Course Description
M. Tech. In Computer Science and Engineering

Title of Course: Advanced Computer Networks **Course Code: CS501**
L-T Scheme: 3-0 **Course Credits: 3**

Prerequisite: Students must have knowledge of Programming, Operating System, and Computer Networks.

Objectives: To be able to design, configure, and edit network structures and protocols.

Learning Outcomes: After completion of this student will be able to:

Course Outcome	Description
CO1	Comprehend the responsibility of each layer of TCP/IP reference model.
CO2	Describe issues in internet protocols.
CO3	Demonstrate understanding of advanced routing protocols.
CO4	Apply congestion control methods and to provide solution to related issues.
CO5	Analyze MPLS, VPN, and SDN protocols.

Course Contents:

Unit-I: Reference models the Open system interconnect and the TCP/IP. Need of protocols at each layer.
Unit-II: IPv4 and IPv6 – features, addressing, and transition mechanisms**,**

Unit-III: Routing inside and outside of networks, advanced routing protocols, IGP and EGP. QoS-aware routing: metrics and algorithms.

Unit-IV: TCP Congestion control (AIMD, TCP Reno, TCP Vegas, BBR, etc.), TCP variants and performance tuning, Flow and error control mechanisms, SCTP, QUIC – modern transport protocols, Multipath TCP (MPTCP).

Unit-V: MPLS architecture, label switching, traffic engineering. VPNs (Site-to-site, Remote Access), IPSec, GRE tunneling. NAT, PAT, and IPv6 tunneling techniques, Software-Defined Networking (SDN): architecture, OpenFlow. Network Function Virtualization (NFV)

Evaluation Scheme:

Exams	**Marks**	**Coverage**
Test-1	15 Marks	Syllabus covered upto Test-1
Test-2	25 Marks	Syllabus covered upto Test-2
Test-3	35 Marks	Full Syllabus
Assignment	10 Marks	
Tutorials	5 Marks	
Quiz	5 Marks	
Attendance	5 Marks	
Total	**100 Marks**	

References

1. **Computer Networks** by Andrew S. Tanenbaum & David Wetherall.
2. Michael A. Gallo, William M. Hancock, Computer Communication and Networking Technologies, Thomson Press, 2002.
3. Dimitri Bertsekas, Robert Gallager, Data Networks, second edition, PHI, 1992.
4. **Computer Networking: Principles, Protocols and Practice** – Olivier Bonaventure.
5. RFCs – especially for TCP, BGP, MPLS, QUIC, etc.

Course Description

Title of Course: Advanced Database Systems **Course Code: CS502**
L-T Scheme: 3-0 **Course Credits: 3**

Prerequisite: Student must have knowledge about Database Management Systems.

Objectives: To develop the ability to design, implement and manipulate databases as well as to build Database management systems.

Learning Outcomes:

- Ability to build normalized data bases made a secure application for real time systems.
- Ability to design systems by using Models.
- Ability to develop skills of writing applications by using SQL and PL SQL.
- Ability to understand query optimization techniques.
- Understanding of transaction processing.
- Ability to handle recovery and concurrency issues in the database.
- Ability to design web services will be use in different applications.

Course Contents:

Simple database applications and Users: Single user, Multi User, Review of relational databases- EER model, object oriented model, relational algebra, Transforming data models into database design, determination of data requirements for databases. SQL for database processing, advanced SQL, Ranked queries, joins among relations, advanced database manipulation using SQL and PL SQL.

Query Processing and Optimization: Evaluation of Relational Operations, Transformation of Relational Expressions, Indexing and Query Optimization, Data access from disk, Index based access, Sort and Join Processing, Physical plan selection, Limitations of Relational Data Model;

Distributed Query Processing and Optimization: Distributed Transaction Modeling and Concurrency Control, Distributed Deadlock, Commit Protocols, Design of Parallel Databases, Parallel Query Evaluation. Parallel and Distributed Databases: Distributed Data Storage, Fragmentation & Replication, Location and Fragment Transparency.

Advanced Transaction Processing: Nested and Multilevel Transactions, Compensating Transactions and Saga, Long Duration Transactions, Weak Levels of Consistency, Transaction Work Flows, And Transaction Processing Monitors.

Secured Database applications: Database Security - Database security risks, database users and privileges. Authentication, authorization, access control methods, data encryption, Database Recovery- Failure classifications, recovery and atomicity, log based recovery, shadow paging, recovery with concurrent transactions, Buffer management, Multimedia and Mobile databases: Multimedia databases, web-enabled database, temporal databases, Mobile databases.

Evaluation Scheme:

Exams	**Marks**	**Coverage**
Test-1	15 Marks	Syllabus covered upto Test-1
Test-2	25 Marks	Syllabus covered upto Test-2
Test-3	35 Marks	Full Syllabus
Assignment	10 Marks	
Tutorials	5 Marks	
Quiz	5 Marks	
Attendance	5 Marks	
Total	**100 Marks**	

Text Book

1. R. Ramakrishnan, J. Gehrke, Database Management Systems, McGraw Hill, 2004
2. Database System Concepts" – *Abraham Silberschatz, Henry F. Korth, and S. Sudarshan Publisher: McGraw-Hill*

References

1. "Fundamentals of Database Systems" – *Ramez Elmasri and Shamkant B. Navathe Publisher: Pearson.*
2. "Oracle PL/SQL Programming" – *Steven Feuerstein Publisher: O'Reilly Media.*
3. "Learning PHP, MySQL & JavaScript" – *Robin Nixon Publisher: O'Reilly Media.*
4. "Information Security: Principles and Practice" **–** *Mark Stamp Publisher: Wiley.*
5. "Mobile Database Systems" – *Vijay Kumar Publisher: Wiley-Interscience.*
6. "Multimedia Database Management Systems" **–** *B. Prabhakaran Publisher: Springer.*

Course Description

Title: Research Methodology and IPR Code: HS520
L-T-P scheme: 2-0-0 **Credits: 2**

Prerequisite: There is no prerequisite for this course.

Objective: This course is designed to enable students:
1. To identify the issues and concepts salient to the research.
2. To understand the overall processes of designing a research study from its inception to its report.
3. To use basic techniques of quantitative and qualitative research.
4. To gain knowledge to develop data analytics skills and meaningful interpretation to the data sets so as to solve the research problem.
5. To conduct the research work and draft research articles, synopsis and report.

Learning Outcomes:

Course Outcome	Description
CO1	Develop understanding of basic concepts of research and its methodologies, various kinds of research, objectives of doing research, research process, and ethical dimensions of conducting research.
CO2	Select and define appropriate research problem & identify various sources of information for literature review and data collection.
CO3	Outline various concepts of research design, sample design, measurement and scaling techniques, IPR and their application.
CO4	Formulation of research hypotheses and applying various techniques of data analysis for hypothesis testing with the aspects of IPR
CO5	Describe the real-world problems using basic techniques of descriptive and inferential data analysis and to reach a conclusion based on the data analysis, and to make and defend a recommended course of action.

Course Outline

1. Introduction to Research Methodology: Meaning of Research, Objectives of Research, Motivations in Research, Types and Approaches of Research, Significance of Research, Research Methods v/s Methodology, Research and Scientific Methods, Research Process, Criteria of Good Research.

2. Defining the Research Problem: What is Research Problem?, Selecting the Problem, Necessity of and Techniques in defining the problem

3. Research design: Basic research process, Meaning and necessity of research design, Features of good research design, Different research designs, Basic principles

4. Sample Design: Sample survey, Implications of a sample design, Steps in sampling design, Need for sampling, Important sampling distributions, Criteria of selecting a sampling procedure, Characteristic of a good sample design, Different types of samples design

5. Measurement and scaling techniques: Measurement in research, Measurement scales, Sources of errors in measurement, Techniques of developing measurement tools, Scaling, Scale classification bases, important scaling techniques

6. Methods of Data Collection and Presentation: Various methods of Primary and secondary data collection Selection of appropriate method for data collection; Case Study Method, Guidelines for developing questionnaire, successful interviewing. Survey v/s experiment, Methods of Data presentation

7. Processing and analysis of data: Frequency distributions, Measure of central tendency, Measures of dispersion , Simple regression analysis, Multiple correlation and regression, Estimation of population means, Sample size and its determination, Chi- square test, Analysis of Variance and Co-variance

8. Research Ethics: Ethics introduction, ethics in research work, concept of similarity, Plagiarism, Academic Integrity, Departmental Academic Integrity Panel (DAIP), Institutional Academic Integrity Pane (IAIP)

9. **Intellectual property rights:** Intellectual Property – The concept of IPR, Evolution and development of concept of IPR, IPR development process, Trade secrets, utility Models, IPR & Bio diversity, Role of WIPO and WTO in IPR establishments, Right of Property, Common rules of IPR practices, Types and Features of IPR Agreement, Trademark, Functions of UNESCO in IPR maintenance., Trademark, Patents.

Evaluation Scheme:

Exams	Marks	Coverage
Test-1	15 Marks	Based on Unit-1 & Unit-2
Test-2	25 Marks	Based on Unit-3 & Unit-4 and around 30% from coverage of Test-1
Test-3	35 Marks	Based on Unit-5 and around 30% from coverage of Test-2
Assignment	10 Marks	
Tutorials	5 Marks	
Quiz	5 Marks	
Attendance	5 Marks	
Total	**100 Marks**	

Text Book

- Research Methodology-Methods and Techniques by C.R. Kothari; New Age publication.
- Research Methodology by R.Panneerselvam ; PHI publication.
- Research Methodology by P.K. Sharma
- H.A. Taha. (1992) Operations Research- An Introduction, New York: Macmillan.
- F. S. Hiller and G. J. Liberman. Introduction to Operations Research, San Francisco: Holden- Day.
- Intellectual Property: A Very Short Introduction, Siva Vaidhyanathan
- Intellectual Property: The Law of Trademarks, Copyrights, Patents, and Trade Secrets" Deborah E. Bouchoux

Course Description

Title of Course: High Performance Computer Architecture **Course Code: CS503**
L-T Scheme: 3-0 **Course Credits: 3**

Prerequisite: Student must have knowledge about the hardware and software component about the computer and basic of computer architecture and parallel computing.

Objectives: The main aim of this course is intended to develop, implement, and debug assembly language programs that meet stated specifications. To explain bus transactions, memory organization and address decoding, basic I/O interfaces and port addressing. To understand how to control components of a computer system through the use of hardware and software interrupts.

At the conclusion of the course, following learning objectives are expected to be achieved:

1. You will broaden your knowledge of computer system organization and Architectures.
2. You will gain knowledge in technical aspects of computer system design.
3. You will gain understanding of computer arithmetic both integer and floating point.
4. You will acquire the background for understanding next-generation CPUs.
5. You will learn a computer programming model at a level that enables you to write assembly language programs for the processor meeting given specifications.

Learning Outcomes: The students shall acquire the generic skills to know about the all aspect of computer like working of memory, memory calculations, design and implement of computer architecture along with analysis of practical aspects.

Course Outcome	**Description**
CO1	Implement and simulate VHDL models using behavioral, dataflow, structural, and mixed modeling approaches.
CO2	Analyze the functional organization of computer systems, including pipelining, RTL, memory hierarchy, and performance metrics.
CO3	Evaluate modern performance enhancement techniques such as branch prediction, dynamic scheduling, and multithreading.
CO4	Apply and assess multiprocessing and parallel architectures including Amdahl’s Law, cache coherence, and GPU-based computing.
CO5	Explain and compare distributed computing paradigms including grid, cluster, and cloud computing systems.

Course Contents:

- **Introduction to VHDL:** Implement basic VHDL constructs, Implement modeling structures of VHDL: Behavioral, dataflow, structural, mixed design, use VHDL building blocks: entity architecture, subprograms, package, package declarations, package body, test –Bench, State machine modeling, fault analysis and hazard detection.
- **Functional Organization:** Review the concepts of Computer Architecture – RTL, Micro program, Pipelining and ILP, Memory and I/O. Processor and storage hierarchy, system performance, Performance – Benchmarks, Metrics and their Limitations, Fault tolerance, pipelining timing analysis, and area performance analysis.
- **Performance Enhancements:** Branch Prediction, Dynamic scheduling, Speculative Execution (software/hardware), Superscalar Architecture, Out-of-order Execution, and Multithreading, VLIW and EPIC Architectures, Power Aware Computing, SPEC Mark vs. DTMR cache performance.
- **Multiprocessing:** Amdahl's Law, Multicore and Multithreaded Processors, Flynn's Taxonomy: Multiprocessor Structures and Architectures, Memory Synchronization and Cache Coherence, Interconnection Networks, Programming Multiprocessor Systems, GPU and Special-Purpose Graphics Processors. Case Study – Blue Gene, Road Runner
- **Distributed Systems:** Grids, Cluster Computing, Cloud Computing

Evaluation Scheme:

Exams	**Marks**	**Coverage**
Test-1	15 Marks	Syllabus covered upto Test-1
Test-2	25 Marks	Syllabus covered upto Test-2
Test-3	35 Marks	Full Syllabus
Assignment	10 Marks	
Tutorials	5 Marks	
Quiz	5 Marks	
Attendance	5 Marks	
Total	**100 Marks**	

References

1. Patterson D A, Hennessy J L, "Computer Architecture – A Quantitative Approach" Elsevier, 3rd Edition.
2. Kai Hwang, "Advanced Computer Architecture – Parallelism, Scalability, Programmability", Mc Graw Higher Education, 1992.

3. Michael J. Flynn, "Computer Architecture-pipelined and parallel processor design", Narosa publication, 8th Reprint, 2008.
4. J. E. Smith, "Characterizing Computer Performance with a Single Number", Communications of ACM, Volume 31, Issue 10 (October 1988), Pages: 1202 – 1206, 1988.
5. D. W. Wall, "Limits of Instruction Level Parallelism", Architectural Support for Programming Languages and Operating Systems, Santa Clara, Pages: 176 – 188, 1991.

Course Description

Title of Course: Software System Lab-I **Course Code: CS601**
L-T-P scheme: 0-0-2 **Course Credit: 1**

Prerequisite: Students must have good knowledge of C++/JAVA.

Objectives: To develop an understanding of the various components of the UNIX/Linux operating systems from a system programmer's perspective including both the shell and programming interfaces. To develop the ability to use a NS-2 simulator for a designed network.

Learning Outcomes: Students will be able to:

1. Efficient in system programming
2. Creating a process and making a IPC connection
3. Socket programming and System Call functions
4. Comfortable in Network Simulator

Course Contents:

Introduction to UNIX commands and Shell programming, Process model, Process environment, Process creation and termination, Interprocess communication, Process synchronization, Pipes, Named pipes, Introduction to assembly language, Introduction to Compiler, Introduction to NS-2, Mini Project.

Note: There will be three minis Project and each project will be done by different groups.

Exams		**Marks**	**Coverage**
P-1		15 Marks	**Based on Lab Exercises: upto P1**
P-2		15 Marks	**Based on all Lab Exercises**
Day-to-Day Work	Viva	20 Marks	**70 Marks**
	Demonstration	20 Marks	
	Lab Record	15 Marks	
	Attendance & Discipline	15 Marks	
Total		**100 Marks**	

References

1. Kay A. Robbins, Steven Robbins, "UNIX Systems Programming: Communication, Concurrency, and Threads", Prentice Hall 2003.
2. Alfred V. Aho, Ravi Sethi, Jeffrey D. Ullman, "Compilers: Principles, Techniques, and Tools", Pearson Education 2003.
3. Jeff Duntemann, "Assembly Language Step-by-Step: programming with DOS and Linux" Wiley Dreamtech India Pvt. Ltd., New Delhi, 2003.
4. Arnold Robbins, "UNIX in a Nutshell", O'Reilly 4th edition.

5. David I. Schawartz,"Introduction to UNIX", Prentice Hall, second edition .

Title of Course: Artificial Intelligence in Manufacturing **Course Code: CS701**
L-T-P Scheme: 3-0-0 **Course Credits: 3**

Course Objectives:

- Introduce the fundamentals of AI and its importance in manufacturing.
- Explore machine learning, robotics, and knowledge-based systems as applied to manufacturing.
- Equip students with the ability to analyze manufacturing problems and propose AI-based solutions.
- Familiarize students with Industry 4.0 technologies integrated with AI for smart factories.

Course Outcomes:

Course Outcomes	Description
CO1	Understand the fundamentals of AI and its relevance to manufacturing.
CO2	Apply machine learning techniques in manufacturing contexts.
CO3	Analyze and design AI-based solutions for automation and optimization of manufacturing processes.
CO4	Evaluate case studies and current practices of AI in smart manufacturing.
CO5	Appreciate the role of AI in Industry 4.0 and future manufacturing trends.

Course Contents:

Unit 1: Introduction to AI in Manufacturing

Evolution of manufacturing systems, Introduction to AI and Industry 4.0, AI technologies in smart manufacturing

Unit 2: Knowledge-based Systems in Manufacturing

Expert systems and decision support, Knowledge representation and reasoning, Applications in process planning and quality assurance

Unit 3: Machine Learning for Manufacturing

Supervised and unsupervised learning algorithms, Predictive maintenance and defect detection, Optimization of manufacturing processes using ML

Unit 4: Robotics and Intelligent Automation

AI in industrial robots, Computer vision applications, Human-robot collaboration in manufacturing environments

Unit 5: Advanced Applications and Case Studies

AI in supply chain and inventory management, Digital twins and simulation, Case studies on smart factories, predictive maintenance

Evaluation Scheme:

Exams	Marks	Coverage
Test-1	15 Marks	Syllabus covered upto Test-1
Test-2	25 Marks	Syllabus covered upto Test-2
Test-3	35 Marks	Full Syllabus
Assignment	10 Marks	
Tutorials	5 Marks	
Quiz	5 Marks	
Attendance	5 Marks	
Total	**100 Marks**	

Textbook:

1. Srinivas, K., & Anandakrishnan, V. (2020). Artificial Intelligence for Industrial Applications. Springer.

Reference Books:

1. Russell, S., & Norvig, P. (2020). Artificial Intelligence: A Modern Approach (4th Edition). Pearson.
2. Lee, J., Bagheri, B., & Kao, H.-A. (2015). "A Cyber-Physical Systems Architecture for Industry 4.0-based Manufacturing Systems", Manufacturing Letters, Elsevier.
3. Li, B., Hou, B., Yu, W., Lu, X., & Yang, C. (2017). "Applications of Artificial Intelligence in Intelligent Manufacturing: A Review", Frontiers of Information Technology & Electronic Engineering.
4. Zhang, X. (2019). Machine Learning Applications in Nontraditional Manufacturing Processes. Springer.

Course Description

Title of Course: Advanced Algorithm Lab **Course Code: CS805**
L-T-P scheme: 0-0-2 **Course Credit: 1**

Prerequisite: Good knowledge of Computer programming, Data Structures and algorithms. Proficiency in a high-level programming language is required.

Objectives:

- To practically implement advanced algorithms and data structures.
- To strengthen the ability to apply algorithm design paradigms such as greedy programming and dynamic programming to solve problems.
- To empirically analyze the performance and correctness of algorithms.
- To gain hands-on experience with algorithms from various domains, including computational geometry and cryptography.

Course Outcomes:

Course Outcomes	Description
CO1	Implement and evaluate the time complexity of various sorting and searching algorithms.
CO2	Develop and code solutions for optimization problems using Greedy and Dynamic Programming techniques.
CO3	Implement standard algorithms for string matching and computational geometry.
CO4	Create programs that use basic cryptographic and primarily testing algorithms.

Course Contents:

The lab will consist of programming assignments based on the topics covered in the "Advanced Algorithms Lab" course. Students will be expected to implement the following:

Sorting & Searching Algorithms:

- Implementation and empirical performance analysis of Quick Sort and Merge Sort.
- Implementation of non-comparison based sorting algorithms like Radix Sort and Bucket Sort.
- Implementation of a Hash Table with a chosen collision resolution strategy.

Greedy Algorithms:

- Implementation of Dijkstra's algorithm for the Shortest Path Problem.

- Implementation of Prim's and Kruskal's algorithms for finding a Minimum Spanning Tree.

Dynamic Programming:
- Implementation of a solution for the 0/1 Knapsack problem.
- Implementation of a solution for the Matrix Chain Multiplication problem.

String Matching:
- Implementation of the Rabin-Karp algorithm.
- Implementation of the Knuth-Morris-Pratt (KMP) algorithm.

Computational Geometry:
- Implementation of an algorithm, such as Graham scan, to find the Convex Hull of a set of points.

Cryptographic & Number Theory Algorithms:
- Implementation of the RSA public-key cryptosystem.
- Implementation of the Miller-Rabin primality test as part of the RSA implementation.

Evaluation Scheme:

Exams		**Marks**	**Coverage**
P-1		15 Marks	**Based on Lab Exercises: upto P1**
P-2		15 Marks	**Based on all Lab Exercises**
Day-to-Day Work	Viva	20 Marks	**70 Marks**
	Demonstration	20 Marks	
	Lab Record	15 Marks	
	Attendance & Discipline	15 Marks	
Total		**100 Marks**	

References

1. Algorithms: S. Dasgupta, C.H. Papadimitriou, and U.V. Vazirani, McGraw-Hill Science/Engineering/Math;
2. Cormen, Thomas H., Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, 2nd Edition, 2001.
3. Sedgewick, Robert. *Algorithms in C*. 3rd edition. Addison Wesley, 2002.

Course Title: Advanced Database Systems LAB Course Code: CS803

L-T Scheme: 0-0-2 Course Credits: 1

Prerequisite: Basic knowledge of Database Management Systems (DBMS)

Lab Objectives:

1. To provide practical experience in the design and implementation of advanced database concepts.
2. To enhance skills in writing complex SQL and PL/SQL programs.
3. To implement distributed, parallel, and secure database applications.
4. To explore recovery, indexing, and query optimization techniques through experiments.
5. To develop real-world web and mobile-enabled secure database applications

Lab Learning Outcomes:

Course Outcome	Description
CO1	Design and develop secure, normalized, and efficient database systems.
CO2	Write complex SQL, PL/SQL procedures, functions, and triggers.
CO3	Understand and implement indexing, query optimization, and distributed transactions.
CO4	Apply concurrency control and recovery techniques to real-time database environments.
CO5	Develop secure web and mobile-integrated database systems.

Lab Experiments:

Exp. No.	Title of the Experiment
1	**Design and Normalize a Complex Relational Schema** Model a realistic scenario (e.g., university or hospital system) by creating ER diagrams, mapping to relational schema, and normalizing up to 3NF/BCNF to eliminate redundancy.
2	**Advanced SQL Queries: Nested, Ranked, Analytical Functions** Craft advanced SQL using nested and correlated sub queries, window functions such as RANK, ROW_NUMBER, LEAD/LAG, analytic aggregates (SUM, AVG), ROLLUP/CUBE.

3 **Stored Procedures, Functions, and Exception Handling in PL/SQL** Implement procedural business logic with PL/SQL by writing stored procedures, functions, using cursors, handling exceptions (NO_DATA_FOUND, OTHERS), and structuring robust error control.

4 **Triggers for Data Integrity and Auditing** Create BEFORE/AFTER INSERT/UPDATE/DELETE triggers to enforce constraints; maintain audit logs (user, timestamp, old/new values.

5 **Designing Indexes and Query Optimization Analysis** Design primary, secondary, composite, and function-based indexes. Analyze EXPLAIN PLAN outputs, tuning queries using join strategies, cost metrics, and index selectivity

6 **Implementation of Fragmentation and Replication Strategies** Simulate horizontal and vertical fragmentation across distributed sites. Plan and test snapshot, transactional, and merge replication to ensure availability and consistency across nodes.

7 **Distributed Query Processing and Optimization** Develop SQL queries across distributed datasets, incorporate site-selection, join ordering, data localization, and network cost in optimization.

8 **Concurrency Control: Transaction Isolation Levels and Deadlock Simulation** Demonstrate and compare RCC, RC, RR, and Serializable isolation levels. Create and resolve deadlock scenarios using locking, timestamps, and multi-granularity techniques.

9 **Failure Recovery and Log-based Recovery Simulation** Simulate system crashes, apply UNDO/REDO recovery using Write-Ahead Logging. Understand ARIES-style checkpoint protocols to restore ACID properties

10 **Secure Database Access with Role-based Authorization** Define roles and grant privileges (SELECT, INSERT, EXECUTE) to roles/users. Set up schemas and enforce fine

11 **Development of a Web-based Secure Database Application (Mini Project –1)**
Build a secure front-end (HTML/Java/JDBC or Python/Flask) and back-end DB (Oracle/MySQL/Postgres). Implement form-based CRUD operations, parameterized queries, authentication, and session managemen

12 **Integration with Mobile/Cloud-based Storage (Mini Project -2)**

Evaluation Scheme:

Exams		**Marks**	**Coverage**
P-1		15 Marks	**Based on Lab Exercises: upto P1**
P-2		15 Marks	**Based on all Lab Exercises**
Day-to-Day Work	Viva	20 Marks	**70 Marks**
	Demonstration	20 Marks	
	Lab Record	15 Marks	
	Attendance & Discipline	15 Marks	

Total	**100 Marks**

Lab Software & Tools Required:

- Oracle Database 11g/12c or higher
- PostgreSQL / MySQL Server
- PL/SQL Developer / pgAdmin / MySQL Workbench
- Apache/PHP or Python Flask/Django for web interface
- Java or Kotlin (optional) for mobile interfacing
- Firebase / REST API tools for mobile DB connectivity
- Git for version control

Lab References:

1. Steven Feuerstein, *Oracle PL/SQL Programming*, O'Reilly Media
2. Robin Nixon, *Learning PHP, MySQL & JavaScript*, O'Reilly Media
3. Mark Stamp, *Information Security: Principles and Practice*, Wiley
4. Ramez Elmasri & Shamkant Navathe, *Fundamentals of Database Systems*, Pearson
5. Vijay Kumar, *Mobile Database Systems*, Wiley-Interscience

Course Description

Title of Course: Advanced Operating Systems **Course Code: CS505**
L-T Scheme: 3-0-0 **Course Credits: 3**
Prerequisite: Students must have good knowledge of "Introduction to Computer Architecture and Operating System".

This course aims to:

1. Study advanced components of operating systems, including their functions, mechanisms, and policies.
2. Understand implementation techniques with examples from Unix/Linux.
3. Explain how different OS modules interact to provide core services.
4. Develop insight into real-world operating-system design and architecture.

Learning Outcomes:

Course Outcome	**Description**
CO1	Explain and compare different operating system architectures and evaluate their design trade-offs using real systems.
CO2	Analyze problems of process communication, synchronization, and deadlocks in distributed environments, and apply appropriate prevention/avoidance/detection techniques.
CO3	Evaluate distributed mutual exclusion and concurrency control algorithms with respect to correctness, scalability, performance, and fault tolerance.
CO4	Differentiate types of device drivers and explain their internal operation using case studies; describe file-system organization and recovery mechanisms.
CO5	Apply fundamental OS security principles to identify threats and propose secure authentication and network protection mechanisms in operating systems.

Course Contents:

UNIT–I: Overview of Operating Systems & Design Principles: Introduction to advanced OS concepts; review of principles in Linux, VxWorks, and other systems. Concepts of OS structure monolithic, layered, and microkernel architectures. Comparative evaluation of Minix and µCOS with focus on merits, demerits, and design trade-offs.

UNIT–II: Process Communication & Synchronization: Process synchronization mechanisms and coordination challenges in distributed environments. Deadlocks: detection, avoidance, and prevention strategies. Resource models, event ordering, timestamps, and the concepts of local vs. global states. Practical limitations of distributed operating systems are emphasized.

UNIT–III: Concurrency Control in Distributed Systems: Distributed mutual exclusion algorithms token-based and non-token-based approaches. Performance comparison of concurrency strategies. Discussion on correctness, scalability, fault tolerance, and real-world scenarios where concurrency control is critical.

UNIT–IV: Device Drivers & File Systems: Types of device drivers: character, block, network, serial, and USB. Case studies—keyboard, HDD, and LAN card drivers. Concepts behind hot-pluggable USB driver architecture. Introduction to file systems: file handling, recovery techniques, and case studies of FAT and EXT2FS.

UNIT–V: Security & Operating System Design Studies: OS security fundamentals: authentication mechanisms, attack types, and secure network communication frameworks.

Text Books

1. Advanced Concepts in Operating Systems: M. Singhal, N. G. Shivaratri, 1st Ed., Tata McGraw-Hill, 1994.

References

1. Linux Device Drivers : Alessandro Rubini, 3rd Ed., O'Reilly, 2005.
2. Real-Time Systems: C. M. Krishna, K. G. Shin, 1st Ed., McGraw-Hill, 1997.
3. MicroC/OS-II, The Real-Time Kernel: J. J. Labrosse, 2nd Ed., CMP Books, 2002.
4. Real-Time Systems: J. W. S. Liu, 1st Ed., Addison-Wesley, 2000.
5. The Linux Kernel Module Programming Guide: Peter Jay Salzman, Michael Burian, Ori Pomerantz, 2.6.1 Edition, Linux Documentation Project, 2005.
6. Distributed Operating Systems: Tanenbaum, A. S 1st Ed., Prentice-Hall, Englewood Cliffs, NJ, 1995.

Title of Course: Advanced Software Engineering **Course Code: CS506**
L-T Scheme: 3 **Course Credits: 3**

Prerequisite: Students must have good knowledge of "Software Engineering and Project Management".

Objectives: To engineer good quality software from its specification.

Learning Outcomes:

Course Outcome	Description
CO1	Understand and summarize core software engineering concepts
CO2	Explore maintenance and reengineering approaches for legacy systems
CO3	Apply PSP/TSP practices for time tracking, planning, defect management, and improving product/process quality
CO4	Use software metrics for measurement and estimation and apply effort/cost estimation methods.
CO5	Study Agile methods and advanced testing techniques

Course Contents:

Review of software engineering: Requirement Engineering, Software Process Models, Analysis and Design, UML Diagrams.

Software Reengineering: Software reengineering, forward engineering, reverse engineering, program comprehension.

PSP and TSP: Time management, Tracking time, Product planning, Project Plans, Defect finding, projecting a defect, Quality: Product and Process quality.

Software Metrics: Introduction to Metrics, challenges in Metrics, classification of Metrics. Product Metrics: Metrics for Analysis, Metrics for design, Metrics for code, Metrics for Testing. Process Metrics: Size, cost, effort Estimation, COCOMO Model. Project Metrics: Metric for Quality.

Agile Methods: Agile development, Classification of methods, Agile principals, Agile project management, SCRUM, extreme Programming.

Advance software Testing: Object Oriented Testing , GUI Testing , Real Time system Testing, Automated Testing, cyclomatic complexity, black box testing, data flow testing, graph based testing, regression testing.

Text Book

1. Software Engineering: A practitioner's approach: Roger S. Pressman, McGraw-Hill Publications (Sixth Edition)

References

1. Software Testing Techniques, B. Beizer (Second Edition)

2. Agile and Iterative Development: A Manager's Guide, Craig Larman
3. Software Engineering, Sommerville, Addison Wesley
4. Design Patterns, Erich Gamma, Johnson, Pearson Education
5. IEEE International Conference on Software Maintenance (ICSM)
6. ACM Transactions on Software Engineering Methodology.
7. IEEE Transactions on Software Engineering.

Title of Course: Advance Algorithm **Course Code: CS504**
L-T Scheme: 3-0-0 **Course Credits: 3**

Prerequisite: Students should have knowledge of data structure and programming concepts.

Objectives:

- To understand the proof of correctness and running time of the algorithms for the classic problems in various domains.
- To apply algorithmic design paradigms and methods of analysis in common engineering design situation

Learning Outcomes:

Course Outcome	Description
CO1	Understand and apply mathematical preliminaries to the analysis and design of various types of algorithms.
CO2	Understand and apply the probabilistic analysis and randomized algorithms.
CO3	Apply various algorithmic strategies and paradigms to model engineering problems.
CO4	Understand NP completeness-related issues and approximation to np-complete problems.
CO5	Understand and apply the parallel algorithms in various research problems.

Course Contents:

UNIT 1: Sorting, Graph Algorithms, and Amortized Analysis

Sorting: Review of various sorting algorithms, topological sorting Graph: Definitions and Elementary Algorithms: Shortest path by BFS, shortest path in edge-weighted case (Dijkstra's), depth-first search and computation of strongly connected components, emphasis on correctness proof of the algorithm and time/space analysis, example of amortized analysis.

UNIT 2: Matroids, Graph Matching, and Flow Networks

Matroids: Introduction to greedy paradigm, algorithm to compute a maximum weight maximal independent set. Application to MST. Graph Matching: Algorithm to compute maximum matching. Characterization of maximum matching by augmenting paths, Edmond's Blossom algorithm to compute augmenting path. Flow-Networks: Maxflow-mincut theorem, Ford-Fulkerson Method to compute maximum flow, Edmond-Karp maximum-flow algorithm.

UNIT 3: Dynamic Programming, Number Theory, and Fourier Transforms

Shortest Path in Graphs: Floyd-Warshall algorithm and introduction to dynamic programming paradigm. More examples of dynamic programming. Modulo Representation of integers/polynomials: Chinese Remainder Theorem, Conversion between base-representation and modulo representation. Extension to polynomials. Application: Interpolation problem. Discrete Fourier Transform (DFT): In complex field, DFT in modulo ring. Fast Fourier

Transform algorithm. Schonhage-Strassen Integer Multiplication algorithm

UNIT 4: Linear Programming, NP-Completeness, and Advanced Topic

Linear Programming: Geometry of the feasibility region and Simplex algorithm NP-completeness: Examples, proof of NP-hardness and NP-completeness. One or more of the following topics based on time and interest Approximation algorithms, Randomized Algorithms, Interior Point Method. Advanced Number Theoretic Algorithm Recent Trends in problem solving paradigms using recent searching and sorting techniques by applying recently proposed data structures.

Unit-5: Parallel Algorithm

Performance Measures of Parallel Algorithms, Parallel Merging/Sorting Algorithms on CREW/EREW, and Parallel searching algorithms

Text Books

- Coreman, Rivest, Lisserson, “Algorithm”, PHI, 2009

Reference Books

- The Design and Analysis of Computer Algorithms" by Aho, Hopcroft, Ullman.

Algorithm Design by Kleinberg and Tardos.

Title of Course: Machine Learning and Applications **Course Code: CS711**
L-T Scheme: 3-0-0 **Course Credits: 3**

Prerequisite: Students must have knowledge of statistical techniques

Objectives: This course aims to provide an in-depth understanding of machine learning theory and system components, focusing on mathematical foundations, algorithmic design, optimization strategies, and evaluation methodologies. Students will critically analyze learning mechanisms, develop and implement advanced models, and apply machine learning techniques to complex, real-world and research-oriented problems.

Learning Outcomes:
The students will have

1. Concepts of machine learning.
2. Ability to analyze data using mathematical approach
3. Good understanding of Classifications and Regression
4. Good knowledge of Ensemble Learning Techniques
5. Good knowledge of Unsupervised Learning Techniques

At the end of the course students will be able to:

Course Outcome	**Description**
CO1	Demonstrate understanding of various machine learning approaches.
CO2	Develop insights by analyzing data using various statistical tools.
CO3	Design models to solve machine learning problems using regression / classification approaches.
CO4	Evaluate the performance of machine learning solutions and fine-tuning.
CO5	Analyze the importance of performance indicators of the machine learning solution.

Course Contents:

Unit-I Classification: MNIST Training a Binary Classifier, Performance Measures, Measuring Accuracy.

Unit-II Computational Complexity, Gradient Descent, Polynomial Regression, Learning Curves, Regularized Linear Models, Logistic Regression, Estimating Probabilities, Training and Cost Function, and Decision Boundaries.

Unit-III Support Vector Machines Linear SVM Classification, Soft Margin Classification, Nonlinear SVM Classification, Polynomial Kernel, Adding Similarity Features, Gaussian RBF Kernel, Computational Complexity, SVM Regression.

Unit-IV Decision Trees Training and Visualizing a Decision Tree, The CART Training Algorithm, Computational Complexity, Gini Impurity or Entropy, Regularization of hyper parameters, and Random Forests.

Unit-V Dimensionality Reduction, Projection, Manifold Learning, PCA, Preserving the Variance, Principal Components, Choosing the Right Number of Dimensions Clustering, K-Means, Limits of K-Means, Using clustering for image segmentation, Using Clustering for Pre-processing and for Semi-Supervised Learning

Evaluation Scheme:

Exams	**Marks**	**Coverage**
Test-1	15 Marks	Syllabus covered upto Test-1
Test-2	25 Marks	Syllabus covered upto Test-2
Test-3	35 Marks	Full Syllabus
Assignment	10 Marks	
Tutorials	5 Marks	
Quiz	5 Marks	
Attendance	5 Marks	
Total	**100 Marks**	

References

1: Machine Learning, TOM M MITCHELL, TMH
2: Introduction to Machine Learning, 2nd Ed, Ethem Alpaydin, The MIT Press Cambridge, Massachusetts,London, England.
3: Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Ed, Aurelien Geron, O'RIELLY
4: Deep Learning, Ian Goodfellow, Yoshua Bengio, and Aaron Courville, MIT Press

Online Resources:

Introductory Course Introduction to ML: link is given below
NPTEL https://nptel.ac.in/courses/106106139

Title of Course: Swarm Intelligence & Applications **Course Code: CS727**
L-T Scheme: 3-0 **Course Credits: 3**

Prerequisite: Students should have basic knowledge of Data Structures, Algorithms, Probability & Statistics, and Fundamentals of Artificial Intelligence or Soft Computing.

Objectives: The main aim of this course is to introduce students to swarm intelligence principles and bio-inspired optimization techniques derived from the collective behavior of natural systems such as ants, birds, bees, and fish. The course focuses on understanding, designing, and applying swarm-based algorithms to solve complex optimization and real-world computational problems.

Learning Outcomes: Students will acquire skills to apply swarm-based computational techniques for optimization and decision-making problems.

Course Outcome	**Description**
CO1	Explain fundamentals of swarm intelligence and self-organization.
CO2	Analyze ACO and PSO algorithms.
CO3	Apply swarm techniques to optimization problems.
CO4	Evaluate performance of swarm algorithms.
CO5	Design swarm-based solutions for real-world applications.

Course Contents:

Unit-I Introduction to Swarm Intelligence:

Overview of Swarm Intelligence, Biological Inspiration, Characteristics of Swarm Systems, Self-organization, Emergence, Collective Behavior, Comparison with Traditional AI and Soft Computing Techniques.

Unit-II Ant Colony Optimization (ACO):

Biological Inspiration of Ants, Pheromone Communication, ACO Meta-heuristic Framework, Ant System, Ant Colony System, MAX-MIN Ant System, Applications in Traveling Salesman Problem, Routing, and Scheduling.

Unit-III Particle Swarm Optimization:

Bird Flocking and Fish Schooling Behavior, PSO Algorithm, Velocity and Position Update Rules, Convergence Analysis, Variants of PSO, Applications in Continuous Optimization and Neural Network Training.

Unit-IV-Other Algorithms:

Artificial Bee Colony (ABC), Firefly Algorithm, Bat Algorithm, Cuckoo Search, Glowworm Swarm Optimization, Hybrid Swarm Intelligence Techniques.

Unit-V Applications of Swarm Intelligence:

Optimization in Engineering Problems, Network Routing, Task Scheduling, Data Clustering, Feature Selection, Image Processing, Machine Learning Optimization, Robotics and Multi-Agent Systems.

Evaluation Scheme:

Exams	Marks	Coverage
Test-1	15 Marks	Syllabus covered upto Test-1
Test-2	25 Marks	Syllabus covered upto Test-2
Test-3	35 Marks	Full Syllabus
Assignment	10 Marks	
Tutorials	5 Marks	
Quiz	5 Marks	
Attendance	5 Marks	
Total	**100 Marks**	

References

- Kennedy J., Eberhart R., "Swarm Intelligence", Morgan Kaufmann, 2001.
- Dorigo M., Stützle T., "Ant Colony Optimization", MIT Press, 2004.
- Yang X. S., "Nature-Inspired Optimization Algorithms", Elsevier, 2014.
- Engelbrecht A. P., "Computational Intelligence: An Introduction", Wiley, 2nd Edition.
- Blum C., Merkle D., "Swarm Intelligence: Introduction and Applications", Springer.

Title of Course: Software System Lab-II **Course Code: CS602**
L-T-P scheme: 0-0-2 **Course Credit: 1**

Prerequisite:

Students should have proficiency in at least one object-oriented programming language such as Java, Python, or C++. They should be familiar with core object-oriented programming concepts including classes, inheritance, polymorphism, and interfaces. Basic understanding of software engineering practices, modular programming, and version control systems is expected. Introductory knowledge of web technologies, RESTful APIs, and basic command-line usage will be beneficial.

Objectives:

The objective of this course is to provide students with practical exposure to modern software system development practices, with a focus on software design patterns, containerization, and microservices architecture. The course aims to strengthen students' ability to design modular, reusable, and maintainable software systems and to deploy such systems using contemporary tools and frameworks commonly used in industry.

Learning Outcomes:

Course Outcome	Description
CO1	Students will apply fundamental software design principles to develop well-structured and maintainable software systems.
CO2	Students will implement and evaluate commonly used object-oriented design patterns in practical scenarios..
CO3	Students will demonstrate the ability to build, containerize, and manage applications using Docker.
CO4	Students will design and deploy microservices-based applications using containerized environments.
CO5	Students will exhibit professional practices including proper documentation, collaboration, and ethical coding standards.

Course Content:

Unit-1: This unit introduces the foundations of software design with an emphasis on object-oriented design thinking. It covers key design principles such as abstraction, modularity, encapsulation, and separation of concerns. The unit also introduces SOLID design principles and discusses their role in building flexible, maintainable, and scalable software systems. Practical exercises focus on identifying design issues and improving existing code using these principles.
Unit-2: This unit focuses on the concept of design patterns as reusable solutions to recurring software design problems. It introduces the motivation and benefits of using design patterns in large-scale systems. Students implement and analyze commonly used creational patterns

such as Singleton, Factory, and Abstract Factory; structural patterns such as Adapter, Decorator, and Proxy; and behavioral patterns such as Observer, Strategy, and Command. Emphasis is placed on selecting appropriate patterns and understanding trade-offs through hands-on implementations and case studies.

Unit-3: This unit introduces containerization as a modern approach to application deployment. It covers Docker architecture, core components, and the differences between containers and virtual machines. Students learn to create Docker images using Dockerfiles, manage containers, and work with Docker registries. The unit also introduces Docker Compose for defining and managing multi-container applications, with practical exercises focusing on containerizing real-world applications.

Unit-4: This unit explores microservices architecture as an alternative to monolithic system design. It discusses the characteristics, advantages, and challenges of microservices-based systems. Students learn about microservice design principles, service boundaries, and communication mechanisms such as REST-based interaction. The unit also introduces configuration management and basic service discovery concepts, emphasizing design considerations for scalable and loosely coupled systems.

Unit-5: This unit integrates design patterns, Docker, and microservices into a cohesive workflow. Students learn to containerize individual microservices and manage their interaction using Docker Compose. The unit provides an overview of orchestration concepts using tools such as Docker Swarm or Kubernetes. Topics such as scaling, fault tolerance, and basic monitoring are introduced. The unit culminates in a mini-project where students design, implement, and deploy a microservices-based application.

Evaluation Scheme:

Exams		Marks	Coverage
P-1		15 Marks	Uni1-1 & Unit-2
P-2		15 Marks	Unit-3, Unit-4 & Unit-5
Day-to-Day Work	Viva	20 Marks	70 Marks
	Demonstration	20 Marks	
	Lab Record	15 Marks	
	Attendance & Discipline	15 Marks	
Total		100 Marks	

Resources:

1. *Design Patterns: Elements of Reusable Object-Oriented Software* – Erich Gamma et al.
2. *Clean Code: A Handbook of Agile Software Craftsmanship* – Robert C. Martin
3. Docker Official Documentation – https://docs.docker.com/
4. Microservices architecture articles, case studies, and online resources.

Title of Course: Advanced Operating Systems Lab **Course Code: CS804**
L-T-P scheme: 0-0-2 **Course Credit: 1**

Prerequisite:

Basic knowledge of Operating Systems (processes, memory management, file systems, synchronization)

Objectives:

1. To provide hands-on experience with advanced operating system concepts.
2. To strengthen skills in process management, synchronization, and memory handling.
3. To implement concurrency control, scheduling, and IPC mechanisms.
4. To explore virtualization, distributed OS concepts, and performance analysis.
5. To develop secure and real-world system-level applications.

Learning Outcomes:

Course Outcome	Description
CO1	Design and implement efficient process, thread, and memory management solutions.
CO2	Apply synchronization, deadlock handling, and concurrency control techniques.
CO3	Understand and implement scheduling, IPC, and distributed OS mechanisms.
CO4	Analyze OS performance, reliability, and recovery techniques.
CO5	Develop secure, virtualized, and cloud-integrated OS-level applications.

Course Content:

Lab Experiments:

1. Process Creation and Management
2. Multithreading and Thread Synchronization
3. CPU Scheduling Algorithms Simulation
4. Inter-Process Communication (IPC)
5. Deadlock Detection, Prevention, and Avoidance
6. Memory Management and Virtual Memory
7. File System Implementation and Performance Analysis
8. Classical Synchronization Problems
9. Failure Recovery and Checkpointing
10. Secure Operating System Mechanisms

Evaluation Scheme:

<table>
<tr><td colspan="2">Exams</td><td>Marks</td><td>Coverage</td></tr>
<tr><td colspan="2">P-1</td><td>15 Marks</td><td>Up To P1</td></tr>
<tr><td colspan="2">P-2</td><td>15 Marks</td><td>Complete</td></tr>
<tr><td rowspan="4">Day-to-Day Work</td><td>Viva</td><td>20 Marks</td><td rowspan="4">70 Marks</td></tr>
<tr><td>Demonstration</td><td>20 Marks</td></tr>
<tr><td>Lab Record</td><td>15 Marks</td></tr>
<tr><td>Attendance & Discipline</td><td>15 Marks</td></tr>
<tr><td colspan="2">Total</td><td colspan="2">100 Marks</td></tr>
</table>

Resources:

- Modern Operating Systems — Andrew S. Tanenbaum & Herbert Bos
- Advanced Concepts in Operating Systems — Mukesh Singhal & Niranjan G. Shivaratri
- Operating Systems: Design and Implementation — Andrew S. Tanenbaum & Albert S. Woodhull
- Operating Systems: Three Easy Pieces — Remzi H. Arpaci-Dusseau & Andrea C. Arpaci-Dusseau
- Linux Kernel Development— Robert Love

Title: Research Methodology and IPR **Code: HS520**
L-T-P scheme: 2-0-0 **Credits: 2**

Prerequisite: There is no prerequisite for this course.

Objective: This course is designed to enable students:
1. To identify the issues and concepts salient to the research.
2. To undersatand the overall processes of designing a research study from its inception to its report.
3. To use basic techniques of quantitative and qualitative research.
4. To gain knowledge to develop data analytics skills and meaningful interpretation to the data sets so as to solve the research problem.
5. To conduct the research work and draft research articles, synopsis and report.

Learning Outcomes:

Course Outcome	Description
CO1	Develop understanding of basic concepts of research and its methodologies, various kinds of research, objectives of doing research, research process, and ethical dimensions of conducting research.
CO2	Select and define appropriate research problem & identify various sources of information for literature review and data collection.
CO3	Outline various concepts of research design, sample design, measurement and scaling techniques, IPR and their application.
CO4	Formulation of research hypotheses and applying various techniques of data analysis for hypothesis testing with the aspects of IPR
CO5	Describe the real-world problems using basic techniques of descriptive and inferential data analysis and to reach a conclusion based on the data analysis, and to make and defend a recommended course of action.

Course Outline

1. Introduction to Research Methodology: Meaning of Research, Objectives of Research, Motivations in Research, Types and Approaches of Research, Significance of Research, Research Methods v/s Methodology, Research and Scientific Methods, Research Process, Criteria of Good Research.

2. Defining the Research Problem: What is Research Problem?, Selecting the Problem, Necessity of and Techniques in defining the problem

3. Research design: Basic research process, Meaning and necessity of research design, Features of good research design, Different research designs, Basic principles

4. Sample Design: Sample survey, Implications of a sample design, Steps in sampling design, Need for sampling, Important sampling distributions, Criteria of selecting a sampling procedure, Characteristic of a good sample design, Different types of samples design

5. Measurement and scaling techniques: Measurement in research, Measurement scales, Sources of errors in measurement, Techniques of developing measurement tools, Scaling, Scale classification bases, important scaling techniques

6. Methods of Data Collection and Presentation: Various methods of Primary and secondary data collection Selection of appropriate method for data collection; Case Study Method, Guidelines for developing questionnaire, successful interviewing. Survey v/s experiment, Methods of Data presentation

7. Processing and analysis of data: Frequency distributions, Measure of central tendency, Measures of dispersion , Simple regression analysis, Multiple correlation and regression, Estimation of population means, Sample size and its determination, Chi- square test, Analysis of Variance and Co-variance

8. Research Ethics: Ethics introduction, ethics in research work, concept of similarity, Plagiarism, Academic Integrity, Departmental Academic Integrity Panel (DAIP), Institutional Academic Integrity Pane (IAIP)

9. Intellectual property rights: Intellectual Property – The concept of IPR, Evolution and development of concept of IPR, IPR development process, Trade secrets, utility Models, IPR & Bio diversity, Role of WIPO and WTO in IPR establishments, Right of Property, Common rules of IPR practices, Types and Features of IPR Agreement, Trademark, Functions of UNESCO in IPR maintenance., Trademark, Patents.

Evaluation Scheme:

Exams	Marks	Coverage
Test-1	15 Marks	Based on Unit-1 & Unit-2
Test-2	25 Marks	Based on Unit-3 & Unit-4 and around 30% from coverage of Test-1
Test-3	35 Marks	Based on Unit-5 and around 30% from coverage of Test-2
Assignment	10 Marks	
Tutorials	5 Marks	
Quiz	5 Marks	
Attendance	5 Marks	
Total	**100 Marks**	

Text Book

- Research Methodology-Methods and Techniques by C.R. Kothari; New Age publication.
- Research Methodology by R.Panneerselvam ; PHI publication.
- Research Methodology by P.K. Sharma
- H.A. Taha. (1992) Operations Research- An Introduction, New York: Macmillan.
- F. S. Hiller and G. J. Liberman. Introduction to Operations Research, San Francisco: Holden- Day.
- Intellectual Property: A Very Short Introduction, Siva Vaidhyanathan
- Intellectual Property: The Law of Trademarks, Copyrights, Patents, and Trade Secrets" Deborah E. Bouchoux

Title of Course: Minor Project **Course Code: CS604**
L-T-P Scheme: 0-0-6 **Course Credits: 3**

The Minor Project course for M. Tech. (Computer Science and Engineering) aims to provide students with hands-on experience in applying advanced concepts of computer science to solve research-oriented or real-world problems. The course enables students to identify a focused problem in emerging areas such as algorithms, machine learning, artificial intelligence, and data analytics, cloud computing, cyber security, computer vision, Internet of Things, or software engineering.

Students are required to carry out a comprehensive literature survey, design an appropriate system architecture or algorithmic solution, and implement the proposed solution using modern programming tools and development platforms. Emphasis is placed on experimentation, performance evaluation, result analysis, and professional technical documentation. This course prepares students for their major projects (Dissertation-I and II), research work, and industry-level problem-solving.

Learning Outcomes:

Course Outcome	Description
CO1	Identify and define a focused research or development problem in the domain of Computer Science and Engineering.
CO2	Perform an extensive literature survey using scholarly resources to understand existing solutions, gaps, and research trends.
CO3	Design and implement an appropriate methodology, algorithm, model, or system using modern engineering and computational tools.
CO4	Apply experimental, analytical, or simulation-based methods to evaluate the performance, correctness, and efficiency of the proposed solution.
CO5	Analyze experimental or simulation results and interpret findings using quantitative and qualitative techniques.
CO6	Prepare a well-structured technical report and effectively present the project work demonstrating professional and ethical skills.

Evaluation Scheme:

Exams	Marks	Coverage
Mid Semester Viva	20 Marks	
Final Viva	30 Marks	
Project Report	20 Marks	
Day to Day Work	30 Marks	
Total	**100 Marks**	