NATIONAL INSTITUTE OF ELECTRONICS AND INFORMATION TECHNOLOGY

DEEMED TO BE UNIVERSITY UNDER DISTINCT CATEGORY

(AN AUTONOMOUS INSTITUTION UNDER MINISTRY OF ELECTRONICS & IT, GOVT. OF INDIA)

Curriculum and Syllabi for

Master of Computer Applications (MCA)

(2026-27 Scheme)

Main Campus at Ropar and Constituent Units at Aizawl, Agartala, Aurangabad, Calicut, Gorakhpur, Imphal, Itanagar, Kekri, Kohima, Patna and Srinagar

Vision

To be a leading center of excellence in computer applications education and research, fostering innovation, digital transformation, and sustainable development through technology.

Mission

1. To provide comprehensive education in computer applications with a strong foundation in theoretical and practical knowledge.
2. To develop problem-solving, analytical, and programming skills relevant to industry, academia, and entrepreneurship.
3. To foster ethical practices, social responsibility, and effective communication among computing professionals.
4. To enable students to adapt to emerging technologies through lifelong learning and advanced education.
5. To contribute to the national and global digital ecosystem.

Program Education Objectives

Graduates of the MCA program will:
1. Apply computing principles and domain knowledge to design and develop effective software solutions for real-world problems.
2. Succeed in IT careers, research, or higher studies through strong technical and interpersonal competencies.
3. Uphold ethical, legal, and professional standards while working in multidisciplinary teams.
4. Engage in continuous learning to adapt to the evolving landscape of technology and industry needs.
5. Contribute to societal progress through innovative, inclusive, and sustainable technological solutions.

Program Outcomes

Upon successful completion of the MCA program, students will be able to:
1. Apply Computing Knowledge: Use knowledge of computer science and mathematics to analyze and solve software-related problems.
2. Design and Develop Applications: Develop efficient software systems using modern tools, techniques, and methodologies.
3. Analyze and Interpret Data: Apply data structures, algorithms, and analytics to extract insights and support decision-making.
4. Demonstrate Ethics and Communication: Exhibit professional ethics, communication skills, and teamwork in diverse environments.
5. Pursue Lifelong Learning: Engage in lifelong learning to stay updated with emerging technologies and best practices.

Program Specific Outcomes

MCA graduates from NIELIT Deemed to be University will be able to:
1. Build secure, scalable, and efficient applications using modern programming languages and frameworks.
2. Apply database, networking, and cloud computing concepts in designing enterprise-level IT solutions.
3. Integrate emerging technologies such as AI, IoT, blockchain, or data science in solving domain-specific challenges.

CATEGORY WISE CREDIT DISTRIBUTION

Category	Category Code	Credits
Audit Courses (without grade or credit)	AU	0
Open Elective Courses	OE	8
Professional Elective Courses	PE	16
Program Core Courses	PC	20
Skill / Employment Enhancement Courses (Project/Internship/Seminar/Dissertation/Research)	EE	36
Total Credits (Common track)		**80**

EVALUATION AND ASSESSMENT

1. Performance of a student in a semester shall be evaluated through continuous Class assessment, Tutorial/Lab assessment, Mid-Semester Examination (MSE) and End-Semester Examination (ESE). Both the MSE and ESE shall be the University examination and will be conducted as notified by the Controller of Examinations (CoE) of the University.

2. The continuous assessment shall be based on assignments, tutorials, paper presentation / quizzes/ viva-voce / flipped classes, lab work / projects / fieldwork and attendance, etc.

3. The MSE/ESE shall be comprising of written papers.

4. The overall assessment of the students will be done as per the following scheme:

S. No.	Assessment Type	Marks Weightage
1.	Mid Semester Examination	20
2.	End Semester Examination	40
3.	Continuous Assessment - Practical /Lab/ Tutorial	25
4.	Continuous Assessment - Theory	15
Total		**100**

For more details, please refer the applicable ordinance for this programme and Assessment SOPs.

CURRICULUM

Semester 1						
Semester Code	**Course Code**	**Course Title**	**L**	**T**	**P**	**Cr**
PC-MCAAPP-101	DASH250072	Mathematical Foundations of Computer Science	3	1	0	4
PC-MCAAPP-102	DCSE250113	Programming in Python	2	0	4	4
PC-MCAAPP-103	DCSE250006	Advanced Database Management Systems	3	0	2	4
PE-MCAAPP-104	Elective	Program Elective				4
OE-MCAAPP-105	Elective	Open Elective				4
AU-MCAAPP-106	Elective	Audit Elective				0

Semester 2						
Semester Code	**Course Code**	**Course Title**	**L**	**T**	**P**	**Cr**
PC-MCAAPP-201	DCSE250004	Advanced Data Structures and Algorithms	3	0	2	4
PC-MCAAPP-202	DCSA250071	Software Engineering	3	1	0	4
PE-MCAAPP-203	Elective	Program Elective				4
PE-MCAAPP-204	Elective	Program Elective				4
EE-MCAAPP-205	EE	Internship/Seminar/Mini project				4
AU-MCAAPP-206	Elective	Audit Elective				0

Semester 3						
Semester Code	**Course Code**	**Course Title**	**L**	**T**	**P**	**Cr**
PE-MCAAPP-301	Elective	Program Elective				4
OE-MCAAPP-302	Elective	Open Elective				4
EE-MCAAPP-303	EE	Dissertation-I/Industrial Project				12

Semester 4						
Semester Code	**Course Code**	**Course Title**	**L**	**T**	**P**	**Cr**
EE-MCAAPP-401	EE	Dissertation-II				20

Elective Courses

Elective Courses for PE Category						
Course Code	**Course Title**	**L**	**T**	**P**	**Cr**	**For Sem./Code**
DOAI260001	AI (Industry)-Applied Ethics for AI	3	0	2	4	
DOAI260002	AI (Industry)-Introduction to Artificial Intelligence	3	0	2	4	
DOAI260003	AI (Industry)-Introduction to Generative AI	3	0	2	4	
DOAI260004	AI (Industry)-Introduction to Machine Learning	3	0	2	4	
DASH250095	Research Methodology and IPR	3	1	0	4	
DCSA250003	Mobile Application Development using Android	2	1	2	4	
DCSA250031	Full Stack Web Development	3	0	2	4	
DCSA250059	Object-Oriented Programming with Java	2	0	4	4	
DCSA250084	Web Technologies	3	0	2	4	
DCSE250003	OS Principles and Applied Linux	2	0	4	4	
DCSE250010	Design and Analysis of Algorithms	3	0	2	4	
DCSE250020	Cloud Computing	3	0	2	4	
DCSE250029	Computer Graphics	3	1	0	4	
DCSE250035	Computer Networks	3	0	2	4	
DCSE250043	Cryptography and Network Security	3	0	2	4	
DCSE250081	Fuzzy Logic and Fuzzy Sets	3	1	0	4	
DCSE250099	Object Oriented Analysis and Design	3	1	0	4	
DOAI250009	Big Data Analytics	3	0	2	4	
DOAI250015	Data Analytics and Visualization	2	0	4	4	
DOAI250019	Data Mining and Warehousing	3	0	2	4	
DOAI250030	Introduction to Artificial Intelligence	3	1	0	4	

Elective Courses for OE Category
Students may opt courses under MOOC, NPTEL, SWAYAM, NSQF/NCVET aligned courses or other relevant technical subjects not included in their core curriculum. The exercise of option shall be subject to the Course Schedule of the respective Constituent Unit, the credit requirements and availability of seats. Guidelines for choosing MOOC/Online courses are given separately and shall have to be adhered to.

Elective Courses for AU (Audit) Category					
Course Code	**Course Title**	**L**	**T**	**P**	**Cr**
DASH240027	Disaster Management	2	0	0	0
DASH240047	English for Research Paper Writing	2	0	0	0
DASH240052	Environmental Science	3	0	0	0
DASH240085	Pedagogy Studies	2	0	0	0
DASH240086	Personality Development through Life Enlightenment Skills	2	0	0	0
DASH240097	Sanskrit for Technical Knowledge	2	0	0	0
DASH240103	Stress Management by Yoga	2	0	0	0

DASH240104	Technical Presentation and Report Writing	0	0	2	0
DASH240106	Value Education	2	0	0	0
DASH240124	Constitution of India - AU	2	0	0	0
DASH250023	Constitution of India	2	1	0	0
DASH250027	Disaster Management	3	0	0	0
DASH250034	Energy and Environmental Engineering	3	0	0	0
DASH250047	English for Research Paper Writing	2	0	0	0
DASH250054	Essence of Indian Knowledge and Tradition	2	0	0	0
DASH250085	Pedagogy Studies	2	0	0	0
DASH250086	Personality Development	2	0	0	0
DASH250097	Sanskrit for Technical Knowledge	3	0	0	0
DASH250103	Stress Management by Yoga	3	0	0	0
DASH250106	Value Education	2	0	0	0
DASH250117	Innovative Thinking and Entrepreneurship Skills	1	0	0	0
DASH250118	Intellectual Property Rights	2	0	0	0
DCSA240080	Training on Computer Networking	0	0	2	0
DCSA240304	Lab Based on Statistical Package	0	0	2	0
Any other Scheduled Course as per the Course Schedule of the respective Constituent Unit can also be chosen, subject to availability of seats. However, there shall be no assessment and grading for the course chosen as Audit Course.					

The choice of all type of Electives available shall be as per the Course Schedule of the respective Constituent Unit.

Guidelines for Massive Open Online Courses (MOOC) / approved Online Courses

1. Relevant Swayam, MOOC, NPTEL, NIELIT NSQF and any other online courses approved by UGC/AICTE shall also be allowed as Open Electives(OE).

2. One credit of MOOC course should be minimum of 30 hours.

3. A student can choose any approved online course as Open Elective, subject to minimum credit requirements.

4. The students willing to opt OE under MOOC in their next semester, will submit their request to the MOOC Coordinator at their respective campus.

5. Once approved, the campus shall display the list of accepted courses.

6. The student has to submit certificate issued by the institution offering the MOOCs course, along with the number of credits and grades, to get credits transferred into his/her marks certificate issued by NDU.

7. The transfer of credits shall be as adopted by NDU.

Detailed Syllabus

Course Code	Course Name	L	T	P	Cr
DASH250072	Mathematical Foundations of Computer Science	3	1	0	4

Course Outcomes:

CO1:Understand and apply the foundational concepts of probability, distributions, and statistical inference essential for machine learning and data analysis.
CO2:Analyze and model stochastic systems using Markov chains and steady-state analysis in applications like simulation and AI.
CO3:Perform estimation and hypothesis testing using both classical and modern statistical techniques, including z-tests, t-tests, and chi-square tests.
CO4:Design and evaluate multivariate models, including regression and classification, and assess their performance using appropriate metrics and validation techniques.
CO5:Apply graph theory concepts to analyze graphs and solve combinatorial problems using fundamental counting and logical principles.

Module 1:

Basics of probability (9 hrs)
Sample space, events, conditional probability, Bayes' theorem, Random variables: discrete and continuous, Probability mass function (PMF), probability density function (PDF), and cumulative distribution function (CDF), Discrete distributions: Bernoulli, Binomial, Geometric, Poisson, Continuous distributions: Uniform, Normal, Exponential, Expectation, variance, covariance, moment-generating functions, Applications of probability in computer science.

Module 2:

Central Limit Theorem and Law of Large Numbers, Markov chains (9 hrs)
Markov property, transition matrix, steady-state analysis, Sampling techniques and distributions, Estimation methods: Point and interval estimation, Method of Moments, Maximum Likelihood Estimation (MLE), Applications in simulation, AI, and modeling.

Module 3:

Statistical inference (9 hrs)
Theory of estimation, Hypothesis testing: null and alternative hypotheses, types of errors (Type I & II), p-value, significance level, one-tailed and two-tailed tests, z-test, t-test, chi-square test.

Module 4:

Introduction to multivariate statistical models (9 hrs)
Regression models: linear and logistic, Classification techniques, principal components analysis, Overfitting and underfitting, evaluation metrics, Model validation: cross-validation, confusion matrix, ROC curve, AUC.

Module 5:

Graph Theory (9 hrs)
Graph fundamentals: graphs, subgraphs, directed and undirected graphs, multigraphs, weighted graphs, Paths, walks, cycles, connected components, Eulerian and Hamiltonian graphs, Planar graphs and Kuratowski's Theorem, Graph coloring, isomorphism, Permutations and combinations (with/without repetition), Pigeonhole principle, inclusion-exclusion principle.

Labs / Practicals:

N/A

References:

1. N.G. Das – Statistical Methods (Volumes I & II combined), McGraw Hill Education India
2. T. Veerarajan, Probability, Statistics and Random Processes, Tata McGraw Hill.
3. S. C. Gupta and V. K. Kapoor, Fundamentals of Mathematical Statistics, Sultan Chand & Sons.
4. K. Trivedi, Probability and Statistics with Reliability, Queuing, and Computer Science Applications, Wiley.
5. M. Mitzenmacher and E. Upfal, Probability and Computing: Randomized Algorithms and Probabilistic Analysis, Cambridge University Press.
6. Narsingh Deo, Graph Theory with Applications to Engineering and Computer Science, Prentice Hall of India.
7. B. S. Grewal, Higher Engineering Mathematics, Khanna Publishers.
8. John Vince, Foundation Mathematics for Computer Science, Springer.
9. Alan Tucker, Applied Combinatorics, Wiley.

Course Code	Course Name	L	T	P	Cr
DCSE250113	Programming in Python	2	0	4	4

Course Outcomes:

CO1: Remember syntax rules, basic operators, and built-in features of Python 3.

CO2: Understand guiding principles and coding conventions from Zen of Python.

CO3: Apply object-oriented programming paradigms in design and practice.

CO4: Analyze the runtime efficiency and resource utilization of Python programs.

CO5: Evaluate the outcomes of programs manually with dry runs and trace tables.

CO6: Create modular and maintainable programs following Pythonic practices.

Module 1:

Constants, types, identifiers, keywords, comments, print, input, help functions.
Operator precedence and associativity, arithmetic operators (**, *, /, //, %, +, -).
Binary number system, bin function, bitwise operators (shift, &, ^, |), bit masking.
Relational and equality operators (== vs "is"), not, short-circuit evaluation (and, or).

Module 2:

Selection: if, elif, else, mutual exclusion, indent (space vs tab), nesting, pass.
Iteration: while loop, range of integers, in operator, for loop, break vs continue.
Function: parameters, arguments, return value, global, recursion, lambda, closure.
Modules: from, import, as, __name__, top-level module, avoiding circular imports.

Module 3:

ASCII and UTF-8, ord and chr functions, escape sequence, strings, docstring, regex.
Mutability, id function, object vs reference, is vs ==, interning, datetime module.
Ordered: list, tuple, negative indexing, slicing, sort with key, pack and unpack.
Unordered: set, dictionary (keys, values, items), subscript vs get method, None.

Module 4:

Class definition, self parameter in methods, constructor, attributes, instantiation.
Dot notation, type function, __str__ vs __repr__, methods for built-in operators.
Static attributes, annotations, private name mangling, getter and setter methods.
Inheritance, object, super, C3 linearization, ambiguous MRO, abstract base class.

Module 5:

Syntax errors vs runtime exceptions, traceback, try, except, else, finally, raise.
Matching except clause, except with multiple exceptions, finally with return.
Built-in exceptions, user-defined exception classes, exceptions with arguments, as.
Command-line arguments, file opening modes (read/write/append), with statement.

Labs / Practicals:

01. Test the precedence and associativity of arithmetic operators.

02. Test the bitwise operators and verify the results with bin function.

03. Test the relational operators (with and without chaining).

04. Test the short-circuit evaluation of and, or operators using function calls as operands.

05. Convert from metric prefix (KB/MB/GB/TB) to binary prefix (KiB/MiB/GiB/TiB).

06. Use if-elif-else ladder to print the grade as per the score obtained in a subject.

07. Find the mean, median, mode, variance, and standard deviation for a list of scores.

08. Print values using additive/subtractive rules of roman numerals: I, V, X, L, C, D, M.

09. Use string repetition with multiplication to generate patterns without nested loops.

10. Use nested loops to generate asymmetric and symmetric patterns of arbitrary size.

11. Check whether a given string is a palindrome (or not) without reversing the string.

12. Test positional arguments, default arguments, and keyword arguments for functions.

13. Write a function to check whether an indexed collection of numbers is already sorted.

14. Write a function for primality testing, and use it to find first n pairs of twin primes.

15. Write a function to generate the Collatz sequence (up to 1) for a positive integer.

16. Generate the terms of Fibonacci sequence using iterative and recursive approaches.

17. Print optimal sequence of moves for a given instance of “Towers of Hanoi” puzzle.

18. Find all combinations to make a given payment from a limited set of coin types.

19. Check if a letter is a vowel using two approaches: if-elif-else and in operator.

20. Find vowel count in a string using a lookup table for case-insensitive vowel check.

21. Write a regular expression to check whether a string is a valid email ID (or not).

22. Use case-insensitive and dot-insensitive matching to check if two email IDs are same.

23. Display a countdown timer in HH:MM:SS format using carriage return ("\r").

24. Get current time and find clockwise angles of hour hand, minute hand, second hand.

25. Find the day of week for any date in Gregorian calendar using doomsday algorithm.

26. Define Matrix class and implement these operators: unary +, unary -, +, -, *, ==.

27. Define Person class in a module and extend it with Student class in another module.

28. Incrementally generate roll numbers in Student constructor using a static counter.

29. Populate a list with Student instances and write their attributes in a new CSV file.

References:

1. "Python Tutorial" by Guido van Rossum and the Python development team
https://bugs.python.org/file47781/Tutorial_EDIT.pdf

2. "Think Python: How to Think Like a Computer Scientist (Second Edition)" by Allen Benjamin Downey
https://greenteapress.com/thinkpython2/thinkpython2.pdf

3. "Python for Everybody: Exploring Data in Python 3" by Charles Russell Severance
https://do1.dr-chuck.com/pythonlearn/EN_us/pythonlearn.pdf

4. "Beej's Guide to Python Programming" by Brian Hall
https://beej.us/guide/bgpython/pdf/bgpython_a4_c_2.pdf

5. "Learning Python: Powerful Object-Oriented Programming (Sixth Edition)" by Mark Lutz
https://learning-python.com

6. "Fluent Python: Clear, Concise, and Effective Programming (Second Edition)" by Luciano Ramalho
https://www.fluentpython.com

Course Code	Course Name	L	T	P	Cr
DCSE250006	Advanced Database Management Systems	3	0	2	4

Course Outcomes:

CO1: Explain database concepts, architectures, data models and distributed database.
CO2: Design relational databases using ER/EER models and normalization techniques.
CO3: Implement and manage databases using SQL and transaction processing.

Module 1:

Introduction: Databases and Database System Environment, Characteristics of Database Approach, Advantages of using DBMS approach, Categories of Data models, schemas and instances; Three-schema architecture and data independence.
Data Modeling using the ER Model: Database Design using High-Level Conceptual Data Models, Entity Types, Entity Sets, Attributes and Keys; Relationship types, Relationship Sets, Roles and Structural Constraints; Weak Entity Types; ER Diagrams, Enhanced Entity-Relationship (EER) Model.

Module 2:

Relational Model and Relational Algebra: Relational Model Concepts; Relational Model Constraints and Relational Database Schemas; Relational Operations- SELECT and PROJECT; Relational Algebra Operations from Set Theory; Binary Relational Operations - JOIN and DIVISION, Relational Database Design by ER- and EER-to-Relational Mapping.

Functional Dependencies and Normalization: Functional Dependencies; Normal Forms Based on Primary Keys; General Definitions of Second and Third Normal Forms; Boyce-Codd Normal Form, Algorithms for Relational Database Schema Design; Multivalued Dependencies and Fourth Normal Form; Join Dependencies and Fifth Normal Form.

Module 3:

SQL: Data Definition and Data Types; Specifying basic constraints in SQL; Schema change statements in SQL; Basic queries in SQL; More complex SQL Queries. Insert, Delete and Update statements in SQL; Views (Virtual Tables) in SQL; Embedded SQL, Dynamic SQL; Database stored procedures

Module 4:

Distributed Database and NOSQL: Distributed Database Concepts, Data Fragmentation, Replication, and Allocation Techniques for Distributed Database Design, Overview of Concurrency Control and Recovery in Distributed Databases
Big Data Technologies Based on MapReduce and Hadoop: Big Data, Introduction to MapReduce and Hadoop, Hadoop Distributed File System, MapReduce: Additional Details, Hadoop v2 alias YARN

Module 5:

Transaction Processing: Read and Write Operations, Transaction states, ACID Properties; Schedules and Characterizing Schedules, Serializability and Testing for Conflict Serializability, Lock- Based Concurrency Control: Types of Locks, Serializability by 2-Phase Locking, Dealing with Deadlock and starvation.
Database Recovery: Recovery concepts and techniques, NO-UNDO/REDO Recovery Based on Deferred Update, Recovery Techniques Based on Immediate Update, Shadow paging, ARIES recovery algorithm.

Labs / Practicals:

Preparing

References:

1.Elmasri & S. Navathe, Fundamentals of Database Systems, 7th Ed., Addison-Wesley.
2.A. Silberschatz, H.F. Korth and S. Sudarshan, Database System Concepts, 5th Edition, TMH.
3.C.J. Date, An Introduction to Database Systems, 8th Edition, Pearson.
4.R. Ramakrishnan and Johannes Gehrke, Database Management Systems, 3rd Ed., McGraw-Hill.

Course Code	Course Name	L	T	P	Cr
DCSE250004	Advanced Data Structures and Algorithms	3	0	2	4

Course Outcomes:

CO1: Analyze the performance of basic data structures and sorting/searching techniques.
CO2: Implement and apply advanced trees, heaps, and hash tables to solve complex problems.
CO3: Solve problems using graph algorithms and greedy strategies.
CO4: Design algorithms using divide and conquer, backtracking, and dynamic programming.
CO5: Develop efficient and scalable solutions using appropriate algorithmic techniques.

Module 1:

Review of Fundamental Concepts
Pointers and Dynamic Memory Allocation, Arrays, Dynamically Allocated Arrays, Structures, Algorithm Specification, Data Abstraction, Performance Analysis, Performance Measurement, Time and space complexity

Module 2:

Basic Data Structures and Advanced Trees
Stacks, Queues, Linked Lists, Hash Tables, Searching and Sorting Techniques: Linear Search, Bubble Sort, Selection Sort, Insertion Sort, Binary Trees, Binary Tree Traversals, Applications of Binary Trees, Threaded Binary Trees, Binary Search Trees, Advanced Trees: AVL Trees: Rotations, insertion and deletion, Red-Black Trees, Splay Trees, B Trees, B+ Trees, Binary Heap: Min/Max heap operations, Priority Queue applications.

Module 3:

Divide and Conquer, Backtracking, String Matching algorithms
Merge Sort, Quick Sort, Binary Search, Backtracking: N-Queens, Graph Coloring, String matching algorithms: KMP, Rabin-Karp

Module 4:

Graph Algorithms
Graph Representations: Adjacency list, Adjacency Matrix, BFS, DFS, Shortest Path: Dijkstra’s, Greedy algorithms: Activity Selection, Huffman Coding Minimum Spanning Tree: Prim’s and Kruskal’s Algorithms

Module 5:

Dynamic Programming
Basic concepts: Overlapping subproblems, optimal substructure, 0/1 Knapsack, Longest Common Subsequence, Matrix Chain Multiplication

Labs / Practicals:

Preparing

References:

1. "Introduction to Algorithms (Fourth Edition)" by Thomas H. Cormen, Charles Eric Leiserson, Ronald Linn Rivest, Clifford Seth Stein
https://mitpress.mit.edu/9780262046305/introduction-to-algorithms

2. "A Second Course in Algorithms" by Timothy Avelin Roughgarden
https://timroughgarden.org/w16/l

3. "Algorithm Design" by Jon Michael Kleinberg and Eva Tardos
https://dl.acm.org/doi/10.5555/1051910

4. "Computational Intractability: A Guide to Algorithmic Lower Bounds" by Erik D. Demaine, William Ian Gasarch, and Mohammad Taghi Hajiaghayi
https://hardness.mit.edu/drafts/2025-02-02.pdf

5. "The Design and Analysis of Algorithms" by Dexter Campbell Kozen
https://www.cs.cornell.edu/kozen/Papers/daa.pdf

6. "The Art of Computer Programming" by Donald Ervin Knuth
https://www-cs-faculty.stanford.edu/~knuth/taocp.html

Course Code	Course Name	L	T	P	Cr
DCSA250071	Software Engineering	3	1	0	4

Course Outcomes:

CO1: Understand Software Engineering Concepts – Explain core principles, ethical responsibilities, software processes, and critical system properties.
CO2: Analyze Software Requirements – Differentiate functional and non-functional requirements and apply requirements engineering processes.
CO3: Apply System Modeling & Project Management – Develop system models and manage software projects, including planning, scheduling, and risk management.
CO4: Design & Develop Software – Implement architectural and object-oriented design, agile methodologies, and software evolution strategies.
CO5: Perform Software Testing & Cost Estimation – Conduct verification, validation, software testing, team management, and cost estimation techniques

Module 1:

Overview - Introduction, FAQs about software engineering, Professional and ethical responsibility; Socio-technical systems - Emergent system properties, Systems engineering; Critical systems and Software processes - Critical systems, A simple safety-critical system, System dependability, Availability and reliability; Software processes - Software process models, Process iteration, Process activities, The Rational Unified Process, Computer-Aided Software Engineering.

Module 2:

Requirements - Software requirements, Functional and non-functional requirements, User requirements, System requirements, Interface specification, The software requirements document; Requirements engineering processes - Feasibility studies, Requirements elicitation and analysis, Requirements validation, Requirements management.

Module 3:

System models and Project Management - System models, Context models, Behavioural models, Data models, Object models, Structured methods; Project management - Management activities, Project planning, Project scheduling, Risk management

Module 4:

Software Design - Architectural design, Architectural design decisions, System organisation, Modular decomposition styles, Control styles; Object-oriented design - Objects and object classes, An object-oriented design process, Design evolution; Development - Rapid software development, Agile methods, Extreme programming, Rapid application development; Software evolution - Program evolution dynamics, Software maintenance, Evolution processes, Legacy system evolution.

Module 5:

Verification and validation - Verification and validation, Planning verification and validation, Software inspections, Automated static analysis, Verification and formal methods; Software testing - System testing, Component testing, Test case design, Test automation; Management - Managing people, Selecting staff, Motivating people, Managing groups, The People Capability Maturity Model; Software cost estimation, Software productivity, Estimation techniques, Algorithmic cost modelling, Project duration and staffing

Labs / Practicals:

N/A

References:

1.Ian Sommerville: Software Engineering, 8th Edition, Pearson Education, 2007.
2.Roger S. Pressman: Software Engineering - A Practitioner’s Approach, 7th Edition, Tata McGraw Hill, 2007.
3.Pankaj Jalote: An Integrated Approach to Software Engineering, Wiley India, 2009.

Course Code	Course Name	L	T	P	Cr
DOAI260001	AI (Industry)-Applied Ethics for AI	3	0	2	4

Course Outcomes:

CO1. Understand and be able to define and explain the terms 'AI for good' and 'responsible AI'. Describe the significance of responsible AI and recognize the potential benefits of AI in society.
CO2. Discuss and examine the impact of ethics on AI decision-making processes. Identify common ethical pitfalls in AI development and deployment. Explore methods for ethical AI practices.
CO3. Understand to apply an ethics-first approach to AI project planning and development. Compare different approaches to ethical AI design which include design of rules based on ethics and virtues. Identify factors influencing ethical decisions in AI development.
CO4. Apply Utilitarian and Kantian ethical principles to AI case studies, scenarios, and dilemmas. Explain the purpose and benefits of AI ethics codes. Develop a code of ethics for AI within an organization.
CO5. Identify the importance of interpretability in AI models. Evaluate the strengths and weaknesses of three different explainable AI tools: SHAP, LIME, and Tree Interpreter.
CO6. Conduct impact assessments for AI solutions. Apply ethical considerations in designing an AI system or solution. Implement measures to mitigate ethical risks and challenges post deployment.

Module 1:

Introduction to AI Ethics – meaning and importance, Role of Ethics in Responsible AI – defining 'AI for good' and 'responsible AI', significance of responsible AI in society, Common Ethical Pitfalls in AI – bias, fairness, accountability, transparency issues in AI development and deployment, Principles of Ethical AI – core principles guiding ethical AI, methods for ethical AI practices in development and deployment.

Module 2:

Ethics-First Approach in the AI Project Cycle – applying ethics at every stage of AI project planning and development, Approaches for Designing Ethical AI – rules-based ethics, values-based design, Introduction to Ethical Frameworks – overview of major ethical frameworks applied to AI, Virtue-Based Ethics – virtue ethics principles and the 4-box method for analysing ethical dilemmas, Applying Bio-Ethics Framework in AI – bioethical principles applied to AI in healthcare and biotechnology.

Module 3:

Utilitarianism for Ethical AI – utilitarian principles applied to AI case studies and dilemmas, Kantianism for Ethical AI – Kantian ethical principles applied to AI scenarios, Code of AI Ethics for Organizations – purpose, benefits and development of an organizational AI code of ethics, Data Protection and AI – risks of data breaches, privacy violations, responsible data handling.

Module 4:

Explainable AI – importance of interpretability in AI models, Intel XAI Toolkit, impact of explainability on decision-making and trust, Explainable AI in Practice – SHAP, LIME and Tree Interpreter tools: strengths, weaknesses and methods for applying them to analyze and interpret AI applications.

Module 5:

Impact Assessment of AI Solutions – conducting impact assessments for AI systems, Mitigating Ethical Issues Post Deployment – strategies to identify and resolve ethical risks after deployment, Project Creation – creating an AI startup with data card, model card, code of ethics, explainability report, impact self-assessment and transparency report.

Labs / Practicals:

1. Develop comprehensive Data Cards for datasets, as outlined in the Data Cards Playbook by Google, to ensure transparency and accountability in AI training data.
2. Design Responsible AI Best Practices.

3. Apply Human-Centered AI Canvas to design AI systems that address user needs and ethical considerations.
4. Learn to use The Box to embed AI principles into organizational workflows and operations.
5. Apply bio-ethics principles to AI case studies, scenarios, and dilemmas; evaluate the impact of virtue-based ethics on AI systems.
6. Develop AI ethics cards for given AI scenarios. Analyse a variety of ethical dilemmas by following the 4-box method from Virtue Ethics.
7. Analyse a variety of ethical dilemmas by walking through the ethical frameworks: Virtue Ethics, Bioethics, Utilitarian, Kantian, and Moral System Agnostic.
8. Develop an AI Code of Ethics for a fictitious company.
9. Use the UXAI Toolkit to brainstorm and plan different explainable interfaces for AI applications. Use the explainable AI library SHAP to create an explainability report for income prediction on the standard adult census income dataset.
10. Conduct impact assessments for AI solutions; create your own AI startup, develop business documents, write documentation (data card, model card, code of ethics, explainability report, impact self-assessment) and create and analyze transparency report.

References:

1. Intel AI for Future Workforce – Applied Ethics for AI Course Content, Intel Digital Readiness Program.
2. Virginia Eubanks – Automating Inequality: How High-Tech Tools Profile, Police, and Punish the Poor, St. Martin's Press.
3. Cathy O'Neil – Weapons of Math Destruction, Crown Publishers.
4. Kate Crawford – Atlas of AI: Power, Politics, and the Planetary Costs of Artificial Intelligence, Yale University Press.
5. Google Data Cards Playbook – https://pair-code.github.io/datacardsplaybook/

Course Code	Course Name	L	T	P	Cr
DOAI260002	AI (Industry)-Introduction to Artificial Intelligence	3	0	2	4

Course Outcomes:

CO1. Describe what AI is and what it is not. Appreciate the history of AI and its evolution over time.
CO2. Identify and analyze current trends in AI by correlating them with other technologies like IoT, Big Data and 5G.
CO3. Identify 3 common domains of AI – Natural Language Processing, Computer Vision and Statistical Data – based on the type of underlying data.
CO4. Examine and appreciate typical steps involved in an AI Project through the AI Project Cycle.
CO5. Examine the immediate impacts of AI practices on society and appreciate the ethical concerns of AI applications.
CO6. Classify Supervised Learning, Unsupervised Learning and Reinforcement Learning with the help of examples. Discuss the roles played by different key stakeholders in a typical AI team.

Module 1:

Demystification of AI – what AI is and what it is not, History and Evolution of AI, Current trends in AI, AI and Technology convergence, Industry 4.0 and Digitalization, Role of IoT, Big Data and 5G in AI, Domains of AI – Natural Language Processing, Computer Vision and Statistical Data.

Module 2:

AI Project Cycle-I – Problem Scoping and Data Acquisition, AI Project Cycle-II – Modelling and Evaluation, Societal Impact of AI – ethical concerns and responsible practices, Elements of ML – Supervised Learning, Unsupervised Learning and Reinforcement Learning with examples.

Module 3:

Elements of AI – types and characteristics, Data as Fuel for AI – structured and unstructured data, methods of data mining and data storage, AI/Data Science Team – roles and responsibilities of key stakeholders.

Module 4:

Tools to implement AI – No-Code tools (Teachable Machine, Orange, Roboflow, Chatfuel) and Code-based tools, Introduction to Neural Networks – working of simple Neural Networks, difference between common Deep Learning models with examples and applications.

Module 5:

AI Project-I – Natural Language Processing use case using no-code tool Chatfuel, AI Project-II – Statistical Data use case using no-code tool Orange Data Mining, AI Project-III – Computer Vision use case using no-code tool Roboflow, Future Possibilities of AI – emerging software and hardware technology trends and their direct impact on development of AI.

Labs / Practicals:

1. Familiarize with AI tools and techniques, including statistical data analysis using raw graphs, computer vision applications like Vision API, and generative AI tools such as DALL·E, Stable Diffusion, Gemini, and ChatGPT.
2. Apply the AI Project Cycle to real-world scenarios by identifying problems, collecting data, developing models, and evaluating their effectiveness to propose impactful AI-based solutions.
3. Deconstruct the working behind simple Neural Networks and outline the difference between some common DL models with the help of examples and applications.
4. Explore and utilize no-code tools, such as Teachable Machine, to build AI projects.
5. Project Building using No-code tool – Orange Data Mining for Statistical Data Use cases.
6. Project Building using No-code tool – Roboflow for Computer Vision Use cases.
7. Project Building using No-code tool – Chatfuel for Natural Language Processing Use cases.
8. Classify the type of data into structured and unstructured and evaluate data quality using real datasets.

9. Explore various AI domains by running Vision API experiments and comparing outcomes from NLP, CV, and Statistical Data use cases.
10. Demonstrate the AI Project Cycle on a chosen real-world problem from an industrial or social impact perspective and present findings.

References:

1. Intel AI for Future Workforce – Introduction to AI Course Content, Intel Digital Readiness Program.
2. Stuart Russell and Peter Norvig – Artificial Intelligence: A Modern Approach, Pearson Education.
3. Andreas Mueller and Sarah Guido – Introduction to Machine Learning with Python, O'Reilly Media.
4. Ian Goodfellow, Yoshua Bengio, and Aaron Courville – Deep Learning, MIT Press.
5. Aurélien Géron – Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, O'Reilly Media.

Course Code	Course Name	L	T	P	Cr
DOAI260003	AI (Industry)-Introduction to Generative AI	3	0	2	4

Course Outcomes:

CO1. Define Gen AI and distinguish its role in the broader context of AI. Describe how the basic model of Gen AI works. Discuss the wider societal impact of Gen AI.
CO2. Understand how NLP has evolved over time and the shift from Traditional AI to Gen AI. Understand Prompt Engineering and its impact on the output of NLP tasks.
CO3. Learn the limitations of current AI generation tools for video and audio. Recognize the challenges in ethical considerations and responsibilities when using generative AI tools.
CO4. Describe the stages of a generative AI project and learn to select and use appropriate AI tools for development.
CO5. Understand the basic structure of neural networks, architecture of transformer models, workings of diffusion models, and the pre-training process for generative AI models.
CO6. Learn to fine-tune generative AI models, understand RLHF, prototype Gen AI applications, and apply principles of responsible and ethical use of LLMs.

Module 1:

Introduction to Gen AI – definition and scope, role of Gen AI in the broader context of AI, how the basic model of Gen AI works, societal impact of Gen AI, Demystifying Gen AI Models – types of generative models (GANs, VAEs, Diffusion Models, LLMs), evolution of NLP to Gen AI, shift from Traditional AI to Gen AI.

Module 2:

Prompt Engineering-I – introduction to prompt engineering, basic applications and impact on NLP task outputs, text generation and conversation initiation, Prompt Engineering-II – advanced prompt engineering techniques for diverse NLP applications, real-world examples such as Mint Mobile Commercial, Basics of Video and Audio for Generation using Gen AI for Productivity-I – introduction to video generation tools (Runway ML, Genmo), text-to-video and image-to-video generation, limitations of current AI generation tools for video and audio.

Module 3:

Basics of Video and Audio for Generation using Gen AI for Productivity-II – audio and music generation using AIVA, code generation using Codeium, text generation using ChatGPT, Gemini and Perplexity, Generative AI-Ethics – ethical considerations, responsibilities and challenges when using generative AI tools, Generative AI Project Lifecycle – stages of a Gen AI project, selection of appropriate AI tools, Gen AI Tools for Development – open-source tools such as Stable Diffusion XL, proprietary tools such as Bing Image Creator powered by DALLE3.

Module 4:

Neural Network Fundamentals – basic structure and functioning of neural networks, how neural networks process data to make predictions, Transformers and the Evaluation of LLMs – architecture of transformer models, methods to evaluate large language models for performance and reliability, walkthrough of OpenAI Playground and parameter exploration, Diffusion Models and the Evaluation of Text to Image Generation – workings of diffusion models, evaluation of quality of text-to-image generation, Pre-Training Gen AI Models-I – pre-training process for generative AI models, importance of pre-training in building effective AI systems, modifying classification labels of Cohere datasets and its impact on fine-tuning and model confidence.

Module 5:

Fine-Tuning Gen AI Models – techniques for fine-tuning generative AI models to adapt them for specific tasks and applications, RLHF – Evaluating and Optimizing Gen AI Models: reinforcement learning with human feedback, evaluation and improvement of performance of generative AI models, Prototyping Gen AI Applications – skills to design, develop and test prototypes of generative AI applications, LangChain components – LLM, Agents, Prompts, Chains, Document loaders and utils with code-based implementation, LLM Application and Responsible use of AI – practical applications of large language models, ethical and responsible use of LLMs, building an AI

Tutor using open-source models such as Dolly 3B and Intel Neural Chat 7B.

Labs / Practicals:

1. Apply prompt engineering skills in basic NLP tasks, such as text generation, conversation initiation, and simple problem-solving.
2. Develop advanced prompt engineering skills for diverse NLP applications. Examine real-world prompt engineering examples, such as the Mint Mobile Commercial, to discern its role in marketing.
3. Explore cutting-edge Gen AI tools, their features, and capabilities for content generation including video and audio.
4. Explore open-source models such as Stable Diffusion XL and proprietary tools such as Bing Image Creator (powered by DALLE3) for image generation.
5. Text-to-video generation and image-to-video generation using tools like Runway ML and Genmo. Generate audio and music files using the tool AIVA.
6. Generate code by prompts using the tool Codeium. Generate text using tools like ChatGPT, Gemini, and Perplexity.
7. Walkthrough of OpenAI Playground. Explore how changing parameters can affect responses generated by different models.
8. Explore how modifying classification labels of Cohere datasets impacts fine-tuning and model confidence.
9. Explore basic components of LangChain like LLM, Agents, Prompts, Chains, and Document loaders and utils, along with a code-based implementation.
10. Utilize open-source text generation models such as Dolly 3B and Intel Neural Chat 7B for building an AI Tutor application.

References:

1. Intel AI for Future Workforce – Introduction to Generative AI Course Content, Intel Digital Readiness Program.
2. Antony Hagiwara – Generative Deep Learning: Teaching Machines to Paint, Write, Compose, and Play, O'Reilly Media.
3. Ashish Vaswani et al. – Attention is All You Need, Advances in Neural Information Processing Systems.
4. OpenAI – GPT-4 Technical Report, OpenAI.
5. Harrison Chase – LangChain Documentation – https://docs.langchain.com/

Course Code	Course Name	L	T	P	Cr
DOAI260004	AI (Industry)-Introduction to Machine Learning	3	0	2	4

Course Outcomes:

CO1. Describe Machine Learning and how Deep Learning is a subset of the wider field of Machine Learning.
CO2. Understand the importance of dashboards and discuss various applications of dashboards across different industries.
CO3. Explain the mathematical working behind different ML models. Implement different classification and regression ML models using Python.
CO4. Outline the difference between supervised, unsupervised and reinforcement learning algorithms. Examine the working of different reinforcement learning models.
CO5. Describe the working of Neural Networks and contrast it with biological Neurons. Describe various applications of neural networks. Outline different methods to overcome variance and bias in DL models.
CO6. Discuss and interpret the future of ML based on current and upcoming trends. Develop Python-based AI Projects incorporating Supervised Learning, Unsupervised Learning and Deep Neural Networks.

Module 1:

Introduction to Machine Learning – definition, types and applications, Relationship between AI, ML and Deep Learning, Tools to Implement AI (Python & Mathematics) – I: Python basics for ML, NumPy, Pandas, data manipulation.

Module 2:

Tools to Implement AI (Python & Mathematics) – II: mathematical foundations – linear algebra, statistics and probability, No-Code tool for Data Exploration – Tableau dashboard creation, data visualization techniques, statistical concepts implementation using no-code visualization tools.

Module 3:

AI (Modeling) Supervised Learning-I – Linear Regression, Logistic Regression, mathematical working behind models, classification and regression using Python, AI (Modeling) Supervised Learning-II – SVM, Decision Tree, industrial applications of ML models, AI (Modeling) Unsupervised Learning-I – K-Means Clustering, mathematics of clustering algorithms, AI (Modeling) Unsupervised Learning-II – Recommendation Systems, mathematics behind recommendation system, applications of recommendation systems.

Module 4:

Reinforcement Learning – definition, difference from supervised and unsupervised learning, working of reinforcement learning models with applications, Neural Network and Deep Learning-I – biological vs artificial Neurons, working of Neural Networks, applications of neural networks, Deep Learning-II – Convolutional Neural Networks, Recurrent Neural Networks, methods to overcome variance and bias in DL models, implementation using Python and deep learning frameworks, exploring OpenAI's Gym for reinforcement learning.

Module 5:

ML Project – Supervised Learning AI models: end-to-end Python project using classification/regression, ML Project – Unsupervised Learning AI models: end-to-end Python project using clustering, ML Project – Deep Neural Networks: building and training DNN in Python, Future of Machine Learning and Deep Learning – current and upcoming trends in ML and DL.

Labs / Practicals:

1. Interpret and understand the basic tools required for building ML Projects through a selected list of topics from Mathematics and Python Programming.
2. Develop and create a simple dashboard for visualizing data through Tableau.
3. Implement statistical concepts using the no-code visualization tool and create your own dashboard with the help of a no-code visualization tool.

4. Implement supervised machine learning algorithms, such as SVM and Decision Tree, using Python to develop a deeper understanding of their practical applications.
5. Implement unsupervised machine learning algorithms, such as K-Means, using Python to develop a deeper understanding of their practical applications.
6. Implement neural networks and create simple generative AI models using Python and deep learning frameworks, including exploring OpenAI's Gym for hands-on experience with AI and reinforcement learning.
7. Develop Python-based use cases and AI Projects incorporating Supervised Learning methods.
8. Develop Python-based use cases and AI Projects incorporating Unsupervised Learning methods.
9. Develop Python-based use cases and AI Projects incorporating Deep Neural Networks.
10. Implement a recommendation system using Python and evaluate its performance on a real-world dataset.

References:

1. Intel AI for Future Workforce – Introduction to Machine Learning Course Content, Intel Digital Readiness Program.
2. Andreas Mueller and Sarah Guido – Introduction to Machine Learning with Python, O'Reilly Media.
3. Ian Goodfellow, Yoshua Bengio, and Aaron Courville – Deep Learning, MIT Press.
4. Aurélien Géron – Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, O'Reilly Media.
5. Christopher Bishop – Pattern Recognition and Machine Learning, Springer.

Course Code	Course Name	L	T	P	Cr
DASH250095	Research Methodology and IPR	3	1	0	4

Course Outcomes:

CO1: Understand the principles and process of research methodology.
CO2: Apply appropriate research design and data collection methods in computing projects.
CO3: Analyze and interpret qualitative and quantitative data using statistical tools.
CO4: Develop ethically sound, well-documented research or project reports conforming to academic and industrial standards.
CO5: Understand various forms of intellectual property and apply processes for protection and patent filing.

Module 1:

Introduction to Research and Research Ethics
Definition and objectives of research, types of research – fundamental, applied, exploratory, empirical, significance of research in computer applications, research ethics and integrity, plagiarism, citation styles (APA, IEEE), tools for plagiarism check.

Module 2:

Research Design and Data Collection
Research problem identification and formulation, hypothesis generation, literature review techniques, types of research design: experimental, descriptive, analytical, sampling techniques, primary and secondary data sources, survey design, case study design.

Module 3:

Data Analysis and Interpretation
Quantitative and qualitative data analysis, measures of central tendency and dispersion, t-test, ANOVA, correlation and regression, chi-square test, software tools: Excel, R, SPSS (demo-level), data visualization and result interpretation.

Module 4:

Technical Writing and Project Report Preparation
Structure of research papers and reports, abstract, introduction, literature review, methodology, results, and conclusions, referencing tools (Mendeley/Zotero), IEEE/ACM/SCI journal guidelines, thesis and dissertation formatting, proposal writing basics.

Module 5:

Intellectual Property Rights and Patent Process
Introduction to IPR: patents, copyrights, trademarks, industrial design, Indian and international patent systems (WIPO, TRIPS), criteria for patentability, patent filing process in India (IPINDIA portal), open-source licenses, patent drafting basics, IPR in software and AI innovations.

Labs / Practicals:

N/A

References:

1. C.R. Kothari & Gaurav Garg – Research Methodology: Methods and Techniques, New Age International.
2. Ranjit Kumar – Research Methodology – A Step-by-Step Guide, Sage Publications India.
3. P. Narayan – Intellectual Property Law, Eastern Book Company.
4. Neeraj Pandey & Khushdeep Dharni – Intellectual Property Rights, PHI Learning.
5. Yogesh Kumar Singh – Fundamentals of Research Methodology and Statistics, New Age International.
6. Bharat Bhushan – Research Methodology in Computer Science, Himalaya Publishing.
7. WIPO and IPIndia Manuals – Guidelines for Filing Patents in India.

Course Code	Course Name	L	T	P	Cr
DCSA250003	Mobile Application Development using Android	2	1	2	4

Course Outcomes:

CO1: Understand the architecture, lifecycle, and development environment of Android applications.
CO2: Design and build responsive user interfaces using Android layout and widget components.
CO3: Develop Android applications using activities, intents, and persistent data storage.
CO4: Integrate multimedia, location, and device services such as sensors and GPS in Android apps.
CO5: Deploy Android applications and apply debugging, testing, and publishing techniques.

Module 1:

Introduction to Android and Development Environment
Overview of Mobile App Development Landscape
Introduction to Android OS: History, Features, and Architecture
Android Studio: Installation and Setup
Android Project Structure (Manifest, Java/Kotlin files, Res folder)
Activity Lifecycle
Hello World App
Emulator and Device Testing Basics

Module 2:

User Interface Design and Layout Management
UI Components: TextView, EditText, Button, ImageView, CheckBox, RadioButton, Spinner
Layouts: LinearLayout, RelativeLayout, ConstraintLayout, FrameLayout
Event Handling and Listeners
Toasts, Dialog Boxes, and Snackbars
RecyclerView and Adapters
Menus: Options Menu, Context Menu

Module 3:

Activities, Intents, and Data Persistence
Intents: Explicit and Implicit
Activity Communication (startActivityForResult)
Fragments: Introduction and Usage
Shared Preferences and Internal Storage
SQLite Database: CRUD Operations
Room Persistence Library (Basics)

Module 4:

Advanced Features and Device Integration
Multimedia: Audio, Video Playback
Camera Integration
Accessing Device Sensors: Accelerometer, Gyroscope
Location and Maps: GPS and Google Maps API (Basic)
Background Tasks: AsyncTask, WorkManager (Intro)
Permissions and Security

Module 5:

App Deployment, Debugging and Project Development
Debugging Tools in Android Studio
Unit Testing and UI Testing Basics
App Signing and Versioning
Publishing App to Google Play (Overview)
Mini Project / Capstone Project
Best Practices in Android Development

Labs / Practicals:

1. Create a “Hello World” Android app using Android Studio and test on emulator.
2. Design a login UI screen using different layout managers and widgets.
3. Create an app with multiple activities and pass data using intents.
4. Store user preferences using SharedPreferences and display them.
5. Develop a simple SQLite-based app for student record management.
6. Integrate Google Maps to display current location.
7. Capture image using device camera and display in the app.
8. Create a media player app that plays audio or video from the device.
9. Develop an app with sensor-based interaction, such as shake detection.
10. Final Mini Project: A complete Android app with at least one form of persistent data storage and sensor/location integration.

References:

Kushwaha, D. S., & Misra, R.
Android Application Development: A Beginner's Guide
McGraw Hill Education, 2018.

Mednieks, Z., Dornin, L., Meike, G., & Nakamura, M.
Programming Android
O’Reilly Media, 3rd Edition, 2012.

Donn Felker
Android Application Development for Dummies
Wiley Publishing, 3rd Edition, 2015.

Joseph Annuzzi Jr., Lauren Darcey, and Shane Conder
Android Wireless Application Development (Volume I & II)
Addison-Wesley, 2015.

Wei-Meng Lee
Beginning Android Programming with Android Studio
Wiley, 4th Edition, 2021.

Online Resources

Android Developers Official Documentation
https://developer.android.com

Google Codelabs – Android Development Tutorials
https://developer.android.com/codelabs

Android Jetpack (UI, Room, WorkManager)
https://developer.android.com/jetpack

Kotlin Language Documentation
https://kotlinlang.org/docs/home.html

Udacity Course: Developing Android Apps with Kotlin
https://www.udacity.com/course/developing-android-apps-with-kotlin--ud9012

Course Code	Course Name	L	T	P	Cr
DCSA250031	Full Stack Web Development	3	0	2	4

Course Outcomes:

CO1: Explain full stack development concepts, client-server model, and use essential development tools effectively.
CO2: Develop interactive web pages using HTML, CSS, JavaScript events, form validation, and dynamic content.
CO3: Build React applications with components, JSX, props, state, interactive forms, and conditional rendering.
CO4: Create backend APIs using Node.js and Express, connect with frontend, and exchange JSON data.
CO5: Integrate frontend and backend features into a functional mini project with data persistence.

Module 1:

What is full stack development? Difference between front-end and back-end, Understanding how websites work (client/server model),Introduction to tools: VS Code, Chrome, Node.js, GitHub.

Module 2:

Simple web page structure using HTML tags, CSS for styling layout and elements JavaScript basics: variables, events, interaction, Handling user actions like clicks and input, Creating and validating simple forms, Dynamic content display using JavaScript.

Module 3:

Introduction to React and benefits, Setting up a React project, Understanding components and JSX, Passing data with props and using useState, Creating interactive forms, /Handling input changes and button clicks, Rendering lists and conditional content.

Module 4:

Introduction to Backend and APIs, What is Node.js? What is a server? Creating a simple Express serve, Sending and receiving JSON. Simple REST API with Express, Create routes: GET, POST, Connecting Backend to Frontend: Calling backend from React using fetch, Displaying backend data in UI, Submitting form data to backend.

Module 5:

Developing a mini project Using the Features: Frontend form (React), Backend data saving (Node), displaying data list.

Labs / Practicals:

preparing

References:

1. "Full Stack Development with React and Node.js",By David Choi
2. "Pro MERN Stack" (2nd Edition),By Vasan Subramanian
3. JavaScript and JQuery: Interactive Front-End Web Development",By Jon Duckett
4. "Learning Full-Stack JavaScript Development",By Eric Bush

Course Code	Course Name	L	T	P	Cr
DCSA250059	Object-Oriented Programming with Java	2	0	4	4

Course Outcomes:

CO1: Describe the procedural and object oriented paradigm with concepts of streams, classes, functions, data and objects.
CO2: Understand dynamic memory management techniques using pointers, constructors, destructors, etc
CO3: Describe the concept of function overloading, operator overloading, virtual functions and polymorphism.
CO4: Classify inheritance with the understanding of early and late binding, usage of exception handling, generic programming.
CO5: Demonstrate the use of various OOPs concepts with the help of program

Module 1:

An Overview of Java
Object-Oriented Programming, A First Simple Program, A Second Short Program, Two Control Statements, Using Blocks of Code, Lexical Issues, The Java Class Libraries, Data Types, Variables, and Arrays: Java Is a Strongly Typed Language, The Primitive Types, Integers, Floating-Point Types, Characters, Booleans, A Closer Look at Literals, Variables, Type Conversion and Casting, Automatic Type Promotion in Expressions, Arrays, A Few Words About Strings, Arithmetic Operators, The Bitwise Operators, Relational Operators, Boolean Logical Operators, The Assignment Operator, The ? Operator, Operator Precedence, Using Parentheses, Control Statements: Java's Selection Statements, Iteration Statements, Jump Statements.

Module 2:

Object and Class
Class Fundamentals, Declaring Objects, Assigning Object Reference Variables, Introducing Methods, Constructors, The this Keyword, Garbage Collection, The finalize( ) Method, A Stack Class, A Closer Look at Methods and Classes: Overloading Methods, Using Objects as Parameters, A Closer Look at Argument Passing, Returning Objects, Recursion, Introducing Access Control, Understanding static, Introducing final, Arrays Revisited, Inheritance: Inheritance, Using super, Creating a Multilevel Hierarchy, When Constructors Are Called, Method Overriding, Dynamic Method Dispatch, Using Abstract Classes, Using final with Inheritance, The Object Class.

Module 3:

Interface and Packages
Defining Interfaces, Extending Interfaces, Implementing Interfaces, Accessing Interface Variables. Java API Packages, Using System Packages, Naming Conventions, Creating Packages, Accessing a Package, Using a Package, Adding a Class to a Package, Hiding Classes, Static Import.

Module 4:

Exception Handling and I/O Exceptions
Handling-Fundamentals, exception types, using try and catch, multiple catch clause, nested try statements – throw, throws and finally, built-in exceptions, creating own exceptions. Input / Output Basics – Streams – Byte streams and Character streams – Reading and Writing Console – Reading and Writing Files

Module 5:

Multithreaded Programming
Creating Threads, Extending the Thread Class, Stopping and Blocking a Thread, Life Cycle of a Thread, Using Thread Methods, Thread Exceptions, Thread Priority, Synchronization, Implementing the 'Runnable' Interface,

Inter-thread communication. vectors, lists, maps.

Labs / Practicals:

Preparing

References:

1.Java: The Complete Reference, Twelfth Edition Paperback – 23 December 2021 by Herbert Schildt
2.Java Performance: The Definitive Guide: Getting the Most Out of Your Code by Scott Oaks

Course Code	Course Name	L	T	P	Cr
DCSA250084	Web Technologies	3	0	2	4

Course Outcomes:

CO1: Explain the fundamentals of web technology, including its history, evolution, client-server architecture, web browsers, web servers, types of web pages, and development environments.
CO2: Develop structured and styled web pages using HTML and CSS, incorporating responsive design principles.
CO3: Apply JavaScript for programming logic, event handling, DOM manipulation, form validation, and AJAX-based interactions.
CO4: Build dynamic web applications using server-side scripting (PHP/Node.js) and integrate them with databases (MySQL).
CO5: Demonstrate skills in website deployment, hosting, web security, content management systems, and version control with Git.

Module 1:

Definition of Web Technology, history, and evolution of the web. Understanding client-server architecture. Web Browsers and Web Servers: Definition, features, and functionalities. Web page types: static, dynamic, responsive, and interactive. Introduction to HTML, CSS, and JavaScript. Overview of web development environments.

Module 2:

Basic structure of an HTML document, HTML tags, attributes, and elements. Creating and formatting text, links, images, lists, tables, and forms. Introduction to Cascading Style Sheets, syntax, selectors, and types (inline, internal, external). Styling text, layouts, and images. Responsive design with media queries.

Module 3:

Introduction, syntax, variables, operators, and control structures. Functions, events, and event handling. DOM manipulation and traversal. Form validation using JavaScript. Introduction to AJAX for asynchronous data loading. Integrating JavaScript with HTML and CSS.

Module 4:

Introduction to server-side scripting languages: PHP, Node.js. Setting up a local server environment. Database concepts: introduction to MySQL, SQL basics, and connecting a web application with a database.

Module 5:

Types of hosting services, domain name registration, and hosting providers. Uploading and managing websites on servers. FTP and file management. Web security measures: authentication, encryption (SSL), and data protection. Introduction to content management systems (CMS) like WordPress. Version control using Git.

Labs / Practicals:

1. Create a simple web page with your name, photo, and contact details.
2. Design a student registration form using different input controls (text, radio, checkbox, etc.).
3. Create an HTML page using tables to display a college timetable.
4. Create a web page with internal and external linking.
5. Write a HTML program to design a form which should allow to enter your personal data (Hint: make use of text field, password field, e-mail, lists, radio buttons, checkboxes, submit button).
6. Write HTML Code to demonstrate the use of Anchor Tag for the Following:-

a. Creating a web link that opens in a new window.
b. Creating a web link that opens in the same window
c. Reference within the same html document.
d. Reference to some image.
e. Making an image a hyperlink to display second image.

7. Create an html page with following specifications. Title should be about my City, Place your City name at the top of the page in large text and in blue color. Add names of landmarks in your city each in a different color, style and typeface. One of the landmark, your college name should be blinking. Add scrolling text with a message of your choice.
8. Create an html page with 7 separate lines in different colors. State color of each line in its text.
9. Design a webpage with a navigation bar using lists.
10. Apply inline, internal, and external CSS to format text and layout.
11. Create a webpage with a fixed header and footer using CSS.
12. Design a responsive layout using Flexbox or Grid.
13. Create a webpage with hover effects on buttons and menus.
14. Build a styled form using CSS (with focus and validation effects).
15. Write a JavaScript program to validate a user login form.
16. Write a JavaScript program to display the current date and time.
17. Create a simple calculator using JavaScript.
18. Write a JavaScript program for form input validation (e.g., email, number).
19. Use JavaScript to change the content and style of an HTML element dynamically.
20. Implement an image slideshow or gallery using JavaScript.
21. Write a PHP script to display your name and current date/time.
22. Create a simple PHP calculator (Add, Subtract, Multiply, Divide).
23. Write a PHP script to process and display form data (POST/GET).
24. Write a PHP program to connect to MySQL database and display data.

References:

1. Web Technologies by Uttam K. Roy ,Oxford University Press
2. Web Technology: Theory and Practice by M. Srinivasan, Pearson Education
3. Internet and Web Technologies by Rajkamal, Tata McGraw Hill
4. Web Programming using HTML, CSS, JavaScript & PHP by Dr. Neeraj Kumar, Neeraj Publications

Course Code	Course Name	L	T	P	Cr
DCSE250003	OS Principles and Applied Linux	2	0	4	4

Course Outcomes:

CO1: Understand the fundamental concepts and architecture of modern operating systems.
CO2: Analyse and implement process scheduling, memory management, and file systems.
CO3: Perform essential Linux operations using command-line tools.
CO4: Write and debug shell scripts for task automation.
CO5: Manage users, permissions, networking, and packages in a Linux environment.

Module 1:

Introduction to Operating Systems
Definition and functions of Operating System, OS structures: Monolithic, Layered, Microkernel, Hybrid, System calls and OS interface, Types of OS: Batch, Time-sharing, Distributed, Real-time, Case study: Linux vs Windows architecture

Module 2:

Process and CPU Management
Process concept, Process Control Block (PCB), Process states and transitions, Threads and Multithreading, CPU Scheduling: FCFS, SJF, Round Robin, Priority, Inter-process Communication (IPC), Semaphores, Deadlock

Module 3:

Memory and File System Management
Memory hierarchy, Address binding, Dynamic loading, Paging, Segmentation, Virtual Memory, Page replacement algorithms: FIFO, LRU, Optimal, File system interface: Directory, File operations, File system implementation: Allocation methods, Free-space management

Module 4:

Linux Fundamentals and Shell Programming
Linux architecture and file system hierarchy, Linux shell and types (Bash, sh, etc.), File and directory commands, Permissions, File redirection, Shell scripting: Variables, Conditional statements, Loops, Filters: grep, sed, awk; Pipes and command chaining

Module 5:

Linux System Administration and Networking
User and group management, Disk management and file system mounting, Process management: ps, top, kill, Package installation (apt, yum, rpm), Network tools: ifconfig, ping, ssh, scp, Crontab and system services, Log file management

Labs / Practicals:

1.Install and set up a Linux distribution (Ubuntu/Fedora) on a virtual machine using VirtualBox or VMware. Demonstrate basic navigation and system information commands.
2.Write a C program to implement basic system calls: fork(), getpid(), exec(), and wait(), Display how new processes are created and managed.
3.Simulate FCFS, SJF, and Round Robin CPU scheduling algorithms in C or Python. Use arrival time and burst time as input.
4.Write a C program to demonstrate inter-process communication (IPC) using shared memory.

5.Write a program in C to implement producer-consumer problem using semaphores.
6.Monitor and manage system processes using Linux commands: ps, top, kill, nice, and renice
7.Simulate paging and implement LRU/FIFO page replacement algorithms in C/Python.
8.Use df, du, lsblk, and mount commands to check disk space and mount/unmount drives.
9.Use Linux file system commands to perform the following operations: Create, delete, rename, move, compress, and decompress files/directories.
10. Write a shell script to check whether a number is prime or not.
11.Write a shell script to automate backup of a directory to another location with a timestamp.
12.Write a shell script that accepts a filename and displays:
i. Number of lines
ii. Number of words
iii. Number of characters
13. Create a shell script that logs system resource usage every 10 seconds into a file.
14.Create and manage users and groups using Linux commands: useradd, passwd, usermod, groupadd, chmod, chown
15.Set up a basic SSH server on your Linux machine and connect from another system using ssh. Enable public/private key authentication.

References:

1.Abraham Silberschatz, Peter B. Galvin, Greg Gagne Operating System Concepts, 10th Edition, Wiley.
2.Richard Blum, Christine Bresnahan Linux Command Line and Shell Scripting Bible, 4th Edition, Wiley.
3.Andrew S. Tanenbaum, Herbert Bos Modern Operating Systems, 4th Edition, Pearson.
4. Neil Matthew, Richard Stones Beginning Linux Programming, 4th Edition, Wrox.
5. Evi Nemeth et al. UNIX and Linux System Administration Handbook, 5th Edition, Pearson.

Course Code	Course Name	L	T	P	Cr
DCSE250010	Design and Analysis of Algorithms	3	0	2	4

Course Outcomes:

CO1: Understand the fundamental concepts of algorithm design and analysis.
CO2: Implement and analyze algorithms using different design paradigms.
CO3: Have the mathematical foundation in analysis of algorithms
CO4: Understand different algorithmic design strategies like Divide and conquer, Dynamic programming, greedy Algorithms, backtracking.
CO5: Analyze the efficiency of algorithms using time and space complexity theory

Module 1:

Notion of Algorithm, Role of algorithms in computing, Growth of functions- Asymptotic notations, Mathematical Analysis of Non-Recursive and Recursive Algorithms, Brute Force Approaches - Introduction, Selection Sort and Bubble Sort, Sequential Search. Brute Force String Matching.

Module 2:

Divide and Conquer: General Method, Defective Chess Board, Binary Search, Merge Sort, Quick Sort, Heap sort and its performance.

Module 3:

The Greedy Method: The General Method, Amortized Complexity, Knapsack Problem, Job Sequencing with Deadlines, Minimum-Cost Spanning Trees: Prim's Algorithm, Kruskal's Algorithm; Single Source Shortest Paths;
Dynamic Programming:
The General Method, Matrix Chain Multiplication, Longest Common Subsequence, Warshall's Algorithm, Floyd's Algorithm for the All-Pairs Shortest Paths Problem, Single-Source Shortest Paths: General Weights, 0/1 Knapsack, The Traveling Salesperson problem.

Module 4:

Decrease-And-Conquer Approaches, Space-Time Tradeoffs
Decrease and Conquer Approaches: Introduction, Insertion Sort, Depth First Search and Breadth First Search, Topological Sorting, Space-Time Tradeoffs: Introduction, Sorting by Counting, Input Enhancement in String Matching.

Module 5:

Limitations of Algorithmic Power and Coping with them: Lower Bound Arguments, Decision Trees, P, NP, and NP-Complete Problems, Challenges of
Numerical Algorithms;
Coping with Limitations of Algorithmic Power:
Backtracking: n-Queens problem, Hamiltonian Circuit Problem, Subset – Sum Problem. Branch-and-Bound: Assignment Problem, Knapsack Problem, Traveling Salesperson Problem. Approximation Algorithms for NP-Hard Problems, Traveling Salesperson Problem, Knapsack Problem.

Labs / Practicals:

Each algorithm should be implemented in C or Python, and operate on runtime inputs.

01. Implement a linear time greedy algorithm for the fractional knapsack problem.

02. Implement a greedy algorithm for optimal solution of job scheduling with deadlines.

03. Implement Kruskal's algorithm for minimum spanning tree of a connected graph.

04. Implement Prim’s algorithm for minimum spanning tree of a connected graph.

05. Implement Dijkstra’s algorithm for single-source shortest paths (no negative edge).

06. Implement Bellman-Ford algorithm for single-source shortest paths on a digraph.

07. Implement the Euclidean algorithm to find the GCD of two non-negative integers.

08. Implement the binary exponentation algorithm for efficient modular exponenation.

09. Implement binary search, uniform binary search, and interpolation search.

10. Implement quicksort with Hoare partitioning and Lomuto partioning schemes.

11. Implement an adaptive variant of mergesort that runs in linear time for best case.

12. Implement Floyd-Warshall algorithm for all-pairs shortest-path on a weighted graph.

13. Use dynamic programming to find optimal grouping in matrix chain multiplication.

14. Implement a solver for n-queens puzzle, along with a polynomial-time verifier.

15. Implement a polynomial time solver for 2-SAT problem expressed as Krom formula.

References:

1. "Computer Science III" by Chris Bourke
https://cse.unl.edu/~cbourke/ComputerScienceThree.pdf

2. "Algorithms in C (Third Edition)" by Robert Sedgewick
https://www.oreilly.com/library/view/algorithms-in-c/9780768685312

3. "Algorithms Illuminated (Omnibus Edition)" by Timothy Avelin Roughgarden
https://www.algorithmsilluminated.org

4. "The Algorithm Design Manual (Third Edition)" by Steven Sol Skiena
https://www.algorist.com/Algorist_ed2

5. "The Design and Analysis of Computer Algorithms" by Alfred Vaino Aho, John Edward Hopcroft, Jeffrey David Ullman
https://dl.acm.org/doi/10.5555/578775

6. "Computers and Intractability; A Guide to the Theory of NP-Completeness" by Michael Randolph Garey and David Stifler Johnson
https://dl.acm.org/doi/10.5555/578533

Course Code	Course Name	L	T	P	Cr
DCSE250020	Cloud Computing	3	0	2	4

Course Outcomes:

CO1: Explain the evolution, architecture, and key characteristics of cloud computing, including service models (IaaS, PaaS, SaaS) and deployment types (public, private, hybrid).

CO2: Describe various types and implementation levels of virtualization, including CPU, memory, storage, and network virtualization in cloud environments.

CO3: Analyze layered cloud architecture and address design challenges related to inter-cloud resource management and platform deployment.

CO4: Apply distributed and parallel programming paradigms such as MapReduce and Hadoop for cloud-based application development and understand various cloud platforms like AWS, OpenStack, and Google App Engine.

CO5: Evaluate cloud security challenges and solutions, including data and application security, VM security, and risk management strategies in cloud environments.

Module 1:

Introduction to Cloud Computing

Evolution of Cloud Computing, System Models for Distributed and Cloud Computing, NIST Cloud Computing Reference Architecture, Features of Cloud Computing, Cloud Services – IaaS, PaaS, SaaS, Cloud service Providers – Public , Private and Hybrid Clouds.

Module 2:

Introduction to component virtualization

Basics of Virtualization, Types of Virtualization, Implementation Levels of Virtualization, Virtualization of CPU, Memory, I/O Devices, Desktop Virtualization, Server Virtualization, Storage Virtualization, Network Virtualization.

Module 3:

Architectural Design of Compute and Storage Clouds

Layered Cloud Architecture Development, Design Challenges, Inter Cloud Resource Management, Resource Provisioning and Platform Deployment, Global Exchange of Cloud Resources.

Module 4:

Parallel and Distributed Programming Paradigms

Map Reduce, Twister and Iterative MapReduce, Hadoop Library from Apache, Mapping Applications, Programming Support, Google App Engine, Amazon AWS, Cloud Software Environments - Eucalyptus, Open Nebula, OpenStack.

Module 5:

Security Overview

Cloud Security Challenges, Software-as-a-Service Security, Security Governance, Risk Management, Security Monitoring, Security Architecture Design, Data Security, Application Security, Virtual Machine Security.

Labs / Practicals:

1. Exploring Cloud Deployment and Service Models
2. Simulating Public, Private, and Hybrid Clouds
3. Virtualization of CPU and Memory
4. Network and Storage Virtualization
5. Deployment of a Layered Cloud Architecture
6. Resource Provisioning and Scaling
7. Deploying a Web Application
8. Simulating Security Attacks and Defenses in a Cloud VM
9. Implementing Access Control and Encryption on Cloud Storage

References:

[1] R. Buyya, C. Vecchiola, and T. Selvi, Mastering Cloud Computing: Foundations and Applications Programming. New Delhi, India: McGraw-Hill Education, 2013.
[2] A. T. Velte, T. J. Velte, and R. Elsenpeter, Cloud Computing: A Practical Approach. New Delhi, India: McGraw-Hill Education, 2010.
[3] K. Hwang, G. C. Fox, and J. J. Dongarra, Distributed and Cloud Computing: From Parallel Processing to the Internet of Things. Burlington, MA, USA: Morgan Kaufmann, 2012.
[4] T. Erl, R. Puttini, and Z. Mahmood, Cloud Computing: Concepts, Technology & Architecture. Boston, MA, USA: Pearson Education, 2013.
[5] G. Reese, Cloud Application Architectures: Building Applications and Infrastructure in the Cloud. Sebastopol, CA, USA: O'Reilly Media, 2009.

Course Code	Course Name	L	T	P	Cr
DCSE250029	Computer Graphics	3	1	0	4

Course Outcomes:

CO1: Understand the basics of computer graphics systems and their architecture. Discuss various algorithms for scan conversion and filling of basic objects and their comparative analysis.
CO2: Apply geometric transformations and model graphical objects using OpenGL.
CO3: Develop interactive applications with input and event handling.
CO4: Illustrate concepts of 3D viewing, projections, and shading models.
CO5: Implement algorithms for clipping, rasterization, and rendering pipeline stages.

Module 1:

Introduction to Computer Graphics and OpenGL
Introduction to Computer Graphics: Applications, physical vs. synthetic images,
Graphics Systems: Imaging systems, synthetic camera model, Graphics Architectures: Programmable pipelines, performance metrics, Graphics Programming: Sierpinski gasket, 2D applications, OpenGL Basics: OpenGL API, primitives, attributes, colors Viewing & Control: Viewing, control functions, recursive graphics, 3D Gasket & Implicit Functions

Module 2:

Interaction and Input Mechanisms
Interaction & Input Devices: Event handling, input models, Client-Server Architecture, Display Lists and Modeling: Reusability, efficiency, Event Driven Programming: Menus, picking, Building Interactive Models: CAD systems, Animation and Logic Operations

Module 3:

Geometric Transformations and Modeling
Geometric Objects: Scalars, points, vectors, Three-dimensional Primitives; Coordinate Systems and Frames; Modeling a Colored Cube; Affine Transformations; Rotation, Translation and Scaling; Geometric Objects and Transformations; Transformation in Homogeneous Coordinates; Concatenation of Transformations; OpenGL Transformation Matrices; Interfaces to 3D applications; Quaternion's.

Module 4:

Viewing and Shading
Classical and computer viewing; Viewing with a Computer; Positioning of the camera; Simple projections; Projections in OpenGL; Hidden- surface removal; Interactive Mesh Displays; Parallel-projection matrices; Perspective-projection matrices; Projections and Shadows. Light and Matter; Light Sources; The Phong Lighting model; Computation of vectors; Polygonal Shading; Approximation of a sphere by recursive subdivisions; Light sources in OpenGL; Specification of materials in OpenGL; Shading of the sphere model; Global Illumination.

Module 5:

Implementation Techniques and Rendering Pipeline
Basic Implementation Strategies; Four major tasks; Clipping; Line-segment clipping; Polygon clipping; Clipping of other primitives; Clipping in three dimensions; Rasterization; Bresenham's algorithm; Polygon Rasterization; Hidden-surface removal; Antialiasing; Display considerations.

Labs / Practicals:

NA

References:

1.Edward Angel: Interactive Computer Graphics A Top-Down Approach with OpenGL, 5th Edition, Pearson Education, 2008.
2.Donald Hearn and Pauline Baker: Computer Graphics OpenGL Version 3th Edition, Pearson Education, 2004.
3.F.S. Hill Jr.: Computer Graphics Using OpenGL, 3th Edition, PHI, 2909.
4.James D Foley, Andries Van Dam, Steven K Feiner, John F Hughes, Computer Graphics, Pearson Education 1997.

Course Code	Course Name	L	T	P	Cr
DCSE250035	Computer Networks	3	0	2	4

Course Outcomes:

CO1: Understand Network Basics – Learn network models, topologies, and transmission media.
CO2: Analyse Data Link Layer – Evaluate error detection, MAC protocols, and wireless communication.
CO3: Implement Network Layer Functions – Apply IP addressing, subnetting, and routing algorithms.
CO4: Apply Transport & Application Layer Concepts – Compare TCP/UDP and analyse key protocols.
CO5: Demonstrate Security & Emerging Technologies – Identify threats, implement cryptography, explore SDN, IoT, and cloud networking.

Module 1:

Introduction to Computer Networks Overview of Computer Networks, Network Types: LAN, WAN, MAN, PAN, Network Topologies: Bus, Star, Ring, Mesh, Hybrid, OSI Model & TCP/IP Model – Layers and Functions, Transmission Media: Wired & Wireless Technologies, Switching Techniques: Circuit, Packet, and Message Switching

Module 2:

Data Link Layer & MAC Protocols
Error Detection and Correction: Parity, Checksum, CRC, Hamming Code, Flow Control Mechanisms: Stop-and-Wait, Sliding Window Protocol, MAC Protocols: ALOHA, CSMA/CD, CSMA/CA, Ethernet Standards (IEEE 802.3), VLANs, Spanning Tree Protocol, Wireless LAN (Wi-Fi), Bluetooth, and Mobile Networks

Module 3:

Network Layer & Routing Algorithms
IPv4 and IPv6 Addressing, Subnetting, and CIDR, Routing Algorithms: Distance Vector, Link-State, and Hybrid Routing, Routing Protocols: RIP, OSPF, EIGRP, and BGP, Congestion Control Mechanisms and Quality of Service (QoS), Network Address Translation (NAT) and DHCP

Module 4:

Transport Layer & Application Layer Protocols
TCP and UDP: Connection Establishment, Flow Control, and Error Control, Congestion Control: TCP Reno, TCP Tahoe, and Fair Queuing, Application Layer Protocols: HTTP, HTTPS, FTP, SMTP, POP3, IMAP, DNS and URL Resolution, Peer-to-Peer Networks, Security in Transport & Application Layers: TLS, SSL, Firewalls

Module 5:

Network Security and Emerging Trends
Cryptography Fundamentals: Symmetric and Asymmetric Encryption, Authentication and Digital Signatures, Network Security Attacks: DoS, DDoS, Phishing, Man-in-the-Middle, Wireless and IoT Security Challenges, Software-Defined Networking (SDN) and Cloud Networking, Introduction to 5G, Edge Computing, and Quantum Networking

Labs / Practicals:

Preparing

References:

1. "Computer Networking: A Top Down Approach (Eighth Edition)" by James F. Kurose and Keith W. Ross

https://gaia.cs.umass.edu/kurose_ross/about.php

2. "An Introduction to Computer Networks (Second Edition)" by Peter Lars Dordal
https://intronetworks.cs.luc.edu/current2/ComputerNetworks.pdf

3. "Computer Networking: Principles, Protocols and Practice (Third Edition)" by Olivier Bonaventure
https://github.com/cnp3/ebook/releases/download/draft-3rd/CNP3-2021.pdf

4. "Computer Networks: A Systems Approach (Sixth Edition)" by Larry L. Peterson and Bruce S. Davie
https://github.com/SystemsApproach/book/releases/download/v6.1/book.pdf

5. "Computer Networks (Fifth Edition)" by Andrew Stuart Tanenbaum and David J. Wetherall
https://www.oreilly.com/library/view/computer-networks-fifth/9780133485936

6. "Internetworking with TCP/IP: Principles, Protocols, and Architecture (Sixth Edition)" by Douglas Earl Comer
https://www.oreilly.com/library/view/internetworking-with-tcpip/9780137464197

7. "UNIX Network Programming: Volume 1 (Third Edition)" by William Richard Stevens, Bill Fenner, and Andrew M. Rudoff
https://www.oreilly.com/library/view/the-sockets-networking/0131411551

8. "Computer and Network Organization: An Introduction" by Maarten van Steen and Henk Sips
https://www.distributed-systems.net/index.php/books/computer-and-network-organization

9. "Beej's Guide to Network Programming" by Brian Hall
https://beej.us/guide/bgnet/pdf/bgnet_a4_c_2.pdf

Course Code	Course Name	L	T	P	Cr
DCSE250043	Cryptography and Network Security	3	0	2	4

Course Outcomes:

CO1: Explain security fundamentals, including the CIA Triad and various cyber threats.
CO2: Describe security services and mechanisms, such as authentication and encryption.
CO3: Understand the history and evolution of cryptography, including classical techniques.
CO4: Recognize the role of cryptanalysis in breaking cryptographic systems.
CO5: Apply mathematical foundations, including number theory, modular arithmetic, and random number generation, in cryptographic algorithms.

Module 1:

Introduction to Cryptography & Security Concepts
Basics of Security, Security Goals: Confidentiality, Integrity, Availability (CIA), Threats and Attacks: Passive vs. Active Attacks, Security Services & Mechanisms, Cybersecurity Principles, Introduction to Cryptography, History & Evolution of Cryptography, Classical Cryptographic Techniques: Caesar Cipher, Vigenère Cipher, Playfair Cipher, Modern Cryptography & Cryptanalysis
Mathematical Foundations of Cryptography: Number Theory Basics: Prime Numbers, Modular Arithmetic, Euler's Theorem, Greatest Common Divisor (GCD), Fermat's Theorem, Random Number Generation

Module 2:

Symmetric & Asymmetric Cryptography
Symmetric Key Cryptography, Block Ciphers vs. Stream Ciphers, Data Encryption Standard (DES) & Triple DES, Advanced Encryption Standard (AES), Modes of Operation: ECB, CBC, CFB, OFB, CTR, Asymmetric Key Cryptography, Public Key Cryptosystems: RSA Algorithm, Key Exchange Mechanisms: Diffie-Hellman Key Exchange, Elliptic Curve Cryptography (ECC), Comparison of Symmetric & Asymmetric Cryptography

Module 3:

Hash Functions, Digital Signatures & Authentication Cryptographic Hash Functions, Properties of Hash Functions, Common Hash Algorithms: MD5, SHA-1, SHA-256, SHA-3, Message Authentication, Message Authentication Codes (MAC), HMAC (Hash-based Message Authentication Code), Digital Signatures & Certificates, RSA Digital Signature Algorithm, Digital Certificates & Public Key Infrastructure (PKI), SSL/TLS Encryption and HTTPS Authentication Mechanisms, Password-Based Authentication, Multi-Factor Authentication (MFA), Biometric Authentication

Module 4:

Network Security & Secure Protocols
Network Security Fundamentals, Security in OSI & TCP/IP Models, Firewalls and Intrusion Detection Systems (IDS/IPS), Virtual Private Networks (VPNs) & IPsec, Secure Network Protocols, Secure Email: PGP & S/MIME, Secure Web Transactions: HTTPS, SSL/TLS
Wireless Security: WPA, WPA2, WPA3, Network Attacks & Defense Mechanisms, Denial of Service (DoS) & Distributed DoS (DDoS) Attacks, Man-in-the-Middle (MITM) & Eavesdropping, SQL Injection & Cross-Site Scripting (XSS)

Module 5:

Emerging Trends & Cybersecurity Applications
Blockchain & Cryptography, Role of Cryptography in Blockchain, Bitcoin & Ethereum Security, Cloud Security & Zero Trust Security Model, Cloud Encryption Mechanisms, Zero Trust Architecture & Access Control
Cybersecurity & Ethical Hacking, Introduction to Ethical Hacking, Penetration Testing Basics, Legal & Ethical

Issues in Cybersecurity, Future Trends in Cryptography & Security

Labs / Practicals:

Preparing

References:

1. "The Joy of Cryptography" by Michael Rosulek
https://joyofcryptography.com/pdf/book.pdf

2. "Cryptography and Computer Security" by Chris Bourke
https://cse.unl.edu/~cbourke/cryptoNotes.pdf

3. "Applied Cryptography (20th Anniversary Edition)" by Bruce Schneier
https://www.schneier.com/books/applied-cryptography

4. "Cryptography and Network Security (Eighth Edition)" by William Stallings
http://williamstallings.com/Cryptography/Crypto8e-Student

5. "Cracking Codes With Python: An Introduction To Building And Breaking Ciphers" by Al Sweigart
https://inventwithpython.com/cracking

6. "Yet Another Introductory Number Theory Textbook (Cryptology Emphasis Version)" by Jonathan Adam Poritz
https://www.poritz.net/jonathan/share/yaintt.pdf

Course Code	Course Name	L	T	P	Cr
DCSE250081	Fuzzy Logic and Fuzzy Sets	3	1	0	4

Course Outcomes:

CO1: Understand and explain the fundamentals of fuzzy sets, including operations, properties, and extensions.
CO2: Apply fuzzy logic and approximate reasoning to model uncertainty and imprecision in real-world scenarios.
CO3: Implement fuzzy inference systems for decision-making and control applications.
CO4: Develop and apply fuzzy clustering and pattern recognition algorithms to analyze complex data.
CO5: Integrate fuzzy logic concepts with conventional systems to enhance performance in intelligent applications.

Module 1:

Fundamentals of Fuzzy Sets: Basic concepts on fuzzy sets, Basic Types, Fuzzy sets versus crisp sets, Properties of alpha-cuts, Representation of fuzzy sets, Extension principle.

Module 2:

Operations on Fuzzy Sets and Relations: Operation on fuzzy sets, Fuzzy union, intersection and complement, combinations of operations, Fuzzy arithmetic – Fuzzy numbers, Arithmetic operations on fuzzy numbers, Fuzzy relations, Binary fuzzy relations, Fuzzy equivalence and compatibility relations, Fuzzy ordering relations, Fuzzy morphism.

Module 3:

Fuzzy Measures and Fuzzy Logic: Fuzzy measures, Belief and possibility measures, Evidence theory, Possibility theory versus Probability theory, Fuzzy logic, Multivalued logic, Fuzzy propositions, Fuzzy qualifiers.

Module 4:

Approximate Reasoning and Fuzzy Expert Systems: Approximate reasoning – Fuzzy expert system (an overview), Fuzzy implications, Selection of fuzzy implication, Multiconditional approximate reasoning.

Module 5:

Fuzzy systems: Fuzzy system & neural network, Fuzzy automata, Fuzzy clustering, Fuzzy pattern recognition.

Labs / Practicals:

NA

References:

1.S. N. Sivanandam, S. Sumathi and S. N. Deepa, Introduction to Fuzzy Logic using MATLAB, Springer.
2.S. N. Sivanandam, S. Sumathi and S. N. Deepa, Introduction to Fuzzy Logic using MATLAB, Springer.
3.Timothy J. Ross, Fuzzy Logic With Engineering Applications, Wiley, 3rd edition.

Course Code	Course Name	L	T	P	Cr
DCSE250099	Object Oriented Analysis and Design	3	1	0	4

Course Outcomes:

CO1: Understand and apply the principles of object-oriented modeling and system abstraction.
CO2: Design effective class, state, and interaction models using UML.
CO3: Apply systematic processes to system conception, domain analysis, and system design.
CO4: Develop class designs and model implementation while integrating legacy systems.
CO5: Identify and utilize appropriate design patterns and idioms for building scalable systems.

Module 1:

Introduction, Modeling Concepts & Class Modeling:
Object Orientation Concepts-What is OO development? OO themes and assumptions, OO development usefulness and history; Modeling as a Design Technique - Modeling and abstraction, Three models: class, state, interaction; Class Modeling- Objects and classes, links and associations, Generalization and inheritance, Sample class model and navigation

Module 2:

Advanced Class Modeling and State Modeling:
Advanced Class Modeling - N-ary associations, association ends, aggregation, Abstract classes, multiple inheritance, Reification, metadata, derived data, constraints, Packages and modeling tips; State Modeling - Events, states, transitions, and conditions, State diagrams and behaviour, State modeling practicals

Module 3:

Advanced State and Interaction Modeling:
Advanced State Modeling - Nested diagrams and concurrency, Signal generalization, sample models, Relationship between class and state models; Interaction Modeling - Use case models, sequence models, activity models, Use case relationships, Procedural models and special constructs in activity modeling

Module 4:

Process, System Conception & Analysis:
Process Overview - Life cycle and development stages; System Conception - System concept, elaboration, and problem statement; Domain Analysis - Domain class, state, and interaction models, Iterative analysis process; Application Analysis & System Design - Application interaction, class, and state models, Performance estimation, reuse planning, Subsystem design, concurrency, and resource management, Architectural styles and ATM system example

Module 5:

Class Design, Implementation & Patterns:
Class Design & Implementation Modeling - Bridging gaps, use case realization, Refactoring and recursion, Design optimization and inheritance adjustments, Realizing associations and testing; Legacy Systems - Reverse engineering - class, interaction, state models, Wrapping and maintenance; Design Patterns – I & II - What makes a pattern? Pattern categories and relations, Communication Patterns: Forwarder-Receiver, Publisher-Subscriber, Management Patterns: Command Processor, View Handler; Idioms - Introduction, usage, style, Example: Counted Pointer Idiom

Labs / Practicals:

NA

References:

1.Michael Blaha, James Rumbaugh: Object-Oriented Modelling and Design with UML, 2nd Edition, Pearson Education, 2005.
2.Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal: Pattern-Oriented Software Architecture, A System of Patterns, Volume 1, John Wiley and Sons, 2007.
3. Grady Booch et al: Object-Oriented Analysis and Design with Applications, 3rd Edition, Pearson Education, 2007.
4.Brahma Dathan, Sarnath Ramnath: Object-Oriented Analysis, Design, and Implementation, Universities Press, 2009.
5.Hans-Erik Eriksson, Magnus Penker, Brian Lyons, David Fado: UML 2 Toolkit, Wiley- Dreamtech India, 2004.
6.Simon Bennett, Steve McRobb and Ray Farmer: Object-Oriented Systems Analysis and Design Using UML, 2 Edition, Tata McGraw-Hill, 2002.

Course Code	Course Name	L	T	P	Cr
DOAI250009	Big Data Analytics	3	0	2	4

Course Outcomes:

CO1: Describe big data characteristics, challenges, and applications across industries and domains.

CO2: Evaluate and apply NoSQL databases and distributed storage architectures for big data management.

CO3: Install, configure, and operate Hadoop and HDFS for batch-oriented big data processing.

CO4: Develop and implement data analysis workflows using Hadoop ecosystem tools including Hive, Pig, and HBase.

CO5: Design and implement real-time data processing and analytics using Spark Streaming and PySpark.

CO6: Apply visualization and regression techniques for analytical insights from large-scale data.

Module 1:

Introduction to Big Data Ecosystem

Big Data Characteristics: Volume, Variety, Velocity, Veracity, and Value, Business applications and case studies of big data analytics, Challenges in conventional systems, Big Data Analytics Lifecycle, Tools and Technologies in Big Data, Hadoop vs. Traditional Systems, NoSQL Overview (Key-Value, Document, Columnar, Graph databases), Basics of MongoDB and Cassandra for big data storage.

Module 2:

Hadoop Framework for Big Data

History and Evolution of Hadoop, Hadoop Distributed File System (HDFS): Architecture, Internals, and Operations, Block-level Storage, Fault Tolerance, MapReduce Paradigm: Concepts, Anatomy of a MapReduce Job, Failures, Scheduling, Shuffle, Sort, and Task Execution, Input / Output Formats in MapReduce, Hadoop Streaming and Integrations, Developing MapReduce Applications with Java and Python APIs.

Module 3:

Data Analysis with Hadoop Ecosystem

Hive: Data Modeling, Data Types, File Formats, HiveQL (DDL, DML, DQL), Querying Structured Data, Hive Services and Optimization.
Pig: Data Flow Language, Pig Latin Programming, Grunt Shell, Data Models, Operators, Joins, Developing and Testing Pig Scripts.
HBase: Architecture, Data Model, Integration with Hadoop, Use Cases.
Introduction to Data Lakes using Hadoop.
Hands-on with Hadoop Ecosystem for ETL pipelines.

Module 4:

Real-Time Big Data Analytics and Streaming

Stream Processing Fundamentals, Stream Data Models, Stream Architecture, Concepts of Sampling, Filtering, Counting in Streams, Real-time Sentiment Analysis, Stock Market Analysis using Streams, Introduction to Apache Kafka for Streaming Data Ingestion, Spark Basics: RDD, DataFrames, Spark SQL, In-Memory Computing, PySpark APIs, Spark Streaming for Real-Time Processing, Real-Time Analytics Applications, Case

Studies.

Module 5:

Analytical Models and Visualization for Big Data

Regression Techniques: Simple Linear Regression, Multiple Linear Regression, Model Interpretation for Large-Scale Data, Feature Engineering for Big Data, Visualization Techniques: Interactive Dashboards, Graphical Tools for Big Data (Tableau / PowerBI concepts), Visual Analytics: Trends, Patterns, Correlations, Interaction Models, Real-World Applications of Big Data Visualizations.

Labs / Practicals:

1. Setting up Hadoop, HDFS, and YARN on local and cloud-based environments (AWS / Azure / GCP).

2. Developing basic MapReduce jobs using Java and Python.

3. Implementing ETL pipelines using Hive and Pig.

4. Performing analytics with HBase and Hive.

5. Real-time data streaming with Apache Kafka and Spark Streaming (using PySpark).

6. Implementing basic ML pipelines over Spark (MLlib).

7. Visualizing Big Data using interactive tools (Matplotlib, Plotly, Tableau / PowerBI).

8. Group Project: Building and deploying an end-to-end Big Data Analytics pipeline on a real-world dataset.

References:

1. Tom White, Hadoop: The Definitive Guide, 4th Edition, O'Reilly Media, 2015.

2. Anand Rajaraman, Jeffrey David Ullman, Mining of Massive Datasets, Cambridge University Press, 3rd Edition, 2020.

3. Bill Franks, Taming the Big Data Tidal Wave: Finding Opportunities in Huge Data Streams with Advanced Analytics, John Wiley & Sons, 2012.

4. Jiawei Han, Micheline Kamber, Jian Pei, Data Mining: Concepts and Techniques, 3rd Edition, Morgan Kaufmann / Elsevier, 2011.

5. Holden Karau, Andy Konwinski, Patrick Wendell, Matei Zaharia, Learning Spark: Lightning-Fast Big Data Analysis, 2nd Edition, O'Reilly Media, 2020.

6. Paul Zikopoulos, Dirk deRoos, Krishnan Parasuraman, Thomas Deutsch, James Giles, David Corrigan, Harness the Power of Big Data – The IBM Big Data Platform, Tata McGraw Hill, 2012.

7. Michael Berthold, David J. Hand, Intelligent Data Analysis, Springer, 2007.

8. Da Ruan, Guoqing Chen, Etienne E. Kerre, Geert Wets, Intelligent Data Mining, Springer, 2007.

Official Documentation (Highly Recommended for Labs / Practical Work):

9. Official documentation for Hadoop: https://hadoop.apache.org/

10. Official documentation for Spark: https://spark.apache.org/docs/latest/

11. Official documentation for Hive: https://cwiki.apache.org/confluence/display/Hive/Home

12. Official documentation for Pig: https://pig.apache.org/

13. Official documentation for HBase: https://hbase.apache.org/

14. Official documentation for Kafka: https://kafka.apache.org/documentation/

15. Official documentation for PySpark: https://spark.apache.org/docs/latest/api/python/

Course Code	Course Name	L	T	P	Cr
DOAI250015	Data Analytics and Visualization	2	0	4	4

Course Outcomes:

CO1: Understand the fundamentals of data analytics, data preprocessing, and feature engineering techniques.
CO2: Perform efficient data manipulation, aggregation, and wrangling using Pandas and NumPy.
CO3: Implement numerical computing techniques for high-level mathematical operations and image data analysis.
CO4: Create effective data visualizations using Matplotlib, Seaborn, and Pandas for data-driven insights.
CO5: Utilize advanced visualization tools like Excel, Tableau, and Power BI for interactive and real-time data representation.

Module 1:

Fundamentals of Data Analytics and Preprocessing
Definition, importance, and applications of data analytics. Introduction to structured and unstructured data, data collection, data storage, and preprocessing techniques. Data ingestion, handling missing values, data cleaning, transformation, and feature engineering. Overview of NumPy and Pandas libraries for data analysis, including DataFrames, Series, indexing, slicing, and filtering.

Module 2:

Data Manipulation and Aggregation
Operations on Pandas DataFrames and Series, data wrangling techniques, handling categorical and numerical data, working with missing and duplicate values. Data aggregation, applying functions in Pandas, groupby operations, pivot tables, and cross-tabulations. Merging and concatenating datasets, time series analysis, and handling large datasets efficiently.

Module 3:

Numerical Computing with NumPy
Creating and manipulating multidimensional arrays, broadcasting, vectorized operations, and high-level mathematical functions. Array slicing, fancy indexing, filtering, and statistical analysis using NumPy. Working with structured arrays, handling image data, and performing linear algebra operations for data science applications.

Module 4:

Data Visualization Techniques
Importance of data visualization, choosing appropriate visualization methods, visualizing data insights and findings. Data visualization using Matplotlib: line plots, bar charts, scatter plots, histograms, and subplots. Advanced visualization using Seaborn: pair plots, violin plots, heatmaps, and regression plots. Using Pandas for quick data visualization and integrating visualization tools in data science workflows.

Module 5:

Tools and Advanced Visualization Techniques
Visualization using Excel: charts, pivot tables, and dashboards. Introduction to interactive visualization tools such as Tableau and Power BI. Creating real-time and dashboard-based visualizations, interactive plots using Plotly, and geographic data visualization with Folium. Case studies and real-world applications of data analytics and visualization.

Labs / Practicals:

1. Implement data wrangling techniques like reshaping and pivoting using Pandas.
2. Create Data Frames and Series in Pandas and perform basic data manipulation operations.
3. Load datasets from different file formats (CSV, Excel, JSON) into Pandas.
4. Perform basic exploratory data analysis (EDA) using summary statistics and visualizations.
5. Create data visualizations using Matplotlib (e.g., line plots, histograms, bar charts).
6. Create advanced visualizations using Seaborn (e.g., heatmaps, pair plots, violin plots).
7. Implement box plots, scatter plots, and correlation matrices for data analysis.
8. Perform data aggregation and group-by operations to analyze grouped data.
9. Merge multiple Data Frames using different join operations in Pandas.
10. Implement linear regression model and visualize the results using Matplotlib.
11. Create and customize subplots using Matplotlib to present multiple visualizations.
12. Perform data aggregation using pivot tables and cross-tabulation in Pandas.
13. Create interactive plots using Plotly for web-based visualization.
14. Create a word cloud from a textual dataset to visualize the frequency of words.
15. Perform principal component analysis (PCA) for dimensionality reduction and visualize the result.
16. Visualize time series data and trends using Matplotlib and Seaborn.
17. Use Excel for data visualization, such as creating charts, graphs, and dashboards.
18. Implement correlation analysis and create a heatmap for understanding relationships between variables

References:

1. Python – Data Visualisation by Samuel Burns
2. Storytelling with Data: A Data Visualization Guide for Business Professionals by Cole NussbaumerKnaflic
3. Data Science for Business by Foster Provost and Tom Fawcett:
4. Data Visualization: A Practical Introduction by Kieran Healy
5. Hands-On Data Visualization: Interactive Storytelling from Spreadsheets to Code by Jack Dougherty and IlyaIlyankou
6. Data Visualization with Excel Dashboards and Reports by Dick Kusleika
7. Beautiful Visualization: Looking at Data Through the Eyes of Experts edited by Julie Steele and Noah Iliinsky

Course Code	Course Name	L	T	P	Cr
DOAI250019	Data Mining and Warehousing	3	0	2	4

Course Outcomes:

CO1: Understand and apply data warehousing concepts and architectures.

CO2: Perform clustering, classification, and prediction for pattern discovery.

CO3: Address issues related to data streams and implement stream mining techniques.

CO4: Analyze and apply web mining methodologies for real-world applications.

CO5: Explore current trends and advancements in distributed warehousing and pattern mining.

Module 1:

Data warehousing components, Building a data warehouse, Data warehouse architecture, DBMS schemas for decision support, Data extraction, cleanup, and transformation tools, Metadata, Reporting, Query tools and applications, Online Analytical Processing (OLAP), OLAP and multidimensional data analysis.

Module 2:

Data mining functionalities, Data preprocessing: cleaning, integration, transformation, reduction, discretization, Concept hierarchy generation, Architecture of typical data mining systems, Classification of data mining systems, Efficient and scalable frequent itemset mining methods, Mining various kinds of association rules, From association mining to correlation analysis, Constraint-based association mining.

Module 3:

Issues in classification and prediction, Classification methods: Decision Trees, Bayesian classification, Rule-based classification, Neural networks: backpropagation, Support Vector Machines (SVM), Associative classification, Lazy learners, Other classification methods, Prediction techniques, Accuracy and error measures, Evaluating classifiers/predictors, Ensemble methods, Model selection.

Module 4:

Types of data in clustering, Categorization of major clustering methods, Partitioning methods, Hierarchical methods, Density-based methods, Grid-based methods, Model-based clustering methods, Clustering high-dimensional data, Constraint-based clustering, Outlier analysis.

Module 5:

Mining objects, Spatial data mining, Multimedia data mining, Text mining, Mining the World Wide Web, Multidimensional analysis of complex data, Descriptive mining of complex data objects.

Labs / Practicals:

1. Design a Data Warehouse Schema

Create Star Schema and Snowflake Schema for business data (sales, e-commerce, etc.).

2. ETL Process Implementation

Perform Extract, Transform, Load (ETL) operations on structured data using SQL-based tools (MySQL / PostgreSQL).

3. Association Rule Mining

Apply the Apriori Algorithm for mining frequent itemsets and generating association rules from transactional datasets.

4. Classification Using Decision Trees

Build, visualize, and evaluate Decision Tree Classifiers on datasets like Iris or Titanic.

5. Clustering Using K-Means

Apply K-Means Clustering on multi-dimensional datasets and visualize the results.

6. Data Preprocessing & Dimensionality Reduction

Handle missing values, normalize data, and apply Principal Component Analysis (PCA).

7. OLAP Operations and Cube Construction

Perform OLAP operations (Roll-up, Drill-down, Slice, Dice) using sample datasets.

8. Web Mining Basics

Analyze web log files for pattern discovery (frequent user paths, sessions).

References:

1. Jiawei Han, Micheline Kamber, Jian Pei, Data Mining Concepts and Techniques, Third Edition, Elsevier, 2011.

2. Alex Berson, Stephen J. Smith, Data Warehousing, Data Mining & OLAP, Tata McGraw-Hill, 10th Reprint, 2007.

3. K.P. Soman, Shyam Diwakar, V. Ajay, Insight into Data Mining: Theory and Practice, PHI, 2006.

4. G.K. Gupta, Introduction to Data Mining with Case Studies, PHI, 2006.

5. Pang-Ning Tan, Michael Steinbach, Vipin Kumar, Introduction to Data Mining, Pearson Education, 2007.

Course Code	Course Name	L	T	P	Cr
DOAI250030	Introduction to Artificial Intelligence	3	1	0	4

Course Outcomes:

CO1: Understand AI fundamentals, state-space search, and heuristic problem-solving methods.
CO2: Represent knowledge using logic, rules, and predicate calculus.
CO3: Apply reasoning methods under uncertainty including probabilistic and fuzzy logic approaches.
CO4: Explain key concepts in natural language processing and basics of machine learning.
CO5: Demonstrate the design of expert systems and AI-based planning algorithms.

Module 1:

Introduction and Problems, State Space Search & Heuristic Search Techniques:
What is AI? The AI Problems, The Underlying Assumption, What Is An AI
Techniques, The Level of The Model, Criteria For Success, Some General References, One Final Word, Defining The Problems as a State Space Search, Production Systems, Production Characteristics, Production System Characteristics, And Issues In The Design Of Search Programs, Additional Problems, Generate And-Test, Hill Climbing, Best-First Search, Problem Reduction, Constraint Satisfaction, Means Ends Analysis.

Module 2:

Knowledge representation:
Knowledge Representation Issues -Representations And Mappings, Approaches To Knowledge Representation; Using Predicate Logic - Representation Simple Facts In Logic, Representing Instance And Isa Relationships, Computable Functions And Predicates, Resolution; Representing Knowledge Using Rules -Procedural Versus Declarative Knowledge, Logic Programming, Forward Versus Backward Reasoning.

Module 3:

Reasoning:
Symbolic Reasoning Under Uncertainty -Introduction To Non-monotonic Reasoning, Logics For Non-monotonic Reasoning; Statistical Reasoning -Probability And Bays' Theorem, Certainty Factors And Rule-Base Systems, Bayesian Networks, Dempster-Shafer Theory, Fuzzy Logic; Weak Slot-and-Filler and Strong Slot-and-Filler Structures - Semantic Nets, Frames, Conceptual Dependency, Scripts, CYC

Module 4:

Natural Language Processing and Machine learning and Neural Networks:
Introduction, Syntactic Processing, Semantic Analysis, Semantic Analysis, Discourse and Pragmatic Processing, Spell Checking; Definition and examples of machine learning, supervised learning, unsupervised learning, reinforcement learning, introduction to neural networks.

Module 5:

Expert Systems and AI planning systems:
Representation using domain knowledge, Expert System shell, knowledge acquisition; Definition and examples of Planning Systems, planning as search, operator-based planning, propositional planning.

Labs / Practicals:

NA

References:

1. “Artificial Intelligence” -By Elaine Rich And Kevin Knight (2nd Edition) Tata Mcgraw-Hill
2. “Artificial Intelligence: A Modern Approach”, Stuart Russel, Peter Norvig, PHI
3. Introduction to AI & Expert Sys., Pitterson, D.N