AC: 01/25

Item No: 05

The Bombay Salesian Society's

Don Bosco Institute of Technology, Mumbai

(An Autonomous Institute Affiliated to University of Mumbai)

CURRICULUM STRUCTURE FOR SECOND YEAR ENGINEERING

SEMESTER III

Department of Information Technology

(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 reaffirm our commitment to producing graduates who are not only professionally competent but also dedicated to the greater good of society.

2. Vision and Mission

Vision:

DBIT will be recognized for providing 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 field of sustainable technologies.

Mission:

- To create future engineers who work with honesty and integrity and excel in the use of technology for the benefit of the underprivileged.
- To train engineers to be innovative problem-solvers and entrepreneurs who engage in research and lifelong learning.
- 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 shape the Course Outcomes (COs) and form the foundation for the course structure, delivery, and assessments.

Key design principles include:

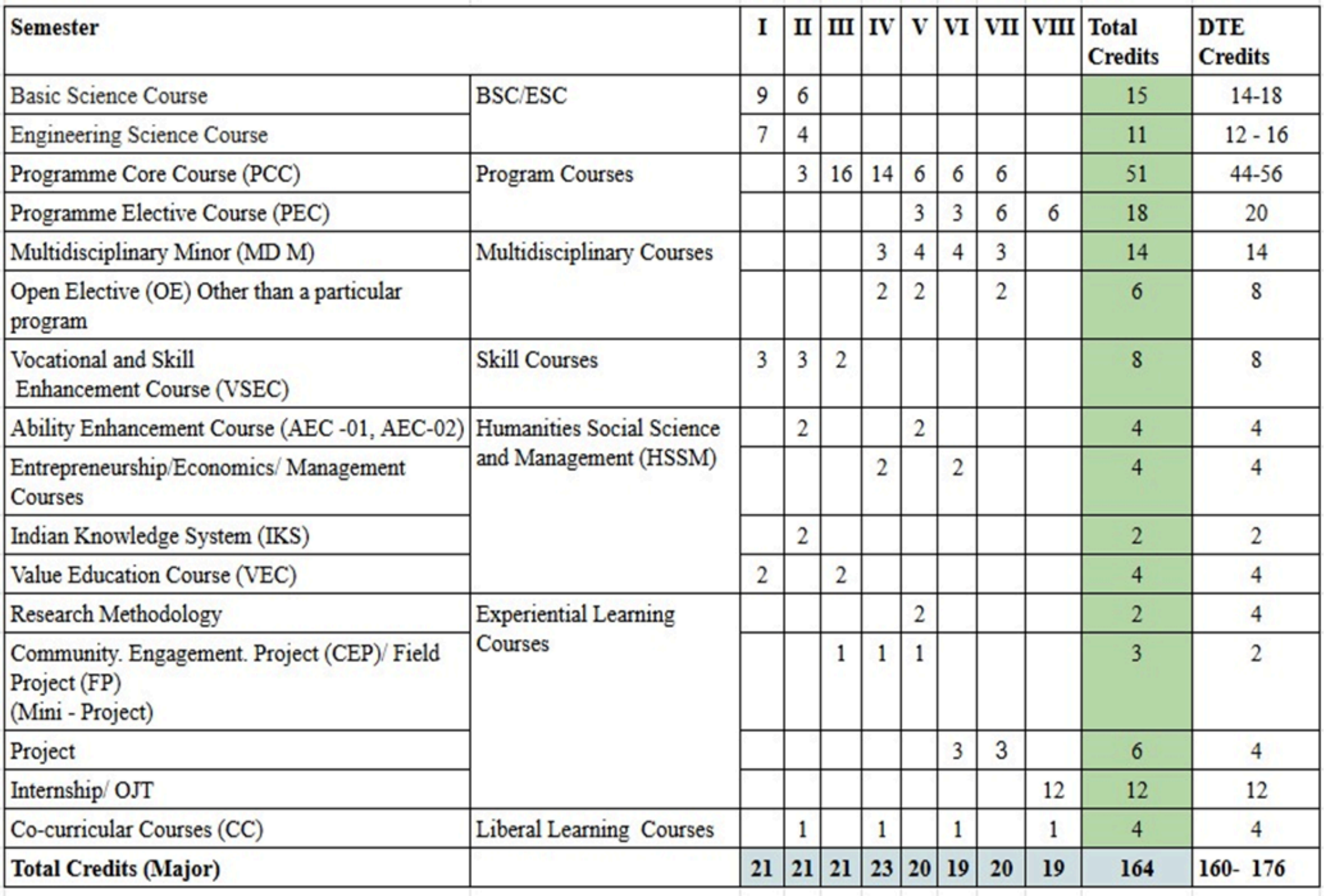
- Emphasis on Outcome-Based Education (OBE) with clear mappings of COs to POs
- Integration of core technical knowledge with interdisciplinary electives
- Inclusion of vocational skills, internships, and community engagement
- Development of entrepreneurship and research aptitude through minor and honors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

4. Credit Guidelines and Allocation

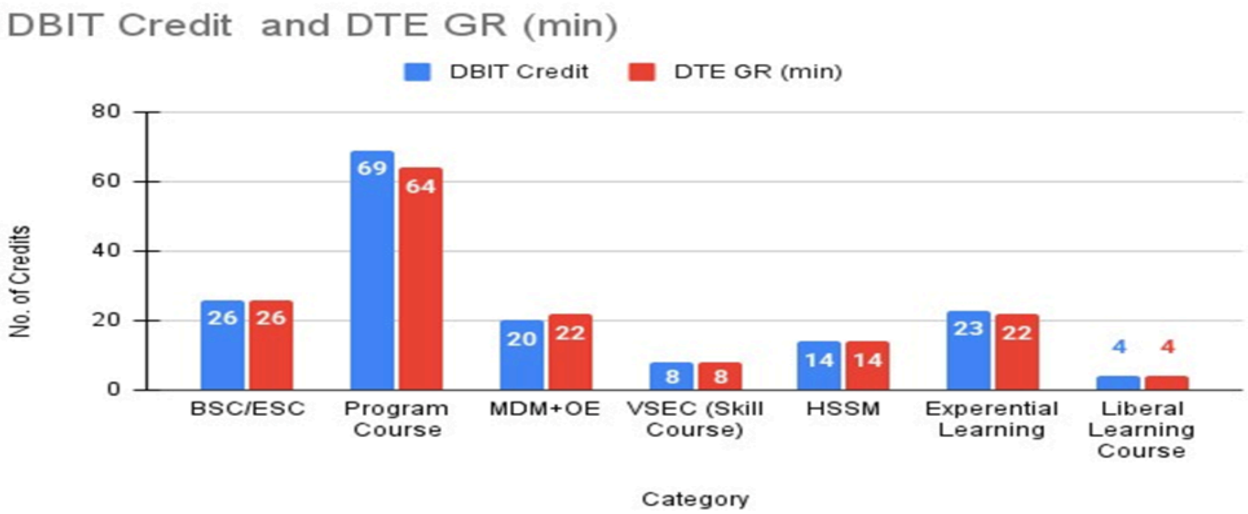
The curriculum is delivered through a structured credit system as follows:

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. - 164 credits
- B.E. Minor/Honors - 182 credits
- B.E. Honors 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 IT Program

Curriculum Scheme and Structure: Semester III									
Course Code	**Course Vertical**	**Course Name**	**Teaching Scheme (Contact Hours)**			**Credits Assigned**			
			L	**P**	**T**	**L**	**P**	**T**	**Total**
25IT3PCC01	PCC	Computational Logic & Structures	3	--	1	3	--	1	4
25IT3PCC02	PCC	Advanced Data Structure & Algorithms	3	2	--	3	1	--	4
25IT3PCC03	PCC	Database Management System	3	2	--	3	1	--	4
25IT3PCC04	PCC	Digital Logic & Computer Organisation and Architecture	3	--	1	3	--	1	4
25IT3VSEC01	VSEC	Full Stack Java Development	--	2*+2	--	--	2	--	2
25IL3VEC02	VEC	Sustainable Development	2	-	-	2	-	-	2
25IT3CEP01	CEP	Community Engagement Project - 1	--	2	--	--	1		1
		Total	**14**	**10**	**2**	**14**	**5**	**2**	**21**

* Two hours of demo/discussion for entire class

Examination and Assessment Structure

Examination Marking Scheme: Semester III									
Course Code	**Course Vertical**	**Course Name**	**Teaching Scheme (Contact Hours)**			**Credits Assigned**			
			CA	**MSE**	**ESE**	**TW**	**OR**	**PR**	**Total**
25IT3PCC01	PCC	Computational Logic & Structures	20	30	50	25	--	--	125
25IT3PCC02	PCC	Advanced Data Structure & Algorithms	20	30	50	25	25	--	150
25IT3PCC03	PCC	Database Management System	20	30	50	25	--	25	150
25IT3PCC04	PCC	Digital Logic & Computer Organization and Architecture	20	30	50	25	--	--	125
25IT3VSEC01	VSEC	Full Stack Java Development	--	--	--	25	25	--	50
25IL3VEC02	VEC	Sustainable Development	50	--	--	--	--	--	50
25IT3CEP01	CEP	Community Engagement Project - 1	--	--	--	25	--	25	50
		Total	**130**	**120**	**200**	**150**	**50**	**50**	**700**

UG Second Year IT Program

Assessment Methodology

Types of Course	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 ● MOOCs Certification (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 Project	**TW-25**	● Active Participation = 05 marks ● Project Report = 10 marks ● Progress Presentations (min 02) & Demonstration = 10 marks
Tutorial	**TW-25**	● Active Participation = 5 marks ● Tutorial Submission = 20 marks Tutorials should cover 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 bythe students during laboratory sessions.

Weightage of COs across all Assessments:

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

Note: Total weightage of all COs should be 100%

Course Code	Course Name	Teaching Scheme (Hrs. / Week)			Credits Assigned				
25IT3PCC01	**Computational Logic & Structures**	L	P	T	L	P	T	Total	
		3	--	1	3	--	1	**4**	
		Examination Scheme							
			CA	**MSE**	**ESE**	**TW**	**OR**	**PR**	**Total**
		Theory	20	**30**	**50**	-	-	-	**100**
		Lab/Tut.	-	-	-	25		-	**25**
		Total	**125**						

Pre-Requisite Courses:	25FE1BSC01 -Fundamentals of Engineering Mathematics -I	
Course Objectives : 1. To understand mathematical reasoning and logic fundamental to computer science. 2. To develop the ability to model and solve computational problems using discrete structures. 3. To study the application of combinatorics, graph theory, set theory and algebra in emerging technologies like AI, DBMS, Blockchain, etc		
Course Outcomes	After successful completion of the course , the students will be able to	
	CO1	Recall the foundational concepts of computational logic. (Remembering)
	CO2	Explain logical equivalences, set operations, recurrence relations and DFA transitions (Understanding)
	CO3	Apply principles of counting, recurrence relations, set theory, and functions to solve mathematical and computational problems. (Applying)
	CO4	Analyze and differentiate among algorithms and systems using graph theory, functions, and automata for effective design. (Analyzing)
	CO5	Evaluate and select the appropriate logical arguments, proof structures and algorithmic correctness using mathematical reasoning techniques. (Evaluating)
	CO6	Design and construct automata, grammar models and discrete structures for solving real-world IT system challenges. (Creating)

Syllabus:

Module No.	Unit No.	Topics	Hours
1	**Logic**		
	1.1	Propositional Logic, Predicate Logic, Logical Equivalences, Quantifiers, Normal Forms, Inference Theory of Predicate Calculus	**7**
	1.2	Mathematical Induction	
	Self-Learning Topics: Fuzzy logic		

2	**Sets, Relations, Functions, and Posets**		
	2.1	Basic concepts of Set Theory: Sets, Venn Diagrams, Set Operations	**8**
	2.2	Relations: Definition, Types of Relations, Representation of Relations, Closures of Relations, Warshall's algorithm, Equivalence relations and Equivalence Classes	
	2.3	Functions: Functions: Injective, Surjective, Bijective; Composition; Inverse	
	2.4	Posets: Partial Order Relations, Poset, Hasse Diagram, Chain and Anti chains	
	Self-Learning Topics: Dilworth's Theorem		
3	**Combinatorics and Recurrences**		
	3.1	Basic Counting Principle: Rule of sum/product, Permutations, Combinations	**8**
	3.2	Pigeonhole Principle, Inclusion–Exclusion	
	3.3	Recurrence Relations, Fibonacci Series, Solving linear recurrence	
	Self-Learning Topics: Generating Functions		
4	**Graph Theory**		
	4.1	Graph Terminologies, Types, Representations, Sub graphs, Operations on Graphs	**9**
	4.2	Walk, Path, Circuit, Connected Graphs, Disconnected Graph, Components,	
	4.3	Homomorphism and Isomorphism of Graphs, Eulerian & Hamiltonian Paths.	
	4.4	Planar Graph, Cut Set, Cut Vertex Applications	
	Self-Learning Topics: 4 - color graph theorem		
5	**Algebraic Structures and Lattices**		
	5.1	Algebraic structures with one binary operation: Algebraic structures: Semi group, Monoid, Groups, Subgroups, Abelian Group, Cyclic group, Isomorphism	**9**
	5.2	Algebraic structures with two binary operations: Ring	
	5.3	Lattice, Types of Lattice, Sub lattice	
	Self-Learning Topics: Cryptography		
6	**Introduction to Automata Theory & Formal Languages**		
	6.1	Regular Expressions, Introduction to Finite State Machines (DFA, NFA)	**4**
	6.2	Grammars: Type 0–3, Applications	
	Self-Learning Topics: Conversion of RE into NFA		
		TOTAL	**45**

Text Books:

1. Bernad Kolman, Robert Busby, Sharon Cutler Ross, Nadeem-ur-Rehman, “Discrete Mathematical Structures”, 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, TataMcGraw 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. J. P. Trembley, R. Manohar “Discrete Mathematical Structures with Applications to Computer Science”, TataMcgraw-Hill

3. Seymour Lipschutz, Marc Lars Lipson, “Discrete Mathematics” Schaum’s Outline, McGrawHill Education

Useful Links:

1. https://www.edx.org/learn/discrete-mathematics
2. https://www.coursera.org/specializations/discrete-mathematics
3. https://nptel.ac.in/courses/106/106/106106094/

Assessment Methodology

Type of Course	Assessment Tool	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 Assignment ● MOOCs Certification (Approved by instructor) ● Open Book Test ● Working model/simulation of a course-based concept.
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
Tutorial	**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

<table>
<tr><th>Course Code</th><th>Course Name</th><th colspan="3">Teaching Scheme (Hrs. / Week)</th><th colspan="5">Credits Assigned</th></tr>
<tr><td rowspan="7">25IT3PCC02</td><td rowspan="7"><b>Advance Data Structure & Algorithms</b></td><td><b>L</b></td><td><b>P</b></td><td><b>T</b></td><td><b>L</b></td><td><b>P</b></td><td><b>T</b></td><td colspan="2"><b>Total</b></td></tr>
<tr><td>3</td><td>2</td><td>-</td><td>3</td><td>1</td><td>-</td><td colspan="2"><b>4</b></td></tr>
<tr><td colspan="8"><b>Examination Scheme</b></td></tr>
<tr><td></td><td><b>CA</b></td><td><b>MSE</b></td><td><b>ESE</b></td><td><b>TW</b></td><td><b>OR</b></td><td><b>PR</b></td><td><b>Total</b></td></tr>
<tr><td><b>Theory</b></td><td>20</td><td><b>30</b></td><td><b>50</b></td><td>-</td><td><b>-</b></td><td><b>-</b></td><td><b>100</b></td></tr>
<tr><td><b>Lab/Tut</b></td><td>-</td><td><b>-</b></td><td><b>-</b></td><td>25</td><td>25</td><td>-</td><td><b>50</b></td></tr>
<tr><td><b>Total</b></td><td colspan="7"><b>150</b></td></tr>
</table>

Pre-Requisite Course:	Data Structure and Analysis, Programming in C

Course Objectives:

1. Develop a strong conceptual understanding of asymptotic notations and apply various methods to determine the time and space complexity of algorithms.
2. Build the ability to critically evaluate algorithmic efficiency, identifying performance bottlenecks and trade-offs in computational resources.
3. Gain proficiency in constructing, manipulating, and evaluating advanced data structures to solve complex computational problems.
4. Analyse problem requirements and choose suitable algorithmic strategies based on complexity considerations and performance goals.
5. Create efficient algorithms, optimising for both time and space, while understanding the implications of design choices.
6. Strengthen analytical and creative problem-solving abilities by applying optimisation and approximation algorithms to real-world scenarios.

<table>
<tr><td rowspan="7"><b>Course Outcomes</b></td><td colspan="2">After successful completion of the course, the students will be able to</td></tr>
<tr><td>CO1</td><td>Identify fundamental data structures, their operations, and the algorithms associated with those operations. (Remembering)</td></tr>
<tr><td>CO2</td><td>Explain the functionality and logic of data structure operations and the algorithms that implement them.(Understanding)</td></tr>
<tr><td>CO3</td><td>Calculate the time and space complexity of data structure operations and algorithms from given code or descriptions.(Apply)</td></tr>
<tr><td>CO4</td><td>Analyze various data structures and algorithms based on complexity, applications, and techniques, including verification of their correctness. (Alanyse)</td></tr>
<tr><td>CO5</td><td>Evaluate the most appropriate data structures and algorithms for specific problems, considering complexity, applicability, and design constraints. (Evaluate)</td></tr>
<tr><td>CO6</td><td>Formulate iterative or recursive algorithms from problem statements or mathematical models. (Design)</td></tr>
</table>

Syllabus :

Module No.	Unit No.	Topics	Hours.
1	**Foundations and Analytical Techniques**		4
	1.1	Algorithm analysis: Amortized complexity & its calculations.	
	1.2	Recurrence relations: Master Theorem, Recurrence Tree, Substitution method.	
	Self Study Topics: Time Complexity calculation for Basic operations of Stack, queue, Linked List, Tree.		
2	**Balanced & AdvanceTrees**		9
	2.1	Red-Black Trees: Rotations, Basic Operation: Insert, Search, Application of Red-Black tree for in-memory storage, Time Complexity.	
	2.2	B-Trees & B+ Trees: Basic Operations: Insert, Search, Delete, Application of B & B+ Tree in Storage, Time Complexity	
	2.3	Binomial Heap and Fibonacci Heap: Binomial Tree, Basic Operations: Create, Insert, Merge, Minimum, Decrease, Application in priority queue, Time Complexity.	
	2.4	Tries: Types of Tries, Basic operation: Insert Search, Delete, Applications in Spell Check, Time Complexity	
	Self Study Topics: Splay Tree, Basic operation and its application		
3	**Graph Structures and Algorithms**		9
	3.1	Graph representations & Traversal: adjacency list, matrix, BFS, DFS traversals.	
	3.2	Shortest path algorithms: Dijkstra, Bellman-Ford - Working, Applications in computer networks, Time & Space Complexity.	
	3.3	Minimum spanning trees: Prim, Kruskal - Working, Applications in network and navigation, Time & Space Complexity	
	3.4	Network Flow Problems: Ford - Fulkerson Algorithm, Edmond Karp Algorithm, Application in Maximum Flow problems,Time & Space Complexity	
	Self Study Topics: Kahn's Algorithm for Topological Sorting in Directed Acyclic Graph		
4	**Algorithmic Technique**		9
	4.1	Introduction to Algorithmic techniques: Brute Force, Divide and Conquer, Greedy Algorithm, Dynamic Programming, Backtracking	

	4.2	Divide and Conquer Technique:- Characteristics of Divide and Conquer technique, Quick Sort , Merge Sort, Application, Time Complexity.	
	4.3	Greedy Technique :- Characteristics of Greedy Technique, Knapsack Problem, Job Sequencing with deadline, Applications of the algorithms, Time Complexity.	
	4.4	DynamicTechnique :- Characteristics of Dynamic technique, Memoization v/s Tabulation, 0/1 Knapsack, Travelling Sales person, Matrix Chain Multiplication, Longest Common Subsequence, Applications, Time Complexity.	
	4.5	Backtracking : Characteristics of Backtracking, N- Queen problem, Application, Time Complexity.	
	Self Study Topics: Optimal binary search tree and its time complexity		
5	**String based Algorithms**		7
	5.1	Introduction to String Matching: Naive String Matching Algorithm and Application of String Matching in Text Editors.	
	5.2	Algorithms: Rabin Karp algorithm, KnuthMorris-Pratt algorithm , Trie-based string matching algorithm, Time and Space Complexity of above algorithms.	
	Self Study Topics: Boyer–Moore string-search algorithm and its complexity		
6	**Advance Algorithms**		7
	6.1	First Order Optimization Algorithms: Characteristics of First Order Optimization Algorithm, Gradient Descent algorithm, Time Complexity, Application.	
	6.2	Population Optimization Method: Characteristics of Population Optimization method, Genetic algorithm(GA), Time Complexity.	
	6.3	Approximation Algorithms: Characteristic of Approximation Algorithm, Vertex-cover problem.	
	6.4	Introduction to NP-Hard and NP-Complete Problems: Hamiltonian path - NP Hard , Boolean Satisfiability (SAT) -- NP Complete.	
	Self Study Topics: Comparison between Approximation, NP Hard and NPC Algorithms.		
	Total		**45**

Text Books:

1. Introduction to Algorithms : Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein
2. Data Structures and Algorithms O.G. Kakde, U.A. Deshpande

Reference Books:

1. Data Structure and Algorithms, Pahuja Sanjay
2. Algorithms: Design and Analysis, Harsh Bhasin, OXFORD.
3. Fundamentals of Computer Algorithms, Horowitz, Sahani, Rajsekaran, Universities Press

Practical list:

Practical No.	Experiment
1	Implement the Master Theorem, to compute the time complexity of a given recurrence relation. Input: Recurrence relation test-cases given by teacher.
2	or given code determine the relation between the amount of execution time and number of elements to execute. Plot graph of the same. Input: Code snippet given by the teacher during the Lab work.
3	Implement the BTree with following operations: Create, Insert, Display Input: As given by the teacher during the Lab work.
4	Implement the Dijkstra algorithm and find the shortest path. Graph operation: Create, Display, Source Node, Display shortest path. Input: As per the graph given by the teacher during the Lab work.
5	Implement the Bellman-Ford and find the shortest path. Graph operation: Create, Display, Source Node, Display shortest path. Input: As per the graph given by the teacher during the Lab work.
6	Implement Prim's algorithm and find the minimum spanning tree. Graph operation: Create, Display, Source Node, Display minimum cost & path. Input: As per the graph given by the teacher during the Lab work.
7	Implement Kruskal's algorithm and find the minimum spanning tree. Graph operation: Create, Display, Source Node, Display minimum cost & path. Input: As per the graph given by the teacher during the Lab work.
8	Implement 0/1 Knapsack algorithm and maximize / minimize the solution Following details will be provided by teacher Capacity of Knapsack: ___ Product weights and associated profits: ___ Teacher led Analysis: The teacher should discuss the problem and algorithmic technique to be used to solve the above problem.
9	Implement the algorithm to solve the above problem. Implement the algorithm to solve the above problem. Implement Fractional Knapsack algorithm and maximize / minimize the solution Following details will be provided by teacher Capacity of Knapsack: ___ Product weights and associated profits: ___ Teacher led Analysis: The teacher should discuss the problem and algorithmic technique to be used to solve the above problem.

10	Implement N - Queen problem using backtracking algorithm
11	Implement Quick sort algorithm . Input: Teacher will give the “n” numbers or alphabets to be sorted and mention the pivot position “k”.
12	Implement String Matching using KnuthMorris-Pratt algorithm Input: Teacher will give string of "m" characters and pattern of "n" characters where n < m
13	Implement Longest Common Subsequence (LCS) to determine the longest substring match between the two strings Input: Teacher give string "S1" of "m" characters and string "S2" of "n" characters, where m, n>=0
14	Implement Trie-based string matching algorithm
15	Implement Genetic algorithm(GA) Input: Teacher to define the target number or string to be developed.
16	Implement Rabin Karp String Matching Algorithm

Assessment Methodology:

Type of Course	Assessment Tool	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 Assignment ● MOOCs Certification (Approved by instructor) ● Open Book Test ● Working model/simulation of a course-based concept.
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
Laboratory	**OR-25**	Oral examination will be based on the entire syllabus.

Course Code	Course Name	Teaching Scheme (Hrs. / Week)			Credits Assigned				
25IT3PCC03	Database Management System	L	P	T	L	P	T	Total	
		3	2	-	3	1	-	4	
		Examination Scheme							
			CA	MSE	ESE	TW	OR	PR	Total
		Theory	20	30	50	-	-	-	100
		Lab/Tut.	-	-	-	25	-	25	50
		Total	150						

Pre-Requisites Courses:	25FE1VESC02 - Problem Solving using C programming
	25FE2PCC04 - Data structure and algorithm

Course Objective:

1. To learn the fundamental database concepts and understand the need of database management systems.
2. To build relational models from ER/EER and understand the process of database normalization.
3. To understand and implement various SQL commands for defining, manipulating and querying database objects.
4. To understand transaction processing, concurrency techniques and recovery mechanisms.
5. To understand and demonstrate indexing techniques for efficient information retrieval.
6. To provide a foundational understanding of distributed databases and fragmentation techniques.

Course Outcome	After successful completion, students will be able to	
	CO1	Recall fundamental concepts, architectural details related to database management systems. (Remembering)
	CO2	Explain the building blocks of ER modeling, draw an ER diagram and an equivalent conversion into relation model for a real life problem. (Understanding)
	CO3	Apply query processing and use SQL commands to define and manipulate database objects, and to retrieve stored information effectively. (Apply)
	CO4	Analyze system requirements and a given database design to perform normalization process to fine tune the database design. (Analyse)
	CO5	Evaluate the given database design and identify approprite indexing and optimization techniques for efficient data management.(Evaluate)
	CO6	Design a scenario and demonstrate transaction management.(Design)

Syllabus :

Module No.	Unit No.	Topics	Hours

1	**DBMS fundamental Concepts and Architecture**		**4**
	1.1	Introduction to DBMS, Comparison with file system, Database characteristics and advantages.	
	1.2	Database architecture, Database implementation types and users.	
	Self-Learning Topic : Case Study based on Amazon Database		
2	**Data Modeling and Normalization**		**9**
	2.1	Data modeling and models, E-R Modeling: Entities, Attributes, Relationships, Cardinality, Participation, ER diagrams, EER, Converting ER model to Relational model.	
	2.2	Extended Entity Relationship (EER), Generalization, Specialization and Association.	
	2.3	Case studies based on ER model (real life examples e.g. Amazon, Spotify, Swiggy, LinkedIn, IRCTC, etc.)	
	2.4	Normalization process, Anomalies, Functional dependencies, Normal forms: 1NF, 2NF and 3NF	
	Self-Learning Topic: BCNF, Case studies on ER modeling and Normalization		
3	**SQL and Relational Algebra**		**12**
	3.1	SQL basics - datatypes, operators, integrity constraints; SQL commands - DDL, DML, DCL, Aggregate functions, Group By, Order By, Subqueries, VIEWs and SQL JOINs.	
	3.2	Advance SQL – Procedures, Triggers and Cursors.	
	3.3	Relational Algebra: Union, Selection, Projection, Cartesian Product, Joins	
	Self-Learning Topic: SET Operations and SQL Function		
4	**Transactions Processing, Concurrency and Database Recovery**		11
	4.1	Database transaction, ACID properties, Transaction states diagram, TCL (Commit, Rollback, Savepoints), Schedules, Serializability.	
	4.2	Database recovery mechanism, Deadlock detection and prevention	
	4.3	Concurrency control techniques - Lock-based, Timestamp-based	
	Self-Learning Topic: Checkpoints and Wait - for - Graph		
5	**Query optimization and Database programming**		**5**
	5.1	Query processing, performance and optimization, Indexing - Basic concepts, types and implementation.	
	5.2	Database programming with JDBC – CRUD Operations	
	Self-learning Topics: Case study – Domain specific database and UI design		
6	**Introduction to Distributed and Schema-less Databases**		**4**
	6.1	Distributed database – Partitioning and data allocation, Comparison with centralized database; Database scaling – vertical and horizontal (Replication,	

		Sharding and Caching).	
	6.2	Schema-less database, introduction to NoSQL.	
	Self-Learning Topic: Installation and query processing - MongoDB, Cassendra		
Total			**45**

Practical List :

Expt No.	List of Experiments
1	Draw an ER diagram and convert it into relational schema for a real life problem
2	Perform SQL DDL commands with various Constraints
3	Perform SQL DML commands with Where clause and various Operators
4	Execute Aggregate functions, Group By and Subqueries.
5	Perform various types of SQL JOINs
6	Create SQL VIEWs and Indexing
7	Create SQL Procedure and Trigger for the given scenario
8	Perform CRUD operations using JDBC API
9	Simulate transaction control using TCL (Commit, Rollback, Savepoint).
10	Demonstrate locking and concurrency using basic scenarios.
Self	Perform DCL, and basic read/write queries on MongoDB or Cassendra (optional)

Text Books:

1. "FundDatabase System Concepts" – Abraham Silberschatz, Henry Korth, and S. Sudarshan (McGraw Hill)
2. Funamentals of Database Systems" – Ramez Elmasri and Shamkant Navathe (Pearson)

References:

1. "SQL For Dummies" – Allen G. Taylor (Wiley).
2. "Principles of Distributed Database Systems" – M. Tamer Özsu and Patrick Valduriez.
3. "NoSQL Distilled" – Pramod J. Sadalage, Martin Fowler (Addison-Wesley)

Online Platforms:

1. MIT OpenCourseWare - https://ocw.mit.edu/courses/6-830-database-systems-fall-2010/pages/readings/.
2. NPTEL - https://onlinecourses.nptel.ac.in/noc22_cs91/preview.
3. CoursEra - https://www.coursera.org/learn/database-management.
4. IBM/Edx - https://www.edx.org/certificates/professional-certificate/ibm-sql-nosql-and-relational-database-fundamentals.

Assessment Methodology:

Types of Course	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 ● MOOCs Certification (Approved by Instructor) ● Open Book Test ● Working model / simulation of a course-based concept.
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
Laboratory	PR-25	Practical examination will be based on the experiments performed by the students during laboratory sessions.

<table>
<tr><th>Course Code</th><th>Course Name</th><th colspan="3">Teaching Scheme
(Hrs. / Week)</th><th colspan="5">Credits Assigned</th></tr>
<tr><td rowspan="7">25IT3PCC04</td><td rowspan="7">Digital Logic & Computer Organization and Architecture</td><td>L</td><td>P</td><td>T</td><td>L</td><td>P</td><td>T</td><td colspan="2">Total</td></tr>
<tr><td>3</td><td>-</td><td>1</td><td>3</td><td>-</td><td>1</td><td colspan="2">4</td></tr>
<tr><td colspan="8">Examination Scheme</td></tr>
<tr><td></td><td>CA</td><td>MSE</td><td>ESE</td><td>TW</td><td>OR</td><td>PR</td><td>Total</td></tr>
<tr><td>Theory</td><td>20</td><td>30</td><td>50</td><td>-</td><td>-</td><td>-</td><td>100</td></tr>
<tr><td>Lab/Tut.</td><td>-</td><td>-</td><td>-</td><td>25</td><td>-</td><td>-</td><td>25</td></tr>
<tr><td>Total</td><td colspan="7">125</td></tr>
</table>

Course Objectives:

1. To understand the principles of number systems, Boolean algebra, logic gates and digital circuits.
2. To design combinational and sequential circuits for given applications.
3. To comprehend the key components of computer architecture and their organization
4. To explore memory organization, I/O systems, and modern trends like RISC, CISC, and parallel processing for efficient computing.

<table>
<tr><td rowspan="7"><b>Course Outcomes</b></td><td colspan="2">After successful completion of the course, the students will be able to</td></tr>
<tr><td>CO1</td><td>Know the various building blocks of digital circuits and their Functionality. (Remembering)</td></tr>
<tr><td>CO2</td><td>Explain the working of combinational and sequential circuits, organization and architecture of a basic computer system. (Understanding)</td></tr>
<tr><td>CO3</td><td>Apply the principles of Digital circuits ,computer architecture and microprocessors to test digital system performance. (Apply)</td></tr>
<tr><td>CO4</td><td>Analyze digital circuits, processor architectures and microprocessor operations to evaluate performance characteristics. (Analysing)</td></tr>
<tr><td>CO5</td><td>Evaluate the characteristics of combinational and sequential circuits , processors, memory systems and compare various memory technologies and I/O interfaces. (Evaluating)</td></tr>
<tr><td>CO6</td><td>Design using basic concepts of combinational and sequential circuits , RISC/CISC processors , memory , I/O devices to solve advanced problems. (Creating)</td></tr>
</table>

Syllabus:

<table>
<tr><th>Module No.</th><th>Unit No.</th><th>Topics</th><th>Hours</th></tr>
<tr><td rowspan="2"><b>1</b></td><td colspan="2"><b>Number System and codes</b></td><td rowspan="2">4</td></tr>
<tr><td>1.1</td><td>Introduction to number system, Binary , octal and hexadecimal number system</td></tr>
</table>

	1.2	Conversion between the different number systems	
	1.3	2’s complement of numbers , binary addition	
	Self Study Topics: Gray to Excess 3 conversion in number system		
2	**Combinational circuits**		11
	2.1	Introduction to logic gates and their Truthtables and excitation tables	
	2.2	Implementation of basic gates using Universal gates	
	2.3	Properties of Boolean Algebra and Reduction of logical expressions using Boolean algebra and K maps	
	2.4	DeMorgan's theorem	
	2.5	Mux, demux, designing using Mux and Demux	
	2.6	Encoder, and decoder and their application	
	2.7	Half adder, fulladder and design of full adder using half adder	
	Self Study Topics: Study of ALU, Study of comparator		
3	**Sequential circuits**		9
	3.1	Flipflops – SR, JK, T, D and their truth tables and excitation tables, conversion of flip flops	
	3.2	Shift registers – SISO, SIPO,PISO,PIPO	
	3.3	Synchronous and asynchronous counter and design of upto 4 digit counters	
	Self Study Topics: Introduction to design of Moore and Mealy Machine		
4	**Overview of Processor Architecture & Organization**		5
	4.1	Introduction of Computer Organization and Architecture.	
	4.2	Von Neumann model and Harvard Architecture	
	4.3	Basic Organization of computer and block level description of the functional units.	
	4.4	Instruction Cycle: Fetch, Decode, Execute	
	Self Study Topics: Instruction pipeline and pipeline hazards		
5	**Memory Organization**		11
	5.1	Introduction to Memory and Memory hierarchy	
	5.2	Memory characteristics and Classifications of memories	
	5.3	Cache memory - Level 1and Level 2 cache and Cache mapping techniques and cache organization	
	5.4	Cache coherency, Cache snooping and cache write policies	
	5.5	I/O Interface and Modes (Programmed, Interrupt-driven, DMA) I/O Mapping: Memory-Mapped I/O and I/O-Mapped I/O	

	Self Study Topics: Study of ISA Bus		
6	**Module 6: Introduction to Microprocessors**		5
	6.1	Introduction to Microprocesssor, microcontroller ,RISC and CISC processors	
	6.2	Addressing modes and Instruction types	
	6.3	Flynns Classification	
	Self Study Topics: Overview of PCI and USB Bus Architecture		
		Total	**45**

Tutorial list:

Tutorial No.	Title
1	Realization of Boolean algebra using gates
2	Verify De'morgans Theorem
3	Design of half adder and full adder
4	Implementation of Multiplexer and De multiplexer
5	Implementation of Encoder and decoder
6	To verify and observe the operation of flipflops
7	To perform conversion between flip flops
8	To implement counter
9	Study of PC mother board
10	Case study on cache memory concepts
11	Case study on CISC and RISC processors
12	Case study on primary and secondary memory
13	Case study on emerging processors
14	Case study on instruction pipelining

Text Books:

1. R. P. Jain, “Modern Digital Electronics”, Tata McGraw Hill.
2. William Stallings, Computer Organization and Architecture: Designing for Performance, Eighth Edition, Pearson
3. M. Morris Mano, “Digital Logic and computer Design”, PHI

Reference Books:

1. A. Anand Kumar, “Fundamentals of Digital Circuits ”, Prentice Hall India
2. B. Govindarajulu,, Computer Architecture and Organization: Design Principles and Applications,Tata McGraw-Hill
3. John P. Hayes, Computer Architecture and Organization, Third Edition., McGraw-Hill

Useful Links:

1. https://nptel.ac.in/courses/117106086
2. https://nptel.ac.in/courses/117105080

Assessment Methodology:

Type of Course	Assessment Tool	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 • MOOCs Certification (Approved by Instructor) • Open Book Test • Working model / simulation of a course-based concept.
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.

Course Code	**Course Name**	**Teaching Scheme (Hrs. / Week)**			**Credits Assigned**			
25IL3VEC01	**Sustainable Development**	**L**	**P**	**T**	**L**	**P**	**T**	**Total**
		2	-	-	2	-	-	**2**
		Examination Scheme						

			CA	MSE	ESE	TW	OR	PR	Total
		Theory	50	**-**	**-**	-	**-**	**-**	**50**
		Lab/Tut.	-	**-**	**-**	-	-	-	**-**
		Total	**50**						

* Two hours of demo/discussion for entire class

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.

Course Outcomes	After successful completion of the course, the students will be able to:	
	CO1	Understand fundamental concepts of AI, IoT, and ICT for sustainable development.(Remembering)
	CO2	Identify key areas of sustainable development in India that can benefit from technology.(ynderstanding)
	CO3	Work collaboratively on an activity to address a societal or environmental issue. (Apply)
	CO4	Analyse real-world case studies and technology-led development models. (Ana;yse)
	CO5	Evaluate the ethical, environmental, and policy implications of digital interventions. (Evaluation)
	CO6	Propose innovative solutions to local and national challenges using AI, IoT, and ICT. (Evaluation)

Syllabus:

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).		
	Fundamentals of AI, IoT, and ICT		

2	2.1	Sustainability enablers using AI: ML, Deep Learning, NLP IoT: Sensors, connectivity, cloud platforms	**5**
	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**

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.

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.

Assessment Methodology:

Types of Course	**Assessment Tools**	**Marks Distribution**
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.

Course Code	**Course Name**	**Teaching Scheme (Hrs. / Week)**	**Credits Assigned**

25IT3VSEC01	**Full Stack Java Development**	**L**	**P**	**T**	**L**	**P**	**T**	**Total**	
		---	2*+2	-	--	1	--	**2**	
		Examination Scheme							
			CA	**MSE**	**ESE**	**TW**	**OR**	**PR**	**Total**
		Theory	--	--	--	--	--	--	--
		Lab/Tut.	--	--	--	25	25	--	**50**
		Total	**50**						

Pre-Requisites Course:	25FE1VSEC02 - Problem Solving using C programming

Course Objective:

1. To introduce concepts, technologies, and workflows involved in full-stack development, covering frontend, backend, databases, DevOps tools, and modern development practices.
2. To develop proficiency in Core and Advanced Java, with an emphasis on object-oriented principles, multithreading, collections, file handling, and JDBC for backend logic and data access.
3. To enable to build RESTful APIs and integrate frontend with backend using Spring Boot and MVC, culminating in the development and deployment of a full-stack application.

	After successful completion, students will be able to	
Course Outcome	CO1	Recall and describe the fundamental concepts of full-stack development and its technology stack. (Remember)
	CO2	Explain core Java object-oriented principles like encapsulation, inheritance, and polymorphism with examples. (Understand)
	CO3	Apply Java collections, file handling, and JDBC to build simple backend programs. (Apply)
	CO4	Analyze RESTful API architecture and Spring MVC structure to identify effective data flow and communication. (Analyze)
	CO5	Evaluate the differences between traditional Spring and Spring Boot to determine the benefits of rapid development. (Evaluate)
	CO6	Design and develop a complete full-stack application integrating frontend, backend, and database with Git & Docker. (Create)

Syllabus :

Module No.	Unit No.	Topics	Hours
1	**Introduction to Full Stack Development**		
	1.1	Overview of full-stack development: Definition of Full Stack Development, Components of Full Stack: Frontend, Backend, Database, DevOps, Full Stack Developer Roles & Skills	**5**
	1.2	Technology Stack Overview: Frontend: HTML, CSS, JavaScript, React.js	
	1.3	Backend: Spring Boot (Java), Express.js (Node), Django (Python), Laravel (PHP) Database: MySQL/MariaDB, PostgresSQL	
	1.4	Tools: Version Control using Git, Docker, Postman, Authentication : JWT, OAuth2, Passport.js, Firebase Auth, CI/CD tools	
	1.5	Real-world applications of full stack	
	Self-Learning Topics: Create a GitHub repo and push sample code to learn Git, Install Docker and run a sample containerized app		
2	**Core Java - Fundamentals**		
	2.1	Java Language Basics: JVM, JRE, JDK. Data Types, Variables, Operators. Control Flow Statements: if-else, switch, loops (for, while, do-while).	**5**
	2.2	Object-Oriented Programming in Java: Classes and Objects in Java. Constructors, this keyword. Encapsulation: Access Modifiers (public, private, protected, default), Getters and Setters. Inheritance: extends keyword, super keyword, Method Overriding. Polymorphism: Method Overloading, Dynamic Method Dispatch. Abstraction: Abstract Classes and Methods, Interfaces.	
	2.3	Packages and Access Control: Organizing classes into packages. Understanding package visibility.	
	2.4	Exception Handling: try, catch, finally blocks. throw and throws keywords. Checked vs. Unchecked Exceptions. Custom Exceptions.	
	Self-Learning Topics: Comparison on python and java in terms of object oriented features		
3	**Java - Advanced Concepts**		**5**
	3.1	Collections Framework: Introduction to Collections: List, Set, Map interfaces. Implementations: ArrayList, LinkedList, HashSet, TreeSet, HashMap, TreeMap. Iterators. Generics.	
	3.2	Input/Output (I/O) Streams: File Handling: File class. Byte Streams vs. Character Streams. FileInputStream, FileOutputStream, FileReader, FileWriter. Buffered Streams. Serialization.	

	3.3	Multithreading: Concepts of Threads, Processes. Creating Threads: Thread class, Runnable interface. Thread Lifecycle. Thread Synchronization (synchronized keyword, locks).	
	3.4	JDBC (Java Database Connectivity) Basics: Connecting to a database. Executing SQL queries (SELECT, INSERT, UPDATE, DELETE). Statement, PreparedStatement, ResultSet. Introduction to connection pooling.	
	Self-Learning Topics: JDBC performance tuning and security		
4	**Spring Boot Fundamentals**		
	4.1	Introduction to Spring Framework: History and philosophy of Spring. Dependency Injection (DI) and Inversion of Control (IoC) Container. Spring Modules (overview).	5
	4.2	Introduction to Spring Boot: Why Spring Boot? Simplifies Spring development. Auto-configuration, Starter dependencies. Creating a simple Spring Boot application. @SpringBootApplication annotation.	
	4.3	Building RESTful APIs with Spring Boot: @RestController, @RequestMapping, @GetMapping, @PostMapping, @PutMapping, @DeleteMapping. Request Parameters, Path Variables, Request Body. JSON serialization/deserialization.	
	4.4	Data Persistence with Spring Data JPA: Introduction to ORM (Object-Relational Mapping). Setting up H2 Database (in-memory). @Entity, @Id, @GeneratedValue. JpaRepository interface for CRUD operations.	
	4.5	Spring Boot Application Configuration: application.properties and application.yml.	
	Self-Learning Topics:		
5	**RESTful API Development**		
	5.1	Introduction to REST: What is an API? REST Architecture Style REST vs. SOAP: Key Differences	5
	5.2	HTTP Basics for REST: HTTP Methods: GET, POST, PUT, DELETE, HTTP Status Codes (200, 201, 400, 404, 500 etc.)	
	5.3	RESTful Design Principles: Resource-Based URIs (e.g., /api/products) Statelessness, Use of Proper HTTP Status Codes	
	5.4	REST API with Spring Boot: Setting up Spring Boot project with REST dependencies, Creating REST Controllers using @RestController, Mapping HTTP Methods: @GetMapping, @PostMapping, @PutMapping, @DeleteMapping	
	5.5	Request and Response Handling: Using @RequestBody, @PathVariable, @RequestParam, Returning JSON data, Exception Handling with @ControllerAdvice	

	Self-Learning Topics: Testing the API, Testing with Postman or curl, Validating responses and debugging		
6	**Spring MVC and Full Stack Integration**		
	6.1	Introduction to Model-View-Controller (MVC) Architecture: Understanding the roles of Model, View, and Controller. Benefits of MVC in web development.	5
	6.2	Spring MVC Fundamentals: DispatcherServlet. @Controller, @RequestMapping. View Resolvers (e.g., Thymeleaf, JSP). Model-View data transfer.	
	6.3	Form Handling and Validation: HTML Forms in Spring MVC. @ModelAttribute, @Valid. Validation annotations.	
	6.4	Frontend Integration (Brief Overview): Introduction to basic HTML, CSS, JavaScript for a simple UI. Connecting frontend (e.g., using Fetch API or Axios) to Spring Boot REST APIs.	
	Self-Learning Topics: Building a Simple Full Stack Application		
Total			**30**

Text Books:

1. Head First Design Patterns by Eric Freeman, Elisabeth Robson, Bert Bates, and Kathy Sierra (O'Reilly Media).
2. Core Java Volume I -- Fundamentals by Cay S. Horstmann (Pearson).
3. Spring in Action by Craig Walls (Manning Publications) - In-depth guide to the Spring framework.
4. Beginning Spring Boot 2.0 by K. Siva Prasad Reddy (Apress) - Practical approach to Spring Boot development.

Reference Books:

1. Spring 5.0 Projects: Build seven web development projects with Spring MVC, Angular 6, JHipster, WebFlux, and Spring Boot 2- Packt Publishing Limited.

Useful Links:

1. Java Documentation : https://docs.oracle.com/en/java/
2. Learn Java : https://dev.java/learn/
3. Spring Documentation : https://docs.spring.io/spring-framework
4. Spring Boot documentation: https://docs.spring.io/spring-boot/index.html
5. https://www.lucidchart.com/
6. https://online.visual-paradigm.com

Practical List :

Expt No.	List of Experiments
1	Write a java program for a given problem demonstrating Classes and Polymorphism.
2	Demonstrate creation of packages with access control and Inheritance for chosen case study.

3	Implement a mini contact book using Java Collections (ArrayList/HashMap) and handle exceptions using try-catch-finally and custom exceptions.
4	Perform File I/O operations: read/write using FileReader and BufferedWriter.
5	Create and manage threads to simulate a multi-threaded banking application.
6	Connect to a MySQL/Postgres/Sqlight database using JDBC to perform basic CRUD operations.
7	Develop a RESTful API using Spring Boot to manage student records.
8	Use Postman to test the RESTful API endpoints developed for student record manageme.
9	Create a Spring Boot project using Spring Initializr, write a basic @RestController and Implement CRUD operations with Spring Data JPA and H2 database.
10	Develop a form submission app with Spring MVC using Thymeleaf templates.
11	Connect frontend HTML/CSS with Spring Boot backend using Fetch API

Assessment Methodology:

Type of Course	**Assessment Tool**	**Marks Distribution**
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.
Laboratory	**OR-25**	Oral examination will be based on the entire syllabus.

Course Code	**Course Name**	**Teaching Scheme (Hrs. / Week)**	**Credits Assigned**

25ITIL3CEP01	Community Engagement Project - 1	L	P	T	L	P	T	Total
		--	--	--	--	1	-	**2**
		Examination Scheme						
			CA	**MSE**	**ESE**	**TW**	**OR**	**PR**	**Total**
		Theory	--	--	--	--	--	--	--
		Lab/Tut.	--	--	--	25	--	25	50
		Total	50						

Pre-Requisites Courses:	25FE1VSEC02 - Problem Solving using C Programming
	25FE2PCC04 - Data Structures and Algorithms

Course Objective:

1. To enable students to identify real-world problems through surveys, interactions, or domain research and define a focused problem statement relevant to industry or society.
2. To inculcate project planning and time management skills using tools like Gantt Charts, PERT, or CPM for effective tracking of project progress.
3. To develop a working full-stack application using Java Spring Boot, with implementation of software engineering principles including modularity, APIs, and database integration.
4. To promote self-learning, collaboration, and documentation including technical project report by encouraging students to maintain logbooks, conduct peer discussions, and actively engage with their project guide.
5. To enhance critical thinking and decision-making by evaluating alternative solutions, selecting the optimal one, and implementing it with justification.

Course Outcome	After successful completion, students will be able to ..	
	CO	Recall and describe fundamental concepts of community engagement, social responsibility, and ethical considerations relevant to technology projects. (Remembering)
	CO2	Explain the socio-economic and environmental context of a chosen community problem and outline possible IT-based interventions. (Understanding)
	CO3	Apply IT tools, frameworks, and project management techniques to address a real-world community issue through a mini-project. (Apply)
	CO4	Analyse community needs, stakeholder expectations, and resource constraints to design an effective and sustainable solution. (Analyse)
	CO5	Evaluate the proposed solution in terms of usability, scalability, societal impact, and alignment with Sustainable Development Goals (SDGs). (Evaluate)
	CO6	Create and present a functional prototype or proof-of-concept that addresses the identified community challenge, documenting the process and learnings.(Design)

Guidelines for Community Engagement Mini Project

1. Group Formation

- Each group shall consist of **3-4 students**.
- Groups should ensure gaining coding, UI/UX design, database handling, communication, and documentation skills.

2. Problem Identification & Need Assessment

- Conduct **community surveys, interviews, or field visits** to identify a real need/problem within society.
- Convert the identified need into a **clear, concise problem statement** in consultation with the help of a guide.
- Map the identified problem to **at least one UN Sustainable Development Goal (SDG)**.

3. Planning & Documentation

- Prepare an **implementation plan** in the form of a **Gantt chart / PERT chart / CPM chart** covering weekly activities for the semester.
- Maintain a **logbook** to record weekly progress, with guide's verification and feedback notes.
- Propose **multiple possible solutions** and select the most feasible one based on stakeholder discussion.

4. Technical Scope & Expectations

- **Minimum Project Scale:** At least **2–3 core modules** (e.g., User Management, Dashboard, Analytics/Reports).
- **Suggested Technology Stack:**

  - Frontend: *React.js* (or equivalent) with API integration.
  - Backend: *Java Spring Boot REST API.*
  - Database: *MySQL* with *JPA/Hibernate*.
  - Security: JWT-based authentication and role-based access control.

- **Integration:** End-to-end connection between frontend, backend, and database.
- **Testing:** API testing (Postman), frontend validation, error handling.
- **Deployment:** Local server or cloud hosting (optional but encouraged).

5. Development & Validation

- Convert the selected solution into a **working prototype** using domain-relevant components.
- Validate the solution with **real users/stakeholders** and collect feedback.
- Ensure the application meets **usability, scalability, security, and accessibility** standards.
- Compile a final report in **standard academic format** including introduction, methodology, testing, results, conclusion, and future scope.

6. Project Deliverables

1. **Project Proposal**
   - Problem statement, objectives, scope, target community, and mapped SDG.
2. **Design Documentation**
   - System architecture diagram (MVC + API flow)
   - Database ER diagram/schema
   - UI wireframes/mockups

3. **Implementation**
    - Source code repositories (GitHub or equivalent) for frontend and backend.
    - RESTful API endpoints with authentication and role-based access control.
4. **Testing**
    - Test cases and results (API & UI validation).
5. **Final Report**
    - Problem background, methodology, solution design, implementation, testing summary, results, societal impact, challenges faced, and future scope.
6. **Presentation & Demo**
    - 8–10 minute live demo with architecture explanation, prototype walk-through, and Q&A.

7. Assessment Process

- **Progress Monitoring**
- At least **two formal reviews** per semester:
- **Review 1:** Problem finalization & proposed solution.
- **Review 2:** Implementation, testing, and validation.

8. Assessment Criteria

- Quality of **survey & need identification**.
- **Clarity** in problem definition and scope.
- **Innovativeness** and novelty in the proposed solution.
- **Feasibility** and practicality of implementation.
- **Dataset collection & usage** (if applicable).
- **Societal impact** and sustainability.
- Quality of **prototype** and alignment with requirements.
- Effective use of **technical skills & standard engineering practices**.
- Individual **contribution & leadership** within the team.
- **Clarity** in written and oral communication during the final presentation.

Assesment Methodology:

TW-25	● Active Participation = 05 marks ● Project Report = 10 marks ● Progress Presentations (min 02) & Demonstration = 10 marks