Marathwada Mitramandal's College of Engineering

Karvenagar, Pune- 52

An Autonomous Institute Affiliated to SPPU

Curriculum for

Third Year B. Tech Artificial Intelligence and Data Science

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

Emerge globally as a leading center for Artificial Intelligence and Data Science education, contributing to sustainable development.

Department Mission

M1: Equip students with advanced Artificial Intelligence knowledge for industry-ready skills.

M2: Cultivate a culture of research and entrepreneurship to drive innovation and problem-solving.

M3: Inculcate social and professional values to develop sustainable AI solutions.

M4: Foster a lifelong learning that supports upskilling and global engagement.

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)

Graduates of the AI & DS program will be able to:

PEO1: Apply, analyze, and evaluate computing skills for problem-solving.

PEO2: Engage in continuous learning through professional development, certifications, emerging technologies in Artificial Intelligence and Data Science.

PEO3: Demonstrate ethical behaviour, effective communication and leadership skills in diverse environments.

PEO4: Build successful careers in industry or entrepreneurship, demonstrating competence and adaptability in a dynamic global environment.

Program Specific Outcomes (PSOs)

Upon successful completion of the program, graduates will be able to:

PSO1: Demonstrate the ability to understand, apply, and develop fundamental computer programs and AI and Data Science solutions using key concepts such as data analytics, machine learning, deep learning and natural language processing.

PSO2: Develop and implement AI and data-driven models by critically analyzing complex problems, designing effective solutions, and evaluating their impact across diverse real-world applications.

PSO3: Utilize emerging AI and Data Science technologies to innovate, promote entrepreneurial ventures, and develop successful careers in industry and research, while embracing lifelong learning and higher education.

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

Third Year B.Tech Tech Artificial Intelligence and Data Science 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
AI24PCC301	Artificial Intelligence & Machine Learning	PCC	3		-	-	40	60	-		-	100	3		-	-	3
AI24PCC302	Computer Network & Cloud Computing	PCC	3		-	-	40	60	-		-	100	3		-	-	3
AI24PCC303	Automata Theory and Applications	PCC	3	-	-	-	40	60	-	-	-	100	3	-	-	-	3
AI24PEC304	A. AR & VR B. Industry driven-High Performance Computing C. Edge Computing D. Game Development:From Design to Prototype	PEC	3	-	-	-	40	60	-	-	-	100	3	-	-	-	3
AI24MDM305	*(Multidisiplinary Minor theory)	MDM	3	-	-	-	40	60		-	-	100	3	-	-	-	3
AI24PCC306	Artificial Intelligence & Machine Learning Laboratory	PCC	-	2	-	-	-	-	-	50	-	50	-	1	-	-	1
AI24PCC307	Computer Network & Cloud Computing Laboratory	PCC	-	2	-	-	-	-	-	50	-	50	-	1	-	-	1
AI24PCC308	Engineering Design and Innovation-I	PCC	-	4	-	-	-	-	25	-	-	25	-	2	-	-	2
AI24MDM309	*(Multidisiplinary Minor Lab)	MDM	-	2	-	-	-	-	25			25		1	-	-	1
AI24OEL310	*(OEL Theory)	OEL	-	-	-	2	-	-	50	-	-	50	-	-	-	2	2
AI24AEC311	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 Booklet for OE and MDM**

SEMESTER V

Third Year B.Tech Artificial Intelligence & Data Science														
Semester-V														
Course Code: AI24PCC301					**Course Name: Artificial Intelligence & 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: Statistical Analysis

Course Objectives:

- To understand and apply intelligent agents, search techniques, games, and constraint satisfaction in artificial intelligence.
- To represent knowledge and perform logical reasoning using logical formalisms and inference mechanisms in AI systems.
- To develop the ability to use various regression methods to model relationships and generate predictions across real-world problems
- To implement classification techniques and clustering approaches on data.
- To design and apply ensemble learning and reinforcement learning models.

Course Outcomes:

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

CO1: Apply intelligent agent, search games and CSP techniques to solve AI problems
CO2: Represent knowledge and perform logical reasoning in AI systems
CO3: Design ML models to make predictions by using linear, non-linear and logistic regression techniques
CO4: Implement classification and clustering models for unlabeled data
CO5: Apply ensemble learning and reinforcement learning and its algorithms for different applications

Unit	Contents	Duration (Hrs.)
1	**Foundations of Artificial Intelligence:** Intelligent agents, Environment types, Uninformed and Informed search, Greedy Best First Search, A*, Heuristic design and admissibility and Search performance comparison. **Adversarial Search and Games:** Heuristic Alpha–Beta Tree Search, Monte Carlo Tree Search, Stochastic Games, Partially Observable Games, Constraint Satisfaction Problems (CSP), Constraint Propagation	8
Case Study	**Artificial Intelligence at Amazon: Building Intelligent Decision-Making Systems for Global Logistics**	
2	**Knowledge Representation and Logical Reasoning** Knowledge-Based Agents, The Wumpus World, Types of knowledge, Knowledge representation techniques, Propositional Logic and First Order Logic (FOL) - Syntax and semantics, Using Prepositional Logic and FOL, Inference Mechanisms	8
Case Study	**Artificial Intelligence in Healthcare: Building a Knowledge-Based Clinical Decision Support System**	
3	**Introduction to Machine Learning and Regression** ML vs AI vs Data Science, Supervised, Unsupervised, Semi-supervised and Reinforcement Learnings, PCA and LDA, Difference between Regression and Correlation, Types of Regression, Regression Techniques, Evaluation Metrics	8
Case Study	**Analyze and predict Big Mart sales using selected predictor variables by applying Lasso, Ridge, and Elastic Net regression techniques**	
4	**Classification and Clustering** Balanced and Imbalanced Classification Problems, Binary Classification, Multiclass Classification, Classification Algorithms, Hierarchical clustering algorithms,	8

	Centroid-based clustering algorithms, Density-based clustering algorithms (DBSCAN), Distribution-based clustering algorithms	
Case Study	**To segment customers into distinct groups based on similarities using clustering techniques.**	
5	**Ensemble Learning and Reinforcement Learning** Types of ensemble methods, Basic and Advanced Ensemble Learning Techniques, Random Forest Ensemble. Types of Reinforcement, Elements of Reinforcement Learning. Markov's Decision Process, Bellman equation Q Learning.	8
Case Study	**Use ensemble learning to detect anomalies in real-time patient health data, and apply reinforcement learning to recommend adaptive treatment or alert strategies based on patient responses.**	
	Total Hours	40

Text Books

1. Stuart Russell and Peter Norvig, "Artificial Intelligence: A Modern Approach", Third edition, Pearson, 2021, ISBN: 9781292401171, 1292401176
2. Peter Flach: "Machine Learning: The Art and Science of Algorithms that Make Sense of Data", Cambridge University Press, Edition 2012

Reference Books

1. Nilsson Nils J , "Artificial Intelligence: A new Synthesis", Morgan Kaufmann Publishers Inc. San Francisco, CA, ISBN: 978-1-55-860467-4
2. Dr. LavikaGoel, "Artificial Intelligence: Concepts and Applications", Wiley publication, ISBN: 9788126519934
3. Shalev-Shwartz, Shai, and Shai Ben-David, "Understanding machine learning: From theory to algorithms", Cambridge university press, 2014

Online References

1. https://cs.calvin.edu/courses/cs/344/kvlinden/resources/AIMA-3rd-edition.pdf
2. https://www.cin.ufpe.br/~tfl2/artificial-intelligence-modern-approach.9780131038059.25368.pdf
3. http://aima.cs.berkeley.edu/

Third Year B.Tech Artificial Intelligence & Data Science

Semester-V

Course Code: AI24PCC302 **Course Name: Computer Network & 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

Prerequisites: Programming Basics, Operating Systems, Database Management Systems

Course Objectives:

- To understand the importance and learn the fundamentals of computer networks.
- To demonstrate the use of addressing, routing, and socket programming to solve real-world problems in the context of today's internet.
- To introduce cloud computing concepts, architectures, and service models.
- To analyze cloud virtualization and storage.
- To evaluate cloud networking, cloud security and ML services.

Course Outcomes:

CO1: Select network architecture, topology and essential components to design computer Networks.
CO2: Implement the client server application using socket.
CO3: Illustrate the cloud computing models, architectures, and platforms.
CO4: Investigate and apply the resource virtualization and storage technique for a given business case.
CO5: Develop secure, scalable, and cost-efficient cloud solutions.

Unit I	Introduction to Computer Network	Duration (Hrs.)
1	**Introduction, Physical:** Introduction to Computer networks, network components & Network Devices, Types of networks, Network topologies. OSI and TCP/IP reference model, Transmission media, Bandwidth, Data rate, Switching techniques, multiplexing, Line Coding. **Data link layer:** Services and framing, Error detection and Correction techniques, flow control mechanisms, MAC addressing, Multiple access protocols. **Network layer:** Addressing, Subnetting, Routing and routing algorithms, Supporting protocols- ARP and ICMP. congestion control .	8
Case Study	1.Identify the components to establish a Small LAN. 2.Identify MAC addresses,Observe Ethernet frames. Study HDLC protocol 3. A mid-sized enterprise has multiple departments such as Administration, HR, Finance, IT, and Production. Design network layer solution using CIDR and subnetting 4.Network Analysis Tools : Packet Tracer,Wireshark	
Unit II	**Transport and Application Layers**	
2	**Transport laye**r: Process-to-Process Communication, UDP and TCP protocols, port numbers and sockets, reliable data transfer, and flow and congestion control,socket programming. **Application layer:** client–server and peer-to-peer models DNS, HTTP and HTTPS, FTP, SMTP, POP and IMAP. RESTful web services and APIs.	8
Case Study	1. Email Communication Reliability. 2. Design a reliable file transfer application using TCP sockets and measure transfer time and throughput. 3. Design a quiz application where questions are broadcast via UDP and answers are sent back to the server.	

Unit III	**Cloud Computing Fundamentals**	
3.	Distributed and grid computing. Introduction to cloud computing, Cloud service models: IaaS, PaaS and SaaS, IaaS and deployment models: public, private, hybrid and community clouds. cloud architecture, Cloud computing vs. Cluster computing vs. Grid computing, Cloud providers, benefits and limitations, Regions and availability zones, billing in cloud.	8
Case Study	1. Students list at least 5 applications they use daily and identify: Type of cloud service (SaaS / PaaS / IaaS) and Cloud provider (if known) 2. Divide students into groups. Each group represents: A hospital, A college, A bank, A startup. They decide: Which cloud deployment model to use and why it is suitable. 3. Distinguish cloud service models.	
Unit IV	**Cloud Virtualization and Storage**	
4.	Virtualization concepts and benefits, Types of Virtualizations: Server, Storage, Network, Virtual Machines and Containers, Hypervisors: Type I and Type II,VM provisioning and resource management, Cloud services- EC2, Elastic load balancer, AWS Elastic Beanstalk, AWS Lambda, ECS/EKS Cloud File Systems, Cloud Storage as a Service, Connecting to VM using SSH/RDP Auto-scaling and load balancing. Amazon S3, Amazon S3 Glacier, AWS Block Storage - EBS, AWS - EFS, RDS/ Aurora/ Elasti Cache- DynamoDB	8
Case Study	• A startup wants to deploy: A web server, A database server, A testing environment but has only one physical machine. Students must use virtualization to simulate this cloud-like setup. • A college wants to store: Student documents, Video lectures,Application data for a web portal. The storage solution must be scalable, secure, cost-effective, and accessible from anywhere. Students must design a cloud storage solution for this scenario.	
Unit V	**Cloud Private Network and ML**	
5.	Amazon VPC, AWS Snowball, AWS DataSync, AWS Data Pipeline, AWS Database Migration Service.DevOps- CI CD, Infrastructure Provisioning and Server Con, AWS Cloud Formation, AWS Elastic Beanstalk, AWS Ops Works, AWS Cloud Shell Cloud security fundamentals: Data security, Network security, Privacy and compliance issues, ML- Sage Maker, Kendra, Applications of Cloud Computing in AI & Data Science	8
Case Study	• A software company wants to deploy: A public web application, A private database, Internal management servers. The company needs a secure private network in the cloud so that: Users can access the web app, The database remains private, Internal systems are protected. Students must design a cloud private network solution. • A startup wants to build a system that can predict whether a student will pass or fail based on: Attendance, Study hours, Internal marks. The startup chooses cloud-based ML to avoid buying expensive hardware. Students must design this cloud-based ML solution.	
	Total Hours	**40**

Text Books

1. James F. Kurose, and Keith W. Ross," A Top-Down Approach," 4th edition, Publisher: Addison-Wesley ISBN: 0-321-49770-8
2. Behrouz A. Forouzan, "Data Communication and Networking", 4th edition, Tata McGraw Hill

3. Andrew S. Tanenbaum, "Computer Networks" ,5th Edition, Pearson Education
4. Andreas Wittig & Michael Wittig, "*Amazon Web Services in Action*" – Comprehensive guide to AWS services with real-world examples
5. Quentin Docter, Emmett Dulaney& Timothy L. Warner, "*AWS Certified Cloud Practitioner Complete Study Guide*"

Reference Books

1. Holger Karl and Andreas Willig,"Protocols and Architectures for Wireless Sensor Network", Wiley, ISBN: 0-470-09510-5
2. C. Siva Ram Murthy and B. S. Manoj, "Ad Hoc Wireless Networks: Architectures and Protocols", Prentice Hall, 2004
3. *AWS Lambda in Action* by DaniloPoccia – Focuses on serverless computing with AWS Lambda and event-driven architectures.
4. *Serverless Architectures on AWS* (2nd Ed.) – Teaches designing and building serverless apps, a key AWS skill.

Online References

1. www.nptelvideos.in https://www.my-mooc.com/en/categorie/computer-networking
2. AWS Certified Cloud Practitioner

Third Year B. Tech Artificial Intelligence and Data Science														
Semester V														
Course Code: AI24PCC303										**Course Name: Automata Theory and Applications**				
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**

Pre-Requisites: Discrete Mathematics

Course Objectives:

- Understand basic models of computation and formal languages.
- Design simple automata for recognizing patterns.
- Apply grammars to structured input analysis.
- Understand computation using abstract machines.
- Recognize the capabilities and limits of algorithmic problem-solving.

Course Outcomes:

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

CO1: Apply concepts of formal languages and automata.
CO2: Design finite automata for simple pattern-matching problems.
CO3: Apply grammars and pushdown automata for structured inputs.
CO4: Analyse problem-solving using Turing Machines.
CO5: Evaluate decidable and undecidable problems.

Unit	Contents	Duration (Hrs.)
1	**Foundations of Automata and Formal Languages** Introduction to Automata Theory, Alphabets, Strings, and Languages, Operations on Languages, Language Classification (Chomsky Hierarchy), Applications of Formal Languages. Suggested Applications to be discussed: Counting valid strings over an alphabet, Checking if a string belongs to a given language, Basic token separation in text, Simple pattern rules for data validation	8
Case Study	A software company is designing an input validation module for a web application. The module uses formal languages and automata to validate user inputs such as IDs, codes, and commands.	
2	**Finite Automata and Regular Languages** Deterministic Finite Automata (DFA), Non-Deterministic Finite Automata (NFA), ε-NFA and Conversions, DFA Minimization, Regular Languages. Suggested Applications to be discussed: Binary string validation (e.g., strings ending with 01), Password format checking, Traffic light state simulation. Lexical Analysis (Compiler Design - Use of finite automata for recognizing tokens such as identifiers, keywords, and operators).	8
Case Study	1.A company is developing a control and validation system that uses finite automata to: • Validate binary data • Check password formats 2. Simulate traffic light behavior Model these tasks using DFA, NFA, and ε-NFA.	
3	**Regular Expressions and Context-Free Grammars** Regular Expressions (RE), Conversion between RE and Automata, Limitation of suitability of REs in handling recursive or nested structures, Context-Free Grammars (CFG), Parse Trees, Ambiguous Grammars	8

	Suggested Applications to be discussed: Searching patterns using regular expressions, Input format validation (email, phone number), Arithmetic expression grammar, Simple sentence structure checking, SQL Query Tree Generation and Parsing.	
Case Study	**1)** An e-commerce company is building an automated system to process customer orders. The system must: • Validate product codes • Recognize order commands • Check the structure of order expressions **2)** Regular Expression based case study of arithmetic expression grammar design.	
4	**Pushdown Automata and Context-Free Languages** Pushdown Automata (PDA), Acceptance by Final State and Empty Stack, Equivalence of PDA and CFG, Simplification of Grammars. Suggested Applications to be discussed: Checking balanced parentheses, Matching opening and closing tags, Expression validation using stack, Nested structure recognition.	8
Case Study	A software company is developing a code editor that automatically checks whether source code is syntactically well-formed. The source contains Balanced Parentheses Checking (e.g. [and]), Matching Opening and Closing Tags (e.g. <item><price></price></item>), Expression Validation Using Stack (e.g. (id + id) * id).	
5	**Turing Machines and Decidability** Turing Machines and Variants, Church–Turing Thesis, Recursive and Recursively Enumerable Languages, Decidable and Undecidable Problems, Halting Problem Suggested Applications to be discussed: Unary addition using Turing Machine, Palindrome checking, String reversal, Demonstration of non-terminating programs..	8
Case Study	A software company is developing a program analysis tool to: • Simulate basic computations • Detect infinite loops • Classify problems as decidable or undecidable The tool uses Turing Machines (TM) as its underlying model of computation.	
	Total Hours	**40**

Text Books

1. Hopcroft J, Motwani R, Ullman, Addison-Wesley, "Introduction to Automata Theory, Languages and Computation", Second Edition, ISBN 81-7808-347-7.
2. Michael Sipser, "Introduction to Theory of Computation", Third Edition, Course Technology, ISBN 10: 053494728X.

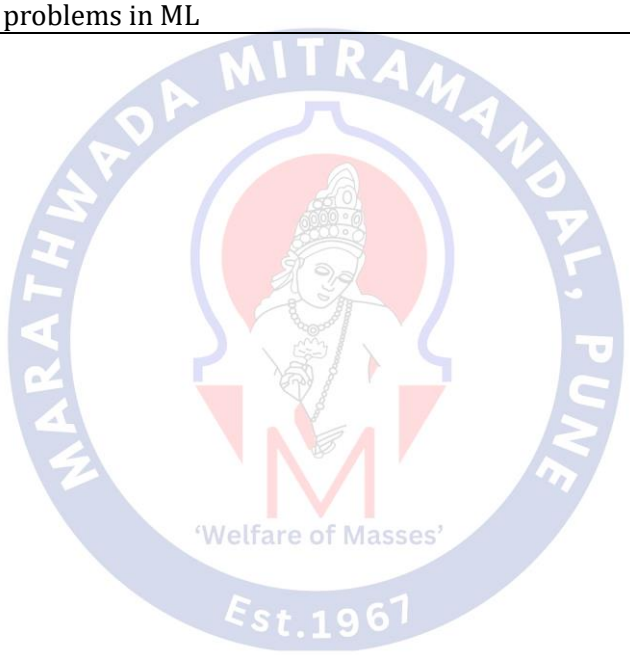
Reference Book

1. Mishra K.L.P, N. Chandrasekaran, "Theory of Computer Science: Automata, Languages and Computation", Third Edition, PHI, ISBN 978-8120329683.
2. John C. Martin," Introduction to Languages and The Theory of Computation", Fourth Edition, McGraw Hill,ISBN 978-0-07-319146-1.
3. Vivek Kulkarni "Theory of Computation", Oxford University Press, 2013, ISBN-13 978-0198084587
4. Daniel I.A. Cohen, "Introduction to Computer Theory", 2nd Edition, John Willy & Sons Inc., Sons Inc., 2000

Online References

1. https://nptel.ac.in/courses/106104148
2. https://www.inf.ed.ac.uk/teaching/courses/nlu/assets/reading/Gurney_et_al.pdf
3. https://nptel.ac.in/courses/117105101

Suggested Tools & Languages
• Python (Automata simulation) • JFLAP (Visualization and modeling) • NLTK (Grammar and parsing) • Regex libraries • Graph-based automata simulators (e.g. automata verse)
Optional Suggested Experiments / Mini Projects
• Design a DFA for sentiment keyword detection • Implement regex-based data pre-processing pipeline • Implement arithmetic operations (+,-,*,/,%, parenthesis (), identifiers, literals) with operator precedence and associativity-based parser by defining formal language and write a Context Free Grammar and implement using YACC • SQL Select Query parsing. Order of evaluation of query and optimization. • CFG-based chatbot sentence parser • Turing Machine simulator for basic arithmetic • Case study on NP-hard problems in ML



Third Year B.Tech Artificial Intelligence & Data Science														
Semester-V														
Course Code: AI24PEC304A					**Course Name: AR&VR(Augumented Reality & Virtual Reality)**									
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: Programming Fundamentals, Data Structures & Algorithms (Basics), Computer Graphics (Introductory Level), Mathematics for Engineering, Artificial Intelligence Basics.

Course Objectives:

- To introduce the fundamental concepts, evolution, and applications of Augmented Reality (AR), Virtual Reality (VR), and Mixed Reality (MR).
- To understand the architecture, components, and working principles of AR/VR systems including displays, tracking, and input devices.
- To develop knowledge of 3D graphics, transformations, rendering, lighting, shading, and physics for immersive environments.
- To explore the role of AI, Machine Learning, Computer Vision, and Data Science in enhancing AR/VR experiences.
- To provide hands-on exposure to AR/VR development platforms and SDKs while addressing UX design, ergonomics, and ethical issues.

Course Outcomes:

CO1: Explain the concepts, differences, challenges, and future scope of AR, VR, and MR technologies.
CO2: Analyze AR/VR system components, display technologies, tracking methods, and interaction techniques.
CO3: Apply 3D graphics and modeling principles to design and render immersive AR/VR environments.
CO4: Develop basic AR/VR applications using platforms such as Unity, Unreal Engine, AR Core, ARKit, and Vuforia.
CO5: Evaluate AR/VR applications based on user experience, performance, ergonomics, and ethical and social considerations.

Unit	Contents	Duration (Hrs.)
1	**Introduction to AR & VR** Definition of AR, VR, MR (Mixed Reality), History and evolution of AR/VR, Applications of AR/VR (Education, Gaming, Healthcare, Industry, Defense), AR vs VR vs MR, Challenges and future scope, 3D transformations & rendering, Lighting, shading, physics UX principles for XR	8
Case Study	Identify and classify AR, VR, and MR devices based on display type, level of immersion, and application domain. Compare AR vs VR vs MR in terms of hardware requirements, user interaction, and use cases. Analyze UX principles for XR by reviewing an existing AR/VR application.	
2	**Virtual Reality** VR system components, VR display systems (HMD, CAVE, desktop VR), Tracking systems (optical, magnetic, inertial), VR input devices (gloves, controllers, sensors), Stereoscopic vision and depth perception, VR environment design, Interaction techniques, Performance optimization	8
Case Study	Identify the components of a VR system including HMDs, tracking systems, and input devices. Observe and compare VR display systems such as HMD-based VR, CAVE, and desktop VR.Study tracking techniques (optical, magnetic, inertial) used in VR systems.	

3	**Augmented Reality** AR system architecture, Marker-based and marker-less AR, Registration and tracking techniques, Spatial mapping and SLAM, AR display technologies (mobile AR, smart glasses), Interaction techniques in AR/VR, User experience (UX) design principles, AR Core, ARKit, Vuforia, SLAM & tracking, Mobile AR applications	8
Case Study	1.Identify the architecture of an AR system and its core components. 2.Compare marker-based and marker-less AR techniques with respect to tracking accuracy and performance. 3.Study SLAM and spatial mapping techniques used in AR Core and ARKit.	
4	**AI & Data Science in AR/VR &3D Graphics & Modeling for AR/VR** Computer vision for AR, ML-based gesture & eye tracking, XR analytics. Basics of 3D graphics, Coordinate systems and transformations, 3D object Modeling, Textures, lighting, shading, Animation basics for AR/VR environments.	8
Case Study	• Identify the role of computer vision in AR applications such as object detection and tracking. • Study ML-based gesture and eye-tracking techniques used in XR systems. • Analyze XR analytics data to understand user interaction and performance metrics.	
5	**AR/VR Development Platforms** Unity 3D fundamentals, Unreal Engine basics, ARCore and ARKit overview, Vuforia SDK, Scene creation and asset integration, Motion sickness and ergonomics, Gesture and voice interaction Ethical and social issues in AR/VR	8
Case Study	• A company plans to deploy an AR-based product catalog for customers while keeping internal 3D design assets and analytics private. The AR app must run on mobile devices, support image tracking, and ensure data security. • A manufacturing company wants to train workers using VR simulations instead of physical equipment to reduce cost and risk.	
	Total Hours	40

List of Activities

Note: Submission of these activities is mandatory for evaluation.

1. Create a simple 3D scene in Unity containing at least five primitive objects (Cube, Sphere, Capsule, Cylinder, and Plane). Apply different materials and colors, position the objects appropriately, add a directional light, and configure the Main Camera to view the scene. Enter Play Mode to verify that the scene renders correctly and navigate the scene using the Scene and Game views. Observe the differences between Edit Mode and Play Mode. *(Save the Unity project on external storage or cloud.)*
2. Extend the Unity scene created in Practical 1 by attaching built-in components (such as Rigidbody and Box Collider) to the existing primitive objects. Create a C# script and attach it as a custom component to one of the objects. Modify the object's properties (such as position, rotation, scale, or color) using the script and observe the changes by running the project in Play Mode. Experiment with Inspector-exposed variables to understand how scripts interact with GameObjects through components.
3. Create a Unity application that incorporates a user interface consisting of buttons, text, and images. Design a simple menu with options to display object information, enable or disable a 3D

object, and reset the scene to its initial state. Test the functionality by interacting with the UI elements during Play Mode.

4. Design a simple obstacle scene where a rolling sphere navigates through obstacles using keyboard input. Apply Rigidbody physics and create an animated object (such as a rotating fan or opening door) using Unity Animator or a simple script.
5. Develop a marker-based AR application using Vuforia Engine. Configure an Image Target and display a 3D model when the marker is detected. Ensure the model appears correctly aligned with the target and remains visible while the marker is in view.
6. Develop an Augmented Reality application using Vuforia Engine that recognizes multiple Image Targets. Display a different 3D model for each target and implement interactive UI controls that allow users to rotate, scale, or reset the displayed model. Ensure the correct model is displayed when the corresponding Image Target is detected.
7. Configure a basic Virtual Reality (VR) scene in Unity using the XR Interaction Toolkit. Create a simple environment containing multiple primitive 3D objects and make them interactable using the XR Grab Interactable component. Enable users to pick up, move, place, and throw objects within the virtual environment, allowing the objects to interact naturally using Unity's physics system.
8. Develop a Virtual Reality application using the XR Interaction Toolkit that contains multiple interactable objects with different behaviors. Create designated interaction zones where users can place objects to trigger actions such as changing object colors, displaying information, or activating simple animations. Verify that object interactions function correctly in the VR environment.

Text Books

1. Virtual Reality, Steven M. LaValle, Cambridge University Press, 2017.
2. Augmented Reality, Mark Billinghurst, Adrian Clark, Gun Lee, Foundations and Trends in Human–Computer Interaction, 2015.
3. Learning Virtual Reality: Developing Immersive Experiences and Applications, Tony Parisi, O'Reilly Media, 2015.

Reference Books

1. Augmented Reality: Principles and Practice, Dietrich Schmalstieg& Tobias Höllerer Addison-Wesley, 2016.
2. Virtual Reality Technology, Burdea& Philippe Coiffet, Wiley-IEEE Press, 2003 / 2014 Edition.
3. Computer Graphics: Theory and Practice, Ken Perlin& Holly Rushmeier (Editors), 3rd Edition, Pearson, 201.
4. Understanding Augmented Reality: Concepts and Applications, Alan B. Craig, Morgan Kaufmann, 2013.

Online References

1. Introduction to Augmented Reality and ARCore (Google) – Beginner-friendly AR fundamentals and ARCore tools.Introduction to AR &ARCore course (Class Central)
2. Handheld AR App Development with Unity – Learn mobile AR using Unity's AR Foundation (ARKit&ARCore).Unity AR Foundation course (Class Central)
3. WIZAPE AR Development Tools & Techniques – Free online modules covering AR basics, SDKs, and tools.AR Development Tools (WIZAPE)

Third Year B.Tech Artificial Intelligence & Data Science

Semester-V

Course Code: AI24PEC304B **Course Name: Industry Driven High Performance Computing**

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

Prerequisite: Operating System, Fundamentals of Computer Architecture

Course Objectives:

- Understand fundamental concepts, models, and levels of parallel processing.
- Analyze parallel architectures, operating systems, and core HPC design issues.
- Apply parallel programming techniques using OpenMP, MPI, Pthreads, CUDA, and related tools.
- Evaluate performance, scalability, and limitations of parallel algorithms and HPC systems.

Explore advanced HPC technologies including accelerators, cloud platforms, power-aware design, and future computing trends.

Course Outcomes:

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

CO1: Explain parallel processing concepts, models, and architectures used in high performance computing.
CO2: Analyze HPC system design issues including scheduling, synchronization, load balancing, and resource management.
CO3: Develop parallel programs using shared and distributed memory programming models and standard HPC tools.
CO4: Assess the performance, scalability, and efficiency of parallel algorithms and HPC systems using benchmarking techniques.
CO5: Apply advanced HPC technologies such as accelerators, cloud-based platforms, and power-aware designs to real-world computing problems.

Unit	Contents	Duration (Hrs.)
1	**Parallel Processing Fundamentals & Architectures** Parallel processing concepts, levels of parallelism, data flow models, demand-driven computation, superscalar architectures, multi-core architectures, multi-threaded architectures, server architectures, cloud-based architectures	8
Case Study	**Multi-Core Processor Performance Comparison** • Compare execution of a matrix multiplication program on single-core vs multi-core CPU • Analyze speedup using superscalar vs multi-threaded architecture concepts • Demonstrate task-level vs data-level parallelism using Python multiprocessing	

	Tools: Python, multiprocessing, NumPy, Google Colab	
2	**Design Issues & System Software for HPC** Fundamental design issues in HPC, load balancing, scheduling techniques, synchronization mechanisms, resource management, operating systems for scalable HPC, process management, thread management, memory management, scalable storage systems, RAID architectures, SSD cache, SAS, SAN.	8
Case Study	**Load Balancing & Resource Management in a Cluster** • Simulate job scheduling and load balancing across multiple worker nodes • Compare static vs dynamic scheduling for uneven workloads • Observe process/thread/memory management using system monitoring tools **Tools:** Python, Slurm (or a simple custom scheduler script), htop/top, Linux VM	
3	**Parallel Programming Models & Environments** Parallel languages and programming environments, shared memory programming model, distributed memory programming model, Pthreads, OpenMP, MPI, Java for parallel programming, Cilk, introduction to accelerator programming, GPU programming overview, CUDA programming basics.	8
Case Study	**Parallel Image Processing using MPI, OpenMP & CUDA** • Sequential vs multi-threaded (OpenMP) vs multi-process (MPI) image filtering • GPU-accelerated filtering using CUDA • Compare programming complexity and performance across models **Tools:** OpenMP, MPI4Py, CUDA (CuPy/Numba), OpenCV, Google Colab	
4	**Performance, Benchmarking & HPC Platforms** Performance analysis of parallel algorithms, speedup and efficiency, scalability, Amdahl's law, Gustafson's law, fundamental limitations in HPC, bandwidth limitations, latency, latency hiding techniques, benchmarking of HPC systems, scientific applications, engineering applications, commercial workloads, cluster computing, grid computing, cloud-based HPC	8
Case Study	**Speedup & Scalability Analysis of a Parallel Algorithm** • Benchmark a sorting/matrix algorithm on varying thread/node counts • Calculate speedup and efficiency using Amdahl's Law and Gustafson's Law • Identify latency/bandwidth bottlenecks in cluster communication **Tools:** Python, MPI4Py, matplotlib (for speedup graphs), Google Colab	
5	**Advanced HPC Technologies & Future Trends** HPC as a service, infrastructure as a service, platform as a service, computation as a service, economic models of HPC, accelerated high performance computing-based HPC systems, FPGA-based HPC systems, Xeon Phi, Cell Broadband Engine, power-aware HPC design, energy-efficient computing, communication and interconnect design, processor and memory design, power management techniques, peta-scale computing, big data processing, optics in HPC, quantum computing **HPC for AI and Data Science Applications** • HPC in machine learning and deep learning	8

	• Parallel data processing frameworks (MapReduce concept) • Scalable training of ML models • Case studies: HPC in healthcare, finance, weather prediction Introduction to cloud-based HPC	
Case Study	**Cloud-Based HPC for Scalable ML Model Training** • Train a deep learning model using distributed data parallelism • Deploy and scale training jobs on a cloud GPU instance (HPC as a Service) • Compare training time/cost across CPU vs GPU vs multi-GPU cloud setup **Tools:** PyTorch/TensorFlow, Google Colab (GPU), AWS/GCP free-tier HPC instance.	
	Total Hours	**40**

List of Activities

Note: Submission of these activities is mandatory for evaluation.

1. Develop and execute a multithreaded program using Pthreads to perform a computational task (e.g.,vector addition or matrix multiplication) and analyze speedup compared to serial execution.
2. Implement a shared-memory parallel program using OpenMP directives for loop-level parallelism and evaluate performance improvement by varying the number of threads.
3. Design and execute an MPI program for data-parallel computation (e.g., parallel sum, matrix multiplication) and study communication overhead and scalability.
4. Develop a CUDA program for parallel vector or matrix operations and compare GPU execution performance with CPU-based parallel implementations.
5. Analyze the performance of parallel programs using metrics such as execution time, speedup, efficiency, and scalability under different workloads and system configurations.
6. Load Balancing and Task Scheduling Simulation
   Design and implement a program simulating static and dynamic load balancing techniques (e.g., round-robin, work-stealing) across multiple worker threads/processes handling an uneven task queue. Compare completion time and CPU utilization for each scheduling strategy.
   Tools: Python/C with threading or MPI, custom scheduler logic.
7. Parallel MapReduce Implementation for Big Data Processing
   Implement a MapReduce-based program (e.g., word count, or aggregation on a large dataset) using Hadoop or a manual multiprocessing-based simulation, and analyze how execution time scales with increasing data size and number of mapper/reducer nodes.
   Tools: Hadoop (pseudo-distributed mode) or PySpark, Python multiprocessing.
8. Memory Access and Cache Performance Analysis in Parallel Programs
   Develop a parallel program (using OpenMP or Pthreads) that performs matrix operations with different memory access patterns (row-major vs column-major traversal) and study the impact of cache locality, false sharing, and memory bandwidth on overall performance.
   Tools: OpenMP/Pthreads (C), perf or Valgrind/Cachegrind for profiling

Text Books

1. 1 AnanthGrama, Anshul Gupta, George Karypis, and Vipin Kumar; Introduction to parallel computing; second edition., Addison-Wesley, 2003, ISBN: 0201648652
2. David Kirk, Wen-meiHwuCUDA: Programming Massively Parallel Processors: A Hands-On Approach. © ELSEVIER Inc.

3. Jason Sanders and Edward KandrotCUDA by Example: An Introduction to General-Purpose GPU Programming"

Reference Books

1. 1. Hwang and Briggs,"Computer Architecture and Parallel Processing", Tata McGraw Hill Publication ISBN 13: 9780070315563.
2. John Cheng, Max Grossman, Ty McKercherProfessional CUDA C Programming,
3. CUDA C PROGRAMMING GUIDE by NVIDIA

Online References

1. www.nptelvideos.in
2. http://developer.nvidia.com/

Third Year B.Tech Artificial Intelligence & Data Science														
Semester-V														
Course Code: AI24PEC304C							**Course Name: Edge 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: Programming Fundamentals, Data Structures & Algorithms, Computer Networks, Operating Systems, IoT Fundamentals.

Course Objectives:

- Understand edge computing architecture and its role in AI systems
- Study low-latency and scalable edge–cloud solutions
- Understand techniques for optimizing AI models at the edge
- Gain hands-on experience with industry tools and platforms
- Examine real-world industry case studies .

Course Outcomes:

CO1: Explain edge computing concepts, architectures, and reference models
CO2: Compare edge, fog, and cloud computing paradigms
CO3: Analyze networking and resource management challenges at the edge
CO4: Implement edge solutions using industry hardware and platforms
CO5: Evaluate AI-enabled edge applications and emerging trends

Unit	Contents	Duration (Hrs.)
1	**Foundations of Edge Computing & Industry Applications** Evolution: Cloud → Fog → Edge, Limitations of cloud computing for latency-sensitive AI , Edge computing definition, characteristics, and benefits Edge vs Cloud vs Fog (comparative analysis) ,Edge computing architectures and deployment models , Edge computing reference models: OpenFog, ETSI MEC,Latency-sensitive AI applications.	8
Case Study	**Smart City Traffic Management System** Analysis of how edge nodes process real-time traffic video data to reduce latency and congestion compared to cloud-only processing. **Tools:** Architecture modeling tools , OpenFog & ETSI MEC documentation	
2	**Edge Hardware & Embedded Platforms** Edge devices, gateways, and edge nodes, Microcontrollers vs Single Board Computers , ARM architecture overview, Edge hardware platforms: Raspberry Pi , NVIDIA Jetson Nano, Intel NUC , Sensors, actuators, and edge gateways.	8
Case Study	**Industrial IoT Sensor Gateway** Deployment of Raspberry Pi as an edge gateway for real-time factory sensor data filtering before cloud transmission. **Tools:**Raspberry Pi OS , NVIDIA JetPack SDK ,Python, C++	
3	**Introduction to networking and communication in edge computing.** Study of edge computing protocols: OPC UA, DDS, gRPC, REST/HTTP, WebSockets. Role of 5G and Multi-access Edge Computing (MEC) in low-latency communication. Performance challenges: latency, bandwidth utilization, and Quality of Service (QoS).	8
Case study	5G-enabled autonomous vehicle communication – MEC reducing decision latency in vehicle-to-infrastructure (V2I) communication. **Tools for practical analysis**: Wireshark and MQTT client tools..	

4	**Edge Software Stack, Platforms and Resource Management** Linux fundamentals for edge devices , Containerization using Docker, lightweight orchestration: K3s, MicroK8s , Edge platforms: AWS IoT Greengrass , Azure IoT Edge , EdgeX Foundry , Resource management and load balancing at edge,Edge–Cloud Integrated Deployment Models.	8
Case Study	**Retail Analytics Edge Platform** Deployment of containerized video analytics at retail stores using Docker and AWS IoT Greengrass. **Tools :** Linux CLI , Docker ,AWS IoT / Azure IoT	
5	**AI, Security and Edge–Cloud Integration** Data processing and analytics at edge, AI inference at the edge ,Model optimization: quantization, pruning , Edge AI frameworks: TensorFlow Lite, ONNX Runtime, OpenVINO , Edge security challenges , Device authentication, encryption (TLS) ,Edge–cloud collaboration and MLOps basics,Challenges in Edge Device Upgrade and Debugging.	8
Case Study	**Healthcare Edge AI for Patient Monitoring** Real-time health data analysis and anomaly detection at the edge with secure cloud synchronization. **Tools:** TensorFlow Lite , OpenCV , Secure MQTT (TLS)	
	Total Hours	40

List of Activities

1. ***Note: Submission of these activities is mandatory for evaluation.***

1.Environment Setup for Edge Computing
Install Raspberry Pi OS / JetPack SDK.
Configure network and SSH access.
Verify Python and system libraries.
Tools: Raspberry Pi / Jetson Nano, Linux, Python.

2.Sensor Data Acquisition at Edge
Interface DHT temperature/humidity sensor
Read and display sensor values
Log data locally on edge device
Tools: Raspberry Pi, GPIO, Python

3.Edge Gateway for Industrial IoT
Collect sensor data.
Perform data filtering and aggregation at edge.
Forward processed data to cloud.
Tools: Raspberry Pi, Python.

4.Performance Analysis of Edge Communication
Send data packets using TCP/UDP.
Measure delay and packet loss.
Compare results.
Tools: Python, Wireshark.

5.AI Inference at the Edge
Perform iLoad a TensorFlow Lite model
Perform inference on live data
Measure latency and accuracy

Tools: TensorFlow Lite, Python

6.Model Optimization for Edge AI

Apply quantization/pruning
Compare model size and performance
Tools: TensorFlow Lite, ONNX

7.Secure Communication at the Edge

Enable TLS for MQTT.
Authenticate edge devices.
Verify secure data transfer.
Tools: Secure MQTT, TLS certificates.

8. Energy and Resource Monitoring on Edge Devices

Monitor CPU usage, memory, and power consumption of the edge device during workload execution.
Compare resource usage for different tasks (e.g., idle vs sensor processing vs ML inference)
Tools: Raspberry Pi/Jetson Nano, Python (psutil), power monitoring sensor (e.g., INA219) or software-based estimation.

Text Books

- Weisong Shi et al., *Edge Computing: A Primer*, IEEE Press
- Rajkumar Buyya et al., *Fog and Edge Computing*, Wiley

Reference Books

- Fog and Edge Computing: Principles and Paradigms, Rajkumar Buyya et al., Wiley, 2019.
- Building the Internet of Things, Maciej Kranz, Wiley, 2016.
- Distributed Systems: Concepts and Design, Coulouris et al., Pearson, 5th Edition.

Online References

- https://services.conferences.computer.org/2026/edge/
- https://aws.amazon.com/documentation-overview/iot-core/
- https://learn.microsoft.com/en-us/azure/iot/iot-mqtt-connect-to-iot-hub

Third Year B. Tech Artificial Intelligence and Data Science

Semester-V

Course Code : AI24PEC304D **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.

1. Deconstruct one published 2D game on paper, documenting its core loop, mechanics, dynamics, and win/lose conditions.
2. Design and playtest a paper/board game prototype in pairs; document playtester feedback and the resulting design iteration.
3. Construct a static Unity 2D scene using sprites, camera setup, and basic environment layout.
4. Write a C# script implementing keyboard-controlled player movement constrained within defined screen boundaries.
5. Implement a collectible pickup mechanic using Rigidbody2D and Collider/Trigger components, with an increasing score on pickup.
6. Add a live in-game UI (score display, restart button) and integrate one sound effect into the existing project.
7. 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

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

Third Year B.Tech Artificial Intelligence & Data Science

Semester-V

Course Code: AI24PCC306 Course Name: Artificial Intelligence and Machine Learning Lab

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: Statistical Analysis

Course Objectives:

- To apply and implement various search strategies for Artificial Intelligence
- To formulate and implement constraints in search problems
- To apply regression, classification and clustering algorithms for creation of ML models
- To introduce and integrate models in the form of advanced ensembles

Course Outcomes:

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

CO1: Design and develop a system using different uninformed, informed search or heuristic approaches

CO2: Apply basic principles of AI in solutions that require problem solving, inference, perception, knowledge representation, and learning.

CO3: Apply regression, classification and clustering algorithms for creation of ML models

CO4: Apply reinforcement learning and its algorithms for real world applications

CO5: Design and develop an interactive AIML application

List of Experiments

Sr. No.	Name of the Experiment	Duration (Hrs.)
1	Implement depth first search algorithm and Breadth First Search algorithm. Use an undirected graph and develop a recursive algorithm for searching all the vertices of a graph or tree data structure.	02
2	Implement A star (A*) Algorithm for any game search problem.	04
3	Implement Alpha-Beta Tree search for any game search problem.	04
4	Implement a solution for a Constraint Satisfaction Problem using Branch and Bound and Backtracking for n-queens problem or a graph coloring problem.	02
5	Implement K-Nearest Neighbours' algorithm on Social network ad dataset. Compute confusion matrix, accuracy, error rate, precision and recall on the given dataset.	02
6	Implement K-Means clustering on Iris.csv dataset.	02
7	Use a different voting mechanism and Apply AdaBoost (Adaptive Boosting), Gradient Tree Boosting (GBM), XGBoost classification on Iris dataset	02
8	Implement Reinforcement Learning using an example of a maze environment that the agent needs to explore.	02
9	Identify AI-generated fake videos or audio.	04
10	Develop AIML chatbot for any suitable customer interaction application.	04
	Total Hours	26

Text Books

- Stuart Russell and Peter Norvig, "Artificial Intelligence: A Modern Approach", Third edition, Pearson, 2021, ISBN : 9781292401171, 1292401176

- Peter Flach: "Machine Learning: The Art and Science of Algorithms that Make Sense of Data", Cambridge University Press, Edition 2012

Reference Books

- Nilsson Nils J , "Artificial Intelligence: A new Synthesis", Morgan Kaufmann Publishers Inc. San Francisco, CA, ISBN: 978-1-55-860467-4
- Dr. Lavika Goel, "Artificial Intelligence: Concepts and Applications", Wiley publication, ISBN: 9788126519934.
- Shalev-Shwartz, Shai, and Shai Ben-David, "Understanding machine learning: From theory to algorithms", Cambridge university press, 2014

Online References

1. https://cs.calvin.edu/courses/cs/344/kvlinden/resources/AIMA-3rd-edition.pdf
2. https://www.cin.ufpe.br/~tfl2/artificial-intelligence-modern-approach.9780131038059.25368.pdf
3. http://aima.cs.berkeley.edu/

Third Year B.Tech Artificial Intelligence & Data Science

Semester-V

Course Code: AI24PCC307 **Course Name: Computer Network & Cloud Computing 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: Statistical Analysis

Course Objectives:

- To understand and apply fundamental networking concepts through hands-on experiments.
- To design and develop client–server communication models using socket programming techniques.
- To understand and apply cloud computing fundamentals.
- Use core cloud services to build cloud-based solutions.
- Apply cloud services to real-world problem scenarios.

Course Outcomes:

After learning the course, the students will be able to:
CO1: Demonstrate the understanding of networking fundamentals.
CO2: Implement socket programming.
CO3: Analyze cloud adoption for specific application.
CO4: Implement different core cloud services.
CO5: Design and implement end to end cloud solution.

List of Experiments

Sr. No.	Name of the Experiment	Duration (Hrs.)
	Part A	
1	Design and simulate a computer network for a smart campus supporting CCTV, biometric attendance, IoT sensors, and high-speed internet access. Analyze bandwidth and security requirements.	4
2	Real-Time Notification System- Create a socket-based server that sends instant alerts or notifications to connected clients.	4
3	Analyze cloud adoption for an e-commerce platform.	4
4	An e-commerce company experiences fluctuating user traffic, especially during seasonal sales and promotional events. The existing on-premise infrastructure is unable to scale dynamically, resulting in performance degradation, increased downtime, and inefficient resource utilization. Design and implement a cloud-based solution that uses virtualization and cloud storage technologies to support the e-commerce platform. The solution should ensure high availability, scalability, data reliability, and cost efficiency.	6
5	Design and implement a Virtual Private Cloud (VPC) that provides controlled access, network segmentation, and secure communication between application components.	4
6	An e-commerce platform aims to improve customer experience and business performance by leveraging machine learning (ML) capabilities such as personalized recommendations, demand forecasting, and fraud detection.	4

	Design a solution using cloud-based machine learning services that enables efficient model development, training, deployment, and real-time inference for the e-commerce platform.	
	Total Hours	**26**

Text Books

1. Behrouz A. Forouzan ,"Data Communication and Networking", 4th edition, Tata McGraw Hill
2. W. Richard Stevens, "Unix Network Programming" Third Edition,ISBN-13: 9780131411555, Addison-Wesley Professional
3. Judith Hurwitz, R.Bloor, M.Kanfman, F.Halper, "Cloud Computing for Dummies", Wiley India.
4. Ronald Krutz and Russell Dean Vines, "Cloud Security", Wiley-India
5. Andreas Wittig & Michael Wittig, "*Amazon Web Services in Action"* – Comprehensive guide to AWS services with real-world examples
6. Quentin Docter, Emmett Dulaney& Timothy L. Warner, "*AWS Certified Cloud Practitioner Complete Study Guide"*

Reference Books

1. Kurose, Ross,"Computer Networking a Top Down Approach Featuring the Internet||", Pearson; 6th edition (March 5, 2012), ISBN-10: 0132856204
2. Antohy T Velte, et.al, "Cloud Computing : A Practical Approach", McGraw Hill.
3. Barrie Sosinsky, "Cloud Computing Bible", Wiley India

Online References

- https://www.my-mooc.com/en/categorie/computer-networking
- AWS Certified Cloud Practitioner

Third Year B.Tech Artificial Intelligence & Data Science

Semester-V

Course Code: AI24PCC308 **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: Programming fundamentals (Python/Java/C/C++), Data Structures, Database Management Systems, Operating System, Applied Statistics, Software Engineering basics, and domain-specific knowledge relevant to the selected project topic.

Course Objectives:

1. To introduce students to structured engineering problem-solving methodologies by mapping real-world challenges to foundational engineering concepts.
2. Design the system architecture, modules, dataset, and technology stack for the proposed solution.
3. Implement, integrate, and refine project modules using suitable algorithms and development tools.
4. Validate, test, and evaluate the performance of the developed system.
5. Document the work through technical/research reports and present it via demonstration and viva-voce.

Course Outcomes:

After completing the mini project, the students will be able to:

CO1: Formulate a problem statement and objectives based on literature survey and gap analysis. (Part I)

CO2: Design system architecture and a module-wise development plan for the proposed system. (Part II)

CO3: Implement and integrate project modules using appropriate algorithms and technologies. (Part III)

CO4: Test, validate, and evaluate the performance of the implemented system. (Part III)

CO5: Prepare project documentation and a research paper, and present the project through demonstration and viva-voce. (Part IV)

Sr. No.	Mini Project Task / Activity
1.	Foundations of Software Engineering & Methodology
	Part I
2.	Finalization of domain and submission of 3 topics(Societal problem/ Research Papers from Elsevier, Web of Science etc. /Hackathon Repository) .
3.	Submission of the approved, signed Base Paper (for baseline-driven projects), OR Submission of an approved Novel Concept Proposal (for original ideas without a specific base paper).
4.	Submission of 1 page abstract and Synopsis.
5.	Project Review 1 - Motivation, Literature Survey, Gap Analysis and Objectives finalization.
	Part II
6.	Project Review 2 - Proposed System Design- Architecture, Mathematical Model, Selection of Algorithm, Dataset Details.
	Part III
7.	Project Review 3 - 50% System Implementation, Models Refinement & Test Case Writing
8.	Project Review 4 - 100% Module Implementation and Project Technical Evaluation (Code Review).

9.	Project Functional Evaluation, Validation and Testing.
	Part IV
10.	Camera-ready submission of the implementation research article or Patent Draft.(Optional)
11.	Project Report Writing & Presentation.

Overall Perspective — Engineering Design and Innovation-I (EDI-I)-(Mini Project)

Engineering Design and Innovation-I (EDI-I) is a dedicated project-based course designed to introduce students to the fundamentals of design thinking and innovation early in their engineering journey. EDI-I is structured as a hands on studio in which student teams design, build, and validate solutions to real world problems by using fundamental engineering knowledge. It spans in wide range of domains.

The problem statements for Mini Project should ensured that each team works on a well scoped, industry/academically curated challenge. This allows the course to maintain consistency in difficulty and learning outcomes across teams, while still exposing students to diverse domains depending on the statement allotted to their group

General Guidelines for Mini Project Development Work:

1. The Mini project should visibly integrate knowledge from at least 3-4 foundational/core courses completed up to the current year of study.
2. Each group should ideally consist of 3-4 students, working under one allotted project guide.
3. A literature survey(10 relevant research papers) /Gap Analysis/fundamental concept etc. must be completed and reviewed by the guide before finalizing the objectives.
4. Regular logbook entries and review-wise progress (as per the review schedule) are mandatory for continuous assessment.
5. The final submission must include a working model/prototype/executable software, a project report, a camera ready research Article / Patent(optional), and a viva-voce/demonstration.
6. Local laboratory readiness, tools/software availability, safety, and faculty mentoring capacity should be considered while finalizing the topic.

Guide Allotment:

1. Mini Project titles are finalized and provided by the department/college.
2. Guide allocation is done as per the domain expertise of faculty.
3. Groups in consultation with their allotted guide, will prepare the Project Synopsis/Abstract based on the given title.

Project Review / Evaluation:

1. The Project Coordinator, along with the Head of Department, shall constitute a review committee composed of faculty members from the department.
2. The review committee will confirm the mini project group and allotted title. A discussion/presentation may be arranged covering the topics listed in the synopsis.
3. The review committee will evaluate the timely progress of the mini project through continuous internal evaluation, where students present their understanding of the given title, literature survey, and proposed approach/design. Each group must be mandatory to appear for all the reviews as per the project calendar.
4. Final term-work evaluation will be conducted by the internal guide, along with an internal examiner, through an oral examination, where students are expected to demonstrate a working model/prototype.

Mini Project Work Progress Monitoring:

1. Every student group project management/tracking tool for documentation and progress tracking.
2. Every group must meet with their allocated guide during allotted slot to track and discuss project progress. Attendance at these meetings is mandatory. A record of each meeting, including the discussions and progress made, must be maintained by the group in the individual/group Project Logbook..

Report:

1. The report should be prepared as per the standard template provided by the department.

2. The plagiarism free project report (AI plagiarism check should be below 20%) must be submitted in both soft copy (PDF format) and hard copy. The hard copy must be duly signed by the respective Project Guide, Head of the Department and the Principal.

Project Deliverables :

As an outcome of the mini project work, each group is expected to complete at least one activity from the list below:

- Develop a working model, functional prototype, or demonstrable application.
- Submit a research article or patent, or present the project at a college-level student technical symposium.
- Participate in college-level / inter college project competitions / Hackathon/ etc or exhibitions.
- Develop a basic application or software module that addresses the approved project title.

Text Books

1. C. R. Kothari, Research Methodology: Methods and Techniques, 2nd (revised) edition2004,New Age International Publishers.
2. Roger S. Pressman, Software Engineering: A Practitioner's Approach, McGraw-Hill Education, 9th edition,2020 ,ISBN 978-1259872976
3. Ian Sommerville, Software Engineering, Pearson Education, 10th edition,2016 , 9th edition2020McGraw-Hill Education, ISBN 978-1259872976Boston, ISBN 978-0133943030

Reference Books

1. Wayne C. Booth, Gregory G. Colomb, Joseph M. Williams, The Craft of Research, 5th edition (with Joseph Bizup & William T. FitzGerald),2024,University of Chicago Press.
2. David Kmiec & Bernadette Longo, *The IEEE Guide to Writing in the Engineering and Technical Fields*, 1st edition, 2018, Wiley-IEEE Press.

Online References

1. IEEE Xplore Digital Library, https://ieeexplore.ieee.org
2. Google Scholar, https://scholar.google.com
3. IEEE Editorial Style Manual for Authors.

Third Year B.Tech Artificial Intelligence and Data Science

Semester-V

Course Code: AI24AEC311 **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	-	-	25	-	-	1	-	-	1

Prerequisite: Basic Mathematical Concepts

Course Objectives:

- To develop fast and accurate mental calculation skills using Vedic Mathematics techniques for arithmetic operations.
- To strengthen students' understanding of number systems, divisibility rules, and numerical properties for efficient problem solving.
- To enhance competency in percentages, profit–loss, discounts, ratios, and partnerships for real-life and competitive exam applications.
- To build logical reasoning abilities through linear, circular, and complex puzzle-based arrangements.
- To 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) • Cube Roots and Square Roots of perfect numbers	2
2	**Number Systems** • Classification (Rational, Irrational, Prime, Composite) • Divisibility Rules (2 to 19) • Unit Digit and Cyclicity • Factorial theory and Number of Zeroes • HCF & LCM (Models: Remainder cases) • Remainder Theorem basics	2
3	**Percentages** • Percentage Increase/Decrease and Multiplying Factors	2

	• Successive Percentage Change Formula • Product Constancy Rule	
4	**Profit, Loss & Discount** • CP, SP, and Profit/Loss Percentage • Relation between Markup, Discount, and Profit • Successive Discounts and Single Equivalent Discount	2
5	**Ratio, Proportion & Partnership** • Ratio Scaling and Merging (A:B and B:C to A:B:C) • Mean, Third, and Fourth Proportional • Coin-based problems • Partnership: Profit sharing based on (Capital $\times$ Time) • 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 neighbors) • 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** • Standard Propositions (All, Some, No, Some Not) • Venn Diagram Method • "Only a Few" and "Few" cases • Possibility-based conclusions • 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.