Module - 4

1.Which components have you used in Load Runner?

1. HP Load runner :

- is the most popular performance testing tool on the market today.
- This tool is capable of simulating hundreds of thousands of users, putting applications under real life loads to determine their behavior under expected loads.
- Load runner features a virtual user generator which simulates the actions of live human users.

2.HTTP Load :

- a throughput testing tool aimed at testing web servers by running several http or https fetches simultaneously to determine how a server handles the workload.

3. Proxy Sniffer:

- one of the leading tools used for load testing of web and application servers.
- It is a cloud based tool that's capable of simulating thousands of users.

2.How can you set the number of Vusers in Load Runner?

- In LoadRunner, you can set the number of virtual users (Vusers) during the Scenario Creation or Runtime Settings phase. Here are the main ways to set the number of Vusers:

1. **During Scenario Creation in the Controller:**

- When creating or configuring a scenario in the LoadRunner Controller, you can specify the number of Vusers for each script or group.
- Go to the **"Scenario"** tab.
- In the **"Vusers"** section, add the number of Vusers you want to run.
- You can distribute Vusers across different scripts or groups depending on your test plan.

2.In Runtime Settings:

- While running the scenario, you can also adjust Vusers in the **Runtime Settings** before starting the test.
- This is useful if you want to control the pacing or ramp-up of users.

3.Using LoadRunner VuGen:

- You can set the number of Vusers when running a script in the **VuGen** tool by configuring the run options, but typically the main control is through the Controller.

3.What is Correlation?

- correlation is the process of capturing dynamic data returned by a server (like session IDs or tokens) and using it in subsequent requests within a test script, particularly in performance and load testing.

- **Why is Correlation needed?**

  1.Dynamic values:
  Web applications often generate dynamic values (e.g., session IDs, unique tokens, timestamps) that are essential for maintaining user state and navigating the application

  2. Script Replay:
  If these dynamic values are not handled correctly, a recorded script might fail to replay accurately, as the server will likely reject requests using outdated or incorrect dynamic values.

  3.Load Testing:
  In load testing, multiple virtual users (VUs) simulate real user behavior. If each VU uses the same hardcoded dynamic value, it will lead to errors and inaccurate results.

4.What is the process for developing a Vuser Script?

1. Understanding the Application/Service

- Gather Requirements: Before you start creating scripts, it's essential to understand the business processes, user flows, and typical actions that users will take within the application.
- Identify Key Scenarios: Pick the most common and critical workflows that you need to test. These could include login, search, checkout, etc.
- Identify Load Parameters: Know the expected user load, the actions that users will perform, and how long each action might take.

2. Recording the Script

- **Use a Script Recording Tool**: Tools like LoadRunner, JMeter, or other performance testing tools allow you to record user actions as you interact with the application.

- **Perform the Action Manually**: While the tool is recording, manually perform the actions (e.g., login, browsing, form submission) that a typical user would do.

- **Save the Recorded Script**: After recording, the tool generates a script that represents the sequence of actions you performed.

3. Enhance the Script

- **Parameterization**: Replace hardcoded values (e.g., user credentials, search terms) with variables so that the script can use different inputs for each virtual user. This mimics real user behavior.

- **Correlation**: Dynamic data (e.g., session IDs, tokens) generated by the server must be captured and reused in subsequent requests. Correlation ensures that the script maintains state and can simulate realistic user flows.

- **Add Think Time**: To simulate realistic user interaction, add think time (pauses) between actions to mimic real-world delays that users experience when interacting with the application.

- **Add Validation**: Check for the correct response from the server after actions like logging in, submitting forms, etc., to ensure the script is functioning correctly.

4. Customization and Logic

- **Custom Logic**: Depending on the testing scenario, you may need to add custom logic to the script. This could include conditional loops, if-else statements, or logic to handle different paths based on the application's response.

- **Error Handling**: Implement error-handling mechanisms to deal with issues like timeouts, page not found errors, or server issues.

5. Test and Debug the Script

- **Run the Script in Debug Mode**: Before running the full load test, test the script using a smaller number of virtual users to make sure that it behaves as expected.

- **Fix Any Issues**: If you encounter errors, debug the script and fix any issues related to session management, dynamic data handling, or incorrect request/response handling.

6. Load Test Configuration

- **Configure Load Parameters**: Set up how many virtual users (Vusers) will execute the script, the ramp-up period, the duration of the test, and the load distribution (whether you want constant load or peak load).

- **Set Up Load Generators**: These are machines that simulate the virtual users. Ensure they are distributed correctly to simulate traffic from various locations if needed.

.7. Execute the Load Test

- **Run the Test**: Once everything is configured, execute the load test with the defined number of virtual users. Monitor system performance during this time.

- **Track Metrics**: Collect metrics such as response times, throughput, error rates, server CPU usage, memory, and database performance to understand how well the application performs under load.

8. Analyze and Report Results

- **Analyze the Results**: After the test, analyze the collected data to identify bottlenecks or performance issues. Look at key metrics like response time, server performance, and error rates.

- **Generate a Report**: Summarize the results of the test, including any findings, bottlenecks, and performance improvement recommendations.

5.How Load Runner interacts with the application?

1. LoadRunner Components Overview:

LoadRunner is made up of several components that work together to simulate virtual users and generate load on an application:

- Vuser Scripts: These scripts represent the actions performed by virtual users. They are created by recording user actions or writing custom scripts.
- Controller: The Controller orchestrates the execution of tests, including managing the Vusers, distributing load, and collecting performance data.
- Load Generators: These machines simulate the Vusers and generate the load. They are distributed across different machines to simulate different geographies and traffic conditions.
- Analysis: After the test is completed, LoadRunner's Analysis component is used to analyze performance metrics, identify bottlenecks, and produce detailed reports.

Step 1: Script Recording :

- Virtual User (Vuser) Script Creation: LoadRunner first records a user's interaction with the application via a protocol-based recording. During this phase, LoadRunner records the requests (HTTP, SOAP, Database queries, etc.) sent from the user to the server.
- The user action could be anything from logging in, navigating through different pages, filling out forms, or submitting requests to the server.
- Recording Tools: LoadRunner uses its VuGen (Virtual User Generator) to capture these actions and generate a script in a language such as C, JavaScript, or other supported protocols.

Step 2: Parameterization and Correlation:

- **Parameterization**: In order to simulate realistic behavior, the script will replace hardcoded data (like usernames, passwords, search terms) with dynamic variables. These variables are then populated with different values from a dataset during runtime.

- **Correlation**: When recording, LoadRunner captures dynamic data such as session IDs, tokens, or timestamps generated by the server, which are essential

for maintaining state across requests. The correlation step ensures that this dynamic data is handled correctly in the script and reused in subsequent transactions.

-

Step 3: Script Customization

- After the recording, you may need to modify or enhance the script. This could involve adding **think time** (delays between actions), custom **validation** checks (to verify correct application responses), and **error handling** (to deal with failed requests).

- **Think Time**: Think time simulates real users' response times between actions. For example, after clicking a button, a real user might wait a few seconds before proceeding.

Step 4: Test Execution (Controller) :

- **Test Plan Configuration**: In LoadRunner Controller, you configure the load test scenario, specifying the number of Vusers, the duration of the test, and other parameters like ramp-up period (how fast Vusers start).
- **Test Execution**: The Controller orchestrates the execution of the script, launching the defined number of Vusers (simulating real users), who start interacting with the application simultaneously.
- **Simulating Load**: Each Vuser runs the script from the **Load Generator** machine, sending requests to the server as if they were real users. The requests might involve interactions like opening pages, submitting forms, or making database queries.

Step 5: Monitoring the Application's Response

- **Performance Data Collection**: LoadRunner continuously monitors key metrics such as response times, error rates, throughput, and resource utilization (CPU, memory, network bandwidth) while the test is running.
- **Application Interaction**: During the test, LoadRunner sends the requests that represent real user actions (like navigating to a page or performing a transaction), and the application responds with the relevant data (e.g., HTML pages, database records).

Step 6: Data Analysis (Analysis Component):

- After the load test completes, LoadRunner's **Analysis** component is used to analyze and interpret the data collected during the test. It provides performance

reports, visualizations, and charts that help identify any bottlenecks, failures, or areas of improvement in the application's performance.

- The results might include information on response times for different transactions, throughput rates, error rates, resource utilization, and more.

Step 7: Report Generation:

- LoadRunner generates reports based on the performance data collected during the test. These reports highlight key metrics and allow you to draw conclusions about the application's ability to handle load.

6.How many VUsers are required for load testing?

1. Expected Traffic Volume

- How many concurrent users do you expect during peak usage? If you're testing for a scenario where hundreds or thousands of users may interact with the service at the same time, you will need to simulate that level of traffic using virtual users.

2. Purpose of the Load Test

- **Baseline Testing:** You might want to determine the performance of the system under normal usage conditions. This might involve simulating the expected number of users interacting with the system in typical conditions.

- **Stress Testing:** This aims to push the system beyond its capacity to identify how it breaks or where it fails. In this case, you could increase the number of VUsers well beyond the expected maximum usage, possibly simulating tens of thousands of users.

- **Scalability Testing:** To see how well the system scales when adding more VUsers incrementally, you could start from a baseline and gradually increase the number of VUsers until performance degrades or the system begins to fail.

3. Type of Interactions

- If users will perform intensive operations like generating large texts or performing

complex API calls (e.g., processing queries with ChatGPT), you might need fewer VUsers but with heavier load per VUser.

- On the other hand, if users are simply browsing or submitting smaller, less complex queries, the number of VUsers needed might be higher to simulate higher traffic.

4. Performance Metrics and Goals

- Determine what key performance indicators (KPIs) you're measuring: response times, throughput, error rates, etc.

- How many users do you need to simulate to reach those KPIs under load?

7.What is the relationship between Response Time and Throughput?

1.Response Time :

- **Definition:** Response time is the total duration a system takes to process a user request and deliver a response. It's the time between a user taking action (like clicking a button) and the application acknowledging the action is complete.
- **Impact**: Fast response times are crucial for a positive user experience. Delays can lead to frustration and potential user abandonment.
- **Factors Influencing Response Time:**
- Server processing power and efficiency.
- Network conditions like latency and bandwidth.
- Application and database efficiency (query speed, code optimization).
- Concurrency levels (number of users accessing the system).

2. Throughput:

- **Definition:** Throughput measures the volume of work a system can process and deliver within a specific timeframe. It's often measured in terms of transactions per second (TPS) or requests per second (RPS).
- **Impact:** High throughput indicates a system's capacity to handle a large volume

of operations efficiently. Low throughput can indicate performance bottlenecks and scalability issues.

- **Factors Influencing Throughput:**
- Server resources (CPU, memory, disk I/O).
- Database and backend efficiency (query optimization, indexing).
- Network bandwidth and infrastructure.
- Application architecture and design.
- Number of concurrent users or requests.

8.To test the Performance testing on "Tops Technologies website" :-https://www.saucedemo.com/

1. to Record all top level menu

- **What is a Top-Level Menu?**
  A "top-level menu" typically refers to the primary navigation bar or the main menu at the top of the website (e.g., Home, Products, About Us, etc.). These menu items are essential in simulating how users navigate the website.

- Recording the Interactions:
  Depending on the tool you're using, the procedure will slightly vary, but here's the general flow:

For LoadRunner:

- *Open LoadRunner and start a new Virtual User Generator (VuGen) script.*

- Choose Web - HTTP/HTML as the protocol.

- Start recording and navigate through the website's top-level menu (Home, Products, Cart, etc.).
- Stop the recording once you have captured all the relevant actions.
- Save the script.

For NeoLoad:

- Open NeoLoad and create a new Scenario.
- Add Virtual Users and set the desired load (10 VUsers).

- Record the HTTP requests by navigating through the top-level menu.
- Save the script once recording is done.

2. to Record minimum 10 Vuser on this website

- In the Thread Group (JMeter) or Virtual User Generator (LoadRunner/NeoLoad), you need to set the number of virtual users (VUsers) to 10. This simulates 10 concurrent users navigating through the site.
- In JMeter:
  - Go to your Thread Group and set the Number of Threads (Users) to 10.
- Set Ramp-Up Period to 10 seconds (or as required).
- Set Loop Count to 1 (or more, if you want to repeat actions).

In LoadRunner:

- Set Virtual Users to 10 and choose the appropriate Pacing and Think Time.

In NeoLoad:

- Set the Number of Virtual Users to 10 and adjust the Pacing and Duration as needed.

3. save all (Script,Design,Graph) :

- In JMeter: After recording, right-click the Test Plan and select Save As to save your script.

- In LoadRunner: After recording, save the VuGen script.

- In NeoLoad: After recording, save the Scenario script.

9. Create a normal script of the above website with correlate using hp default website.

1. Script Recording (using VuGen)

1. **Start VuGen (Virtual User Generator):**
   - Open VuGen (the scripting tool in LoadRunner).
   - Select the HTTP/HTML protocol for recording.
   - Click Record.

2. **Record Actions:**
   - In the browser window that opens during the recording, navigate to the HP Default Website (e.g., www.hp.com).
   - Perform basic actions like:
   - Login (if applicable, using sample credentials).
     - Browse to a product page (e.g., click on a product category).-
     - Search for a product (e.g., search for "laptops").

3.Stop the Recording:

- After performing the desired actions, stop the recording in VuGen.
- VuGen generates an HTTP script that simulates the actions performed during the session.

10.What is Automation Testing?

- Test automation is the use of software to control the execution of tests, the comparison of actual outcomes to predicted outcomes, the setting up of test preconditions, and other test control and test reporting functions.
- Commonly, test automation involves automating a manual process already in place that uses a formalized testing process.
- Although manual tests may find many defects in a software application, it is a laborious and time consuming process.
- In addition it may not be effective in finding certain classes of defects.
- Test automation is a process of writing a computer program to do testing that would otherwise need to be done manually.

- Once tests have been automated, they can be run quickly.
- This is often the most cost effective method for software products that have a long maintenance life, because even minor patches over the lifetime of the application can cause features to break which were working at an earlier point in time.

11.Which Are The Browsers Supported By Selenium Ide?

1.Firefox

2.Google Chrome

12.What are the benefits of Automation Testing?

- 70% faster than the manual testing
- Wider test coverage of application features
- Reliable in results
- Ensure Consistency
- Saves Time and Cost
- Improves accuracy
- Human Intervention is not required while execution
- Increases Efficiency
- Better speed in executing tests
- Re-usable test scripts
- Test Frequently and thoroughly
- More cycle of execution can be achieved through automation
- Early time to market

13.What are the advantages of Selenium?

- Open source, free to use, and free of charge.
- Highly extensible
- Can run tests across different browsers
- Supports various operating systems
- Supports mobile devices
- Can execute tests while the browser is minimized to be visible on the desktop.
- Can execute tests in parallel.

14.Why testers should opt for Selenium and not QTP?

1. Cost-Effectiveness

1. Selenium:

- Free and Open Source: Selenium is a completely free and open-source automation tool. This makes it highly cost-effective, especially for organizations that need a powerful automation framework without the associated licensing costs.
- No Licensing Fees: Selenium doesn't require any ongoing license renewals, and testers can use it without worrying about software costs or vendor-specific fees.

2. QTP/UFT:

- **Expensive Licensing**: QTP/UFT is a commercial product that requires the purchase of a license. The cost of licensing can be quite high, especially for organizations with large teams or those scaling their automation efforts.

2. Cross-Browser and Cross-Platform Support:

1. **Selenium**:
- **Cross-Browser Testing**: Selenium supports all major browsers (Chrome, Firefox, Safari, Edge, etc.), which means you can run tests across different browsers simultaneously. This is crucial for web applications where compatibility across browsers is a key requirement.
- **Cross-Platform Compatibility**: Selenium can run on multiple platforms, including Windows, Linux, and Mac OS, making it a versatile tool for global teams and diverse environments.
- **Mobile Testing**: Selenium can also be integrated with mobile testing frameworks such as Appium for testing mobile applications.

2.QTP/UFT:

- **Limited Browser Support**: While QTP supports multiple browsers, its compatibility with newer browsers is sometimes slower due to updates or lack of support for cutting-edge features in the browser market.
- **Limited OS Support**: QTP mainly runs on Windows and doesn't have the broad cross-platform support that Selenium offers.

3. Language Flexibility :

1.Selenium:

- **Multiple Language Support**: Selenium allows testers to write test scripts in a variety of programming languages, including Java, Python, Ruby, C#, JavaScript, and PHP. This flexibility makes it easier for testers to work in the language they are most familiar with, or the one used by their development team.
- **Integration with Popular IDEs**: Selenium can be used with popular IDEs like Eclipse, IntelliJ IDEA, and Visual Studio, which provide robust support for debugging, version control, and test management.

2.QTP/UFT:

- **Limited Language Support**: QTP/UFT primarily uses VBScript as its scripting language. While VBScript is simple to use, it can be limiting for more advanced automation or integration with complex systems.
- **Less Flexibility**: Testers are restricted to VBScript or other limited scripting languages for writing test cases.

4. Community and Ecosystem :

1.Selenium:

- **Large and Active Community**: Being open-source, Selenium has a vast and active community of users, developers, and contributors. The community frequently shares solutions, tutorials, bug fixes, and new features.
- **Continuous Development**: Selenium has constant updates and is actively maintained with improvements in functionality, bug fixes, and compatibility with new browser versions and technologies.
- **Rich Ecosystem**: Selenium integrates seamlessly with a wide range of tools such as TestNG, JUnit, Maven, Jenkins (for CI/CD), Appium (for mobile automation), and many more.

2.QTP/UFT:

- **Proprietary Product**: QTP/UFT has a smaller community compared to Selenium and is primarily supported by Micro Focus (formerly HP). Although UFT has a user base, the community is not as broad or active as Selenium’s open-source ecosystem.

- **Limited Integrations**: While UFT integrates with other HP tools like ALM (Application Lifecycle Management), it doesn't have the same level of flexibility or integration with other modern tools used in DevOps pipelines.

5. Flexibility and Customization :

1.Selenium:

- **Customizable Frameworks**: Selenium offers great flexibility for creating highly customizable test automation frameworks. Whether you need data-driven, keyword-driven, or behavior-driven testing, Selenium can accommodate various automation strategies and frameworks.
- **CI/CD Compatibility**: Selenium works well with continuous integration tools like Jenkins, Bamboo, and TeamCity, enabling automated testing in modern DevOps pipelines.

2.**QTP/UFT**:

- **Less Flexibility**: While QTP/UFT has built-in features for common test automation needs, it is less flexible than Selenium when it comes to building custom frameworks or handling complex automation scenarios.
- **Less Integration with CI/CD**: Although UFT supports integration with some CI/CD tools, it is less seamless than Selenium when it comes to automating test scripts within a continuous integration pipeline.

6. Speed and Lightweight :

1.Selenium:

- **Faster Execution**: Since Selenium is a lightweight tool, it tends to execute faster, especially when paired with modern test frameworks and CI/CD pipelines. It also has a quicker startup time compared to UFT.
- **Smaller Memory Footprint**: Selenium's minimalistic design makes it less resource-intensive, meaning it can run more tests concurrently with less strain on the system.

2.QTP/UFT:

- **Slower Execution**: UFT can be slower in comparison, particularly when executing larger test suites, due to its heavier architecture and reliance on proprietary libraries.
- **Higher System Requirements**: QTP/UFT requires more system resources (memory, CPU) to run efficiently, which can be a bottleneck in large-scale test automation efforts.

7. Support for Modern Web Technologies :

1.Selenium:

- Support for Ajax, JavaScript, and HTML5: Selenium is well-suited to testing modern web applications that use AJAX, JavaScript frameworks (e.g., React, Angular), and HTML5. It interacts with dynamic content and is designed for modern web architectures.

2.QTP/UFT:

- Slower Adaptation to New Web Technologies: QTP/UFT has had a slower adaptation to modern web technologies like single-page applications (SPA) or web apps built with JavaScript frameworks. Testers may face challenges when automating tests for newer web standards.

15.To validate the tops technologies website Contact us page and enter your friend detail at last

“Login and sidemenu” https://www.saucedemo.com/

1. In JMeter:

- When you finish recording your interactions with the website, you will have a Test Plan which includes the recorded HTTP requests (browser interactions).

- To save your script:

1. Right-click on the Test Plan in the left panel (on the Test Plan node).

2. Select Save As.

3. Choose a location on your system and give the file a name (e.g., TopsTech_Performance_Test.jmx).

4. Click Save.

2. In LoadRunner (VuGen):

- Once you've finished recording the website interactions using VuGen (Virtual User Generator):

    1. Stop the recording by clicking the Stop Recording button in VuGen.

    2. To save your script:
        - Click File in the menu bar.
        - Select Save Script.
        - Choose the location where you want to save the script, and give it a meaningful name (e.g., TopsTech_VuGen_Script).
        - Click Save.

3. In NeoLoad:

- After completing the recording of website interactions in NeoLoad:

1. Click on the "Stop" button in the recording toolbar to stop recording.

2. To save your Scenario:
    - Click on File in the top menu.
    - Select Save Scenario.
    - Choose a location on your system to save the scenario file.
    - Provide a name for your scenario (e.g., TopsTech_Scenario).
    - Click Save.