TADIPATRI ENGINEERING COLLEGE

(AUTONOMOUS)

Accredited by NAAC, Approved by AICTE, New Delhi & Affiliated to JNTUA, Anantapur)

M. Tech (Regular-Full Time)

(Effective for the students admitted into I year I Sem from the Academic Year **2025-26** onwards)

M.TECH.-*SOFTWARE ENGINEERING*

I, II, III & IV SEMESTER COURSE STRUCTURE & SYLLABUS

DEPARTMENT COMPUTER SCIENCE AND ENGINEERING

M. TECH SOFTWARE ENGINEERING

EFFECTIVE FROM ACADEMIC YEAR 2025-26 ADMITTED BATCH

R-25 COURSESTRUCTURE AND SYLLABUS

I YEAR I - Semester

CODE	Course Title	Category	L	T	P	Credits
TEC25D58101	Software Requirements & Estimation	PC	3	0	0	3
TEC25D25101	Advanced Data Structures	PC	3	0	0	3
TEC25D25102a TEC25D25102b TEC25D25102c TEC25D25102d TEC25D25102e	1. Enterprise Cloud Concepts 2. Dev-Ops 3. Database Programming 4. Network Programming 5. Internet of things	PE	3	0	0	3
TEC25D58103a TEC25D58104b TEC25D25103a	1. Software Project Management 2. Software Metrics 3. Software Reliability 4. Software Agents 5. Secure Coding Practices	PE	3	0	0	3
TEC25D58105	Advanced Data Structures Lab	PC	0	0	4	2
TEC25D25104	Professional Elective-I Lab	PC	0	0	4	2
TEC25MRM101	Research Methodology& IPR	MC	2	0	0	2
TEC25DAC101a	Full stack Development Using MERN	SE	0	1	2	2
TEC25MAC01A TEC25MAC01B TEC25MAC01C	Audit Course-I English for Research Paper Writing Disaster Management Essence of Indian Traditional Knowledge	AC	2	0	0	0
	Total		**16**	**1**	**10**	**20**

Professional Elective-I and Professional Elective-I Lab must be of same course

I YEAR II- Semester

Course Code	Course Title	Categoy	L	T	P	Credits
TEC25MSE0003T	Design Patterns	PC	3	0	0	3
TEC25MSE0004T	Software Quality Assurance	PC	3	0	0	3
TEC25MSEPE03Ta TEC25MSEPE03Tb TEC25MSEPE03Tc TEC25MSEPE03Td	1. Software Testing Methodologies 2. Mobile Application and API Development 3. Full Stack Development 4. Functional Programming	PE	3	0	0	3
TEC25MSEPE04Ta TEC25MSEPE04Tb TEC25MSEPE04Tc TEC25MSEPE04Td	1. Secure Software Engineering 2. Object Oriented Software Engineering 3. Human Computer Interaction 4. Machine Learning	PE	3	0	0	3
TEC25MSE0020P	Design Patterns Lab	PC	0	0	4	2
TEC25MSEPE20P	Professional Elective-III Lab	PC	0	0	4	2
TEC25MRMC02	Quantum Technologies And Applications	MC	2	0	0	2
TEC25MCVO01	Comprehensive Viva Voce	PC	0	0	0	2
TEC25MAC02A TEC25MAC02B TEC25MAC02C	Audit Course-II Pedagogy Studies Personality Development Through Life Enlightenment Skills Yoga For Stress Management	AC	2	0	0	0
	Total		**14**	**0**	**8**	**20**

Professional Elective-III and Professional Elective-III Lab must be of same course

**Students have to undergo an Industry Internship after I Year II Semester for a duration of 6 to 8 weeks

III SEMESTER

S.No.	Code	Course Name	Category	Hours per week			Credits
				L	T	P	
1.	TEC25MSEPE05T	**Program Elective – V** 1. Agile Development Methodologies 2. Web Services Testing 3. Ad-hoc Networks 4. Mobile Application Security	PE	3	0	0	3
2.	TEC25MSEOE01T	Open Elective-I	OE	3	0	0	3
3.	TEC25MDP301	Dissertation Phase – I	PR	0	0	20	10
4.	TEC25MII301	Industry Internship		0	0	0	2
5.	TEC25MCC301	Co- Curricular Activities		0	0	0	1
Total							19

IV SEMESTER

S.No.	Corse Codes	Course Name	Category	Hours per			Credits
				L	T	P	
1.	TEC25MDP401	**Dissertation Phase – II**	PR	0	0	32	16
Total							**16**

Open Elective:

1. Cyber Security
2. Fault Tolerance Systems
3. Intrusion Detection Systems
4. Optimization Techniques
5. Cyber Physical Systems
6. Graph Analytics
7. Network Programming

8. Agile Development Methodologies
9. Secure Software Engineering
10. Principles of Software Engineering

SOFTWARE REQUIREMENTS & ESTIMATION (PC-I)

M.Tech SE I Year I Sem.

L	T	P	C
3	0	0	3

Course Objectives:

- Students will demonstrate knowledge of the distinction between critical and non- critical systems.
- Students will demonstrate the ability to manage a project including planning, scheduling and risk assessment /management.
- Students will author a software requirements document.
- Students will demonstrate an understanding of the proper contents of a software requirements document.
- Students will author a formal specification for a software system.
- Students will demonstrate an understanding of distributed system architectures and application architectures.
- Students will demonstrate an understanding of the differences between real-time and non-real time systems.
- Students will demonstrate proficiency in rapid software development techniques.
- Students will demonstrate proficiency in software development cost estimation
- Students will author a software testing plan.

Course Outcomes:

- Explain the concepts of software requirements and their impact on project success.
- Apply effective requirements elicitation techniques and document software requirements accurately.
- Analyze and prioritize software requirements to align with project goals and risk management.
- Use different software estimation methods to predict project size, effort, and cost effectively.
- Evaluate and select suitable requirements management and estimation tools for software projects.

UNIT-I

Software Requirements: What and Why Essential Software requirement, Good practices for requirements engineering improving requirements processes, Software requirements and risk management

Software Requirements Engineering Requirements elicitation, requirements analysis documentation, review, elicitation techniques, analysis models, Software quality attributes, risk reduction through proto typing, setting requirements priorities, verifying requirements quality.

UNIT-II

Software Requirements Management Requirements Management Principles and practices, Requirements attributes, Change Management Process, Requirements Traceability Matrix, Links in requirements chain

Software Requirements Modeling Use Case Modeling, Analysis Models, Data flow diagram, state transition diagram, class diagrams, Object analysis, Problem Frames

UNIT-III

Software Estimation Components of Software Estimations, Estimation methods, Problems associated with estimation, Key project factors that influence estimation

Size Estimation

Two views of sizing, Function Point Analysis, Mark IIFPA, Full Function Points, LOC Estimation, Conversion between size measures.

UNIT-IV

Effort, Schedule and Cost Estimation What is Productivity? Estimation Factors, Approaches toEffort and Schedule Estimation, COCOMO II, Putnam Estimation Model, Algorithmic models, Cost

UNIT-V

Tools for Requirements Management and Estimation Requirements Management Tools: Benefits of using a requirements management tool, commercial requirements management tool, Rational Requisite pro, Caliber – RM, implementing requirements management automation,

Software Estimation Tools: Desirable features in software estimation tools, IF PUG, USC's COCOMO II, and SLIM (Software Life Cycle Management) Tools

Text Book:

1. Software Requirements and Estimation by *Rajesh Naik and Swapna Kishore*, Tata Mc Graw Hill.

Reference Books:

1. Software Requirements by Karl E. Weigers, Microsoft Press.
2. Managing Software Requirements, Dean Leffing well& Don Widrig, Pearson Education, 2003.
3. Mastering the requirements process, second edition, Suzanne Robertson & James Robertson, Pearson Education, 2006.
4. Estimating Software Costs, Second edition, Capers Jones, TMH, 2007.
5. Practical Software Estimation, M.A. Parthasarathy, PearsonEducation,2007.
6. Measuring the software process, William A. Flora c & Anita D. Carleton, Pearson Education, 1999.

ADVANCED DATA STRUCTURES (PC-II)

M.Tech SE I Year I Sem.

L	T	P	C
3	0	0	3

Pre requisites: A course on "Data Structures"

Course Objectives:

- Introduces the heap data structure such as leftist trees, binomial heaps, Fibonacci and min-max heaps
- Introduces a variety of data structures such as disjoint sets, hash tables, search structures and digital search structures

Course Outcomes:

- Ability to select the data structures that efficiently model the information in a problem
- Ability to understand how the choice of data structures impact the performance of programs
- Design programs using a variety of data structures, including hash tables, search structures and digital search structures

UNIT –I

Introduction: Introduction, Stacks, queues, linked lists.

Heap Structures: Introduction, Min-Max Heaps, Binomial Heaps, Fibonacci heaps.

UNIT-II

Hashing and Collisions

Introduction, Hash Tables, Hash Functions, different Hash Functions: Division Method, Multiplication Method, Mid-Square Method, Folding Method, Collisions

UNIT-III

Search Structures: OBST, AVL trees, Red-Black trees, Splay trees,

Multiway Search Trees: B-trees, 2-3 trees

UNIT-IV

Digital Search Structures

Digital Search trees, Binary tries and Patricia, Multiway Tries, Suffix trees, Standard Tries, Compressed Tries

UNIT -V

Pattern matching

Introduction, Brute force, the Boyer- Moore algorithm, Knuth-Morris-Pratt algorithm, Naïve String, Hars pool, Rabin Karp

TEXTBOOKS:

1. Fundamentals of data structures in java Sahni, Horowitz, Mehatha, Universities Press.
2. Introduction to Algorithms, TH Cormen, PHI

REFERENCES:

1. Design methods and analysis of Algorithms, SK Basu, PHI.
2. Data Structures & Algorithm Analysis in java, Mark Allen Weiss, Pearson Education.
3. Fundamentals of Computer Algorithms, 2nd Edition, Ellis Horowitz, Sartaj Sahni, Sanguthevar Rajasekaran, Universities Press.

ENTERPRISE CLOUD CONCEPTS (Professional Elective-I)

M.Tech SE I Year I Sem.

L	T	P	C
3	0	0	3

Course Objectives:

- Knowledge on significance of cloud computing and its fundamental concepts and models.

Course Outcomes:

- Understand importance of cloud architecture
- Illustrating the fundamental concepts of cloud security
- Analyze various cloud computing mechanisms
- Understanding the architecture and working of cloud computing.

UNIT-I

Understanding Cloud Computing: Origins and influences, Basic Concepts and Terminology, Goals and Benefits, Risks and Challenges.

Fundamental Concepts and Models: Roles and Boundaries, Cloud Characteristics, Cloud Delivery Models, Cloud Deployment Models.

UNIT-II

Cloud-Enabling Technology: Broadband Networks and Internet Architecture, Data Center Technology, Virtualization Technology.

Cloud Computing Mechanisms:

Cloud Infrastructure Mechanisms: Logical Network Perimeter, Virtual Server, Cloud Storage Device, Cloud Usage Monitor, Resource Replication.

UNIT-III

Cloud Management Mechanisms: Remote Administration System, Resource Management System, SLA Management System, Billing Management System, Case Study Example

Cloud Computing Architecture

Fundamental Cloud Architectures: Workload Distribution Architecture, Resource Pooling Architecture, Dynamic Scalability Architecture, Elastic Resource Capacity Architecture, Service Load Balancing Architecture, Cloud Bursting Architecture, Elastic Disk Provisioning Architecture, Redundant Storage Architecture, Case Study Example

UNIT-IV

Cloud-Enabled Smart Enterprises: Introduction, Revisiting the Enterprise Journey, Service-Oriented Enterprises, Cloud Enterprises, Smart Enterprises, The Enabling Mechanisms of Smart Enterprises

Cloud-Inspired Enterprise Transformations: Introduction, The Cloud Scheme for Enterprise Success, Elucidating the Evolving Cloud Idea, Implications of the Cloud on Enterprise Strategy, Establishing a Cloud-Incorporated Business Strategy.

UNIT-V

Transitioning to Cloud-Centric Enterprises: The Tuning Methodology, Contract Management in the Cloud

Cloud-Instigated IT Trans formations

Introduction, Explaining Cloud Infrastructures, A Briefing on Next-Generation Services, Service Infrastructures, Cloud Infrastructures, Cloud Infrastructure Solutions, Clouds for Business Continuity, The Relevance of Private Clouds, The Emergence of Enterprise Clouds

TEXTBOOKS:

1. Erl Thomas, Puttin iRicardo, Mahmood Zaigham,CloudComputing:Concepts,Technology&Architecture1stEdition,
2. Pethuru Raj, Cloud Enterprise Architecture, CRC Press

REFERENCES:

1. James Bond, The Enterprise Cloud, O'Reilly Media, Inc.

DEV-OPS (DEVELOPMENT OPERATIONS) (PE-I)

M.Tech SE I Year I Sem.

L	T	P	C
3	0	0	3

Pre-Requisites:

- Software Engineering
- Software Project Management

Course Objectives:

- Understand the skill sets and high-functioning teams involved in Agile, DevOps and related Methods to reach a continuous delivery capability.
- Implement automated system update and DevOps lifecycle.

Course Outcomes:

- Understand the various components of DevOps environment.
- Identify Software development models and architectures of DevOps
- Use different project management and integration tools.
- Select an appropriate testing tool and deployment model for project.

UNIT- I

Introduction to DevOps:

Introduction, Agile development model, DevOps and ITIL. DevOps process and Continuous Delivery, Release management, Scrum, Kanban, delivery pipeline, identifying bottlenecks.

UNIT- II

Software development models and DevOps:

DevOps Lifecycle for Business Agility, DevOps, and Continuous Testing. DevOps influence on Architecture: Introducing software architecture, The monolithic scenario, Architecture rules of thumb, The separation of concerns, Handling database migrations, Micro services and the data tier, DevOps, Architecture and resilience.

UNIT- III

Introduction to project management:

The need for source code control, the history of source code management, Roles and code, source code management system and migrations, shared authentication, Hosted Git servers, Different Git server implementations, Docker intermission, Gerrit, The pull request model, GitLab.

UNIT- IV

Integrating the system:

Build systems, Jenkins build server, Managing build dependencies, Jenkins plugins, and file system layout, The host server, Build slaves, Software on the host, Triggers, Job chaining and build pipelines, Build servers and infrastructure as code, Building by dependency order, Build phases, Alternative build servers, Collating quality measures.

UNIT- V

Testing Tools and Deployment:

Various types of testing, Automation of testing Pros and cons, Selenium - Introduction, Selenium Features, JavaScript testing, testing backend integration points, Test-driven development, REPL-driven development. Deployment of the system: Deployment systems, Virtualization stacks, code execution at the client, Puppet master and agents, Ansible, Deployment tools: Chef, Salt Stack and Docker.

TEXT BOOKS:

1. Joakim Verona., Practical DevOps, Packt Publishing, 2016.

REFERENCE BOOKS:

1. Deepak Gaikwad, Viral Thakkar. DevOps Tools from Practitioner's Viewpoint. Wiley publications.
2. Len Bass, Ingo Weber, Liming Zhu. DevOps: A Software Architect's Perspective. Addison Wesley.

DATABASE PROGRAMMING (Professional Elective-I)

M.Tech SE I Year I Sem.

L	T	P	C
3	0	0	3

Course Objectives:

- Knowledge on significance of SQL fundamentals.
- Evaluate functions and triggers of PL/SQL
- Knowledge on control structures, packages in PL/SQL and its applications

Course Outcomes:

- Understand importance of PL/SQL basics
- Implement functions and procedures using PL/SQL
- Understand the importance of triggers in data base

UNIT-I

PL/SQL Basics: Block Structure, Behavior of Variables in Blocks, Basic Scalar and Composite DataTypes,ControlStructures,Exceptions,BulkOperations,Functions,Procedures,andPackages,Transa ctionScope.

UNIT-II

Language Fundamentals & Control Structures: Lexical Units, Variables and Data Types, Conditional Statements, Iterative Statements, Cursor Structures, Bulk Statements, Introduction to Collections, Object Types: V array and Table Collections, Associative Arrays, Oracle Collection API.

UNIT-III

Functions and Procedures: Function and Procedure Architecture, Transaction Scope, Calling Subroutines, Positional Notation, Named Notation, Mixed Notation, Exclusionary Notation, SQL Call Notation, Functions, Function Model Choices, Creation Options, Pass-by-Value Functions, Pass-by-Reference Functions, Procedures, Pass – by – Value Procedures, Pass – by – Reference Procedures, Supporting Scripts.

UNIT-IV

Packages: Package Architecture, Package Specification, Prototype Features, Serially Reusable Precompiler Directive, Variables, Types, Components: Functions and Procedures, Package Body, Proto type Features, Variables, Types, Components: Functions and Procedures, Definer vs. Invoker Rights Mechanics, Managing Packages in the Database Catalog, Finding, Validating, and Describing Packages, Checking Dependencies, Comparing Validation Methods: Time stamp vs. Signature.

UNIT-V

Triggers: Introduction to Triggers, Database Trigger Architecture, Data Definition Language Triggers, Event Attribute Functions, Building DDL Triggers, Data Manipulation Language Triggers, Statement-Level Triggers, Row-Level Triggers, Compound Triggers, INSTEAD OF Triggers, System and Database Event Triggers, Trigger Restrictions, Maximum Trigger Size, SQL Statements, LONG and LONG RAW Data Types.

TEXTBOOKS:

1. Oracle Database 12cPL/SQL Programming Michael Mc Laughlin, McGraw Hill Education

REFERENCES:

1. Benjamin Rosenzweig, Elena Silvestrova Rakhimov,OraclePL/SQL by example FifthEdition
2. Dr. P.S. Deshpande, SQL&PL/SQLforOracle11gBlackBook

NETWORK PROGRAMMING (Professional Elective - I)

M.Tech SE I Year I Sem.

L	T	P	C
3	0	0	3

Course Objectives:
- To understand inter process and inter-system communication
- To understand socket programming in its entirety
- To understand usage of TCP/UDP / Raw sockets
- To understand how to build network applications

Course Outcomes:
- Understand socket API based programs
- Implement client-server applications using TCP and UDP sockets
- Understand Domain name servicing and multicasting routing protocols
- Analyze network programs using Linux Network based API

UNIT - I

Introduction to Network Programming: OSI model, UNIX standards, TCP and UDP & TCP connection establishment and Format, Buffer sizes and limitation, standard internet services, Protocol usage by common internet application.

Sockets: Address structures, value – result arguments, Byte ordering and manipulation function and related functions Elementary TCP sockets – Socket, connect, bind, listen, accept, fork and exec function, concurrent servers. Close function and related function.

UNIT - II

TCP client server: Introduction, TCP Echo server functions, Normal startup, terminate and signal handling server process termination, Crashing and Rebooting of server host shutdown of server host.

Elementary UDP sockets: Introduction UDP Echo server function, lost datagram, summary of UDP example, Lack of flow control with UDP, determining outgoing interface with UDP.

I/O Multiplexing: I/O Models, select function, Batch input, shutdown function, poll function, TCP Echo Server

UNIT - III

Socket options: get sock opt and set sock opt functions. Socket states, Generic socket option IPV6 socket option ICMPV6 socket option IPV6 socket option and TCP socket options.

Advanced I/O Functions-Introduction, Socket Timeouts, recv and send Functions, read v and write v Functions, recv msg and send msg Functions, Ancillary Data, How Much Data Is Queued?, Sockets and Standard I/O, T/TCP: TCP for Transactions.

UNIT - IV

Elementary name and Address conversions: DNS, get host by Name function, Resolver option, Function and IPV6 support, u name function, other networking information.

Daemon Processes and in etd Super server- Introduction, sys log d Daemon, sys log Function, daemon_init Function, inetd Daemon, daemon_inetd Function

Broadcasting- Introduction, Broadcast Addresses, Unicast versus Broadcast, dg_cli Function Using Broadcasting, Race Conditions

Multicasting- Introduction, Multicast Addresses, Multicasting versus Broadcasting on A LAN, Multicasting on a WAN, Multicast Socket Options, mcast_join and Related Functions, dg_cli Function Using Multicasting, Receiving MBone Session Announcements, Sending and Receiving, SNTP: Simple Network Time Protocol, SNTP

UNIT - V

Raw Sockets-Introduction, Raw Socket Creation, Raw Socket Output, Raw Socket Input, Ping Program, Traceroute Program, An ICMP Message Daemon,

Datalink Access- Introduction, BPF: BSD Packet Filter, DLPI: Data Link Provider Interface

Linux: SOCK_PACKET, libpcap: Packet Capture Library, Examining the UDP Checksum Field.

Remote Login: Terminal line disciplines, Pseudo-Terminals, Terminal modes, Control Terminals, rlogin Overview, RPC Transparency Issues.

TEXT BOOKS:

1. UNIX Network Programming, by W. Richard Stevens, Bill Fenner, Andrew M. Rudoff, Pearson Education
2. UNIX Network Programming, 1st Edition, - W. Richard Stevens. PHI.

REFERENCE BOOKS:

1. UNIX Systems Programming using C++ T CHAN, PHI.
2. UNIX for Programmers and Users, 3rd Edition Graham GLASS, King abls, Pearson Education
   Advanced UNIX Programming 2nd Edition M. J. ROCHKIND, Pearson Education

INTERNET OF THINGS (Professional Elective - I)

M.Tech SE I Year I Sem.

L	T	P	C
3	0	0	3

Course Objectives:

- To introduce the terminology, technology and its applications
- To introduce the concept of M2M (machine to machine) with necessary protocols
- To introduce the Python Scripting Language which is used in many IoT devices
- To introduce the Raspberry PI platform, that is widely used in IoT applications
- To introduce the implementation of web-based services on IoT devices

Course Outcomes:

- Interpret the impact and challenges posed by IoT networks leading to new architectural models.
- Compare and contrast the deployment of smart objects and the technologies to connect them tothe network.
- Appraise the role of IoT protocols for efficient network communication.
- Elaborate the need for Data Analytics and Security in IoT.
- Illustrate different sensor technologies for sensing real world entities and identify the applicationsof IoT in Industry.

UNIT - I

Introduction to Internet of Things –Definition and Characteristics of IoT, Physical Design of IoT – IoT Protocols, IoT communication models, Iot Communication APIs IoT enabled Technologies – Wireless Sensor Networks, Cloud Computing, Big data analytics,and Communication protocols, Embedded Systems, IoT Levels and Templates Domain Specific IoTs – Home, City, Environment, Energy, Retail, Logistics, Agriculture, Industry, healthand Lifestyle

UNIT - II

IoT and M2M – Software defined networks, network function virtualization, difference between SDN and NFV for IoT
Basics of IoT System Management with NETCOZF, YANG- NETCONF, YANG, SNMP NETOPEER

UNIT - III

Introduction to Python - Language features of Python, Data types, data structures, Control of flow, functions,modules, packaging, file handling, data/time operations, classes, Exception handling
Python packages - JSON, XML, HTTP Lib, URL Lib, SMTP Lib

UNIT - IV

IoT Physical Devices and Endpoints - Introduction to Raspberry PI-Interfaces (serial, SPI, I2C) Programming – Python program with Raspberry PI with focus of interfacing external gadgets, controlling output, and reading input from pins.

UNIT - V

IoT Physical Servers and Cloud Offerings – Introduction to Cloud Storage models and communication APIs Web server – Web server for IoT, Cloud for IoT, Python web application framework
Designing a RESTful web API

TEXT BOOKS:

1. Internet of Things - A Hands-on Approach, Arshdeep Bahga and Vijay Madisetti, Universities Press,2015, ISBN: 9788173719547
2. Getting Started with Raspberry Pi, Matt Richardson & Shawn Wallace, O'Reilly (SPD), 2014, ISBN: 9789350239759.

SOFTWARE PROJECT MANAGEMENT (Professional Elective-II)

M.Tech SE I Year I Sem.

L	T	P	C
3	0	0	3

Prerequisites: A course on "Software engineering".

Course Objectives:

- To develop skills in software project management
- The topics include - software economics; software development life cycle; artifacts of the process; work flows; check points; project organization and responsibilities; project control and process instrumentation;

Course Outcomes:

- Gain knowledge of soft ware economics, phases in the lifecycle of software development, project organization, project control and process instrumentation.
- Analyze the major and minor milestones, artifacts and metrics from management and technical perspective
- Design and develop software products using conventional and modern principles of software project management

UNIT-I

Conventional Software Management: The water fall model, conventional software Management performance. Evolution of Software Economics: Software economics, pragmatic software cost estimation.

UNIT-II

Improving Software Economics: Reducing Software product size, improving software processes, improving team effectiveness, improving automation, Achieving required quality, peer inspections.
The old way and the new: The principles of conventional software Engineering, principles of modern software management, transitioning to an iterative process.

UNIT-III

Life cycle phases: Engineering and production stages, inception, Elaboration, construction, Transition phases.
Artifacts of the process: The artifact sets, Management artifacts, Engineering artifacts, programmatic artifacts.
Model based soft ware architectures: A Management perspective and technical perspective.
Work Flows of the process: Software process work flows, Iteration work flows**.**

UNIT-IV

Check points of the process: Major milestones, Minor Milestones, Periodic status assessments.
Iterative Process Planning: work break down structures, planning guidelines, cost and schedule estimating, Iteration planning process, Pragmatic planning.
Project Organizations and Responsibilities: Line-of-Business Organizations, Project Organizations, evolution of Organizations. Process Automation: Automation Building blocks, The Project Environment.

UNIT-V

Project Control and Process instrumentation: The seven core Metrics, Management indicators, quality indicators, lifecycle expectations, pragmatic Software Metrics, Metrics Automation.
Tailoring the Process: Process discriminates.
Future Software Project Management: Modern Project Profiles, Next generation Software economics, modern process transitions.
Case Study: The command Center Processing and Display system- Replacement (CCPDSR).

Text Books:

1. Software Project Management, Walker Royce: Pearson Education, 2005.

Reference Books:

1. Software Project Management, Bob Hughes and Mike Cotterell: Tata McGraw-Hill Edition.
2. Software Project Management, Joel Henry, Pearson Education.
3. Software Project Management in practice, Pankaj Jalote, Pearson Education.2005.

SOFTWARE METRICS (Professional Elective-II)

M.Tech SE I Year I Sem.

L	T	P	C
3	0	0	3

Course Objectives:

- To gain basic knowledge about metrics, measurement theory and related terminologies
- To learn measure the quality level of internal and external attributes of the software product
- To introduce the basics of software reliability and to illustrate how to perform planning, executing and testing for software reliability
- To explore various metrics and models of software reliability
- To compare various models of software reliability based on its application

Course Outcomes: Upon completion of the course, students shall be able to

- Identify and apply various software metrics, which determines the quality level of software
- Identify and evaluate the quality level of internal and external attributes of the software product
- Compare and Pick out the right reliability model for evaluating the software
- Evaluate the reliability of any given software product
- Design new metrics and reliability models for evaluating the quality level of the software based on the requirement.

UNIT-I

What Is Software Quality: Quality: Popular Views, Quality Professional Views, Software Quality, Total Quality Management, and Summary.

Fundamentals of Measurement Theory: Definition, Operational Definition, and Measurement, Level of Measurement, Some Basic Measures, Reliability and Validity, Measurement Errors, Be Careful with Correlation, Criteria for Causality, Summary.

Software Quality Metrics Overview: Product Quality Metrics, In Process Quality Metrics, Metrics for Software Maintenance, Examples for Metrics Programs, Collecting software Engineering Data.

UNIT-II

Applying the Seven Basic Quality Tools in Software Development: Ishikawa's Seven Basic Tools, Checklist, Pareo Diagram, Histogram, Run Charts, Scatter Diagram, Control Chart, Cause, and Effect Diagram.

The Rayleigh Model: Reliability Models, the Rayleigh Model Basic Assumptions, Implementation, Reliability and Predictive Validity.

UNIT-III

Complexity Metrics and Models: Lines of Code, Halstead's Software Science, Cyclomatic Complexity Syntactic Metrics, An Example of Module Design Metrics in Practice.

Metric and Lessons Learned for Object Oriented Projects: Object Oriented Concepts and Constructs, Design and Complexity Metrics, Productivity Metrics, Quality and Quality Management Metrics, Lessons Learned For object-oriented Projects.

UNIT-IV

Availability Metrics: Definition and Measurement of System Availability, Reliability Availability and Defect Rate, Collecting Customer Outage Data for Quality Improvement, In Process Metrics for Outage and Availability.

Conducting Software Project Assessment: Audit Ad Assessment, Software Process Maturity Assessment and Software Project Assessment, Software Process Assessment A Proponed Software Project Assessment Method.

UNIT-V

Dos and Don'ts of Software Process Improvement: Measuring Process Maturity, Measuring Process Capability, Staged Versus Continuous Debating Religion, Measuring Levels Is Not Enough, Establishing the Alignment Principle, Take Time Getting Faster, keep it Simple or Face Decomplexification, Measuring the Value of Process Improvement, Measuring Process Compliance,

Celebrate the Journey Not Just the Destination. Using Function Point Metrics to Measure Software Process Improvement: Software Process Improvement Sequences, Process Improvement Economies, Measuring Process Improvement at Activity Levels.

Text Books:

1. Norman E Fentor and Share Lawrence P flieger. ”Software Metrics”. International Thomson Computer Press, 1997.
2. Stephen H Khan: Metrics and Models in Software Quality Engineering, Pearson 2nd edition 2013.

References:

1. S.A. Kelkar, “Software quality and Testing, PHI Learning, Pvt., Ltd., New Delhi 2012.
2. Watts S Humphrey, “Managing the Software Process”, Pearson Education Inc, 2008.
3. MaryBethChrissis,MikeKonradandSandyShrum,“CMMI”,PearsonEducation(Singapore)Pvt.Ltd .,2003
4. Philip B Crosby," Quality is Free: The Art of Making Quality Certain", Mass Market, 1992.

FULL STACK DEVELOPMENT USING MERN
(Skill Enhancement Course)

L	T	P	C
0	1	2	2

M.Tech SE I Year I Sem.

Course Objectives:

The course aims to:

- Provide strong foundations in web development technologies (HTML, CSS, JavaScript, ES6).
- Introduce server-side programming with Node.js and Express.js for building scalable applications.
- Enable students to work with relational (MySQL) and non-relational (MongoDB) databases.
- Impart skills to design and develop interactive user interfaces using ReactJS.
- Enhance problem-solving abilities through full-stack web application development experiments.

Course Outcomes (COs) :

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

CO1: Apply fundamental web technologies (HTML, CSS, JavaScript, ES6) to design responsive web pages.
CO2: Develop server-side applications using Node.js and Express.js with REST API integration.
CO3: Perform database operations using MySQL and MongoDB and integrate them with backend services.
CO4: Design and implement dynamic, component-based user interfaces using ReactJS.
CO5: Develop and deploy full-stack applications by combining frontend, backend, and database skills.
CO6: Demonstrate problem-solving, debugging, and version control skills in web development projects.

Module 1: Web Development Fundamentals

Fundamentals of Web Design, Webpage and Website, Web application HTML Typography, Images, Tables, Lists, Hyperlinks etc. CSS Syntax and usage, CSS Selectors, CSS on body, CSS on Text, CSS on Links, CSS on Tables, CSS on Lists, CSS on Forms, CSS on Images, CSS on DIV, W3.CSS Framework

List of Experiments :

- **HTML & CSS Basics** – Create a personal portfolio webpage using HTML (headings, lists, tables, hyperlinks, forms) and style it with CSS selectors.
- **Responsive Layout** – Develop a responsive webpage using DIV, CSS box model, and W3.CSS framework.
- **Styled Components** – Design a webpage for a college event with images, tables, and styled navigation menu using CSS.

Module 2: JavaScript and ECMA Script 6

JavaScript Fundamentals - Grammar and types, Control flow and error handling - Loops, Function - Objects, Arrays, Promises - ES6 Let and const, Template literals - Arrow Function, Default parameter, Async Await

List of Experiments :

- **JavaScript Fundamentals** – Build a simple calculator app using functions, loops, and control flow.
- **Array & Object Manipulation** – Write a program using ES6 features (let/const, arrow functions, template literals) to manage student records.
- **Async Programming** – Create a webpage that fetches and displays random user data from a public API using Promises and Async/Await.

Module 3: Node.js

overview, Node.js - basics and setup - Node.js console, Node.js command utilities - Node.js modules, concepts - Node.js events, database access - Node.js with Express.js, Express.js Request/Response -

Express.js Get, Express.js Post - Express.js Routing, Express.js Cookies - Express.js File Upload, Middleware - Express.js Scaffolding, Template

List of Experiments:

- **Node.js Basics** – Write a Node.js script to create a local server and display "Hello World" in the browser.
- **Express.js Routing** – Build a REST API with Express.js that handles GET and POST requests for a student information system.
- **File Handling** – Develop a Node.js application to upload, read, and display a text/JSON file using Express middleware.

Module 4: MySQL and MongoDB

MySQL Concepts - Create, Read, Update, Delete Operation - SQL and NoSQL concepts - Create and manage MongoDB - Migration of data into MongoDB - MongoDB with NodeJS - Services offered by MongoDB

List of Experiments :

- **MySQL CRUD** – Create a MySQL database for employee records and perform Create, Read, Update, Delete (CRUD) operations.
- **MongoDB CRUD with Node.js** – Build a Node.js application that connects to MongoDB and manages student data.
- **Migration Project** – Write a script to migrate data from MySQL to MongoDB and display it through a Node.js API.

Module 5: React JS

ReactJS introduction and overview - ReactJS installation and environment setup - Introducing JSX, Rendering Elements - Components and Props - State and Lifecycle - Handling Events - Conditional Rendering - Lists and Keys, Forms - Lifting State Up

List of Experiments :

- **React Components** – Build a React app to display a list of courses using functional components and props.
- **State & Events** – Create a counter and a form component in React using useState and event handling.
- **Conditional Rendering & Lists** – Develop a React to-do list application with add/delete functionality and conditional rendering of completed tasks.

Textbooks

1. **Alex Banks, Eve Porcello** – *Learning React: Modern Patterns for Developing React Apps*, O'Reilly.
2. **StoyanStefanov** – *React Up & Running: Building Web Applications*, O'Reilly.
3. **Mario Casciaro, Luciano Mammino** – *Node.js Design Patterns*, Packt.
4. **Seyed M.M. Iravani** – *Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics*, O'Reilly.

Reference Books

1. **Robin Wieruch** – *The Road to React*, Leanpub.
2. **Carl Rippon** – *React 18 Design Patterns and Best Practices*, Packt.
3. **KirupaChinnathambi** – *Learning React: A Hands-On Guide to Building Web Applications*, Addison-Wesley.
4. **Ethan Brown** – *Web Development with Node and Express: Leveraging the JavaScript Stack*, O'Reilly.
5. **Kristina Chodorow** – *MongoDB: The Definitive Guide*, O'Reilly.
6. **Ben Forta** – *SQL in 10 Minutes, Sams Teach Yourself*, Sams Publishing.

SOFTWARE RELIABILITY (Professional Elective-II)

M.Tech SE I Year I Sem. **L T P C**

3 0 0 3

Course Objectives:

- To learn about the engineering techniques for developing and maintaining reliable of software systems.
- To measure the reliability of software systems.
- To understand about fault prevention, fault removal, fault tolerance and failure fore casting in software systems.
- To learn different time dependent and time independents of software reliability models and design reliability models for software systems.

Course Outcomes: After completing this course, students will be able to

- Explain the differences between software and hardware reliability, and use reliability metrics to assess software dependability.
- Apply techniques for preventing, managing, and forecasting software failures, and understand dependability concepts such as maintenance and reliability modeling.
- Use various software reliability models, including time-dependent and time-independent models, to measure and predict software reliability.
- Perform reliability testing using operational profiles, and apply methods for regression testing and test selection.
- Analyze the predictive accuracy of reliability models, recalibrate models, and improve prediction techniques for more reliable software systems.

UNITI

Basic Ideas of Software Reliability, Hardware reliability vs. Software reliability, Reliability metrics, Failure and Faults – Prevention, Removal, Tolerance, Forecast, Dependability Concept – Failure Behavior, Characteristics, Maintenance Policy, Reliability and Availability Modeling, Reliability Evaluation Testing methods, Limits, Starvation, Coverage, Filtering, Microscopic Model of Software Risk.

UNIT II

Computation of software reliability, Functional and Operational Profile, Operational Profiles – Difficulties, Customer Type, User Type, System Mode, Test Selection-Selecting Operations, Regression Test.

UNIT III

Classes of software reliability Models, Time Dependent Software Reliability Models: Time between failure reliability Models, Fault Counting Reliability Models.

UNIT IV

Time Independent Software Reliability Models: Fault injection model of Software Reliability, Input Domain Reliability Model, Orthogonal defect classification, Software availability Models.
Software Reliability Modeling: A general procedure for reliability modeling.

UNIT V

Short and Long Term Prediction, Model Accuracy, Analyzing Predictive Accuracy – Outcomes, PLR, U and Y Plot, Errors and Inaccuracy, Recalibration – Detecting Bias, Different Techniques, Power of Recalibration, Limitations in Present Techniques, Improvements.

TEXTBOOKS:

1. J.D. Musa, *Software Reliability Engineering*,McGrawHill,NewYork,2004
2. H.Pham, *Software Reliability*, Springer Verlag,NewYork,2000

REFERENCEBOOK:

1. Patric D.T.O Connor, *Practical Reliability Engineering,4th Edition*, John Wesley & Sons,2003
2. D. Reled, *Software Reliability Methods*, Springer Verlag,NewYork,2001

SOFTWARE AGENTS (Professional Elective-II)

M.Tech SE I Year I Sem.

L	T	P	C
3	0	0	3

Course Objectives: The objective of this course is to make students to

- Learn the principles and fundamentals of designing agents
- Study the architecture design of different agents.
- Learn to do detailed design of the agents
- Explore the role of agents in assisting the users in day to day activities

Course Outcomes: After completing this course, students will be able to

- Explain the concepts of intelligent agents and multi-agent systems, including their types and environments.
- Apply methodologies like Prometheus for specifying system goals and functionalities when building agents.
- Design agent architectures, including interaction diagrams and protocols, ensuring completeness and consistency in system design.
- Create detailed designs for agent systems, refining descriptors and ensuring consistency between various artifacts.
- Develop agents that improve user experience, facilitate information sharing, and assist in intelligent tasks.

UNIT I

INTRODUCTION

Agents and Multi Agent Systems-Intelligent Agent-Concepts of Building Agent-Situated Agents-Proactive and Reactive agents - Challenging Agent Environment - Social Agents - Agent Execution Cycle - Prometheus Methodology - Guidelines for using Prometheus - Agent Oriented Methodologies – System Specification – Goal Specification – Functionalities – Scenario Development – Interface Description –Checking for Completeness and Consistency.

UNIT II

ARCHITECTURALDESIGN

Agent Types - Grouping Functionalities - Agent Coupling - Develop Agent Descriptors – Interactions - Interaction Diagram from Scenarios - Interaction Protocol from Interaction Diagram - Develop Protocol and Message Descriptors – Architectural Design- Identifying the Boundaries of Agent System – Percepts and Action - Shared Data Objects – System Overview – Checking for Completeness and Consistency.

UNIT III

DETAILEDDESIGN

Capability Diagrams - Sub Tasks - Alternative Programs - Events and Messages - Action and Percept Detailed Design - Data - Develop and Refine Descriptors - Missing or Redundant Items - Consistency between Artifacts - Important Scenarios - Implementing Agent Systems - Agent Platform-JACK

UNIT IV

AGENTS ANDUSEREXPERIENCE

Interact with Agents - Agents from Direct Manipulation to Delegation - Interface Agents - Designing Agents - Direct Manipulation versus Agents - Agents for Information Sharing and Coordination – Agents that Reduce Work and Information Overload – Kid Sim: Programming Agents without a Programming Language.

UNIT V

AGENTSFORINTELLIGENTASSISTANCE

Computer Characters - Software Agents for Cooperative Learning - Integrated Agents - Agent Oriented Programming - KQML as an Agent Communication Language-Agent Based Framework for Inter operability - Agents for Information Gathering - KaoS - Communicative Actions for Artificial Agents -Mobile Agents.

TEXTBOOKS:

1. Lin Padgham and Michael Winikoff. Developing Intelligent Agent Systems: A Practical Guide: JohnWiley&sons Publication,2004.
2. JeffreyM.Bradshaw.SoftwareAgents:MITPress,1997.REFERENCEBOOK:StevenF.RailsBack and Volker Grimm. Agent – Based and Individual Based modeling: A Practical Introduction:PrincetonUniversityPress,2012.

SECURE CODING PRACTICES (Professional Elective - V)

M.Tech SE I Year I Sem.

L	T	P	C
3	0	0	3

Course Objectives:

- To understand the various security attacks
- To learn how to recognize to coding errors
- To understand techniques for developing a secure application.

Course Outcomes: On completion of this course, the student will be able to :

- Understand various attacks like DoS, buffer over flow, web specific, database specific, web-spoofing attacks.
- Demonstrate skills needed to deal with common programming errors that lead to most security problems and to learn how to develop secure applications.
- Identify the nature of the threats to software and incorporate secure coding practices throughout the planning and development of the product.

UNIT – I

Introduction: Security, CIAT riad, Viruses, Trojans, and Worms in a Nutshell, Security Concepts exploit, threat, vulnerability, risk, attack. Malware Terminology: Root kits, Trapdoors, Botnets, Key loggers, Honey pots. Active and Passive Security Attacks. IP Spoofing, Teardrop, DoS, DDoS, XSS, SQL injection, Man in middle Attack, Format String attack. Types of Security Vulnerabilities-buffer over flows, Invalidated input, race conditions, access-control problems, weaknesses in authentication, authorization, or cryptographic practices. Access Control Problems.

UNIT – II

Need for secure systems: Proactive Security development process, Secure Software Development Cycle (S-SDLC), Security issues while writing SRS, Design phase security, Development Phase, Test Phase, Maintenance Phase, Writing Secure Code – Best Practices SD3 (Secure by design, default and deployment), Security principles and Secure Product Development Timeline.

UNIT – III

Threat modeling process and its benefits: Identifying the Threats by Using Attack Trees,
Risk Mitigation Techniques and Security Best Practices. Security techniques, authentication, authorization.
Secure Coding Techniques: Protection against DoS attacks, Application Failure Attacks, CPU Starvation Attacks. Security Issues in C Language: String Handling, Avoiding Integer Over flows and Underflows and Type Conversion Issues- Memory Management Issues, Code Injection Attacks.

UNIT – IV

Database and Web-specific issues: SQL Injection Techniques and Remedies, Race conditions, Time of Check, Time of Use and its protection mechanisms. Validating Input and Inter process Communication, Securing Signal Handlers and File Operations. XSS scripting attack and its types – Persistent and Non persistent attack XSS Counter measures and bypassing the XSS Filters.

UNIT – V

Testing Secure Applications: Security code overview, secure software installation. The Role of the Security Tester, Building the Security Test Plan. Testing HTTP- Based Applications, Testing File-Based Applications.

Suggested Reading:

1. Michael Howard and David LeBlanc,‖ Writing Secure Code‖, Microsoft Press, 2nd Edition, 2004.
2. Jason Deckard, ¯Buffer Overflow Attacks: Detect, Exploit, Prevent¯, Syngress,1st Edition, 2005.
3. Frank Swiderski and Window Snyder, ¯Threat Modeling‖, Microsoft Professional,1st Edition, 2004.

ADVANCED DATA STRUCTURES LAB (Lab - I)

M.Tech SE I Year I Sem.

L	T	P	C
0	0	4	2

Pre requisites: A course on Computer Programming & Data Structures

Course Objectives:

- Introduces the basic concepts of Abstract Data Types.
- Reviews basic data structures such as stacks and queues.
- Introduces a variety of data structures such as hash tables, search trees, tries, heaps, graphs, and B-trees.
- Introduces sorting and pattern matching algorithms.

Course Outcomes:

- Ability to select the data structures that efficiently model the information in a problem.
- Ability to assess efficiency trade-offs among different data structure implementations or combinations.
- Implement and know the application of algorithms for sorting and pattern matching.
- Design programs using a variety of data structures, including hash tables, binary and general tree structures, search trees, tries, heaps, graphs, and B-trees.

List of Programs

1. Write a program to perform the following operations:
   a. Insert an element in to a binary search tree.
   b. Delete an element from a binary search tree.
   c. Search for a key element in a binary search tree.

2. Write a program for implementing the following sorting methods:
   a) Merge sort b)Heap sort c)Quick sort

3. Write a program to perform the following operations:
   a. Insert an element into a B-tree.
   b. Delete an element from a B-tree.
   c. Search for a key element in a B-tree.

4. Write a program to perform the following operations:
   a. Insert an element in to a Min-Max heap
   b. Delete an element from a Min-Max heap
   c. Search for a key element in a Min-Max heap
5. Write a program to perform the following operations:
   a. Insert an element into a Leftist tree
   b. Delete an element from a Leftist tree
   c. Search for a key element in a Leftist tree
6. Write a program to perform the following operations:
   a. Insert an element in to a binomial heap
   b. Delete an element from a binomial heap.
   c. Search for a key element in a binomial heap

7. Write a program to perform the following operations:
   a. Insert an element into a AVL tree.
   b. Delete an element from a AVL search tree.
   c. Search for a key element in a AVL search tree.

8. Write a program to perform the following operations:
   a. Insert an element into a Red-Black tree.
   b. Delete an element from a Red- Black tree.
   c. Search for a key element in a Red-Black tree.

9. Write a program to implement all the functions of a dictionary using hashing.

10. Write a program for implementing Knuth-Morris-Pratt pattern matching algorithm.

11. Write a program for implementing Brute Force pattern matching algorithm.

12 Write a program for implementing Boyer pattern matching algorithm.

TEXTBOOKS:

1. Fundamentals of Data structures inC,E. Horowitz, S.Sahni and Susan Anderson Freed,2nd Edition, Universities Press
2. DataStructuresUsingC-A.S.Tanenbaum,Y.Langsam,andM.J.Augenstein,PHI/Pearsoneducation.
3. Introduction to Data Structures in C , Ashok Kamthane,1stEdition, Pearson.

REFERENCES:

1. The C Programming Language, B.W. Kernighan, Dennis M. Ritchie, PHI / Pearson Education
2. C Programming with problem solving, J.A. Jones & K. Harrow, Dream tech Press
3. Data structures: A Pseudo code Approach with C, R.F.Gilberg And B.A. Forouzan, 2nd Edition, Cengage Learning.

ENTERPRISE CLOUD CONCEPTS LAB (Professional Elective- I Lab)

M.Tech SE I Year I Sem.

L	T	P	C
0	0	4	2

Course Objectives:

- Knowledge on significance of cloud computing and its fundamental concepts and models.

Course Outcomes:

- Understand importance of cloud architecture
- Illustrating the fundamental concepts of cloud security
- Analyze various cloud computing mechanisms
- Understanding the architecture and working of cloud computing.

List of Experiments:

1. Install Virtual box/ VMware Work station with different flavors of linux or windows OS on top of windows 7 or 8.
2. Install a C compiler in the virtual machine created using virtual box and execute Simple Programs
3. Install Google App Engine. Create a hello world app and other simple web applications using python / java.
4. Findaproceduretotransferthefilesfromonevirtualmachinetoanothervirtualmachine.
5. Find a procedure to launch virtual machine using try stack (Online Open stack Demo Version)
6. Install Hadoop single node cluster and run simple applications like word count.

E-Resources:

1. https://www.iitk.ac.in/nt/faq/vbox.html
2. https://www.google.com/urlsa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwjqrNG0za73AhXZt1YBHZ21DWEQFnoECAMQAQ&url=http%3A%2F%2Fwww.cs.columbia.edu%2F~sedwards%2Fclasses%2F2015%2F1102-fall%2Flinuxvm.pdf&usg=AOvVaw3xZPuF5xVgk-AQnBRsTtHz
3. https://www.cloudsimtutorials.online/cloudsim/
4. https://edwardsamuel.wordpress.com/2014/10/25/tutorial-creating-openstack-instance-in-trystack/
5. https://www.edureka.co/blog/install-hadoop-single-node-hadoop-cluster

DEV-OPS LAB (PE-1LAB)

M.Tech SE I Year I Sem.

L	T	P	C
0	0	4	2

Course Objectives:

- Develop a sustainable infrastructure for applications and ensure high scalability. DevOps aims to shorten the software development lifecycle to provide continuous delivery with high-quality.

Course Outcomes:

- Understand the need of DevOps tools.
- Understand the environment for a software application development.
- Apply different project management, integration and development tools.
- Use Selenium tool for automated testing of application.

List of Experiments:

1. Write code for a simple user registration form for an event.
2. Explore Git and GitHub commands.
3. Practice Source code management on GitHub. Experiment with the source code in exercise 1.
4. Jenkins installation and setup, explore the environment.
5. Demonstrate continuous integration and development using Jenkins.
6. Explore Docker commands for content management.
7. Develop a simple containerized application using Docker.
8. Integrate Kubernetes and Docker
9. Automate the process of running containerized application for exercise 7 using Kubernetes.
10. Install and Explore Selenium for automated testing.
11. Write a simple program in JavaScript and perform testing using Selenium.
12. Develop test cases for the above containerized application using selenium.

TEXT BOOK:

1. Joakim Verona., Practical DevOps, Packt Publishing, 2016.

REFERENCE BOOKS:

1. Deepak Gaikwad, Viral Thakkar. DevOps Tools from Practitioner's Viewpoint. Wiley publications.
2. Len Bass, Ingo Weber, Liming Zhu. DevOps: A Software Architect's Perspective. Addison Wesley.

DATABASE PROGRAMMING LAB (Lab–PE1)

M.Tech SE I Year I Sem.

L	T	P	C
0	0	4	2

Course Objectives:

- Knowledge on significance of SQL fundamentals.
- Evaluate functions and triggers of PL/SQL
- Knowledge on control structures, packages in PL/SQL and its applications

Course Outcomes:

- Understand importance of PL/SQL basics
- Implement functions and procedures using PL/SQL
- Understand the importance of triggers in database

List of Experiments:

1. Write a Pl/SQL program using FOR loop to insert ten rows into a database table.
2. Given the table EMPLOYEE (Employee No, Name, Salary, Designation, Dept ID), write a cursor to select the five highest paid employees from the table.
3. Illustrate how you can embed PL/SQL in a high-level host language such as C/Java and demonstrate show a banking debit transaction might be done.
4. Given an integer i, write a PL/SQL procedure to insert the tuple (i, 'xxx') into a given relation.
5. Write a PL/SQL program to demonstrate Exceptions.
6. Write a PL/SQL program to demonstrate Cursors.
7. Write a PL/SQL program to demonstrate Functions.
8. Write a PL/SQL program to demonstrate Packages.
9. Write PL/SQL queries to create Procedures.
10. Write PL/SQL queries to create Triggers.

NETWORK PROGRAMMING LAB

M.Tech SE I Year I Sem.

L	T	P	C
0	0	4	2

Course Objectives:

- Understand inter process and inter-system communication
- Understand socket programming in its entirety
- Understand usage of TCP/UDP / Raw sockets
- Understand how to build network applications

Course Outcomes:

- Write socket API based programs
- Design and implement client-server applications using TCP and UDP sockets
- Analyze network programs

List of Experiments

1. Implement programs for Inter Process Communication using PIPE, Message Queue and Shared Memory.
2. Write a programme to create an integer variable using shared memory concept and increment the variable simultaneously by two processes. Use semaphores to avoid race conditions.
3. Design TCP iterative Client and server application to reverse the given input sentence
4. Design TCP iterative Client and server application to reverse the given input sentence
5. Design TCP client and server application to transfer file
6. Design a TCP concurrent server to convert a given text into upper case using multiplexing system call "select"
7. Design a TCP concurrent server to echo given set of sentences using poll functions
8. Design UDP Client and server application to reverse the given input sentence
9. Design UDP Client server to transfer a file
10. Design using poll client server application to multiplex TCP and UDP requests for converting a given text into upper case.
11. Design a RPC application to add and subtract a given pair of integers

TEXT BOOKS:

1. UNIX Network Programming, by W. Richard Stevens, Bill Fenner, Andrew M. Rudoff, Pearson Education.
2. UNIX Network Programming, 1 st Edition, - W. Richard Stevens. PHI.

INTERNET OF THINGS LAB (PE-1LAB)

M.Tech SE I Year I Sem.

L	T	P	C
0	0	4	2

Course Objectives:

- Understand the basic concepts and architecture of the Internet of Things (IoT).
- Learn the implementation of IoT devices and protocols.
- Develop skills in integrating sensors, actuators, and communication interfaces.
- Familiarize with IoT platforms and cloud services for data management.
- Acquire knowledge of various IoT applications in different domains.
- Develop skills in programming IoT devices for real-time data processing

Course Outcomes: At the end of this course, students will be able to

- Demonstrate the working of various IoT devices and sensors.
- Program IoT devices for different applications using relevant protocols.
- Interface various sensors and actuators with IoT platforms.
- Analyze data collected from IoT devices using cloud-based services.
- Apply IoT technologies to solve real-world problems.
- Design and implement a basic IoT system.

Python Basic exercises

1. Write a Python program that reads 10 integers from keyboard and prints the average of even numbers and odd numbers separately
2. Write a Python program that prints the grade of a student when internal and external marks are given. A candidate is declared Failed (Grade=F), if Total marks <50orExternalmarks<25.
3. If a candidate is passed, then Grade is given as follows:

Condition	Grade
50 <=total marks<60	E
60 <=total marks<70	D
70 <=total marks<80	C
80 <=total marks<90	B
Total marks>=70	A

4. Create a table in MySQL that stores the status of devices in a house with the following data (Device ID, Device Name and Device State, last altered date and time). Now write a Python program that reads and alters the state of a given device. The date format is “YYYY-MM-DD: HH-mm-ss” where mm is minutes and ss is seconds.
5. Write a Python program that loads all the states of the devices into a dictionary from the table mentioned above.
6. Write a Python program that sorts the device states based on the last altered time.
7. Write a Python program that reads a string from keyboard and prints the count of each alphabet in the string.
8. Write a Python program that reads a page from internet and prints it on the screen.
9. Write a Python program that reads and modifies an XML file.
10. Write a Python program that reads and alters JSON data from a data base table.
11. Write a client-server Python program that uses socket connection to implement a time server. The client will connect to the server and the server sends the current time as “YYYY-MM-DD: HH-mm-ss” format. This value should be printed on the client side.
12. Write a Python program that generates 10 random numbers and stores them in a text file one per line. Now write another Python program that reads this data into a list and shows them
13. Write a program that reads key-value pair data from a file and stores the min a database table
14. Write a Python program that reads a time string in the format of “YYYY-MM-DD:HH-mm-ss” and prints its components separately.
15. Write a Python program that reads data from a table and writes it to a text file using tab as field separator and new line as record separator and vice versa.

Raspberry Pi Experiments*:* Use Raspberry Pi for all the experiments

1. Connect an LED to GPIO pin 25 and control it through command line
2. Connect an LED to GPIO pin 24and aSwitchtoGPIO25 and control the LED with the switch. The state of LED should toggle with every press of the switch
3. UseDHT11temperature sensor and print the temperature and humidity of the room with an interval of 15 seconds
4. Use joy stick and display the direction on the screen
5. Use Light Dependent Resistor (LDR) and control an LED that should switch-on/off depending on the light.
6. Create a traffic light signal with three colored lights (Red, Orange and Green) with a duty cycle of 5-2-10seconds.
7. User rotary encoder and print the position of the shaft on the console
8. Control a servomotor angle that is taken from the key board
9. Switch on and switch of a DC motor based on the position of a switch
10. Convert an analog voltage to digital value and show it on the screen.
11. Create a door lock application using a reed switch and magnet and give a beep when the door is opened.
12. Control a 230Vdevice(Bulb) with Rasp berry Pi using a relay
13. Control a 230Vdevice using a threshold temperature, using temperature sensor.
14. Simulate an earthquake alarm using vibration sensor and give an alarm when vibration is detected.
15. Create an application that hast here LEDs (Red, Green and white).The LEDs should follow the cycle(All Off, Red On, Green On, White On)for each clap(use sound sensor).
16. Create a web application for the above applications wherever possible with suitable modifications to get input and to send output.

RESEARCH METHODOLOGY & IPR

M.Tech SE I Year I Sem.

L	T	P	C
2	0	0	2

Pre requisite: None

Course Objectives:

- To understand the research problem
- To know the literature studies, plagiarism and ethics
- To get the knowledge about technical writing
- To analyze the nature of intellectual property rights and new developments
- To know the patent rights

Course Outcomes: At the end of this course, students will be able to

- Understand research problem formulation.
- Analyze research related information
- Follow research ethics
- Understand that today's world is controlled by Computer, Information Technology, but tomorrow world will be ruled by ideas, concept, and creativity.
- Understanding that when IPR would take such important place in growth of individuals & nation, it is needless to emphasis the need of information about Intellectual Property Right to be promoted among students in general & engineering in particular.
- Understand that IPR protection provides an incentive to inventors for further research work and investment in R & D, which leads to creation of new and better products, and in turn brings about, economic growth and social benefits.

UNIT-I:

Meaning of research problem, Sources of research problem, Criteria Characteristics of a good research problem, Errors in selecting a research problem, Scope and objectives of research problem. Approaches of investigation of solutions for research problem, data collection, analysis, interpretation, Necessary instrumentations

UNIT-II:

Effective literature studies approaches, analysis, Plagiarism, Research ethics

UNIT-III:

Effective technical writing, how to write report, Paper Developing a Research Proposal, Format of research proposal, a presentation and assessment by a review committee

UNIT-IV:

Nature of Intellectual Property: Patents, Designs, Trade and Copyright. Process of Patenting and Development: technological research, innovation, patenting, development. International Scenario: International cooperation on Intellectual Property. Procedure for grants of patents, Patenting under PCT.

UNIT-V:

Patent Rights: Scope of Patent Rights. Licensing and transfer of technology. Patent information And databases. Geographical Indications. New Developments in IPR: Administration of Patent System. New developments in IPR; IPR of Biological Systems, Computer software etc. Traditional Knowledge Case Studies, IPR and IITs.

TEXTBOOKS

1. StuartMelvilleandWayneGoddard,"Researchmethodology:anintroductionforscience&engineering students'"
2. WayneGoddardandStuartMelville,"ResearchMethodology:AnIntroduction"

REFERENCES:

1. RanjitKumar,2ndEdition,“ResearchMethodology:AStepbyStepGuideforbeginners”
2. Halbert,“ResistingIntellectualProperty”,Taylor&FrancisLtd,2007.
3. Mayall,“Industrial Design”,McGrawHill,1992.
4. Niebel, “Product Design”, McGraw-Hill, 1974.
5. Asimov,“IntroductiontoDesign”,PrenticeHall,1962.
6. Robert P. Merges, Peter S. Menell, Mark A. Lemley, “Intellectual Property in New Technological Age”,2016.
7. T.Ramappa,“IntellectualPropertyRightsUnderWTO”,S.Chand,2008

Design Patterns (PC–III)

M.Tech SE I Year II Sem.

L	T	P	C
3	0	0	3

Pre requisites:

- A Course on Software Engineering"
- A Course on "Object Oriented Programming Through Java"

Course Objectives:

- The aim of the course is to appreciate the idea behind Design Patterns in handling common problems faced during building an application
- This course covers all pattern types from creational to structural, behavioral to concurrency and highlights the scenarios when one pattern must be chosen over others.

Course Outcomes:

- Create software designs that are scalable and easily maintainable
- Understand the best use of Object-Oriented concepts for creating truly OOP programs
- Use creational design patterns in software design for class instantiation
- Use structural design patterns for better class and object composition
- Use behavioral patterns for better organization and communication between the objects
- Use refactoring to compose the methods for proper code packaging
- Use refactoring to better organize the class responsibilities of current code

UNIT-I:

Introduction: What is a design pattern? Design patterns in Smalltalk MVC, Describing Design Patterns, The Catalog of Design Patterns, Organizing the Catalog, How Design Patterns Solve Design Problems, How to Selecta Design Pattern, How to Use a Design Pattern.

UNIT-II:

Designing a Document Editor: Design Problems, Document Structure, Formatting, Embellishing the User Interface, Supporting Multiple Look-and-Feel Standards, Supporting Multiple Window Systems, User Operations Spelling Checking and Hyphenation, Summary

UNIT-III:

Creational Patterns: Abstract Factory, Builder, Factory Method, Prototype, Singleton, Discussion of Creational Patterns.

UNIT-IV:

Structural Pattern: Adapter, Bridge, Composite, Decorator, Façade, Fly weight, Proxy

UNIT-V:

Behavioral Patterns: Chain of Responsibility, Command, Interpreter, Iterator, Mediator, Memento, Observer, State, Strategy, Template Method, Visitor.

Text Book:

1. Design Patterns, Erich Gamma, Pearson Education

Reference Books:

1. Pattern's in Java, Vol-I, Mark Grand, Wiley Dream Tech.
2. Patterns in Java, Vol-II, Mark Grand, Wiley Dream Tech.
3. Java Enterprise Design Patterns Vol-III, Mark Grand, Wiley Dream Tech.
   Head First Design Patterns, Eric Free man, O' reily publications

SOFTWARE QUALITY ASSURANCE (Professional Elective – V)

M.Tech SE I Year II Sem.

L	T	P	C
3	0	0	3

Course Objectives:

- Assurance of software quality using metrics, factors and process standards.
- Course Outcomes:
- Understand the software errors and need of software quality management
- Classify and understand the software quality factors in various stages of development.
- Analyze the different software metrics and their implementation costs.
- Understand the Software Quality Assurance project standards and its management roles.

UNIT - I

Introduction to Software Quality:

Define Software, Software errors, faults and failures, Classification of the causes of software errors, Software quality – definition, Software quality assurance – definition and objectives, Software quality assurance and software engineering. The uniqueness of software quality assurance, the environments for which SQA methods are developed.

UNIT - II

Software quality factors:

The need for comprehensive software quality Requirements, Classifications of software requirements into software quality factors, Product operation software quality factors, Product revision software quality factors, Product transition software quality factors, Alternative models of software quality factors, quality Requirements, Software compliance with quality factors.

UNIT - III

Software quality metrics and their Costs

Objectives of quality measurement, Classification of software quality metrics, Process metrics, Product metrics, Implementation of software quality metrics, Limitations of software metrics. Objectives of cost of software quality metrics, the classic model of cost of software quality, an extended model for cost of software quality, Application of a cost of software quality system, Problems in the application of cost of software, quality metrics.

UNIT - IV

SQA project process standards and Management:

Structure and content of IEEE software engineering standards, IEEE/EIA Std 12207 – software life cycle processes, IEEE Std 1012 – verification and validation, IEEE Std 1028 – reviews. Top management's quality assurance activities, Department management responsibilities for quality assurance, Project management responsibilities for quality assurance

UNIT - V

The SQA unit and its Actors:

The SQA unit, SQA trustees and their tasks, SQA committees and their tasks, SQA forums – tasks and Methods of operation.

TEXT BOOK:

1. Software Quality Assurance, From theory to implementation, DANIEL GALIN, Pearson Education, 2004.

REFERENCE BOOK:

1. Metrics and Models in Software Quality Engineering, Kan Pearson Education, 2003.

SOFTWARE TESTING METHODOLOGIES (PE–III)

M.Tech SE I Year II Sem.

L	T	P	C
3	0	0	3

Pre requisites: A course on “Software Engineering”

Course Objectives:

- To provide knowledge of the concepts in software testing such as testing process, criteria, strategies, and methodologies.
- To develop skills in software test automation and management using latest tools.

Course Outcomes:

- Ability to design and develop the best test strategies in accordance to the development models
- Acquire skills to perform data flow testing, domain testing, logic testing.

UNIT-I

Introduction: Purpose of testing, Dichotomies, model for testing, consequences of bugs, taxonomy of bugs Flow graphs and Path testing: - Basics concepts of path testing, predicates, path predicates and achievable paths, path sensitizing, path instrumentation, application of path testing.

UNIT-II:

Transaction Flow Testing: Transaction flows, transaction flow testing techniques.
Data flow testing: Basics of data flow testing, strategies in data flow testing, application of data flow testing.
Domain Testing: Domains and paths, nice & ugly domains, domain testing, domains and interfaces’ testing, domain and interface testing, domains and testability.

UNIT-III:

Paths, Path products and Regular expressions: path products & path expression, reduction procedure, applications, regular expressions& flow anomaly detection.
Logic Based Testing: overview, decision tables, path expressions, kv charts, specifications.

UNIT-IV:

State, State Graphs and Transition testing: state graphs, good & bad state graphs, state testing, Test ability tips.

UNIT-V:

Graph Matrices and Application: Motivational overview, matrix of graph, relations, power of a matrix, node reduction algorithm, building tools. (Student should be given an exposure to a tool like J Meteror Win-runner).

Text Books:

1. Software Testing techniques- Boris Beizer, Dream tech, second edition.
2. Software Testing Tools-Dr.K.V.K.K. Prasad, Dream tech.

MOBILE APPLICATION AND API DEVELOPMENT (Professional Elective – III)

M.Tech SE I Year II Sem.

L	T	P	C
3	0	0	3

Course Objectives: Your studies will enable you to develop

- Structure your app, design flexible and interactive interfaces, run services in the back ground, make your app work on various smart phones and tablets
- Build efficient and secure REST ful web APIs in Java
- Design solutions to produce, consume and visualize Restful web services using WADL,RAML, and Swagger
- Familiarize the role of Restful APIs usage in emerging technology trends like Cloud, IoT, and Social Media.
- Introduce yourself to the Restful soft ware architectural style and the RESTAPI design principles
- Make use of the JSR353API,JSR374API,JSR367 API and Jackson API for JSON processing
- Build portable Restful web APIs,makinguseoftheJAX-RS2.1API
- Simplify API development using the Jersey and REST Easy extension APIs
- Secure your Restful web services with various authentication and authorization mechanisms
- Get to grips with the various metadata solutions to describe, produce, and consume Restful web services
- Understandthedesignandcodingguidelinestobuildwell-performingRESTfulAPIs
- See how the role of Restful web services changes with emerging technologies and trends

Course Outcomes:

- Explain the key concepts of Restful API architecture.
- Use JAX-RS API annotations to create Restful web services.
- Apply security measures like O Auth to protect APIs.
- Set up the Android development environment and create basic applications.
- Build functional mobile apps that consume APIs and deploy them successfully.

UNIT-I

Introduction to Restful API Architecture: Introduction to REST Architecture style, architectural elements, methods. Java API for JSON processing

UNIT-II

JAX-RS API: JAX-RSAPI Annotations, Building API, advance features, filters and interceptors, Jersey and HATEOAS models, REST frame works and extensions.

UNIT-III

Design and Securing RESTAPI: Securing API using OAuth, Restful modeling language, Swagger, RESTAPI design guidelines, package and deploy JAX-RS applications.

UNIT-IV

Android Development Basics: Understanding Android Platform, Setting up the environment, pre-requisites for building application

UNIT-V

Build Mobile Apps using API: Setting up the Android environment, building basic app, activities and layout, run the app, watch the progress and go-live.

Text Books:

1. Best for Visual Learners: Head First Android Development: A Brain-Friendly Guide by Dawn Griffitha and David Griffitha
2. Restful Java Web Services: A pragmatic guide to designing and building Restful APIs using Java,3rd Edition by Bogunuva Mohanram Balachandar

Reference Books:
Best for Programmers with Java Experience: Android Programming: The Big Nerd Ranch Guide

1. Best for Quick Answers: Android Cook book: Problems and Solutions for Android Developers
2. Building Progressive Web Apps- Bringing the Power of Native to the Browser

FULL STACK DEVELOPMENT (Professional Elective–III)

M.Tech SE I Year II Sem.

L	T	P	C
3	0	0	3

Course Objectives:

- To learn the core concepts of both the frontend and backend programming course.2.To get familiar with the latest web development technologies.
- To learn all about databases.
- To learn complete web development process
- To provide an in-depth study of the various web development tools

Course Outcomes:

- Develop a fully functioning website and deploy on a web server.
- Gain Knowledge about the front end and backend Tools
- Find and use code packages based on their documentation to produce working results in a project.
- Create web pages that function using external data.

UNIT-I:

Web Development Basics: Web development Basics-HTM L& Web servers Shell-UNIXCLI Version control- Git & Git hub HTML, CSS

UNIT-II:

Frontend Development: Java script basics OOPS Aspects of Java Script Memory usage and Functions in JS AJAX for data exchange with server JQuery Framework JQuery events, UI components etc. JSON data format.

UNIT-III:

REACT JS: Introduction to React React Router and Single Page Applications React Forms, FlowArchitectureandIntroductiontoReduxMoreReduxandClient-ServerCommunication

UNIT-IV:

Java Web Development: JAVA PROGRAMMING BASICS, Model View Controller (MVC)Pattern MVC Architecture using Spring Restful API using Spring Framework Building an application using Maven

UNIT-V:

Databases & Deployment: Relational schemas and normalization Structured Query Language (SQL) Data persistence using Spring JDBC Agile development principles and deploying application in Cloud

TEXTBOOKS:

1. Web Design with HTML, CSS, Java Script and JQuery Set Book by Jon Duckett Professional Java Script for Web Developers Book by Nicholas C.Zakas
2. LearningPHP,MySQL,JavaScript,CSS&HTML5:AStep-by-StepGuideto Creating Dynamic Websites by Robin Nixon
3. Full Stack JavaScript: Learn Backbone.js, Node.js and Mongo DB. Copyright © 2015 BY Azat Mardan

REFERENCES:

1. Full-Stack Java Script Development by Eric Bush.
2. Mastering Full Stack React Web Development Paperback-April28, 2017 by Tomasz Dyl, Kamil Przeorski, Maciej C zarnecki

FUNCTIONAL PROGRAMMING (Professional Elective-III)

M.Tech SE I Year II Sem.

L	T	P	C
3	0	0	3

Course Objectives:

- Understand the concepts and terms used to describe languages that support the imperative, functional, object - oriented, and logic programming paradigms.
- Solve problems using the functional paradigm.
- Solve problems using the object-oriented paradigm.
- Solve problems using the logic programming paradigm.

Course Outcomes: At the end of the course the student will be able to:

- Write programs in a functional style.
- Reason formally about functional programs.
- Use polymorphism and higher-order functions.
- Reason about the time and space complexity of programs.

Program Specific Outcomes:

- Develop a strong foundation in functional programming principles applicable to various programming languages.
- Gain proficiency in using Python for both functional programming and general-purpose tasks.
- Acquire the ability to analyze and solve problems using functional programming techniques.
- Understand the significance of modularity and maintainability in software development.
- Prepare for advanced programming roles by mastering multiple functional programming languages and concepts.

UNIT-I

Functional Programming: Introduction, Differences between Functional Programming and Object-Oriented Programming, concepts of functional programming, Functional Programming in Python: Introduction to Python, Built-in Functions, Dictionary Methods, String Methods, LIST/ ARRAY Methods, Tuple Methods, Set Methods

UNIT-II

Python Exceptions, File Handling, Tuple Methods, Defining Iteration, Conditional Iterations, Random Module, Math Module, C Math Module, Python File I/O

UNIT-III

Python Sending Mail, Python CSV, Python OOP Concepts, Python Iterators, Python Generators, Python Decorators, Python Data base Connections

UNIT-IV

Introduction to Haskell and Laziness, Structure, Modularity, Maintainability, Polymorphism, higher order functions, strings & characters, lazy evaluation, Data Types using Patterns

UNIT-V

LISP Programming: Basic LISP Programming, Data Types, Functions, Editing, Loading, Compiling LISP Programs, Control Structures: Recursions and Conditionals, LISTS, SETS, Structural Recursion with LISTS, Symbols

TEXTBOOKS:

1. The Haskell School of Expression: Learning Functional Programming through Multimedia, Paul Hudak.
2. Functional Programming in Python, David Mertz, O'Reilly Media.
3. LISP, Patrick Henry Winston, Bertbold Klaus Paul Horn, Pearson Education

SECURE SOFTWARE ENGINEERING (PE - IV)

M.Tech SE I Year II Sem.

L	T	P	C
3	0	0	3

Course Objectives:

- Students will demonstrate knowledge of the distinction between critical and non-critical systems.
- Students will demonstrate the ability to manage a project including planning, scheduling and risk assessment / management.
- Students will author a software requirements document.
- Students will demonstrate an understanding of the proper contents of a soft ware requirements document.
- Students will author a formal specification for a software system.
- Students will demonstrate an understanding of distributed system architectures and application architectures.
- Students will demonstrate an understanding of the differences between real-time and non-real time systems.
- Students will demonstrate proficiency in rapid software development techniques.
- Students will be able to identify specific components of a soft ware design that can be targeted for reuse.
- Students will demonstrate proficiency in software development cost estimation.
- Students will author a soft ware testing plan.

Course Outcomes:

- Explain the key concepts of software security and its significance in the software development process.
- Identify common threats to software security and evaluate their potential impact on applications.
- Apply secure requirements engineering techniques, including the SQUARE process model, to prioritize security needs.
- Design secure software architectures using established security principles and guidelines.
- Implement secure coding practices and perform security testing to detect vulnerabilities in software systems.

UNIT–I

Security a software Issue: Introduction, the problem, Software Assurance and Soft ware Security,Threats to software security, Sources of software insecurity, Benefits of Detecting Software Security**WhatMakesSoftwareSecure:**PropertiesofSecureSoftware,Influencingthesecuritypropertiesof software,Assertingandspecifyingthedesiredsecurityproperties?

UNIT–II

Requirements Engineering for secure software: Introduction, the SQUARE process Model, Requirements elicitation and prioritization

UNIT–III

Secure Software Architecture and Design: Introduction, software security practices for architecture and design: architectural risk analysis, software security knowledge for architecture and design: security principles, security guidelines and attack patterns

Secure coding and Testing: Code analysis, Software Security testing, Security testing considerations throughput the SDLC

UNIT–IV

Security and Complexity: System Assembly Challenges: introduction, security failures, functional and attacker perspectives for security analysis, system complexity drivers and security

UNIT–V

Governance and Managing for More Secure Software: Governance and security, Adopting an enterprise software security frame work, How much securityis enough?, Security and project management, Maturity of Practice

Textbook:

1. Software Security Engineering: Julia H .Allen, Pearson Education

References:

1. Developing Secure Software: Jason Grembi, Cengage Learning
2. Software Security: Richard S inn, Cengage Learning

OBJECT ORIENTED SOFTWARE ENGINEERING (Professional Elective- IV)

M.Tech SE I Year II Sem.

L	T	P	C
3	0	0	3

Course Objectives:

- Understand the various software life cycle models and their applicability in different software development scenarios.
- Learn the principles of planning and estimation in software projects, including different metrics and models.
- Explore the workflows involved in the software life cycle, focusing on requirements gathering, analysis, design, implementation, and testing.
- Gain knowledge of software quality assurance practices and standards to ensure the delivery of high-quality software products.
- Understand software configuration management and maintenance processes to manage changes and defects effectively.

Course Outcomes:

- Describe different software life cycle models and select appropriate models for specific project requirements.
- Apply estimation techniques, such as LOC and Function Points, to assess project size and complexity.
- Conduct requirement elicitation and documentation using techniques like use case specifications and activity diagrams.
- Design software systems using object-oriented principles, including design patterns, coupling, and cohesion.
- Implement testing strategies and quality assurance practices to ensure software reliability and maintainability.

UNIT–I

Software life cycle models: Waterfall, RAD, Spiral, Open-source, Agile process
Understanding software process: Process metric, CMM levels

UNIT–II

Planning & Estimation: Product metrics, Estimation- LOC, FP, COCOMO models.
Project Management: Planning, Scheduling, Racking.

UNIT-III

Work flow of Software life cycle,
Requirement Work flow: Functional and Nonfunctional, Characteristics of Requirements, Requirement Elicitation Techniques, Requirement Documentation – Use case specification, Activity Diagram
Analysis workflow: Static Analysis, Identifying Object – Methods of identifying objects and types - Boundary, Control, Entity, **Dynamic Analysis-Identifying Interaction –** Sequence and Collaboration diagrams, State chart diagram
Design Workflow: System Design Concept – Coupling and Cohesion, Architectural Styles Identifying Subsystems and Interfaces, Design Patterns

UNIT-IV

Implementation Workflow: Mapping models to Code, Mapping Object Model to Database Schema
Testing: FTR – Walkthrough and Inspection, Unit Testing, Integration, System and Regression Testing, User Acceptance Testing
Software Quality: Quality Standards, Quality Matrices Testing & SQA: FTR, unit testing, integration testing, product testing, and acceptance testing

UNIT-V

Software Configuration Management: Managing and controlling Changes, Managing and controlling versions
Maintenance: Types of maintenance, Maintenance Log and defect reports. Reverse and re-engineering
References:

1. Bernd Bruegge, “Object oriented software engineering” ,Second Edition, Pearson Education.
2. Stephan R. Schach, “Object oriented software engineering”, Tata Mc Graw Hill.
3. Roger Pressman, “Software Engineering”, sixth edition, Tata Mc Graw Hill.

HUMAN COMPUTER INTERACTION (Professional Elective- IV)

M.Tech SE I Year II Sem.

L	T	P	C
3	0	0	3

Course Objectives:

- Understand the importance of user interfaces and the principles of good design in enhancing user experience.
- Explore the history and evolution of user interface design, focusing on graphical user interfaces (GUIs) and web interfaces.
- Learn the design process for creating effective user interfaces, considering human characteristics and interaction speeds.
- Gain skills in screen design, including organization of screen elements, navigation, and visual composition.
- Familiarize with various interaction devices and software tools for interface design and development.

Course Outcomes:

- Describe the significance of user interface design and its impact on user satisfaction and usability.
- Analyze the characteristics of effective graphical and web user interfaces, applying design principles to real-world scenarios.
- Create well-organized screen layouts that prioritize user needs and enhance information retrieval.
- Select appropriate navigation schemes and controls for different types of user interfaces.
- Demonstrate proficiency in using software tools for interface specification and development, as well as understanding various interaction devices.

UNIT I

Introduction: Importance of user Interface – definition, importance of good design. Benefits of good design. A brief history of Screen design,
The graphical user interface – popularity of graphics, the concept of direct manipulation, graphical system, Characteristics, Web user- Interface popularity, characteristics - Principles of user interface.

UNIT II

Design process-Human interaction with computers, importance of human characteristics human consideration, Human interaction speeds, and understanding business junctions.
Screen Designing:- Design goals – Screen planning and purpose, organizing screen elements, ordering of screen data and content - screen navigation and flow - Visually pleasing composition - amount of information - focus and emphasis - presentation information simply and meaningfully - information retrieval on web-statistical graphics-Technological consideration in interface design.

UNIT III

Windows - New and Navigation schemes selection of window, selection of devices based and screen based controls.
Components – text and messages, Icons and increases – Multimedia, colors, uses problems, choosing colors.

UNIT IV

Software tools- Specification methods, interface- Building Tools.

UNIT V

Interaction Devices - Keyboard and function keys - pointing devices - speech recognition digitization and generation-image and video displays - drivers.

TEXTBOOKS:

1. The essential guide to user interface design, Wilbert O Galitz, Wiley Dream Tech.
2. Designing the user interface, 3rd Edition Ben Shneider mann, Pearson Education Asia.

REFERENCES:

1. Human – Computer Interaction. Alan Dix, Janet Fincay, Gre Goryd, A bowd, Russell Bealg, Pearson Education
2. Interaction Design Prece, Rogers, Sharps. Wiley Dream tech.
3. User Interface Design, Soren Lauesen, Pearson Education.
4. Human-Computer Interaction, D.R. Olsen, Cengage Learning.
5. Human-Computer Interaction, Smith - Atakan, Cengage Learning.

MACHINE LEARNING (PE-III)

M.Tech SE I Year II Sem.

L	T	P	C
3	0	0	3

Course Objectives:
1. To introduce students to the basic concepts and techniques of Machine Learning.
2. To have a thorough understanding of the Supervised and Unsupervised learning techniques
3. To study the various probability-based learning techniques

Course Outcomes:
1. Distinguish between, supervised, unsupervised and semi-supervised learning.
2. Understand algorithms for building classifiers applied on datasets of non-linearly separable classes
3. Design an ensemble model to increase the classification accuracy
4. Understand the principles of RL evolutionary computing algorithms

UNIT - I

Introduction to Machine Learning: Types of Human learning, machine learning process, Well-posed learning problem, Types of machine learning and comparison, applications of machine learning. Model Preparation, Evaluation and feature engineering: Machine learning activities, Types of data in machine learning, dataset understanding, plotting and exploration, checking data quality, remediation, data pre-processing, selecting a model, predictive and descriptive models, supervised learning model training, cross-validation and boot strapping, lazy vs eager learner, interpreting the model- under fitting, over fitting, bias-variance trade-off. Parameter for evaluating performance of classification, regression, and clustering model. Improving performance of a model.

UNIT - II

Feature Engineering: Feature transformation - feature construction, feature extraction by PCA, SVD, LDA. Feature subset selection – feature relevancy and redundancy measures. Feature selection process and approaches. **Review of Probability concepts:** joint probability, conditional probability, bayes rule, Common discrete and continuous distributions, dealing with multiple random variables, central limit theorem. Bayes classifier, Multi-class Classification, Naïve Bayes classifier, Bayesian belief network.

UNIT - III

Supervised Learning - Introduction to supervised learning,
Regression: Introduction of regression, Regression algorithms: Simple linear regression, Multiple linear Regression, Polynomial regression model, Logistic regression, Maximum likelihood estimation.
Classification: Classification model and learning steps, Classification algorithms: Naïve Bayes classifier, Distance measures, k-Nearest Neighbor (kNN), Decision tree, Support vector machines, Kernel trick, Random Forest.

UNIT - IV

Unsupervised Learning: Introduction to unsupervised learning, unsupervised vs supervised learning, Application of unsupervised learning, Clustering and its types, Partitioning method: k-Means and KMedoids, Hierarchical clustering, Density-based methods – DBSCAN.

UNIT - V

Artificial Neural Network: Biological neuron, Artificial neuron, Activation functions, neural network architecture, perceptron, learning process in ANN, Back propagation. Introduction to deep learning, overview of reinforcement learning, Representation learning, Evolutionary learning. Case-study of ML applications: Image recognition, Email spam filtering, Online fraud detection.

TEXT BOOKS:
1. Saikat Dutt, S. Chjandramouli, Das - Machine Learning, Frist Edition, Pearson
2. M N Murty, Anathanarayana V S - Machine Learning, First Edition, University Press
3. Tom M Mitchell, ¯Machine Learning, First Edition, McGraw Hill Education, 2013.

REFERENCE BOOKS:
1. Stephen Marsland, ¯Machine Learning - An Algorithmic Perspective, Second Edition,
2. Chapman and Hall/CRC Machine Learning and Pattern Recognition Series, 2014.
3. Ethem Alpaydin, ¯Introduction to Machine Learning 3e
(Adaptive Computation and Machine Learning Series), Third Edition, MIT Press, 2016
4. Machine Learning, Anuradha Srinivasa raghavan, Vincy Joseph,Wiley

DESIGN PATTERNS LAB (Lab- III)

M.Tech SE I Year II Sem.

L	T	P	C
0	0	4	2

Course Objectives:

- The objective of this lab is to get an over view of the various machine learning techniques and can able to demonstrate them using python.

Course Outcomes: After the completion of the "**Machine Learning**" lab, the student can able to:

- Understand complexity of Machine Learning algorithms and their limitations;
- Understand modern notions in data analysis- oriented computing;
- Be capable of confidently applying common Machine Learning algorithms in practice and implementing their own;
- Be capable of performing experiments in Machine Learning using real- world data.

List of Experiments

1. The probability that it is Friday and that a student is absent is 3 %. Since there are 5 schooldays in a week, the probability that it is Friday is 20%. What is the probability that a student is absent given that today is Friday? Apply Baye's rule in python to get the result. (Ans: 15%)
2. Extract the data from data base using python
3. Implement k-nearest neighbors classification using python
4. Given the following data, which specify classifications for nine combinations of VAR1 andVAR2 predict a classification for a case where VAR1=0.906 and VAR2=0.606, using the result of k-means clustering with3means(i.e.,3centroids)

VAR1	VAR2CLASS	
1.713	1.586	0
0.180	1.786	1
0.353	1.240	1
0.940	1.566	0
1.486	0.759	1
1.266	1.106	0
1.540	0.419	1
0.459	1.799	1
0.773	0.186	1

5. The following training examples map descriptions of individuals on to high, medium and low credit-worthiness.
   Medium skiing design single twenties no-> high Risk
   High golf trading married forties yes->low Risk
   Low speed way transport married thirties yes->med Risk
   Medium foot ball banking single thirties yes -> low Risk
   High flying media married fifties yes-> high Risk
   Low football security single twenties no -> med Risk
   Medium golf media single thirties yes -> med Risk
   Medium golf transport married forties yes -> low Risk
   High skiing banking single thirties yes->high Risk
   Low golf unemployed married forties yes->high Risk
   Input attributes are (from left to right) income, recreation, job, status, age-group, home-owner. Find the unconditional probability of `golf' and the conditional probability of `single' given `med Risk' in the data set?
6. Implement linear regression using python.
7. Implement Naïve Bayes theorem to classify the English text
8. Implement an algorithm to demonstrate the significance of genetic algorithm
9. Implement the finite words classification system using Back-propagation algorithm

Learning Objectives: After completing this course, the student should be able to

- Understand the concept of patterns and the Catalog.
- Discuss the Presentation tier design patterns and their affection: sessions, client access,

validation and consistency.
- Understand the variety of implemented bad practices related to the Business and Integration tiers.
- Highlight the evolution of patterns.
- How to add functionality to designs while minimizing complexity
- What design patterns really are, and are not
- About specific design patterns.
- What code qualities you need to maintain to keep code flexible.
- How to use design patterns to keep code quality high without over design.

Course Outcomes:
- Identify and explain key design patterns such as abstract factory, adapter, bridge, command, composite, facade, observer, proxy, and strategy.
- Successfully design and implement applications that apply specified design patterns to solve programming challenges.
- Evaluate and select the appropriate design pattern for various software development scenarios, enhancing design decisions.
- Demonstrate proficiency in using programming languages to implement design patterns, ensuring code reusability and maintainability.
- Reflect on and articulate the advantages and trade-offs of using design patterns in software design through written reports or presentations.

List of Sample Problems:

1. Design and execute an application, where “abstract factory” pattern is applied.
2. Design and execute an application, where “adapter” pattern is applied.
3. Design and execute an application, where “bridge” pattern is applied.
4. Design and execute an application, where “command” pattern is applied.
5. Design and execute an application, where “composite” pattern is applied.
6. Design and execute an application, where “facade” pattern is applied.
7. Design and execute an application, where “observer” pattern is applied.
8. Design and execute an application, where “proxy” pattern is applied.
9. Design and execute an application, where “strategy” pattern is applied.
10. Specify and implement the most convenient pattern to apply to the following scenarios:
    i. A component on the web tier requires access to business components.
    ii. There a need to provide several buttons on a web form which executes different actions.
    iii. Messages need to be sent to citizen search time a typhoon approaches.

Text Books:
1. Machine Learning- Tom M. Mitche ll, MGH
2. Fundamentals of Speech Recognition By Lawrence Rabiner and Biing- Hwang Juang.

Reference Book:
1. Machine Learning: An Algorithmic Perspective, Stephen Mars l and, Taylor & Francis

SOFTWARE TESTING METHODOLOGIES LAB (Lab– IV) (PE-3 LAB)

M.Tech SE I Year II Sem.

L	T	P	C
0	0	4	2

Pre requisites: A basic knowledge of programming.

Course Objectives:

- To provide knowledge of Software Testing Methods.
- To develop skills in software test automation and management using latest tools.

Course Outcomes:

- Design and develop the best test strategies in accordance to the development model.

List of Experiments

1. Recording in context sensitive mode and analog mode.
2. GUI checkpoint for single property.
3. GUI checkpoint for single object / window.
4. GUI checkpoint for multiple objects.
5. a)Bitmap checkpoint for object / window.
   b) Bitmap checkpoint for screen area.
6. Database check point for Default check
7. Database check point for custom check
8. Database check point for runtime record check
9. a)Data driven test for dynamic test data submission
   b) Data driven test through flat files
   c) Data driven test through front grids
   d) Data driven test through excel test
10. a) Batch testing without parameter passing
    b) Batch testing with parameter passing
11. Data driven batch
12. Silent mode test execution without any interruption
13. Test case for calculator in windows application

Text Books:

1. Software Testing techniques, Baris Beizer, 2nd edition, Dream tech.
2. Software Testing Tools, Dr. K.V.K.K. Prasad, Dream tech.

References:

1. The craft of software testing, Brian Marick, Pearson Education.
2. Software Testing Techniques-SPD(Oreille)
3. Software Testing in the Real World, Edward Kit, Pearson.
4. Effective methods of Software Testing, Perry, John Wiley.
5. Art of Software Testing, Meyers, John Wiley.

MOBILE APPLICATION AND API DEVELOPMENT LAB (Lab – IV) (PE-3 LAB)

M.Tech SE I Year II Sem.

L	T	P	C
0	0	4	2

Course Objectives:

- To learn how to develop Applications in android environment.
- To learn how to develop user interface applications.
- To learn how to develop URL related applications.
- The student is expected to be able to do the following problems, though not limited.

Course Outcomes:

- Students will create basic applications that demonstrate knowledge of Android development fundamentals, including UI elements and event handling.
- Students will design and implement complex user interfaces that effectively utilize layouts and fragments, responding to user interactions.
- Students will develop applications that utilize intents for various actions (calling, sending SMS) and create notifications for user alerts.
- Students will implement user registration and login functionalities using databases, ensuring secure data management practices.
- Students will develop applications that track and display the user's current location using GPS and integrate with Google Maps.

List of Experiments:

1.a) Create an Android application that shows Hello + name of the user and run it on an emulator.
b) Create an application that takes the name from a text box and shows hello message along with the name entered in text box, when the user clicks the OK button.

2. Create a screen that has input boxes for User Name, Password, Address, Gender (radio buttons for male and female), Age (numeric), Date of Birth (Date Picket), State (Spinner) and a Submit button. On clicking the submit button, print all the data below the Submit Button. Use (a) Linear Layout (b) Relative Layout and (c) Grid Layout or Table Layout.

3. Develop an application that shows names as a list and on selecting a name it should show the details of the candidate on the next screen with a “Back” button. If the screen is rotated to landscape mode (width greater than height), then the screen should show list on left fragment and details on right fragment instead of second screen with back button. Use Fragment transactions and Rotation event listener.

4. Develop an application that uses a menu with 3 options for dialing a number, opening a website and to send an SMS. On selecting an option, the appropriate action should be invoked using intents.

5. DevelopanapplicationthatinsertssomenotificationsintoNotificationareaand whenever a notification is inserted, it should show a toast with details of the notification.

6. Create an application that uses a text file to store user names and passwords (tab separated fields and one record per line).When the user submits a login name and password through a screen, the details should be verified with the text file data and if they match, show a dialog saying that login is successful. Otherwise, show the dialog with Login Failed message.

7. Create a user registration application that stores the user details in a database table.

8. Create a database and a user table where the details of login names and pass words are stored. Insert some names and passwords initially. Now the login details entered by the user should be verified with the database and an appropriate dialog should be shown to the user.

9. Create an admin application for the user table, which shows all records as a list and the admin can select any record for edit or modify. The results should be reflected in the table.

10. Develop an application that shows all contacts of the phone along with details like name, phone number, mobile number etc.

11. Create an application that saves user information like name, age, gender etc. in shared preference and

retrieves them when the program restarts.

12. Create an alarm that rings every Sunday at 8:00AM. Modify it to use a time picker to set alarm time.

13. Create an application that shows the given URL (from a text field) in a browser.

14. Develop an application that shows the current location's latitude and longitude continuously as the device is moving (tracking).

15. Create an application that shows the current location on Google maps. Note:
Android Application Development with MIT App Inventor: For the first one week, the student is advised to go through the App Inventor from MIT which gives insight into the various properties of each component. The student should pay attention to the properties of each components, which are used later in Android programming. Following a reuse ful links:

1. http://ai2.appinventor.mit.edu
2. https://drive.google.com/file/d/0B8rTtW_91YclTWF4czdBMEpZcWs/view

FUNCTIONAL PROGRAMMING LAB (Professional Elective- III Lab)

M.Tech SE I Year II Sem.

L	T	P	C
0	0	4	2

Course Objectives:

- To be able to introduce core programming basics and program design with functions using functional programming languages.
- To understand a range of Object-Oriented Programming, as well as in-depth data and information processing techniques.

Course Outcomes:

- Students should be able to understand the basic concepts of scripting and the contributions of Functional Programming Languages.
- Ability to explore python specially the object – oriented concepts and the built in objects of Python.

List of Programs:

1. Write a program to demonstrate different number data types in Python.
2. Write a program to perform different Arithmetic Operations on numbers in Python.
3. Write a program to create, concatenate and print a string and accessing sub-string from a given string.
4. Write a python script to print the current date in the following format “Sun May 2902:26:23 IST 2017”
5. Write a program to create, app end, and remove lists in python.
6. Write a program to demonstrate working with tuples in python.
7. Write a program to demonstrate working with dictionaries in python.
8. Write a python program to find largest of three numbers.
9. Write a Python program to convert temperatures to and from Celsius, Fahrenheit. [Formula :c/5=f-32/9]
10. Write a recursive function, (de fun nth (pos list)???), that returns the n'th item from a list. Assume the list has at least n items. (nth1aList) Is to return the first item in a List.
11. Write simple lisp functions such as the following. Take into account lists which are too short.

- (Remove- first'(abc...))->(bc...) ---remove the first item from the list.
- (Remove- second'(abc...))-> (ac...)–remove the second item from the list.
- (Insert- as – second 'b'(ac...))-> (abc...) ---insert as the second element.

12. Write a Lisp macro my case that translates the following macro call. Assume the input will be error free. The input lists can be any length. You must document your solution.

    (my case(C1C2...Cn)(P1P2...Pn))
    Translates to the following
    (Mycond (C1P1)(C2 P2)...(Cn Pn))

13. Write a Lisp macro mycase that translates the following macro callas shown. Assume the input will be error free. The input lists can be any length. Use standard Lisp functional. If you need support functions, your answer should have only non-recursive support functions.

    (Mycase (C1C2...Cn)(P1P2...Pn))
    Translates to the following
    (Mycond (C1 (P1P2...Pn))(C2(P2...Pn))...(Cn(Pn)))

14. Assume the following forms have been typed into the Lisp interpreter and evaluated.

    (De fun a(x) (values (list x) x))(Set qa'(ab))
    (De fun b(x) `(x, x))(Set qb(cdra))

    (set qc(cara))(set qdc)
    (set qe((lambda(x)(list x))d))

15. What will the following forms evaluate to?(consc (cara))
    (conseb)(evala)
    (let ((a b) (y a)) (append a y))(multiple-value-list(aa))

(bc)
(set(cara)(cdra))
(setf(cara)(cdra))

TEXTBOOKS:

1. The Haskell School of Expression: Learning Functional Programming through Multimedia, Paul Hudak.
2. Functional Programming in Python, DavidMertz,O'ReillyMedia.

AGILE DEVELOPMENT METHODOLOGIES (Professional Elective-V)

M.Tech SE II Year I Sem.

L	T	P	C
3	0	0	3

Course Objectives:

- Understand the principles and practices of Agile Development methodologies, focusing on Extreme Programming (XP).
- Explore the importance of collaboration and customer involvement in successful Agile projects.
- Learn the strategies for releasing software with high quality and fast delivery.
- Develop skills in planning iterations, managing risks, and estimating project timelines effectively.
- Gain practical experience in incremental development techniques, including test-driven development and refactoring.

Course Outcomes:

- Demonstrate knowledge of Agile methodologies and their application in real-world software development.
- Collaborate effectively in teams, utilizing Agile practices to enhance productivity and communication.
- Produce high-quality software releases through effective version control and continuous integration.
- Create and manage effective release plans and iterations while identifying and mitigating risks.
- Apply incremental development practices, including customer testing and exploratory testing, to enhance software quality.

UNIT-I

Introduction Extreme Programming (XP)- Agile Development

Why Agile-Understanding Success, Beyond Deadlines, Importance of Organizational Success, Introduction to Agility

How to Be Agile - Agile methods, Don't make your own method, Road to mastery Understanding XP (Extreme Programming) - XP life cycle, XP team, XP Concepts Adopting XP- Knowing whether XP issuitable, Implementing XP,assessingAgility

Practicing XP- Thinking – Pair Programming, Energized work, Informative Work space, Root cause Analysis, Retrospectives

UNIT-II

Collaborating

Trust, Sit together, Real customer involvement, Ubiquitous language, meetings, coding standards, Iteration demo, Reporting

UNIT-III

Releasing

Bug free Release, Version Control, fast build, continuous integration, Collective ownership, Documentation

UNIT-IV

Planning

Version, Release Plan, Risk Management, Iteration Planning, Slack, Stories, Estimating

UNIT-V

Developing:

Increment al requirements, Customer tests, Test driven development, Refactoring,

Increment al design and architecture, spike solutions, Performance optimization, Exploratory testing

Text Book:

1. The art of Agile Development, James Shore and Shane Warden, 11th Indian Reprint, O'Reilly,2018

References:

1. Learning Agile, Andrew Stellman and Jennifer Greene,O'Reilly,4th Indian Reprint,2018
2. Practices of an Agile Developer, VenkatSubramaniam and Andy Hunt, SPD, 5th IndianReprint,2015
3. Agile Project Management – Jim High smith, Pearson Low price Edition 2004.

WEB SERVICES TESTING (Professional Elective- V)

M.Tech SE II Year I Sem.

L	T	P	C
3	0	0	3

Course Objectives: Your studies will enable you to develop:

- Become more proficient in testing web services included in your service – oriented solutions
- Find, analyze, reproduce bugs effectively by adhering to best web service testing approaches
- Learn with clear step – by – step instructions and hands-one examples on various topics related to web services testing using soap UI

Course Outcomes:

- Demonstrate proficiency in creating and managing soapUI projects for web services testing.
- Design and execute functional and non-functional test cases for web services, including load and performance tests.
- Implement advanced testing techniques for security and RESTful services using soapUI.
- Develop automated testing scripts using Groovy and integrate them with other testing frameworks.
- Validate web services for compliance with industry standards using WS-I validation and other best practices.

UNIT-I

Introduction to Web Services Testing and soapUI: SOA and web services, SOAP, SOAPUI basics,projectpre-requisites.Designingandimplementingwebservices.

UNIT-II

First hands-on experience: Understanding web services definition, creation of soapUI projects, generate SOAP faults. Creating test suites, test cases, assertions, properties in test cases and suites.

UNIT-III

Load and Performance Testing: Non-functional testing, planning for web services performance testing using soapUI, load assertions. Web services simulation with soapUI, mock services in action.

UNIT-IV

Advance testing scenarios: Advanced functional testing, WS-Addressing, WS-Security, ApacheAxis2, security with binding and authentication. REST testing, REST parameters, REST functional testing using soapUI, testing databases using soapUI and assertions, JMS testing with soapUI and assertions.

UNIT-V

Automated testing using scripts: Introduction to Groovy script, Groovy scripting in soapUI, modelitems,requestandresponsehandlingusingscripts,testautomation,Junitintegration,soapUIcomman d line execution, maven soapUI plug-in, WS-I validation, sending attachments with SOAP messages using soapUI.

Text Books:

1. Web Services Testing with soap UI by Charitha Kankanamge
2. The Art of Application Performance Testing by Ian Molyneaux

Reference Books:

1. Testing Applications on the Web by Hung Nguyen, Bob Johnson m and Michael Hack

ADHOC NETWORKS (Professional Elective-V)

M.Tech SE II Year I Sem.

L	T	P	C
3	0	0	3

Course Objectives: This course will enable students to

- Explain fundamental principles of Adhoc Networks
- MAC Protocols for Adhoc Wireless Networks
- Routing Protocols for Adhoc Wireless Networks
- Transport Layer Protocols for Ad-hoc Networks
- Security Protocols for Adhoc Networks
- Wireless Sensor Networks

Course Outcomes:

- Understand Mac Protocols for Adhoc networks
- Understand Routing protocols
- Understand Transport layer protocols
- Understand Security related issues
- Understand Wireless sensor networks

UNITI

Ad-hoc Wireless Networks: Introduction, Issues in Ad-hoc Wireless Networks, Ad-hoc Wireless Internet;

MAC Protocols for Ad-hoc Wireless Networks: Introduction, Issues in Designing a MAC Protocol, Design Goals of MAC Protocols, Classification of MAC protocols, Contention-Based Protocols, Contention-Based Protocols with Reservation Mechanisms.

UNITII

Routing Protocols for Ad-hoc Wireless Networks: Introduction, Issues in Designing a Routing Protocol for Ad-hoc Wireless Networks, Classification of Routing Protocols, Table Driven Routing Protocols; On-Demand Routing Protocols, Hybrid Routing Protocols.

UNITIII

Transport Layer Protocols for Ad hoc Networks: Introduction, Issues in Designing a Transport Layer Protocol, Design Goals of a Transport Layer Protocol, Classification of Transport Layer Solutions, TCP over Adhoc Wireless Networks

UNITIV

Security Protocols for Adhoc Networks: Security in Adhoc Wireless Networks, Issues and Challenges in Security Provisioning, Network Security Attacks, Key Management, Secure Routing in Adhoc Wireless Networks.

UNITV

Wireless Sensor Networks: Introduction, Sensor Network Architecture, Data Dissemination, Data Gathering, MAC Protocols for Sensor Networks

TEXTBOOKS:

1. C. Siva Ram Murthy & B.S. Manoj: Adhoc Wireless Networks: Architectures and Protocols, 2nd Edition, Pearson Education, 2011

REFERENCEBOOKS:

1. Ozan K.Tonguz and Gianguigi Ferrari:Ad-hocWirelessNetworks,JohnWiley,2007.
2. Xiuzhen Cheng, Xiao Hung, Ding-Zhu Du: Ad-hoc Wireless Networking, Kluwer Academic Publishers, 2004.
3. C.K. Toh: Ad-hoc Mobile Wireless Networks – Protocols and Systems, Pearson

MOBILE APPLICATION SECURITY (Professional Elective-V)

M.Tech SE II Year I Sem.

L	T	P	C
3	0	0	3

Course Objectives:

- To understand the mobile issues and development strategies
- To understand the WAP and mobile security issues
- To understand the Bluetooth security issues.

Course Outcomes:

- Demonstrate the ability to identify vulnerabilities in mobile applications and propose security solutions.
- Apply secure coding practices and development strategies to enhance mobile application security.
- Analyze and mitigate common security threats associated with WAP, mobile HTML, and SMS.
- Evaluate Bluetooth security protocols and recommend best practices for secure Bluetooth communications.
- Implement enterprise security measures on various mobile operating systems to protect sensitive data.

UNIT-I

Top Mobile Issues and Development Strategies: Top Issues Facing Mobile Devices, Physical Security , Secure Data Storage (on Disk), Strong Authentication with Poor Keyboards , Multiple-User Support with Security, Safe Browsing Environment , Secure Operating Systems, Application Isolation, Information Disclosure, Virus, Worms, Trojans, Spyware, and Malware , Difficult Patching/UpdateProcess,StrictUseandEnforcementofSSL,Phishing,Cross-SiteRequestForgery(CSRF),Location Privacy/Security, Insecure Device Drivers, Multifactor Authentication, Tips for Secure Mobile Application Development.

UNIT-II

WAP and Mobile HTML Security: WAP and Mobile HTML Basics, Authentication on WAP/MobileHTMLSites,Encryption,ApplicationAttacksonMobileHTMLSites,Cross-SiteScripting,SQLInjection, Cross-Site Request Forgery, HTTP Redirects, Phishing, Session Fixation, Non-SSL Login, WAP and Mobile Browser Weaknesses, Lack of HTTP Only Flag Support, Lack of SECURE Flag Support, Handling Browser Cache, WAP Limitations.

UNIT-III

Bluetooth Security: Over view of the Technology, History and Standards, Common Uses, Alternatives , Future , Bluetooth Technical Architecture , Radio Operation and Frequency, Bluetooth Network Topology , Device Identification , Modes of Operation , Bluetooth Stack ,Bluetooth Profiles ,Bluetooth Security Features , Pairing , Traditional Security Services in Bluetooth, Security “Non-Features”, Threats to Bluetooth Devices and Networks, Bluetooth Vulnerabilities, Bluetooth Versions Prior to v 1.2, Bluetooth Versions Prior to v 2.1.

UNIT-IV

SMS Security: Overview of Short Message Service, Overview of Multimedia Messaging Service, Wireless Application Protocol (WAP), Protocol Attacks, Abusing Legitimate Functionality, Attacking Protocol Implementations, Application Attacks, iPhone Safari, Windows Mobile MMS, Motorola RAZRJPG Over flow, Walk through, Sending PDUs, Converting XML to WBXML.

UNIT-V

Enterprise Security on the Mobile OS: Device Security Options, PIN, Remote, 346 Secure Local Storage, Apple iPhone and Keychain, Security Policy Enforcement, Encryption, Full Disk Encryption, E-mail Encryption, File Encryption, Application Sandboxing, Signing, and Permissions, Applications

and boxing, Application Signing, Permissions, Buffer Overflow Protection, Windows Mobile, iPhone, Android, BlackBerry, Security Feature Summary.

Text Book:

1. “Mobile Application Security”, Himanshu Dwivedi, Chris Clark, David Thiel, TATA McGraw-Hill.

References:

1. “Mobile and Wireless Network Security and Privacy”, KamiS.Makki,et al,Springer.
2. “Android Security Attacks Defenses”, Abhishek Dubey, CRCPress

RESEARCH PAPER WRITING IN ENGLISH (Audit Course- I &II)

M.Tech SE I Year I Sem. / II Sem.

L	T	P	C
2	0	0	0

Pre requisite: None

Course Objectives: Students will be able to:

- Understand that how to improve your writing skills and level of read ability
- Learn about what to write in each section
- Understand the skills needed when writing a Title Ensure the good quality of paper at very first-time submission

Course Outcomes:

- Create well-structured paragraphs and sentences.
- Clarify the roles of authorship in research.
- Effectively highlight findings and results.
- Produce clear and concise abstracts and introductions.
- Conduct thorough literature reviews and discussions.

UNIT-I:

Planning and Preparation, Word Order, Breaking up long sentences, Structuring Paragraphs and Sentences, Being Concise and Removing Redundancy, Avoiding Ambiguity and Vagueness

UNIT-II:

Clarifying Who Did What, Highlighting Your Findings, Hedging and Criticizing, Paraphrasing and Plagiarism, Sections of a Paper, Abstracts. Introduction

UNIT-III:

Review of the Literature, Methods, Results, Discussion, Conclusions, the Final Check.

UNIT-IV:

Key skills are needed when writing a Title, key skills are needed when writing an Abstract, key skills are needed when writing an Introduction, skills needed when writing a Review of the Literature,

UNIT-V:

Skills are needed when writing the Methods, skills needed when writing the Results, skills are needed when writing the Discussion, and skills are needed when writing the Conclusions. useful phrases, how to ensure paper is as good as it could possibly be the first-time submission

TEXTBOOKS/REFERENCES:

1. Gold bort R (2006) Writing for Science, Yale University Press(available on Google Books)
2. Day R(2006)How to Write and Publish a Scientific Paper, Cambridge University Press
3. High man N (1998), Hand book of Writing for the Mathematical Sciences, SIA M. Highman's book.
4. Adrian Wall work, English for Writing Research Papers, Springer New York Dordrecht Heidel berg London, 2011

DISASTER MANAGEMENT (Audit Course- I & II)

M.Tech SE I Year I Sem. / II Sem

L	T	P	C
2	0	0	0

Pre requisite: None

Course Objectives: Students will be able to

- Learn to demonstrate a critical understanding of key concepts in disaster risk reduction and humanitarian response.
- Critically evaluate disaster risk reduction and humanitarian response policy and practice from multiple perspectives.
- Develop an understanding of standards of humanitarian response and practical relevance in specific types of disasters and conflict situations.
- Critically understand the strengths and weaknesses of disaster management approaches, Planning and programming in different countries, particularly their home country or the countries they work in.

Course Outcomes:

- Understand the definitions and significance of disasters and hazards.
- Evaluate the economic damage and human loss caused by different types of disasters.
- Identify disaster-prone areas in India and their specific risks.
- Assess disaster preparedness measures and management strategies.
- Analyze techniques for disaster risk assessment and the importance of community participation

.

UNIT-I:

Introduction:

Disaster: Definition, Factors and Significance; Difference between Hazard and Disaster; Natural and Manmade Disasters: Difference, Nature, Types and Magnitude.

Disaster Prone Areas in India:

Study of Seismic Zones; Areas Prone to Floods and Droughts, Landslides and Avalanches; Areas Prone to Cyclonic and Coastal Hazards with Special Reference to Tsunami; Post-Disaster Diseases and Epidemics

UNIT-II:

Repercussions of Disasters and Hazards:

Economic Damage, Loss of Human and Animal Life, Destruction of Ecosystem. Natural Disasters:Earthquakes,Volcanisms,Cyclones,Tsunamis,Floods,DroughtsandFamines,LandslidesandAvalanches, Man-made disaster: Nuclear Reactor Meltdown, Industrial Accidents, Oil Slicks and Spills, Outbreaks of Disease and Epidemics, War and Conflicts.

UNIT-III:

Disaster Preparedness and Management:

Preparedness: Monitoring of Phenomena Triggering A Disasteror Hazard; Evaluation of Risk: Application of Remote Sensing, Data from Meteorological and Other Agencies, Media Reports: Governmental and Community Preparedness.

UNIT-IV:

Risk Assessment Disaster Risk:

Concept and Elements, Disaster Risk Reduction, Global and National Disaster Risk Situation. Techniques of Risk Assessment, Global Co-Operation in Risk Assessment and Warning, People's Participation in Risk Assessment. Strategies for Survival.

UNIT-V:

Disaster Mitigation:

Meaning, Concept and Strategies of Disaster Mitigation, Emerging Trends in Mitigation. Structural Mitigation and Non-Structural Mitigation, Programs of Disaster Mitigation in India.

TEXTBOOKS/REFERENCES:

1. R. Nishith, Singh AK," Disaster Management in India: Perspectives, issues and strategies"' New Royal book Company.
2. Sahni, Pardeep Et.Al.(Eds.),"Disaster Mitigation Experiences and Reflections", Prentice Hall of India, New Delhi.
3. Goel S. L., Disaster Administration and Management Text and Case Studies", Deep & Deep Publication Pvt. Ltd., New Delhi.

SANSKRIT FOR TECHNICAL KNOWLEDGE (Audit Course – I & II)

M.Tech SE I Year I Sem. / II Sem.

L	T	P	C
2	0	0	0

Pre requisite: None

Course Objectives:

- To get a working knowledge in illustrious Sanskrit, the scientific language in theworld.
- Learning of Sanskrit to improve brain functioning.
- Learning of Sanskrit to develop the logic in mathematics, science & other subjects enhancing the memory power.
- The engineering scholars equipped with Sanskrit will be able to explore the huge knowledge from ancient literature.

Course Outcomes: Students will be able to

- Understanding basic Sanskrit language
- Ancient Sanskrit literature about science & technology can be understood
- Being a logical language will help to develop logic in students

UNIT-I:
Alphabets in Sanskrit,

UNIT-II:
Past/Present/Future Tense, Simple Sentences

UNIT-III:
Order, Introduction of roots,

UNIT-IV:
Technical information about Sanskrit Literature

UNIT-V:
Technical concepts of Engineering- Electrical, Mechanical, Architecture, Mathematics

TEXTBOOKS/REFERENCES:

1. "Abhyas pustakam"-Dr. Vishwas, Samskrita-BhartiPublication,NewDelhi
2. "Teach Yourself Sanskrit" Prathama Deeksha- Vempati Kutumb shastri, Rashtriya Sanskrit Sansthanam, New Delhi Publication
3. "India's Glorious Scientific Tradition" Suresh Soni, Ocean books (P) Ltd., New Delhi.

VALUE EDUCATION (Audit Course- I & II)

M.Tech SE I Year I Sem./ II Sem.

L	T	P	C
2	0	0	0

Pre requisite: None

Course Objectives: Students will be able to
- Understand value of education and self- development
- Imbibe good values in students
- Let the should know about the importance of character

Course Outcomes: Students will be able to
- Knowledge of self-development
- Learn the importance of Human values
- Developing the overall personality

UNIT-I:
Values and self-development – Social values and individual attitudes. Work ethics, Indian vision of humanism. Moral and non-moral valuation. Standards and principles. Value judgments

UNIT-II:
Importance of cultivation of values. Sense of duty. Devotion, Self-reliance. Confidence, Concentration. Truthfulness, Cleanliness. Honesty, Humanity. Power of faith, National Unity. Patriotism. Love for nature, Discipline

UNIT-III:
Personality and Behavior Development - Soul and Scientific attitude. Positive Thinking. Integrity and discipline, Punctuality, Love and Kindness.

UNIT-IV:
Avoid fault Thinking. Free from anger, Dignity of labor. Universal brother hood and religious tolerance. True friendship. Happiness Vs suffering, love for truth. Aware of self-destructive habits. Association and Cooperation.Doingbestfor saving nature

UNIT-V:
Character and Competence –Holy books vs. Blind faith. Self-management and Good health.Scienceof reincarnation, Equality, Nonviolence, Humility, Role of Women. All religions and same message. Mind your Mind, Self- control. Honesty, Studying effectively.

TEXTBOOKS/REFERENCES:
1. Chakroborty,S.K.“ValuesandEthicsfororganizationsTheoryandpractice”,OxfordUniversityPress,NewDelhi

CONSTITUTION OF INDIA (Audit Course- I & II)

M.Tech SE I Year I Sem./ II Sem.

L	T	P	C
2	0	0	0

Pre requisite: None

Course Objectives: Students will be able to:

- Understand the premises informing the twin themes of liberty and freedom from a civil rights perspective.
- To address the growth of Indian opinion regarding modern Indian intellectuals' constitutional role and entitlement to civil and economic rights as well as the emergence of nationhood in the early years of Indian nationalism.
- To address the role of socialism in India after the commencement of the Bolshevik Revolutionin1917and its impact on the initial drafting of the Indian Constitution.

Course Outcomes: Students will be able to:

- Discuss the growth of the demand for civil rights in India for the bulk of Indians before the arrival of Gandhi in Indian politics.
- Discuss the intellectual origins of the frame work of argument that informed the conceptualization of social reforms leading to revolution in India.
- Discuss the circumstances surrounding the foundation of the Congress Socialist Party [CSP] under the leadership of Jawaharlal Nehru and the eventual failure of the proposal of direct elections through adult suffrage in the Indian Constitution.
- Discuss the passage of the Hindu Code Bill of1956.

UNIT-I:

History of Making of the Indian Constitution: History Drafting Committee, (Composition & Working), **Philosophy of the Indian Constitution:** Preamble**,** Salient Features.

UNIT-II:

Contours of Constitutional Rights & Duties: Fundamental Rights Right to Equality**,** Right to Freedom**,** Right against Exploitation**,** Right to Freedom of Religion**,** Cultural and Educational Rights**,** Right to Constitutional Remedies, Directive Principles of State Policy**,** Fundamental Duties.

UNIT-III:

Organs of Governance: Parliament**,** Composition**,** Qualifications and Disqualifications**,** Powers and Functions**,** Executive**,** President**,** Governor**,** Council of Ministers, Judiciary, Appointment and Transfer of Judges, Qualification, Powers and Functions.

UNIT-IV:

Local Administration: District's Administration head: Role and Importance, Municipalities: Introduction, Mayor and role of Elected Representative, CEO of Municipal Corporation. Pachayati raj: Introduction, PRI: Zila Pachayat. Elected officials and their roles, CEO Zila Pachayat: Position and role. Block level: Organizational Hierarchy (Different departments), Village level: Role of Elected and Appointed officials, Importance of grass root democracy.

UNIT-V:

Election Commission: Election Commission: Role and Functioning. Chief Election CommissionerandElectionCommissioners.StateElectionCommission:RoleandFunctioning.Instituteand Bodies For the welfare of SC/ST/OBC and women.

TEXTBOOKS/REFERENCES:

1. The Constitution of India,1950(Bare Act),Government Publication.
2. Dr. S.N. Busi, Dr.B.R. Ambedkar framing of Indian Constitution, 1st Edition, 2015.
3. M.P.Jain, Indian Constitution Law, 7thEdn.,Lexis Nexis, 2014.
4. D.D. Basu, Introduction to the Constitution of India, Lexis Nexis, 2015.

PEDAGOGY STUDIES (Audit Course- I & II)

M.Tech SE I Year I Sem. / II Sem.

L	T	P	C
2	0	0	0

Pre requisite: None

Course Objectives: Students will be able to:

- Review existing evidence on the review topic to inform programme design and policy making under taken by the DfID, other agencies and researchers.
- Identify critical evidence gaps to guide the development.

Course Outcomes: Students will be able to understand:

- What pedagogical practices are being used by teachers informal and informal classrooms in developing countries?
- What is the evidence on the effectiveness of these pedagogical practices, in what conditions, and with what population of learners?
- How can teacher education (curriculum and practicum) and the school curriculum and guidance materials best support effective pedagogy?

UNIT-I:

Introduction and Methodology: Aims and rationale, Policy background, Conceptual framework and terminology Theories of learning, Curriculum, Teacher education. Conceptual frame work, Research questions**.** Over view of methodology and Searching.

UNIT-II:

The maticover view: Pedagogical practices are being used by teacher's informal and informal classrooms in developing countries. Curriculum, Teacher education.

UNIT-III:

Evidence on the effectiveness of pedagogical practices, Methodology for the in depth stage: quality assessment of included studies. How can teacher education (curriculum and practicum) and the Scholl curriculum and guidance materials best support effective pedagogy? Theory of change. Strength and nature of the body of evidence for effective pedagogical practices. Pedagogic theory and pedagogical approaches. Teachers' attitudes and beliefs and Pedagogic strategies.

UNIT-IV:

Professional development: alignment with classroom practices and follow-up support, Peer support, Support from the head teacher and the community. Curriculum and assessment, Barriers to learning: limited resources and large class sizes

UNIT-V:

Research gaps and future directions: Research design**,** Contexts**,** Pedagogy**,** Teacher education**,** Curriculum and assessment**,** Dissemination and research impact.

TEXTBOOKS/REFERENCES:

1. Ackers J, Hardman F (2001) Classroom interaction in Kenyan primary schools, Compare, 31(2):245-261.
2. Agrawal M (2004) Curricular reform in schools: The importance of evaluation, Journal ofCurriculumStudies,36(3):361-379.
3. Akyeampong K (2003) Teacher training in Ghana - does it count? Multi-site teacher educationresearchproject(MUSTER)countryreport1.London:DFID.
4. Akyeampong K, Lussier K, Pryor J, Westbrook J (2013) Improving teaching and learning of basic maths and reading in Africa: Does teacher preparation count? International JournalEducationalDevelopment,33(3):272-282.

5. Alexander RJ (2001) Culture and pedagogy: International comparisons in primary education. Oxford and Boston: Blackwell.
6. Chavan M (2003) Read India: A mass scale, rapid,‘ learning to read’ campaign.
7. www.pratham.org/images/resource%20working%20paper%202.pdf.

STRESS MANAGEMENT BY YOGA (Audit Course – I & II)

M.Tech SE I Year I Sem. / II Sem.

L	T	P	C
2	0	0	0

Pre requisite: None

Course Objectives:
- To achieve overall health of body and mind.
- To overcome stress.

Course Outcomes: Students will be able to:
- Develop healthy mind in a healthy body thus improving social health also.
- Improve efficiency.

Program Specific Outcomes:
- Demonstrate an understanding of the eight parts of yoga (Ashtanga) and their significance.
- Apply the principles of Yam and Niyam in personal and professional life.
- Identify and practice essential do's and don'ts in life for a balanced lifestyle.

UNIT-I:
Definitions of Eight parts of yoga.(Ashtanga)

UNIT-II:
YamandNiyam.

UNIT-III:
Do`s and Don'ts in life.
i) Ahinsa, satya, astheya, bramhacharya and aparigraha
ii) Shaucha, santosh, tapa, swadhyay, ishwarpranidhan

UNIT-IV:
Asan and Pranayam

UNIT-V:
i) Various yoga poses and their benefits for mind & body
ii) Regularizationofbreathingtechniquesanditseffects-Typesofpranayam

TEXTBOOKS/REFERENCES:
1. 'Yogic Asanas for Group Tarining-Part-I": Janardan Swami Yogabhyasi Mandal, Nagpur
2. "Raja yoga or conquering the Internal Nature" by Swami Vivekananda, Advaita Ashrama (Publication Department), Kolkata.

PERSONALITY DEVELOPMENT THROUGH LIFE ENLIGHTENMENT SKILLS
(Audit Course-I&II)

M.Tech SE I Year I Sem. / II Sem. **L T P C**
2 0 0 0

Pre requisite: None

Course Objectives:

- To learn to achieve the highest goal happily
- To become a person with stable mind, pleasing personality and determination
- To awaken wisdom in students

Course Outcomes: Students will be able to

- Study of Shrimad – Bhagwad Geeta will help the student in developing his personality and achieve the highest goal in life.
- The person who has studied Geeta will lead the nation and mankind to peace and prosperity
- Study of Neeti shatakam will help in developing versatile personality of students

Program Specific Outcomes:

- Demonstrate holistic personality development through the teachings of Neeti Shatakam.
- Cultivate a role model personality based on the values derived from ancient texts.
- Develop a deeper understanding of ethics and moral values for personal and professional growth.
- Engage in reflective practices to enhance self-awareness and wisdom.

UNIT-I: Neeti satakam-Holistic development of personality

- Verses-19,20,21,22(wisdom)
- Verses-29,31,32(pride & heroism)
- Verses-26,28,63,65(virtue)

UNIT-II: Neeti satakam-Holistic development of personality

- Verses-52,53,59(dont's)
- Verses-71,73,75,78(do's)

UNIT-III: Approach to day to day work and duties.

- ShrimadBhagwadGeeta:Chapter2-Verses41,47,48,
- Chapter3-Verses13, 21, 27,35,Chapter 6-Verses5,13,17, 23,35,
- Chapter18-Verses45,46,48.

UNIT-IV: Statements of basic knowledge.

- ShrimadBhagwadGeeta:Chapter2-Verses56,62,68
- Chapter12-Verses13,14,15,16,17, 18
- Personality of Role model. Shrimad Bhagwad Geeta:

UNIT-V:

- Chapter2-Verses17,Chapter3-Verses36,37,42,
- Chapter4-Verses18,38,39
- Chapter18- Verses37,38,63

TEXTBOOKS/REFERENCES:

1. "Shrimad Bhagavad Gita" by Swami Swarupananda Advaita Ashram (Publication Department), Kolkata.
2. Bhartrihari's Three Satakam (Niti-sringar-vairagya) by P.Gopinath, Rashtriya Sanskrit Sansthanam, New Delhi.

TECMRMC02	QUANTUM TECHNOLOGIES AND APPLICATIONS	L	T	P	C
		2	0	0	2

Course Objectives

1. Present core quantum principles such as superposition and entanglement without mathematical formalism.
2. Develop conceptual clarity on qubits, quantum states, and information frameworks.
3. Examine the theoretical challenges in realizing scalable quantum systems.
4. Introduce foundational ideas in quantum communication and computing.
5. Highlight applications, industrial adoption, and future research directions in quantum technologies.

Course Outcomes

- Upon completion, the learner will be able to:
- Explain fundamental quantum concepts conceptually.
- Distinguish classical information systems from quantum information frameworks.
  Identify the principal theoretical limitations in building quantum computers.
- Describe the conceptual basis of quantum communication and computation.
- Discuss current applications, technological trajectories, and career opportunities in the quantum domain.

Unit 1: Foundations of Quantum Theory and Technologies

Transition from classical to quantum physics. Key conceptual principles: Superposition, Entanglement, Uncertainty, Wave-particle duality. Quantum states and measurement; the role of the observer. Representative quantum systems: electrons, photons, atoms. Concept of quantization and discrete energy levels. Strategic relevance of quantum technologies. Overview of major domains: Computing, Communication, Sensing. Global quantum initiatives: India's National Quantum Mission, EU Quantum Flagship, USA, China.

Unit 2: Conceptual Structure of Quantum Information

Qubits: qualitative understanding using spin and polarization. Classical bits vs quantum bits: distinctions and implications. Quantum systems (non-engineering perspective): trapped ions, superconducting qubits, photonics. Coherence and decoherence mechanisms. Abstract notions: quantum states, measurement operators, Hilbert space—interpretation without mathematics. Entanglement and non-locality as foundational resources. Quantum vs classical information principles; philosophical considerations.

Unit 3: Building a Quantum Computer – Challenges and Requirements

Conceptual prerequisites for functional quantum hardware. Fragility of quantum states: decoherence, noise, stability issues. Requirements: isolation, error resilience, scalability, control. Why maintaining entanglement is difficult; theoretical necessity of quantum error correction. Comparative overview of hardware platforms (superconducting circuits, trapped ions, photonics).Current progress vs scientific constraints; conceptual view of quantum software's role.

Unit 4: Quantum Communication and Computing

(Redundant explanations removed, retaining only unique themes.)Quantum vs classical communication paradigms. Essentials of Quantum Key Distribution (QKD) and its security rationale. Entanglement-enabled communication protocols. Concept of the Quantum Internet and secure global networking. Introduction to quantum computing and quantum parallelism.Conceptual comparison of classical and quantum gate operations. Challenges: decoherence, noise, and the necessity of error correction frameworks.

Unit 5: Applications, Industry, and Future Directions

Application domains: Healthcare and drug discovery, Material science and chemistry, Optimization and logistics, Quantum sensing and precision timing. Case studies: IBM, Google, Microsoft, PsiQuantum. Ethical, societal, and policy considerations. Barriers to adoption: cost, skilled workforce, standards. Emerging research and career landscapes; India's strategic opportunity in the global quantum ecosystem.

Textbooks

1. Nielsen & Chuang, Quantum Computation and Quantum Information, Cambridge University Press, 2010.
2. Rieffel & Polak, Quantum Computing: A Gentle Introduction, MIT Press, 2011.
3. Chris Bernhardt, Quantum Computing for Everyone, MIT Press, 2019.

Reference Books

1. David McMahon, Quantum Computing Explained, Wiley, 2008.
2. Kaye, Laflamme, Mosca, An Introduction to Quantum Computing, OUP, 2007. Scott Aaronson, Quantum Computing Since Democritus, CUP, 2013.
3. Susskind & Friedman, Quantum Mechanics: The Theoretical Minimum, Basic Books, 2014. Rosenblum & Kuttner, Quantum Enigma, OUP, 2011.
4. Benenti et al., Principles of Quantum Computation and Information, World Scientific, 2004. DST India and MeitY: Official Quantum Mission Reports, 2020 onwards.
5. Quantum Flagship EU: Roadmaps and Strategy Documents.

Online Learning Resources

- IBM Quantum Experience & Qiskit Textbook Coursera – Quantum Mechanics and Quantum
- Computation (UC Berkeley) edX – Quantum Internet & Quantum Computers YouTube – Quantum Computing for the Determined (Michael Nielsen).

CYBER SECURITY (Open Elective)

M.Tech SE II Year I Sem.

L	T	P	C
3	0	0	3

Course objectives:

- To understand various types of cyber-attacks and cyber-crimes
- To learn threats and risks within context of the cyber security
- To have an overview of the cyber laws & concepts of cyber forensics
- To study the defensive techniques against these attacks

Course Outcomes:

- Analyze and evaluate the cyber security needs of an organization.
- Understand Cyber Security Regulations and Roles of International Law
- Design and develop a security architecture for an organization.
- Understand fundamental concepts of data privacy attacks

UNIT - I

Introduction to Cyber Security: Basic Cyber Security Concepts, layers of security, Vulnerability, threat, Harmful acts, Internet Governance – Challenges and Constraints, Computer Criminals, CIA Triad, Assets and Threat, motive of attackers, active attacks, passive attacks, Software attacks, hardware attacks, Cyber Threats- Cyber Warfare, Cyber Crime, Cyber terrorism, Cyber Espionage, etc., Comprehensive Cyber Security Policy.

UNIT - II

Cyberspace and the Law & Cyber Forensics: Introduction, Cyber Security Regulations, Roles of International Law. The INDIAN Cyberspace, National Cyber Security Policy.

Introduction, Historical background of Cyber forensics, Digital Forensics Science, The Need for Computer Forensics, Cyber Forensics and Digital evidence, Forensics Analysis of Email, Digital Forensics Lifecycle, Forensics Investigation, Challenges in Computer Forensics

UNIT - III

Cybercrime: Mobile and Wireless Devices: Introduction, Proliferation of Mobile and Wireless Devices, Trends in Mobility, Credit card Frauds in Mobile and Wireless Computing Era, Security Challenges Posed by Mobile Devices, Registry Settings for Mobile Devices, Authentication service Security, Attacks on Mobile/Cell Phones, Organizational security Policies and Measures in Mobile Computing Era, Laptops.

UNIT- IV

Cyber Security: Organizational Implications: Introduction, cost of cybercrimes and IPR issues, web threats for organizations, security and privacy implications, social media marketing: security risks and perils for organizations, social computing and the associated challenges for organizations

UNIT - V

Privacy Issues: Basic Data Privacy Concepts: Fundamental Concepts, Data Privacy Attacks, Datalinking and profiling, privacy policies and their specifications, privacy policy languages, privacy in different domains- medical, financial, etc

Cybercrime: Examples and Mini-Cases

Examples: Official Website of Maharashtra Government Hacked, Indian Banks Lose Millions of Rupees, Parliament Attack, Pune City Police Bust Nigerian Racket, e-mail spoofing instances. **Mini- Cases:** The Indian Case of online Gambling, An Indian Case of Intellectual Property Crime, Financial Frauds in Cyber Domain.

TEXT BOOKS:

1. Nina Godbole and SunitBelpure, Cyber Security Understanding Cyber Crimes, Computer Forensics and Legal Perspectives, Wiley
2. B. B. Gupta, D. P. Agrawal, Haoxiang Wang, Computer and Cyber Security: Principles, Algorithm, Applications, and Perspectives, CRC Press, ISBN 9780815371335,2018.

REFERENCES:

1. Cyber Security Essentials, James Graham, Richard Howard and Ryan Otson, CRC Press.
2. Introduction to Cyber Security, Chwan-Hwa(john) Wu,J. David Irwin, CRC Press T&F Group.

FAULT TOLERANCE SYSTEMS (OPEN ELECTIVE)

M.Tech SE II Year I Sem.

L	T	P	C
3	0	0	3

Course Objectives:

1. To know the different advantages and limits of fault avoidance and fault tolerance techniques.
2. To impart the knowledge about different types of redundancy and its application for the design of computer system being able to function correctly even under presence of faults and data errors.
3. To understand the relevant factors in evaluating alternative system designs for a specific set of requirements.
4. To understand the subtle failure modes of "fault-tolerant" distributed systems.

Course Outcomes: Upon the Successful Completion of the Course, the Students would be able to:

1. Become familiar with general and state of the art techniques used in design and analysis of fault tolerant digital systems.
2. Be familiar with making system fault tolerant, modeling and testing, and benchmarking to evaluate and compare systems.

UNIT - I

Introduction to Fault Tolerant Computing: Basic concepts and overview of the course; Faults and their manifestations, Fault/error modeling, Reliability, availability and maintainability analysis, System evaluation, performance reliability tradeoffs.

UNIT - II

System level fault diagnosis: Hardware and software redundancy techniques. Fault tolerant system design methods, Mobile computing and Mobile communication environment, Fault injection methods.

UNIT - III

Software fault tolerance: Design and test of defect free integrated circuits, fault modeling, built in self-test, data compression, error correcting codes, simulation software/hardware, fault tolerant system design, CAD tools for design for testability.

UNIT - IV

Information Redundancy and Error Correcting Codes: Software Problem. Software Reliability Models and Robust Coding Techniques, Reliability in Computer Networks Time redundancy. Re execution in SMT, CMP Architectures, Fault Tolerant Distributed Systems, Data replication.

UNIT - V

Case Studies in FTC: ROC, HP Non-Stop Server. Case studies of fault tolerant systems and current research issues.

TEXT BOOK:

1. Fault Tolerant Computer System Design by D. K. Pradhan, Prentice Hall.

REFERENCES:

1. Fault Tolerant Systems by I. Koren, Morgan Kauffman.
2. Software Fault Tolerance Techniques and Implementation by L. L. Pullum, Artech House Computer Security Series.
3. Reliability of Computer Systems and Networks: Fault Tolerance Analysis and Design by M. L. Shooman, Wiley.

INTRUSION DETECTION SYSTEMS (Open Elective)

M.Tech SE II Year I Sem.

L	T	P	C
3	0	0	3

Prerequisites: Computer Networks, Computer Programming

Course Objectives:

1. Compare alternative tools and approaches for Intrusion Detection through quantitative analysis to determine the best tool or approach to reduce risk from intrusion.
2. Identify and describe the parts of all intrusion detection systems and characterize new and emerging IDS technologies according to the basic capabilities all intrusion detection systems share.

Course Outcomes: After completion of the course, students will be able to:

1. Possess a fundamental knowledge of Cyber Security.
2. Understand what vulnerability is and how to address most common vulnerabilities.
3. Know basic and fundamental risk management principles as it relates to Cyber Security and Mobile Computing.
4. Have the knowledge needed to practice safer computing and safeguard your information usingDigital Forensics.
5. Understand basic technical controls in use today, such as firewalls and Intrusion Detection systems.
6. Understand legal perspectives of Cyber Crimes and Cyber Security.

UNIT - I

The state of threats against computers, and networked systems-Overview of computer security solutions and why they fail-Vulnerability assessment, firewalls, VPN's -Overview of Intrusion Detection and Intrusion Prevention, Network and Host-based IDS

UNIT - II

Classes of attacks - Network layer: scans, denial of service, penetration Application layer: software exploits, code injection-Human layer: identity theft, root access-Classes of attackers-Kids/hackers/sop Hesitated groups-Automated: Drones, Worms, Viruses

UNIT - III

A General IDS model and taxonomy, Signature-based Solutions, Snort, Snort rules, Evaluation of IDS, Cost sensitive IDS

UNIT - IV

Anomaly Detection Systems and Algorithms-Network Behavior Based Anomaly Detectors (rate based)-Host- based Anomaly Detectors-Software Vulnerabilities-State transition, Immunology, Payload Anomaly Detection

UNIT - V

Attack trees and Correlation of alerts- Autopsy of Worms and Botnets-Malware detection - Obfuscation, polymorphism- Document vectors.
Email/IM security issues-Viruses/Spam-From signatures to thumbprints to zero-day detection-Insider Threat issues-Taxonomy-Masquerade and Impersonation Traitors, Decoys and Deception-Future: Collaborative Security

TEXT BOOKS:

1. Peter Szor, The Art of Computer Virus Research and Defense, Symantec Press ISBN 0-321- 30545-3.
2. Markus Jakobsson and Zulfikar Ramzan, Crimeware, Understanding New Attacks and Defenses.

REFERENCE BOOKS:

1. Saiful Hasan, Intrusion Detection System, Kindle Edition.
2. Ankit Fadia, Intrusion Alert: An Ethical Hacking Guide to Intrusion Detection.

Online Websites/Materials:

1. https://www.intechopen.com/books/intrusion-detection-systems/

Online Courses:

1. https://www.sans.org/course/intrusion-detection-in-depth
2. https://www.cybrary.it/skill-certification-course/ids-ips-certification-training-course

OPTIMIZATION TECHNIQUES (Open Elective)

M.Tech SE II Year I Sem.

L	T	P	C
3	0	0	3

Prerequisite: Mathematics –I, Mathematics –II

Course Objectives:

1. To introduce various optimization techniques i.e classical, linear programming, transportation problem, simplex algorithm, dynamic programming
2. Constrained and unconstrained optimization techniques for solving and optimizing electrical and electronic engineering circuits design problems in real world situations.
3. To explain the concept of Dynamic programming and its applications to project implementation.

Course Outcomes: After completion of this course, the student will be able to:

1. explain the need of optimization of engineering systems.
2. understand optimization of electrical and electronics engineering problems.
3. apply classical optimization techniques, linear programming, simplex algorithm, transportation problem.
4. apply unconstrained optimization and constrained non-linear programming and dynamic programming.
5. Formulate optimization problems.

UNIT - I

Introduction and Classical Optimization Techniques: Statement of an Optimization problem – design vector – design constraints – constraint surface – objective function – objective function surface - classification of Optimization problems.

Linear Programming: Standard form of a linear programming problem – geometry of linear programming problems – definitions and theorems – solution of a system of linear simultaneous equations – pivotal reduction of a general system of equations – motivation to the simplex method – simplex algorithm.

UNIT - II

Transportation Problem: Finding initial basic feasible solution by north – west corner rule, least cost method and Vogel's approximation method – testing for optimality of balanced transportation problems. Degeneracy.

Assignment problem – Formulation – Optimal solution - Variants of Assignment Problem; Traveling Salesman problem.

UNIT - III

Classical Optimization Techniques: Single variable Optimization – multi variable Optimization without constraints – necessary and sufficient conditions for minimum/maximum – multivariable Optimization with equality constraints: Solution by method of Lagrange multipliers – Multivariable Optimization with inequality constraints: Kuhn – Tucker conditions.

Single Variable Nonlinear Unconstrained Optimization: Elimination methods: Uni Model function-its importance, Fibonacci method & Golden section method.

UNIT - IV

Multi variable nonlinear unconstrained optimization: Direct search methods – Univariant method, Pattern search methods – Powell's, Hooke - Jeeves, Rosenbrock's search methods. Gradient methods: Gradient of function & its importance, Steepest descent method, Conjugate direction methods: Fletcher-Reeves method & variable metric method.

UNIT - V

Dynamic Programming: Dynamic programming multistage decision processes – types – concept of sub optimization and the principle of optimality – computational procedure in dynamic programming – examples illustrating the calculus method of solution - examples illustrating the tabular method of solution.

TEXT BOOKS:

1. Optimization Techniques & Applications by S.S.Rao, New Age International.
2. Optimization for Engineering Design by Kalyanmoy Deb, PHI

REFERENCES:

1. George Bernard Dantzig, Mukund Narain Thapa, "Linear programming", Springer series in Operations Research 3rd edition, 2003.
2. H. A. Taha, "Operations Research: An Introduction", 8th Edition, Pearson/Prentice Hall, 2007.
3. Optimization Techniques by Belegundu&Chandrupatla, Pearson Asia.
4. Optimization Techniques Theory and Practice by M.C. Joshi, K.M. Moudgalya, Narosa Publications

CYBER PHYSICAL SYSTEMS (Open Elective)

M.Tech SE II Year I Sem.

L	T	P	C
3	0	0	3

Course Objective: To learn about design of cyber-physical systems

Course Outcomes: Upon the Successful Completion of the Course, the Students would be able to:

1. Understand the core principles behind CPS
2. Identify Security mechanisms of Cyber physical systems
3. Understand Synchronization in Distributed Cyber-Physical Systems

UNIT - I

Symbolic Synthesis for Cyber-Physical Systems

Introduction and Motivation, Basic Techniques - Preliminaries, Problem Definition, Solving the Synthesis Problem, Construction of Symbolic Models, Advanced Techniques: Construction of Symbolic Models, Continuous-Time Controllers, Software Tools

UNIT - II

Security of Cyber-Physical Systems

Introduction and Motivation, Basic Techniques - Cyber Security Requirements, Attack Model, Countermeasures, Advanced Techniques: System Theoretic Approaches

UNIT - III

Synchronization in Distributed Cyber-Physical Systems: Challenges in Cyber-Physical Systems, A Complexity-Reducing Technique for Synchronization, Formal Software Engineering, Distributed Consensus Algorithms, Synchronous Lockstep Executions, Time-Triggered Architecture, Related Technology, Advanced Techniques

UNIT - IV

Real-Time Scheduling for Cyber-Physical Systems

Introduction and Motivation, Basic Techniques - Scheduling with Fixed Timing Parameters, Memory Effects, Multiprocessor/Multicore Scheduling, Accommodating Variability and Uncertainty

UNIT - V

Model Integration in Cyber-Physical Systems

Introduction and Motivation, Causality, Semantic Domains for Time, Interaction Models for Computational Processes, Semantics of CPS DSMLs, Advanced Techniques, ForSpec, The Syntax of CyPhyML, Formalization of Semantics, Formalization of Language Integration.

TEXT BOOKS:

1. Raj Rajkumar, Dionisio De Niz, and Mark Klein, Cyber-Physical Systems, Addison-Wesley Professional.
2. Rajeev Alur, Principles of Cyber-Physical Systems, MIT Press, 2015

GRAPH ANALYTICS (Open Elective)

M.Tech SE II Year I Sem.

L	T	P	C
3	0	0	3

Course Objectives:

1. To explore the concept of Graphs and related algorithms.
2. To learn new ways to model, store, retrieve and analyze graph-structured data.
3. To be aware of advanced concepts in graph analytic techniques and its applications.

Course Outcomes: Upon the Successful Completion of the Course, the Students would be able to:

1. Understand Large-scale Graph and its Characteristics
2. Analyze Breadth-First Search Algorithm
3. Illustrate Recent Advances in Scalable Network Generation

UNIT - I

Introduction and Application of Large-scale Graph: Characteristics, Complex Data Sources - Social Networks, Simulations, Bioinformatics; Categories- Social, Endorsement, Location, Co- occurrence graphs; Graph Data structures, Parallel, Multicore and Graph Algorithms

UNIT - II Algorithms: Search and Paths

A Work-Efficient Parallel Breadth-First Search Algorithm (or How To Cope With the Nondeterminism of Reducers), Multi-Objective Shortest Paths

UNIT - III Algorithms: Structure

Multicore Algorithms for Graph Connectivity Problems, Distributed Memory Parallel Algorithms for Massive Graphs, Massive-Scale Distributed Triangle Computation and Applications

UNIT - IV Models

Recent Advances in Scalable Network Generation, Computational Models for Cascades in Massive Graphs, Executing Dynamic Data-Graph Computations Deterministically Using Chromatic Scheduling.

UNIT - V Frameworks and Software

Graph Data Science Using Neo4j, A Cloud-Based Approach to Big Graphs, Interactive Graph Analytics at Scale in Arkouda

TEXT BOOKS:

1. David A. Bader, Massive Graph Analytics, CRC Press

REFERENCES:

1. Stanley Wasserman, Katherine Faust, "Social Network Analysis: Methods and Applications", (Structural Analysis in the Social Sciences), Cambridge University Press, 1995.
2. Matthew O. Jackson, "Social and Economic Networks", Princeton University Press, 2010.
3. Tanja Falkowski, "Community Analysis in Dynamic Social Networks", (Dissertation), University Magdeburg, 2009.

NETWORK PROGRAMMING (Open Elective)

M.Tech SE II Year I Sem.

L	T	P	C
3	0	0	3

Course Objectives:

- To understand inter process and inter-system communication
- To understand socket programming in its entirety
- To understand usage of TCP/UDP / Raw sockets
- To understand how to build network applications

Course Outcomes:

- Understand socket API based programs
- Implement client-server applications using TCP and UDP sockets
- Understand Domain name servicing and multicasting routing protocols
- Analyze network programs using Linux Network based API

UNIT - I

Introduction to Network Programming: OSI model, Unix standards, TCP and UDP & TCP connection establishment and Format, Buffer sizes and limitation, standard internet services, Protocol usage by common internet application.

Sockets: Address structures, value – result arguments, Byte ordering and manipulation function and related functions Elementary TCP sockets – Socket, connect, bind, listen, accept, fork and exec function, concurrent servers. Close function and related function.

UNIT - II

TCP client server: Introduction, TCP Echo server functions, Normal startup, terminate and signal handling server process termination, Crashing and Rebooting of server host shutdown of server host.

Elementary UDP sockets: Introduction UDP Echo server function, lost datagram, summary of UDP example, Lack of flow control with UDP, determining outgoing interface with UDP.

I/O Multiplexing: I/O Models, select function, Batch input, shutdown function, poll function, TCP Echo Server

UNIT - III

Socket options: get sock opt and set sock opt functions. Socket states, Generic socket option IPV6 socket option ICMPV6 socket option IPV6 socket option and TCP socket options.

Advanced I/O Functions-Introduction, Socket Timeouts, recv and send Functions, read v and write v Functions, recv msg and send msg Functions, Ancillary Data, How Much Data Is Queued?, Sockets and Standard I/O, T/TCP: TCP for Transactions.

UNIT - IV

Elementary name and Address conversions: DNS, get host by Name function, Resolver option, Function and IPV6 support, u name function, other networking information.

Daemon Processes and in etd Super server- Introduction, sys log d Daemon, sys log Function, daemon_init Function, inetd Daemon, daemon_inetd Function

Broadcasting- Introduction, Broadcast Addresses, Unicast versus Broadcast, dg_cli Function Using Broadcasting, Race Conditions

Multicasting- Introduction, Multicast Addresses, Multicasting versus Broadcasting on A LAN, Multicasting on a WAN, Multicast Socket Options, mcast_join and Related Functions, dg_cli Function Using Multicasting, Receiving MBone Session Announcements, Sending and Receiving, SNTP: Simple Network Time Protocol, SNTP

UNIT - V

Raw Sockets-Introduction, Raw Socket Creation, Raw Socket Output, Raw Socket Input, Ping Program, Traceroute Program, An ICMP Message Daemon,

Datalink Access- Introduction, BPF: BSD Packet Filter, DLPI: Data Link Provider Interface

Linux: SOCK_PACKET, libpcap: Packet Capture Library, Examining the UDP Checksum Field.

Remote Login: Terminal line disciplines, Pseudo-Terminals, Terminal modes, Control Terminals, rlogin Overview, RPC Transparency Issues.

TEXT BOOKS:

3. UNIX Network Programming, by W. Richard Stevens, Bill Fenner, Andrew M. Rudoff, Pearson Education
4. UNIX Network Programming, 1st Edition, - W. Richard Stevens. PHI.

REFERENCE BOOKS:

3. UNIX Systems Programming using C++ T CHAN, PHI.
4. UNIX for Programmers and Users, 3rd Edition Graham GLASS, King abls, Pearson Education
5. Advanced UNIX Programming 2nd Edition M. J. ROCHKIND, Pearson Education

AGILE DEVELOPMENT METHODOLOGIES (Open Elective)

M.Tech SE II Year I Sem.

L	T	P	C
3	0	0	3

Course Objectives:

- Understand the principles and practices of Agile Development methodologies, focusing on Extreme Programming (XP).
- Explore the importance of collaboration and customer involvement in successful Agile projects.
- Learn the strategies for releasing software with high quality and fast delivery.
- Develop skills in planning iterations, managing risks, and estimating project timelines effectively.
- Gain practical experience in incremental development techniques, including test-driven development and refactoring.

Course Outcomes:

- Demonstrate knowledge of Agile methodologies and their application in real-world software development.
- Collaborate effectively in teams, utilizing Agile practices to enhance productivity and communication.
- Produce high-quality software releases through effective version control and continuous integration.
- Create and manage effective release plans and iterations while identifying and mitigating risks.
- Apply incremental development practices, including customer testing and exploratory testing, to enhance software quality.

UNIT-I

Introduction Extreme Programming (XP)- Agile Development

Why Agile-Understanding Success, Beyond Deadlines, Importance of Organizational Success, Introduction to Agility How to Be Agile - Agile methods, Don't make your own method, Road to mastery Understanding XP (Extreme Programming) - XP life cycle, XP team, XP Concepts AdoptingXP- Knowing whetherXPi ssuitable, Implementing XP, assessing Agility Practicing XP- Thinking – Pair Programming, Energized work, Informative Work space, Root cause Analysis, Retrospectives

UNIT-II

Collaborating

Trust, Sit together, Real customer involvement, Ubiquitous language, meetings, coding standards, Iteration demo, Reporting

UNIT-III

Releasing

Bug free Release, Version Control, fast build, continuous integration, Collective ownership, Documentation

UNIT-IV

Planning

Version, Release Plan, Risk Management, Iteration Planning, Slack, Stories, Estimating

UNIT-V

Developing:

Increment al requirements, Customer tests, Test driven development, Refactoring,
Increment al design and architecture, spike solutions, Performance optimization, Exploratory testing

Text Book:

The art of Agile Development, James Shore and Shane Warden, 11th Indian Reprint, O'Reilly,2018

References:

Learning Agile, Andrew Stellman and Jennifer Greene,O'Reilly,4th Indian Reprint,2018

Practices of an Agile Developer, VenkatSubramaniam and Andy Hunt, SPD, 5th Indian Reprint, 2015

Agile Project Management – Jim High smith, Pearson Low price Edition 2004.

SECURE SOFTWARE ENGINEERING (Open Elective)

M.Tech SE II Year I Sem.

L	T	P	C
3	0	0	3

Course Objectives:

- Students will demonstrate knowledge of the distinction between critical and non-critical systems.
- Students will demonstrate the ability to manage a project including planning, scheduling and risk assessment / management.
- Students will author a software requirements document.
- Students will demonstrate an understanding of the proper contents of a soft ware requirements document.
- Students will author a formal specification for a software system.
- Students will demonstrate an understanding of distributed system architectures and application architectures.
- Students will demonstrate an understanding of the differences between real-time and non-real time systems.
- Students will demonstrate proficiency in rapid software development techniques.
- Students will be able to identify specific components of a soft ware design that can be targeted for reuse.
- Students will demonstrate proficiency in software development cost estimation.
- Students will author a soft ware testing plan.

Course Outcomes:

- Explain the key concepts of software security and its significance in the software development process.
- Identify common threats to software security and evaluate their potential impact on applications.
- Apply secure requirements engineering techniques, including the SQUARE process model, to prioritize security needs.
- Design secure software architectures using established security principles and guidelines.
- Implement secure coding practices and perform security testing to detect vulnerabilities in software systems.

UNIT–I

Security a software Issue: Introduction, the problem, Software Assurance and Soft ware Security, Threats to software security, Sources of software insecurity, Benefits of Detecting Software Security

What Makes Software Secure: Properties of Secure Software, Influencing the security properties of software, Asserting and specifying the desired security properties?

UNIT–II

Requirements Engineering for secure software: Introduction, the SQUARE process Model, Requirements elicitation and prioritization

UNIT–III

Secure Software Architecture and Design: Introduction, software security practices for architecture and design: architectural risk analysis, software security knowledge for architecture and design: security principles, security guidelines and attack patterns

Secure coding and Testing: Code analysis, Software Security testing, Security testing considerations throughput the SDLC

UNIT–IV

Security and Complexity: System Assembly Challenges: introduction, security failures, functional and attacker perspectives for security analysis, system complexity drivers and security

UNIT–V

Governance and Managing for More Secure Software: Governance and security, Adopting an enterprise software security frame work, How much securityis enough?, Security and project management, Maturity of Practice

Textbook:

1. Software Security Engineering: Julia H .Allen, Pearson Education

References:

1. Developing Secure Software: Jason Grembi, Cengage Learning Software Security: Richard S inn,CengageLearning

PRINCIPALS OF SOFTWARE ENGINEERING (Open Elective)

M.Tech SE II Year I Sem.

L	T	P	C
3	0	0	3

Course Objectives

- The aim of the course is to provide an understanding of the working knowledge of the techniques for estimation, design, testing and quality management of large software development projects.
- Topics include process models, software requirements, software design, software testing, software process/product metrics, risk management, quality management and UML diagrams

Course Outcomes

- Ability to translate end-user requirements into system and software requirements, using e.g.
- UML, and structure the requirements in a Software Requirements Document (SRD).
- Identify and apply appropriate software architectures and patterns to carry out high level design of a system and be able to critically compare alternative choices.
- Will have experience and/or awareness of testing problems and will be able to develop a simple testing report.

UNIT - I

Introduction to Software Engineering: The evolving role of software, changing nature of software, software myths.

A Generic view of process: Software engineering- a layered technology, a process framework, the capability maturity model integration (CMMI)

Process models: The waterfall model, Incremental model, Spiral model and Agile Development

UNIT - II

Software Requirements: Functional and non-functional requirements, user requirements, system requirements, interface specification, the software requirements document.

Requirements engineering process: Feasibility studies, requirements elicitation and analysis, requirement validation, requirements management.

UNIT - III

Design Engineering: Design process and design quality, design concepts, the design model.

Creating an architectural design: software architecture, data design, architectural styles and patterns, architectural design, conceptual model of UML, basic structural modeling, class diagrams, sequence diagrams, collaborationdiagrams, use case diagrams, component diagrams.

UNIT - IV

Testing Strategies: A strategic approach to software testing, test strategies for conventional software, black-box and white-box testing, validation testing, system testing, the art of debugging.

Metrics for Process and Products: Software measurement, metrics for software quality.

UNIT - V

Risk management: Reactive Vs proactive risk strategies, software risks, risk identification, risk projection, risk refinement, RMMM

Quality Management: Quality concepts, software quality assurance, software reviews, formal technical reviews, statistical software quality assurance, software reliability, the ISO 9000 quality standards.

TEXT BOOKS:

1. Software Engineering, A practitioner's Approach- Roger S. Pressman, 7th edition, McGraw Hill International Edition.
2. Software Engineering- Sommerville, 7th edition, Pearson Education.

REFERENCES:

1. The unified modeling language user guide Grady Booch, James Rambaugh, Ivar Jacobson, Pearson Education.
2. Software Engineering, an Engineering approach- James F. Peters, Witold Pedrycz, John Wiley.
3. Software Engineering principles and practice- Waman S Jawadekar, The McGraw-Hill Companies.

Fundamentals of object-oriented design using UML Meiler page-Jones: Pearson Education.