**MODULE – 5**

1. **What is Automation Testing?**

**Automation Testing** is a software testing technique that uses specialized tools and scripts to execute tests automatically, reducing the need for manual intervention. It involves validating that software performs as expected by automating repetitive and time-consuming tasks.

**Key Features of Automation Testing:**

1. **Tool-Driven**: Tools like Selenium, Appium, JUnit, and TestNG are commonly used to create and execute tests.
2. **Repeatability**: Test cases can be run multiple times without human effort, ensuring consistent results.
3. **Efficiency**: Speeds up the testing process, especially for regression, load, or performance testing.
4. **Accuracy**: Eliminates human errors associated with manual testing.
5. **Reusability**: Once written, automated test scripts can be reused for different versions of the software.

**Benefits:**

* **Saves Time and Effort**: Automates repetitive test cases, reducing manual labor.
* **Increases Test Coverage**: Enables testing a large number of test cases in less time.
* **Improves Accuracy**: Reduces the risk of human error.
* **Supports CI/CD Pipelines**: Integrates seamlessly into Continuous Integration and Continuous Deployment workflows.
* **Cost-Effective in the Long Run**: While the initial investment is higher, automation saves costs over time by reducing manual effort.

**Types of Tests Suitable for Automation:**

* Regression Testing
* Load Testing
* Performance Testing
* Functional Testing
* Smoke and Sanity Testing

**Common Tools for Automation Testing:**

* **Selenium**: For web application testing.
* **Appium**: For mobile application testing.
* **JMeter**: For performance testing.
* **Postman**: For API testing.
* **Cucumber**: For behavior-driven development (BDD).

**When to Use Automation Testing:**

* When tests need to be run repeatedly.
* When testing involves complex calculations or logic.
* When time-to-market is critical, and regression testing is extensive.
* When consistency and reliability of tests are crucial.

Automation testing is most effective when combined with manual testing to ensure comprehensive software quality assurance.

1. **Which Are The Browsers Supported By Selenium Ide?**

Selenium IDE (Integrated Development Environment) is supported by the following browsers:

1. **Google Chrome**  
   Selenium IDE is available as a Chrome extension and is fully supported on the Chrome browser.
2. **Mozilla Firefox**  
   Selenium IDE is also available as a Firefox add-on, supporting Firefox.

These two browsers are the primary ones supported by Selenium IDE as it functions as a browser extension for automated testing. Other browsers may not be directly supported by Selenium IDE, but the Selenium WebDriver framework can be used for broader cross-browser automation testing.

1. **What are the benefits of Automation Testing?**

**1. Increased Efficiency**

* Automation allows tests to be executed faster than manual testing, reducing the time needed for repetitive and regression testing.

**2. Improved Accuracy**

* Automated tests eliminate the possibility of human error in repetitive tasks, ensuring consistent and accurate results.

**3. Cost-Effective in the Long Run**

* While the initial setup may involve costs, automation testing reduces the need for repetitive manual testing, saving time and money over time.

**4. Enhanced Test Coverage**

* Automation allows for the execution of a large number of test cases across different scenarios, browsers, devices, and environments, ensuring comprehensive coverage.

**5. Reusability**

* Test scripts can be reused across multiple test cycles or projects, further reducing effort in creating new tests for similar functionalities.

**6. Support for Continuous Integration/Continuous Deployment (CI/CD)**

* Automated tests can be integrated into CI/CD pipelines, enabling rapid feedback and continuous delivery of software updates.

**7. Fast Feedback Cycle**

* Developers receive immediate feedback on their code, allowing for quicker identification and resolution of bugs.

**8. Improved Scalability**

* Automation tools can run multiple tests simultaneously on different configurations, making it easier to scale testing efforts.

**9. Support for Non-Functional Testing**

* Automation tools can be used for performance, load, stress, and scalability testing, which are challenging to perform manually.

**10. Ease of Reporting**

* Automation tools generate detailed logs and reports automatically, making it easier to analyze results and track issues.

1. **What are the advantages of Selenium?**

**1. Open Source and Free to Use**

* Selenium is open-source and free, making it accessible to individual developers, small teams, and large enterprises without licensing costs.

**2. Cross-Browser Support**

* Selenium supports multiple browsers, including:
  + Google Chrome
  + Mozilla Firefox
  + Safari
  + Microsoft Edge
  + Opera
  + Internet Explorer (legacy support)

**3. Cross-Platform Compatibility**

* Selenium supports testing on various operating systems such as:
  + Windows
  + macOS
  + Linux/Unix

**4. Multiple Programming Language Support**

* Selenium supports many programming languages for writing test scripts, including:
  + Java
  + Python
  + C#
  + Ruby
  + JavaScript
  + Kotlin
  + PHP

**5. Support for Multiple Testing Frameworks**

* Selenium can be easily integrated with testing frameworks like:
  + TestNG
  + JUnit
  + PyTest
  + NUnit

**6. Integration with CI/CD Tools**

* Selenium integrates seamlessly with Continuous Integration/Continuous Deployment tools such as:
  + Jenkins
  + Bamboo
  + CircleCI
  + Travis CI

**7. Wide Range of Testing Types**

* Selenium supports:
  + Functional Testing
  + Regression Testing
  + End-to-End Testing
  + Smoke and Sanity Testing

**8. Parallel Test Execution**

* Selenium Grid allows for the execution of multiple tests in parallel across different browsers, operating systems, and devices, saving time.

**9. Supports Mobile Testing**

* With tools like Appium and Selendroid, Selenium can also be extended to automate testing on mobile devices.

**10. Community Support**

* Selenium has a large and active community, providing extensive support, tutorials, plugins, and libraries.

1. **Why testersshould opt for Selenium and not QTP?**

Testers often choose Selenium over QuickTest Professional (QTP), now known as Micro Focus Unified Functional Testing (UFT), for various reasons. Here’s a detailed comparison highlighting why Selenium is often preferred:

**1. Cost**

* **Selenium**: Open-source and free to use, making it ideal for individual developers, startups, and organizations with limited budgets.
* **QTP/UFT**: A commercial tool with high licensing costs, making it less feasible for smaller teams or organizations.

**2. Programming Language Support**

* **Selenium**: Supports multiple programming languages like Java, Python, C#, Ruby, JavaScript, Kotlin, and PHP, giving testers flexibility to use their preferred language.
* **QTP/UFT**: Primarily supports VBScript, which limits flexibility for testers familiar with other languages.

**3. Browser Compatibility**

* **Selenium**: Supports almost all major browsers, including:
  + Google Chrome
  + Mozilla Firefox
  + Safari
  + Microsoft Edge
  + Opera
  + Legacy Internet Explorer
* **QTP/UFT**: Supports fewer browsers compared to Selenium and may require frequent updates for compatibility.

**4. Platform Compatibility**

* **Selenium**: Compatible with multiple operating systems, including Windows, macOS, Linux, and Unix.
* **QTP/UFT**: Primarily designed for Windows, with limited support for other platforms.

**5. Open Source and Extensibility**

* **Selenium**: Open-source, allowing for greater customization and integration with third-party tools.
* **QTP/UFT**: Proprietary software with limited extensibility and customization options.

**6. Test Execution**

* **Selenium**: Supports parallel and distributed test execution through Selenium Grid, significantly speeding up test cycles.
* **QTP/UFT**: Does not inherently support distributed or parallel execution without additional tools or configurations.

**7. Support for Modern Web Applications**

* **Selenium**: Supports automation of modern web applications, including single-page applications (SPAs), through advanced WebDriver APIs.
* **QTP/UFT**: Better suited for traditional client-server applications and may struggle with modern web technologies.

**8. Integration with CI/CD Tools**

* **Selenium**: Integrates seamlessly with CI/CD tools like Jenkins, CircleCI, Travis CI, Bamboo, and GitLab for continuous testing.
* **QTP/UFT**: While it can integrate with some CI/CD tools, the process is less straightforward compared to Selenium.

**9. Community Support**

* **Selenium**: Has a vast and active community, offering extensive resources, tutorials, and forums for support.
* **QTP/UFT**: Relatively smaller community due to its proprietary nature, with most support provided by the vendor.

**10. Flexibility and Scalability**

* **Selenium**: Highly flexible and can be scaled for complex automation scenarios using various plugins and libraries.
* **QTP/UFT**: More rigid and less adaptable to custom or complex testing needs.

**11. Mobile Testing**

* **Selenium**: Supports mobile application testing through tools like Appium, which integrates seamlessly with Selenium.
* **QTP/UFT**: Requires additional tools or modules (like UFT Mobile) for mobile testing, increasing costs.

**12. Ease of Learning**

* **Selenium**: Has a straightforward API and many learning resources, making it easier for new testers to pick up.
* **QTP/UFT**: Steeper learning curve, especially for testers not familiar with VBScript.

**13. Test Maintenance**

* **Selenium**: Provides better support for dynamic elements in modern web applications through robust locators and dynamic waits.
* **QTP/UFT**: Test maintenance can be cumbersome, especially when dealing with dynamic elements.