**Performance Testing**

* **Definition**: A broad term that evaluates how well a system performs under various conditions.
* **Objective**: To assess system behavior, responsiveness, and stability under different scenarios.
* **Focus Areas**:
  + Speed (response time)
  + Scalability (how well the system scales with increasing workload)
  + Stability (behavior under sustained load)
* **Examples**:
  + Testing response times for a website under normal conditions.
  + Checking API response times under different network conditions.

Performance testing includes various subtypes, each focusing on different aspects of system performance. Here's an overview of testing types under performance testing and their compatibility with **JMeter**:

**1. Load Testing**

* **Purpose**: Validate system performance under expected user load.
* **Example**: Simulating 1,000 concurrent users accessing a web application.
* **Can Be Done in JMeter?**: ✅ Yes. JMeter is widely used for load testing.

**2. Stress Testing**

* **Purpose**: Determine the system's breaking point by applying extreme load conditions.
* **Example**: Pushing the system with 10,000+ users to see when it crashes.
* **Can Be Done in JMeter?**: ✅ Yes. You can incrementally increase load to find the limits.

**3. Endurance Testing (Soak Testing)**

* **Purpose**: Test system stability and performance under a sustained load over a long period.
* **Example**: Running 500 users for 8 hours to check memory leaks or resource exhaustion.
* **Can Be Done in JMeter?**: ✅ Yes. JMeter supports long-running tests to observe system endurance.

**4. Spike Testing**

* **Purpose**: Assess system behavior when load suddenly increases or decreases.
* **Example**: Simulating a sudden surge of 5,000 users, then dropping back to 100 users.
* **Can Be Done in JMeter?**: ✅ Yes. Use custom Thread Groups or scripts for spike scenarios.

**5. Scalability Testing**

* **Purpose**: Evaluate how the system handles increasing user load while scaling resources.
* **Example**: Testing a system with 500, 1,000, and 5,000 users to see if it scales effectively.
* **Can Be Done in JMeter?**: ✅ Yes. Use variable thread counts to simulate increasing load.

**6. Volume Testing**

* **Purpose**: Test the system with large volumes of data.
* **Example**: Uploading 1 million records to a database and checking response time.
* **Can Be Done in JMeter?**: ✅ Yes. JMeter can handle bulk data operations with appropriate configurations.

**7. Failover Testing**

* **Purpose**: Check system performance when components fail and recovery mechanisms activate.
* **Example**: Simulating a server failure during a high load test to test load balancers.
* **Can Be Done in JMeter?**: ⚠️ Partially. JMeter can generate load, but simulating failures requires integration with external tools or manual interventions.

**8. Configuration Testing**

* **Purpose**: Assess system performance with different configurations (e.g., server settings, hardware).
* **Example**: Testing the impact of changing cache size on response time.
* **Can Be Done in JMeter?**: ✅ Yes. Use JMeter to simulate traffic for various configurations.

**9. Latency Testing**

* **Purpose**: Measure delay in data communication and assess impact on performance.
* **Example**: Checking API latency under load.
* **Can Be Done in JMeter?**: ✅ Yes. JMeter provides detailed latency metrics.

**Summary of Tests JMeter Supports**

| **Testing Type** | **Can Be Done in JMeter?** |
| --- | --- |
| Load Testing | ✅ Yes |
| Stress Testing | ✅ Yes |
| Endurance Testing | ✅ Yes |
| Spike Testing | ✅ Yes |
| Scalability Testing | ✅ Yes |
| Volume Testing | ✅ Yes |
| Failover Testing | ⚠️ Partially |
| Configuration Testing | ✅ Yes |
| Latency Testing | ✅ Yes |

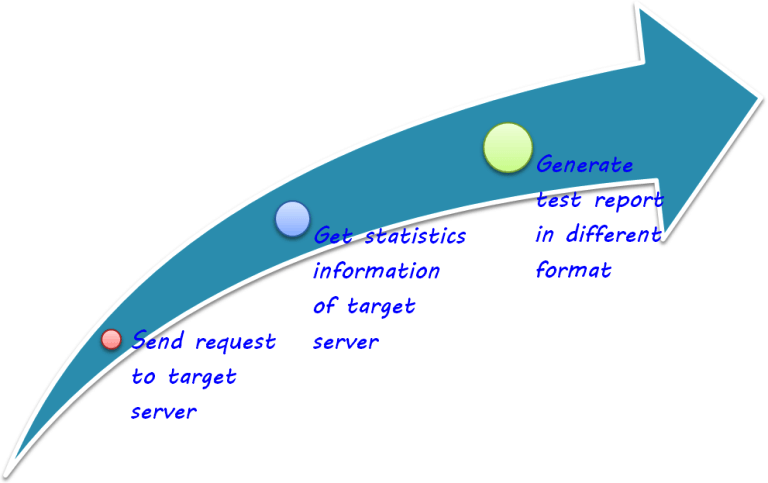
**How JMeter Supports These Tests**

* **Thread Groups**: Define user load and test duration.
* **Timers and Controllers**: Simulate varying load patterns (spike, endurance).
* **Listeners**: Analyze results, including latency, throughput, and error rates.
* **Integrations**: Combine JMeter with CI/CD tools for automated performance testing.

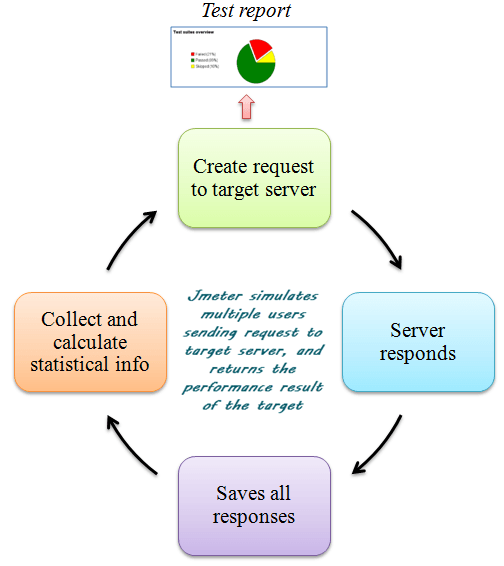
**How does JMeter work?**

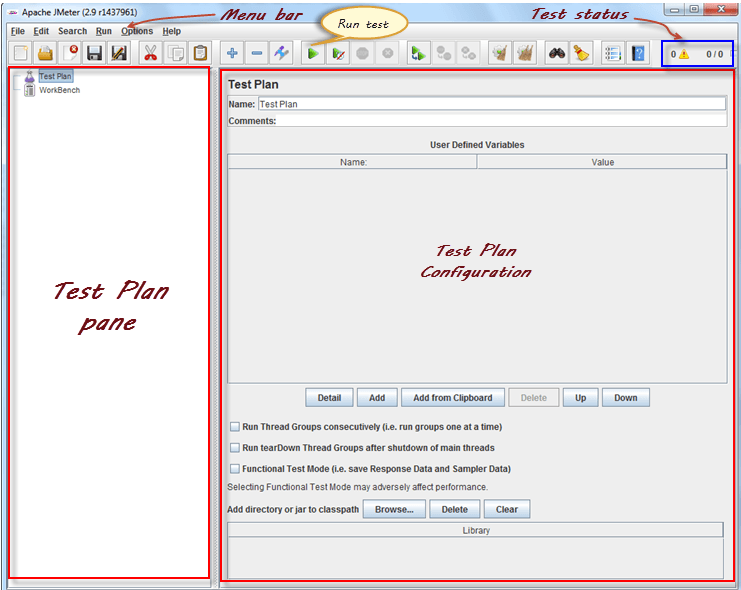
The basic workflow of JMeter as shown in the figures below

JMeter simulates a group of users sending requests to a target server, and return statistics information of target server through graphical diagrams

[](https://www.guru99.com/images/IMG6(2).png)

The completed workflow of JMeter as shown in the figure below

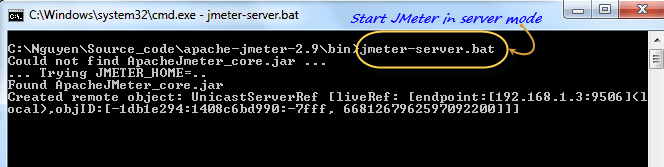
[](https://www.guru99.com/images/IMG7(1).png)



**Start JMeter in Server Mode**

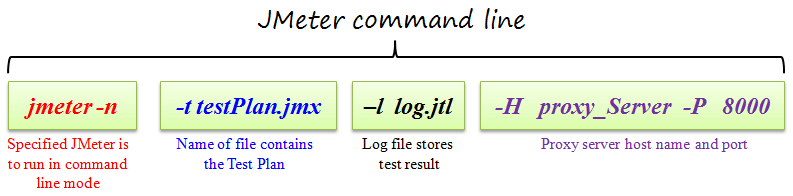
Server mode is used for **distributed** testing. This[Testing](https://www.guru99.com/software-testing.html)works as **a client-server** model. In this model, JMeter runs on a server computer in **server**mode. On a client computer, JMeter runs in **GUI** mode.

To start the server mode, you run the bat file bin\**jmeter-server.bat**as below figure

[](https://www.guru99.com/images/ApacheJmeterServer.png)

**Start JMeter in command line mode**

JMeter in GUI mode consumes much computer memory. For saving the resource, you may choose to run JMeter without the GUI. To do so, use the following command options

[](https://www.guru99.com/images/ApacheJmeterCommandLine.png)

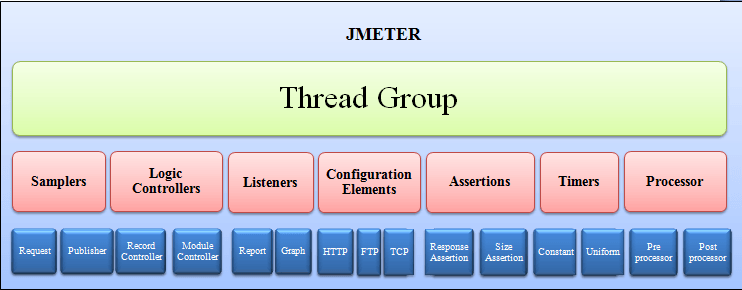
This is a command line example

$jmeter -n -t testPlan.jmx - l log.jtl -H 127.0.0.1 -P 8000

**What is Element in JMeter?**

The different components of JMeter are called Elements. Each Element is designed for a specific purpose.

The figure below gives some common elements in JMeter.

[](https://www.guru99.com/images/Jmeter.png)

Studying all the components in one go is an invitation to confusion and boredome. Here, we will discuss the must-know components before you can start [Testing](https://www.guru99.com/software-testing.html) in JMeter.

Initially we start with 4 elements to learn

Thread group – number of users

Samplers – type of request user needs to send

Listeners – for type of report to represent

Configurations- variable or cofiguratiosn used by samplers

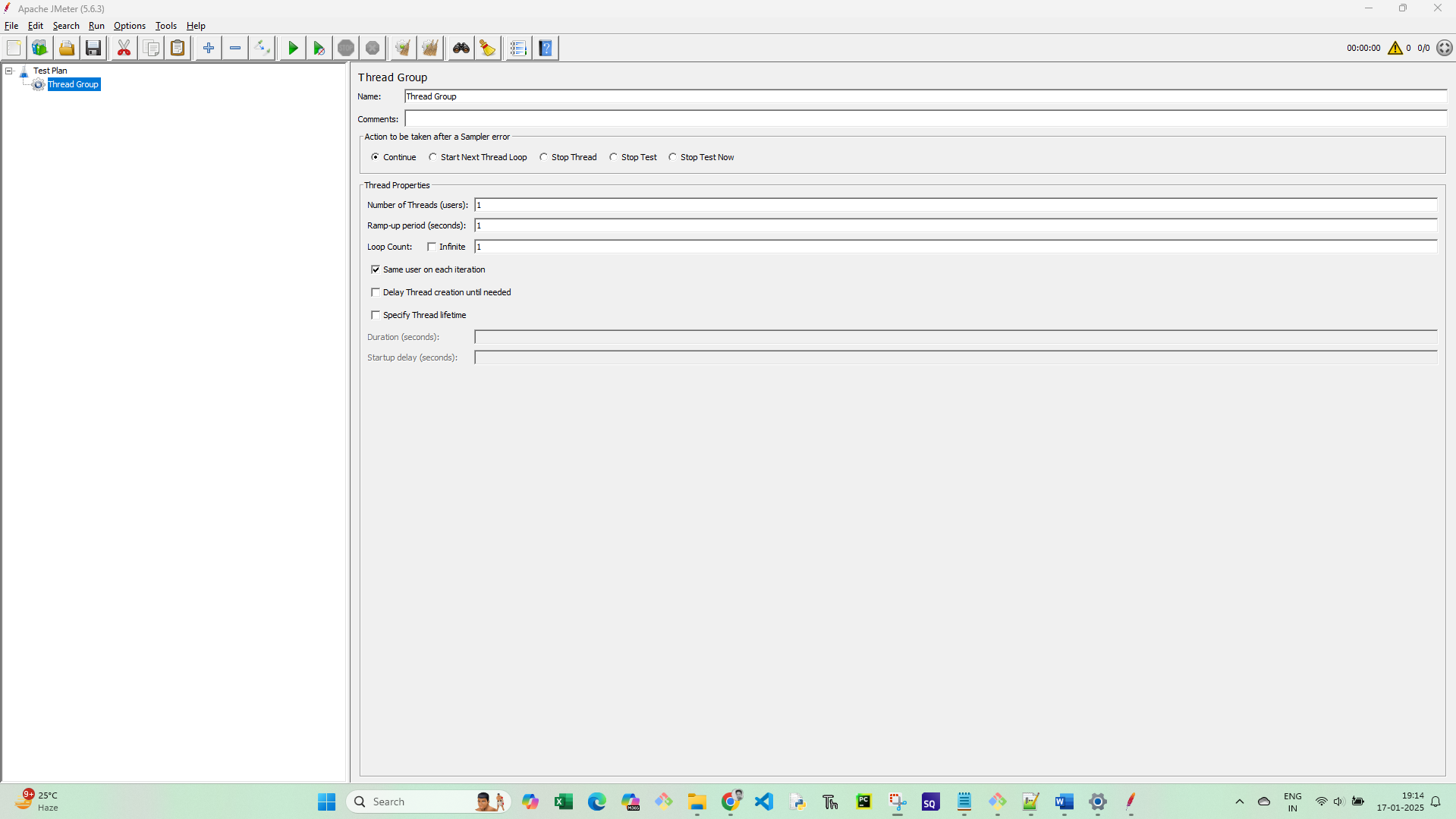
First test :

Start the tool , new test plan will be created

Create a thread group by right click and add

Set a name and set the thread properties

* Number of threads – 5
* Ramp up period – 3 – for every three secods each user request will be sent , like an interval
* Loop – 1- will loopeach time based on the count



Next sampler;

What kinf of request we are sending to the server

We can create the sampler only from the thread group

We start with http request :

Add ip and path

Next listeners:

We just add view results in tree and table

And then save the plan and run it

Check the result it must have pass