PRESIDENCY UNIVERSITY

Presidency University Act, 2013 of the Karnataka Act No. 41 of 2013 | Established under Section 2(f) of UGC Act, 1956
Approved by AICTE, New Delhi

PRESIDENCY SCHOOL OF INFORMATION SCIENCE

Program Regulations and Curriculum
2026-2028

MASTER OF COMPUTER APPLICATIONS (MCA)

based on Choice Based Credit System (CBCS) and Outcome Based Education (OBE)

Table of Contents

PART A – PROGRAM REGULATIONS

1. Vision & Mission of the University and the School / Department

1.1 Vision of the University

To be a Value-driven Global University, excelling beyond peers and creating professionals of integrity and character, having concern and care for society.

1.2 Mission of the University

- Commit to be an innovative and inclusive institution by seeking excellence in teaching, research and knowledge-transfer.
- Pursue research and development and its dissemination to the community, at large.
- Create, sustain and apply learning in an interdisciplinary environment with consideration for ethical, ecological and economic aspects of nation building.
- Provide knowledge-based technological support and services to the industry in its growth and development.
- To impart globally-applicable skill-sets to students through flexible course offerings and support industry's requirement and inculcate a spirit of new-venture creation.

1.3 Vision of Presidency School of Information Science

To be a global centre of excellence in information science and research, fostering innovation and producing professionals with integrity and ethical responsibility.

1.4 Mission of Presidency School of Information Science

- To provide high-quality education in information science, equipping students with strong technical expertise and problem-solving skills.
- To promote research and innovation in information science and technology, addressing real-world challenges through industry collaboration.
- To nurture graduates with strong ethical values and a commitment for lifelong learning for sustained professional growth in the IT sector and allied fields.

2. Preamble to the Program Regulations and Curriculum

This is the subset of Academic Regulations and it is to be followed as a requirement for the award of MCA degree.

The Curriculum is designed to take into the factors listed in the Choice Based Credit System (CBCS) with focus on Social Project Based Learning, Industrial Training, and Internship to enable the students to become eligible and fully equipped for employment in industries, choose higher studies or entrepreneurship.

In exercise of the powers conferred by and in discharge of duties assigned under the relevant provision(s) of the Act, Statutes and Academic Regulations, of the University, the Academic Council hereby makes the following Regulations.

3. Short Title and Applicability

a. These Regulations shall be called the Master of Computer Applications (MCA) Degree Program Regulations and Curriculum 2026-2028
b. These Regulations are subject to, and pursuant to the Academic Regulations.
c. These Regulations shall be applicable to the MCA Degree Programs of the 2026-2028 batch, and to all other MCA Degree Programs which may be introduced in future.
d. These Regulations shall supersede all the earlier MCA Degree Program Regulations and Curriculum, along with all the amendments thereto.
e. These Regulations shall come into force from the Academic Year 2026-2027.

4. Definitions

In these Regulations, unless the context otherwise requires:

a. "Academic Calendar" means the schedule of academic and miscellaneous events as approved by the Vice Chancellor;
b. "Academic Council" means the Academic Council of the University;
c. "Academic Regulations" means the Academic Regulations, of the University;
d. "Academic Term" means a Semester or Summer Term;
e. "Act" means the Presidency University Act, 2013;
f. "AICTE" means All India Council for Technical Education;
g. "Basket" means a group of courses bundled together based on the nature/type of the course;
h. "BOE" means the Board of Examinations of the University;
i. "BOG" means the Board of Governors of the University;
j. "BOM" means the Board of Management of the University;
k. "BOS" means the Board of Studies of a particular Department/Program of Study of the University;
l. "CGPA" means Cumulative Grade Point Average as defined in the Academic Regulations;
m. "Clause" means the duly numbered Clause, with Sub-Clauses included, if any, of these Regulations;
n. "COE" means the Controller of Examinations of the University;
o. "Course In Charge" means the teacher/faculty member responsible for developing and organising the delivery of the Course;
p. "Course Instructor" means the teacher/faculty member responsible for teaching and evaluation of a Course;
q. "Course" means a specific subject usually identified by its Course-code and Course-title, with specified credits and syllabus/course-description, a set of references, taught by some teacher(s)/course-instructor(s) to a specific class (group of students) during a specific Academic Term;
r. "Curriculum Structure" means the Curriculum governing a specific Degree Program offered by the University, and, includes the set of Baskets of Courses along with minimum credit requirements to be earned under each basket for a degree/degree with specialization/minor/honours in addition to the relevant

details of the Courses and Course catalogues (which describes the Course content and other important information about the Course). Any specific requirements for a particular program may be brought into the Curriculum structure of the specific program and relevant approvals should be taken from the BOS and Academic Council at that time.

s. "DAC" means the Departmental Academic Committee of a concerned Department/Program of Study of the University;
t. "Dean" means the Dean / Director of the concerned School;
u. "Degree Program" includes all Degree Programs;
v. "Department" means the Department offering the degree Program(s) / Course(s) / School offering the concerned Degree Programs / other Administrative Offices;
w. "Discipline" means specialization or branch of MCA Degree Program;
x. "HOD" means the Head of the concerned Department;
y. "L-T-P-C" means Lecture-Tutorial-Practical-Credit – refers to the teaching – learning periods and the credit associated;
z. "MOOC" means Massive Open Online Courses;
aa. "MOU" means the Memorandum of Understanding;
bb. "NPTEL" means National Program on Technology Enhanced Learning;
cc. "Parent Department" means the department that offers the Degree Program that a student undergoes;
dd. "Program Head" means the administrative head of a particular Degree Program/s;
ee. "Program Regulations" means the MCA Degree Program Regulations and Curriculum, 2026-2028;
ff. "Program" means the Master of Computer Applications (MCA) Degree Program;
gg. "PSIS" means the Presidency School of Information Science;
hh. "Registrar" means the Registrar of the University;
ii. "School" means a constituent institution of the University established for monitoring, supervising and guiding, teaching, training and research activities in broadly related fields of studies;
jj. "Section" means the duly numbered Section, with Clauses included in that Section, of these Regulations;
kk. "SGPA" means the Semester Grade Point Average as defined in the Academic Regulations.
ll. "Statutes" means the Statutes of Presidency University;
mm. "Sub-Clause" means the duly numbered Sub-Clause of these Program Regulations;
nn. "Summer Term" means an additional Academic Term conducted during the summer break (typically in June-July) for a duration of about eight (08) calendar weeks, with a minimum of thirty (30) University teaching days;
oo. "SWAYAM" means Study Webs of Active Learning for Young Aspiring Minds.
pp. "UGC" means University Grant Commission;
qq. "University" means Presidency University, Bengaluru; and
rr. "Vice Chancellor" means the Vice Chancellor of the University.

5. Program Description

The Programme shall be called Master of Computer Applications, abbreviated as MCA. The MCA Degree Program Regulations and Curriculum 2026-2028 are subject to, and, pursuant to the Academic Regulations. These Program Regulations shall be applicable to the ongoing MCA Degree Program of 2026-2028 offered by the Presidency School of Information Science.

5.1 These Program Regulations shall be applicable to other similar programs, which may be introduced in future.

5.2 These Regulations may evolve and get amended or modified or changed through appropriate approvals from the Academic Council, from time to time, and shall be binding on all concerned.

5.3 The effect of periodic amendments or changes in the Program Regulations, on the students admitted in earlier years, shall be dealt with appropriately and carefully, so as to ensure that those students are not subjected to any unfair situation whatsoever, although they are required to conform to these revised Program Regulations, without any undue favour or considerations.

6. Minimum and Maximum Duration

6.1 MCA Degree Program is a Two-Year, Full-Time Semester based program. The minimum duration of the MCA Program is Two (02) years and each year comprises of two academic Semesters (Odd and Even Semesters) and hence the duration of the MCA program is four (04) Semesters.

6.2 A student who for whatever reason is not able to complete the Program within the normal period or the minimum duration (number of years) prescribed for the Program, may be allowed a period of two years beyond the normal period to complete the mandatory minimum credits requirement as prescribed by the concerned Program Regulations and Curriculum. In general, the permissible maximum duration (number of years) for completion of Program is 'N' + 2 years, where 'N' stands for the normal or minimum duration (number of years) for completion of the concerned Program as prescribed by the concerned Program Regulations and Curriculum.

6.3 The time taken by the student to improve Grades/CGPA, and in case of temporary withdrawal/re-joining (Refer to Clause 16.1 of Academic Regulations), shall be counted in the permissible maximum duration for completion of a Program.

6.4 In exceptional circumstances, such as temporary withdrawal for medical exigencies where there is a prolonged hospitalization and/or treatment, as certified through hospital/medical records, women students requiring extended maternity break (certified by registered medical practitioner), and, outstanding sportspersons representing the University/State/India requiring extended time to participate in National/International sports events, a further

extension of one (01) year may be granted on the approval of the Academic Council.

6.5 The enrolment of the student who fails to complete the mandatory requirements for the award of the concerned Degree (refer Section 19.0 of Academic Regulations) in the prescribed maximum duration (Clauses 18.1 and 18.2 of Academic Regulations), shall stand terminated and no Degree shall be awarded.

7 Programme Educational Objectives (PEO)

After 2-3 years of successful completion of the program, the graduates will:

PEO1: Establish themselves as competent professionals or pursue higher education by applying technical knowledge, analytical thinking, and problem-solving skills in the field of computer applications.

PEO2: Demonstrate ethical responsibility, strong communication skills, and a commitment to lifelong learning by adapting to emerging technologies and evolving industry practices.

PEO3: Lead and contribute to project teams in designing and implementing innovative software solutions that support organizational and societal needs.

8 Programme Outcomes (PO) and Programme Specific Outcomes (PSO)

8.1 Programme Outcomes (PO)

On successful completion of the Program, the students shall be able to:

PO 1: Foundation Knowledge: Apply knowledge of mathematics, programming logic and coding fundamentals for solution architecture and problem solving.

PO 2: Problem Analysis: Identify, review, formulate and analyse problems for primarily focusing on customer requirements using critical thinking frameworks.

PO 3: Development of Solutions: Design, develop and investigate problems with as an innovative approach for solutions incorporating ESG/SDG goals.

PO 4: Modern Tool Usage: Select, adapt and apply modern computational tools such as development of algorithms with an understanding of the limitations including human biases.

PO 5: Individual and Teamwork: Function and communicate effectively as an individual or a team leader in diverse and multidisciplinary groups. Use methodologies such as agile.

PO 6: Project Management and Finance: Use the principles of project management such as scheduling, work breakdown structure and be conversant with the principles of Finance for profitable project management.

PO 7: Ethics: Commit to professional ethics in managing software projects with financial aspects. Learn to use new technologies for cyber security and insulate customers from malware.

PO 8: Life-long learning: Change management skills and the ability to learn, keep up with contemporary technologies and ways of working.

8.2 Program Specific Outcomes (PSOs):

On successful completion of the program, the students shall be able to:

PSO1: Design and develop software solutions for complex real-world problems in specialized domains such as Full-Stack Development, Artificial Intelligence, and Cybersecurity by applying software engineering principles and advanced programming skills.

PSO2: Identify, analyse and implement use cases across diverse sectors including Healthcare, Agriculture, Banking, and Insurance by utilizing modern tools, effective teamwork, and project management practices.

PSO3: Demonstrate professional ethics, embrace lifelong learning of emerging technologies, and actively contribute to the field of computer applications.

9 Admission Criteria (as per the concerned Statutory Body)

The University admissions shall be open to all persons irrespective of caste, class, creed, gender or nation. All admissions shall be made on the basis of merit in the qualifying examinations; provided that forty percent of the admissions in all Programs of the University shall be reserved for the students of Karnataka State and admissions shall be made through a Common Entrance Examination conducted by the State Government or its agency and seats shall be allotted as per the merit and reservation policy of the State Government from time to time**.** The admission criteria to the MCA Program are listed in the following Sub-Clauses:

9.1 An applicant who has successfully completed BCA / Bachelor's Degree in Computer Science Engineering or equivalent degree OR Passed B.Sc./ B.Com./ B.A. with Mathematics at 10+2 Level or at Graduation Level (with additional bridge Courses as per the norms of the concerned University).

9.2 The applicant must have appeared Karnataka PG-CET or any other State-level Entrance Examinations.

9.3 Reservation for SC/ST and other backward Sections/Classes/Categories shall be made in accordance with the directives issued by the Government of Karnataka from time to time.

9.4 Admissions are offered to Foreign Nationals and Indians living abroad in accordance with the rules applicable for such admission, issued from time to time, by the Government of India.

9.5 Candidates must fulfil the medical standards required for admission as prescribed by the University.

9.6 If, at any time after admission, it is found that a candidate had not in fact fulfilled all the requirements stipulated in the offer of admission, in any form whatsoever, including possible misinformation and any other falsification, the Registrar shall report the matter to the Board of Management (BOM), recommending revoking the admission of the candidate.

9.7 The decision of the BOM regarding the admissions is final and binding.

10 Transfer Students requirements

10.1 Transfer of student(s) from another recognized University to the 2nd year (3rd Semester) of the MCA Program of the University

A student who has completed the 1st Year (i.e., passed in all the Courses / Subjects prescribed for the 1st Year) of the MCA, two-Year Degree Program from another recognized University, may be permitted to transfer to the 2nd Year (3rd Semester) of the MCA Program of the University as per the rules and guidelines prescribed in the following Sub-Clauses:

10.1.1 Candidates seeking transfer may be required to complete specified bridge Courses, if any, as prescribed by the University. Such bridge Courses shall not be included in the CGPA computations.

10.1.2 All the existing Regulations and Policies of the University shall be binding on all the students admitted to the Program through the provision of transfer.

10.1.3 The student shall submit the Application for Transfer along with a non-refundable Application Fee (as prescribed by the University from time to time) to the Presidency University no later than the last date of registration of the concerned year/semester for admission to the 2nd Year (3rd Semester) MCA Program commencing on August 1 on the year concerned.

10.1.4 The student shall submit copies of the respective Marks Cards / Grade Sheets / Certificates along with the Application for Transfer.

10.1.5 The transfer may be provided on the condition that the Courses and Credits completed by the concerned student in the 1st Year of the MCA Degree Program from the concerned University, are declared equivalent and acceptable by the Equivalence Committee constituted by the Vice Chancellor for this purpose. Further, the Equivalence Committee may also prescribe the Courses and Credits the concerned students shall have to mandatorily complete, if admitted to the 2nd Year of the MCA Program of the University.

11. Mandatory Bridge Course for Non-Computer Science Discipline Students

Students who have studied Mathematics at the 10+2 level or at the graduation level, but have not undergone Computer Science related courses, are required to undergo the following Bridge Courses before starting of the regular first semester classes to facilitate them with foundational knowledge in computer science.

Sl. No.	Course Code	Course Name	Contact Hours (Modular Basis)
1	CSA9001	C PROGRAMMING AND DATA STRUCTURES	15
2	CSA9002	Fundamentals of Information Technology	15
		TOTAL	**30**

Requirement: Students must successfully complete the Bridge Course and obtain a minimum of 50% marks in the qualifying examination to proceed with their regular academic program.

12. Specific Regulations regarding Assessment and Evaluation (including the Assessment Details of NTCC Courses, Weightages of Continuous Assessment and End Term Examination for various Course Categories)

12.1 The academic performance evaluation of a student in a Course shall be according to the University Letter Grading System based on the class performance distribution in the Course.

12.2 Academic performance evaluation of every registered student in every Course registered by the student is carried out through various components of Assessments spread across the Semester. The nature of components of Continuous Assessments and the weightage given to each component of Continuous Assessments (refer Clause 8.8 of Academic Regulations) shall be clearly defined in the Course Plan for every Course, and approved by the DAC.

12.3 Format of the End-Term examination shall be specified in the Course Plan.

12.4 Grading is the process of rewarding the students for their overall performance in each Course. The University follows the system of Relative Grading with statistical approach to classify the students based on the relative performance of the students registered in the concerned Course except in the following cases:

- Non-Teaching Credit Courses (NTCC)
- Courses with a class strength less than 30

Absolute grading method may be adopted, where necessary with prior approval of concerned DAC.

Grading shall be done at the end of the Academic Term by considering the aggregate performance of the student in all components of Assessments prescribed for the Course. Letter Grade (Clause 8.10 of Academic Regulations) shall be awarded to a student based on her/his overall performance relative to the class performance distribution in the concerned Course. These Letter Grades not only indicate a qualitative assessment of the student's performance but also carry a quantitative (numeric) equivalent called the Grade Point.

12.5 Assessment Components and Weightage

Table 1: Assessment Components and Weightage for different category of Courses

Nature of Course and Structure	Evaluation Component	Weightage
Lecture-based Course L component in the L-T-P Structure is predominant (more than 1) (Examples: 3-0-0; 3-0-2; 2-1-0; 2-0-2, 2-0-4 etc.)	Continuous Assessments	50%
	End Term Examination	50%

Lab/Practice-based Course P component in the L-T-P Structure is predominant (Examples: 0-0-4; 1-0-4; 1-0-2; etc.)	Continuous Assessments	75%
	End Term Examination	25%
Skill based Courses like Industry Internship, Major project, Research Dissertation, Integrative Studio, Interdisciplinary Project, Summer / Short Internship, Social Engagement / Field Projects, Portfolio, and such similar Non-Teaching Credit Courses, where the pedagogy does not lend itself to a typical L-T-P structure	Guidelines for the assessment components for the various types of Courses, with recommended weightages, shall be specified in the concerned Program Regulations and Curriculum / Course Plans, as applicable.	

The exact weightages of Evaluation Components shall be clearly specified in the concerned PRC and respective Course Plan.

Normally, for Practice/Skill based Courses, without a defined credit structure (L–T–P) [NTCC], but with assigned Credits (as defined in Clause 5.2 of the Academic Regulations), the method of evaluation shall be based only on Continuous Assessments. The various components of Continuous Assessments, the distribution of weightage among such components, and the method of evaluation/assessment, shall be as decided and indicated in the Course Plan/PRC. The same shall be approved by the respective DAC.

12.6 Minimum Performance Criteria:

12.6.1 Theory only Course and Lab/Practice Embedded Theory Course

A student shall satisfy the following minimum performance criteria to be eligible to earn the credits towards the concerned Course:

a. A student must obtain a minimum of 30% of the total marks/weightage assigned to the End Term Examinations in the concerned Course.
b. The student must obtain a minimum of 40% of the AGGREGATE of the marks/weightage of the components of Continuous Assessments, Mid Term Examinations and End Term Examinations in the concerned Course.

12.6.2 Lab/Practice only Course and Project Based Courses

The student must obtain a minimum of 40% of the AGGREGATE of the marks/weightage of all assessment components in the concerned Course.

12.6.3 A student who fails to meet the minimum performance criteria listed above in a Course shall be declared as "Fail" and given "F" Grade in the concerned Course. For theory Courses, the student shall have to re-appear in the "Make-Up Examinations" as scheduled by the University in any subsequent semester, or, re-appear in the End Term Examinations of the same Course when it is scheduled at the end of the following Semester or Summer Term, if

offered. The marks obtained in the Continuous Assessments (other than the End Term Examination) shall be carried forward and be included in computing the final grade, if the student secures the minimum requirements (as per the sub-clauses 8.9.1 and 8.9.2 of Academic Regulations) in the "Make-Up Examinations" of the concerned Course. Further, the student has an option to re-register for the Course and clear the same in the summer term/ subsequent semester if he/she wishes to do so, provided the Course is offered.

13. Additional clarifications - Rules and Guidelines for Transfer of Credits from MOOC, etc. – Note: These are covered in Academic Regulations

The University allows students to acquire credits from other Indian or foreign institutions and/or Massive Open Online Course (MOOC) platforms, subject to prior approval. These credits may be transferred and counted toward fulfilling the minimum credit requirements for the award of a degree. The process of transfer of credits is governed by the following rules and guidelines:

13.1 The transfer of credits shall be examined and recommended by the Equivalence Committee (Refer ANNEXURE B of Academic Regulations) and approved by the Dean - Academics.

13.2 Students may earn credits from other Indian or foreign Universities/Institutions with which the University has an MOU, and that MOU shall have specific provisions, rules and guidelines for transfer of credits. These transferred credits shall be counted towards the minimum credit requirements for the award of the degree.

13.3 Students may earn credits by registering for Online Courses offered by *Study Web of Active Learning by Young and Aspiring Minds* (SWAYAM) and *National Program on Technology Enhanced Learning* (NPTEL), or other such recognized Bodies/ Universities/Institutions as approved by the concerned BOS and Academic Council from time to time. The concerned School/Parent Department shall publish/include the approved list of Courses and the rules and guidelines governing such transfer of credits of the concerned Program from time to time. The Rules and Guidelines for the transfer of credits specifically from the Online Courses conducted by SWAYAM/ NPTEL/ other approved MOOCs are as stated in the following Sub-Clauses:

13.3.1 A student may complete SWAYAM/NPTEL/other approved MOOCs as mentioned in Clause 13.3 (as per Academic Regulations) and transfer equivalent credits to partially or fully complete the mandatory credit requirements of Discipline Elective Courses prescribed in the concerned Curriculum Structure. However, it is the sole responsibility of the student to complete the mandatory credit requirements of the Discipline Elective Courses prescribed by the Curriculum Structure of the concerned Program.

13.3.2 SWAYAM/NPTEL/ other approved MOOCs as mentioned in Clause 13.3 (as per Academic Regulations) shall be approved by the concerned Board of Studies and placed (as Annexures) in the concerned PRC.

13.3.3 Parent Departments may release a list of SWAYAM/NPTEL/other approved MOOCs for Pre-Registration as per schedule in the Academic Calendar or through University Notification to this effect.

13.3.4 Students may Pre-Register for the SWAYAM/NPTEL/other approved MOOCs in the respective Departments and register for the same Courses as per the schedule announced by respective Online Course Offering body/institute/ university.

13.3.5 A student shall request for transfer of credits only from such approved Courses as mentioned in Sub-Clause 13.3.2 above.

13.3.6 SWAYAM/NPTEL/other approved MOOCs Courses are considered for transfer of credits only if the concerned student has successfully completed the SWAYAM/NPTEL/other approved MOOCs and obtained a certificate of successful/satisfactory completion.

13.3.7 A student who has successfully completed the approved SWAYAM/NPTEL/ other approved MOOCs and wants to avail the provision of transfer of equivalent credits, must submit the original Certificate of Completion, or such similar authorized documents to the HOD concerned, with a written request for the transfer of the equivalent credits. On verification of the Certificates/Documents and approval by the HOD concerned, the Course(s) and equivalent Credits shall be forwarded to the COE for processing of results of the concerned Academic Term.

13.3.8 The credit equivalence of the SWAYAM/NPTEL/other approved MOOCs are based on Course durations and/or as recommended by the Course offering body/institute/university. The Credit Equivalence mapped to SWAYAM/ NPTEL approved Courses based on Course durations for transfer of credits is summarised in Table shown below. The Grade will be calculated from the marks received by the Absolute Grading Table 8.11 in the Academic Regulations.

Table 2: Durations and Credit Equivalence for Transfer of Credits from SWAYAM-NPTEL/ other approved MOOC Courses

Sl. No.	Course Duration	Credit Equivalence
1	4 Weeks	1 Credit
2	8 Weeks	2 Credits
3	12 Weeks	3 Credits

13.3.9 The maximum permissible number of credits that a student may request for credit transfer from MOOCs shall not exceed 20% of the

mandatory minimum credit requirements specified by the concerned Program Regulations and Curriculum for the award of the concerned Degree.

13.3.10 The University shall not reimburse any fees/expense; a student may incur for the SWAYAM/NPTEL/other approved MOOCs.

13.4 The maximum number of credits that can be transferred by a student shall be limited to forty percent (40%) of the mandatory minimum credit requirements specified by the concerned Program Regulations and Curriculum for the award of the concerned Degree. However, the grades obtained in the Courses transferred from other Institutions/MOOCs, as mentioned in this Section (13), shall not be included in the calculation of the CGPA.

13.5 Mandatory Non-Credit Course Completion Requirements: All mandatory non-credit courses shall be satisfactorily completed by the student as part of the degree requirements. These courses will be evaluated and awarded letter grades based on the following criteria:

- S (Satisfactorily Completed): Awarded when the student successfully completes all prescribed course requirements.
- NC (Not Completed): Awarded when the student fails to meet the prescribed course requirements.

A student receiving an NC grade must reappear for and complete the course in accordance with the guidelines prescribed by the University.

In the case of non-taught and non-credited mandatory courses—where students are advised to undertake learning through MOOC platforms—there shall be a clearly defined Course Catalogue and a corresponding Course Plan. The Course Plan shall outline the assessment components, which will form the basis for evaluation.

PART B: PROGRAM STRUCTURE

14. Structure / Component with Credit Requirements Course Baskets & Minimum Basket wise Credit Requirements

The MCA Program Structure (2026-2028) totalling 80 credits. Table 3 summarizes the type of baskets, number of courses under each basket and the associated credits that are mandatorily required for the completion of the Degree.

Table 3: MCA 2026-2028: Summary of Mandatory Courses and Minimum Credit Contribution from various Baskets

Sl. No.	Baskets	Credit Contribution
1	Program Core (PC)	42
2	Elective Course (EC)	15
3	Project Work (Project)	15
4	Foundation Course (FC)	8
	Total Credits	**80**

In the entire Program, the practical and skill-based course component contribute to an extent of approximately 82% out of the total credits of 80 for MCA Computer Applications program of Two-year duration.

15. Minimum Total Credit Requirements of Award of Degree

A minimum of 80 credits is required for the award of an MCA degree.

16. Other Specific Requirements for Award of Degree, if any, as prescribed by the Statutory Bodies,

16.1 The award of the Degree shall be recommended by the Board of Examinations and approved by the Academic Council and Board of Management of the University**.**

16.2 A student shall be declared to be eligible for the award of the concerned Degree if she/he:

a. Fulfilled the Minimum Credit Requirements and the Minimum Credits requirements under various baskets;
b. Secure a minimum CGPA of 5.0 in the concerned Program at the end of the Semester/Academic Term in which she/he completes all the requirements for the award of the Degree as specified in Sub-Clause 19.2.1 of Academic Regulations**;**
c. No dues to the University, Departments, Hostels, Library, and any other such Centers/ Departments of the University; and
d. No disciplinary action is pending against her/him.

PART C: CURRICULUM STRUCTURE

17.Curriculum Structure – Basket Wise Course List

Table 3.1: List of Program Core

Sl. No.	Course Code	Course Name	L	T	P	C
Program Core						
		Theory Course				
1	CSA4201	Data Structures and Algorithms	3	0	0	3
2	CSA4207	Database Systems	3	0	0	3
3	CSA4203	Computer Networks and Security	3	0	0	3
4	CSA4501	Cloud Computing	2	0	0	2
5	CSA4204	Object Oriented Programming using Java	2	0	0	2
6	CSA4206	Software Engineering	3	0	0	3
7	CSA4502	Machine Learning	2	0	0	2
8	CSA4503	Data Analytics and Visualization	2	0	0	2
9	CSA4606	MERN Full Stack Development	1	0	4	3
		Practical Course				23
10	CSA4303	Web Technology	1	0	4	3
11	CSA4305	Advanced Python Programming	1	0	4	3
12	CSA4306	UI/UX Design	1	0	4	3
13	CSA4301	Data Structures and Algorithms Lab	0	0	2	1
14	CSA4307	Database Systems Lab	0	0	2	1
15	CSA4601	Cloud Computing Lab	0	0	2	1
16	CSA4304	Object Oriented Programming using Java Lab	0	0	4	2
17	CSA4602	Machine Learning Lab	0	0	2	1
18	CSA4603	Data Analytics and Visualization Lab	0	0	2	1
19	CSA4605	Mobile Application Development using Flutter	1	0	4	3
		Total				19 (42)

Table 3.2: List of Project Work

Sl. No.	Course Code	Course Name	L	T	P	C
Foundation Courses						
1	CSA8100	Mini Project	0	0	0	3
2	CSA8300	Major Project	0	0	0	12
						15

Table 3.3: List of Foundation Courses

Sl. No.	Course Code	Course Name	L	T	P	C
Foundation Courses						
1	ENG5001	English for Employability	2	1	0	3
2	MAT4001	Probability and Statistics	3	0	0	3
3	PPS4008	Quantitative skills and logical reasoning	1	0	2	2
4	CSA4001	Design Thinking and Innovation	0	0	2	0
						8

18. Practical / Skill based Courses – Internships / Thesis / Dissertation / Major Project Work / Portfolio / Mini project

Practical / Skill based Courses like internship, project work, major project, research project / dissertation, and such similar courses, where the pedagogy does not lend itself to a typical L-T-P-C Structure as defined in Clause 5.1 of the Academic Regulations, are simply assigned the number of Credits based on the quantum of work / effort required to fulfill the learning objectives and outcomes prescribed for the concerned Courses. Such courses are referred to as Non-Teaching Credit Courses (NTCC). These Courses are designed to provide students with hands-on experience and skills essential for their professional development. These courses aim to equip students with abilities in problem identification, root cause analysis, problem-solving, innovation, and design thinking through industry exposure and project-based learning. The expected outcomes are first level proficiency in problem solving and design thinking skills to better equip MCA graduates for their professional careers. The method of evaluation and grading for the Practical / Skill based Courses shall be prescribed and approved by the concerned Departmental Academic Committee (refer Annexure A of the Academic Regulations). The same shall be prescribed in the Course Handout.

18.1 Mini Project

A student may opt to do a Project Work for a period of 6-8 weeks in an Industry / Company or academic / research institution or the University Department(s) as an equivalence of Internship during the 3rd Semester as applicable, subject to the following conditions:

18.1.1 The Project Work shall be approved by the concerned HOD and be carried out under the guidance of a faculty member.

18.1.2 The student may do the project work in an Industry / Company or academic / research institution of her / his choice subject to the above-mentioned condition (Sub-Clause 18.1.1). Provided further, that the Industry / Company or academic / research institution offering such project work confirms to the University that the project work will be conducted in accordance with the Program Regulations and requirements of the University.

18.2 Major Project

A student may undergo a Project for a period of 12-14 weeks in an industry / company or academic / research institution in the 4th Semester as applicable, subject to the following conditions:

18.2.1 The Project shall be conducted in accordance with the Project Policy prescribed by the University from time to time.

18.2.2 The selection criteria (minimum CGPA, pass in all Courses as on date, and any other qualifying criteria) as applicable / stipulated by the concerned Industry / Company or academic / research institution for award of the Major Project to a student;

18.2.3 The number of Projects available for the concerned Academic Term shall determine the allocation of Major Projects. Further, the available number of Major Project shall be awarded to the students by the University on the basis of merit using the CGPA secured by the student. Provided further, the student fulfils the criteria, as applicable, specified by the Industry / Company or academic / research institution providing the Major Project, as stated in Sub-Clause 18.2.2 above.

18.2.4 A student may opt for Project in an Industry / Company or academic / research institution of her / his choice, subject to the condition that the concerned student takes the responsibility to arrange the Project on her/his own. Provided further, that the Industry / Company or academic / research institution offering such Major Project confirms to the University that the Major Project shall be conducted in accordance with the Program Regulations and Internship Policy of the University.

18.2.5 A student selected for a Project in an industry / company or academic / research institution shall adhere to all the rules and guidelines prescribed in the Major Project Policy of the University.

18.3 Research Project / Dissertation

A student may opt to do a Research Project / Dissertation for a period of 12-14 weeks in an Industry / Company or academic / research institution or the University Department(s) as an equivalence of Major Project, subject to the following conditions:

18.3.1 The Research Project / Dissertation shall be approved by the concerned HOD and be carried out under the guidance of a faculty member.

The student may do the Research Project / Dissertation in an Industry / Company or academic / research institution of her / his choice subject to the above-mentioned condition (Sub-Clause 18.3.1). Provided further, that the Industry / Company or academic / research institution offering such Research Project / Dissertation confirms to the University that the Research Project / Dissertation work will be conducted in accordance with the Program Regulations and requirements of the University.

19.List of Elective Courses under various Specialisations / Stream Basket

Table 3.4: List of Discipline Electives

Sl. No.	Course Code	Course Name	L	T	P	C
Full Stack Stream						
1	CSA4701	Agile Methodology and DevOps	3	0	0	3
2	CSA4731	Responsive Web Designing	2	0	2	3
3	CSA4732	Enterprise Computing with Java	2	0	2	3
4	CSA4704	Data Modelling with NoSQL Databases	2	0	2	3
5	CSA4705	Backend Development with Node.js	2	0	2	3
Artificial Intelligence and Machine Learning Stream						
1	CSA4706	Computer Vision	3	0	0	3
2	CSA4707	Natural Language Processing	2	0	2	3
3	CSA4708	Reinforcement Learning	3	0	0	3
4	CSA4709	Deep Learning	2	0	2	3
5	CSA4710	Generative AI	2	0	2	3
Cloud and Cyber Security Stream						
1	CSA4711	Cyber Security and Ethical Hacking	3	0	0	3
2	CSA4712	Web Application Security	2	0	2	3
3	CSA4713	Cybersecurity Testing	3	0	0	3
4	CSA4714	Cloud Security	2	0	2	3
5	CSA4715	AI in Cyber Security	2	0	2	3
Fintech Stream						
1	CSA4716	Applied AI in Financial Risk & Credit Modelling	2	0	2	3
2	CSA4717	Smart Contract Engineering & Decentralized Finance (DeFi)	2	0	2	3
3	CSA4718	Advanced FinTech Cybersecurity & Compliance Architecture	2	0	2	3
4	CSA4719	Portfolio Optimization & Robo-Advisory Algorithms	2	0	2	3
5	CSA4720	RegTech Systems, Fraud Analytics & Compliance Automation	2	0	2	3
AI and Data Science, Big Data & IoT Stream						
1	CSA4721	Applied Data Science and Feature Engineering	3	0	0	3
2	CSA4722	Big Data Analytics using Hadoop and Spark	2	0	2	3
3	CSA4723	Time Series Analysis and Forecasting Techniques	3	0	0	3
4	CSA4724	IoT and Edge Computing Systems	2	0	2	3
5	CSA4725	Data Engineering and Pipeline Architecture	2	0	2	3
Quantum Computing Stream						
1	CSA4726	Quantum Information Theory	3	0	0	3
2	CSA4727	Quantum Algorithms and Programming	2	0	2	3
3	CSA4728	Quantum Optimization Techniques and Applications	3	0	0	3
4	CSA4729	Quantum Cryptography and Post-Quantum Security	2	0	2	3
5	CSA4730	Advanced Quantum Algorithms	2	0	2	3

20. List of MOOC (NPTEL) Courses

20.1 Presidency University students are given the opportunity to study abroad in International Universities through a selection process coordinated by the Office of International Affairs (OIA). Such selected students need to complete their credits for the semester that they are abroad in the following way:

20.1.1 The student needs to study and complete School Core and Program Core Courses in offline mode only.

20.1.2 Massive Open Online Course (MOOC) courses maybe given for Discipline Elective Courses. These courses need to be approved by the concerned BOS and Academic Council from time to time.

20.1.3 SWAYAM/NPTEL/ other approved MOOCs shall be approved by the concerned Board of Studies and placed in the concerned PRC.

20.1.4 Student shall register for these courses in the ERP of Presidency University.

20.1.5 For these MOOC courses faculty coordinators are identified. These faculty should have undergone similar MOOC courses and therefore should be familiar with the mode of class conduction, types of assessments and evaluation procedures.

20.1.6 Study materials shall be provided to the students as video lectures shared by the MOOCs Coordinator(s), or the students may access the approved MOOCs Portal directly. The mode of class conduction is determined by the MOOCs coordinator(s) as detailed in the Course Catalogue and Course Plan.

20.1.7 The question paper shall be prepared by the MOOCs coordinator(s).

20.1.8 Students write the exams in online mode. These exams are scheduled and conducted by the School.

20.1.9 Results are evaluated by School and given to the Office of the Controller of Examinations (CoE).

20.1.10 The details of the duration, credits and evaluation are given below:

Sl.No	Duration	Credits	Evaluation
1.	12 weeks	3	Continuous Assessment –50 Marks Mid Term –50 Marks End Term-100 Marks
2.	8 weeks	2	Mid Term-50 Marks End Term-100 Marks
3	4 weeks	1	End Term-100 Marks

20.2 MOOC Courses for MCA Program

Table 3.9: MOOC Discipline Elective Courses
Discipline Elective Courses duration is 4 weeks (01 credit)/ 8 weeks (02 credits)/ 12 weeks (03 credits)

Sl.No	NPTEL Course Code	Course Code	Course Name	Credits
1			Scalable Data Science - Anirban	3

			Dasgupta and Sourangshu Bhattacharyya	
2			Machine Learning for Earth System Sciences - Adway Mitra	3
3			Introduction to Graph Algorithms - C Pandu Rangan	3
4			Hardware Modeling Using Verilog - Indranil Sengupta	3
5			Distributed Systems - Rajiv Misra	3
6			Data Science for Engineers - Raghunathan Rengasamy and Shankar Narasimhan	3
7			Artificial Intelligence for Economics - Adway Mitra, Dripto Bakshi and Palash Dey	3

Table 3.10: MOOC Open Elective Courses
Discipline Elective Courses duration is 4 weeks (01 credit)/ 8 weeks (02 credits)/ 12 weeks (03 credits)

Sl.No	NPTEL Course Code	Course Code	Course Name	Credits
1			Film Appreciation - Aysha Viswamohan	3
2			Financial Accounting - Varadaraj Bapat	3
3			Introduction to Biostatistics - Shamik Sen	3
4			Introduction to Urban Planning - Harshit Lakra	3
5			Knowledge Management - KBL Srivastava	3
6			Petroleum Technology - Sonali Sengupta	3

The following AI Enhanced / AI Extra Edge courses were proposed for inclusion:

S. No	Course Code	Course Name	L	T	P	C	Basket	Category	Program	Sem
1.	CSA3804	Prompt Engineering and AI Tools for Productivity	3	0	0	3	DSE	AI Extra Edge	MCA	–
2.	CSA3805	Responsible AI and Bias Auditing	3	0	0	3	DSE	AI Extra Edge	MCA	–
3.	CSA3806	Project Management and MLOps	3	0	0	3	DSE	AI Extra Edge	MCA	–

21. Recommended Semester Wise Course Structure / Flow including the Programme / Discipline Elective Paths / Options

Semester 1:

Sl. No.	Course Code	Course Name	Credit Structure				Contact Hours	Type of Course
			L	T	P	C		
1.	ENG5001	English for Employability	2	1	0	3	3	FC
2.	MAT4001	Probability and Statistics	3	0	0	3	3	FC
3.	CSA4201	Data Structures and Algorithms	3	0	0	3	3	PC
4.	CSA4207	Database Systems	3	0	0	3	3	PC
5.	CSA4203	Computer Networks and Security	3	0	0	3	3	PC
6.	CSA4303	Web Technology	1	0	4	3	5	PC
7.	CSA4305	Advanced Python Programming	1	0	4	3	5	PC
8.	CSA4301	Data Structures and Algorithms Lab	0	0	2	1	2	PC
9.	CSA4307	Database Systems Lab	0	0	2	1	2	PC
10.	CSA4001	Design Thinking and Innovation	0	0	2	0	2	FC
		TOTAL	**18**	**1**	**12**	**23**	**31**	

Semester 2:

Sl. No.	Course Code	Course Name	Credit Structure				Contact Hours	Type of Course
			L	T	P	C		
1.	CSA4501	Cloud Computing	2	0	0	2	2	PC
2.	CSA4502	Machine Learning	2	0	0	2	2	PC
3.	CSA4204	Object Oriented Programming using Java	2	0	0	2	2	PC
4.		Elective 1	3	0	0	3	3	EC
5.	PPS4008	Quantitative Skills and Logical Reasoning	1	0	2	2	3	FC
6.	CSA4306	UI/UX Design	1	0	4	3	5	PC
7.		Elective 2	1	0	4	3	5	EC
8.	CSA4601	Cloud Computing Lab	0	0	2	1	2	PC
9.	CSA4304	Object Oriented Programming using Java Lab	0	0	4	2	4	PC
10.	CSA4602	Machine Learning Lab	0	0	2	1	2	PC
		TOTAL	**12**	**0**	**18**	**21**	**30**	

Semester 3:

Sl. No.	Course Code	Course Name	Credit Structure				Contact Hours	Type of Course
			L	T	P	C		

1	CSA4206	Software Engineering	3	0	0	3	3	PC
2	CSA4503	Data Analytics and Visualization	2	0	0	2	2	PC
3	CSA4504	MERN Full Stack Development	1	0	4	3	5	PC
4		Elective 3	3	0	0	3	3	EC
5		Elective 4	2	0	2	3	4	EC
6		Elective 5	2	0	2	3	4	EC
7	CSA4603	Data Analytics and Visualization Lab	0	0	2	1	2	PC
8	CSA4605	Mobile Application Development using Flutter	1	0	4	3	5	PC
9	CSA8100	Mini Project	0	0	0	3		Project
TOTAL			**14**	**0**	**14**	**24**	**28**	

Semester 4:

Sl. No.	**Course Code**	**Course Name**	**Credit Structure**				**Contact Hours**	**Type of Course**
			L	**T**	**P**	**C**		
1	CSA8300	Major Project	0	0	0	12	0	Project
TOTAL			**0**	**0**	**0**	**12**		

22. Course Catalogue

Course Catalogue of all Courses Listed including the Courses Offered by other School / Department and Discipline / Programme Electives – Course Code, Course Name, Prerequisite, Anti-requisite, Course Description, Course Outcome, Course Content (with Blooms Level, CO, No. of Contact Hours), Reference Resources.

<table>
<tr><td>Course Code:
ENG5001</td><td colspan="3">Course Title: English for Employability
Type of Course: Program Core</td><td>L-T- P- C</td><td>2</td><td>1</td><td>0</td><td>3</td></tr>
<tr><td>Version No.</td><td colspan="8">3.0</td></tr>
<tr><td>Course Pre-requisites</td><td colspan="8">Graduate Level English Language Proficiency</td></tr>
<tr><td>Anti-requisites</td><td colspan="8">NIL</td></tr>
<tr><td>Course Description</td><td colspan="8">This course is designed to enhance students' communication skills in English with a focus on employability. It develops listening, speaking, reading, and writing (LSRW) skills through structured activities and assessments. The course also equips students with research writing skills, enabling effective academic and professional communication.</td></tr>
<tr><td>Course Objective</td><td colspan="8">The objective of the course is EMPLOYABILITY enhancement of students through PARTICIPATIVE LEARNING techniques.</td></tr>
<tr><td>Course Outcomes</td><td colspan="8">On successful completion of this course, the students shall be able to:
CO1: Understand and interpret spoken English by identifying main ideas and supporting details. [Understand]
CO2: Apply effective listening and speaking strategies for professional communication. [Apply]
CO3: Apply reading strategies to analyze, interpret, and evaluate written content. [Apply]
CO4: Analyze different communication contexts including workplace and academic settings. [Analyze]
CO5: Apply research writing techniques including report writing, referencing, and academic formatting. [Apply]
CO6: Create structured academic and professional documents such as reports and research articles. [Create]</td></tr>
<tr><td>Course Content:</td><td colspan="8"></td></tr>
<tr><td>Module 1</td><td colspan="2">Active Listening – CO1, CO2</td><td>Listening to audio clips and answering questions</td><td>Listening skills and Vocabulary Building</td><td colspan="4">10 Sessions</td></tr>
<tr><td colspan="9">1. Listening to Speeches for Vocabulary and Intonation – TED Talks and Podcasts
2. Barriers to Effective Listening
3. Types of Listening – Informational, Discriminative, Critical, Empathetic, Appreciative
4. Listening and Note Taking – Activity</td></tr>
<tr><td>Module 2</td><td colspan="2">Effective Speaking – CO2, CO4</td><td>Presentation</td><td>Speaking Skills</td><td colspan="4">12 Sessions</td></tr>
<tr><td colspan="9">1. Workplace Communication and Communication Etiquette
2. Practical frameworks to improve speaking
3. Attending Interviews
4. Asking and responding to questions, Formal and Informal Communication</td></tr>
</table>

5. Expressing views, opinions and preferences 6. Presentation Skills 7. Short speeches				
Module 3	**Reading Strategies – CO3, CO4**	**Reading Research Articles**	**Reading Skills**	**12 Sessions**
1. Components of reading 2. Improving thinking skills, analytical abilities, and decision making through reading 3. Reading Strategies 4. Reading and Note Making – Activity				
Module 4	**Scientific Writing / Writing Dissertation – CO5, CO6**	**Writing Reports**	**Writing Skills**	**10 Sessions**
1. Report Writing – Types of reports, Components, Structuring 2. Referencing Skills for Academic Writing 3. Writing a Research Article 4. Writing Bibliography				
Project work/Assignment: 1. Listening analysis of TED Talks / Podcasts 2. Mock interviews and presentations 3. Reading analysis of research articles 4. Writing technical reports and research articles				
Topics related to **1. Problem Solving:** Develop communication strategies to interpret, analyze, and present information effectively in academic and professional scenarios. **2. Employability:** Enhances skills in communication, presentation, interview handling, and academic writing required for corporate and research careers.				
Textbook(s): **T1.** Stuart Redman, *"English Vocabulary in Use"*, Cambridge University Press, Updated Edition, 2020**.** **T2.** Michael McCarthy, Felicity O'Dell, *"English Vocabulary in Use (Upper-Intermediate)"*, Cambridge University Press, 2021 (Latest Reprint). **T3.** Nigel D. Turton, *"ABC of Common Grammatical Errors"*, Macmillan, Reprint Edition.				
References **R1.** Steve Hart et al., *"Embark: English for Undergraduates"*, Cambridge University Press, 2020. **R2.** M. Hari Prasad et al., *"Strengthen Your Steps: A Multimodal Course in Communication Skills"*, Maruti Publications. **R3.** Martin Hewings, *"Advanced Grammar in Use"*, Cambridge University Press, 2019 (Latest Edition) **E-Resources:** 1. https://www.ted.com/talks 2. https://nptel.ac.in/courses/ 3. https://owl.purdue.edu/ 4. https://www.cambridgeenglish.org/learning-english/				

Course Code: **MAT4001**	**Course Title:** **Probability and Statistics** **Type of Course:** **Foundation Course**	**L-T- P- C**	3	0	0	3
Version No.	1.0					

Course Pre-requisites	Knowledge of Central Tendency and Measure of Dispersion			
Anti-requisites	**NIL**			
Course Description	This course introduces the fundamental concepts of probability theory and statistical analysis. It focuses on methods to collect, organize, analyze, and interpret data using mathematical models to understand randomness and uncertainty. The course covers probability distributions, statistical methods, correlation, regression, and hypothesis testing. Applications of these concepts are emphasized across engineering, science, business, and social sciences to support data-driven decision-making.			
Course Objective	**The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.**			
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1:** Apply probability concepts including conditional probability, Bayes' theorem, and independence of events. **[Apply]** **CO2:** Apply discrete and continuous random variables and their probability distributions in problem-solving. **[Apply]** **CO3:** Analyze different types of probability distributions such as binomial, Poisson, normal, and exponential distributions. **[Analyze]** **CO4:** Apply statistical methods including correlation, regression, and curve fitting techniques. **[Apply]** **CO5:** Analyze sampling distributions and perform hypothesis testing using statistical tests such as z-test, t-test, and chi-square test. **[Analyze]** **CO6:** Evaluate data-driven problems and interpret statistical results for informed decision-making. **[Evaluate]**			
Course Content:				
Module 1	**Basic Probability – CO1**	**Quiz**	**Problem Solving**	**4 Sessions**
Probability of an Event, multiplication rule, permutations, combinations, Addition Law, Multiplication Law, Conditional Probability, Bayes's Theorem and Problems.				
Module 2	**Random Variables and Bivariate Distributions – CO2, CO3**	**Assignment**	**Case Study**	**18 Sessions**
Random Variables (discrete and continuous), Probability Mass/Density Functions, Mathematical Expectations, discrete probability distributions - Binomial distribution, Poisson distribution, geometric distribution, Continuous uniform distribution - exponential distribution, normal distribution, gamma distribution. Bivariate distributions and their properties, distribution of sums and quotients, conditional densities, Bayes' rule. Joint Probability distribution for two discrete random variables, expectation and covariance.				
Module 3	**Statistical Methods – CO4**	**Problem Solving**	**Case Study**	**11 Sessions**
Descriptive Statistics - Moments, skewness and Kurtosis, Correlation - Karl Pearson's coefficient of correlation and rank correlation (with & Without repetition, Multiple Correlation - Problems. Regression analysis - lines of regression, Multiple regression - Problems. Curve Fitting (Straight Line ($y = a + bx$), Parabola ($y = a + bx + cx^2$), Exponential Curves ($y = aebx$, $y = abx$ and $y = axb$)				
Module 4	**Sampling Theory – CO5, CO6**	**Assignment**	**Case Study**	**12 Sessions**
Random sampling, sampling distributions, Standard Error, Type I & Type II errors, Testing of Hypothesis, Test of significance - Large sample test for single proportion, difference of proportions, single mean, difference of means, and difference of standard deviations, Test for single mean, difference of means and correlation coefficients, test for ratio of variances - Chi-square test for goodness of fit and independence of attributes.				
Project work/Assignment: 1. Apply probability models to solve real-world engineering and business problems. 2. Perform statistical analysis using datasets and interpret results.				

3. Use R software for data visualization and statistical computations.
Topics related to **1. Problem Solving:** Apply probability and statistical techniques to analyze uncertainty, model real-world problems, and derive meaningful conclusions. **2. Employability:** Hands-on experience with statistical tools like R enables students to work in roles such as Data Analyst, Business Analyst, and Research Associate.
Textbook(s): **T1.** Douglas C. Montgomery, George C. Runger, "Applied Statistics and Probability for Engineers", 7th Edition, Wiley, 2021. **T2.** Sheldon M. Ross, "Introduction to Probability and Statistics for Engineers and Scientists", 6th Edition, Academic Press, 2021. **T3.** Jay L. Devore, "Probability and Statistics for Engineering and the Sciences", 10th Edition, Cengage Learning, 2021.
References **R1.** Robert V. Hogg, Elliot Tanis, Dale Zimmerman, "Probability and Statistical Inference", 10th Edition, Pearson, 2020. **R2.** Larry Wasserman, "All of Statistics: A Concise Course in Statistical Inference", Springer, 2020. **R3.** Peter Bruce, Andrew Bruce, Peter Gedeck, "Practical Statistics for Data Scientists", 2nd Edition, O'Reilly, 2020. **E-Resources:** 1. https://nptel.ac.in/courses/109104124 2. https://nptel.ac.in/courses/111106051 3. https://nptel.ac.in/courses/111102137

Course Code: **CSA4201**	**Course Title: Data Structures and Algorithms** **Type of Course: Program Core**	**L-T- P- C**	3	0	0	3
Version No.	1.0					
Course Pre-requisites	**NIL**					
Anti-requisites	**NIL**					
Course Description	This course provides a comprehensive understanding of advanced data structures and algorithms with emphasis on abstraction and efficient problem solving. It covers fundamental data structures, dictionary implementations, hashing techniques, trees, skip lists, and text processing algorithms. The course enables students to analyze, design, and implement efficient algorithms for real-world applications. It also introduces concepts from computational geometry and prepares students for advanced problem-solving in software development.					
Course Objective	**The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.**					
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1:** Apply fundamental concepts of data structures such as stacks, queues, lists, and graphs in problem solving. **[Apply]** **CO2:** Apply advanced abstract data types (ADT) and hashing techniques for efficient data storage and retrieval. **[Apply]** **CO3:** Analyze and implement tree-based data structures such as BST, AVL, Splay Trees, and Heaps. **[Analyze]** **CO4:** Apply string processing algorithms such as pattern matching and trie-based techniques. **[Apply]** **CO5:** Analyze algorithm efficiency and design appropriate data structures for different computational					

	problems. **[Analyze]** **CO6:** Analyze the optimized algorithmic solutions for real-world applications using appropriate data structures. **[Analyze]**	
Course Content:		
Module 1	**Review of traditional Data Structures – CO1, CO5**	**11 Sessions**
Based on the background of students, revise programming in C/C++, Stacks, Queues, Lists and Graphs (Dijekstra's algorithm, Spanning tree algorithms).		
Module 2	**Dictionaries and Hash Tables – CO2, CO5, CO6**	**11 Sessions**
Definition, Dictionary Abstract Data Type, Implementation of Dictionaries. Hashing: Review of Hashing, Hash Function, Collision Resolution Techniques in Hashing, Separate Chaining, Open Addressing, Linear Probing, Quadratic Probing.		
Module 3	**Skip Lists AND Trees – CO3, CO5, CO6**	**11 Sessions**
Need for Randomizing Data Structures and Algorithms, Binary Search Trees, AVL Trees, Splay Trees, Heap		
Module 4	**Text Processing – CO4, CO5, CO6**	**12 Sessions**
String Operations, Brute-Force Pattern Matching, The Boyer - Moore Algorithm, The Knuth-Morris-Pratt Algorithm, Standard Tries, Compressed Tries, Suffix Tries, The Huffman Coding Algorithm, The Longest Common Subsequence Problem (LCS).		

Project work/Assignment:

1. Implementation of Hashing techniques for efficient data retrieval
2. Implementation of Binary Search Tree and AVL Tree
3. Implementation of String-Matching Algorithms
4. Case study on selecting appropriate data structures for real-world applications

Topics related to

1. Problem Solving: Apply data structures and algorithmic techniques to design efficient solutions for computational problems.

2. Employability: Hands-on experience in implementing data structures and algorithms prepares students for roles such as Software Developer, Backend Engineer, and Algorithm Engineer.

Textbook(s):

T1. Mark Allen Weiss, "Data Structures and Algorithm Analysis in C++", 4th Edition, Pearson, 2021.

T2. Michael T. Goodrich, Roberto Tamassia, Michael H. Goldwasser, "Data Structures and Algorithms in C++", 2nd Edition, Wiley, 2020.

T3. Steven S. Skiena, "The Algorithm Design Manual", 3rd Edition, Springer, 2020.

References

R1. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, "Introduction to Algorithms", 4th Edition, MIT Press, 2022.

R2. Robert Sedgewick, Kevin Wayne, "Algorithms", 4th Edition (Updated), Addison-Wesley, 2021.

R3. Adam Drozdek, "Data Structures and Algorithms in C++", Cengage Learning, 2020.

E-Resources:

1. https://sites.cs.ucsb.edu/~suri/cs130a/cs130a
2. https://www.seas.upenn.edu/~swati/ee22003.html
3. https://visualgo.net/en

Course Code: **CSA4207**	**Course Title:** **Database Systems** **Type of Course:** **Program Core**	**L-T- P- C**	3	0	0	3
Version No.	1.0					

Course Pre-requisites	**NIL**	
Anti-requisites	**NIL**	
Course Description	This course provides a comprehensive foundation in database systems, focusing on both theoretical concepts and practical applications in data management. It introduces the relational data model, database design principles, and structured query language (SQL) for efficient data manipulation. The course further explores advanced database topics including normalization, transaction management, and NoSQL databases such as MongoDB and Cassandra. Students will also gain insights into emerging database systems such as object-oriented, spatial, and distributed databases. Emphasis is placed on solving real-world data management problems, enabling students to design, implement, and manage scalable database solutions for modern applications.	
Course Objective	**The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.**	
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1:** Apply fundamental database concepts, including database architecture, data models, and normalization techniques in real-world scenarios. **[Apply]** **CO2:** Apply relational database design principles using ER models, schema design, and key constraints. **[Apply]** **CO3:** Analyze and construct SQL queries for efficient data retrieval, manipulation, and database management. **[Analyze]** **CO4:** Apply database programming concepts using MySQL to solve complex data-driven problems**. [Apply]** **CO5:** Analyze NoSQL database models such as MongoDB and Cassandra for handling unstructured and distributed data. **[Analyze]** **CO6:** Evaluate advanced database systems including object-oriented, spatial, and distributed databases for modern applications. **[Evaluate]**	
Course Content:		
Module 1	**Introduction to Database – CO1**	**12 Sessions**
Introduction-Database System, Applications-Advantages of Database approach-Database Architecture-Roles of DBA,Data Independence-Database Languages- Database Design-Database Model-Data Storage and Querying- Transaction Management-serialization -concurrency control		
Module 2	**Relational Databases – CO2, CO3**	**11 Sessions**
Relational Algebra-ER Diagrams -Introduction to the Relational Model-Structure of Relational Databases- -Database Schema- Keys-Schema Diagrams- -Mapping to ER Model to Relational model- data redundancy and anomalies -Normalization-functional dependencies and types--inference rule-types of normalization-		
Module 3	**MySQL and NoSQL Database – CO4, CO5**	**11 Sessions**
Mysql and NoSQL Databases: MySQL Introduction-MySQL Features- Data types- Variables- MySQL Database Creation-Table-Queries-MySQL Clauses-views-triggers, nested queries, PL/SQL. Introduction to NoSQL: MongoDB CRUD Operation-Insert- Update-Delete-Query-Indexing-Replication-Using MongoDB with Python-Advanced MongoDB Features – Cassandra: Data Model- Table Operations, CRUD Operations.		
Module 4	**Advanced Database Systems – CO6**	**11 Sessions**
Object Oriented Databases-Need for Complex Data Types - The Object-Oriented Data Model-Object-Oriented Languages-Spatial Databases: Spatial Data Types-Spatial Relationships-Spatial Data Structures–Mobile Databases-Multimedia Databases-Overview of PostgreSQL database		
Project work/Assignment: **Mini Project - Applying database skills to a real-world application:** Create a system that helps streamline tasks, improve efficiency, and provide real-time reports on organizational operations.		

Sample coding assignments include: 1. Healthcare Management System 2. Financial Data Analysis 3. Online Learning Platform 4. Employee Management System
Topics related to **1. Problem Solving:** Apply database design techniques and query optimization strategies to solve real-world data management challenges. **2. Employability:** Hands-on experience with MySQL, MongoDB, and Cassandra tools to enhance skills for roles such as Database Developer and Data Analyst.
Textbook(s): **T1.** Carlos Coronel, Steven Morris, "Database Systems: Design, Implementation, and Management", 14th Edition, Cengage Learning, 2022. **T2.** Abraham Silberschatz, Henry F. Korth, S. Sudarshan, "Database System Concepts", 7th Edition, McGraw Hill, 2020. **T3.** Pramod J. Sadalage, Martin Fowler, "NoSQL Distilled", Updated Edition, Addison-Wesley, 2021.
References **R1.** Thomas Connolly, Carolyn Begg, "Database Systems", 7th Edition, Pearson, 2021. **R2.** Guy Harrison, "Next Generation Databases", Apress, 2020. **R3.** Dan Sullivan, "NoSQL for Mere Mortals", 2nd Edition, Addison-Wesley, 2020. **E-Resources:** 1. https://dev.mysql.com/doc/ 2. https://www.mongodb.com/docs/ 3. https://cassandra.apache.org/

Course Code: **CSA4203**	**Course Title:** **Computer Networks and Security** **Type of Course:** **Program Core**	**L-T- P- C**	3	0	0	3
Version No.	1.0					
Course Pre-requisites	**NIL**					
Anti-requisites	**NIL**					
Course Description	This course provides a comprehensive foundation in computer networking and security, focusing on both theoretical principles and practical applications. It covers networking models, signal transmission, error detection, routing protocols, and transport mechanisms. The course further explores network security concepts, cryptographic techniques, and secure communication protocols such as SSL/TLS. Emphasis is placed on real-world applications, enabling students to design secure and efficient communication systems.					
Course Objective	**The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.**					

Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1:** Apply core networking concepts including OSI and TCP/IP models, signal transmission, and error detection techniques. **[Apply]** **CO2:** Apply network layer and transport layer protocols including IPv4, IPv6, routing algorithms, and flow control mechanisms. **[Apply]** **CO3:** Analyze network security fundamentals including security principles, attacks, and cryptographic techniques. **[Analyze]** **CO4:** Analyze symmetric and asymmetric cryptographic algorithms and secure communication protocols such as SSL/TLS. **[Analyze]** **CO5:** Apply routing, flow control, and error handling mechanisms to design efficient communication systems. **[Apply]** **CO6:** Apply network security strategies and cryptographic solutions for secure data transmission in real-world scenarios. **[Apply]**	
Course Content:		
Module 1	**Fundamentals of Data Communication and Error Control - CO1, CO2, CO5**	**11 Sessions**
Physical Layer: Introduction - Network Models: OSI Model - TCP/ IP Protocol Suite. Data and Signals: Basics of Analog and Digital Signals - Transmission Impairment. Guided and Unguided Media - Circuit Switched Networks - Datagram Networks. Data Link Layer: Error Detection and Correction: Types of Errors –Parity Check, Two-Dimensional Parity Check, Checksum and CRC, Hamming Distance.		
Module 2	**Concepts of Network Communication and Protocols – CO1, CO2, CO5**	**11 Sessions**
Network Layer: IPv4 – Subnetting, Routing - Distance Vector Routing – Link State Routing, IPv6. Transport and Application Layer: Flow control - Sliding Window, Go-Back N ARQ, Selective Repeat ARQ. UDP, TCP, Congestion Control.		
Module 3	**Foundations of Network Security and Cryptographic Techniques – CO3, CO4, CO6**	**11 Sessions**
Security Concepts: Introduction, The need for security, Security approaches, Principles of security, Types of Security attacks, Security services, Security Mechanisms, A model for Network Security. Cryptography Concepts and Techniques: Introduction, plain text and cipher text, substitution techniques, transposition techniques, encryption and decryption, symmetric and asymmetric key cryptography, steganography, possible types of attacks.		
Module 4	**Cryptographic Techniques and Secure Communication Protocols – CO3, CO4, CO6**	**12 Sessions**
Symmetric key Ciphers: Block Cipher principles, DES, AES, Blowfish, IDEA. Asymmetric key Ciphers: RSA algorithm, Diffie-Hellman Key Exchange. Transport-level Security: Web security considerations, Secure Socket Layer and Transport Layer Security.		
Project work/Assignment: 1. Solve numerical problems on Parity Check, CRC, and Hamming Code error detection. 2. Explain Distance Vector Routing and Link State Routing with examples. 3. Manually encrypt and decrypt text using substitution and transposition ciphers. 4. Demonstrate RSA encryption and decryption with numerical examples. 5. Project: Subnetting, Routing Algorithms, and Transport Protocols 6. Project: Symmetric, Asymmetric Encryption and Web Security Protocols		
Topics related to **1. Problem Solving:** Apply networking and cryptographic techniques to solve real-world communication, data security, and transmission challenges. **2. Employability:** Hands-on exposure to networking protocols, routing techniques, and cryptographic implementations prepares students for roles in networking, cybersecurity, and system administration.		
Textbook(s): **T1.** Behrouz A. Forouzan, “Data Communications and Networking”, 6th Edition, McGraw Hill, 2021.		

T2. William Stallings, "Cryptography and Network Security: Principles and Practice", 8th Edition, Pearson, 2023.
T3. Charlie Kaufman, Radia Perlman, Mike Speciner, "Network Security: Private Communication in a Public World", 3rd Edition, Pearson, 2021.

References

R1. James F. Kurose, Keith W. Ross, "Computer Networking: A Top-Down Approach", 8th Edition, Pearson, 2023.
R2. Andrew S. Tanenbaum, David J. Wetherall, "Computer Networks", 6th Edition, Pearson, 2021.
R3. Behrouz A. Forouzan, Debdeep Mukhopadhyay, "Cryptography and Network Security", McGraw Hill, 2021.

E-Resources:

1. https://www.netacad.com/catalogs/learn/networking
2. https://inl.info.ucl.ac.be/cnp3.html
3. https://github.com/ssllabs/research/wiki/SSL-and-TLS-Deployment-Best-Practices

Course Code: CSA4303	**Course Title:** Web Technology **Type of Course:** Program Core	**L-T- P- C**	1	0	4	3
Version No.	1.0					
Course Pre-requisites	Basic Programming and Database Concepts.					
Anti-requisites	**NIL**					
Course Description	This course introduces the fundamental concepts and architecture of the World Wide Web. It enables students to design and develop dynamic and interactive web applications using HTML, CSS, JavaScript, and PHP. The course focuses on both client-side and server-side technologies, along with modern frameworks such as Bootstrap. Emphasis is placed on building responsive web interfaces, integrating databases, and developing full-stack web applications for real-world use cases.					
Course Objective	**The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.**					
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1:** Apply Internet concepts and develop static web pages using HTML. **[Apply]** **CO2:** Apply CSS and Bootstrap frameworks to design responsive web interfaces. **[Apply]** **CO3:** Apply client-side scripting using JavaScript and jQuery for dynamic web applications. **[Apply]** **CO4:** Apply server-side scripting using PHP to develop database-driven web applications. **[Apply]** **CO5:** Analyze and integrate front-end and back-end technologies to build complete web solutions. **[Analyze]** **CO6:** Analyze and design scalable and interactive web applications for real-world scenarios. **[Analyze]**					
Course Content:						
Module 1	**Introduction to Internet Standards and HTML – CO1, CO5**				**15 Sessions (L-3, P-12)**	
Basics Of Internet Client/Server Computing: Introduction to WWW, WWW Architecture, Web Browsers, Web servers, SMTP, POP3, MIME, File Transfer Protocol, Overview of HTTP, HTTP request-response, Types of Web servers, Error Response Codes. Markup Language (HTML): Introduction to HTML and HTML5, Basic Structure of HTML Page, Formatting, Commenting, Anchors, Images, Hyperlinks, Lists, Tables, HTML Forms.						
Module 2	**UI Design – CO2, CO5, CO6**				**18 Sessions (L-4, P-14)**	
Cascading Style Sheet (CSS): The need for CSS, Introduction to CSS, Basic syntax and structure, Inline Styles, Embedding Style Sheets, Linking External Style Sheets, Levels of CSS, Selectors, Font, color and Text Properties, BOX Model Backgrounds, manipulating text, Margins, and Padding - Positioning using CSS. Responsive Design, CSS frameworks.						

Module 3	Introduction to JavaScript – CO3, CO5, CO6	20 Sessions (L-4, P-16)
Introduction to Client-Side Scripting, JavaScript Features, Programming Constructs, Arrays and Functions, Document Object Model, Event Handling, Browser functions, Form handling and Validation. Introduction to JQuery, Syntax, JQuery Fundamentals, Event handling, JQuery Event Model		
Module 4	**Server-Side Development – CO4, CO5, CO6**	**22 Sessions (L-4, P-18)**
Introduction to server-side Development with PHP, PHP structure, Data Types, Arrays, $GET and $POST, Reading/Writing Files, PHP Sessions and Objects, Object Oriented Design, Working with Databases, SQL, Database APIs, Managing MySQL Database. Accessing MySQL in PHP.		

Project work/Assignment:

1. Develop a complete web application integrating frontend and backend technologies
2. Build responsive web pages using HTML, CSS, and Bootstrap
3. Implement dynamic features using JavaScript and jQuery
4. Develop database-driven applications using PHP and MySQL

Topics related to

1. Problem Solving: Design and implement web applications using modern web technologies to solve real-world problems.

2. Employability: Hands-on experience with full-stack development tools such as HTML, CSS, JavaScript, PHP, and MySQL prepares students for roles like Web Developer and Full Stack Developer.

List of Laboratory Tasks:

Experiment No. 1: Demonstration of HTML document and formatting tags

Level 1: Design a paragraph about MCA course. Bold, italicize this text and set font color and size also.

Level 2: Design a page with a background image and demonstrate all attributes of a background image. Set the image properties.

Experiment No. 2: Demonstration of HTML List

Level 1: Design an unordered list and an ordered list of different items.

Level 2: Develop a web page with a Menu and a nested menu with ordered and un-ordered lists.

Experiment No. 3: Demonstration of HTML Hyper Link

Level 1: Design a web page with different schools of the Presidency University. When user clicks on the school, the next page should display the details about the school.

Level 2: Design and develop static web pages for an online bookstore and link all pages and show the link on same page also.

Experiment No. 4: Demonstration of HTML table tag

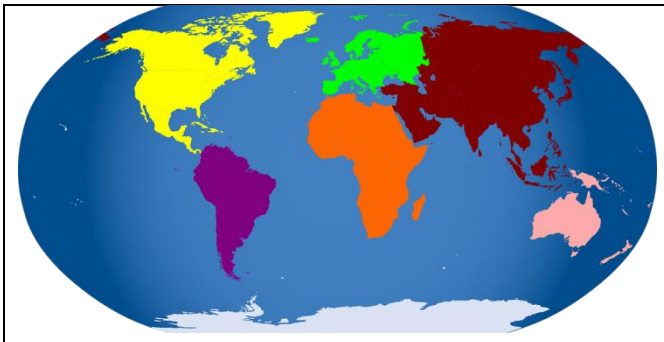
Level 1: Demonstrate the various courses of the university and link those courses using a table and a link tag.

Level 2: Design a Resume page using table and image tags. The page should have all necessary information, photo and signature also.

Experiment No. 5: Demonstration of HTML frame,iframe and IMagemap

Level 1: Design a page using frame and iframe.

Level2: Demonstrate image mapping for any four countries.



Experiment No. 6: Demonstration of HTML form

Level 1: Design a Login form with a submit and reset button, When the user clicks on the Submit button, it displays the message "Login successful.", reset button clears the form.

Level 2: The University is organizing a cultural festival, and the organizing team wants to collect registrations for various events with the help of a web page. Design a registration form for collecting the participant details.

Experiment No. 7: Demonstration of different types of CSS

Level 1: Design the webpage by applying the different styles using inline stylesheets

Level 2: Design the webpage by applying the different styles using external & internal style sheets.

Experiment No. 8: Demonstration of CSS image styles

Level 1: Create a web page to change the background color of elements.

Level 2: Create a web page to set the background image, and repeat the image horizontally and fixed background image.

Experiment No. 9: Application of CSS in web designing

Level 1: Design a document using HTML and CSS to create a catalog of items for online shopping.

Level 2: Create an HTML document for employees' information in a table and display the same using a cascaded style sheet.

Experiment No. 10: Application of CSS in web designing

Level1: Design a Web page to show various Cvarious CSSerties

Level 2: Create a following calculator interface with HTML and CSS

Experiment No. 11: Application of Bootstrap

Level 1: Design a basic Bootstrap page with a responsive fixed width container.

Level 2: Design a basic Bootstrap page with Bootstrap Grid Structure.

Experiment No. 12: Application of Bootstrap

Level 1: Design an Image gallery page using Bootstrap Image**.**

Level 2: Create a panel with Hello World using Bootstrap**.**

Experiment No. 13: Demonstration of JavaScript

Level 1: Write a Java Script program that on clicking a button, displays scrolling text which moves from left to right with a small delay

Level 2: Write a JavaScript code to change the background color at frequent intervals

Experiment No. 14: Demonstration of JavaScript

Level 1: To write a JavaScript program to define a user defined function for sorting the
values in an array. Use HTML5 for the user interface.

Level 2: Develop and demonstrate JavaScript with POP-UP boxes and functions for the following problems:

a) Input: Click on the Display Date button using onclick ()
function Output: Display date in the textbox

b) Input: A number n obtained using prompt
Output: Factorial of n number using alert

c) Input: A number n obtained using prompt and add
another number using confirm
Output: Sum of the entire n numbers using alert

Experiment No. 15: Demonstration of JavaScript

Level 1: Write a JavaScript to change the size of the image when the user clicks on the button.

Level 2: Write a JavaScript program to open a web page after confirming from the user. Otherwise, the window should be closed. Use confirms, open, and close methods

Experiment No. 16: Demonstration of JavaScript Validation

Level 1: Write a JavaScript program to give access to some web pages only by presidency University students.

Level 2: Write JavaScript to validate the following fields of the above registration page.

1. Name (Name should contains alphabets and the length should not be less than 6 characters).
2. Password (Password should not be less than 6 characters length).
3. E-mail id (should not contain any invalid characters and must follow the standard pattern (name@domain.com)
4. Phone number (Phone number should contain 10 digits only).

Experiment No. 17: Demonstration of jQuery

Level 1: Develop a page to show Blink text using jQuery.

Level 2: Write a code to Disable right click menu in html page using jquery.

Experiment No. 18: Demonstration of jQuery

Level 1: Develop the page to detect when a textbox's content has changed using jQuery.

Level 2: Develop a page to set the background-image using the jQuery CSS property.

Experiment No. 19: Demonstration of jQuery

Level 1: Develop the page to detect when a textbox's content has changed using jQuery.

Level 2: Develop a page to set the background-image using the jQuery CSS property.

Experiment No. 20: Demonstration of jQuery

Level 1: Write a code to access HTML form data using jQuery.

Level 2: Design a page to animate an element, by changing its height and width using jQuery

Experiment No. 21: Web design using PHP

Level 1: Write a PhP program to Get name of the user from a form and show greeting text.

Level 2: Write a PHP Script to find out the Sum of the Individual Digits.

Experiment No. 22: Web design using PHP

Level 1: Write a PHP Program to display current Date, Time and Day.

Level 2: Write a php program to find largest values of two numbers using nesting of function.

Experiment No. 23: **Web design using PHP**

Level 1: Write a php program to show Array manipulation.

Level 2: A web application that takes a name as input and on submit it shows a hello <name> page where name is taken from the request. It shows the start time at the right top corner of the page and provides a logout button. On clicking this button, it should show a logout page with Thank You <name > message with the duration of usage (hint:Use session to store name and time).

Experiment No 24: Web design using PHP

Level 1: Write a PHP program to read the personal information of a person such as first name, last name, age, permanent address, and pin code entered by the user into a table created in MySQL. Read the same information from the database and display it on a web page

Level 2: Using PHP develop a web page that accepts book information such as ISBN number, title, authors, edition, and publisher and store information submitted through the web page in MySQL database.

Experiment No 25: Web design using PHP

Level 1: Write a php program to Read from existing file

Level 2: Create a web page to advertise a product of the company using images and audio.

Experiment No 26: PHP Cookies and Filters

Level 1: Write a PHP program to show Cookie concepts

Level 2: Write a PHP program to show different filters.

Experiment No 27: PHP session

Level 1: Write a PHP program to show session concepts

Level 2: Write a PHP program to store page views count in SESSION, to increment the count on each refresh, and to show the count on web page.

Experiment No. 28: Building a website.

Level 1: Build a website for organizing an International Conference. The conference website must be able to collect the author's details and upload a file.

Level 2: Develop the PHP code for partial web pages for ordering vegetables from Bigbasket

Textbook(s):

T1. Paul Deitel, Harvey Deitel, Abbey Deitel, "Internet & World Wide Web: How to Program", 5th Edition, Pearson, 2021.
T2. Robin Nixon, "Learning PHP, MySQL & JavaScript", 6th Edition, O'Reilly, 2021.
T3. Jon Duckett, "HTML and CSS: Design and Build Websites", Updated Edition, Wiley, 2022.

References

1. **R1.** Ivan Bayross, "HTML, DHTML, JavaScript, Perl CGI", BPB Publications, 2022.
2. **R2.** John Pollock, "JavaScript: A Beginner's Guide", 5th Edition, McGraw Hill, 2020.
3. **R3.** Ben Frain, "Responsive Web Design with HTML5 and CSS", 4th Edition, Packt, 2022.

E-Resources:

1. https://www.w3schools.com/
2. https://www.tutorialspoint.com/internet_technologies
3. https://developer.mozilla.org/

Course Code: CSA4305	**Course Title:** Advanced Python Programming **Type of Course:** Program Core – Lab Integrated	**L-T- P- C**	1	0	4	3
Version No.	1.0					
Course Pre-requisites	Basic Python Programming.					
Anti-requisites	**NIL**					
Course Description	This course focuses on advanced Python programming concepts for application development and data-driven problem solving. It covers object-oriented programming, modular programming, multithreading, data analysis, machine learning basics, visualization, GUI development, and web scraping. The course emphasizes hands-on learning through experiments and projects, enabling students to build intelligent applications across domains such as data science, machine learning, and automation.					
Course Objective	**The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.**					
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1:** Apply object-oriented programming concepts in Python to develop modular and reusable applications. **[Apply]** **CO2:** Apply advanced Python features such as modules, multithreading, and functional programming. **[Apply]** **CO3:** Apply data analysis and visualization techniques using Python libraries such as NumPy, Pandas, and Matplotlib. **[Apply]** **CO4:** Apply machine learning techniques and database connectivity using Python for real-world applications. **[Apply]** **CO5:** Analyze and integrate Python-based tools for GUI development, web scraping, and data processing. **[Analyze]** **CO6:** Analyze and design end-to-end Python-based solutions for data-driven and intelligent applications. **[Analyze]**					
Course Content:						
Module 1	**Object Oriented Programming Concepts – CO1, CO2, CO5**					**21 Sessions (L-5, P-16)**
Overview of Python basics and syntax, Variables, data types, conditional statements, Lists, Tuples, Sets, Dictionary, Functions, Overview of object-oriented programming (OOP) concepts and principles.						
Module 2	**Advanced Python Concepts – CO2, CO5, CO6**					**20 Sessions (L-4, P-16)**
Regular Expressions and Pattern Matching, Python Modules, Creating and importing own modules, Multithreading, Multiprocessing, Sharing Data between processes, Python Testing Frameworks, Lambda functions, Map functions.						
Module 3	**Python Essentials for Data Analysis and Machine Learning – CO3, CO4, CO5, CO6**					**20 Sessions (L-4, P-16)**
Introduction to NumPy: Vectors, Matrix, Matrix manipulation, Array operations, Slicing, Basic data analysis using Pandas, Panda's data structures, Machine Learning, Regression Analysis, Data Cleaning, Data Visualization using Matplotlib, Seaborn, Plotly, MySQL Connectivity.						
Module 4	**GUI Development, Case Study and Project – CO3, CO4, CO5, CO6**					**14 Sessions (L-2, P-12)**
Building interfaces with Tkinter or PyQt, GUI Building Libraries, Digital Image Processing, Web Scraping and Data Analysis - Case Study: Python for Data Analysis, Python for Data Visualization and Interactive Dashboards.						
Project work/Assignment: 1. Develop Python-based applications using OOP and modular programming						

2. Perform data analysis and visualization using real-world datasets
3. Implement machine learning models using Python
4. Develop GUI-based and web scraping applications
5. End-to-end project: Data collection → processing → visualization → prediction

Topics related to

1. Problem Solving: Solve real-world computational and data-driven problems using Python programming techniques.**2. Employability:** Hands-on experience with Python, machine learning, data analysis, and GUI tools prepares students for roles such as Data Analyst, Python Developer, and AI/ML Engineer.

Experiment 1: Exploring Lists and Tuples

LEVEL 1 Demonstrate creation and manipulation of Lists, Tuples.
LEVEL 2 Perform operations like sorting a list of dictionaries, filtering sets.

Experiment 2: Exploring sets and dictionaries.
LEVEL 1 Demonstrate creation and manipulation of Sets, and Dictionaries.
LEVEL 2 Perform operations like aggregating data (e.g., summing values).

Experiment 3: Looping and Conditional statements

LEVEL 1 Create a Floyd's Triangle [odd, even, center and reverse.
LEVEL 2 Implement Simple Calculator using Conditional Statements.

Experiment 4: Regular expressions for pattern matching

LEVEL 1 Write a Python program that matches a word at the beginning and ending of a string.
LEVEL 2 Validate Email Address with Regular Expressions (RegEx)

Experiment 5: Regex Functions

LEVEL 1 Write a Python program that Validate a String Using Regular Expressions
LEVEL 2 Extract All Dates in DD-MM-YYYY Format

Experiment 6: Modules and Packages

LEVEL 1 Create a module math_utils.py with functions for factorial, prime checking, and GCD.
LEVEL 2 Use datetime, os, and sys modules in a script to show file stats and runtime info.

Experiment 7: Building a Custom Python Module

LEVEL 1 Develop a Python module with utility functions for mathematical operations (e.g., prime checking, matrix addition).
LEVEL 2 Develop a Python module with utility functions for mathematical operations USING matrix addition.

Experiment 8: Multiprocessing

LEVEL 1: Perform Print squares and find cubes and sum of numbers using multi-processing.
LEVEL 2 Use multiprocessing to perform heavy computations (e.g., matrix multiplication) on a dataset.

Experiment 9: Functional Programming

LEVEL 1Create a Lambda function that adds 15 to a given number.
LEVEL 2Create a Lambda function that multiplies two arguments and prints the result and use a lambda function inside another function.

Experiment 10: Arrays in Numpy

LEVEL 1 Create a NumPy array from a list and print its shape, size, and datatype.

LEVEL 2 Perform Array operations like Slice Arrays in Multiple Dimensions,Reshaping and Sorting Arrays.

Experiment 11: Data Analysis on a Real-world dataset.

LEVEL 1 Load a Weather Data Analysis Using NumPy.
Dataset: Weather data (temperature, humidity, rainfall)
LEVEL 2 Perform basic statistical and mathematical analysis on temperature and humidity, identify trends over time.

Experiment 12: Advanced Pandas Operations

LEVEL 1 Demonstrate data cleaning by handling missing values, duplicates, and outliers in a large dataset.
LEVEL 2 Analyze temporal trends in stock price data using Pandas time series methods.

Experiment 13: Data Exploration and Cleaning

LEVEL 1 Load and analyze a weather dataset using Pandas
LEVEL 2 Explore the ways to detect NaN values in Python, using NumPy and Pandas.

Experiment 14: Regression Analysis in Machine Learning

LEVEL 1 Given a data set from UCI repository, implement the simple linear regression algorithm
LEVEL 2Plot the learning curves using Matplotlib and seaborn

Experiment 15: Predictions using Machine Learning

LEVEL 1 Train a simple Linear regression model using Scikit-learn
LEVEL 2 To predict house prices and visualize the line of best fit.

Experiment 16: Sports performance using Matplotlib

LEVEL 1 Create interactive plots to track the performance of athletes over time
LEVEL 2 Use maps to visualize geospatial data.

Experiment 17: Data Visualization using Seaborn library

LEVEL 1Load iris dataset using the Seaborn library in Python.
LEVEL 2Create effective visualizations using the Seaborn library in Python.Add titles, color palettes, style, or size arguments.

Experiment 18: Data Visualization with Interactive Dashboards

LEVEL 1 Create interactive dashboards using Plotly
LEVEL 2 Use maps (via Plotly) to visualize geospatial data.

Experiment 19: MySQL Database Connectivity

LEVEL 1 Connect to a MySQL/SQLite database
LEVEL 2 Perform CRUD operations and display database records.

Experiment 20: Student Database using MySQL with Python

LEVEL 1 Connect MySQL database and create table for Student.
LEVEL 2 Perform Student record management system and manage student profiles and grades.

Experiment 21: GUI Development using Tkinter

LEVEL 1Create a GUI to accept and display user input.
LEVEL 2Add a button labeled “Submit” and while clicking the button display the entered name.

Experiment 22: GUI Development using Tkinter

LEVEL 1Create a window with a title, fixed size, and a simple label.
LEVEL 2Accept username and password and display login status (e.g., “Login successful” or “Invalid credentials”).Add a button to close the window.

Experiment 23: GUI Development using Tkinter

LEVEL 1 Create a student database using MySQL and perform CRUD operations
LEVEL 2 Build a GUI-based calculator using Tkinter

Experiment 24: Web Scraping:BeautifulSoup

LEVEL 1 Fetch the HTML content of a webpage and parse it using BeautifulSoup.
LEVEL 2Extract Links and images from a Website

Experiment 25: Web Scraping and Data Analysis

LEVEL 1 Create a Patient database using MySQL and perform CRUD operations
LEVEL 2 Build a GUI-based calculator using Tkinter

Experiment 26: Testing and Debugging using employee database

LEVEL 1 Create a employee database using MySQL and perform CRUD operations
LEVEL 2 Build a GUI-based calculator using Tkinter

Experiment 27: Testing and Debugging using library database

LEVEL 1 Create a library database using MySQL and perform CRUD operations
LEVEL 2 Build a GUI-based calculator using Tkinter

Experiment 28: Web Scraping and Data Analysis

LEVEL 1 Scrape data from a live website (e.g., weather data, product prices) using BeautifulSoup or Scrapy. Save the data as a CSV.
LEVEL 2 Use Pandas to clean, manipulate, and analyze the scraped data. Visualize findings with Matplotlib and Seaborn.

Experiment 29: Testing and Debugging

LEVEL 1 Writing unit tests with unittest or pytest
LEVEL 2Debugging techniques and tools

Experiment 30: End-to-End Data Analysis Project

LEVEL 1 Identify a dataset from Kaggle (e.g., Global Warming Data). Scrape, clean, and preprocess the data.
LEVEL 2 Visualize insights using an interactive dashboard or multi-chart report. Include predictive analysis using a simple ML model.

Textbook(s):

T1. Mark Lutz, “Learning Python”, 5th Edition (Updated), O’Reilly Media, 2025.
T2. Quan Nguyen, “Advanced Python Programming”, 2nd Edition, Packt Publishing, 2021.
T3. Fabio Nelli, “Python Data Analytics with Pandas, NumPy and Matplotlib”, 2nd Edition, Apress, 2021.
T4. Paul Deitel, Harvey Deitel, “Python for Programmers”, Pearson, 2021.

References

R1. John V. Guttag, “Introduction to Computation and Programming Using Python”, 3rd Edition, MIT Press, 2021.
R2. Eric Matthes, “Python Crash Course”, 2nd Edition (Updated), No Starch Press, 2021.
R3. Aurélien Géron, “Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow”, 3rd Edition, O’Reilly, 2022.
E-Resources:

1. https://www.python.org/
2. https://nptel.ac.in/courses/
3. https://www.coursera.org/learn/python
4. https://www.udemy.com/course/

Course Code: CSA4301	**Course Title:** Data Structures and Algorithms Lab **Type of Course:** Lab Oriented	**L-T- P- C**	0	0	2	1
Version No.	1.0					
Course Pre-requisites	**NIL**					
Anti-requisites	**NIL**					
Course Description	This course will provide exposure to understand the ADT/libraries, the necessary mathematical abstraction and choose appropriate data structures. It familiarizes students with advanced data structures and paradigms. Course includes theory as well as practical components. Topics to Include: Review of traditional data structures, Dictionaries, Implementation of Dictionaries. Hashing, Skip Lists, Trees, Text Processing and introduction to Computational Geometry					
Course Objective	**The objective of the course is to familiarize the learners with the concepts of Advanced Data Structure and Algorithms and attain Skill Development through Experiential Learning techniques.**					
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1:** Apply abstract data types (ADT) such as stacks, queues, lists, and graphs in solving computational problems. **[Apply]** **CO2:** Apply hashing techniques and dictionary implementations for efficient data storage and retrieval. **[Apply]** **CO3:** Apply tree-based data structures such as heaps, binary trees, and Red-Black trees in algorithm design**. [Apply]** **CO4:** Apply graph algorithms such as shortest path and minimum spanning tree for problem solving. **[Apply]** **CO5:** Analyze and implement advanced data structures such as Fibonacci heaps, binomial queues, and tries. **[Analyze]** **CO6:** Analyze and design optimized algorithmic solutions using appropriate data structures for real-world applications. **[Analyze]**					

List of Experiments:

Experiment 1: Implementing Lists, stacks in C++

Level 1: Understanding the concepts of lists and stacks

Level 2: Implementation of a significant subset of STL vector and list classes

Experiment No. 2: Implementing Queues and Graphs

Level 1: Prioritize the significance of Queue structure

Level 2: Implementation of a significant subset of STL vector and list classes

Experiment No. 3: Implementations of dictionaries & Hash Tables

Level 1: Building the classic algorithms quadratic probing and separate chaining

Level 2: Analyzing the cuckoo hashing concepts and hopscotch hashing

Experiment No. 4: Implementation Tree algorithms and Heap

Level 1: Importance of general-purpose Tree algorithms

Level 2: Implementation of binary heaps and concepts of pairing heap

Experiment No. 5: Implementation of sorting algorithms

Level 1: Understanding of sorting conditions with comparison-based sorting**.**

Level 2: Implementation of methods to solve O(N2) and O(N log N) sorting algorithms

Experiment No. 6: Implementation Minimum spanning Tree

Level 1: Analyze an MST with minimum possible total edge weight

Level 2: Implement MST for Prim's algorithm and Kruskal's algorithm

Experiment No. 7: Implementation of Shortest-Path Algorithms

Level 1: Analyze the representation of Graphs, Acyclic graphs and negative edge costs

Level 2: Implementation of unweighted shortest path and Dijkstra's Algorithm

Experiment No. 8: Implementation of Fibonacci heaps

Level 1: Analyze the concepts of lazy merging for Binomial Queues and Implement the Fibonacci heap operations

Level 2: Implement proof of the Time bound in Fibanacci Heaps

Experiment No. 9: Implementation of Binominal Queues

Level 1: Understanding of Binomial Queue structure and Binomial Queues operations

Level 2- Amortized Analysis of Data structures is carried on queues

Experiment No. 10: Implementation of Huffman coding Algorithm

Level 1- Apply simple scheduling problem (Greedy algorithms)

Level 2- Implement Huffman codes and with approximate Bin packing

Experiment No. 11: Implementation of Knuth-Morris-Pratt (KMP) Algorithm

Level 1- Analyze of worst-case complexity of KMP algorithm o(n+m) and comparision with Naïve algorithm worst case complexity of O(m(n-m+1)).

Level 2- Implement the pattern searching using KMP algorithm

Experiment No. 12: Implementation coding Tries, Suffix Tries

Level 1- Applying tree-like data structure that stores and retrieves strings based on shared prefixes

Level 2- Implement Suffix Tries for searching large sequences like genomes.

Experiment No. 13: Implementation of Longest common Subsequence Problem (LCS)

Level 1- Calculate the common subsequence for the given set of strings (Computational problems)

Level 2- Implement the problems of DNA sequence analysis using LCS algorithm

Experiment 14: KMP Algorithm

Level 1: Implement LPS Array Construction
Level 2: Analyze the Advanced Data Structures concepts of Red-Black trees with Bottom-up Insertion, Top-Down Red-Black

Experiment 15: LCS

Level 1: Compute the length of the Longest Common Subsequence (LCS) between two given strings using the Dynamic Programming approach. Clearly illustrate the construction of the DP table and explain how the final LCS length is obtained

Level 2 : Determine and print the Longest Common Subsequence (LCS) of two given strings using Dynamic Programming. Construct the DP table to compute the LCS length and demonstrate the backtracking process to retrieve and display the actual LCS sequence

Textbook(s):

T1. Mark Allen Weiss, *"Data Structures and Algorithm Analysis in C++"*, 4th Edition, Pearson, 2021.
T2. Michael T. Goodrich, Roberto Tamassia, Michael H. Goldwasser, *"Data Structures and Algorithms in Python"*, Wiley, 2021.
T3. Robert Sedgewick, Kevin Wayne, *"Algorithms"*, 4th Edition (Updated), Addison-Wesley, 2021.

References

R1. Mark Allen Weiss, "Data Structures and Algorithm Analysis in C++", 4th Edition, Pearson, 2021.
R2. Thomas H. Cormen et al., "Introduction to Algorithms", 4th Edition, MIT Press, 2022.
R3. Robert Sedgewick, Kevin Wayne, "Algorithms", Addison-Wesley, 2021.

E-Resources:

1. https://visualgo.net
2. https://www.geeksforgeeks.org/data-structures/
3. https://www.programiz.com/dsa

Course Code: CSA4307	**Course Title:** Database Systems Lab **Type of Course:** Lab Oriented	**L-T- P- C**	0	0	2	1
Version No.	1.0					
Course Pre-requisites	**Should be selected only from the subjects studied by the students in the previous semesters.**					
Anti-requisites	**NIL**					
Course Description	This course is designed to introduce the fundamental concepts of database design, implementation, and utilization, covering topics like data models, MySQL, database design principles, and transaction management. It presents the fundamental concepts of database design and use. It provides a study of data models, data description languages, and query facilities including relational algebra and SQL, data normalization, transactions and their properties, physical data organization and indexing, security issues and object databases. It also looks at the new trends in databases.					

Course Objective	**The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.**
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1:** Apply database concepts using open-source DBMS tools such as MySQL, MongoDB, and Cassandra. **[Apply]** **CO2:** Apply Data Definition Language (DDL) and Data Manipulation Language (DML) commands to create and manage databases. **[Apply]** **CO3:** Apply SQL queries to retrieve, filter, and manipulate structured data efficiently. **[Apply]** **CO4:** Apply NoSQL database concepts for handling unstructured and semi-structured data**. [Apply]** **CO5:** Analyze database operations including indexing, aggregation, and optimization techniques. **[Analyze]** **CO6:** Evaluate and design database solutions for real-world applications using relational and NoSQL systems. **[Evaluate]**

Experiment No. 1:

SmartMart Database Setup – Create database, tables (customers, products, suppliers, employees, orders, order_items, payments) with PK/FK constraints.

Level 1:

1. Create a database named SmartMart.
2. Create the following tables with appropriate **data types**, **primary keys**, and **foreign key constraints**:
    - customers (customer_id, name, email, phone, city, membership_type)
    - products (product_id, name, category, price, stock_quantity, supplier_id)
    - suppliers (supplier_id, name, contact_person, phone, city)
    - employees (employee_id, name, designation, salary, hire_date, manager_id)
    - orders (order_id, customer_id, employee_id, order_date, total_amount, status)
    - order_items (order_item_id, order_id, product_id, quantity, unit_price)
    - payments (payment_id, order_id, payment_date, amount, method)
3. Define foreign keys:
    - products.supplier_id references suppliers(supplier_id)
    - orders.customer_id references customers(customer_id)
    - orders.employee_id references employees(employee_id)
    - order_items.order_id references orders(order_id)
    - order_items.product_id references products(product_id)
    - payments.order_id references orders(order_id)

Level 2:

1. Add a **CHECK constraint** on products.price (> 0) and orders.total_amount (>= 0).
2. Write the SQL statements and execute them.

Experiment No. 2:

Inserting Sample Records & Basic Retrieval – INSERT, SELECT, display customer purchase history, product price lists, employee records

Level 1:

1. Insert at least **5 records** into each table (customers, suppliers, employees, products, orders, order_items, payments). Ensure referential integrity.
2. Write queries to:
    - Display all customer details.
    - Display product name, price, and category for all products.
    - Show employee names and designations.
3. Display **purchase history** for a specific customer (given customer_id) – show order_id, order_date, product_name, quantity, unit_price.

Level 2:

1. List all orders placed after a specific date (e.g., '2026-10-01').

2. Show all products with price less than ₹500.

Experiment No. 3:

Filtering & Sorting Data – WHERE, BETWEEN, ORDER BY, LIKE; price range queries, customer location sorting, monthly sales extraction

Level 1:

1. Retrieve products whose **price is between ₹200 and ₹1000**. Sort by price descending.
2. List customers from **'Bangalore'** and **'Mumbai'** only, sorted by name alphabetically.

Extract **monthly sales records** for October 2026 – show order_date, total_amount.

Use WHERE with BETWEEN '2026-10-01' AND '2026-10-31'.

Level 2:

1. Find employees whose **salary > 40000** and designation contains 'Manager'. Sort by salary descending.
2. Display orders where **status is 'Pending'** or 'Shipped' and total_amount > 1000

Experiment No. 4:

Aggregate Functions & Grouping – SUM, COUNT, AVG, GROUP BY, HAVING; total monthly sales, city-wise customer counts, top-selling categories

Level 1:

1. Compute **total monthly sales** (sum of total_amount) for each month of 2026. Group by month.
2. Find **city-wise customer count**. Display city and number of customers.
3. Identify **top-selling product categories** – display category and total quantity sold (sum of order_items.quantity). Join orders, order_items, products. Group by category and sort by total quantity descending. Limit to top 3.

Level 2:

1. Calculate **average salary of employees** by designation.
2. Show **number of orders placed by each customer** (customer name, count of orders). Exclude customers with zero orders.

Experiment No. 5:

Multi-table Operations (Joins & Subqueries) – INNER JOIN, LEFT JOIN, subqueries; customer order summaries, employees with most orders, products never purchased

Level 1:

1. Generate **customer order summaries** with product details – show customer_name, order_id, order_date, product_name, quantity, unit_price. Use INNER JOIN.
2. Find **employees who handled the highest number of orders** – display employee name, count of orders handled. Use LEFT JOIN and ordering.
3. Identify **products that have never been purchased** – use NOT EXISTS or LEFT JOIN ... WHERE product_id IS NULL.

Level 2:

1. List **suppliers who supply products that have never been ordered** – use subquery.
2. Write a query to find **customers who have placed orders worth more than ₹5000 in total** (using subquery with GROUP BY and HAVING).

Experiment No. 6:

Database Views – CREATE VIEW for customer order summaries, monthly sales, top-selling products, employee performance reports

Level 1:

1. Create a view CustomerOrderSummary that shows customer_name, order_id, order_date, total_amount.
2. Create a view MonthlySalesReport that shows month, year, total_sales (sum of total_amount).
3. Create a view TopSellingProducts that shows product_name, total_quantity_sold, rank them by quantity

Level 2:

1. Create a view EmployeePerformance that shows employee_name, number_of_orders_handled, total_sales_generated.
2. Query the views to verify. Then, attempt to **update** a view (e.g., change customer name via view) and explain the result.

Experiment No. 7:

Triggers – AFTER INSERT on orders to reduce inventory; BEFORE INSERT to prevent order when stock unavailable; AFTER DELETE log for customers

Level 1:

1. Create a trigger reduce_inventory – **AFTER INSERT** on order_items. For each inserted row, decrease products.stock_quantity by the quantity ordered.
2. Create a trigger prevent_order_if_stock_unavailable – **BEFORE INSERT** on order_items. Check if the product's stock_quantity is sufficient; if not, raise an error (SIGNAL SQLSTATE '45000').
3. Create a trigger auto_generate_billing_record – **AFTER INSERT** on orders – automatically insert a record into payments with amount = total_amount, method = 'Pending', payment_date = NULL.

Level 2:

1. Create a trigger log_deleted_customers – **BEFORE DELETE** on customers – insert the deleted customer details into a customers_audit table (create it first: customer_id, name, email, deleted_at timestamp).\
2. Test all triggers with sample INSERT/DELETE statements

Experiment No. 8:

Stored Procedures – Generate monthly sales report, add new customer, update product prices by category, calculate loyalty points

Level 1:

1. Create a stored procedure GenerateMonthlySalesReport(IN month INT, IN year INT) that outputs total_sales for that month.
2. Create a procedure AddNewCustomer(IN name VARCHAR, IN email VARCHAR, IN phone VARCHAR, IN city VARCHAR, IN membership VARCHAR) that inserts into customers. Also return the new customer_id.
3. Create a procedure UpdateProductPricesByCategory(IN category VARCHAR, IN percentage DECIMAL) that increases price by given percentage for all products in that category.

Level 2:

1. Create a procedure CalculateLoyaltyPoints(IN customer_id INT, OUT points INT) – points = total_amount spent / 100 (integer division).
2. Call each procedure and verify.

Experiment No. 9:

User-Defined Functions – Calculate discount based on membership & amount, compute tax, determine total order value

Level 1:

1. Create a function CalculateDiscount(membership_type VARCHAR, purchase_amount DECIMAL) returns DECIMAL. Logic:
    - 'Gold' → 15% off
    - 'Silver' → 10% off
    - 'Bronze' → 5% off
    - others → 0% off

2. Create a function ComputeTax(product_id INT) returns DECIMAL – tax = price * 0.18 (GST 18%).:

Level 2:

1. Create a function GetTotalOrderValue(order_id INT) returns DECIMAL – sum of (quantity * unit_price) from order_items for that order.
2. Use the functions in a SELECT query: display order_id, total_order_value, tax, discount (assuming customer's membership), and final payable amount.

Experiment No. 10:

Cursors – Row-by-row processing: customer-wise purchase summary, sequential update of seasonal discounts

Level 1:

1. Write a stored procedure using a cursor to generate customer-wise purchase summary – iterate over all

customers, for each customer compute total amount spent, and insert into a new table customer_summary (customer_id, total_spent).

2. Write another procedure using cursor to update seasonal discounts for products sequentially. Assume 'Electronics' gets 5% discount, 'Clothing' gets 10% discount. Use cursor to fetch each product and update a discounted_price column (add if not exists).

Level 2:

1. Demonstrate row-by-row processing with error handling (continue on error).
2. Compare cursor-based approach with a set-based UPDATE in terms of performance (discussion).

Experiment No. 11:

Transaction Control & Exception Handling – START TRANSACTION, COMMIT, ROLLBACK; multi-step payment processing, integrity during complex updates

Level 1:

1. Implement a multi-step payment processing system for an order:
    - Start transaction.
    - Insert payment record into payments (amount, method 'Credit Card').
    - Update order status to 'Paid'.
    - Reduce stock (call trigger or manual update).
    - If any step fails, rollback; else commit.
2. Write a procedure that transfers ₹1000 from order total to refund (simulate). Use DECLARE EXIT HANDLER FOR SQLEXCEPTION to rollback.

Level 2:

1. Simulate a deadlock scenario by running two concurrent transactions (use two MySQL sessions) that update the same row in opposite order. Use SELECT ... FOR UPDATE. Observe and resolve.
2. Demonstrate SAVEPOINT and partial rollback within a transaction.

Experiment No. 12:

MongoDB: Database Design – Create collections (users, products, orders, reviews) with nested documents, flexible schema design

Level 1:

1. Create a database QuickShop.
2. Create collections (without strict schema):
    - users – fields: _id, name, email, address (nested: street, city, zip), membership
    - products – fields: _id, name, category, price, tags (array), stock, supplier_info (nested)
    - orders – fields: _id, user_id, order_date, items (array of objects: product_id, qty, price), total, status
    - reviews – fields: _id, product_id, user_id, rating, comment, created_at

Level 2:

1. Insert at least 3 documents in each collection with nested/array structures.
2. Write queries to display all documents

Experiment No. 13:

MongoDB CRUD Operations – insert, update, delete, query; add products, modify order status, remove inactive users, retrieve cart info

Level 1:

1. Insert a new product with nested supplier info.
2. Update order status from 'Pending' to 'Shipped' for a specific order.
3. Remove (delete) a user who is inactive (no orders in last 6 months) – simulate by checking orders collection.

Level2:

1. Retrieve shopping cart information for a specific user by finding all orders with status 'Cart'.
2. Use find() with projection to show only product name and price.
3. Update multiple documents: Increase price of all products in 'Electronics' category by 5%.

Experiment No. 14:

Advanced MongoDB – Aggregation pipeline for sales analysis, product recommendation queries, index creation for search optimization

Level 1:

1. Write an aggregation pipeline to compute total sales per product category for the last 30 days.

Stages: $match, $unwind (on items), $lookup (to products for category), $group, $sort.

2. Create a product recommendation query – find products frequently bought together. Use aggregation with $group on order items to find pairs.
3. Create an index on products.category and another compound index on orders.user_id and order_date. Explain performance benefits.

Level2:

1. Use explain() to analyze a query before and after indexing.
2. Write a map-reduce function to calculate total orders per user (optional, if time permits).

Experiment No. 15:

Cassandra – Create keyspace, column families for user activity logs, order histories, multi-warehouse inventory; CRUD with CQL, partitioning & replication strategies

Level 1:

1. Install and start Cassandra (or use Docker). Create a keyspace QuickShop with NetworkTopologyStrategy and replication factor 1 for development.
2. Design column families (tables) for:
   - user_activity_log – partition key: user_id, clustering key: timestamp (to store login, page views)
   - order_history – partition key: user_id, clustering key: order_date DESC (to retrieve recent orders)
   - inventory_tracking – partition key: warehouse_id, clustering key: product_id
3. Write CQL to:
   - Insert 5 activity logs for two different users.
   - Query last 3 activities of a specific user.
   - Insert order history records.
   - Update inventory quantity in a warehouse.

Level 2:

1. Demonstrate data partitioning – explain how partition key affects data distribution.
2. Explain replication strategy and consistency levels (ONE, QUORUM, ALL) with examples.

References/Manual/Software:

1.Abraham Silberschatz, Henry F.Korth,"Database Ssystem Concepts", McGraw HIll,7th Edition, 2020.

2.Elvis C. Foster, Shripad V. Godbole, "Database Systems", Apress,2014 .

3. Daniel Nichter, Efficient MY SQL Performance Best Practices and Techniques, O Reily Media First Edition, Jan 2022

4. Michael E Kirshteyn,Mastering NoSQL Database Design: A Comprehensive Guide to Building Scalable, High-Performance, and Flexible Data Systems,April 2024.

5.Vinicius M. Grippa,Learning MySQL: Get a Handle on Your Data 2nd Edition,O'Reilly Media,October 2021.

Course Code: CSA4001	**Course Title: Design Thinking and Innovation** **Type of Course: Foundation Course**	**L-T- P- C**	0	0	2	0
Version No.	1.0					
Course Pre-requisites	**NIL**					
Anti-requisites	**NIL**					

Course Description	This course introduces the fundamental principles and processes of Design Thinking and Innovation. It focuses on developing empathy, creativity, collaboration, and human-cantered problem-solving skills. Students learn to apply Design Thinking methodologies to real-world challenges through research, ideation, prototyping, testing, and iterative refinement. The course emphasizes innovation across diverse domains and equips learners with essential skills for engineering practice, entrepreneurship, and user-centric solution development.
Course Objective	The objective of the course is EMPLOYABILITY of student by using PARTICIPATIVE LEARNING techniques.
Course Outcomes	On successful completion of this course, the students shall be able to: **CO1:** Demonstrate an understanding of the Design Thinking mindset, user-centered design principles, and their application in solving real-world problems. [**Understand]** **CO2:** Apply user research techniques such as interviews, observation, contextual inquiry, empathy mapping, and persona development to identify user needs and insights**. [Apply]** **CO3:** Analyze user experiences through journey mapping, pain-point identification, and Point-of-View (POV) statements to define meaningful problem statements**. [Analyze]** **CO4:** Generate innovative solutions using ideation techniques such as How Might We (HMW) questions, brainstorming, Crazy Eights, SCAMPER, and mind mapping. **[Apply]** **CO5:** Develop and evaluate prototypes using wireframes, storyboards, and low-fidelity models, incorporating user feedback through iterative testing and refinement. **[Apply]** **CO6:** Design and present feasible, impactful, and sustainable solutions by integrating emerging technologies, innovation strategies, and implementation planning. **[Apply]**

Experiment No. 1:
Understanding Design Thinking Mindset
Level 1:
Identify characteristics of Design Thinking through case examples.
Level 2:
Analyze a real-world problem and explain how Design Thinking can address it.

Experiment No. 2:
Exploring User-Centered Design
Level 1:
Study user-centered design principles using a selected product.
Level 2:
Evaluate the product and suggest user-centric improvements.

Experiment No. 3:
Conducting User Interviews
Level 1:
Prepare and conduct structured interviews with 3 users.
Level 2:
Conduct semi-structured interviews and derive meaningful insights.

Experiment No. 4:
Observation and Contextual Inquiry
Level 1:
Observe user behavior in a chosen environment and record findings
Level 2:
Perform contextual inquiry and identify hidden user needs.

Experiment No. 5:
Creating Empathy Maps
Level 1:
Develop an Empathy Map based on interview data.
Level 2:

Compare multiple Empathy Maps and identify common patterns.
Experiment No. 6: Persona Development **Level 1:** Create a user persona from collected research data. **Level 2:** Design multiple personas representing diverse user groups.
Experiment No. 7: User Journey Mapping **Level 1:** Develop a basic User Journey Map for a selected service. **Level 2:** Identify pain points and improvement opportunities in the journey.
Experiment No. 8: Problem Definition and POV Statements **Level 1:** Formulate Point-of-View (POV) statements from user insights. **Level 2:** Refine POV statements and validate them with users.
Experiment No. 9: Crafting 'How Might We' Questions **Level 1:** Generate HMW questions from identified user problems. **Level 2:** Prioritize HMW questions based on innovation potential.
Experiment No. 10: Brainstorming and Idea Generation **Level 1:** Conduct a brainstorming session and generate 20 ideas. **Level 2:** Apply Crazy Eights and SCAMPER techniques to generate innovative solutions.
Experiment No. 11: Mind Mapping and Concept Development **Level 1:** Create a mind map around a chosen problem statement. **Level 2:** Combine multiple mind maps to generate integrated solutions.
Experiment No. 12: Idea Selection and Prioritization **Level 1:** Use a prioritization matrix to evaluate generated ideas. **Level 2:** Select the best solution considering feasibility and impact.
Experiment No. 13: Low-Fidelity Prototyping **Level 1:** Develop paper sketches or storyboard prototypes. **Level 2:** Create detailed wireframes incorporating user requirements.

Experiment No. 14: Prototype Testing and Feedback Collection **Level 1:** Conduct usability testing with at least 3 users. **Level 2:** Analyze feedback and redesign the prototype iteratively.
Experiment No. 15: Innovation Implementation and Future Trends **Level 1:** Prepare a plan for implementing the proposed solution. **Level 2:** Present the solution incorporating AI, sustainability, or emerging technology concepts.
References: 1. Universal Methods of Design, Rockport Publishers, 2019. 2. The Design Thinking Playbook, **Michael Lewrick, Patrick Link, and Larry Leifer**, Wiley, 2018 **E-Resources:** • IDEO Design Thinking Resources • IDEO.org Design Kit (Human-Centered Design Toolkit)

SEMESTER II

Course Code: **CSA4501**	**Course Title: Cloud Computing** **Type of Course: Program Core**	**L-T- P- C**	2	0	0	2
Version No.	1.0					
Course Pre-requisites	**Computer Networks**					
Anti-requisites	**NIL**					
Course Description	This course provides a hands-on comprehensive study of Cloud concepts and capabilities across the various Cloud service models including Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). It dives into all of the details that a student needs to know in order to plan for developing applications on the cloud and what to look for when using applications or services hosted on a cloud.					
Course Objective	The objective of the course is to familiarize the learners with the concepts of Cloud Computing and attain Skill Development through Experiential Learning techniques.					
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1**: Apply fundamental concepts of cloud computing and service models (IaaS, PaaS, SaaS) in real-world scenarios [**APPLY]** **CO2:** Implement virtualization techniques and analyze their role in efficient resource utilization [**APPLY]** **CO3**: Analyze cloud infrastructure, QoS parameters, and Service Level Agreements (SLAs) for performance optimization [**APPLY]** **CO4:** Evaluate cloud security mechanisms and management strategies for secure cloud environments. [**APPLY]** **CO5:** Develop cloud-based applications using modern programming models and containerization technologies [**APPLY]**					

	CO6: Design and deploy scalable cloud solutions using platforms such as AWS or similar cloud environments **[ANALYZE]**	
Course Content:		
Module 1	**Introduction to Cloud services - CO1**	**7 Sessions**
Evolution of cloud computing, Computing Platforms and Technologies, Cloud Computing Architecture, IaaS, PaaS, SaaS, Types of Clouds, Cloud Computing Environments.		
Module 2	**Virtualization Techniques – CO2**	**7 Sessions**
Basics of Virtualization - Types of Virtualizations, Taxonomy of Virtualization Techniques, Implementation Levels of Virtualization.		
Module 3	**Cloud QoS and Management – CO3**	**8 Sessions**
Cloud Infrastructure Mechanisms, SLAs, Specialized Cloud Mechanisms, Cloud Management Mechanisms, Cloud Security Mechanisms		
Module 4	**Application development in Cloud – CO4**	**8 Sessions**
Programming Models for Cloud Computing - Software Development in Cloud - Service creation environments to develop cloud-based applications. Development environments for service development (Demonstration using AWS Cloud); Dockers and Containers.		

Project work/Assignment:

1. **Automation of performance analysis of students through the Cloud**
2. **Chatbots development using Cloud resources**
3. **Blog creation using Cloud computing**
4. **Implement a simple cloud-based application using AWS (EC2 / S3)**
5. **Demonstrate containerization using Docker and deploy a sample application**
6. **Perform cloud security analysis by identifying threats and suggesting mitigation techniques**

Topics related to

1. Problem Solving: Apply cloud computing techniques to solve real-world challenges such as scalable storage, distributed computing, virtualization, and secure data management in dynamic environments.

2. Employability: Hands-on experience with cloud platforms (AWS/Azure/GCP), virtualization tools, Docker containers, cloud deployment, and security practices to prepare for roles such as Cloud Engineer, DevOps Engineer, and System Architect.

Textbook(s):

T1. Daniel Vaughan, "Cloud Native Development with Google Cloud". O'Reilly Media Publishers. 1st Edition 2023.

T2. Rajkumar Buyya, Christian Vecchiola, and S. Thamarai Selvi, *Mastering Cloud Computing*, 2nd Edition, McGraw Hill, 2021

T3. Thomas Erl, Ricardo Puttini, and Zaigham Mahmood, *Cloud Computing: Concepts, Technology & Architecture*, 2nd Edition, Pearson, 2021

References

R1. Michael J. Kavis, *Architecting the Cloud: Design Decisions for Cloud Computing Service Models*, Wiley, 2021

R2. Josh Stella, *Cloud Native Security*, O'Reilly Media, 2023

R3. Justin Garrison and Kris Nova, *Cloud Native Infrastructure*, O'Reilly Media, 2021.

E-Resources:

1. https://aws.amazon.com/training/
2. https://learn.microsoft.com/en-us/training/azure/
3. https://cloud.google.com/training
4. https://www.docker.com/resources/what-container

Course Code: CSA4502	**Course Title:** Machine Learning **Type of Course:** Program Core	**L-T- P- C**	2	0	0	2
Version No.	1.0					
Course Pre-requisites	**NIL**					
Anti-requisites	**NIL**					
Course Description	This course builds the foundational insight of understanding principles of machine learning process and to explore various ML algorithms and practices to accelerate strategic decision making with data. Overall, this course gives a comprehensive insight to appropriately choose ML algorithms on real time problems to construct an intelligent ML model.					
Course Objective	The objective of the course is to familiarize the learners to explore the competence and comprehend with potential machine learning algorithms and techniques to revolutionize with real-world problems and create prominent solutions (with ML models) to attain Employability Skills through Experiential Learning techniques.					
Course Outcomes	**On successful completion of the course the students shall be able to:** **CO1:** Apply fundamental concepts of Machine Learning to analyze real-world datasets and select appropriate models [**APPLY]** **CO2:** Implement supervised learning algorithms such as Decision Trees, Naïve Bayes, and SVM for classification problems [**APPLY]** **CO3**: Develop and evaluate neural network models and ensemble techniques for improved prediction accuracy [**APPLY]** **CO4:** Apply unsupervised learning techniques like clustering and dimensionality reduction for data analysis[**APPLY]** **CO5:** Analyze reinforcement learning models and optimize decision-making processes in dynamic environments [**APPLY]** **CO6:** Design and build end-to-end machine learning solutions for real-world applications with performance evaluation[**APPLY]**					
Course Content:						
Module 1	**Introduction to Machine Learning - CO1**					**7 Sessions**
Introduction to Machine Learning: Types of Machine Learning, Supervised Learning: The Machine Learning Process, Performance measures, The Bias-Variance Tradeoff, Learning with Trees: Using Decision Trees, Constructing Decision Trees, Classification and Regression Trees (CART), Turning Data into Probabilities: The Naïve Bayes' Classifier, Bayesian Networks.						
Module 2	**Supervised Learning – CO2**					**7 Sessions**
Neural Networks: The Brain and The Neuron, Neural Networks, The Perceptron, Linear Separability, The Multi-Layer Perceptron: Going Forwards, Going Backwards Back-Propagation of Error, The Multi-Layer Perceptron in Practice, Deriving Back-Propagation. Support Vector Machines: Optimal Separation, Kernels, The Support Vector Machine Algorithm, Dimensionality Reduction: The Curse of Dimensionality, Linear Discriminant Analysis (LDA), Principal Components Analysis (PCA), Evolutionary Learning: The Genetic Algorithm (GA), Generating Offspring: Genetic Operators, Using Genetic Algorithms. Ensemble Learning: Boosting, Bagging and Random Forests.						
Module 3	**Unsupervised Learning – CO3**					**8 Sessions**
Unsupervised Learning: The k-means algorithm, Hierarchical Clustering, The Self-Organising Feature Map, Explanation based Learning, Markov Decision Process- Reinforcement Learning and Evaluating Hypotheses: Introduction, Learning Task, Q Learning, Non-Deterministic Rewards and Actions.						
Module 4	**Symmetric weights and Deep Belief Networks – CO4**					**8 Sessions**
The EM Algorithm: Estimate Means of K Gaussians, General Statement of EM Algorithm, Extensions to the SVM, Active Reinforcement Learning, Energetic Learning: The Hopfield Network, Stochastic Neurons: The Boltzmann Machine, Deep Belief Networks (DBF)						

Project work/Assignment:

1. **Implement Decision Tree and Naïve Bayes classifiers on a real dataset and compare performance**
2. **Build a Neural Network model using Python (TensorFlow/Keras) for classification**
3. **Apply PCA and LDA on a dataset and visualize dimensionality reduction**
4. **Perform clustering using K-Means and Hierarchical Clustering and analyze results**
5. **Develop a machine learning model for stress detection or prediction using EEG or similar datasets**
6. **Implement an ensemble model (Random Forest / Boosting) and compare with individual models**

Topics related to

1. Problem Solving: Apply machine learning techniques such as classification, clustering, and optimization to solve real-world problems like prediction, pattern recognition, and decision-making using structured and unstructured data.

2. Employability: Hands-on experience with Python libraries (NumPy, Pandas, Scikit-learn, TensorFlow), data preprocessing, model building, evaluation, and deployment to enhance skills required for roles such as Data Analyst, Machine Learning Engineer, and AI Developer.

Textbook(s):

T1. Aurélien Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, 3rd Edition, O'Reilly Media, 2022

T2. Ethem Alpaydin, *Introduction to Machine Learning*, 4th Edition, MIT Press, 2020

T3. Introduction to Machine Learning, Ethem Alpaydin, 4th Edition, The MIT press, ISBN:978-0-262-043-793, March 2020.

References

R1. Hands-On Machine learning with Scikit-Learn, Keras, and Tensorflow: Concepts, Tools, and Techniques to Build Intelligent Systems, Aurelien Geron, 3rd Edition, O'Reilly Media, ISBN: 978-9355421982, October 2022.

R2. Kevin P. Murphy, *Probabilistic Machine Learning: An Introduction*, MIT Press, 2022

R3. Christopher M. Bishop and Hugh Bishop, *Deep Learning: Foundations and Concepts*, Springer, 2023

R4. Andreas C. Müller and Sarah Guido, *Introduction to Machine Learning with Python*, O'Reilly Media, 2021.

E-Resources:

1. https://www.coursera.org/learn/machine-learning
2. https://developers.google.com/machine-learning
3. https://scikit-learn.org/stable/
4. https://www.kaggle.com/learn

Course Code: CSA4204	**Course Title:** Object Oriented Programming using Java **Type of Course:** Program Core	**L-T- P- C**	2	0	0	2
Version No.	1.0					
Course Pre-requisites	**NIL**					
Anti-requisites	**NIL**					
Course Description	The main objective is to learn the basic concept and techniques, which form the object-oriented programming paradigm. Object-oriented programming is a new way of thinking about problem using models organized around real-world concept. It investigates the software engineering principles of encapsulation, information hiding and code reuse, and discusses how these concepts are used to build abstract data types. The object-oriented programming features of Sessions, inheritance, polymorphism and composition are studied, along with constructors and method overloading. Students implement Java programs incorporating features from the Java programming language.					
Course Objective	The objective of the course is to familiarize learners with Object-Oriented Programming concepts using Java, while developing general-purpose applications with database connectivity through experiential learning and object-oriented design principles.					

<table>
<tr><td>Course Outcomes</td><td colspan="2">On successful completion of this course, the students shall be able to:
CO1: Apply Object-Oriented Programming principles in Java to design module and reusable code that efficiently solve the real-world problems. (Apply)
CO2: Utilize the concepts of Inheritance, Multithreading, and Exception Handling to develop efficient and robust code. [Apply]
CO3: Develop Serverside java applications using Servlet and JSP concepts. (Apply)
CO4: Construct basic applications that demonstrate efficient interaction with relational database using JDBC through the java.sql package. (Apply)
CO5: Analyze the design and implementation of integrated Java-based applications by combining Core Java, JDBC, and Web technologies using object-oriented design principles for real-world problem solving. (Analyze)
CO6: Apply professional practices such as teamwork, effective use of development tools, debugging strategies, and self-directed learning in Java application development. (Apply)</td></tr>
<tr><td>Course Content:</td><td colspan="2"></td></tr>
<tr><td>Module 1</td><td>Introduction to OOPs - CO1</td><td>7 Sessions</td></tr>
<tr><td colspan="3">Java buzzwords, characteristics, Java environment; Java program structure and source file structure; Java compilation process (javac, bytecode, JVM); Data types, variables, arrays; Operators; Control statements; Classes and objects; Methods and method overloading; Access control; Static and final keywords; Constructors; Recursion; Nested and inner classes; String, StringBuffer, StringBuilder</td></tr>
<tr><td>Module 2</td><td>Inheritance, Exception Handling and Multithreading – CO2</td><td>7 Sessions</td></tr>
<tr><td colspan="3">Inheritance concepts; Super keyword; Method overriding; Dynamic method dispatch; Abstract classes; Final keyword with inheritance; Constructor chaining; Object class methods: toString(), equals(), hashCode();
Packages and interfaces; Exception hierarchy; try, catch, multiple catch, nested try; throw, throws, finally; Built-in and user-defined exceptions;
Multithreading concepts; Java thread model; Main thread; Thread creation; Thread methods; Thread synchronization.</td></tr>
<tr><td>Module 3</td><td>Collections. Servlets and Java Server Pages – CO3, CO4</td><td>8 Sessions</td></tr>
<tr><td colspan="3">Java Collections Framework; ArrayList, LinkedList, HashSet, TreeSet, PriorityQueue; Iterator interface; HashMap, TreeMap; Comparator interface; Lambda expressions; Servlet architecture; Servlet API; Servlet life cycle; Session tracking; Cookies; JSP architecture; JSP life cycle; Basic JSP tags; JSP action tags; JSP implicit objects.</td></tr>
<tr><td>Module 4</td><td>IO Operations, JDBC, Hibernate – CO5, CO6</td><td>8 Sessions</td></tr>
<tr><td colspan="3">Java I/O streams; java.io package; Byte streams; Character streams; Java NIO introduction; File class; FileReader, FileWriter; BufferedReader, BufferedWriter;
JDBC architecture; Types of JDBC drivers; JDBC API; Connection, Statement, PreparedStatement, ResultSet; java.sql package; CRUD operations using JDBC.</td></tr>
<tr><td colspan="3">Project work/Assignment:
• Develop a Java program demonstrating OOP concepts such as classes, objects, constructors, and method overloading
• Implement inheritance and polymorphism using a real-world example (e.g., employee management system)
• Design and implement exception handling for a banking or student management system
• Create a multithreaded Java application demonstrating thread creation and synchronization
• Develop a Java application using Collections Framework (ArrayList, HashMap) for data management
• Build a database-driven application using JDBC to perform CRUD operations</td></tr>
<tr><td colspan="3">Topics related to
1. Problem Solving: Apply object-oriented programming principles to design modular, reusable, and efficient solutions for real-world problems involving data handling, exception management, multithreading, and database interaction.
2. Employability: Hands-on experience in Java programming, OOP design, multithreading, web development (Servlets & JSP), and database connectivity (JDBC, Hibernate) to prepare for roles such as Java Developer, Backend Developer, and Software Engineer.</td></tr>
<tr><td colspan="3">Textbook(s):</td></tr>
</table>

T1. Herbert Schildt, *Java: The Complete Reference*, 12th Edition, McGraw Hill, 2021
T2. Cay S. Horstmann, *Core Java Volume I – Fundamentals*, 12th Edition, Pearson, 2022
T3. Paul Deitel and Harvey Deitel, *Java: How to Program*, 12th Edition, Pearson, 2021

References

R1. Joshua Bloch, *Effective Java*, 3rd Edition (Updated Edition), Addison-Wesley, 2021
R2. Raoul-Gabriel Urma, Mario Fusco, and Alan Mycroft, *Modern Java in Action*, Manning Publications, 2021
R3. Jeanne Boyarsky and Scott Selikoff, *OCP Java SE 17 Developer Study Guide*, Wiley, 2022
R4. Venkat Subramaniam, *Programming Java: Functional Programming and Advanced Techniques*, Pragmatic Bookshelf, 2021

E-Resources:

1. https://docs.oracle.com/en/java/
2. https://www.geeksforgeeks.org/java/
3. https://www.javatpoint.com/java-tutorial
4. https://www.w3schools.com/java/

Course Code: PPS4008	**Course Title:** Quantitative Skills and Logical Reasoning **Type of Course:** Program Core	**L-T- P- C**	1	0	2	2
Version No.	1.0					
Course Pre-requisites	Students should know the basic Mathematics & aptitude along with understanding of English					
Anti-requisites	**NIL**					
Course Description	The objective of this course is to prepare the trainees to tackle the questions on various topics and various difficulty levels based on Quantitative Ability, and Logical Reasoning asked during the placement drives. There will be sufficient focus on building the fundamentals of all the topics, as well as on solving the higher order thinking questions. The focus of this course is to teach the students to not only get to the correct answers, but to get there faster than ever before, which will improve their employability factor.					
Course Objective	The objective of the course is to familiarize the learners with the concepts of Aptitude and attain Skill Development through Problem Solving techniques.					
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1**: Apply fundamental quantitative aptitude concepts such as percentages, ratios, and averages to solve real-world problems **[APPLY]** **CO2:** Solve problems involving time, speed, distance, and work using appropriate mathematical techniques **[APPLY]** **CO3**: Apply permutation, combination, and probability concepts in analytical problem-solving scenarios**[APPLY]** **CO4:** Analyze and interpret data using logical reasoning and data interpretation techniques **[ANALYZE]** **CO5:** Apply complex logical reasoning problems such as puzzles, coding-decoding, and syllogisms**[APPLY]** **CO6:** Analyze efficient problem-solving strategies to improve speed and accuracy in competitive and placement exams **[Analyze]**					
Course Content:						
Module 1	**Quantitative Ability - CO1, CO2, CO3**				**22 Sessions**	
Introduction to Aptitude, Percentages, Profit & Loss, Ratio & Proportion, Averages, Time & Work, Time speed & Distance, Permutation and Combination, Probability, Data Interpretation						
Module 2	**Logical Reasoning – CO4, CO5, CO6**				**23 Sessions**	

Linear & Circular Arrangement Puzzle, Coding & Decoding, Blood Relations, Directions, Number Series, Wrong number series, Visual Reasoning, Critical thinking, Syllogism
Project work/Assignment: **1. Solve real-time quantitative aptitude questions based on percentages, profit & loss, and ratios** **2. Analyze and solve time & work and time, speed & distance problems using shortcut techniques** **3. Solve permutation, combination, and probability problems with practical applications** **4. Perform data interpretation using charts, graphs, and tables** **5. Solve logical reasoning puzzles (seating arrangement, blood relations, coding-decoding)** **6. Practice mixed aptitude tests under time constraints to improve speed and accuracy**
Topics related to **1. Problem Solving:** Apply mathematical and logical reasoning techniques to solve real-world problems involving data interpretation, decision-making, pattern recognition, and analytical thinking. **2. Employability:** Develop aptitude, logical reasoning, and analytical skills required for competitive exams, campus placements, and job roles requiring critical thinking, problem-solving, and decision-making abilities.
Textbook(s): **T1**. R.S. Aggarwal, *Quantitative Aptitude for Competitive Examinations*, Revised Edition, S. Chand Publishing, 2021. **T2**. Arun Sharma, *How to Prepare for Quantitative Aptitude for CAT*, McGraw Hill, 2022. **T3**. Abhijit Guha, *Quantitative Aptitude for Competitive Examinations*, 6th Edition, McGraw Hill, 2023
References **R1**. Nishit K. Sinha, *Quantitative Aptitude for Competitive Examinations*, Pearson, 2021 **R2**. A.K. Gupta, *Logical and Analytical Reasoning*, Ramesh Publishing House, 2021 **R3**. BS Sijwali and Indu Sijwali, *A New Approach to Reasoning*, Arihant Publications, 2020 **E-Resources:** 1. https://www.indiabix.com/aptitude/questions-and-answers/ 2. https://www.geeksforgeeks.org/aptitude-gq/ 3. https://www.practiceaptitudetests.com/ 4. https://www.khanacademy.org/math

Course Code: **CSA4306**	**Course Title:** **UI/UX Design** **Type of Course:** **Lab Integrated course**	**L-T- P- C**	1	0	4	3
Version No.	1.0					
Course Pre-requisites	**NIL**					
Anti-requisites	**NIL**					
Course Description	The UI/UX Design brings a design-centric approach to user interface and user experience design, and offers practical, skill-based instruction centered on a visual communications perspective, rather than on one focused on marketing or programming alone. User interface and user experience design is a high-demand field, but the skills and knowledge you will learn in this Specialization are applicable to a wide variety of careers, from marketing to web design to human-computer interaction. The course is foundational and hands-on learning in using popular design tools such as Figma.					
Course Objective	The objective of the course is to familiarize the learners with the concepts of UI/UX Design and attain Employability Skills through Experiential Learning techniques.					

Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1:** Apply UI/UX design principles and interaction design concepts to create user-friendly interfaces [**APPLY]** **CO2:** Implement user-cantered design methodologies to analyze user needs and design appropriate solutions [**APPLY]** **CO3:** Develop wireframes and prototypes using design tools such as Figma [**APPLY]** **CO4:** Evaluate usability and accessibility of interfaces using testing techniques and design principles [**APPLY]** **CO5:** Design responsive and inclusive user interfaces for multiple platforms and diverse user groups [**APPLY]** **CO6:** Create complete UI/UX solutions from user research to prototyping for real-world applications [**APPLY]**	
Course Content:		
Module 1	**Introduction to UI/UX – CO1**	**19 Sessions (4T + 15P)**
Introduction to User Experience, Importance of UX-design, Different sub- disciplines within UX, job opportunities in UX field/domain. RoI, KPI, Stakeholders of UX team, trade-offs, UX Design definition. Basics of Interaction Design, User Research, Visual Design, Motion Design.		
Module 2	**User and User Centered Design – CO2**	**19 Sessions (4T + 15P)**
Users and end users, User Centered design framework, 7 principles of UX design, 4 stages of user centered design, 5-elements framework. Design thinking process, Lean UX, Double Diamond, designing for the next billion users, designing for multiple platforms, the four Cs of designing for multiple platform		
Module 3	**Design Methodologies – CO3, CO4**	**19 Sessions (4T + 15P)**
Universal design, 7 principles of universal design, inclusive design and accessible design, and equity-focused design. Equality and equity. Designing for accessibility, Lenses of Accessibility, assistive technology, design sprints. Wireframing, importance of wireframing. Compatibility with wearable devices.		
Module 4	**Personas, developing mockups using Figma – CO5, CO6**	**18 Sessions (3T + 15P)**
Basics of personas, creating personas, perspectives on personas. Gestalt principles of perception, Usability Testing, acceptance testing, creating mockups and prototypes in Figma.		

List of Laboratory Tasks:

Experiment No. 1: Installation and Interface of Balsamiq and/or Figma
Level 1: Ensure that both Balsamiq and Figma are up and running with user accounts.
Level 2: Download and import design files from internet to familiarize with them.

Experiment No. 2: Create wireframe of the login screen of a mobile app
Level 1: Make first wireframe of one login page
Level 2: Make two pages that are hyperlinked and critique the design

Experiment No. 3: Final wireframe experiment.
Level 1: Prepare the wireframe of all the pages of a selected website
Level 2: Change the wireframe to make the design changes to the website

Experiment No. 4: First Figma experiment.
Level 1: Figma interface, shortcuts and tools.
Level 2: Create and move between frames.

Experiment No. 5: Design App Screen
Level 1: Create layout, layers, fill colours
Level 2: Set layer opacity, lock and unlock layers

Experiment No. 6: Logo and icon
Level 1: Boolean operations on shapes, pen tool
Level 2: Make smiley face

Experiment No. 7: Create an app face
Level 1: Insert image, design nav bar using logo and icons
Level 2: Duplicate Frames

Experiment No. 8: Create a prototype
Level 1: Use designing and prototyping modes
Level 2: Create connections between frames and layers

Experiment No. 9: Create prototype of food delivery app
Level 1: Replicate inner pages of app
Level 2: Improve the inner page design

Experiment No. 10: Create prototype of a desktop website
Level 1: Replicate pages on desktop app
Level 2: Export files and share in LinkedIn

Targeted Application & Tools that can be used:
Figma

Project work/Assignment:

1. **Conduct user research and create user personas for a selected application**
2. **Design wireframes for a mobile or web application using low-fidelity sketches**
3. **Develop high-fidelity UI designs using Figma for a real-world problem**
4. **Perform usability testing and document findings with improvements**
5. **Design responsive UI screens for multiple devices (mobile, tablet, desktop)**
6. **Apply accessibility principles to redesign an existing application**

Topics related to
1. Problem Solving: Apply design thinking and user-centered design approaches to identify user problems, analyze requirements, and create intuitive, efficient, and accessible interface solutions.
2. Employability: Hands-on experience with tools like Figma, wireframing, prototyping, usability testing, and accessibility design to prepare for roles such as UI/UX Designer, Product Designer, and Interaction Designer.

Textbook(s):
T1. Ben Shneiderman et al., *Designing the User Interface: Strategies for Effective Human-Computer Interaction*, 6th Edition, Pearson, 2021
T2. Don Norman, *The Design of Everyday Things (Revised and Expanded Edition)*, Basic Books, 2021
T3. Russ Unger and Carolyn Chandler, *A Project Guide to UX Design*, 3rd Edition, New Riders, 2021

References
R1. Jon Yablonski, *Laws of UX: Using Psychology to Design Better Products*, O'Reilly Media, 2020
R2. Steve Krug, *Don't Make Me Think (Revisited)*, 3rd Edition Updated, 2021
R3. Laura Klein, *Build Better Products*, Rosenfeld Media, 2021
R4. David Travis and Philip Hodgson, *Think Like a UX Researcher*, CRC Press, 2022

Course Code: **CSA4601**	**Course Title:** **Cloud Computing Lab** **Type of Course:** **Lab**	**L-T- P- C**	0	0	2	1
Version No.	1.0					
Course Pre-requisites	Computer Networks					
Anti-requisites	**NIL**					

Course Description	This course provides a hands-on comprehensive study of Cloud concepts and capabilities across the various Cloud service models including Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). It dives into all of the details that a student needs to know in order to plan for developing applications on the cloud and what to look for when using applications or services hosted on a cloud.
Course Objective	The objective of the course is to familiarize the learners with the concepts of Cloud Computing and attain Skill Development through Experiential Learning techniques.
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1**: **CO1:** Apply cloud computing concepts by creating and managing virtual machines in a cloud environment [**APPLY]** **CO2:** Implement storage and compute services using cloud platforms such as AWS/Azure/GCP [**APPLY]** **CO3**: Deploy and manage applications on cloud infrastructure using appropriate service models [**APPLY]** **CO4:** Analyze cloud performance using monitoring tools and evaluate Service Level Agreements (SLAs) **CO5:** Implement containerization using Docker and manage applications using containers [**APPLY]** **CO6**: Design and deploy scalable and secure cloud-based solutions for real-world applications [**APPLY]**

List of Experiments / Exercises

Experiment No. 1:
Level 1: Create a basic form in Zoho Creator to collect student details (Name, Roll No, Department, Email).
Level 2: Add field validations and make the email field mandatory. Test the form for different inputs.

Experiment No. 2:
Level 1: Create a new application in Zoho Creator to manage a library system (Books, Authors, Issued Books).
Level 2: Establish relationships between forms (Books and Authors) using lookup fields.

Experiment No. 3:
Level 1: Create a leave management system with a form to apply for leave.
Level 2: Implement an approval workflow with notifications for pending and approved leaves.

Experiment No. 4:
Level 1: Create an online appointment scheduling system in Zoho Creator. Design forms to capture customer details, appointment date/time, and service required. Build a calendar view to display booked appointments.
Level 2: Deploy the application on AWS and provide users with access through a custom domain. Set up Elastic Load Balancing (ELB) to ensure that the application can scale efficiently for multiple users.

Experiment No. 5:
Level 1: Create a cloud inventory management application using Zoho Creator with forms for product name, quantity, and price. Set up reports to display the product list and low stock alerts.
Level 2: Design a cloud-based survey application in Zoho Creator to capture user responses with questions like Name, Age, and Feedback. Use Deluge scripting to automatically calculate the response count for each question.

Experiment No. 6:
Level 1: Install Oracle VirtualBox/VMware Workstation. Create a virtual machine and install Ubuntu/Linux OS.
Level 2: Configure virtual hardware (RAM, CPU, storage) and observe performance changes after modification.

Experiment No. 7:
Level 1: Set up and run a Windows virtual machine in VirtualBox. Enable Guest Additions and test features like shared clipboard and drag-and-drop.
Level 2: Share folders between host and guest operating systems and test access from both sides.

Experiment No. 8:
Level 1: Install and configure Hyper-V on a Windows system. Create and run a virtual machine.
Level 2: Enable and test nested virtualization on Hyper-V.

Experiment No. 9:
Level 1: Evaluate and compare different cloud storage systems to understand their features, performance, and pricing.
Level 2: Assess cloud Storage systems and Cloud security, the risks involved, its impact and develop cloud application.

Experiment No. 10:
Level 1: Using the Cloud Analyst Tool, simulate traffic distribution across multiple regions (North America and Europe) for an e-commerce platform. Write a program to configure load balancing across these regions, ensuring the traffic is intelligently routed to the closest region based on latency and server load. Display the simulated load distribution and response times for each region.
Level 2: Your organization is planning to migrate services to the cloud. Using Cloud Analyst, describe the step-by-step process to simulate the deployment of two data centers and three user bases. Based on the simulation results, how would you estimate the operational cost, and what configuration changes could you make to reduce the overall cost?

Experiment No. 11:
Level 1: Write a Reducer to aggregate and analyze the extracted data (e.g., count occurrences).
Level 2: Write a Mapper to extract key information (e.g., IP addresses) from log entries.

Experiment No. 12:
Level 1: Install Google App Engine. Create hello world app and other simple web applications using python/java.
Level 2: Find a procedure to launch virtual machine using trystack (Online Openstack Demo Version)

Experiment No. 13:
Level 1: Set up a basic Hadoop cluster in a local or virtualized environment. Configure HDFS (Hadoop Distributed File System) to store data across multiple nodes. Verify the setup by running basic Hadoop commands such as hdfs dfs -ls and hdfs dfs -put.
Level 2: Extend the setup to include YARN (Yet Another Resource Negotiator) for resource management. Implement Hadoop MapReduce to process data stored in HDFS. Create a simple program to process a text file, count the occurrences of words, and store the results back in HDFS.

Experiment No. 14:
Level 1: Write a basic MapReduce program in Java or Python that processes a dataset (such as a text file) and performs a word count. Execute the program on a Hadoop cluster and store the output in HDFS.
Level 2: Modify the MapReduce program to process a larger, more complex dataset. For example, implement a program that computes the average temperature from a weather dataset, storing the result in HDFS. Test the program's scalability by running it on larger data sets.

References/Manual/Software:
Targeted Applications:

Developing applications on Cloud Platforms via Virtual machines

Cloud Tools:

- CloudSim
- VMWare
- Amazon EC2
- Google Compute Engine
- Microsoft Azure

1. Daniel Vaughan, "Cloud Native Development with Google Cloud". O'Reilly Media Publishers. 1st Edition 2023.
2. Rajkumar Buyya, Christian Vecchiola, and Thamarai Selvi, "*Mastering Cloud Computing*", McGraw Hill Education, 2017edition.
3. John Rittinghouse and James Ransome, "*Cloud Computing, Implementation, Management and Security*", CRC

Press, 2010 edition.

Course Code: **CSA4304**	**Course Title:** **Object Oriented Programming using Java Lab** **Type of Course:** **Program Core**	**L-T- P- C**	0	0	4	2
Version No.	1.0					
Course Pre-requisites	**Basic programming Skills**					
Anti-requisites	**NIL**					
Course Description	The main objective is to learn the basic concept and techniques which form the object-oriented programming paradigm. Object-oriented programming is a new way of thinking about problem using models organized around real-world concept. It investigates the software engineering principles of encapsulation, information hiding and code reuse, and discusses how these concepts are used to build abstract data types. The object-oriented programming features of Sessions, inheritance, polymorphism and composition are studied, along with constructors and method overloading. Students implement Java programs incorporating features from the Java programming language.					
Course Objective	The objective of the course is to familiarize learners with Object-Oriented Programming concepts using Java, while developing general-purpose applications with database connectivity through experiential learning and object-oriented design principles.					
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1**: Apply Object-Oriented Programming principles in Java to design module and reusable code that efficiently solve the real-world problems. **(Apply)** **CO2:** Utilize the concepts of Inheritance, Multithreading, and Exception Handling to develop efficient and robust code. (Apply) **CO3**: Develop Serverside java applications using Servlet and JSP concepts. **(Apply)** **CO4:** Construct basic applications that demonstrate efficient interaction with relational database systems through JDBC and Hibernate frameworks. **(Apply)** **CO5:** Analyze the design and implementation of integrated Java-based applications by combining Core Java, JDBC, and Web technologies using object-oriented design principles for real-world problem solving. **(Analyze)** **CO6:** Analyze professional practices such as teamwork, effective use of development tools, debugging strategies, and self-directed learning in Java application development**. (Analyze)**					

List of Experiments / Exercises

Experiment No. 1:
Level 1: Write a program to display your name and course name. Read the input using Command Line arguments
Level 2: Write a program to convert Celsius to Fahrenheit. Read the input using Scanner

Experiment No. 2:
Level 1: A teacher in a school wants to calculate the average marks of a student in three subjects to assess their overall performance. The teacher needs a program that accepts the marks of three subjects and calculates the average. Read the data using the Scanner Class.
Level 2: Presidency University collects student information during enrolment for the even semester. The program uses the Scanner class to take input (such as name, course, gender, age, fees and fees paid status) from the student and displays the information entered. Help the Presidency University by writing an application in Java to perform the same. (Hint: Use Scanner Class to take the Input from the User)

Experiment No. 3:

Level 1: A power distribution company charges its customers based on the number of units consumed. The tariff rates are as follows:

- For the first 100 units: ₹1.50 per unit
- For the next 200 units (101-300): ₹2.50 per unit
- For units above 300: ₹4.00 per unit

Level 2: Given an integer, perform the following conditional actions:

If n is odd, print Weird,

If n is even and in the inclusive range of to, print Not Weird

If n is even and in the inclusive range of to, print Weird,

If n is even and greater than, print Not Weird

Complete the Java code provided in your editor to print whether or not is weird.

Input Format:

A single line containing a positive integer, Constraints

1<=n<=100

Output Format

Print Weird if the number is weird; otherwise, print Not Weird.

Sample Input 0

3

Sample Output 0

Weird

Sample Input 1

24

Sample Output 1

Not Weird

Experiment No. 4:

Level 1: Write a Java program that calculates the total cost of an order based on the number of items and a per-item cost that varies depending on the type of item.

Purse Rs 500, Laptop Bag Rs 1200 and Head Phone 1000. If the user enters any other number, print invalid order item.

Level 2: Write a program to find the factorial value of any number entered by the user. Input the first line contains an integer T, total number of test cases. Then follow T lines, each line contains an integer N. Output: display the factorial of the given number N.

Sample input:

2

5 8

Sample output:

120

40320

Experiment No. 5:

Level 1: Write a program to identify common elements or numbers between two given arrays. You should not use any inbuilt methods. Input Length of arrays. Output : Common Elements.

Sample input

5 5

1 2 3 4 6

3 4 5 6 7

Sample output

3 4 6

Level 2: Write a program to find the roots of a Quadratic Equation.

Experiment No. 6:

Level 1: Write a program to count no of alphabets, digits, special symbols in the given string. Use Scanner class to read the input.

Sample input

hello123@

Sample output:

Letter 5

Digit 3

Symbol 1

Level 2: A university maintains Library applications which keep track of the books or magazines the students read in every month using matrix form. A student visits the university library in the month of June, July, and August. He reads fiction, non-fiction books and magazines, both in paper copies and online. The library application shows how many different types of books and magazines the student read. Use the information given below for better understanding.

Experiment No. 7:

Level 1: Presidency University stores the faculty details such as facid, facname, gender, course, and date of joining. Write a Java application to store all the details using a constructor, a parameterized constructor, and a method to display the data.

Level 2: Design a class to represent a bank account. Include the following members: Data Members: - Name of the Customer, Account Number, Type of Account, Balance amount in the account and methods: To assign initial values, to deposit an amount, to withdraw an amount after checking balance, to display name and the balance. Write a Javaprogram to demonstrate the working of the various class members and store five customers' details using Object Array.

Experiment No. 8:

Level 1: Write a Java program to create a class called "IDGenerator" with a static variable 'nextID' and a static method "generateID()" that returns the next ID and increments 'nextID'. Demonstrate the usage of generateID in the main method.

Level 2: An engineer wants to calculate the area of different shapes for the painting. Write a java application to do the same with the help of overloading.

Experiment No. 9:

Level 1: Write a program to show String and String Buffer methods. Read the input using Scanner.

Level 2: Write a Java program to find the length of each word in given sentence and concatenate or join those words length and display the output.

Sample Input

Welcome to Java

Sample output

724

Experiment No. 10:

Level 1: Write a Code to show single level Inheritance

Level 2: Create a class called Rectangle and store length, width using the constructor. Calculate the area using that. Create a tabletop using rectangle class and calculate the cost of painting that tabletop

Experiment No. 11:

Level 1: Create a class 'Emp' by extending Person class with additional data member empno, position with following features:

a. Default constructor b. Parameterized constructor c.Input method which takes values from user and assigns to data members and calls input method of Person d.Output method to display all data and calls output method of Person

Define a class Manager by extending Emp with data member bonus. Provide necessary constructors and override input and output methods. Create objects of manager in main class

Level 2: A company has two types of employees. 1. Salaried and 2. Daily wage. Salaried employees have their position and basic salary. Daily waged employees have basic and number of days worked. The accounts department wants to calculate the salary for both the type of employees and want to display all details about the employee. Write a code to implement this concept with the help of Inheritance.

Experiment No. 12:

Level 1: Write a program to create interface A in this interface, we have two methods meth1 and meth2. Implement this interface in another class named MyClass. Include the interfaces in packages

Level 2: Write a Java program to create an interface Sortable with a method sort () that sorts an array of integers in ascending order. Create two classes BubbleSort and SelectionSort that implement the Sortable interface and provide their own implementations of the sort () method.

Sample Input

5 7 2 19 34 3

Sample Output

Bubble Sort : 2 3 5 7 19 34

Selection Sort : 2 3 5 7 19 34

Experiment No. 13:

Level 1: Shayam reads the integers from the user and calculates the division of the numbers. Help him to handle the scenario if the user enters string values or zero in second integer.

Level 2: Write a java program which contains method bintodecimal. If an invalid binary number is given as input, the method must throw NumberFormat Exception and display the exception message in the catch block.

Sample Input:

3

Sample Output

Number is not a Binary Number

Experiment No. 14:

Level 1: Write a Java program to create a method that takes a string as input and throws a NotaVowelException if the string does not contain vowels.

Level 2: Kumar wants to withdraw the amount from the bank. If his balance is less than the withdrawl amount, throw the insufficient_ fund_exception. Otherwise, display the current balance

Experiment No. 15:

Level 1: Write a Java program to show multiple threads and display their names.

Level 2: Anshul wants to print odd numbers and even numbers with the help of threads. Help him to complete the task by reading the limit from him.

Experiment No. 16:

Level 1: Shashi is developing java application using multiple threads. He ensures the main thread is the last one to exit from the program. Help him to understand this concept with the help of an example.

Level 2: Write a Java program that calculates the sum of all prime numbers up to a given limit using multiple threads.

Experiment No. 17:

Level 1: **Write a program to show synchronization of threads using Synchronized block.**

Level 2: **Vishal, Anu, and Deepak are going to withdraw the money from the ATM. ATM has a limited amount of cash only for withdrawals. All three are trying to withdraw at the same time. Solve the problem with the help of threads and develop a solution for withdrawing the amount one by one. If the balance in the ATM is less than the withdrawal amount or empty, display the insufficient balance in the ATM.**

Experiment No. 18:

Level 1: **Write a Java program to read input from the console and print the output to the Java console**

Level 2: **Write a Java program to find the longest word in a text file**

Experiment No. 19:

Level 1: **Write a program to show the file properties**

Level 2: **Write a Java program to compare two files lexicographically**

Experiment No. 20:

Level 1: **Write a Java program to create an array list, add some colors (strings), iterate, update, update and print out the collection.**

Level 2: **Write a Java program to implement a lambda expression to find the average of a list of doubles:**

Sample Output:

Original values: [3.5, 7.5, 4.3, 4.7, 5.1]

Average of the original values: 5.0200000000000005

Experiment No. 21:

Level 1: **Write a code to reverse the priorities of the String values in the PriorityQueue**

Level 2: **Write a Java program to implement a lambda expression to convert a list of strings to uppercase and lowercase.**

Sample output:

Original strings:

Red

Green

Blue

PINK

Lowercase strings:

red

green

blue

pink

Uppercase strings:

RED

GREEN

BLUE

PINK

Experiment No. 22:

Level 1: **Write a Java Program to establish a connection with the database.**

Level 2: **Presidency University Hr interested in storing all faculty details in the database. Write a program to help them store, retrieve and delete the faculty details in the database.**

Experiment No. 23:
Level 1: Write a program to demonstrate the use of PreparedStatement and ResultSet interface.
Level 2: Create a Application (Ticket Booking System) with login page & CRUD operation.

Experiment No. 24:
Level 1: Write a servlet that displays the welcome message.
Level 2: Write a Servlet program to send username and password using HTML forms and authenticate the user.

Experiment No. 25:
Level 1: Write a Java Servlet to display employee's name and id using Parameters.
Level 2: Create a web Application (Ticket Booking System) with login page & CRUD operation and deploy the same in Web Server.

Experiment No. 26:
Level 1: Develop a simple JSP web application.
Level 2: Develop a web application to pass the values from the servlet to JSP.

Experiment No. 27: Programming Exercises on JSP
Level 1: Write a jsp program to implement the single text field calculator.
Level 2: The University is organizing a cultural festival and the organizing teams wants to collect registrations for various events with the help of web page. Design a registration form for collecting the participant details and store the data in the database.

Mini Application using Core Java + JDBC

Programming: Implementation of given scenario using Java

1. Develop a Library management system with basic modules and users like

Database module: This has two functions – Insertion of data and extraction of data.

Report module: For the borrowed books list to display.

Availability module: To view the availability of books.

Search Module: search facility for books and members.

Users in the system:

Librarian

Student

User functions:

Librarian: Add, view, delete the book details and user details, issue and return books.

Student: view and requesting books, returning books.

2. Design an employee payroll management system with basic modules and its processes as

Admin:

Admin can Add/Edit/delete the employees.

Admin can Add/Edit/delete the schedule the work of the employees.

Admin can Add and calculate/Edit/Delete the Salary of the employee.

Employee:

Employees can view his/her schedule set by Admin.

Employees can check his/her attendance.

Employees can update his/her details.

Employees can View their salary details.

3. Design an online Quiz system with basic modules and its processes as follows

Users of the System

Teacher

Student

Functional Requirements

Teacher:

Can create quiz after getting logged in.

Can enter subjects and enter question with its options and answer at the time of creating quiz.

10 Question for each quiz required to be completed.

Student:

Can search quiz according to their interest.

select the id of quiz and ready to start it.

After completing all questions, result will be displayed automatically.

Can view the description about each and every question in the respective quiz

References/Manual/Software:

1. Cay Horstmann,"Core Java -Volume 1: Fundamentals", 12th Edition, Oracle Press, 2021.
2. Bruce Eckel, **Thinking in Java. 4th ed**.
3. R. Nageswara Rao**, Core Java: An Integrated Approach, New: Includes All Versions upto Java 8**
4. Brett McLaughlin, **Head First Object-Oriented Analysis and Design: A Brain Friendly Guide to OOA&D**
5. NPTEL Course on "Java Programming", Prof. Debasis Samanta, https://archive.nptel.ac.in/courses/106/105/106105191/
6. "Head First Java" by Kathe Siera and Bert Bates, 2nd edition https://www.rcsdk12.org/cms/lib/NY01001156/Centricity/Domain/4951/Head_First_Java_Second_Edition.pdf.
7. "Building java programs" https://presiuniv.knimbus.com/user#/searchresult?searchId=java%20programming&_t=1662620793642

Course Code: CSA4602	**Course Title: Machine Learning Lab** **Type of Course: Lab**	**L-T-P-C**	0	0	2	1
Version No.	1.0					
Course pre-requisites	**NIL**					
Anti-requisites	**NIL**					
Course Description	This course builds the foundational insight of understanding principles of machine learning process and to explore various ML algorithms and practices to accelerate strategic decision making with data. Overall, this course gives a comprehensive insight to appropriately choose ML algorithms on real time problems to construct an intelligent ML model.					
Course Objective	The objective of the course is to familiarize the learners to explore the competence and comprehend with potential machine learning algorithms and techniques to revolutionize with real-world problems and create prominent solutions (with ML models) to attain Employability Skills through Experiential Learning techniques.					
Course Outcomes	**On successful completion of the course the students shall be able to:** **CO1:** Apply simple and multiple linear regression models to perform forecasting using real-world datasets. **(Apply)** **CO2:** Analyze categorical datasets using ID3 decision tree models with Gini Index and Information Gain for classification problems. **(Apply)**					

	CO3: Apply probabilistic learning models, including Bayesian Classifiers and Bayesian Belief Networks, for inference and classification tasks. **(Apply)** **CO4:** Apply margin-based and neural network models such as Support Vector Machines and Artificial Neural Networks for supervised learning problems. **(Analyze)** **CO5:** Analyze deep learning and ensemble learning models including Backpropagation, AdaBoost, XGBoost, and Random Forests for predictive performance. **(Analyze)** **CO6:** Evaluate datasets using dimensionality reduction and unsupervised learning techniques such as PCA, LDA, K-means, and Hierarchical Clustering. **(Evaluate)**
Course Content:	

List of Laboratory Tasks:

Experiment 1: Implementation and Evaluation of Simple Linear Regression Model

Level 1: Use dataset for generating ML model for sales forecasting

Level 2:

Experiment 2: Implementation and Evaluation of Multi-Linear regression Model

Level 1: Practice multi-output regression model to capture non-linear relationships for energy consumption forecasting application

Experiment 3: Implement and Evaluate decision Tree Model using ID3 Algorithm with GINI Index

Level 1: Use generating categorical inputs for Credit card Fraud detection

Level 2: Apply the GINI Index to develop ML model

Experiment 4: Building a decision Tree using ID3 Algorithm and Information Gain and Evaluate the results.

Level 1: Apply ID3 algorithm for Customer Churn Prediction (telecom/SaaS)

Level 2: Apply lead code problems on Decision tree using ID3 algorithm

Experiment 5: Implement Bayesian Classification Model

Level 1: Develop ML model for classifying text/email

Experiment 6: Apply Bayesian Belief Network

Level 1: Develop BFN based model for disease prediction

Experiment 7: Implement Support Vector Machine Model

Level 1: Practice various types of Supervised learning algorithms

Level 2: Practice different types of classification algorithms (Handwritten digits, classify tumors as malignant or benign.

Experiment 8: Build an Artificial Neural Network Model

Level 1: Use ANN to identify objects and make decisions/ detect anomalies in X-rays or MRIs

Experiment 9: Build a Back Propagation Model

Level 1: Apply back propagation to generate information retrieval model from custom document

Experiment 10: Implement PCA on a given dataset and implement any classification model

Level 1: Show how dimensionality reduction retains its accuracy and removes noise and redundancy features for facial feature dataset

Experiment 11: Implement ADA Boosting Algorithm

Level 1: Understand to handle imbalanced data, working with noisy data and non-linear relationships

Level 2: Use misclassified fraudulent transaction data that uses AdaBoost to train the sequence of weak classifiers

Experiment 12: Implement XG Boost Classifier

Level 1: Demonstrate an online advertising via a click-through rate prediction and generate a customer churn prediction by applying various service apps.

Experiment 13: Implement LDA on a Given Dataset and Implement any Classification model

Level 1: List various classification/Regression models

Level 2: Apply LDA on an 3D image.

Experiment No.14: Implementation of Random Forest Algorithm

Level 1: Level 1: List various classification/Regression models

Level 2: Use a credit scoring dataset to generate a ML model for approve/reject loan

Experiment 15: Implementation of K-means Clustering

Level 1: Identify the generative model for clustering

Level 2: Use to create models to group similar documents using features like TF-IDF.

Experiment No. 16: Implementation of Hierarchical Clustering

Level 1: Identify the generative model for clustering
Level 2: Use to create models on personalized treatment recommendations based on patient clusters

References:

1. Jupyter Notebook/ JupyterLab, VS code(Code Editor), PyCharm (Python IDE) - IDEs
2. Flask/FastAPI – Python web frameworks
3. Docker – Containerize and ship ML apps
4. Google Colab / Kaggle Kernels – Cloud-based GPU access
5. Pytorch, Scikit-learn, Tensorflow

T1. Machine Learning: An Algorithmic Perspective, Stephen Marshland, 2nd Edition, CRC Press, TayLevelr & Francis group, ISBN: 978-1-4665-8333-7, November 2014.
T2. Machine Learning in Action, Peter Harrington, ISBN: 978-935-004-4131, April 2012.
T3. Introduction to Machine Learning, Ethem Alpaydin, 4th Edition, The MIT press, ISBN:978-0-262-043-793, March 2020.

Textbooks

T1: Alpaydin, Ethem. *Introduction to Machine Learning*. 4th ed., MIT Press, 2020.
T2: Géron, Aurélien. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. 3rd ed., O'Reilly Media, 2022.

Reference Books

T1: Bishop, Christopher M. *Pattern Recognition and Machine Learning*. Springer, 2016.
T2: Murphy, Kevin P. *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.

Web Resources

1. Scikit-learn Documentation. https://scikit-learn.org
2. TensorFlow Documentation. https://www.tensorflow.org
3. Kaggle Learn. https://www.kaggle.com/learn

SEMESTER III

Course Code: CSA4206	**Course Title:** Software Engineering **Type of Course:** Program Core	**L-T- P- C**	3	0	0	3
Version No.	1.0					
Course Pre-requisites	**NIL**					
Anti-requisites	**NIL**					
Course Description	This course is intended to provide the students with an overall view over the Software Engineering Discipline and with insight into the processes of software development. This course also provides students with a theoretical as well as practical understanding of agile software development practices, how small teams can apply them to create high-quality software and insights of different software architectures.					
Course Objective	The objective of the course is EMPLOYBILITY of students by using PARTICIPATIVE LEARNING techniques.					
Course Outcomes	On successful completion of this course the students shall be able to: **[CO1]** Describe software engineering principles and activities involved in building large software programs. **[Understand]** **[C02]** Implement the process of requirement gathering, requirement **[Apply]** **[C03]** Demonstrate the importance of software architecture. **[Apply]** **[CO4]** Implement appropriate software architectural styles to design a system that meets given functional and non-functional requirements. **[Apply]** **[C05]** Apply software quality factors, quality standards, and SQA practices to evaluate, ensure, and improve the quality and security of software systems. **[Apply]** **[CO6]** Analyze and utilize Agile development practices, Scrum frameworks, scaled Agile methodologies, and JIRA tools to improve software project planning, execution and delivery. **[Analyze]**						
Course Content:						
Module 1	**Software and Software Engineering and System Models**					**10 Sessions**
Introduction, Nature of software, Defining the discipline, Software Process, Software Application Domains, Software Myths, Terminologies, Role of management in software development. Software Process Models: Generic Models, Defining Framework Activity, Process Assessment and Improvement, Prescriptive Process Models, Requirements Definition, Preliminary Architectural Design, Resource Estimation, Cost estimation - COCOMO model.						
Module 2	**Software Prototyping and Specification**					**10 Sessions**
Prototype construction and evolution, Understanding Requirements, Requirements Engineering Types of Requirements, Feasibility Studies, Requirements Elicitation, Developing Use Cases, Requirements - Analysis Documentation, Software Requirement and Specification (SRS)						
Module 3	**Architectural Design**					**12 Sessions**
Software Architecture, Importance of Software Architecture, Its Role, Views, Component & Connector View and its architecture style, Architectural Considerations, Assessing Alternative Architectural Designs, Architectural Reviews. Software testing - Types of testing.						
Module 4	**Software Quality and Security**					**13 Sessions**

Software Quality, Quality Factors, the Software Quality Dilemma, Risk, Achieving Software Quality. Quality control and assurance, Elements of Software Quality Assurance, SQA Tasks, Goals, and Metrics, Formal Approaches to SQA. Introduction to Agility and Process, Agility Principles and Agile Development, Scrum and Other Frameworks. Tools: Agile tracking tools such as JIRA; Scaled agile frameworks: SAFe, Scrum@Scale, Disciplined Agile
Targeted Applications & Tools that can be used: JIRA, Confluence
Project work/Assignment: Mention the Type of Project /Assignment proposed for this course
Case Study: Health Care support System: Develop an SRS for the mentioned system
References T1] K.K. Aggarwal, Yogesh Singh, “Software engineering”, New Age International Publisher, Fourth edition,2022 T2] James Shore, Shane Warden, “The Art of Agile Development”, Second Edition, O'Reilly,2021 R1] Ari Takanen, Jared de Mott, Charlie Miller, "Fuzzing for Software Security Testing and Quality Assurance", Artech House Publishers, Second Edititon,2023 **Web resources:** **1.https://onlinecourses.nptel.ac.in/noc20_cs68/preview** **2.https://onlinecourses.nptel.ac.in/noc24_mg01/preview** **3. https://www.geeksforgeeks.org/software-engineering/software-engineering**

Course Code: CSA4503	**Course Title:** Data Analytics and Visualization **Type of Course:** Program Core	**L-T- P- C**	2	0	0	2
Version No.	1.0					
Course Pre-requisites	**Probability and Statistics, Advanced Python Programming**					
Anti-requisites	**NIL**					
Course Description	The Course consists of two parts where first Part covers advanced analytics that covers topics necessary to give businesses greater insight into their data than they could ordinarily, and the Second Part covers data visualization concepts. Primary concepts include machine learning, data mining, predictive analytics, location analytics, big data analytics, and location intelligence. Visualization for Time series, Geolocated data, Correlations, connections, Hierarchies, networks, and interactivity.					
Course Objective	The objective of the course is EMPLOYBILITY of students by using PARTICIPATIVE LEARNING techniques.					
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1**: Apply fundamental concepts of Data Analytics and the Data Science process using Python and essential libraries. **(Apply)** **CO2:** Implement data manipulation and preprocessing techniques using NumPy and pandas to analyse and clean diverse datasets. (**Apply)** **CO3**: Analyse data visualization techniques using matplotlib, seaborn, and pandas to interpret patterns, identify trends, and derive insights from datasets effectively. **(Analyse)** **CO4:** Utilize advanced data aggregation and time series analysis methods to handle and analyse real-world time-dependent datasets. **(Apply)** **CO5:** Examine time-series data and identify patterns for real-world applications. **(Apply)** **CO6:** Develop data-driven solutions and dashboards for decision-making scenarios. **(Apply)**					
Course Content:						
Module 1	**Introduction to Data Analytics**	**Sessions: 7**				
Introduction to Data Analytics, Exploratory Data Analysis and Data Science Process. Motivation for using Python for Data Analysis. Python Libraries: NumPy, pandas, matplotlib, SciPy, scikit-learn, stats models.						
Module 2	**Data Analytics with Python**	**Sessions 8**				
Arrays and vectorized computation, Introduction to pandas Data Structures, Essential Functionality, Summarizing and Computing Descriptive Statistics. Data Loading, Storage and File Formats. Reading and Writing Data in Text Format Data Cleaning and Preparation. Handling Missing Data, Data Transformation, String Manipulation.						
Module 3	**Introduction to Data Visualization**	**Sessions 7**				
Data Wrangling: Hierarchical Indexing, Combining and Merging Data Sets Reshaping and Pivoting. Data Visualization matplotlib: Basics of matplotlib, plotting with pandas and seaborn, other python visualization tools and advance visualization library of python.						
Module 4	**Data Aggregation and Visualization**	**Sessions: 8**				
Data Aggregation and Group operations: Group by Mechanics, Data aggregation, General split-apply-combine, Pivot tables and cross tabulation. Time Series Data Analysis: Date and Time Data Types and Tools, Time series Basics, date Ranges, Frequencies and Shifting, Time Zone Handling, Periods and Periods Arithmetic, Resampling and Frequency conversion, Moving Window Functions.						
Targeted Application & Tools that can be used: Exploratory Data Analysis, Data Wrangling, Statistical Analysis, Data Visualization, and Time Series Analysis using Python						

libraries such as NumPy, pandas, matplotlib, seaborn, SciPy, scikit-learn, and stats models.
Project work/Assignment: 1. **Customer Segmentation using EDA and Clustering** 2. **COVID-19 Data Analysis and Visualization** 3. **Stock Market Trend Analysis** 4. **Sales Data Cleaning and Reporting Dashboard** 5. **Social Media Sentiment Analysis (Basic Level)**
Topics related to 1. Problem Solving: Developing and optimizing machine learning algorithms for predicting stock market trends using real-time data. 2. Employability: Simulation of AI-based customer support chatbots for automating customer service workflows in e-commerce platforms using NLP and sentiment analysis.
References **T1.** McKinney, W. (2022). Python for Data Analysis: Data Wrangling with Pandas, NumPy and IPython. 3rd edition. O'Reilly Media. **T2**. Zaki, Mohammed J., Wagner Meira Jr, and Wagner Meira. *Data mining and machine learning: Fundamental concepts and algorithms*. Cambridge University Press, 2020. **R1**. George, Nathan. *Practical data science with Python: learn tools and techniques from hands-on examples to extract insights from data*. Packt Publishing Ltd, 2021. **R2**. Gutman, Alex J., and Jordan Goldmeier. *Becoming a data head: How to think, speak, and understand data science, statistics, and machine learning*. John Wiley & Sons, 2021. **E-Resources:** **https://scikit-learn.org/stable/** **https://www.istqb.org/** **https://machinelearningmastery.com/time-series-forecasting-with-python/** **https://matplotlib.org/** **https://www.deeplearning.ai/**

Course Code: **CSA4606**	**Course Title:** **MERN Full Stack Development** **Type of Course:** **Program Core, Theory and Laboratory Integrated**	**L-T- P- C**	1	0	4	3
Version No.	1.0					
Course Pre-requisites	**Web Technology, DBMS**					
Anti-requisites	**NIL**					
Course Description	It provides students with a comprehensive understanding of full-stack development, covering both front-end and backend technologies. MERN Stack course includes MongoDB (database), Express.js (backend framework), ReactJS (frontend), and Node.js (runtime environment). The course offers hands-on experience in building real-world web applications using the Mern Stack. This practical approach allows students to apply theoretical concepts in a practical setting, enhancing their learning experience. This course incorporates modern technologies and practices such as NoSQL databases, RESTful APIs, and single-page application development. By studying MERN Stack development, students become familiar with these technologies, preparing them for the demands of the tech industry.					

Course Objective	**This course aims to equip learners with industry-relevant skills in Full Stack Web Development using the MERN (MongoDB, Express.js, React.js, Node.js) stack.**	
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1**: Explain and experiment with version control using Git and manage data using MongoDB and MySQL databases **[Understand]** **CO2:** Apply Node.js modules and ReactJS concepts to build interactive user interfaces **[Apply]** **CO3** Develop dynamic and responsive web pages using modern web technologies **[Apply]** **CO4:** Build and deploy web applications using scripting techniques and development tools **[Apply]** **CO5:** Design and integrate full-stack applications using MongoDB, Express.js, ReactJS, and Node.js with efficient data handling and API communication **[Analyze]** **CO6 :** Apply React.js techniques to integrate frontend applications with backend REST APIs for data retrieval and manipulation.[**Apply]**	
Course Content:		
Module 1	**Foundations of Full Stack Development**	**19 Sessions (4T + 15P)**
Overview of Web Development Ecosystem, Introduction to Full Stack Development, Overview of MERN Stack, Role of MongoDB, Express.js, React.js and Node.js, Version Control using Git and basic Git commands, Introduction to Databases, Comparison of MySQL and MongoDB		
Module 2	**Backend Development with Node.js**	**19 Sessions (4T + 15P)**
Introduction to Node.js, Node.js Architecture and Event Loop, Node.js Modules, Node Package Manager (NPM), Creating a basic Node.js application, Core Node modules, Working with Events and Event Listeners, Timers in Node.js, Callbacks and asynchronous programming.		
Module 3	**Backend APIs and MongoDB Integration**	**19 Sessions (4T + 15P)**
Introduction to Express.js framework, Creating Express applications, Middleware concepts in Express.js, Routing in Express.js, Development of RESTful APIs, MongoDB architecture and installation, MongoDB data model with collections and documents, CRUD operations in MongoDB, Connecting MongoDB with Node.js.		
Module 4	**Frontend Development with React.js**	**18 Sessions (3T + 15P)**
Introduction to React.js, React application architecture, JSX syntax, React.js components, Props and state in React, Event handling in React, Introduction to React Hooks, Routing using React Router, Integrating React with backend REST APIs.		
List of Laboratory Tasks: **Experiment No. 1:** **Level 1**: Install Git on local machines. Initialize a local Git repository. Perform basic Git operations such as adding files, committing changes, and viewing the commit history. **Level 2:** Practice creating branches and merging changes. Push changes to a remote repository (e.g., GitHub, GitLab). **Experiment No. 2:** **Level 1:** Install Node.js and npm on local machines. **Level 2:** Create a simple Node.js application. Use npm to manage dependencies and install packages. **Experiment No. 3:** **Level 1:** Implement basic server functionalities such as handling HTTP requests. **Level 2:** Implement basic server functionalities such as handling HTTP responses. **Experiment No. 4:** **Level 1:** Run and test the Node.js application locally. **Level 2**: Creating Application with npm init and Installed Modules.		

Experiment No. 5:

Level 1: Create a message.txt file in the same directory and add some content in node.js

Level 2: Creating Web-based Node Application.

Experiment No. 6:

Level 1: Creating a server in Node

Level 2: Read request and return response in Node.

Experiment No. 7:

Level 1: Create a Calculator Node.js Module with functions add, subtract and multiply. And use the Calculator module in another Node.js file.

Level 2: Create node.js program with the content 'FS module' to perform file operations.

Experiment No. 8:

Level 1: Install MongoDB on local machines or a virtual environment. Configure MongoDB to run as a service.

Level 2: Access the MongoDB shell and perform basic database operations (e.g., creating databases, collections, inserting documents).

Experiment No. 9:

Level 1: Perform CRUD operations (Insert, Update, Delete and Query Documents) on 'Student' Database.

Level 2: Do queries involving MongoDB update operator.

Experiment No. 10:

Level 1: Perform different query modifiers on 'Student' Database

Level 2: Implement different aggregation commands on 'Student' Database.

Experiment No. 11:

Level 1: Perform CRUD operations (Insert, Update, Delete and Query Documents) on Employee Information Database.

Level 2: Do queries involving MongoDB update operator.

Experiment No. 12:

Level 1: Perform different query modifiers on Employee Database

Level 2: Implement different aggregation commands on Employee Information Database.

Experiment No. 13:

Connect a Node.js application to MongoDB using the official MongoDB Node.js driver

Experiment No. 14:

Implement basic server functionalities such as handling HTTP requests and responses. Run and test the Node.js application locally

Experiment No. 15:

Level 1: Learn about the basics of npm commands such as npm init, npm install, and npm publish.

Level 2: Explore the npm registry to search for and install existing Node.js modules.

Experiment No. 16:

Create a simple Node.js project and install external modules using npm.

Experiment No. 17:

Level 1: Create React Application implements input box for a floating number input.

Level 2: Create a simple ReactJS Application to Pass Data from One Component to Another Component in.

Experiment No. 18:

Design responsive components such as navigation menus, cards, and tables using Bootstrap classes and utilities.

Experiment No. 19:

Create a React.js application for food delivery website where users can order food from a particular restaurant listed in the website.

Experiment No. 20:

Create a web application to manage the TO-DO list of users, where users can login and manage their to-do items using MERN stack (MongoDB, ReactJS, NodeJS).

Targeted Application & Tools that can be used:

MongoDB, Express.js, React.js, Node.js, Visual Studio Code, Sublime Text, Atom, Git and GitHub.

Project work/Assignment:

1. **Create dynamic, interactive, and scalable web applications.**

Topics related to

1. MongoDB: Ensure proper schema design and indexing to optimize queries. Use tools like MongoDB Compass for visual debugging
2. Express.js: Implement middleware for error handling and logging. Use tools like Postman to test API endpoints
3. React.js: Utilize React Developer Tools for inspecting component hierarchies and state. Handle errors gracefully with error boundaries.
4. Node.js: Use debugging tools like Node Inspector and logging libraries like Winston to track server-side issues.

Textbook(s):

T1. Vasan Subramanian, 'Pro MERN Stack, Full Stack Web App Development with Mongo, Express, React, and Node', Second Edition, Apress, 2020.

T2. Brad Dayley, Brendan Dayley, Caleb Dayley, 'Node.js, MongoDB and Angular Web Development', Addison-Wesley, Second Edition, 2021.

References

1. https://www.tutorialspoint.com/the_full_stack_web_development/index.asp
2. https://www.coursera.org/specializations/full-stack-react
3. https://www.udemy.com/course/the-full-stack-web-development

Course Code: CSA4603	**Course Title:** Data Analytics and Visualization Lab **Type of Course:** Program Core	**L-T- P- C**	0	0	2	1
Version No.	1.0					
Course pre-requisites	Probability and Statistics, Advanced Python Programming					
Anti-requisites	**NIL**					

Course Description	This laboratory course is designed to provide hands-on experience in data analytics using Python and its powerful libraries such as NumPy, pandas, matplotlib, seaborn, SciPy, and stats-models. The focus is on applying analytical techniques to real-world datasets, enabling learners to explore, clean, visualize, and interpret data effectively. Through structured experimentation, students will develop the practical skills necessary for implementing data science workflows, statistical analysis, and time series forecasting.
Course Objective	**The objective of the course is EMPLOYBILITY of students by using EXPERIENTIAL LEARNING techniques.**
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1**: Apply NumPy and pandas to perform fundamental data manipulation, indexing, filtering, and cleaning tasks on structured datasets. (Apply) **CO2:** Use Python libraries to load, process, and transform datasets, including handling missing data, performing string operations, and converting formats. (Apply) **CO3**: Apply visualization techniques using matplotlib, pandas, and seaborn to generate graphical representations that highlight data patterns, distributions, and relationships for effective data interpretation. (Apply) **CO4:** Apply aggregation techniques, time series analysis, and hierarchical indexing to perform advanced analytical operations on real-time data**.** (Apply) CO5: Analyze complete data analytics workflows to evaluate the effectiveness of visualization and preprocessing techniques.(Apply) CO6: Apply time series analysis techniques for forecasting and trend analysis in real-world datasets.(Apply)

List of Experiments / Exercises

Experiment No. 1:
Level 1: Introduction to NumPy arrays – Creation, indexing, slicing
Level 2: Mathematical operations and broadcasting using NumPy

Experiment No. 2:
Level 1: Getting started with pandas – Series and Data Frame creation
Level 2: Basic data inspection, selection, filtering using pandas

Experiment No. 3:
Level 1: Introduction to matplotlib – Line and bar charts
Level 2: Customizing plots with titles, labels, and legends

Experiment No. 4:
Level 1: Overview of Python libraries – Exploring SciPy and stats-models
Level 2: Use of simple statistical functions for descriptive analysis

Experiment No. 5:
Level 1: Data loading from CSV and Excel using pandas
Level 2: Writing cleaned data to different file formats

Experiment No. 6:
Level 1: Handling missing values – Detection and imputation techniques
Level 2: Data transformation and type conversions

Experiment No. 7:
Level 1: String operations and regular expressions in pandas

Level 2: Cleaning and formatting textual data

Experiment No. 8:
Level 1: Summary statistics using describe, mean, std functions
Level 2: Custom aggregations and value counts

Experiment No. 9:
Level 1: Combining datasets – Merge, join, and concat operations
Level 2: Reshape datasets using pivot and melt

Experiment No. 10:
Level 1: Plotting with pandas' built-in visualization
Level 2: Create statistical plots using seaborn

Experiment No. 11:
Level 1: Hierarchical indexing – Creation and manipulation
Level 2: Group-wise transformations using multi-index

Experiment No. 12:
Level 1: Group By mechanics and applying aggregation functions
Level 2: Create pivot tables and crosstab reports

Experiment No. 13:
Level 1: Time series basics – Parsing dates, datetime index
Level 2: Resampling, shifting, and moving average

Experiment No. 14:
Level 1: Working with time zones and frequency conversion
Level 2: Periods arithmetic and time series forecasting preparation

References

1. **T1.** McKinney, W. (2022). Python for Data Analysis: Data Wrangling with Pandas, NumPy and I Python. 3rd edition. O'Reilly Media.
2. **T2**. Zaki, Mohammed J., Wagner Meira Jr, and Wagner Meira. *Data mining and machine learning: Fundamental concepts and algorithms*. Cambridge University Press, 2020.
3. **R1**. George and Nathan. *Practical data science with Python: Learn tools and techniques from hands-on examples to extract insights from data*. Packt Publishing Ltd, 2021.
4. **R2**. Gutman, Alex J., and Jordan Goldmeier. *Becoming a data head: How to think, speak, and understand data science, statistics, and machine learning*. John Wiley & Sons, 2021.

E-Resources:

1. **https://scikit-learn.org/stable/**
2. **https://www.istqb.org/**
3. **https://machinelearningmastery.com/time-series-forecasting-with-python/**
4. **https://matplotlib.org/**
5. **https://www.deeplearning.ai/**

Course Code: CSA4605	Course Title: Mobile Application Development using Flutter Type of Course: Program Core, Theory and Laboratory Integrated	L-T- P- C	1	0	4	3

Version No.	1.0	
Course Pre-requisites	Python, Java, or C++, HTML, CSS, and JavaScript, Knowledge of SQL and NoSQL databases	
Anti-requisites	**NIL**	
Course Description	This course introduces students to Mobile Application development using Flutter, a modern cross-platform framework. It covers Dart programming, UI/UX design, state management, backend integration with Firebase, and deployment. Students will develop interactive mobile applications through hands-on lab sessions. The course emphasizes real-world app development, API integration, and best practices for performance optimization.	
Course Objective	**This course equips students with the skills to develop cross-platform mobile applications using Flutter and Dart. It covers UI/UX design, state management, backend integration with Firebase, and API communication. Students will gain hands-on experience in building, debugging, and optimizing mobile apps. The course also focuses on app deployment to the Google Play Store and Apple App Store.**	
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1**: Apply the fundamentals of Flutter and Dart to build basic mobile applications. ***(Apply)*** **CO2:** Design and develop intuitive, user-friendly mobile application interfaces using Flutter widgets. ***(Apply)*** **CO3:** Integrate backend services (such as APIs and databases) and manage application state effectively. ***(Apply)*** **CO4:** Analyze application performance, debug errors, and optimize mobile applications for efficiency and responsiveness. ***(Analyze)*** **CO5:** Develop and deploy complete cross-platform mobile applications following industry best practices. ***(Apply)*** **CO6 : Apply** deployment techniques to develop and deploy feature-rich cross-platform mobile applications**.[Apply]**	
Course Content:		
Module 1	**Introduction to Mobile App Development & Dart Programming**	**25Sessions [5T + 20P]**
Overview of Web Development Ecosystem, Introduction to Full Stack Development, Overview of MERN Stack, Role of MongoDB, Express.js, React.js and Node.js, Version Control using Git and basic Git commands, Introduction to Databases, Comparison of MySQL and MongoDB		
Module 2	**Flutter Basics & UI Development**	**25Sessions [5 T + 20P]**
Flutter Architecture & Widget Tree, Understanding Stateful & Stateless Widgets, Commonly Used Widgets: Text, Image, Button, GridView, AppBar, Bottom Navigation Bar, Layouts & Styling: Column, Row, Stack, Container, Padding, Margin, Alignments, Themes & Custom Styling, User Input & Forms Handling		
Module 3	**State Management & Firebase Integration**	**25Sessions [5 T + 20P]**
State Management Approaches: setState, Provider, Riverpod, Bloc, Working with API & JSON Data, Database Integration: Firebase Authentication (Google, Email-Password, OTP), Firebase Firestore for Real-time Database, Local Database (SQLite, Hive), Notifications: Push & Local, Advanced Flutter & Deployment, Animations & Custom UI, Gesture Detection & Touch Interactions, Maps & Location Services, Background Services & App Lifecycle.		
List of Laboratory Tasks: **Lab sheet 1: Environment Setup & Dart Basics** **Level 1: Flutter Development Environment Setup** • Installation of Flutter SDK, Dart, and Android Studio • Configuration of emulator and execution of a sample application **Level 2: Dart Basics – Variables & Data Types** • Development of a Dart program demonstrating various data types **Lab sheet 2: Control Flow & Functions**		

Level 1: Dart Control Flow – Loops & Conditionals

- Implementation of a program to check prime numbers

Level 2: Functions & Exception Handling

- Development of a factorial function with proper error handling

Lab sheet 3: Object-Oriented Programming & Async Programming

Level 1: OOP in Dart – Classes & Objects

- Design of a Student Management System using classes

Level 2: Asynchronous Programming

- Implementation of async/await and Future concepts

Lab sheet 4: Introduction to Flutter UI

Level 1: Basic Flutter Application

- Development of a simple app displaying text and images

Level 2: Stateless & Stateful Widgets

- Creation of an application with dynamic theme switching

Lab sheet 5: User Input & List Handling

Level 1: Forms & User Input Handling

- Development of a login form with validation using TextFormField

Level 2: ListView & GridView

- Building a contacts list application using ListView.builder()

Lab sheet 6: Navigation & Theming

Level 1: Navigation & Routing

- Implementation of multi-screen navigation using Navigator

Level 2: Styling & Theming

- Customization of application UI using ThemeData and colors

Lab sheet 7: UI Components & Gestures

Level 1: Bottom Navigation & Drawer

- Development of an app with BottomNavigationBar and Drawer

Level 2: Gesture Detection

- Implementation of tap and swipe gestures

Lab sheet 8: Animation & State Management

Level 1: Basic Animation

- Creation of a fade-in effect using AnimatedContainer

Level 2: State Management using setState

- Development of a counter application

Lab sheet 9: Advanced State Management & API Integration

Level 1: Provider State Management

- Development of a shopping cart application using Provider

Level 2: REST API Integration

- Fetching and displaying weather data using HTTP package

Lab sheet 10: Firebase Authentication

Level 1: Firebase Authentication

- Implementation of Google Sign-In in a Flutter application

Lab sheet 11: Firebase Firestore

Level 1: Cloud Database (Firestore)

- Development of a CRUD application (Create, Read, Update, Delete)

Lab sheet 12: Local Storage

Level 1: Persistent Storage

- Saving user preferences using Hive/Shared Preferences

Lab sheet 13: Push Notifications

Level 1: Firebase Cloud Messaging (FCM)

- Implementation of push notifications in a Flutter app

Lab sheet 14: **Media Handling**

Level 1: Image Upload & Storage

- Capturing and uploading images to Firebase Storage

Lab sheet 15: **Final Mini Project**

Level 1: Application Development Project

- Development of a complete mobile application integrating UI design, state management, Firebase services, and API integration

Targeted Application & Tools that can be used:

1. Android Studio (Recommended)
2. Visual Studio Code (VS Code)
3. Xcode (For iOS Development - macOS Only)

Project work/Assignment:

The course includes hands-on projects and assignments focusing on real-world mobile application development. Students will work on: To-Do List, Weather App, or Expense Tracker to reinforce Flutter concepts. Fetch and display data from REST APIs (e.g., News, Weather, or Movie API). Implement user authentication, real-time database, and cloud storage using Firebase. Students will generate APK files and optionally publish their apps on the Google Play Store.

Topics related to:

1. Fundamentals of Flutter framework and Dart programming, including data types, control structures, OOP, and asynchronous programming
2. Design and development of user interfaces using Flutter widgets, layouts, theming, and responsive components
3. Implementation of application features such as navigation, state management, user input handling, and animations
4. Integration of backend services including REST APIs, Firebase (Authentication, Firestore, Storage), and local data persistence

Textbook(s):

T1. Marco L. Napoli, *Beginning Flutter: A Hands-On Guide to App Development*, O'Reilly Media, 2019.

T2. Alessandro Biessek, *Flutter for Beginners: An introductory guide to building cross-platform mobile applications with Flutter and Dart 2*, Packt Publishing, 2020.

T3. Rap Payne, *Flutter & Dart Cookbook: Practical recipes for building cross-platform mobile apps*, Packt Publishing, 2022.

References

R1. Thomas Bailey, *Flutter for Dummies*, Wiley, 2020.

R2. Carmine Zaccagnino, *Flutter Apprentice: Beyond the Basics*, Razeware LLC, 2022.

R3. Martin Aguinis & Google Flutter Team, *Flutter Complete Reference: Create Modern Cross-Platform Apps*, BPB Publications, 2021.

Web Based Resources:

W1: https://www.geeksforgeeks.org/flutter/flutter-tutorial

W2: https://codelabs.developers.google.com/codelabs/flutter-codelab-first#0

Course Code: CSA8100	**Course Title:** Mini Project **Type of Course: Project Work**	**L-T- P- C**	0	0	0	3
Version No.	1.0					

Course Pre-requisites	Knowledge and Skills related to all the courses studied in previous semesters.
Anti-requisites	**NIL**
Course Description	Students observe science and technology in action, develop an awareness of the method of scientific experimentation, and often get an opportunity to see, study and operate sophisticated and costly equipment. They also learn about the implementation of the principles of management they have learnt in class, when they observe multidisciplinary teams of experts from engineering, science, economics, operations research, and management deal with techno-economic problems at the micro and macro levels. Finally, it enables them to develop and refine their language, communication and inter-personal skills, both by its very nature, and by the various evaluation components, such as seminar, group discussion, project report preparation, etc. The broad-based core education, strong in mathematics and science and rich in analytical tools, provides the foundation necessary for the student to understand properly the nature of real-life problems. The students have options to pursue this course as either Project Work and Dissertation at the university, or Project Work in an Industry/ Company/ Research Laboratory, or Internship Program in an Industry/Company.
Course Objective	**The objective of the course is to familiarize the learners with the concepts of Professional Practice and attain Employability Skills through Experiential Learning techniques.**
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1:** Identify real world computing problems related to local, regional, national or global needs. **[Apply]** **CO2:** Apply appropriate techniques or modern tools for solving the intended problem. **[Apply]** **CO3:** Design the experiments as per the standards and specifications. **[Apply]** **CO4:** Interpret the events and results for meaningful conclusions. **[Apply]** **CO5:** Appraise project findings and communicate effectively through scholarly publications. **[Analyze]** **CO6:** Evaluate and design end-to-end for data-driven and intelligent applications. **[Evaluate]**

Course Broad Schedule

Topics	Mark		Rubrics
1. Domain, Technology, Project Scope and Planning	10	5%	Domain and technology (5) Scope and planning (5)
2. Importance and relevance of working technology in solving the problem	10	5%	Importance and relevance of working technology (8) and identifying the applications (2)
3. Contributions to the project in solving real-world problems.	10	5%	Feature/characteristics/Applications selection and Constraints Identification (5), Analysis and Feature finalization subject to constraints and Design (5).
4. Application or usage of the project and impact on the society	20	10%	Application or usage of the project (10) and impact on the society (10) (SDG)
5. End Term Viva	100	50%	Technical Competency, Scope fulfilment, originality of problem solving, analysis and discussion of results, presentation content (delivery, clarity, supporting material quality), cost

			and impact analysis.
6. Project Report	10	5%	Compliance of report as per format (10)
7. *Supervisor	10	5%	Individual, Team work, discipline, Project Progression (10)
8. Publication/Patent	20	10%	Research Publication (15) and patenting the project work (5)
9. Git Hub Repository	10	5%	Uploading the complete coding, Reports, publication proof, patent proof and other necessary documents in Git Hub(10)
Total	200	100%	** For S.No 1-4 , 5 marks should be allotted for communication and documentation skills*

SEMESTER IV

Course Code: **CSA8300**	**Course Title: Major Project** **Type of Course: Project Work**	**L-T- P- C**	0	0	0	12
Version No.	1.0					
Course Pre-requisites	Knowledge and skills gained from all courses studied in previous semesters, including those applied in the mini-project.					
Anti-requisites	**NIL**					
Course Description	Students observe science and technology in action, develop an awareness of the method of scientific experimentation, and often get an opportunity to see, study and operate sophisticated and costly equipment. They also learn about the implementation of the principles of management they have learnt in class, when they observe multidisciplinary teams of experts from engineering, science, economics, operations research, and management deal with techno-economic problems at the micro and macro levels. Finally, it enables them to develop and refine their language, communication and inter-personal skills, both by its very nature, and by the various evaluation components, such as seminar, group discussion, project report preparation, etc. The broad-based core education, strong in mathematics and science and rich in analytical tools, provides the foundation necessary for the student to understand properly the nature of real-life problems. The students have options to pursue this course as either Project Work and Dissertation at the university, or Project Work in an Industry/ Company/ Research Laboratory, or Internship Program in an Industry/Company.					
Course Objective	**The objective of the course is to familiarize the learners with the concepts of Professional Practice and attain Employability Skills through Experiential Learning techniques.**					

Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1:** Identify real world computing problems related to local, regional, national or global needs. **[Understand]** **CO2:** Apply appropriate techniques or modern tools for solving the intended problem. **[Apply]** **CO3:** Design the experiments as per the standards and specifications. **[Apply]** **CO4:** Interpret the events and results for meaningful conclusions. **[Apply]** **CO5:** Appraise project findings and communicate effectively through scholarly publications. **[Analyze]** **CO6:** Evaluate and design end-to-end for data-driven and intelligent applications. **[Evaluate]**

Course Broad Schedule

Topics	Mark		Rubrics
1. Domain, Technology, Project Scope and Planning	10	5%	Domain and technology (5) Scope and planning (5)
2. Importance and relevance of working technology in solving the problem	10	5%	Importance and relevance of working technology (8) and identifying the applications (2)
3. Contributions to the project in solving real-world problems.	10	5%	Feature/characteristics/Applications selection and Constraints Identification (5), Analysis and Feature finalization subject to constraints and Design (5).
4. Application or usage of the project and impact on the society	20	10%	Application or usage of the project (10) and impact on the society (10) (SDG)
5. End Term Viva	100	50%	Technical Competency, Scope fulfilment, originality of problem solving, analysis and discussion of results, presentation content (delivery, clarity, supporting material quality), cost and impact analysis.
6. Project Report	10	5%	Compliance of report as per format (10)
7. *Supervisor	10	5%	Individual, Team work, discipline, Project Progression (10)
8. Publication/Patent	20	10%	Research Publication (15) and patenting the project work (5)
9. Git Hub Repository	10	5%	Uploading the complete coding, Reports, publication proof, patent proof and other necessary documents in Git Hub (10)

Total	200	100%	** For S.No 1-4 , 5 marks should be allotted for communication and documentation skills*

Discipline Electives

Full Stack Stream

Course Code: **CSA4701**	**Course Title:** **Agile Methodology and DevOps** **Type of Course:** **Discipline Elective**	**L-T- P- C**	3	0	0	3
Version No.	1.0					
Course Pre-requisites	**Adaptive Software Engineering**					
Anti-requisites	**NIL**					
Course Description	This course provides a strong foundation in Agile methodologies and DevOps practices for modern software development. It covers Agile principles, values, estimation techniques, and frameworks such as Scrum, Extreme Programming (XP), Unified Process, and EVO. The course also introduces DevOps concepts, lifecycle, and tools, along with practical exposure to Git for version control. Through assignments, case studies, and hands-on activities, students learn to apply Agile thinking and DevOps automation for efficient and high-quality software delivery.					
Course Objective	**The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.**					
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1:** Apply Agile concepts, values, principles, and estimation techniques to differentiate Agile from traditional development. ***(Apply)*** **CO2:** Analyze Agile frameworks such as Scrum, XP, EVO, and Unified Process. ***(Analyze)*** **CO3:** Analyze DevOps principles and their integration in software development lifecycle models. ***(Analyze)*** **CO4:** Implement version control using Git for repository management and collaboration. ***(Apply)*** **CO5:** Create a DevOps pipeline demonstrating CI/CD principles. ***(Create)*** **CO6:** Evaluate Agile and DevOps practices in team-based environments for efficiency and quality. ***(Evaluate)***					
Course Content:						
Module 1	**Foundations of Agile Methodology and Estimation – CO1**					**10 Sessions**
Introduction to Agile; Iterative and evolutionary development; Agile values and principles; Agile manifesto; Agile project management; Agile team interactions; Comparison with traditional models; Benefits of Agile; Estimation techniques.						
Module 2	**Agile Frameworks and Their Industry Significance – CO2**					**10 Sessions**
Agile evolution; Scrum framework; Sprint planning and backlog; Adaptive planning; Issues with waterfall model; Extreme Programming (XP); Unified Process; EVO methodology; Lifecycle phases, roles, and practices.						
Module 3	**Introduction to DevOps and Software Development Lifecycle – CO3, CO5, CO6**					**12 Sessions**
Basic Linux commands; SDLC models (Waterfall, Agile, Lean); Comparison of models; DevOps overview; Core elements; DevOps lifecycle; DevOps adoption; Tools; Build, integration, and deployment processes.						

Module 4	Version Control and Git Operations in DevOps – CO4	13 Sessions
DevOps tools; Version control concepts; Git fundamentals; Git workflow; Git vs GitHub; Installation and setup; Repository structure; File lifecycle; Staging, committing; Local and remote repositories.		
Project work/Assignment: • **Compare Agile with traditional software development models** • **Map DevOps lifecycle with tools** • **Demonstrate Git version control** **Projects:** • **AI-Powered Resume Analyzer** • **DevOps Pipeline Simulation** • **Smart Attendance System with Face Recognition**		
Topics related to **Problem Solving:** Apply efficient solutions to real-world challenges such as scalable systems, traffic management, and distributed computing. **Employability:** Simulate real-world environments including Agile lifecycle, DevOps pipelines, cloud deployment, and cybersecurity practices.		
Textbook(s): **T1: Agile and Iterative Development – Craig Larman (2006)** **T2: Brilliant Agile Project Management – Edward Scater (2015)** **T3: DevOps for Beginners – Craig Berg (2020)**		
References **R1: Story Card Maturity Model – Chetankumar Patel** **R2: Agile Software Engineering – Hazza & Dubinsky** **R3: Modern DevOps Practices – Gaurav Agarwal** **R4: Learning DevOps – Mikael Krief** **E-Resources:** https://www.agilealliance.org/agile101 https://www.atlassian.com/agile https://learn.microsoft.com/en-us/devops https://www.atlassian.com/git/tutorials		

Course Code: CSA4731	Course Title: Responsive Web Designing Type of Course: Discipline Elective	L-T- P- C	2	0	2	3
Version No.	1.0					
Course Pre-requisites	Web Technology					
Anti-requisites	NIL					
Course Description	This course introduces the fundamentals of modern web development, focusing on building structured, responsive, and interactive websites. Students will learn HTML for content structure, CSS for styling and layout, and JavaScript for dynamic behaviour. The course also covers Responsive Web Design using tools like Sass, Flexbox, and CSS Grids for mobile-friendly interfaces. Finally, students explore Web Project Management, including planning, budgeting, development, and deployment processes. Practical assignments and projects enhance hands-on skills for real-world web development.					

Course Objective	**The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.**	
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1:** Design structured web pages using fundamental HTML and HTML5 elements including text, links, images, tables, and forms. **[Apply]** **CO2:** Apply CSS and JavaScript to style web pages and add interactive client-side functionalities. **[Apply]** **CO3**: Develop responsive websites using modern techniques like Flexbox, CSS Grid, Sass, and mobile-first design principles. **[Apply]** **CO4**: Demonstrate an understanding of the web project lifecycle, including planning, execution, documentation, testing, and deployment. **[Apply]** **CO5**: Construct a fully functional, responsive website by integrating HTML, CSS, JavaScript, and a CSS framework to solve a real-world problem. **[Create]** **CO6**: Evaluate the usability and responsiveness of a web application across different devices and screen sizes using browser-based developer tools. **[Evaluate]**	
Course Content:		
Module 1	**Foundations of Web Design with HTML - CO1**	**15 Sessions (8T + 7P)**
Working of Web; HTML Markup for Structure; Creating simple page; Marking up text; Adding Links; Adding Images; Table Markup; Forms; HTML5.		
Module 2	**Styling and Scripting for Dynamic Web Pages – CO2**	**15 Sessions (8T + 7P)**
CSS; Formatting text; Colours and Background; Padding, Borders and Margins; Floating and positioning; Page Layout with CSS; Transition, Transforms and Animation; Javascript; Using Java Script.		
Module 3	**Techniques for Responsive Web Design – CO3, CO5, CO6**	**15 Sessions (8T + 7P)**
Sass for Responsive Web Design; Marking Content with HTML5; Mobile-First or Desktop-First; CSS Grids, CSS Frameworks, UI Kits, and Flexbox for RWD; Designing small UIs by Large Finger; Images and Videos in Responsive Web Design; Meaningful Typography for Responsive Web Design.		
Module 4	**Managing Web Development Projects – CO4**	**15 Sessions (6T + 9P)**
Project Life Cycle; Project Definition; Discovery and Requirements; Project Schedule and Budgeting; Running the project; Technical Documentation; Development, Communication, Documentation; QA and testing; Deployment; Support and operations.		
List of Laboratory Tasks: **Experiment No. 1:** Create a personal homepage using semantic HTML5 elements. (Level 1) **Experiment No. 2:** Design a student registration form using various HTML5 input types and validation attributes. (Level 1) **Experiment No. 3:** Style a blog post page using CSS to format text, add colors, and manage backgrounds. (Level 1) **Experiment No. 4:** Create a product catalog page demonstrating CSS box model properties. (Level 1) **Experiment No. 5:** Develop a photo gallery that uses CSS transitions and transforms for hover effects. (Level 1) **Experiment No. 6:** Create an interactive webpage using JavaScript to calculate and display the sum of two numbers. (Level 1) **Experiment No. 7:** Build an image slider/carousel using JavaScript to manipulate the DOM and handle events. (Level 1) **Experiment No. 8:** Analyze a modern website's HTML layout; identify semantic tags, forms, tables, and media. Write a report. (Level 2) **Experiment No. 9:** Compare layout models (float, flex, grid); create a webpage that toggles layouts using JavaScript. (Level 2) **Experiment No. 10:** Build a responsive navigation bar that collapses into a hamburger menu on mobile devices. (Level 2) **Experiment No. 11:** Develop a responsive product listing page using CSS Flexbox and Grid for mobile, tablet, and desktop views. (Level 2) **Experiment No. 12:** Implement a dark/light theme switcher using CSS custom properties and JavaScript local storage. (Level 2)		

Experiment No. 13: Create a responsive landing page for a business using a CSS framework (e.g., Bootstrap 5) and custom CSS. (Level 2)

Experiment No. 14: Draft a basic web project plan for a mini e-commerce site, including objectives, timeline, and testing strategies. (Level 2)

Experiment No. 15: Project: Personal Portfolio Website - Develop a fully responsive personal portfolio website integrating HTML, CSS, and JavaScript. (Level 2)

Experiment No. 16: Project: Responsive Business Landing Page - Build a complete, multi-section responsive landing page for a fictional business. (Level 2)

Experiment No. 17: Project: Mini E-commerce Product Page - Develop a functional product page with a gallery and shopping cart summary. (Level 2)

Targeted Application & Tools that can be used:

Web Applications. HTML, CSS, JavaScript, Sass, Flexbox, CSS Grids, Bootstrap, Browser Developer Tools.

Project work/Assignment:

1. **Create a fully responsive personal portfolio website.**
2. **Build a complete multi-section responsive landing page for a fictional business.**

Topics related to

Problem Solving: Choose Responsive Web Design Techniques to Optimize User Experience Across Devices.

Employability: Simulation of Real-Time Web Page Development using HTML, CSS, and JavaScript Frameworks.

Textbook(s):

T1: Jennifer Niederst Robbins, "Learning Web Design", O'REILLY 5th Edition, 2020.

T2: Ricardo Zea, "Mastering Responsive Web Design", PACKT Publishing, 2021.

T3: Justin Emond, Chris Steins, "Pro Web Project Management", Apress, 2020.

References

R1: Jon Duckett, "HTML and CSS: Design and Build Websites", John Wiley and Sons, edition 2021.

R2: Faithe Wempen, "Step by Step HTML 5", South Asian Edition, Microsoft Press and PHI Learning.

R3: Adam Freeman, "Pro jQuery 2.0", Apress, Latest Edition.

R4: Jeremy McPeak, "Beginning JavaScript", Wrox Publication, Latest Edition.

E-Resources:

1. **https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Responsive_Design**
2. **https://www.w3schools.com/html/html_responsive.asp**
3. **https://www.google.com/search?q=https://developers.google.com/web/fundamentals/design-and-ux/responsive**
4. **https://jsfiddle.net/**

Course Code: CSA4732	**Course Title:** Enterprise Computing with Java **Type of Course:** Discipline Elective	**L-T- P- C**	2	0	2	3
Version No.	1.0					

Course Pre-requisites	**Web Technology, DBMS**	
Anti-requisites	**NIL**	
Course Description	This course provides a comprehensive understanding of **Java enterprise development**, covering both foundational and advanced Java programming concepts. Learners will gain hands-on experience with **Java I/O**, **Generics**, **Lambdas**, **Servlets**, **JSP**, and **JPA with Hibernate** for building robust web applications. The course delves into **Spring Core**, **Spring MVC**, and **Spring Boot REST APIs**, enabling rapid enterprise-level development. It also introduces **automation tools** such as **Apache Maven** and **Selenium**, equipping students to manage projects efficiently and perform automated web testing. A blend of theory and intensive programming practice ensures industry-readiness in full-stack Java application development.	
Course Objective	**The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.**	
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1**: Apply Java programming concepts including I/O streams, collections, generics, and lambda expressions to build functional components in enterprise applications. **(Analyze)** **CO2:** Analyze the behavior of Java EE web applications using servlets, JSP, and MVC architecture to manage client-server interactions effectively. **(Apply)** **CO3**: Apply JPA and Hibernate ORM features to develop scalable and maintainable data persistence layers in enterprise solutions. **(Apply)** **CO4:** Analyze the components of the Spring framework to develop secure, RESTful web services and enterprise applications. **(Analyze)** **CO5:** Apply automation tools like Maven and Selenium WebDriver to automate the build, test, and deployment process in real-time projects. **(Apply)** **CO06:** Evaluate enterprise applications in terms of performance, security, and reliability. ***(Evaluate)***	
Course Content:		
Module 1	**Advanced Java Programming and Core Concepts – CO1**	**10 Sessions (6T + 4P)**
Review of Java; Java I/O; Advanced concepts of Java and Java New Features of Java; Collection framework, Annotation, Java generics, Lambda Expression, JDBC.		
Module 2	**Java EE Web Development with Servlets and JSP – CO2**	**10 Sessions (6T + 4P)**
Introduction to Eclipse & Tomcat; Servlet API Fundamentals; Servlet Context, Session, Cookies; Request Redirection Techniques; JSP Fundamentals; Reading HTML form Data with JSP; State Management with Java; JSP Standard Tag Library - Core & Function Tags; Building MVC App with Servlets & JSP; Complete App- Integrating JDBC with MVC App		
Module 3	**Data Persistence with JPA and Hibernate ORM – CO3**	**10 Sessions (6T + 4P)**
Fundamentals of Java Persistence with Hibernate: JPA for Object/Relational Mapping, querying database using JPQL and Criteria API (JPA). Hibernate: Architecture, HQL, Querying, Caching, Performance and Concurrency; First & Second Level Caching, Batch Fetching, Optimistic Locking & Versioning; Entity Relationships, Inheritance Mapping & Polymorphic Queries.		
Module 4	**Spring Framework and RESTful Web Services – CO4**	**15 Sessions (9T + 6P)**
Spring Core, Spring MVC, Spring Boot REST API: Understanding Spring Framework; Spring AOP (Aspect Oriented Programming); Using Spring MVC; Building a Database Web App with Spring and Hibernate Implementing Spring Security; Developing Spring REST API; Using Spring Boot for Rapid Development		
Module 5	**Automation Testing with Maven and Selenium – CO5**	**15 Sessions (9T + 6P)**
Introduction to Automation Tools; Apache Maven: Maven Fundamentals, Software Setup – Command lineand Eclipse, pom.xml and Directory Structure, Multi-Module Project Creation, Scopes, Dependency Management, Profiles; Functional/BDD Testing using Selenium, Selenium Fundamentals and IDE, Selenium WebDriver, Installation and Configuration, Locating Web Elements, Driver Commands, Web Element Command		

List of Laboratory Tasks:

Experiment No. 1:

Level 1: Use Serialization and Deserialization mechanism to develop a console application.

Level 2: Build a console application by using Collection framework and Annotation.

Experiment No. 2:

Level 1: Build a console application by using Collection framework and Lambda Expression

Level 2: Develop a console application that connect with MySQL Database and perform database transactions.

Experiment No. 3:

Level 1: Build a web application to connect with a database using Servlet that perform database manipulations.

Level 2: Build web applications to connect with a database using JSP that perform database manipulations.

Experiment No. 4:

Level 1: Construct a login application in respect of the MVC model.

Level 2: Implement a web application based on the MVC design pattern, to create an Employee Registration module using JSP, Servlet, JDBC, and MySQL database.

Experiment No. 5:

Level 1: Create Student mark processing project using Hibernate with Maven.

Level 2: Create a User Registration project using JSP, Servlet, Hibernate Framework, and MySQL database.

Experiment No. 6:

Level 1: Develop a User Login Form and will validate username and password with the MySQL database using the Hibernate framework.

Level 2: Build a complete Hibernate application with HQL CRUD operations using MAVEN, JSP, Servlet, Hibernate Framework, JPQL and MySQL database.

Experiment No. 7:

Level 1: Build CRUD RESTful API using Spring Boot 3, Spring Data JPA (Hibernate), and MySQL database.

Level 2: Build login or sign-in and registration or signup REST API using Spring boot, Spring Security, Hibernate, and MySQL database.

Experiment No. 8:

Level 1: Create Spring web application to implement SpringMVC framework using eclipse IDELanguage identifier

Targeted Application & Tools that can be used:

Java SE & EE, Eclipse, Tomcat, JSP, Servlets, JDBC, JPA, Hibernate, Spring Boot, Spring MVC, Spring Security, Apache Maven, Selenium WebDriver, MySQL.

Project work/Assignment:

1. **Build a console-based Library Management System using Java collections, generics, and I/O.**
2. **Develop a Student Registration Portal using Servlets and JSP.**
3. **Create an Employee Management System using JPA and Hibernate.**
4. **Build a Spring Boot-based Inventory Management System with RESTful API support.**
5. **Create a Test Automation Framework using Selenium WebDriver and Apache Maven.**

Topics related to

1. Problem Solving: Develop a full-stack Java web application integrating JSP, Servlets, JDBC, and Spring to address real-world business challenges such as inventory management or customer tracking.

2. Employability: Simulate industry-level scenarios using tools like Apache Maven and Selenium for automated testing, and apply Spring Boot REST APIs to build scalable web services, enhancing job-readiness in enterprise Java development.

Textbook(s):

T1. Fender, Young, "Front-end Fundamentals", Leanpub, 2021. **T2.** Horstmann, "Core Java Volume II - Advanced Features", 12th Edition – Pearson, 2023
References **R1.** Soni, Ravi Kant. "Full Stack AngularJS for Java Developers: Build a Full-Featured Web Application fromScratch Using AngularJS with Spring RESTful." ,Apress, 2021. **R2.** Mardan, Azat. "Full Stack JavaScript: Learn Backbone.js, Node.js and MongoDB.",Apress, 2015

Course Code: **CSA4704**	**Course Title: Data Modelling with NoSQL Databases** **Type of Course: Discipline Elective**	**L-T- P- C**	2	0	2	3
Version No.	1.0					
Course Pre-requisites	**RDBMS**					
Anti-requisites	**NIL**					
Course Description	This course is designed to provide the core concepts of NoSQL databases. It provides a comprehensive introduction to NoSQL databases to design, create, manage, and query databases using MongoDB and Cassandra with practical hands-on experience. It covers topics like database structure, installation, data manipulation, query optimization, security, and administration tools, enabling learners to effectively store and retrieve data in a distributed database environment.					
Course Objective	**The objective of the course is SKILL DEVELOPMENT of students using PARTICIPATIVE LEARNING techniques.**					
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1:** Understand the different types of NoSQL databases. ***(Understand)*** **CO2:** Understand the impact of clustering on database design. ***(Understand)*** **CO3:** Apply NoSQL development tools across different database types. ***(Apply)*** **CO4:** Apply indexing techniques in NoSQL databases for efficient query execution. ***(Apply)*** **CO5:** Apply and manage distributed database systems using sharding, replication, and CAP theorem. ***(Apply)*** **CO6:** Analyze performance and suitability of different NoSQL database types for real-world applications. ***(Analyze)***					
Course Content:						
Module 1	**Introduction to NoSQL and Database Modeling – CO1**					**14 Sessions (6L + 8P)**
Introduction to databases; Evolution from relational databases; Early DBMS; Relational database revolution; Need for NoSQL; Types of NoSQL databases; Distributed data management; ACID vs BASE; CAP Theorem; SQL vs NoSQL.						
Module 2	**NoSQL Database Types – CO2**					**14 Sessions (8L + 6P)**
Key-Value databases (features, hash functions, TTL keys); Document databases (collections, embedded documents, schemaless design); Column-family databases (architecture, terminology); Graph databases (network modeling); Partitioning techniques.						
Module 3	**NoSQL Technologies – CO3**					**16 Sessions (8L + 8P)**
MongoDB introduction; Data modeling; CRUD operations; Querying; Replication; Sharding; Deployment; Server administration; MapReduce; CouchDB (introduction, API, document handling, connectivity).						

Module 4	Distributed Databases – CO4, CO5, CO6	16 Sessions (8L + 8P)

Cassandra architecture; Data model; Keyspace and table operations; CQL; Collections; CAP theorem; Data modeling; Cassandra vs HBase; HBase shell and commands; Graph database basics; Redis data structures (keys, strings).

List of Laboratory Tasks:

Experiment No. 1:
- **Level 1: Connect to MongoDB and explore data**
- **Level 2: Setup MongoDB Compass**

Experiment No. 2:
- **Level 1: Create and manage databases and collections**
- **Level 2: Work with documents and collections**

Experiment No. 3:
- **Level 1: Perform CRUD operations**
- **Level 2: Switch between databases**

Experiment No. 4:
- **Level 1: Insert documents using insertOne()**
- **Level 2: Insert multiple documents using insertMany()**
- **Level 3: Delete using deleteOne() and deleteMany()**

Experiment No. 5:
- **Level 1: Retrieve using find() and findOne()**
- **Level 2: Update using updateOne() and updateMany()**

Experiment No. 6:
- **Level 1: Use MongoDB Query Language (MQL)**
- **Level 2: Create indexes using createIndex()**

Experiment No. 7:
- **Level 1: Complex queries using operators**
- **Level 2: Use operators ($gt, $lt, $in, $ne, $and, $or)**

Experiment No. 8:
- **Level 1: Aggregation pipeline ($match, $group)**
- **Level 2: Sorting and filtering**

Experiment No. 9:
- **Level 1: Use $lookup for joins**
- **Level 2: Complex transformations**

Experiment No. 10:
- **Level 1: Understand ACID in MongoDB**

Experiment No. 11:
- **Level 1: Transaction commit and rollback**

Experiment No. 12:
- **Level 1: MongoDB integration with Python**

Experiment No. 13:
- **Level 1: Text search and filtering**

Experiment No. 14:
- **Level 1: Install Cassandra**
- **Level 2: Create sample tables**

Experiment No. 15:
- **Level 1: Perform CQL operations (INSERT, SELECT)**
- **Level 2: Perform CQL operations (UPDATE, DELETE)**

Targeted Application & Tools that can be used:
MongoDB (v4.4+), Apache Cassandra, MongoDB Compass, Python, JS

Project work/Assignment:
1. **Blogging platform using MongoDB**

2. **Product catalog management system**
3. **Inventory management system**
4. **To-do list application**
5. **Social media mini platform**
6. **Contact management system**
7. **File sharing system**
8. **Bookstore management system**
9. **Real-time analytics using Cassandra**

Topics related to

1. **Distributed database design**
2. **Real-time data processing**
3. **Scalable application development**
4. **Industry-relevant NoSQL tools**

Textbook(s):

T1: MongoDB: The Definitive Guide – Shannon Bradshaw (2020)
T2: Mastering NoSQL Databases with MongoDB – Greyson Chesterfield (2024)
T3: NoSQL for Mere Mortals – Dan Sullivan (2015)

References

R1: Seven Databases in Seven Weeks – Eric Redmond
R2: Cassandra: The Definitive Guide – Jeff Carpenter
R3: Seven NoSQL Databases in a Week – Packt

Online Resources

- NPTEL NoSQL Course
- SWAYAM Courses
- MongoDB Certification
- Coursera & Simplilearn courses
- GeeksforGeeks MongoDB Tutorial

Course Code: CSA4705	**Course Title:** Backend Development with Node.js **Type of Course:** Discipline Elective	**L-T- P- C**	2	0	2	3
Version No.	1.0					
Course Pre-requisites	**Web Technology**					
Anti-requisites	**NIL**					
Course Description	It provides students with a comprehensive understanding of Server-side Scripting backend development, how to build efficient and scalable backend applications using Node.js. Starting from the basics and moving towards advanced concepts, this course will equip you with the skills necessary to develop high-performance server-side applications. This course incorporates modern technologies and practices such as Node.js, RESTful APIs. By studying Server-side Scripting, students become familiar with these					

	technologies, preparing them for the demands of the tech industry.	
Course Objective	**The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.**	
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1:** Apply server-side JavaScript in web application development **(Apply)** **CO2:** Analyze the web application development using HTTP, FS and Buffer modules **(Analyze)** **CO3:** Develop dynamic websites **(Apply)** **CO4:** Apply the node express, JSON, Socket.IO to allow high scalability with asynchronous code **(Apply)** **CO5:** Integrate MongoDB or MySQL with Node.js to build data-driven applications with full CRUD functionality **(Apply)** **CO6:** Construct real-time features using Socket.IO and manage collaborative development workflows using Git and GitHub **(Create)**	
Course Content:		
Module 1	**Getting Started with Node.JS – CO1**	**14 Sessions (L6 + P8)**
Introducing Node.JS, Node Package Manager (npm), Custom NPM modules, Installing Node, use Node.js REPL. Explore and use built-in modules of Node.js, Use of Node.JS and GitHub, collaborate on code with others using the git tool.		
Module 2	**Node JS Modules – CO2**	**14 Sessions (L8 + P6)**
Working with Node packages; Using Node package manager; Creating a simple Node.js application; Using Events; Listeners; Timers; Callbacks; Handling Data I/O; Implementing HTTP services in Node.js.		
Module 3	**Handling Data I/O in Node.js – CO3**	**16 Sessions (L8 + P8)**
Working with fs module, working with JSON, Using Buffer Module to Buffer Data, Using Stream Module to Stream Data, Compressing and Decompressing Data with Zlib. Implementing HTTP Services in Node.JS: Introduction to HTTP module, Processing URLs, Processing Query Strings and Form Parameters, Understanding Request, Response and Server Objects.		
Module 4	**Web Development with Node.JS – CO4, CO5, CO6**	**16 Sessions (L8 + P8)**
Introducing Express, More on Express, GET, POST, body Parser. Creating Middleware with Connect: What is Middleware? Middleware in Connect, Access Control with Middleware. Socket Services in Node.js: Understanding Network Sockets, A Socket.IO Chat Server, A Streaming Twitter Client.		
List of Laboratory Tasks: **Experiment No. 1:** Level 1: Install Node.js and npm on local machines. Level 2: Use npm to manage dependencies and install packages. **Experiment No. 2:** Level 1: Install Git on local machines. Initialize a local Git repository. Level 2: Perform basic Git operations such as adding files, committing changes, and viewing the commit history. **Experiment No. 3:** Level 1: Learn about the basics of npm commands such as npm init, npm install and npm publish. Level 2: Explore the npm registry to search for and install existing Node.js modules. **Experiment No. 4:** Level 1: Connect a Node.js application to MySQL. Level 2: Connect a Node.js to MySQL using Node.js driver. **Experiment No. 5:** Level 1: Install MongoDB on local machines or a virtual environment. Level 2: Configure MongoDB to run as a service. **Experiment No. 6:** Level 1: Access the MongoDB shell and perform basic database operations. Level 2: MongoDB shell Commands - Creating databases, Collections, inserting documents, Update, Delete.		

Experiment No. 7:
Level 1: Install external modules using node package manager (npm).
Level 2: Create a simple Node.js project.
Experiment No. 8:
Level 1: Create JavaScript Objects and functions Using object literal.
Level 2: Create JavaScript Objects and functions by creating instance of Object directly.
Experiment No. 9:
Level 1: Working with the arrays using Node.js.
Level 2: Assessing file system from Node.js.
Experiment No. 10:
Level 1: Create a basic food delivery website using node.js.
Level 2: Upload the project to GitHub repository.
Experiment No. 11:
Level 1: Implement basic socket services.
Level 2: Enhanced Socket Service with Rooms and Broadcasting.
Experiment No. 12: Create a website to manage the TO-DO list of users, where users can login and manage their to-do items using NodeJS and MongoDB using the official MongoDB Node.js driver.

Targeted Application & Tools that can be used:
Web Applications. MongoDB, Express.js, Node.js, Visual Studio Code, Sublime Text, Atom, Git and GitHub.

Project work/Assignment:
1. Create dynamic, interactive, and scalable web applications.

Topics related to
Problem Solving: Choose building scalable server-side applications using Node.js to handle asynchronous data processing and real-time communication.
Employability: Simulation of full-stack web applications using Node.js, Express, and Socket.IO to demonstrate industry-relevant backend development skills.

Textbook(s):
T1: Professional Node.js: Building JavaScript Based Scalable Software, 2020.

T2: Sams Teach Yourself Node.js in 24 Hours, 2021.

T3: Learn PostgreSQL: Build and manage high-performance database solutions using PostgreSQL, 2020

References

R1: Chris Northwood, 'The Full Stack Developer: Your Essential Guide to the Everyday Skills Expected of a Modern Full Stack Web Developer', Apress; 1st edition, 2021.

R2: Kirupa Chinnathambi, 'Learning React: A Hands-On Guide to Building Web Applications Using React and Redux', Addison-Wesley Professional, 4th edition, 2020.

E-Resources:

- https://www.tutorialspoint.com/nodejs/index.htm
- https://www.w3schools.com/nodejs/
- https://www.udemy.com/course/the-full-stack-web-development/

Artificial Intelligence Stream

Course Code: CSA4706	**Course Title:** Computer Vision **Type of Course: Theory & Integrated Laboratory**	**L-T- P- C**	**2**	**0**	**2**	**3**
Version No.	1.0					
Course Pre-requisites	CSA4201 CSA4305					
Anti-requisites	Nil					
Course Description	This course explores the foundations and advancements in digital image processing and computer vision. The course covers image formation, feature detection and matching, edge and corner detection, image segmentation, texture analysis, stereo vision, motion analysis, object recognition, and 3D scene understanding. Learners will gain hands-on experience in analysing visual data, and understanding geometric techniques for depth and motion analysis . Emphasis is placed on both theoretical underpinnings and practical implementations across real-world applications like face recognition, object tracking, and scene reconstruction.					
Course Objective	**The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.**					
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1**: Explain fundamentals of computer vision and image formation [**Apply**] **CO2:** Explain and apply image transformations for geometric and intensity-based image processing tasks**. [Apply]** **CO3:** Apply image processing techniques such as filtering, edge detection, segmentation, and feature extraction for visual data analysis [**Apply**] **CO4:**Explain the principles of scale-invariant feature detection and matching techniques used for robust object recognition **[Apply]** **CO5:** Develop pattern classification and face recognition techniques to solve real-world vision problems. **[Apply]** **CO6:** Evaluate Structure from Motion (SfM) and depth estimation methods for reconstructing 3D structures from multiple images and video sequences. [**Analyze**]					
Course Content:						
Module 1	**Foundations of Computer Vision - CO1 ,CO2**			**16 Sessions (10T + 6 L)**		
Introduction to Computer Vision: Definition and Applications of Computer Vision, Human vision vs computer vision, Digital Image Fundamentals, Image Formation, Types of Images, Components of Computer Vision Systems. **Camera Models:** Digital Camera, Pin hole Camera, orthographic and perspective projections, weak perspective projection, Intrinsic and extrinsic camera parameters, linear and nonlinear approaches of camera calibration, Human Color Perception.						
Module 2	**Image Processing Fundamentals -CO2**			**16 Sessions (10T + 6L)**		
Image Processing: Image Transformations, Image sampling and quantisation, Image Enhancement, Image filtering, Image Segmentation, Noise Removal techniques						
Module 3	**Image Descriptors and Features-CO3,CO4**			**15 Sessions (7T + 8 L)**		

Feature Detection: Edge detection-Canny, LOG, DOG, Interest points and corners, Harris and Hessian affine, local image features, feature matching and Hough transform, model fitting and RANSAC, scale invariant feature matching, Texture - Local Texture Representations Using Filters, Shape from Texture

Module 4	**Multiple View Geometry and Computer Vision Applications CO5,CO6**			**13 Sessions (7T + 6 L)**

Sterovision: Stereo camera geometry and Epipolar constraints, Stereo vision and stereopsis, Essential and fundamental matrix, image rectification, local methods for stereo matching: correlation and multi-scale approaches, optical flow, Structure from Motion (SfM), Depth estimation

Applications: Object Detection, Face Recognition, Pattern Classification

List of Laboratory Tasks:

Exp 1: Image Acquisition and Display using OpenCV
Level 1: Read and display grayscale and color images.
Level 2: Analyze image properties such as resolution, histogram, and pixel intensity distribution.
Exp 2: Image Enhancement Techniques
Level 1: Apply brightness, contrast, and gamma correction.
Level 2: Compare enhancement methods using quantitative image quality measures.
Exp 3: Image Filtering using Mean, Median, and Gaussian Filters
Level 1: Implement mean, median, and Gaussian filters.
Level 2: Evaluate filter performance on noisy images and compare results.
Exp 4: Edge Detection using Sobel, Prewitt, and Canny Operators
Level 1: Detect edges using Sobel and Prewitt operators.
Level 2: Implement Canny Edge Detection and compare edge maps.
Exp 5: Corner Detection using Harris Corner Detection
Level 1: Apply Harris Corner Detection on sample images.
Level 2: Detect and visualize key points using Shi-Tomasi Corner Detection
Exp 6: Principal Component Analysis (PCA) for Images
Level 1: Perform PCA on an image dataset.
Level 2: Reduce image dimensionality and reconstruct images using principal components.
Ex 7: Image Geometric Transformations
Level 1: Apply scaling, translation, rotation, and reflection.
Level 2: Implement affine and perspective transformations for image rectification.
Exp 8: Camera Calibration
Level 1: Capture calibration images using a checkerboard pattern.
Level 2: Estimate camera intrinsic and extrinsic parameters and remove distortion.
Exp 9: Stereo Vision and Depth Estimation
Level 1: Generate disparity maps using stereo image pairs.
Level 2: Compute depth maps and analyze distance estimation accuracy.
Exp 10: Object Tracking
Level1: Track moving objects using frame differencing.
Level 2: Implement tracking using Kalman Filter or Optical Flow techniques.
Exp 11: Image Classification using Machine Learning
Level 1: Train a classifier using image features.
Level2: Compare SVM, KNN, and Decision Tree classifiers for image recognition.
Exp 12: Detection
Level1: Detect objects using Haar Cascade or HOG features.
Level 2: Implement real-time object detection using pre-trained deep learning models.
Exp 13: Image Segmentation
Level 1: Perform threshold-based segmentation.
Level 2: Apply K-Means clustering or Watershed segmentation and evaluate results.
Exp 14: Face Recognition System
Level1: Detect faces using OpenCV face detector.
Level 2: Develop a face recognition system using PCA (Eigenfaces) or LBPH algorithm.

Exp 15: Mini Project: Computer Vision Application

Level 1: Develop a basic vision application such as object counting or image classification.

Level 2: Design and evaluate a complete computer vision solution incorporating detection, recognition, or tracking techniques.

Targeted Application & Tools that can be used:

TensorFlow, Keras, PyTorch, OpenCV, and Jupyter Notebook.

Project work/Assignment:

1. **Image Filtering and Feature Extraction**
2. **Geometric Transformation and Camera Calibration**
3. **Machine Learning for Image Classification**
4. **Project: Face Recognition System Development**
5. **Project: 3D Reconstruction from 2D Images**
6. **Project: Object Detection and Segmentation using Machine Learning**

Topics related to

1. Problem Solving: Develop and implement innovative image processing techniques to address challenges in segmentation, feature extraction, and pattern recognition in complex visual datasets.

2. Employability: Design and simulate a complete computer vision application such as a face recognition or object detection system that demonstrates practical, industry-relevant skills.

Textbooks

T1. Forsyth, D. A. and Ponce, J., "Computer Vision: A Modern Approach", Pearson Education India , 2nd Ed. 2015

T2.Richard Szeliski, Computer Vision: Algorithms and Applications, Springer-Verlag London Limited 2023.

References

R1. Richard Hartley and Andrew Zisserman, Multiple View Geometry in Computer Vision, 2ndEdition, Cambridge University Press, March 2004.

R2. Bishop; Pattern Recognition and Machine Learning, Springer, 2009.

R3. Muhammad Sarfraz, “Computer Vision and Image Processing in Intelligent Systems and Multimedia Technologies”, IGI Global, 2014.

E-Resources:

1. https://docs.opencv.org/master/d9/d0c/group__calib3d.html
2. https://web.stanford.edu/class/cs231a/
3. https://www.geeksforgeeks.org/computer-vision/computer-vision/
4. https://github.com/PacktPublishing/Practical-Computer-Vision/tree/master/Chapter05

Course Code: CSA4707	**Course Title:** Natural Language Processing **Type of Course:** Discipline Elective	**L-T- P- C**	2	0	2	3
Version No.	1.0					
Course Pre-requisites	**Data Structures and Algorithms**					
Anti-requisites	**NIL**					
Course Description	This course provides a comprehensive introduction to Natural Language Processing (NLP), combining theoretical foundations with hands-on programming. Learners will explore fundamental text preprocessing, word and text representation techniques, sequence labeling with probabilistic models, and practical NLP applications like sentiment analysis, machine translation, and text summarization. The course integrates machine learning concepts such as embeddings, RNNs, LSTMs, and attention mechanisms, preparing students to build real-world NLP systems using modern toolkits and language models.					
Course Objective	**The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.**					
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1**: Implement appropriate text preprocessing techniques such as tokenization, stemming, lemmatization, and stop word removal to clean and prepare textual data for NLP tasks. **[APPLY]** **CO2:** Implement word and sentence representation techniques including TF-IDF, Word2Vec, and n-gram models to extract meaningful features from text for downstream applications**. [APPLY]** **CO3**: Investigate sequence labeling problems using models like Hidden Markov Models and evaluate performance using algorithms like Viterbi for tasks such as Named Entity Recognition. **[APPLY]** **CO4:** Investigate and implement real-world NLP applications such as sentiment analysis, text summarization, and machine translation by integrating linguistic features with machine learning models. **[APPLY]** **CO5**: Communicate and present NLP concepts, methodologies, and project outcomes effectively through oral presentations, technical reports, and demonstrations. **[APPLY]** **CO6:** Design and develop a mini project or case study that applies Natural Language Processing techniques to solve a real-world problem using appropriate datasets and tools**.[APPLY]**					
Course Content:						
Module 1	**Foundations of Natural Language Processing**					**12 Sessions (6T + 6P)**
Introduction to Natural Language Processing, terminologies, empirical rules, why NLP is hard, why NLP is useful, NLP Processing pipeline, Corpus Cleaning techniques – word tokenization, sentence tokenization, word frequency distribution, stemming, lemmatization, dictionary, Introduction to Part of Speech Tagging, Textual Pre-Processing techniques – Stop words removal, regular expression, lower case, text standardization.						
Module 2	**Word and Text Representations in NLP**					**16 Sessions (8T + 8P)**
Word relationships, Word Embeddings techniques- bag of words, TF-iDF, Word2Vec and optimization. Simple N-gram models. Estimating parameters and smoothing. Negative Sampling Evaluating language models. Logistic regression – Sigmoid and Softmax. Perceptron and backpropagation. RNN, LSTM, CNN. Attention. Pre-trained language models. Multilinguality.						
Module 3	**Sequence Labeling and Parsing Techniques**					**16 Sessions (8T + 8P)**

Sequence Labeling, Hidden Markov Models. Best Emission Probability, Best Forward Probability and Viterbi Decoding Algorithms. Analysis of Viterbi Algorithm. Named Entity Recognition. Constituency Parsing.

Module 4	Real-World Applications of NLP	16 Sessions (8T + 8P)

Application of NLP. Lexical Resource Creation. Machine Translation. Sentiment Analysis. Lexical Simplification. Text Summarization.

List of Laboratory Tasks:

1. End-to-End NLP Pipeline

Build a simple NLP pipeline that takes raw text input and performs tokenization, cleaning, stop-word removal, and displays outputs at each stage.

2. Tokenization in Practice

Implement both word and sentence tokenization using an NLP library (such as NLTK or spaCy) and display the results for a sample dataset.

3. Custom Tokenizer Design

Develop a custom tokenizer that correctly handles punctuation, emojis, and special characters, and compare its output with a standard library tokenizer.

4. Text Cleaning and Normalization

Write a program to clean a noisy dataset by removing HTML tags, URLs, numbers, and special symbols using regular expressions.

5. Stopword Removal and Text Standardization

Perform stopword removal, lowercase conversion, and whitespace normalization, then compare raw vs processed text.

6. Frequency Analysis and Insights

- Generate a word frequency distribution from a cleaned corpus and identify:
- Top 10 frequent words
- Least frequent words
- Explain how this impacts NLP models.

7. Data Visualization for NLP

Visualize word frequency using bar charts or word clouds and interpret the patterns observed in the dataset.

8. Stemming vs Lemmatization

Apply both stemming and lemmatization techniques and compare outputs. Analyze which method preserves meaning better for real-world applications.

9. Part-of-Speech Tagging and Analysis

Perform POS tagging on sentences and identify patterns such as noun phrases and verb usage. Analyze tagging errors in ambiguous sentences.

10. Semantic Relationships

Using lexical resources (e.g., WordNet), find synonyms, antonyms, and compute similarity between word pairs. Analyze challenges with polysemous words.

11. Feature Extraction: BoW vs TF-IDF

Implement Bag of Words and TF-IDF representations for a document set. Compare both approaches and explain which is better for classification tasks.

12. Word Embeddings with Word2Vec

- Train a Word2Vec model (CBOW or Skip-gram) and:
- Display vector representations
- Perform similarity and analogy tasks

13. N-gram Language Modeling

Build unigram and bigram models and compute probabilities of given sentences. Apply smoothing techniques and evaluate using perplexity.

14. Text Classification using Machine Learning

Build a text classification model using logistic regression or Naive Bayes with TF-IDF features. Evaluate using accuracy,

precision, recall, and F1-score.

15. Deep Learning for NLP

Develop a simple deep learning model (RNN/LSTM or CNN) for text classification or sentiment analysis. Compare its performance with a traditional ML model.

Targeted Application & Tools that can be used:

Python (NLTK, spaCy, Gensim, Scikit-learn), TensorFlow, PyTorch, Hugging Face Transformers, Jupyter Notebook, Google Colab, Stanford NLP, TextBlob, Flair

Project work/Assignment:

1. Text Classification App
2. Sentiment Analysis Dashboard
3. Chatbot using RNN or Transformers
4. Named Entity Recognition System
5. Machine Translation App
6. Text Summarizer
7. Resume Parser for HR Systems
8. Topic Modeling on Research Papers
9. Speech-to-Text Processing Pipeline
10. Grammar and Spell Checker

Topics related to

Problem Solving: Text preprocessing, POS tagging, NER, sentiment analysis, and machine translation challenges. Employability: Building NLP pipelines, deploying models, chatbot development, word embeddings, and real-world applications using Python libraries.

Textbook(s):

T1. Daniel Jurafsky, and James H. Martin. Speech and Language Processing. (3rd Edition Draft,February 2024)

T2. Aditya Joshi, and Pushpak Bhattacharyya. Natural Language Processing. 1st Edition. Wiley Publishers. December 2023.

References

R1. Chris Manning and Hinrich Schutze, “Foundations of Statistical Natural Language Processing”,1st Edition, MIT Press

R2. Pawan Goyal. “Natural Language Processing”. 1st Edition, 2016.

E-Resources:

Weblinks

1. https://drive.google.com/file/d/10nbwAJd-dv6htOOZVBgAvLd1WscI0RqC/view
2. https://web.stanford.edu/~jurafsky/slp3/
3. https://nptel.ac.in/courses/106106211
4. https://nptel.ac.in/courses/106105158
5. https://nptel.ac.in/courses/106101007
6. https://nptel.ac.in/courses/106105572

Course Code: CSA4708	**Course Title:** Reinforcement Learning **Type of Course:** Discipline Elective	**L-T- P- C**	3	0	0	3
Version No.	1.0					
Course Pre-requisites	**Machine Learning**					
Anti-requisites	**NIL**					
Course Description	This course introduces the foundational concepts and practical applications of Reinforcement Learning (RL). Learners will explore core elements like agent-environment interactions, rewards, and policies through the lens of Markov Decision Processes (MDP). The course progresses through Monte Carlo methods, Temporal Difference learning (including SARSA and Q-Learning), and culminates with Multi-Armed Bandit (MAB) problems and an introduction to Deep Reinforcement Learning (DRL). Practical implementation using the OpenAI Gym environment is emphasized throughout to build hands-on skills in solving real-world sequential decision-making problems.					
Course Objective	**The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.**					
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1**: Infer the basic reinforcement learning environments to solve simple MDP problems using appropriate policy evaluation techniques. **[APPLY]** **CO2:** Build Monte Carlo prediction and control algorithms for model-free environments and evaluate their performance. **[APPLY]** **CO3**: Analyse the functionality of SARSA and Q-learning algorithms and determine their impact on policy optimization**. [APPLY]** **CO4:** Develop exploration strategies in Multi-Armed Bandit problems and demonstrate the selection of suitable algorithms for various real-time scenarios. **[APPLY]** **CO5:** Apply temporal-difference learning methods such as TD(0), TD(λ), and eligibility traces to improve prediction accuracy in reinforcement learning tasks. **[APPLY]** **CO6:** Design and implement deep reinforcement learning models using neural networks and evaluate their performance in complex, high-dimensional environments. **[APPLY]**					
Course Content:						
Module 1	**Foundations of Reinforcement Learning and MDPs - CO1**					**11 Sessions**
Elements of RL, Agent,environment Interface,Goals and rewards, RL platforms, Applications of RL, Markov decision proces s (MDP), RL environment as a MDP, Maths essentials of RL, Policy and its types, episodic and continuous tasks, return and discount factor, fundamental functions of RL – value and Q functions, model-based and model-free learning, types of RL environments, Solving MDP using Bellman Equation, Algorithms for optimal policy using Dynamic Programming -Value iteration and policy iteration, Example : Frozen Lake problem, Limitations and Scope.						
Module 2	**Monte Carlo Methods for Prediction and Control – CO2**					**11 Sessions**
Monte Carlo methods, prediction and control tasks, Monte Carlo prediction: algorithm, types of MC prediction, examples, incremental mean updates, Monte Carlo Control: algorithm, on-policy MC control, MC with epsilon-greedy policy, off-policy MC control. Limitations of MC method.						
Module 3	**Temporal Difference Learning Techniques – CO3 – CO5**					**11 Sessions**
Temporal difference learning: TD Prediction, TD Control: On-policy TD control – SARSA, computing the optimal policy using SARSA, Off-policy TD control – Q learning, computing optimal policy using Q learning, Examples, Difference between SARSA and Q-learning, Comparison of DP, MC and TD methods.						
Module 4	**Multi-Armed Bandits and Introduction to Deep Reinforcement Learning – CO4 – CO6**					**12 Sessions**

Understanding the MAB problem, Various exploration strategies – epsilon-greedy, softmax exploration, upper confidence bound and Thompson sampling, Applications of MAB - finding the best advertisement banner for a web site, Contextual bandits, introduction to Deep Reinforcement Learning (DRL) Algorithm – Deep Q Network (DQN).
Project work/Assignment: 1. Assignment on MDP Formulation. 2. Comparison Report: SARSA vs. Q-Learning. 3. Exploration Strategy Evaluation. 4. Project: Frozen Lake Navigation using Policy Iteration 5. Project: Train an Agent using Q-Learning to Play CartPole 6. Project: Contextual Multi-Armed Bandit for Ad Recommendation
Topics related to **1. Problem Solving:** Design a reinforcement learning-based strategy to solve the Frozen Lake navigation problem using Value Iteration and Policy Iteration. **2. Employability:** Simulate real-world decision-making environments using Multi-Armed Bandit algorithms to recommend optimal content or advertisements.
Textbook(s): **T1**. Maxim Lapan, *Deep Reinforcement Learning Hands-On*, 2nd Edition, Packt Publishing, 2020. **T2**. Sudharshan Ravichandiran, "Deep Reinforcement Learning with Python", Packt Publishers, Second Edition, 2020. T3. Miguel Morales, *Grokking Deep Reinforcement Learning*, Manning Publications, 2020
References **R1**. Laurra Graesser and Wan Loon Keng, "Foundations of Deep Reinforcement Learning", Pearson, 2022. **E-Resources:** 1. https://www.davidsilver.uk/teaching/ 2. https://spinningup.openai.com/en/latest/ 3. https://www.geeksforgeeks.org/temporal-difference-learning/ 4. https://lilianweng.github.io/lil-log/2018/05/22/exploration-strategies-in-deep-reinforcement-learning.html

Course Code: **CSA4709**	**Course Title: Deep Learning** **Type of Course: Theory & Integrated Laboratory**	**L-T- P- C**	2	0	2	3
Version No.	1.0					
Course Pre-requisites	**Machine learning**					
Anti-requisites	**NIL**					
Course Description	This course provides a comprehensive introduction to Deep Learning, equipping students with foundational knowledge and practical experience in building and training deep neural networks. Starting with feedforward networks and optimization techniques, the course progresses through convolutional neural networks for image and text classification, sequence modeling using RNNs and LSTMs, and culminates in the exploration of autoencoders for unsupervised learning and representation. Emphasis is placed on hands-on assignments, quizzes, and real-world projects to ensure a deep understanding of both theory and application.					
Course Objective	**The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.**					

Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1**: Apply optimization techniques such as stochastic gradient descent and momentum to train feedforward neural networks. **(Apply)** **CO2:** Implement convolutional neural networks for image and text classification tasks. **(Apply)** **CO3**:Implement recurrent neural networks and LSTM architectures for sequence modeling problems. **(Apply)** **CO4:** Develop autoencoder-based models to perform dimensionality reduction and feature extraction on real-world data. **(Apply)** **CO5:** Analyze the performance of deep learning models using evaluation metrics, loss functions, and validation techniques, and interpret their effectiveness across different datasets. **(Analyze)** **CO6:** Analyze and optimize deep learning architectures by selecting appropriate hyperparameters, regularization techniques, and training strategies for real-world applications. **(Analyze)**	
Course Content:		
Module 1	**Foundations of Deep Learning and Feedforward Networks - CO1**	**16 Sessions (8T + 8 L)**
Introduction: Historical context and motivation for deep learning; basic supervised classification task, optimizing logistic classifier using gradient descent, stochastic gradient descent, momentum, and adaptive sub-gradient method. Neural Networks: Feedforward neural networks, deep networks, regularizing a deep network, model exploration, and hyper parameter tuning.		
Module 2	**Convolutional Neural Networks for Image and Text – CO2**	**16 Sessions (8T + 8 L)**
Introduction to convolution neural networks: stacking, striding and pooling, applications like image, and text classification.		
Module 3	**Sequence Modeling with Recurrent Neural Networks – CO3 , CO4**	**15 Sessions (7T + 8 L)**
Unfolding computational graphs, recurrent neural networks (RNNs), bidirectional RNNs, encoder-decoder sequence to sequence architectures, deep recurrent networks, LSTM networks.		
Module 4	**Autoencoders for Representation Learning – CO5, CO6**	**13 Sessions (7T + 6 L)**
Autoencoders: Undercomplete autoencoders, regularized autoencoders, sparse autoencoders, denoising autoencoders, representational power, layer, size, and depth of autoencoders, stochastic encoders and decoders.		

List of Laboratory Tasks:

List of Laboratory Tasks:

Experiment No. 1:

Level 1: Implement logistic regression classification with (a) gradient descent and (b) stochastic gradient descent method. Plot cost function over iteration.

Level 2: Experiment with logistic regression by adding momentum term, and adaptive sub gradient method

Experiment No. 2:

Level 1: Implement a feed-forward neural network for solving (a) regression and (b) 2-class classification problem. Also experiment with hyper-parameter tuning.

Level 2: Train and test a feed-forward neural network for multi-class classification using softmax layer as output.

Experiment No. 3:

Level 1: Create a 2D and 3D CNN for image classification. Experiment with different depth of network, striding and pooling values.

Level 2: CNN-based model for sentiment analysis on a text dataset such as movie reviews or tweets.

Experiment No. 4:

Level 1: Implement (a) RNN for image classification, (b) GRU network and (c) Implement LSTM networks

Level 2: Simple Recurrent Neural Network (RNN) for predicting next word in a sentence.

Experiment No. 5:

Level 1: Bidirectional RNN for Sequence Classification

Level 2: Encoder-Decoder Architecture for Machine Translation

Experiment No. 6:
Level 1: LSTM Networks for Time-Series Prediction
Level 2: Implement an auto-encoder, denoising autoencoders and sparse autoencoders.

Experiment No. 7:
Level 1: Design stochastic encoders and decoders.

Targeted Application & Tools that can be used:
Image classification, text classification, sequence prediction, and representation learning using tools such as TensorFlow, Keras, PyTorch, OpenCV, and Jupyter Notebook.

Project work/Assignment:

1. Handwritten Digit Recognition using CNN (MNIST Dataset)
2. Sentiment Analysis using RNN or LSTM
3. Image Denoising with Autoencoders
4. Real-Time Object Detection System
5. Stock Price Prediction using LSTM
6. Face Recognition System using Deep Neural Networks

Topics related to
1. **Problem Solving**: Choose appropriate deep learning architectures (CNN, RNN, Autoencoders) for specific real-world data challenges such as image classification or sequence prediction.
2. **Employability**: Simulation of deep learning-based systems such as image classification, object detection, or time-series forecasting to develop industry-relevant skills.

Textbook(s):
T1. Chollet, F. (2021). *Deep Learning with Python (2nd Edition)*. Manning Publications

T2. Stevens, E., Antiga, L., & Viehmann, T. (2020). *Deep Learning with PyTorch*. Manning Publications.

T3. Raff, E. (2022). *Inside Deep Learning*. Manning Publications.

References

R1. Deng, L., & Yu, D. (2009). Deep Learning: Methods and Applications (Foundations and Trends in Signal Processing). Publishers Inc.
R2. Hall, M.L, (2011). Deep Learning. VDM Verlag
R3. David Foster, "Generative Deep Learning" O'Reilly Publishers, 2020.
R4. John D Kellehar, "Deep Learning", MIT Press, 2020.

E-Resources:
Weblinks

1. https://onlinecourses.nptel.ac.in/noc22_cs22/preview
2. https://www.coursera.org/learn/neural-networks-deep-learning?specialization=deep-learning
3. https://www.deeplearning.ai/
4. http://imlab.postech.ac.kr/dkim/class/csed514_2019s/DeepLearningBook.pdf

Course Code: CSA4710	**Course Title:** Generative AI **Type of Course:** Elective	**L-T- P- C**	2	0	2	3
Version No.	1.0					
Course Pre-requisites	**CSA4709 – Deep Learning** **CSA4707 – Natural Language Processing** **CSA4502 - Machine Learning**					
Anti-requisites	**NIL**					
Course Description	This course builds the foundational insight of understanding generative AI models and to explore various architectures, algorithms and practices of Gen AI skills to accelerate strategic decision making with data and deliver cutting-edge products faster with GenAI-augmented software development and leverage Gen AI tools to optimize workflows.					
Course Objective	**The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.**					
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1:** Infer the concepts of generative AI models and prompt engineering in tailoring customized outputs.**(Apply)** **CO2:** Demonstrate attention mechanism and transformers architecture with practical Applications **(Apply)** **CO3:** Practice advanced generative AI techniques using Langchain Python framework **(Apply)** **CO4:** Solve real-time applications using multi-modal generative AI models. **(Apply)** **CO5:** Apply prompt optimization techniques and chaining strategies to improve the quality and consistency of outputs generated by large language models. **(Apply)** **CO6:** Implement generative AI solutions using pre-trained models and APIs for tasks such as text generation, summarization, and conversational agents. **(Apply)**					
Course Content:						
Module 1	**Introduction to Generative AI-CO1**					**14 Sessions (L6+P8)**
Introduction to Generative models: Historical perspective and evolution, Applications, Types of Generative models for different data modalities, Large Language Models (LLMs) – Introduction, evolution, Generative pre-trained transformers (GPT) and its variants, Google DeepMind's, PaLM2, LLaMa and its series of models by Meta AI, Claud and its variants by Anthropic, Prompt Engineering-basic prompting						
Module 2	**Text-based Generative models-CO2**					**14 Sessions (L8 + P6)**
Text-based Generative models: State-of-the Art models, RNN, LSTM, Transformer Architecture, Transformer based Generative models: BERT, GPT, Training and Fine tuning LLMs for Generative task, Open AI's Pre-trained transformers for Text Generation: ChatGPTs, Limitations of LLMs: Lack of context and Hallucination risks, Techniques to mitigate these limitations: chaining and retrieval augmentation, Workflow of an LLM application.						
Module 3	**Introduction to Lang Chain – CO3 , CO4**					**16 Sessions (L8 + P8)**
Introduction to Lang chain: Types, Components, Information retrieval using agents and tools in Lang chain, Retrieval Augmented Language Models (RaLM): Understanding Retrieval and vectors: Embeddings, Vector storage, Vector indexing, Vector Libraries, Vector Databases, Chatbot using memory and conversation buffer						
Module 4	**Generative models for other Data modalities-CO5, CO6**					**16 Sessions (L8 + P8)**
Generative Adversarial Networks (GAN): GAN Architecture, GAN variants, Neural Style transfer with GAN, Training GANs and common challenges, GAN applications in image and text generation, Variational Auto Encoders (VAEs) and its variants, Image generation models: Dall-E, MidJourney and stable diffusion: Architecture and components of stable						

diffusion, Text-to-image Generation, Parameter tuning, Image-to-image generation, Training custom models, In-Painting: Exchanging classes, Multi-modal generative models using Whisper for Audio: Speech-to-Text generation.

List of Laboratory Tasks:

List of Laboratory Tasks:

Experiment No.1: Setting up Python IDE(Spyder) and OpenAI API key. Introduction to OpenAI playground and prompting
Level 1: Document the installation and the process for generating models in OpenAI
Level 2: Solve various GenAI models of OpenAI from Playground using prompts

Experiment No.2: Text classification, summarization, sentiment analysis, chatbot application, code explanation with generating single and multiple response(S).
Level 1: Practice the text generation model of OpenAI and Spyder IDE to implement various applications.

Experiment No.3: Embeddings – for words, similarity between words, text embeddings, plagiarism check of documents
Level 1: Use generating embeddings for words, text and documents
Level 2: Apply the embeddings API to develop applications for plagiarism check

Experiment No.4: Image generation using Dall E. Using GPT-Vision model for text to image generation and image-to-text
Level 1: Apply GPT-vision model for text-to-image generation and image-to-image

Experiment No.5: Transformer based text and email classification
Level 1: Develop transformer-based AI models for classifying text/email

Experiment No.6: BERT for masked token generation
Level 1: Develop BERT based model for generating masked tokens

Experiment No.7: Creating applications using different types of LangChains – Simple Sequential, Sequential and map reduce
Level 1: List the various types of chains in Langchain
Level 2: Practice different types of chains using Spyder IDE and OpenAI

Experiment No.8: Information retrieval using agents and tools in Langchain
Level 1: Use agents and tools with Langchain for information retrieval

Experiment No.9: Custom Document loading and retrieval in LangChain using ChromaDB
Level 1: Understand ChromeDb
Level 2: Apply chromed with Langchain to generate information retrieval model from custom document

Experiment No.10: Create a GPT like Chatbot using the memory component and RALM in LangChain
Level 1: Show GPT like chatbot using memory component and retrieval augmented language model

Experiment No.11: Using action agents, human as a tool and plan and execute agents for information retrieval.
Level 1: Understand action agents and plan and execute agents
Level 2: Use agents and tools for information retrieval

Experiment No.12: Implement GAN for neural style transfer
Level 1: Demonstrate a style transfer algorithm using generative models and experiment with the transformation of

images by applying different artistic styles, assessing both the technical aspects and the aesthetic outcomes

Experiment No.13: Text to Image generation using Dall-e/stable diffusion using prompts
Level 1: List various image generation models
Level 2: Use an image generation model to generate image from prompts

Experiment No.14: Image to Image generation using stable diffusion
Level 1: Apply stable diffusion to generate image from an image using prompts

Experiment No.15: Speech to text and multi-modal generative models using Whisper for Audio
Level 1: Identify the generative model for text, image and audio data
Level 2: Use Langchain to create models for generating different data modalities. Ex: Audio-to-text

Targeted Application & Tools that can be used:
Open AI Generative AI models: GPT 3.5 Turbo, GPT 4.0 vision model, Dall-E 3.0, Lang Chain Framework in Python, Python IDE, Stable Diffusion, Gemini, Hugging Face

Project work/Assignment:

1. Mini-Project Titles:
2. Conversational Chatbot that interacts with documents: create a conversational chatbot to engage users in meaningful dialogues, answer queries, offer recommendations, and aid tasks using provided documents as inputs.
3. Sentiment Analysis/Intent Analysis/Toxicity Analysis
4. Natural Language Translation – Instruction Tuning using FLAN (Finetuned language Net) model
5. Questions and Answering systems – Extractive & Generative
6. Text Summarization – Medicine – Med-PaLM
7. Given the Academic guidelines of the University, generate the student Handbook with FAQs and solutions.
8. Generating Cartoon based story telling
9. Simulate various driving conditions to improve safety and performance in Autonomous vehicles
10. In Financial management, generate synthetic financial data for stress testing and scenario analysis
11. Personalized recommendations/Product suggestions/tailored content based personalized design studio
12. Simulate characters for Games
13. Create conversational agents
14. Tutor in a range of preferred subjects
15. Generate codes
16. Draft documents
17. Answer questions about any knowledge base
18. Create an application which uses LangChain to connect OpenAI API to DALL-E. This image generation application turns written descriptions into lifelike pictures and artwork.
19. Embark on building a personalized language model with Falcon-7b. Utilize personalized LLM technique to explore text generation capabilities by providing task examples as inputs.
20. Use OpenAI's DALL-E and Gradio UI to develop an innovative logo builder. Th app creates unique and stunning logos from text prompts, revolutionizing the logo design process.
21. Crafting an AI powered HR Assistant: Develop a virtual assistant designed to answer queries related to Audi HR policy. Leverage Python libraries and OpenAI's GPT model for accurate and efficient query responses.

Topics related to
1. **Problem Solving**: Choose appropriate deep learning architectures (CNN, RNN, Autoencoders) for specific real-world data challenges such as image classification or sequence prediction.
2. **Employability**: Simulation of deep learning-based systems such as image classification, object detection, or time-series forecasting to develop industry-relevant skills.

Textbook(s):

T1. Generative AI with LangChain, Ist Edition by Ben Auffarth, Packt. Inc. ISBN: 978-1-83508-346-8, Decemeber 2023

T2. Generative Deep Learning,2nd Edition by David Foster, O'Reilly Media, Inc. ISBN: 9781098134181, May 2023.

T3. Prompt Engineering for Generative AI, by James Phoenix, Mike Taylor, O'Reilly Media, Inc., ISBN:9781098153373, July 2024

References

R1. Bandi, A., Adapa, P. V. S. R., & Kuchi, Y. E. V. P. K. (2023). The power of Generative AI: a review of requirements, models, Input–Output formats, evaluation metrics, and challenges. Future Internet, 15(8), 260. https://doi.org/10.3390/fi15080260

R2. Barachini, F., & Stary, C. (2022). From digital twins to digital selves and beyond. In Springer eBooks. https://doi.org/10.1007/978-3-030-96412-2

R3. Hadi, M. U., Tashi, Q. A., Qureshi, R., Shah, A., Muneer, A., Irfan, M., Zafar, A., Shaikh, M. B., Akhtar, N., Wu, J., & Mirjalili, R4. S. (2023). Large Language Models: A Comprehensive Survey of its Applications, Challenges, Limitations, and Future Prospects. https://doi.org/10.36227/techrxiv.23589741.v4

R4. Hai-Jew, S. (n.d.). Generative AI in Teaching and Learning. IGI Global.

R5. Salvaris, M., Dean, D., & Tok, W. H. (2018). Generative adversarial networks. In Apress eBooks (pp. 187–208). https://doi.org/10.1007/978-1-4842-3679-6_8

E-Resources:

W1. https://openai.com

W2: https://python.langchain.com/v0.2/docs/introduction/

W3: https://www.udemy.com/course/master-ai-image-generation-using-stable-diffusion/?kw=Image+generation+using&src=sac&couponCode=LETSLEARNNOWPP

W4: https://huggingface.co/google-t5/t5-base

W5: https://dominguezdaniel.medium.com/exploring-image-generative-ai-models-9359705b15d3

W6: https://cloud.google.com/use-cases/retrieval-augmented-generation?hl=en#

W7: https://ig.ft.com/generative-ai/

W8: https://medium.com/@samia.khalid/bert-explained-a-complete-guide-with-theory-and-tutorial-3ac9ebc8fa7c

MOOC's/Swayam Courses/Online Courses:

h https://onlinecourses.swayam2.ac.in/imb24_mg116/preview

Global Certification Course by Google :

1. https://www.cloudskillsboost.google
   a. Introduction to Generative AI (Beginner)
   b. Gemini for Google Cloud (Intermediate)
   c. Generative AI for Developers (Advanced)

2. https://www.credly.com/badges/90e3eae0-87f3-44e3-af82-658e837aad3d/public_url

3. https://www.coursera.org/learn/generative-ai-with-llms

4. https://www.coursera.org/specializations/prompt-engineering

5.

Cyber Security Stream

Course Code: CSA4711	**Course Title:** Cyber Security and Ethical Hacking **Type of Course:** Discipline Elective	**L-T- P- C**	3	0	0	3
Version No.	1.1					
Course Pre-requisites	**Basic knowledge of Computer Networks, Operating Systems, and Programming Concepts**					
Anti-requisites	**NIL**					
Course Description	This course offers a comprehensive introduction to cyber security and ethical hacking. It addresses core concepts in security architecture, threat analysis, penetration testing, and system defense mechanisms. Students will learn how to ethically identify vulnerabilities in systems and mitigate them using appropriate tools and techniques, guided by industry best practices and legal frameworks.					
Course Objective	**The objective of the course is to familiarize the learners with the concepts of principles of cyber security and the practical skills and attain SKILL DEVELOPMENT through PARTICIPATIVE LEARNING techniques.**					
Course Outcomes	**On successful completion of this course the students shall be able to:** **CO1:**Understand and explain the fundamental concepts of cyber security, threats, vulnerabilities, and risk management. **[Understand]** **CO2:** Apply techniques to secure networks and systems using tools such as firewalls, intrusion detection systems, and endpoint security solutions. **[Apply]** **CO3;** Apply identity and access management practices, including cryptographic methods, to enforce secure authentication and data confidentiality. **[Apply]** **CO4:** Apply ethical hacking methodologies to identify vulnerabilities and simulate cyber-attacks in a controlled environment. **[Apply]** **CO5:** Analyze ethical hacking tools and techniques such as reconnaissance, scanning, and exploitation to identify potential security weaknesses**. [Analyze]** (Module 3 related) **CO6:** Apply vulnerability assessment and penetration testing methodologies to evaluate system security and recommend appropriate countermeasures. **[Apply]** (Module 4 related)					
Course Content:						
Module 1	**Introduction to Cyber Security**					**12 Sessions**
Introduction to Cyber Security : Concept of Cyber Security, Types of Cyber Attacks, Threats and Vulnerabilities, Security Goals (CIA Triad), Information Assurance, Security Policies, Risk Management, and Cyber Security Standards (ISO 27001, NIST.						
Module 2	**System and Network Security**					**12 Sessions**
Operating System Security, File Permissions and Access Control, Malware (Viruses, Worms, Trojans), Network Protocols (TCP/IP, DNS, HTTP), Firewalls, Intrusion Detection/Prevention Systems (IDS/IPS), VPNs, Wireless Security (WPA2, WPA3), Network Scanning Tools (Nmap, Wireshark)						
Module 3	**Ethical Hacking Tools and Techniques**					**10 Sessions**
Phases of Ethical Hacking: Reconnaissance, Scanning, Gaining Access, Maintaining Access, Clearing Tracks. Password Cracking, Social Engineering, SQL Injection, XSS, Buffer Overflow, Metasploit Framework, Kali Linux tools (Aircrack-ng, John the Ripper), Enumeration Techniques						
Module 4	**Penetration Testing and Countermeasures**					**11 Sessions**
Penetration Testing Methodology, Vulnerability Assessment, Reporting and Documentation, Web Application Security, OWASP Top 10, Security Audit, Countermeasures, Patch Management, Incident Response, Digital Forensics Overview						
Project work/Assignment:						

1. **Identify different types of cyber-attacks (Phishing, DoS, Malware) and explain their impact with real-world examples.**
2. **Analyze CIA Triad (Confidentiality, Integrity, Availability) using a case study of an online banking system.**
3. **Demonstrate file permissions and access control mechanisms in an operating system environment.**
4. **Perform basic network scanning using tools like Nmap and interpret open ports and services.**
5. **Compare different malware types (Virus, Worm, Trojan, Ransomware) with attack flow diagrams.**
6. **Explain phases of Ethical Hacking with a practical example of website security testing.**
7. **Demonstrate password cracking concepts using dictionary attack and brute force (conceptual/simulated).**
8. **Study OWASP Top 10 vulnerabilities and explain any three with examples (SQL Injection, XSS, CSRF).**
9. **Prepare a vulnerability assessment report for a sample web application (theoretical).**
10. **Project: Network Security Implementation using Firewall, IDS/IPS, and VPN concepts.**
11. **Project: Ethical Hacking Methodology and Penetration Testing Report for a Sample System.**
12. **Project: Web Application Security and OWASP Top 10 Risk Mitigation Strategy.**

Topics related to

1.Problem Solving: Apply cyber security principles to identify threats, analyze vulnerabilities, perform basic penetration testing, and recommend countermeasures to protect systems, networks, and web applications from real-world cyber-attacks.

2.Employability: Hands-on exposure to ethical hacking tools, network scanning, vulnerability assessment, malware analysis, firewall configuration, and security auditing to develop practical skills for careers in cyber security, ethical hacking, penetration testing, and security analysis.

Textbook(s):

T1. Charles J. Brooks, Christopher Grow, Philip Craig, Donald Short, "Cybersecurity Essentials," 2nd Edition, Jones & Bartlett Learning, 2022 [ISBN: 9781284235667]

T2. Craig S. Wright, "Security Management Practices: A Step-by-Step Guide", 1st Edition, Auerbach Publications, USA, 2021. [ISBN-978-1032092223].

T3. Erdal Ozkaya, Cybersecurity – The Beginners Guide: A comprehensive guide to getting started in cybersecurity, 2nd Edition, Packt Publishing, 2023. [ISBN: 9781804616436]

References

R1. Ric Messier, Certified Ethical Hacker (CEH) v12 Study Guide, Wiley, 2023. [ISBN: 9781394162325]

R2. Jon DiMaggio, The Art of Cyberwarfare: An Investigator's Guide to Espionage, Ransomware, and Organized Cybercrime, No Starch Press, 2022. [ISBN: 9781718502147]

R3. Daniel G. Graham, Practical Ethical Hacking: The Hands-On Guide to Breaking In, No Starch Press, 2022. [ISBN: 9781718502192].

E-Resources:

1. https://www.jblearning.com/catalog/productdetails/9781284235667
2. https://nptel.ac.in/courses/106105031
3. https://www.cybrary.it/
4. https://www.hackthebox.com/
5. https://owasp.org/www-project-top-ten/

Course Code: CSA4712	**Course Title:** Web Application Security **Type of Course:** Discipline Elective	**L-T- P- C**	2	0	2	3
Version No.	1.0					

Course Pre-requisites	**Web Technology**	
Anti-requisites	**NIL**	
Course Description	This course introduces foundational and advanced concepts in web application and API security. It explores real-world threats, secure coding practices, encryption techniques, and vulnerability assessments. Learners will engage with hands-on assignments and projects aimed at building secure systems and defending against evolving cyber threats.	
Course Objective	The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.	
Course Outcomes	**On successful completion of the course, the students shall be able to:** **CO1:** Analyze the roles of authentication, authorization, session management, and input validation in securing web applications, and examine their interdependencies in mitigating common vulnerabilities. **(Analyze)** **CO2:** Apply secure development lifecycle models such as SDL and CLASP to design and implement secure web application architectures. **(Apply)** **CO3:** Apply security mechanisms including OAuth2, encryption, and service mesh for securing APIs in microservices and IoT-based systems. **(Apply)** **CO4:** Apply vulnerability assessment techniques such as scanning and penetration testing on platforms like web, wireless, and mobile to identify and address security flaws. **(Apply)** **CO5:** Design and justify security design decisions, threat models, and mitigation strategies through technical reports and oral presentations. **(Analyze)** **CO6:** Design and develop a secure web or API-based application as a course project by integrating authentication, authorization, encryption, and vulnerability mitigation techniques. **(Apply)**	
Course Content:		
Module 1	**Foundations of Web Application Security - CO1**	**16 Sessions (8T + 8P)**
The history of Software Security-Recognizing Web Application Security Threats, Web Application Security, Authentication and Authorization, Secure Socket layer, Transport layer Security, Session Management-Input Validation		
Module 2	**Secure Software Development Lifecycle (SDLC) – CO2**	**16 Sessions (8T + 8P)**
Web Applications Security - Security Testing, Security Incident Response Planning, The Microsoft Security Development Lifecycle (SDL), OWASP Comprehensive Lightweight Application Security Process (CLASP), The Software Assurance Maturity Model (SAMM)		
Module 3	**API and Micro services Security – CO3**	**16 Sessions (8T + 8P)**
Encryption, Audit logging, securing service-to-service APIs: API Keys, OAuth2, Securing Micro service APIs: Service Mesh, Locking Down Network Connections, Securing Incoming Requests.		

Module 4	Vulnerability Assessment and Penetration Testing – CO4	12 Sessions (6T + 6P)

Network-based vulnerability scanners, Database based vulnerability scanners, Types of Penetration Tests: External Testing, Web Application Testing, Internal Penetration Testing, SSID or Wireless Testing, Mobile Application Testing.

Lab Sheet 1: Cybersecurity Lab Orientation & Network Protocol Basics

Level 1:
Introduction to cybersecurity lab, lab safety, ethical guidelines, and overview of security tools. Install and configure Wireshark.

Level 2:
Capture live traffic and identify TCP, HTTP, and DNS packets. Analyze the role of each protocol in web communication.

Lab Sheet 2: TCP Communication Analysis

Level 1:
Capture TCP packets and identify SYN, SYN-ACK, and ACK packets involved in the TCP three-way handshake.

Level 2:
Analyze sequence numbers, acknowledgments, and packet retransmissions to explain TCP reliability.

Lab Sheet 3: Secure vs Non-Secure Web Traffic

Level 1:
Capture HTTP and HTTPS traffic using Wireshark and observe packet structures.

Level 2:
Compare encrypted and unencrypted payloads and explain why HTTPS ensures confidentiality and integrity.

Lab Sheet 4: SSL/TLS and Digital Certificates

Level 1:
Inspect SSL/TLS handshake messages and view server certificates using browser developer tools.

Level 2:
Analyze certificate chains, validity, and the role of Certificate Authorities (CA).

Lab Sheet 5: Authentication and Session Management

Level 1:
Analyze cookies and session tokens generated during user authentication.

Level 2:
Evaluate session management weaknesses such as session fixation and hijacking risks.

Lab Sheet 6: Web Input Validation

Level 1:
Test web application input fields with invalid and unexpected inputs.

Level 2:
Identify missing input validation controls and discuss associated vulnerabilities.

Lab Sheet 7: OWASP ZAP Tool Familiarization

Level 1:
Install OWASP ZAP and configure it as a proxy for browser-based testing.

Level 2:
Explore spidering and passive scanning features of OWASP ZAP.

Lab Sheet 8: OWASP Top 10 Vulnerability Identification

Level 1:
Perform automated vulnerability scanning on a test web application using OWASP ZAP.

Level 2:
Analyze the scan report and map identified issues to OWASP Top 10 categories.

Lab Sheet 10: REST API Development – Basics

Level 1:
Develop a REST API using Python (Flask/FastAPI) implementing GET and POST methods.

Level 2:
Validate request payloads and analyze HTTP status codes returned by the API.

Lab Sheet 10: REST API Development – CRUD Operations

Level 1:
Extend the REST API to include PUT and DELETE operations.

Level 2:
Test all CRUD operations using Postman and analyze API behavior for invalid inputs.

Lab Sheet 11: API Authentication using API Keys

Level 1:
Implement API key–based authentication for REST endpoints.

Level 2:
Analyze unauthorized access attempts and discuss limitations of API key security.

Lab Sheet 12: OAuth2-Based API Security

Level 1:
Implement OAuth2 authentication to protect REST API endpoints.

Level 2:
Analyze access tokens and refresh tokens and explain their lifecycle.

Lab Sheet 13: Encryption for Secure API Communication

Level 1:
Implement AES or RSA encryption for sensitive API data.

Level 2:
Analyze encrypted traffic and explain how encryption ensures confidentiality and integrity.

Lab Sheet 14: Web Application Vulnerability Exploitation

Level 1:

Install and configure Burp Suite and intercept HTTP requests.

Level 2:
Detect and exploit SQL Injection and XSS vulnerabilities using Burp Suite in a controlled environment.

Lab Sheet 15: Advanced Security Assessment & Mini Project

Level 1:
Perform network vulnerability scanning (Nmap/Nessus), wireless SSID analysis, and social engineering awareness simulation.

Level 2:
Design, develop, and present a secure web/API-based application integrating authentication, authorization, encryption and vulnerability mitigation.

Targeted Application & Tools that can be used:

AI-driven Intrusion Detection Systems, Fraud Detection Engines, Secure Email Filtering, Cyber Threat Intelligence Platforms using tools such as Python, Scikit-learn, TensorFlow, Keras, OpenCV, Wireshark, Splunk, and Jupyter Notebook.

Project work/Assignment:

1. **AI-Powered Intrusion Detection System**
2. **Phishing Email Detection Using NLP**
3. **Fraudulent Transaction Detector**
4. **Adversarial Attack Simulation**
5. **Anomaly Detection in IoT Devices**

Topics related to

1. Problem Solving: Designing and implementing AI models for real-time cybersecurity threat detection and response.
2. Employability: Simulation of AI-driven intrusion detection systems and fraud detection tools using machine learning and NLP techniques.

Textbook(s):

T1. Andrew Hoffman, Web Application Security: Exploitation and Countermeasures for Modern

Web Applications, First Edition, 2020, O'Reilly Media, Inc.

T2. Bryan Sullivan, Vincent Liu, Web Application Security: A Beginners Guide, 2012, The McGraw-Hill Companies

References

R1: Neil Madden, API Security in Action, 2020, Manning Publications Co., NY, USA.

R2: Singh, Sri N. Electric power generation: transmission and distribution. PHI Learning Pvt. Ltd.,

2008.

R3: IS 4023:1996, Recommendations for Testing of Three-Phase Induction Motors, Bureau of

Indian Standards, New Delhi.

R4: IEEE Std 3002.7-2018, IEEE Recommended Practice for Conducting Field Surveys of

Industrial and Commercial Power Systems, IEEE Standards Association.

E-Resources:

1.https://www.coursera.org/projects/web-application-security-testing-with-owsap-zap

2. https://www.coursera.org/learn/web-application-security

Course Code: **CSA4713**	**Course Title:** **Cybersecurity Testing** **Type of Course:** **Discipline Elective**	**L-T- P- C**	3	0	0	3
Version No.	1.0					
Course Pre-requisites	**Students should have basic knowledge of Python programming, cybersecurity fundamentals, machine learning, and network security. Familiarity with ML frameworks like TensorFlow and scikit-learn, along with networking concepts such as TCP/IP and IDS/IPS, is recommended. A solid understanding of mathematics, particularly statistics, probability, and linear algebra, is essential. This background will help students engage effectively with cybersecurity testing. A foundation in Computer Science, IT, or Cybersecurity will be beneficial.**					
Anti-requisites	**NIL**					
Course Description	The Cybersecurity Testing course provides a comprehensive overview of methods and tools used to assess and enhance the security of systems, networks, and applications. It focuses on testing techniques like penetration testing, vulnerability scanning, and security auditing to identify potential risks. The course covers automated testing tools, manual testing methodologies, and compliance with security standards. Students will gain hands-on experience with real-world cybersecurity testing scenarios, learning how to detect and mitigate threats effectively while ensuring system robustness and data protection.					
Course Objective	**To understand the principles and methodologies of cybersecurity testing.** **To explore various testing techniques, including penetration testing, vulnerability assessment, and security auditing.** **To apply automated tools and manual methods for identifying and mitigating security vulnerabilities.** **To ensure compliance with industry security standards and best practices during testing.** **To provide hands-on experience with real-world scenarios for testing and securing systems, networks, and applications.**					
Course Outcomes	**CO1:** Explain the importance of cybersecurity testing methods in identifying vulnerabilities and improving security. (Understand, Describe, Identify) **CO2:** Implement cybersecurity testing techniques such as penetration testing, vulnerability scanning, and security auditing. (Apply, Execute, Perform) **CO3**: Analyze test results to evaluate system security and identify potential threats. (Analyze, Evaluate, Assess) **CO4:** Develop and deploy mitigation strategies based on test outcomes to enhance overall security. (Develop, Deploy, Optimize) **CO5:** Evaluate web application security using OWASP Top 10 vulnerabilities and recommend secure development practices. (Evaluate, Recommend) — Module 3 **CO6:** Apply cybersecurity governance, risk management, and compliance frameworks to implement secure organizational policies and incident response strategies. (Apply, Implement) — Module 4					
Course Content:						
Module 1	**Introduction to Cybersecurity**					**10 Sessions**
Cybersecurity Fundamentals, Cyber Threats & Attack Types, Security Principles, Authentication & Access Control, Encryption Basics, Security in Network & Systems, Emerging Trends in Cybersecurity						
Module 2	**Security Testing Methodologies**					**12 Sessions**
Introduction to Security Testing, Types of Security Testing, Threat Modeling & Risk Assessment, Security Testing Tools Overview, Application Security Testing, API Security & Mobile App Testing, Cloud Security Testing, Case Study on Security						

Breaches & Testing Approaches.		
Module 3	**OWASP and Web Security**	**12 Sessions**
Introduction to OWASP, OWASP Top 10 Vulnerabilities, Web Security Testing Methodologies, Best Practices for Secure Web Development, Industry Case Studies on Web Security Breaches.		
Module 4	**Security Implementation & Best Practices**	**11 Sessions**
Introduction to Risk Management in Cybersecurity, Cybersecurity Governance & Policies, Risk Assessment Frameworks, Security Compliance & Regulations, Incident Response & Disaster Recovery Planning, Security Audits & Vulnerability Management, Case Studies on Cyber Risk Management Failures		

Project work/Assignment:

7. **To understand the application of cybersecurity Testing in daily lives the following assignments, Quizzes and Tests are included:**
8. **Assignment: 1] Module 3**
9. **Assignment: 2] Module 4**

Topics related to

1. Problem Solving: Apply cybersecurity testing methodologies such as threat modelling, vulnerability scanning, OWASP-based web testing, and risk assessment frameworks to identify security weaknesses and recommend mitigation strategies for real-world systems, networks, and applications.

2. Employability: Hands-on exposure to security testing tools, web vulnerability assessment, API testing, cloud security evaluation, compliance auditing, and incident response planning to develop industry-relevant skills for careers in penetration testing, security analysis, vulnerability assessment, and cybersecurity auditing.

Textbook(s):

1] Kutub Thakur, Al-Sakib Khan Pathan, Cybersecurity Fundamentals: A Real-World Perspective, 2020.
2] Bryan Sullivan, Vincent Liu, Web Application Security: A Beginner's Guide, Updated Edition, 2024.
3] Krag Brotby, Information Security Governance: A Practical Development and Implementation Approach, Updated Edition, 2024.

References

R1. Ric Messier, Certified Ethical Hacker (CEH) v12 Study Guide, Wiley, 2023.
R2. Daniel G. Graham, Practical Ethical Hacking: The Hands-On Guide to Breaking In, No Starch Press, 2022.
R3. Kim Crawley, 8 Steps to Better Security: A Simple Cyber Resilience Guide for Business, Wiley, 2021.

E-Resources:

6. https://youtu.be/3DZLItfbqtQ
7. https://www.geeksforgeeks.org/last-minute-notes-computer-network/
8. https://owasp.org/www-project-webgoat/
9. https://tryhackme.com/

Course Code: **CSA4714**	**Course Title:** **Cloud Security** **Type of Course:** **Discipline Elective**	**L-T- P- C**	2	0	2	3
Version No.	1.0					
Course Pre-requisites	**Cloud Computing**					
Anti-requisites	**NIL**					

Course Description	The Cloud Security course provides an in-depth understanding of the principles, techniques, and best practices for securing cloud environments. The course covers a broad spectrum of security challenges associated with cloud computing, including data security, network security, application security, and infrastructure security. Students will learn to design, implement, and manage security solutions within public, private, and hybrid cloud environments, focusing on real-world scenarios and industry standards. The course emphasizes security frameworks such as NIST, ISO 27001, and OWASP, along with regulatory compliance including GDPR, HIPAA, and PCI DSS.	
Course Objective	**The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.**	
Course Outcomes	**On successful completion of this course, the students shall be able to:** CO1: Apply encryption methods, including homomorphic encryption and Public Key Infrastructure to secure cloud data. (Apply) CO2: Develop secure virtualization environments, including hypervisor security and virtual machine (VM) isolation (Apply) CO3: Implement Security as a Service (SECaaS) solutions in cloud applications (Apply) CO4: Analyze security threats and intrusion detection in cloud environments (Apply) CO5: Implement identity and access management, logging, and monitoring mechanisms to secure cloud infrastructure and applications. (Apply) — Module 3 CO6: Evaluate cloud security standards and compliance frameworks such as NIST, ISO 27001, and PCI DSS for secure cloud deployment. (Evaluate) — Module 4	
Course Content:	The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.	
Module 1	**Cloud Security Fundamentals- CO1**	**14 Sessions (6T + 8P)**
Cloud Security Fundamentals Cloud computing security challenges – cloud computing security architecture – data security life-cycle - Security Patterns and architectural elements - Planning key Strategies for secure operation. Cloud Application Security Encryption techniques – homomorphic encryption - securing data Redaction - secure bitcoin – Public key infrastructure (PKI) – key management - open web application security project (OWASP) Cloud Top 10 Security Risks - Security as a service (SECaaS)		
Module 2	**Cloud Infrastructure Security – CO2**	**14 Sessions (8T + 6P)**
Security Management & Privacy Managed Security Service Provider (MSSP): Availability management – configuration management - vulnerability management - identity management. - Privacy: privacy, compliance and the cloud - privacy enhancing encryption. Risk Management & Security Threats Risk management – principles - assessing the risk – strategies for managing risk – risk analysis framework – security threats - intrusion detection		
Module 3	**Security Management – CO3**	**16 Sessions (8T + 8P)**
Security Management & Privacy Managed Security Service Provider (MSSP): Availability management – configuration management - vulnerability management - identity management. - Privacy: privacy, compliance and the cloud - privacy enhancing encryption. Risk Management & Security Threats Risk management – principles - assessing the risk – strategies for managing risk – risk analysis framework – security threats - intrusion detection		
Module 4	**Cloud Standards and Compliance – CO4**	**16 Sessions (8T + 8P)**
Cloud Standards and Compliance Cloud security alliance – cloud controls matrix - cloud security standards guidance – security compliance - NIST – PCI data security standards – SAS 70 - ISO 27001 – HIPAA – ITIL - FISMA - FIPS 140- 2. Cloud-Based IT Audit Process – System and Infrastructure lifecycle management for the cloud - governance, risk management and compliance (GRC)		
List of Laboratory Tasks: **Experiment No. 1:**		

Level 1: Set up a cloud environment using AWS, Azure, or Google Cloud Platform (GCP).

Level 2: Implement basic security configurations, such as firewalls, access controls, and encryption.

Experiment No. 2:

Level 1: Store and retrieve data securely using cloud storage services.

Level 2: Apply encryption techniques like AES, RSA, and homomorphic encryption to secure data at rest and in transit

Experiment No. 3:

Level 1: Install and configure a hypervisor (e.g., VMware, Hyper-V, or KVM).

Level 2: Secure virtual machines (VMs) and ensure isolation between VMs using secure configurations.

Experiment No. 4:

Level 1: Design a secure virtual network within a cloud environment

Level 2: Implement network security measures, including security groups, network access control lists (NACLs), and virtual private clouds (VPCs).

Experiment No. 5:

Level 1: Develop a simple web application and deploy it to the cloud

Level 2: Identify and mitigate security vulnerabilities using the OWASP Cloud Top 10 Security Risks.

Experiment No. 6:

Level 1: Configure IAM policies and roles to manage user access in the cloud

Level 2: Implement multi-factor authentication (MFA) and single sign-on (SSO) for secure access.

Experiment No. 7:

Level 1: Create and configure a virtual private network (VPN) within the cloud.

Level 2: Secure data transmission between on-premises and cloud environments using VPNs.

Experiment No. 8:

Level 1: Conduct vulnerability scanning on cloud infrastructure using tools like Nessus or OpenVAS.

Level 2: Remediate identified vulnerabilities and enhance security posture.

Experiment No. 9:

Level 1: Set up a cloud-based intrusion detection and prevention system (IDPS).

Level 2: Analyze and respond to security incidents using IDPS tools.

Experiment No. 10:

Level 1: Set up and configure cloud monitoring tools (e.g., AWS CloudWatch, Azure Monitor).

Level 2: Analyse logs and generate security reports for cloud environments

Experiment No. 11:

Level 1: Set up a cloud environment for IaaS, PaaS, and SaaS models and describe the security responsibilities of cloud providers and customers.
Level 2: Implement security measures in an IaaS setup on AWS/Azure/Google Cloud and simulate unauthorized access to verify mitigation.

Experiment No. 12:

Level 1: Enable data encryption at rest for a file uploaded to cloud storage and verify encryption.
Level 2: Implement encryption in transit using SSL/TLS while transferring data to/from cloud storage.

Experiment No. 13:

Level 1: Set up a Virtual Private Network (VPN) in the cloud and test the secure communication between on-premises and cloud resources.

Level 2: Configure firewall rules and network security groups to restrict access based on IP and test unauthorized access attempts.

Experiment No. 14:

Level 1: Enable logging and monitoring for cloud resources and track activity using cloud-native tools (e.g., AWS CloudTrail, Azure Monitor).

Level 2: Set up an automated alert system for suspicious activities and test it by simulating security incidents.

Experiment No. 15:

Level 1: Configure backup and disaster recovery for cloud resources using cloud-native services (AWS Backup / Azure Backup / GCP Backup) and verify data restoration.

Level 2: Implement high availability and failover mechanisms using load balancers and multi-region deployment, and test system resilience during simulated service failure.

Targeted Application & Tools that can be used:

AI-driven Intrusion Detection Systems, Fraud Detection Engines, Secure Email Filtering, Cyber Threat Intelligence Platforms using tools such as Python, Scikit-learn, TensorFlow, Keras, OpenCV, Wireshark, Splunk, and Jupyter Notebook.

Project work/Assignment:

1. **AI-Powered Intrusion Detection System**
2. **Phishing Email Detection Using NLP**
3. **Fraudulent Transaction Detector**
4. **Adversarial Attack Simulation**
5. **Anomaly Detection in IoT Devices**

Topics related to

1. Problem Solving: Designing and implementing AI models for real-time cybersecurity threat detection and response.

2. Employability: Simulation of AI-driven intrusion detection systems and fraud detection tools using machine learning and NLP techniques.

Textbook(s):

T1. John R. Vacca, "Cloud Computing Security: Foundations and Challenges", 1st Edition, CRC Press, USA, 2020. [ISBN-978-0367331656].

T2. Craig S. Wright, "Security Management Practices: A Step-by-Step Guide", 1st Edition, Auerbach Publications, USA, 2021. [ISBN-978-1032092223].

T3: Cloud Security Alliance (CSA), "Cloud Computing Compliance Controls Catalog (C5): A Compendium of Cloud Provider Requirements", 1st Edition, CSA, USA, 2021. [ISBN-978-0989567009]

References

R1. Kim Crawley, 8 Steps to Better Security: A Simple Cyber Resilience Guide for Business, Wiley, 2021.

R2. Brien Posey, Microsoft Azure Security Technologies (Exam AZ-500) Guide, Packt Publishing, 2022.

R3. Ashish Rajan, Cloud Security Cookbook: Practical Solutions for Securing Cloud Environments, O'Reilly Media, 2021.

E-Resources:

1. https://cloudsecurityalliance.org
2. https://owasp.org/www-project-cloud-security
3. https://www.nist.gov/topics/cloud-computing
4. https://www.vmware.com/security.html
5. https://learn.microsoft.com/en-us/security/azure

Course Code: CSA4715	**Course Title:** AI in Cyber Security **Type of Course:** Discipline Elective	**L-T- P- C**	2	0	2	3
Version No.	1.0					
Course Pre-requisites	**Students should have basic knowledge of Python programming, cybersecurity fundamentals, machine learning, and network security. Familiarity with ML frameworks (TensorFlow, scikit-learn), networking concepts (TCP/IP, IDS/IPS), and mathematics (statistics, probability, linear algebra) is recommended. A background in Computer Science, IT, or Cybersecurity will be beneficial.**					
Anti-requisites	**NIL**					
Course Description	The AI in Cybersecurity course explores how artificial intelligence enhances cybersecurity by detecting threats, preventing cyberattacks, and automating security responses. It covers AI-driven malware detection, intrusion detection systems, and secure AI implementation. Students will gain practical experience using AI models for security applications.					
Course Objective	The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques. 1. To understand the role of AI in modern cybersecurity. 2. To explore AI techniques for threat detection and security automation. 3. To apply machine learning for cybersecurity applications like malware detection and fraud prevention. 4. To implement and evaluate AI-based security solutions.					
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1:** Explain AI applications in cybersecurity and security automation. *(Understand, Describe, Identify)* **CO2:** Implement AI techniques for detecting and preventing cyber threats. *(Apply, Develop, Construct)* **CO3:** Analyze AI-driven cybersecurity models for fraud detection and network security. *(Analyze, Evaluate, Visualize)* **CO4:** Secure AI systems and optimize cybersecurity solutions. *(Secure, Deploy, Optimize)* ***CO5****: Develop AI-based intrusion detection and anomaly detection models using machine learning and deep learning techniques for network security. (Develop) — Module 2* ***CO6:*** *Evaluate adversarial attacks and implement secure AI model deployment techniques for robust cybersecurity systems. (Evaluate, Implement) — Module 4*					
Course Content:	The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.					
Module 1	**Introduction to AI in Cybersecurity**					**16 Sessions (8T + 8P)**
Cybersecurity Fundamentals: Overview of cybersecurity concepts, threat landscapes, vulnerabilities, and risk management. Discussion of common attack vectors (e.g., phishing, DDoS, ransomware) and defensive strategies. AI and Machine Learning Basics: Introduction to machine learning algorithms (supervised, unsupervised, reinforcement learning) used in security applications.Overview of deep learning, neural networks, and decision trees as applied to cybersecurity tasks.						
Module 2	**Cloud Infrastructure Security – CO2**					**16 Sessions (8T + 8P)**
Intrusion Detection and Prevention Systems (IDS/IPS): Detailed study of IDS/IPS architectures and how AI algorithms improve detection rates, Comparison of signature-based versus anomaly-based detection systems. Deep Learning for Network Traffic Analysis: Use of deep neural networks, auto encoders, and recurrent neural networks (RNNs) to model network behaviour, Techniques for feature extraction and dimensionality reduction in large network datasets.						

Module 3	Security Management – CO3	16 Sessions (8T + 8P)
Financial Fraud Detection: Analysis of transactional data using classification and clustering techniques to uncover fraudulent patterns. Study of real-world case studies such as credit card fraud, insurance fraud, and identity theft. Phishing Detection and Email Security: Application of Natural Language Processing (NLP) and machine learning to analyse email content, URLs, and sender behaviour. Techniques for training models to differentiate between legitimate and phishing emails.		
Module 4	**Cloud Standards and Compliance – CO4**	**12 Sessions (6T + 6P)**
Adversarial Machine Learning: In-depth analysis of adversarial attacks such as evasion attacks, data poisoning, and model inversion. Strategies for designing robust AI models that can resist adversarial inputs. Securing AI Systems: Best practices for securing the training, deployment, and maintenance of AI models used in cybersecurity. Techniques for continuous monitoring and updating of AI models to prevent exploitation.		

List of Laboratory Tasks:

1. Packet Filtering Firewall (Scapy & Netfilterqueue):

Leve;-1 Write a Python script to capture all incoming TCP packets and print their source IP addresses and destination ports.

Level-2 :- Write a Python script to capture all incoming UDP packets and print their source IP addresses, destination IP addresses, and destination ports in real-time.

2.Phishing URL Detection (Machine Learning):

Level-1: Basic Phishing URL Classifier using Decision Tree.

Level-2: Advanced Phishing URL Classifier using Random Forest + Feature Engineering.

3.Intrusion Detection System (IDS) Using K-Means Clustering:

Level-1:- Design an Intrusion Detection System (IDS) that uses K-Means clustering to classify network traffic as either normal or suspicious based on features like packet size, packet rate, and duration. Evaluate the performance of the model by calculating its accuracy and plotting the clustered traffic.

Level-2:- Develop an anomaly-based IDS using K-Means clustering to detect suspicious network traffic. Implement a method to compute anomaly scores based on the distance from the cluster center, and classify traffic as normal or anomalous using a defined threshold.

4.Static Malware Detection Using Decision Trees:

Level-1: - Write a Python program that uses a Decision Tree Classifier to classify PE (Portable Executable) files as either malware or benign based on static features such as file size, entropy, and number of imports.

Level 2: - Develop a more advanced Static Malware Detection System using a Decision Tree Classifier. The system should analyze additional features such as section headers, file metadata, and imports, and classify files based on these features. Implement a function to visualize the decision tree and interpret how specific features contribute to the classification.

5.Keylogger Simulation for Ethical Hacking Awareness:

Level-1: - Write a basic Python program to simulate a keylogger that records keystrokes and saves them to a text file. The program should run in the background and log each keystroke until the user manually stops it.

Level-2: - Enhance the keylogger simulation by adding features for data encryption and stealth operation. The keylogger should encrypt the keystroke data before saving it to a file and should hide its process from the task

manager to prevent detection.

6.Spam and Phishing Email Detection Using NLP:

Level-1: Write a Python program to build a spam email detection model using NLP techniques such as text preprocessing, TF-IDF, and Support Vector Machine (SVM).

Level-2: Build an advanced spam and phishing email detection system using Natural Language Processing (NLP) and deep learning techniques. The system should classify emails as spam, phishing, or legitimate based on email content and metadata.

7.Deep Learning for Network Anomaly Detection: Train a Recurrent Neural Network (RNN) :

Level-1: Develop a Recurrent Neural Network (RNN) for network anomaly detection using Keras. The model should classify network traffic as normal or anomalous based on features like packet size, protocol type, and duration.

Level-2: Enhance the network anomaly detection system by integrating autoencoders for unsupervised learning. Use a deep learning model (RNN + Autoencoder) to detect anomalies in network traffic data and reconstruct the traffic patterns.

8.Autoencoder for Cyber Threat Detection:

Level- 1: Implement an Autoencoder model for cyber threat detection in network traffic. The model should be trained on normal network traffic and detect anomalies that could indicate potential cyber threats based on reconstruction errors.

Level-2: Develop an advanced cyber threat detection system using an Autoencoder with LSTM layers for detecting anomalies in time-series network traffic. The system should effectively distinguish between normal and anomalous traffic patterns and detect sophisticated cyber threats.

9.Credit Card Fraud Detection Using AI:

Level-1: - Build a Credit Card Fraud Detection System using XGBoost to classify transactions as fraudulent or legitimate based on features like transaction amount, time of transaction, and user behaviour.

Level-2: - Enhance the Credit Card Fraud Detection System by using ensemble methods (e.g., Random Forests or Stacking) and incorporate time-series analysis for transaction behavior patterns to improve fraud detection accuracy.

10.AI-Based Password Strength Classifier:

Level-1 :- Build a Password Strength Classifier using machine learning to classify passwords as weak, medium, or strong based on features like length, entropy, and the presence of special characters.

Level-2 :- Enhance the Password Strength Classifier by using a Neural Network and incorporating additional features such as common words (e.g., dictionary checks) and patterns (e.g., repeating characters) for more robust classification.

11.Adversarial Attack on an AI Model (FGSM Method):

Level-1: Implement an Adversarial Attack on a simple image classification model using the Fast Gradient Sign Method (FGSM) to generate adversarial examples and analyze their impact on model predictions.

Level-2: Enhance the FGSM attack by incorporating multiple adversarial examples and defense mechanisms (e.g., adversarial training or gradient masking) to make the model more resilient to adversarial perturbations.

12.IoT Security - Anomaly Detection in Smart Home Devices:

Level-1: Develop a basic anomaly detection system for IoT smart home devices using Isolation Forests to detect unusual behaviour based on features like device usage patterns, network traffic, and sensor data.

Level-2 : Enhance the IoT anomaly detection system by incorporating time-series analysis and deep learning techniques (e.g., LSTM Autoencoders) to detect more complex, subtle anomalies in the behaviour of smart home devices.

13. Secure AI Model Deployment Using Homomorphic Encryption:

Level-1: Implement Homomorphic Encryption to perform secure predictions using a simple AI model (e.g., Logistic Regression or Decision Tree) on encrypted data without decrypting it, ensuring data privacy.

Level-2: Enhance the AI model's deployment by implementing a more complex system with Homomorphic Encryption for neural networks and performing predictions on encrypted inputs using advanced encryption techniques such as Fully Homomorphic Encryption (FHE).

14.Blockchain for Secure Cybersecurity Logging:

Level-1: Implement a simple blockchain-based logging system using Python and Web3.py to securely store cybersecurity logs and ensure immutability, making it resistant to tampering.

Level-2:- Enhance the blockchain-based cybersecurity logging system by adding advanced features such as log verification, timestamping, and access control using smart contracts on a public Ethereum network.

15.Quantum Computing Simulation in Cybersecurity:

Level-1 : Simulate Shor's Algorithm using Qiskit to demonstrate how quantum computers could potentially break RSA encryption, illustrating the difference in computational power between classical and quantum systems.

Level-2 : Implement Quantum Key Distribution (QKD) using Qiskit to demonstrate how quantum communication could provide secure communication channels, immune to eavesdropping and interception, and analyze its implications for future cybersecurity.

Targeted Application & Tools that can be used:

AI-driven Intrusion Detection Systems, Fraud Detection Engines, Secure Email Filtering, Cyber Threat Intelligence Platforms using tools such as Python, Scikit-learn, TensorFlow, Keras, OpenCV, Wireshark, Splunk, and Jupyter Notebook.

Project work/Assignment:

6. **AI-Powered Intrusion Detection System**
7. **Phishing Email Detection Using NLP**
8. **Fraudulent Transaction Detector**
9. **Adversarial Attack Simulation**
10. **Anomaly Detection in IoT Devices**

Topics related to

1. Problem Solving: Designing and implementing AI models for real-time cybersecurity threat detection and response.

2. Employability: Simulation of AI-driven intrusion detection systems and fraud detection tools using machine learning and NLP techniques.

Textbook(s):

T1. I. Priyadarshini, R. Sharma, Eds., Artificial Intelligence and Cybersecurity: Advances and Innovations, Routledge, 2025.

T2. S. Mahajan, M. Khurana, and V. V. Estrela, Eds., Applying Artificial Intelligence in Cybersecurity Analytics and

Cyber Threat Detection, Wiley, 2024. **T3. Leslie F. Sikos, AI in Cybersecurity: A Practical Guide to Machine Learning Security, Springer, 2023.**
References **[1] L. Bass, P. Clements, and R. Kazman, Software Architecture in Practice, 4th ed., Addison-Wesley, 2021.** **[2] C. K. Hargreaves, Machine Learning for Cybersecurity Cookbook, Packt Publishing, 2020.** **[3]R3. Soma Halder, Sinan Ozdemir, Hands-On Machine Learning for Cybersecurity, Packt Publishing, 2021.** **E-Resources:** **https://www.coursera.org/specializations/ai-for-cybersecurity?utm_source=chatgpt.com** https://www.kaggle.com/datasets (Cybersecurity datasets for AI-based threat detection and fraud analysis) https://www.tensorflow.org/tutorials (Deep learning tutorials for AI-based cybersecurity model development)

Data Science, Big Data and IOT Stream

Course Code: CSA4721	**Course Title:** Applied Data Science and Feature Engineering **Type of Course:** Program Core	**L-T- P- C**	3	0	0	3
Version No.	1.0					
Course Pre-requisites	**NIL**					
Anti-requisites	**NIL**					
Course Description	This course provides a comprehensive foundation in applied data science, emphasizing both statistical principles and practical data analysis techniques. It begins with an overview of the data science process, including data preparation, exploratory data analysis, and modelling workflows used to extract insights from large and complex datasets. The course then introduces inferential statistics concepts such as probability distributions, confidence intervals, and hypothesis testing for data-driven decision making. Building on this, students learn techniques for anomaly detection, predictive modelling, and time series analysis used in real-world applications such as business analytics and social media analysis. The course also covers advanced feature engineering methods for text, image, and time-series data, along with automated feature engineering tools and data pipelines. Through assignments and case studies, students develop practical skills in data preprocessing, feature engineering, model development, and interpretation of analytical results for real-world problem solving.					
Course Objective	**The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.**					
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1**: Demonstrate fundamental understanding of the data science, data science lifecycle, and the role of data science techniques in solving real-world problems. [**Apply**] **CO2:** Apply statistical methods and exploratory data analysis techniques to summarize and interpret data characteristics. [**Apply**]					

	CO3: Interpret probability distributions and perform hypothesis testing for data-driven decision making. **[Apply]** **CO4:** Analyze datasets to identify patterns, relationships, and anomalies using appropriate analytical techniques**. [Analyze]** **CO5:** Develop predictive models using data science methods and evaluate their performance using suitable metrics. **[Apply/[Analyze]** **CO6:** Apply data science techniques such as time-series analysis and anomaly detection to solve real-world problems in various domains. [**Apply]**	
Course Content:		
Module 1	**Fundamentals of Data Science CO1**	**11 Sessions**
Overview of Data Science, Data Science process and workflow, Data volume, dimensionality and complexity, Data science tasks and examples, Overview of data preparation and modelling techniques.		
Module 2	**Inferential Statistics and Hypothesis – CO2**	**11 Sessions**
Probability distributions: Normal distribution, Poisson distribution, central limit Theorem, Confidence intervals, Hypothesis testing, Z-test and t-test, Statistical inference for data science.		
Module 3	**Anomaly Detection and Predictive Modelling – CO3**	**11 Sessions**
Introduction to anomaly detection, Types of anomalies, Statistical methods for anomaly detection, Density-based anomaly detection, Predictive modelling techniques, Model evaluation methods. Time series Analysis: Time series components, Time series forecasting techniques, Case studies in social media analytics, Business Intelligence and finance.		
Module 4	**Advanced Feature Engineering: – CO4**	**12 Sessions**
Feature engineering for text analytics (NLP), Feature engineering for Image data, Feature engineering for time-series forecasting, Automated Feature engineering tools, Feature stores and data pipelines, Case studies in Applied Data science.		
Project work/Assignment: 1. **Perform Exploratory Data Analysis on a real-world dataset (e.g., COVID – 19 data, sales data) using Python libraries** 2. **Conduct descriptive and inferential statistical analysis on a dataset to identify patterns and relationships.** 3. **Create an interactive dashboard to visualize trends and insights from a dataset using visualization tools** 4. **Analyze social media trends using data from Twitter or Instagram** 5. **Project: Forecast future values using: Moving averages, ARIMA model** 6. **Project: Does Online learning improve student performance? Perform a Null and alternative hypotheisi formulation and also perform Z-test and t-test and interpret statistical results**		
Topics related to 1. **Problem Solving**: Apply data science techniques such as data preprocessing, exploratory data analysis, statistical inference (hypothesis testing, confidence intervals), anomaly detection methods, predictive modelling, time-series forecasting, and advanced feature engineering to analyze datasets and solve real-world problems in domains such as business analytics, healthcare, finance, and social media analysis. 2. **Employability:** Hands-on experience with data analysis and modelling using tools such as Python, Pandas, Scikit-learn, and visualization libraries like Matplotlib, including tasks such as data preprocessing, statistical analysis, anomaly detection, predictive modelling, time-series forecasting, and feature engineering to develop practical skills for careers in data science, business analytics, and artificial intelligence.		
Textbook(s): **T1.** Ramcharan Kakarla, Sundar Krishnan, Balaji Dhamodharan and Venkata Gunnu, "Applied Data Science Using PySpark: Learn the End-to-End Predictive Model-Building Cycle", 2nd Edition, A press, 2024, ISBN: 979-8868808197. **T2.** Jaiteg Singh, S. B. Goyal, Rajesh Kumar Kaushal, Naveen Kumar and Sukhjit Singh Sehra, "Applied Data Science and Smart Systems", 1st Edition, CRC Press, 2024, ISBN: 978-1032425221. **T3.** Max Kuhn and Kjell Johnson, "Feature Engineering and Selection: A Practical Approach for Predictive Models", 1st Edition, Chapman & Hall/CRC Press, 2019, ISBN: 978-1138079229.		
References		

R1. Alice Zheng and Amanda Casari, "Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists", 1st Edition, O'Reilly Media, 2018, ISBN: 978-1491953242.
R2. Boba Samuels, Donna Kotsopoulos and Douglas G. Woolford, "Applied Data Science: Data Translators Across the Disciplines", 1st Edition, Springer, 2024, ISBN: 978-3031299391.
R3. Vidya Subramanian, "Applied Machine Learning for Data Science Practitioners", 1st Edition, Wiley, 2025, ISBN: 978-1394155378.
R4. Gintautas Dzemyda, Jolita Bernatavičienė and Janusz Kacprzyk, "Data Science in Applications: Towards AI-Driven Approaches", 1st Edition, Springer, 2025, ISBN: 978-3031884856.

E-Resources:

1. https://ocw.mit.edu
2. https://nptel.ac.in/courses
3. https://github.com/ssllabs/research/wiki/SSL-and-TLS-Deployment-Best-Practices
4. https://www.edx.org/learn/data-science

Course Code: CSA4722	**Course Title:** **Big Data Analytics using Hadoop and Spark** **Type of Course:** **Discipline Elective**	**L-T- P- C**	2	0	2	3
Version No.	1.0					
Course Pre-requisites	**NIL**					
Anti-requisites	**NIL**					
Course Description	This course provides a comprehensive introduction to Big Data concepts, tools, and technologies, equipping students with the skills to manage, process, and analyze large-scale data. Students will learn the foundations of Big Data, explore NoSQL databases, and gain hands-on experience with the Hadoop ecosystem and its components. The course covers Hadoop architecture, HDFS, MapReduce programming, and Hadoop ecosystem tools such as Hive, Pig, HBase, Sqoop, and Flume					
Course Objective	**The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.**					
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1**: Understand Big Data concepts, characteristics, and industry applications. **CO2:** Apply NoSQL data management techniques to model, store, and retrieve structured, semi-structured, and unstructured data. **CO3**: Apply Hadoop frameworks to perform distributed data storage, processing, and analysis using HDFS. **CO4:** Apply Apache Spark to perform data queries using Spark					
Course Content:						
Module 1	**Understanding Big Data**				**15 Sessions(8T+8P)**	
Introduction to Big Data, Characteristics, Industry applications and web analytics, Big Data technologies, Traditional vs Big Data systems, Types of data (structured, semi-structured, unstructured). Big Data applications and use cases, Overview of Big Data ecosystem, Cloud computing & Big Data, Mobile business intelligence.						
Module 2	**NoSQL Data Management**				**15 Sessions(8T+8P)**	
Introduction to NoSQL, Data models: Key-value, Document, Graph databases, Schema-less databases, Replication & consistency, Cassandra: Data model, Examples & clients.						
Module 3					**15 Sessions(7T+7P)**	

	Basics of Hadoop and Hadoop ecosystem	
Introduction to Hadoop, Hadoop architecture, HDFS (Hadoop Distributed File System), Blocks, replication, NameNode, DataNode, Hadoop cluster setup basics), Data formats & analysis using Hadoop, Hadoop I/O, Compression, Serialization, Avro & data structures, Integration with databases. **Hadoop ecosystem**: Apache Hive (HiveQL, tables), Apache Pig (Pig Latin basics,) Apache HBase.		
Module 4	**Introduction to Apache Spark**	**15 Sessions(7T+7P)**
Overview of Apache Spark, Spark vs Hadoop MapReduce, Spark architecture, RDD (Resilient Distributed Datasets), Transformations and Actions Spark Programming: Spark Core concepts, Spark SQL, DataFrames and Datasets, Spark Streaming basics.		

List of Laboratory Tasks:

Experiment No. 1:
Level 1: Install and download Windows Hadoop and HDFS.Configure and run Hadoop and HDFS.
Level 2: Install Java and verify Java Installation.

Experiment No. 2:
Level 1: Install and configure MongoDB/ HBase to execute NoSQL Commands.
Level 2: NoSQL and MongoDB – Basic CRUD Operations.

Experiment No. 3:
Level 1: Develop a MapReduce program to calculate the frequency of a given word in a given file.
Level 2: Develop a MapReduce program to find the grades of student.

Experiment No. 4:
Level 1: Develop a MapReduce program to implement Matrix Multiplication.
Level 2: Develop a MapReduce to find the maximum electrical consumption in each year given electrical consumption for each month in each year.

Experiment No. 5:
Level 1: Set up a single-node Hadoop cluster and verify HDFS status. Create a directory and upload sample files to HDFS.
Level 2: Write and run a basic MapReduce program to count word frequency in a text file.

Experiment No. 6:
Level 1: Create a directory in HDFS and Upload a local file to HDFS.
Level 2: List files in HDFS and display contents of a file.

Experiment No. 7:
Level 1: Retrieve all records and display names of students with marks > 80 in HDFS.
Level 2: Find the average marks of students using aggregate functions**.**

Experiment No. 8:
Level 1: Counting Number of Files in HDFS Directory.
Level 2: Move or rename a file in HDFS,Deleting a File from HDFS.

Experiment No. 9:
Level 1: Filter data based on a keyword or pattern from a text file in HDFS.
Level 2: Sorting Records of a File in HDFS.

Experiment No. 10:

Level 1: Checking File Size and Storage in HDFS.
Level 2: Counting Words in a File Stored in HDFS.

Experiment No. 11:
Level 1: Create HBase table and insert sample records.
Level 2: Demonstrate HBase row-level access and column family management.Create column families and store/retrieve data.

Experiment No. 12:
Level 1: Creating and Managing Tables in Hive and loading Data into Hive Tables.
Level 2: Querying Data using HiveQL (SELECT, WHERE).

Experiment No. 13:
Level 1: Filtering and Sorting in Hive.
Level 2: Find average salary per department in Hive.

Experiment No. 14:
Level 1: Filtering Data using Pig Latin.
Level 2: Grouping and Aggregating Data in Pig.

Experiment No. 15:
Level 1: Create a simple RDD from a list of numbers and apply basic transformations (map, filter) and count actions on RDDs.
Level 2: Create a DataFrame from a CSV file and perform simple queries using Spark SQL.

Software Required for Big Data:
1. Java JDK (8 or above): Required for Hadoop, HBase, Hive, Pig, and Spark
2. Apache Hadoop
3. MongoDB and HBase
4. Apache Hive,Apache Pig,Apache Spark

Programming / IDE Tools: Eclipse / IntelliJ IDEA / VS Code/Python

References/Manual/Software:

1. https://hadoop.apache.org/docs/stable/\
2. Hadoop: Setting up a Single Node Cluster. https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-common/SingleCluster.html
3. Java Download: ttps://www.oracle.com/java/technologies/downloads/
4. Install MongoDB: https://www.mongodb.com/docs/manual/installation/
5. Apache HBase Official Documentation: https://hbase.apache.org/book.html#quickstar
6. Apache Hive Installation Guide: https://hive.apache.org/
7. Apache Pig, https://pig.apache.org/docs/latest/
8. Apache Spark Official Documentation: https://spark.apache.org/docs/latest/

Python (for PyMongo / Spark): https://www.python.org/downloads/

Project work/Assignment:

1. Big Data Analytics for E-Commerce Customer Insights
2. Implementing Hadoop and Spark for Large-Scale Data Processing and Analytics
3. E-Commerce Customer Analytics Using Big Data Technologies
4. Analyzing Social Media Data Using Hadoop and Spark"
5. Real-Time Sensor Data Processing with Apache Spark Streaming
6. Customer Behavior Analysis Using Hive, Pig, and HBase

Topics related to

1. Problem Solving: Understanding Big Data, Problem-solving focuses on identifying business challenges that require Big Data solutions, analyzing structured, semi-structured, and unstructured datasets, and designing strategies for data collection, storage, and processing. Students apply Big Data technologies to real-world scenarios such as web analytics,

mobile business intelligence, and industry-specific use cases.
2. **Employability**: Hands-on experience with Big Data Analytics significantly enhances employability by equipping students with highly sought-after skills in data management, processing, and analysis. Graduates gain practical experience with Hadoop, Spark, Hive, Pig, HBase, Flume, and NoSQL databases, enabling them to handle large-scale, complex datasets across industries such as e-commerce, finance, healthcare, and social media.

Textbook(s):

T1.Greyson Chesterfield," Big Data Analytics: How to Process and Analyze Large Datasets with AI: A Practical Guide to Apache Spark, Hadoop, and Data Engineering" .Kindle Edition.March 2025
T2. Dr. Brinthakumari.S, Radhika Ajit patil, Dr. Sivaraja. P M, Dr. K. Jagadheesh,Mastering Big Data Analytics with Hadoop: From Basics to Advanced Techniques, September 2024
T3.Simhadri Govindappa," Ultimate Big Data Analytics with Apache Hadoop: Master Big Data Analytics with Apache Hadoop Using Apache Spark, Hive, and Python ",September 2024

References

R1.Rathinaraja Jeyaraj. "Big Data with Hadoop MapReduceEbook, A Classroom Approach". Apple Academic Press Publisher,May 2020
R2.Diego Rodrigues, "Fundamentals of Big Data: With Hadoop and Spark" ,2024 Edition
R3.Dr. Haynesh Desai, Jahnvi Masrani ,"Data Analytics with Hadoop" , 2025 Edition,
R4.Tom White Hadoop: The Definitive Guide, 4th Edition, O'Reilly Media ,2015

E-Resources:

1. https://www.perlego.com/book/1479034/big-data-with-hadoop-mapreduce-a-classroom-approach-pdf?utm_source=chatgpt.com
2. https://www.coursera.org/learn/introduction-to-big-data-with-spark-hadoop?utm_source=chatgpt.com
3. https://www.coursera.org/learn/big-data-processing-with-hadoop-and-spark?utm_source=chatgpt.com

Course Code: CSA4723	**Course Title:** Time Series Analysis and Forecasting Techniques **Type of Course:** Program Core	**L-T- P- C**	3	0	0	3
Version No.	1.0					
Course Pre-requisites	**NIL**					
Anti-requisites	**NIL**					
Course Description	Time Series Analysis and Forecasting Techniques focuses on analysing data collected over time and predicting future values. The course introduces key components such as trend, seasonality, and irregular variations. It covers methods for visualization, decomposition, and stationarity analysis. Students learn autocorrelation techniques and statistical models like AR, MA, ARMA, and ARIMA. Emphasis is given to the Box–Jenkins methodology for model building and validation. Forecasting methods such as moving averages and exponential smoothing are discussed along with accuracy evaluation. The course highlights real-world applications in business and engineering. By the end, students will be able to analyses time series data and apply suitable forecasting techniques.					
Course Objective	**The objective of the course is to familiarize the learners with the concepts of Time Series Analysis and Forecasting Techniques and attain skill development through experiential learning techniques in analyzing time-dependent data, building statistical models, and applying forecasting methods for real-world applications.**					

Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1:** Understand the fundamental concepts of time series data, its components (trend, seasonal, cyclic, irregular), and basic decomposition technique [**Understand**] **CO2:** Understand the concept of stationarity, white noise, and random processes in time series data. [**Understand**] **CO3:** Apply autocorrelation (ACF) and partial autocorrelation (PACF) techniques to analyze time series behavior. [**Apply**] **CO4:** Understand the structure and characteristics of AR, MA, and ARMA models. [**Understand**] **CO5:** Apply ARIMA modeling and Box–Jenkins methodology for building time series forecasting models. [**Apply**] **CO6:** Apply appropriate forecasting techniques such as moving averages and exponential smoothing to predict future values and evaluate model performance. [**Apply**]	
Course Content:		
Module 1	**Introduction to Time Series – CO1**	**10 Sessions**
Definition and components of time series (Trend, Seasonality, Cyclic, Irregular) - Types of time series data - Applications in business, economics, engineering - Time series visualization techniques - Decomposition methods (Additive & Multiplicative models)		
Module 2	**Stationary Processes & Correlation Analysis - CO2 & CO3**	**11 Sessions**
Stationarity (strict & weak) - White noise and random walk - Autocorrelation Function (ACF) - Partial Autocorrelation Function (PACF) - Identification of time series models		
Module 3	**Time Series Models – CO4 & CO5**	**12 Sessions**
Autoregressive (AR) models - Moving Average (MA) models - ARMA models - non-stationary series and differencing - ARIMA modeling - Box–Jenkins's methodology (Identification, Estimation, Diagnostic Checking)		
Module 4	**Forecasting Techniques – CO6**	**12 Sessions**
Naïve forecasting methods - Moving averages - Exponential smoothing (Simple exponential smoothing, Holt's linear method, Holt-Winters seasonal method) - Regression-based forecasting - Forecast accuracy measures (MSE, RMSE, MAPE)		
Project work/Assignment: 1. Time Series Decomposition Analysis 2. Stationarity and Autocorrelation Study 3. ARIMA Model Implementation 4. Exponential Smoothing Techniques 5. Forecast Accuracy Evaluation		
Topics related to **1. Problem Solving:** Apply time series decomposition, stationarity testing, and ARIMA modeling techniques to analyze real-world datasets (such as stock prices, weather data, or sales data) and develop accurate forecasting solutions for decision-making. **2. Employability:** Gain hands-on experience in analyzing datasets, building forecasting models, evaluating performance metrics (MSE, RMSE, MAPE), and using software tools, thereby enhancing skills for careers in data analytics, business forecasting, and research.		
Textbook(s): **T1.** Introduction to Time Series and Forecasting, Peter J. Brockwell and Richard A. Davis, 3rd Edition, 2021, Springer **T2**. Time Series Analysis: Forecasting and Control, George E. P. Box, Gwilym M. Jenkins, Gregory C. Reinsel and Greta M. Ljung, 5th Edition, 2022, Wiley.		
References **R1.** Time Series Analysis and Its Applications, Robert H. Shumway and David S. Stoffer, 4th Edition, 2020, Springer. **R2**. Forecasting: Principles and Practice, Rob J. Hyndman and George Athanasopoulos, 3rd Edition, 2021, OTexts. **R3.** Time Series Analysis, James D. Hamilton, 1st Edition, 1994, Princeton University Press. **R4.** The Analysis of Time Series: An Introduction, Chris Chatfield, 6th Edition, 2021, CRC Press. **R5.** Time Series: Theory and Methods, Peter J. Brockwell and Richard A. Davis, 2nd Edition, 2022, Springer.		

E-Resources:
1. http://www.otexts.com/fpp3/ 2. http://www.otexts.com/fpppy/ 3. http://www.ptsf.netlify.app/ 4. http://www.coursera.org/learn/practical-time-series-analysis/ 5. http://www.edx.org/learn/data-analysis/time-series-analysis

Course Code: CSA4724	**Course Title: IoT and Edge Computing Systems** **Type of Course:**	**L-T- P- C**	2	0	2	3
Version No.	1.0					
Course Pre-requisites	**Iot and Edge Computing**					
Anti-requisites	**NIL**					
Course Description	This course introduces the concepts of Internet of Things (IoT) and Edge Computing, focusing on architecture, communication protocols, smart devices, data processing, and real-world applications. It integrates theoretical foundations with hands-on lab exercises to enable students to design, implement, and deploy IoT systems with edge intelligence.					
Course Objective	The objective of this course is: • To understand IoT architecture, devices, and communication protocols • To analyze and implement real-time data processing using edge computing technologies					
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1:** Apply IoT architecture concepts to design smart systems and applications. **(Apply)** **CO2:** Analyze IoT communication protocols and networking technologies. **(Analyze)** **CO3:** Apply sensor and actuator interfacing techniques to build IoT solutions. **(Apply)** **CO4:** Analyze edge computing frameworks for real-time data processing. **(Analyze)** **CO5:** Apply cloud-edge integration techniques for scalable IoT systems. **(Apply)** **CO6:** Evaluate security challenges and implement secure IoT applications. **(Evaluate)**					
Course Content:						
Module 1	**Introduction to IoT and Architecture – CO1**					**15 Sessions (6T + 9P)**
Definition, characteristics, and evolution of IoT, IoT ecosystem: physical devices, connectivity, platforms, and applications, IoT Architecture: • 3-layer architecture (Perception, Network, Application) • 5-layer architecture (Perception, Transport, Processing, Application, Business), IoT vs Machine-to-Machine (M2M) communication, Design considerations: scalability, interoperability, energy efficiency, Overview of IoT hardware platforms: Arduino, Raspberry Pi, NodeMCU, Introduction to real-world IoT applications (smart homes, healthcare, smart cities)						
Module 2	**IoT Devices and Communication Protocols – CO2**					**15 Sessions (6T + 9P)**
Sensors and actuators: types, working principles, and interfacing basics Embedded systems overview and microcontroller fundamentals, Communication protocols in IoT: 1. MQTT (publish-subscribe model, broker architecture) 2. CoAP (lightweight REST-based protocol),3. HTTP/REST APIs for IoT communication, Wireless communication technologies: Wi-Fi, Bluetooth, Zigbee, LoRaWAN , Network topologies and addressing in IoT ,Data transmission challenges: latency, bandwidth, reliability						
Module 3	**IoT Data Management and Cloud Integration – CO3**					**15 Sessions (6T + 9P)**

Data acquisition from sensors and preprocessing techniques,Data formats: JSON, XML in IoT communication , Data storage techniques: local storage vs cloud storage, Introduction to cloud platforms: AWS IoT, Microsoft Azure IoT Hub (overview) , API-based integration between IoT devices and cloud ,Real-time data streaming and visualization techniques ,Basics of data analytics in IoT systems

Module 4	**Spring Framework and RESTful Web Services – CO4**	**15 Sessions (6T + 9P)**

Introduction to Edge Computing and its need in IoT, Comparison: Cloud vs Fog vs Edge computing, Edge architecture: edge devices, gateways, and local processing units, Real-time data processing and low-latency requirements, Data filtering, aggregation, and preprocessing at the edge, Introduction to Edge AI and lightweight machine learning models, Use cases: autonomous systems, industrial IoT, smart surveillance

Module 5	**Automation Testing with Maven and Selenium – CO5, CO6**	**15 sessions (3T + 12P)**

Security challenges in IoT: device vulnerability, network threats, Authentication and authorization mechanisms, Encryption techniques for secure communication, Data privacy, integrity, and trust management, Secure firmware updates and device management, Case studies: 1. smart home automation, 2.Smart healthcare monitoring , 3.Industrial IoT (IIoT) systems ,Emerging trends: A IoT (AI + IoT), 5G-enabled IoT, Edge intelligence

List of Laboratory Tasks:

Experiment No. 1:
Level 1: Install Arduino IDE and perform LED blinking experiment

Level 2: Interface multiple LEDs and control using push buttons

Experiment No. 2:
Level 1: Read analog data from temperature sensor

Level 2: Display sensor data on serial monitor with calibration.

Experiment No. 3:
Level 1: Interface humidity sensor with Arduino
Level 2: Build temperature-humidity monitoring system
Experiment No. 4:
Level 1: Setup Raspberry Pi and control GPIO pins
Level 2: Build LED control system using Raspberry Pi
Experiment No. 5:
Level 1: Send sensor data using HTTP protocol
Level 2: Send sensor data to cloud using MQTT protocol
Experiment No. 6:
Level 1: Create simple IoT dashboard for data visualization
Level 2: Build real-time dashboard using cloud platform
Experiment No. 7:
Level 1: Develop REST API for IoT device
Level 2: Integrate IoT device with REST API and database
Experiment No. 8:
Level 1: Edge processing using Raspberry Pi (basic filtering)
Level 2: Implement real-time edge analytics system
Experiment No. 9:
Level 1: Smart home simulation using sensors
Level 2: Build home automation system with mobile control
Experiment No. 10:
Level 1: Implement basic security (password-based access)
Level 2: Implement encrypted communication between devices

Targeted Application & Tools that can be used:

Arduino, Raspberry Pi, NodeMCU, MQTT Broker, AWS IoT, Azure IoT Hub, Python, Embedded C, REST APIs, ThingSpeak, Blynk
Project work/Assignment: **1. Build a Smart Home Automation System using IoT devices** **2. Develop a Smart Agriculture Monitoring System** **3. Create a Health Monitoring System using wearable sensors** **4. Build an Edge-based Traffic Monitoring System** **5. Develop Industrial IoT (IIoT) monitoring solution**
Topics related to 1. Problem Solving: Design and develop real-world IoT systems integrating sensors, communication protocols, cloud platforms, and edge computing to solve problems such as smart monitoring and automation. 2. Employability: Use industry tools like Raspberry Pi, MQTT, AWS IoT, and edge computing frameworks to build scalable IoT solutions, enhancing job readiness in IoT, embedded systems, and cloud domains.
Textbook(s): T1: Raj Kamal, *Internet of Things: Architecture and Design Principles*, 4th Edition, McGraw Hill, 2023 T2: Pethuru Raj & Anupama C. Raman, *The Internet of Things: Enabling Technologies, Platforms, and Use Cases*, CRC Press, 2022 T3: Qingyang Wang, *Edge Computing: Fundamentals and Applications*, Wiley, 2021
References **R1:** David Hanes et al., *IoT Fundamentals: Networking Technologies, Protocols, and Use Cases*, Cisco Press, 2021 **R2:** Kai Hwang, *Cloud and Edge Computing Systems*, Morgan Kaufmann, 2022 **R3:** Vijay Madisetti, *Internet of Things: A Hands-on Approach*, Universities Press, 2021

Course Code: **CSA4725**	**Course Title:** **Data Engineering and Pipeline Architecture** **Type of Course:** **Program Core**	**L-T- P- C**	2	0	2	3
Version No.	1.0					
Course Pre-requisites	**NIL**					
Anti-requisites	**NIL**					
Course Description	This course introduces the fundamental concepts and practical techniques required to design, develop, and manage modern data engineering systems and data pipeline architectures. It focuses on the end-to-end lifecycle of data including data ingestion, transformation, storage, processing, and delivery for analytics and machine learning applications. Students will learn how to build scalable and reliable ETL/ELT pipelines, understand distributed data processing frameworks, and implement pipeline orchestration using modern tools and cloud platforms. The course also covers data quality, pipeline monitoring, performance optimization, and real-time data processing in large-scale data environments. Data pipelines automate stages such as data ingestion, processing, storage, and consumption, enabling continuous data flow for analytics and intelligent applications					
Course Objective	The objective of the course is to familiarize learners with Data Engineering & Pipeline Architecture, while developing general-purpose applications with database connectivity through experiential learning.					

Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1:** Understand the principles of data engineering and modern data pipeline architectures. **(Understand)** **CO2:** Implement data ingestion and transformation pipelines using ETL/ELT techniques. **(Apply)** **CO3**: Apply distributed data processing frameworks for large-scale data processing. **(Apply)** **CO4:** Develop scalable and reliable data pipelines using orchestration tools. **(Apply)** **CO5:** Evaluate data quality, monitoring, and performance optimization techniques in data pipelines. **(Analyze)** **CO6:** Build practical data engineering workflows integrating cloud platforms and big data technologies. **(Analyze)**	
Course Content:		
Module 1	**Introduction to Data Engineering and Data Pipeline Architecture**	**12 Sessions (6T+6P)**
Fundamentals of Data Engineering, Role of Data Engineers in Data Ecosystem, Data Engineering Lifecycle, Introduction to Data Pipeline Architecture, Batch Processing vs Stream Processing, Data Pipeline Components and Architecture Patterns.		
Module 2	**Data Ingestion and ETL/ELT Processing**	**16 Sessions (8T +8P)**
Data Sources and Data Ingestion Techniques, ETL vs ELT Architecture, Data Transformation Techniques, Data Integration and Data Warehousing, Data Cleaning and Data Quality Management, Tools for ETL and Data Integration		
Module 3	**Big Data Processing and Distributed Data Systems**	**16 Sessions (8T + 8P)**
Introduction to Big Data Technologies, Distributed Data Processing Concepts, Hadoop Ecosystem (HDFS, MapReduce, Hive), Apache Spark for Data Processing, Data Storage Systems and Data Lakes, Data Modeling for Large-Scale Data Systems		
Module 4	**Data Pipeline Orchestration and Monitoring**	**16 Sessions (8T + 8P)**
Pipeline Orchestration Concepts Workflow Scheduling Tools (Airflow, Prefect), Real-time Data Processing (Kafka, Streaming Systems), Data Pipeline Monitoring and Logging, Performance Optimization and Scalability, Security, Governance, and Emerging Trends in Data Engineering		

Lab 1: Introduction to Data Pipelines

Level 1: Design a simple batch pipeline to read CSV data and display it in a table.

Level 2: Modify the pipeline to handle multiple data formats (CSV, JSON, XML) with logging.

Lab 2: Batch vs Stream Processing

Level 1: Implement a batch job to calculate daily sales totals from a static dataset.

Level 2: Implement a streaming job using Kafka or Spark Streaming to compute live metrics.

Lab 3: Data Pipeline Architecture

Level 1: Draw and implement a basic pipeline with ingestion, processing, and storage components.

Level 2: Add monitoring and error handling with logs and alerts for pipeline failures

Lab 4: ETL Pipeline with Python

Level 1: Extract data from a CSV file, clean nulls, and load into SQLite.

Level 2: Integrate multiple sources (CSV + API), transform data, and load into PostgreSQL.

Lab 5: Data Transformation Techniques

Level 1: Perform filtering, aggregation, and sorting on sample data using Python or Spark DataFrame.

Level 2: Implement a reusable transformation module that can be applied to multiple datasets.

Lab 6: Data Cleaning and Quality Management

Level 1: Identify missing or inconsistent data in a dataset and correct it.

Level 2: Implement automated validation rules for incoming datasets with error logging.

Lab 7: Data Warehousing

Level 1: Create a simple star schema for sales data and load sample data.

Level 2: Implement slowly changing dimensions and summarize fact tables for analytics.

Lab 8: Hadoop HDFS & MapReduce

Level 1: Upload data to HDFS and run a basic MapReduce job to count words.

Level 2: Implement a MapReduce job that joins two datasets and computes summary statistics.

Lab 9: Apache Spark RDDs and DataFrames

Level 1: Load CSV data into Spark DataFrame, filter rows, and show results.

Level 2: Perform aggregations and joins on multiple datasets and save output to HDFS.

Lab 10: Data Modeling for Large-Scale Systems

Level 1: Design a simple schema for a large-scale user activity dataset.

Level 2: Implement schema in Hive/Spark and optimize for queries with partitioning and bucketing.

Lab 11: Data Storage Systems & Data Lakes

Level 1: Store sample data in AWS S3 or local data lake.

Level 2: Implement a pipeline that reads raw data, transforms it, and writes curated datasets to the lake.

Lab 12: Workflow Orchestration with Airflow

Level 1: Create a DAG in Airflow to schedule a simple ETL task daily.

Level 2: Implement multiple dependent DAGs with error handling and email notifications.

Lab 13: Real-Time Data Processing

Level 1: Consume a Kafka topic and display incoming messages in real-time.

Level 2: Aggregate streaming data (e.g., count events per minute) and store results in a database.

Lab 14: Pipeline Monitoring & Logging

Level 1: Add logging to an existing pipeline and capture basic metrics.

Level 2: Implement a monitoring dashboard that tracks pipeline performance and failures.

Lab 15: Performance, Scalability, Security

Level 1: Benchmark a Spark job with sample data and observe execution time.

Level 2: Optimize the pipeline using partitioning, caching, and configure role-based access for data security.

Targeted Application & Tools that can be used:

ETL Pipelines, Data Warehousing, Real-Time Streaming, Machine Learning Data Preparation, Analytics & Reporting using tools such as Python, Apache Spark, PySpark, Apache Kafka, Apache Airflow, dbt, Snowflake, BigQuery, Pandas, and Jupyter Notebook.

Project work/Assignment:

- Build ETL pipelines to ingest, clean, transform, and load data into a warehouse or database.
- Implement batch and real-time streaming pipelines using Spark and Kafka with live analytics dashboards.
- Design data warehouses with star/snowflake schemas and perform CRUD operations using SQL/JDBC.
- Orchestrate workflows with Airflow/Prefect, including logging, error handling, and monitoring.
- Develop a data lake solution integrating multiple sources, distributed processing, and data validation.

Topics related to

Problem Solving: Apply data engineering principles to design modular, scalable, and efficient data pipelines handling batch and real-time data, ensuring data quality, transformation, and error management across distributed systems.

Employability: Hands-on experience with ETL pipelines, Apache Spark, Kafka, Airflow, data warehousing, and database integration to prepare for roles such as Data Engineer, Big Data Developer, and Pipeline/Backend Engineer.

Textbook(s):

T1. Joe Reis and Matt Housley, *Fundamentals of Data Engineering: Plan and Build Robust Data Systems*, O'Reilly Media, 2022.

T2. Paul Crickard, *Data Engineering with Python*, Packt Publishing, 2020

T3. James Densmore, *Data Pipelines Pocket Reference*, O'Reilly Media, 2021

References

R1. Valliappa Lakshmanan, *Machine Learning Design Patterns*, O'Reilly Media, 2020

R2. Manish Kumar and Ruchi Verma, Big Data Analytics and Data Engineering, Springer, 2021.

R3. Holden Karau, Andy Konwinski, Patrick Wendell, Matei Zaharia, Learning Spark, O'Reilly Media, 2020

E-Resources:

1. Data Engineering Cookbook | PDF | Apache Spark | Apache Hadoop
2. GitHub - PacktPublishing/Data-Engineering-with-Python: Data Engineering with Python, published by Packt · GitHub
3. https://www.udemy.com/course/data-engineering-essentials-sql-python-and-spark/
4. https://learn.microsoft.com/en-us/training/courses/dp-700t00

Quantum Computing Stream

Course Code: CSA4726	**Course Title:** Quantum Information Theory **Type of Course:** Discipline Elective	**L-T- P- C**	3	0	0	3
Version No.	1.0					
Course Pre-requisites	**NIL**					
Anti-requisites	**NIL**					
Course Description	This course introduces the fundamental principles of quantum mechanics applied to information processing. It covers quantum states, entanglement, quantum gates, and key algorithms, along with communication protocols like quantum teleportation and cryptography. The course also addresses noise, de-coherence, and quantum error correction, providing a foundation for understanding modern quantum technologies.					
Course Objective	**The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.**					
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1**: Explain the fundamental principles of quantum mechanics and represent quantum states using appropriate mathematical frameworks. **[Understand]** **CO2**: Analyze quantum entanglement, measurement processes, and correlations in multi-qubit systems. **[Analyze]** **CO3**: Apply quantum gates and circuits to implement basic quantum algorithms and information processing tasks. **[Apply]** **CO4**: Evaluate quantum communication protocols and error correction techniques in the presence of noise and de-coherence. **[Evaluate]** **CO5:** Apply the concepts of quantum decoherence, noise models, mitigation techniques, and fault tolerance principles, including the threshold theorem, to analyze the reliability of quantum computing systems.**[Apply]** **CO6:** Apply the principles of quantum network architecture and communication protocols to explain and evaluate applications of quantum networks in secure communication and distributed quantum computing.**[Apply]**					
Course Content:						
Module 1	**Foundations of Quantum Mechanics for Information - CO1**					**08 Sessions**
Linear Algebra Review: Vector spaces, inner products, eigenvalues, eigenvectors, unitary operators. Hilbert Spaces: State representation, basis, orthonormality. Postulates of Quantum Mechanics: State evolution, measurement, observables. Qubits: Physical realization, Bloch sphere representation, single qubit operations. Composite Systems: Tensor products, multi-qubit systems, separable vs entangled states. Measurement Theory: Projective measurements, POVMs, measurement postulate. Density Matrix Formalism: Pure vs mixed states, trace, and partial trace.						
Module 2	**Quantum Entanglement and Correlations – CO2**					**10 Sessions**
Classical vs Quantum Correlations: Differences and examples. Entanglement: Definition, properties, physical significance. Bell States: Construction and properties. Bell Inequalities: CHSH inequality and violations. Bell's Theorem: Non-locality and implications. Entropy Measures: Shannon entropy vs von Neumann entropy. Entanglement Measures: Entropy of entanglement, concurrence, negativity.						
Module 3	**Quantum Information Processing – CO3**					**13 Sessions**
Quantum Gates: Pauli gates, Hadamard, phase gates. Quantum Circuits: Circuit model, gate composition, measurement in circuits. Universal Gate Sets: Single qubit + CNOT universality						

Quantum Algorithms: Shor's algorithm (factoring), Grover's algorithm (search). Quantum Teleportation: Protocol steps and circuit. Super-dense Coding: Information capacity and protocol. No-Cloning Theorem: Proof and implications.

Module 4	**Quantum Communication & Error Correction – CO4**	**14 Sessions**

Quantum Channels: Noise models (bit flip, phase flip, depolarizing). Quantum Error Correction: Shor code, Steane code, syndrome measurement. Quantum Cryptography: BB84 protocol, security concepts. Channel Capacity: Classical vs quantum capacity, Holevo bound

De-coherence: Sources, models, and mitigation. Fault Tolerance: Threshold theorem basics

Quantum Networks: Basic architecture and applications.

Project work/Assignment:

1. **Explain the concept of a qubit and represent it using the Bloch sphere. How does it differ from a classical bit?**
2. **Analyze a Bell state and determine whether it is entangled. Show the necessary calculations.**
3. **Design a simple quantum circuit using basic gates (Hadamard, Pauli-X, CNOT) to create an entangled state.**
4. **Implement quantum teleportation using a quantum circuit (e.g., Qiskit) and explain each step involved.**
5. **Explain how the BB84 protocol enables secure quantum communication and compare it with classical cryptography.**

Topics related to

1. Problem Solving: Apply network design techniques (like subnetting, routing algorithm selection, and flow control methods) and cryptographic strategies (encryption/decryption) to solve real-world communication, data security, and transmission reliability challenges.

2. Employability: Hands-on experience with configuring IPv4/IPv6 networks, analyzing TCP/UDP traffic, simulating routing protocols, implementing cryptographic algorithms, and securing data using SSL/TLS protocols to enhance practical skills for careers in networking, cybersecurity, and system administration.

Textbook(s):

T1. Giacomo Mauro D'Ariano, Paolo Perinotti, "Quantum Information and Foundations", MDPI, 2020, ISBN: 978-3039219438.

T2. Sumeet Khatri, Mark M. Wilde, "Principles of Quantum Communication Theory: A Modern Approach", Cambridge University Press, 2020, ISBN: 978-1108491457

References

R1. J. J. Sakurai, Jim Napolitano, "Modern Quantum Mechanics", 3rd Edition, Cambridge University Press, 2020, ISBN: 978-1108473224

R2. Horacio Casini, Marina Huerta, "Lectures on Entanglement in Quantum Field Theory", Springer, 2022, ISBN: 978-3030902711.

R3. Joseph D. Lykken, "Quantum Information for Particle Theorists", Springer, 2020, ISBN: 978-3030523510

E-Resources:

4. https://learning.quantum.ibm.com/course/basics-of-quantum-information
5. https://learning.quantum.ibm.com/learning-path/understanding-quantum-information-and-computation
6. https://learning.quantum.ibm.com/learning-path/getting-started-with-qiskit
7. https://qiskit.qotlabs.org/docs/tutorials
8. https://softhints.com/quantum-computing-for-beginners/

Course Code: CSA4727	**Course Title: Quantum algorithm and programming** **Type of Course: Discipline Elective**	**L-T- P- C**	2	0	2	3
Version No.	1.0					
Course Pre-requisites						
Anti-requisites	**NIL**					
Course Description	This course introduces the basic concepts of quantum computing in a simple and clear way. It helps students understand how quantum systems work and how they are different from classical computers. The course also covers quantum frameworks and quantum gates, which are the building blocks of quantum circuits. Students will learn basic quantum algorithms to understand how problems can be solved using quantum computing. In addition, the course explains quantum error correction methods to handle errors in quantum systems. Finally, it provides an introduction to quantum cryptography, showing how quantum principles can be used for secure communication.					
Course Objective	**The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.**					
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1**: Apply fundamental concepts of Quantum Computation and Quantum Information to analyze and construct quantum circuits using qubits, quantum gates, measurements, and linear algebra formulations. **(Apply)** **CO2:** Analyze and apply the principles of quantum state representation, system evolution, measurement, entanglement, and quantum gate operations to solve basic quantum computing problems. **(Apply)** **CO3**: Apply and analyze fundamental quantum algorithms to solve computational problems using quantum computing principles. **(Apply)** **CO4:** Analyze and apply quantum error correction techniques along with fault-tolerant quantum operations and quantum teleportation, to ensure reliable quantum computation. **(Analyze)** **CO5:** Analyze the principles of private key cryptography, privacy amplification, information reconciliation, and quantum key distribution protocols (including BB84) to evaluate their role in secure communication. **(Analyze)** **CO6:** Evaluate the security aspects of QKD protocols, quantum error correction techniques, and the limitations of quantum operations in practical quantum systems. **(Analyze)**					
Course Content:						
Module 1	**Introduction To Quantum Computing– CO1**				**15 Sessions (3T + 12P)**	
History of quantum computation and quantum information - Quantum bits – Quantum computation: Single qubitgates – Multiple qubit gates – Measurements – Quantum circuits – Qubit Copy – Bell States –Circuit Model of Computation – Linear algebra Formulation of the Circuit Model of Computation: NOT gate – CNOT gates –Reversible Computation – Dirac Notation and Hilbert space – Tensor product - Schmidt decomposition Theorem						
Module 2	**Quantum Frameworks and Quantum Gates – CO2**				**15 Sessions (3T + 12P)**	
State of a Quantum system: State space postulate – Bloch sphere - Time evolution of a closed system: Evolution Postulate – Schrodinger Equation – Composite Systems: Composition of system postulate – Entanglement – EPR pair– Measurement: Measurement Postulate – Mixed state – Partial trace – Quantum gates: Working of Quantum Pauli gates – Rotation gates – CNOT gate – Hadamard gate						
Module 3	**Quantum Algorithms – CO3**				**15 Sessions (3T + 12P)**	
Quantum Fourier Transform - The Deutsch algorithm – Deutsch-Jozsa algorithm – Simon's Algorithm – Shor's Algorithm - Grover's search algorithm						

Module 4	Quantum Error Correction and Fault Tolerant Computation – CO4	15 Sessions (3T + 12P)
Introduction: Three qubit bit flip code – Three qubit phase flip code – Shor code – Stabilizer code – Fault tolerance: Fundamental issues – Fault tolerant controlled NOT gate – Threshold theorem for quantum computation - Quantum teleportation		
Module 5	**Quantum Cryptography – CO5**	**15 Sessions (3T + 12P)**
Private key cryptography – Privacy amplification and information reconciliation – Quantum key distribution: QKD Protocol – BB84 protocol – Security of Quantum key distribution: Secure BB84 protocol – Quantum error correction protocol - Limitation of quantum operations		

Experiment No. 1:
Level 1: Simulate single qubit gates (X, Y, Z, H) on a qubit.

Level 2: Simulate single qubit gates on multiple qubits and visualize on Bloch sphere.

Experiment No. 2:
Level 1: Implement CNOT gate for 2 qubits.

Level 2: Generate Bell states and verify entanglement experimentally.

Experiment No. 3:
Level 1: Simulate 2-qubit quantum circuit using NumPy/PyTorch.

Level 2: Apply tensor product to compose multi-qubit states and measure correlations.

Experiment No. 4:
Level 1: Apply Pauli and rotation gates on a single qubit and observe outcome.

Level 2: Simulate a 2-qubit system with entanglement and partial measurement.

Experiment No. 5:
Level 1: Generate EPR pair and measure qubit correlation.

Level 2: Implement simple quantum teleportation protocol for a qubit.

Experiment No. 6:
Level 1: Implement Deutsch algorithm for 1-bit function.

Level 2: Implement Deutsch-Jozsa algorithm for n-bit function.

Experiment No. 7:
Level 1: Implement Simon's algorithm for 2-bit input.

Level 2: Implement Grover's search algorithm for 4-element database.

Experiment No. 8:
Level 1: Implement Quantum Fourier Transform on 2 qubits.

Level 2: Implement inverse QFT on 3-qubit system.

Experiment No. 9:
Level 1: Implement 3-qubit bit-flip error correction code.

Level 2: Implement 3-qubit phase-flip error correction code.

Experiment No. 10:
Level 1: Demonstrate simple quantum teleportation circuit.

Level 2: Combine teleportation with error correction and measure fidelity.

Experiment No. 11:
Level 1: Implement BB84 protocol for Quantum Key Distribution (QKD) with 2 qubits.

Level 2: Implement secure BB84 with error detection and privacy amplification.

Experiment No. 12:
Level 1: Simulate quantum key exchange using simple qubit states.

Level 2: Implement quantum error correction in QKD protocol.

Experiment No. 13:
Level 1: Implement basic quantum gates and measurement to simulate classical logic.

Level 2: Build a small multi-gate quantum circuit with entanglement.

Experiment No. 14:
Level 1: Analyze limitations of single qubit operations.

Level 2: Integrate multiple qubit operations to study interference and entanglement.

Experiment No. 15:
Level 1: Simulate basic quantum cryptography operations (key exchange).

Level 2: Implement an end-to-end QKD simulation with entanglement, measurement, and error correction.

Targeted Application & Tools that can be used:
Jupyter Notebook, NumPy, Matplotlib ,Plotly ,Qiskit , Cirq ,Pennylane ,Qiskit Aer imulator , Git / GitHub , VS Code , PyCharm , Eclipse , Atom , Anaconda , Scipy, SymPy

Project work/Assignment:

1. Simulate single qubit gates (X, Y, Z, H) on a qubit and extend to multiple qubits and visualize on Bloch sphere.
2. Implement CNOT gate for 2 qubits
   Generate Bell states and verify entanglement experimentally.
3. Simulate 2-qubit quantum circuits using NumPy/PyTorch
   Apply tensor product to compose multi-qubit states and measure correlations.
4. Apply Pauli and rotation gates on a single qubit
   Simulate a 2-qubit system with entanglement and partial measurement.
5. Generate EPR pair and measure qubit correlation
   Implement simple quantum teleportation protocol for a qubit.

Topics related to
1. Problem Solving: Develop quantum programs using quantum algorithms and frameworks (such as Qiskit or Cirq) to address real-world computational challenges like optimization, cryptography, and search problems.
2. Employability: Simulate industry-relevant quantum computing scenarios by implementing quantum circuits, analyzing algorithm performance, and utilizing hybrid quantum-classical approaches to build scalable solutions, enhancing readiness for careers in quantum software development.

Textbook(s):
T1. Michael A.Nielsen, Issac L. Chuang, "Quantum computation and Quantum information", Cambridge university press, Reprint 2016.

References

R1. Parag K Lala, "Quantum Computing, A Beginners Introduction", Mc Graw Hill Education, 2020 (Unit IV and Unit V).

R2. Shani vishal, "Quantum computing", Tata McGraw Hill, 2007 (Unit I and Unit III).

R3. Chris Bernhardt, "Quantum Computing for Everyone", The MIT press, 2019. (Unit I and Unit III).

R4. Chuck Easttom, "Quantum computing fundamentals", Addition Wesley, 2021 (Unit V).

R5. P.Kaye R. Laflamme, and M.Mosca, "An Introduction to Quantum Computing", Oxford University Press, 2006.

Course Code: CSA4728	**Course Title:** Quantum Optimization Techniques and Applications **Type of Course:** Program Core	**L-T- P- C**	3	0	0	3
Version No.	1.0					

Course Pre-requisites	• Basic knowledge of linear algebra (vectors, matrices) and probability concepts. • Understanding of discrete mathematics and fundamental mathematical concepts. • Familiarity with algorithms and data structures is desirable.
Anti-requisites	**NIL**
Course Description	This course introduces the fundamental concepts of quantum computing and their application to optimization problems. It covers the basic principles of quantum mechanics, qubits, quantum gates, and circuit models required to understand quantum computation. The course also provides an overview of classical optimization techniques, including linear, nonlinear, and combinatorial optimization, along with heuristic approaches, to build a strong foundation for comparison with quantum methods. The course further explores quantum optimization techniques, focusing on variational quantum algorithms and their role in solving complex optimization problems. It highlights the application of these techniques in real-world domains such as finance, logistics, machine learning, chemistry, and aerospace. The course aims to equip students with the ability to analyze and apply emerging quantum optimization approaches and understand their advantages over classical methods
Course Objective	• To introduce the fundamental concepts of quantum computing and quantum mechanics relevant to computation. • To understand the mathematical foundations required for quantum computing, including linear algebra and probability. • To provide knowledge of classical optimization techniques such as linear, nonlinear, and combinatorial optimization. • To study heuristic methods used for solving complex optimization problems. • To understand the principles of quantum optimization and variational quantum algorithms. • To analyze and compare classical and quantum approaches for solving optimization problems. • To explore the design and modeling of optimization problems using quantum techniques. • To examine real-world applications of quantum optimization in domains such as finance, logistics, and machine learning.
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1:** Apply fundamental concepts of quantum computing such as qubits, superposition, and quantum gates to represent computational problems. **[Apply]** **CO2:** Apply classical optimization techniques including linear programming, nonlinear optimization, and heuristic methods to solve real-world problems. **[Apply]** **CO3:** Analyze combinatorial optimization problems and evaluate their computational complexity. **[Apply]** **CO4:** Apply quantum optimization algorithms to model and solve optimization problems. **[Apply]** **CO5:** Analyze the working of variational quantum algorithms and compare them with classical optimization approaches. **[Apply]** **CO6:** Evaluate the applicability of quantum optimization techniques in domains such as finance, logistics, machine learning, and scientific computing. **[Evaluate]**
Course Content:	

Module 1	Fundamentals of Quantum Computing - CO1, CO2	8 Sessions
Basic concepts of quantum computing: complex numbers, vectors, matrices, and linear algebra basics. Postulates of quantum mechanics and global phase (conceptual overview). Quantum bits (qubits), representation of qubits, and superposition. Quantum gates and circuits: universal logic gates, basic single-qubit and multi-qubit gates, circuit development, and introduction to quantum error correction.		
Module 2	**Classical Optimization Techniques – CO2, CO3**	**11 Sessions**
Introduction to optimization problems, objective functions, constraints, and classification. Linear Programming: formulation, graphical and simplex methods. Basics of nonlinear optimization and convexity. Combinatorial optimization: TSP, Knapsack, NP-hard problems. Heuristic methods: Genetic Algorithm, Simulated Annealing, Particle Swarm Optimization. Complexity classes: P, NP, NP-hard and limitations of classical approaches.		
Module 3	**Quantum Optimization Algorithms -CO4, CO5**	**11 Sessions**
Introduction to quantum optimization and problem mapping. Variational Quantum Algorithms (VQA) and hybrid optimization. Quantum Approximate Optimization Algorithm (QAOA) and Variational Quantum Eigensolver (VQE). Quantum Annealing and adiabatic optimization. Overview of advanced methods (basics of Warm-start QAOA, MA-QAOA, CVaR-VQE, QRAO, and Pauli Correlation Encoding).		
Module 4	**Applications of Quantum Optimization - CO6**	**12 Sessions**
Applications of quantum optimization in combinatorial problems such as scheduling, routing, and resource allocation. Applications in finance, logistics, and supply chain management for efficient decision-making. Applications in chemistry, drug discovery, and aerospace for modeling and planning. Applications in machine learning, cryptography, and energy systems, with comparison to classical approaches.		

Project work/Assignment:

- **Solve numerical problems on qubit representation, superposition, and quantum gate operations.**
- **Construct and analyze simple quantum circuits using single-qubit and multi-qubit gates.**
- **Formulate and solve Linear Programming and nonlinear optimization problems.**
- **Apply Genetic Algorithm, Simulated Annealing, and Particle Swarm Optimization to optimization problems.**
- **Project: Classical Optimization Techniques for Scheduling, Routing, and Resource Allocation.**
- **Project: Quantum Optimize**

Topics related to

1. Problem Solving: Apply classical and quantum optimization techniques to solve real-world decision-making, scheduling, routing, resource allocation, and combinatorial optimization problems.

2. Employability: Hands-on experience with quantum circuit design, Qiskit programming, optimization modeling, heuristic algorithms, and quantum optimization techniques to enhance skills for careers in quantum computing, data science, AI, and operations research.

Textbook(s):

T1. Parag K. Lala, Mc Graw Hill Education, *"Quantum Computing: A Beginner's Introduction"*, First Edition (1 November 2020).

T2. Chris Bernhardt, The MIT Press, *"Quantum Computing for Everyone"*, Reprint Edition (8 September 2020).

References

1. **R1**. Mykel J. Kochenderfer and Tim A. Wheeler, MIT Press, *"Algorithms for Optimization"*, First Edition, 2019.
2. **R2**. Michael A. Nielsen and Isaac L. Chuang, Cambridge University Press, *"Quantum Computation and Quantum Information"*, Tenth Edition, 2010.

E-Resources:

3. https://github.com/SMU-Quantum/quantum-optimization-algorithms?tab=readme-ov-file
4. https://www.bqpsim.com/quantum-optimization/quantum-optimization-algorithms-guide

Course Code: CSA4729	**Course Title:** Quantum Cryptography and Post-Quantum Security **Course:** Program Core	**L-T- P-C**	2	0	2	3
Version No.	1.0					
Course Prerequisites	**Mathematics, Classical Cryptography, Quantum Computing Basics, Cybersecurity fundamentals**					
Antirequisites	**NIL**					
Course Description	This course introduces the fundamentals of **quantum cryptography** and **post-quantum security**, focusing on the impact of quantum computing on classical cryptographic systems. The theoretical component covers classical cryptography basics, quantum computing concepts (qubits, superposition, entanglement), and key quantum algorithms such as Shor's and Grover's algorithms, along with their implications for security. The course further explores quantum cryptographic protocols, including quantum key distribution (QKD), and post-quantum cryptographic techniques such as lattice-based, hash-based, and code-based methods. The practical component emphasizes hands-on implementation and analysis of cryptographic algorithms, simulation of quantum concepts using software tools, and evaluation of quantum-resistant schemes. Students will gain experience in programming, security analysis, and applying post-quantum techniques to real-world scenarios.					
Course Objective	To understand the fundamentals of classical cryptography, including encryption, hashing, and digital signatures To explain the core principles of quantum computing, such as qubits, superposition, and entanglement To analyse the impact of quantum algorithms (e.g., Shor's and Grover's) on existing cryptographic systems To study quantum cryptographic techniques, including quantum key distribution (QKD) and their security properties To explore post-quantum cryptographic approaches such as lattice-based, hash-based, and code-based methods To develop the ability to evaluate and compare classical and quantum-resistant cryptographic schemes To gain practical skills in implementing and analyzing cryptographic algorithms using appropriate tools and programming languages To understand current challenges and strategies in transitioning to post-quantum secure systems To apply theoretical knowledge to solve real-world security problems in a quantum computing context					
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1:** Recall fundamental concepts of classical cryptography and quantum computing relevant to secure communication systems**.** ***(Remember)*** **CO2:** Identify key components of quantum computing such as qubits, quantum gates, and entanglement. ***(Remember)*** **CO3:** Explain the working principles of classical cryptographic techniques including encryption, hashing, and digital signatures. ***(Understand)*** **CO4:** Describe the impact of quantum algorithms on existing cryptographic systems and security mechanisms. ***(Understand)*** **CO5:** Apply quantum cryptographic protocols such as quantum key distribution (QKD) to solve basic security problems. ***(Apply)*** **CO6:** Apply post-quantum cryptographic techniques to implement simple quantum-resistant solutions. ***(Apply)***					

Course Content:		
Module 1	**Foundations of Cryptography and Security- CO1, CO3**	**6 Sessions +6 Practical**
Introduction to information security and cryptography – classical cryptographic techniques including symmetric and asymmetric encryption – hash functions and digital signatures – public key infrastructure (PKI) – security models and common cryptographic attacks.		
Module 2	**Fundamentals of Quantum Computing - CO1, CO2, CO4**	**8 Sessions +8 Practical**
Introduction to quantum computing – qubits, superposition, and measurement – quantum gates and quantum circuits – entanglement and quantum states – overview of quantum algorithms including Shor's and Grover's algorithms – impact of quantum computing on classical cryptography.		
Module 3	**Quantum Cryptography- CO2, CO4, CO5**	**8 Sessions +8 Practical**
Introduction to quantum cryptography – principles of quantum key distribution (QKD) – BB84 protocol and its variants – security analysis of quantum cryptographic protocols – quantum attacks and eavesdropping models – practical challenges in quantum		
Module 4	**Post-Quantum Cryptography CO3, CO5, CO6**	**8 Sessions +8 Practical**
Introduction to post-quantum cryptography – need for quantum-resistant algorithms – lattice-based cryptography – hash-based cryptography – code-based and multivariate cryptography – post-quantum standardization efforts (NIST) – migration strategies for quantum-safe systems.		

Project work/Assignment:

Study and analysis of classical cryptographic algorithms and their vulnerabilities in the presence of quantum computing
Implementation of basic cryptographic techniques such as encryption, hashing, and digital signatures using programming tools
Case study on the impact of quantum algorithms (Shor's and Grover's) on existing security systems
Design and simulation of quantum key distribution (QKD) protocols
Comparative analysis of post-quantum cryptographic algorithms based on security and performance
Mini-project on developing a simple quantum-resistant secure communication model

Laboratory Experiments

Experiment No. 1
Level 1: Implement classical encryption algorithms such as AES and RSA.
Level 2: Implement SHA family hash functions.
Experiment No. 2
Level 1: Implement digital signature generation and verification.
Level 2: Demonstrate basic quantum states using Qiskit.
Experiment No. 3
Level 1: Implement quantum gates and simple quantum circuits.
Level 2: Demonstrate quantum superposition and measurement concepts.
Experiment No. 4
Level 1: Demonstrate entanglement in quantum systems.
Level 2: Implement Grover's search algorithm simulation.
Experiment No. 5
Level 1: Illustrate the working of Shor's algorithm through simulation.
Level 2: Implement the BB84 quantum key distribution protocol.
Experiment No. 6
Level 1: Analyze quantum attacks on classical cryptographic systems.
Level 2: Implement a basic lattice-based cryptographic scheme.
Experiment No. 7
Level 1: Implement a hash-based signature scheme.
Level 2: Analyze the performance of classical and post-quantum cryptographic algorithms.
Experiment No. 8
Level 1: Demonstrate a simple quantum-safe cryptographic application design.

Level 2: Design and evaluate a quantum-safe cryptographic application.
Experiment No. 9
Level 1: Implement AES encryption and decryption operations.
Level 2: Implement RSA key generation and encryption operations.
Experiment No. 10
Level 1: Implement SHA-256 hash generation.
Level 2: Implement SHA-512 hash generation.
Experiment No. 11
Level 1: Implement digital signature generation using cryptographic keys.
Level 2: Implement digital signature verification and integrity checking.
Experiment No. 12
Level 1: Demonstrate single-qubit quantum state simulation.
Level 2: Implement quantum gates on qubits.
Experiment No. 13
Level 1: Demonstrate quantum superposition using Hadamard gates.
Level 2: Illustrate quantum measurement outcomes.
Experiment No. 14
Level 1: Demonstrate Bell states and quantum entanglement.
Level 2: Implement a basic simulation of Grover's search algorithm.
Experiment No. 15
Level 1: Illustrate the concepts of Shor's algorithm and BB84 protocol.
Level 2: Design a simple quantum-safe cryptographic mini project.

Topics related to
Problem Solving: Ability to analyze complex security problems and develop effective cryptographic solutions.
Employability: Enhances job readiness by building skills relevant to cybersecurity and emerging quantum technologies.
Critical Thinking: Enables evaluation of cryptographic methods and identification of potential vulnerabilities.
Analytical Skills: Develops the ability to interpret data, algorithms, and security models systematically.
Computational Thinking: Promotes logical reasoning and algorithmic approaches to solving security challenges.
Research and Innovation: Encourages exploration of new quantum-resistant techniques and emerging security models.
Design Thinking: Supports the creation of user-centric and secure cryptographic systems.
Technical Proficiency: Builds hands-on expertise in implementing cryptographic and quantum computing concepts.
Cybersecurity Awareness: Provides understanding of modern security threats and protection mechanisms.
Ethical and Professional Responsibility: Emphasizes ethical practices in handling sensitive data and security systems.
Communication Skills: Improves the ability to present technical concepts clearly and effectively.
Teamwork and Collaboration: Encourages working effectively in teams on security and research projects.
Lifelong Learning: Promotes continuous learning to keep up with evolving quantum and cybersecurity technologies.
Adaptability to Emerging Technologies: Prepares students to work with rapidly advancing quantum computing innovations.
Security and Risk Assessment: Develops skills to evaluate and mitigate risks in cryptographic systems.

Textbook(s):
T1 - Introduction to Quantum Cryptography – Provides a clear introduction to quantum cryptography concepts, including QKD, with examples and exercises suitable for undergraduate and postgraduate courses.
T2- Post-Quantum Cryptography – Covers core post-quantum algorithms such as lattice-based, hash-based, and code-based cryptography, widely used as a reference text in universities.
T3- Physical-Layer Security, Quantum Key Distribution, and Post-Quantum Cryptography – Integrates classical cryptography, QKD, and PQC with modern communication security applications.
T4-Mathematical Foundations for Post-Quantum Cryptography – Focuses on the mathematical foundations required for advanced post-quantum cryptographic systems.

References
R1. Quantum Cryptography and Annealing for Securing Industrial IoT – Explores practical applications of quantum cryptography in modern systems like IoT.
R2 Mathematical Foundations of Post-Quantum Cryptography – Covers algebra, number theory, and probability foundations essential for PQC learning.

E-Resources:
"NIST Post-Quantum Cryptography Project"
https://csrc.nist.gov/Projects/Post-Quantum-Cryptography

"NIST PQC Program Overview" https://www.nist.gov/programs-projects/post-quantum-cryptography "Post-Quantum Cryptography Visualization Tool" https://pqc.nithinram.com/ "PQC NOW Platform" https://pqcnow.com/ "Quantum Security Defence – PQC Learning Resources" https://www.quantumsecuritydefence.com/pqc

Course Code: **CSA4730**	**Course Title: Advanced Quantum Algorithms** **Type of Course: Program Core**	**L-T- P- C**	2	0	2	3
Version No.	1.0					
Course Pre-requisites	**Quantum Algorithms**					
Anti-requisites	**NIL**					
Course Description	This course explores advanced quantum algorithms beyond foundational concepts, focusing on both theory and practical implementation. It covers Quantum Fourier Transform, Phase Estimation, quantum walks, and hidden subgroup frameworks. Students study number-theoretic algorithms such as Shor's algorithm and discrete logarithms, along with quantum simulation and linear system solvers like HHL. The course emphasizes variational and NISQ algorithms including VQE and QAOA, integrating hybrid quantum-classical approaches. Hands-on labs use modern tools like Qiskit and Pennylane, preparing students for research and real-world applications in optimization, machine learning, and scientific computing.					
Course Objective	**The objective of the course is EMPLOYBILITY of student by using PARTICIPATIVE LEARNING techniques.**					
Course Outcomes	**On successful completion of this course, the students shall be able to:** **CO1:** Apply advanced quantum algorithmic techniques such as Quantum Fourier Transform and Phase Estimation to solve complex computational problems. **(Apply)** **CO2:** Analyze number-theoretic and algebraic quantum algorithms including Shor's algorithm and discrete logarithms. **(Analyze)** **CO3:** Apply quantum simulation methods and linear system solvers like the HHL algorithm for scientific computations. **(Apply)** **CO4:** Analyze variational quantum algorithms such as VQE and QAOA in NISQ environments**. (Analyze)** **CO5:** Apply hybrid quantum-classical approaches for optimization and quantum machine learning tasks. **(Apply)** **CO6:** Design and evaluate research-oriented quantum solutions for real-world applications**. (Create)**					
Course Content:						
Module 1	**Quantum Algorithmic Foundations – CO1**					**12 Sessions (6T + 6P)**
Advanced Quantum Fourier Transform (QFT): mathematical formulation, circuit optimization, and applications. Phase Kickback: principle and role in algorithm design. Quantum Phase Estimation (QPE): detailed working, eigenvalue extraction, and accuracy analysis. Quantum Walk Algorithms: discrete and continuous-time models, search applications. Hidden Subgroup Problem (HSP): general framework, relation to quantum speedups, complexity analysis and limitations in practical systems.						
Module 2	**Number-Theoretic and Advanced Algorithms – CO2**					**12 Sessions (6T + 6P)**
Shor's Algorithm: full architecture, modular exponentiation, period finding, and complexity advantages over classical						

methods. Order Finding: mathematical basis and implementation. Discrete Logarithm Problem: quantum approach and cryptographic implications. Applications of Hidden Subgroup Problem in algebraic structures. Quantum Monte Carlo Methods: amplitude estimation, probabilistic sampling, and applications in finance and optimization.		
Module 3	**Quantum Simulation and Linear Systems – CO3**	**12 Sessions (6T + 6P)**
Hamiltonian Simulation: digital and analog approaches, sparse Hamiltonians, and real-world applications. Trotterization Techniques: product formulas, error bounds, and optimization strategies. Variational Simulation Methods: parameterized circuits for dynamic systems. HHL Algorithm: solving linear systems, assumptions, complexity, and limitations. Introduction to hybrid quantum-classical computation for simulation and numerical problem solving.		
Module 4	**Variational, NISQ and Quantum Machine Learning (QML) – COFml4, CO5**	**12 Sessions (6T + 6P)**
Variational Quantum Eigensolver (VQE): ansatz design, cost functions, and optimization challenges. Quantum Approximate Optimization Algorithm (QAOA): combinatorial optimization and parameter tuning. Quantum Machine Learning (QML): data encoding (amplitude, angle), quantum feature maps, variational quantum classifiers, quantum neural networks. Error Mitigation: noise handling in NISQ devices. Hybrid Quantum-Classical Frameworks: integration strategies and real-world applications in optimization and learning systems.		
Module 5	**Quantum Error Correction and Fault-Tolerant Computing– CO5, CO6**	**12 Sessions (6T + 6P)**
Introduction to quantum noise and decoherence, error models (bit-flip, phase-flip, depolarizing). Quantum Error Correction Codes: Shor code, Steane code, surface codes. Fault-Tolerant Quantum Computing: threshold theorem, logical qubits, error-resilient circuit design. Syndrome measurement and recovery operations. Overview of scalable quantum architectures and challenges in building fault-tolerant quantum systems. Applications and relevance in large-scale quantum algorithms.		

Experiment No. 1
Level 1: Implement basic single-qubit gates (X, Y, Z, H) using a quantum simulator.
Level 2: Design multi-qubit circuits to demonstrate superposition and entanglement.

Experiment No. 2
Level 1: Implement measurement operations and visualize quantum state probabilities.
Level 2: Analyze Bloch sphere representation of qubit states.

Experiment No. 3
Level 1: Implement Quantum Fourier Transform (QFT) circuit.
Level 2: Apply QFT for simple signal/phase analysis problems.

Experiment No. 4
Level 1: Demonstrate **Phase** Kickback using controlled gates.
Level 2: Analyze its role in quantum algorithms.

Experiment No. 5
Level 1: Implement basic Quantum Phase Estimation (QPE).
Level 2: Develop full QPE for eigenvalue estimation.

Experiment No. 6
Level 1: Simulate Quantum Walk (discrete-time).
Level 2: Apply quantum walk for search problem simulation.

Experiment No. 7
Level 1: Construct oracle-based circuits.
Level 2: Demonstrate Hidden Subgroup Problem concepts.

Experiment No. 8
Level 1: Implement Shor's Algorithm for small integers.

Level 2: Perform Order Finding using quantum circuits.

Experiment No. 9
Level 1: Simulate Discrete Logarithm Problem using quantum approach.
Level 2: Analyze cryptographic implications of quantum algorithms.

Experiment No. 10
Level 1: Implement Hamiltonian Simulation for simple systems.
Level 2: Apply Trotterization techniques and analyze error bounds.

Experiment No. 11
Level 1: Implement Variational Simulation using parameterized circuits.
Level 2: Optimize parameters using classical optimizers.

Experiment No. 12
Level 1: Implement HHL Algorithm for solving linear equations.
Level 2: Analyze performance and limitations.

Experiment No. 13
Level 1: Perform Quantum Monte Carlo simulation.
Level 2: Apply amplitude estimation for probabilistic analysis.

Experiment No. 14
Level 1: Implement Variational Quantum Eigensolver (VQE).
Level 2: Apply VQE for ground state energy estimation.

Experiment No. 15
Level 1: Implement QAOA for optimization problems.
Level 2: Develop a Quantum Machine Learning (QML) model such as a quantum classifier.

Targeted Application & Tools that can be used:

Qiskit, Pennylane, Cirq, IBM Quantum Experience, Google Cirq Simulator, Python (NumPy, SciPy), Jupyter Notebook, Matplotlib, TensorFlow Quantum.

Project work/Assignment:

1. Design and implement a complete Quantum Phase Estimation or QFT-based application using Qiskit or Pennylane, with performance analysis.
2. Develop a simulation of Shor's Algorithm for integer factorization and analyze its impact on classical cryptographic systems.
3. Implement Hamiltonian simulation or HHL algorithm for solving real-world scientific or engineering problems.
4. Design and optimize a Variational Quantum Eigensolver (VQE) or QAOA model for solving optimization problems.
5. Develop a Quantum Machine Learning model such as a quantum classifier or regressor using hybrid quantum-classical techniques and evaluate its performance.

Topics related to

1. Problem Solving: Implement advanced quantum algorithms (QFT, QPE, Shor's Algorithm) and simulate quantum Circuits for real-world applications in optimization, linear systems, and cryptography.
2. Employability: Apply variational algorithms (VQE, QAOA) and Quantum Machine Learning models using hybrid Quantum-classical frameworks and quantum simulators to develop industry-relevant quantum computing skills.

Textbook(s):

T1. Nielsen, Michael A., and Isaac L. Chuang. Quantum Computation and Quantum Information: 10th Anniversary Edition. Cambridge University Press, 2021.

T2. Schuld, Maria, and Francesco Petruccione. Machine Learning with Quantum Computers. Springer, 2021.

T3. Cerezo, M., et al. "Variational quantum algorithms." Nature Reviews Physics 3, no. 9 (2021): 625–644.

References

R1. Preskill, John. "Quantum Computing in the NISQ Era and beyond." Quantum 2 (2018): 79. (Updated 2021)

R2. Hidary, Jack D. Quantum Computing: An Applied Approach. 2nd ed. Springer, 2021.

R3. Thé, Jesse Van Griensven, Roydon Andrew Fraser, and Jose Rosas Bustos. Quantum Computing and Quantum Machine Learning for Engineers and Developers. Springer, 2025.

E-References

E1. IBM Quantum Documentation: https://qiskit.org

E2. Microsoft Quantum Documentation — https://learn.microsoft.com/quantum/

E3. Xanadu Quantum Learning Resources — https://docs.xanadu.ai/