Module-4

Automation Core Testing (Load Runner Up and Selenium IDE)

1. **Which components have you used in Load Runner?**

- **Virtual User Generator (VuGen)**
- **Controller**
- **Load Generators**
- **Agent Process**
- **Analysis**
- **Monitoring Tools**
- **Protocols**

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

In LoadRunner, the number of Virtual Users (Vusers) can be set through the LoadRunner Controller. Here's how you can do it:

- **Open LoadRunner Controller:** Launch the LoadRunner Controller application.
- **Create or Open a Scenario:** You can either create a new scenario or open an existing one.
- **Define Vuser Groups:** In LoadRunner, virtual users are organized into groups called Vuser groups. Each Vuser group represents a set of users executing a specific set of actions. You can create multiple Vuser groups to simulate different user behaviors.
- **Set the Number of Vusers:** Within each Vuser group, you can specify the number of virtual users to simulate. This can be done by entering the desired number in the "Number of Vusers" field or by using the sliders provided.
- **Distribute Vusers:** If you have multiple load generators, you can distribute the Vusers across these generators. LoadRunner allows you to specify how many Vusers should run on each load generator.
- **Configure Ramp-Up:** Ramp-up refers to the gradual increase in the number of Vusers over time. You can configure ramp-up settings to simulate realistic user behaviour, such as a gradual increase in user load during peak hours.

- **Run the Scenario:** Once you've configured the number of Vusers and other settings, you can start the scenario execution. The Controller will distribute the specified number of Vusers across the load generators and begin simulating user activity on the application under test.
- **Monitor Execution:** During scenario execution, you can monitor various performance metrics such as response times, throughput, and server resource utilization to assess the application's performance under load.

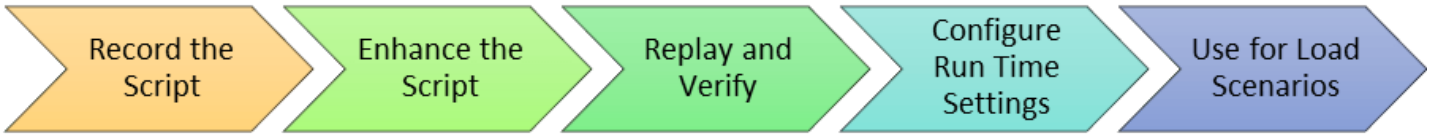
By adjusting the number of Vusers and other parameters in the LoadRunner Controller, you can simulate different levels of user load and analyse how the application behaves under various conditions.

3. What is Correlation?

Correlation, in the context of LoadRunner or performance testing, refers to the process of capturing and replacing dynamic values in a recorded script with parameters. Dynamic values are those that change with each user session or interaction, such as session IDs, authentication tokens, timestamps, or any other unique identifiers generated by the server.

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

- **The script development process in VUGen**



- **Record the Script:** Usually, this is the first step of scripting where every user action is recorded into a script.
- **Replay and Verify:** Once the script is recorded, reply the script to ensure its working right. Verify any impact through application frontend or database.
- **Enhance the Script:** Once recording has been verified, enhance script by adding checkpoints, validating data, adding transactions and rendezvous points.
- **Replay and Verify:** As earlier, re-play your script and verify that everything is working as intended.

- **Configure Runtime Settings:** Configure and control pacing duration, think time variation, proxy settings and whether you wish to ignore any external resources.
- **Use for Load Scenarios:** Formulate load scenarios based on test objectives. Use load distribution and geo-wide agents to make real like scenarios.

5. How Load Runner interacts with the application?

Here's how LoadRunner typically interacts with an application:

- **Scripting:** Testers create scripts using LoadRunner's scripting language, usually by recording user interactions or manually coding the scenarios. These scripts simulate user actions like logging in, browsing pages, filling forms, etc.
- **Parameterization:** LoadRunner allows for parameterization, where variables can be used to represent dynamic data such as user names, passwords, or input data. This enables realistic simulation of user behaviour.
- **Scenario Design:** Testers design scenarios that represent different types of user loads and behaviours. For example, a scenario might simulate a specific number of users logging in simultaneously or performing specific actions at different rates.
- **Controller Setup:** LoadRunner's Controller component orchestrates the execution of scenarios. Testers configure the desired number of virtual users, ramp-up rates, and other parameters to mimic real-world usage patterns.
- **Execution:** The Controller distributes the load across multiple load generators (machines or virtual machines running LoadRunner's Virtual User Generator or Vuser scripts). Each load generator simulates multiple virtual users executing the scripts against the application under test.
- **Monitoring:** During test execution, LoadRunner collects performance metrics such as response times, throughput, CPU and memory usage, database performance, etc., from the application servers, web servers, and other infrastructure components.
- **Analysis:** After the test completes, testers analyse the collected data using LoadRunner's Analysis component. They can identify performance bottlenecks, pinpoint areas for optimization, and evaluate the application's scalability, reliability, and responsiveness under different loads.
- **Reporting:** LoadRunner generates comprehensive reports summarizing the test results, including graphs, charts, and statistics, to help stakeholders understand the application's performance characteristics and make informed decisions.

Overall, LoadRunner provides a robust framework for assessing an application's performance under realistic conditions, helping organizations ensure that their software meets performance objectives and user expectations.

6. **How many VUsers are required for load testing?**

Determining the number of Virtual Users (VUsers) required for load testing depends on several factors, including the objectives of the test, the nature of the application, the expected user load in production, and the available infrastructure. Here are some considerations:

- **Business Requirements**
- **Expected User Load**
- **Scalability Goals**
- **Infrastructure Capacity**
- **Test Scenarios**
- **Ramp-Up Strategy**
- **Monitoring and Analysis**
- **Iterations**

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

While response time and throughput are related, they are not directly proportional. In some cases, as throughput increases, response time may also increase, and vice versa.

A system with high throughput may still have high response times if it's struggling to process a large number of concurrent requests.

Conversely, a system with low throughput may have low response times if it's not under heavy load.

The relationship between response time and throughput is influenced by various factors, including system resources, concurrency, network latency, application design, and workload characteristics.

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

- **to Record all top level menu**
- **to Record minimum 10 Vuser on this website**
- **save all (Script,Design,Graph)**

```
Action()
{

	web_url("gts1c3.der",
		"URL=http://pki.goog/repo/certs/gts1c3.der",
		"Resource=1",
		"RecContentType=application/pkix-cert",
		"Referer=",
		"Snapshot=t5.inf",
		LAST);

	web_url("gtsr1.der",
		"URL=http://pki.goog/repo/certs/gtsr1.der",
		"Resource=1",
		"RecContentType=application/pkix-cert",
		"Referer=",
		"Snapshot=t6.inf",
		LAST);

	web_url("gts1c3.der_2",
		"URL=http://pki.goog/repo/certs/gts1c3.der",
		"Resource=1",
		"RecContentType=application/pkix-cert",
		"Referer=",
		"Snapshot=t7.inf",
		LAST);

	web_url("gtsr1.der_2",
		"URL=http://pki.goog/repo/certs/gtsr1.der",
		"Resource=1",
		"RecContentType=application/pkix-cert",
		"Referer=",
		"Snapshot=t8.inf",
		LAST);

	web_url("RapidSSLTLSRSACAG1.crt",
		"URL=http://cacerts.rapidssl.com/RapidSSLTLSRSACAG1.crt",
		"Resource=1",
		"RecContentType=application/pkix-cert",
		"Referer=",
		"Snapshot=t9.inf",
```

```
                LAST);

        return 0;
    }
```

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

```
Action()
{

    web_url("index.htm",
            "URL=http://127.0.0.1:1080/WebTours/index.htm",
            "Resource=0",
            "Referer=",
            "Snapshot=t1.inf",
            "Mode=HTML",
            LAST);

    web_url("header.html",
            "URL=http://127.0.0.1:1080/WebTours/header.html",
            "Resource=0",
            "Referer=http://127.0.0.1:1080/WebTours/index.htm",
            "Snapshot=t2.inf",
            "Mode=HTML",
            LAST);

    web_url("welcome.pl",
            "URL=http://127.0.0.1:1080/cgi-bin/welcome.pl?signOff=true",
            "Resource=0",
            "RecContentType=text/html",
            "Referer=http://127.0.0.1:1080/WebTours/index.htm",
            "Snapshot=t3.inf",
            "Mode=HTML",
            EXTRARES,
            "Url=http://pki.goog/repo/certs/gts1c3.der", "Referer=", ENDITEM,
            "Url=http://pki.goog/repo/certs/gtsr1.der", "Referer=", ENDITEM,
            LAST);

    lr_save_string(lr_decrypt("6620c77998a0b4f6"), "PasswordParameter");

    web_submit_data("login.pl",
            "Action=http://127.0.0.1:1080/cgi-bin/login.pl",
```

```
            "Method=POST",
            "RecContentType=text/html",
            "Referer=http://127.0.0.1:1080/cgi-bin/nav.pl?in=home",
            "Snapshot=t4.inf",
            "Mode=HTML",
            ITEMDATA,
            "Name=userSession",
"Value=138794.996921831HVDQczHpzftVzzzHtciDfpzAfQcf", ENDITEM,
            "Name=username", "Value=jojo", ENDITEM,
            "Name=password", "Value={PasswordParameter}", ENDITEM,
            "Name=login.x", "Value=47", ENDITEM,
            "Name=login.y", "Value=8", ENDITEM,
            "Name=JSFormSubmit", "Value=off", ENDITEM,
            LAST);

    return 0;
}
```

10. What is Automation Testing?

Automation testing is a software testing technique that involves using automated tools and scripts to execute test cases and verify the behaviour and performance of software applications. Instead of manual intervention, automation testing relies on pre-written scripts to perform repetitive and complex test scenarios, thereby increasing testing efficiency, repeatability, and accuracy.

11. Which Are The Browsers Supported By Selenium Ide?

1. **Mozilla Firefox:** Selenium IDE is primarily developed for Mozilla Firefox and offers the most comprehensive support for this browser.
2. **Google Chrome:** Selenium IDE also provides experimental support for Google Chrome. There is a separate Chrome extension available, but the functionality may be limited compared to the Firefox version, and it may still be in development or testing phases.
3. **Microsoft Edge:** Selenium IDE may also have experimental support for Microsoft Edge, either through a dedicated Edge extension or compatibility with Chrome extensions (since Edge is based on Chromium).
4. **Other Browsers:** Support for other browsers may vary, and it's recommended to check the latest documentation or announcements from the Selenium IDE project for updates on browser support.

12. What are the benefits of Automation Testing?

Automation testing offers numerous benefits that contribute to improving the efficiency, effectiveness, and quality of software development processes. Here are some of the key benefits:

- **Faster Time-to-Market**
- **Increased Testing Coverage**
- **Repeatability and Consistency**
- **Cost Savings**
- **Improved Accuracy**
- **Early Detection of Defects**
- **Support for Continuous Integration/Continuous Deployment (CI/CD**
- **Scalability**
- **Enhanced Developer Productivity**
- **Improved Software Quality**

  Overall, automation testing is a valuable practice that offers numerous benefits for software development teams, helping them deliver high-quality products more efficiently and effectively.

13. What are the advantages of Selenium?

Selenium is a widely used open-source tool for automating web browsers. It offers several advantages that make it a preferred choice for automated testing of web applications:

- **Cross-Browser Compatibility**
- **Platform Independence**
- **Support for Multiple Programming Languages**
- **Integration with Testing Frameworks**
- **Rich Set of Features**
- **Community Support**
- **Extensibility**
- **Open-Source**
- **Browser Interaction Simulation**
- **Support for Headless Testing**

Overall, Selenium offers a powerful and flexible platform for automating web browser testing, helping organizations improve the efficiency, reliability, and quality of their web applications.

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

Choosing between Selenium and QTP (now known as UFT - Unified Functional Testing) depends on various factors, including the project requirements, budget, skillset of the testing team, and the specific needs of the organization. Here are some reasons why testers might opt for Selenium over QTP/UFT:

- **Open Source vs. Commercial Tool**
- **Platform Independence**
- **Programming Language Support**
- **Community Support and Flexibility**
- **Integration with Testing Frameworks**
- **Web Application Testing Focus**
- **Cost Considerations**

Ultimately, the choice between Selenium and QTP/UFT depends on the specific needs and priorities of the organization, as well as factors such as budget, expertise, and project requirements. While Selenium offers many advantages, QTP/UFT may still be a suitable option for organizations that prioritize commercial support, comprehensive testing capabilities, and integration with other Micro Focus products.