

Department of Artificial Intelligence & Machine Learning
School of Computer Science & Engineering

SYLLABUS

SEMESTER III

AIM2101: DATA STRUCTURES & ALGORITHMS

[3 1 0 4]

Introduction: algorithm specification; Performance analysis: time and space complexity, asymptotic notation; C concepts: pointers, functions, arrays, passing arrays to functions through pointers, dynamic memory allocation, bubble sort, insertion sort, selection sort, structures, arrays of structures, passing structures to functions; **List:** ADT, array and its types, implementation, operations, linked list and its types, implementation and operations; **Stack:** ADT, implementations using array and linked list, operations and its applications; **Queue:** ADT, implementations using array and linked list, operations and its applications; **Tree:** terminologies, different types, representation of binary tree using array and linked structure, binary search tree, different operations (recursive and non-recursive), heap, heap sort, priority queue, AVL trees, B-tree; **Graph:** Introduction, representation, operations and applications; Searching techniques and hashing.

References:

1. Aaron M. Tenenbaum, Yedidyah Langsam, Moshe J. Augenstein, *Data Structures using C*, Pearson Education, 2013.
2. M. Tenenbaum et al., *Data Structures using C*, First edition, Pearson Education, 2019.
3. Ellis Horowitz, Sartaj Sahni and Susan Anderson-Freed, *Fundamentals of Data Structures in C*, University Press (India) Pvt. Ltd., 2014.
4. Alfred V. Aho, John E. Hopcroft and Jeffrey D. Ullman, *Data Structures and Algorithms*, Pearson Education, 2012.
5. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein, *Introduction to algorithms*, PHI, Third Edition, 2009.
6. Seymour Lipschutz, *Data Structures with C* (Schaum's Outline Series), McGraw Hill Education Private Limited, 2011.
7. Mark Allen Weiss, *Data structures and Algorithm Analysis in C*, Pearson, Second edition, 2014.

AIM2102: RELATIONAL DATABASE MANAGEMENT SYSTEM

[3 1 0 4]

Introduction: DBMS Concepts, Database System Vs File System, Data Models, Schema & Instance, Schema architecture, Data independence, Data Base Languages and interfaces, Database system applications, Database users, Functions of DBA. **Data Modeling using the Entity Relationship Model:** ER model concepts, Entities, Attributes, Relationship & types, Relationship Constraints, Extended ER-Model Concept - Generalization, Specialization and Aggregation, Transforming ER diagram into the tables. **Relational Data models:** Domains, Tuples, Attributes, Relations, Characteristics of relations, Keys, Key attributes of relation, Relational database, Schemas, Integrity constraints. Referential integrity, Relational Algebra and Relational Calculus, Relational algebra operators – Unary, Binary, Set Operations. Tuple oriented and domain oriented relational calculus and its operations. **SQL:** Basic SQL Query, Creating Table and Views, SQL as DML, DDL and DCL, SQL Algebraic Operations, Joins, Sub-Queries, Aggregate Operations, Cursors, Dynamic SQL, Integrity Constraints in SQL, Triggers



Data Base Design: Introduction to Normalization, Functional dependency, Normal forms, Decomposition, Armstrong's Axioms, Canonical Cover, Lossless Join & Dependency preservation Problems with null valued and dangling tuples, multivalued dependencies. **Transaction Processing Concepts:** Transaction Properties & States, Schedules, Serial & Concurrent Schedule, Serializability of schedules, conflict & view serializable schedule, Recoverability, Recovery from transaction failures, log-based recovery, checkpoints, Deadlock handling. **Concurrency Control Techniques:** Concurrency control, Concept of Locks, Concurrency Control Protocols - Two Phase Locking Protocols, Time stamping protocols, validation-based protocol, multiple granularities, Multi version schemes, Recovery with concurrent transactions. **File Structures:** File Organization, Indexing, Primary, Clustered, Secondary Indexes, Hashing, Multilevel Indexing with B-Tree, B+ Tree.

References:

1. H. F. Korth, S. Sudarshan and A. Silverschatz, *Database System Concepts*, Six edition, TMH, New Delhi, 2017.
2. R. Elmasri and S. Navathe, *Fundamentals of Database systems*, Seven edition, Pearson Education, 2017.
3. C. J. Date, *Database Systems*, Eight edition, Prentice Hall of India, New Delhi, 2012.

AIM2103: PRINCIPLES OF ARTIFICIAL INTELLIGENCE

[3 1 0 4]

Introduction: Foundations, scope, problems, and approaches of AI; **Intelligent agents:** Reactive, deliberative, goal-driven, utility-driven, and learning agents; Artificial Intelligence programming techniques; **Problem-solving through Search:** Problem formulation, State-space, Production systems, Control Strategies, Informed and uninformed search, heuristic search methods, Forward and backward reasoning problem-reduction, A, A*, AO*, minimax, constraint satisfaction problem **Knowledge Representation and Reasoning:** Knowledge Based systems, Propositional Logic, syntax, semantics, inference, propositional theorem proving, Resolution- Horn clauses, Forward chaining and backward chaining, First Order Logic -Representation, Syntax and semantics, quantifiers, Inference in First Order Logic. Inductive, deductive learning. **Planning:** Planning as search, partial order planning, construction, and use of planning graphs; Blocks world problems **Representing and Reasoning with Uncertain Knowledge:** probability, connection to logic, independence, Bayes rule, Bayesian networks, probabilistic inference, sample applications; Hidden Markov Model, Maximum Entropy Markov Model, Conditional Random Field.

References:

1. S. Russell and P. Norvig, *Artificial Intelligence A Modern Approach*, Fourth edition, Pearson 2020
2. E. Rich et al., *Artificial Intelligence*, Third edition, Tata McGraw Hill, 2017
3. G. Antoniou et al., *A Semantic Web Primer*, MIT Press, 2012.

AIM2120: OBJECT ORIENTED PROGRAMMING USING PYTHON

[3 1 0 4]

Introduction: Programming a computer, Programming languages; **Python basics:** Getting started with Python, Essentials of a Python program; Integers, Floating-point numbers, Strings; **Variables and scope:** Variables, Modifying values, Type conversion; **Selection control statements:** Selection: if statement, Boolean values, operators, and expressions; **Collections:** Lists, Tuples, Sets, Ranges, Dictionaries, Conversion, Sequences; **Loop control statements:** while, for statements, Nested loops, Iterables, iterators and generators, Comprehensions, The break and continue statements **Functions:** Input parameters, Return values, Default parameters, *args and **kwargs, Decorators, Lambdas. Generator functions and yield. **Object-Oriented programming:** OOP's Concepts, Classes, and Objects: Defining and using



a class, Instance attributes, Class attributes, Class, decorators, inspecting an object, Constructor, Abstraction, Composition. Inheritance: Types of Inheritance. overriding magic methods; I/O and Errors Handling: Errors, exceptions, handling exceptions, Debugging programs, Logging, Testing. **Packaging:** Modules, Packages, Documentation, **File Handling:** Introduction, Access Methods, Read and write operation, Working with directories. **Python Libraries:** Pandas, Matplotlib, NUMPY, Introduction to GUI programming with Tkinter.

References:

1. D. Phillips, *Python 3 Object-Oriented Programming Build robust and maintainable software with object-oriented design patterns in Python 3.8*, Third edition, Packt Publishing, January 2018
2. W. J. Chun, *Core Python Applications Programming*, Third edition, Prentice Hall Publishers, 2012
3. J. Grus, *Data Science from Scratch: First Principles with Python*, First edition, O'Reilly Media, 2015.

AIM2121: OBJECT ORIENTED PROGRAMMING USING JAVA

[3 1 0 4]

Basics of Java: Features of Java, Byte Code and Java Virtual Machine, JDK, Data types, Operator, Control Statements – If , else, nested if, if-else ladders, Switch, while, do-while, for, for-each, break, continue, Basics of Java: Features of Java, Byte Code and Java Virtual Machine, JDK, Data types, Operator, Control Statements – If , else, nested if, if-else ladders, Switch, while, do-while, for, for-each, break, continue. **Classes, Objects and Methods:** Class, Object, Object reference, Constructor, Constructor Overloading, Method Overloading, Recursion, Passing and Returning object form Method, new operator, this and static keyword, finalize () method, Access control, modifiers, Nested class, Inner class, Anonymous inner class, Abstract class. **Inheritance and Interfaces:** Use of Inheritance, Inheriting Data members and Methods, constructor in inheritance, Multilevel Inheritance – method overriding Handle multilevel constructors – super keyword, Stop Inheritance - Final keywords, Creation and Implementation of an interface, Interface reference, instance of operator, Interface inheritance, Dynamic method dispatch, Understanding of Java Object Class, Comparison between Abstract Class and interface. **Package:** Use of Package, CLASSPATH, Import statement, Static import, Access control. **Exception Handling:** Exception and Error, Use of try, catch, throw, throws and finally, Built in Exception, Custom exception, Throwable Class. **Multithreaded Programming:** Use of Multithread programming, Thread class and Runnable interface, Thread priority, Thread synchronization, Thread communication, Deadlock. IO Programming: Introduction to Stream, Byte Stream, Character stream, Readers and Writers, File Class, File Input Stream, File Output Stream, Input Stream Reader, Output Stream Writer, File Reader, File Writer, Buffered Reader. Collection Classes: List, Abstract List, Array List, LinkedList, Enumeration, Vector, Properties, Introduction to Java.util package.

References:

1. Brett McLaughlin, Gary Pollice, and David West, *Head First Object-Oriented Analysis and Design*, First edition, O'Reilly Media.

AIM2130: DATA STRUCTURES & ALGORITHMS LAB

[0 0 2 1]

One Dimensional Arrays and Two Dimensional Arrays: Static and Dynamic Allocation, Passing of Arrays to Functions, Stack implementation using Array, Using Structures and Pointers, Programs on Evaluation of Expressions in Infix, Prefix and Postfix Notations, Programs on Conversion from One Notation to Other, **Queue** : Implementation Linear Queue, Circular Queue, Priority Queue using Array, Using Structures and Pointers ,Tower of Hanoi,



GCD, Fibonacci Definition, Binary Search, Prefix to Postfix etc. , **Link List:** Implementation of Singly, Doubly and Circular Linked Lists Using Pointers, Polynomial Addition, Sparse Matrices etc., **Tree:** Implementation of Binary Search Tree through Arrays and Pointers, Tree Traversals, Various Operations on Binary Search Tree, Huffman Algorithm, Josephus Problem etc, Implementation through Arrays and Pointers, Transitive Closure and Searching and Sorting algorithms.

References:

1. E. Horowitz et al., *Fundamentals of Data Structures in C*, University Press (India) Pvt. Ltd., 2014
2. M. Tenenbaum et al., *Data Structures using C*, First edition, Pearson Education, 2019
3. V. Aho et al., (1e), *Data Structures and Algorithms*, Pearson, 2012
4. T. H. Cormen et al., *Introduction to algorithms*, Third edition, PHI, 2010.
5. S. Lipschutz, *Data Structures with C (Schaum's Outline Series)*, First edition, Tata McGraw Hill Education Private Limited, 2011

AIM2131: RELATIONAL DATABASE MANAGEMENT SYSTEM LAB

[0 0 2 1]

Introduction to SQL and its different command categories i.e., DDL, DML, DQL and DCL, Data Integrity Constraints and Built-in Functions, Design and implementing the data requirements of a simple DB application, Experiments on views, indexing, triggers, stored procedures, transaction.

References:

1. I. Bayross, *Teach yourself SQL & PL/SQL using Oracle 8i & 9i with SQLJ*, Third edition, BPB Publications, 2010.
2. A. Silberschatz et al., *Database System Concepts*, Six edition, McGraw Hill, 2013
3. R. Elmasri and S. B. Navathe, *Fundamentals of Database Systems*, Seven edition, Addison-Wesley, 2017.

AIM2170: PROJECT BASED LEARNING -I [0 0 2 2]

This course aims at developing innovative skills in the students whereby they apply in totality the knowledge and skills gained through the course work in the solution of problem or by undertaking a project. Project work, therefore, should match the strengths of students. The project assignment can be individual assignment or a group assignment. The project work identified in collaboration with industry should be preferred. Each teacher is expected to guide the project work of 5-6 students. The project assignments may consist of: Programming customer-based applications, Web page designing (Only dynamic), Database applications, Software Development etc. Execution through mentor mentee policy.

SEMESTER IV

AIM2201: DESIGN & ANALYSIS OF ALGORITHMS

[3 1 0 4]

Introduction: Algorithm Definition and Criteria of Algorithms, Iterative and Recursive algorithms, Performance Analysis: Priori and Posteriori Analysis, Asymptotic Notations, Space Complexity, Time Complexity, Performance measurement of iterative and recursive algorithms, **Solving Recurrence Relations:** Substitution Method, Iterative Method, Recursive Tree Method, Master Method, **Divide and Conquer:** Introduction, Binary Search, Finding Maximum and Minimum, Merge Sort, Quick Sort, Randomized Quick Sort, Closest Pair of Points, Integer Multiplication, Fast Fourier Transforms Graph Search Algorithm: Graph representation, Breadth First Search and Depth First Search, **Greedy Strategy:** Introduction, Knapsack Problem, Job Sequencing with Deadlines, Huffman Coding, Union and Find Operation (Set and Disjoint Set), Minimum Cost Spanning Tree Algorithms (Prim's and Kruskal's), Optimal Merge Patterns, Single Source Shortest Path (Dijkstra's Algorithm), Dynamic Programming:



Introduction, Single Source Shortest Path (Bellman and Ford Algorithm), All Pair Shortest Path (Floyd Warshall's Algorithm), Optimal Binary Search Trees, 0/1 Knapsack Problem, Travelling Salesperson Problem, Longest Common Subsequence, Matrix Chain Multiplication, Edit distance, Viterbi algorithm **Backtracking:** Introduction, N-Queens Problem, Graph Colouring and Hamiltonian Cycles, **Branch and Bound:** Introduction, FIFO and LC Branch and Bound, 0/1 Knapsack Problem, Travelling Salesman Problem, **String Matching:** Naïve String Matching, Rabin Karp Algorithm, Knuth-Morris-Pratt Algorithm, Boyer-Moore Algorithm, **Complexity Classes:** NP, NP-Complete and NP-Hard Problems, Polynomial time reductions, Satisfiability, Reduction from Satisfiability to Vertex Cover, Cook's Theorem.

References:

1. E. Horowitz et al., *Fundamental of Computer Algorithms*, Second edition, Universities Press, 2008.
2. T. H. Cormen et al., *Introduction to Algorithms*, Third edition, MIT press, 2010.
3. T. Roughgarden, *Algorithms Illuminated (Part 2): Graph Algorithms and Data Structures First edition*, Wiley, 2018

AIM2202: OPERATING SYSTEMS

[3 1 0 4]

Introduction: Definition of operating systems, Single and multi-processor systems, Operating system services, System commands and system calls, Interrupt, System boot, Operating system structure, Types of OS, Multi-user, Multitasking, Embedded, Real-time, Network, Distributed. **Process and Thread:** Process concept, Operations on processes, Inter-process communication, UNIX pipes, Multithreading, Multithreaded models, Programs using PThread. **Process Scheduling:** Basic concepts, Scheduling criteria, Scheduling algorithms. **Synchronization:** Critical section problem, Dekker's algorithm, Peterson solution, Synchronization hardware, Semaphores, Classical problems of synchronization, Deadlock, Methods for handling deadlock- prevention, avoidance, detection, and recovery. **Memory Management:** Address binding, Logical vs Physical address space, Swapping, Contiguous memory allocation, Paging, Structure of Page Table, Segmentation, Demand Paging, Page Replacement Policies, Allocation of Frames, Thrashing. **File System Interface and Implementation:** File Concept, Access Methods, Directory and Disk Structure, File System Mounting, File System Structure, File System Implementation, Allocation Methods, Free Space Management. **Disk Management:** Disk Scheduling Algorithms, Disk Management, Swap Space Management. **Case Studies:** Linux, Windows, iOS, Android.

References:

1. A. Silberschatz, et al., *Operating System Concepts*, (9e), Wiley, 2018
2. A.S. Tanenbaum and H. Bos, *Modern Operating Systems*, (4e), Pearson, 2015
3. W. Stallings, *Operating Systems: Internals and Design Principles*, (9e), Pearson, 2018.

AIM2220: SOFTWARE ENGINEERING & PROJECT MANAGEMENT

[3 1 0 4]

Software Engineering – importance – emergence - Phases of software development - Feasibility study Phases and Life cycle models of Software Development. Requirement Analysis, Design, Implementation, Testing, and Maintenance phases Different Software Life Cycle Models - Classical waterfall, Iterative, prototyping, Spiral, and Agile - Compare Lifecycle models. Requirements Analysis and **Design Requirement Analysis** – Analysis process, Requirements specification, desirable characteristics of an SRS, structure of an SRS document, Data Flow Diagrams - Role of Software Architecture and Architecture. **Software Design** - Software design concepts - Function Oriented Design and its Complexity Metrics -



Object Oriented Design and its Complexity Metrics - Detailed Design. **Software Implementation and Testing-** Software Coding- Programming principles and coding guidelines - method of incrementally developing code - managing the evolving code Testing - Unit testing and Code Inspection - Testing concepts and testing process - Design of Test case and Test plan - Black-box testing - White box testing. **Software Project Management-**Software Project Management Framework - methods to estimate project time and cost, Resource. Planning for a Software Project. Management, Identification, Analysis, mitigation, and monitoring of Project Risks - Ensuring Project. Quality and quality management, Configuration Management, Change management, CMMI, Quality standards -ISO.

References:

1. B. Hughes et al., *Software Project Management*, Sixth Edition, McGraw Hill, 2017
2. P Jalote., *Software Project Management in Practice*, First Edition, Addison Wesley Professional, 2010.
3. R. S. Pressman, *Software Engineering: A practitioner's approach*, Eighth Edition, McGraw Hill, 2014
4. S. A. Kelkar, *Software Project Management: a concise study*, Third Edition, PHI Learning- New Delhi, 2013
5. S. H. Kan, *Metrics and Models in Software Quality Engineering*, Second Edition, Pearson, 2010.

AIM2221: AGILE SOFTWARE DEVELOPMENT

[3 1 0 4]

Software Engineering — importance — emergence - Phases of software development - Feasibility study. Phases and Life cycle models of Software Development Requirement Analysis, Design, Implementation, Testing, and Maintenance phases Different Software Life Cycle Models - Classical waterfall, Iterative, prototyping, Spiral, and Agile Model. **Agile Model:** Need of Agile software development, agile context– Manifesto, Principles, Methods, Values, Roles, Artifacts, Stakeholders, and challenges. Business benefits of software agility. **Agile Project Planning:** Recognizing the structure of an agile team– Programmers, Managers, Customers. User stories– Definition, Characteristics, and content. Estimation– Planning poker, Prioritizing, and selecting user stories with the customer, projecting team velocity for releases and iterations. **Agile Project Design:** Fundamentals, Design principles–Single responsibility, Open-closed, Liskov substitution, Dependency-inversion, Interface-segregation. **Agile Project Design Methodologies:** Need of scrum, Scrum practices –Working of scrum, Project velocity, Burn down chart, Sprint backlog, Sprint planning and retrospective, Daily scrum, Scrum roles– Product Owner, Scrum Master, Scrum Team. Extreme Programming- Core principles, values, and practices. Kanban, Feature-driven development, Lean software development. **Agile Project Testing:** The Agile lifecycle and its impact on testing, test driven development– Acceptance tests and verifying stories, writing a user acceptance test, Developing effective test suites, Continuous integration, Code refactoring. Risk based testing, Regression tests, Test automation.

References:

1. Ken Schawber, Mike Beedle, *Agile Software Development with Scrum*, International Edition, Pearson.
2. Robert C. Martin, *Agile Software Development, Principles, Patterns and Practices*, First International Edition, Prentice Hall.
3. Pedro M. Santos, Marco Consolaro, and Alessandro Di Gioia, *Agile Technical Practices Distilled: A learning journey in technical practices and principles of software design*, First edition, Packt Publisher.
4. Lisa Crispin, Janet Gregory, *Agile Testing: A Practical Guide for Testers and Agile Teams*, International edition, Addison Wesley.



5. Alistair Cockburn, *Agile Software Development: The Cooperative Game*, Second edition, Addison-Wesley.

AIM2230: DESIGN & ANALYSIS OF ALGORITHMS LAB

[0 0 2 1]

Sorting & Searching Algorithm: Insertion sort, selection sort, binary search. Basic data structures: stacks and queues, graphs and trees, binary trees; Algorithmic paradigms: Recursion, divide-and-conquer, Merge sort, Quick sort. **Greedy:** Knapsack, Huffman encoding, dynamic programming, lower bounds and optimal algorithms; **Heaps:** Heaps, priority queues, min-max heaps, heap sort. Dynamic search structures, Binary search trees, height balancing, B-trees; **Algorithms on arrays:** Linear-time median finding, sorting in linear time (counting sort, radix sort, bucket sort), String matching (Rabin-Karp and Knuth-Morris-Pratt algorithms); **Graph algorithms Traversal:** (BFS, DFS, topological sort), Minimum spanning trees (Prim and Kruskal algorithms), shortest paths (Dijkstra's and Floyd-Warshall algorithms). Mini-Projects & Case Studies.

References:

1. E. Horowitz *et al.*, *Fundamental of Computer Algorithms*, Second edition, Universities Press, 2008
2. T. H. Cormen *et al.*, *Introduction to Algorithms*, Third edition, MIT press, 2010.
3. T. Roughgarden, *Algorithms Illuminated (Part 2): Graph Algorithms and Data Structures First edition*, Wiley, 2018

AIM2230: OPERATING SYSTEMS LAB

[0 0 2 1]

Basic Linux commands: Illustration of shell functions, wild cards, redirection, pipes, sequencing, grouping, background processing, command substitution, sub shells, Shell programming. **System Calls:** File and process, I/O Redirection, IPC using Pipe and Signals. **PThread API:** Multithreaded programs, Synchronization programs using PThreads and Semaphores, CPU Scheduling, Deadlock, Memory Management. **Creating a Virtual Machine:** Virtual Machine Files and Snapshots, Virtual Machine Cloning and Exporting.

References:

1. W. R. Stevens and S. A. Rago, *Advanced Programming in the UNIX Environment*, Third edition, Addison-Wesley, 2017.
2. S. Das, *Unix Concepts and Applications*, Fourth edition, McGraw Hill, 2017
3. K. A. Robbins and S. Robbins, *Unix Systems Programming: Communication, Concurrency, and Threads*, Second edition, Prentice Hall, 2015.

AIM2270: PROJECT BASED LEARNING 2

[0 0 2 2]

This course aims at developing innovative skills in the students whereby they apply in totality the knowledge and skills gained through the course work in the solution of problem or by undertaking a project. Project work, therefore, should match the strengths of students. The project assignment can be individual assignment or a group assignment. The project work identified in collaboration with industry should be preferred. Each teacher is expected to guide the project work of 5-6 students. The project assignments may consist of: Programming customer-based applications, Web page designing (Only dynamic), Database applications, Machine Learning application, Software Development etc. Execution through mentor mentee policy.

PROGRAM ELECTIVE 1

AIM2140: COMPUTER ORGANIZATION & ARCHITECTURE

[3 0 0 3]

Basic Structure of Computers: Computer Types, Functional Units, Basic Operational Concepts, Number Representation and Arithmetic Operations, Performance, **Instruction Set Architecture:** Memory Locations and Addresses, Instructions, and Instruction Sequencing

Addressing Modes. **Arithmetic:** Addition and Subtraction of Signed Numbers, Design of Fast Adders, Multiplication of Positive Numbers, Signed Operand Multiplication, Fast Multiplication. **Memory Hierarchy:** Basics of Caches, Measuring and Improving Cache Performance. Processor Datapath and Control: Single and Multiple Bus Organization, Microprogram Control Unit, Hardware Control Unit Pipeline: Basic Concepts, Data Hazard, Instruction Hazard, Influence on Instruction Hazard. Multicores, **Multiprocessors and Clusters:** Flynn's classification, multi-core, superscalar, vector processor and GPU.

References:

1. C. Hamacher, Z. Vranesic and S. Zaky, *Computer Organization and Embedded Systems*, Six edition, McGraw Hill, 2017.
2. D. A. Patterson and J. L. Hennessy, *Computer Organization and Design: The Hardware and Software Interface*, Six edition, Morgan Kaufmann Publishers, 2020.
3. J. P. Hayes, *Computer Architecture and Organization*, Third edition, McGraw Hill, 2012.

AMI2141: CLOUD COMPUTING

[3 0 0 3]

Introduction: Distributed Computing and Enabling Technologies, Cloud Fundamentals: Cloud Definition, Evolution, Architecture, Applications, deployment models, and service models. **Implementation:** Study of Cloud Computing Systems like Amazon EC2 and S3, Google App Engine, and Microsoft Azure, Build Private/Hybrid Cloud using open-source tools, Deployment of Web Services from Inside and Outside Cloud Architecture. MapReduce and its extensions to Cloud Computing, HDFS, and GFS. **Interoperability and Service Monitoring:** Issues with interoperability, Vendor lock-in, Interoperability approaches. SLA Management, Metering Issues, and Report generation. Resource Management and Load Balancing: Distributed Management of Virtual Infrastructures, Server consolidation, Dynamic provisioning and resource management, Resource Optimization, Resource dynamic reconfiguration, Scheduling Techniques for Advance Reservation, Capacity Management to meet SLA Requirements, and Load Balancing, various load balancing techniques. Migration and Fault Tolerance: Broad Aspects of Migration into Cloud, Migration of virtual Machines and techniques. Fault Tolerance Mechanisms. **Advances:** Grid of Clouds, Green Cloud, Mobile Cloud Computing.

References:

1. R. Buyya, J. Broberg, A. Goscinski, *Cloud Computing Principles and Paradigms*, Wiley Publishers, 2013.
2. B. Sosinsky, *Cloud Computing Bible*, Wiley, 2011.
3. M. Miller, *Cloud Computing: Web-based Applications that change the way you work and collaborate online*, Pearson, 2008.
4. D. S. Linthicum, *Cloud Computing and SOA Convergence in Your Enterprise: A Step-by-Step Guide*, Addison Wesley Information Technology Series, 2010.
5. T. Velte, A. T. Velte, R. Elsenpeter, *Cloud Computing: A Practical Approach*, McGraw Hill, 2017.

SEMESTER V

AIM3101: Machine Learning

[3 1 0 4]

Introduction to Machine Learning: Basics of Machine Learning, labelled, unlabelled and, Types of Machine Learning: supervised, unsupervised, semi-supervised and reinforcement learning, parametric vs non-parametric models, learning theory, bias-variance tradeoff, model selection, Data Visualization Concepts, Data Sampling Techniques. **Regression Techniques and Evaluation:** Linear Regression, Gradient descent for Linear Regression, Ridge and Lasso



Regression. Logistic Regression. Performance measures for Regression models. Cost functions, Underfitting, Overfitting. **Core Classification Algorithms:** k Nearest Neighbours (KNN), Support Vector Machines: kernel functions, Naïve Bayes, Decision Trees. Performance & Evaluation of Classifiers: Confusion matrix, F1 score, Accuracy, Recall, Precision, Precision-recall curve, ROC curve. **Advanced Classification and Ensemble Techniques:** Ensemble Learning: Concept of weak learners, Bagging and Boosting, Adaptive Boosting, Extreme Gradient Boosting (XGBoost), Random Forests, Bayesian Networks, Model Validation: K-Fold and Cross-Validation, holdout method, Gaussian Mixture Models (GMM). **Unsupervised learning:** Clustering: Partitioning, Hierarchical and Density based methods, Agglomerative Clustering, K-Means, K-Medoids. Association Rule Mining. **Dimensionality Reduction:** Feature Extraction and Feature Selection. Principal Component Analysis (PCA), Partial Least Squares, Latent Factors, Factor Analysis.

References:

1. Ethem Alpaydin, *Introduction to Machine Learning*, MIT Press, 4th Edition, 2020.
2. Murphy, K. P. *Machine learning: A probabilistic perspective*. MIT Press, 2012.
3. Géron, A. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*, 3rd ed. O'Reilly Media, 2022.

AIM3102 Automata Theory & Compiler Design

[3 1 0 4]

Introduction to Automata Theory: Alphabets, Strings, and Languages, Deterministic Finite Automata (DFA), Non-Deterministic Finite Automata (NFA), Equivalence of DFA and NFA, Regular Expressions and their equivalence with Finite Automata, Applications of Finite Automata; **Context-Free Grammars and Pushdown Automata:** Context-Free Grammars (CFGs), Derivations and Parse Trees, Ambiguity in Grammars, Simplification of CFGs, Pushdown Automata (PDA) – Definition and Language Recognition, Equivalence of PDA and CFG; **Turing Machines and Computability:** Introduction to Turing Machines, Language Acceptance by Turing Machines, Variants of Turing Machines, Church-Turing Thesis, Recursive and Recursively Enumerable Languages, Undecidability and the Halting Problem; **Introduction to Compiler Design:** Structure of a Compiler, Lexical Analysis – Role, Specification of Tokens, Lex Tools, Syntax Analysis – Context-Free Grammars, Top-Down Parsing (Recursive Descent, LL(1)), Bottom-Up Parsing (Shift-Reduce, LR, SLR, LALR), Parse Trees and Abstract Syntax Trees (AST); **Semantic Analysis and Code Optimization:** Intermediate Code Generation – Three Address Code, Syntax-Directed Translation, Type Checking and Symbol Table, Code Optimization, DAGs, Code Generation.

References:

1. Hopcroft, Ullman, and Motwani – *Introduction to Automata Theory, Languages and Computation*
2. Alfred V. Aho, Monica S. Lam, Ravi Sethi, and Jeffrey D. Ullman – *Compilers: Principles, Techniques, and Tools* (Dragon Book)
3. Peter Linz – *An Introduction to Formal Languages and Automata*
4. John C. Martin – *Introduction to Languages and the Theory of Computation*.
5. Dick Grune et al. – *Modern Compiler Design*

AIM3120 Computer Networks [3 1 0 4]



Introduction to Computer Networking Concepts: Layered Network Protocol Architectures OSI and TCP; Personal, Local, Metropolitan and Wide Area Networks; Telecommunications and Cellular Networks overview; **Physical Layer:** Data Transmission: Concepts and Terminology, Analog and Digital Data Transmission & capacity; **Transmission Media:** Guided and Wireless Transmission, Wireless Propagation, Line-of-Sight Transmission. Signal Encoding Techniques: Analog and Digital Signals; **Digital-To-Digital Conversion:** Line Coding Schemes, Block Coding, Scrambling, Analog-To-Digital Conversion: Pulse Code Modulation, Delta Modulation; **Data Link Layer and Logical Link Control (LLC) sub-layer:** Framing: Reliable transmission and Automatic Repeat Request (ARQ), Go-back-N, Selective Repeat. Performance analysis of ARQ protocols. Example protocols such as HDLC and PPP; **Medium Access Control (MAC) sub-layer:** Shared media systems: Bus, Star and Ring topologies; TDMA, FDMA, CSMA, CSMA/CD, Ethernet, CSMA/CA protocols; Shared and Switched Ethernet; Related protocols such as ICMP, NAT, ARP and RARP; **Network Layer:** Internet Protocol (IP) suite; IPv4 and IPv6 addressing and headers; Routing protocols including distance-vector and link-state approaches; **Transport Layer:** Reliable end-to-end transmission protocols; UDP header; Details of TCP header and operation including options headers and congestion control; **Application Layer:** Socket Interface and Socket programming; Example protocols such as DNS, SMTP, FTP, and HTTP.

References:

1. J.F. Kurose and K.W. Ross, *Computer Networking - A top-down approach*, (7e), Pearson, 2017.
2. A. S. Tanenbaum, *Computer Networks*, (5e), Pearson Education India, 2013.
3. W. Stallings, *Data & Computer Communications* (9e), Pearson Education Inc., Noida, 2017.
4. L. L. Peterson and B.S. Davie, *Computer Networks- A Systems approach*, (5e), Elsevier, 2016
5. B. A. Forouzan and F.Mosharraf, *Computer Networks A Top-Down Approach*, Mc-Graw Hill, 2017

AIM3121 Data Communication and Networking

[3 1 0 4]

Introduction: Data communications, Networks, Network types, Standards. Network Protocols: Introduction, Need for protocol architecture, OSI Model, TCP/IP protocol architecture; **Data Transmission:** Concepts and terminology, Analog and digital data transmission, Transmission impairments, Channel capacity, Transmission Media. Signal Encoding Techniques: Analog and digital Signals, Digital-to-digital conversion: Line coding schemes, Block coding, scrambling, Analog To-Digital Conversion: Pulse code modulation, Delta modulation; **Digital Data Communication Techniques:** asynchronous and synchronous transmission, Types of errors, Error detection, Error correction, Line configurations; **Data Link Layer:** Introduction, Flow control, Error control, High-level data link control. Multiplexing and Spread Spectrum: Frequency-division multiplexing, Time-division multiplexing, Code-division multiple access, Space division multiplexing, Spread Spectrum; **Media Access Control (MAC):** Random access, Aloha, Carrier sense multiple access (CSMA), CSMA with collision detection, CSMA with collision avoidance, Code-division multiple access. Wired LANs (Ethernet)

References:

1. B. Forouzan, *Data Communication & Networking*, (5e), McGraw Hill Education, 2013.
2. W. Stallings, *Data and Computer Communications*, (10e), Pearson Education, 2018.



AIM3140 Web Technologies

[3 0 0 3]

Web Basics and Page Design: Introduction to Internet, WWW, and HTTP protocols, HTML & XHTML, Basic tags, lists, tables, images, links; Hyperlinking and navigation; HTML Forms and user input handling. **Styling and Dynamic Client-Side Programming:** CSS: Inline, Internal, and External Styling; JavaScript Basics: Events, Functions, Document Object Model (DOM); Bootstrap: Grid system, components, responsive design. **Introduction to Backend with Python:** Basics of Python for web; CGI Programming: Handling GET/POST requests; Introduction to MySQL: Basic CRUD operations using command line. **Advanced Python Web Development:** Python DB connectivity using MySQL Connector; JSON parsing, JQuery for asynchronous web updates (AJAX); NodeJS for lightweight server-side scripting; AngularJS for interactive front-end interfaces. **Django Web Framework:** Django Architecture (MVT Pattern), Models, Views, Templates, Admin Interface, Django ORM for database operations, Dynamic page configuration using URL routing. **Project Development:** Mini Project Development and Evaluation

References:

1. S. Dauzon, A. Bendoraitis and A. Ravindran, Django, *Web Development with Python*, (1e), Packt
2. H. M Deitel et al., *Internet & World Wide Web How to Program*, (5e), Pearson Education, 2011
3. C. Bates, *Web Programming: Building Internet Application*, (3e), Wiley India, 2012
4. W. J. Chun, *Core Python Applications Programming*, (3e), Prentice Hall Publishers, 2012
5. R. Connolly and R. Hoar, *Fundamentals of Web Development*, (1e), Pearson Education India, 2015

AIM3141: NoSQL Databases

[3 0 0 3]

Foundations of NoSQL & Distributed Systems: Overview of traditional RDBMS limitations and the emergence of NoSQL. Types of NoSQL databases: Document, Key-Value, Column-Family, and Graph. Key concepts: ACID vs. BASE, CAP Theorem, and Eventual Consistency. The PACELC Theorem: Understanding the trade-off between Latency and Consistency during normal operation. Data Partitioning Strategies: Horizontal vs. Vertical Sharding and Consistent Hashing. Use cases and industry motivation. **MongoDB – Architecture & Basic Operations:** Introduction to MongoDB, installation, and environment setup (Shell, Compass, and Atlas). Data model: Documents, Collections, and BSON types. CRUD operations, projections, and querying embedded documents. Indexing, filtering, and sorting. Advanced Schema Design: The Outlier Pattern, Computed Pattern, and Extended Reference Pattern. **Aggregation & Application Development:** Aggregation pipeline, stages, and operators. Data modeling: Embedded vs. Referenced documents and Schema validation constraints, and indexes. Introduction to application development using PyMongo (Python) or Node.js. Reactive Data Streams: Using MongoDB Change Streams for real-time application updates. Hands-on project: Building a CRUD-based application. **Advanced Concepts & Cloud Ecosystem:** Data Partitioning, Sharding, and Replica Sets for High Availability. MongoDB Security: User Roles, Authentication, and Encryption at Rest. Backup, Restore, Monitoring, and Performance Optimization. Introduction to Database-as-a-Service (DBaaS): Managing NoSQL in the Cloud (AWS DynamoDB and MongoDB Atlas). **Vector Databases & Case Studies:** Introduction to Vector Databases: Storing and querying high-dimensional embeddings for AI. Vector Search Fundamentals: Similarity metrics (Cosine, Euclidean) and Approximate Nearest Neighbor algorithms. Overview of Redis for Caching and Neo4j for Graph-based recommendations. AI

Case Study: Building a simple RAG (Retrieval-Augmented Generation) back-end using a Vector Store (e.g., Pinecone or Milvus). Case studies of MongoDB in real- world applications. Comparative matrix and guidelines for selecting a NoSQL solution.

References:

1. Dan Sullivan, *NoSQL for Mere Mortals*, Pearson Education, 2015.
2. Kristina Chodorow, *MongoDB: The Definitive Guide*, O'Reilly Media.
3. Eric Redmond and Jim R. Wilson, *Seven Databases in Seven Weeks*, O'Reilly Media.
4. MongoDB Documentation: <https://www.mongodb.com/docs>

AIM3142 Linux System Administration and Shell Scripting

[3 0 0 3]

Introduction to Linux Operating System: Overview of Linux, its history, evolution, distributions, Basic Linux commands and file system hierarchy, **Installation:** Prerequisites for installing Linux, step-by-step installation, partitioning, disk management, network and system settings, and post-installation configurations, **Networking:** Configuring network interfaces, wired and wireless setup, using command-line tools like ifconfig and ip, troubleshooting network issues, and configuring DNS and static IP addresses, **Utilities:** Overview of common Linux utilities such as grep, find, awk, file compression, disk usage, and system monitoring tools , **File Systems:** Introduction to Linux file systems, mounting/unmounting, file system permissions, access control, and disk management tools, **The Shell and Popular Editors:** Introduction to the shell, text editors, and basic shell scripting with syntax, variables, and commands, Advanced Bash scripting with conditionals, loops, and functions **Programming the Bourne Again Shell:** Advanced Bash techniques, file I/O operations, input/output redirection, and debugging, **Linux System Administration:** User and group management, file permissions, process management, job control, cron jobs, system logs, and monitoring services, **Web Server Configuration:** Configuring Apache servers, virtual hosts, SSL, PHP, MySQL, and troubleshooting web server issues, **File Server Configuration:** Configuring NFS and SMB for file sharing, and troubleshooting file server issues, **Samba Servers:** Installing and configuring Samba, user authentication, configuring shares **Network File Systems:** Introduction to NFS, configuring NFS server and client, mounting NFS shares, network file system security.

References:

1. Shotts, W E (2019) *The Linux command line: A complete introduction*, (2nd ed) No Starch Press
2. Negus, C (2021) *Linux Bible*, (9th ed) Wiley
3. Korman, M J (2009) *Linux networking: A beginner's guide to networking*, McGraw-Hill
4. Blum, R (2007) *Linux command line and shell scripting bible*, Wiley
5. Nemeth, E, Snyder, G, Hein, B, & Whaley, D (2017) *Linux system administration handbook*, (5th ed) Prentice Hall.

AIM3143 Recommender Systems

[3 0 0 3]

Introduction: Recommender system functions, applications of recommendation systems, understanding ratings (explicit vs. implicit feedback), issues in recommender systems (sparsity, scalability, cold-start problem). **Mathematical Basic:** linear algebra concepts: Matrix representation, matrix multiplication, similarity measures (Cosine Similarity, Pearson Correlation). **Collaborative Filtering:** User-based nearest neighbor recommendation, Item-



based nearest neighbor recommendation, limitations of memory-based methods. **Model-Based Approaches:** Matrix factorization techniques (SVD, ALS), latent factor models, handling sparsity and scalability, implicit feedback models. **Neural Collaborative Filtering:** Basic concepts and architecture. **Pre-processing Techniques:** Data normalization, handling missing values, dimensionality reduction. **Security and Robustness:** Attacks on collaborative recommender systems (Shilling attacks) and basic defense mechanisms. Evaluation of collaborative filtering models. **Content-Based recommendation:** High level architecture of content-based systems, Advantages and drawbacks of content-based filtering, Item profiles, discovering features of documents, obtaining item features from tags, representing item profiles, Methods for learning user profiles, Similarity based retrieval, Classification algorithms. **Knowledge based recommendation:** Knowledge representation and reasoning, Constraint based recommenders, Case based recommenders. **Hybrid approaches:** Opportunities for hybridization, **Monolithic hybridization design:** Feature combination, Feature augmentation, **Parallelized hybridization design:** Weighted, Switching, Mixed, Pipelined hybridization design: Cascade Meta-level, Limitations of hybridization strategies. Evaluating Recommender System: Introduction, General properties of evaluation research, Evaluation designs, Evaluation on historical datasets, Error metrics, Decision-Support metrics, User-Centred metrics.

References:

1. C.C. Aggarwal, *Recommender Systems: The Textbook*, (1e), Springer, 2016.
2. N Manouselis, H. Drachsler, K. Verbert and E. Duval., *Recommender Systems for Learning*, (1e), Springer 2013.
3. F. Ricci, L. Rokach, D. Shapira and B.P. Kantor, *Recommender Systems Handbook*, (1e), Springer, 2011.

AIM3144: High Performance Computing

[3 0 0 3]

Introduction to Parallel Computing: Definition and applications of parallel computing, Evolution of High-Performance Computing, Scope of Parallel Programming Platforms, Implicit Parallelism, Trends in Microprocessor and Architectures, Limitations of Memory, System Performance, Need for HPC in AI, Data Science, and Scientific **Computing. Types of HPC systems:** clusters, grids, clouds, HPC Applications; HPC Scalable and parallel architectures – Shared memory and distributed memory; Flynn's Classification (SISD, SIMD, MISD, MIMD). **Types of Parallelism:** Data Parallelism, Task Parallelism, Instruction-Level Parallelism, Pipeline Parallelism, Challenges in HPC Systems. **Parallel Programming Models and Architecture:** Parallel programming: Principles of Parallel Algorithm Design, Decomposition Techniques, Characteristics of Tasks and Interactions. **Parallel Programming Models:** Message Passing Interface (MPI), Introduction to OpenMP, Dependency Graphs and Task Scheduling, Communication Overhead Concepts, Scalability Concepts, Granularity of Parallelism, Amdahl's Law and Gustafson's Law, Data Parallelism vs Task Parallelism, Parallel Execution Models. **GPU Computing: CUDA programming:** Introduction to CUDA programming, CPU vs GPU, Rise of GPU Computing, GPU Architecture, CUDA Architecture, Applications of CUDA, GPU Computing Fundamentals, CPU vs GPU Architecture Comparison, Role of GPUs in Deep Learning and Scientific Applications. **Performance Modelling and Optimization:** Performance Metrics for Parallel Systems: speedup, efficiency, and scalability, optimal execution time; Dense Matrix Algorithms: Matrix-Vector Multiplication, Matrix-Matrix Multiplication; Pi Calculation using MPI, Load Balancing



Concepts, Bottleneck Analysis, Throughput and Latency, Energy Efficiency in HPC, Cost–Performance Tradeoff. **HPC Applications and Case Studies:** Applications of HPC: HPC in Artificial Intelligence, HPC in Big Data Analytics, HPC in Healthcare and Bioinformatics, HPC for Climate Modeling and Weather Prediction, Future Trends in HPC. **Case Study:** HPC Facilities at MUJ – Overview of GPU-enabled HPC infrastructure, parallel computing resources, AI/deep learning applications.

References:

1. D. B. Kirk, W. W. Hwu, *Programming Massively Parallel Processors: A Hands-on Approach*, 3rd Edition, Morgan Kaufmann, 2016.
2. Ananth Grama, Anshul Gupta, George Karypis, Vipin Kumar, *Introduction to Parallel Computing*, 2nd Edition, Pearson Education, 2003.

AIM3130: Machine Learning Lab

[0 0 2 1]

Data Preprocessing and Exploration: NumPy, Pandas, Matplotlib; **Supervised Learning:** Regression Linear, Logistic; **Supervised Learning: Classification:** Decision Trees, Random Forests, Support Vector Machines, K-Nearest Neighbors, Model Evaluation and Hyperparameter Tuning, **Unsupervised Learning:** K-Means clustering, Density-Based Spatial Clustering; Dimensionality Reduction: PCA, LDA.

References:

1. Ian Goodfellow, Yoshua Bengio, and Aaron Courville – *Deep Learning*, MIT Press, 2016.
2. Ethem Alpaydin, *Introduction to Machine Learning*, MIT Press, 4th Edition, 2020.

AIM3131 Computer Networks Lab

[0 0 2 1]

Cisco Packet Tracer: Introduction to packet tracer and networking device components; **Router Mode:** Switch/Router basic commands; **Network Utilities Commands:** PING, NETSTAT, IPCONFIG, IFCONFIG, ARP, TRACE-ROUTE, NETSTAT, NSLOOKUP, PATHPING; **Designing of star topology:** using HUB and Switch; **IP configuration of end devices:** Configuring DHCP server, Static routing, RIP, OSPF, VLAN and NAT; **Network Utilities Tools:** NMAP, Wireshark, Network Scanner; **Network Programming:** Socket Programming using UDP Socket; Socket Programming using TCP Socket; **Case Study / Mini Project:** Build a Small Network and Scale to Larger Networks, Troubleshooting Scenarios

References:

1. B. A. Forouzan, *TCP/IP Protocol Suite*, (5e), Tata McGraw Hill, 2013.
2. A. S. Tanenbaum, *Computer Networks*, (5e), Pearson Education, 2010.

AIM3170: PROJECT-BASED LEARNING-3

[0 0 3 3]

Course Introduction & Project Briefing, Problem Identification & Literature Review, Project Proposal Presentation, System Design & Planning, Initial Implementation, Mid-Term Review (Progress report & feedback), Final Implementation, Documentation & Report Writing (Technical report), Final Project Presentation (Demo & viva).

References:

1. Clive L. Dym, Patrick Little, Elizabeth Orwin, *Engineering Design: A Project-Based Introduction*, (5th Edition), (2013).
2. Richard E. Fairley, *Managing and Leading Software Projects*.
3. Aurélien Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, (2022) 3rd Edition.



4. Christopher Bishop, *Pattern Recognition and Machine Learning*, (2019).
5. Charles Platt, *Make: Electronics: Learning Through Discovery*, 2nd Edition (2015).
6. Sharon J. Gerson & Steven M. Gerson, *Technical Writing: Process and Product*, (2016).
7. Jeremy Blum, *Exploring Arduino: Tools and Techniques for Engineering Wizardry*, 2nd Edition, (2019).

SEMESTER VI

AIM3201 Deep Learning

[3 1 0 4]

Introduction: History of Deep Learning, Deep Learning applications, Feed forward Neural networks, Gradient descent and the back propagation algorithm, vanishing/Exploding gradient problem. Various activation functions, ReLU Heuristics for avoiding bad local minima, Optimizers: Gradient Descent (GD), Momentum Based GD, Nesterov Accelerated GD, Stochastic GD, ADAM, AdaGrad, RMSprop. **Convolutional Neural Networks:** Architectures, convolution / pooling layers, LeNet, AlexNet, EfficientNet, VGGNet, GoogLeNet, ResNet, DenseNet; Regularization, Dropout, Early stopping, Dataset augmentation, Dropout; Better weight initialization methods, Batch Normalization. **Recurrent Neural Networks:** Vanishing/exploding gradients in RNNs LSTM, GRU. **Segmentation** Semantic Segmentation, Instance Segmentation, FCN, Unet Object Localization, Region Proposal Networks, RCNN, RFCN, YOLO; Encoder Decoder Models. **Advanced Deep Learning Concepts:** Boltzmann machine, Auto encoder, Regularization in auto encoders, Denoising auto encoders, Sparse auto encoders Autoencoders, Variational Auto-encoders, Generative Adversarial Networks, Attention Mechanism, Attention over images and Transformer.

References:

1. Goodfellow, I., Bengio, Y., and Courville, A., *Deep Learning*, (1e), MIT Press, 2016.
2. Chollet, F., *Deep Learning with Python*, (1e), Manning Publications, 2017.
3. Géron, A., *Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow*, (2e), O'Reilly Media, 2019

AIM3240: Computer Vision

[3 0 0 3]

Introduction of computer vision, Image formation, geometric transformation and image processing, point operations, linear and non-linear filtering, sampling, aliasing, transform filtering techniques, interpolation and multi resolution, **Feature Detection and Matching:** human color perception and inference from color feature descriptor, matching, edge detection, lines detection, Hough transform, object recognition, SHIFT and HOG feature. **Camera Model:** pinhole model, camera calibration, single and multiple view geometry, stereovision, shape detection and segmentation: active contours, snakes, region split and merge, graph cut and energy-based methods, medical image segmentation. **Object Detection:** The Sliding Window Method, Face Detection, Detecting Humans, Detecting Boundaries. Motion Estimation: parametric motion, spline-based motion, optical flow, Kalman filtering, application of motion estimation to video stabilization, medical image registration. **Application of CV:** Texture classification and recognition, Activity recognition, Computational Photography, and super-resolution.

REFERENCE BOOKS

1. Gonzalez and Woods, *Digital Image Processing*, (4e), Pearson Education, 2017.
2. Joseph Howse, Joe Minichino, *Learning OpenCV 4 Computer Vision with Python 3*, (3e), Packt Publishers, 2020.

AIM3241: Natural Language Processing

[3 0 0 3]

Introduction: Overview of NLP and its applications in real-world tasks, Challenges in multilingual NLP. **Study of Words:** Frequency N-grams, Word alignment in parallel corpora - length based, word based. Word Sense Disambiguation – Supervised, Unsupervised, and Pretrained embedding-based approaches. **Grammar:** Hidden Markov Model, POS tagging. Parsing techniques – Dependency parsing, Constituency parsing. **Text Preprocessing and Normalization:** Regular expressions, Tokenization, lemmatization, stop word removal and stemming. Subword tokenization (BPE, SentencePiece), Text cleaning for noisy social media/text data. **Word Embeddings:** Understanding word representation techniques like Bag of words, TF-IDF, word2vec, one hot encoding and GLOVE. Classical embeddings (BoW, TF-IDF, word2vec, GloVe) and Contextual embeddings. **Attention Mechanisms and Transfer Learning:** Transformer-based models like BERT, Roberta. Transformer-based models (BERT, RoBERTa, GPT family, T5). **Applications in Sentiment Analysis, Question Answering, Conversational AI (Chatbots), and Information Retrieval.**

References:

1. S Andreas, *NLP: New Technology (1e)*, William Morrow, 2011
2. Raymond Lee, *Natural Language Processing: A Textbook with Python Implementation (2e)*, Springer, 2025.
3. Hobson Lane and Maria Dyshel, *Natural Language Processing in Action (2e)*, Manning Publications, 2025.
4. Yannis Haralambous, *A Course in Natural Language Processing*, Springer, 2024.
5. Andrew Freed, Cari Jacobs, and Eniko Rozsa, *Effective Conversational AI*, Manning Publications, 2025.

AIM3242: Opinion mining and Sentiment Analysis

[3 0 0 3]

Introduction: Basics of Opinion Mining and Sentiment Analysis and classification, NLP foundations, applications in e-commerce, politics, healthcare, education, research trends, subjective vs objective text, comparative and aspect-based opinions. **Text Processing & Sentence Level Analysis:** Tokenization, stop-word removal, stemming, lemmatization, POS tagging, sentence subjectivity detection, sentence-level polarity classification, negation handling, sentiment composition. **Aspect Based Sentiment Analysis:** Aspect and entity extraction, feature selection, opinion target extraction, supervised and unsupervised approaches, rating prediction, review analytics. **Sentiment Lexicons & Advanced Mining:** Dictionary-based and corpus-based lexicon generation, word embeddings, comparative opinion mining, debate and stance analysis, abnormal pattern discovery, intent mining, review quality prediction.

References:

1. F. Pozzi, E. Fersini, E. Messina, and B. Liu, *Sentiment Analysis in Social Networks*. Morgan Kaufmann, 2016.
2. Moreno and C. A. Iglesias, *Sentiment Analysis for Social Media*. MDPI, 2020.
3. Sharaff, G. R. Sinha, and S. Bhatia, *New Opportunities for Sentiment Analysis and Information Processing*. IGI Global, 2021.
4. D. S. Rajput, R. S. Thakur, and S. M. Basha, *Sentiment Analysis and Knowledge Discovery in Contemporary Business*. IGI Global, 2018.

AIM3243: Artificial Intelligence in Cybersecurity

[3 0 0 3]



Fundamentals of Cyber Security: Cryptography, security services, attacks, Mechanisms. Basic Cryptographic algorithms, Symmetric (DES, AES & Stream Cipher) vs. Asymmetric key cryptography techniques (RSA, Diffie-Hellman & ECC). Types of cybercrimes and classifications, Legal Perspectives and Case Studies. **Fundamentals of AI for Security:** Modern AI tools to mitigate the Cyber Attacks- Intrusion Detection Systems, Network traffic analysis, Behavioural analytics, and Real-time threat monitoring; **Web Application Security:** Injection, Broken authentication, Sensitive data exposure, XML External Entities (XXE), Broken access control, Security misconfiguration, Cross-Site Scripting (XSS). **Secure Web:** Handshake mechanisms of HTTPS and TLS protocols, PKI, and Secure key exchange. **Deep learning applications:** Deep learning fundamentals from a security perspective, Pattern and Anomalous behaviour detection techniques, Convolutional Neural Networks (CNNs) for malware detection, Recurrent Neural Networks for sequence-based attacks and Autoencoders for anomaly detection) and **Case studies;** Secure AI Development: foundations of secure software design, secure programming, and security testing. A basic understanding of Application Programming Interface (API) and example APIs of companies referred to are: Darktrace, Vectra and Cylance; Adversarial AI & Ethical Issues: AI-based attacks as deepfakes, Evasion techniques, Bias and fairness in AI security. **Impact of AI on Cyber Security:** Understanding through message authentication, Secure hash algorithms, Threat hunting in memory, network mapping, packet sniffing, and firewall configurations.

Reference

1. L.F. Sikos (Ed.), *AI in Cybersecurity*, Springer International Publishing, 2019
4. E. Tsukerman, *Machine Learning for Cybersecurity Cookbook*, Packt Publishing, 2019
2. S. Halder, *Hands-On Machine Learning for Cybersecurity: Safeguard your system by making your machines intelligent using the Python ecosystem*, (1e), Packt Publishing, 2018

AIM3244 Explainable AI

[3 0 0 3]

Introduction: Foundations of Explainable **AI: Motivation:** Black-box vs. Glass-box models, Definitions, stakeholders, ethical and legal aspects, Taxonomy of XAI methods: global vs. local, model specific vs. model-agnostic, post-hoc vs. ante-hoc, Responsible AI principles, AI governance frameworks, transparency regulations (GDPR, EU AI Act, India DPDP Act), **Human Factors in Explainability:** Interpretability & cognitive aspects, Trust calibration and human-in-the-loop explanations, Evaluation with end users through usability studies, User-centered explanation design, fairness perception, interactive explanation dashboards, **Inherently Interpretable Models:** Decision trees, rule-based models, generalized additive models (GAMs), case-based reasoning, risk scores, linear models, monotonic neural networks, Explainable Boosting Machines (EBMs), Post-hoc Explanation Methods and its Architecture: Feature attribution (LIME, SHAP, Grad-CAM, Integrated Gradients), Concept-based methods (TCAV, ACE), Sample-based methods (Influence functions, ProtoPNet), Counterfactual explanations, Anchors, Partial Dependence Plots (PDP), Accumulated Local Effects (ALE), attention visualization, **Interpretable Architectures:** Concept bottleneck models (CBMs), SENNs, interpretable variants of CNN/RNN/Transformers, **Evaluation of Explanations:** Faithfulness, stability, completeness, Sanity checks, consistent issues in explanations,

disagreement problem, **Applications and Case Studies:** Healthcare, finance, autonomous driving, recommendation systems, criminal justice.

References

1. Molnar, C., *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable*, (2e), Leanpub, 2022.
2. Taly, A., Kim, B., & Sayres, R., *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, Springer, 2021.
3. Rothman, D., *Explainable AI with Python: Interpret, Visualize and Explain Deep Learning Models*, (2e), Packt Publishing, 2022.

AIM3245: Generative AI

[3 0 0 3]

Introduction to Generative AI: Definition and scope of Generative AI, Historical evolution and milestones. Applications in text, image, audio, video, and real-time generative applications (e.g., chatbots, copilots, digital twins). Introduction to Foundation Models and Large Language Models (LLMs). Prompt Engineering basics and design principles. **Mathematical and Deep Learning Foundations:** Probability and Statistics for Generative Models, Linear Algebra and Matrix Operations, Neural Networks and Backpropagation with focus on deep architectures used in generative models, Loss functions for generative tasks, Optimization techniques (SGD, Adam, RMSprop). **Generative Models:** Variational Autoencoders (VAEs), Generative Adversarial Networks (GANs), DCGAN, Conditional GAN, CycleGAN, StyleGAN, Diffusion Models (DDPMs, Stable Diffusion, Imagen), Transformer-based Generative Models (GPT, BERT for generation, DALL·E). Retrieval-Augmented Generation (RAG). Fine-tuning techniques: LoRA, PEFT. Multimodal models. **Tools, Frameworks, and APIs:** TensorFlow and PyTorch for model building, Hugging Face Transformers and Diffusers, **APIs:** OpenAI, Stability AI, Replicate with integration in applications, Dataset preprocessing pipelines and data augmentation. **Evaluation and Applications:** Evaluation metrics: FID, IS for image generation, BLEU, ROUGE, METEOR for text generation, Case studies: Ablation in GANs, VAEs, and Transformer-based models.

References

1. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). *Generative adversarial nets*. *Advances in Neural Information Processing Systems*, 27.
2. Kingma, D. P., & Welling, M. (2014). *Auto-Encoding Variational Bayes*. arXiv preprint arXiv:1312.6114.

AIM3230 Deep Learning lab

[0 0 2 1]

Introduction: Exploration of **feedforward neural networks**, including various **activation functions** and the application of **gradient descent** and **backpropagation**. implement different **optimizers** such as **ADAM**, **RMSprop**, and **Stochastic Gradient Descent (SGD)** to address optimization challenges. **Convolutional Neural Networks (CNNs)** by building and analysing classic architectures like **LeNet**, **AlexNet**, and **ResNet**, while also applying **regularization techniques** like **dropout** and **batch normalization**. **Recurrent Neural Networks (RNNs)**, specifically implementing **LSTMs** and **GRUs** to manage sequence data and address the vanishing/exploding gradient problem. **semantic and instance segmentation, object**

localization using models like **YOLO** and **RCNN**, and a study of **encoder-decoder models**, **autoencoders**, **Generative Adversarial Networks (GANs)**, and the **Attention Mechanism** to understand their practical applications and internal workings.

References:

1. Charu C. Aggarwal, *Neural Networks and Deep Learning: A Textbook*, Springer, 2023.
2. Vishnu Sharma, Somnath Dey, *Practical Deep Learning for Cloud, Mobile, and Edge, 2nd Edition*, O'Reilly Media, 2023.
3. Palash Goyal, Sumit Pandey, Karan Jain, *Deep Learning for Natural Language Processing, 2nd Edition*, Apress, 2023.
4. Satya Mallick & Tzutalin, *Practical Computer Vision with Deep Learning*, Packt Publishing, 2024.
5. Recent Research Articles (2022–2025) from IEEE TPAMI, CVPR, NeurIPS, ICLR, ACM Computing Surveys on CNNs, Transformers, GANs, and Attention mechanisms.

AIM3202 Professional Practices

[0 0 2 1]

Professional Ethics: Philosophy of Ethics, Computing Profession, Computing Ethics, Strengthen Professional Ethics, ethical issues in AI and Machine Learning (AI Governance), and sustainability, Sustainable Development and Green Computing, ethical and social responsibilities in sustainable technology practices and Green IT initiatives. **Computing Professions and Laws:** The Structure of Organizations, Finance and Accounting, Anatomy of a Software organization, roles in computing professions, and workplace ethics. **Legal Aspects:** Computer Contracts, Intellectual Property Rights, The Framework of Employee Relations Law, Human Resource Management and IT, Health and Safety at Work, Software Liability, Computer Misuse and the Criminal Law, cyber laws Regulation of personal data. **Code of Conduct and Ethics:** Overview of the British Computer Society Code of Conduct, General Research Ethics and ethical principles, and responsible AI practices. Accountability and Auditing, Accountability in computing, IT auditing, risk management, ethics in social media, AI misuse and deepfakes.

Reference

1. J. Kizza, *Ethical and Social Issues in the Information Age*, 6th Edition, Springer, 2017, ISBN: 978-3319707112

AIM3231: Emerging Tools and Technologies Lab

[0 0 2 1]

Introduction: Microsoft Power BI installation, workspace layout, data import from Excel/CSV/SQL. Data Transformation with Power Query: cleaning, reshaping, merging queries, and custom column creation. DAX Fundamentals & Report Building: calculated columns, measures, KPI cards, slicers, drill-through, and publishing to Power BI Service. **Jira Fundamentals:** creating a Scrum project, defining epics, stories, and tasks, setting up the backlog. Sprint Planning and Agile Boards: running a sprint, using Kanban boards, tracking velocity and burndown charts. Advanced Jira: custom workflows, JQL queries, Jira reports, and integration with Confluence and GitHub. **Microsoft Azure Fundamentals & Portal**



Navigation: subscriptions, resource groups, Azure CLI basics, creating VMs and Storage Accounts. **Azure App Service & Databases:** deploy a web app, configure Azure SQL Database and Cosmos DB, set up Azure Monitor alerts. **Azure DevOps & CI/CD Pipelines:** connect repos, create build and release pipelines, integrate with containerized workloads. **Docker Fundamentals:** installing Docker, pulling images, running containers, and understanding layers and namespaces. **Building Custom Images & Dockerfile:** write Dockerfiles, build and tag images, push to Docker Hub registry. **Docker Compose & Multi-Container Apps:** define multi-service stacks with compose, networking, volumes, and scaling. **Overleaf LaTeX Basics in Overleaf:** document structure, math typesetting, figures, tables, bibliography management with BibTeX. **Collaborative Technical Report:** template-based report authoring, reviewer comments, version history, and export to PDF.

References:

1. G. Deckler and B. Powell, *Mastering Microsoft Power BI: Expert Techniques to Create Interactive Insights for Effective Data Analytics and Business Intelligence*, 2nd ed. Birmingham, U.K.: Packt Publishing, 2022.
2. P. Li, *Jira 8 Essentials: Effective Issue Management and Project Tracking with the Latest Jira Features*, 5th ed. Birmingham, U.K.: Packt Publishing, 2019.
3. J. Cheshire, *Exam Ref AZ-900 Microsoft Azure Fundamentals*, 3rd ed. Redmond, WA, USA: Microsoft Press, 2022.
4. N. Poulton, *Docker Deep Dive: Zero to Docker in a Single Book*, 2025 ed. Independently published, 2025.
5. S. P. Kane and K. Matthias, *Docker: Up and Running—Shipping Reliable Containers in Production*, 3rd ed. Sebastopol, CA, USA: O'Reilly Media, 2023.

AIM3270: PROJECT-BASED LEARNING - 4

[0 0 6 3]

Course Introduction & Project Briefing (Team formation, guidelines), Problem Identification & Literature Review (Research, idea validation), Project Proposal Presentation (Proposal submission & peer review), System Design & Planning (Architecture, tools, timeline), Initial Implementation (Prototyping), Mid-Term Review (Progress report & feedback), Final Implementation (Testing, debugging), Documentation & Report Writing (Technical report), Patent/Conference paper and research paper writing, Final Project Presentation (Demo & viva)

References:

1. "Engineering Design: A Project-Based Introduction" (5th Edition), (2013) By Clive L. Dym, Patrick Little, Elizabeth Orwin
2. "Managing and Leading Software Projects" By Richard E. Fairley
3. "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow" (2022) 3rd Edition) By Aurélien Géron
4. "Pattern Recognition and Machine Learning" (2019) By Christopher Bishop
5. "Make: Electronics: Learning Through Discovery" (2nd Edition) (2015) By Charles Platt
6. "Technical Writing: Process and Product" (2016) By Sharon J. Gerson & Steven M. Gerson
7. "Exploring Arduino: Tools and Techniques for Engineering Wizardry" (2019) (2nd Edition) By Jeremy Blum



AIM4140: BigData Analytics

[3 0 0 3]

Introduction to Big Data: Types of Digital Data-Characteristics of Data, Evolution, Definition, Challenges, 3Vs, non-definitional traits, Business Intelligence vs. Big Data, Data warehouse and Hadoop environment, Coexistence. **Big Data Analytics:** Classification of analytics, Data Science, Terminologies, CAP Theorem, BASE Concept. **Introduction to NoSQL Database:** Types of Databases, Advantages, Comparison of NewSQL, SQL vs. NOSQL vs NewSQL. **Introduction to Hadoop:** Features, Advantages, Versions, Overview of Eco-systems, Distributions, Hadoop vs. SQL, RDBMS vs. Hadoop, Components, Architecture, HDFS. Map Reduce: Mapper, Reducer, Combiner, Partitioner, Searching, Sorting, Compression. Hadoop 2 (YARN): Architecture, Interacting with Hadoop Eco systems. **Apache Spark:** Introduction to Apache Spark, Spark architecture and components, Spark vs MapReduce, Resilient Distributed Datasets (RDDs), Transformations and Actions, Spark SQL, Spark Data Frames and Datasets. **MongoDB:** Introduction, Features, Data types, MongoDB Query language, CRUD operations, MongoDB Import & Export. **Cassandra:** Introduction, Features, Data types, CQLSH, Key spaces, CRUD operations, Collections, Counter, TTL, alter commands, Import and Export, Querying System tables. **Hadoop Eco systems:** Hive introduction and architecture; data types and file formats; Hive Query Language (HQL); SerDe concepts; user-defined functions; Hive integration with Hadoop ecosystem.

References:

1. T. Erl, W. Khattak, and P. Buhler, *Big Data Fundamentals: Concepts, Drivers & Techniques*, Upper Saddle River, NJ, USA: Prentice Hall, 2016.
2. S. Acharya, *Big Data and Analytics*. Wiley India Pvt. Ltd., 2015.
3. V. Prajapati, *Big Data Analytics with R and Hadoop*. Birmingham, U.K.: Packt Publishing Ltd., 2013.
4. A. Holmes, *Hadoop in Practice*, 2nd ed. Shelter Island, NY, USA: Manning Publications, 2015.
5. Bill Chambers & Matei Zaharia, "*Spark: The Definitive Guide - Big Data Processing Made Simple(Greyscale Indian Edition)* ", O'Reilly Media, Feb 2018
6. Z. Zhang, *Big Data Analytics: Methods and Applications*, Cham, Switzerland: Springer, 2024.

AIM4141: Human Computer Interaction

[3 0 0 3]

Introduction: Importance of User Interface, good design, history of screen design. Graphical user interface – concept of direct manipulation, characteristics. Web user interface – popularity, principles of user interface design. **Human interaction with computers** – human characteristics and considerations, interaction speeds, understanding business functions. Principles of user interface design. Usability engineering basics. **Screen Design & Components:** goals, screen planning and purpose, organization, navigation and flow, visually pleasing composition, focus and emphasis, Color theory, multimedia elements, window design and navigation schemes, device-based and screen-based controls. **HCI in Software Development life cycle:** iterative design and prototyping, design rules and standards, golden rules and heuristics, HCI patterns, design rationale. **Design Evaluation techniques:** goals, expert analysis vs user participation, choosing evaluation methods. Universal design principles, accessibility considerations, multi-modal interaction



basics. **Emerging Topics & Visualization:** Overview of cognitive models (GOMS), virtual reality basics, augmented reality applications information and data visualization principles, current trends and case studies.

References:

1. Dix, J. Finlay, G. D. Abowd, and R. Beale, *Human–Computer Interaction*, 3rd ed. Harlow, U.K.: Pearson Education, 2004.
2. D. R. Olsen, *Human–Computer Interaction*, 2nd ed. Boston, MA, USA: Cengage Learning, 2012.
3. W. O. Galitz, *The Essential Guide to User Interface Design*, 3rd ed. New Delhi, India: Wiley DreamTech, 2007.
4. J. Preece, Y. Rogers, and H. Sharp, *Interaction Design: Beyond Human–Computer Interaction*, 4th ed. Hoboken, NJ, USA: Wiley, 2015.
5. S. Lauesen, *User Interface Design*. Harlow, U.K.: Pearson Education, 2005.
6. Y. Rogers, J. Preece, and H. Sharp, *Interaction Design: Beyond Human–Computer Interaction*, 6th ed. Hoboken, NJ, USA: Wiley, 2023.

AIM4142: Machine Learning Operations(MLOps)

[3 0 0 3]

Introduction to MLOps: MLOps Fundamentals, comparison with DevOps, ML lifecycle vs software lifecycle, Key challenges in ML operations, Overview of MLOps platforms (MLflow, Kubeflow, TFX, MetaflowSeldon). **Common ML frameworks:** Scikitlearn, TensorFlow, PyTorch **Data and Model Versioning:** Data versioning tools (DVC, LakeFS, Pachyderm), Model versioning and experiment tracking (MLflow, Weights & Biases), Introduction to Feature stores and metadata management. **CI/CD for ML system:** Principles of Continuous Integration and Continuous Deployment, CI/CD tools for ML (GitHub Actions, Jenkins), Automated testing and retraining workflows, Managing reproducibility. **Software Development Paradigms** i.e., OOP, YAML language, software testing, code refactoring **Model Deployment & Monitoring:** Flask, FastApi, TensorFlow serving, TensorFlow lite for optimization latency. Model packaging: Docker, Conda env, Deployment strategies: REST APIs, batch, streaming, Monitoring performance, drift detection, Tools: Prometheus, Grafana, Evidently, Why Logs. **MLOps in the Cloud:** MLOps on AWS, Azure, GCP, ML pipeline automation using Kubeflow or Airflow, Model governance, fairness, and security, AutoML.

Reference:

1. E. Raj, *Engineering MLOps*. Birmingham, U.K.: Packt Publishing, 2021.
2. M. Treveil, N. Omont, C. Stenac, K. Lefevre, D. Phan, J. Zentici, and A. Moreau, *Introducing MLOps*. Sebastopol, CA, USA: O'Reilly Media, 2020.
3. H. Hapke and C. Nelson, *Building Machine Learning Pipelines: Automating Model Life Cycles with TensorFlow*. Sebastopol, CA, USA: O'Reilly Media, 2022.
4. K. Kapelonis, *Kubernetes for Machine Learning: Implement Machine Learning Workflows on Kubernetes*. Birmingham, U.K.: Packt Publishing, 2022.
5. N. Gift and A. Deza, *Practical MLOps*. Sebastopol, CA, USA: O'Reilly Media, 2021.

AIM4143: Reinforcement Learning

[3 0 0 3]

Introduction: Basics of probability and linear algebra, Definition of a stochastic multi-armed bandit, Definition of regret, Achieving sublinear regret, UCB algorithm, KL-UCB, Thompson Sampling. **Markov Decision Processes and Dynamic Programming:** Markov Decision Problem, policy, and value function, Reward models (infinite discounted, total, finite horizon, and average), Episodic & continuing tasks, Bellman's optimality operator, and Value iteration & policy iteration. **Foundations of Reinforcement Learning:** The Reinforcement Learning problem, prediction and control problems, Model-based algorithm, Monte Carlo methods for prediction, and Online implementation of Monte Carlo policy evaluation Bootstrapping; **Temporal Difference Learning, Model Free Control and Function Approximation:** TD(0)



algorithm; Convergence of Monte Carlo and batch TD(0) algorithms; Model-free control: Q-learning, Sarsa, Expected Sarsa. n-step returns; TD(λ) algorithm; Need for generalization in practice; Linear function approximation and geometric view; Linear TD(λ). **Function Approximation, Policy Search, and Practical Algorithms Policy:** Tile coding; Control with function approximation; Policy search; Policy gradient methods; Experience replay; Fitted Q Iteration; Case studies.

References:

1. R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. Cambridge, MA, USA: MIT Press, 2020.
2. M. Sugiyama, *Statistical Reinforcement Learning: Modern Machine Learning Approaches*. Boca Raton, FL, USA: CRC Press, 2015.
3. B. Belousov, H. Abdulsamad, P. Klink, S. Parisi, and J. Peters, *Reinforcement Learning Algorithms: Analysis and Applications*. Cham, Switzerland: Springer, 2021.
4. A. Zai and B. Brown, *Deep Reinforcement Learning in Action*. Shelter Island, NY, USA: Manning Publications, 2020.
5. A. Lazaric and M. Ghavamzadeh, *Foundations of Reinforcement Learning*. Cham, Switzerland: Springer, 2023.

SEMESTER VIII

AIM4270: Major Project

[0 0 0 12]

Introduction In this course, students are required to select and carry out a major project based on a topic of their interest in emerging areas of Computer Science and Engineering, including Artificial Intelligence, Machine Learning, Data Science, IoT, Cyber Security, Cloud Computing, and Software Engineering. The project work will involve problem identification, literature survey, system design, implementation, testing, documentation, and presentation. Periodic evaluations will be conducted by the project supervisor to monitor the progress and implementation of the work. This project, initiated in the eighth semester, will be evaluated through both internal and external assessment processes.

References:

1. Research papers, whitepapers, technical reports, patents, open-source documentation, IEEE/ACM digital library resources, and industry standards relevant to the selected project domain.