

**Batch: B1 Roll No.: 16010421119 Experiment No.: 5 Aim:** To perform performance testing and Load Testing using JMeter.

**Resources needed:** Internet, Apache JMeter

# Theory:

The **Apache JMeter** is pure Java **open source** software, which was first developed by Stefano Mazzocchi of the Apache Software Foundation, designed to load test functional behaviour and measure performance. You can use JMeter to analyze and measure the performance of web application or variety of services. Performance Testing means testing a web application against heavy load, multiple and concurrent user traffic. JMeter originally is used for testing Web Application or FTP application. Nowadays, it is used for functional test, database server test etc.

Apache JMeter may be used to test performance both on static and dynamic resources, Web dynamic applications. It can be used to simulate a heavy load on a server, group of servers, network or object to test its strength or to analyze overall performance under different load types.

# Procedure:

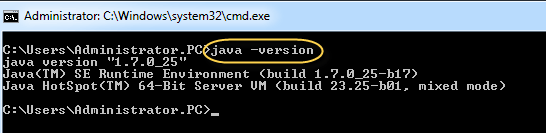
1. **Install Java**

Because JMeter is pure Java desktop application, it requires a fully compliant JVM 6 or higher. You can download and install the latest version of Java SE Development Kit.

Link: <https://www.java.com/en/download/help/windows_manual_download.xml>

After installation is finished, you can use the following procedure to check whether Java JDK is installed successfully in your system

* + In Window/Linux, go to Terminal
  + Enter command java -version

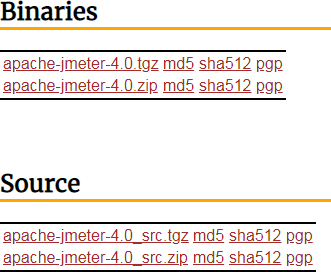
[](https://cdn.guru99.com/images/Administrator.png)If Java runtime environment is installed successfully, you will see the output as figure below

If nothing displays, please re-install Java SE runtime environment

# Download JMeter

As of this writing, the latest version of JMeter is Apache JMeter 4.0. Link: <http://jmeter.apache.org/download_jmeter.cgi>

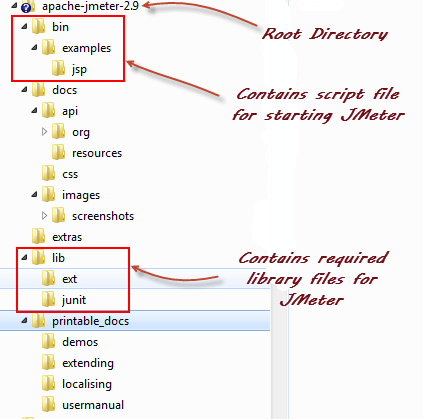
Choose the Binaries file (either zip or tgz) to download as shown in figure below



# Installation

Installation of JMeter is extremely easy and simple. You simply unzip the zip/tar file into the directory where you want JMeter to be installed. There is no tedious installation screen to deal with. Simple unzip and you are done.

Once the unzipping is done installation directory structure should look like as figure below:

[](https://cdn.guru99.com/images/ApacheJmeter2_9.png)

Given below is the description of the JMeter directories and its importance. JMeter directory contains many files and directory

* + /bin: Contains JMeter script file for starting JMeter
  + /docs: JMeter documentation files
  + /extras: ant related extra files
  + /lib/: Contains the required Java library for JMeter
  + /lib/ext: contains the core jar files for JMeter and the protocols
  + /lib/junit: [Junit](https://www.guru99.com/junit-tutorial.html) library used for JMeter
  + /printable\_docs:

# Launch JMeter

You can start JMeter in two modes:

* + GUI Mode
  + Server Mode

Start JMeter in GUI Mode

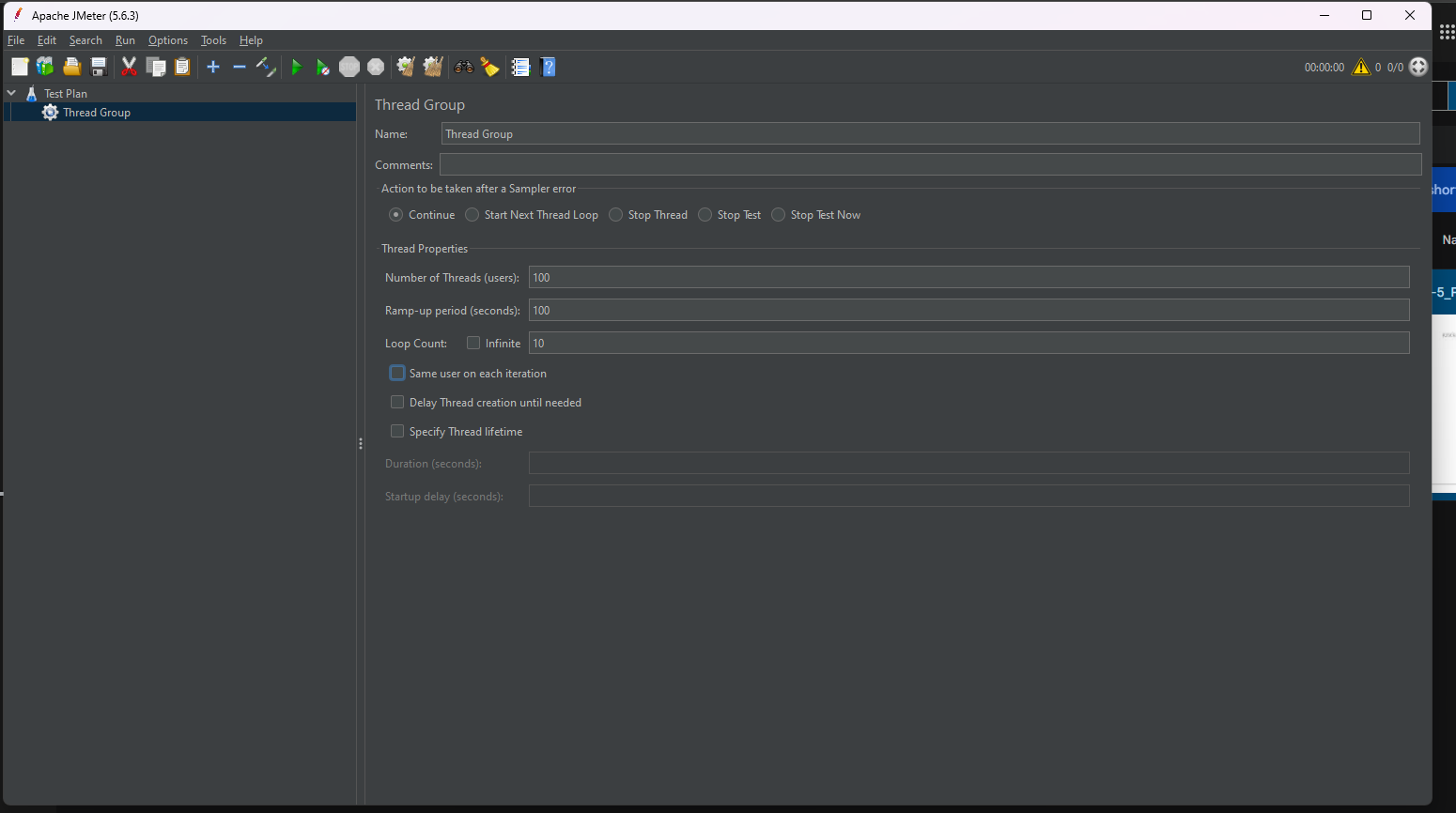
If you are using Window, just run the file /bin/jmeter.bat to start JMeter in GUI mode.

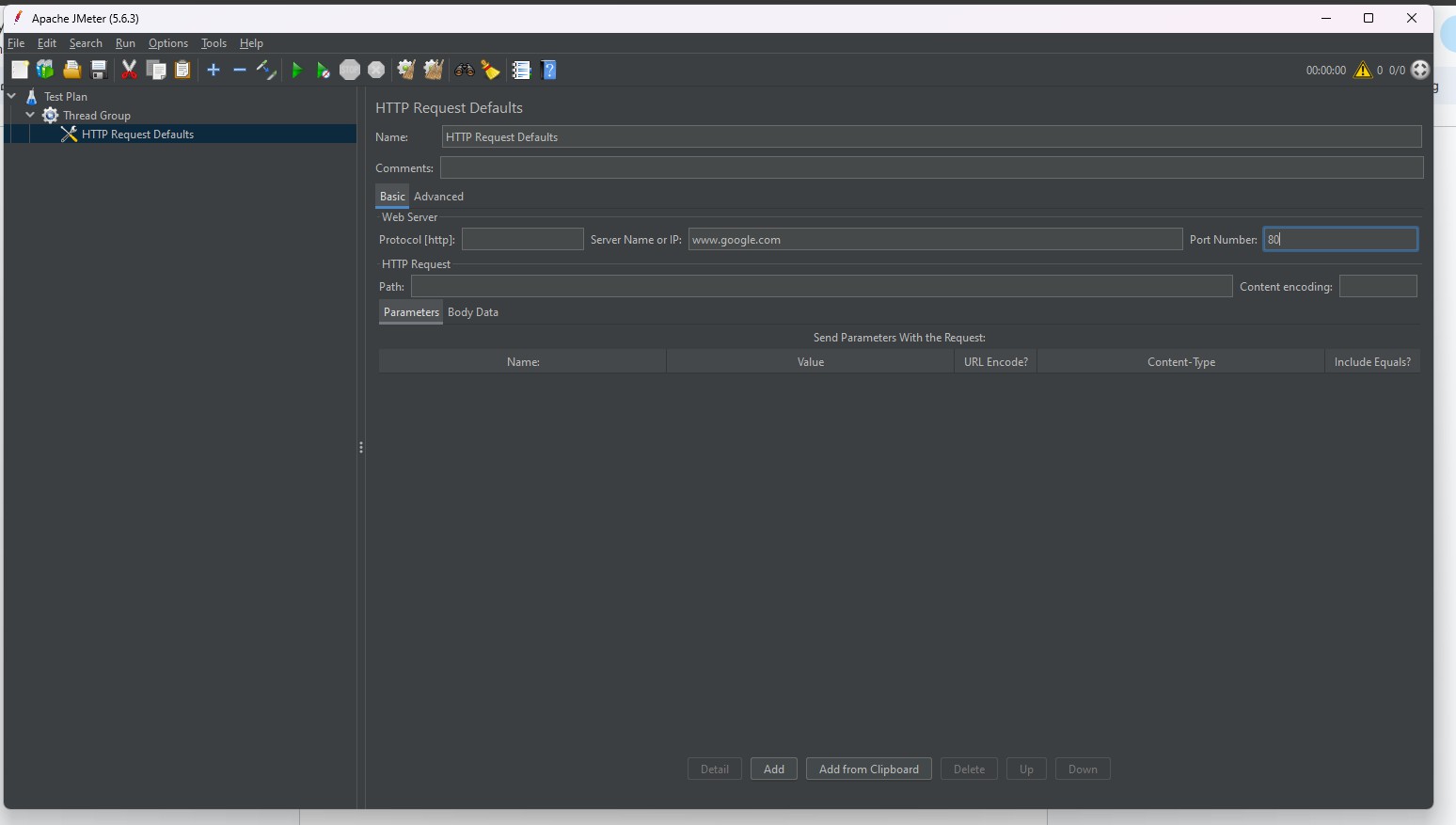
Perform the following on Apache JMeter:

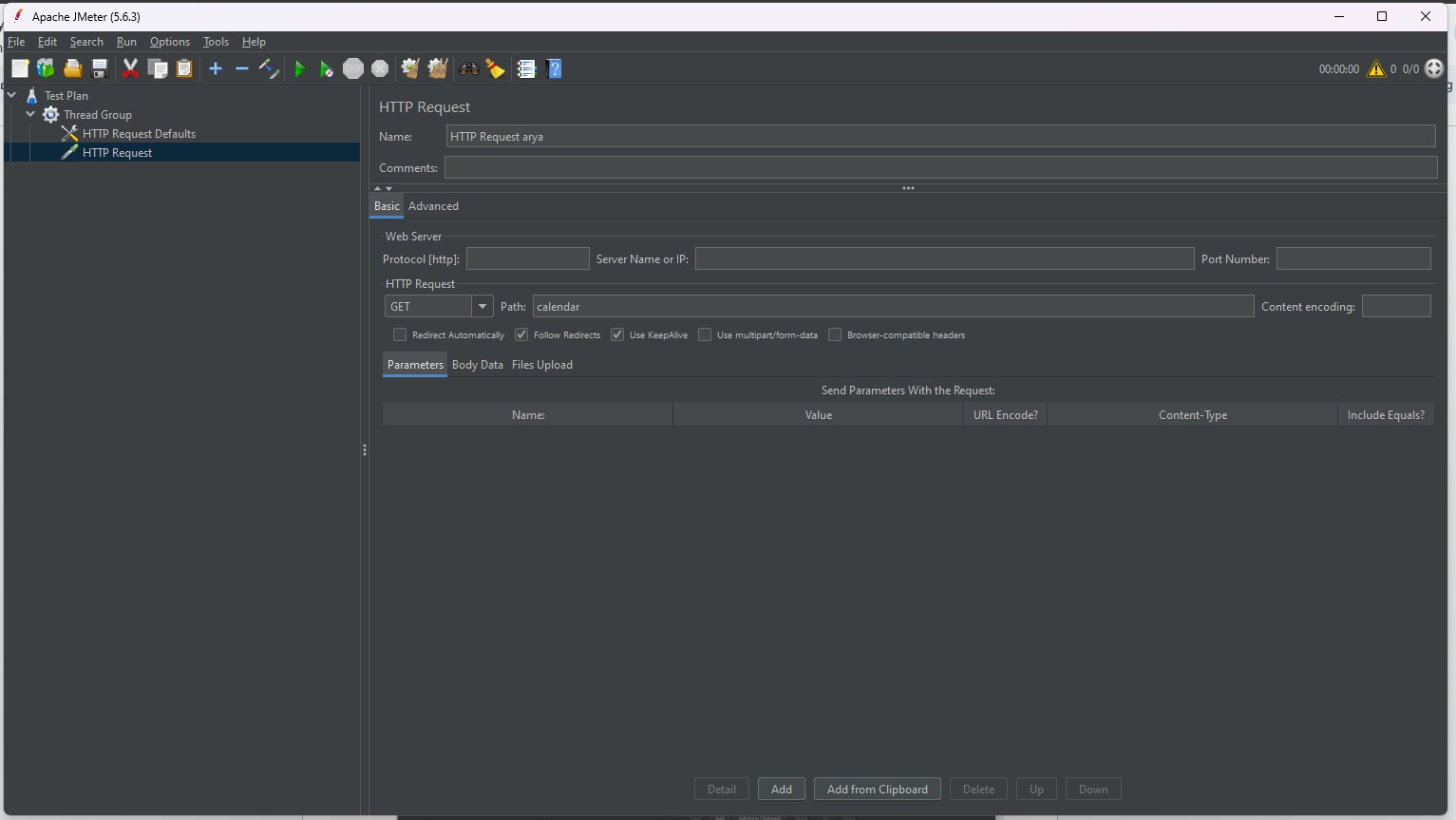
1. Create a Performance Test Plan
2. Use Constant Timer
3. Use Response Assertion
4. Loop Controller
5. Post Processor
6. Explore Load Testing

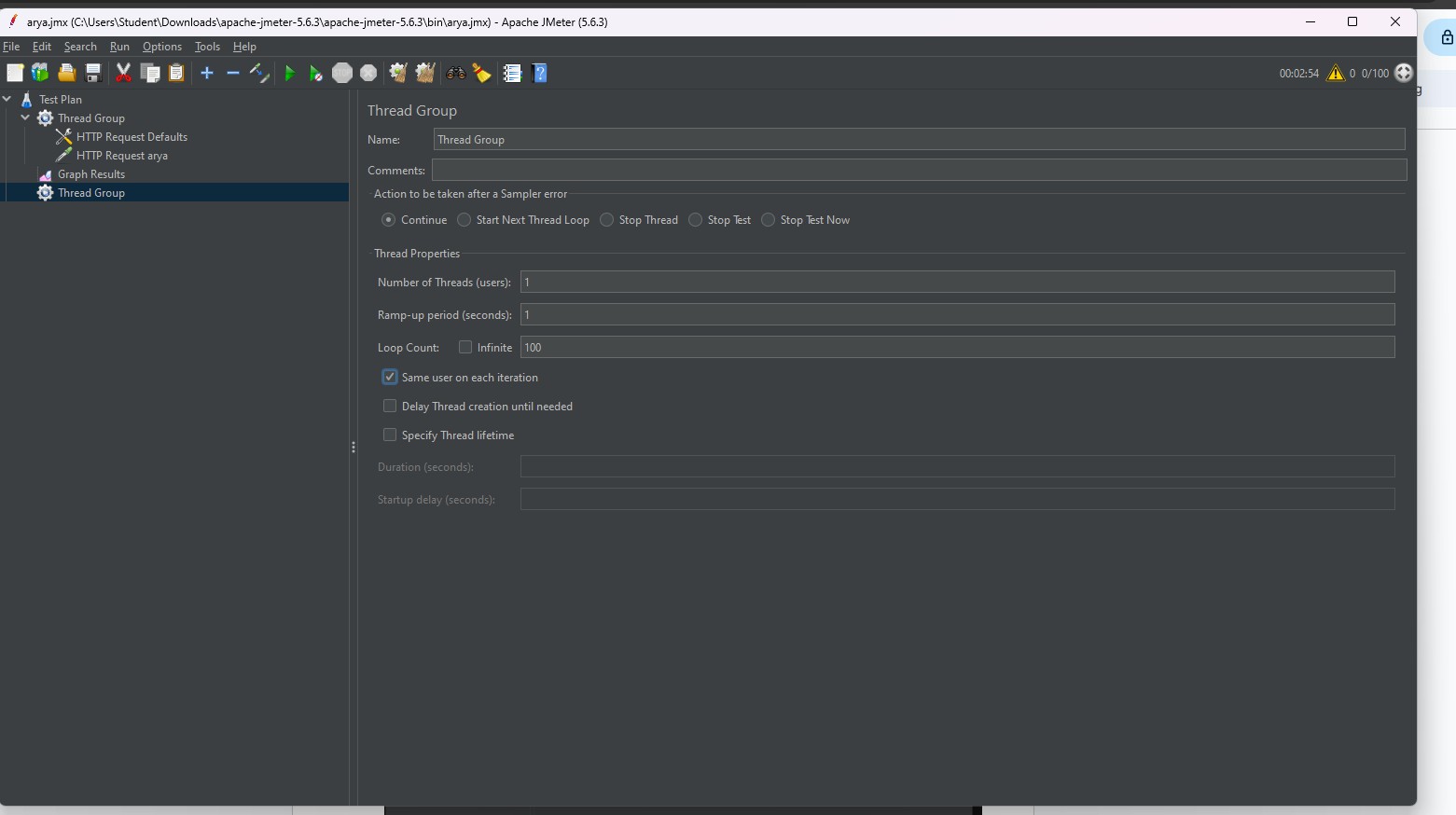
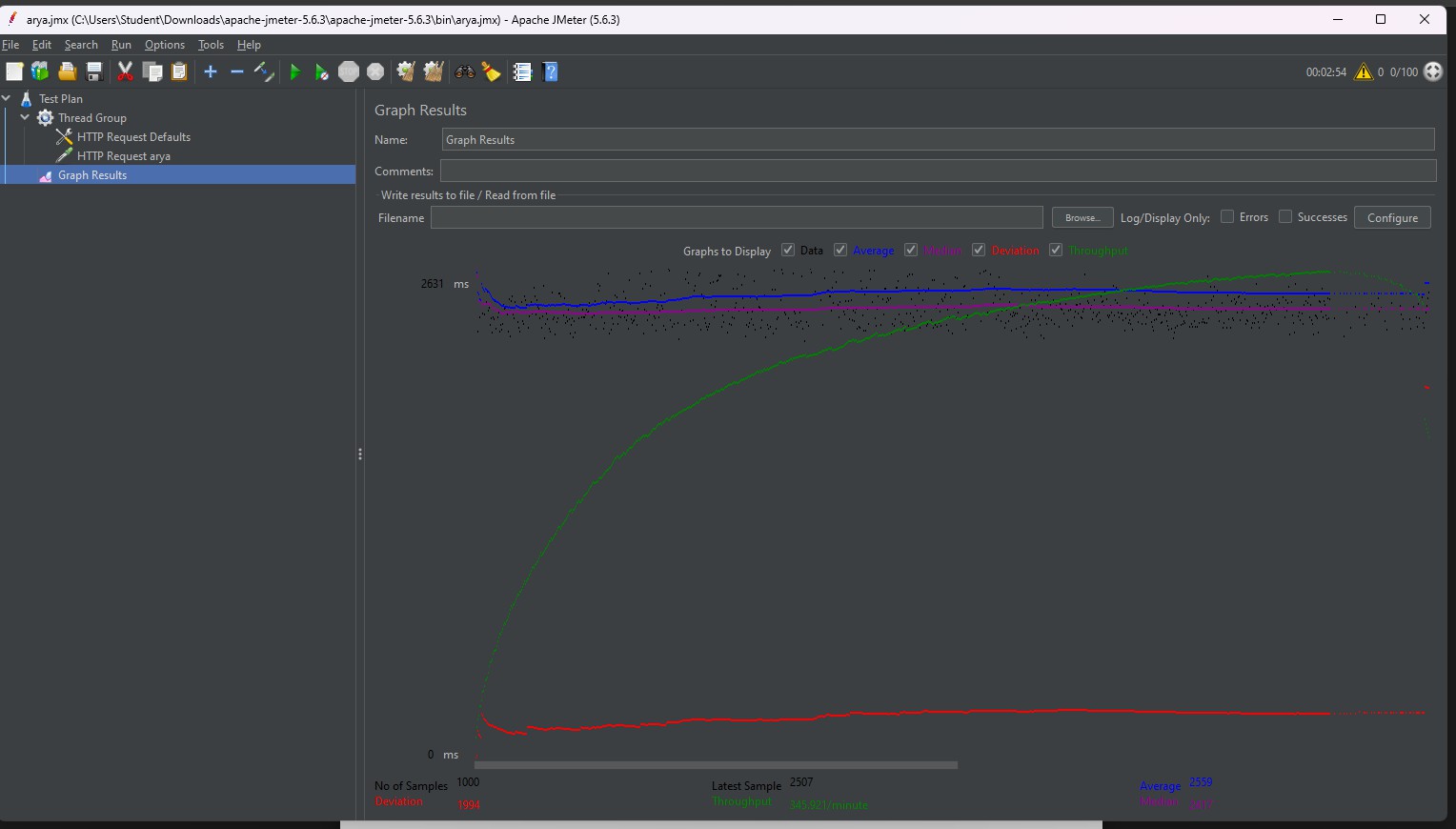
# Results: (Document printout as per the format)

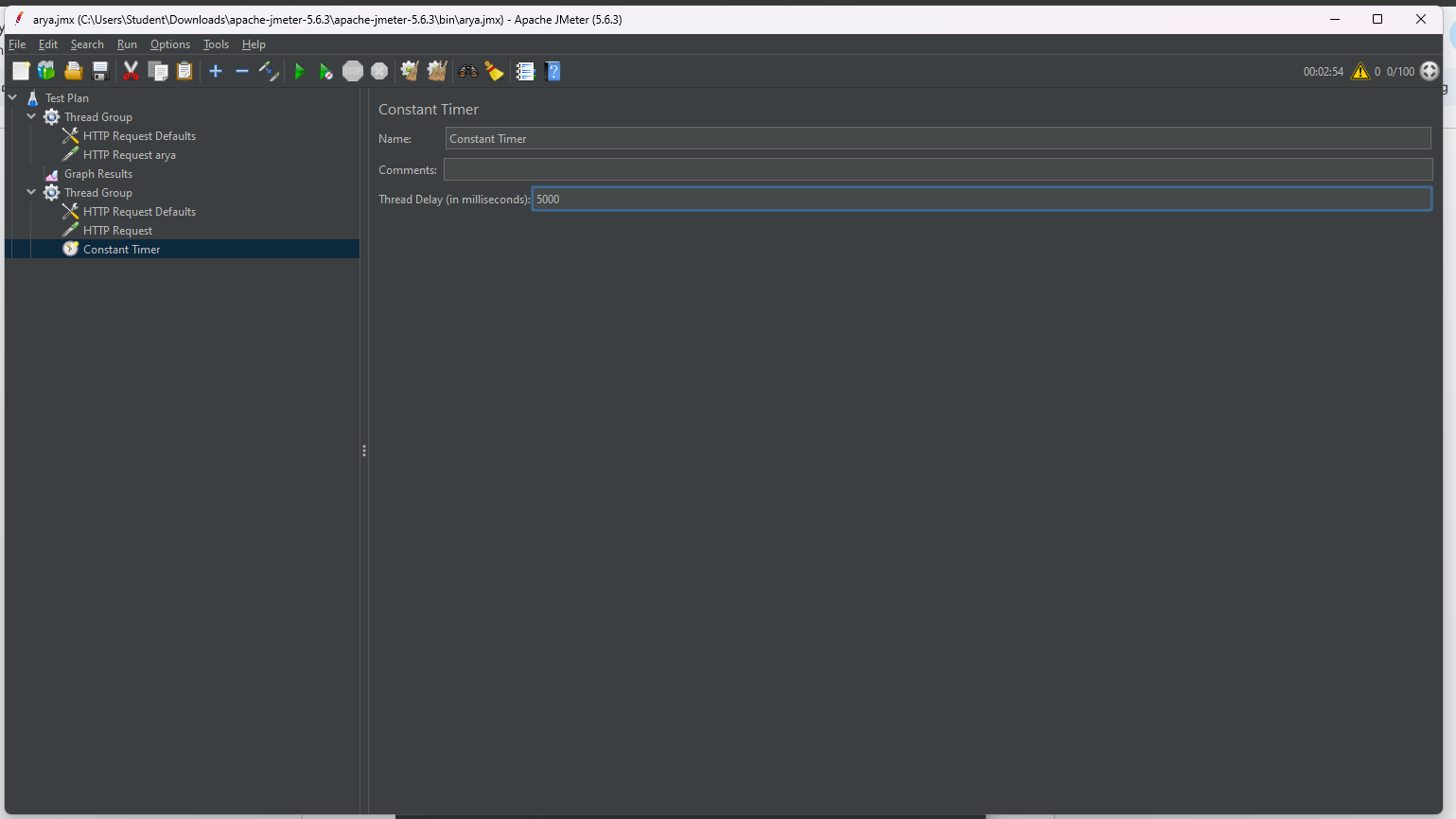


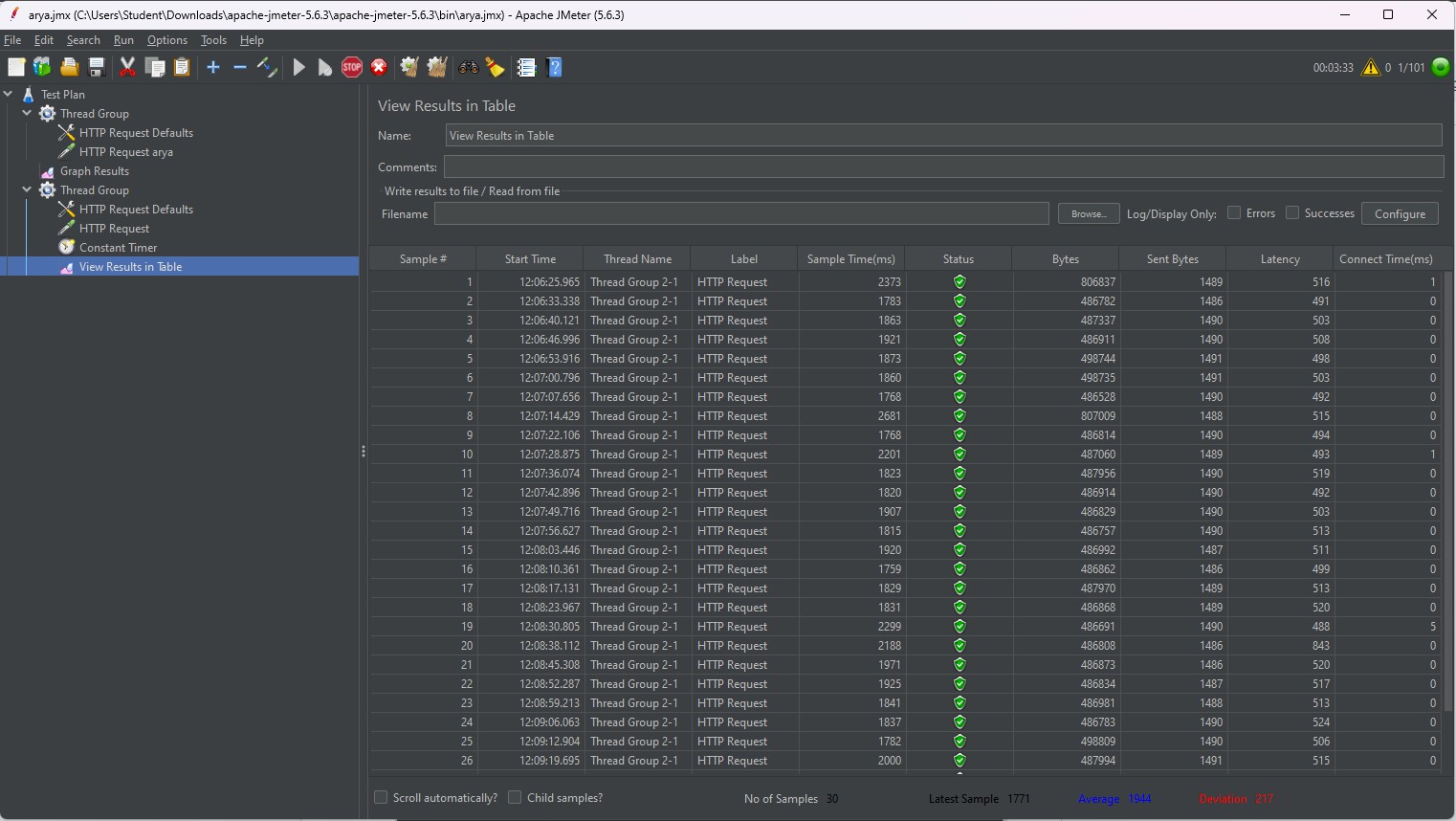


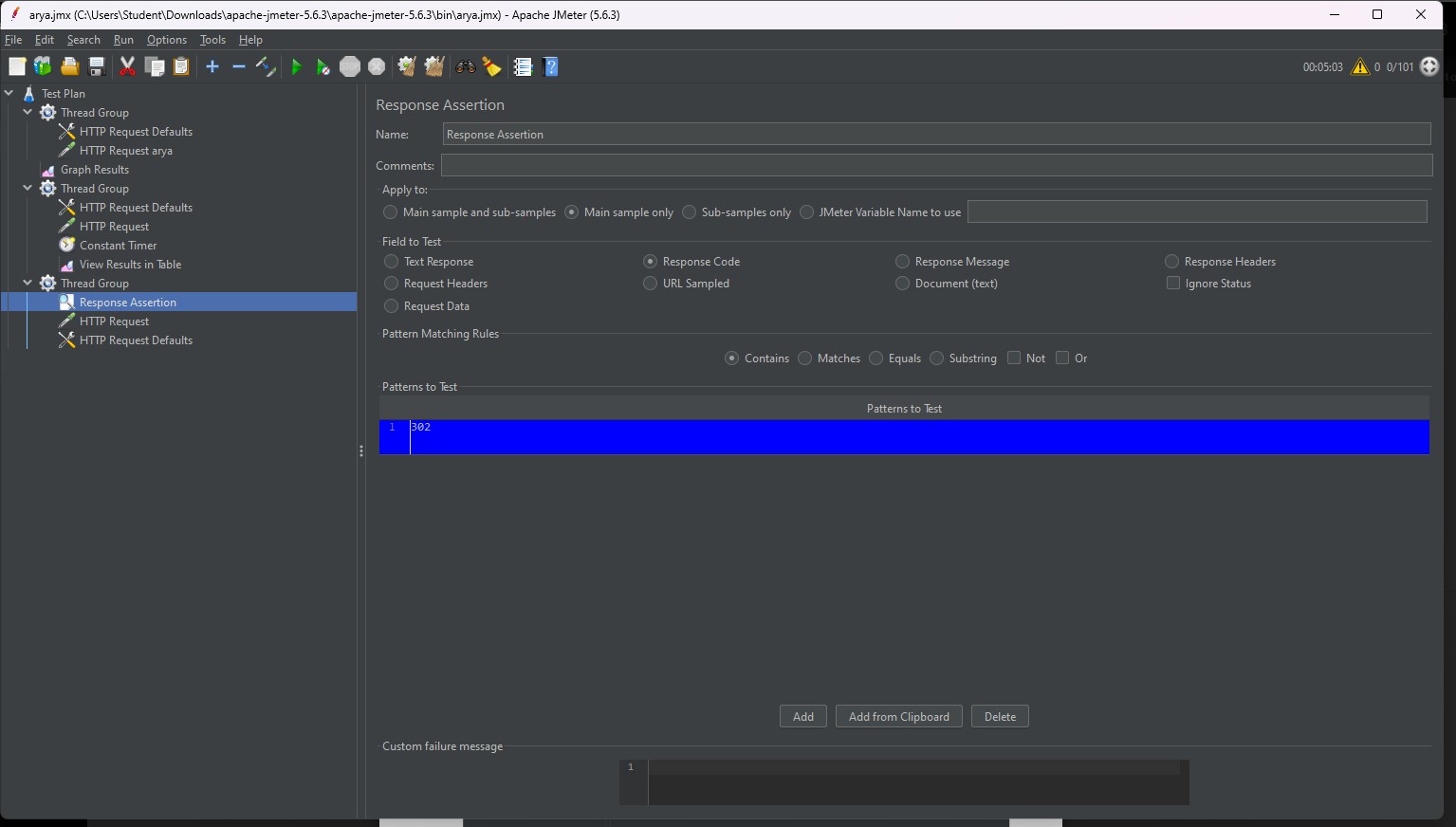


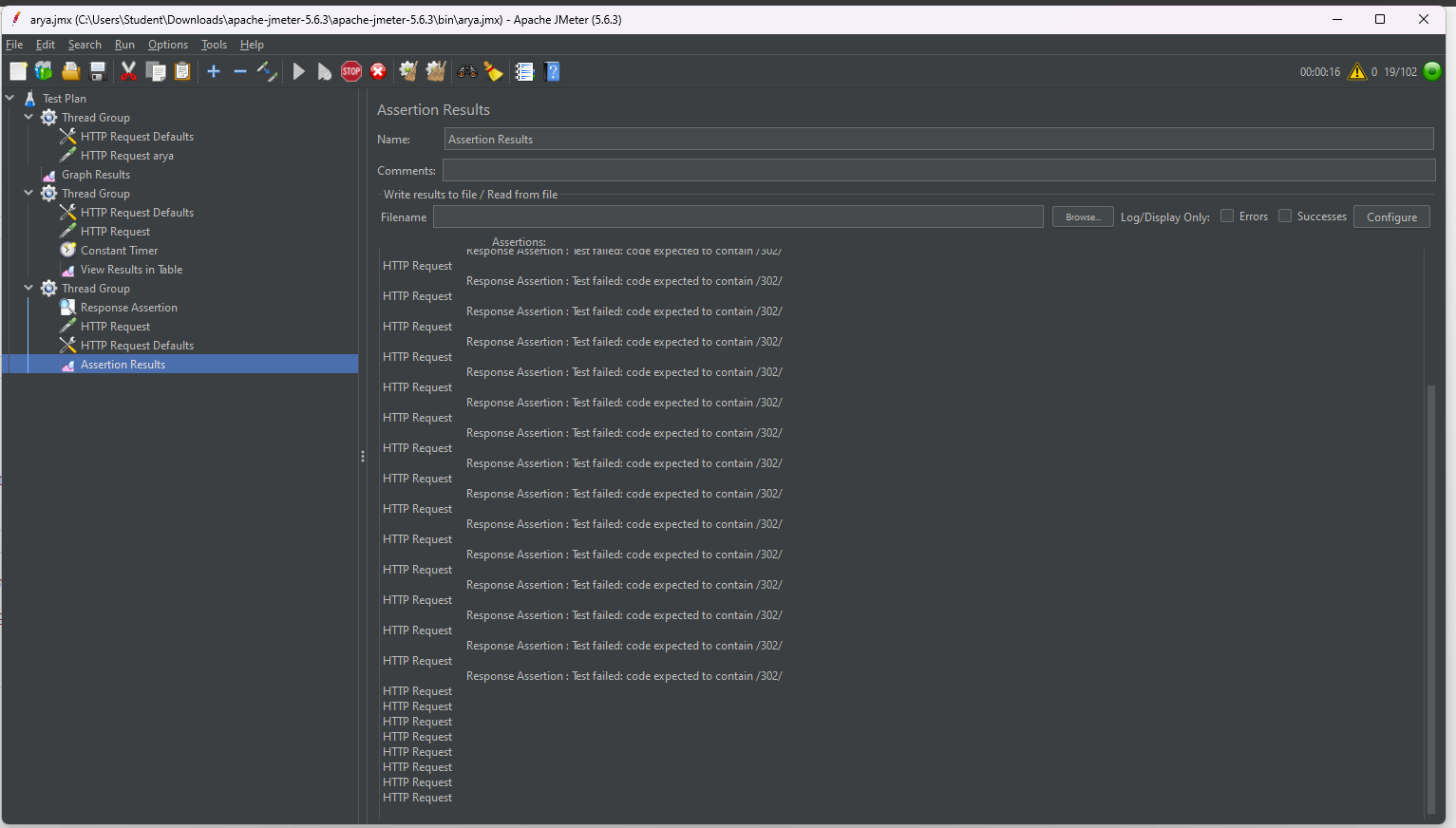


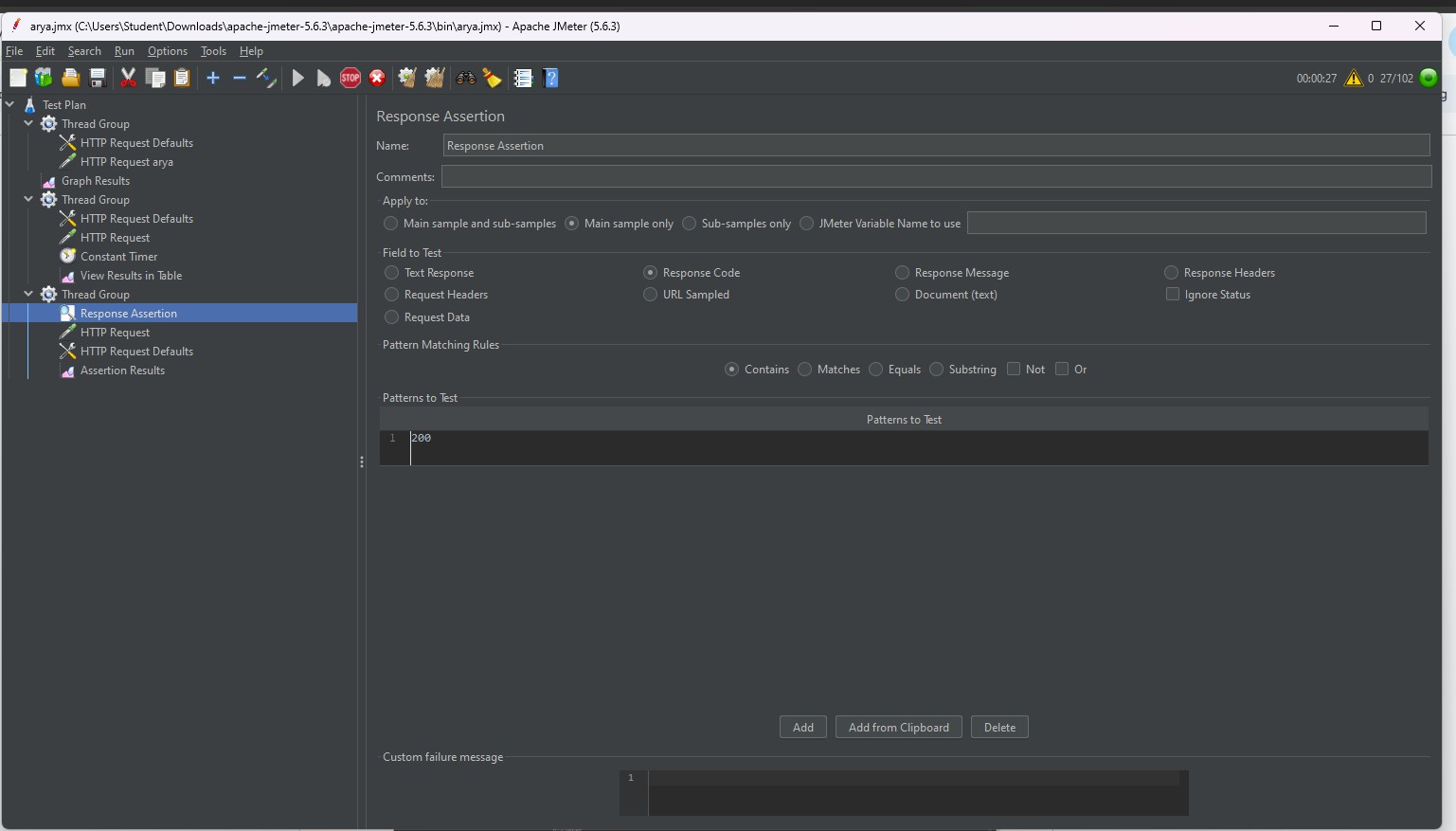


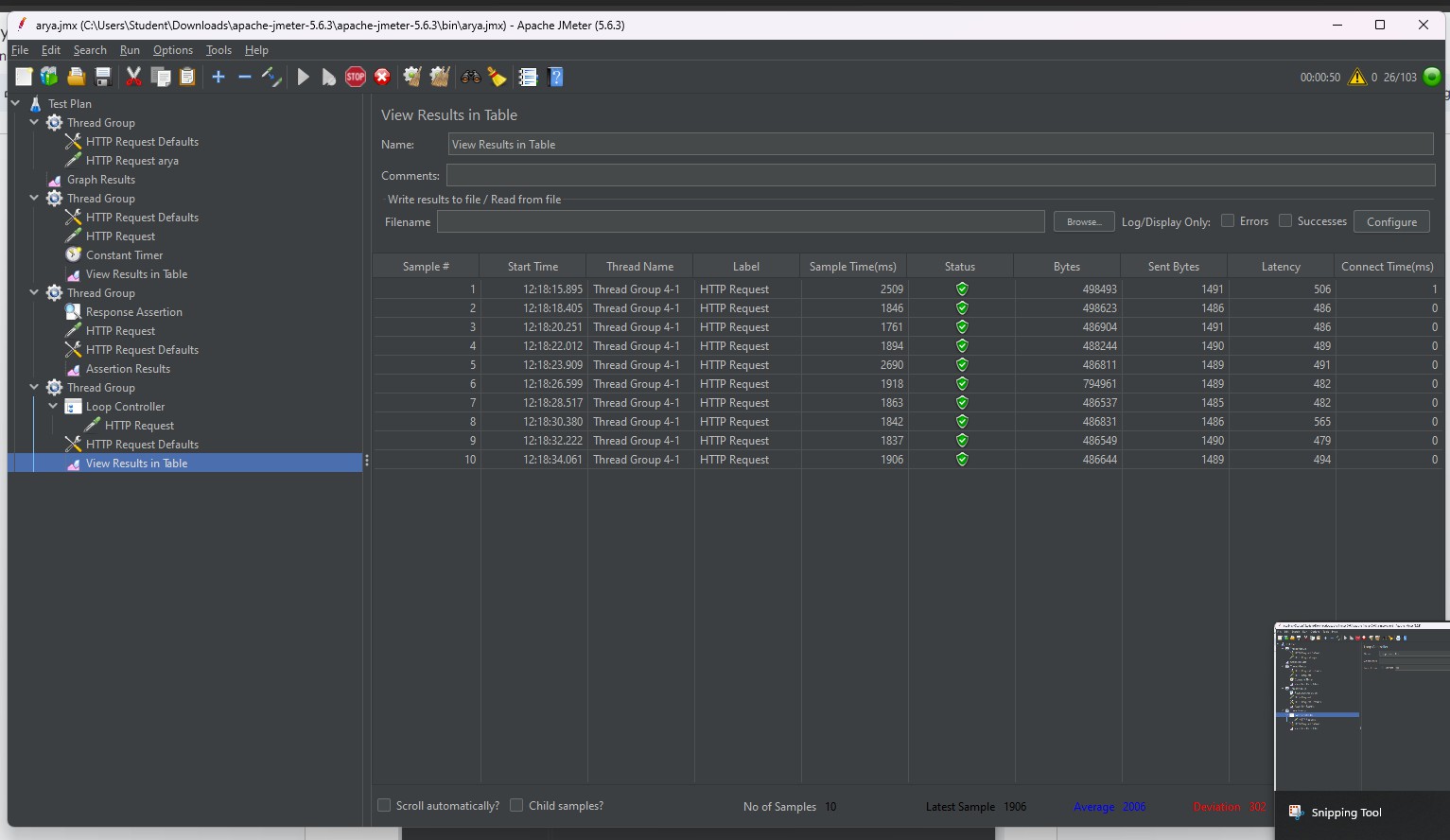
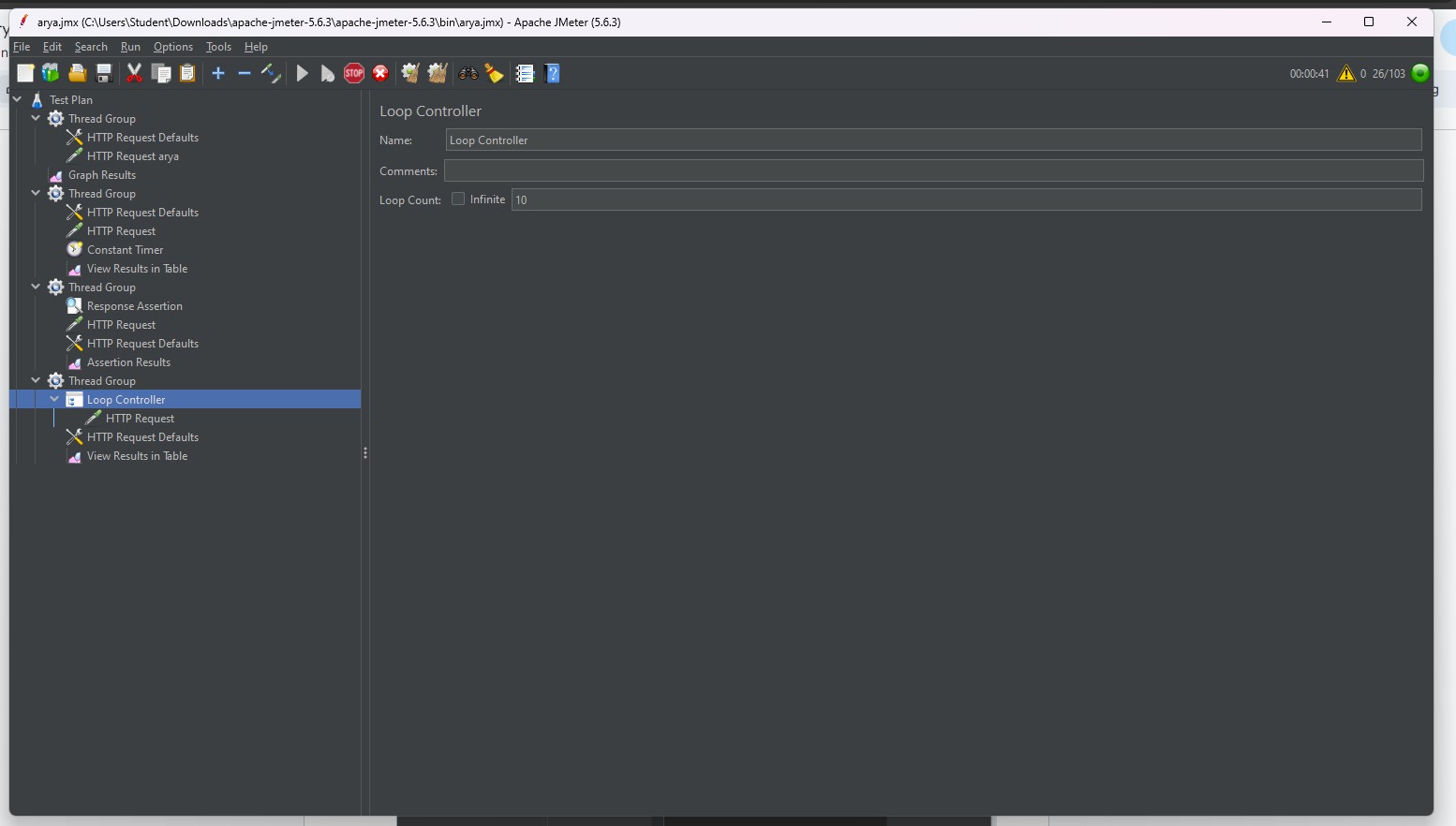


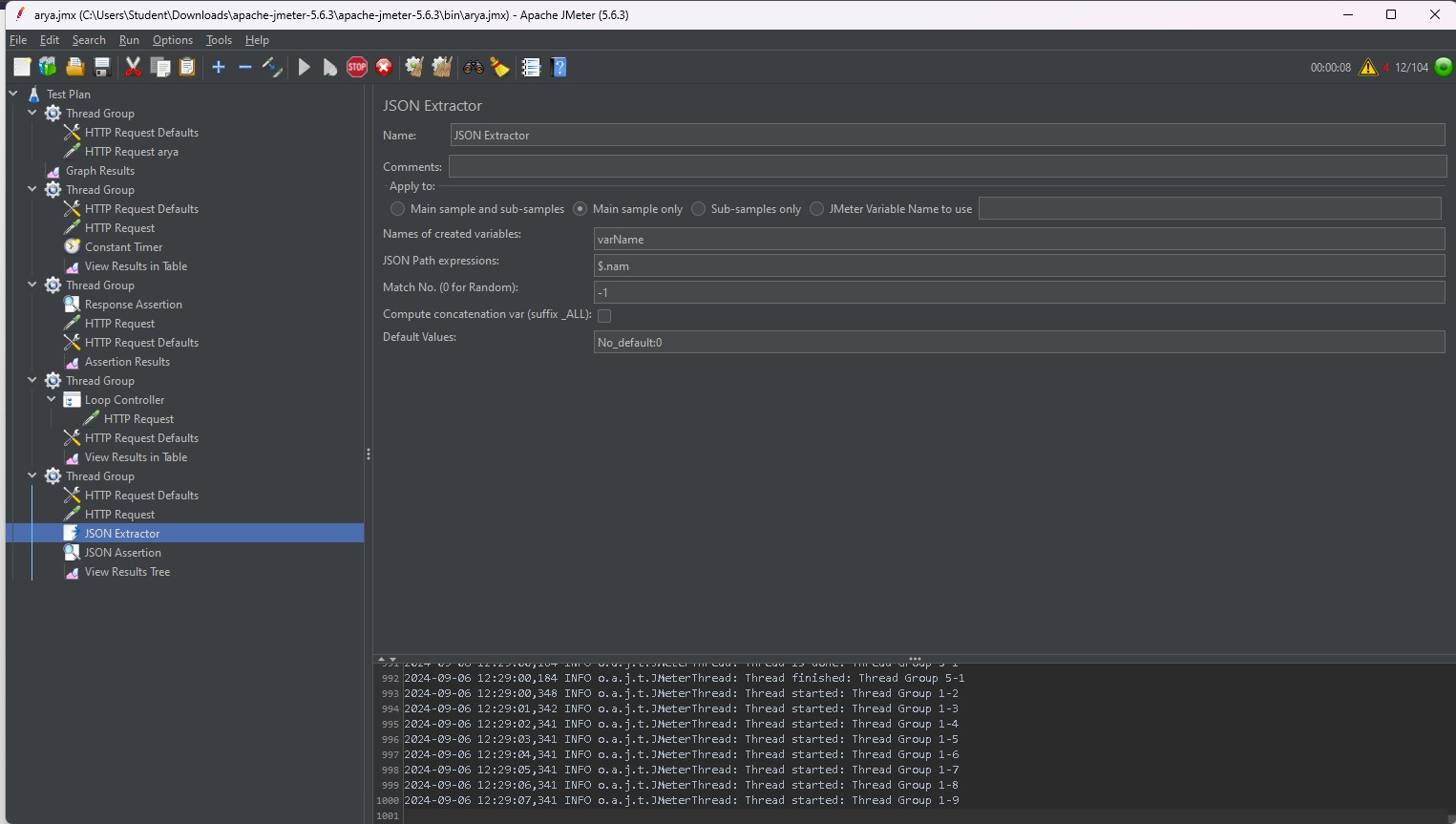














**Questions:**

1. Why performance testing is considered highly important for web applications? Outline the performance parameters against which testing can be performed.

Performance testing is crucial for web applications for several reasons:

1. User Experience: Slow or unresponsive web applications lead to a poor user experience, which can drive users away. Ensuring that an application performs well under load helps

retain users and improve satisfaction.

1. Scalability: Performance testing helps determine how well an application scales with increasing user load or data volume. This is important for planning capacity and ensuring

that the application can handle future growth.

1. Reliability: It helps identify potential issues before they impact end-users, ensuring that the application remains reliable and stable under various conditions.
2. Resource Optimization: It helps in identifying bottlenecks and optimizing resource usage, such as server CPU, memory, and network bandwidth.
3. Business Impact: Poor performance can lead to loss of revenue and damage to brand reputation. Performance testing helps mitigate these risks by ensuring that the application



performs well under expected conditions. Performance Parameters for Testing

Performance testing involves evaluating various parameters to ensure that a web application meets performance standards. Key performance parameters include:

1. Response Time: The time taken by the system to respond to a request. This includes:
   1. Page Load Time: Time to fully load a web page.
   2. Server Response Time: Time for the server to process a request and start sending a

response.

1. Throughput: The amount of data processed by the application in a given period. It is often measured in transactions per second (TPS) or requests per second (RPS).
2. Concurrency: The ability of the application to handle multiple simultaneous users or transactions without performance degradation.
3. Latency: The delay before a transfer of data begins following an instruction for its transfer.

This is critical for applications that rely on real-time data processing.

1. Error Rate: The frequency of errors occurring during tests, which can indicate issues with the application under load.
2. Scalability: The application’s ability to maintain performance as the number of users or transactions increases.
3. Resource Utilization: Measurement of how well the application uses system resources such as CPU, memory, disk I/O, and network bandwidth.
4. Load Handling: How well the application performs under a specific load, often measured using different load levels (e.g., light, moderate, heavy).
5. What are the major challenges in performance testing?

Major Challenges in Performance Testing

1. Accurate Simulation of Production Environment: Creating a test environment that accurately reflects the production environment in terms of hardware, network configuration,



and

KJSCE/IT/LYBTech/SEMVII/STQA/2024-25

data can be difficult.

1. Test Data Management: Managing and maintaining relevant and realistic test data is often challenging, especially for large-scale applications with complex data requirements.
2. Identifying Bottlenecks: Pinpointing the exact cause of performance issues can be complex due to the interplay of various system components, including databases, application servers,

and network infrastructure.

1. Scaling Issues: Testing how well the application scales and predicting how it will behave under different levels of load can be difficult, especially for applications with dynamic or

unpredictable usage patterns.

1. Complex Interactions: Modern web applications often integrate with various third-party services and APIs, making it challenging to account for the performance impact of these external dependencies.
2. Resource Constraints: Performance testing often requires significant computational resources and can be expensive in terms of time and infrastructure.
3. Tool Limitations: Performance testing tools might have limitations in terms of the scenarios they can simulate or the metrics they can capture, which may impact the

comprehensiveness of the testing.

1. Real-World Variability: The performance experienced by end-users can be influenced by many factors that are hard to replicate in a test environment, such as varying network

conditions and user behaviors.

1. What is the significance of load testing?

Load testing is crucial for evaluating how a system performs under various conditions of demand. Its significance lies in several key areas:

Performance Evaluation: Load testing helps determine how a system behaves under normal and peak load conditions. It assesses factors like response time, throughput, and resource utilization to ensure the system can handle the expected user load.



KJSCE/IT/LYBTech/SEMVII/STQA/2024-25

Identifying Bottlenecks: By simulating different levels of load, load testing helps identify

performance bottlenecks, such as limitations in the hardware, software, or network infrastructure. This allows developers to address these issues before they affect end users.

Scalability Testing: It verifies whether the system can scale up or down as required. This is particularly important for systems expected to handle varying loads, such as e-commerce sites during sales events or applications with growing user bases.

Reliability and Stability: Load testing assesses the system’s reliability and stability under stress, ensuring that it can handle unexpected spikes in demand without crashing or degrading in performance.

User Experience: By identifying potential performance issues before they reach production, load testing helps ensure a smooth user experience. This can be crucial for user satisfaction and retention, particularly for high-traffic applications.

Compliance and SLAs: For systems with specific performance requirements or Service Level Agreements (SLAs), load testing ensures that these requirements are met and helps demonstrate compliance to stakeholders or clients.

# Outcomes: CO3 Apply recent automation tools for testing software.

**Conclusion: (Conclusion to be based on outcomes)**

use jmeter to perform testing on http requests

# Grade: AA / AB / BB / BC / CC / CD /DD Signature of faculty in-charge with date

**Reference Websites:**

1. [https://www.guru99.com/jmeter-performance-testing.html](http://www.guru99.com/jmeter-performance-testing.html)
2. <https://dzone.com/articles/jmeter-performance-and-load-testing>
3. <https://jmeter.apache.org/>