Department of CSE
Details of Lesson Plan

S.No.	Particulars	Details
1.	Course Name	Software Testing and Automation
2.	Course Code	COM-702(B)
3.	Academic Year	2024-2025
4.	Semester	7th
5.	Number of Lesson plans	43
6.	Faculty Assigned	Parul Sharma

[illegible]

Lesson Plan No. 1.1	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Overview of Software Testing**
Objectives	At the end of the lesson, the student shall be able to: a) Provide an understanding of what software testing is and its necessity in software development. b) Explain the goals of software testing and the value it adds to the software development lifecycle.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube
Teaching Development	1. Introduction (5 minutes) a. Ask questions: What do you think software testing involves? Why is it important? 2. Defining Software Testing (10 minutes) a. Define software testing and its critical role in identifying and resolving defects before the software goes live. b. Discuss the impact of software testing on product quality and user satisfaction. 3. Goals of Software Testing (10 minutes) a. Explain the main goals such as finding defects, ensuring quality, verifying compliance with requirements, and assessing user satisfaction. 4. Value Added by Testing in Software Development (10 minutes) a. Illustrate how effective testing contributes to stability, security, and usability of software products. b. Case examples showing benefits of thorough testing in reducing maintenance costs and enhancing customer trust. 5. Summary (5 minutes) a. Recap the key points discussed. b. Ask questions to ensure understanding of the topic.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Reading books: a) Srinivasan Desikan, "Software Testing: Principles and Practices", Pearson Education, 1st (2005) b) Ali Mili, Fairouz Tchier, "Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: Write a brief paragraph explaining the importance of

	software testing and how it contributes to the success of software development. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** Asking open-ended questions on Worst-case analysis through nearpods **3.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 1.2	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Principles of Software Testing**
Objectives	At the end of the lesson, the student shall be able to: a. Discuss fundamental principles of software testing. b. Examine how these principles guide the testing process to ensure quality and reliability.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube
Teaching Development	1. **Introduction (5 minutes)** a. Ask questions: Why are principles important in software testing? Can you guess any fundamental testing principles? 2. **Exploration of Key Testing Principles (10 minutes)** a. Discuss each principle such as: o Testing shows the presence of defects o Exhaustive testing is impossible o Early testing o Defect clustering o Pesticide paradox o Testing is context-dependent o Absence-of-errors fallacy b. Provide real-world scenarios for each principle to show its application and relevance. 3. **Importance of Testing Principles (10 minutes)** a. Explain how these principles form the backbone of any testing strategy and help testers make informed decisions. 4. **Interactive Activity (10 minutes)** a. Small group activity: Assign each group a principle to create a short presentation on how it can be applied in a hypothetical testing project. 5. **Summary and Q&A (5 minutes)** a. Summarize the principles discussed and their importance. b. Open the floor for any questions related to the principles.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Reading books: a) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) b) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015)

	Home work: **Activity**: Choose one principle of software testing and describe a scenario where it would be crucial to apply this principle.Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**a.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **b.** Asking open-ended questions on Worst-case analysis through nearpods **c.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 1.3	**Course Name: Software Testing and Automation**	**Course No.: COM 702B**

Topics	**Testing Process and Its Phases**
Objectives	At the end of the lesson, the student shall be able to: a. Outline the different phases of the software testing process. b. Describe the activities involved in each phase and their importance in the overall testing strategy.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube
Teaching Development	1. **Introduction (5 minutes)** a. Start with a question: "Can anyone explain what they think the software testing process involves?" 2. **Overview of the Testing Process (10 minutes)** a. Present an overview of the testing lifecycle: o Requirement analysis o Planning o Test case development o Environment setup o Test execution o Defect reporting o Test closure 3. **Deep Dive into Each Phase (10 minutes)** a. Discuss each phase in detail: o Requirement Analysis: Understanding what needs to be tested and why. o Planning: Resources, tools, and schedule planning. o Test Case Development: Creating detailed test cases and scripts. o Environment Setup: Configuring the necessary hardware and software. o Test Execution: Running the tests and logging the results. o Defect Reporting: Documenting the defects for correction. o Test Closure: Ensuring all test cases are executed and all critical defects are fixed. 4. **Interactive Group Activity (5 minutes)** a. Break the class into groups and assign each a phase to develop a quick plan on how they would execute it for a given software project. 5. **Discussion and Q&A (10 minutes)** a. Groups present their plans and discuss any challenges they foresee. b. Open the floor to questions about the phases.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation

	from students on these. 2. Suggested Reading books: a) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) b) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: reate a simple flowchart that outlines the phases of the software testing process. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** Asking open-ended questions on Worst-case analysis through nearpods **3.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 1.4	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Software Development Life Cycle (SDLC) and Testing**
Objectives	At the end of the lesson, the student shall be able to: a. Illustrate where and how testing fits into the SDLC. b. Discuss different SDLC models and the role of testing in each.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube
Teaching Development	1. **Introduction (5 minutes)** a. Start with a question: "Can anyone explain what they think the software testing process involves?" 2. **Overview of the Testing Process (10 minutes)** a. Present an overview of the testing lifecycle: o Requirement analysis o Planning o Test case development o Environment setup o Test execution o Defect reporting o Test closure 3. **Deep Dive into Each Phase (10 minutes)** a. Discuss each phase in detail: o Requirement Analysis: Understanding what needs to be tested and why. o Planning: Resources, tools, and schedule planning. o Test Case Development: Creating detailed test cases and scripts. o Environment Setup: Configuring the necessary hardware and software. o Test Execution: Running the tests and logging the results. o Defect Reporting: Documenting the defects for correction. o Test Closure: Ensuring all test cases are executed and all critical defects are fixed. 4. **Interactive Group Activity (5 minutes)** a. Break the class into groups and assign each a phase to develop a quick plan on how they would execute it for a given software project. 5. **Discussion and Q&A (10 minutes)** a. Groups present their plans and discuss any challenges they foresee. b. Open the floor to questions about the phases.

Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Reading books: a) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) b) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: List the stages of the SDLC where testing is performed and explain why testing is important at each stage.Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	1. Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 2. Asking open-ended questions on Worst-case analysis through nearpods **3.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 1.5	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Types of Software Defects**
Objectives	At the end of the lesson, the student shall be able to: a. Define what software defects are and their impact on software performance. b. Classify different types of software defects and provide examples of each.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube
Teaching Development	**1. Introduction (5 minutes)** Begin with a question: "What do you think a software defect is, and why is it important to find them?" **2. Definition and Impact of Defects (10 minutes)** Define software defects and discuss their potential impact on software functionality, user experience, and overall system performance. **3. Classification of Software Defects (15 minutes)** a. Explain various types of defects: b. Functional Defects: Issues in the system that cause it to deviate from its functional requirements. c. Performance Defects: Problems that affect the application’s performance and efficiency. d. Usability Defects: Flaws that affect user interaction and satisfaction. e. Security Vulnerabilities: Weaknesses that could potentially be exploited by attackers. f. Integration Defects: Issues that occur when components or systems do not work well together. g. Provide real-world examples for each type of defect to illustrate their implications. **4. Summary and Question Activity (10 minutes)** Ask the class to identify a type of defect. Have them come up with an example of their assigned defect type, including potential causes and the impact on users.

Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Reading books: a) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) b) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity** Identify and write a brief description of a software defect you have encountered in any software application, classifying it based on the types discussed in class. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	1. Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 2. Asking open-ended questions on Worst-case analysis through nearpods **3.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 1.7	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Introduction to Test Automation**
Objectives	At the end of the lesson, the student shall be able to: a. Provide an overview of test automation and its role in software development. b. Highlight the importance of automated tools in improving efficiency, speed, and accuracy in software testing.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube
Teaching Development	Here's the structured outline of your session with bullet points and detailed formatting: 1. Introduction (5 minutes) - Begin with a question to engage participants: "What advantages do you think automation brings to software testing compared to manual testing?" 2. Definition and Advantages of Test Automation (10 minutes) - Define test automation and discuss its primary benefits, such as consistency, speed, and efficiency in executing repetitive test cases. 3. Common Scenarios for Test Automation (10 minutes) - Explain typical scenarios where test automation is highly beneficial: - Regression testing to ensure new changes don't break existing functionality. - Performance testing to measure system performance under load. - Continuous integration and deployment environments to enable frequent code integration and quick testing. 4. Overview of Popular Test Automation Tools (10 minutes) - Selenium: Discuss its use for automating web applications, support for multiple languages and browsers. - Appium: Cover its application in mobile testing for both iOS and Android platforms. - JUnit and TestNG: Explain their use in unit testing within Java projects. - Cucumber: Introduce Behavior-Driven Development (BDD) with Cucumber and its use in facilitating collaboration between technical and

	non-technical team members. 5. Discussion and Q&A (5 minutes) - Conclude the session with a Q&A round to revise and clarify the topics covered.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Reading books: a) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) b) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: List three advantages of test automation over manual testing with a real-world example for each. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	1. Reflective Questions (What, Why, Who?). Allow students to answer and discuss. 2. Asking open-ended questions on Worst-case analysis through nearpods **3.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 1.8	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Integration of Testing in SDLC**
Objectives	At the end of the lesson, the student shall be able to: a. Discuss how testing is integrated at various stages of the SDLC. b. Analyze the benefits of early and continuous testing in software projects.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube
Teaching Development	Here's your session outline with clear formatting and bullet points: 1. Introduction (5 minutes) - Start by asking: "Why is it important to integrate testing throughout the SDLC rather than just at the end?" 2. Testing in Various SDLC Models (10 minutes) - Discuss the role of testing in different SDLC models: - Waterfall: Testing in the later stages after development is complete. - Agile: Continuous testing throughout the lifecycle with each iteration. - DevOps: Emphasis on automated testing in continuous integration and deployment processes. 3. Benefits of Early Testing (10 minutes) - Explain the advantages of integrating testing early in the SDLC, such as identifying defects early, reducing the cost of defect fixes, and improving product quality. 4. Case Study Discussion (10 minutes) - Present a case study of a project that integrated testing at various stages of the SDLC. Discuss the outcomes and lessons learned. 5. Summary and Q&A (5 minutes) - Summarize the key points about the integration of testing in the SDLC. - Open the floor for questions to clarify any doubts and reinforce understanding.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Reading books:

	a. Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) b. Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: Discuss the potential impacts on a project if testing is only performed at the end of the development cycle, rather than integrated throughout the SDLC.. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** Asking open-ended questions on Worst-case analysis through nearpods **3.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 2.1	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	Introduction to Manual Testing Techniques
Objectives	At the end of the lesson, the student shall be able to: a. Introduce the concept of manual testing and its role in the software development lifecycle. b. Explain the importance of manual testing despite the increase in automation.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube
Teaching Development	• **Introduction to Manual Testing Techniques (5 minutes)** Begin the session by introducing the concept of manual testing and discussing its crucial role in the software development lifecycle. Highlight the relevance of manual testing even in an era where automated testing is prominent. • **Importance of Manual Testing (10 minutes)** Explain why manual testing is indispensable, emphasizing its ability to detect usability and user interface issues that automated tests might miss. Discuss the human element that manual testing brings, particularly in understanding the user experience and the intuitive use of the software. • **Overview of Manual Testing Processes (10 minutes)** Walk through the typical processes involved in manual testing, including test planning, test case development, test execution, and issue reporting. Explain how these processes are interlinked and how each contributes to the overarching goal of quality assurance. • **Challenges in Manual Testing (10 minutes)** Address common challenges faced in manual testing, such as the time-intensive nature of creating and executing tests, the potential for human error, and the difficulty in achieving complete coverage. Discuss strategies to mitigate these challenges, such as proper test planning and the integration of exploratory testing sessions. • **Q&A and Wrap-up (5 minutes)** Conclude the lecture by answering participant questions and summarizing the key points covered. Encourage participants to appreciate the value

	of manual testing as a complement to automated testing within the software testing spectrum.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Reading books: a) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) b) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: Write a short essay on the role of manual testing in the software development lifecycle and discuss why it remains relevant despite the rise of automated testing. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** Asking open-ended questions on Worst-case analysis through nearpods **3.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 2.2	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Test Case Design and Execution**
Objectives	At the end of the lesson, the student shall be able to: a) Discuss how to design test cases that effectively capture both functional and non-functional requirements. b) Teach methods for executing test cases and documenting the outcomes.
Teaching Aids (if any)	d. PPTs. e. Green board (Chalk and Talk). f. Video Lectures by NPTEL / Youtube
Teaching Development	• **Introduction to Test Case Design (5 minutes)** Start by defining what a test case is and its components, including the test steps, expected results, and test data. Emphasize the role of well-designed test cases in ensuring effective and efficient testing. • **Methods for Designing Test Cases (10 minutes)** Discuss various methods for designing test cases, focusing on capturing both functional and non-functional requirements. Explain the use of requirements traceability to ensure that all requirements are covered by test cases. • **Executing Test Cases (10 minutes)** Explore the execution phase of test cases, detailing how testers perform tests based on the test cases and record the results. Discuss the importance of environmental setup before test execution and how to handle test case failures. • **Documentation and Reporting (10 minutes)** Teach the importance of thorough documentation and reporting in manual testing. Discuss how to document test cases and results effectively for future reference and for communicating with other team members, including developers and project managers. • **Q&A and Practical Considerations (5 minutes)** End the session with a Q&A, allowing participants to clarify their doubts and discuss real-world applications of the concepts learned. Recap the importance of rigorous test case design and execution as the backbone of successful manual testing efforts. •

Closure	3. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 4. Suggested Reading books: c) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) d) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: Create a set of five test cases for a simple application (like a calculator or a to-do list app) that demonstrate an understanding of functional requirements and test execution steps. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**4.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **5.** Asking open-ended questions on Worst-case analysis through nearpods **6.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 2.3	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Test Coverage Essentials**
Objectives	At the end of the lesson, the student shall be able to: c) Explain the concept of test coverage and its importance in ensuring software quality. d) Explore methods to measure and improve test coverage in manual testing scenarios.
Teaching Aids (if any)	g. PPTs. h. Green board (Chalk and Talk). i. Video Lectures by NPTEL / Youtube
Teaching Development	• **Introduction to Test Coverage (5 minutes)** Start the lecture by explaining what test coverage is and why it's crucial for ensuring comprehensive software testing. Describe the concept of test coverage as a metric used to measure the effectiveness of the testing process in covering the code base and functionality. • **Different Types of Test Coverage (10 minutes)** Discuss various types of test coverage such as statement coverage, branch coverage, and path coverage. Explain how each type contributes to understanding the depth and breadth of testing efforts. • **Measuring Test Coverage (10 minutes)** Go into the specifics of how test coverage is measured, including the tools and techniques used to track coverage metrics. Describe common tools used in the industry for measuring coverage and how they integrate with testing environments. • **Improving Test Coverage (10 minutes)** Talk about strategies to improve test coverage, including the use of additional test cases, reviewing existing tests for gaps, and prioritizing high-risk areas of the application for increased scrutiny. • **Discussion and Wrap-Up (5 minutes)** Conclude with a discussion session, encouraging participants to share their experiences or challenges related to test coverage in their projects. Summarize the key points about the importance of test coverage and best practices for maximizing it.
Closure	5. Summarize the Lesson Learning Outcomes and get affirmation

	from students on these. 6. Suggested Reading books: e) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) f) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: Research and summarize different types of test coverage (e.g., statement, branch, path) and explain how each type contributes to software quality. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**7.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **8.** Asking open-ended questions on Worst-case analysis through nearpods **9.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 2.4	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	Black-box Testing Fundamentals
Objectives	At the end of the lesson, the student shall be able to: e) Introduce techniques such as equivalence partitioning and boundary value analysis. f) Explain how black-box testing can be applied to ensure system functionality without knowing the internal workings
Teaching Aids (if any)	j. PPTs. k. Green board (Chalk and Talk). l. Video Lectures by NPTEL / Youtube
Teaching Development	• **Introduction to Black-box Testing (5 minutes)** Begin with an introduction to black-box testing, defining it as a method where the tester does not have knowledge of the internal workings of the application. Highlight its relevance in testing the functionality of the software from an end-user's perspective. • **Black-box Testing Techniques (10 minutes)** Introduce specific techniques used in black-box testing, such as equivalence partitioning, boundary value analysis, and decision table testing. Explain each technique with examples to illustrate how they help in creating effective test cases. • **Applying Black-box Testing Techniques (10 minutes)** Discuss the application of these black-box testing techniques in real-world scenarios. Walk through how to apply these techniques to various types of software applications, emphasizing their role in uncovering a wide range of potential issues. • **Challenges in Black-box Testing (10 minutes)** Address common challenges encountered in black-box testing, such as the potential for missing critical test cases and difficulties in achieving complete coverage without understanding the internal code structure. • **Q&A and Practical Tips (5 minutes)** Finish the lecture by opening up for questions and providing practical tips for overcoming the challenges discussed. Encourage sharing of personal experiences and strategies among participants to enhance learning.

Closure	7. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 8. Suggested Reading books: g) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) h) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity** Develop three test scenarios using black-box testing techniques such as equivalence partitioning or boundary value analysis for a hypothetical user login feature. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**10.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **11.** Asking open-ended questions on Worst-case analysis through nearpods **12.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 2.5	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**White-box Testing Strategies**
Objectives	At the end of the lesson, the student shall be able to: g) Discuss techniques such as statement coverage and branch coverage. h) Teach how to apply white-box testing to evaluate the internal structures of the application.
Teaching Aids (if any)	m. PPTs. n. Green board (Chalk and Talk). o. Video Lectures by NPTEL / Youtube
Teaching Development	• **Introduction to White-box Testing (5 minutes)** Begin by defining white-box testing, emphasizing its focus on the internal logic and structure of the code. Explain its importance in verifying the internal aspects of the software that cannot be seen from the outside. • **Overview of White-box Testing Techniques (10 minutes)** Introduce key white-box testing techniques such as statement coverage, branch coverage, and condition coverage. Describe how each technique helps ensure that different parts of the codebase are tested thoroughly. • **Implementing White-box Testing (10 minutes)** Discuss practical steps for implementing white-box testing in a software project. Explain the use of automated tools that help in analyzing code and generating test cases that cover different paths. • **Advantages and Challenges (10 minutes)** Talk about the advantages of white-box testing, including the ability to identify hidden errors and optimize the code. Also, discuss the challenges, such as the need for testers to have a good understanding of the programming languages and complex algorithms used in the application. • **Discussion and Conclusion (5 minutes)** Conclude with a discussion on the integration of white-box testing into the overall test strategy. Invite questions from participants and provide a summary of key points to ensure clarity and retention of the discussed concepts.

Closure	9. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 10. Suggested Reading books: i) Srinivasan Desikan, “ Software Testing: Principles and Practices”, Pearson Education, 1st (2005) j) Ali Mili, Fairouz Tchier, “ Software Testing: Concepts and Operations”, Wiley, 3rd (2015) **Home work**: **Activity**: Select a simple piece of code (such as a function that calculates the sum of two numbers) and write down what tests would be necessary to achieve 100% statement coverage. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**13.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **14.** Asking open-ended questions on Worst-case analysis through nearpods **15.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 2.6	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**User Acceptance Testing (UAT)**
Objectives	At the end of the lesson, the student shall be able to: i) Explain the process of planning and conducting User Acceptance Testing. j) Discuss how UAT can serve as a final verification against business requirements.
Teaching Aids (if any)	p. PPTs. q. Green board (Chalk and Talk). r. Video Lectures by NPTEL / Youtube
Teaching Development	• **Recap of Basic Black-box Testing Techniques (5 minutes)** Quickly recap the basic techniques of black-box testing covered in previous lectures to set the foundation for more advanced techniques. • **Deep Dive into Decision Table Testing and State Transition Testing (15 minutes)** Explore decision table testing, explaining how it can be used to handle complex business rules and logic by mapping different input combinations to their expected outcomes. Then, introduce state transition testing, which is used to analyze the behavior of an application for different input sequences. • **Practical Application Examples (10 minutes)** Provide examples of how decision table testing and state transition testing can be applied to real-world scenarios. This could include case studies or hypothetical scenarios that highlight how these techniques can identify unique defects. • **Combining Techniques for Robust Testing (5 minutes)** Discuss how combining different black-box techniques can lead to more robust and comprehensive testing outcomes. Highlight the importance of selecting the right testing technique based on the application's complexity and the specific testing needs. • **Interactive Q&A (5 minutes)** End the session with an interactive Q&A, encouraging participants to ask questions about the techniques discussed. Provide further resources for study and practical application in their own projects.
Closure	11. Summarize the Lesson Learning Outcomes and get affirmation

	from students on these. 12. Suggested Reading books: k) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) l) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity** Create a decision table for a common user interface element, such as a form with multiple fields and conditions that determine the form's output or next steps. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**16.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **17.** Asking open-ended questions on Worst-case analysis through nearpods **18.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 2.7	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Usability Testing Basics**
Objectives	At the end of the lesson, the student shall be able to: k) Define usability testing and its significance in product development. l) Teach techniques for conducting usability testing and interpreting the results to improve product design.
Teaching Aids (if any)	s. PPTs. t. Green board (Chalk and Talk). u. Video Lectures by NPTEL / Youtube
Teaching Development	• **Introduction to User Acceptance Testing (UAT) (5 minutes)** Define User Acceptance Testing and explain its role in the development lifecycle. Discuss how UAT serves as a final verification phase to ensure the system meets business requirements and is ready for live deployment. • **Usability Testing Overview (10 minutes)** Shift focus to usability testing, detailing its purpose to evaluate how user-friendly, efficient, and satisfactory a software product is from the end user's perspective. Discuss various usability testing methods like heuristic evaluation, user testing, and A/B testing. • **Conducting UAT and Usability Tests (10 minutes)** Outline the steps involved in planning and conducting UAT and usability tests. Include the preparation of test scenarios, selection of appropriate end users, and setup of the testing environment. • **Challenges and Best Practices (10 minutes)** Identify common challenges in UAT and usability testing, such as participant selection bias, logistical issues, and interpreting subjective feedback. Provide best practices to overcome these challenges and achieve meaningful test results. • **Wrap-up and Discussion (5 minutes)** Conclude with a summary of key points and open the floor for discussion. Allow participants to share their experiences or hypothetical approaches to UAT and usability testing, fostering a collaborative learning environment.
Closure	13. Summarize the Lesson Learning Outcomes and get affirmation from students on these.

	14. Suggested Reading books: m) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) n) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: Draft a UAT plan for a recent software update or feature addition in a familiar application, including objectives, criteria for acceptance, and roles and responsibilities. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**19.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **20.** Asking open-ended questions on Worst-case analysis through nearpods **21.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 2.8	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Creating Your First Test Script with Selenium in Python**
Objectives	At the end of the lesson, the student shall be able to: m) Introduce Selenium WebDriver and its integration with Python. n) Guide students through the process of writing a simple script to automate a web login process.
Teaching Aids (if any)	v. PPTs. w. Green board (Chalk and Talk). x. Video Lectures by NPTEL / Youtube
Teaching Development	• **Introduction to Usability Testing (5 minutes)** Begin the lecture by defining usability testing and explaining its crucial role in evaluating how user-friendly, accessible, and efficient a software product is from the end user's perspective. Emphasize the goal of identifying potential areas for improvement to enhance user satisfaction. • **Key Components of Usability Testing (10 minutes)** Discuss the critical components that make up usability testing, including the identification of the target user demographic, the creation of representative user scenarios, and the selection of appropriate tasks for users to perform during testing sessions. • **Conducting Usability Testing (10 minutes)** Explain the steps involved in conducting effective usability testing, from recruiting participants and preparing test materials to executing the test sessions themselves. Highlight the importance of creating a controlled environment that mimics real-world usage as closely as possible. • **Analyzing Usability Test Results (10 minutes)** Cover the methods for analyzing data gathered from usability testing. Discuss how to interpret the results to draw meaningful conclusions about user behavior, satisfaction, and interaction patterns. Explain how these insights can be translated into actionable improvements in the product design. • **Q&A and Practical Implementation Tips (5 minutes)** Conclude the session by inviting questions from the participants, aiming to clarify any uncertainties regarding the usability testing process. Offer practical tips on implementing the insights gained from usability testing to make real-world

	improvements to software products.
Closure	15. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 16. Suggested Reading books: o) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) p) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: Design a simple usability test for any website or mobile app you frequently use. Outline the test's goals, the tasks participants would perform, and the questions you would ask them afterward to gather feedback. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**22.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **23.** Asking open-ended questions on Worst-case analysis through nearpods **24.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 3.1	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	Introduction to Test Automation
Objectives	At the end of the lesson, the student shall be able to: • Define test automation and explain its significance in modern software development. • Describe the benefits of test automation over manual testing processes.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube
Teaching Development	• **Introduction to Test Automation (5 minutes)** Start the session by explaining what test automation is and discuss its role in modern software development. Highlight the importance and benefits of test automation in improving the efficiency, accuracy, and coverage of software testing. • **Benefits of Test Automation (10 minutes)** Elaborate on specific benefits such as faster feedback cycles, increased test coverage, and improved accuracy. Share examples or case studies where test automation significantly improved the software development process. • **Challenges and Considerations (10 minutes)** Address common challenges faced when implementing test automation, such as initial setup costs, maintenance of test scripts, and the skill required to manage complex test suites. Discuss the strategic considerations necessary for successful test automation, including choosing the right tools and frameworks, planning for long-term maintenance, and ensuring team readiness and training. • **Overview of Automation Frameworks (10 minutes)** Provide an introduction to different types of test automation frameworks (linear, modular, data-driven, keyword-driven, hybrid). Explore the pros and cons of each framework type, assisting participants in understanding which might be suitable for their specific testing needs. • **Q&A and Wrap-up (5 minutes)** Conclude the session by opening the floor for questions to address any uncertainties or details regarding test automation. Summarize the key points

	discussed and provide a quick overview of what to expect in the next session on choosing test automation tools.
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Reading books: a) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) b) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: Write a brief essay on the importance of test automation and how it can improve the software development lifecycle. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** Asking open-ended questions on Worst-case analysis through nearpods **3.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 3.2	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Types of Test Automation Frameworks**
Objectives	At the end of the lesson, the student shall be able to: a) Introduce various automation frameworks such as linear, modular, data-driven, and keyword-driven. b) Discuss the pros and cons of each framework to help identify the most suitable for different testing needs.
Teaching Aids (if any)	d. PPTs. e. Green board (Chalk and Talk). f. Video Lectures by NPTEL / Youtube
Teaching Development	• **Introduction to Test Automation Frameworks (5 minutes)** Begin by briefly revisiting the concept of test automation and its importance. Define what a test automation framework is and the role it plays in organizing and executing tests effectively. • **Linear Scripting Framework (5 minutes)** Explain the concept of linear scripting, its simplicity, and when it is appropriate to use. Use a basic example of a linear script to illustrate the point. • **Modular Testing Framework (10 minutes)** Describe the modular testing approach where tests are broken down into separate scripts that can be reused. Provide a detailed example to show how modular design aids in maintenance and scalability. • **Data-Driven Framework (10 minutes)** Discuss the data-driven approach where test data is separated from the script logic, allowing for easy data iteration without script modifications. Show how to set up a simple data-driven test using Excel or CSV files. • **Keyword-Driven and Hybrid Frameworks (5 minutes)** Introduce keyword-driven testing and how it allows non-programmers to participate in writing automated test scripts using predefined keywords. Touch on hybrid frameworks that combine the strengths of the aforementioned frameworks to optimize testing efforts. • **Q&A and Summary (5 minutes)** Finish the session by addressing any remaining questions about the different

	frameworks. Summarize the session and discuss how to decide on a framework based on specific project requirements.
Closure	3. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 4. Suggested Reading books: c) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) d) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: Research and create a summary comparison table of different automation frameworks discussed (linear, modular, data-driven, keyword-driven). Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**4.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **5.** Asking open-ended questions on Worst-case analysis through nearpods **6.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 3.3	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Choosing the Right Test Automation Tools**
Objectives	At the end of the lesson, the student shall be able to: c) Provide criteria for selecting appropriate test automation tools based on project requirements. d) Review popular test automation tools and their key features.
Teaching Aids (if any)	g. PPTs. h. Green board (Chalk and Talk). i. Video Lectures by NPTEL / Youtube
Teaching Development	• **Introduction to Test Automation Tools (5 minutes)** Begin the session by explaining the critical role of choosing the right tools in the success of test automation efforts. Emphasize the diversity of tools available and the need for alignment with project requirements. • **Criteria for Tool Selection (10 minutes)** Discuss various criteria to consider when selecting a test automation tool, such as the technology stack of the application under test, the expertise of the team, budget constraints, and specific features offered by the tools. • **Review of Popular Test Automation Tools (15 minutes)** Provide an overview of some popular test automation tools, detailing their key features, strengths, and typical use cases. Include tools like Selenium for web applications, Appium for mobile apps, and TestComplete. Illustrate how each tool fits into different testing scenarios. • **Case Studies and Examples (5 minutes)** Share real-world case studies or examples where specific tools were selected based on unique project needs. Highlight the outcomes and lessons learned from these selections. • **Q&A and Wrap-up (5 minutes)** Allow time for participants to ask questions regarding test automation tools and their applications. Summarize the key points covered and hint at the upcoming session, which will focus on setting up an automation testing environment.
Closure	5. Summarize the Lesson Learning Outcomes and get affirmation

	from students on these. 6. Suggested Reading books: e) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) f) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: Choose any two test automation tools you find interesting, describe their features, and discuss why they would be suitable for a hypothetical project. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**7.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **8.** Asking open-ended questions on Worst-case analysis through nearpods **9.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 3.4	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Setting Up an Automation Testing Environment**
Objectives	At the end of the lesson, the student shall be able to: e) Guide on setting up the necessary environment for running automated tests. f) Discuss the integration of test automation tools with existing development and testing workflows.
Teaching Aids (if any)	j. PPTs. k. Green board (Chalk and Talk). l. Video Lectures by NPTEL / Youtube
Teaching Development	• **Introduction to Automation Testing Environments (5 minutes)** Open with a discussion on the importance of a properly configured testing environment for the success of automation efforts. Highlight the impact of the environment on test accuracy and repeatability. • **Components of a Test Environment (10 minutes)** Describe the essential components of an automation testing environment, including hardware, software, network configurations, and the test data setup. Discuss the role of each component in ensuring comprehensive testing coverage. • **Setting Up the Environment (15 minutes)** Walk through the process of setting up a typical automation testing environment. Discuss considerations for virtual versus physical environments, the use of containers or cloud platforms for scalability, and the integration of source control for test scripts. • **Best Practices and Common Pitfalls (5 minutes)** Outline best practices in setting up and maintaining test environments, such as regular updates and backups, clear documentation, and environment-specific configuration management. Address common pitfalls to avoid, like configuration drift and environment inconsistencies. • **Q&A and Practical Tips (5 minutes)** Conclude with a session for questions and provide practical tips and troubleshooting techniques for common issues faced when managing test environments. Recap the session's main points and set expectations for the next steps in learning about test automation scripting.

Closure	7. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 8. Suggested Reading books: g) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) h) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity** Create a checklist of necessary components and steps required to set up a test automation environment for a web application. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**10.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **11.** Asking open-ended questions on Worst-case analysis through nearpods **12.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 3.5	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	Introduction to Scripting in Test Automation
Objectives	At the end of the lesson, the student shall be able to: g) Explain the role of scripting in test automation and how it differs from general programming. h) Provide an overview of scripting languages commonly used in test automation.
Teaching Aids (if any)	m. PPTs. n. Green board (Chalk and Talk). o. Video Lectures by NPTEL / Youtube
Teaching Development	• **Introduction to the Role of Scripting in Test Automation (5 minutes)** Start by explaining the difference between scripting and programming with an emphasis on how scripting is used specifically in the context of test automation. Discuss why scripting is an essential skill for testers and the types of problems it solves in automation. • **Overview of Scripting Languages Used in Test Automation (10 minutes)** Discuss various scripting languages that are prevalent in the field of test automation, such as Python, JavaScript, and Ruby. Highlight the advantages and disadvantages of each, with a focus on their applicability to different types of automation tasks. • **Scripting Basics for Test Automation (10 minutes)** Go through the basic concepts of scripting necessary for writing effective automation scripts. Cover essential elements such as variables, control structures, functions, and error handling within the context of writing test scripts. • **Introduction to Common Scripting Frameworks (10 minutes)** Introduce participants to common frameworks and libraries that enhance scripting capabilities in test automation, like Selenium WebDriver for Python or Jest for JavaScript. Explain how these tools can be integrated with scripting languages to create more robust automation solutions. • **Q&A and Summary (5 minutes)** Finish the session by answering any questions from the participants. Summarize the key points covered, particularly emphasizing how scripting forms the backbone of most test automation strategies. Provide

	resources for further learning and practice in scripting for automation.
Closure	9. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 10. Suggested Reading books: i) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) j) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: Write a short script in any programming language you are familiar with that demonstrates basic conditional statements and loops. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**13.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **14.** Asking open-ended questions on Worst-case analysis through nearpods **15.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 3.6	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	Writing Effective Automation Scripts
Objectives	At the end of the lesson, the student shall be able to: i) Teach best practices for writing robust and maintainable test automation scripts. j) Demonstrate basic script writing that covers common functionalities in test automation.
Teaching Aids (if any)	p. PPTs. q. Green board (Chalk and Talk). r. Video Lectures by NPTEL / Youtube
Teaching Development	• **Introduction to Writing Effective Automation Scripts (5 minutes)** Begin with the importance of well-structured and maintainable scripts in test automation. Explain how good scripts reduce maintenance costs, improve reusability, and increase the reliability of automation efforts. • **Best Practices in Script Writing (10 minutes)** Discuss best practices for writing automation scripts, including coding standards, use of comments, maintaining readability, and error handling. Emphasize the importance of consistency and simplicity in script development. • **Structuring Scripts for Reusability and Maintainability (10 minutes)** Explain techniques for enhancing the reusability and maintainability of scripts, such as modularizing code, using external data sources (like CSV files or databases) for test data, and implementing object-oriented approaches where appropriate. • **Common Challenges in Script Writing and Solutions (10 minutes)** Address common challenges faced by automation testers when writing scripts, such as dealing with dynamic elements, synchronization issues, and cross-browser compatibility. Offer solutions and workarounds for these challenges, including examples of how to implement them in scripts. • **Q&A and Wrap-Up (5 minutes)** Allow time for a final Q&A session where participants can discuss issues they encounter in their scripting work. Recap the main lessons from today's lecture and discuss what they can expect in the upcoming

	sessions focused on advanced scripting and execution.
Closure	11. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 12. Suggested Reading books: k) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) l) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: Develop a simple script that could be used to automate a daily routine task (e.g., checking emails, logging into a particular website). Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**16.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **17.** Asking open-ended questions on Worst-case analysis through nearpods **18.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 3.7	**Course Name: Software Testing and Automation**	**Course No.: COM 702B**

Topics	**Introduction to Scripting for Test Automation**
Objectives	At the end of the lesson, the student shall be able to: k) Differentiate scripting for testing from programming languages. l) Introduce various types of scripting languages used in test automation, focusing on their applicability in different scenarios.
Teaching Aids (if any)	s. PPTs. t. Green board (Chalk and Talk). u. Video Lectures by NPTEL / Youtube
Teaching Development	• **Introduction to Scripting for Test Automation (5 minutes)** Start by differentiating scripting for testing from traditional programming languages. Highlight the specialized nature of scripting in the context of automation and its direct application to testing scenarios. • **Overview of Scripting Languages (10 minutes)** Discuss various scripting languages commonly used in test automation, such as Python, JavaScript, and Ruby. Highlight how each is suited to different aspects of automation, like Python for its simplicity and integration with tools like Selenium, or JavaScript for its use in web-based test environments. • **Where and When to Use Scripting in Test Automation (10 minutes)** Explain the typical scenarios where scripting is most beneficial in test automation, such as in repetitive tasks, complex test scenarios, or when integrating with Continuous Integration/Continuous Deployment (CI/CD) pipelines. • **Types of Automation Scripts (10 minutes)** Introduce different types of scripts used in test automation, such as setup scripts, teardown scripts, and utility scripts. Discuss their roles in a comprehensive test strategy and the importance of each within the test cycle. • **Q&A and Summary (5 minutes)** Conclude with a session for questions and answers, allowing participants to clarify doubts and deepen their understanding of the material covered. Summarize the key points and prepare them for the practical applications they will encounter in the next lecture.
Closure	13. Summarize the Lesson Learning Outcomes and get affirmation from students on these.

	14. Suggested Reading books: m) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) n) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: Identify a repetitive task you do on a computer and outline how you would automate this task using a script. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**19.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **20.** Asking open-ended questions on Worst-case analysis through nearpods **21.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 3.8	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Creating Your First Test Script with Selenium in Python**
Objectives	At the end of the lesson, the student shall be able to: m) Introduce Selenium WebDriver and its integration with Python. n) Guide students through the process of writing a simple script to automate a web login process.
Teaching Aids (if any)	v. PPTs. w. Green board (Chalk and Talk). x. Video Lectures by NPTEL / Youtube
Teaching Development	• **Introduction to Selenium WebDriver (5 minutes)** Introduce Selenium WebDriver as a powerful tool for automating web browsers. Explain how it can be used with Python to create flexible and powerful test scripts. • **Setting Up Selenium with Python (10 minutes)** Walk through the process of setting up Selenium with Python, including installing the necessary packages via pip and setting up the WebDriver for the desired browser. • **Writing a Basic Selenium Script in Python (10 minutes)** Guide participants through the process of writing a simple Selenium script in Python. The script will automate opening a web page, navigating to a login form, entering credentials, and submitting the form. • **Running the Script and Analyzing Output (10 minutes)** Demonstrate how to run the Selenium script from a Python environment. Show how Selenium interacts with the browser and explain how to interpret the results of the script, including handling potential errors. • **Q&A and Wrap-Up (5 minutes)** Allow time for a question and answer session where participants can ask specific questions about Selenium, Python, or scripting in general. Recap the session's main points and discuss the next steps for enhancing their scripts in future sessions.
Closure	15. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 16. Suggested Reading books: o) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005)

	p) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: Write a Python script using Selenium to open a browser and navigate to your favorite website, then close the browser. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**22.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **23.** Asking open-ended questions on Worst-case analysis through nearpods **24.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 3.9	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Enhancing Selenium Scripts**
Objectives	At the end of the lesson, the student shall be able to: o) Teach how to enhance Selenium scripts with more complex scenarios like handling dropdowns and checkboxes. p) Discuss how to incorporate assertions to verify test outcomes.
Teaching Aids (if any)	y. PPTs. z. Green board (Chalk and Talk). aa. Video Lectures by NPTEL / Youtube
Teaching Development	• **Review of Basic Selenium Scripting (5 minutes)** Begin by reviewing the basic Selenium script created in the previous session. Reiterate the key functions used, such as locating elements and interacting with them, to set the foundation for more advanced topics. • **Introduction to Advanced Selenium Techniques (10 minutes)** Introduce advanced Selenium techniques such as working with various web element locators (CSS selectors, XPath), handling dropdowns, checkboxes, and managing asynchronous operations with explicit and implicit waits. • **Enhancing Scripts with Assertions and Error Handling (10 minutes)** Discuss the importance of assertions in verifying test conditions and how to implement them in Selenium scripts. Cover basic error handling techniques to make scripts more robust and reliable, particularly in handling scenarios where elements are not found or do not behave as expected. • **Integrating Selenium Scripts with Test Suites (10 minutes)** Explain how individual Selenium scripts can be integrated into larger test suites. Discuss using testing frameworks like pytest or unittest in Python to organize tests, run them in batches, and generate reports. • **Q&A and Practical Tips (5 minutes)** Conclude with a session for questions and answers, providing practical tips on troubleshooting common issues in Selenium scripting. Summarize the enhancements discussed and their impact on the effectiveness and reliability of automation scripts.
Closure	17. Summarize the Lesson Learning Outcomes and get affirmation from students on these.

	18. Suggested Reading books: q) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) r) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: Enhance the script from the previous homework to include actions like filling out a form or navigating through multiple pages. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**25.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **26.** Asking open-ended questions on Worst-case analysis through nearpods **27.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 3.10	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Debugging and Troubleshooting Automation Scripts**
Objectives	At the end of the lesson, the student shall be able to: q) Provide strategies for debugging scripts when tests fail. r) Teach methods to optimize and troubleshoot scripts to improve reliability and performance.
Teaching Aids (if any)	bb. PPTs. cc. Green board (Chalk and Talk). dd. Video Lectures by NPTEL / Youtube
Teaching Development	• **Introduction to Debugging Automation Scripts (5 minutes)** Start with an overview of common issues that can arise when writing and running automation scripts, such as synchronization issues, incorrect locator strategies, or environmental inconsistencies. • **Tools and Techniques for Debugging (10 minutes)** Introduce tools and techniques for debugging scripts, including the use of logging, breakpoints, and inspection tools within IDEs. Highlight the use of Selenium's own debugging features to troubleshoot scripts effectively. • **Case Studies: Common Script Failures and Solutions (10 minutes)** Present several case studies that illustrate common script failures and discuss step-by-step how these issues were diagnosed and resolved. This real-world application helps participants understand the debugging process in context. • **Best Practices in Script Maintenance and Troubleshooting (10 minutes)** Cover best practices for maintaining automation scripts, such as regular code reviews, keeping scripts up to date with changes in the application, and effective version control strategies. Discuss strategies to anticipate and mitigate issues before they become significant obstacles. • **Q&A and Recap (5 minutes)** Finish the session with a final Q&A where participants can share their experiences and seek advice on specific challenges they have encountered. Recap the session by emphasizing the importance of systematic debugging and maintenance in the success of test automation efforts.

Closure	19. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 20. Suggested Reading books: s) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) t) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: Given a sample Selenium script with intentional errors, identify and correct the errors and explain what each correction fixes. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**28.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **29.** Asking open-ended questions on Worst-case analysis through nearpods **30.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 4.1	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Test Automation Tools and Frameworks**
Objectives	At the end of the lesson, the student shall be able to: a) Understand the concept of test automation and its role in modern software development. b) Explore the benefits and importance of using automated tools and frameworks for testing.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube
Teaching Development	- **Introduction (5 minutes)** - Ask questions: What is test automation. - What are its benefits? Why test automation is essential for improving software quality and development speed? - **Development (30 minutes)** - Define Test Automation. - Benefits of Test Automation: - Speed and Efficiency - Accuracy and Reliability - Reusability of Test Scripts - Early Bug Detection - Cost-Effectiveness - Define Test Automation Frameworks - Types of Test Automation Frameworks - Linear Scripting Framework - Modular Testing Framework - Data-Driven Framework - Keyword-Driven Framework - Hybrid Framework - Behavior-Driven Development (BDD) Framework - Popular Test Automation Tools - Designing a Test Automation Framework - **Key Components** test Automation Framework - Test Scripts - Test Data - Test Libraries - Test Execution Engine - Reporting - Best Practices for Designing Automation Frameworks - Continuous Integration and Test Automation -

	- **Exercise** (10 minutes) – - Have a discussion to summarize the lecture - Ask Questions Related to Topics
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Reading books: a) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) b) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: Discuss how Test automation tools and frameworks are essential for speeding up the testing process, increasing accuracy, and ensuring the quality of modern software applications. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** Asking open-ended questions on Worst-case analysis through nearpods **3.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 4.2	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Test Automation Tools and Frameworks**
Objectives	At the end of the lesson, the student shall be able to: a) Provide an overview of test automation and its role in software development. b) Highlight the importance of automated tools in improving efficiency, speed, and accuracy in software testing.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube
Teaching Development	- **Introduction (5 minutes)** - Ask questions: What is the benefits of using test automation in comparison to manual testing. - Name the common scenarios where test automation is used (e.g., regression testing, performance testing, and continuous integration). - **Development (30 minutes)** - Overview of Popular Test Automation Tools - Define Test Automation Tools. - Importance of Choosing the Right Tool - Different tools are suited for different types of testing (e.g., web, mobile, performance). - Factors to consider when choosing a tool: platform support, language compatibility, cost, ease of use, and integration with CI/CD pipelines. - Selenium: The Most Popular Web Automation Tool - Overview Selenium - Selenium is an open-source tool used to automate web applications across different browsers and platforms. - It supports multiple programming languages, including Java, Python, C#, Ruby, and JavaScript. - Appium: Mobile Application Automation Tool - Overview:Appium is an open-source tool used for automating mobile applications, including native, hybrid, and mobile web apps on both iOS and Android platforms. - JUnit and TestNG: Unit Testing Frameworks - **JUnit:**A widely used framework for writing and running unit tests in Java applications. It helps developers test small units of code like methods and classes.. - TestNG: A testing framework inspired by JUnit but designed to cover a wider range of test categories, including unit, functional, and integration testing.

	- Cucumber: Behavior-Driven Development (BDD) Tool - Overview: Cucumber is a BDD tool that uses plain language (Gherkin syntax) to define test cases, making it easier for non-technical stakeholders to understand and contribute. - Choosing the Right Tool for the Job (10 minutes) - Factors to Consider: - Application Type**.** - Programming Language - CI/CD Integration - Team Skillset - **Exercise** (10 minutes) – - Have a discussion to summarize the lecture - Ask Questions Related to Topics
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Reading books: a) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) b) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Activity**: discuss why test automation tools are crucial in today's fast-paced development environment to ensure software quality and reliability. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** Asking open-ended questions on Worst-case analysis through nearpods **3.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 4.3	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Selenium**
Objectives	At the end of the lesson, the student shall be able to: a) Understand what Selenium is and its importance in web automation. b) Learn about its key features and use cases in test automation. c) Role of Selenium in automating functional tests for web applications across various browsers and platforms. d) The advantages of Selenium over manual testing (speed, accuracy, repeatability).
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube
Teaching Development	- **Introduction (5 minutes)** - Ask questions: What is test automation? - Do you any test automation tool? - **Development (30 minutes)** - Selenium Ecosystem and Components - Overview of Selenium Components: - Selenium WebDriver - Selenium IDE - Selenium Grid - Comparison of Components - Setting Up Selenium WebDriver - **Steps to Setup Selenium with Java:** - Install the **Selenium WebDriver** library in your preferred IDE. - Download the appropriate browser driver (e.g., ChromeDriver). - Key Selenium WebDriver Commands - Selenium Integration with Test Frameworks - **Exercise** (10 minutes) – - Have a discussion to summarize the lecture - Ask Questions Related to Topics
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Reading books: a) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) b) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015)

	Home work: **Activity**: Discuss how selenium grid allows for executing tests in parallel across different browsers and machines. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** Asking open-ended questions on Worst-case analysis through nearpods **3.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 4.4	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	Appium
Objectives	At the end of the lesson, the student shall be able to: a) Understand what Appium is and its role in automating mobile applications. b) Learn about the types of mobile applications supported by Appium.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube
Teaching Development	- **Introduction (5 minutes)** - Ask questions: What is Appium? - Do you any test automation tool? - **Development (30 minutes)** - Appium Architecture - Client-Server Architecture: - Appium Client: Test scripts written in any language like Java, Python, C#, etc., using the WebDriver protocol. - Appium Server: Receives client requests and translates them into device-specific commands using mobile automation frameworks (e.g., UIAutomator2 for Android, XCUITest for iOS). - Appium's Cross-Platform Nature - Setting Up Appium - Prerequisites: - Install the Java Development Kit (JDK) and configure the environment variables. - Install the Appium Desktop application (provides a graphical interface for Appium Server) or run Appium via the command line. - Set up Android SDK and/or Xcode (for iOS development). - Install device emulators or connect physical devices. - Steps to Set Up Appium - Download and install Appium Desktop or use Appium Server from the command line. - Install the Appium Java client library in your IDE (e.g., Eclipse or IntelliJ). - Configure Android SDK and/or Xcode for emulators or devices. - Install platform-specific drivers (e.g., UIAutomator2 for Android). - Writing a Basic Appium Test for Android

	- Key Appium Commands and Interactions - Parallel Testing with Appium and Selenium Grid **(10 minutes)** - Advantages and Limitations of Appium (10 minutes) - **Exercise** (10 minutes) – - Have a discussion to summarize the lecture - Ask Questions Related to Topics
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Reading books: a) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) b) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: Justify why appium is a versatile and powerful tool for mobile app automation.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** Asking open-ended questions on Worst-case analysis through nearpods **3.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 4.5	Course Name: Software Testing and Automation	Course No.: 702B

Topics	**Creating automated test scripts,**
Objectives	At the end of the lesson, the student shall be able to: a) Understand the purpose and benefits of automated testing in software development. b) Learn the basics of creating and executing automated test scripts..
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube
Teaching Development	- **Introduction (5 minutes)** - Ask questions: What is benefits of automated testing software? - **Development (30 minutes)** - Steps in Creating Automated Test Scripts (10 minutes) - **1.** Identify Test Cases to Automate: - 2. Select an Automation Tool: - 3. Set Up the Testing Environment: - Writing a Basic Automated Test Script - **Test Scenario:** Automate the login functionality of a web application. - Steps: - Navigate to the login page. - Enter the username and password. - Click the login button. - Verify the login success message. - - **Key Concepts:** - WebDriver - Locators - Assertions - Using Test Frameworks for Organizing Test Scripts - Running Automated Test Scripts - Best Practices for Writing Automated Test Scripts - 1. Use Descriptive Test Names: - 2. Modularize Test Scripts - 3. Handle Dynamic Elements - 4. Avoid Hardcoding Test Data - 5. Maintain Clear Reporting and Logging - 6. Regularly Update Tests - **Exercise** (10 minutes) – - Have a discussion to summarize the lecture - Ask Questions Related to Topics

Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Reading books: a) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) b) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: Create an automated test scripts.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** Asking open-ended questions on Worst-case analysis through nearpods **3.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 4.6	**Course Name: Software Testing and Automation**	**Course No.: Com 702B**

Topics	**Executing automated test scripts**
Objectives	At the end of the lesson, the student shall be able to: a) Understand the process of executing automated test scripts. b) Learn about the tools and environments required for successful test script execution. c) Learn about various libraries and functions
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube
Teaching Development	- **Introduction (5 minutes)** - Ask questions: Have you executed any automated test scripts - **Development (30 minutes)** - Types of Test Execution - **1.** Local Test Execution - 2. Remote Test Execution - 3. Continuous Integration (CI) Execution - Setting Up Test Execution Environments - **1.** Local Test Execution - 2. Remote Test Execution - 3. Continuous Integration (CI) Execution - Running Automated Test Scripts - Integration of Test Execution with CI/CD - **Exercise** (10 minutes) – - Have a discussion to summarize the lecture - Ask Questions Related to Topics
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Reading books: a) Srinivasan Desikan, “ Software Testing: Principles and Practices”, Pearson Education, 1st (2005) b) Ali Mili, Fairouz Tchier, “ Software Testing: Concepts and Operations”, Wiley, 3rd (2015) **Home work**: **Activity**: Executing automated test scripts are a crucial step in ensuring the quality of applications through quick, repeatable, and reliable testing processes. Justify why.

Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** Asking open-ended questions on Worst-case analysis through nearpods **3.** MCQ / Sessional Test / Assignments
	Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 4.7	**Course Name: Software Testing and automation**	**Course No.: COM 702B**

Topics	**Test data management**
Objectives	At the end of the lesson, the student shall be able to: a) Understand the importance of test data in software testing and its role in ensuring test accuracy and efficiency. b) Define test data management (TDM) and its goals in the context of software development and testing.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube
Teaching Development	- **Introduction (5 minutes)** - Ask questions: what is test data management? - **Development (30 minutes)** - The Role of Test Data in Software Testing - Define Test Data - **Types of Test Data:** - Valid Data - Invalid Data - Boundary Data - Null/Empty Data - **Importance of Test Data** - Challenges in Test Data Management - Approaches to Test Data Management - 1. Manual Test Data Creation: - 2. Automated Test Data Generation: - 3. Synthetic Test Data: - 4. Production Data Masking - 5. Data Subsetting: - 6. Data Virtualization: - - Test Data Management Best Practices - Tools for Test Data Management - Test Data Management in Agile and DevOps Environments - **Exercise** (10 minutes) – - Have a discussion to summarize the lecture - Ask Questions Related to Topics
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Reading books: a) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) b) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley,

	3rd (2015) **Home work**: **Activity**: Test data management is a crucial aspect of ensuring high-quality testing processes. Justify why.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** Asking open-ended questions on Worst-case analysis through nearpods **3.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 4.8	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Test data automation**
Objectives	At the end of the lesson, the student shall be able to: a) Understand the concept of test data automation and its importance in the software testing lifecycle. b) Learn how automated test data generation and management contribute to effective test execution.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube
Teaching Development	- **Introduction (5 minutes)** - Ask questions: what is test data automation? - **Development (30 minutes)** - The Role of Test Data in Automation - Define Test Data Automation - Key Aspects of Test Data Automation (15 minutes) - **1.** Automated Test Data Generation - 2. Data Masking and Obfuscation - 3. Synthetic Data Generation - 4. Data Subsetting and Cloning - Automation Tools for Test Data Management - 1. CA Test Data Manager. - 2. Delphix - 3. Mockaroo - Integrating Test Data Automation into CI/CD Pipelines - 1. Continuous Integration and Test Data - 2. Data Provisioning in CI/CD - 3. Example CI/CD Integration - Best Practices for Test Data Automation - 1**.** Use Dynamic Test Data - 2. Automate Data Cleanup - 3. Maintain Data Privacy - 4. Version Control for Test Data - Challenges in Test Data Automation - 1. Complexity of Data Sources - 2. Managing Large Data Sets - 3. Data Integrity - **Exercise** (10 minutes) – - Have a discussion to summarize the lecture - Ask Questions Related to Topics
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these.

	2. Suggested Reading books: a) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) b) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: Discuss how test data automation simplifies and accelerates the process of creating and managing data for testing, enabling more efficient testing cycles and better test coverage.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** Asking open-ended questions on Worst-case analysis through nearpods **3.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 4.9	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Test data automation**
Objectives	At the end of the lesson, the student shall be able to: a) Understand the concept of continuous integration (CI) and its importance in modern software development. b) Explore how CI contributes to faster, more reliable software delivery.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube
Teaching Development	- Introduction (5 minutes) - Ask questions: what is continuous integration? - Development (30 minutes) - Key Principles of Continuous Integration - 1. Frequent Code Integration - 2. Automated Builds and Tests - 3. Fast Feedback - 4. Single Source Code Repository - 5. Fix Broken Builds Immediately - Benefits of Continuous Integration (10 minutes) - 1. Early Detection of Bugs - 2. Reduced Integration Risk - 3. Improved Software Quality - 4. Faster Delivery - 5. Increased Team Collaboration - CI Workflow and Process (15 minutes) - 1. Code Commit - 2. Automated Build - 3. Automated Tests - 4. Code Quality Checks - 5. Deployment (Optional) - 6. Feedback - Popular CI Tools (10 minutes) - 1. Jenkins - 2. GitLab CI - 3. CircleCI - 4. Travis CI - Steps to Set Up a Continuous Integration Pipeline - Best Practices for Continuous Integration - **Exercise (10 minutes) –** - Have a discussion to summarize the lecture - Ask Questions Related to Topics

Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Reading books: a) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) b) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: o **Activity**: Discuss how by following CI best practices and using the right tools, teams can improve collaboration, detect bugs early, and reduce the risks associated with integrating code changes.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** Asking open-ended questions on Worst-case analysis through nearpods **3.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 4.9	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Test data automation**
Objectives	At the end of the lesson, the student shall be able to: a) Understand what test automation is and its role in modern software development. b) Explore how automating tests improves testing efficiency and reliability.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube
Teaching Development	- **Introduction (5 minutes)** - Ask questions: what is test automation? - **Development (30 minutes)** - Key Components of Test Automation (10 minutes) - 1. Test Scripts - 2. Automation Tools - 3. Test Frameworks - Types of Tests to Automate - 1. Unit Tests - 2. Functional/End-to-End Tests - 3. Regression Tests - 4. Performance and Load Tests - Automation Testing Tools Overview - 1. Selenium (Web Automation) - 2. Appium (Mobile Automation) - 3. JUnit and TestNG (Unit Testing) - 4. Cucumber (Behavior-Driven Development - BDD) - 5. Apache JMeter (Performance Testing) - Setting Up an Automation Framework - 1. Framework Components: - Test Scripts: The core of the framework, written to automate specific tests. - Test Data: External data sources (e.g., CSV, Excel) that are used for data-driven testing to execute tests with multiple data sets. - Test Libraries: Reusable functions that can be shared across multiple test scripts. - Test Reports: Automated reporting mechanisms to generate results, logs, and screenshots of test execution. - 2. Page Object Model (POM) - 3. Data-Driven Testing - 4. Integration with CI/CD Tools - Running Automated Test Scripts

	- **Exercise (10 minutes) –** - Have a discussion to summarize the lecture - Ask Questions Related to Topics
Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Reading books: a) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) b) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: Justify that test automation is a critical component in modern software development, providing faster feedback, greater test coverage, and more reliable testing processes.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** Asking open-ended questions on Worst-case analysis through nearpods **3.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 5.1	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Performance Testing Fundamentals**
Objectives	At the end of the lesson, the student shall be able to: • Introduce the principles of performance testing, including the types of performance tests commonly conducted (e.g., stress, load, spike). • Explain how to design performance tests that accurately simulate user behavior and environmental conditions.
Teaching Aids (if any)	a. PPTs. b. Green board (Chalk and Talk). c. Video Lectures by NPTEL / Youtube
Teaching Development	• **Introduction to Performance Testing (5 minutes)** Begin by defining performance testing and its critical role in ensuring software applications can handle their intended load efficiently. Highlight the difference between performance testing and other types of testing. • **Types of Performance Tests (10 minutes)** Discuss different types of performance tests including stress testing, load testing, and spike testing. Describe scenarios where each type is applicable and the expected outcomes from these tests. • **Designing Performance Tests (10 minutes)** Explain how to design performance tests that accurately simulate user behavior and environmental variables. Discuss the importance of realistic test scenarios to ensure the results are relevant to real-world usage. • **Executing Performance Tests (10 minutes)** Detail the steps involved in setting up and executing a performance test. Emphasize the use of specific tools and software that are commonly used in the industry. • **Analyzing Test Results (5 minutes)** Cover the basics of analyzing performance test results. Explain how to interpret data to find bottlenecks and areas of improvement. • **Conclusion and Discussion (5 minutes)** Conclude with a review of the key points discussed. Open the floor for any questions, allowing participants to discuss their thoughts or prior experiences with performance testing.

Closure	1. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 2. Suggested Reading books: a) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) b) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: Research and write a summary of the differences between stress testing, load testing, and spike testing, including real-world scenarios where each would be applicable. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**1.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **2.** Asking open-ended questions on Worst-case analysis through nearpods **3.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 5.2	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Load Testing Techniques**
Objectives	At the end of the lesson, the student shall be able to: a) Discuss the process of setting up and executing load tests to assess system behavior under expected peak load conditions. b) Teach how to analyze load testing results to identify bottlenecks and performance degradation.
Teaching Aids (if any)	d. PPTs. e. Green board (Chalk and Talk). f. Video Lectures by NPTEL / Youtube
Teaching Development	• **Introduction to Load Testing (5 minutes)** Start by explaining what load testing is and how it differs from other types of performance testing. Discuss its importance in validating the application's performance under expected user loads. • **Setting Up Load Tests (10 minutes)** Discuss the technical setup required for load testing, including the selection of tools and configuration of test environments. Explain the role of these tools in simulating multiple users accessing the application simultaneously. • **Executing Load Tests (10 minutes)** Walk through the process of executing a load test, from initiating the test to monitoring performance metrics. Stress the importance of continuous monitoring to adjust test parameters in real-time for optimal results. • **Analyzing Load Testing Data (10 minutes)** Explain how to analyze the data gathered during load testing. Discuss the identification of performance degradation, understanding throughput, and response times, and how these metrics influence decisions on performance improvements. • **Q&A and Best Practices (5 minutes)** End with a session to answer any queries from the participants. Provide best practices for conducting effective load tests, including tips on iterative testing and using test results to guide performance optimizations. • **Summary (5 minutes)** Wrap up by summarizing the importance of load testing in the software development lifecycle. Encourage participants to integrate load testing early

	in the development stages to detect and mitigate potential performance issues before deployment.
Closure	3. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 4. Suggested Reading books: c) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) d) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: Create a hypothetical load testing plan for a popular website, detailing the types of loads to simulate and the key performance indicators to measure. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**4.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **5.** Asking open-ended questions on Worst-case analysis through nearpods **6.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 5.3	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	Security Testing Overview
Objectives	At the end of the lesson, the student shall be able to: c) Define security testing and its importance in identifying vulnerabilities in software applications. d) Explore different types of security tests, including penetration testing and risk assessment.
Teaching Aids (if any)	g. PPTs. h. Green board (Chalk and Talk). i. Video Lectures by NPTEL / Youtube
Teaching Development	• **Introduction to Security Testing (5 minutes)** Start the session by defining security testing and its importance in the software development lifecycle. Explain how security testing helps identify vulnerabilities and ensures that the software is protected against potential attacks. • **Types of Security Tests (10 minutes)** Discuss various types of security tests such as penetration testing, security scanning, and risk assessments. Explain the specific objectives of each type of test and when they are most effectively employed in a project. • **Common Security Threats and Vulnerabilities (10 minutes)** Introduce common security threats like SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF). Explain the typical vulnerabilities that lead to these threats and demonstrate with examples how these vulnerabilities can be exploited. • **Security Testing Tools and Techniques (10 minutes)** Outline the tools and techniques commonly used in security testing. Include both commercial tools and open-source solutions, discussing how they fit into a comprehensive security testing strategy. • **Discussion and Real-World Applications (5 minutes)** Conclude the lecture with a discussion on how security testing has been crucial in preventing major security breaches in real-world applications. Invite participants to share any experiences or concerns regarding security testing in their work environments.

	• **Summary and Next Steps (5 minutes)** Wrap up the session by summarizing the key points covered and discussing the next steps for participants who wish to deepen their knowledge in security testing. Provide resources for further learning and certification in security testing.
Closure	5. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 6. Suggested Reading books: e) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) f) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**Identify three common security threats (e.g., SQL injection, XSS, CSRF) and describe preventive measures that could be implemented during the development phase to mitigate these risks. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**7.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **8.** Asking open-ended questions on Worst-case analysis through nearpods **9.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 5.4	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Vulnerability Assessment Procedures**
Objectives	At the end of the lesson, the student shall be able to: e) Outline the steps for conducting a comprehensive vulnerability assessment. f) Describe how to use tools and techniques to identify security weaknesses in software systems.
Teaching Aids (if any)	j. PPTs. k. Green board (Chalk and Talk). l. Video Lectures by NPTEL / Youtube
Teaching Development	• **Introduction to Vulnerability Assessments (5 minutes)** Define what a vulnerability assessment is and distinguish it from penetration testing. Highlight its role in identifying, quantifying, and prioritizing vulnerabilities in a system. • **Steps in Conducting a Vulnerability Assessment (10 minutes)** Describe the step-by-step process for conducting a vulnerability assessment, including preparation, scanning, analysis, and reporting. Emphasize the importance of a systematic approach to ensure comprehensive coverage of all potential vulnerabilities. • **Tools Used in Vulnerability Assessments (10 minutes)** Discuss various tools that are essential for performing effective vulnerability assessments, such as Nessus, OpenVAS, and Qualys. Explain the capabilities and strengths of each tool to help participants choose the right tool for their needs. • **Analyzing and Prioritizing Vulnerabilities (10 minutes)** Teach how to analyze the results from vulnerability scans and prioritize vulnerabilities based on their severity, exploitability, and impact on the organization. Discuss strategies for mitigating the highest priority vulnerabilities. • **Best Practices and Common Pitfalls (5 minutes)** Share best practices for conducting vulnerability assessments and discuss common pitfalls that can compromise the effectiveness of the assessments. • **Q&A and Conclusion (5 minutes)** Open the floor for questions, allowing participants to clarify doubts and discuss how vulnerability assessments can be integrated into their

	security practices. Summarize the key takeaways from the session and encourage ongoing learning in the field of cybersecurity.
Closure	7. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 8. Suggested Reading books: g) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) h) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity** Choose a software application you use regularly and list potential security vulnerabilities that might be present based on general knowledge about similar applications. Suggest methods for assessing these vulnerabilities. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**10.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **11.** Asking open-ended questions on Worst-case analysis through nearpods **12.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 5.5	**Course Name: Software Testing and Automation**	**Course No.: COM 702B**

Topics	**Mobile Application Testing Strategies**
Objectives	At the end of the lesson, the student shall be able to: g) Provide an overview of the challenges unique to mobile application testing, including device fragmentation and mobile-specific user interactions. h) Discuss methodologies for effective testing across various mobile platforms and devices.
Teaching Aids (if any)	m. PPTs. n. Green board (Chalk and Talk). o. Video Lectures by NPTEL / Youtube
Teaching Development	• **Introduction to Mobile Application Testing (5 minutes)** Start the session by discussing the unique challenges and importance of testing mobile applications, highlighting the differences from traditional desktop testing due to diverse operating systems, screen sizes, and user interaction methods. • **Key Aspects of Mobile Testing (10 minutes)** Delve into the key areas specific to mobile testing, such as touch interactions, device orientation, and location services. Discuss how to effectively test these functionalities across different devices and platforms. • **Tools and Environments for Mobile Testing (10 minutes)** Introduce tools and environments that are essential for mobile testing, including both simulators and real device testing. Discuss popular tools like Appium and TestComplete, explaining their use in automating mobile tests. • **Handling Device and OS Fragmentation (10 minutes)** Explain strategies for managing testing efforts across various device types and operating systems to ensure broad coverage. Discuss the importance of prioritizing devices based on market data and project requirements. • **Interactive Q&A and Wrap-Up (5 minutes)** Conclude with a question-and-answer session to address specific concerns or challenges participants may have about mobile testing. Summarize the session by reinforcing the need for comprehensive testing strategies tailored to the mobile environment.

Closure	9. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 10. Suggested Reading books: i) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) j) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**Write a comparative analysis of two mobile testing tools of your choice, focusing on features, ease of use, and the types of testing each tool is best suited for. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**13.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **14.** Asking open-ended questions on Worst-case analysis through nearpods **15.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 5.6	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	**Test Management Techniques**
Objectives	At the end of the lesson, the student shall be able to: i) Illustrate how to organize and manage test activities within a project, including planning, scheduling, and resource allocation. j) Explain the use of test management tools to track test cases, results, and metrics.
Teaching Aids (if any)	p. PPTs. q. Green board (Chalk and Talk). r. Video Lectures by NPTEL / Youtube
Teaching Development	• **Introduction to Test Management (5 minutes)** Begin by defining test management and its role in successful software development projects. Discuss how test management integrates with overall project management to enhance product quality and ensure deliverables meet expectations. • **Test Planning and Organization (10 minutes)** Cover the process of test planning, including defining objectives, scope, resources, and timelines. Discuss how to organize testing activities to align with development cycles and business goals. • **Using Test Management Tools (10 minutes)** Introduce participants to common test management tools like JIRA, Quality Center, and TestRail. Explain how these tools help in tracking test cases, managing test cycles, and reporting on test progress. • **Metrics and Reporting in Test Management (10 minutes)** Discuss the importance of metrics and reporting in test management. Teach how to interpret test data to provide actionable insights and make informed decisions about the testing process. • **Best Practices and Discussion (5 minutes)** Share best practices in test management, focusing on communication, documentation, and continuous improvement. Open the floor for a discussion to allow participants to share experiences or ask questions about implementing effective test management strategies.
Closure	11. Summarize the Lesson Learning Outcomes and get affirmation

	from students on these. 12. Suggested Reading books: k) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) l) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity** Create a test plan outline for a small software project, including sections on scope, objectives, resources, schedule, and risk management strategies. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**16.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **17.** Asking open-ended questions on Worst-case analysis through nearpods **18.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 5.7	Course Name: Software Testing and Automation	Course No.: COM 702B

Topics	Defect Tracking and Reporting
Objectives	At the end of the lesson, the student shall be able to: k) Discuss methods for tracking defects from discovery through to resolution. l) Teach best practices for documenting defects and communicating findings to development teams.
Teaching Aids (if any)	s. PPTs. t. Green board (Chalk and Talk). u. Video Lectures by NPTEL / Youtube
Teaching Development	• **Introduction to Defect Tracking (5 minutes)** Define defect tracking and discuss its critical role in identifying, isolating, and resolving defects in software projects. Explain how defect tracking contributes to the overall quality of the software. • **Setting Up a Defect Tracking System (10 minutes)** Describe the key components of a defect tracking system and how to set it up effectively. Discuss the use of various tools that facilitate defect tracking and management. • **Effective Defect Reporting Techniques (10 minutes)** Teach how to write effective defect reports that clearly communicate the issue, steps to reproduce, expected vs. actual results, and severity. Emphasize the importance of accuracy and clarity in reporting to facilitate quick resolutions. • **Analyzing Defect Trends and Metrics (10 minutes)** Explain how to analyze defect trends and use metrics to improve testing processes. Discuss how to use defect data to prioritize bug fixes and understand underlying causes of defects. • **Q&A and Best Practices (5 minutes)** End with a session for questions and share best practices in defect management, focusing on collaboration between testers and developers to efficiently resolve defects.
Closure	13. Summarize the Lesson Learning Outcomes and get affirmation from students on these. 14. Suggested Reading books: m) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) n) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015)

	Home work: • **Activity**: Choose a defect tracking tool (like JIRA or Bugzilla) and create a detailed guide on how to log a defect, including screenshots and explanations of each field to be completed. □ Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**19.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **20.** Asking open-ended questions on Worst-case analysis through nearpods **21.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents

Lesson Plan No. 5.8	**Course Name: Software Testing and Automation**	**Course No.: COM 702B**

Topics	**Integrating Testing Practices Across Development Cycles**
Objectives	At the end of the lesson, the student shall be able to: m) Explain how to integrate various testing practices effectively throughout the development lifecycle for maximum impact. n) Explore strategies for maintaining testing relevance and effectiveness in agile and continuous delivery environments.
Teaching Aids (if any)	v. PPTs. w. Green board (Chalk and Talk). x. Video Lectures by NPTEL / Youtube
Teaching Development	• **Overview of Testing in Development Cycles (5 minutes)** Begin with an overview of how testing integrates into different development methodologies, such as Agile, Waterfall, and DevOps. Highlight the importance of adapting testing strategies to fit the development approach. • **Testing in Agile Environments (10 minutes)** Discuss specific strategies for integrating testing into Agile development cycles, including continuous integration, test-driven development (TDD), and behavior-driven development (BDD). • **Testing in Continuous Delivery Models (10 minutes)** Explain the role of testing in continuous delivery models. Discuss how to set up automated pipelines that include robust testing stages to ensure that every release is ready for production. • **Challenges and Solutions (10 minutes)** Address common challenges in integrating testing across different development cycles and provide practical solutions to overcome these challenges. Discuss the importance of communication and collaboration among cross-functional teams. • **Wrap-Up and Interactive Discussion (5 minutes)** Conclude the session with a summary of key points and an interactive discussion where participants can share their experiences or strategies for integrating testing practices effectively in their development environments.
Closure	15. Summarize the Lesson Learning Outcomes and get affirmation from students on these.

	16. Suggested Reading books: o) Srinivasan Desikan, " Software Testing: Principles and Practices", Pearson Education, 1st (2005) p) Ali Mili, Fairouz Tchier, " Software Testing: Concepts and Operations", Wiley, 3rd (2015) **Home work**: **Activity**: Prepare a short report discussing the benefits and challenges of implementing continuous testing in a DevOps environment. Include recommendations for smooth integration and examples of tools that support continuous testing. Spend 5 minutes to wrap up and consolidate the leanings.
Evaluation	**22.** Reflective Questions (What, Why, Who?). Allow students to answer and discuss. **23.** Asking open-ended questions on Worst-case analysis through nearpods **24.** MCQ / Sessional Test / Assignments Spend 5 minutes to evaluate student assimilation of the lesson contents