JMeter Assignment Questions

Q1.

To run a basic load test in Apache JMeter to demonstrate the basic elements such as Thread Group, Sampler, Assertions, Listener, and Timers, we can follow these steps:

Install JMeter: First, download and install Apache JMeter from the official website if we haven't already.

Launch JMeter: Start JMeter by executing the JMeter.bat (for Windows) or JMeter.sh (for Unix) file in the bin directory of the JMeter installation.

Create a Test Plan:

Right-click on the Test Plan (root) and select Add -> Threads (Users) -> Thread Group.

In the Thread Group, we can specify the number of threads (virtual users), ramp-up period, and loop count according to wer load testing requirements.

Add Sampler:

Right-click on the Thread Group, select Add -> Sampler -> HTTP Request.

In the HTTP Request sampler, enter the server name or IP address, protocol, method (GET/POST), and path.

Add Assertions:

Right-click on the HTTP Request sampler, select Add -> Assertions -> Response Assertion.

Configure the Response Assertion to specify the conditions that must be met for the sampler to be considered successful.

Add Timers:

Right-click on the Thread Group or individual samplers, select Add -> Timer -> Constant Timer or Uniform Random Timer.

Timers are used to introduce delay between requests to simulate real-world scenarios.

Add Listeners:

Right-click on the Thread Group, select Add -> Listener -> View Results Tree.

Listeners are used to view the test results. We can add multiple listeners according to wer requirements.

Configure Test Plan:

Configure any additional settings such as timeouts, proxy settings, etc., as needed.

Run the Test:

Click on the green play button to start the test.

JMeter will execute the test plan according to the configuration we provided.

Analyze Results:

Once the test completes, we can analyze the results using the various listeners we added.

View Results Tree, Summary Report, and Aggregate Report are commonly used listeners to analyze test results.

Adjust and Retest:

Based on the results, we can adjust the configuration of wer test plan (e.g., number of threads, ramp-up time, assertions) and retest as needed.

Save Test Plan:

Save wer test plan for future use by selecting File -> Save Test Plan As.

By following these steps, we can create and execute a basic load test using Apache JMeter, demonstrating the key elements such as Thread Group, Sampler, Assertions, Listener, and Timers.

Q2. To run a basic load test on a Selenium site using JMeter, we need to first record the interactions using JMeter's Proxy Server, as JMeter itself does not have native support for executing Selenium scripts. Follow these steps to create a basic load test for the Selenium site for Training Support:

Set Up JMeter:

Ensure we have Apache JMeter installed and ready to use.

Configure JMeter's Proxy Server:

Open JMeter and navigate to the Workbench.

Right-click on the Workbench and select Add -> Non-Test Elements -> HTTP(S) Test Script Recorder.

Configure the HTTP(S) Test Script Recorder with the desired port (e.g., 8888). Make sure to set up wer browser to use this port as a proxy.

Configure Browser to Use JMeter Proxy:

Open wer browser's settings and configure it to use the proxy server we set up in the previous step. Set the proxy host to localhost and the port to 8888.

Record Interactions:

Start the HTTP(S) Test Script Recorder in JMeter.

Now, perform the actions on the Selenium site (Training Support site) that we want to include in wer load test. JMeter will record these interactions.

Stop Recording:

Once we've recorded the interactions we want to include in wer load test, stop the HTTP(S) Test Script Recorder.

Correlate Dynamic Values:

Some dynamic values such as session IDs or CSRF tokens may need to be correlated to ensure realistic test scenarios. Use JMeter's Post-Processors like Regular Expression Extractor or XPath Extractor to extract and store these values for subsequent requests.

Add Thread Group:

Right-click on the Test Plan and select Add -> Threads (Users) -> Thread Group.

Configure the Thread Group with the desired number of users, ramp-up period, and loop count.

Add HTTP Request Samplers:

Add HTTP Request samplers for each recorded interaction from the Training Support site. We can find these under the Recording Controller.

Add Listeners:

Add Listeners such as View Results Tree, Summary Report, or Aggregate Report to analyze the test results.

Configure Timers and Assertions (if needed):

Depending on wer test requirements, we may add Timers and Assertions to wer test plan.

Run Test:

Save wer test plan and then run the test by clicking the play button.

Analyze Results:

Once the test completes, analyze the results using the added Listeners to identify any bottlenecks or issues.

By following these steps, we can create and execute a basic load test on the Selenium site for Training Support using Apache JMeter.

Q3. To perform a basic load test on a website that requires Basic Authentication using JMeter, we need to configure JMeter to include the necessary authentication headers in the HTTP requests. Here's how we can do it:

Add Thread Group:

Right-click on the Test Plan and select Add -> Threads (Users) -> Thread Group.

Configure the Thread Group with the desired number of users, ramp-up period, and loop count.

Add HTTP Authorization Manager:

Right-click on the Thread Group and select Add -> Config Element -> HTTP Authorization Manager.

In the HTTP Authorization Manager, configure the following:

Base URL: The URL of the website we're testing.

Username: The username for Basic Authentication.

Password: The password for Basic Authentication.

Add HTTP Request Sampler:

Right-click on the Thread Group and select Add -> Sampler -> HTTP Request.

Configure the HTTP Request with the following:

Protocol: HTTP or HTTPS depending on wer website.

Server Name or IP: The domain name or IP address of the website.

Path: The path to the specific resource we want to request.

Configure HTTP Authorization Manager in HTTP Request Sampler:

Within the HTTP Request sampler, scroll down to the "Authorization" tab.

Choose "HTTP Authorization Manager" from the dropdown list.

Add Listeners:

Add Listeners such as View Results Tree, Summary Report, or Aggregate Report to analyze the test results.

Configure Timers and Assertions (if needed):

Depending on wer test requirements, we may add Timers and Assertions to wer test plan.

Run Test:

Save wer test plan and then run the test by clicking the play button.

Analyze Results:

Once the test completes, analyze the results using the added Listeners to identify any bottlenecks or issues.

By following these steps, we can create and execute a basic load test on a website that requires Basic Authentication using Apache JMeter. Ensure that we have the correct credentials and permissions to access the website during the load test.

Q4. To implement a basic load test using pre and post processors on the Selenium site for Training Support, we can use Apache JMeter. Since JMeter does not directly execute Selenium scripts, we'll use the HTTP Request samplers to simulate user actions. Below are the steps to create a basic load test with pre and post processors:

Set Up JMeter:

Ensure we have Apache JMeter installed and ready to use.

Record User Actions:

Use JMeter's HTTP(S) Test Script Recorder to record the user actions on the Training Support Selenium site. Configure wer browser to use JMeter's proxy while recording.

Add Thread Group:

Right-click on the Test Plan and select Add -> Threads (Users) -> Thread Group.

Configure the Thread Group with the desired number of users, ramp-up period, and loop count.

Add HTTP Authorization Manager (if needed):

If the site requires authentication, add an HTTP Authorization Manager and configure it with the necessary credentials.

Add HTTP Request Samplers:

For each recorded user action, add an HTTP Request sampler. We can find these under the Recording Controller.

Add Pre-Processors (if needed):

If we need to preprocess any data before sending the request, we can use Pre-Processors like Regular Expression Extractor or XPath Extractor. For example, if we need to extract CSRF tokens, use Regular Expression Extractor.

Add Post-Processors (if needed):

If we need to process any data from the response, we can use Post-Processors. For example, if we need to extract specific data from the response for further use, we can use Regular Expression Extractor or JSON Extractor.

Add Listeners:

Add Listeners such as View Results Tree, Summary Report, or Aggregate Report to analyze the test results.

Configure Timers and Assertions (if needed):

Depending on wer test requirements, we may add Timers and Assertions to wer test plan.

Run Test:

Save wer test plan and then run the test by clicking the play button.

Analyze Results:

Once the test completes, analyze the results using the added Listeners to identify any bottlenecks or issues.

By following these steps, we can create and execute a basic load test on the Selenium site for Training Support using Apache JMeter, including pre and post processors as needed. Make sure to adjust the test plan according to wer specific requirements and ensure that wer load test scenario accurately represents real-world usage.

Q5. A demonstration of how the HTTP Link parser works

The HTTP Link parser in JMeter is used to extract links from HTTP responses and store them for further processing. This can be useful in scenarios where we need to extract URLs from web pages to navigate to other pages or perform additional actions.

Here's a step-by-step demonstration of how the HTTP Link parser works in JMeter:

Set Up Test Plan:

Open JMeter and create a new Test Plan.

Add Thread Group:

Right-click on the Test Plan and select Add -> Threads (Users) -> Thread Group.

Add HTTP Request Sampler:

Right-click on the Thread Group and select Add -> Sampler -> HTTP Request.

Configure the HTTP Request sampler with the URL of the web page from which we want to extract links.

Add HTTP Link Parser:

Right-click on the HTTP Request sampler and select Add -> Post Processors -> HTTP URL Re-writing Modifier.

HTTP URL Re-writing Modifier is the HTTP Link parser in JMeter.

Configure HTTP Link Parser:

In the HTTP URL Re-writing Modifier, configure the following:

Apply to: Select "Main sample and sub-samples."

Link URL: Provide a regular expression to match the links we want to extract. For example, to extract all links <a href="...">, we can use <a href="([^"]\*)">.

Add Listener:

Right-click on the Thread Group and select Add -> Listener -> View Results Tree.

View Results Tree will allow we to view the extracted links.

Run Test:

Save wer Test Plan and then run the test by clicking the play button.

Analyze Results:

Once the test completes, go to the View Results Tree listener.

In the Sampler Result tab, we should see the extracted links under the "Response Data" section.

Here's an example regular expression we can use in the HTTP Link parser:

css

Copy code

<a href="([^"]\*)">

This regular expression captures all the URLs within <a> tags.

Remember to adjust the regular expression according to wer specific requirements and the structure of the HTML content we're parsing.

By following these steps, we can demonstrate how the HTTP Link parser works in JMeter by extracting links from HTTP responses.

Q6. A Spidering example using the Selenium website

To perform a spidering (also known as web crawling) example using the Selenium website with Apache JMeter, you can simulate the behavior of a web crawler by navigating through links on web pages and extracting URLs. Below are the steps to create a basic spidering test plan:

Set Up JMeter:

Ensure you have Apache JMeter installed and ready to use.

Create a Test Plan:

Open JMeter and create a new Test Plan.

Add Thread Group:

Right-click on the Test Plan and select Add -> Threads (Users) -> Thread Group.

Add WebDriver Sampler:

Right-click on the Thread Group and select Add -> Sampler -> jp@gc - WebDriver Sampler.

WebDriver Sampler allows you to execute Selenium WebDriver scripts within JMeter.

Configure WebDriver Sampler:

In the WebDriver Sampler, write Selenium WebDriver code to navigate through the Selenium website, find links, and extract URLs.

Here's an example Selenium WebDriver script in Java syntax to navigate through the Selenium website and extract URLs:

java

Copy code

WDS.sampleResult.sampleStart()

WDS.browser.get("https://www.selenium.dev/");

List<WebElement> links = WDS.browser.findElements(By.tagName("a"));

for (WebElement link : links) {

String url = link.getAttribute("href");

if (url != null && !url.isEmpty()) {

WDS.log.info("Found URL: " + url);

}

}

WDS.sampleResult.sampleEnd()

Add Listener:

Right-click on the Thread Group and select Add -> Listener -> View Results Tree.

View Results Tree will allow you to view the extracted URLs.

Run Test:

Save your Test Plan and then run the test by clicking the play button.

Analyze Results:

Once the test completes, go to the View Results Tree listener.

In the Sampler Result tab, you should see the extracted URLs printed in the log.

This basic example demonstrates how you can use Apache JMeter along with the WebDriver Sampler to perform spidering on the Selenium website. You can further enhance this test plan by adding additional logic to handle dynamic content, crawling depth, and URL filtering as per your requirements.

To connect to a database and run a load test using Apache JMeter, you can follow these steps:

Set Up JMeter:

Ensure you have Apache JMeter installed and ready to use.

Download Database Driver:

If you haven't already, download the appropriate database driver for the database you want to connect to. This driver is usually a JAR file.

Copy Driver to JMeter's lib Directory:

Copy the downloaded database driver JAR file to the "lib" directory within your JMeter installation folder.

Create a Test Plan:

Open JMeter and create a new Test Plan.

Add Thread Group:

Right-click on the Test Plan and select Add -> Threads (Users) -> Thread Group.

Add JDBC Connection Configuration:

Right-click on the Thread Group, select Add -> Config Element -> JDBC Connection Configuration.

Configure the JDBC Connection Configuration with the following details:

Database URL: The JDBC URL for your database (e.g., jdbc:mysql://localhost:3306/mydatabase).

JDBC Driver class: The fully qualified class name of the JDBC driver for your database.

Username and Password: Your database credentials.

Add JDBC Request Sampler:

Right-click on the Thread Group and select Add -> Sampler -> JDBC Request.

Configure the JDBC Request sampler with the SQL query you want to execute against the database.

Add Listeners:

Right-click on the Thread Group and select Add -> Listener -> View Results Tree or any other appropriate listener to view test results.

Configure Thread Group:

Configure the Thread Group with the desired number of threads (simulated users), ramp-up period, and loop count.

Run Test:

Save your Test Plan and then run the test by clicking the play button.

Analyze Results:

Once the test completes, analyze the results using the added Listeners to identify any performance bottlenecks or issues.

Adjust and Retest:

Based on the results, you may need to adjust your test plan parameters or SQL queries and rerun the test to validate improvements.

By following these steps, you can connect to a database and run a load test using Apache JMeter. Make sure to replace placeholder values with actual database connection details and SQL queries relevant to your scenario.