Marathwada Mitra Mandal's College of Engineering

Karvenagar, Pune- 52

An Autonomous Institute Affiliated to SPPU

Curriculum for

Third Year B. Tech Computer Engineering

A.Y. 2026-27

Permanently Affiliated to SPPU | Accredited with A++ grade by NAAC
Recipient of Best College award by SPPU | Accredited by NBA
Recognized under 2(f) and 12(B) of UGC Act 1956
www.mmcoe.edu.in

CONTENTS

Institute Vision

To be a globally renowned institution through excellence in engineering education for sustainable and holistic development

Institute Mission

M1: Empower students with cutting-edge technologies and global competencies

M2: Foster culture of research and entrepreneurial mindset

M3: Imbibe social and professional values

M4: Provide an inclusive environment for lifelong learning

Department Vision

To excel in computer engineering education for sustainable development.

Department Mission

M1: To develop globally competent professionals through interdisciplinary learning and Center of Excellence.

M2: To empower research, innovation, and entrepreneurial thinking

M3: To promote ethical values and holistic development

M4: To prepare graduates for lifelong learning and dynamic careers

Knowledge and Attitude Profile (WK)

WK1: A systematic, theory-based understanding of the natural sciences applicable to the discipline and awareness of relevant social sciences.

WK2: Conceptually-based mathematics, numerical analysis, data analysis, statistics and formal aspects of computer and information science to support detailed analysis and modelling applicable to the discipline.

WK3: A systematic, theory-based formulation of engineering fundamentals required in the engineering discipline.

WK4: Engineering specialist knowledge that provides theoretical frameworks and bodies of knowledge for the accepted practice areas in the engineering discipline; much is at the forefront of the discipline.

WK5: Knowledge, including efficient resource use, environmental impacts, whole-life cost, re-use of resources, net zero carbon, and similar concepts, that supports engineering design and operations in a practice area.

WK6: Knowledge of engineering practice (technology) in the practice areas in the engineering discipline.

WK7: Knowledge of the role of engineering in society and identified issues in engineering practice in the discipline, such as the professional responsibility of an engineer to public safety and sustainable development.

WK8: Engagement with selected knowledge in the current research literature of the discipline, awareness of the power of critical thinking and creative approaches to evaluate emerging issues.

WK9: Ethics, inclusive behavior and conduct. Knowledge of professional ethics, responsibilities, and norms of engineering practice. Awareness of the need for diversity by reason of ethnicity, gender, age, physical ability etc. with mutual understanding and respect, and of inclusive attitudes.

Program Outcomes (PO)

PO1: Engineering Knowledge: Apply knowledge of mathematics, natural science, computing, engineering fundamentals and an engineering specialization as specified in WK1 to WK4 respectively to develop to the solution of complex engineering problems.

PO2: Problem Analysis: Identify, formulate, review research literature and analyze complex engineering problems reaching substantiated conclusions with consideration for sustainable development. (WK1 to WK4)

PO3: Design/Development of Solutions: Design creative solutions for complex engineering problems and design/develop systems/components/processes to meet identified needs with consideration for the public health and safety, whole-life cost, net zero carbon, culture, society and environment as required. (WK5)

PO4: Conduct Investigations of Complex Problems: Conduct investigations of complex engineering problems using research-based knowledge including design of experiments, modelling, analysis & interpretation of data to provide valid conclusions. (WK8).

PO5: Engineering Tool Usage: Create, select and apply appropriate techniques, resources and modern engineering & IT tools, including prediction and modelling recognizing their limitations to solve complex engineering problems. (WK2 and WK6)

PO6: The Engineer and The World: Analyze and evaluate societal and environmental aspects while solving complex engineering problems for its impact on sustainability with reference to economy, health, safety, legal framework, culture and environment. (WK1, WK5, and WK7).

PO7: Ethics: Apply ethical principles and commit to professional ethics, human values, diversity and inclusion; adhere to national & international laws. (WK9)

PO8: Individual and Collaborative Team work: Function effectively as an individual, and as a member or leader in diverse/multi-disciplinary teams.

PO9: Communication: Communicate effectively and inclusively within the engineering community and society at large, such as being able to comprehend and write effective reports and design documentation, make effective presentations considering cultural, language, and learning differences

PO10: Project Management and Finance: Apply knowledge and understanding of engineering management principles and economic decision-making and apply these to one's own work, as a member and leader in a team, and to manage projects and in multidisciplinary environments.

PO11: Life-Long Learning: Recognize the need for, and have the preparation and ability for i) independent and life-long learning ii) adaptability to new and emerging technologies and iii) critical thinking in the broadest context of technological change. (WK8)

Program Educational Objectives (PEO)

PEO1: Graduates will demonstrate strong foundationalknowledge to address global and multidisciplinary challenges

PEO2: Graduates will engage in innovation, research,and entrepreneurship across multidisciplinary domains

PEO3: Graduates will practice ethical conduct, professionaland social responsibility

PEO4: Graduates will exhibit lifelong learning andcontinuous skill enhancement

Program Specific Outcomes (PSOs)

A Graduate of the Computer Engineering Program will be able to

PSO1: Apply problem solving skills to design effectivesolutions in High Performance Computing, Artificial Intelligence and Cyber Security

PSO2: Develop advanced skill-based solutions usingstandard Software Engineering practices

Abbreviations	
BSC:	Basic Science Course
UG:	Undergraduate Programme
ESC:	Engineering Science Course
PCC:	Program Core Courses
PEC:	Program Elective Courses
MDM:	Multidisciplinary Minor Courses
OEL:	Open Elective
VSE:	Vocational & Skill Enhancement Course
AEC:	Ability Enhancement Course
EEM:	Entrepreneurship/Economics/Management
IKS:	Indian Knowledge System
VEC:	Value Education Course
RMD:	Research Methodology
CEP/FPR:	Comm. Eng. Project (CEP)/Field Project (FP)
PRJ:	Project
INT/OJT:	Internship/On-Job Training
CCC:	Co-Curricular Courses
IT:	Internal Tool
ET:	External Tool
ETE:	End –Term Examination
CIE:	Continuous Internal Evaluation
TW:	Term work
OR:	Oral Examination
PR:	Practical Examination
L:	Lecture
P:	Practical
T:	Tutorial
OL:	Online Teaching
ODL:	Open Distance Learning

Curriculum Structure Sem- V & VI

Third Year B.Tech Computer Engineering- SEMESTER V																	
Course Code	Course Name	Course Type	Teaching Scheme(Hrs/week)				Examination Scheme						Credits				
			L	P	T	ODL	CIE	ETE	TW	PR	OR	Total	L	P	T	ODL	Total
CE24PCC301	System Programming and Operating System	PCC	3	-	-	-	40	60	-	-	-	100	3	-	-	-	3
CE24PCC302	Data Science and Machine Learning	PCC	3	-	-	-	40	60	-	-	-	100	3	-	-	-	3
CE24PCC303	Computer Networks and Security	PCC	3	-	-	-	40	60	-	-	-	100	3	-	-	-	3
CE24PEC304	Elective I: A. Prompt Engineering B. Cloud Computing C. Information Security D. Game Development: From Design to Prototype	PEC	3	-	-	-	40	60	-	-	-	100	3	-	-	-	3
CE24MDM305	* Multidisciplinary Minor	MDM	3	-	-	-	40	60	-	-	-	100	3	-	-	-	3
CE24PCC306	System Programming and Operating System Laboratory	PCC	-	2	-	-	-	-	-	50	-	50	-	1	-	-	1
CE24PCC307	Data Science and Machine Learning Laboratory	PCC	-	2	-	-	-	-	-	50	-	50	-	1	-	-	1
CE24PCC308	Engineering Design and Innovation-I	PCC	-	4	-	-	-	-	25	-	-	25	-	2	-	-	2
CE24MDM309	*Multidisciplinary Minor Laboratory	MDM	-	2	-	-	-	-	25	-	-	25	-	1	-	-	1
CE24OEL310	**Open Elective	OEL	-	-	-	2	-	-	50	-	-	50	-	-	-	2	2
CE24AEC311	Reasoning and Aptitude Development- I	AEC	-	-	1	-	-	-	25			25	-	-	1	-	1
Total			15	10	1	2	200	300	125	100	-	725	15	5	1	2	23

* Refer separate MDM Booklet
** Refer separate OE Booklet

L- Lecture P- Practical T- Tutorial CIE-Continuous Internal Evaluation

ETE- End Term Examination TW- Term work PR- Practical OR- Oral

L : 1 Hr.= 1 credit P: 2 Hr. = 1 Credit T: 1 Hr. = 1 Credit

Third Year B.Tech Computer Engineering- SEMESTER VI																	
Course Code	Course Name	Course Type	Teaching Scheme (Hrs/week)				Examination Scheme						Credits				
			L	P	T	ODL	CIE	ETE	TW	PR	OR	Total	L	P	T	ODL	Total
CE24PCC351	Artificial Intelligence	PCC	3	-	-	-	40	60	-	-	-	100	3	-	-	-	3
CE24PCC352	Design and Analysis of Algorithms	PCC	3	-	-	-	40	60	-	-	-	100	3	-	-	-	3
CE24PCC353	Theory of Computation	PCC	3	-	-	-	40	60	-	-	-	100	3	-	-	-	3
CE24PEC354	Elective II A. Developments and Operations B. Distributed Systems C. High Performance Computing D. Digital Forensic	PEC	3	-	-	-	40	60	-	-	-	100	3	-	-	-	3
CE24MDM355	* Multidisciplinary Minor	MDM	3	-	-	-	40	60	-	-	-	100	3	-	-	-	3
CE24PCC356	Artificial Intelligence Laboratory	PCC	-	2	-	-	-	-	-	50	-	50	-	1	-	-	1
CE24PCC357	Design and Analysis of Algorithms Laboratory	PCC	-	2	-	-	-	-	-	50	-	50	-	1	-	-	1
CE24PCC358	Engineering Design and Innovation-II	PCC	-	2	-	-	-	-	25	-	-	25	-	1	-	-	1
CE24OEL359	**Open Elective	OEL	-	-	-	2	-	-	50	-	-	50	-	-	-	2	2
CE24VSE360	Server-Side Programming	VSE	-	4	-	-	-	-	50	-	-	50	-	2	-	-	2
CE24AEC361	Reasoning and Aptitude Development - II	AEC	-	-	-	1	-	-	25	-	-	25	-	-	-	1	1
Total			15	10	-	3	200	300	150	100	-	750	15	5	-	3	23

* Refer separate MDM Booklet
** Refer separate OE Booklet

L- Lecture P- Practical T- Tutorial CIE-Continuous Internal Evaluation

ETE- End Term Examination TW- Term work PR- Practical OR- Oral

L : 1 Hr.= 1 credit P: 2 Hr. = 1 Credit T: 1 Hr. = 1 Credit

SEMESTER V

Third Year B.Tech Computer Engineering

Semester-V

Course Code: CE24PCC301 **Course Name: System Programming and Operating System**

Teaching Scheme (Hours/Week)					Examination Scheme						Credits			
L	P	T	OL	ODL	CIE	ETE	TW	OR	PR	TOTAL	L	P	T	TOTAL
3	-	-	-	-	40	60	-	-	-	100	3	-	-	3

Prerequisite: CE24PCC203: Digital Electronics & Computer Organization

Course Objectives:

The course aims to:

- Analyze the role of program translation tools.
- Develop the ability to solve process scheduling and synchronization problems.
- Apply memory and file management techniques for efficient resource utilization.

Course Outcomes:

After learning the course, the students will be able to:

CO1: Identify design process of two-pass assembler.
CO2: Analyze roles of system softwares macros, linkers and loaders in program translation.
CO3: Compare different Process Scheduling algorithms.
CO4: Solve Classical Synchronization Problems using semaphores and mutex.
CO5: Evaluate memory management techniques.

Unit	Contents	Duration (Hrs.)
1	**Systems Programming:** Introduction**,** Types of software: system software and application software, Evolution of components of Systems. **Assembly Language Programming:** Assembly Language statements, A simple Assembly scheme, Pass Structure of Assembler. Design of two pass Assembler: Processing of declaration statements, Assembler Directives and Imperative statements, Advanced Assembler Directives, Intermediate code forms, Pass I and Pass II of Two Pass Assembler. **Introduction to Compilers:** Phases of Compiler with one example, Comparison of Compiler and Interpreter. Case Study: Role of System Software in Modern Operating Systems	8
2	**Macro Processor:** Introduction, Features of a Macro facility: Macro instruction arguments, Conditional Macro expansion, Macro calls within Macros, Macro instructions, Defining Macro, Design of two pass Macro processor. **Linker and Loader:** Introduction, Loader schemes: Compile and Go, General Loader Scheme, Absolute Loaders, Subroutine Linkages, Relocating Loaders, Direct linking Loaders, Overlay structure, Self-relocating programs, Linker**-** Introduction**,** Types of linking. Case Study: Performance Impact of Macro Expansion in System Programs	9
3	**Introduction to Operating System** Introduction to Operating System: Evolution, Services, Functions, Types, Virtualizing The CPU, Virtualizing Memory, Concurrency, Persistence. **Process Management:** Process, States: 5 and 7 state model, Process Control Block(PCB), Threads, Thread lifecycle, Multithreading Model, Process Scheduling Types: Preemptive, Non-preemptive, Schedulers, Scheduling Algorithms: FCFS, SJF, RR, Priority, Completely Fair Scheduler, Multilevel Feedback Queue. Case Study: Scheduling and Synchronization in xv6	7

4	**Synchronization and Concurrency Control** Synchronization Vs. Concurrency Control, Critical Section Problem, Mechanisms, Classical Synchronization Problems **Deadlocks**: Introduction to Deadlock, Conditions for Deadlock, Resource Allocation Graph (RAG), Wait-for Graph (WFG), Detection, Prevention, Avoidance - Banker's Algorithm, Recovery. Case Study: CPU Virtualization	8
5	**Memory Management and File Management** **Memory Management :** Concepts, requirements, Memory Partitioning, Fragmentation, TLB, Paging, Segmentation,Placement Strategies, **Virtual Memory:** Concept, Swapping, Page Replacement Policies: FIFO, LRU, Optimal. Swapping issues: Thrashing **File Management:** Files, File Systems, File structure. File Organization and Access Methods, Allocation Methods, Directories, File Sharing, Record Blocking, Introduction to FAT16, FAT32 and NTFS Case Study: I/O Management and The xv6 filesystem	8
	Total Hours	**40**

Text Books

1. John J. Donovan, Systems Programming, McGraw-Hill, 1st Edition , ISBN-10: 0074604821 • ISBN-13: 978-0074604823
2. Dhananjay M. Dhamdhere, Systems Programming and Operating Systems (Second Revised Edition), Tata McGraw-Hill Education, 2nd Edition, 1999, ISBN-10: 0074635794 • ISBN-13: 978-0074635797
3. Operating Systems: Three Easy Pieces, Andrea Arpaci-Dusseau and Remzi Arpaci-Dusseau, Arpaci-Dusseau Books, 1st Edition, 2012, ISBN-10: 0985673529 • ISBN-13: 978-0985673529
4. William Stallings, Operating Systems: Internals and Design Principles, Pearson, 9th Edition, 2018, ISBN-10: 0134670957 • ISBN-13: 9780134670959
5. Abraham Silberschatz, Peter Baer Galvin, and Greg Gagne, Operating System Concepts, John Wiley & Sons, Inc., 10th Edition, 2018, ISBN-10: 1119439256 • ISBN-13: 978-1119439257

Reference Books

1. Tom Adelstein and Bill Lubanovic, Linux System Administration, O'Reilly Media, 1st Edition, 2007, ISBN-10: 0596009526 • ISBN-13: 978-0596009526
2. Harvey M. Deitel, Paul J. Deitel, and David R. Choffnes, Operating Systems, Prentice Hall, 3rd Edition, 2003, ISBN-10: 0131828274 • ISBN-13: 978-0131828278
3. Thomas W. Doeppner, Operating Systems in Depth: Design and Programming, Wiley, 1st Edition, 2010, ISBN-10: 0471687235 • ISBN-13: 978-0471687238
4. Andrew S. Tanenbaum and Herbert Bos, Modern Operating Systems, Pearson, 5th Edition, 2022, ISBN-13: 9780137618880
5. Milan Milenkovic, Operating Systems: Concepts and Design, 2nd Edition, McGraw-Hill Education, 2001, ISBN-10: 0074632728 • ISBN-13: 978-0074632727

Online References

- NPTEL Course: Prof. Sorav Bansal, IIT Delhi, "Operating Systems" https://nptel.ac.in/courses/106102132
- NPTEL Course: Prof. Santanu Chattopadhyay, IIT Kharagpur, "Operating System Fundamentals" https://nptel.ac.in/courses/106105214
- NPTEL Course: Prof. Chester Rebeiro, IIT Madras, "Introduction to Operating Systems," https://nptel.ac.in/courses/106106144

Third Year B.Tech Computer Engineering

Semester- V

Course Code: CE24PCC302 **Course Name: Data Science and Machine Learning**

Teaching Scheme (Hours/Week)					Examination Scheme						Credits			
L	P	T	OL	ODL	CIE	ETE	TW	OR	PR	TOTAL	L	P	T	TOTAL
3	-	-	-	-	40	60	-	-	-	100	3	-	-	3

Prerequisite: CE24PCC251 Database Management System

Course Objectives:

The Course aims to

- Apply data science life cycle and data collection methods on real world data.
- Analyze preprocessing methods and feature selection techniques.
- Apply machine learning approaches for solving business problems.
- Implement regression and classification models.
- Determine clustering methods and reinforcement strategies.

Course Outcomes:

After learning the course, the students will be able to:

CO1: Apply data science concepts to real-world scenarios.
CO2: Analyze datasets using feature engineering and EDA.
CO3: Apply machine learning approaches for business problems.
CO4: Develop predictive models using supervised learning techniques.
CO5: Evaluate clustering and reinforcement learning-based solutions.

Unit	Contents	Duration (Hrs.)
1	**Introduction to Data Science:** Basics concepts, Applications, Data explosion, 5 V's of Big Data, Business intelligence versus Data Science, Data Science Life Cycle, Key Roles for a Successful Analytics Project, Data Types, Data Collection methods.	7
2	**Feature Engineering:** Concept of Feature, Preprocessing of data: Removing Duplicates, Transformation of Data using function or mapping, replacing values, Normalization and Scaling, Standardization, Managing missing values, Feature Selection Techniques, Dimensionality Reduction, Principal Component Analysis (PCA), Feature Extraction, Exploratory Data Analysis: Univariate analysis, Bivariate/Multivariate analysis Case study: Telecom churn dataset preprocessing	9
3	**Introduction To Machine Learning:** Introduction, Comparison of Machine learning(ML) with traditional programming, ML vs Artificial IntelligenceI vs Data Science, Types of ML, Models of ML. Case Study: Uber database to identify ML model used.	6
4	**Supervised Learning :Regression and Classification** Regression: Linear regression, Lasso regression, Ridge regression, Gradient descent algorithm, Bias, Variance, Underfitting, Overfitting, Evaluation Metrics Classification: K-nearest neighbour, Decision tree, Random forest, Naive Bayes, Ensemble Learning, Evaluation Metrics. Case study: Stock market price prediction and —--	10

5	**Unsupervised Learning and reinforcement learning:** Basics of Clustering, Methods: K-means, DBSCAN, Spectral Clustering, Evaluation methods, Hierarchical Clustering, Expectation maximization clustering, Agglomerative Clustering. Reinforcement Learning: Reinforcement learning through feedback network, function approximation Case study: Customer Behavior Analysis for an E-Commerce Platform	8
	Total Hours	**40**

Text Books

1. Bishop, Christopher M., and Nasser M. Nasrabadi, "Pattern recognition and machine learning", Vol. 4.No. 4. New York: Springer, 2006.
2. Ethem Alpaydin, " Introduction to Machine Learning", PHI 2nd Edition-2013
3. Géron, A., "Hands-On Machine Learning with Scikit-Learn...", O'Reilly, 3rd Ed., ISBN: 9781098125974.
4. Deisenroth, M. et al., "Mathematics for Machine Learning", Cambridge Univ. Press, ISBN: 9781108455145.
5. Shalev-Shwartz, S. & Ben-David, S., "Understanding Machine Learning", Cambridge Univ. Press, ISBN: 9781107057135.

Reference Books

1. EMC Education Services, "Data Science and Big Data Analytics: Discovering, Analyzing, Visualizing and Presenting Data", Wiley, 2015, ISBN: 9781118876138.
2. Jiawei Han, Micheline Kamber, and Jian Pei, "Data Mining: Concepts and Techniques", Elsevier, 3rd Ed., ISBN: 9780123814791.
3. Wes McKinney, "Python for Data Analysis", O'Reilly, ISBN: 9781449319793.
4. Trent hauk, "Scikit-learn", Cookbook , Packt Publishing, ISBN: 9781787286382
5. Tom Mitchell, " Machine learning", McGraw-Hill series in Computer Science, 1997

Online References

- NPTEL Course: Introduction to Machine Learning, IIT Madras, Prof. Balaraman Ravindran- https://nptel.ac.in/courses/106106139
- NPTEL Course: Machine Learning for Engineering & Science, IIT Madras, Prof. Balaji Srinivasanand Prof. Ganapathy Krishnamurthi- https://nptel.ac.in/courses/106106198.
- NPTEL Course: Reinforcement Learning, IIT Madras, Prof. Balaraman Ravindran- https://nptel.ac.in/courses/106106143

Third Year B.Tech Computer Engineering

Semester- V

Course Code: CE24PCC303 **Course Name: Computer Networks and Security**

Teaching Scheme (Hours/Week)					Examination Scheme						Credits			
L	P	T	OL	ODL	CIE	ETE	TW	OR	PR	TOTAL	L	P	T	TOTAL
3	-	-	-	-	40	60	-	-	-	100	3	-	-	3

Prerequisites: –

Course Objectives:

The Course aims to

- To understand the basic concepts of computer network
- To implement techniques used for framing, error control and flow control
- To apply network layer functions including logical addressing, routing and packet forwarding in a network
- To develop client server communication using transport layer protocols
- To apply the fundamental concepts of network security

Course Outcomes:

After learning the course, the students will be able to:

CO1: Apply fundamental networking concepts to basic networking scenarios

CO2: Apply the functions of data link layer such as flow control and error control for network communication

CO3: Demonstrate Logical addressing, subnetting & Routing Mechanisms

CO4: Implement client-server applications using sockets

CO5: Apply network security concepts to identify security requirements

Unit	Contents	Duration (Hrs.)
1	Introduction Basic concepts, OSI Reference Model, TCP/IP Model, Network Topologies, Transmission Media, Network devices. Line Coding Schemes: Manchester and Differential Manchester Encodings, Spread Spectrum Techniques Case Study: MMCOE Campus Network Study	8
2	**Data Link Layer** Design issues, Framing, Flow Control, ARQ strategies, Error detection and correction, PPP and HDLC. Multiple Access Protocols: Pure and Slotted ALOHA, CSMA, CSMA/CD, CSMA/CA. IEEE network standards Case Study: Creating a Wireless LAN in a small office or lab.	8
3	**Network Layer** Design issues, IP protocol, IP Addressing, Network Address Translation(NAT), CIDR, Subnetting, ARP, RARP, ICMP, Static and dynamic Routing, Distance Vector Routing, Link State Routing, multicast routing, OSPF,BGP,MPLS,Quality of Service (QoS),Software Defined Network(SDN) Case Study:Routing Protocol Demo using Cisco Packet Tracer	8
4	**Transport and Application Layer** Services, Addressing, Connection establishment, Connection release, Error control and Flow control, TCP, UDP. Web and HTTP, DNS, SMTP, FTP, TELNET, DHCP. Case Study:Demonstration of Transport layer protocols on Simulator	8

5	**Security** Basic concepts, Threats and Vulnerabilities, Types of Attacks, ITU-T X.800 Security Architecture for OSI, Security Policy and mechanisms, Operational Model of Network Security, Symmetric and Asymmetric Key Cryptography. Security in Network and Transport layer, Overview of IDS and Firewalls. Case Study:Study and analyse the performance of Application Layer packets using packet analyser tools	8
	Total Hours	**40**

Text Books

1. A. S. Tanenbaum, N. Feamster, and D. J. Wetherall, *Computer Networks*, 6th ed. Noida, India: Pearson Education, 2021. ISBN: 1-292-37406-3
   Pearson Education. ISBN: 9780132126953
2. Forouzan, Behrouz A. — *Data Communications and Networking with TCP/IP Protocol Suite*, 6th Edition. McGraw-Hill. ISBN: 978 9355320940
3. Kurose, James F., and Keith W. Ross. *Computer Networking: A Top-Down Approach*, 9th Edition. Pearson Education. ISBN: 9780135415603
4. William Stallings, "Cryptography and Network Security: Principles and Practice", 7th Edition

Reference Books

1. Peterson, Larry L., and Bruce S. Davie. *Computer Networks: A Systems Approach*, 5th Edition. Morgan Kaufmann. ISBN: 9780123850591
2. Leon-Garcia, Alberto, and Indra Widjaja. — *Communication Networks: Fundamental Concepts and Key Architectures*, 2nd Edition. McGraw-Hill. ISBN: 978 0070595019
3. Stallings, William. *Data and Computer Communications*, 10th Edition.Pearson Education. ISBN: 9780137561704

Online References

- Computer Networks And Internet Protocol-Prof. Soumya Kanti Ghosh Prof. Sandip Chakraborty - https://nptel.ac.in/courses/106105183
- The Bits and Bytes of Computer Networking- Coursera https://www.coursera.org/learn/computer-networking
- Networking Essential -Cisco Networking Academy https://www.netacad.com/courses/networking/networking-essentials

Third Year B.Tech Computer Engineering

Course Code : CE24PEC304A **Course Name: Prompt Engineering**

Teaching Scheme (Hours/Week)					Examination Scheme						Credits			
L	P	T	OL	ODL	CIE	ETE	TW	OR	PR	TOTAL	L	P	T	TOTAL
3	-	-	-	-	40	60	-	-	-	100	3	-	-	3

Pre-requisites: Communications, Problem Solving Abilities, and Analytical Thinking

Course Objectives:

This Course aims to:

- Introduce foundations of Large Language Models (LLMs) and the role of prompt engineering in modern Generative AI applications.
- Develop students' ability to design, structure, and refine prompts for diverse technical and non-technical tasks.
- Enable students to build basic LLM-based applications using APIs, structured outputs, and simple tool use.
- Expose students to Retrieval-Augmented Generation (RAG), domain-specific assistants, and basic AI agent concepts.
- Sensitize students to safety, security, bias, and ethical issues in designing and deploying LLM systems.

Course Outcomes :

After completing the course, the students will be able to:

CO1: Describe the fundamentals, capabilities, and limitations of LLMs and justify the need for prompt engineering.

CO2: Design and optimize prompts using core patterns such as zero-shot, few-shot, role prompting, and stepwise reasoning.

CO3: Implement simple LLM-powered applications using APIs with structured outputs and basic tool use integration.

CO4: Construct small RAG-style and domain-specific assistant workflows relevant to computer engineering problems.

CO5: Apply responsible AI principles to analyze and mitigate safety, security, and ethical risks in LLM-based solutions.

Unit	Contents	Duration (Hrs)
1	**Foundations of LLMs and Prompt Engineering** Evolution of NLP to Generative AI: Rule-based systems, statistical NLP, deep learning, transformers, and LLMs.. Key concepts: tokens, embeddings, context window, parameters, latency, temperature, top-p, and cost trade-offs. Capabilities and limitations of LLMs: hallucinations, sensitivity to wording, context length constraints, non-determinism. Concept of a "prompt": instruction prompts, conversational prompts, system prompts, and few-shot prompts. Prompt engineering vs traditional programming; "prompt guessing" vs systematic engineering. Overview of popular LLM platforms (Claude, GPT, Gemini, open-source models) – basic similarities and differences.	8

2	**Core Prompting Patterns and Best Practices** Instruction design: clear goals, constraints, and context; specifying role (e.g., "You are a senior software architect"). Zero-shot, one-shot, and few-shot prompting: concepts, when to use each, and how to choose good examples. Use of delimiters and structure: sections, bullet points, stepwise instructions, and explicit input/output segments. Controlling outputs: requesting tabular output, bullet lists, code blocks, or JSON-style responses. Prompt decomposition: breaking complex problems into smaller sub-tasks and chaining prompts. Introduction to chain-of-thought and self-reflective prompts for analytical and reasoning tasks.	8
3	**LLM APIs, Structured Outputs, and Tool Use** Basics of LLM APIs: authentication, request/response structure, rate limits, error handling. Prompt templates and parameterization: separating prompt text from variables in code. Structured outputs: designing JSON schemas, key–value pairs, enforcing formats, using pre-filled response skeletons. Deterministic behavior: controlling randomness via temperature and seeds, validating returned outputs. Introduction to tool/function calling: concept of the model deciding when to call tools such as calculators, search, or internal APIs.	8
4	**RAG, Domain-Specific Assistants, and Agents** Motivation for Retrieval-Augmented Generation (RAG): reducing hallucinations and grounding answers in trusted data. Basics of retrieval: embeddings, vector stores, chunking, and metadata filtering (conceptual level). Generic RAG pipeline: document ingestion → retrieval → context construction → prompt to LLM → answer generation. Designing prompts that incorporate retrieved context, encourage citation of sources, and handle "no-answer" situations. Concept of AI agents: planning vs acting, multi-step workflows, and tool orchestration (high-level view). Domain-specific assistants for computer engineering: documentation helper, lab manual bot, network troubleshooting assistant, etc.	8
5	**Responsible & Secure Prompt Engineering, Evaluation** Safety, security, and ethics: prompt injection, jailbreak attempts, data leakage, handling personal/sensitive data. Bias and fairness: recognizing and mitigating bias, maintaining neutrality, documenting limitations. Evaluation of prompts and systems: qualitative review, accuracy checks, small "golden" evaluation sets. Using LLMs as critics (LLM-as-judge) and designing refine/evaluate loops for prompt improvement. Deployment considerations: logging, monitoring, cost awareness, and basic observability concepts.	8
	Total Hours	**40 Hrs**

Text Books

- Phoenix, J., & Taylor, M. – Prompt Engineering for Generative AI, O'Reilly Media, 2024.

- Koul, N. – Prompt Engineering for Large Language Models, 1st Edition, 2025.
- Richardson, E. C. – Prompt Engineering, BPB Publications, 2025.

Reference Books

1. Nayyar, A., Vairamani, A. D., & Kaswan, K. – Mastering Prompt Engineering: Deep Insights for Optimizing Large Language Models (LLMs), Elsevier, 2025.
2. Vemula, A. – Generative AI System Design: A Practical Guide, Amazon KDP, 2024.
3. Berryman, J., & Ziegler, A. – Prompt Engineering for LLMs: The Art and Science of Building Large Language Model Applications, O'Reilly Media, 2024 (in development/early access).

Online References

- Prompt Engineering Guide: https://www.promptingguide.ai/
- Prompt Engineering Example: https://www.coursera.org/articles/prompt-engineering-examples

Third Year B.Tech Computer Engineering

Semester-V

Course Code: CE24PEC304B **Course Name: Cloud Computing**

Teaching Scheme (Hours/Week)					Examination Scheme						Credits			
L	P	T	OL	ODL	CIE	ETE	TW	OR	PR	TOTAL	L	P	T	TOTAL
3	-	-	-	-	40	60	-	-	-	100	3	-	-	3

Prerequisite: CE24PCC302:Computer Network

Course Objectives:

The Course aims to

- Use cloud computing concepts and cloud-based data storage in real time applications
- Implement virtualization techniques in cloud computing environments
- Understand advanced applications and techniques in cloud computing
- Examine risks and vulnerabilities in cloud computing
- Apply knowledge of cloud platforms and services in practical scenarios.

Course Outcomes :

After completing the course, the students will be able to:

CO1: Apply cloud computing concepts and cloud data storage methods in practical scenarios
CO2: Analyze virtualization technology and install virtualization software
CO3: Use advance techniques in Cloud Computing
CO4: Analyze cloud risks and basic cloud security measures
CO5: Develop and deploy applications on Cloud

Unit	Contents	Duration (Hrs.)
1	**Introduction to Cloud Computing and Data Storage** Basic concepts, Pros and Cons, Migrating into a cloud, Deployment models, Service Models, Architecture, Cloud Modeling and Design. Data Storage: Introduction to Enterprise data storage and management,File systems,Cloud data stores. Case study: Cloud Computing Model of AWS.	8
2	**Virtualization in Cloud Computing** Introduction, Characteristics, Taxonomy of virtualization techniques, Pros and cons,Technology examples,Virtualization Architecture and software, Virtual clustering, Virtualization Application Case Study: Online Book Marketing Service.	8
3	**Cloud Applications and Advanced Techniques in Cloud Computing** Scientific applications, Business and consumer applications**:**CRM and ERP. Social networking, Advanced Topics, Energy efficiency in clouds,Market-based management of clouds, Federated clouds/InterCloud, Third-party cloud services,Distributed Cloud Computing, Edge Computing, Containers, Docker, and Kubernetes. Case study: Xen: Para virtualization.	8
4	**Cloud computing and security:** Risk Management, Enterprise-Wide Risk Management, Types of Risks. Data Security in Cloud: Security Issues, Challenges, Advantages, Disadvantages, Cloud Digital persona and Data security, Content Level Security. Cloud Security Services: Confidentiality, Integrity and Availability, Security Authorization Challenges in the Cloud, Secure Cloud Software Requirements Case Study:Cloud Security Tool: Acunetix	8

5	**Cloud Platforms and Services** Amazon Web Services (AWS):Compute services, Storage services, Communication services, Amazon Storage system, Amazon database services: DynamoDb, Google AppEngine:Architecture and core concepts, Application life cycle, Cost model, Microsoft Azure:Azure core concepts, Database services and fundamental security concepts. Case study: Dev Ops - DocuSign, Forter	8
	Total Hours	**40**

Text Books

1. A. Srinivasan, J. Suresh, "Cloud Computing: A Practical Approach for Learning and Implementation", Pearson, ISBN: 978-81-317-7651-3
2. Rajkumar Buyya, Christian Vecchiola, and Thamrai Selvi, Mastering Cloud Computing McGraw-Hill Education, 2nd Edition (McGraw-Hill) — ISBN 978-9355329509.

Reference Books

1. Toby Velte, Anthony Velte, Cloud Computing: A Practical Approach, McGraw-Hill Education, 2017.
2. Dr. Kris Jamsa, "Cloud Computing: SaaS, PaaS, IaaS, Virtualization and more", Wiley Publications, ISBN: 978-0-470-97389-9
3. George Reese, Cloud Application Architectures: Building Applications and Infrastructure in the Cloud, O'Reilly Publication, 1st Edition.
4. James Bond ,"The Enterprise Cloud", O'Reilly Media, Inc. ISBN: 9781491907627.

Online References

- NPTEL Course: Prof. Soumya Kanti Ghosh, IIT Kharagpur, "Cloud Computing" https://onlinecourses.nptel.ac.in/noc21_cs14/preview
- NPTEL Course: Prof. Rajiv Misra, IIT Patna , "Cloud Computing and Distributed System" https://onlinecourses.nptel.ac.in/noc21_cs15/preview
- NPTEL Course: Prof. Soumya Kanti Ghosh, IIT Kharagpur , "Cloud Computing" https://onlinecourses.nptel.ac.in/noc25_cs107/preview
- NPTEL Course: Prof. Rajiv Misra, IIT Patna ," Foundation of Cloud IoT Edge ML" https://nptel.ac.in/courses/106104242

Third Year B.Tech Computer Engineering														
Semester- V														
Course Code: CE24PEC304C										**Course Name: Information Security**				
Teaching Scheme (Hours/Week)					**Examination Scheme**						**Credits**			
L	**P**	**T**	**OL**	**ODL**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**TOTAL**	**L**	**P**	**T**	**TOTAL**
3	**-**	**-**	**-**	**-**	**40**	**60**	**-**	**-**	**-**	**100**	**3**	**-**	**-**	**3**

Prerequisite: CE24PCC303: Computer Networks and Security, CE24MDM204: Mathematical Foundation for Gen AI

Course Objectives:

The course aims to

- Understand foundational security principles and symmetric encryption techniques.
- Apply the concepts of number theory in public-key cryptographic mechanisms and data integrity techniques.
- Explore advanced cryptographic methods such as ZKP, HE, SMPC and techniques ensuring privacy in modern systems.
- Understand security requirements for Computer Networks and software systems.
- Acquire the knowledge of Cybercrimes and Indian IT Act.

Course Outcomes:

After learning the course, the students will be able to:

CO1: Apply classical and symmetric cryptographic algorithms in real life system
CO2: Apply major asymmetric cryptographic algorithms in real life system
CO3: Apply modern cryptographic protocols used for privacy-preserving computation.
CO4: Analyze network and software security mechanisms to evaluate secure access management.
CO5: Demonstrate the use of standards and cyber laws to enhance information security.

Unit	Contents	Duration (Hrs.)
1	**Fundamentals of Security** Security Basics: Computer Security Concepts, The OSI security Architecture, Security attacks, Security services, Security mechanism, A model for Network Security **Symmetric Key Cryptography:** Classical Encryption Techniques: Stream Ciphers and Block Ciphers, Substitution Techniques, Transposition Techniques, symmetric Key Algorithms: Data Encryption standards, 3DES, Advanced Encryption standard Case Study: Scrambled word puzzle race using Rail fence, columnar transposition	8
2	**Asymmetric Key Cryptography** Number theory: Prime number, Fermat and Euler theorems , Testing for primality, Chinese reminder theorem, discrete logarithm, Public Key Cryptography concept, Public key cryptography algorithm: RSA, Elliptic Curve Cryptography, Diffie-Hellman key exchange, **Data Integrity Algorithms:** Cryptographic Hash Functions: Applications of Cryptographic Hash Functions, Requirements and Security, Hash function algorithms **Digital Signatures:** Digital Signatures and Schemes Case Study: Multi-server password verification using Chinese reminder theorem	8

3	**Modern Cryptographic Algorithms** Zero-Knowledge Proofs , Homomorphic Encryption (partially & fully HE), Secure Multi-Party Computation (SMPC) protocols, Federated learning & secure aggregation, Cryptographic randomness and entropy failures Case Study: Password-less Access to a Library Database using Zero-Knowledge Proof	8
4	**Network and Software Security** Access Control, Firewall: Firewall characteristics and access policy, Types of Firewall, DMZ networks, Intrusion prevention system: Host based, Network based, Hybrid. **Software Security:** Access control, Buffer Overflow, Operating system Security, trusted computing and multilevel security. Case Study: Hospital EHR (Electronic Health Records) Access Management	8
5	**Cyber Security and Indian Law** Introduction, Cybercrime and Information Security, Classification of Cybercrimes, The legal perspectives-An Indian perspective, Global perspective, Categories of Cybercrime, Indian IT Act, Challenges to Indian Law and Cybercrime Scenario in India, Amendments to the Indian IT Act Case Study: UPI scam detection using Malicious fake Mobile Applications (India)	8
	Total Hours	**40**

Text Books

1. William Stallings, "Cryptography and Network Security Principles and Practice", 7th edition, Pearson , ISBN : 978-1-292-15858
2. William Stallings, Lawrie Brown, "Computer Security Principles and Practice", 3rd_Edition, Pearson , ISBN : 978-0-13-3777392-7
3. Jonathan Katz, Yehuda Lindell, "Introduction to Modern Cryptography", 2nd Edition, CRC Press, ISBN: 9781466570269
4. Nina Godbole, Sumit Belapure, "Cyber Security", Wiley, ISBN: 978-81-265-2179- 1

Reference Books

1. Atul Kahate,"Cryptography and Network Security", 3rd Edition, McGraw Hill Education, ISBN: 9781259029882
2. Behrouz Forouzan, Debdeep Mukhopadhyay, "Cryptography and Network Security", 3rd edition, McGraw Hill Publication, ISBN: 9339220951
3. Joseph Kizza, "Computer Network Security and Cyber Ethics", McFarland & Company, Inc., Publishers , 4th Edition, ISBN: 978-1-4766-1560-8

Online References

- NPTEL Course: Prof. Sourav Mokhopadhyay, IIT Kharagpur, "Cryptography and Network Security" : https://nptel.ac.in/courses/106105162
- NPTEL Course: Prof. Chester Rebeiro , IIT Madras, "Information Security - 5 - Secure Systems Engineering ": https://onlinecourses.nptel.ac.in/noc26_cs71/preview

Third Year B. Tech Computer Engineering

Semester-V

Course Code : CE24PEC304D **Course Name: Game Development: From Design to Prototype**

Teaching Scheme (Hours/Week)					Examination Scheme						Credits			
L	P	T	OL	ODL	CIE	ETE	TW	PR	OR	TOTAL	L	P	T	TOTAL
3	-	-	-	-	40	60	-	-	-	100	3	-	-	3

Note: *As this is a lab-weighted industry elective with no traditional end-semester theory examination, evaluation is based fully on continuous in-semester assessment (CIE) and a final term-end practical/capstone evaluation (ETE), as detailed in the Assessment Structure below.*

Prerequisite: Basic computer literacy and familiarity with using a desktop operating system are assumed.

Course Objectives:

- To introduce students with no prior exposure to the fundamental concepts of game design and game development.
- To build proficiency in navigating and using the Unity game engine for 2D game development.
- To develop working knowledge of C# scripting for implementing game logic, player control, and interactivity.
- To enable students to apply physics-based collision systems and scoring mechanics within a game.
- To cultivate the ability to design, build, playtest, and iterate on a complete, playable 2D game independently.

Course Outcomes:

After completing the course, the students will be able to:

CO1: Explain core game design concepts, including the core loop, mechanics, dynamics, and aesthetics (MDA framework).

CO2: Design and playtest a simple game on paper, and apply playtester feedback through iteration.

CO3: Navigate the Unity editor and apply its core building blocks (GameObjects, Components, Prefabs) to construct 2D scenes.

CO4: Write basic C# scripts to control player movement and implement game logic.

CO5: Implement physics-based collision detection and scoring systems, and build and present one complete, playable 2D game.

Unit	Contents	Duration (Hrs.)
1	**Foundations of Game Design:** What Even Is a Game: Games vs. simulations vs. toys, the core loop, Mechanics, Dynamics, and Aesthetics (MDA framework, simplified). Paper Prototyping: Why prototype before code, defining rules, win/lose conditions and player goals, reading and applying playtester feedback.	8

2	**Unity Orientation and Environment Setup:** Navigating the Unity Editor: Scene, Hierarchy, Inspector, and Project panels. GameObjects and Components. Introduction to Prefabs (concept only). Building static 2D scenes: placing sprites, setting up a camera, and arranging a simple environment.	5
3	**Scripting Basics in C#:** Fundamentals of C# for game logic: variables, conditionals (if/else), and loops in a game context. Working with Transform.position and the Update() loop. Reading keyboard input and implementing keyboard-controlled player movement with screen/scene boundaries.	9
4	**Physics, Collision, and Gameplay Logic:** Rigidbody2D and Collider components. Differentiating colliders vs. triggers. Collision detection fundamentals and driving gameplay logic through collisions. Implementing a collectible pickup mechanic with disappearing objects and an increasing score.	6
5	**Game Feel, UI, and Capstone Integration:** Why feedback ('game feel'/juice) matters to player experience. Building basic in-game UI: score display, buttons, and simple sound effects. Capstone Project: integrating movement, collision, scoring, and UI into one complete playable 2D game; structured peer playtesting and feedback.	11
	Total Hours	**39**

List of Activities

Note: Submission of these activities is mandatory for evaluation.

- Deconstruct one published 2D game on paper, documenting its core loop, mechanics, dynamics, and win/lose conditions.
- Design and playtest a paper/board game prototype in pairs; document playtester feedback and the resulting design iteration.
- Construct a static Unity 2D scene using sprites, camera setup, and basic environment layout.
- Write a C# script implementing keyboard-controlled player movement constrained within defined screen boundaries.
- Implement a collectible pickup mechanic using Rigidbody2D and Collider/Trigger components, with an increasing score on pickup.
- Add a live in-game UI (score display, restart button) and integrate one sound effect into the existing project.
- Build and submit one complete, playable 2D game integrating movement, collision, scoring, and UI; participate in structured peer play testing.

Text Books

1. Introduction to Game Design, Prototyping, and Development — Author: Jeremy Gibson Bond, Publisher: Addison-Wesley / Pearson
2. The Art of Game Design: A Book of Lenses — Author: Jesse Schell — Publisher: CRC Press
3. Unity in Action: Multiplatform Game Development in C# — Author: Joseph Hocking — Publisher: Manning Publications

Reference Book

1. Game Programming Patterns — Author: Robert Nystrom — Publisher: Genever Benning

2. Rules of Play: Game Design Fundamentals — Authors: Katie Salen Tekinbaş & Eric Zimmerman — Publisher: MIT Press
3. Beginning C# Game Programming — Author: Kelvin Sung et al. — Publisher: CRC Press

Online References

- Unity Learn — Official Unity Technologies tutorials and learning pathways for 2D game development.
- Coursera Course: Michigan State University, Game Design and Development with Unity.
- Coursera Course: University of Colorado System, Introduction to Game Development.
- Unity Documentation — Official Unity Manual and Scripting API Reference.

Third Year B.Tech Computer Engineering

Semester- V

Course Code: CE24PCC306 Course Name: System Programming and Operating System Laboratory

Teaching Scheme (Hours/Week)					Examination Scheme						Credits			
L	P	T	OL	ODL	CIE	ETE	TW	OR	PR	TOTAL	L	P	T	TOTAL
-	2	-	-	-	-	-	-	-	50	50	-	1	-	1

Prerequisite: CE24PCC207- Object Oriented Programming Laboratory

Course Objectives:

The course aims to:

- Analyze the working of assemblers and macro processors.
- Compare process scheduling algorithms using suitable schedulers.
- Develop the ability to implement classical synchronization problems using semaphores
- and mutex.
- Apply memory management techniques for efficient resource utilization.

Course Outcomes :

After completing the course, the students will be able to:

CO1: Implement assembly and macro processing techniques to generate intermediate and machine code.

CO2: Implement Process Scheduling algorithms using Schedulers.

CO3: Implement Classical Synchronization Problems using semaphores and mutex.

CO4: Compare page replacement policies for efficient memory management.

Suggested List of Experiments (Any 6)

Sr. No.	Name of the Experiment	Duration (Hrs.)
1	Source Code Preprocessing- Pass 1 Assembler Simulate the first pass of an assembler for a simple assembly language program. The assembler should read the source code line by line, identify and process labels, symbols, and literals, maintain a Symbol Table and Literal Table, and handle assembler directives such as START, END, DC, and DS. The input is a sample assembly program containing symbols, directives, and instructions, and the output should include the Symbol Table, Literal Table, and intermediate code prepared for Pass 2 of the assembler.	2
2	**Machine Code Generation- Pass 2 Assembler** Simulate the second pass of an assembler that converts intermediate code from Pass 1 into final machine code. The assembler should read the intermediate code, use the Symbol Table and Literal Table generated in Pass 1, and resolve all addresses for instructions, labels, and literals. It should process assembler directives like START, END, DC, and DS, replacing placeholders in the intermediate code with actual memory addresses.	4
3	**Macro Table Construction- Macro Processor Pass 1** Design and implement the first pass of a macro processor for a simple assembly language program. The program should read the source code line by line, identify all macro definitions, and construct a Macro Name Table (MNT) to store macro names, starting addresses in the Macro Definition Table (MDT), and the number of parameters, along with a Macro Definition Table (MDT) to store the actual instructions of each macro with parameter	4

	placeholders. The input is a sample assembly program containing multiple macro definitions with or without parameters, and the output is the MNT and MDT ready for further processing by macro processor pass 2.	
4	**Macro Expansion and Source Code Generation: Macro Processor Pass 2** Design and implement the second pass of a macro processor that expands all macro calls in a source assembly program using the MNT and MDT generated in Pass 1. The program should read the source code line by line, identify macro calls, replace them with the corresponding instructions from the MDT, and substitute macro parameters with the actual arguments provided in the call. The input includes the source program containing macro calls and the MNT and MDT from Pass 1, and the output is the fully expanded source code, along with the updated MNT and MDT for reference	2
5	**Hospital Emergency Room- CPU Scheduling** A hospital emergency room receives patients with varying illness severity, and each patient requires different treatment durations. Critical patients should be prioritized, but non-critical patients must also be attended in reasonable time. Each patient is represented as a process with arrival time, burst time (treatment time), and priority (severity: 1 = critical, 2 = moderate, 3 = mild). Implement two CPU scheduling algorithms: Preemptive Priority Scheduling, where a newly arrived patient with higher severity can interrupt the current treatment, and Non-Preemptive Priority Scheduling, where once treatment starts, it runs to completion before the next patient is scheduled. Calculate waiting time and turnaround time for each patient and display the treatment order using Gantt charts for both algorithms.	2
6	**Call Center Customer Support- CPU Scheduling** A customer support call center receives multiple incoming calls from clients simultaneously. Each call requires different handling times depending on the complexity of the query. To ensure fairness, no single caller should be kept waiting too long, and every call should get attention in a timely manner. Each call is represented as a process with an arrival time and burst time (handling duration). Implement Round Robin scheduling with a fixed time quantum to allocate the CPU to calls in a cyclic order. Calculate waiting time and turnaround time for each call and display the service order using a Gantt chart.	2
7	**Online Order Processing- Producer–Consumer Problem** Simulate an online food delivery system where multiple orders are placed by customers and processed by a limited number of chefs. Orders act as produced items, and chefs act as consumers. Use counting semaphores to control resource availability and synchronize producer and consumer threads, and add a mutex to ensure that only one thread accesses the shared buffer (critical section) at a time. The program should manage a shared buffer where producers add orders and consumers remove them, preventing the buffer from overflowing or running empty while maintaining correct order processing.	2
8	**Library Database Access- Reader Writer Problem** Simulate a library information system where multiple students can view book availability at the same time, but adding new books or updating records must be done exclusively. Implement Reader Priority, allowing concurrent access for readers while writers perform updates exclusively. Use proper synchronization to avoid race conditions, deadlocks, and starvation, ensuring reliable access to library data.	2

9	**Bank Loan Resource Management - Banker's Algorithm** Simulate a banking system where multiple customers request loans, and each loan requires allocation of limited resources such as cash, staff approval, and credit verification. Implement the Banker's Algorithm for deadlock avoidance to determine whether the system is in a safe state. The program should take as input the number of processes (loan requests), available resources, maximum demand for each process, and currently allocated resources. It should evaluate resource requests, allocate resources safely if possible, and identify unsafe states or potential deadlocks.	2
10	**College Computer Lab - Page Replacement Algorithms** Simulate a College computer lab where students open different programs on computers with limited memory. Each program requires memory pages, and when memory is full, some pages must be removed to load new programs. Implement page replacement algorithms such as FIFO, LRU, or Optimal to decide which pages to remove. The program should take as input a sequence of program requests and the memory size, then calculate the number of page faults for each algorithm.	2
	Total Hours	**24**
Mini Projects List **Note:** **1. The Mini Project should be implemented in a group of 3-4 students.** **2. Students can select any one from the following mini project titles.** **3. Students can select a mini project title of their own choice.**		
P1	**Manufacturing System** Design and develop a program that applies the Shortest Job First (SJF) scheduling algorithm using C/C++/Java to optimize task management in a manufacturing system. The system will manage a series of tasks in the production line, where each task requires different processing times (burst times). The goal is to prioritize shorter tasks, ensuring that they are completed before longer ones, thereby optimizing throughput and reducing overall waiting times in the production process.	6
P2	**Customer Service Center** Design and develop a system that applies the Round Robin scheduling algorithm using C/C++/Java to manage the allocation of customer queries to available agents in a customer service center for an e-commerce website. The system should ensure that each customer is served in turn, with an equal time slice given to each agent to handle incoming queries. The goal is to minimize customer wait times and ensure fairness in query handling, preventing any agent from being overwhelmed while others remain idle.	
P3	**Multimedia Streaming Service (e.g. Netflix, YouTube)** Design a solution to simulate the Producer-Consumer problem in a multimedia streaming service (e.g., Netflix, YouTube). In this system, the producer represents the servers that generate or stream video/audio data, while the consumer represents the devices (smartphones, laptops, etc.) that retrieve and play the media data. The program must manage a shared buffer that holds the streaming data, ensuring smooth playback by properly synchronizing the producer and consumer. The producer will generate video/audio data at a specific rate, while the consumer will process and display it. If the buffer is full, the producer must wait for space to be available; if the buffer is empty, the consumer must wait for data to be produced.	
P4	**Collaborative Document Editing Systems (e.g. Google Docs)**	

	Design a solution to implement the following scenario, In a collaborative document editing system, the goal is to enable multiple users to read the document simultaneously while ensuring that only one user can edit a specific section of the document at any given time. The system should implement a reader-priority synchronization approach, where reading operations are non-blocking unless a user is actively editing the document. The solution must ensure that multiple users can view the document concurrently without interference, while preventing conflicts during editing by granting exclusive access to the document when needed. The system must handle concurrent read and write operations efficiently, facilitating real-time collaboration while preventing race conditions and ensuring data consistency.	
P5	**Printer Spooler System** Design a solution to simulate a Printer Spooler System in an office network using synchronization techniques inspired by the Dining Philosophers Problem. The program must efficiently manage multiple users (threads) sending print jobs to a shared printer while avoiding deadlocks and ensuring fair access to the printer resource.	
P6	**Airline Reservation Systems** Implement a solution to manage seat availability in an airline reservation system, ensuring that flight bookings do not exceed the available seat capacity. The system must handle multiple customers attempting to book tickets simultaneously, and efficiently allocate seats without causing overbooking. The program should use the Banker's Algorithm to manage seat allocation, checking whether a customer's request can be safely granted without leading to an unsafe overbooking situation. This approach will ensure that the reservation process remains within capacity, avoiding resource contention and guaranteeing smooth operations, even when multiple booking systems or interfaces are interacting with the same flight data.	
P7	Search Engine Query Results Design and develop a system that applies the Least Recently Used (LRU) caching algorithm for a search engine system. The program should manage the cache of search query results, ensuring that the least recently used results are evicted when the cache reaches its limit, while the most frequently accessed results are retained. The system should efficiently update the cache as new queries are processed and old queries are no longer needed. This implementation should optimize the search process by reducing search times and enhancing the overall user experience by minimizing the need to recompute results for previously searched queries.	
P8	**File System** Design and implement a file system for an operating system that demonstrates both CPU and memory virtualization concepts. The file system should operate on a virtual disk and support core file system functionalities such as disk initialization, directory management, file creation, reading, writing, and deletion. The system must internally manage disk blocks, free space, and file metadata using inode- or FAT-based allocation techniques, without relying on the host operating system's file system services. Additionally, the project should simulate process execution and memory usage for file operations to illustrate how CPU time sharing and virtual memory mapping are applied during file access.	
	Total Hours	**30**

Guidelines for Laboratory Conduction:

1. Assignments on all concepts are mandatory.
2. Assignments should be implemented on coding platforms such as GitHub, HackerRank, CodeChef.
3. Use of open-source software is encouraged.
4. Operating System recommended: - 64-bit Open-source Linux or its derivative.
5. Programming tools recommended: - G++/GCC, Eclipse.

Guidelines for Students:

1. The laboratory assignments are to be submitted by students in the form of a journal.
2. Journal consists of prologue, certificate, table of contents and handwritten write-up of each assignment (Title, Objectives, Problem Statement, Outcomes, Theory- Concept, algorithm , program code with suitable input and output, conclusion).

Guidelines for Laboratory Assessment:

1. Continuous assessment of laboratory work is done based on overall performance and Laboratory performance of students.
2. Each Laboratory assignment should be evaluated based on parameters with appropriate weightage.
3. Suggested parameters for overall assessment as well as each Laboratory assignment assessment include- timely completion, performance, innovation, efficiency, punctuality and neatness.

Third Year B.Tech Computer Engineering

Semester-V

Course Code:CE24PCC307 Course Name: Data Science and Machine Learning Laboratory

Teaching Scheme (Hours/Week)					Examination Scheme						Credits			
L	P	T	OL	ODL	CIE	ETE	TW	OR	PR	TOTAL	L	P	T	TOTAL
-	2	-	-	-	-	-	-	-	50	50	-	1	-	1

Prerequisite: CE24PCC251 Database Management System

Course Objectives:

The Course aims to

- Apply feature engineering for real-world datasets.
- Perform exploratory data analysis and visualization.
- Build supervised models for regression and classification.
- Implement ensemble learning and unsupervised learning techniques.
- Design reinforcement learning agents using feedback.

Course Outcomes:

After learning the course, the students will be able to:

CO1: Apply preprocessing techniques for real-world datasets.
CO2: Analyze data to extract meaningful insights.
CO3: Evaluate regression and classification model performance.
CO4: Implement ensemble learning and clustering algorithms effectively.
CO5: Develop reinforcement learning agents using feedback.

A list of assignments with required datasets is provided below. Instructors may adapt these activities and select alternative assignments and datasets following similar structure and complexity.

Suggestive List of Experiments

Sr. No.	Name of the Experiment	Duration (Hrs.)
1	Perform feature engineering: remove duplicates, transform/replace values, normalize/scale, impute missing, wrapper selection (forward/backward), encoding categorical data, PCA/Kernel.	2
2	Perform Exploratory Data Analysis on a dataset, including summary statistics, correlation analysis, and visualization of feature distributions and relationships using histograms, scatter plots, and box plots.	2
3	Create various visualizations (bar plots, pie charts, heatmaps, histograms) to gain insights from the dataset and highlight trends and patterns.	2
4	Build predictive models using supervised learning techniques. Implement linear, ridge, and lasso regression using gradient descent, analyze bias–variance tradeoff, identify underfitting and overfitting, and evaluate models using MAE, RMSE, R^2 score, and cross-validation.	2
5	Develop a classification system using supervised learning. Apply K-Nearest Neighbors, decision tree. Evaluate model performance using confusion matrix, accuracy, precision, recall, F1-score, ROC curve, and AUC.	4
6	Implement ensemble learning models using bagging and boosting techniques by combining multiple weak learners, compare their performance and evaluate results using suitable classification or regression metrics.	4

7	Apply clustering techniques including K-means, DBSCAN. Evaluate clustering results.	4
8	Implement a reinforcement learning model using a feedback-based learning framework. Apply reward and penalty mechanisms, use function approximation techniques, and analyze agent performance over multiple learning episodes.	4
	Total Hours	**24**

List of datasets suggested but not limited to-

1. Netflix Movies and TV Show
   https://www.kaggle.com/datasets/shivamb/netflix-shows
2. Credit Card Transactions Dataset
   https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud
3. Loan Default Prediction Dataset
   https://www.kaggle.com/datasets/wordsforthewise/lending-club
4. Stock Market Historical Data (Multiple Companies)
   https://www.kaggle.com/datasets/jacksoncrow/stock-market-dataset
5. Yahoo Finance Stock Data
   https://finance.yahoo.com
6. Online Learning Engagement Dataset
   https://www.kaggle.com/datasets/aljarah/xAPI-Edu-Data
7. Cricket Match Statistics Dataset
   https://www.kaggle.com/datasets/hijest/ipl-matches-2008-2020
8. Social Media Analytics (Tweets Dataset)
   https://www.kaggle.com/datasets/kazanova/sentiment140

Mini-Project Instructions:

1. **Group Formation:** Students can form groups of 3 members each.
2. **Individual Project Option:** Any student not willing to join a group project may choose to work independently on a project in their own domain of interest.
3. **Project Objective:** Develop a machine learning model to solve a specific problem using a relevant dataset.
4. **Algorithm Selection:** Choose one or more appropriate algorithms suitable for the problem.
5. **Model Development:** Train and test the model(s) thoroughly, ensuring proper data preprocessing and feature engineering.
6. **Performance Evaluation:** Compare the model(s) using appropriate evaluation metrics.
7. **Documentation:** Submit a project report including problem statement, dataset description, methodology, results, and conclusions.
8. **Presentation:** Groups/individuals may be required to present their project outcomes at the end of the term.

1	Develop a data-driven system that analyzes IPL ball-by-ball and player statistics to evaluate player performance and predict player form for upcoming matches. The system should preprocess large-scale match data, engineer meaningful features such as consistency score and strike rate trends, and apply supervised learning models to classify players as "In-Form" or "Out-of-Form." The final system should provide visual insights to support team selection decisions. **Key Tasks** ● Data cleaning and preprocessing ● Feature engineering and selection ● EDA and visualization ● Classification model implementation ● Performance evaluation	
2	Build a user segmentation and recommendation analysis system for an OTT platform using viewer behavior data. The system should cluster users based on	

	watch history, genre diversity, and engagement metrics. Insights from clustering should be used to suggest personalized content strategies and subscription plans. **Key Tasks** ● EDA on user behavior ● Clustering using K-Means and DBSCAN ● Cluster evaluation ● Visualization and insights	6
3	Design a data-driven system for an educational institution that analyzes student academic records, attendance, internal assessments, and engagement metrics to predict student performance and identify at-risk students. The system should help faculty take proactive interventions to improve learning outcomes. **Key Tasks** ● Data cleaning and preprocessing of academic and behavioral data ● Feature engineering (attendance consistency, assessment trends, engagement score) ● Exploratory Data Analysis (EDA) and visualization of performance patterns ● Classification model implementation to predict "At-Risk" vs "Not At-Risk" students ● Model performance evaluation using appropriate metrics ● Interpretability of results to support academic decision-making	
4	Design a credit risk prediction system for a fintech lending platform using applicant demographic and financial data. The system should predict loan default risk and assist lenders in making informed decisions. **Key Tasks** ● Handling missing and skewed data ● Feature engineering ● Classification model evaluation ● Interpretability of results	
5	Develop a forecasting system to predict energy consumption patterns using smart meter data. The system should apply regression and time-based feature engineering to help city planners optimize energy usage. **Key Tasks** ● Data preprocessing and EDA ● Regression modeling ● Forecast evaluation ● Visualization of consumption trends	
6	Build an academic analytics system that predicts student performance and dropout risk using attendance, assessment scores, and engagement data. The system should assist institutions in early intervention planning. **Key Tasks** ● Data preprocessing and feature creation ● Supervised learning models ● Evaluation and insights ● Ethical considerations	
	Total Hours	**30**

Guidelines for Laboratory Conduction:

1. All the assignments are mandatory.
2. Programming tools recommended: - Eclipse/vscode/Jupyter Notebook

Guidelines for Students:

1. The laboratory assignments are to be submitted by students in the form of a journal.

2. Journal consists of Vision Mission of Institute, Department, certificate, table of contents and handwritten write-up of each assignment (Title, Objectives, Problem Statement, Outcomes,Date of Completion, Assessment grade/marks and assessor's sign, Theory- Concept,algorithm, time complexity, sample input and expected output, conclusion).

Guidelines for Laboratory Assessment:

1. Continuous assessment of laboratory work is done based on overall performance and Laboratory performance of students.
2. Each Laboratory assignment assessment should assign grade/marks based on rubrics with appropriate weightage.
3. Suggested rubrics for overall assessment as well as each Laboratory assignment assessment include timely completion, performance, innovation, efficiency, punctuality and neatness.

Third Year B.Tech Computer Engineering

Semester- V

Course Code: CE24PCC308 **Course Name: Engineering Design and Innovation-I**

Teaching Scheme (Hours/Week)					Examination Scheme						Credits			
L	P	T	OL	ODL	CIE	ETE	TW	OR	PR	TOTAL	L	P	T	TOTAL
-	4	-	-	-	-	-	25	-	-	25	-	2	-	2

Prerequisite: CE24CEP208- Project Based Learning

Course Objectives:

The course aims to

- Develop critical thinking and problem solving ability by exploring and proposing solutions to realistic/social problems.
- Evaluate alternative approaches, and justify the use of selected tools and methods,
- Emphasize learning activities that are long-term, inter-disciplinary and student-centric.
- Develop an ecosystem to promote entrepreneurship and research culture among the Students.

Course Outcomes :

After completing the course, the students will be able to:

CO1: Analyze a real-world engineering problem, formulate system requirements, and design an appropriate solution by integrating concepts from foundational and core courses.

CO2: Create innovative solutions for real-world social challenges by applying critical thinking and structured problem-solving techniques.

CO3:Demonstrate effective teamwork, project planning, technical documentation, and communication skills while presenting and evaluating the developed solution.

Preamble:

The Engineering Design and Innovation Lab provides students with an opportunity to integrate the knowledge and skills gained through foundational and second-year core courses, including **programming, data structures, object-oriented programming, digital electronics, database management systems, software engineering, programming languages, and web technologies**. Through a structured mini-project, students apply these concepts to solve real-world problems by following the engineering design process—from problem identification and requirement analysis to solution design, implementation, testing, documentation, and presentation. The focus extends beyond developing a prototype to fostering system-level thinking, innovation, teamwork, effective communication, and sound engineering design practices.

The laboratory promotes experiential and project-based learning by bridging the gap between theory and practice while nurturing critical thinking, computational problem-solving, and the use of modern engineering tools. Students are encouraged to develop sustainable, user-centric solutions for societal, industrial, healthcare, educational, agricultural, and smart city applications. Successful completion of the mini-project prepares students for advanced coursework, internships, research, entrepreneurship, and larger capstone projects by strengthening both their technical competence and professional skills.

Group Structure:

- There should be a team/group of 3-5 students.
- A supervisor/mentor teacher assigned to individual groups.
- It is useful to group students of different abilities together.

Contact Pattern: Weekly project studio, faculty mentoring, reviews, demonstrations, and documentation support.

Selection of Project/Problem:

- Students must focus to initiate the task/idea .The idea inception and consideration shall be from following areas as a real world problem:
- Health Care, Agriculture, Defense, Education, Smart City, Smart Energy, Swaccha Bharat Abhiyan, Environment, Women Safety etc.
- This is the sample list to start with. Faculty and students are free to include other areas which meet the society requirements at large.
- The model begins with the identifying of a problem, often growing out of a question or "wondering". This formulated problem then stands as the starting point for learning. Students design and analyze the problem/project within an articulated disciplinary subject frame/domain.
- A problem can be theoretical, practical, social, technical, symbolic, cultural, and/or scientific and grows out of students' wandering within different disciplines and professional environments. A chosen problem has to be exemplary. The problem may involve an interdisciplinary approach in both the analysis and solving phases.
- By example, a problem needs to refer back to a particular practical, scientific, social and/or technical domain. The problem should stand as one specific example or manifestation of more general learning outcomes related to knowledge and/or modes of inquiry.

Teacher's Role :

- The teacher is not the source of solutions, rather he will act as the facilitator and mentor.
- To utilize the principles of problem solving, critical thinking and metacognitive skills of the students.
- To make the group aware about time management.
- Commitment to devote the time to solve student's technical problems and interested in
- helping students to empower them better.

Student's Role:

- Students must have the ability to initiate the task/idea .They should not be mere imitators.
- They must learn to think.
- Students working in projects must be responsible for their own learning.
- Students must quickly learn how to manage their own learning, instead of passively receiving instruction.
- Students in projects are expected to work in groups.
- They have to develop interpersonal and group process skills, such as effective listening or
- coping creatively with conflicts.

Suggested List of Activities

Sr. No.	Title	Foundational Corses	Duration (Hrs)
1	Library Issue-Return and Fine Automation	Subjects: OOP, DBMS, Data Structures, Software Engineering,Web Technology Focus: entity relationships, transaction handling, hashing/search, validation	52
2	Hostel Complaint Tracker with Escalation Matrix	Subjects: OOP, DBMS, Data Structures, Software Engineering,Web Technology Focus: state transitions, priority queues, audit trails, stakeholder views	
3	Placement Eligibility and Resume Screening Portal	Subjects: DBMS, Data Structures, OOP, Software Engineering Focus: filtering rules, indexed search, structured candidate data, reports	

4	Canteen Order and Inventory Management System	Subjects: OOP, DBMS, Data Structures, Software Engineering,Web Technology Focus: queue handling, stock updates, transaction consistency, UI flow	
5	Question Bank and Quiz Engine	Subjects: DBMS, OOP, Data Structures, Software Engineering Focus: randomization, tagging, scoring logic, reusable components	
6	CPU Scheduling Simulator	Subjects: Data Structures, Digital Electronics and Computer Organization, OOP, Principles of Programming Languages Focus: process queues, scheduling metrics, state transitions, visualization	
7	Traffic Junction Logic Controller	Subjects: Digital Electronics and Computer Organization, OOP, Data Structures, Software Engineering Focus: state machines, timing logic, event handling, simulation	
8	Secure File Sharing over Department LAN	Subjects: OOP, DBMS, Software Engineering, Data Structures Focus: user roles, file indexing, access control, logging	
9	Student Marks Analytics and CO Attainment Tool	Subjects: DBMS, OOP, Data Structures, Software Engineering Focus: academic data modelling, aggregation, dashboards, reporting	
10	Leave and Payroll Management System	Subjects: DBMS, OOP, Data Structures, Software Engineering Focus: rule engines, calculations, normalized schema, report design	
11	Exam Seating and Invigilation Planner	Subjects: Data Structures, DBMS, OOP, Software Engineering Focus: allocation logic, constraint satisfaction, report generation, testing	
12	Alumni Mentoring Portal	Subjects: OOP, DBMS, Data Structures, Software Engineering Focus: profile management, matching logic, interaction records, modular design	

Text Books

1. A new model of problem based learning. By Terry Barrett. All Ireland Society for higher education (AISHE). ISBN:978-0-9935254-6-9; 2017
2. Problem Based Learning. By Mahnazmoallem, woei hung and Nada Dabbagh, Wiley Publishers. 2019.
3. Stem Project based learning and integrated science, Technology, Engineering and mathematics approach. By Robert Robart Capraro, Mary Margaret Capraro

Reference Books

- De Graaff E, Kolmos A., red.: Management of change: Implementation of problem-based and project-based learning in engineering. Rotterdam: Sense Publishers. 2007.
- Project management core textbook, second edition, Indian Edition , by Gopalan.
- The Art of Agile Development. By James Shore & Shane Warden.

Third Year B.Tech Computer Engineering

Semester-V

Course Code: CE24AEC311 Course Name: Reasoning and Aptitude Development- I

Teaching Scheme (Hours/Week)					Examination Scheme						Credits					
L	P	T	OL	ODL	CIE	ETE	TW	OR	PR	TOTAL	L	P	T	OL	ODL	TOTAL
-	-	1	-	-	-	-	25	-	-	-	-	-	1	-	-	1

Prerequisite: Basic Mathematical Concepts

Course Objectives:

The course aims to

- Develop fast and accurate mental calculation skills using Vedic Mathematics techniques for arithmetic operations.
- Strengthen students' understanding of number systems, divisibility rules, and numerical properties for efficient problem solving.
- Enhance competency in percentages, profit–loss, discounts, ratios, and partnerships for real-life and competitive exam applications.
- Build logical reasoning abilities through linear, circular, and complex puzzle-based arrangements.
- Improve analytical thinking by applying concepts of syllogisms, blood relations, and direction sense.
- Prepare students for aptitude-based assessments and competitive examinations by integrating speed, accuracy, and logical reasoning skills.

Course Outcomes:

After learning the course, the students will be able to:

1. Perform rapid mental calculations involving addition, subtraction, multiplication, squaring, square roots, and cube roots with high accuracy.
2. Analyze and solve number system problems involving divisibility, unit digits, factorials, HCF–LCM, and remainders.
3. Apply percentage, profit–loss, discount, ratio, and partnership concepts to solve quantitative aptitude problems efficiently.
4. Solve seating arrangement and puzzle-based problems, including linear, circular, matrix, and scheduling puzzles, using logical reasoning.
5. Evaluate syllogism problems and blood relation cases using Venn diagrams, family trees, and logical inference.
6. Demonstrate improved speed, accuracy, and confidence in quantitative and reasoning sections of placement and competitive examinations.

Unit	Contents - Foundations & Numerical Proficiency	Duration (Hrs.)
1	**Vedic Math & Mental Calculation** ● Speed Addition & Subtraction (Left-to-Right method) ● Base Method Multiplication (Numbers near 10, 100, 1000) ● Squaring Techniques (Base 50/100 and numbers ending in 5)	2

	● Cube Roots and Square Roots of perfect numbers	
2	**Number Systems** 1. Classification (Rational, Irrational, Prime, Composite) 2. Divisibility Rules (2 to 19) 3. Unit Digit and Cyclicity 4. Factorial theory and Number of Zeroes 5. HCF & LCM (Models: Remainder cases) 6. Remainder Theorem basics	2
3	**Percentages** 1. Percentage Increase/Decrease and Multiplying Factors 2. Successive Percentage Change Formula 3. Product Constancy Rule	2
4	**Profit, Loss & Discount** 1. CP, SP, and Profit/Loss Percentage 2. Relation between Markup, Discount, and Profit 3. Successive Discounts and Single Equivalent Discount	2
5	**Ratio, Proportion & Partnership** 1. Ratio Scaling and Merging (A:B and B:C to A:B:C) 2. Mean, Third, and Fourth Proportional 3. Coin-based problems 4. Partnership: Profit sharing based on (Capital $\times$ Time) 5. Problems on Ages	2
6	**Linear Arrangements** ● Single Row: All facing North/South ● Dual Row: Facing each other ● Directional cues (Left of, Second to the right, Immediate neighbours) ● Ranking and Ordering (Tallest/Shortest, Top/Bottom)	2
7	**Circular & Polynomial Arrangements** ● Circular: Facing Centre vs. Facing Outwards ● Inward/Outward Mixed configurations ● Square and Rectangular Table arrangements ● Relationship-based seating (Combining Seating + Blood Relations)	2
8	**Complex Puzzles** ● Grid/Matrix Puzzles (3 or 4 variables: Name, City, Profession, Color) ● Scheduling Puzzles (Days of the week, Months, Years) ● Floor-based Puzzles ● Box-based Puzzles (Stacking logic)	2
9	**Syllogisms** 5. Standard Propositions (All, Some, No, Some Not) 6. Venn Diagram Method 7. "Only a Few" and "Few" cases 8. Possibility-based conclusions 9. Complementary pairs (Either-Or cases)	2
10	**Blood Relations & Direction Sense** ● Family Tree Construction (Standard symbols) ● Coded Blood Relations ● Direction Compass and Degree Turns	2
	Total Hours	**20**
Reference Books		

Quant Basics	R.S. Aggarwal	Concept Clarity
Speed Math	M. Tyra	Time Management
Logic/Puzzles	K. Kundan / R.S. Aggarwal	Exam-specific Patterns
Critical Thinking	M.K. Pandey	Logical Derivations
DI / Advanced	Arun Sharma	Complex Data & CAT

Assessment

Assessment will be done by the Course Teacher. MCQ Test can be conducted based on the syllabus.

SEMESTER VI

Third Year B.Tech Computer Engineering

Semester-VI

Course Code: CE24PCC351 **Course Name: Artificial Intelligence**

Teaching Scheme (Hours/Week)					Examination Scheme						Credits			
L	P	T	OL	ODL	CIE	ETE	TW	OR	PR	TOTAL	L	P	T	TOTAL
3	-	-	-	-	40	60	-	-		100	3		-	3

Prerequisite: AI24PCC151:Foundation of Artificial Intelligence, CE24PCC20:Data Structures

Course Objectives:

The course aims to

- Apply core AI problem solving methods, including search techniques, heuristics, and logic-based reasoning.
- Analyze real world problems and design effective solutions using knowledge representation and automated planning.
- Apply fuzzy logic and genetic algorithm for problem solving

Course Outcomes:

After learning the course, the students will be able to:

CO1: Analyze a given problem to identify required knowledge and formulate a suitable strategy for its solution

CO2: Design logic-based reasoning systems and automated planners to solve real-world AI tasks.

CO3: Apply game theory concepts to make optimal decisions in deterministic, stochastic, and partially observable games

CO4: Apply fuzzy logic techniques to model uncertainty and imprecision in real-world problem scenarios

CO5: Apply genetic algorithms to optimize solutions for computational and engineering problems

Unit	Contents	Duration (Hrs.)
1	**Knowledge-Based Agent** Logical Agents, Knowledge-Based Agents, The Wumpus World, Logic, Propositional Logic: A Very Simple Logic, Propositional Theorem Proving, Effective Propositional Model Checking, Agents Based on Propositional Logic, First-Order Logic, Representation Revisited, Syntax and Semantics of First-Order Logic, Using First-Order Logic, Knowledge Engineering in First-Order Logic. Case Study: Logical Agent Framework for Medical Diagnosis	8
2	**Reasoning and Planning** Inference in First-Order Logic, Propositional vs. First-Order Inference, Unification and First-Order Inference, Forward Chaining, Backward Chaining, Resolution. Automated Planning, Classical Planning, Algorithms for Classical Planning, Heuristics for Planning, Hierarchical Planning Case Study: Intelligent Planning Agent for Autonomous Warehouse Task Scheduling	8

3	**Game Theory and Constraint Satisfaction Problem** Game Theory, Optimal Decisions in Games, Heuristic Alpha–Beta Tree Search, Monte Carlo Tree Search, Stochastic Games, Partially Observable Games, Limitations of Game Search Algorithms, Constraint Satisfaction Problems (CSP), Constraint Propagation: Inference in CSPs, Backtracking Search for CSPs. Case Study: Solving Sudoku Using Constraint Propagation	8
4	**Fuzzy Sets and Logic** Basic concepts of fuzzy logic, Fuzzy sets and Crisp sets, Fuzzy set theory and operations, Properties of fuzzy sets, Fuzzy and Crisp relations, Fuzzy to Crisp conversion. Membership functions, interference in fuzzy logic, fuzzy if-then rules, Fuzzy implications and Fuzzy algorithms, Fuzzyfications and Defuzzifications, fuzzy expert system	8
5	**Genetic Algorithm** Basic concepts, working principle, procedures of GA, flow chart of GA, Genetic representations, (encoding) Initialization and selection, Genetic operators, Mutation, Generational Cycle, Traditional algorithm vs genetic algorithm, simple GA, general genetic algorithm, schema theorem, Classification of genetic algorithm, Holland classifier systems, genetic programming, applications of genetic algorithm, Convergence of GA. Applications and advances in GA, Differences and similarities between GA and other traditional method, applications	8
	Total Hours	**40**

Text Books

1. Stuart Russell and Peter Norvig, "Artificial Intelligence: A Modern Approach", Third
2. edition, Pearson, 2003, ISBN :10: 0136042597
3. Deepak Khemani, "A First Course in Artificial Intelligence", McGraw Hill Education(India),
4. 2013, ISBN : 978-1-25-902998-1
5. Elaine Rich, Kevin Knight and Nair, "Artificial Intelligence", TMH, ISBN-978-0-07-008770-5
6. S. Rajsekaran and G.A. Vijayalakshmi Pai, "Neural Networks, Fuzzy Logic and Genetic Algorithm: Synthesis and Applications" , Prentice Hall of India, ISBN: 0451211243

Reference Books

1. Nilsson Nils J , "Artificial Intelligence: A new Synthesis", Morgan Kaufmann Publishers
2. Inc. San Francisco, CA, ISBN: 978-1-55-860467-4
3. Patrick Henry Winston, "Artificial Intelligence", Addison-Wesley Publishing Company,
4. ISBN: 0-201-53377-4
5. Andries P. Engelbrecht-Computational Intelligence: An Introduction, 2nd Edition-Wiley
6. India- ISBN: 978-0-470-51250-0
7. Dr. Lavika Goel, "Artificial Intelligence: Concepts and Applications", Wiley publication, ISBN: 9788126519934
8. Dr. Nilakshi Jain, "Artificial Intelligence, As per AICTE: Making a System Intelligent", Wiley publication, ISBN: 9788126579945
9. Timothy J. Ross, "Fuzzy Logic with Engineering Applications" , Wiley India, ISBN: 978-0-470-74376-8
10. N.P.Padhy, "Artificial Intelligence and Intelligent Systems" Oxford University Press, ISBN 10: 0195671546

Online References

- NPTEL Course: An Introduction to Artificial Intelligence, IIT Delhi Prof. Mausam https://onlinecourses.nptel.ac.in/noc22_cs56/preview
- NPTELCourse:Fuzzy Sets, Logic and Systems & Applications, IIT Kanpur, Prof. Nishchal Kumar Verma, https://nptel.ac.in/courses/108104157

Third Year B.Tech Computer Engineering

Semester- VI

Code: CE24PCC352 **Course Name: Design and Analysis of Algorithms**

Teaching Scheme (Hours/Week)					Examination Scheme						Credits			
L	P	T	OL	ODL	CIE	ETE	TW	OR	PR	TOTAL	L	P	T	TOTAL
3	-	-	-	-	40	60	-	-	-	100	3	-	-	3

Prerequisite: CE24PCC201:Data Structures

Course Objectives:

The course aims to:

- Apply fundamental algorithmic concepts, including time and space complexity
- Develop problem solving abilities using divide and conquer and Greedy strategies
- Apply Algorithmic strategies to solve computational problems.
- Analyze appropriate algorithmic solutions for complex and real-life engineering problems.

Course Outcomes:

After learning the course, the students will be able to:

CO1: Interpret algorithms and the need of correctness of any algorithm
CO2: Predict the optimal solution using divide and conquer and Greedy method
CO3: Evaluate the problems using Dynamic programming strategy
CO4: Analyze the complex problems using backtracking and branch and bound methods
CO5: Evaluate the problem using mathematical theories

Unit	Contents	Duration (Hrs.)
1	**Algorithms and Problem Solving:** **Algorithm:** The Role of Algorithms in Computing, Algorithms as technology , Design of Algorithm, Need and Confirming correctness of Algorithm, **Algorithm Design Approach:** Iterative and non-iterative design issues, Problem solving Principles, Classification of algorithms, Problem solving strategies, Best case, worst case and average case, growth rate, (O, Ω, Θ), Asymptotic Notations. Case study: Analyzing file search system using Master Theorem	8
2	**Divide and Conquer:** General Strategy, Exponentiation, Quick Sort and Merge Sort, Convex Hull problem **Greedy Method:** General Strategy, Knapsack problem, Job sequencing with Deadlines, activity selection problem, Minimal Spanning Trees and Dijkstra's algorithm. Case Study: Skyline problem, CPU Scheduling in Operating systems	8
3	**Dynamic Programming**, Principle, control abstraction, time analysis of control abstraction,binomial coefficients, OBST, 0/1 Knapsack, Matrix Chain Multiplication, Traveling Salesperson Problem, Flow Shop Scheduling. Case Study: optimized delivery route using Bellman ford algorithm	8
4	**Backtracking and Branch and Bound:** **Backtracking:** Principle, control abstraction, time analysis of control abstraction, 8-queen problem, graph coloring problem, Sum of subsets problem **Branch-and-Bound:** Principle, control abstraction, time analysis of control	8

	abstraction,Strategies- FIFO, LIFO and LC approaches, 0/1 Knapsack, Traveling Salesperson Problem. Case Study: Cryptarithmetic (Alphametic) Puzzles, Airline Crew Scheduling	
5	**Analysis of Algorithms and Complexity Theory:** Counting Dominant operators, polynomial and non-polynomial problems, deterministic and non-deterministic algorithms,P-class problems, NP-class of problems, Approximation algorithms, Polynomial problem reduction NP complete problems- vertex cover, 3-SAT and NP hard problem - Hamiltonian cycle. Case Study: Steiner Tree Problem, Bin Packing Algorithm	8
	Total Hours	**40**

Text Books

1. Ellis Horowitz, Sartaj Sahni, Sanguthevar Rajasekaran, "Fundamentals of Computer Algorithms", Universities Press
2. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein, "Introduction to Algorithms", MIT Press; ISBN 978-0-262-03384-8

Reference Books

1. Michael T. Goodrich, Roberto Tamassia, "Algorithm Design: Foundations", Wiley, ISBN 978-81-265-0986-7
2. Parag Himanshu Dave, Himanshu Bhalchandra Dave, "Design And Analysis of Algorithms", Pearson Education, ISBN 81-7758-595-9
3. Anany Levitin, "Introduction to Design and Analysis of Algorithms",, Person Publication
4. Don Gusfield, "Algorithms on Strings, Trees and Sequences", Cambridge University Press,ISBN:0- 521-67035-7
5. Gilles Brassard, Paul Bratley, "Fundamentals of Algorithmics", PHI

Online References

- NPTEL Course: Prof. Abhiram G Ranade, Prof. Ajit A Diwan, Prof. Sundar Viswanathan, IIT Bombay, "Algorithm Design Techniques" nptel.ac.in/courses/106101060
- NPTEL Course: Prof. Madhavan Mukund, Chennai, Mathematical Institute, "Design and Analysis of Algorithms" https://onlinecourses.nptel.ac.in/noc21_cs22/preview
- NPTEL Course: Prof. Sourav Mukhopadhyay , IIT Kharagpur : https://onlinecourses.nptel.ac.in/noc25_cs33/preview

Third Year B.Tech Computer Engineering

Semester- VI

Course Code: CE24PCC353 **Course Name: Theory of Computation**

Teaching Scheme (Hours/Week)					Examination Scheme						Credits			
L	P	T	OL	ODL	CIE	ETE	TW	OR	PR	TOTAL	L	P	T	TOTAL
3		-	-	-	40	60	-	-		100	3		-	3

Prerequisite: CE24MDM20-Mathematical Foundation

Course Objectives:

The course aims to:

- Understand formal language principles and model problems using finite automata.
- Analyze regular languages using regular expressions.
- Design and simplify context-free grammars for language specification.
- Construct pushdown automata for recognizing context-free languages.
- Build Turing Machine models and study computational complexity classes.

Course Outcomes:

After learning the course, the students will be able to:

CO1: Apply the principles of formal languages to define Finite Automata and its variants.
CO2: Apply the concept of regular expressions for regular language analysis.
CO3: Construct simplified Context-Free Grammars from language specifications.
CO4: Design Pushdown Automata for accepting Context-Free Languages.
CO5: Construct Turing Machine models and analyze complexity classes.

Unit	Contents	Duration (Hrs.)
1	**Finite Automata(FA)** Introduction to symbol notations, Concept of Basic Machine and Finite State Machine, **Finite Automata**: Language accepted by FA, Definition of Regular Language. FA without output: Deterministic and Nondeterministic FA (DFA & NFA), epsilon- NFA and inter-conversion. Minimization of DFAs. FA with output: Moore and Mealy machines, inter-conversion. **Case Study:** Finite State Machine for vending machine	10
2	**Regular Expressions (RE)** Operators of RE, Precedence of operators, Algebraic laws for RE, Language to Regular Expressions, RE Examples, Conversions: RE to DFA, DFA to RE using Arden's theorem, Pumping Lemma for Regular languages. **Case Study:** RE in text search & Replace	6
3	**Grammar** Basic Elements of Grammar, **Context Free Grammar (CFG):** Formal Definition, Sentential form, Derivation- Leftmost, Rightmost, Derivation Tree, Context Free Language (CFL), Ambiguous Grammar and removing ambiguity from grammar, Writing grammar for language, Simplification of CFG, Normal Forms, Chomsky Hierarchy, **Regular grammar**- Definition, left-linear, right-linear grammar, Applications of grammar **Case Study:** Syntax Analysis in Compiler Design	8
4	**Pushdown Automata (PDA)** Introduction, Formal definition, Types of PDA (Deterministic PDA and Non Deterministic PDA), Equivalence of Acceptance by Final State and Empty stack, Construction of PDA (DPDA, NPDA), Equivalence of PDA and CFG: CFG to PDA	8

	conversion, PDA to CFG, Applications of PDA. **Case Study:** Top down and Bottom up parsing	
5	**Turing Machines (TM) and Complexity Theory** Formal definition, Language Acceptability by TM, Design of TM, Description of TM, Techniques for TM Construction, Variants of TM, Halting Problem of TM. Decidable Problems and Undecidable Problems, Complexity Classes: Class P and examples, Class NP and examples, P Problem Versus NP Problem, NP-completeness and NP-hard Problems. **Case Study:** Post Correspondence Problem (PCP)	8
	Total Hours	**40**

Text Books

1. John Martin, "Introduction to Languages and The Theory of Computation", 4th Edition, McGraw-Hill Education, ISBN-13: 978-1-25-900558-9, ISBN-10: 1-25-900558-5
2. John E. Hopcroft, Rajeev Motwani, Jeffrey D.Ullman,"Introduction to Automata Theory Languages and Computation", 3rd Edition Addison-Wesley, ISBN 0-201-44124-1
3. Vivek Kulkarni, "Theory of Computation", Oxford University Press, 1st edition, ISBN 0-19-808458

Reference Books

1. Daniel Cohen, "Introduction to Computer Theory", Wiley & Sons, 2nd edition, ISBN 97881265133454.
2. J. Carroll & D Long, "Theory of Finite Automata", Prentice Hall, ISBN 0-13-913708-45.
3. Kavi Mahesh, "Theory of Computation: A Problem-Solving Approach", Wiley India, ISBN1081265331106.
4. Michael Sipser, "Introduction to the Theory of Computation", Cengage Learning, 3rd Edition ISBN-13:97811331878137.

Online References

- NPTEL Course: Prof. Subrahmanyam Kalyanasundaram, IIT Hyderabad, "Theory of Computation" : https://onlinecourses.nptel.ac.in/noc25_cs70/preview
- NPTEL Course: Prof. Ragunath Tewari, IIT Kanpur, "Theory of Computation": https://onlinecourses.nptel.ac.in/noc19_cs79/preview

Third Year B.Tech Computer Engineering

Semester VI

Course Code : CE24PEC354A **Course Name: Developments and Operations**

Teaching Scheme (Hours/Week)					Examination Scheme						Credits			
L	P	T	OL	ODL	CIE	ETE	TW	OR	PR	TOTAL	L	P	T	TOTAL
3	-	-	-	-	40	60	-	-	-	100	3	-	-	3

Pre-requisites: CE24PEC304B: Cloud Computing

Course Objectives:

The course aims to:

- Apply version control systems to efficiently manage code changes and collaboration.
- Create and manage containerized applications using Docker.
- Develop the skills to deploy, scale, and manage containerized applications using Kubernetes.
- Understand cloud-native security mechanisms, including identity management, access control, and network policies.
- Build and configure a full DevOps pipeline to automate deployment of applications.

Course Outcomes :

After completing the course, the students will be able to:

CO1**:** Use version control systems and automation tools to build and manage continuous integration pipelines.

CO2: Develop containerized applications using Docker.

CO3: Deploy and manage container orchestration using Kubernetes platforms.

CO4: Analyze cloud-native monitoring, logging, and security mechanisms using open-source logging tools.

CO5: Build a complete DevOps pipeline to deploy applications on cloud platforms using open-source tools.

Unit	Contents	Duration (Hrs)
1	**Introduction to Cloud and DevOps:** Basics of Cloud Computing, Introduction to DevOps: CI/CD Concepts, Lifecycle & Principles, Overview of Git, Docker, Jenkins, Kubernetes, Terraform, Ansible. Git Basics: Clone, Branch, Commit, Merge, Rebase, GitHub Basics: Repositories, Issues, Actions, Git Workflows: Feature branching, GitFlow, Fork & Pull Request. **Case Study:** Building a Modern Cloud-Native Application Using DevOps Principles	8
2	**SDLC Automation:** CICD Overview: CodeCommit-Overview, options, securing repository and branches, Triggers and notifications. CodeBuild-Overview: buildspec.yaml, Docker, Electronic Container Registry, Environment variables and ParameterStore, Artifacts and S3 Buckets, Events and Logging CodeDeploy-Overview: Application Deployment Groups, Deployment configurations, Hooks and Environment Variables, Rollbacks, Deploy to AWS Lambda CodePipeline-Overview: Adding CodeCommit, CodeDeploy and CodeBuild, Manual approval steps, Stage Actions, All Integrations **Case Study:** Automating the Software Development Lifecycle (SDLC) for a Web Application	8

3	**Configuration Management and Infrastructure as a Code (IaC):** Introduction to Configuration Management & IaC, Concepts of Configuration-as-Code, Why use IaC Tools, Change Control, Declarative JSON/YAML Templates, Creating Your First Playbook in YAML, Fundamentals of AWS CloudFormation, & Fundamentals of Terraform. **Case Study:** Implementing Configuration Management and Infrastructure as Code in a Cloud-Based Startup	8
4	**Containerization with Dockers:** Containers, Elastic Container Services, ECS Clusters, ECS Tasks Definitions, Docker Architecture & Components, Docker: Images, Containers, Volumes, Networks, Dockerfile & Image Building with Kubernetes. **Case Study:** Containerization and Orchestration for Scalable Application Deployment	8
5	**Security, Monitoring and Logging:** Cloud & Container Security, Identity Access Management, Observability in DevOps, CI/CD best practices & DevSecOps introduction, Logging: ELK Stack, CloudWatch, Prometheus + Grafana, Configure monitoring using Prometheus & Grafana (or CloudWatch). **Case Study:** Implementing Security & Observability in DevOps for Fintech Company	8
	Total Hours	**40**

Text Books

4. Mark Reed, "DevOps: The Ultimate Beginners Guide to Learn DevOps Step-By-Step" , Publishing Factory LLC ISBN: 978-1647710941.
5. Joseph Joyner, "DevOps For Beginners: DevOps Software Development Method Guide For Software Developers and IT Professionals", Kindle Edition, ASIN: B017E4HBXS.
6. Mikael Krief, "Learning DevOps", Packt Publishing Limited, ISBN: 9781838642730

Reference Books

1. Osama Mustafa, "A Complete Guide to DevOps with AWS" APRESS Springer link.
2. Yevgeniy Brikman, "Fundamentals of DevOps and Software Delivery", Orelly.

Online References

- **Microsoft Learning Devops Engineer:** https://learn.microsoft.com/en-us/training/career-paths/devops-engineer
- **AWS Skillbuilder:** https://skillbuilder.aws/search?refid=cd0e962f-7a6e-499a-b161-007a2362b83d&la=cta&cta=topbanner&searchText=devops-engineering-learning-plan&showRedirectNotFoundBanner=true
- **Course Era Devops:** https://www.coursera.org/specializations/packt-devops-complete-course
- **Ethans Cloud Computing Devops:** https://eict.ethans.co.in/cloud-computing-devops/

Development Resources

- **Programming Language**: YAML, JSON, TypeScript, Python and Java
- **IDE**: AWS, Google Cloud, Terraform, Kubernetes, Jenkins, Ansibe, GIT

Third Year B.Tech Computer Engineering

Semester-VI

Course Code: CE24PEC354B **Course Name: Distributed Systems**

Teaching Scheme (Hours/Week)					Examination Scheme						Credits			
L	P	T	OL	ODL	CIE	ETE	TW	OR	PR	TOTAL	L	P	T	TOTAL
3		-	-	-	40	60	-	-		100	3		-	3

Prerequisite: CE24PCC301:Operating systems

Course Objectives:

The course aims to:

- Summarize fundamental principles of distributed systems.
- Illustrate distributed system techniques -communication, and coordination.
- Explore process synchronization in distributed systems.
- Recognize various distinguishing characteristics of distributed systems viz. reliability, fault tolerance, scalability etc.
- Evaluate different distributed systems w.r.t shared memory

Course Outcomes:

After learning the course, the students will be able to:

CO1: Identify the key issues and challenges associated with distributed operating system.
CO2: Explore distributed computing techniques, synchronization and processes.
CO3: Analyze key distributed systems properties
CO4: Design distributed file systems and explore security in distributed systems
CO5: Apply the concepts of distributed shared memory to real world problems

Unit	Contents	Duration (Hrs.)
1	**Introduction to distributed Systems** Definition and goals, Hardware and Software concepts, Design issues, Network Operating Systems, True Distributed System and Time-sharing, Multiprocessor, Operating System, System Architectures.Case Study: Amoeba System Architecture	8
2	**Communication in Distributed System** Computer Network and Layered protocols, Message passing and related issues, synchronization, Client Server model & its implementation, remote procedure call and implementation issues Case Studies: Apache Kafka Distributed Event Streaming Platform, gRPC Open SourceRPC Framework, SUN RPC, DCE, RPC.	8
3	**Synchronization in distributed systems** Clock synchronization and related algorithms, mutual exclusion, Deadlock in distributed systems, Thrashing, Heterogeneous DSM, Resource Management (Load Balancing approach, Load Sharing approach). Case Study: Google Spanner.	8
4	**Processes and Processors in distributed systems** Threads, system model, processor allocation, scheduling in distributed systems: Load balancing and sharing approach, fault tolerance, Real time distributed systems, Process migration and related issues. Case Study :Distributed operating system Amoeba, Chorus	8

5	**Distributed File Systems & Security** Introduction, features & goal of distributed file system, file models, file accessing models, file sharing semantics, file caching scheme, file replication, fault tolerance, trends in distributed file system, Introduction of Security in Distributed OS, Overview of security techniques, features, Need, Access Control, Security Management Case Studies :Sun Network File system, DCE Distributed file service	8
	Total Hours	**40**

Text Books

1. P. K. Sinha, "Distributed Operating Systems: Concepts and Design", 1st ed., New Delhi, India: PHI Learning Pvt. Ltd., 1998. ISBN ISBN-10 8120313801, ISBN-13 978-8120313804.
2. G. F. Coulouris, J. Dollimore, and T. Kindberg, "Distributed Systems: Concepts and Design", 5th ed., Harlow, UK: Pearson (Addison-Wesley) ISBN: 9789332575226, 32575223 Edition: Fifth, 2017
3. A. S. Tanenbaum, "Distributed Operating Systems", illustrated / reprint ed., New Delhi, India: Pearson Education, 1995. ISBN-10 8177581791; ISBN-13 978-8177581799.

Reference Books

1. S. Mahajan and S. Shah, "Distributed Computing", 2nd revised ed., New Delhi / New York: Oxford University Press, 2013. ISBN-10: 0198093489; ISBN-13: 978-0198093480.
2. A. S. Tanenbaum and M. van Steen, "Distributed Systems: Principles and Paradigms", 2nd ed., Upper Saddle River, NJ: Pearson Prentice Hall, 2007. ISBN-10: 0-13-239227-5; ISBN-13: 978-0132392273.
3. H. Attiya and J. Welch, "Distributed Computing: Fundamentals, Simulations, and Advanced Topics", 2nd ed., Hoboken, NJ: John Wiley & Sons (Wiley-Interscience), Mar. 25, 2004. ISBN-10: 0471453242; ISBN-13: 978-0471453246

Online References

- NPTEL Course: Prof. Rajiv Misra, IIT Patna , ' Distributed Systems' https://onlinecourses.nptel.ac.in/noc21_cs87/preview
- NPTEL Course: Prof. Smruti Ranjan Sarangi,IIT Delhi, 'Advanced Distributed Systems' https://nptel.ac.in/courses/106102237
- NPTEL Course : Prof. Ananthanarayana V.S, IIT Madras, ' Distributed Computing Systems', https://nptel.ac.in/courses/106106107

Third Year B.Tech Computer Engineering														
Semester- VI														
Course Code: CE24PEC354C							**Course Name: High Performance Computing**							
Teaching Scheme (Hours/Week)					**Examination Scheme**						**Credits**			
L	**P**	**T**	**OL**	**ODL**	**CIE**	**ETE**	**TW**	**OR**	**PR**	**TOTAL**	**L**	**P**	**T**	**TOTAL**
3		**-**	**-**	**-**	**40**	**60**	**-**	**-**		**100**	**3**		**-**	**3**

Prerequisite: CE24PCC252-Principles of Programming Languages

Course Objectives:

The course aims to:

- To apply the fundamentals of parallel architectures and performance metrics.
- To develop parallel programs using Message Passing Interface for distributed memory systems.
- To apply Pthreads and Open Multi Processing for shared memory parallel programming.
- To analyze the architecture of the Graphics Processing Unit.
- To implement Compute Unified Device Architecture programs.

Course Outcomes:

After learning the course, the students will be able to

CO1: Analyze parallel computer architectures based on standard performance metrics.
CO2: Apply parallel algorithms using Message Passing Interface for distributed memory systems.
CO3: Develop High Performance Computing applications using Shared Memory Programming.
CO4: Evaluate Graphics Processing Unit based parallel algorithms for real world problems.
CO5: Design Compute Unified Device Architecture based High Performance Computing applications.

Unit	Contents	Duration (Hrs.)
1	**Introduction to Parallel Computing** The Von Neumann architecture,Processes, Multitasking,Threads, Modifications to the Von Neumann Model,Parallel Hardware, Parallel Software,Input and Output, Performance,Parallel Program Design,Writing and Running Parallel Programs. Case Study:Medical Image Processing with Parallel Computing	8
2	**Distributed and Shared Memory Programming** Introduction to Message Passing Interface(MPI), MPI Program syntax : MPI Init and MPI Finalize, Communicators, Single Program Multiple Data, MPI Send, MPI Recv, Distributed Memory Programming with MPI. Pthreads and its syntax, Producer Consumer Synchronization , Semaphores, Thread safety, Open Multi Processing (OpenMP), Shared Memory Programming with OpenMP. Case Study**:** Parallel Matrix Multiplication	8
3	**Parallel Communication** Preliminaries, Decomposition Techniques, Characteristics of Tasks and Interactions, Mapping Techniques for Load Balancing,Parallel Algorithm Models, One to All Broadcast, All to One Reduction, All to All Broadcast and Reduction, Collective Communication using MPI, Sources of Overhead in Parallel Programs, Performance Measures and Analysis. Case Study: Real Time Sensor Data Processing in a Smart Factory	8

4	**Graphics Processing Unit Architecture** Understanding Parallelism with Graphics Processing Unit (GPU) ,SIMT, GPU Architecture,Threads, Blocks, Grids, Warps, Scheduling ,Memory Handling, Shared Memory, Global Memory, Constant Memory and Texture Memory,Performance Evaluation of GPU Programs. Case Study: Generative Adversarial Network Training on Graphics Processing Unit	8
5	**Compute Unified Device Architecture** Heterogeneous Computing,Paradigm of Heterogeneous Computing, Compute Unified Device Architecture (CUDA): Programming Model, Managing Memory, Organizing Threads, Write and Launch a CUDA kernel, Handling Errors, Compilation and Execution, Applications of High Performance Computing. Case Study: Real Time Video Processing	8
	Total Hours	**40**

Text Books

1. Grama, A., Gupta, A., Karypis, G., & Kumar, V. ,2003,Introduction to Parallel Computing,2 nd edition, Addison Wesley,ISBN 0-201-64865-2.
2. Seyed H. Roosta, Parallel Processing and Parallel Algorithms: Theory and Computation, Springer-Verlag 2000 ,ISBN 978-1-4612-7048-5.
3. John Cheng, Max Grossman, and Ty McKercher, Professional CUDA C Programming, John Wiley & Sons, Inc., 2014 edition,ISBN: 978-1-118-73932-7.

Reference Books

1. K. Hwang, Scalable Parallel Computing,New York, NY, USA: McGraw-Hill, 1998.
2. G. S. Almasi and A. Gottlieb, Highly Parallel Computing, 2nd ed. Redwood City, CA, USA: Benjamin,Cummings, 1994.
3. J. Sanders and E. Kandrot, CUDA by Example: An Introduction to General-Purpose GPU Programming, Boston, MA, USA: Addison-Wesley Professional, 2010.
4. P. S. Pacheco, An Introduction to Parallel Programming,San Francisco, CA, USA: Morgan Kaufmann, 2005.
5. E. Rieffel and W. Polak,Quantum Computing: A Gentle Introduction, Cambridge, MA, USA: MIT Press, 2011.
6. A. D. Kshemkalyani and M. Singhal,Distributed Computing: Principles, Algorithms, and Systems,Cambridge, UK: Cambridge Univ. Press, 2011.

Online References

- NPTEL Course:High Performance Computing,IISc Bangalore,Prof. Mathew Jacob- https://nptel.ac.in/courses/106108055
- NPTEL Course: Parallel Computing,IIT Delhi,Dr. Subodh Kumar- https://nptel.ac.in/courses/106102114
- NPTEL Course: High Performance Computing,IIT Kanpur and IIT Madras- https://nptel.ac.in/courses/128106014
- NPTEL Course: GPU Architectures and Programming,IIT Kharagpur,Prof. Soumyajit Dey- https://nptel.ac.in/courses/106105220
- NPTEL Course: Introduction to parallel programming with OpenMP and MPI, IIT Delhi, Prof. Yogish Sabharwal - https://nptel.ac.in/courses/106102163

Third Year B.Tech Computer Engineering

Semester- VI

Course Code: CE24PEC354D **Course Name: Digital Forensic**

Teaching Scheme (Hours/Week)					Examination Scheme						Credits			
L	P	T	OL	ODL	CIE	ETE	TW	OR	PR	TOTAL	L	P	T	TOTAL
3		-	-	-	40	60	-	-	-	100	3	-	-	3

Prerequisite: CE24PEC304D-Information Security

Course Objectives:

The Course aims to

- Understand core principles and legal aspects of digital forensics and evidence.
- Apply structured investigation models to analyze cyber incidents.
- Develop skills in collecting, preserving, and validating digital evidence.
- Investigate cybercrimes involving intrusions, malware, and system breaches.
- To Perform network forensic analysis and apply techniques to real-world cases.

Course Outcomes:

After learning the course, the students will be able to:

CO1: Understand fundamental concepts of digital forensics.
CO2: Apply digital investigation process models and scientific methods.
CO3: Analyze collected digital evidence and system artifacts for forensic relevance.
CO4: Evaluate digital traces from crimes and intrusions to reconstruct events.
CO5: Create network forensic analyses and reports using network evidence and reconstruction techniques.

Unit	Contents	Duration (Hrs.)
1	**Introduction to Digital Forensics** Foundations of Digital Forensics:Digital Evidence, Awareness of Digital Evidence, Principles of Digital Forensics, Challenging Aspects of Digital Evidence, Cybertrail, Digital Forensics Research, Language of Computer Crime Investigation,Role of Computers in Crime Case Study: Employee Data Theft in a Company-A software company notices its confidential source code leaked online.	8
2	**Digital Investigations** Process Models, Scaffolding, Scientific Methods, Investigative Scenario: Security Breach,Handling a Digital Crime Scene: Guidelines, Fundamental Principles, Authorization, Preparation, Surveying, Preservation. Case Study: Financial Fraud via Compromised Laptop-A bank employee's laptop is compromised, and money is transferred illegally.	8
3	**Collecting Evidence** Basic concepts, Crime Scenes and Collecting Evidence, Documenting the Scene, Chain of Custody, Cloning, Live System versus Dead System, Hashing, Windows System Artifacts:Deleted Data,Hibernation File, Registry, Print Spooling, Recycle Bin, Metadata, Most Recently Used Case Study: Ransomware Attack on a Small Business.-All business files are encrypted; investigators must determine entry point.	8
4	**Apprehending Offenders** Violent Crime and Digital Evidence:Role of Computers in Violent Crime, Processing the Digital Crime Scene, Investigative Reconstruction, Computer	8

	Intrusions: Investigation, Forensic Preservation of Volatile Data, Post-Mortem Investigation, Investigation of Malicious Computer Programs, Investigative Reconstruction. Case Study: Malware Spread in University Network-A worm spreads across a university network, crashing several machines .	
5	**Network Forensics** Basics Concepts: History of Computer Networks, Technical Overview, Network Technologies, Connecting Networks Using Internet Protocols, Applying Forensic Science to Networks: Preparation and Authorization, Identification, Documentation, Collection and Preservation, Filtering and Data Reduction, Class/Individual Characteristics, Evidence Recovery, Reporting Results Case Study: Email Spoofing & Phishing Campaign-Business email compromise (BEC) attack scams employees into transferring money.	8
	Total Hours	**40**

Text Books

1. Casey, Eoghan. Digital Evidence and Computer Crime: Forensic Science, Computers and the Internet. 3rd Edition, Academic Press (Elsevier), 2011. ISBN: 978-0-12-374268-1.
2. Sammons, John. The Basics of Digital Forensics: The Primer for Getting Started in Digital Forensics. CRC Press, 2012. ISBN: 978-1-59749-661-2.

Reference Books

1. Jones, Keith J., Richard Bejtlich, and Curtis W. Rose. Real Digital Forensics: Computer Security and Incident Response. Addison-Wesley (Pearson Education), 2005. ISBN: 978-0-321-24069-9.
2. Sammes, Tony, & Brian Jenkinson. Forensic Computing: A Practitioner's Guide, 2nd ed., Springer, 2007. ISBN: 978-1-84628-397-0.
3. Brown, Christopher L.T. Computer Evidence: Collection & Preservation, Firewall Media, 1st ed. ISBN: 978-1584504054.

Online References

- NPTEL Course: Information Security -IV,IIT Madras,Prof. V. Kamakoti, Prof. M J Shankar Raman, Prof. Vasan-https://nptel.ac.in/courses/106106178
- NPTEL Course: Digital Forensics,Uttarakhand Open University, Haldwani,Dr. Jeetendra Pande- https://onlinecourses.swayam2.ac.in/nou22_cs05/preview

Third Year B.Tech Computer Engineering

Semester VI

Course Code :CE24PCC356 **Course Name: Artificial Intelligence Laboratory**

Teaching Scheme (Hours/Week)					Examination Scheme						Credits			
L	P	T	OL	ODL	CIE	ETE	TW	OR	PR	TOTAL	L	P	T	TOTAL
-	2	-	-	-	-	-	-	-	50	50	-	1	-	1

Prerequisite: AI24PCC151:Foundation of Artificial Intelligence,CE24PCC201: Data Structures

Course Objective:

The course aims to:

- Enable students to apply core AI problem-solving techniques, including search strategies, heuristics, and logic-based reasoning.
- Enable students to apply fuzzy logic and genetic algorithm for problem-solving

Course Outcomes :

After learning the course, the students will be able to:

CO1: Implement and analyze classical AI problem-solving techniques including search algorithms, game playing, and expert systems.

CO2: Design intelligent applications such as chatbots and rule-based systems for real-world decision-making tasks.

CO3: Implement fuzzy logic and genetic algorithm techniques for solving real-world problems.

Suggestive List of Experiments

Sr. No	Name of the Experiment	Duration (Hrs.)
1	Implement A star Algorithm for 8-Puzzle search problem.	4
2	Write a program to implement a Constraint Satisfaction Problem (CSP) by choosing one application such as N-Queens, Sudoku Solver, Exam Timetable Scheduling, or Hostel Room Allocation	4
3	Implement a two-player Tic-Tac-Toe game where one player is human and the other is an AI. The AI should employ the Minimax algorithm to determine its moves, ensuring it plays optimally.	4
4	Implement any one of the following Expert System I. Information management II.Hospitals and medical facilities III.Help desks management IV.Employee performance evaluation V.Stock market trading VI.Airline scheduling and cargo schedules	4
5	Develop an elementary rule-based chatbot, "Edu-Bot," designed for a customer interaction application in an online education platform. The Edu-Bot should be capable of handling basic customer inquiries related to course information, enrollment status, and frequently asked questions, providing friendly and helpful responses.	2
6	Implement Union, Intersection, Complement and Difference operations on fuzzy sets. Also create fuzzy relation by Cartesian product of any two fuzzy sets and perform max-min composition on any two fuzzy relations	2

7	Implement genetic algorithm for benchmark function (eg. Square, Rosenbrock function etc) Initialize the population from the Standard Normal Distribution. Evaluate the fitness of all its individuals. Then you will do multiple generations of a genetic algorithm. A generation consists of applying selection, crossover, mutation, and replacement. Use: • Tournament selection without replacement with tournament size s • One point crossover with probability Pc • bit-flip mutation with probability Pm • use full replacement strategy	4
	Total Hours	**24**
	Mini Projects **Note:** **1. The mini project should be implemented in a group of 3–4 students.** **2. Students may select any one of the suggested mini project titles provided.** **3. Alternatively, students may propose a mini project title of their own choice, aligned with the syllabus topics.**	
1	Design and implement an intelligent route-finding system that determines the optimal path between a source and a destination in a given map. The system should use an informed search technique (A* algorithm) with suitable heuristic functions to minimize travel distance or cost. The project should allow comparison of different heuristics and evaluate performance based on time complexity and path optimality.	
2	Design and implement a two-player Nim game where a human competes against an intelligent computer opponent. The computer uses the Minimax algorithm to analyze all possible game states and select optimal moves that guarantee a win whenever possible.	6
3	Design and implement an intelligent decision-making system for the Pac-Man game using the Minimax algorithm. In the game environment, Pac-Man acts as the MAX player, whose objective is to maximize score and survival, while the ghosts act as MIN players, whose objective is to minimize Pac-Man's score by trapping or catching him.	
4	Develop an automated exam timetable scheduling system using constraint satisfaction techniques. The system must allocate exams to time slots and rooms while satisfying constraints such as avoiding student conflicts, room capacity limits, and faculty availability.	
5	Design and implement a Fuzzy Expert System (FES) to predict the risk level of heart disease in a patient based on multiple input parameters. The system should mimic human reasoning by handling approximate and ambiguous data	
6	Design a system that uses a Genetic Algorithm to find an optimal path from the start to the finish in a complex maze. Candidate solutions (chromosomes) represent sequences of moves, and the fitness function evaluates distance traveled, collisions with walls, and path efficiency.	
	Total Hours	**30**
	Guidelines for Laboratory Conduction: 1. All the assignments on all concepts are mandatory. 2. Assignments on all concepts covered on Linked list, stack , Queue, Tree and Graph are mandatory and should be implemented on coding platforms such as HackerRank, CodeChef,GitHub, Leetcode. 3. Operating System recommended: - 64-bit Open-source Linux or its derivative.	

	4. Programming tools recommended: - G++/GCC,Java/ Python, Eclipse/Geany/Jupyter Notebook **Guidelines for Students:** 1. The laboratory assignments are to be submitted by students in the form of a journal. 2. Journal consists of Vision Mission of Institute, Department, certificate, table of contents and handwritten write-up of each assignment (Title, Objectives, Problem Statement, Outcomes,Date of Completion, Assessment grade/marks and assessor's sign, Theory- Concept,algorithm, time complexity, sample input and expected output, conclusion). **Guidelines for Laboratory Assessment:** 1. Continuous assessment of laboratory work is done based on overall performance and Laboratory performance of students. 2. Each Laboratory assignment assessment should assign grade/marks based on rubrics with appropriate weightage. 3. Suggested rubrics for overall assessment as well as each Laboratory assignment assessment include timely completion, performance, innovation, efficiency, punctuality and neatness.	

Third Year B.Tech Computer Engineering

Semester- VI

Course Code: CE24PCC357 **Course Name: Design and Analysis of Algorithms Laboratory**

Teaching Scheme (Hours/Week)					Examination Scheme						Credits			
L	P	T	OL	ODL	CIE	ETE	TW	OR	PR	TOTAL	L	P	T	TOTAL
-	2	-	-	-	-	-	-	-	50	50	-	1	-	1

Prerequisite: CE24PCC201:Data Structures, CE24PCC205:Data Structures Lab

Course Objectives:

The course aims to:

- Analyze the asymptotic performance of algorithms in terms of space and time
- Apply algorithmic strategies to find optimal solution

Course Outcomes:

After learning the course, the students will be able to:

CO1: Analyze performance of an algorithm in terms of time and space

CO2: Implement an algorithm using algorithmic designing strategies

Suggestive List of Experiments

Sr. No.	Name of the Experiment	Duration (Hrs.)
1	**Music Theory:** Implement a program to analyze musical compositions and identify patterns related to the Fibonacci sequence, such as the arrangement of notes and the structure of musical pieces. Generate Fibonacci sequence which maps each fibonacci number to a musical note. This should be based on specific musical scales and it should create simple musical patterns. **Leetcode Problem Statement:** Given an array of non-negative integers representing money in houses, you cannot rob adjacent houses. Write the program to return the maximum amount you can rob.	2
2	**Genomic Sequence:** Write a program to sort genomic sequences to identify similarities and differences between organisms. For implementing this , use the merge sort algorithm. Use strings to represent genomic sequences. It must recursively divide the sequence vector and merge the sorted sub sequences. You can use string comparison operators to sort sequences lexicographically. **Leetcode Problem Statement:** For each element, count how many numbers smaller than it appear to its right.	4
3	**Word Search:** Develop a program to find a specific word or phrase within a document using divide and conquer approach. If the target word is found, the function returns its index. If the target word is less than the word at mid, the search is narrowed down to the left half. If the target word is greater than the word at mid, the search is narrowed down to the right half. If the target word is not found after the loop, the function returns -1. **CodeChef Problem Statement:** Given an array A, count the number of pairs (i, j) such that: i < j and A[i] > A[j]	2

4	**Cargo Ship:** Given a cargo ship with a limited weight capacity, select a set of containers to maximize the total value of the cargo. Implement the fractional greedy algorithm to find the optimal solution. Define structure to represent items with their weight and value. Sort items based on their value-to-weight ratio. Iterate through the sorted items and add them to the knapsack until the capacity is reached. If the current item doesn't fit entirely, add a fraction of it to maximize the value.**CodeChef Problem Statement:** Given fuel stations at different distances with different prices, find the minimum cost to reach the destination when fuel can be bought in fractions.(Fuel Station Problem)	4
5	**Job Scheduling on Virtual Machines:** Develop a program for cloud computing platforms. Programs should efficiently allocate virtual machines to incoming jobs using deadlines. Implement the greedy algorithm to schedule jobs with deadlines and profits to maximize the total profit. Create a boolean array slot to keep track of free slots. For each job, iterate backward from its deadline to find the earliest free slot. If a free slot is found, allocate the job to that slot and mark it as occupied. **Leetcode Problem Statement:** Each event has a start day and end day. Write a program to attend at most one event per day. Maximize number of events attended.	4
6	**Manufacturing Company:** Consider a manufacturing company that needs to efficiently package products in a series of operations. The company uses multiple machines in a sequence to package products, and each machine has a different processing time for each item. The company needs to find the optimal order in which to use the machines so that the total time to process all the products is minimized.This scenario is analogous to the Matrix Chain Multiplication problem, where we have a sequence of matrices (machines), and the goal is to find the optimal order of multiplication to minimize the total computational cost. Implement the dynamic programming approach to solve the matrix chain multiplication problem efficiently.Minimize the number of scalar multiplications by determining the optimal matrix multiplication order.Analyze the time and space complexity of the algorithm. **CodeChef Problem Statement:** You are developing a 3D graphics engine. You have a sequence of N transformation matrices that need to be multiplied to apply transformations to objects. Each matrix has different dimensions.Find the order of multiplications that minimizes the total number of scalar multiplications.	4
7	**Marketing Company:** Consider a marketing company that needs to distribute resources across multiple campaigns. The company has a fixed number of resources and needs to decide how to allocate those resources across k different marketing campaigns. The objective is to calculate the number of possible ways to choose k campaigns from the available n campaigns.This is analogous to the binomial coefficient problem, which calculates the number of ways to	4

	choose k elements from a set of n elements, represented as C(n, k) or n choose k.Implement the dynamic programming approach to efficiently calculate the binomial coefficient C(n, k).Analyze the time and space complexity of the Algorithm. **CodeChef Problem Statement:** A bakery is entering a contest where they bake batches of 10 cookies each. Each cookie has a 0.6 probability of being "perfectly baked." The bakery wants to know the probability that at least 7 cookies in a batch are perfectly baked. Write a program that, given the number of batches N, outputs the expected number of batches where at least 7 cookies are perfect.	
	Total Hours	**24**
	Mini Projects **Note:** **4. The Mini Project should be implemented in a group of 3-4 students.** **5. Students can select any one from the following mini project titles.** **6. Students can select a mini project title of their own choice.**	
1	**Drone Application:** Given an 8x8 grid (like a chessboard), the goal is to place 8 drones such that no two drones are in the same row, column, or diagonal, following the rules of a queen's movement in chess. Find all possible configurations for placing the drones on the board. Implement a backtracking algorithm to find all possible valid configurations of drone placements.For each configuration, ensure that no drones threaten each other by checking row, column, and diagonal constraints.After implementing the solution, analyze the time and space complexity of the algorithm.	
2	**Dynamic Programming Based Traffic Signal Optimization System:** Traffic congestion is a major issue in urban areas, leading to increased travel time and fuel consumption. This mini project focuses on designing a Dynamic Programming based traffic signal optimization system that dynamically adjusts signal timings based on traffic density. The system should model intersections as stages and signal timings as decisions, and should minimize total waiting time. The project should analyze scalability and practical feasibility of DP-based solutions in traffic management.	
3	**Minimum Coin Change Problem Solver:** Develop a dynamic programming solution to determine the minimum number of coins required to make a given amount using a set of coin denominations. The system should handle multiple test cases and display the selected coins.	
4	**Sudoku Solver Using Backtracking:** Create a Sudoku solver that fills a partially completed 9×9 grid by satisfying row, column, and sub-grid constraints. Implement the solution using backtracking with constraint checking. Analyze how pruning reduces the search space and improves efficien cy.	

5	**Financial Planning and Budget Optimization System:** Personal and organizational financial planning requires careful budgeting to ensure optimal utilization of funds. This mini project proposes a Dynamic Programming based financial planning system that allocates a fixed budget across multiple expense categories to maximize utility or savings. The system should handle multiple constraints and demonstrate how dynamic programming simplifies complex multi-stage decision problems.	**6**
6	**Automated Timetable Scheduling System:** Develop an automated timetable scheduling system for a college to assign classes, faculty, and classrooms without any conflicts. The system must ensure that no faculty member, classroom, or class batch is scheduled for more than one session at the same time. Implement a backtracking algorithm to explore all possible valid timetable combinations while satisfying given constraints. The application should generate feasible schedules and eliminate conflicting assignments through constraint checking. Analyze the time and space complexity of the proposed solution.	
7	**Resource Constrained Project Scheduling System:** Large projects often involve multiple interdependent tasks competing for limited resources. This mini project proposes a Branch and Bound based project scheduling system that determines the optimal task execution order to minimize project completion time while respecting resource constraints.Partial schedules should be evaluated using bounding functions to eliminate infeasible or suboptimal solutions. The project should demonstrate the applicability of Branch and Bound in multi-constraint optimization problems.	
8	**Network Design Optimization System:** Designing cost-effective networks requires selecting an optimal subset of links while maintaining connectivity and performance constraints. This mini project focuses on implementing a Branch and Bound based network design system that minimizes total network cost subject to connectivity requirements. The system should use bounds based on minimum spanning tree approximations to prune suboptimal network configurations. The project should compare results with heuristic-based solutions.	
9	**Text Similarity and Sequence Analysis Tool:** Text comparison and sequence matching are essential in applications such as plagiarism detection, DNA sequence analysis, and document version control. This mini project aims to develop a Dynamic Programming based tool that computes the similarity between two input sequences using algorithms such as Longest Common Subsequence (LCS) and Edit Distance. The system should accept variable-length inputs, generate the DP table, and output both similarity scores and reconstructed sequences. The project must include a comparative analysis of recursive, greedy, and DP-based solutions, highlighting the importance of overlapping subproblems and optimal substructure.	
	Total Hours	**30**

Guidelines for Laboratory Conduction:

1. Assignments on all concepts are mandatory.
2. Assignments on all concepts covered on Linked list, tree and graph are mandatory and

should be implemented on coding platforms such as HackerRank, CodeChef.
3. Use of open-source software is to be encouraged.
4. Operating System recommended: - 64-bit Open-source Linux or its derivative.
5. Programming tools recommended: - C++/GCC/Python, Eclipse.

Guidelines for Students:

1. The laboratory assignments are to be submitted by students in the form of a journal.
2. Journal consists of prologue, certificate, table of contents and handwritten write-up of each assignment (Title, Objectives, Problem Statement, Outcomes, Date of Completion, Assessment grade/marks and assessor's sign, Theory- Concept, algorithm, time complexity, sample input and expected output, conclusion).

Guidelines for Laboratory /TW Assessment:

1. Continuous assessment of laboratory work is done based on overall performance and Laboratory performance of students.
2. Each Laboratory assignment assessment should assign grade/marks based on parameters with appropriate weightage.
3. Suggested parameters for overall assessment as well as each Laboratory assignment assessment include- timely completion, performance, innovation, efficiency, punctuality and neatness.

Third Year B.Tech Computer Engineering

Semester- VI

Course Code: CE24PCC358 **Course Name: Engineering Design and Innovation-II**

Teaching Scheme (Hours/Week)					Examination Scheme						Credits			
L	P	T	OL	ODL	CIE	ETE	TW	OR	PR	TOTAL	L	P	T	TOTAL
-	2	-	-	-	-	-	25	-	-	25	-	1	-	1

Prerequisite: CE24CEP208- Project Based Learning

Course Objectives:

The course aims to:

- Develop critical thinking and problem solving ability by exploring and proposing solutions to realistic/social problems.
- Evaluate alternative approaches, and justify the use of selected tools and methods,
- Emphasize learning activities that are long-term, inter-disciplinary and student-centric.
- Engage students in rich and authentic learning experiences.
- Provide every student the opportunity to get involved either individually or as a group so as to develop team skills and learn professionalism.
- Develop an ecosystem to promote entrepreneurship and research culture among the students.

Course Outcomes :

After completing the course, the students will be able to:

CO1: Create innovative solutions for real-world social challenges by applying critical thinking and structured problem-solving techniques.

CO2: Synthesize knowledge across multiple disciplines to take ownership of long-term, student-led investigative projects.

CO3:Design original research or entrepreneurial ventures that contribute to a growing ecosystem of innovation and discovery.

Preamble :

Project-Centric Learning (PCL) transforms the abstract rigor of **Design and Analysis of Algorithms (DAA)**, **Theory of Computation (TOC)**, **Artificial Intelligence (AI)**, and **High-Performance Computing (HPC)** from isolated academic hurdles into a unified toolkit for global innovation. In this model, the classroom functions as a high-level research lab where you don't just study the mathematical limits of a Turing machine or the complexity of a sorting algorithm in a vacuum. Instead, you might find yourself applying the formal logic of **TOC** to verify the security of a smart contract or utilizing **DAA** to engineer hyper-efficient routing protocols for autonomous logistics in disaster relief.

As your project grows in complexity, you naturally graduate from basic coding to architectural design, reaching for **AI** to build predictive intelligence and **HPC** to scale those models across distributed clusters. Rather than simply memorizing syntax, you take the lead in solving "big" technical questions—like how to minimize latency in a global healthcare network or how to parallelize massive datasets for climate modeling. With faculty acting as expert mentors rather than lecturers, PCL ensures you move past the "how-to" of setup and into the "why" of system design. By the end of the journey, you aren't just a programmer; you are a computational architect capable of turning the most complex theories of computer science into the digital infrastructure of the future.

These guidelines act as a flexible roadmap rather than a strict set of rules. We've built the Engineering Design labs like a ladder, helping you step up from the basics of Computer Networks—where you learn

how data moves—to the vast world of Cloud Computing, where you learn to manage global scale and storage. Think of this as a journey from understanding industry communication standards to mastering 5G, IoT, and Cyber Security. While the framework is here to guide you, we encourage you to use your creativity to build interdisciplinary solutions that push beyond the syllabus.

Group Structure:

- There should be a team/group of 4-5 students.
- A supervisor/mentor teacher assigned to individual groups.
- It is useful to group students of different abilities and nationalities together.

Selection of Project/Problem:

- Students must focus to initiate the task/idea .The idea inception and consideration shall be from following areas as a real world problem:
- Health Care, Agriculture, Defense, Education, Smart City, Smart Energy, Swaccha Bharat Abhiyan, Environment, Women Safety.
- This is the sample list to start with. Faculty and students are free to include other areas which meet the society requirements at large.
- The model begins with the identifying of a problem, often growing out of a question or "wondering". This formulated problem then stands as the starting point for learning. Students design and analyze the problem/project within an articulated disciplinary subject frame/domain.
- A problem can be theoretical, practical, social, technical, symbolic, cultural, and/or scientific and grows out of students' wandering within different disciplines and professional environments. A chosen problem has to be exemplary. The problem may involve an interdisciplinary approach in both the analysis and solving phases.
- By example, a problem needs to refer back to a particular practical, scientific, social and/or technical domain. The problem should stand as one specific example or manifestation of more general learning outcomes related to knowledge and/or modes of inquiry.

Teacher's Role in PCL :

- The teacher is not the source of solutions, rather he will act as the facilitator and mentor.
- To utilize the principles of problem solving, critical thinking and metacognitive skills of the students.
- To make the group aware about time management.
- Commitment to devote the time to solve student's technical problems and interested in
- helping students to empower them better.

Student's Role in PCL:

- Students must have the ability to initiate the task/idea .They should not be mere imitators.
- They must learn to think.
- Students working in PCL must be responsible for their own learning.
- Students must quickly learn how to manage their own learning, instead of passively receiving instruction.
- Students in PCL are actively constructing their knowledge and understanding of the situation in groups.
- Students in PCL are expected to work in groups.
- They have to develop interpersonal and group process skills, such as effective listening or
- coping creatively with conflicts.

Sample Case Studies(For reference only):

1. Intelligent Traffic Signal Control System(AI)
2. AI-Powered Student Performance Prediction(AI)
3. Password Validation System (TOC)
4. Compiler Syntax Checking (TOC)
5. Virus Pattern Detection (TOC)
6. Emergency Ambulance Route Optimization(DAA)
7. Cloud Task Scheduling (DAA)
8. Weather Forecast Simulation Using Parallel Computing(HPC)

9. Genome Sequence Analysis Using HPC

Text Books: (As per IEEE format)

1. Terry Barrett., "A new model of problem based learning" , All Ireland Society for higher education (AISHE). ISBN:978-0-9935254-6-9; 2017
2. Mahnazmoallem, woei hung and Nada Dabbagh, "Problem Based Learning", Wiley Publishers. 2019.
3. Robert Robart Capraro, Mary Margaret Capraro, "Stem Project based learning and integrated science, Technology, Engineering and mathematics approach"

Reference Books: (As per IEEE format)

- De Graaff E, Kolmos A., red.: Management of change: Implementation of problem-based and project-based learning in engineering. Rotterdam: Sense Publishers. 2007.
- Project management core textbook, second edition, Indian Edition , by Gopalan.
- The Art of Agile Development. By James Shore & Shane Warden.

Third Year B.Tech Computer Engineering

Semester-VI

Course Code: CE24VSE360 **Course Name: Server-Side Programming**

Teaching Scheme (Hours/Week)					Examination Scheme						Credits			
L	P	T	OL	ODL	CIE	ETE	TW	OR	PR	TOTAL	L	P	T	TOTAL
-	4	-	-	-	-	-	50	-		50	-	2	-	2

Prerequisite: CE24VSE257: Web Technology, CE24PCC252: Principles of Programming Languages, CE24PCC251: Database Management Systems, CE24PCC255: Database Management Systems Laboratory

Course Objectives:

The course aims to:

- Differentiate between client-side and server-side scripting for building effective web applications.
- Apply HTTP protocol and request–response mechanisms used in server-side communication.
- Develop backend applications using Spring Boot with database connectivity and RESTful services.
- Apply session and cookie management techniques for secure and stateful user interactions.
- Deploy server-side applications on cloud platforms and local servers.

Course Outcomes:

After learning the course, the students will be able to:

CO1: Analyze server-side and client-side scripting to build robust web applications.
CO2: Apply HTTP protocol and request/response mechanisms to implement web communication
CO3: Develop and manage backend applications using the Spring Boot framework with database connectivity and RESTful web services.
CO4: Implement session and cookie management for secure, stateful user interactions in server-side applications.
CO5: Deploy full-featured server-side applications on cloud platforms and local servers.

Sr. No.	Suggested List of Experiments	Duration (Hrs.)
1	Develop a basic Spring Boot application that accepts a GET request and returns a welcome message.	4
2	Create a Spring Boot REST API that handles GET, POST, PUT, and DELETE methods for a 'Student' resource. Define a Student class and implement CRUD operations using a controller. Test all endpoints using Postman.	4
3	Design an API endpoint that accepts path variables and query parameters and returns dynamic JSON responses. Create endpoints like /student/{id} and /students?course=Java that responds with corresponding data.	4
4	Develop a simple HTML form for feedback submission and handle the form data in a Spring Boot controller. Accept name, email, and message fields from the form. Process the data in a POST endpoint and return a thank-you message as a response.	4
5	Connect a Spring Boot application to a MySQL database and perform insert and fetch operations using Spring Data JPA.Create a Student entity, repository interface, and service layer. Add endpoints to add new students and retrieve them from the database.	4

6	Implement a RESTful API for managing book records in a library system. Create APIs for adding, updating, retrieving, and deleting books. Use JSON for data interchange and test using Postman.	8
7	Implement a basic user login system using sessions in Spring Boot. Accept username and password via a POST request. If valid, create a session and store the username. Display a welcome message for the logged-in user in a separate endpoint.	8
8	Create an endpoint /visit that uses cookies to count and display the number of visits per user. Use cookies to track the number of times a user has visited a specific endpoint.	4
9	Deploy a Spring Boot REST API to a cloud platform such as Render, Railway, or Heroku. Upload your Spring Boot application, configure environment variables, and test your API using the public URL.	8
	Total Hours	**48**

Mini-Project Topics

Design and implement a full-stack or backend-only web application using server-side programming concepts.

Note:

1. **The Mini Project should be implemented in a group of 3-4 students.**
2. **Students can select any one from the following mini project titles.**
3. **Students can select a mini project title of their own choice.**

P1	**Blood Donation and Availability System** Develop a Spring Boot-based backend system that manages blood donor registrations and tracks the availability of different blood types in real-time. Donors can register via a form, and hospitals can check blood availability through REST API endpoints. Admin and hospital login is implemented through session tracking, while cookies store recent search history or preferred blood groups. The project also includes database integration and live deployment for hospital staff to access via browsers or mobile apps.	**12**
P2	**Alumni Connect Platform (Backend)** Design and implement the server-side of an alumni connect platform where alumni and current students can interact. Alumni can register, post mentoring opportunities, and manage their profiles, while students can search for mentors and request interactions. The system handles user roles, session-based logins, and uses cookies to store recent mentor searches. REST APIs are used for all interactions, with full CRUD functionality and backend deployment for use within college systems.	
P3	**Digital Certificate Issuance System** Build a backend system that generates and manages digital certificates for various events and workshops. Event organizers can input event and participant details through APIs, and the system generates downloadable certificate links. Admin authentication is maintained via sessions, while students use token-based access. The system tracks certificate downloads using cookies and stores data in a MySQL database. It is deployed online to allow remote access and verification.	
P4	**Laundry Pickup and Delivery Booking System** Create a backend service for managing laundry pickup and delivery operations for a campus or society. Students or residents can book pickups, track their orders, and view delivery status. CRUD operations are implemented for customers, orders, and delivery agents. Sessions are used for login, and cookies	

	store frequently used laundry preferences. All endpoints follow REST standards, and the application is deployed for use by laundry service staff and customers.	
P5	**Bus Seat Reservation System** Develop a backend system that handles bus reservations for students traveling via college buses. The system allows route creation, seat booking, and cancellation using RESTful APIs. Session management is used for admin and student logins, while cookies track the most frequently traveled routes. Data is stored in a relational database, and the project includes cloud deployment for students to book or modify their bus seats remotely.	
P6	**Online Workshop Registration and Attendance System** Build a backend platform that manages online workshop registrations and marks attendance. Users can register for events, receive confirmations, and get attendance certificates. RESTful APIs allow admins to update sessions, track attendance, and generate certificates. Sessions manage user logins and cookie stores frequently join workshops. MySQL is used to persist data, and the system is deployed to support institutional workshops.	
P7	**Department Asset Management System** Build a backend system to keep track of IT assets such as laptops, projectors, and printers issued to staff and departments. Assets can be added, issued, returned, and serviced through RESTful endpoints. Admins authenticate via session management and cookies keep track of commonly issued items. CRUD operations are implemented with Spring Boot and data is stored in MySQL. The application is deployed for use by department Technical staff.	
P8	**College Event Pass Generator and Validator** Design a backend application for managing event pass generation for college fests. Students can register for events, download digital passes, and organizers can validate these passes at the gate. Sessions manage user and admin login, cookies store frequent registrations, and RESTful APIs provide the event and pass data. Spring Boot with MySQL is used, and the system is deployed on a secure institutional server.	

Text Books

1. Craig Walls, Spring in Action, Manning Publications, 6th Edition, 2022 ISBN-10: 1617297577, ISBN-13: 9781617297571
2. Mark Heckler, Spring Boot: Up and Running, O'Reilly Media, 1st Edition, 2021 ISBN-10: 1492076981, ISBN-13: 9781492076988
3. Felipe Gutierrez, Pro Spring Boot 3, Apress, 3rd Edition, 2023 ISBN-10: 148429147X, ISBN-13: 9781484291473

Reference Books

1. Koushik Kothagal, "Beginning Spring Boot 3: Build Java-Based Cloud and Web Applications", Apress, 1st Edition, 2023 ISBN-10: 148429231X, ISBN-13: 9781484292319
2. Herbert Schildt, "Java: The Complete Reference", McGraw-Hill Education, 12th Edition, 2021 ISBN-10: 1260440230 ISBN-13: 9781260440232
3. Bryan Basham, Kathy Sierra, Bert Bates, "Head First Servlets and JSP", O'Reilly Media, 2nd Edition, 2008, ISBN-10: 0596516681, ISBN-13: 9780596516680

Online References

- **Spring Boot:** https://spring.io/projects/spring-boot
- **HTTP:** https://developer.mozilla.org/en-US/docs/Web/HTTP
- **My SQL:** https://dev.mysql.com/doc/refman/8.0/en/

Guidelines for Laboratory Conduction:

1. Assignments on all concepts are mandatory.
2. Use of open-source software is encouraged.
3. Operating System recommended: - 64-bit Open-source Linux or its derivative.
4. Programming tools recommended: - Eclipse/Geany, JDK version installed compatible with Spring Boot 3.x (Java 17 or later), Web & API Tools -Postman (for REST API testing)

Guidelines for Students:

1. The laboratory assignments are to be submitted by students in the form of a journal.
2. Journal consists of prologue, certificate, table of contents and handwritten write-up of each assignment (Title, Objectives, Problem Statement, Outcomes, Date of Completion, Assessment grade/marks and assessor's sign, Theory- Concept, algorithm, time complexity, sample input and expected output, conclusion).

Guidelines for Laboratory /TW Assessment:

- Continuous assessment of laboratory work is done based on overall performance and Laboratory performance of students.
- Each Laboratory assignment assessment should be assigned grade/marks based on parameters with appropriate weightage.
- Suggested parameters for overall assessment as well as each Laboratory assignment assessment include- timely completion, performance, innovation, efficiency, punctuality and neatness.

Third Year B.Tech Computer Engineering

Semester-VI

Course Code: CE24AEC361 Course Name: Reasoning and Aptitude Development- II

Teaching Scheme (Hours/Week)					Examination Scheme						Credits					
L	P	T	OL	ODL	CIE	ETE	TW	OR	PR	TOTAL	L	P	T	OL	ODL	TOTAL
-	-	-	-	1	-	-	25	-	-	25	-	-	-	-	1	1

Prerequisite: Basic Mathematical Concepts

Course Objectives:

The Course aims to

- Develop numerical computation skills using averages, alligation, time–speed–distance, and time–work concepts for solving real-life quantitative problems.
- Apply mathematical reasoning techniques to special cases involving trains, boats & streams, circular motion, and work-based systems.
- Interpret and analyze data presented in tables, pie charts, bar graphs, and line graphs using efficient calculation and approximation techniques.
- Strengthen logical and analytical reasoning abilities through coding–decoding, series analysis, and structured critical reasoning problems.
- Understand and apply principles of permutations, combinations, and probability to solve counting and chance-based problems accurately.
- Build exam-oriented problem-solving strategies including time management, question selection, elimination methods, and decision-making under constraints.

Course Outcomes:

After learning the course, the students will be able to:

CO1: Solve average and allegation problems involving weighted averages and mixture applications in profit, interest, and speed scenarios.

CO2: Analyze and compute time–speed–distance and time–work problems, including special cases such as trains, boats & streams, circular tracks, and pipes & cisterns.

CO3: Accurately interpret data and assess data sufficiency, applying logical checks and efficient calculation strategies to reach valid conclusions.

CO4: Identify patterns and logical relationships in coding–decoding, number series, and alphanumeric sequences to derive correct solutions.

CO5: Evaluate arguments and deductions critically, distinguishing between strong and weak reasoning in assumptions, conclusions, and cause–effect relationships.

CO6: Apply strategic problem-solving and decision-making techniques to optimize accuracy and time efficiency in competitive and placement examinations.

Unit	Contents - Foundations & Numerical Proficiency	Duration (Hrs.)
1	**Averages & Alligations** 1. Arithmetic Mean of Series (Consecutive numbers) 2. Concept of Weighted Average 3. Deviation Method (Assumed Mean) 4. Alligation Rule (Cross method for mixing two groups) 5. Application of Alligation in Profit, Interest, and Speed	2
2	**Time, Speed & Distance (TSD)**	2

	1. Basic Formula and Unit Conversions (km/hr to m/s) 2. Average Speed (Total Distance / Total Time) 3. Harmonic Mean for equal distance intervals 4. Late/Early arrival logic (Constant Distance) 5. Relative Speed (Same direction vs. Opposite direction)	
3	**TSD Special Cases** 1. Trains: Crossing poles, platforms, and other trains 2. Boats & Streams: Upstream and Downstream 3. Circular Tracks: First meeting point and meeting at starting point	2
4	**Time & Work** ● Efficiency and Days (Inverse Relationship) ● LCM Method for calculating total work ● Alternate day working models ● Negative Work (Pipes and Cisterns / Leaks)	2
5	**Data Interpretation & Sufficiency** 1. Calculation Hacks for DI (Approximation/Elimination) 2. Pie Charts (Degree to Percentage conversion) 3. Bar Graphs and Line Graphs (Growth rates) 4. Table-based DI (Missing data) 5. Data Sufficiency Logic (Check Statement 1, 2, then Both/Either/Neither)	2
6	**Coding-Decoding & Series** 1. Letter-to-Letter and Letter-to-Number coding 2. Number Series (Difference, Double difference, Square/Cube patterns) 3. Alphanumeric Series and Symbol sequences	2
7	**Critical Reasoning I (Arguments)** 1. Identifying Premises, Conclusions, and Intermediate Conclusions 2. Assumption Identification (Negation Test) 3. Strengthening and Weakening Arguments	2
8	**Critical Reasoning II (Deductions)** ● Statement & Conclusion ● Statement & Course of Action ● Cause and Effect relationships ● Assertion and Reason ● Strong vs. Weak Arguments	2
9.	**Permutation, Combination & Probability** ● Fundamental Principles of Counting (Addition vs. Multiplication) ● Arrangements (Word formation, Circular Permutation) ● Selection (Combinations) ● Probability basics (Coins, Dice, Playing Cards) ● Probability of "At least" and "At most" cases	2
10.	**Strategy & Decision Making** ● Question Selection (Identifying "Low-Hanging Fruit") ● Elimination Techniques for Puzzles ● Time-boxing for different sections ● Decision Making (Criterial-based selection sets) ● Full-length Mock Review ●	2
	Total Hours	**20**

Reference Books

Quant Basics	R.S. Aggarwal	Concept Clarity
Speed Math	M. Tyra	Time Management
Logic/Puzzles	K. Kundan / R.S. Aggarwal	Exam-specific Patterns
Critical Thinking	M.K. Pandey	Logical Derivations
DI / Advanced	Arun Sharma	Complex Data & CAT

Assessment

Assessment will be done by Course Teacher. MCQ Test can be conducted based on the syllabus.