SEMESTER	3
Paper Number	MCMS 4311
Paper Title	Software Engineering & Design
No. of credits	6
Theory / Composite	Composite
No. of periods assigned	Th: 4 Pr: 4
Name of faculty member(s)	
Course description / objectives	On completion of this course, the students will be able to: 1. understand the relevance of software engineering and why it is considered as a branch of engineering 2. get introduced to the detailed process of requirements engineering and be able to write SRS 3. recognize importance of design and analyse various design features 4. learn various design concepts
Syllabus	**Theory – 60 marks** Software Engineering Fundamentals: Nature of Software; Software Application Domains, Legacy Software, Changing Nature of Software: WebApps, Mobile Applications, Cloud Computing, Product Line Software; Software Process. Software Requirements Analysis; Requirements engineering: Functional and non-functional requirements, Software requirements document, Requirements specification, Requirements engineering processes, Requirements elicitation and analysis, Requirements validation, Requirements management. Software Design: The Design Process, Software Quality Guidelines and Attributes, Design Concepts: Abstraction, Architecture, Patterns, Separation of Concerns, Modularity, Information Hiding, Functional Independence, Refinement, Aspects, Refactoring; Object-Oriented Design Concepts, Design Classes; Design Model and Design Elements. Effort Estimation: Function Points, COCOMO, Project scheduling and staffing Software Testing: Strategy and Environment, Building Software Testing Process, Types of Software Testing, Black Box and White Box Software Testing Techniques, Software Testing Life Cycle,

	Software Testing Tools. Software Process and Project Management: Process Reference Models, Workflows and Checkpoints of process, Process Planning, Project Organizations, Project Control and process instrumentation. Software Quality Assurance and Testing: Software Quality Assurance and Standards, Quality Standards, SERVQUAL, Software Process Maturity. Component-based Software Development: Component definition, Planning Team Roles for CBD, The Design of Software Component Infrastructures - Software Components and the UML, Component Technologies. **Lab – 40 marks**
Reading/Reference Lists	1. Roger S. Pressman, Software Engineering - A Practitioner's Approach, McGraw- Hill 2. Somerville, Software Engineering, Pearson Education
Evaluation	Total – 100 (Theory – 60, Practical – 40) Theory – CIA – 10 Semester Examination – 50