4/4 B.Tech.   SEVENTH SEMESTER

EM7T5   **SOFTWARE ENGINEERING**   **Credits: 4**

Lecture: 4 periods/week   **Internal assessment: 30 marks**
Tutorial: 1 period /week   **Semester end examination: 70 marks**

Course Objectives

To give an idea about
- Basics of software engineering principles
- Role of a process and a various process models in project development.
- How to prepare requirements document and its validation and testing techniques

Learning Outcomes

After completion of the course Students will be able to
- Apply Software Engineering Techniques and Process models for software development
- Document and evaluate the milestones at every phase development process.
- Analyze the risk happening and its management.
- Know how to test the software at different levels by preparing a proper test plans.

UNIT I:
Introduction to Software Engineering: The Evolving Role of Software, Software, The Changing Nature of Software, Legacy Software, Software Myths.
A Generic View of Process: Software Engineering - A Layered Technology, A Process Framework, The CMMI, Process Patterns, Process Assessment, Personal and Team Process Models, Process Technology, Product and Process.

Unit II
Process Models: Prescriptive Models, The Waterfall Model, Incremental Process Models, Evolutionary Models, Specialized Process Models, The Unified Process
An Agile View of Process: What Is Agility? , What Is an Agile Process? , Agile Process Models.

UNIT III:
Software Engineering Practice: Software Engineering Practice, Communication Practices, Planning Practices, Modeling Practices, Construction Practice, Deployment.
Requirements Engineering: A Bridge To Design and Construction, Requirements Engineering Tasks, Initiating the Requirements Engineering Process, Eliciting Requirements, Developing Use-cases, Building the Analysis Model, Negotiating Requirements, Validating Requirements.

UNIT IV:
Building the Analysis Model: Requirements Analysis, Analysis Modeling Approaches, Data Modeling Concepts, Flow-Oriented Modeling, Creating a Behavioral Model.
Design Engineering: Design within the Context of Software Engineering, Design Process and Design Quality, Design Concepts, The Design Model, Pattern-Based Software Design.

Unit V:
Creating an Architectural Design: Software Architecture, Data Design, Architectural Styles and Patterns, Architectural Design, Assessing Alternative Architectural Designs, Mapping Data Flow into Software Architecture.
Modeling Component-Level Design: What Is a Component? , Designing Class-Based Components, Conducting Component-Level Design, Designing Conventional Components.

Unit VI:
Performing User Interface Design: The Golden Rules, User Interface Analysis and Design, Interface Analysis, Interface Design Steps, Design Evaluation.
Testing Strategies: A Strategic Approach to Software Testing, Strategic Issues, Test Strategies for Conventional Software, Test Strategies for Object-Oriented Software, Validation testing, System testing, The art of debugging.

Unit VII:
Testing Tactics: Software Testing Fundamentals, Black-Box and White-Box Testing, White-Box Testing, Basis Path Testing, Control Structure Testing, Black-Box Testing, Object-Oriented Testing Methods, Testing Methods Applicable at the Class Level, Interclass Test Case Design.
Quality Management : Quality concepts, Software quality assurance, Software Reviews, Formal technical reviews, Statistical Software quality Assurance, Software reliability, The ISO 9000 quality standards.

Unit VIII:
Estimation: Observations on estimation, The project planning process, Software project estimation, Decomposition techniques, Empirical estimation models, Estimation for O-O Projects, Specialized Estimation techniques, The make/buy decision.

Text Book:
1. Roger S.Pressman, Software Engineering- A Practitioner's Approach. 6 ed, Tata McGraw-Hill International
Reference Books:

1. Ian Somerville, Software Engineering. 6 ed, Pearson Education.
2. Carlo Ghezzi, Mehdi Jazayeri and Dino Mandrioli, Fundamentals of Software Engineering. 2 ed, PHI.
3. RajibMall, Fundamentals of Software Engineering. 2