Non-Functional Testing – Project

Web application Load and Capacity Test

LOAD AND CAPACITY TEST PLAN AND REPORT

TABLE OF CONTENTS

1 INTRODUCTION

1.1 PURPOSE OF LOAD TEST OF THE WEB APPLICATION

Load testing is a type of performance testing in which the system is tested for real-time load that the system will encounter after going into production. In other words, it is to see how the system will behave when multiple users use it at same time. It will be helpful since the development team will know in advance if the application will be able to work normally under normal and peak load. There are multiple tools available to perform load testing. In this project, we will be using **JMeter** to do load testing by simulating behavior of different number of users trying to use the address-book application at same time and see how changing the number of users impact the resources.

1.2 PURPOSE OF CAPACITY OR VOLUME TESTING

Capacity testing/volume testing is a type of performance testing which determine how many users the application can handle without degrading the performance of the application. This is also called flood testing, since application is subjected to huge volume of data. In other words, the performance of the application will be examined after increasing the volume of data in the database. In our project we will be performing various CRUD commands on the database. CRUD means create, read, update, and delete commands will be used to add any record, read record from database, update any existing record, or delete the existing record. So, objective is to examine the performance of the application when the volume of data is increased in the database.

1.3 TEST ENVIRONMENT

Test environment gives feedback about the quality and behavior of SUT by allowing us to run the test cases. In general, a test environment includes desktop, server, network, OS, hardware, software, browser. For performing the load and volume testing of the application, we are using personal laptop, personal internet, windows 10 OS, Opera web browser, JMeter, Apache tomcat web server, MYSQL community server, My sql connectors, Command prompt to execute SQL queries.

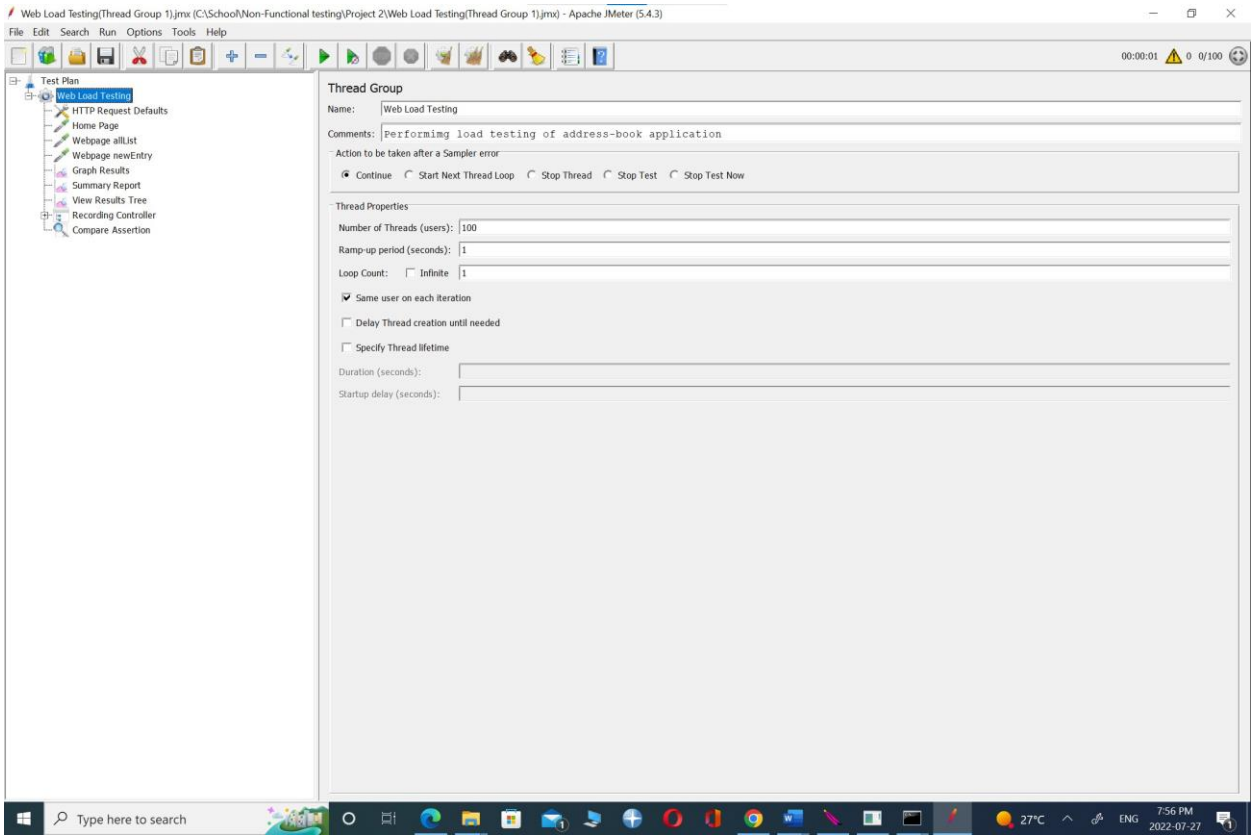
JMeter is a testing tool that will help to make http calls to the application by changing the number of users making calls at a time. Also, it will enable to create JDBC connection to the database and execute various CRUD SQL commands. It allows to view the result in various forms like summary report, Graph results, view result tree and many more.

1.4 TEST DESIGN

To perform the load and volume testing, we used JMeter to create a test plan with two thread group:

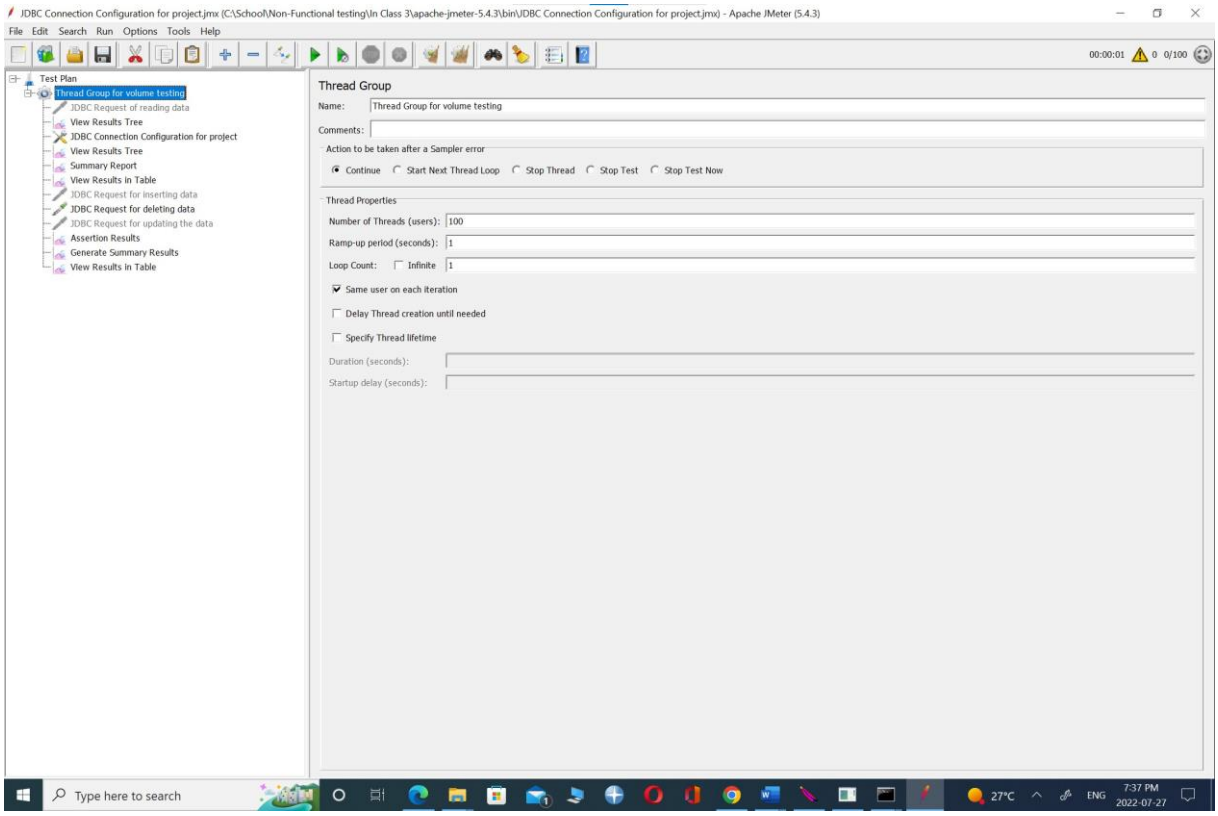
1. **Thread group for web load testing**
2. **Thread group for Volume testing**

1. Thread group for web load testing:



To the Thread group we added, http request, recording controller and assertions. Http request is used to send http requests to webpages of the application: *Home page*, *Webpage allList*, *Webpage newEntry*. We changed the number of users from the Thread properties. Assertions are used to validate the response of the request sent to the server. Added listeners to view the result of the test plan. Recording controller act as a storage for recording steps.

2. Thread group for Volume testing:

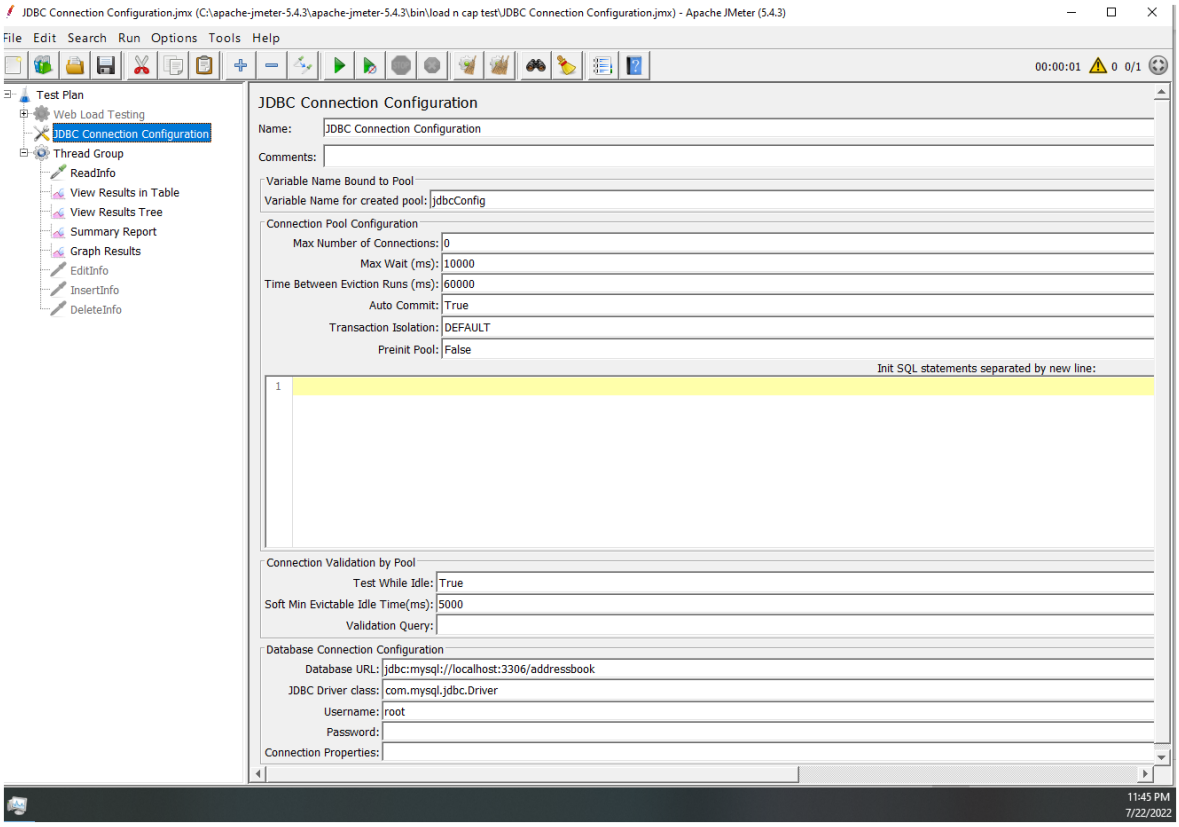


For volume testing, we created JDBC connection to the database using JMeter. The screenshot is provided on the next page.

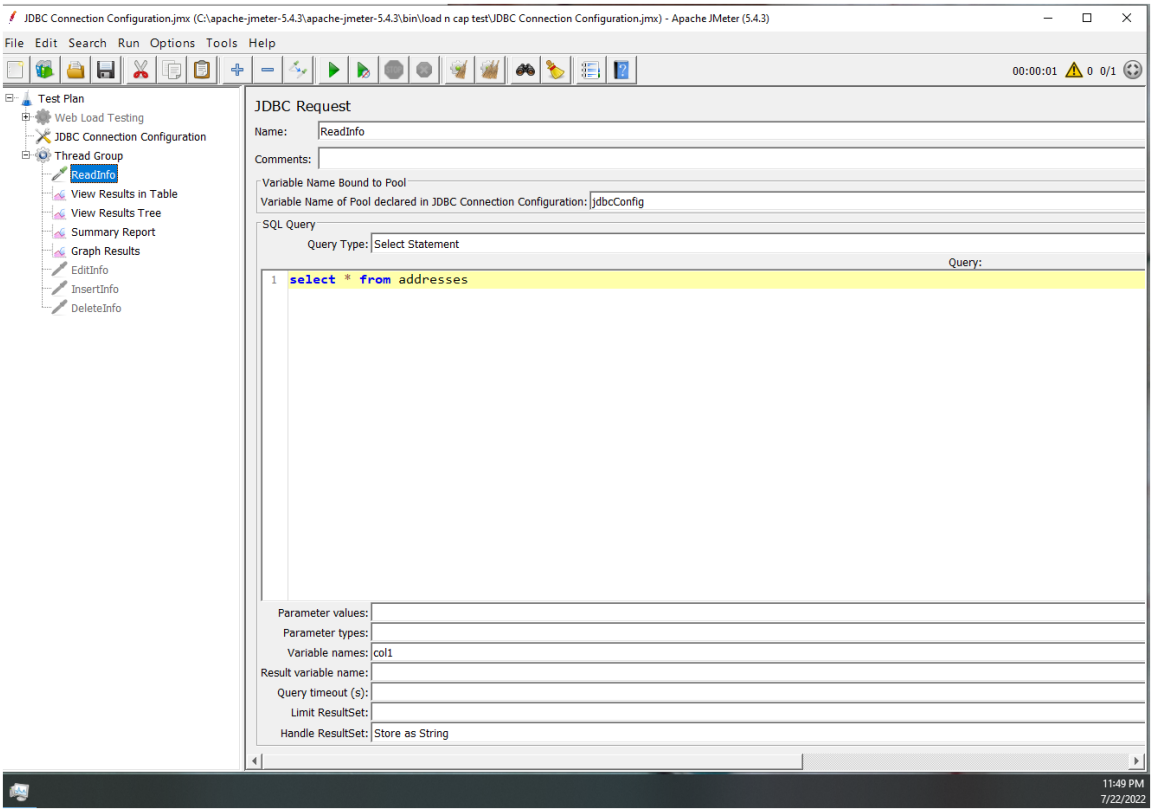
After configuring the JDBC connection, we created JDBC request for different CRUD operations. So, four JDBC requests were created:

- JDBC request for Read.
- JDBC request for Insert.
- JDBC request for Delete.
- JDBC request for Update.

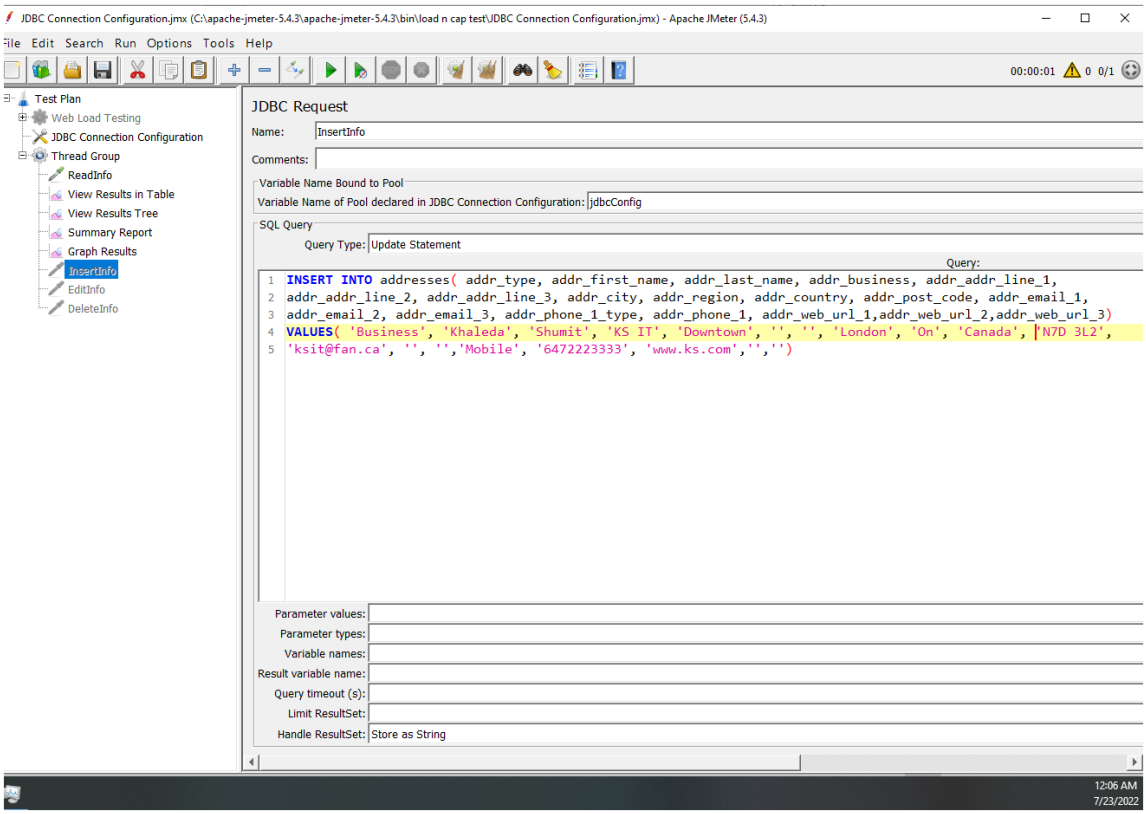
The screenshots of the four JDBC requests mentioned above are provided below:



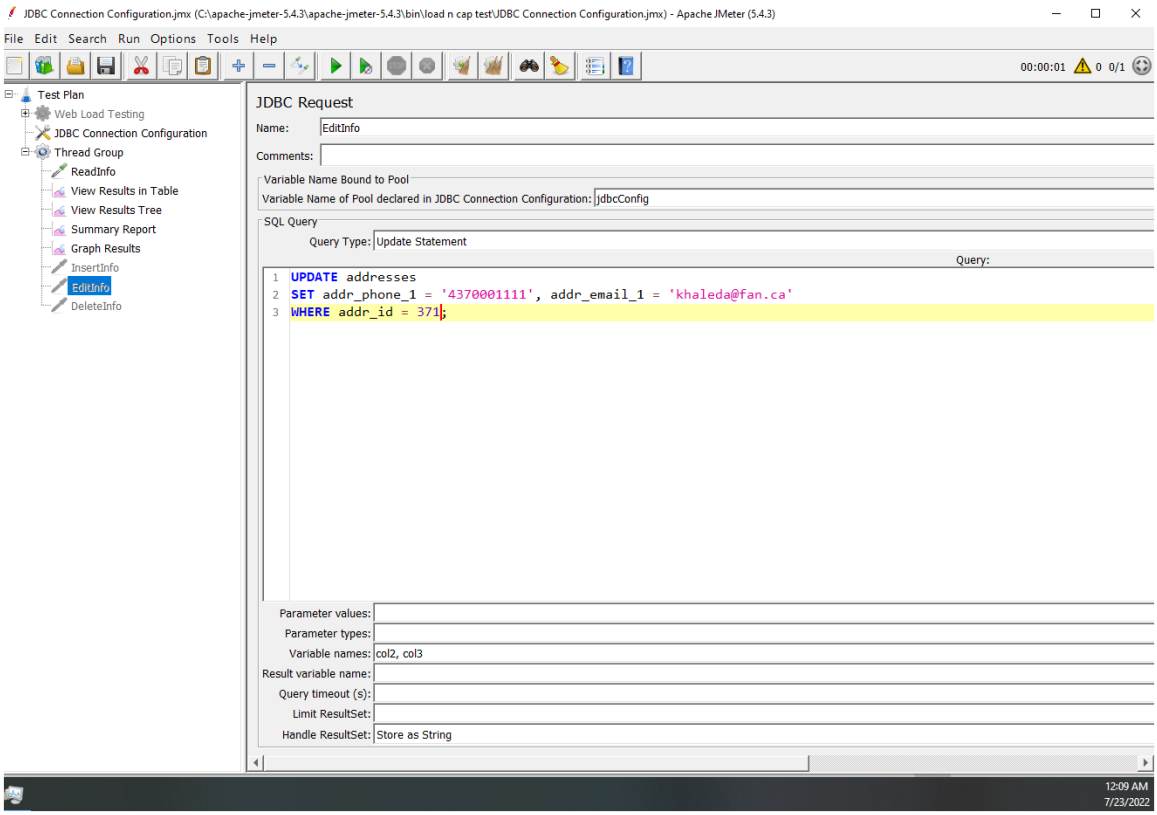
JDBC connection Configuration



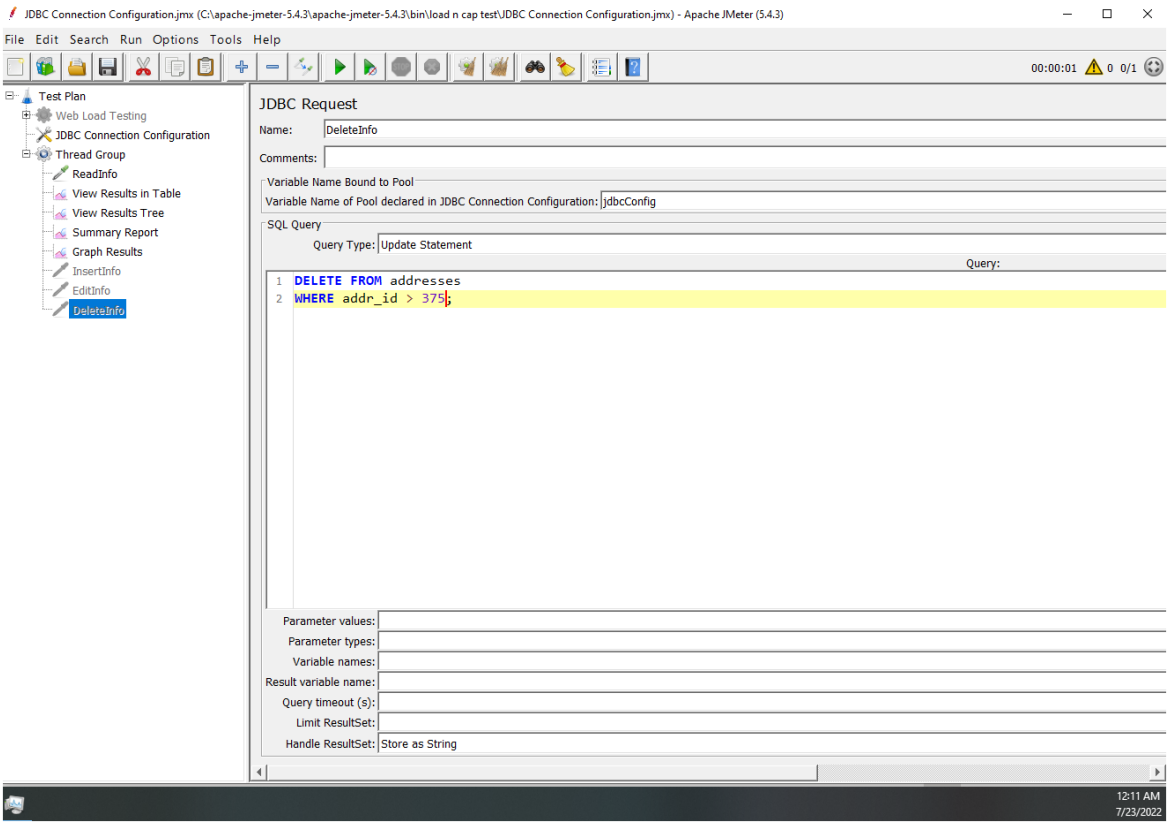
JDBC request for Read



JDBC request for Insert`



JDBC request for Update



JDBC request for Delete

After this, we ran the test plan by changing the number of users and enabling one JDBC request at a time. Also, we captured the results adding various listeners like Summary report, Result view table, Result view tree etc. Results are provided in the test report.

1.5 TEST DATA

Data can be generated in two ways:

1. By going to web application home page and clicking on the Add new Entry.
2. By using Insert command in the JMeter' JDBC request. Also, other commands like read, update, or delete can be used to retrieve or alter any data.

Special consideration should be made about the data type of the data being entered into the database. Data type of the data to be entered in the database is given below:

Input Fields	Type
addr_id	INT
addr_type	VARCHAR
addr_first_name	VARCHAR
addr_last_name	VARCHAR
addr_business	VARCHAR
addr_addr_line_1	VARCHAR
addr_addr_line_2	VARCHAR
addr_addr_line_3	VARCHAR
addr_city	VARCHAR
addr_region	VARCHAR
addr_country	VARCHAR
addr_post_code	VARCHAR
addr_email_1	VARCHAR
addr_email_2	VARCHAR
addr_email_3	VARCHAR
addr_phone_1_type	VARCHAR
addr_phone_1	VARCHAR
addr_phone_2_type	VARCHAR
addr_phone_2	VARCHAR
addr_phone_3_type	VARCHAR
addr_phone_3	VARCHAR
addr_web_url_1	VARCHAR
addr_web_url_2	VARCHAR
addr_web_url_3	VARCHAR

1.6 METRICS

Following response and volume metrics are used to determine the performance of address-book application:

- Average response time: It is the average of the time taken to respond to requests. Time is measured from the moment request is made (request cycle) till the response of the request is sent back.
- Error rate: It is calculated as percentage of number of requests that encountered error to the total number of requests made.
- Concurrent users/Sample: It is the measure of the number of virtual users active at any point in time.
- Requests per Second: It is the count of the request that are sent to the server in a second.
- Throughput: It tells how many requests an application can handle in a minute or an hour.

1.7 TEST CASE(S)

Test Case ID	Description	Inputs	Actual Result	Avg. Response time(MS)	Pass/Fail
TC_101	Read data from database, when users = 1	Select * from addresses	Display all records of the table	340	Pass
TC_102	Read data from database, when users = 10	Select * from addresses	Display all records of the table	23	Pass
TC_103	Read data from database, when users = 100	Select * from addresses	Display all records of the table	111	Pass
TC_104	Read data from database, when users = 250	Select * from addresses	Display all records of the table	64	151 Sample Pass 99 sample failed
TC_105	Read data from database, when users = 500	Select * from addresses	Display all records of the table	124	151 Sample Pass 349 Sample failed
TC_106	Insert data into the database, Where users = 1	Execute insert command	Record gets updated in the database.	13	Pass

TC_107	Insert data into the database, Where users = 100	Execute insert command	Record gets updated in the database	13	Pass
TC_108	Insert data into the database, Where users = 500	Execute insert command	Record gets updated in the database	16	151 sample Pass 349 sample failed
TC_109	Update data from the database, Where users = 1	Execute Update command	Record gets updated in the database	15	Pass
TC_110	Update data from the database, Where users = 100	Execute Update command	Record gets updated in the database	12	Pass
TC_111	Update data from the database, Where users = 500	Execute Update command	Record gets updated in the database	97	151 sample Pass 349 sample failed
TC_112	Delete data from the database, where users = 1	Execute Delete command	Record gets deleted from the database.	22	Pass
TC_113	Delete data from the database, where users = 100	Execute Delete command	Record gets deleted from the database.	13	Pass
TC_114	Delete data from the database, where users = 250	Execute Delete command	Record gets deleted from the database.	16	Pass
TC_115	Delete data from the database, where users = 500	Execute Delete command	Record gets deleted from the database.	99	151 sample Pass 349 sample failed s

TC_116	Make http request to 3 pages of the address book app, user = 1	NA	Request to all 3 pages is sent	24	Pass
TC_117	Make http request to 3 pages of the address book app, user = 100	NA	Request to all 3 pages is sent	300	Pass
TC_118	Make http request to 3 pages of the address book app, user = 500	NA	Request to all 3 pages is sent	423	Pass

2 TEST RESULTS - REPORT

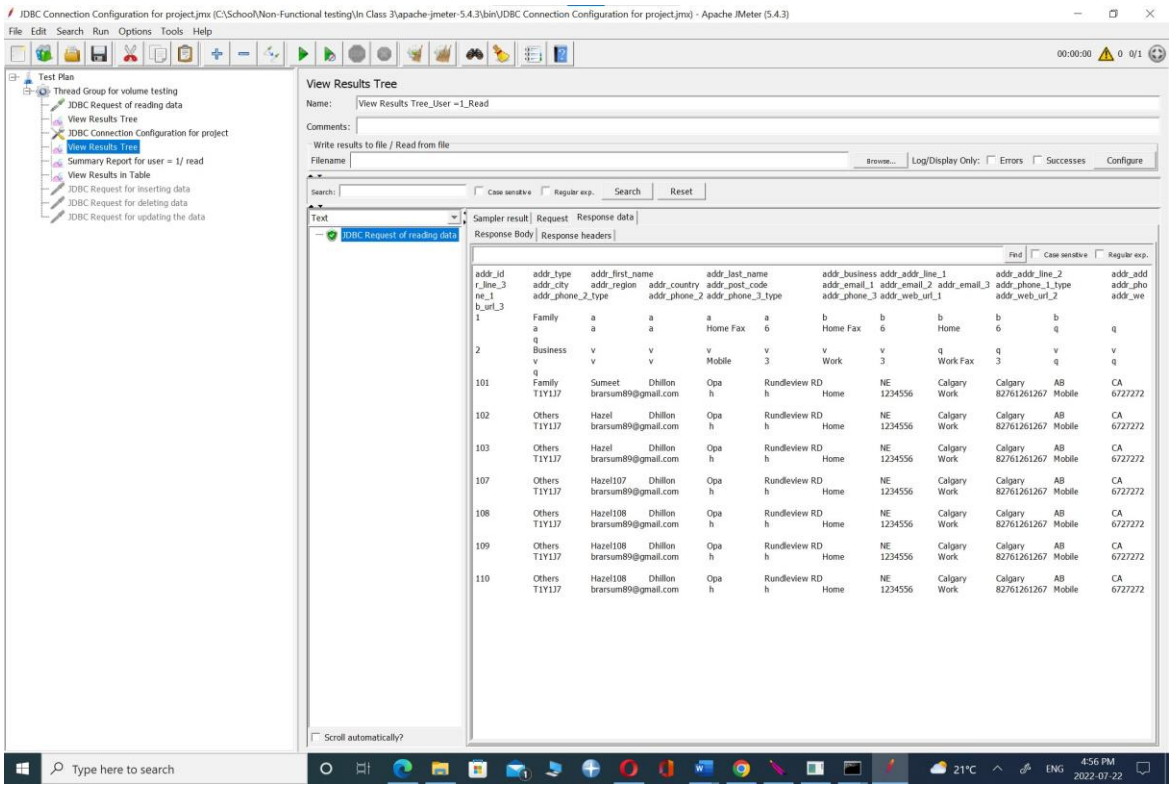
We executed insert, read, delete, and update commands by changing the number of users and received following values for deviation, error rate and throughput:

Command Type	Number of users	Error rate (%)	Deviation	Throughput/sec
Read	1	0	0	2.9
	10	0	5.13	10.7
	100	0	85.73	96.7
	250	39.60	77.54	242.7
	500	69.80	184.83	254.8
	1500	90.00	545.17	460.8
	3000	95.03	947	611
Insert	1	0	0	76.9
	10	0	4.37	81.6
	100	0	4.64	97.9
	250	39.60	51.03	213.5
	500	69.80	21.83	349.2
	1500	90	1049	337

	3000	95	825	585
Delete	1	0	5	183
	10	0	3.83	10.8
	100	0	3.17	97.1
	250	39.60	26.16	246.3
	500	70	210.55	399
	1500	89.9	34.98	775
	3000	94.97	361	1207
Update	1	0	0	66.7
	10	0	3.83	10.8
	100	0	3.17	97.1
	250	39.60	26.16	246.3
	500	70	210	399.4
	1500	89.93	44.89	710.9
	3000	95.7	117	1041

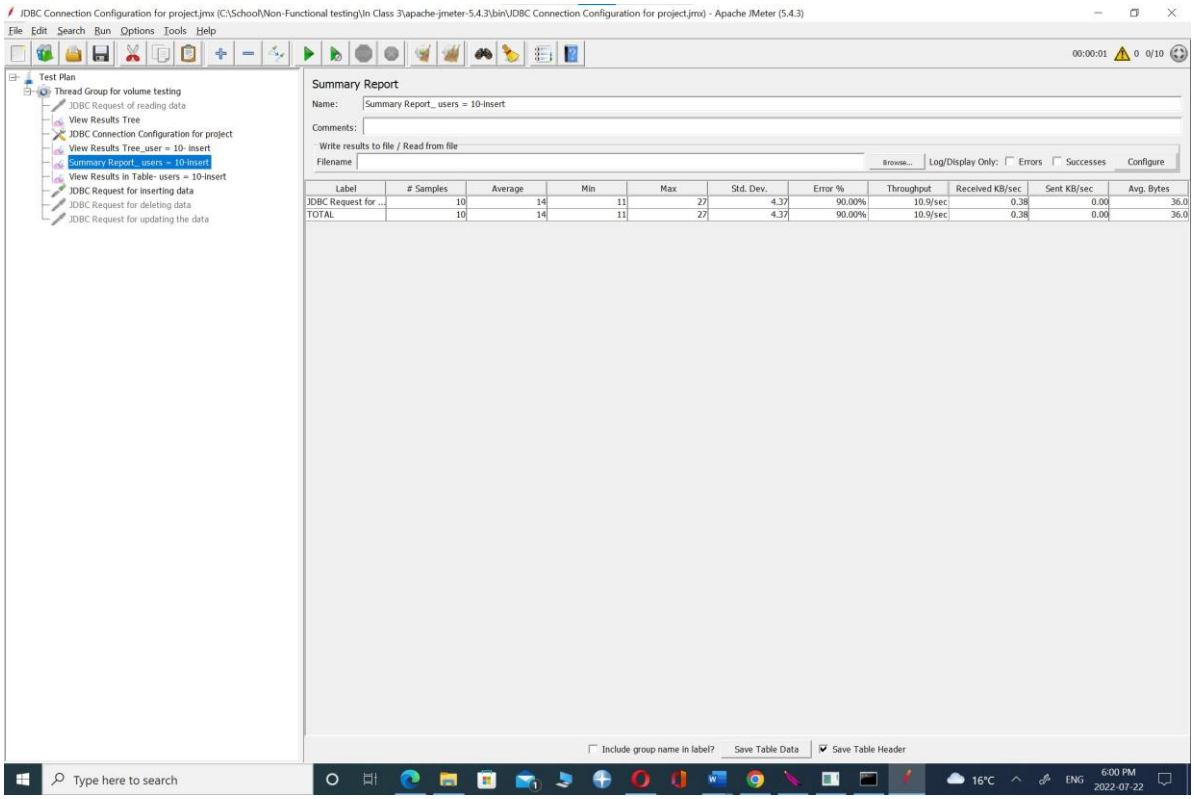
Following are provided the random screenshots from JMeter showing the results captured after running the test cases:

- **Result of Read commands when user = 1**



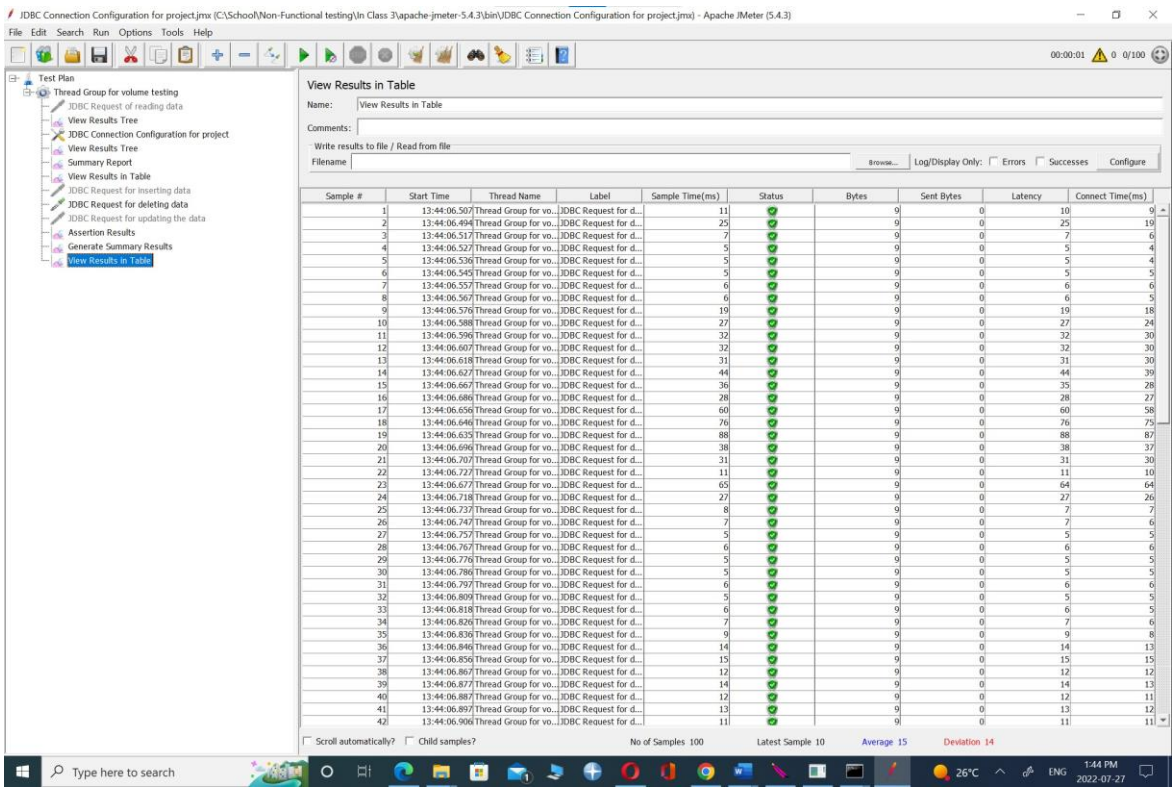
View Result Tree listener

- **Result of Insert command when user = 10**



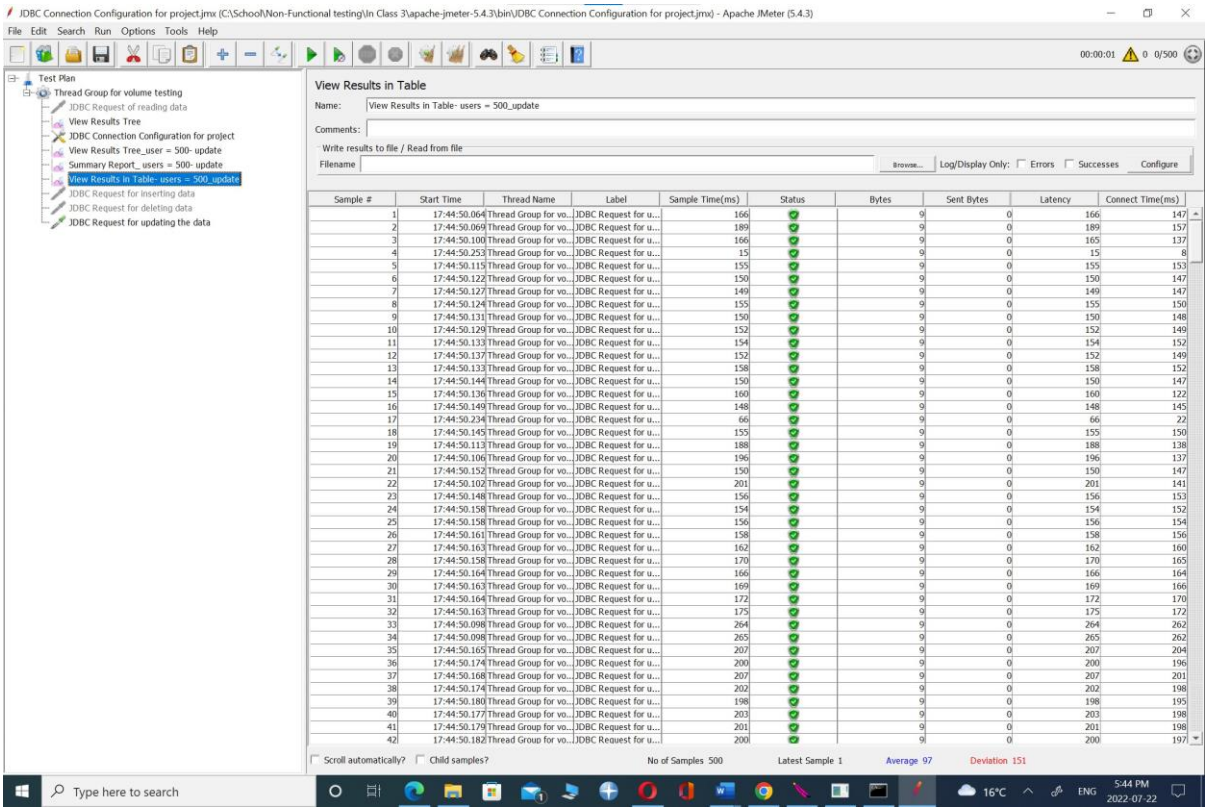
Summary report listener

- **Result of Delete command when users = 100**

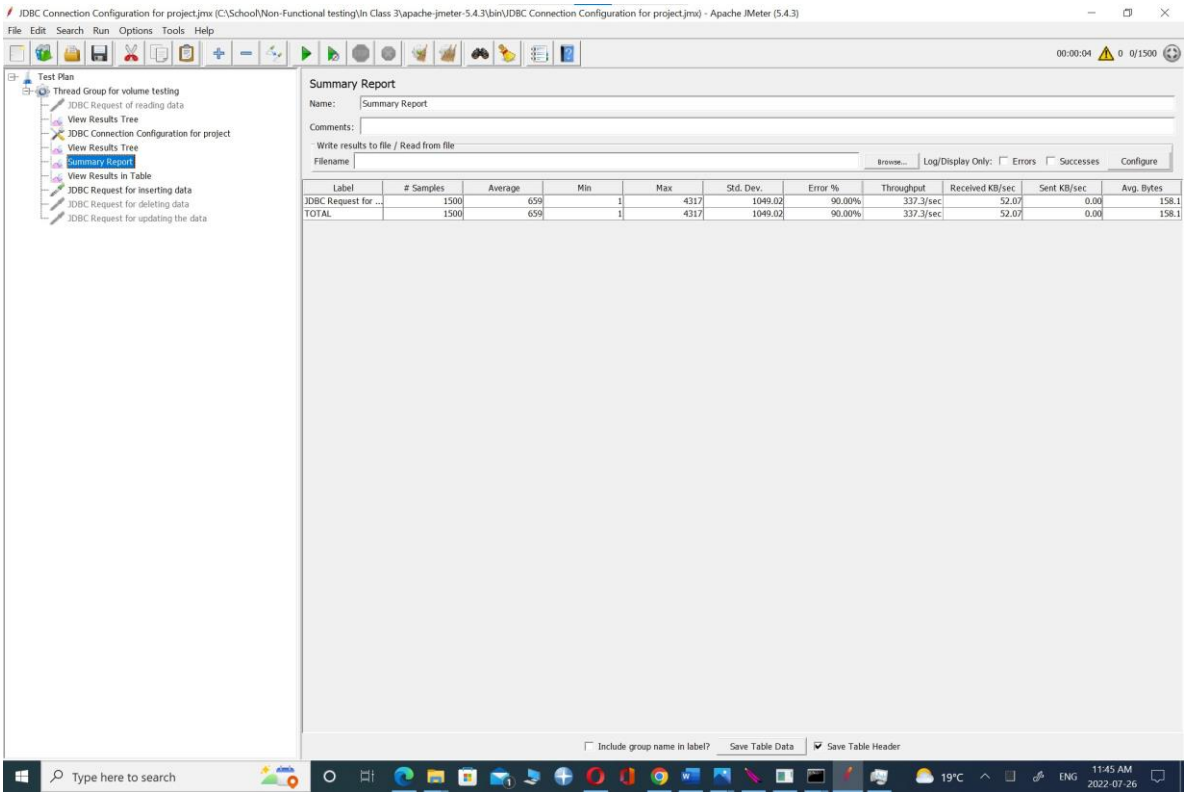


View Result in Table listener

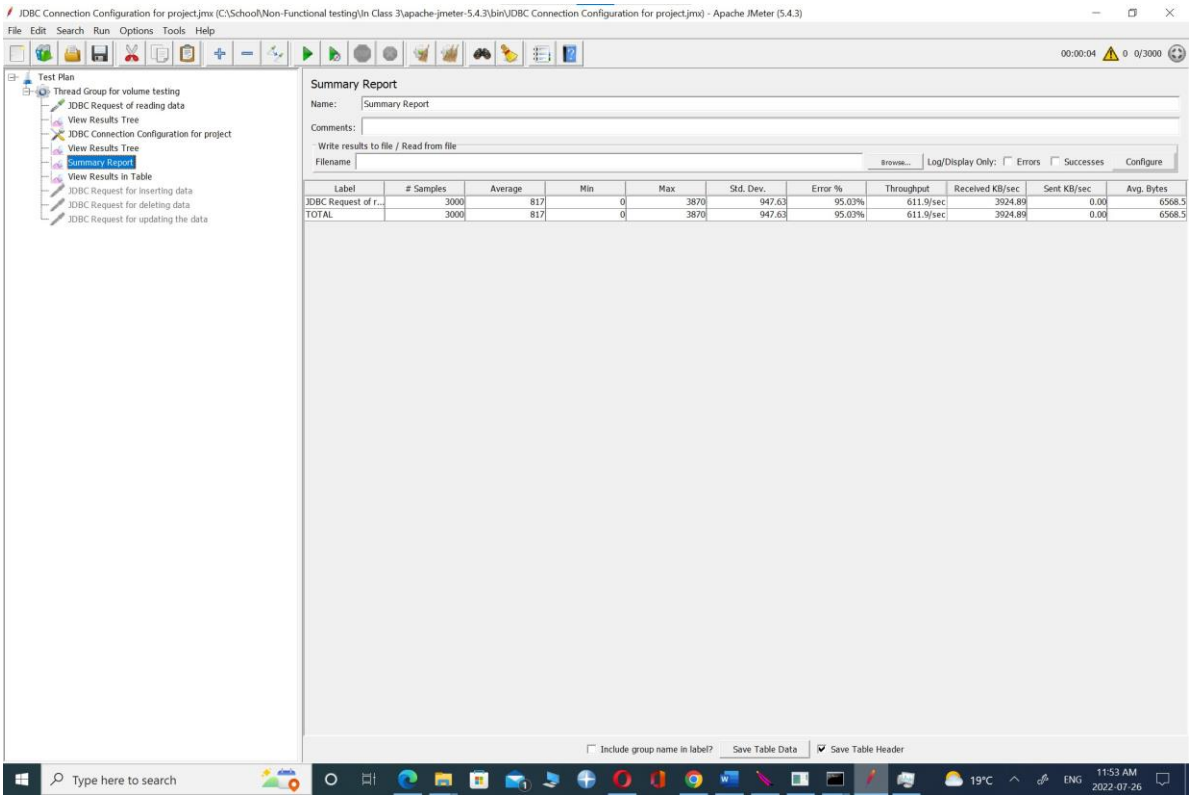
- **Result of Update command when users = 500**



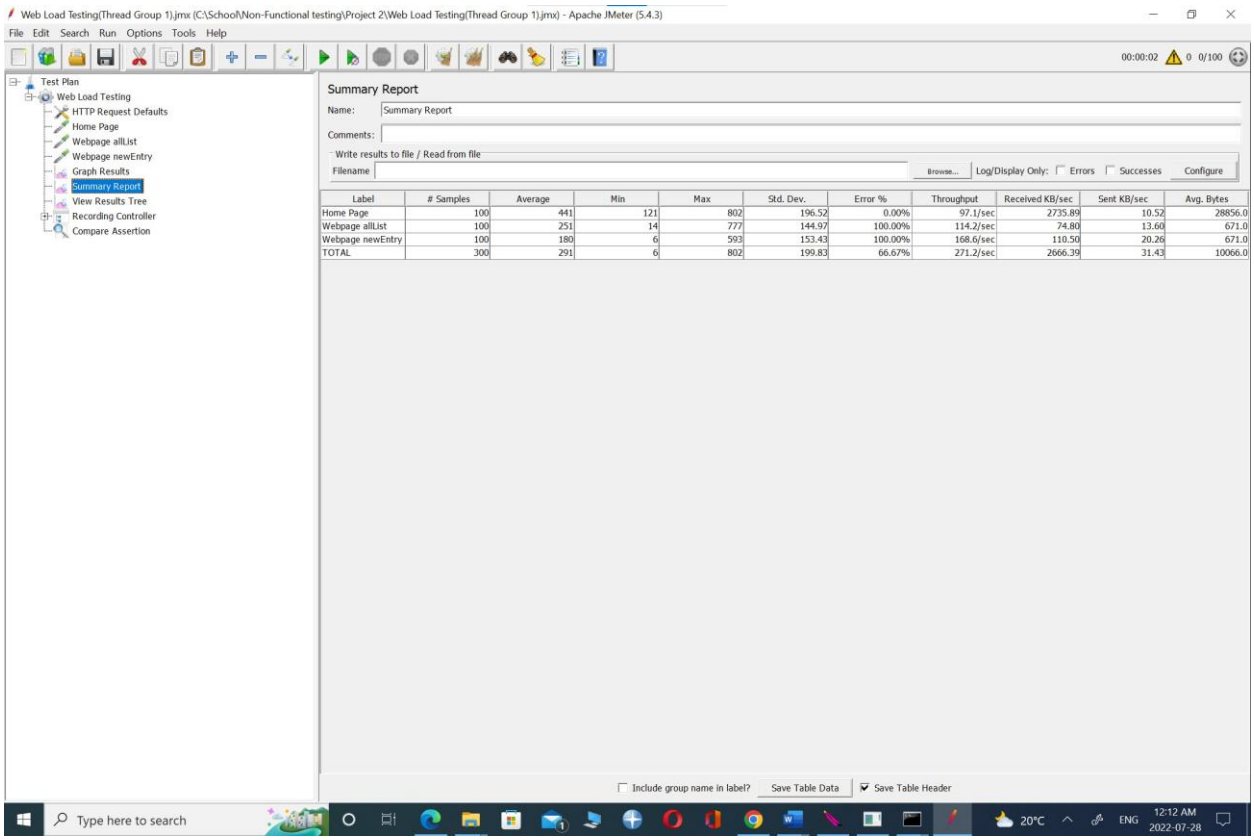
- **Result of Insert command when users = 1500**



- **Result of Read command when users = 3000**

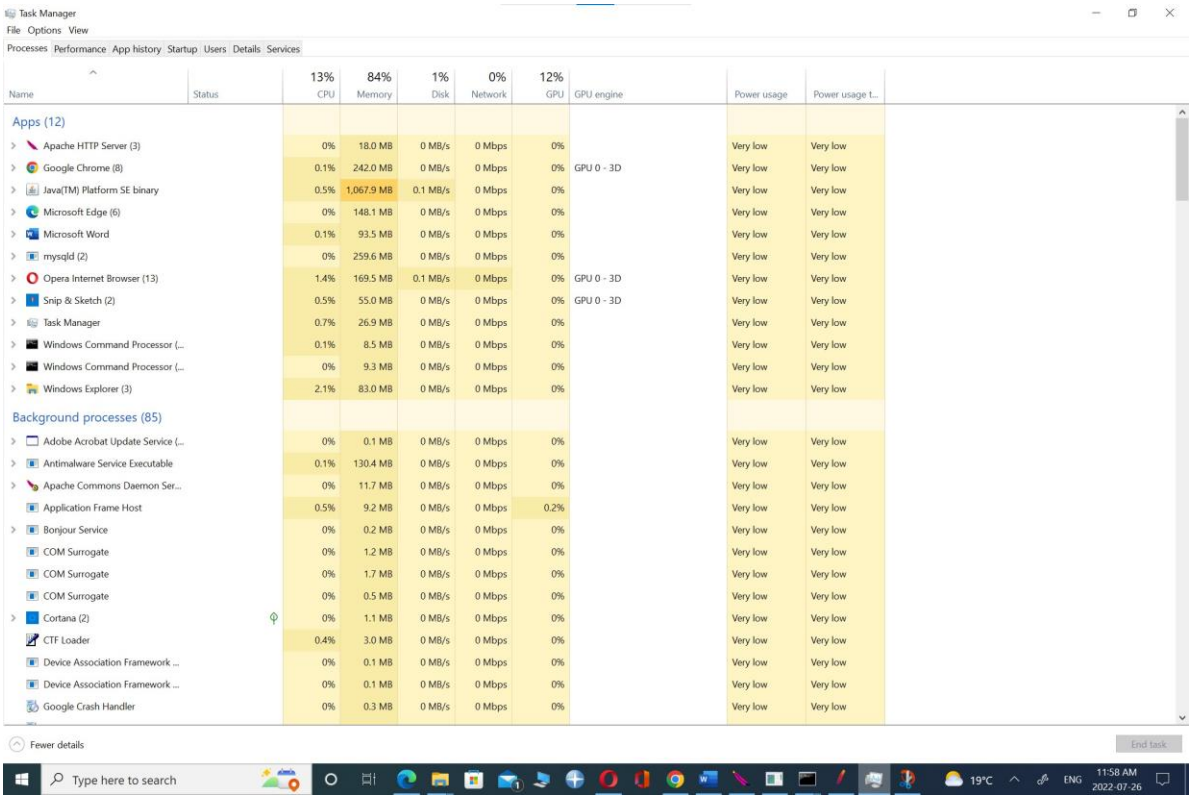


- **http request sent to web pages when users = 100**

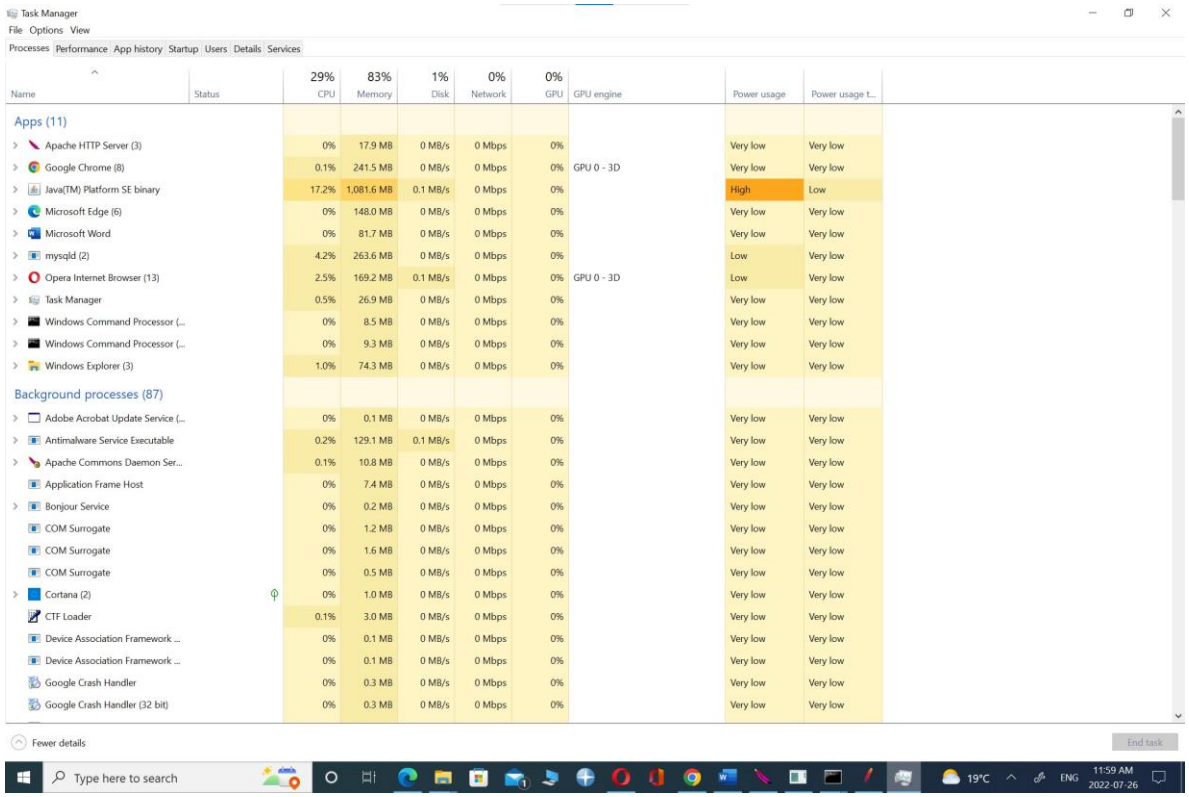


3 DEMONASTRATION OF RESOURCE UTILIZATION

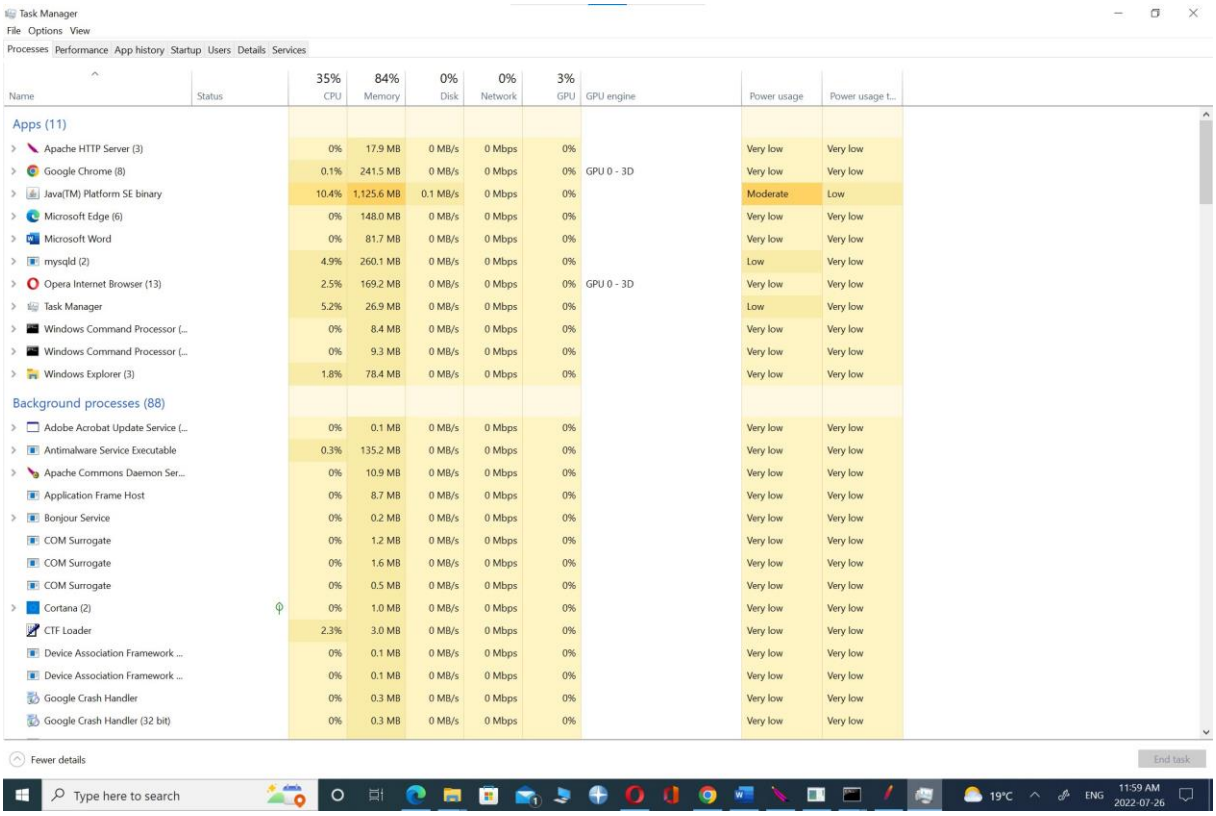
Below are the screenshots of task manager collected while running the test cases on JMeter:



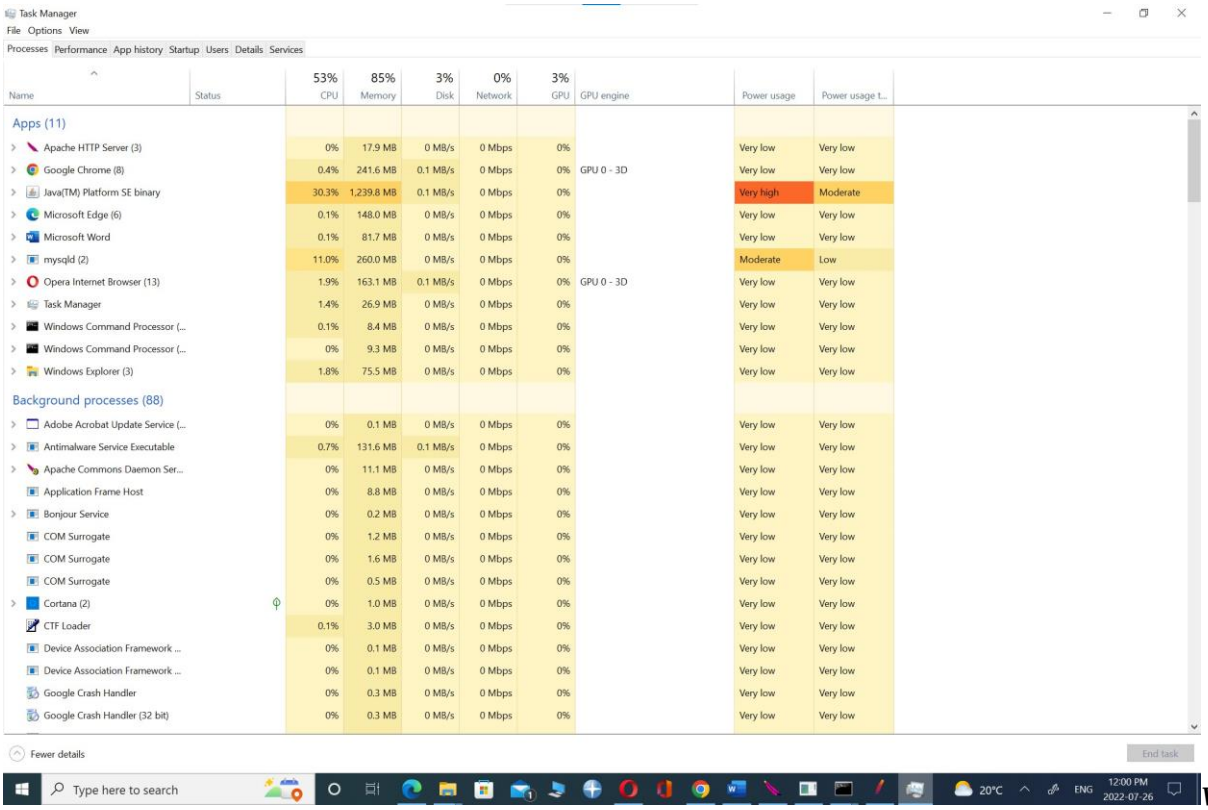
When No. of users = 1



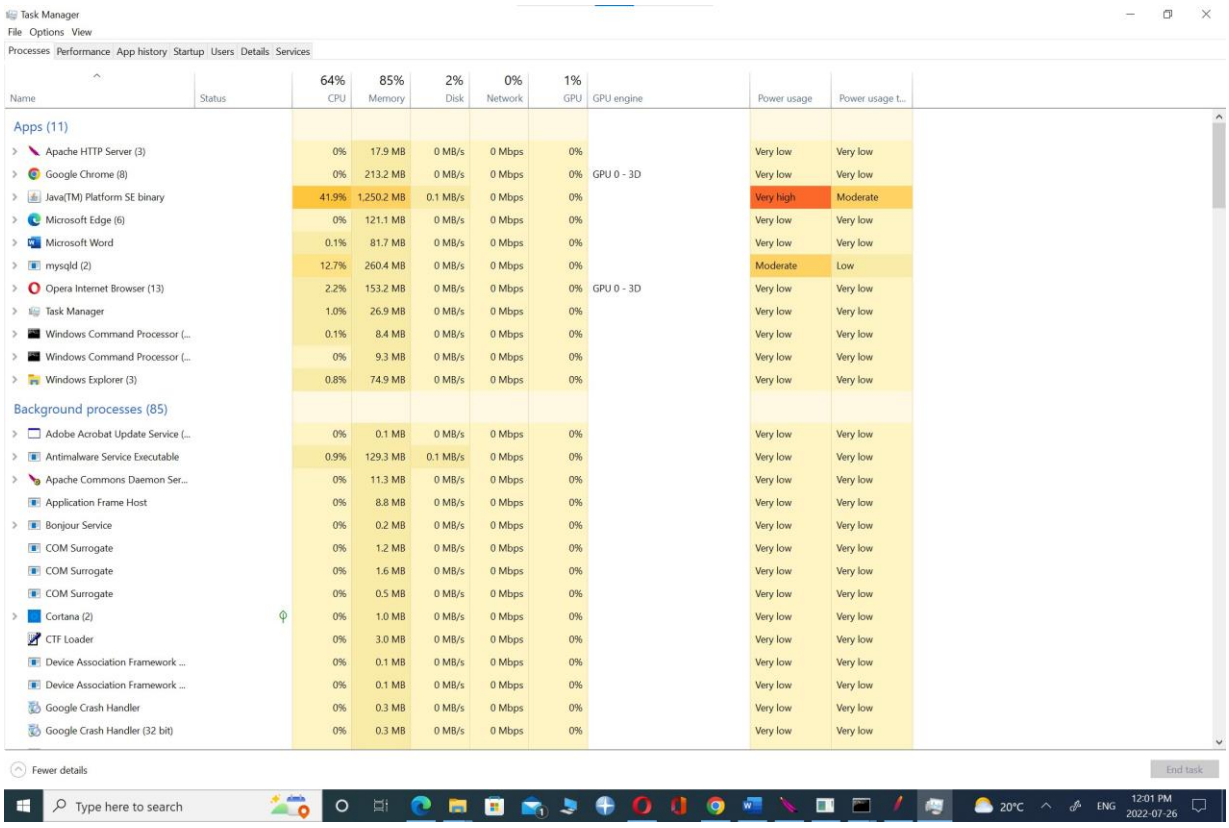
When No. of users = 100



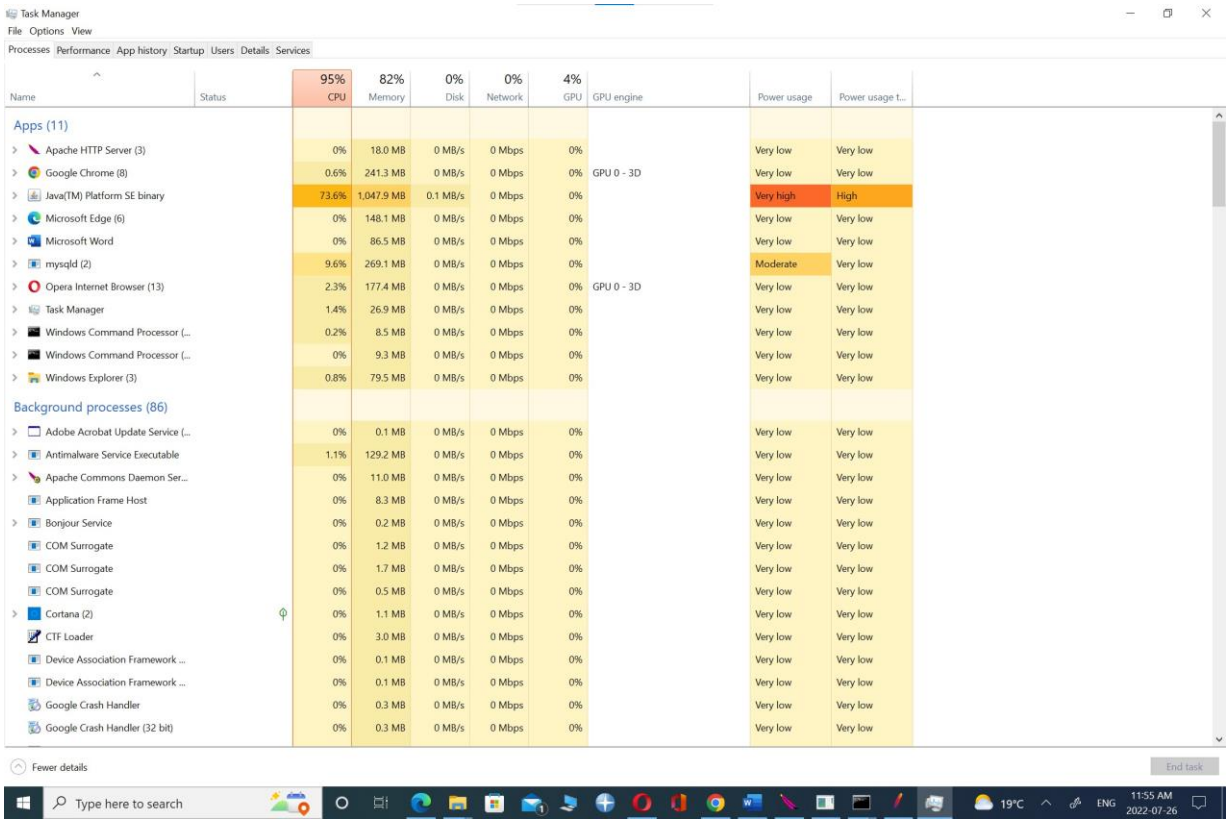
When No. of users = 250



When No. of users = 500



When No. of users = 1000



When No. of users = 1500

From the screenshots above, its can be said that the resources (CPU, memory, power) usage went higher with increase in the number of the users. 95% of CPU was used when users = 1500. Power usage went from low to moderate to very high with increasing load.

4 ANALYSIS AND FINIDINGS

1. **In terms of the application performance, how did the address-book application perform?**
   When considering the load testing, we ran the test cases of http requests made to web pages of the application by varying the number of the users, the avg. response time was good. For the volume testing as well the response time for performing various CRUD commands was good. However, it kept changing every other time the test case was run. Also, the response time kept increasing with the increase in number of users. Another interesting thing that we noticed was read operation was successful for all users when test case run for 250 users, but many transactions failed when same was run after some. So, the error rate varied for same test case when run at different times.

2. **Which metrics did you collect for verifying the performance of the application?**
   We collected average response time for each test case. Also, error rate and throughput.
3. **In terms of the capacity of the database what was your observation?**
   Capacity wise the database was good. We ran 1000s of insert commands and read commands. The throughput was satisfactory.

4. **Which metrics did you collect for analyzing your result?**
   We used response time and error rate as the major metrics to analyze our reports and reach to a conclusion.
5. **What would be your overall recommendation to the business if they want to promote this application to production?**
   The error rate kept increasing with the increase in the user, so that factor needs to be considered before taking the application to production. Otherwise, it will be great problem for the users when accessing the application concurrently.

5 CONCLUSION

To conclude, the performance of the app was good only when the number of users performing CRUD were less. The error rate was 0 for users till 150, however for 200 its 25% and 250 it went to approx.40% and it kept increasing with the increase in number of users. So, it did not pass the volume testing. For any test case, if users > 150, then only 151 samples were success all other were failed. So, it can handle 151 concurrent users at a time. On the other hand, considering the http request sent to the app had good response time without any failure.