**NAME :- MALHOTRA DAXIL**

**MODULE :- 4**

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

🡪 **Load Generator :-** in LoadRunner is the component that **creates virtual users (Vusers)** to simulate real traffic on the application. It runs the test scripts and generates the load by executing them on multiple machines to mimic the expected number of users accessing the system.

🡪 The **Controller** in LoadRunner is the component that **manages and runs** performance tests. It controls the virtual users (Vusers), starts the tests, and monitors their progress. It helps set up test scenarios and ensures everything runs smoothly during the test.

🡪 **Analysis** in LoadRunner is the component that helps you **view and interpret the results** of your performance tests. It shows important metrics like response times, error rates, and system performance, helping you understand how the application performed under load.

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

🡪 To set the number of **Vusers** in LoadRunner:

1. **Open the Controller** and create or open a test scenario.
2. In the **Scenario** tab, enter the number of Vusers you want to simulate.
3. Optionally, set the **ramp-up** (how quickly Vusers start) and **test duration**.
4. **Run the test**, and LoadRunner will simulate the specified Vusers.

**3. What is Correlation?**

🡪 **Correlation** in LoadRunner is the process of handling **changing values** (like session IDs or tokens) in your script. These values change each time a user interacts with the application.

🡪 You use correlation to capture these values during recording and then replace them with variables so the script can work for multiple users or test runs.

🡪 correlation helps make your test scripts **dynamic** and **reliable** by handling values that change.

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

🡪 The process for developing a Vuser Script in LoadRunner involves these simple steps:

1. **Record the Script:**
   * Use VuGen (Virtual User Generator) to record the actions of a real user on the application (like logging in, filling a form, etc.).
2. **Enhance the Script:**
   * Add checks and validation (e.g., to ensure a page loads correctly).
   * Parameterize dynamic data (e.g., replace usernames or session IDs with variables).
   * Use Correlation to handle dynamic values that change with each session.
3. **Replay the Script:**
   * Run the script in VuGen to check if it works correctly and behaves like a real user.
4. **Debug if Needed:**
   * If there are errors or issues, debug the script to fix them.
5. **Save the Script:**
   * Save the script, and it's ready to be used in the LoadRunner Controller for performance testing.

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

🡪 LoadRunner interacts with the application by **simulating virtual users (Vusers)**. Here's how it works:

1. **Record Actions**: It records a real user's actions on the app (like logging in or clicking buttons).
2. **Simulate Users**: It plays back the recorded actions with multiple **Vusers**.
3. **Send Requests**: Vusers send requests to the app, just like real users do.
4. **Measure Performance**: LoadRunner checks how the app responds (e.g., speed, errors) under load.

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

🡪 The number of **Vusers** needed for load testing depends on several factors:

1. **Expected Real Users**: Simulate the number of users you expect to use the app at once.
2. **Peak Load**: Test for the highest traffic the app might experience.
3. **Test Scenario**: The complexity of actions (e.g., browsing vs. transactions) affects the number of Vusers.
4. **Performance Goals**: More Vusers are needed if the goal is to test how the app performs under heavy load or stress.

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

🡪 **Response Time** is how quickly the application responds, while **Throughput** is how many requests it can handle in a certain time

* As **Throughput** increases (more users or requests), **Response Time** often increases (slower response), because the system gets busier.
* The goal is to balance both: high throughput with low response time for better performance.