AC: 01/25
Item no. 04

The Bombay Salesian Society's

Don Bosco Institute of Technology, Mumbai

(An Autonomous Institute affiliated to University of Mumbai)

Department of Computer Engineering

CURRICULUM STRUCTURE FOR SECOND YEAR ENGINEERING SEM III

(As Per NEP 2020)

(Scheme: DB25-V1)
Effective from Academic Year 2025-2026

1. Preamble

Don Bosco Institute of Technology, Kurla, Mumbai, proudly celebrates the achievement of autonomous status—an academic milestone that reaffirms our steadfast commitment to excellence, holistic development, and student-centric learning. This autonomy empowers us to craft and implement a curriculum that is forward-looking, contextually relevant, and deeply rooted in our institutional values and the aspirations of our nation.

As an autonomous institution affiliated with the University of Mumbai, DBIT embraces the opportunity to restructure its academic framework in alignment with the University Grants Commission (UGC) guidelines and the National Education Policy (NEP) 2020. This curriculum framework outlines the undergraduate engineering programs for the EXTC, COMP, IT, and MECH branches. It reflects NEP's emphasis on multidisciplinary learning, flexibility, and outcome-based education, while staying true to the Don Bosco educational philosophy.

The curriculum adopts a top-down approach, beginning with the institutional Vision and Mission, which guides the definition of Program Educational Objectives (PEOs) and Program Outcomes (POs). These outcomes are used to shape Course Outcomes (COs) and the content and assessment methods of each course. This ensures that all academic efforts remain aligned with the broader goals of transforming learners into technically sound, ethically responsible and socially aware citizens. Importantly, this curriculum has been shaped through extensive consultations with stakeholders, including industry experts, academic peers, alumni, and students—to ensure that it remains aligned with contemporary industry requirements and societal expectations. Their inputs have been instrumental in designing a framework that bridges the gap between academic learning and practical applicability.

Key Objectives in developing syllabus are:

1. **Develop Strong Technical Foundations:** Equip students with robust knowledge and skills in core engineering domains to solve real-world problems through design, analysis, and innovation.

2. **Foster Research, Innovation, and Entrepreneurship:** Cultivate a spirit of inquiry, critical thinking, and entrepreneurial mindset to promote research-based problem-solving and startup culture.

3. **Enhance Interdisciplinary and Industry-Ready Competencies:** Integrate emerging technologies, multidisciplinary learning, and practical exposure to prepare

students for dynamic industry requirements and lifelong learning.

4. **Promote Ethical, Sustainable, and Socially Responsible Engineering Practice:** Inculcate ethics, human values, and environmental consciousness to enable students to contribute meaningfully to society and sustainable development.

5. **Empower Communication, Leadership, and Teamwork Abilities:** Strengthen students' soft skills, collaboration, and leadership to perform effectively in diverse professional and global environments.

Academic design includes:

- A Choice-Based Credit System (CBCS) for flexibility
- A range of Minor and Honors options to encourage specialization and research
- Opportunities for field engagement, internships, and experiential learning
- Emphasis on skill enhancement and future workforce needs
- Integration of ethical reasoning, social awareness, and environmental consciousness

As an institution inspired by the values of Saint John Bosco, we strive to create a joyful and inclusive learning environment that fosters creativity, curiosity, and compassion. Through this curriculum framework, we renew our pledge to produce graduates who are not only professionally competent but also committed to the greater good of society.

2. Vision and Mission

Vision:

DBIT will be known to have an innovative, enjoyable, and holistic learning environment that transforms individuals into socially conscious citizens the Don Bosco way, and will lead in research and entrepreneurship in the area of sustainable technologies.

Mission:

1. To create future engineers who work with honesty and integrity and excel in the use of technology for the benefit of the underprivileged.

2. To train engineers to be innovative problem-solvers and entrepreneurs who engage in research and lifelong learning.

3. To provide a diverse and stimulating environment for staff and students to grow holistically.

3. Curriculum Design Philosophy

The curriculum is structured in alignment with the National Education Policy (NEP) 2020 and UGC guidelines. It follows a top-down approach wherein the institutional Vision and Mission guide the Program Educational Objectives (PEOs) and Program Outcomes (POs). These shapes the Course Outcomes (COs) and form the foundation

for the course structure, the delivery, and the assessments.

Key design principles include:

- Emphasis on Outcome-Based Education (OBE) with clear mappings of COs to POs
- Integration of core technical knowledge with interdisciplinary electives
- Inclusion of vocational skills, internships, and community engagement
- Development of entrepreneurship and research aptitude through minor and honors pathways
- Encouragement of ethical, sustainable, and socially responsible engineering practices

This approach ensures that the curriculum remains academically rigorous, industry-relevant, and value-driven.

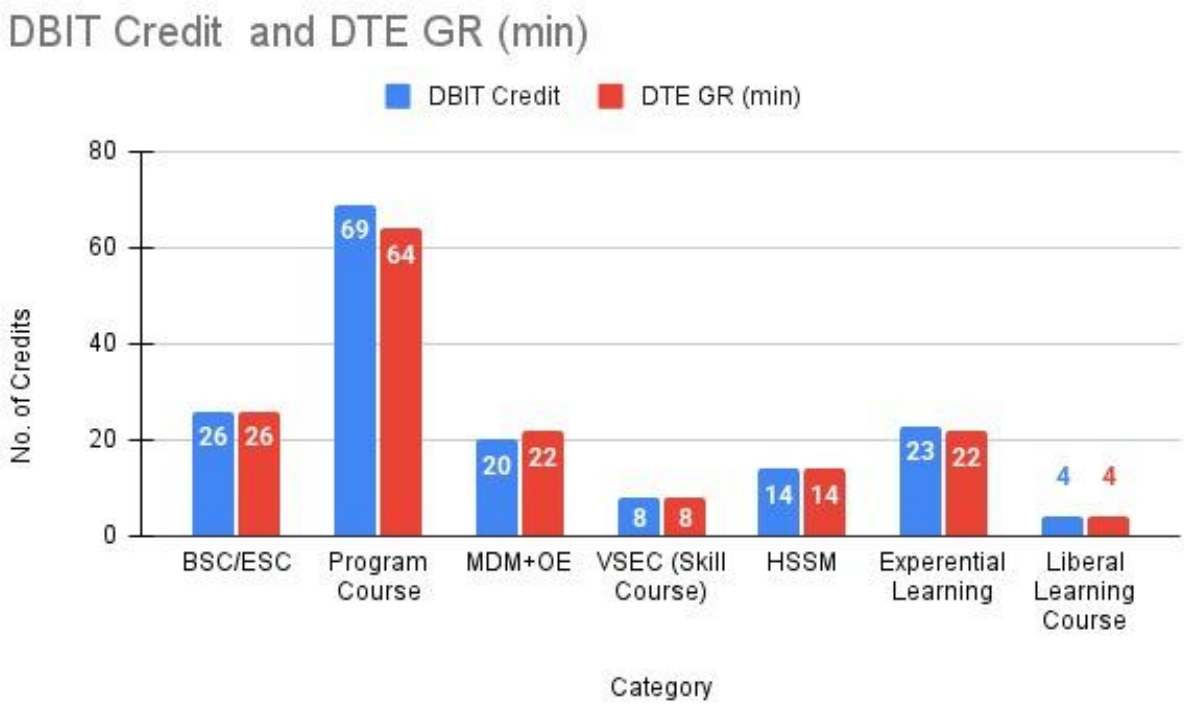
4. Credit Guidelines and Allocation

The curriculum is delivered through a structured credit system as follows:

Activity Type	Credit Definition
Theory Course	1 Credit = 15 Contact Hours
Laboratory / Studio / Workshop	1 Credit = 30 Contact Hours
Internship / Field Work	1 Credit = 40 Hours or 02 weeks
Seminar / Group Discussions	1 Credit = 15 Hours
Community Engagement / Field Project	1 Credit = 30 Hours

DBIT Overall Curriculum Credit Structure

Semester		I	II	III	IV	V	VI	VII	VIII	Total Credits	DTE Credits
Basic Science Course	BSC/ESC	9	6							15	14-18
Engineering Science Course		7	4							11	12 - 16
Programme Core Course (PCC)	Program Courses		3	16	14	6	6	6		51	44-56
Programme Elective Course (PEC)						3	3	6	6	18	20
Multidisciplinary Minor (MD M)	Multidisciplinary Courses				3	4	4	3		14	14
Open Elective (OE) Other than a particular program					2	2		2		6	8
Vocational and Skill Enhancement Course (VSEC)	Skill Courses	3	3	2						8	8
Ability Enhancement Course (AEC -01, AEC-02)	Humanities Social Science and Management (HSSM)		2			2				4	4
Entrepreneurship/Economics/ Management Courses					2		2			4	4
Indian Knowledge System (IKS)			2							2	2
Value Education Course (VEC)		2		2						4	4
Research Methodology	Experiential Learning Courses					2				2	4
Community. Engagement. Project (CEP)/ Field Project (FP) (Mini - Project)				1	1	1				3	2
Project							3	3		6	4
Internship/ OJT									12	12	12
Co-curricular Courses (CC)	Liberal Learning Courses		1		1		1		1	4	4
Total Credits (Major)		**21**	**21**	**21**	**23**	**20**	**19**	**20**	**19**	**164**	**160- 176**



5. Degree Options and Exit Pathways

Students are offered flexible learning pathways through the following options:

Undergraduate Degree Options:

- B.E with MDM – Minimum 164 credits
- B.E with Double Minor/ Honors – 182 credits
- B.E Research with Research – 182 credits

Multiple Entry-Exit Options (Aligned with NEP 2020):

Exit Options	Credits Structure
Certificate after Year 1:	42 Credits + 08 credits (04 credit Exit Course + 04 Summer internship).
Diploma after Year 2:	86 credits + 08 credits (04 credit Exit Course + 04 Summer internship).
B. Vocational Degree after Year 3:	125 credits + 08 credits (04 credit Exit Course + 04 Summer internship).

Credits earned are banked in the Academic Bank of Credits (ABC) for lifelong learning flexibility.

Abbreviations Used:

AEC	Ability Enhancement Course
AEL	Ability Enhancement Laboratory
BSC	Basic Science Course
BSL	Basic Science Laboratory
CEP	Community Engagement Project
CC	Co-curricular Courses
CIE	Continuous Internal Evaluation
EEM	Entrepreneurship, Economics and Management
ELC	Experiential Learning Courses
ESC	Engineering Science Course
ESE	End Semester Examination
ESL	Engineering Science Laboratory
FP	Field Project
HSSM	Humanities Social Science and Management
IKS	Indian Knowledge System
L	Lecture
LLC	Liberal Learning Courses
MDM	Multidisciplinary Minor
MSE	Mid Semester Exam
OE	Open Elective

OJT	On Job Training
P	Practical
PCC	Program Core Course
PCL	Program Core Laboratory
PEC	Program Elective Course
T	Tutorial
VEC	Value Education Course
VSEC	Vocational and Skill Enhancement Course

UG Second Year Computer Engineering Program

Curriculum Scheme and Structure: Semester III

* Two hours of practical class to be conducted for full class as demo/discussion.

Semester wise Curriculum Structure Semester – III								
Course Code	**Course Vertical**	**Course Name**	**Teaching Scheme (Contact Hours)**			**Credits Assigned**		
			L	**P**	**T**	**L**	**P**	**T**	**TOTAL**
25CE3PCC01	PCC	Discrete Structures and Graph Theory	3	-	1	3	-	1	4
25CE3PCC02	PCC	Analysis of Algorithms	3	2	-	3	1	-	4
25CE3PCC03	PCC	Computer Organization and Architecture	3	2	-	3	1	-	4
25CE3PCC04	PCC	Database Management Systems	3	2	-	3	1	-	4
25CE3VSEC01	VSEC	Full Stack Java	-	2*+2	-	-	2	-	2
25IL3VEC01	VEC	AI, IOT and ICT for Sustainable Indian Development	2	-	-	2	-	-	2
25CE3CEP01	CEP	Mini Project I	-	2	-	-	1	-	1
		Total	**14**	**12**	**1**	**14**	**6**	**1**	**21**

* Two hours of demo/discussion for entire class

Examination Marking Scheme: Semester III

Examination Scheme and Assessment Structure: Semester III								
Course Code	**Course Vertical**	**Course Name**	**Examination Marks**						
			CA	**MSE**	**ESE**	**TW**	**OR**	**PR**	**Total**
25CE3PCC01	PCC	Discrete Structures and Graph Theory	20	30	50	25	-	-	125
25CE3PCC02	PCC	Analysis of Algorithms	20	30	50	25	-	25	150
25CE3PCC03	PCC	Computer Organization and Architecture	20	30	50	25	-	-	125
25CE3PCC04	PCC	Database Management Systems	20	30	50	25	25	-	150
25CE3VSEC01	VSEC	Full Stack Java	-	-	-	25	-	25	50
25IL3VEC01	VEC	AI, IOT and ICT for Sustainable Indian Development	50	-	-	-	-	-	50
25CE3CEP01	CEP	Mini Project 1A	-	-	-	25	-	25	50
		TOTAL	**130**	**120**	**200**	**150**	**25**	**75**	**700**

Assessment Methodology

Types of Courses	Assessment Tools	Marks Distribution
Theory	**CA-20**	Certification: NPTEL (20 Marks) (Approved by instructor) OR Any two Pedagogies (10 marks each) • MCQ /Class Test • Case study/Assignment • GATE based Tutorial • Certification: Udemy/Coursera (Approved by instructor) • Open Book Test • Working model / simulation of a course-based concept.
Theory	**CA-25**	Certification: NPTEL (20 Marks) (Approved by instructor) Active Participation and Timely Submission of Laboratory and Programming Assignments (5 Marks) OR Any two Pedagogies (10 marks each) and Active Participation and Timely Submission of Laboratory and Programming Assignments (5 Marks) • MCQ /Class Test • Case study/Assignment • GATE based Tutorial • Certification: Udemy/Coursera (Approved by instructor) • Open Book Test • Working model / simulation of a course-based concept.
Theory (VEC)	**CA-50**	• Active Participation = 5 marks • MCQ /Class Test= 10 marks • Assessment of the Activity carried out by student = 25 marks • Assignment = 10 marks
Workshop	**CA-50**	• Active Participation = 5 marks • Trade 1# = 15 marks • Trade 2# = 15 marks • Trade 3# = 15 marks # Based on the performance and satisfactory completion of trade wise tasks.

Liberal Learning Courses (LLC)	**CA-50**	• Active Participation = 5 marks • Assessment of the Activity carried out by student = 25 marks • Cultural Event Participation = 10 marks • Technical Event Participation = 10 marks
Theory	**MSE**	Question Paper Pattern is as follows: All Questions are compulsory. • Q1 A or B - 10 marks • Q2 A or B - 10 marks • Q3 A or B - 10 marks • For each question, A and B should be based on the same CO. • MSE should be based on 50% syllabus. • Time: 90 minutes (1 hour 30 minutes) • Total Marks: 30
Theory	**ESE**	Question Paper Pattern is as follows: All Questions are compulsory. • Q1 A or B - 10 marks • Q2 A or B - 10 marks • Q3 A or B - 10 marks • Q4 A or B - 10 marks • Q5 A or B - 10 marks • For each question, A and B should be based on the same CO. • ESE should be based on 30% syllabus of MSE and 70% syllabus after MSE. • Time: 120 minutes (2 hours) • Total Marks: 50
Course - Laboratory	**TW- 25**	• Active Participation (Lab) = 5 marks • Laboratory Report = 10 marks • Laboratory performance = 10 marks Based on the performance and satisfactory completion of assigned laboratory work
Community Engagement	**TW-25**	• Active Participation = 5 marks • Project Report = 10 marks • Progress presentation (Minimum 02) & demonstration = 10 marks
Tutorial	**TW-25**	• Active Participation = 5 marks • Tutorial Submission = 20 marks Tutorial based on the entire syllabus

Laboratory	**OR-25**	Oral examination will be based on the entire syllabus.
Laboratory	**PR-25**	Practical examination will be based on the experiments performed by the students during laboratory sessions.

Weightage of COs across all Assessments:

Course Outcomes	Percentage
CO-1, CO-2	**20-30**
CO-3, CO-4	**40-50**
CO-5, CO-6	**20-30**

*Note: Total Weightage of All CO's should be 100%

Heads of Passing

a. Passing Criteria for Theory Course: 40% maximum marks in CA, MSE, ESE taken together
b. Passing Criteria for Laboratory/Tutorial (Term Work): 40% of maximum marks
c. Passing Criteria for Oral/Practical (Term Work): 40% of maximum marks

Discrete Structures and Graph Theory

Course Code	Course Name	Teaching Scheme (Hrs. / Week)			Credits Assigned			
		L	**T**	**P**	**L**	**T**	**P**	**Total**
25CE3PCC01	**Discrete Structures and Graph Theory**	03	01	--	03	01	-	04
			CA	**MSE**	**ESE**	**Term work**		**Total**
		Theory	20	30	50	25		125
		Lab	-	-	-	-		

Pre-Requisite Courses:	**25FE1BSC01** - Mathematical Foundation-I **25FE2BSC01** - Mathematical Foundation-II

Course Objectives:

1. Develop a strong foundation in the principles of logic, set theory, relations, functions, combinatorics, algebraic structures, and graph theory for problem-solving in computer science.
2. Enhance analytical and reasoning abilities through the construction and interpretation of mathematical proofs and models.
3. Apply discrete mathematical concepts to model, analyze, and solve computational, network, and real-world problems.
4. Foster the ability to design and evaluate mathematical structures and algorithms for efficiency, correctness, and applicability in computer engineering domains.

	After successful completion of the course, the students will be able to	
Course Outcomes	CO1	Identify and state the fundamental concepts, symbols, and laws of logic, set theory, relations, functions, algebraic structures, and graph theory. (Remembering)
	CO2	Explain and interpret logical inference, induction, relations, and functions, and differentiate between various counting techniques, algebraic structures, and graph types. (Understanding)
	CO3	Apply combinatorial methods, recurrence relations, and graph construction techniques to solve computational and network-related problems. (Applying)
	CO4	Analyze logical statements, algebraic structures, and graph models to break down problems and derive equivalent or optimized solutions. (Analyzing)
	CO5	Evaluate mathematical models and algorithms by comparing alternative solutions, verifying correctness, and justifying the choice of optimal strategies in coding theory and discrete structures. (Evaluating)
	CO6	Design logic expressions, counting models, and graph structures to model and solve real-world computational problems effectively. (Creating)

Syllabus:

Module No.	Unit No.	Topics	Hours
1		**Module 1: Propositional and Predicate Logic**	**06**
	1.1	Introduction to Logic – Propositional logic, symbols, syntax, semantics	
	1.2	Laws of Logic – Idempotent, associative, distributive, De Morgan's laws, tautology, contradiction.	
	1.3	Quantifiers – Universal and existential, negation rules.	
	1.4	Normal Forms – CNF, DNF, PCNF, PDNF, conversion techniques.	
	1.5	Predicate Logic – Statements, predicates, domain of discourse, translating English statements into logic.	
	1.6	Inference Theory – Rules of inference, resolution method, unification basics.	
	1.7	Mathematical Induction – Principle of mathematical induction, strong induction, applications.	
Self-Learning		• Build a **truth table generator** in Python for any propositional expression. • Use **Prolog** to encode and solve a set of logical puzzles. • **Use online logic solvers** (e.g., Logic Friday) to test logical equivalences.	
2		**Module 2: Relations and Functions**	**07**
	2.1	Set Theory Refresher – Definitions, subsets, power sets, set operations, Venn diagrams.	
	2.2	Relations – Definitions, properties, representation (matrices, digraphs).	
	2.3	Types of Relations – Reflexive, symmetric, transitive, antIsymmetric, equivalence, partial order..	
	2.4	Closures of Relations – Reflexive, symmetric, transitive closure; Warshall's algorithm.	
	2.5	Functions – Definition, domain, codomain, range, types (one-one, onto, bijection).	
	2.6	Identity & Inverse Functions – Existence conditions, examples.	
	2.7	Composition of Functions – Associativity, applications in computer science.	
Self-Learning		Use **Excel or Python Pandas** to represent and query relation matrices. Create a **real-world mapping project**: city → airport → country relations.	
3		**Module 3: Posets and Lattices**	**07**
	3.1	Partial Orders – Definition, properties, examples.	
	3.2	Hasse Diagrams – Construction, interpretation.	
	3.3	Lattices – Definition, join and meet operations, types (bounded, complemented, distributive).	

		Sublattice – Definition, examples.	
	3.4	Chains and Antichains – Definitions, examples, Dilworth's theorem (conceptual).	
Self – Learning		Draw **Hasse diagrams** using Graphviz or Python's NetworkX library.	
4		**Module 4: Combinatorics**	**08**
	4.1	Basic Counting Principles – Sum rule, product rule, worked examples.	
	4.2	Inclusion-Exclusion Principle – Formula, applications in counting problems.	
	4.3	Pigeonhole Principle – Generalized form, examples in computer science (hashing collisions).	
	4.4	Permutations & Combinations – Definitions, formulae, circular permutations, constrained arrangements.	
	4.5	Generating Functions – Definitions, types, application to solving recurrence relations.	
	4.6	Recurrence Relations – Formulation, solving linear recurrence relations with constant coefficients.	
Self – Learning		Create a **seat allocation calculator** using permutations and combinations. Write a Python program to apply **inclusion–exclusion** for counting possible outcomes in overlapping events.	
5		**Module 5: Algebraic Structures**	**09**
	5.1	Introduction – Definition, examples from number theory and modular arithmetic.	
	5.2	Algebraic Structures with One Binary Operation – Semigroup, monoid, groups (definition, properties, examples).	
		Definition and structure of Subgroup, Abelian group, cyclic group	
	5.3	Homomorphism and Isomorphism – Definitions, examples, kernel, image.	
	5.4	Algebraic Structures with Two Binary Operations – Rings, types (commutative, division ring, field), properties, examples.	
Self-learning		• Write a script to test if a given set with an operation forms a **group, ring, or field**. • Apply cyclic groups to **generate secure random numbers**.	
6		**Module 6: Graph Theory**	**08**
	6.1	Introduction – Definition, terminology, degree of vertex, types of graphs (simple, multigraph, pseudograph, directed, weighted).	
	6.2	Graph Representation – Adjacency matrix, adjacency list, incidence matrix.	
	6.3	Subgraphs – Spanning subgraphs, induced subgraphs.	
	6.4	Operations on Graphs – Union, intersection, complement, Cartesian	

		product.	
	6.5	Walk, Path, Circuit – Definitions, connected & disconnected graphs, components.	
	6.6	Graph Isomorphism & Homomorphism – Definitions, examples.	
	6.7	6.7.1 Planar Graphs – Euler's formula, Kuratowski's theorem (conceptual). 6.7.2 Cut Set & Cut Vertex – Definitions, applications in network reliability.	
	6.8	Euler & Hamiltonian Graphs – Definitions, conditions, applications (travelling salesman problem).	
Self-Learning		Use **NetworkX** to create and visualize a social media network, finding shortest paths between friends.	
		TOTAL	**45**

Suggested Tutorial List

Tutorial Questions (Sample Question Set)

1. Define a partial order relation. Give an example.
2. Draw the Hasse diagram for the set {1, 2, 3, 6} under the divisibility relation.
3. Find all chains and antichains in the poset ({a, b, c, d}, ≤) where a ≤ b, a ≤ c, b ≤ d, c ≤ d.
4. Show that the set of natural numbers ≤ 10 with the usual order forms a lattice.
5. What is the difference between a bounded and a complemented lattice?

Tutorial Activities

1. Group Task: Create Hasse diagrams from a given partial order.
2. Assignment: Identify sublattices in given lattices.
3. Class Demo: Explore distributive lattices through Boolean algebra.
4. Story Problem: Design the mathematical model for the given story problem.
5. Prolog: Use the logical programming language and implement the given discrete statement for the problem.

Text Books:

1. Bernad Kolman, Robert Busby, Sharon Cutler Ross, Nadeem-ur-Rehman, "Discrete Mathematical Structures", Edition: Sixth, 2015, Pearson Education.
2. C. L. Liu "Elements of Discrete Mathematics", second edition 1985, McGraw-Hill Book Company Reprinted 2000.
3. K. H. Rosen, "Discrete Mathematics and applications", fifth edition 2003, Tata McGraw Hill Publishing Company

Reference Books:

1. J. L. Mott, A. Kandel, T. P. Baker, "Discrete Mathematics for Computer Scientists and Mathematicians", Second Edition 1986, Prentice Hall of India.

2. Seymour Lipschutz, Marc Lars Lipson, “Discrete Mathematics” Schaum‟s Outline, McGraw Hill Education.

Useful Links:

1. https://nptel.ac.in/courses/106103205
2. https://onlinecourses.nptel.ac.in/noc25_cs127/preview
3. https://onlinecourses.nptel.ac.in/noc25_cs26/preview
4. https://www.coursera.org/specializations/discrete-mathematics
5. https://www.youtube.com/playlist?list=PLHXZ9OQGMqxersk8fUxiUMSIx0DBqsKZS

Assessment Methodology

Assessment Tools	Marks Distribution
Continuous Assessment (CA) **(20 Marks)**	• Certification: NPTEL (20 Marks) (Approved by instructor) ○ OR • Any two Pedagogies (10 marks each) • MCQ /Class Test • Case study/Assignment • GATE based Tutorial • Certification: Udemy/Coursera (Approved by instructor) • Open Book Test • Working model / simulation of a course-based concept.
Mid Semester Examination (MSE) **(30 Marks)**	Question Paper Pattern is as follows: • All Questions are compulsory. • Q1 A or B - 10 marks • Q2 A or B - 10 marks • Q3 A or B - 10 marks For each question, A and B should be based on the same CO. MSE should be based on 50% syllabus. Time: 90 minutes (1 hour 30 minutes) Total Marks: 30

End Semester Examination (ESE) (50 Marks)	Question Paper Pattern is as follows: • All Questions are compulsory. • Q1 A or B - 10 marks • Q2 A or B - 10 marks • Q3 A or B - 10 marks • Q4 A or B - 10 marks • Q5 A or B - 10 marks For each question, A and B should be based on the same CO. ESE should be based on 30% syllabus of MSE and 70% syllabus after MSE. Time: 120 minutes (2 hours) Total Marks: 50
Term Work (25 Marks)	• Active Participation (Lab) = 5 marks • Laboratory Report = 10 marks • Laboratory performance = 10 marks Based on the performance and satisfactory completion of assigned laboratory work

Analysis of Algorithms Syllabus

Course Code	Course Name	Teaching Scheme (Hrs. / Week)			Credits Assigned			
25CE3PCC02	Analysis of Algorithms	**L**	**T**	**P**	**L**	**T**	**P**	**Total**
		3	-	2	3	-	1	4
		Examination Scheme						
			CA		**MSE**	**ESE**		**Total**
		Theory	20		30	50		100
			Term work	Practical				
		Lab	25	25				50

Pre-Requisite Courses:	1. 25FE2PCC01**:** Data structures 2. 25FE1VSEC02: Problem Solving using C Programming

Course Objectives:

1. To introduce foundational concepts of algorithm analysis, including time and space complexity and recurrence solving.
2. To understand and apply various algorithm design techniques for solving computational problems.
3. To develop skills for analyzing and comparing algorithm efficiency using mathematical reasoning.
4. To cultivate the skill of selecting, refining, and adapting algorithms to meet real-world problem requirements.
5. To promote logical thinking and creativity in designing optimized and scalable algorithms.

Course Outcomes	After successful completion, the students will be able to	
	CO1	Recall and define fundamental terms and concepts related to algorithm design and complexity analysis. (Remembering)
	CO2	Explain various algorithm design paradigms and the principles underlying their operation. (Understanding)
	CO3	Apply appropriate algorithmic strategies to develop solutions for computational problems. (Applying)
	CO4	Analyze and interpret the efficiency of algorithms using asymptotic notations and recurrence relations. (Analyzing)
	CO5	Evaluate alternative algorithmic approaches to determine the most suitable solution for a given problem. (Evaluating)
	CO6	Design and construct innovative algorithms by integrating multiple strategies for solving complex and real-world problems. (Creating)

Syllabus:

Module No.	Unit No.	Topics	Hours
1		**Module 1: Introduction to Analysis of Algorithms**	**08**
	1.1	Mathematical Background for Algorithm Analysis: Functions and growth rates, Summations and series, Proof techniques.	

		Space and time complexity: Time Complexity-Worst, average & best and their measurement method, Space complexity calculation. Review of Asymptotic notation: defining and understanding the function with graph and example for: Big O (O), Omega (Ω), Theta (Θ).	
	1.2	Solving Recurrences: Formulating the recurrences relations, Recursion Tree Method, visualizing recursive call as a tree for all cases like highest cost of root, highest cost of leaf nodes & all nodes with same cost, Substitution/ iterative method (forward & backward substitution), and Master's theorem (with fourth condition).	
	1.3	Priori time complexity analysis of selection sort, insertion sort algorithms with examples and logic understanding.	
Self-Learning		NP-completeness basics, Algorithm complexity classes (P, NP, NP-hard),	
2		**Module 2: Divide and Conquer Approach**	
	2.1	General method- concepts, understanding divide & conquer steps- Divide-Conquer-Combine.	**06**
	2.2	Understanding procedure, solving examples and analysis for all cases for the following algorithms: Merge sort, Quick sort, Binary search and Strassen's Matrix Multiplication.	
Self-Learning		Max-Min problem	
3		**Module 3: Greedy Method Approach**	
	3.1	Greedy Method: Introduction, principle of optimality, greedy choice property, optimal substructure, abstract greedy algorithm.	**07**
	3.2	Example solving, mathematical formulation and analysis for: 1. Graphs: Single source shortest path (Dijkstra's algorithm) with limitations 2. Minimum Spanning trees algorithms: Prim's and Kruskal's 3. Fractional knapsack problem	
Self-Learning		Job sequencing with deadline	
4		**Module 4: Dynamic Programming Approach**	
	4.1	Dynamic Programming Approach- Introduction, Key properties: Optimal substructure, Overlapping subproblems, Implementation styles: Top-down (memoization) vs. Bottom-up (tabulation).	**13**
	4.2	Example solving, applications, mathematical formulation and analysis for:	

		1. 0/1 knapsack (table and set method) 2. All pairs shortest paths (Floyd Warshall Algorithm) 3. Single source shortest path (Bellman-Ford Algorithm) 4. Longest Common Subsequence (LCS) 5. Matrix Chain Multiplication 6. Travelling salesperson problem 7. Multistage graphs	
Self-Learning		Fibonacci problem, Coin Changing problem	
5		**Module 5: Backtracking and Branch and bound**	
	5.1	Backtracking: Introduction, Basics of backtracking, Key Characteristics, Applications. Example solving, mathematical formulation and analysis for: 1. N–queen problem 2. Sum of subsets 3. Graph coloring	**07**
	5.2	Branch and bound: Introduction, Basics of Branch and Bounds, Key Characteristics, Difference from Backtracking, Applications. Example solving, mathematical formulation and analysis for: 1. Travelling Salesperson Problem, 2. 15 Puzzle problems.	
Self-Learning		Hamiltonian cycles	
6		**Module 6: String Matching Algorithms**	
	6.1	Concept and Working Principle, Algorithmic Steps, Applications, and Advantages/ Limitations, Complexity Analysis of: 1. The Naïve string-matching algorithm 2. The Rabin Karp algorithm 3. The Knuth-Morris-Pratt algorithm	**04**
Self-Learning		String matching with finite automata	
		TOTAL	**45**

Laboratory Experiments (Minimum 10)

Exp No.	Experiment Title
1	Comparative Growth Analysis of Functions Using Asymptotic Notations
2	Implementation Min-Max Algorithm
3	Implementation of Job Sequencing with deadlines.

4	Experiment based on divide and conquers approach. (Merge sort, Quick sort, Binary search)
5	Experiment based on greedy approach.(Single source shortest path- Dijkstra Fractional Knapsack problem, Minimum cost spanning trees-Kruskal and Prim's algorithm)
6	Experiment using dynamic programming approach (All pair shortest path- Floyd Warshall, 0/1 knapsack)
7	Implementation of Bellman Ford Algorithm using Dynamic programming
8	Travelling salesperson problem, Longest common subsequence
9	Experiment based on graph Algorithms (BFS, DFS)
10	Experiment using Backtracking strategy. (N-queen problem, Sum of subsets, Graph coloring)
11	Travelling Salesperson Problem Using Branch and Bound
12	Experiment based on string matching/amortized analysis (The Naïve string-matching Algorithms, The Rabin Karp algorithm, The Knuth-Morris-Pratt algorithm.
13	Implementation of Strassen's Matrix Multiplication.
14	Implementation of 15 Puzzle problem using Backtracking

Text Books:

1. T.H.Coreman , C.E. Leiserson,R.L. Rivest, and C. Stein, "Introduction to algorithms", 4th Edition, The MIT Press, 2022.
2. Ellis Horowitz, Sartaj Sahni and Susan Anderson-Freed, "Fundamentals of Data Structures in C", 2nd Edition, W. H. Freeman and Company, 2008.

Reference Books:

1. Sanjoy Dasgupta, Christos Papadimitriou, Umesh Vazirani, "Algorithms", Tata McGraw- Hill Edition, 2023.
2. John Kleinberg, Eva Tardos, "Algorithm Design", Pearson, 2005.
3. Michael T. Goodrich, Roberto Tamassia, "Algorithm Design", Wiley Publication, 2014.
4. S. K. Basu, "Design Methods and Analysis of Algorithm", PHI

Useful Links:

1. https://nptel.ac.in/courses/106/106/106106131/
2. https://swayam.gov.in/nd1_noc19_cs47/preview
3. https://www.coursera.org/specializations/algorithms
4. https://www.mooc-list.com/tags/algorithms

Assessment Methodology

Assessment Tools	Marks Distribution
Continuous Assessment (CA) **(20 Marks)**	• Certification: NPTEL (20 Marks) (Approved by instructor) o OR • Any two Pedagogies (10 marks each) • MCQ /Class Test • Case study/Assignment • GATE based Tutorial • Certification: Udemy/Coursera (Approved by instructor) • Open Book Test • Working model / simulation of a course-based concept.
Mid Semester Examination (MSE) **(30 Marks)**	Question Paper Pattern is as follows: • All Questions are compulsory. • Q1 A or B - 10 marks • Q2 A or B - 10 marks • Q3 A or B - 10 marks For each question, A and B should be based on the same CO. MSE should be based on 50% syllabus. Time: 90 minutes (1 hour 30 minutes) Total Marks: 30
End Semester Examination (ESE) (50 Marks)	Question Paper Pattern is as follows: • All Questions are compulsory. • Q1 A or B - 10 marks • Q2 A or B - 10 marks • Q3 A or B - 10 marks • Q4 A or B - 10 marks • Q5 A or B - 10 marks

	For each question, A and B should be based on the same CO. ESE should be based on 30% syllabus of MSE and 70% syllabus after MSE. Time: 120 minutes (2 hours) Total Marks: 50
Term Work (25 Marks)	• Active Participation (Lab) = 5 marks • Laboratory Report = 10 marks • Laboratory performance = 10 marks Based on the performance and satisfactory completion of assigned laboratory work
Practical and Oral (25 Marks)	Practical and Oral examination will be based on the entire syllabus

Computer Organization and Architecture

<table>
<tr><th>Course Code</th><th>Course Name</th><th colspan="3">Teaching Scheme (Hrs. / Week)</th><th colspan="4">Credits Assigned</th></tr>
<tr><td rowspan="8">25CE3PCC03</td><td rowspan="8">Computer Organization and Architecture</td><td>L</td><td>T</td><td>P</td><td>L</td><td>T</td><td>P</td><td>Total</td></tr>
<tr><td>03</td><td>-</td><td>02</td><td>03</td><td>-</td><td>01</td><td>04</td></tr>
<tr><td colspan="7">Examination Scheme</td></tr>
<tr><td></td><td colspan="2">CA</td><td>MSE</td><td>ESE</td><td colspan="2">Total</td></tr>
<tr><td>Theory</td><td colspan="2">20</td><td>30</td><td>50</td><td colspan="2">100</td></tr>
<tr><td></td><td>Term Work</td><td>Practical Exam</td><td>-</td><td>-</td><td colspan="2">Total</td></tr>
<tr><td>Lab</td><td>25</td><td>-</td><td>-</td><td>-</td><td colspan="2">25</td></tr>
</table>

Pre-Requisite Courses:	1. 25FE1ESC02-Basic Electrical and Digital Electronics
	2. 25FE1VSEC02-Problem Solving using C programming
	3. 25FE2VSEC02-Object Oriented Programming using Python

Course Objectives:

1. Build a strong understanding of computer organization, architecture models, and performance measures.
2. Teach data representation, arithmetic operations, and processor control concepts.
3. Explain memory organization, cache, virtual memory, and I/O systems.
4. Introduce advanced concepts like pipelining, parallel processing, and modern processor architectures.

Course Outcomes	After successful completion of the course, the students will be able to	
	CO1	Recall the fundamental concepts of computer organization and Architecture.(Remembering)
	CO2	Describe various parallel processing mechanisms and types of computer buses, highlighting their characteristics and applications. (Understanding)
	CO3	Develop and implement methods for designing control units or

		memory units in computing systems. (Applying)
	CO4	Examine and differentiate memory hierarchies, cache mapping techniques, and virtual memory systems. (Analyzing)
	CO5	Assess the correctness and efficiency of ALU operations for various arithmetic computations. (Evaluating)
	CO6	Develop assembly programs and simulations to demonstrate computational tasks, ALU functions, and cache operations. (Creating)

Syllabus:

Module No.	Unit No.	Topics	Hours
1		**Module 1: Overview of Computer Architecture and Organization**	**4**
	1.1	Introduction of Computer Organization and Architecture, Basic organization of Computer, Von Neumann model, Harvard Model	
	1.2	Performance measures – CPI, speedup, efficiency, throughput and Amdahl's law, numerical on performance parameters	
	Self-Learning Topic: Block level description of the functional units, Evolution of x86 Computers		
2		**Module 2: Data Representation and Arithmetic Algorithms**	**12**
	2.1	Binary Number representation: Sign Magnitude, 1's and 2's Complement representation.	
	2.2	Integer Data Computation: Addition and subtraction using 2's Complement method	
	2.3	Booth's multiplication Algorithms, unsigned division algorithms-Restoring and Non restoring, signed division algorithm	
	2.4	IEEE 754 Floating point representation	
	2.5	Floating point arithmetic: Addition, Subtraction, Multiplication, Division	

	Self-Learning Topic: Modified Booth Encoding – Used in modern processors (Intel) to reduce the number of partial products		
3		**Module 3: Processor Organization and Control Unit Design**	**7**
	3.1	Case study of 8086 Processor: Architecture of 8086 processor, register organization, instruction cycle, phases of instruction, addressing modes.	
	3.2	Control Unit: Soft wired (Micro-programmed) and hardwired control unit design methods. Microinstruction sequencing and execution. Micro operations	
	Self-Learning Topic: Wilkie's microprogrammed control unit		
4		**Module 4: Memory Systems Organization**	**10**
	4.1	Introduction to Memory and Memory parameters, Classification of memories, Memory hierarchy and characteristics.	
	4.2	Cache memory Concepts, Locality of reference, design problems based on mapping techniques. Cache Coherency, Write Policies, Interleaved and Associative memory	
	4.3	Virtual Memory: Concept, Segmentation and Paging, Address translation mechanism	
	Self-Learning Topic: Case study of Pentium Processor Cache Memory Model (MESI Protocol)		
5		**Module 5: I/O Organization**	**04**
	5.1	Buses: Types of Buses, Bus Arbitration	
	5.2	Types of data transfer techniques: Programmed I/O, Interrupt driven I/O and DMA.	
	5.3	Interrupt types, Interrupts handling	
	Self-Learning Topic: Types of Buses-PCI, USB, SATA		
6		**Module 6: Advanced Processor Principles**	**8**
	6.1	Introduction to parallel processing concepts, Difference between Parallel processing and pipelining, Flynn's classifications	

	6.2	Pipelined Architecture: Pipeline Stages, Pipeline Hazards-Structural, data and control ,Mitigation of Hazards	
	6.3	Introduction to Superscalar and VLIW architectures, multithreaded processor, out-of-order execution and multi-threaded processors.	
	6.4	Introduction to Multi-core processor architecture	
	Self-Learning Topic: Case study on Tensor Processing Units (TPU)		
		TOTAL	**45**

Laboratory Experiments (Minimum 08):

Sr. No.	Experiment Titles
1	Installation and configure: DOS, MASM, Debug, X86 Mode or 8086 emulator.
2	Assembly Language Programming based on Arithmetic Operations-addition and subtraction
3	Assembly Language Programming based on Arithmetic Operations-multiplication and division.
4	Assembly Language Programming based on Bit Wise Operations.
5	Implement various String Operations in 8086 using assembly language.
6	Assembly Language Programming based on code conversion
7	Block Transfer and Block Exchange using Index Registers
8	Assembly Language Programming based on flag register contents.
9	Design Password Detection Application using BIOS and DOS interrupts along with 8086 instructions.
10	Implement file operations [DOS Interrupts in C/MASM]
11	Implement I/O interfacing using inbuilt speakers of IBM PC

12	Implement Booth's Multiplication Algorithm.
13	Implement Non-Restoring Division Algorithm.
14	Implement Restoring Division Algorithm.
15	Implementation of Mapping techniques of Cache memory.
16	Simulation of LED blinking using I/O ports.
17	Implementation of ALU operations.

Text Books:

1. William Stalling, "Computer Organization and Architecture: Designing and Performance", Pearson Publication 10TH Edition.

2. John P Hayes, "Computer Architecture and Organization", McGraw-Hill Publication, 3RD Edition.

3.John Uffenbeck, "8086/8088 family: Design Programming and Interfacing", PHI.

Reference Books:

1. Andrew S. Tanenbaum, "Structured Computer Organization", Pearson Publication.

2. B.Govindarajalu, "Computer Architecture and Organization", McGraw-Hill Publication.

3. Carl Hamacher, Zvonko Vranesic and Safwat Zaky "Computer Organization" Tata

McGraw-Hill,5th edition,2002.

Useful Links:

1. https://www.classcentral.com/course/swayam-computer-organization-and-architecture-a-pedagogical-aspect-9824
2. https://www.coursera.org/learn/comparch
3. http://vlabs.iitkgp.ac.in/coa/

Assessment Methodology:

Assessment Tools	Marks Distribution
Continuous Assessment (CA)	• Certification: NPTEL (20 Marks) (Approved by instructor) ○ OR

(20 Marks)	• Any two Pedagogies (10 marks each) • MCQ /Class Test • Case study/Assignment • GATE based Tutorial • Certification: Udemy/Coursera (Approved by instructor) • Open Book Test • Working model / simulation of a course-based concept.
Mid Semester Examination (MSE) **(30 Marks)**	Question Paper Pattern is as follows: • All Questions are compulsory. • Q1 A or B - 10 marks • Q2 A or B - 10 marks • Q3 A or B - 10 marks For each question, A and B should be based on the same CO. MSE should be based on 50% syllabus. Time: 90 minutes (1 hour 30 minutes) Total Marks: 30
End Semester Examination (ESE) (50 Marks)	Question Paper Pattern is as follows: • All Questions are compulsory. • Q1 A or B - 10 marks • Q2 A or B - 10 marks • Q3 A or B - 10 marks • Q4 A or B - 10 marks • Q5 A or B - 10 marks For each question, A and B should be based on the same CO. ESE should be based on 30% syllabus of MSE and 70% syllabus after MSE. Time: 120 minutes (2 hours)

	Total Marks: 50
Term Work (25 Marks)	• Active Participation (Lab) = 5 marks • Laboratory Report = 10 marks • Laboratory performance = 10 marks Based on the performance and satisfactory completion of assigned laboratory work

Database Management Systems

Course Code	Course Name	Teaching Scheme (Hrs. / Week)		Credits Assigned			
		L	**P**	**L**	**T**	**P**	**Total**
		03	**02**	**03**	**-**	**01**	**04**
		Examination Scheme					
			CA	**MSE**	**ESE**	**Total**	
25CE3PCC04	**Database Management Systems**	**Theory**	**20**	**30**	**50**	**100**	
			Term work	**Practical Exam**	**-**	**Total**	
		Lab	**25**	**25**	**-**	**50**	

Pre-Requisite Courses:	25FE2PCC01: Program Core Course (Data Structures)

Course Objectives:

1. To impart foundational knowledge of database concepts, architectures, models, and their practical significance.
2. To develop students' ability to design, model, and implement efficient databases using Entity-Relationship (ER) and Enhanced Entity-Relationship (EER) models, relational schemas, Structured Query Language (SQL) and NoSQL.
3. To enhance analytical skills for evaluating and improving database designs through normalization and transaction management techniques.
4. To introduce students to modern databases like Cloud, Geographic Information Systems (GIS), and NoSQL, helping them adapt to new data management technologies.

Course Outcomes	After successful completion, the students will be able to:	
	CO1	Recognize the need for a database management system and recall its fundamental concepts. (Remembering)
	CO2	Interpret ER and EER diagrams, relational schemas, and normalization concepts in database design. (Understanding)
	CO3	Use relational algebra, SQL and NoSQL to create, manipulate, and retrieve data from databases. (Applying)
	CO4	Examine database schema to detect functional dependencies, anomalies, design issues, and apply techniques for optimizing database designs. (Analyzing)
	CO5	Assess database designs, transaction mechanisms, and concurrency controls for efficiency and reliability. (Evaluating)
	CO6	Design conceptual database schemas using ER/EER modelling for real-world applications. (Creating)

Syllabus:

Module No.	Unit No.	Topics	Hours
1		**Module 1: Introduction to Database Concepts and Entity-Relationship Modelling**	**07**
	1.1	Core DBMS concepts: Introduction, characteristics, architecture, Data abstraction, Data Independence, File system v/s Database system, Database Users and Database Administrator, Real-World Applications of DBMS, Modern Database Paradigms: Cloud and GIS Databases	
	1.2	The Entity-Relationship (ER) Model: Entity types: Weak and strong entity sets, Entity sets, Types of Attributes, Keys, Relationship constraints: Cardinality and Participation, Extended Entity-Relationship (EER) Model: Generalization, Specialization, Aggregation and Categories (Union types)	
	Self-Learning Activity: Time-series databases and their use cases		
2		**Module 2: Relational Model and Relational Algebra**	**08**
	2.1	Introduction to the Relational Model, relational schema and concept of keys	
	2.2	Mapping the ER and EER Model to the Relational Model	
	2.3	Relational Algebra-operators (Unary, Binary, Set Operations, Aggregate functions), Relational Algebra Queries	
	Self-Learning Activity: Create any ten relational algebra queries to extract meaningful insights from the dataset.		
3		**Module 3: Structured Query Language (SQL)**	**10**
	3.1	Overview of SQL, Data Definition Commands (DDL), Integrity constraints: key constraints, Domain Constraints, Referential integrity, check constraints, Data Manipulation commands (DML)	
	3.2	Data Control commands (DCL), Transaction control commands (TCL), Inbuilt functions (Math, Date, String) Set operations, aggregate functions, group by clause and having clause	
	3.3	Views in SQL, joins, Nested and complex queries	
	3.4	Cursor, Stored Procedure, Triggers	
	Self-Learning Activity: Create SQL queries to build complex views and triggers for specific purposes.		
4		**Module 4: Relational-Database Design**	**06**
	4.1	Pitfalls in Relational-Database Designs, Function Dependencies, Inference Rules for Functional Dependencies	
	4.2	Concept of normalization, First Normal Form, 2NF, 3NF, BCNF	
	Self-Learning Activity: Practice identifying the highest normalization level for a given relational schema.		
5		**Module 5: Transactions Management, Concurrency, and**	**06**

		Recovery	
	5.1	Transaction concept, Transaction states, ACID properties, Transaction Control Commands, Concurrent Executions, Serializability-Conflict and View, Concurrency Control: Lock-based, Timestamp-based protocols	
	5.2	Recovery System: Log-based recovery, Deadlock handling	
	5.3	Database design optimization factors, including normalization level, primary and foreign key structure for redundancy control, locking strategy, ACID compliance, and deadlock prevention	
	Self-Learning Activity: Simulation of Deadlock		
6		**Module 6: Introduction of NoSQL**	**08**
	6.1	Overview of NoSQL: Definition, Features, Types, Differences between SQL and NoSQL	
	6.2	Popular NoSQL Systems (MongoDB, Cassandra, Redis, Neo4j)	
	6.3	CRUD operations in MongoDB, Real-world applications of NoSQL	
	Self Learning Activity: Analyze real-world scenarios to identify data needs and recommend the most suitable NoSQL database type with justification.		
		TOTAL	**45**

Laboratory Experiments (Minimum 10)

Sr. No.	**Experiment Titles**
1	Design an ER/ EER model for the assigned Case Study.
2	Mapping the ER/EER Model to the Relational Model.
3	Create a database using DDL commands for the assigned system.
4	Populate database using DML commands for the assigned system.
5	Apply Integrity Constraints for the specified system.
6	Implement TCL and DCL commands.
7	Implement SQL Functions (string case and manipulation operations, Numeric, Date & Time functions), Group By and Order By clause.
8	Perform Sub Queries, Nested Queries and Joins.
9	Perform Views and Triggers.
10	To create and execute a stored procedure that uses a cursor to fetch and display employee records one row at a time from the Employees table, demonstrating row-by-row processing in a database management system.

11	Basic CRUD Operations in MongoDB.
12	Practice using data aggregation techniques to analyze information stored in a MongoDB collection.
13	Develop one database application (for the assigned application) using Flask with MySQL and apply CURD operations.

Text Books:

1. Abraham Silberschatz, Henry F. Korth, and S. Sudarshan, "Database System Concepts", 7th Edition, McGraw Hill in 2019
2. Ramez Elmasri and Shamkant B. Navathe, "Fundamentals of Database Systems", 7th Edition, Pearson Education in 2015
3. Raghu Ramakrishnan and Johannes Gehrke, "Database Management Systems", 3rd Edition, Tata McGraw-Hill Education in 2003

Reference Books:

1. Peter Rob and Carlos Coronel, "Database Systems: Design, Implementation, and Management", 14th Edition, Thomson Learning in 2022
2. G. K. Gupta, "Database Management Systems", McGraw Hill, 2018 Edition
3. Dr. P.S. Deshpande, "SQL and PL/SQL for Oracle 10g", Black Book, Dreamtech Press in 2006

Useful Links:

1. https://nptel.ac.in/courses/106105175
2. https://www.coursera.org/learn/database-structures-and-management-with-mysql
3. https://www.w3schools.com/sql/
4. https://www.geeksforgeeks.org/dbms/dbms/

Assessment Methodology:

Assessment Tools	Marks Distribution
Continuous Assessment (CA) **(20 Marks)**	• Certification: NPTEL (20 Marks) (Approved by instructor) OR • Any two Pedagogies (10 marks each) • MCQ /Class Test • Case study/Assignment • GATE based Tutorial • Certification: Udemy/Coursera (Approved by instructor) • Open Book Test • Working model / simulation of a course-based concept.

Mid Semester Examination (MSE) **(30 Marks)**	Question Paper Pattern is as follows: • All Questions are compulsory. • Q1 A or B - 10 marks • Q2 A or B - 10 marks • Q3 A or B - 10 marks For each question, A and B should be based on the same CO. MSE should be based on 50% syllabus. Time: 90 minutes (1 hour 30 minutes) Total Marks: 30
End Semester Examination (ESE) (50 Marks)	Question Paper Pattern is as follows: • All Questions are compulsory. • Q1 A or B - 10 marks • Q2 A or B - 10 marks • Q3 A or B - 10 marks • Q4 A or B - 10 marks • Q5 A or B - 10 marks For each question, A and B should be based on the same CO. ESE should be based on 30% syllabus of MSE and 70% syllabus after MSE. Time: 120 minutes (2 hours) Total Marks: 50
Term Work (25 Marks)	• Active Participation (Lab) = 5 marks • Laboratory Report = 10 marks • Laboratory performance = 10 marks Based on the performance and satisfactory completion of assigned laboratory work
Practical and Oral (25 Marks)	Practical and Oral examination will be based on the entire syllabus

Full Stack Java Programming

Course Code	Course Name	Teaching Scheme (Hrs. / Week)			Credits Assigned			
		L	**T**	**P**	**L**	**T**	**P**	**Total**
			_	02*+02		--	02	02
		Examination Scheme						
			CA		**MSE**	**ESE**	**Total**	
25SE3VSEC01	**Full Stack Java Programming**	**Theory**	–		–	–	–	
			Termwork	**Practical Exam**	**_**	**_**	**Total**	
		Lab	25	25	_	_	50	

* Two hours of demo/discussion for entire class

Pre-Requisite Courses:	25FE1VSEC02 - Problem Solving using C programming.

Course Objectives:

1. To equip students with a strong foundation in Java programming and OOP concepts for application development.
2. To enable learners to design and integrate client-side and server-side solutions effectively.
3. To foster skills in modern frameworks like React and Spring Boot for building scalable and maintainable applications.
4. To cultivate analytical, debugging, and problem-solving abilities for full stack development projects.

	After successful completion, the students will be able to	
Course Outcomes	CO1	Describe the fundamental concepts of Java programming, OOP principles, and web technologies. (Remembering)
	CO2	Explain and demonstrate use of inheritance, exception handling, JDBC, Servlets, JSP, React, and Spring framework in application development. (Understanding)
	CO3	Implement Java-based programs and integrate client-side and server-side components to build functional applications.
	CO4	Analyze Java programs to identify and fix runtime errors using exception handling and multithreading mechanisms. (Applying)
	CO5	Assess the efficiency, scalability, and maintainability of Java full stack applications and suggest improvements. (Analyzing)
	CO6	Design and develop complete full stack solutions using Java, databases, front-end frameworks, and Spring Boot for real-world use cases. (Creating)

Syllabus:

Module No.	Unit No.	Topics	Hours
1		**Introduction to OOP and Java Basics**	**05**
	1.1	**Java Basics**: Java platform overview: JDK, JRE, JVM, Data types, variables, Methods, operators, expressions, Control structures (if-else, loops),Array and Strings	
	1.2	**OOP concepts:** Objects, class, Encapsulation, Abstraction, Inheritance, Polymorphism, message passing. Branching and looping. Class, object, data members, member functions, Constructors, types, static members and functions, Method overloading Input and output functions in Java, Packages in Java, types, user defined packages	
	Self-Learning	Vectors in Java	
2		**Inheritance, Exception Handling and Multithreading**	**06**
	2.1	**Inheritance:** Types of inheritance, Method overriding, super, abstract class and abstract method, final, Multiple inheritances using interface, extends keyword.	
	2.2	**Exception Handling**: try, catch, finally, throw and throws, Multiple try and catch blocks,	
	2.3	Multithreading in Java	
	Self-Learning	User defined exception handling	
3		**Client side Programming**	**05**
	3.1	**Client Side Scripting:** **HTML:** Elements, Attributes, Head, Body, Hyperlink, Formatting Images, Tables, List, Frames, Forms. **CSS3:**Syntax, Inclusion, Color, Background, Fonts, Tables, lists, CSS3 selectors.	
	3.2	**Java Script:** Introduction to JavaScript: Conditionals Statements, Loops, Functions, Arrays, Objects, Control Flow, Math Function, Browser Object Model, Document Object Model.	
	Self-Learning	Load a text file content into a web page using AJAX	
4		**Advanced Java Concepts**	**05**
	4.1	**Java Database Connectivity (JDBC):** JDBC architecture and drivers Connecting to databases (MySQL, Oracle, etc.) Executing SQL queries using Java Statements.	
	4.2	**Server side programming in Java:** Introduction of Servlet, Servlet lifecycle, Servlet Request, Servlet Response, Servlet Context,	

		HTTP Sessions, Handling forms and user inputs,	
	4.3	**Session management:** Introduction to Java Server Pages, JSP architecture and page directives, Components of a JSP, Scripting elements and Standard actions, Method Definitions, JSTL.	
	Self-Learning	Database Connectivity in Servlets	
5		**Web Programming using React JS**	**05**
	5.1	**Design Pattern:** Understanding MVC architecture Implementing MVC with servlets and JSP Developing a complete web application Solving company's use cases	
	5.2	**React Framework:** Introduction to React JS, Components and Elements of React, Rendering Components, React State and Props, Events, Hooks, Routing Conditional Rendering, Lists and Keys, Forms, create a single page application using React.	
	Self-Learning	Unidirectional data flow in FLUX	
6		**Spring boot for microservices**	**04**
	6.1	**Spring Framework:** Introduction to Microservices, Dependency injection and Inversion of Control (IoC), Spring annotations, Database integration	
	6.2	Creating spring boot applications, Building RESTful APIs with spring boot.	
	Self-Learning	Aspect-oriented programming using Spring	
		TOTAL	**30**

Suggested List of Laboratory Experiments (Any 10 Experiments)

Sr. No	Experiment Title
1	Programs on Classes and Objects
2	Programs on Method and Constructor Overloading
3	Programs on Inheritance and Exception Handling
4	Program on Multi-threading
5	Program on Implementing Generic and HTTP servlet.
6	Design a login webpage in JSP that makes validation through Database using JDBC and call the servlet for various operations
7	Program on Implicit and Explicit objects in JSP
8	Program to create a website using HTML CSS and JavaScript
9	Program using Java Script to validate the email address entered by the user (check an alert box

	reporting the error and ask the user to re-enter it again).
10	Program based on Document Object Model to change the background color of the web page automatically after every 5 seconds.
11	Creating a Single Page website using the concepts in React like Hooks, Router, Props and States.
12	Program to create a Monolithic Application using SpringBoot
13	**Mini Project**

Text books:

1.Herbert Schildt, "Java The Complete Reference" Ninth Edition, Oracle Press

2.Christopher Schmitt and Kyle Simpson, "HTML5 Cookbook",O'Really Press

3.Nicholas C. Zakas,"Professional JavaScript for Web Developers", Wiley Publishing

4. Amuthan G., "Spring MVC, Beginners Guide" Pakt Publication

5. Chris Minnick,"BEGINNING ReactJS Foundations Building User Interfaces with ReactJS", Wrox publication

6.Iuliana Cosmina, Rob Harrop, "Pro Spring 5 An In-Depth Guide to the Spring Framework and Its Tools",Fifth Edition,APress

Reference Books:

1. Laura Lemay, Charles L. Perkins","Teach Yourself JAVA in 21 Days",Sams.net Publishing

2. Eureka, Ribbon, Zuul and Cucumber Moises Macero,"Learn Microservices with Spring Boot A Practical Approach to RESTful Services using RabbitMQ", APress

3.Alex Banks & Eve Porcello," React FUNCTIONAL WEB DEVELOPMENT WITH REACT AND REDUX", O'Really Press

Useful Links

1. https://www.w3schools.com/java/
2. https://docs.oracle.com/javase/tutorial/
3. https://www.w3schools.com/REACT/DEFAULT.ASP
4. https://www.javatpoint.com/html5-tutorial
5. nptel.ac.in/courses/106105191

Assessment Methodology:

Assessment Tools	Marks Distribution
Term Work (25 Marks)	• Active Participation (Lab) = 5 marks

	• Laboratory Report = 10 marks • Laboratory performance = 10 marks Based on the performance and satisfactory completion of assigned laboratory work
Practical and Oral (25 Marks)	Practical and Oral examination will be based on the entire syllabus

Sustainable Development

<table>
<tr><td>Course Code</td><td>Course Name</td><td colspan="4">Teaching Scheme (Hrs. / Week)</td><td colspan="4">Credits Assigned</td></tr>
<tr><td rowspan="8">25IL3VEC01</td><td rowspan="8">Sustainable Development</td><td>L</td><td colspan="2">P</td><td>T</td><td>L</td><td>P</td><td>T</td><td>Total</td></tr>
<tr><td>2</td><td colspan="2">-</td><td>-</td><td>2</td><td>-</td><td>-</td><td>2</td></tr>
<tr><td colspan="9"></td></tr>
<tr><td rowspan="2">Theory</td><td>CA</td><td>MSE</td><td colspan="3">ESE</td><td colspan="2">Total</td></tr>
<tr><td>50</td><td>-</td><td colspan="3">-</td><td colspan="2"></td></tr>
<tr><td rowspan="2">Lab</td><td>TW</td><td>OR</td><td colspan="3">PR</td><td colspan="2">Total</td></tr>
<tr><td>-</td><td>-</td><td colspan="3">-</td><td colspan="2">-</td></tr>
<tr><td>Total</td><td colspan="8">50</td></tr>
</table>

Course Objectives

1. To introduce students to the role of AI, IoT, and ICT in solving India's pressing socio-economic and environmental challenges.
2. To enable learners to understand and apply these technologies for achieving the UN Sustainable Development Goals (SDGs) in the Indian context.
3. To encourage innovative thinking, problem-solving, and use-case development for sustainable growth.

<table>
<tr><td rowspan="6">Course Outcomes</td><td colspan="2">After successful completion of the course, the students will be able to:</td></tr>
<tr><td>CO1</td><td>Understand fundamental concepts of AI, IoT, and ICT for sustainable development. (Remembering)</td></tr>
<tr><td>CO2</td><td>Identify key areas of sustainable development in India that can benefit from technology. (Understanding)</td></tr>
<tr><td>CO3</td><td>Work collaboratively on an activity to address a societal or environmental issue. (Applying)</td></tr>
<tr><td>CO4</td><td>Analyse real-world case studies and technology-led development models. (Analyzing)</td></tr>
<tr><td>CO5</td><td>Evaluate the ethical, environmental, and policy implications of digital interventions. (Evaluating)</td></tr>
</table>

	CO6	Propose innovative solutions to local and national challenges using AI, IoT, and ICT (Creating).

Module No.	Unit No.	Topics	Hours
1	**Introduction to Sustainable Development in India.**		5
	1.1	Overview of UN SDGs and India's priorities (poverty, health, education, environment) **National Missions: Digital India, Smart Cities, Startup India, Skill India**	
	1.2	Technology as an enabler of sustainable growth **Tools for sustainability assessment**	
	Self-Learning Topics: Study the progress and challenges of India in achieving each UN Sustainable Development Goal (SDG).		
2	**Fundamentals of AI, IoT, and ICT**		5
	2.1	Sustainability enablers using AI: ML, Deep Learning, NLP **IoT: Sensors, connectivity, cloud platforms**	
	2.2	ICT: Communication networks, mobile platforms, data systems Synergy between AI, IoT, and ICT	
	Self-Learning Topics: Watch beginner-level videos on Machine Learning, Deep Learning, and NLP (e.g., by Google AI, Fast.ai).		
3	**Sectoral Applications in Indian Context**		5
	3.1	Agriculture: Smart farming, crop prediction, irrigation management Healthcare: Telemedicine, diagnostics, health surveillance	
	3.2	Education: Personalized learning, digital classrooms Case studies: eNAM, eSanjeevani, DIKSHA	
	Self-Learning Topics: Explore eSanjeevani and how telemedicine reached rural India during COVID-19. Case Research: Choose a state and explore how AI/ICT helped during a health crisis.		
4	**Environment, Energy and Urban Development**		5
	4.1	AI and IoT in waste management and pollution control **Smart grids and renewable energy systems**	
	4.2	ICT in climate action and disaster management **Case studies: Smart Cities Mission, Jal Shakti, PM-KUSUM**	
	Self-Learning Topics: Study how smart bins work using IoT and AI (e.g., Swachh Bharat implementations).		
5	**Innovation, Startups, and Ethical Concerns**		5
	5.1	Role of innovation hubs and social enterprises Frugal innovation for rural and tribal India	

	5.2	Data privacy, algorithmic bias, digital inclusion Policy frameworks: NDCP, India AI Strategy, Data Protection Bill	
	Self-Learning Topics: Analyze the impact of the Digital Personal Data Protection Act 2023 on startups and citizens.		
6	**Sustainable Solutions in India (Case India)**		**5**
	6.1	Students identify a sustainable development challenge Discuss an AI/IoT/ICT-based solutions	
	6.2	Review of Digital Transformation initiatives by Government of India	
	Self-Learning Topics: Explore past Smart India Hackathon or Toycathon projects for inspiration. Watch a tutorial on how to build a simple AI/IoT prototype (e.g., Smart Dustbin, Health Monitor). Learn how to use Canva, Figma, or PowerPoint for visualizing your project idea. Practice a pitch using a simple template: Problem → Solution → Impact → Tech Used.		
		TOTAL	**30**

Syllabus

Text Books:

1. *Niti Aayog SDG India Index 2023-24: Towards Viksit Bharat*, NITI Aayog, 2023–24, Government of India Publication.
2. Ahlawat, Ajay. *Sustainable Development Goals: Directive Principles for Sustainable India by 2030*. 8 October 2019, E-Book.
3. Mishra, Ankita, Banerjee, Sourik, & Singh, Brijendra. *Deep Learning Techniques for Smart Agriculture Applications*. Publishing 26 August 2025, IGI Global Publication.
4. Acharya, Biswaranjan (Ed.), Dey, Satarupa (Ed.), & Zidan, Mohammed (Ed.). *IoT-based Smart Waste Management for Environmental Sustainability*. 26 August 2024, CRC Press, Taylor & Francis Group.
5. Satsangi, Prem Saran. *Role of Communities in Achieving Sustainable Development*. 16 May 2024, Academic Foundation India.

Reference Books:

1. "Frugal Innovation: How to Do More with Less" – Navi Radjou, Jaideep Prabhu.
2. "Artificial Intelligence: A Guide for Thinking Humans" – Melanie Mitchell
3. "IoT Fundamentals: Networking Technologies, Protocols, and Use Cases for the Internet of Things" – David Hanes et al.
4. "Information and Communication Technology for Development (ICT4D)" – Tim Unwin
5. "AI and the Future of Humanity" – Rajan Gupta (NIT Rourkela)
6. "Smart Cities: Big Data, Civic Hackers, and the Quest for a New Utopia" – Anthony M. Townsend

Useful Links:

AI / IoT Learning Platforms

1. Teachable Machine by Google – Train simple AI models visually.
2. ThingSpeak IoT Platform – IoT data collection and analysis.
3. Arduino Project Hub – DIY IoT projects for beginners.
4. Google AI Hub – Demos, guides, and tools.

Sustainability and Indian Development Data

1. NITI Aayog SDG India Index – Dashboard for India's SDG progress.
2. India Smart Cities Dashboard – Real-time data and initiatives.
3. PM-KUSUM – Renewable energy for farmers.

Case Study Portals

1. Digital India Case Studies – Real examples of tech-enabled development.
2. eSanjeevani – India's official telemedicine platform.
3. DIKSHA Portal – Digital Infrastructure for Knowledge Sharing (Education).

Educational Videos

1. Fast.ai YouTube Channel – Friendly introductions to ML & Deep Learning.
2. NPTEL AI & ICT Courses – Free government-certified video courses (search: AI, ICT4D, IoT).
3. AI for Social Good by Google – Examples of AI for environmental and humanitarian use.

Term Work:

Students are expected to complete a minimum of 08 experiments and 2 assignments conducted in a batch-wise laboratory setting. Wherever applicable, students are encouraged to undertake computation or simulation-based experiments to deepen their conceptual understanding. Active participation, initiative, and creative engagement in all lab activities are expected.

Students should maintain a professional and positive attitude throughout the course demonstrated through regular attendance, timely submissions, collaboration with peers, and responsiveness to feedback.

Term work marks will be based on overall performance, including the quality and timely completion of experiments and assignments. Continuous grading will be carried out, and the final term work marks will be computed by averaging the converted grades earned across all experiments and assignments.

Course Code	Course Name	Teaching Scheme (Hrs. / Week)			Credits Assigned			
		L	T	P	L	T	P	Total
25CE3CEP01	Community Engagement Project	-	-	1	-	-	1	1

	CA	MSE	ESE	Term work	Total
Theory	-	-	-	25	25
Lab	-	-	-	-	

Pre-Requisites	Basic Programming Knowledge, Basic communication and teamwork abilities

Course Objectives:

1. Introduce the principles of Design Thinking in problem-solving.
2. Encourage empathy-based problem identification from real-world or campus needs.
3. Build skills for teamwork, creativity, and brainstorming.
4. Foster early-stage prototyping with open-source tools or simulations.

Course Outcomes	After successful completion of the course, the students will be able to	
	CO1	Identify real-world problems in community projects through empathy tools such as surveys, interviews, and observations. (Remembering)
	CO2	Explain the user needs and summarize findings to define a clear and concise problem statement. (Understanding)
	CO3	Apply design thinking principles to generate solution ideas through group collaboration and creative techniques. (Applying)
	CO4	Analyze user feedback and design options to compare the feasibility and impact of multiple proposed solutions. (Analyzing)
	CO5	Justify the chosen solution by evaluating alternatives based on usability, societal need, and sustainability. (Evaluating)
	CO6	Develop and demonstrate a conceptual prototype that effectively addresses the defined problem. (Creating)

Detailed Syllabus

Module No.	Unit No.	Topics	Hours
1		**Problem Identification through Empathy Tools**	06
	1.1	Introduction to Design Thinking and the Empathy stage. Identifying user needs through interviews, observations, and surveys.	
	1.2	Create empathy maps, user personas, and problem identification related to real-world or campus challenges.	
2		**User Research Analysis and Problem Definition**	04

	2.1	Analyzing empathy findings to define user needs. Framing clear and actionable problem statements.	
	2.2	Refining problems in community projects by using appropriate tools to understand experiences and perspectives.	
3	**Creative Ideation and Design Exploration**		**06**
	3.1	Creative idea generation using appropriate techniques and frameworks.	
	3.2	Sketching initial solutions and concept diagrams using appropriate methods and platforms.	
4	**Prototype Development Using Software Tools**		**06**
	4.1	Building low-fidelity prototypes using paper models, mockups, or suitable prototyping methods.	
	4.2	Developing partial working software prototypes using appropriate programming or block-based approaches.	
5	**User Testing and Design Evaluation**		**04**
	5.1	Testing prototypes with real users. Gathering feedback, usability testing, and identifying design flaws.	
	5.2	Analyzing feedback and iterating on prototype design to improve usability and functionality.	
6	**Solution Justification and Presentation**		**04**
	6.1	Final presentation of the prototype and solution: storytelling, technical walkthrough, and societal impact.	
	6.2	Preparation of final report/documentation (LaTeX/Word). Suggestions for innovation contests and further development.	
		TOTAL	**30**

Guidelines for Assessment of Mini Project Practical/Oral Examination

1. Students must present **problem statement, design thinking process**, and initial prototype.
2. Emphasis on **design explanation**, **ideation process**, and role clarity.
3. Evaluation focuses on understanding of **user needs**, **creativity**, and planning.
4. Panel of **internal examiners**, ideally including faculty with design innovation experience.
5. Project reports to be submitted in university-specified format (preferably LaTeX).
6. Encourage participation in **idea competitions, exhibitions**, or innovation challenges.

Text Books:

1. Don Norman, *"The Design of Everyday Things"*, Basic Books.
2. Tim Brown, *"Change by Design"*, HarperBusiness.
3. Idris Mootee, *"Design Thinking for Strategic Innovation"*, Wiley.
4. Nigel J. Smith, *"Engineering Project Management"*, Wiley-Blackwell.

Reference Books:

1. Jeanne Liedtka, Andrew King, Kevin Bennett, *"Solving Problems with Design Thinking"*, Columbia Business School Publishing.
2. Michael Lewrick, Patrick Link, Larry Leifer, *"The Design Thinking Playbook"*, Wiley.
3. Vijay Kumar, *"101 Design Methods: A Structured Approach for Driving Innovation"*, Wiley.

Useful Links:

1. https://www.designkit.org – IDEO Design Thinking Toolkit
2. https://dschool.stanford.edu/resources – Stanford d.school Resources
3. https://www.interaction-design.org/literature/topics/design-thinking
4. https://nevonprojects.com/project-ideas/software-project-ideas/
5. https://roadmap.sh/projects

Assessment Methodology

Type of Course	Assessment Tool	Marks Distribution
Community Engagement Project	**TW-25**	• Active Participation = 5 marks • Project Report = 10 marks • Progress presentation (Minimum 02) & demonstration = 10 marks