NAME: KSHATRIYA RONAK OMPRAKASH

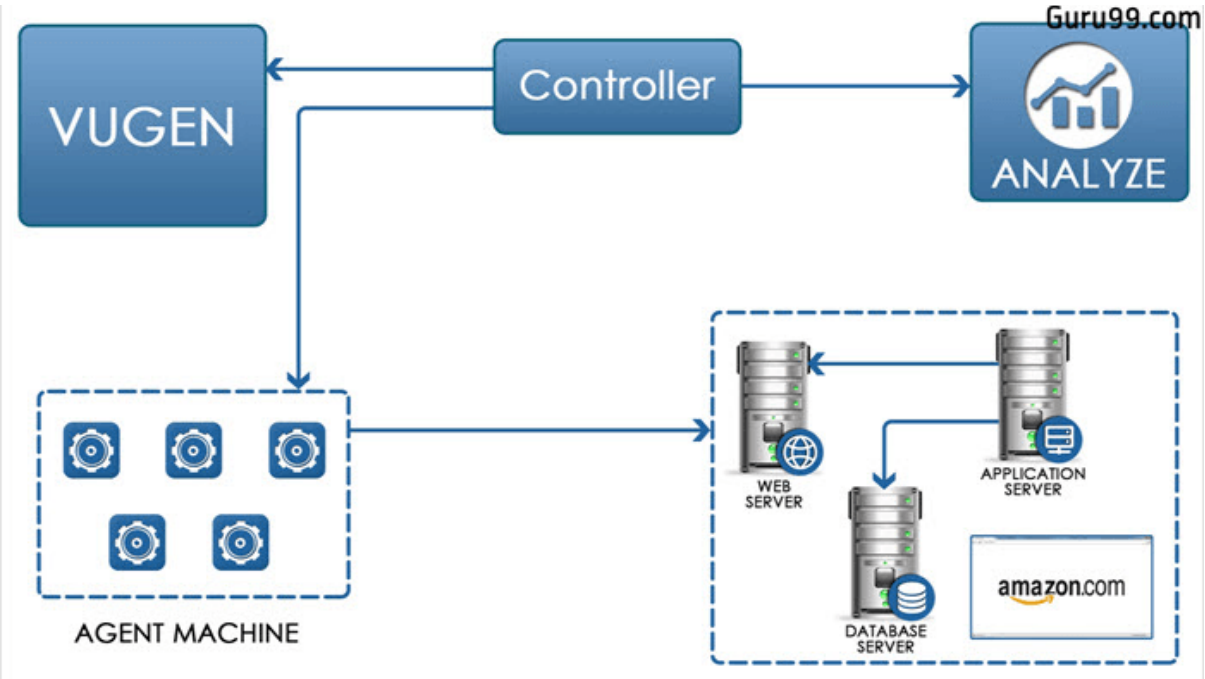
COURSE: SOFTWARE TESTING (MANUAL AND AUTOMATION)

ASSIGNMENT: Automation Core Testing (Load Runner Up and Selenium IDE)

1. Which components have you used in Load Runner?

The main components of Load Runner, a performance testing tool, are:

- **Load Generator**: Generates load on an application by following scripts
- **VuGen (Virtual User Generator)**: Creates and edits scripts
- **Controller**: Launches, controls, and sequences Load Generator instances
- **Agent Process**: Manages the connection between the Controller and Load Generator instances
- **Analysis**: Assembles logs from various Load Generators and creates reports in various formats



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

LoadRunner, you can set the number of virtual users (Vusers) in a few ways:

- **Add a Vuser group**

  In the Run Dashboard, select More > Add Group. Then, enter the number of Vusers to run.

- **Edit a Vuser group**

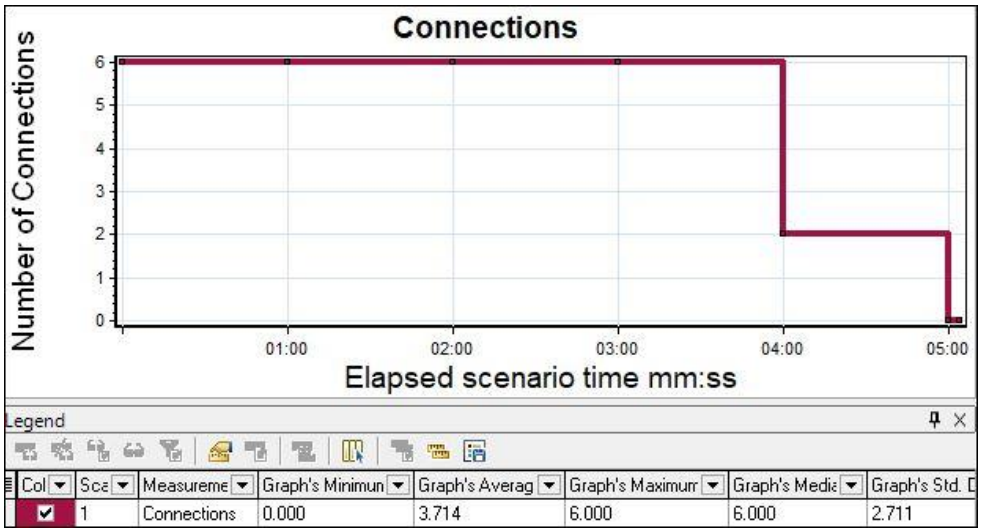
  In the Run Dashboard, open the Groups tab, hover over the group name, and select Edit Group. Then, edit the number of Vusers assigned to the group.

- **Distribute Vusers by number**

  In the Performance Test Designer window, select a group in the Groups grid. Then, enter the number of Vusers to allocate to that group in the Vusers column.

- **Define a schedule**

  Define the number of Vusers when you define the test schedule.



3. What is Correlation?

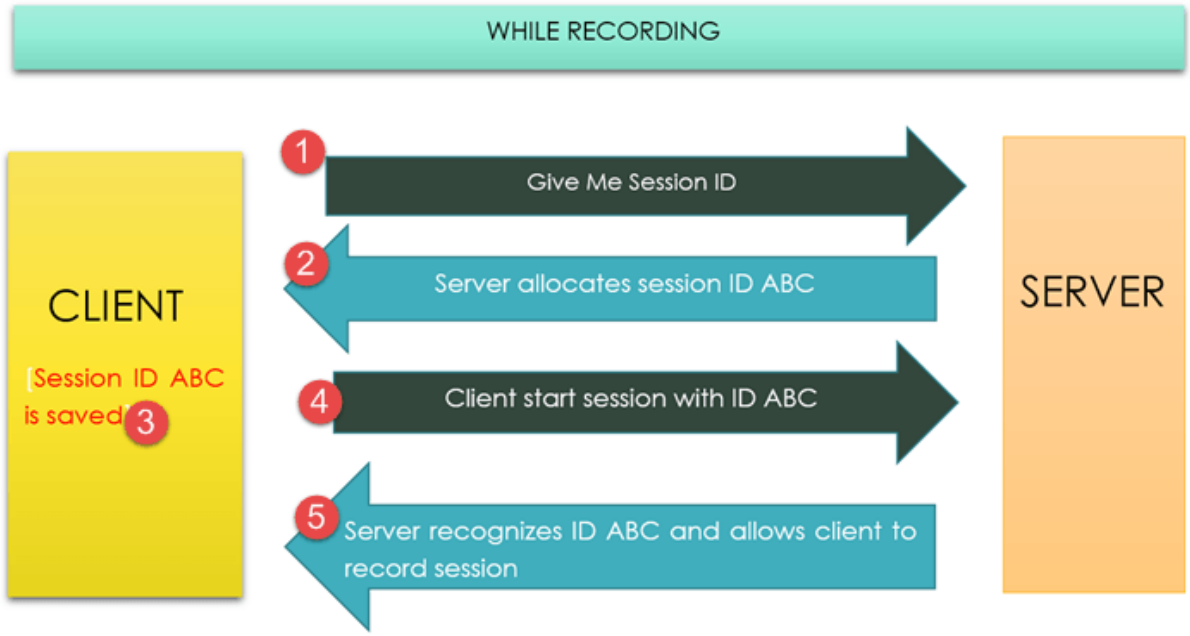
Correlation in Load Runner is a process that captures dynamic values returned by a server and passes them to subsequent requests. This is useful when a test script has dynamic values or generates them during execution. Correlation ensures that the test script executes without errors.

- **Automatic correlation**

Load Runner can automatically perform correlation.

- **Manual correlation**

Users can manually add correlations that the automatic engine missed, or identify all correlations themselves.



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

1. **Record the script**: Record the actions of a typical end-user on the application. For example, if testing a website, the script would emulate a user clicking links, submitting forms, or accessing URLs.
2. **Playback and improve**: Playback the recorded script and make improvements.
3. **Define and test run-time parameters**: Define and test the different run-time parameters.
4. **Use the script in a scenario**: Use the script in a LoadRunner scenario.

Here are some other things to consider when developing a Vuser script:

- **Script verification**

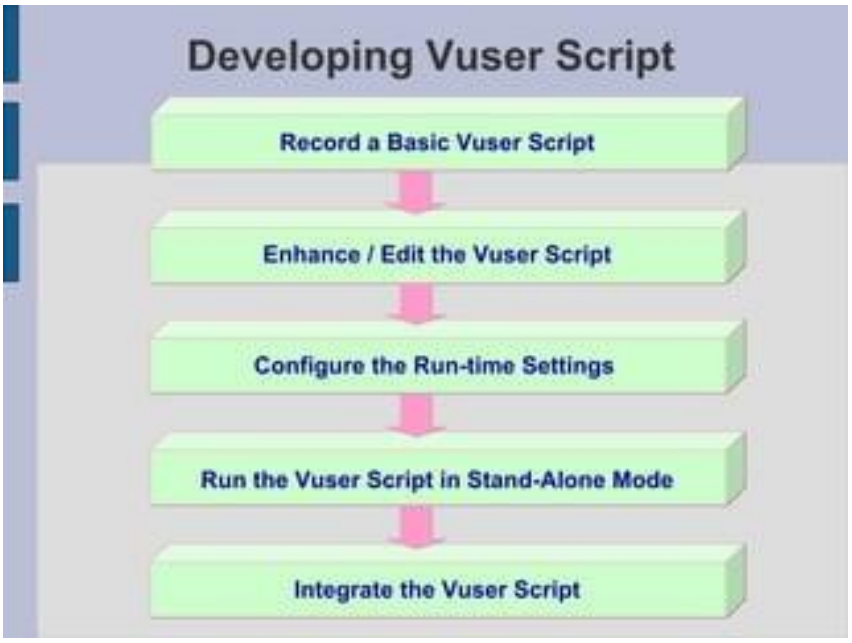
  It's recommended to verify the script's authenticity before putting it under heavy user load. This can help avoid last-minute glitches.

- **Runtime settings**

  Configure runtime settings for the script, such as selecting the browser to use.

- **DevWeb protocol**

  Use the DevWeb Proxy Recorder to record a business process locally or on a remote machine



5. How Load Runner interacts with the application?

LoadRunner simulates user activity by generating messages between application components or by simulating interactions with the user interface such as key presses or mouse movements. The messages and interactions to be generated are stored in scripts.

LoadRunner is a performance testing tool that interacts with an application by simulating user activity:

- **Generating messages**

LoadRunner generates messages between application components.

- **Simulating interactions**

LoadRunner simulates interactions with the user interface, such as key presses and mouse movements.

- **Using scripts**

LoadRunner stores the messages and interactions to be generated in scripts.

- **Using VuGen**

LoadRunner's Virtual User Generator (VuGen) records actions as a user walks through a business process on an application. VuGen then converts these actions into Vuser scripts.

-

- **Using the Load Generator**

The Load Generator simulates multiple virtual users to create realistic user scenarios.

6. How many VUsers are required for load testing?

handle the desired number of VUsers. Here are some general guidelines:

1. **Expected user load**: Estimate the number of real users who will be using the application simultaneously.

2. **Application complexity:** More complex applications may require more VUsers to simulate realistic usage.

3. **Test goals:** Identify what you want to achieve with load testing (e.g., peak performance, stress testing).

4. Hardware resources: Ensure the load testing environment can handle the desired number of VUsers. Here are some general guidelines:

- Low-load testing: 10-50 VUsers (e.g., small applications, development testing)

- Medium-load testing (Stress): 50-200 VUsers (e.g., medium-sized applications, performance testing)

- High-load testing (Scalability): 200-1,000 VUsers (e.g., large applications, stress testing)

• Extreme-load testing (flood): 1,000+ VUsers (e.g., very large applications, extreme stress testing)

Remember, the key is to simulate realistic user behaviour and gradually increase the load to measure the y increase the load to measure the application's performance and identify bottlenecks.

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

   Response Time and Throughput are two related but distinct performance metrics:

i. **Response Time:** The time it takes for an application to respond
to a user request or action.

   It measures how long a user must wait for a response.

ii. **Throughput:** The number of user requests or actions an application can handle within a given time period (e.g., transactions per second).

The relationship between Response Time and Throughput is:

- Inverse correlation: As Throughput increases (more requests handled), Response Time typically decreases (faster responses).

- Trade-off: Improving one metric can impact the other. For example, optimizing for faster Response Times might reduce Throughput, and vice versa.

- Balance: Aim for a balance between Response Time and Throughput to ensure a good user experience and efficient system performance.

- In load testing, analyzing both Response Time and Throughput helps identify performance bottlenecks and optimize application performance.