JMETER

* JMeter is the best open-source load testing tool to measure the performance of an application.

**Difference between load test vs stress test vs endurance test vs scalability test**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| | **Test Type** | **Purpose** | **Key Focus** | **Real-World Example** | | --- | --- | --- | --- | | **Load Testing** | Ensures system stability under expected user load | Measures response time, resource usage, and performance under normal conditions | Example: An **e-commerce website** like Amazon runs load tests before a sale event to ensure smooth performance when thousands of users browse and purchase items. | | **Stress Testing** | Identifies system limits under extreme conditions | Determines breaking points, error handling, and recovery mechanisms | Example: A **banking app** undergoes stress testing to simulate peak loads (e.g., tax filing day) and ensure the system doesn't crash. | | **Endurance Testing** | Verifies long-term stability under sustained load | Detects performance degradation, memory leaks, or slowdowns over time | Example: A **social media platform** like Instagram runs endurance tests to check if performance remains stable after days of continuous activity. | | **Scalability Testing** | Evaluates system ability to handle growth | Tests system behavior under increasing users, data, and infrastructure | Example: A **streaming service** like Netflix performs scalability testing to ensure it can handle millions of new users signing up and watching videos simultaneously. |   **How These Tests Help Businesses**   * **Load Testing** ensures users get a smooth experience under normal traffic conditions. * **Stress Testing** helps businesses avoid crashes and downtime under extreme usage. * **Endurance Testing** ensures applications remain reliable even after long usage. * **Scalability Testing** confirms platforms can expand efficiently as demand grows. |

**Diff btw Web server, App server, Database server**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| | **Server Type** | **Purpose** | **Key Function** | **Real-World Example** | | --- | --- | --- | --- | | **Web Server** | Handles HTTP requests from clients (browsers) | Serves static content (HTML, CSS, images) and forwards dynamic requests to an application server | Example: A user visits **Amazon** to browse products. The web server delivers the homepage with images and CSS styling. | | **Application Server** | Runs business logic and processes dynamic requests | Handles application execution, logic processing, and interacts with the database server | Example: The user adds items to their cart and proceeds to checkout on **Amazon**. The application server processes the order details and calculates the total price. | | **Database Server** | Manages and stores structured data | Processes queries, stores/retrieves data, ensures data integrity and security | Example: When the user confirms the purchase, the **Amazon database server** stores customer details, product inventory, and payment transactions securely. | |

**How They Work Together in Real-Time:**

1. **Step 1: Web Server** serves the website content when a user visits the online store.
2. **Step 2: Application Server** processes dynamic interactions, like adding items to a cart or calculating discounts.
3. **Step 3: Database Server** ensures all customer and order details are securely stored and retrieved.

These servers collaborate seamlessly to provide a smooth experience for users on platforms like Amazon, Netflix, or any banking application.

**Components:**

**1)Thread Group**

* Thread Groups is a collection of Threads.
* Basically, each Thread simulates one real user request to the server.

2)Sampler

* We already know that Thread Groups simulate user request to the server
* But how does a Thread Group know which type of requests (HTTP, FTP etc.) it needs to make?
* The answer is Samplers
* The user request could be FTP Request, HTTP Request, JDBC Request…Etc.

3) **Configuration Elements**

* set up defaults and variables for later use by samplers.

4) **Listeners**

* Listeners: shows the results of the test execution.
* They can show results in a different format such as a tree, table, graph or log file