Selenium Testing
(Manual & Automation)

Introduction to Software Testing

• - Ensures software quality

• - Detects bugs and defects

• - Improves reliability and performance

• - Divided into **Manual Testing** and **Automation Testing**

What is Manual Testing?

- - Process of manually executing test cases
- - Requires human effort without automation tools
- - Used for exploratory, usability, and ad-hoc testing
- - Essential for initial application testing

Advantages & Disadvantages of Manual Testing

- **Advantages:**
- - Detects UI/UX issues
- - Suitable for small projects
- - No need for scripting knowledge

- **Disadvantages:**
- - Time-consuming
- - Prone to human errors
- - Not efficient for large projects

What is Automation Testing?

- - Uses scripts & tools to execute test cases
- - Reduces manual effort & increases efficiency
- - Ideal for regression testing and large applications
- - Selenium, JUnit, TestNG are commonly used tools

Advantages & Disadvantages of Automation Testing

- **Advantages:**
- - Faster execution of test cases
- - High accuracy & reusability
- - Cost-effective in long term

- **Disadvantages:**
- - Initial setup cost is high
- - Requires programming knowledge
- - Not suitable for UI/UX testing

Introduction to Selenium

- - Open-source automation testing framework
- - Supports multiple browsers (Chrome, Firefox, Edge)
- - Compatible with multiple programming languages (Java, Python, C#)
- - Used for web application testing

Selenium Components

- 1. **Selenium WebDriver**: Controls browser actions
- 2. **Selenium IDE**: Record & playback tool
- 3. **Selenium Grid**: Run tests on multiple machines in parallel

Selenium vs Other Automation Tools

- | Feature | Selenium | QTP/UFT | Cypress |
- |---------|----------|---------|---------|
- | License | Open-source | Paid | Open-source |
- | Platform | Web | Web & Desktop | Web |
- | Language Support | Java, Python, C# | VBScript | JavaScript |

Selenium WebDriver Basics

- - Automates browser interactions
- - Supports multiple programming languages
- - Interacts with web elements (buttons, forms, links)

Locators in Selenium

1. ID: driver.findElement(By.id("username"))

2. Name : driver.findElement(By.name("password"))`

3. Class Nam e :driver.findElement(By.className("login"))`

4. CSS Selector: `driver.findElement(By.cssSelector(".btn"))`

5. XPath: `driver.findElement(By.xpath("//button[text()='Submit']"))`

Selenium with TestNG & JUnit

- - TestNG and JUnit are testing frameworks
- - Used for organizing & executing test cases
- - Provides assertions, reports, and parallel execution

Handling Alerts, Frames, and Windows

- - **Alerts**: Accept or dismiss alerts using `driver.switchTo().alert()`
- - **Frames**: Switch between frames using `driver.switchTo().frame()`
- - **Windows**: Handle multiple windows using `driver.switchTo().window()`

Data-Driven Testing with Selenium

- - Uses external data sources (Excel, CSV, JSON)
- - Example: Read test data using Apache POI
- - Helps in testing multiple scenarios efficiently

Best Practices for Selenium Testing

- - Use explicit waits for stability
- - Organize test cases using frameworks (TestNG, JUnit)
- - Avoid hardcoded values
- - Maintain clear and reusable test scripts

Live Demo: Simple Selenium Script

- 1. Launch Chrome WebDriver
- 2. Open a website
- 3. Perform login test case
- 4. Verify test output
- 5. Generate test report