Staff Structured Lesson Plan

Faculty Name: D. Saravana Prakash	**Subject Name**: Software Engineering	**Class** : III-B.Sc – CS – AI

Unit – I: Introduction to Software Engineering:

Syllabus: Introduction: The software engineering discipline, programs vs. software products, why study software engineering, emergence of software engineering, Notable changes in software development practices, computer systems engineering.

Hours	(SLO)/Chapter	Taxonomy Level	General Objective	Learning Objectives	Pedagogy	Question Bank
1	Introduction to Software, Definitions	Remember	Define fundamental concepts in SE	1. To define software. 2. To define software engineering. 3. To state the scope of software engineering.	PPT Lecture	1. What is software? 2. What is software engineering? 3. What is the scope of software engineering?
2	Programs vs. Software Products	Remember	Differentiate software products from programs	4. To define a computer program. 5. To define a software product. 6. To differentiate between a program and a software product.	PPT Lecture & Live Demo	4. What is a computer program? 5. What is a software product? 6. How would you differentiate between a program and a software product?
3	Software Crisis	Understand	Analyze the software crisis	7. To list factors that differentiate software products from programs. 8. To define software crisis. 9. To identify symptoms of the software crisis. 10. To explain the impact of the software crisis on project failures.	PPT Lecture & Group Discussion	7. What are the factors that differentiate software products from programs? 8. What is software crisis? 9. What are the symptoms of the software crisis? 10. How does the software crisis impact project failures?
4	Software Development Projects	Understand	Classify software development projects	11. To define software development project. 12. To classify types of software development projects.	PPT Lecture & Quiz	11. What is a software development project? 12. How would you classify types of software development projects?

Hours	(SLO)/Chapter	Taxonomy Level	General Objective	Learning Objectives	Pedagogy	Question Bank
				13. To list examples of software projects undertaken by Indian companies.		13. Can you list examples of software projects undertaken by Indian companies?
5	Exploratory Development	Understand	Understand exploratory development	14. To define exploratory style of software development. 15. To describe perceived problem complexity in software development. 16. To list principles deployed to overcome human cognitive limitations in software engineering.	PPT Lecture & Case Study	14. What is the exploratory style of software development? 15. How is perceived problem complexity in software development described? 16. What principles are deployed to overcome human cognitive limitations in software engineering?
6	Evolution of Software Engineering	Understand	Trace evolution of software engineering	17. To describe the evolution of software engineering as a discipline. 18. To define early computer programming. 19. To describe high-level language programming.	PPT Lecture & Article Reading	17. How would you describe the evolution of software engineering as a discipline? 18. What is early computer programming? 19. How is high-level language programming described?
7	Design Paradigms	Remember	Define design paradigms	20. To define control flow-based design. 21. To define data structure-oriented design. 22. To define data flow-oriented design. 23. To define object-oriented design.	PPT Lecture	20. What is control flow-based design? 21. What is data structure-oriented design? 22. What is data flow-oriented design? 23. What is object-oriented design?
8	Developments in Software Design	Analyze	Review design practice trends	24. To describe recent developments in software design practices. 25. To list notable changes in software development practices.	PPT Lecture	24. What are the recent developments in software design practices?

Hours	(SLO)/Chapter	Taxonomy Level	General Objective	Learning Objectives	Pedagogy	Question Bank
						25. What are the notable changes in software development practices?
9	Computer Systems Engineering	Remember	Differentiate systems & SE	26. To define computer systems engineering. 27. To list components of a computer-based system. 28. To describe the difference between computer systems engineering and software engineering.	PPT Lecture & Activity Based Learning	26. What is computer systems engineering? 27. What are the components of a computer-based system? 28. How would you describe the difference between computer systems engineering and software engineering?
10	Software Process Concepts	Remember	Explain lifecycle fundamentals	29. To define a model. 30. To define life cycle. 31. To explain life cycle models. 32. To define a software life cycle model. 33. To describe the importance of life cycle models in software development.	PPT Lecture & Quiz	29. What is a model? 30. What is life cycle? 31. What are life cycle models and how are they explained? 32. What is a software life cycle model? 33. Why are life cycle models important in software development?
11	Classical Waterfall Model	Understand	Understand waterfall model	34. To define the classical waterfall model. 35. To list phases of the classical waterfall model. 36. To describe relative effort distribution in waterfall phases.	PPT Lecture & Case Study	34. What is the classical waterfall model? 35. What are the phases of the classical waterfall model? 36. How is the relative effort distributed in waterfall phases?
12	Iterative, V-model, Prototyping	Remember	Examine iterative and prototyping models	37. To define the iterative waterfall model. 38. To define the V-model. 39. To define the prototyping model.	PPT Lecture	37. What is the iterative waterfall model? 38. What is the V-model? 39. What is the prototyping model?

Hours	(SLO)/Chapter	Taxonomy Level	General Objective	Learning Objectives	Pedagogy	Question Bank
				40. To list steps in developing a prototype.		40. What are the steps in developing a prototype?
13	Incremental, Evolutionary, RAD	Remember	Explore incremental and RAD models	41. To describe the incremental development model. 42. To define the evolutionary model. 43. To define rapid application development (RAD). 44. To list activities in the RAD model. 45. To describe the applicability of the RAD model.	PPT Lecture	41. How would you describe the incremental development model? 42. What is the evolutionary model? 43. What is rapid application development (RAD)? 44. What are the activities in the RAD model? 45. How would you describe the applicability of the RAD model?
14	Agile, XP, Scrum, Lean SD	Analyze	Compare agile and traditional models	46. To compare RAD with other models. 47. To define agile development model. 48. To list features of agile models. 49. To define Extreme Programming (XP). 50. To define Scrum. 51. To define Lean Software Development.	PPT Lecture & Case Study	46. How does RAD compare with other software process models? 47. What is the agile development model? 48. What are the features of agile models? 49. What is Extreme Programming (XP)? 50. What is Scrum? 51. What is Lean Software Development?
15	Spiral Model	Understand	Understand spiral model	52. To define the spiral model. 53. To list phases of the spiral model. 54. To describe risk management in the spiral model.	PPT Lecture &	52. What is the spiral model? 53. What are the phases of the spiral model? 54. How is risk management described in the spiral model?
16	Model Selection and Evaluation	Analyze	Evaluate & select process models	55. To list advantages of prototyping model.	PPT Lecture &	55. What are the advantages of the prototyping model?

Hours	(SLO)/Chapter	Taxonomy Level	General Objective	Learning Objectives	Pedagogy	Question Bank
				56. To list disadvantages of the waterfall model. 57. To identify circumstances under which the iterative model is used. 58. To differentiate agile and traditional models. 59. To explain selection criteria for a life cycle model. 60. To describe factors influencing the choice of a model. 61. To define system integration. 62. To define verification in life cycle models. 63. To define validation in life cycle models.	Comparative Analysis	56. What are the disadvantages of the waterfall model? 57. When is the iterative model used? 58. How would you differentiate between agile and traditional models? 59. What are the selection criteria for a life cycle model? 60. What factors influence the choice of a software life cycle model? 61. What is system integration? 62. What is verification in life cycle models? 63. What is validation in life cycle models?

Unit – II: Requirements Analysis and Specification:

Requirements gathering and analysis, Software requirements specification (SRS)Software Design: Good software design, cohesion and coupling, neat arrangement, software design approaches, object- oriented vs function-oriented design

Hours	(SLO)/Chapter	Taxonomy Level	General Objective	Learning Objectives	Pedagogy	Question Bank
1	Requirements Analysis	Remember	To define requirements analysis	1. To define requirements analysis. 2. To explain the purpose of requirements analysis. 3. To list activities involved in requirements analysis.	PPT Lecture & Case Study	1. Define requirements analysis. 2. What is the purpose of requirements analysis? 3. What are the activities involved in requirements analysis?
2	Requirements Gathering Techniques	Remember	To define requirement gathering techniques	4. To define requirements gathering. 5. To list techniques for requirements gathering. 6. To define interviewing as a requirement gathering technique. 7. To define questionnaire as a requirement gathering technique. 8. To define observation as a requirement gathering technique. 9. To define document analysis as a requirement gathering technique.	PPT Lecture & Case Study	4. What is requirement gathering? 5. List techniques for requirement gathering. 6. Define interviewing as a gathering technique. 7. What is a questionnaire in gathering requirements? 8. Define observation as a technique. 9. Define document analysis.
3	Prototyping and JAD	Understand	To explain modern gathering strategies	10. To define joint application development (JAD). 11. To describe brainstorming in requirement gathering. 12. To define prototyping in requirements analysis.	PPT Lecture & Activity Based Learning	10. What is JAD? 11. Describe brainstorming as a requirement gathering method. 12. Define prototyping in requirements analysis.
4	Requirements Specification	Understand	To explain requirements specification	13. To define requirements specification. 14. To explain the purpose of requirements specification.	PPT Lecture	13. Define requirements specification. 14. What is the purpose of requirements specification? 15. Define SRS.

Hours	(SLO)/Chapter	Taxonomy Level	General Objective	Learning Objectives	Pedagogy	Question Bank
				15. To define Software Requirements Specification (SRS). 16. To list characteristics of a good SRS.		16. What are the characteristics of a good SRS?
5	SRS Qualities	Understand	To understand SRS quality aspects	17. To define correctness in SRS. 18. To define unambiguity in SRS. 19. To define completeness in SRS. 20. To define consistency in SRS. 21. To define verifiability in SRS. 22. To define modifiability in SRS. 23. To define traceability in SRS.	PPT Lecture & Quiz	17. Define correctness in SRS. 18. Define unambiguity. 19. Define completeness. 20. Define consistency. 21. Define verifiability. 22. Define modifiability. 23. Define traceability.
6	SRS Components	Remember	To list and describe SRS components	24. To list components of an SRS document. 25. To describe functional requirements in SRS. 26. To describe non-functional requirements in SRS.	PPT Lecture	24. What are the components of an SRS? 25. Describe functional requirements. 26. Describe non-functional requirements.
7	Non-functional Requirements	Remember	To define non-functional attributes	27. To define performance requirements. 28. To define security requirements. 29. To define reliability requirements. 30. To define portability requirements. 31. To define usability requirements. 32. To define interface requirements.	PPT Lecture & Case Study	27. Define performance requirements. 28. Define security requirements. 29. Define reliability requirements. 30. Define portability requirements. 31. Define usability requirements. 32. Define interface requirements.

Hours	(SLO)/Chapter	Taxonomy Level	General Objective	Learning Objectives	Pedagogy	Question Bank
8	System Models	Understand	To describe models used in requirements	33. To describe system models in requirements specification. 34. To define data flow diagram (DFD) in requirements specification. 35. To define entity-relationship diagram (ERD) in requirements specification. 36. To define state transition diagram (STD) in requirements specification.	PPT Lecture & Activity Based Learning	33. What are system models in SRS? 34. Define DFD. 35. Define ERD. 36. Define STD.
9	Importance of SRS	Understand	To explain the value of SRS	37. To explain the importance of SRS in software development. 38. To describe the benefit of user involvement in requirements specification.	PPT Lecture	37. Why is SRS important in development? 38. What is the benefit of user involvement in SRS?
10	Software Design Basics	Remember	To define software design and good design	39. To define software design. 40. To explain the purpose of software design. 41. To define good software design. 42. To list characteristics of good software design.	PPT Lecture	39. Define software design. 40. What is the purpose of software design? 41. Define good software design. 42. What are the characteristics of good software design?
11	Cohesion and Coupling	Understand	To understand modular design	43. To define cohesion in software design. 44. To define coupling in software design. 45. To list types of cohesion. 46. To list types of coupling. 47. To explain the importance of high cohesion. 48. To explain the importance of low coupling.	PPT Lecture & Quiz	43. Define cohesion. 44. Define coupling. 45. List types of cohesion. 46. List types of coupling. 47. Why is high cohesion important? 48. Why is low coupling important? 49. What is a module?

Hours	(SLO)/Chapter	Taxonomy Level	General Objective	Learning Objectives	Pedagogy	Question Bank
				49. To define module in software design. 50. To describe neat arrangement in software design.		50. Describe neat arrangement in design.
12	Design Approaches	Understand	To describe software design techniques	51. To define software design approaches. 52. To define top-down design. 53. To define bottom-up design. 54. To define structured design. 55. To define function-oriented design. 56. To define object-oriented design. 57. To list characteristics of object-oriented design. 58. To list characteristics of function-oriented design. 59. To describe the difference between object-oriented and function-oriented design. 60. To define design documentation. 61. To describe the importance of design documentation.	PPT Lecture	51. What are software design approaches? 52. Define top-down design. 53. Define bottom-up design. 54. What is structured design? 55. What is function-oriented design? 56. What is object-oriented design? 57. List characteristics of object-oriented design. 58. List characteristics of function-oriented design. 59. How is object-oriented design different from function-oriented design? 60. Define design documentation. 61. Why is design documentation important?

Unit – III: Function-Oriented Software Design:

Overview of SA/SD methodology, structured analysis, data flow diagrams (DFDs), Structured Design, Detailed Design

Hours	(SLO)/Chapter	Taxonomy Level	General Objective	Learning Objectives	Pedagogy	Question Bank
1	SA/SD Methodology	Remember	To define foundational concepts	1. To define Structured Analysis. 2. To define Structured Design. 3. To define SA/SD methodology. 4. To list phases of SA/SD methodology.	PPT Lecture	1. What is Structured Analysis? 2. Define Structured Design. 3. What is SA/SD methodology? 4. List the phases of SA/SD methodology.
2	Purpose and Modeling	Understand	To understand the application and modeling of SA/SD	5. To explain the purpose of Structured Analysis. 6. To explain the purpose of Structured Design. 7. To define top-down decomposition. 8. To distinguish Structured Analysis from Structured Design. 9. To list benefits of SA/SD methodology. 10. To define functional decomposition. 11. To explain modelling of system functions.	PPT Lecture	5. Why use Structured Analysis? 6. Why use Structured Design? 7. What is top-down decomposition? 8. How do SA and SD differ? 9. List the benefits of SA/SD. 10. Define functional decomposition. 11. How is system function modeled?
3	Context & DFDs	Remember	To define and differentiate data flow models	12. To define Context Diagram. 13. To explain the scope of a system using Context Diagram. 14. To define Level 0 Data Flow Diagram (DFD). 15. To explain major system functions in Level 0 DFD. 16. To define Level 1 DFD.	PPT Lecture & Activity Based Learning	12. What is a Context Diagram? 13. How does it define scope? 14. Define Level 0 DFD. 15. What are the major functions in Level 0 DFD? 16. Define Level 1 DFD. 17. Explain sub-functions in Level 1 DFD.

Hours	(SLO)/Chapter	Taxonomy Level	General Objective	Learning Objectives	Pedagogy	Question Bank
				17. To explain sub-functions in Level 1 DFD. 18. To define functional primitives in DFDs.		18. What are functional primitives?
4	Data Dictionary	Remember	To define and describe a data dictionary	19. To define Data Dictionary. 20. To list elements of a Data Dictionary. 21. To explain the importance of Data Dictionary for consistency.	PPT Lecture & Demonstration	19. What is a Data Dictionary? 20. List elements of a Data Dictionary. 21. Why is it important for consistency?
5	DFD Symbols	Remember	To identify DFD elements and symbols	22. To define Data Flow Diagram (DFD). 23. To list symbols used in DFDs. 24. To describe the Process symbol in DFD. 25. To describe the Data Flow symbol in DFD. 26. To describe the Data Store symbol in DFD. 27. To describe the External Entity symbol in DFD.	PPT Lecture & Illustration	22. What is a DFD? 23. What symbols are used in DFDs? 24. What is a Process symbol? 25. Describe a Data Flow symbol. 26. What is the symbol for a Data Store? 27. What symbol represents an external entity?
6	Hierarchy & Levels of DFD	Understand	To explain hierarchical DFD development	28. To explain the importance of consistent notation in DFDs. 29. To explain levels of DFDs. 30. To explain hierarchy of DFDs. 31. To distinguish Context Diagram and Level 0 DFD. 32. To explain balancing rules for DFDs. 33. To explain consistency checking between DFD levels.	PPT Lecture & Group Discussion	28. Why is consistent notation essential? 29. What are the levels of DFDs? 30. Explain DFD hierarchy. 31. Differentiate Context Diagram vs Level 0 DFD. 32. What are balancing rules? 33. How is consistency verified?

Hours	(SLO)/Chapter	Taxonomy Level	General Objective	Learning Objectives	Pedagogy	Question Bank
7	DFD Applications	Understand	To use DFDs in software development	34. To identify common mistakes in DFD construction. 35. To explain how DFDs help in identifying system interfaces. 36. To explain how DFDs support user-developer communication.	PPT Lecture & Activity	34. What are common DFD mistakes? 35. How do DFDs help with interfaces? 36. How do DFDs help communication?
8	Structure Charts	Remember	To define and use structure charts	37. To define Structured Design. 38. To explain the transformation of DFDs to Structure Charts. 39. To define Structure Chart. 40. To describe the purpose of Structure Chart in module design.	PPT Lecture	37. What is Structured Design? 38. How are DFDs transformed to structure charts? 39. Define Structure Chart. 40. Why use Structure Charts in design?
9	Cohesion and Coupling	Understand	To understand modular quality	41. To define module cohesion. 42. To list types of module cohesion. 43. To define module coupling. 44. To list types of module coupling. 45. To define fan-in. 46. To define fan-out. 47. To explain the importance of high cohesion. 48. To explain the importance of low coupling.	PPT Lecture & Quiz	41. What is cohesion? 42. List types of cohesion. 43. Define coupling. 44. List types of coupling. 45. What is fan-in? 46. What is fan-out? 47. Why is high cohesion needed? 48. Why should coupling be low?
10	Module Design	Understand	To describe modular implementation	49. To define module specification. 50. To explain modular implementation. 51. To describe the relationship between modules and processes.	PPT Lecture & Case Study	49. Define module specification. 50. Explain modular implementation. 51. How do modules relate to processes?

Hours	(SLO)/Chapter	Taxonomy Level	General Objective	Learning Objectives	Pedagogy	Question Bank
11	Detailed Design	Understand	To explain transition from SD to DD	52. To define Detailed Design. 53. To explain the transition from Structured Design to Detailed Design. 54. To describe the role of algorithm design in Detailed Design.	PPT Lecture	52. What is Detailed Design? 53. How does SD transition to DD? 54. What role does algorithm design play?
12	Detailed Design	Understand	To explain selection of structures	55. To explain the importance of data structure selection in Detailed Design. 56. To explain the interface design between modules in Detailed Design.	PPT Lecture & Illustration	55. Why is data structure choice important? 56. What is interface design in DD?
13	Detailed Design	Remember	To list components of DD	57. To list elements of Detailed Design documentation. 58. To explain how Detailed Design ensures alignment with SRS. 59. To describe tools used in Detailed Design.	PPT Lecture & Demo	57. What are the elements in DD documentation? 58. How does DD align with SRS? 59. What tools are used in DD?
14	Detailed Design	Understand	To explain testing and quality in DD	60. To explain how Detailed Design supports coding. 61. To explain how Detailed Design supports testing. 62. To explain the importance of Detailed Design for system quality.	PPT Lecture & Group Discussion	60. How does DD support coding? 61. How does DD assist testing? 62. Why is DD important for software quality?

Unit – IV: Coding and Testing:

Coding; Code Review; Testing; Testing in the large vs Testing in the small; Unit Testing; Black-box Testing; White-box Testing; Debugging; Program Analysis Tools; Integration Testing; System Testing; General Issues Associated with Testing

Hours	(SLO)/Chapter	Taxonomy Level	General Objective	Learning Objectives	Pedagogy	Question Bank
1	Coding Fundamentals	Remember	To understand the nature of coding and good practices	1. To define coding. 2. To list characteristics of good code. 3. To describe coding standards. 4. To explain code readability. 5. To describe code understandability. 6. To describe code efficiency. 7. To explain code maintainability.	PPT Lecture & Code Review	1. What is meant by 'coding' in software engineering? 2. What are the main characteristics of good code? 3. What are standard coding practices in industry? 4. How does readability improve code quality? 5. What is the difference between understandability and readability in code? 6. How is code efficiency achieved? 7. Why is maintainability a critical factor in writing code?
2	Code Review	Understand	To review code quality through systematic methods	8. To define code review. 9. To explain the purpose of code review. 10. To describe the process of code review.	PPT Lecture & Demo + Peer Review	8. What is code review in the software development process? 9. Why is code review important before deployment? 10. What are the steps involved in performing a formal code review?
3	Software Testing – Concepts	Understand	To understand the goals of software testing	11. To define testing. 12. To explain the purpose of testing. 13. To define verification. 14. To define validation. 15. To distinguish verification from validation.	PPT Lecture	11. What is software testing? 12. What are the major goals of testing a software system? 13. How is 'verification' defined in the software life cycle? 14. What does 'validation' mean in software testing? 15. How do verification and

Hours	(SLO)/Chapter	Taxonomy Level	General Objective	Learning Objectives	Pedagogy	Question Bank
						validation differ with respect to their activities?
4	Unit Testing	Understand	To explain testing at the module level	16. To define unit testing. 17. To describe unit test cases.	PPT Lecture & Exercises	16. What is unit testing and where is it applied in SDLC? 17. What are the components and format of a unit test case?
5	Black-Box Testing	Understand	To apply testing based on functionality	18. To define black-box testing. 19. To describe the advantages of black-box testing. 20. To describe the limitations of black-box testing.	PPT Lecture & Problem Solving	18. What is black-box testing and how does it work? 19. What are the benefits of black-box testing in modular systems? 20. What limitations must developers consider when applying black-box testing?
6	White-Box Testing	Understand	To test code through structural logic	21. To define white-box testing. 22. To list techniques of white-box testing. 23. To describe statement coverage. 24. To describe branch coverage. 25. To describe path coverage.	PPT Lecture & Coding Exercise	21. What is white-box testing? 22. What are common techniques used in white-box testing? 23. How is statement coverage measured and applied? 24. What does branch coverage test in program logic? 25. What is the role of path coverage in code validation?
7	Debugging	Understand	To explain debugging and its strategies	26. To define debugging. 27. To explain the purpose of debugging. 28. To describe debugging process steps.	PPT Lecture & Coding Exercise	26. What is debugging in software engineering? 27. Why is debugging critical in the development phase? 28. What are the standard steps followed in a debugging session?

Hours	(SLO)/Chapter	Taxonomy Level	General Objective	Learning Objectives	Pedagogy	Question Bank
8	Program Analysis Tools	Remember	To understand tools that support debugging	29. To define program analysis tools. 30. To list types of program analysis tools.	PPT Lecture	29. What are program analysis tools? 30. What are different categories of program analysis tools used during software development?
9	Integration Testing	Understand	To evaluate module interaction through testing	31. To define integration testing. 32. To explain the purpose of integration testing.	PPT Lecture	31. What is integration testing? 32. What objectives are achieved through integration testing?
10	System Testing	Understand	To validate full system behavior	33. To define system testing. 34. To explain the purpose of system testing.	PPT Lecture	33. What is system testing? 34. What are the goals of system testing in large-scale projects?
11	Acceptance Testing	Understand	To verify software with the user/client	35. To define alpha testing. 36. To define beta testing. 37. To describe acceptance testing.	PPT Lecture & Group Discussion	35. What is alpha testing? 36. What is beta testing? 37. What is the purpose of acceptance testing in the final stage of development?
12	Test Case Design & Documentation	Understand	To prepare and document effective test plans	38. To define test case. 39. To describe test data. 40. To define test oracles. 41. To explain the purpose of test documentation.	PPT Lecture & Documentation Task	38. What is a test case? 39. How is test data generated and documented? 40. What is a test oracle and how is it used? 41. Why is test documentation critical in projects?
13	Testing Scope & Error Seeding	Understand	To differentiate testing scale and improve coverage	42. To describe testing in the small. 43. To describe testing in the large.	PPT Lecture	42. What is meant by testing in the small? 43. What is testing in the large? 44. How do small-scale and large-scale testing differ? 45. What is error seeding in test

Hours	(SLO)/Chapter	Taxonomy Level	General Objective	Learning Objectives	Pedagogy	Question Bank
				44. To distinguish testing in the small from testing in the large. 45. To define error seeding. 46. To explain the purpose of error seeding.		design? 46. Why do testers use error seeding techniques?
14	Test Coverage & Planning	Understand	To plan and execute effective test cycles	47. To describe test coverage metrics. 48. To explain test plan. 49. To explain test procedure. 50. To define test environment. 51. To describe test report.	PPT Lecture & Practical Walkthrough	47. What metrics are used to measure test coverage? 48. What should a comprehensive test plan include? 49. How is a test procedure different from a test plan? 50. What constitutes a proper test environment? 51. What is the format and use of a test report?
15	Test Execution & Issues	Understand	To describe test implementation and difficulties	52. To explain test case design. 53. To explain test case execution. 54. To describe general issues in testing. 55. To describe the importance of testing documentation. 56. To explain retesting. 57. To define regression testing.	PPT Lecture & Simulation	52. How do you design and structure test cases? 53. How is test case execution handled in live environments? 54. What are common issues encountered during testing? 55. Why is comprehensive documentation required? 56. What is retesting? 57. How is regression testing different from retesting?
16	Test Effectiveness, Best Practices & Challenges	Understand	To ensure quality through efficiency and industry practice	58. To describe the importance of regression testing.	PPT Lecture	58. Why is regression testing essential? 59. What does it mean for a test process to be effective?

Hours	(SLO)/Chapter	Taxonomy Level	General Objective	Learning Objectives	Pedagogy	Question Bank
				59. To explain testing effectiveness. 60. To describe testing efficiency. 61. To list best practices in testing.		60. How is testing efficiency calculated? 61. What are best practices recommended by industry?

Unit – V: Maintenance

Software Maintenance: Characteristic of software maintenance; software reverse engineering; software maintenance process models; estimation of maintenance cost.

Hours	(SLO)/Chapter	Taxonomy Level	General Objective	Learning Objectives	Pedagogy	Question Bank
1	Need for Software Maintenance	Remember / Understand	Define maintenance and its types	1. To define software maintenance. 2. To list types of software maintenance.	PPT Lecture & Group Discussion	1. What is software maintenance? 2. List its types.
2	Types of Software Maintenance	Understand	Understand corrective and adaptive types	3. To explain corrective maintenance. 4. To explain adaptive maintenance.	PPT Lecture & Case Study	3. What is corrective maintenance? 4. Define adaptive maintenance.
3	Types of Software Maintenance	Understand	Understand enhancement-oriented maintenance	5. To explain perfective maintenance. 6. To explain preventive maintenance. 7. To describe maintenance characteristics.	PPT Lecture & Group Exercise	5. Define perfective maintenance. 6. Define preventive maintenance. 7. What are characteristics of maintenance?
4	Concept of Reverse Engineering	Understand / Analyze	Understand reverse engineering concepts	8. To explain reverse engineering. 9. To define software re-engineering. 10. To distinguish reverse engineering and re-engineering.	PPT Lecture & Activity-Based Discovery	8. What is reverse engineering? 9. Define re-engineering. 10. Difference between reverse and re-engineering?
5	Reverse Engineering Process	Understand / Analyze	Study reverse engineering process	11. To explain the importance of reverse engineering.	PPT Lecture	11. Why is reverse engineering important?

Hours	(SLO)/Chapter	Taxonomy Level	General Objective	Learning Objectives	Pedagogy	Question Bank
				12. To describe the reverse engineering process.	& Flowchart-based Activity	12. Describe the reverse engineering process.
6	Software Maintenance Process Models	Understand	Understand process models in maintenance	13. To list maintenance process models. 14. To define the linear sequential model for maintenance. 15. To describe the iterative maintenance model.	PPT Lecture & Diagram Analysis	13. List maintenance process models. 14. What is the linear sequential model? 15. Describe the iterative maintenance model.
7	Change Request and Control Process	Understand / Apply	Handle change effectively	16. To explain impact analysis. 17. To define change request in maintenance. 18. To explain change control process.	PPT Lecture & Case Analysis	16. What is impact analysis? 17. Define change request. 18. Explain change control process.
8	Problem Analysis and Modification	Understand	Analyze and implement modifications	19. To describe problem analysis in maintenance. 20. To explain modification implementation.	PPT Lecture & Role-play / Scenario	19. What is problem analysis? 20. Explain modification implementation.
9	Regression Testing	Understand / Apply	Validate post-modification software	21. To define regression testing. 22. To explain the purpose of regression testing.	PPT Lecture & Test Simulation	21. Define regression testing. 22. Why is regression testing performed?
10	Maintenance Documentation	Understand	Understand role of documentation	23. To define maintenance documentation.	PPT Lecture	23. What is maintenance documentation?

Hours	(SLO)/Chapter	Taxonomy Level	General Objective	Learning Objectives	Pedagogy	Question Bank
				24. To explain the role of documentation in maintenance.	& Sample Document Review	24. Why is it important?
11	Effort Estimation and Cost Analysis	Analyze	Estimate maintenance costs	25. To define maintenance cost. 26. To list factors affecting maintenance cost. 27. To describe effort estimation techniques.	PPT Lecture & Cost Breakdown Task	25. Define maintenance cost. 26. What affects maintenance cost? 27. What are effort estimation techniques?
12	Maintainability and Configuration Management	Understand / Analyze	Understand maintainability and SCM	28. To define software maintainability. 29. To explain maintainability metrics. 30. To describe software configuration management (SCM). 31. To explain how SCM supports maintenance.	PPT Lecture & Diagram and Metrics Activity	28. Define maintainability. 29. List maintainability metrics. 30. What is SCM? 31. How does SCM support maintenance?
13	Challenges and Problems in Maintenance	Analyze	Address challenges and issues	32. To list maintenance challenges. 33. To describe common maintenance problems.	PPT Lecture & Group Discussion	32. What are maintenance challenges? 33. List common problems.
14	Cost Optimization and Software Aging	Analyze / Evaluate	Control cost and identify aging issues	34. To explain strategies to reduce maintenance cost. 35. To explain software aging. 36. To describe software decay.	PPT Lecture	34. How to reduce maintenance cost? 35. What is software aging?

Hours	(SLO)/Chapter	Taxonomy Level	General Objective	Learning Objectives	Pedagogy	Question Bank
						36. Describe software decay.
15	Software Obsolescence and Feedback	Analyze / Evaluate	Address long-term viability of systems	37. To define software obsolescence. 38. To explain documentation updates during maintenance. 39. To describe user feedback in maintenance.	PPT Lecture	37. Define software obsolescence. 38. Why update documentation? 39. How does user feedback help?
16	Maintenance Evolution and Best Practices	Analyze / Evaluate	Link productivity to quality and evolution	40. To explain the role of testing in maintenance. 41. To describe the relationship between maintenance and software evolution. 42. To explain maintenance productivity measurement. 43. To describe quality assurance in maintenance. 44. To explain impact of design on maintenance. 45. To list best practices in software maintenance.	PPT Lecture & Group Discussion	40. Role of testing in maintenance? 41. How is maintenance linked to evolution? 42. Define productivity measurement. 43. QA in maintenance? 44. Impact of design on maintenance? 45. Best practices?