



**Department of Computer Science and Engineering
(IoT, Cybersecurity with Blockchain Technology)**
B.Tech. Sem: VII Subject: Software Testing & Quality Assurance

Experiment 8

Name: Hiren Darji

SAP ID: 60019230114

Aim: - Performance Testing Using Apache JMeter.

Steps:

1) Install JMeter.

Download Apache JMeter from https://jmeter.apache.org/download_jmeter.cgi

The screenshot shows the Apache JMeter download page. At the top, there are links for 'About', 'Download', 'Documentation', 'Tutorials', and 'Binaries'. The 'Download' section is highlighted, showing options for 'Download Releases' and 'Release Notes'. Below this, there is a large heading 'Download Apache JMeter'. A note says: 'We recommend you use a mirror to download our release builds, but you must verify the integrity of the downloaded files using signatures downloaded from our main distribution directories. Recent releases (48 hours) may not yet be available from all the mirrors.' It also mentions the current mirror being used and provides a dropdown for selecting other mirrors. The 'Binaries' section at the bottom is also visible.

Click on Download releases

- Download .zip and Extract the ZIP file and navigate to the /bin folder.

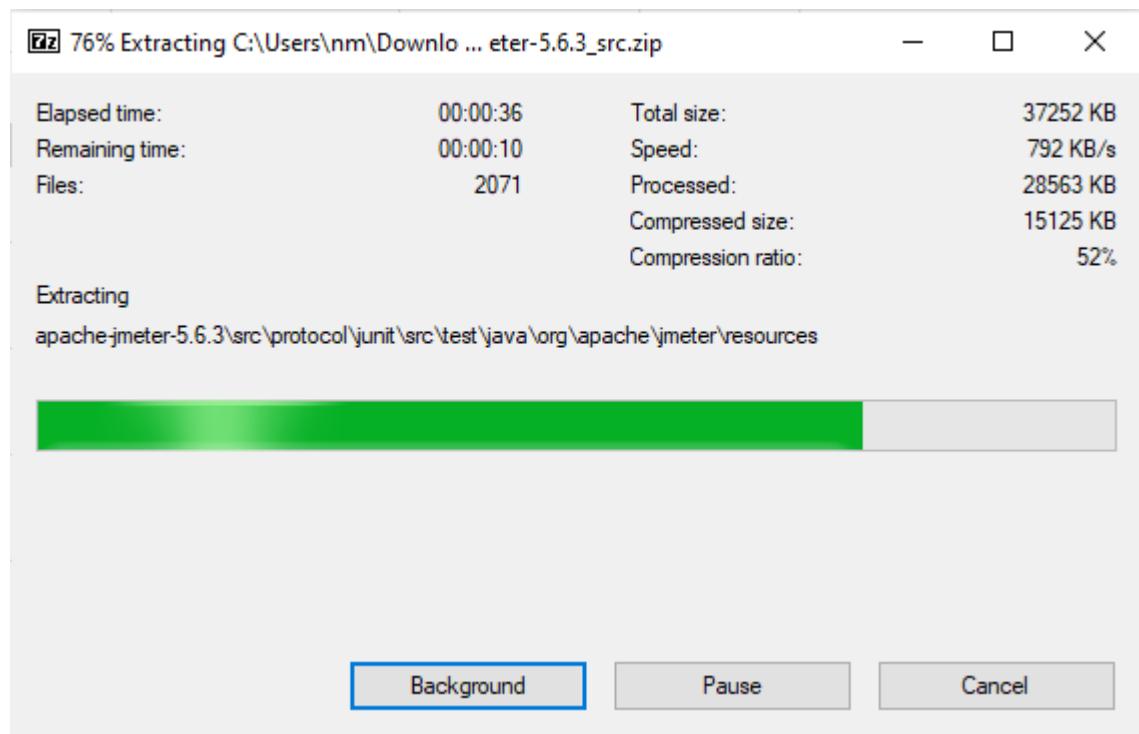


Apache JMeter 5.6.3 (Requires Java 8+)

Binaries

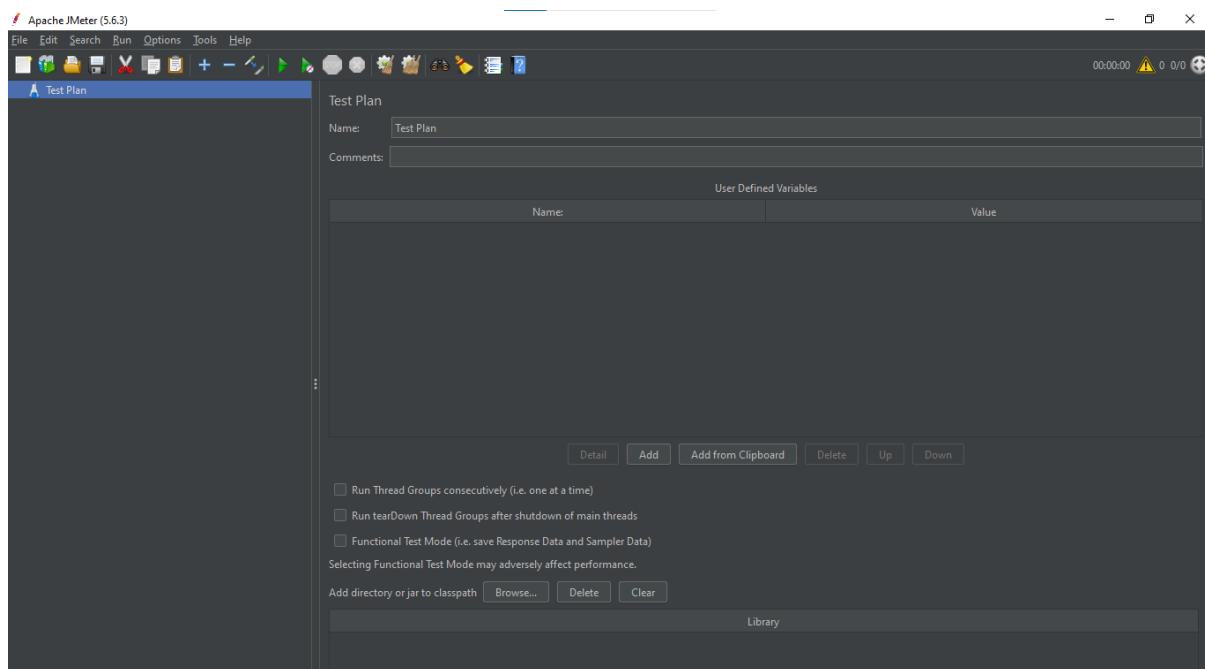
[apache-jmeter-5.6.3.tgz sha512 pgp](#)
[apache-jmeter-5.6.3.zip sha512 pgp](#)

Step 2: Extract the JMeter





Step 3: Download .zip and Extract the ZIP file and navigate to the /bin folder. • Run jmeter.bat

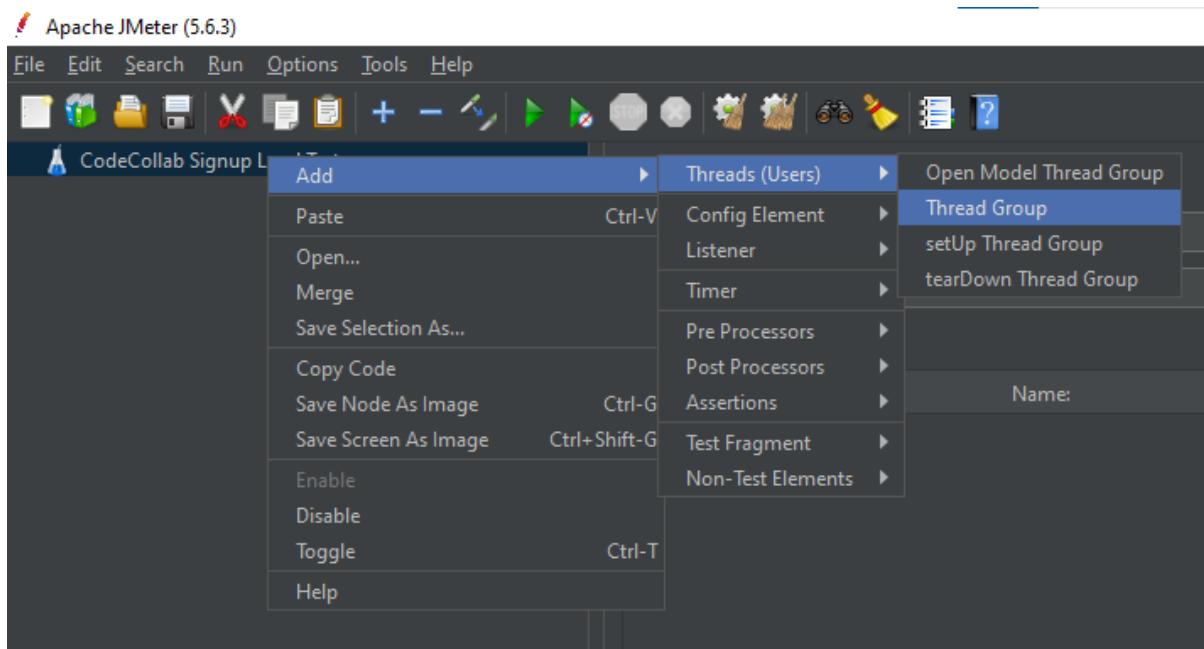


4 — New Test Plan

- Menu: **File → New**
- In left tree, right-click **Test Plan** → **Rename** → e.g. CodeCollab Signup Load Test

5 — Add a Thread Group (virtual users)

- Right-click **Test Plan** → **Add** → **Threads (Users)** → **Thread Group**
- Click **Thread Group** node and set:
 - **Number of Threads (users):** 50 (example)
 - **Ramp-Up Period (seconds):** 10
 - **Loop Count:** 5 (or check **Forever** for long runs)
 - *(This simulates 50 users arriving over 10s, each doing test 5 times.)*



6 — Configure HTTP Request Defaults (so each sampler doesn't repeat host)

- Right-click **Thread Group** → **Add** → **Config Element** → **HTTP Request Defaults**
 - **Server Name or IP:** code-collab-sage.vercel.app
 - **Protocol:** https
 - **Port Number:** leave blank (default for HTTPS)
 - **Path:** leave blank (you'll set path per sampler)
- Save — this reduces typing for each HTTP Request.



Apache JMeter (5.6.3)

File Edit Search Run Options Tools Help

CodeCollab Signup Load Test

Thread Group

Add Sampler

Add Think Times to children Logic Controller

Start Pre Processors

Start no pauses Post Processors

Validate Assertions

Cut Timer

Copy Test Fragment

Paste Ctrl+V Config Element

Duplicate Ctrl+Shift-C Listener

Remove Delete

Open... Add (seconds): 10

Merge Infinite 5

Save Selection As...

Copy Code

Save Node As Image

Save Screen As Image

Enable

Disable

Toggle

Help

oop Stop Thread Stop Test

CSV Data Set Config

HTTP Header Manager

HTTP Cookie Manager

HTTP Cache Manager

HTTP Request Defaults

Bolt Connection Configuration

Counter

DNS Cache Manager

FTP Request Defaults

HTTP Authorization Manager

JDBC Connection Configuration

Java Request Defaults

Keystore Configuration

LDAP Extended Request Defaults

LDAP Request Defaults

Login Config Element

Random Variable

Simple Config Element

TCP Sampler Config

User Defined Variables

HTTP Request Defaults

Name:

Comments:

Basic Advanced

Web Server

Protocol [http]: Server Name or IP: Port Number:

HTTP Request

Path: Content encoding:

Parameters Body Data

Send Parameters With the Request:

Name:	Value	URL Encode?	Content-Type	Include Equals?
<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="text"/>	<input type="checkbox"/>

7 — Add HTTP Cookie Manager & Header Manager

- Right-click **Thread Group** → **Add** → **Config Element** → **HTTP Cookie Manager** (handles cookies like a browser).
- Right-click **Thread Group** → **Add** → **Config Element** → **HTTP Header Manager**
 - Click Header Manager → **Add** a header: Content-Type = application/json (if endpoints accept JSON).
 - Add any auth headers if your app requires tokens already.



Thread Group

- Thread Group
- HTTP Request
- ▼ Add
 - Add
 - Add Think Times to children
 - Start
 - Start no pauses
 - Validate
 - Cut Ctrl-X
 - Copy Ctrl-C
 - Paste Ctrl-V
 - Duplicate Ctrl+Shift-C
 - Remove Delete
 - Open...
 - Merge
 - Save Selection As...
 - Copy Code
 - Save Node As Image Ctrl-G
 - Save Screen As Image Ctrl+Shift-G
 - Enable
 - Disable
 - Toggle Ctrl-T
 - Help
- Sampler
- Logic Controller
- Pre Processors
- Post Processors
- Assertions
- Config Element
 - CSV Data Set Config
 - HTTP Header Manager
 - HTTP Cookie Manager
 - HTTP Cache Manager
 - HTTP Request Defaults
 - Bolt Connection Configuration
 - Counter
 - DNS Cache Manager
 - FTP Request Defaults
 - HTTP Authorization Manager
 - JDBC Connection Configuration
 - Java Request Defaults
 - Keystore Configuration
 - LDAP Extended Request Defaults
 - LDAP Request Defaults
 - Login Config Element
 - Random Variable
 - Simple Config Element
 - TCP Sampler Config
 - User Defined Variables
- Timer
- Test Fragment
- Listener
- Specify Thread lifetime
- Count: Infinite
- Same user on each iteration
- Delay Thread creation
- Up delay (seconds):



The screenshot shows the JMeter interface with a context menu open over a 'Thread Group' element. The 'Add' option is highlighted in blue. A secondary context menu has opened under 'Add', showing various JMeter configuration elements. The 'HTTP Request' option is highlighted in blue within this list.

- Thread Group
 - Add
 - Add Think Times to children
 - Start
 - Start no pauses
 - Validate
 - Cut
 - Copy
 - Paste
 - Duplicate
 - Remove
 - Open...
 - Merge
 - Save Selection As...
 - Copy Code
 - Save Node As Image
 - Save Screen As Image
 - Enable
 - Disable
 - Toggle
 - Help
- Sampler
 - Logic Controller
 - Pre Processors
 - Post Processors
 - Assertions
 - Timer
 - Test Fragment
 - Config Element
 - CSV Data Set Config
 - HTTP Header Manager**
 - HTTP Cookie Manager
 - HTTP Cache Manager
 - HTTP Request Defaults
 - Bolt Connection Configuration
 - Counter
 - DNS Cache Manager
 - FTP Request Defaults
 - HTTP Authorization Manager
 - JDBC Connection Configuration
 - Java Request Defaults
 - Keystore Configuration
 - LDAP Extended Request Defaults
 - LDAP Request Defaults
 - Login Config Element
 - Random Variable
 - Simple Config Element
 - TCP Sampler Config
 - User Defined Variables

8 — Add HTTP Request Sampler(s)

- Right-click **Thread Group** → **Add** → **Sampler** → **HTTP Request**
 - **Name:** Signup Request
 - **Method:** POST (or GET depending endpoint)
 - **Path:** / or specific endpoint, e.g. /api/v1/users (use your actual sign-up endpoint)
 - **Body Data:** paste JSON payload, e.g.

```
{  
  "displayName": "${name}",  
  "email": "${email}",  
  "password": "${password}"  
}
```
 - If using form data, use **Parameters** table instead of Body Data.



The screenshot shows the JMeter interface with a context menu open over a 'Thread Group' element. The menu path is 'Add > Sampler > HTTP Request'. Other options visible in the menu include Logic Controller, Pre Processors, Post Processors, Assertions, Timer, Test Fragment, Config Element, Listener, and various other samplers like JDBC Request, JMS Point-to-Point, etc.

The screenshot shows the 'HTTP Request' configuration dialog in JMeter. The 'Basic' tab is selected. The 'Name' field is set to 'Signup Request'. The 'Protocol' dropdown is set to 'http'. The 'Server Name or IP' field contains 'https://code-collab-sage.vercel.app'. The 'Path' field is set to '/signup'. The 'Content encoding' field is empty. Below these fields, there are checkboxes for 'Redirect Automatically', 'Follow Redirects', 'Use KeepAlive', 'Use multipart/form-data', and 'Browser-compatible headers'. The 'Body Data' tab is selected, showing a JSON payload:

```
1 {
2   "displayName": "${name}",
3   "email": "${email}",
4   "password": "${password}"
5 }
```

9 — Add Assertions (verify success/failure)

- Right-click **HTTP Request** → **Add** → **Assertions** → **Response Assertion**
 - Choose **Response Text** contains expected success text (e.g., "user created" or success code).
 - This allows JMeter to mark failures where functionally incorrect.



The screenshot shows the JMeter interface with a 'Signup Request' thread group selected. A context menu is open, with the 'Listener' option highlighted. A submenu for 'Listener' is displayed, listing various assertion types: JSON Assertion, Size Assertion, JSR223 Assertion, XPath2 Assertion, Compare Assertion, Duration Assertion, HTML Assertion, JSON JMESPath Assertion, MD5Hex Assertion, SMIME Assertion, XML Assertion, XML Schema Assertion, XPath Assertion, and BeanShell Assertion. The 'JSON Assertion' option is currently selected.

10 — Add Listeners (to view results)

- Right-click **Thread Group** → **Add** → **Listener** → **View Results Tree** (debugging only)
- Add **Summary Report**, **Aggregate Report**, **Graph Results**, **Response Time Graph**.
 - Important: use **Summary / Aggregate Report** for summary metrics. Avoid **View Results Tree** for heavy loads (it stores full response content and eats memory).

The screenshot shows the JMeter interface with a 'Response' component selected. A context menu is open, with the 'Listener' option highlighted. A submenu for 'Listener' is displayed, listing various reporting options: View Results Tree, Summary Report, Aggregate Report, Backend Listener, Aggregate Graph, Assertion Results, Comparison Assertion Visualizer, Generate Summary Results, Graph Results, JSR223 Listener, Mailer Visualizer, Response Time Graph, Save Responses to a file, Simple Data Writer, View Results in Table, and BeanShell Listener. The 'View Results Tree' option is currently selected.

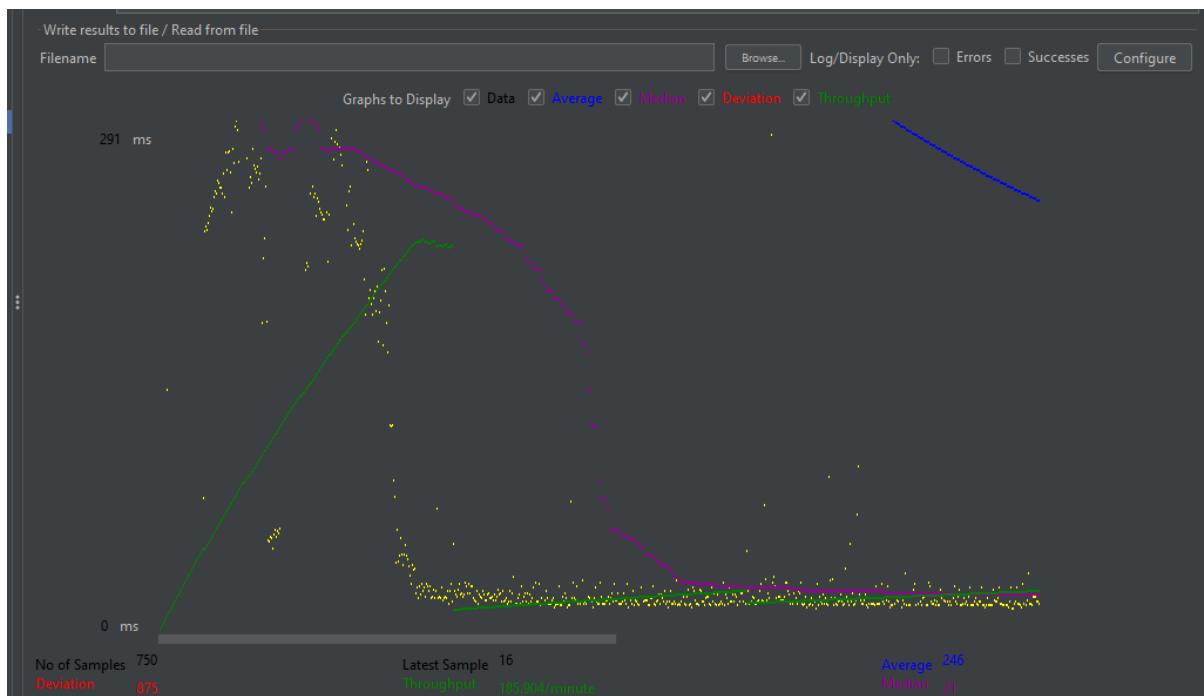


11— Save and Run Tests

Run in GUI (for building & debug)

- Click the green Start (►) button in toolbar.
- Watch results in Listeners. For moderate loads GUI is OK.

Response Graph



Results Tree Table

Sample #	Start Time	Thread Name	Label	Sample Time(...)	Status	Bytes	Sent Bytes	Latency	Connect Time(...)
1	18:26:22.984	Thread Group ...	Signup Request	5900	✗	103	271	5730	5613
2	18:26:27.742	Thread Group ...	Signup Request	1141	✗	103	271	1009	855
3	18:26:27.140	Thread Group ...	Signup Request	1744	✗	103	271	1574	1457
4	18:26:26.544	Thread Group ...	Signup Request	2340	✗	103	271	2170	2053
5	18:26:24.350	Thread Group ...	Signup Request	4535	✗	103	271	4401	4247
6	18:26:28.542	Thread Group ...	Signup Request	341	✗	103	271	209	55
7	18:26:28.747	Thread Group ...	Signup Request	138	✗	103	271	36	29
8	18:26:22.984	Thread Group ...	Signup Request	5899	✗	103	271	5730	5613
9	18:26:23.357	Thread Group ...	Signup Request	5524	✗	103	271	5356	5240
10	18:26:22.984	Thread Group ...	Signup Request	5897	✗	103	271	5766	5615
11	18:26:24.150	Thread Group ...	Signup Request	4730	✗	103	271	4564	4447
12	18:26:25.746	Thread Group ...	Signup Request	3134	✗	103	271	3005	2851
13	18:26:25.945	Thread Group ...	Signup Request	2935	✗	103	271	2806	2652
14	18:26:25.349	Thread Group ...	Signup Request	3531	✗	103	271	3402	3248

Conclusion:

I have successfully performed the complete performance testing of the signup API using Apache JMeter. The test plan was designed with dynamic inputs, appropriate headers, and JSON payloads to simulate real user behavior. Assertions and listeners were added to accurately measure responsiveness, throughput, and success rates of the API. This experiment helped validate the system's stability under load and ensured reliable performance for multiple concurrent user registrations.