Gyanmanjari
Innovative University

Course Syllabus
Faculty of Engineering & Technology
Semester-1(M.Tech)

Subject: Advance Software Engineering-METCE11501

Type of course: Major (Core)

Prerequisite: Basic knowledge of Software Engineering is required.

Rationale:

This course covers advanced software engineering principles and techniques. It includes topics such as software architecture, software analysis, domain-specific techniques, requirements and specifications, advanced design and modeling techniques, formal methods, software testing, and software quality standards.

Teaching and Examination Scheme:

Teaching Scheme			Credits	Examination Marks					Total Marks
CI	T	P	C	Theory Marks		Practical Marks		CA	
				ESE	MSE	V	P	ALA	
4	0	2	5	60	40	10	20	30	150

Legends: CI-Classroom Instructions; T – Tutorial; P - Practical; C – Credit; ESE - End Semester Examination; MSE- Mid Semester Examination; V – Viva; CA - Continuous Assessment; ALA- Active Learning Activities.

GYANMANJARI

Course Content:

Sr. No	Course content	Hrs.	% Weightage
1	**Introduction:** Software product and process, Phases in software development, Software development process models, Agile methodologies, DevOps principles, Software lifecycle models	10	15%
2	**Software Analysis and Design:** Analysis methods, Software requirements specification (SRS), Functional and non-functional requirements, System design methods, Detailed design, Architectural design, Component-level design, User interface design, Design patterns, Use case modeling, UML diagrams	10	15%
3	**Implementation and Testing:** Software testing, Software testing strategies, Testing conventional applications, Object-oriented testing (OO testing), Unit testing, Integration testing, System testing, Development testing, Test-driven development (TDD), Continuous integration and testing, Release testing, User testing, Regression testing, Software maintenance, Quality management	10	25%
4	**Software Quality:** Quality standards, ISO/IEC 9126, ISO/IEC 25010, Capability Maturity Model Integration (CMMI), Quality assurance, Measuring aspects of software quality, Software reliability, Software maintainability, Verification and validation	10	15%
5	**Software Metrics:** Software measurement and metrics, Software quality metrics, Metrics for software products, Metrics	10	20%

GYANMANJARI INNOVATIVE UNIVERSITY • BHAVNAGAR •

	for software processes and projects. Productivity metrics, Complexity metrics, Defect density, Code churn, Function point analysis, Cyclomatic complexity		
6	**Advanced Software Engineering:** Software reuse, Component-based software engineering, Distributed software engineering, Service-oriented architecture (SOA), Cloud-based software engineering, Software product lines, Software evolution, Emerging trends in software engineering.	10	10%

Continuous Assessment:

Sr. No	Active Learning Activities	Marks
1	**Component-Based Software Engineering:** Students will implement a small software module using component-based architecture with tools like Java Beans, .NET Assemblies, or Web Components. They will design reusable components with defined interfaces and demonstrate their interaction within the system. UML diagrams, code snippets, and explanations must be included to showcase the component structure. A well-structured report (PDF) must be submitted individually on the GMIU web portal.	10
2	**Distributed Software Systems:** Students will design and simulate a distributed application such as a chat server or file-sharing system using tools like RabbitMQ, gRPC, REST APIs, or Microservices with Docker. They will demonstrate how components communicate across the network and manage distributed tasks. The submission must include an architecture diagram, communication flow, code samples, and screenshots. A well-structured report (PDF) with detailed explanations must be uploaded individually on the GMIU web portal.	10

3	**Cloud-Based Software Engineering:** Students will deploy a simple application using platforms like AWS, Google Cloud, Heroku, or Firebase. They will design the architecture and demonstrate the live deployment. The report must include deployment steps, architecture, screenshots, and a brief on scalability and reliability. A well-structured PDF must be uploaded individually on the GMIU web portal.	10
	Total	30

Suggested Specification table with Marks (Theory):60

Distribution of Theory Marks **(Revised Bloom's Taxonomy)**						
Level	Remembrance (R)	Understanding (U)	Application (A)	Analyze (N)	Evaluate (E)	Create (C)
Weightage	25%	25%	25%	15%	10%	NA

Course Outcome:

After learning the course, the students should be able to:	
CO1	Explain software engineering principles, lifecycle models, Agile and DevOps practices.
CO2	Analyze requirements and design software using SRS, UML, and design patterns.
CO3	Apply testing techniques for different stages of software development.
CO4	Evaluate software quality using standards like ISO/IEC and CMMI.
CO5	Utilize software metrics to assess product, process, and project performance.
CO6	Implement advanced techniques such as component reuse, distributed and cloud-based development.

List of Practical

Sr. No	Descriptions	Unit No	Hrs
1	Study of SDLC Life Cycle process. Create a JIRA Account and learn interface.	1	02
2	Implement AGILE Framework for any given project definition.	2	02
3	Create use case diagrams to capture functional requirements for a given project definition	3	02
4	Design a software system using UML diagrams for a given project definition.	5	04
5	Design the database schema for a software system using ER diagrams for the given software definition.	3	04
6	Decompose a system into modules and design each module in detail	3	04
7	Develop algorithms and pseudocode for system components	4	02
8	Create a prototype of the user interface for a software application	3	04
9	Evaluate the usability of a user interface design through the testing tools.	3	04

10	Design the architecture of a software system considering scalability and performance	6	02
		Total	30

Text Book :

1. Software Engineering. Pressman. McGraw-Hill. 1992
2. Software Metrics . Fenton & Pfleeger. PWS Publication
3. Watts S. Humphrey. Managing The Software Process. Addison Wesley. 1989
4. Encyclopedia Of Software Engineering . J.J. Marciniak. Ed.. Vols. 1 And 2 John Wiley.1994

REFERENCE BOOKS:

1.Ian Somerville. "Software Engineering". 9th edition. 2010. Pearson Education.

2 Roger S. "Software Engineering – A Practitioner's Approach". 7th Edition. Pressman. 2010.

3.Craig Larman. "Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development". Third Edition. Pearson Education. 2005.

GYANMANJARI INNOVATIVE UNIVERSITY • BHAVNAGAR •