**Software Testing**

**(Module – 4) Automation Core Testing**

**Assignment – 5**

**NAME: - Dhruvin Sathwara**

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

The key components of Load Runner are:

* **Monitoring Tools:** Load Runner integrates with various monitoring tools to capture performance metrics from the application, web servers, databases, and other infrastructure components during test execution.
* **Protocols:** Load Runner supports a wide range of protocols for recording and scripting different types of applications, including web-based applications (HTTP/HTML), web services (SOAP, REST), database protocols (JDBC, ODBC), messaging protocols (MQ), and more.
* **Virtual User Generator (VuGen):** VuGen is used for recording user actions on the application to create scripts for performance testing. It supports multiple scripting languages like C, Java, and VBScript.
* **Controller:** The Controller component is used to design and manage the execution of load testing scenarios. It allows testers to configure the number of virtual users, define scenarios, set up load distribution, and monitor test execution.
* **Load Generators:** Load generators are machines or virtual machines responsible for executing the scripts generated by VuGen. They simulate multiple virtual users to generate load on the application under test.
* **How can you set the number of Vusers in Load Runner?**

You can set the number of Vusers in the controller section while creating your scenarios. Many other advanced options like ramp-up, ramp-down of Vusers are also available in the Controller section.

1. **Open Load Runner Controller**
2. **Create or Open a Scenario**
3. **Define Vuser Groups**
4. **Set the Number of Vusers**
5. **Distribute Vusers**
6. **Configure Ramp-Up**
7. **Run the Scenario**
8. **Monitor Execution.**

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.

* **What is Correlation?**

Correlation is used to obtain data which is unique for each run of your test script (ex: session ids).

Correlation, in the context of Load Runner or performance testing, refers to the process of capturing and replacing dynamic values in a recorded script with parameters.

While recording, these dynamic values are hard-coded in your script causing the script to fail during playback. Correlation is a technique where dynamic values are not hard-coded in your script but are extracted at run-time to avoid failure.

* **What is the process for developing a Vuser Script?**

**1. Record the Script:** Usually, this is the first step of scripting where every user action is recorded into a script.

**2. Replay and Verify:** Once the script is recorded, reply the script to ensure its working right. Verify any impact through application frontend or database.

**3. Enhance the Script:** Once recording has been verified, enhance script by adding checkpoints, validating data, adding transactions and rendezvous points.

**4. Replay and Verify:** As earlier, re-play your script and verify that everything is working as intended.

**5. Configure Runtime Settings:** Configure and control pacing duration, think time variation, proxy settings and whether you wish to ignore any external resources.

**6. Use for Load Scenarios:** Formulate load scenarios based on test objectives. Use load distribution and geo-wide agents to make real like scenarios.

* **How Load Runner interacts with the application?**

Load Runner interacts with applications primarily through a process called performance testing. Load Runner is a performance testing tool that simulates user activity on software applications, allowing testers to measure and analyse system performance under various conditions.

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:** Load Runner'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 Load Runner'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, Load Runner 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 Load Runner'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:** Load Runner 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.
* **How many VUsers are required for load testing?**

The number of VUsers required depends on your system under test, network configurations, hardware settings, memory, operating system, software applications objective of a performance test. There cannot be any generic value for Vuser.

* **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.

* **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.

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

However, Selenium IDE has been evolving, and there have been efforts to support other browsers as well. Here's the browser support status as of my last update:

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.

* **What are the benefits of Automation Testing?**
* **Enhanced Developer Productivity:** Automation testing frees up developers from repetitive and time-consuming testing tasks, allowing them to focus on higher-value activities such as feature development, code optimization, and innovation.
* **Improved Software Quality:** By identifying and addressing defects early in the development process, automation testing contributes to improving the overall quality, reliability, and stability of software applications, leading to higher customer satisfaction and loyalty.
* **Early Detection of Defects:** Automation testing enables early detection of defects in the software development lifecycle, allowing teams to address issues promptly and prevent them from escalating into more significant problems later in the development process.
* **Support for Continuous Integration/Continuous Deployment (CI/CD):** Automation testing is well-suited for integration into CI/CD pipelines, enabling automated testing of code changes with each build or deployment. This facilitates rapid feedback loops and ensures that new features or changes are thoroughly tested before being released to production.
* **Scalability:** Automated tests can handle large test suites and complex scenarios, making them suitable for testing applications with evolving requirements and increasing complexity.
* **Faster Time-to-Market:** Automation testing enables faster execution of test cases, allowing for quicker identification and resolution of bugs. This accelerates the software development lifecycle and helps deliver products to market more rapidly.
* **Increased Testing Coverage:** Automated tests can cover a wide range of test scenarios, including regression testing, integration testing, and performance testing.
* **Repeatability and Consistency:** Automated tests produce consistent and reproducible results, reducing the risk of human error and ensuring consistent test coverage across different test runs and environments.
* **What are the advantages of Selenium?**
* **Integration with Testing Frameworks:** Selenium can be easily integrated with popular testing frameworks such as JUnit, TestNG, NUnit, and Pytest. This integration allows testers to leverage advanced testing features such as test reporting, parallel execution, data-driven testing, and more.
* **Rich Set of Features:** Selenium offers a rich set of features for automating web browser interactions, including form filling, clicking buttons, navigating links, handling alerts and pop-ups, capturing screenshots, and validating page content. These features enable testers to simulate real user interactions and thoroughly test web applications.
* **Community Support:** Selenium has a large and active community of developers and testers who contribute to its development, share knowledge, and provide support through forums, blogs, and online communities. This community-driven approach ensures timely updates, bug fixes, and enhancements to the Selenium framework.
* **Cross-Browser Compatibility:** Selenium supports multiple web browsers, including Google Chrome, Mozilla Firefox, Microsoft Edge, Safari, and others. This cross-browser compatibility allows testers to validate the consistency of web applications across different browsers and platforms.
* **Platform Independence:** Selenium is a platform-independent tool, meaning it can run on various operating systems such as Windows, macOS, Linux, and Unix. This flexibility allows testers to develop and execute tests on their preferred operating system without any compatibility issues.
* **Support for Multiple Programming Languages:** Selenium supports multiple programming languages, including Java, Python, C#, Ruby, and JavaScript. Testers can write automated test scripts using their preferred programming language, making Selenium accessible to a wide range of developers and testers.
* **Open-Source:** Selenium is an open-source tool distributed under the Apache License 2.0, making it freely available for use, modification, and distribution. Organizations can leverage Selenium without incurring licensing costs, making it a cost-effective solution for test automation.
* **Browser Interaction Simulation:** Selenium provides a set of APIs for simulating user interactions with web browsers, enabling testers to automate complex testing scenarios such as filling forms, uploading files, handling cookies, and executing JavaScript actions.
* **Why testers should opt for Selenium and not QTP?**

The best test automation tool for you will depend on your specific needs and requirements. If you are looking for a commercial tool with a wide range of features and support for record-and-playback, then QTP/UFT is a good choice.

If you are looking for an open-source tool that supports a wide range of programming languages and operating systems, then Selenium is a good choice.

For example, let me say your team is already familiar with VBScript and the Windows operating system. Here QTP/UFT may be a better choice. On the other hand, if your team is more experienced with other programming languages and operating systems, Selenium may be a better fit.

* **To test the Performance testing on “Tops Technologies website”:- https://www.saucedemo.com/ 1. to Record all top level menu 2. to Record minimum 10 Vuser on this website 3. 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;

}

* **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;

}