CMPE 273 – Lab1

Canvas Prototype and Calculator Application.

Student ID: **013859989** **Name:** Laxmikant Bhaskar Pandhare

Goals of the above Applications:

The goal of this lab is to develop a Canvas Application wherein it provides the students and faculty with the access to colleges processes. The students can enroll to the students and able to view the coursework. The coursework contains Announcement, Quiz and course materials provided by the faculty of that subject. It has provision to upload images for profile and files of the subject. The entire development is based on React, Node and MySQL. Also, I have created the calculator application which does the calculation depending upon user input. The client side received the data and pass to the server for further processing of it.

Purpose of Developing System:

The main purpose behind making of Canvas Application is to give students the facility to enroll for course, upload assignment, give quiz and all other stuff related to coursework. Also, the faculty is having access to create a course, upload files, create quiz for the subject. Initially, the system validates the user. If user is valid then he can proceed with further functionalities on the dashboard else appropriate error will be thrown at the front end. This project can be utilized as an enrollment and college process during the semester. Also, the calculator is developed to do the mathematical calculations according to user request.

System Design and the implemented system description:

Canvas →

- The design process used to build Canvas system is MERN Stack (Backend for this is MySQL).
- This system is based on 3 tier architecture where backend processes will be handled and processed according to requirement by Node JS. Node JS process is a single thread driven process and all the data fetch and stored in MySQL. The react calls Node JS via axios and pass the input data from the user. After processing the data Node JS will send the response data to the again with response code 200 as a success.
- To connect with MySQL database, the library installed (MySQL) with command NPM install. Also, the sessions on the served side used to handle client processes. Same, I have used for logout process as well. This will check whether user is already logged into the system or not. In the logout process, all these fields reset to the original value.
- The system is designed using React and some part of react-native as a frontend technology and node JS as backend and MySQL as a database.

- To connect node JS and React and for the data passing from front end to backend and vice-versa, I have utilized axios call method. It will handle all the data passing processes. This is the easy way I found for passing the data from client to server and vice-versa.

- The best use of this process is that, the entire application is single page application which is the requirement of this project. In react, I have utilized Redirect method to divert from one page to another depending upon the user actions. For example, once the faculty clicked on Course Creation process it get divert to the Couse Create page. This all has been handled by redirect method in React JS.

- This process is based on user validation as well, if user is part of the system and is it is not faculty then diversion takes place for Student and if it is faculty then it will be diverted towards Faculty Dashboard. This process is performed by axios and redirect in React. This flow in Canvas application heled us to develop enterprise application.

- Also, AES which will be in CTR mode. The MD5 algorithm will be implemented in the next lab as it has more powerful structure as compare to AES. The encrypted password got stored in the table and it will be verified against user when user tries with the UserID which was provided by him at the time of signup process. This algorithm provides the security which will provide the cyber-attacks.

- The Login Page, this page will take and input from the user and check into the database for the user entry. If it is a valid user with valid password then he is allowing access to the services. If not, then appropriate message will be thrown at the front end. There is signup link added in the process.

- The signup process, this will get the data from user. It also gets user profile and store it in the backend node. This will be further get downloaded during Dashboard process.

- Dashboard, in this the student or faculty able to do the process according to his/her needs.

- Quiz, in this tab, the professor can create the quiz and the same quiz get availed by the respective student in the same section at the student side process.

- Announcement, in this tab the professor can make announcement with the coursed. The course id added as the professor may has multiple subjects with him. The student enrolled for that subject gets announcement in announcement tab.

- Files, this will help to have proper communication between professor and student for submission of assignments or sharing of lecture notes from professor end.

- Account tab, this tab will show the details of the student or professor.

- Profile, this tab shows the small profile photo of the student or faculty.

- Course, this tab will allow professor to create the new course.

- Thus, this design provides the most suitable real time application and fast processes.

Calculator →

- On the client side, the user puts his input and the process sends that data to the backend for processing of the data. In this, I have sent the value of the button whether it is addition, subtraction, multiplication or division.
- The server does the calculation of it and sends the response back to the user with the result.

Performance of the System:

The above developed application has below mentioned benefits.

- The system mains the proper data handling from client side to the server and server side to the client.
- These applications are very fast and provides the services in very minimal time.
- The complete flow of the application is as follow:

  React JS (Frontend or Client side) → API → Node JS → MySQL → Node JS → API → React JS (display response.)

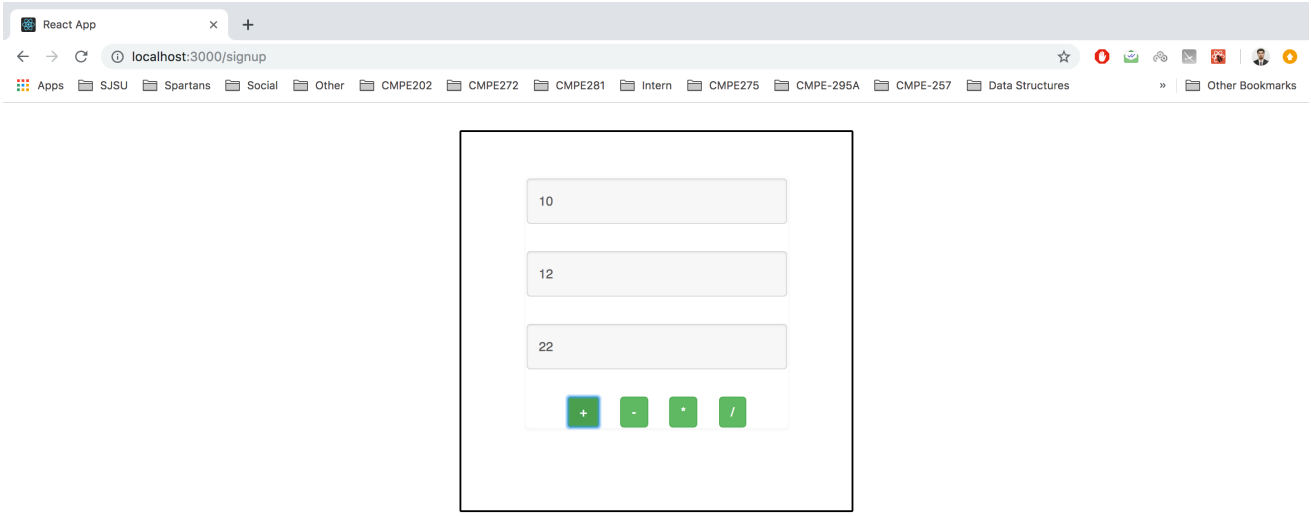
- The React JS is main part of the Frontend Side, the system processes only those parts which was requested by the student or faculty. It will not change the entire page.
- It has implementation of **AES** in **CTR** mode algorithm for password encryption facility. It will help to improve the security of the application. Also, in Node JS, I have added SQL queries in such manner which will prevent the SQL injection attacks.

  The combination of React as client and Node JS as server increases the performance by using functionality provided by both the JS.

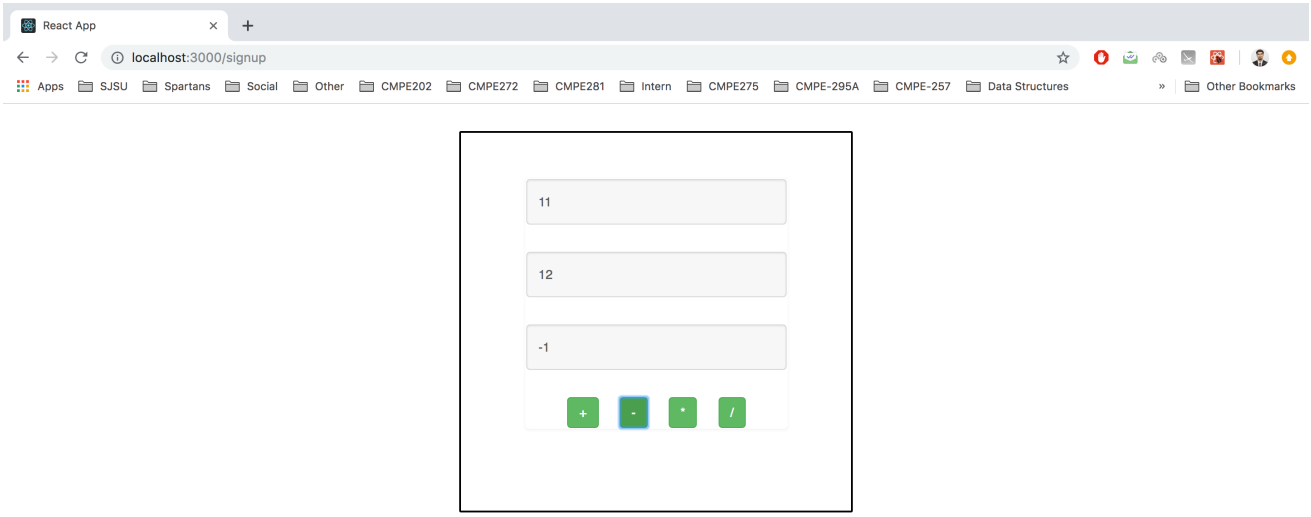
Calculator →

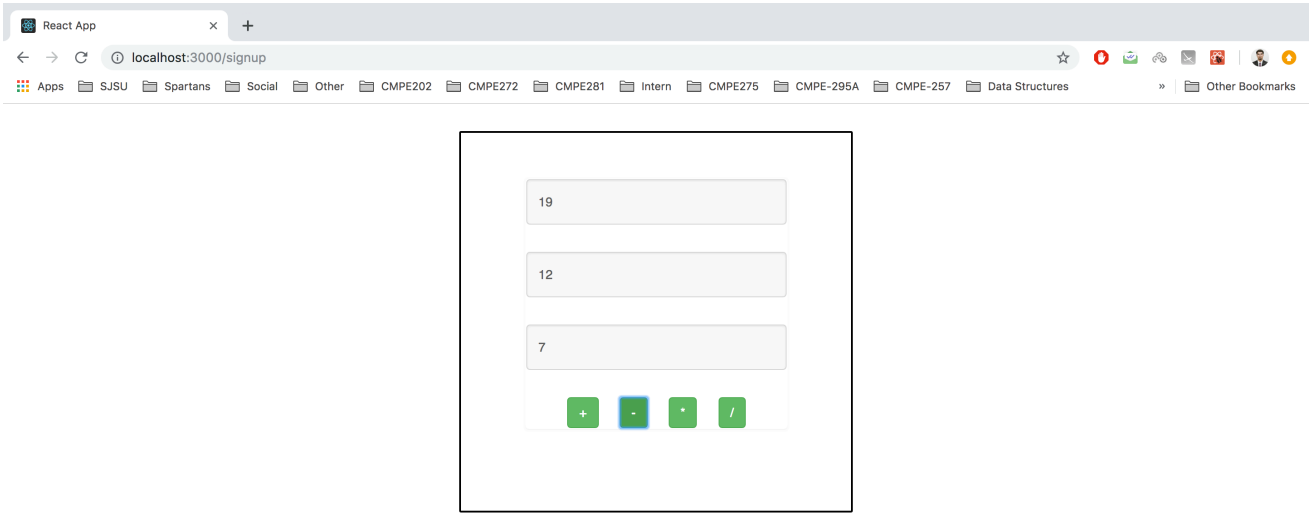
The Client side view and input from the user is:

Addition:

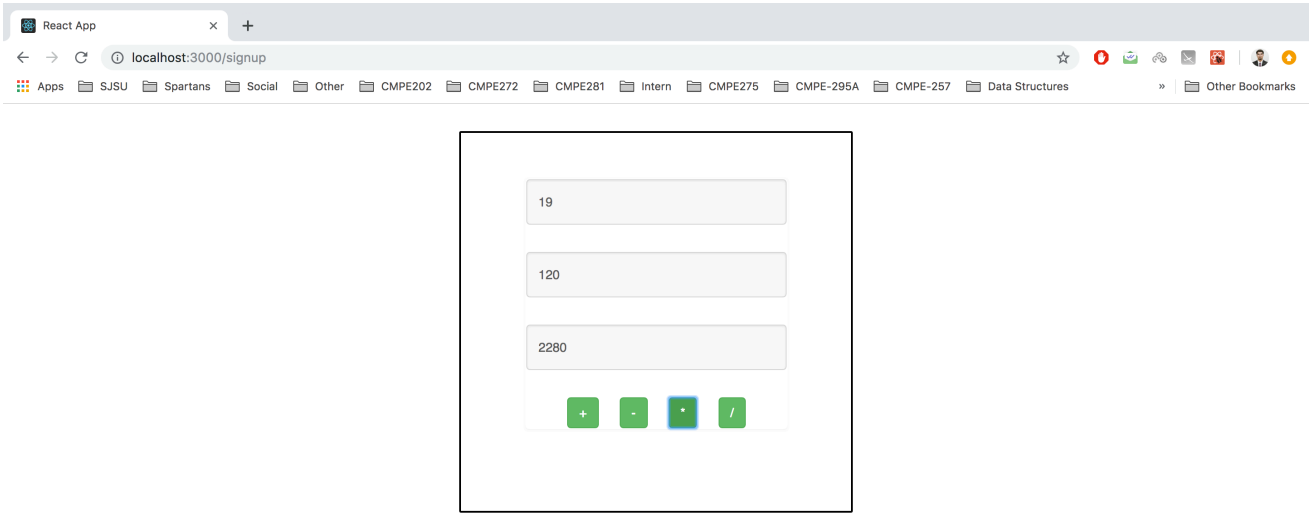


Subtraction:

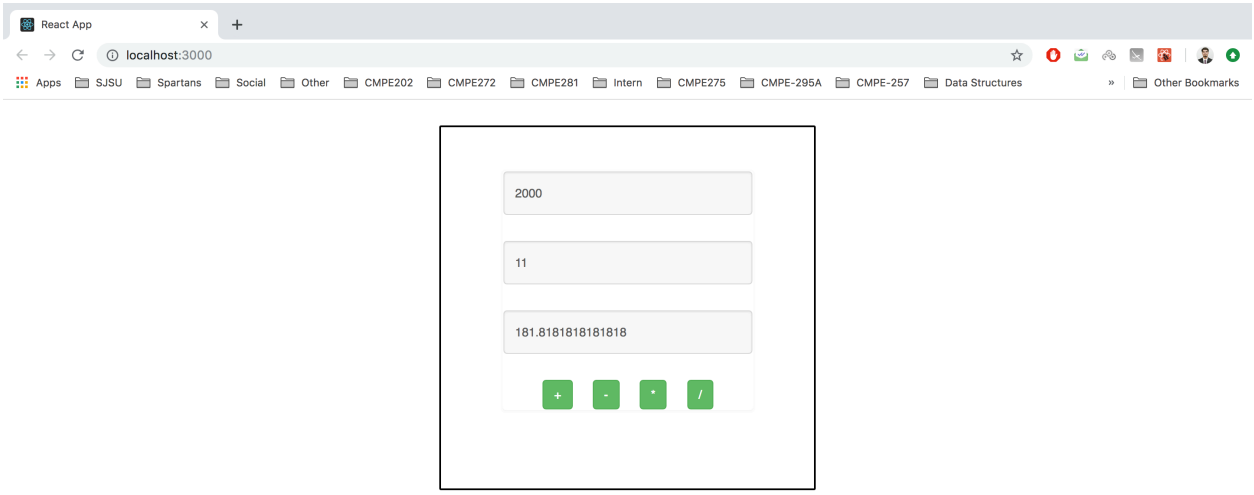




Multiplication:



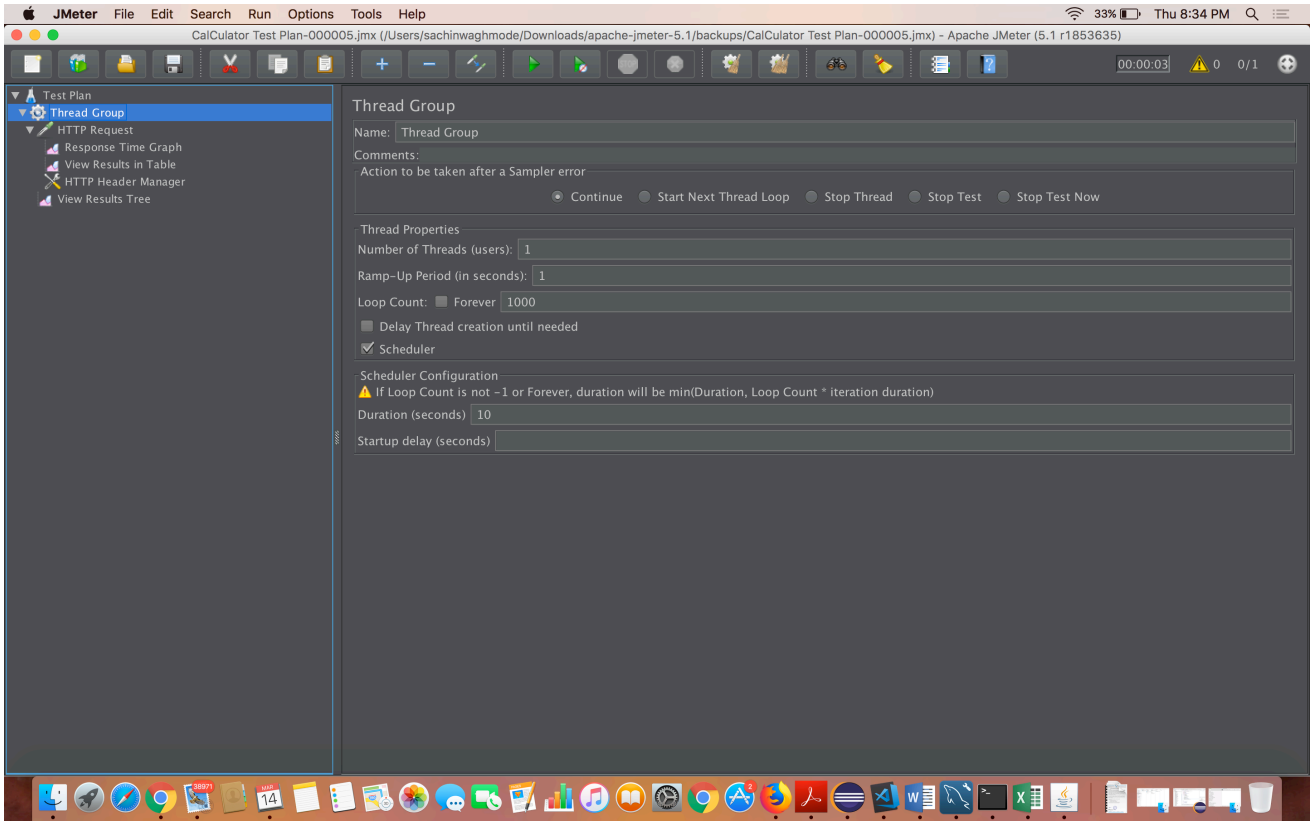
Division:

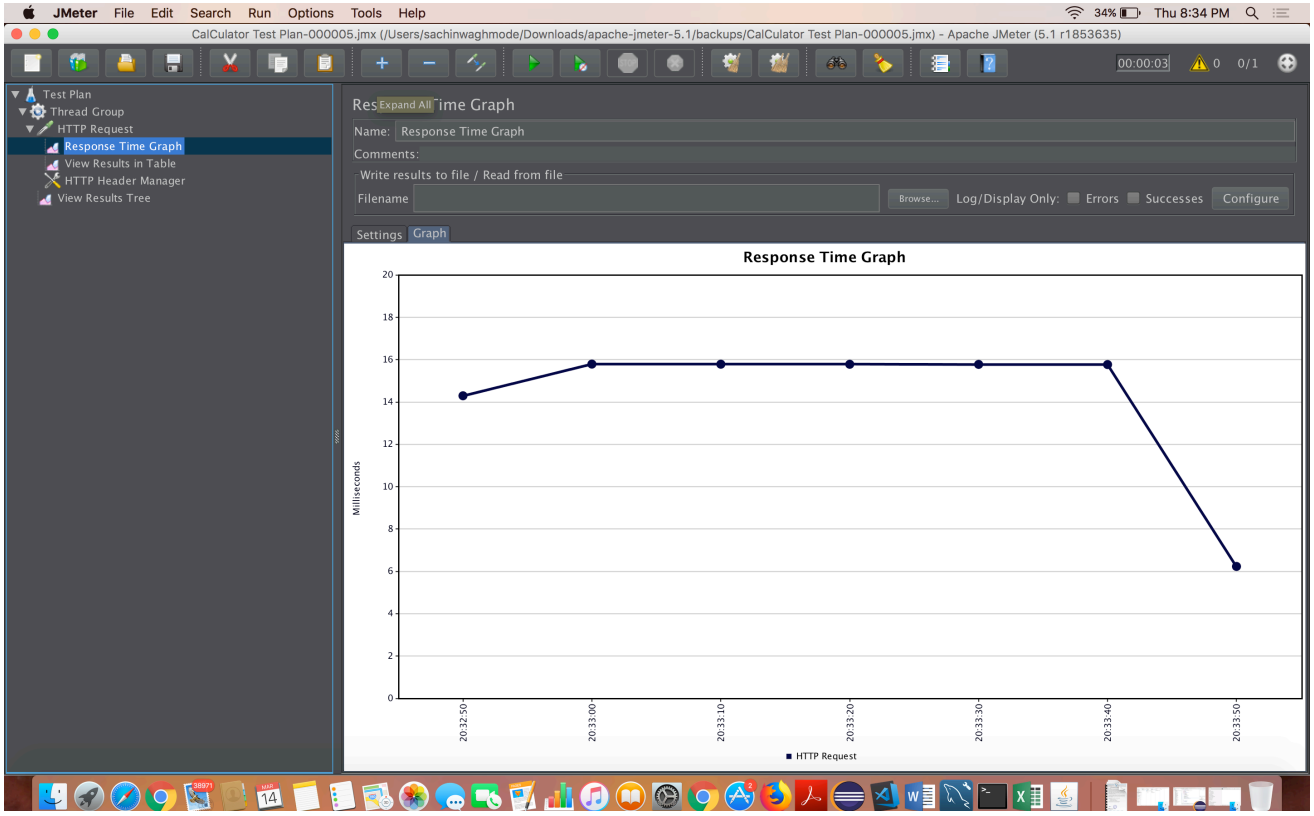


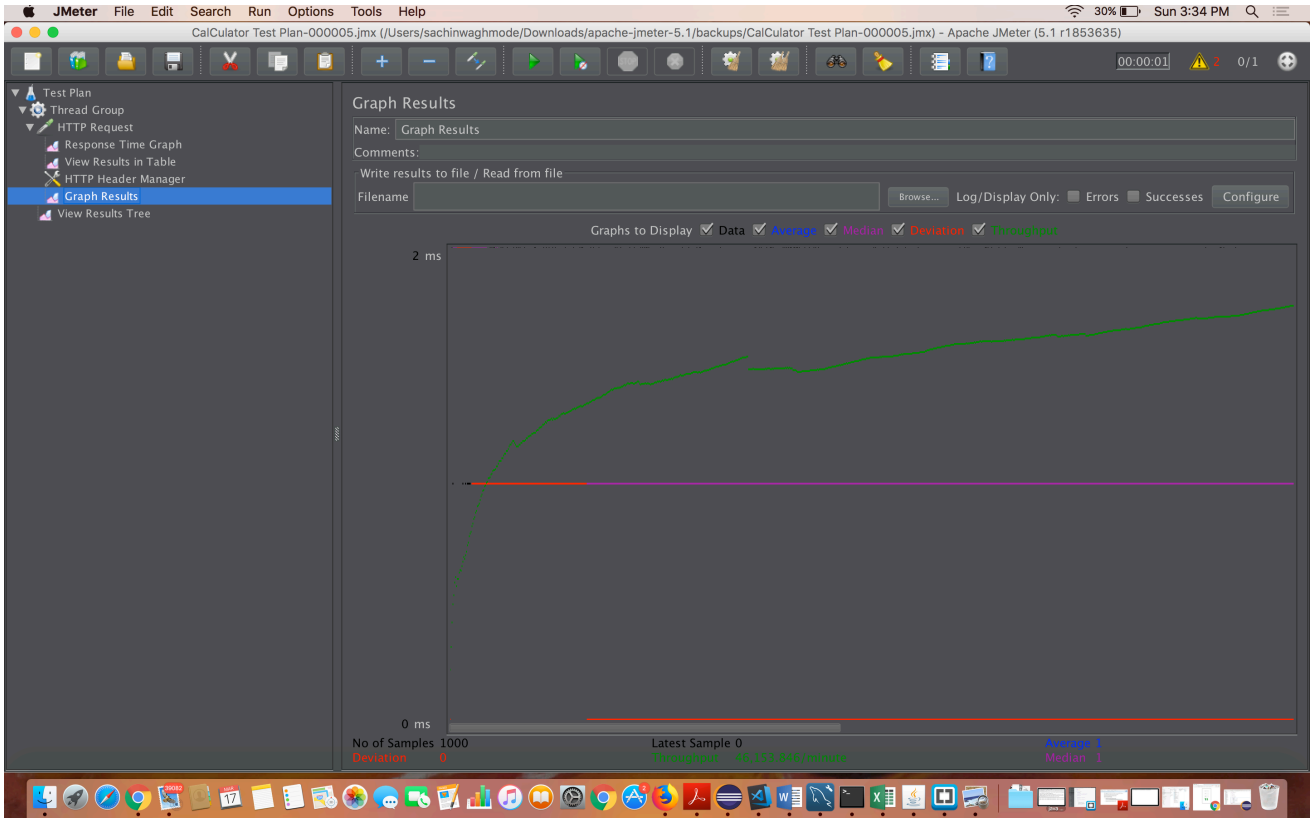
The results at the server side are.

I have tested this application by using Jmeter with different number of calls and users.

a) Invoke 1000 Calculator Calls.



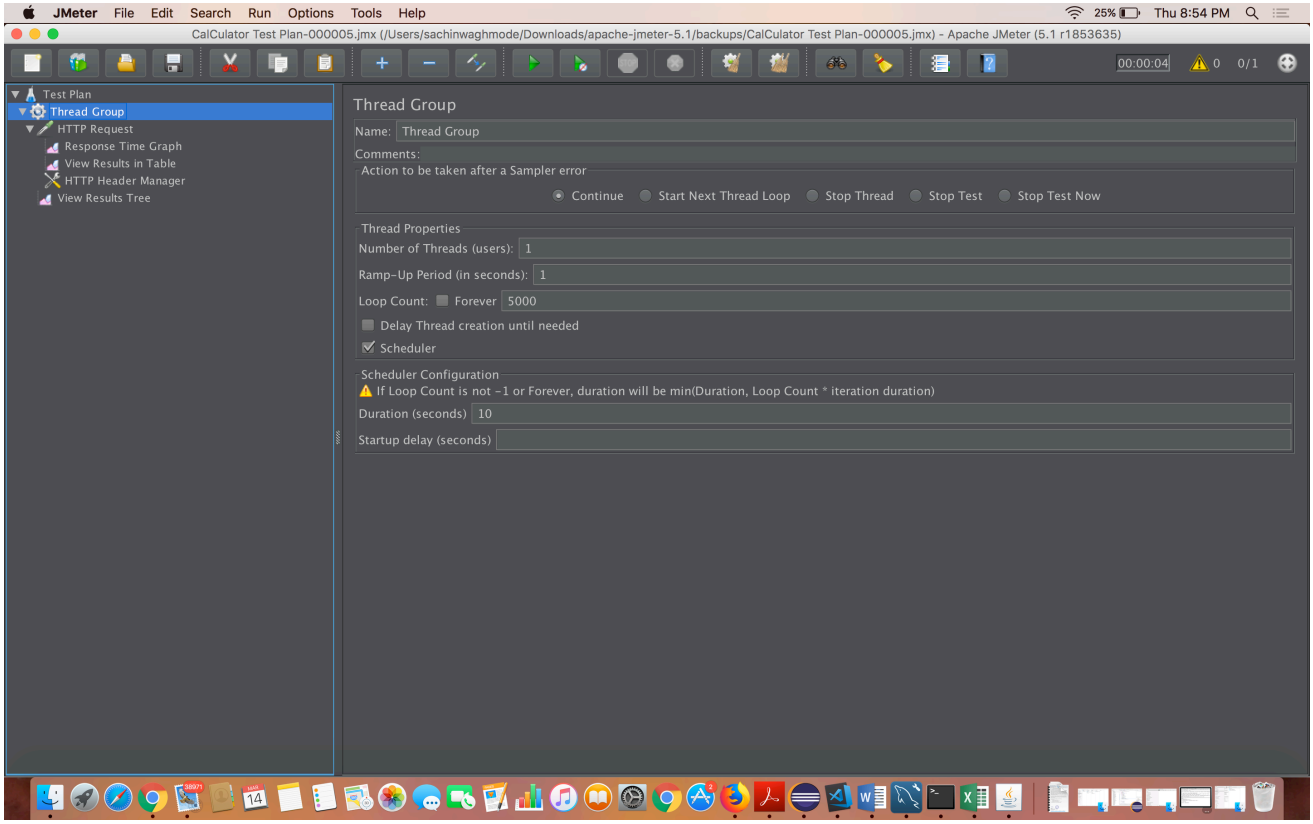


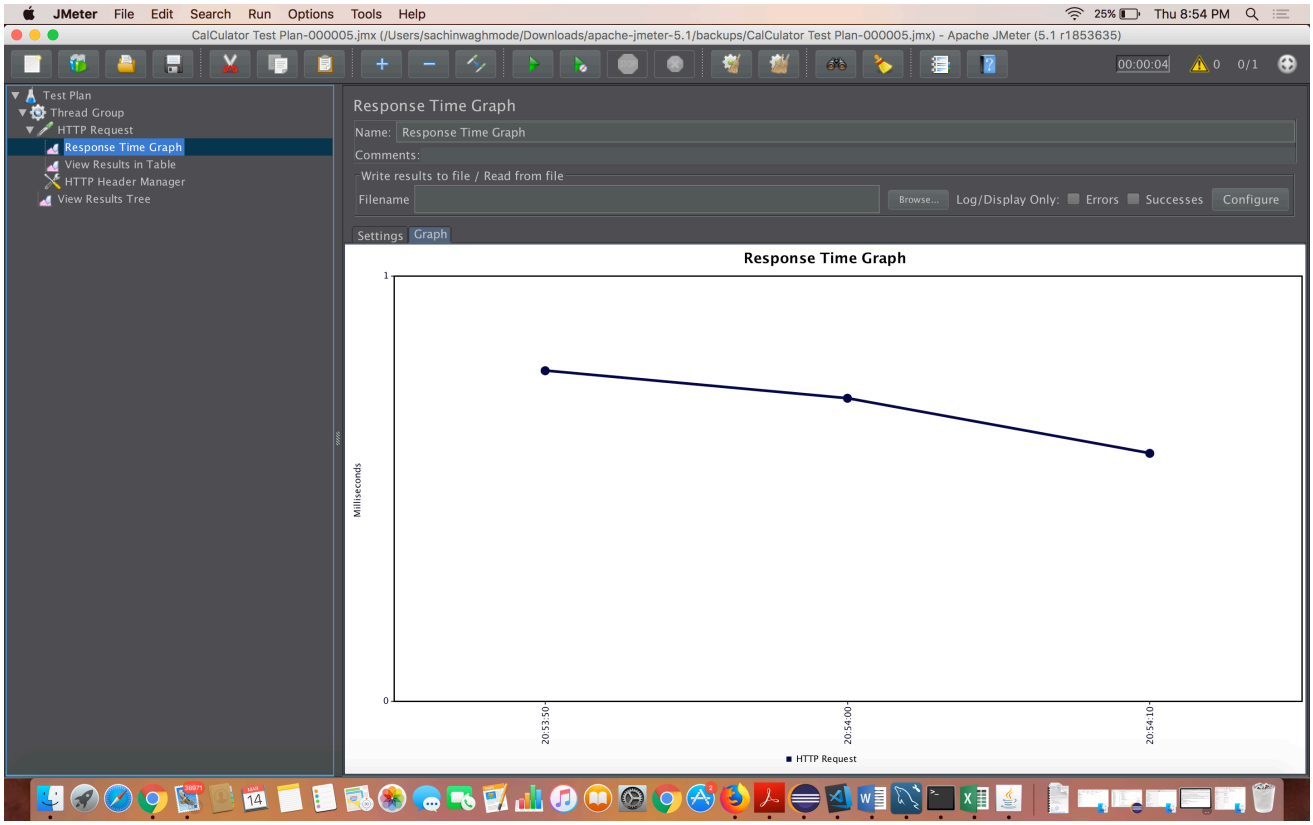


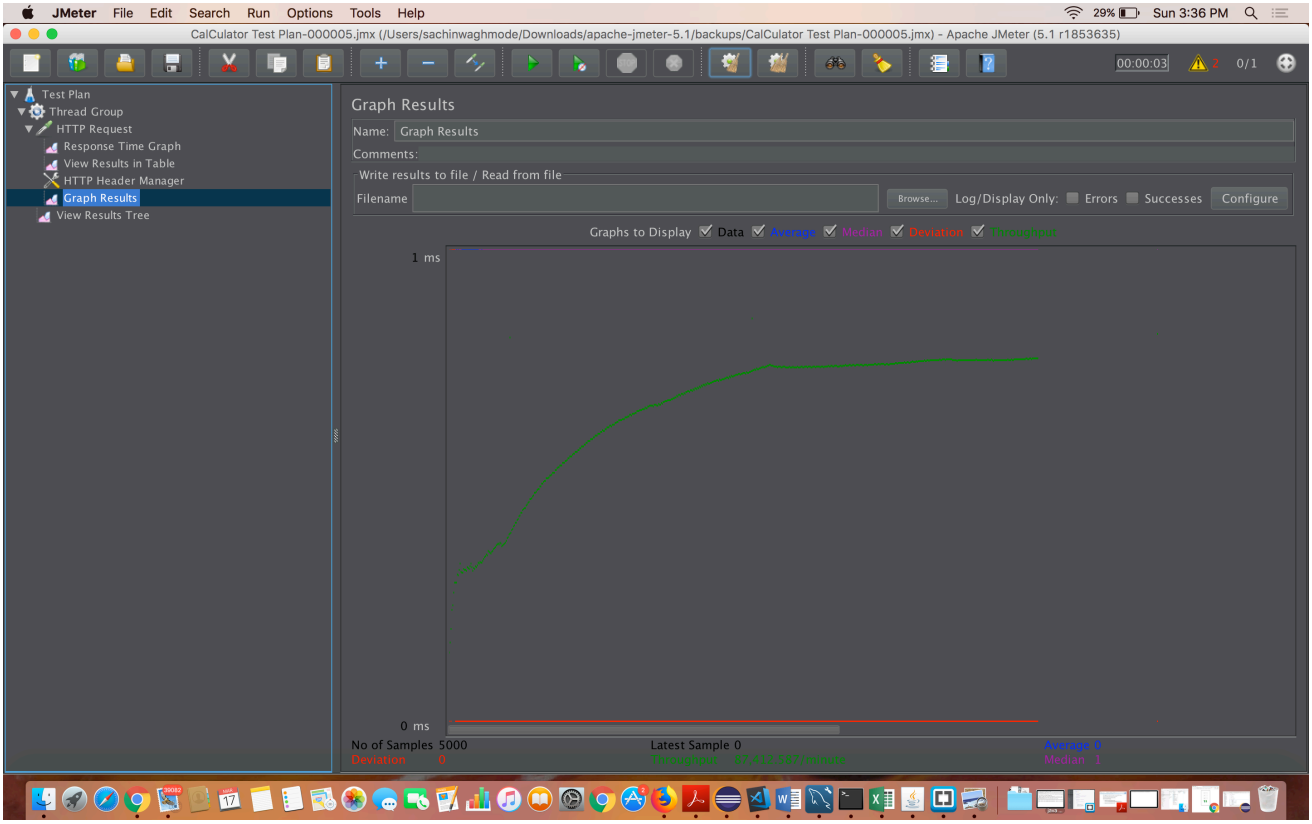
It took almost 3 seconds to perform the complete operations for 1000 calculator calls.

It means it took 3 milliseconds per call.

2) 5000 calls.


JMeter File Edit Search Run Options Tools Help
CalCulator Test Plan-000005.jmx (/Users/sachinwaghmode/Downloads/apache-jmeter-5.1/backups/CalCulator Test Plan-000005.jmx) - Apache JMeter (5.1 r1853635)
Test Plan
Thread Group
HTTP Request
Response Time Graph
View Results in Table
HTTP Header Manager
View Results Tree
Thread Group
Name: Thread Group
Comments:
Action to be taken after a Sampler error
Continue
Start Next Thread Loop
Stop Thread
Stop Test
Stop Test Now
Thread Properties
Number of Threads (users): 1
Ramp-Up Period (in seconds): 1
Loop Count: Forever 5000
Delay Thread creation until needed
Scheduler
Scheduler Configuration
If Loop Count is not -1 or Forever, duration will be min(Duration, Loop Count * iteration duration)
Duration (seconds) 10
Startup delay (seconds)

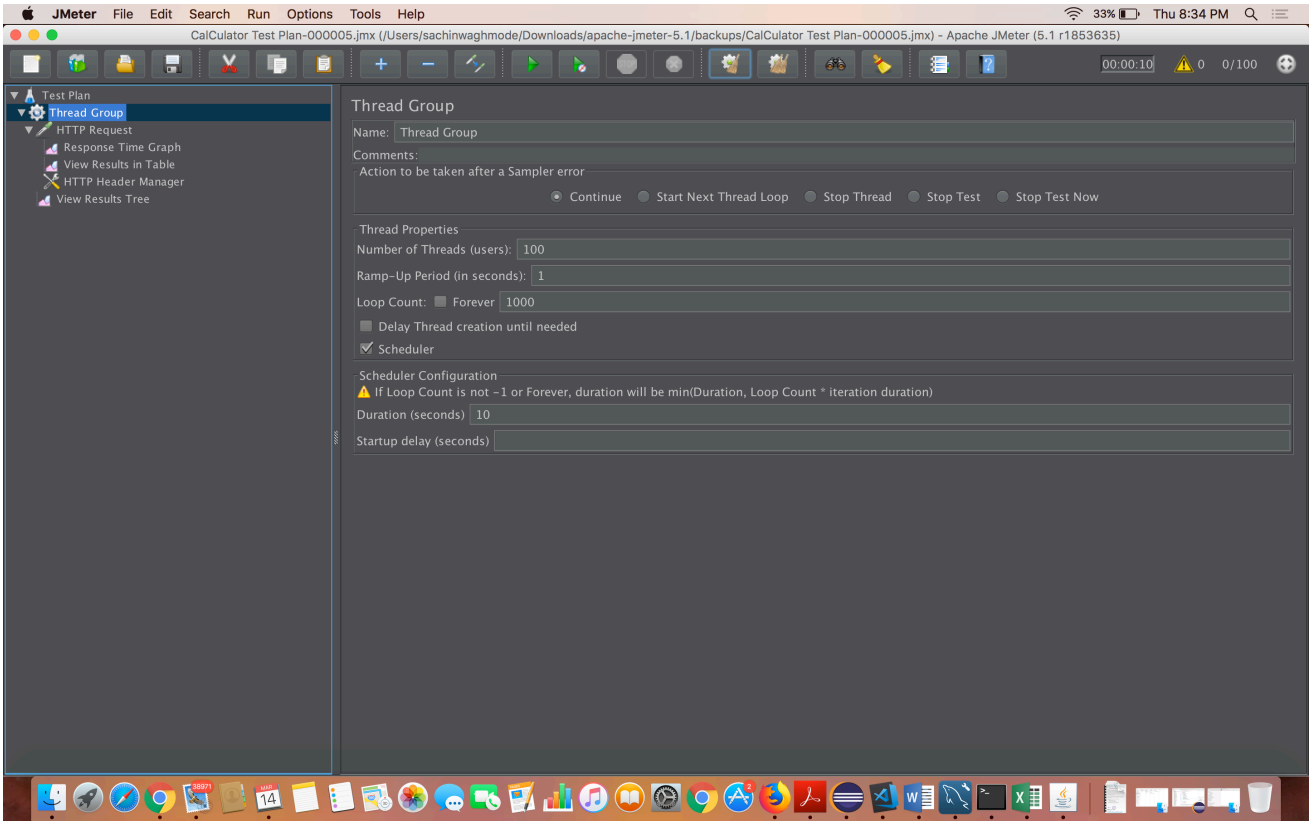

JMeter File Edit Search Run Options Tools Help
CalCulator Test Plan-000005.jmx (/Users/sachinwaghmode/Downloads/apache-jmeter-5.1/backups/CalCulator Test Plan-000005.jmx) - Apache JMeter (5.1 r1853635)
Test Plan
Thread Group
HTTP Request
Response Time Graph
View Results in Table
HTTP Header Manager
View Results Tree
Response Time Graph
Name: Response Time Graph
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only: Errors Successes
Configure
Settings
Graph
Response Time Graph
Milliseconds
1
0
20:53:50
20:54:00
20:54:10
HTTP Request

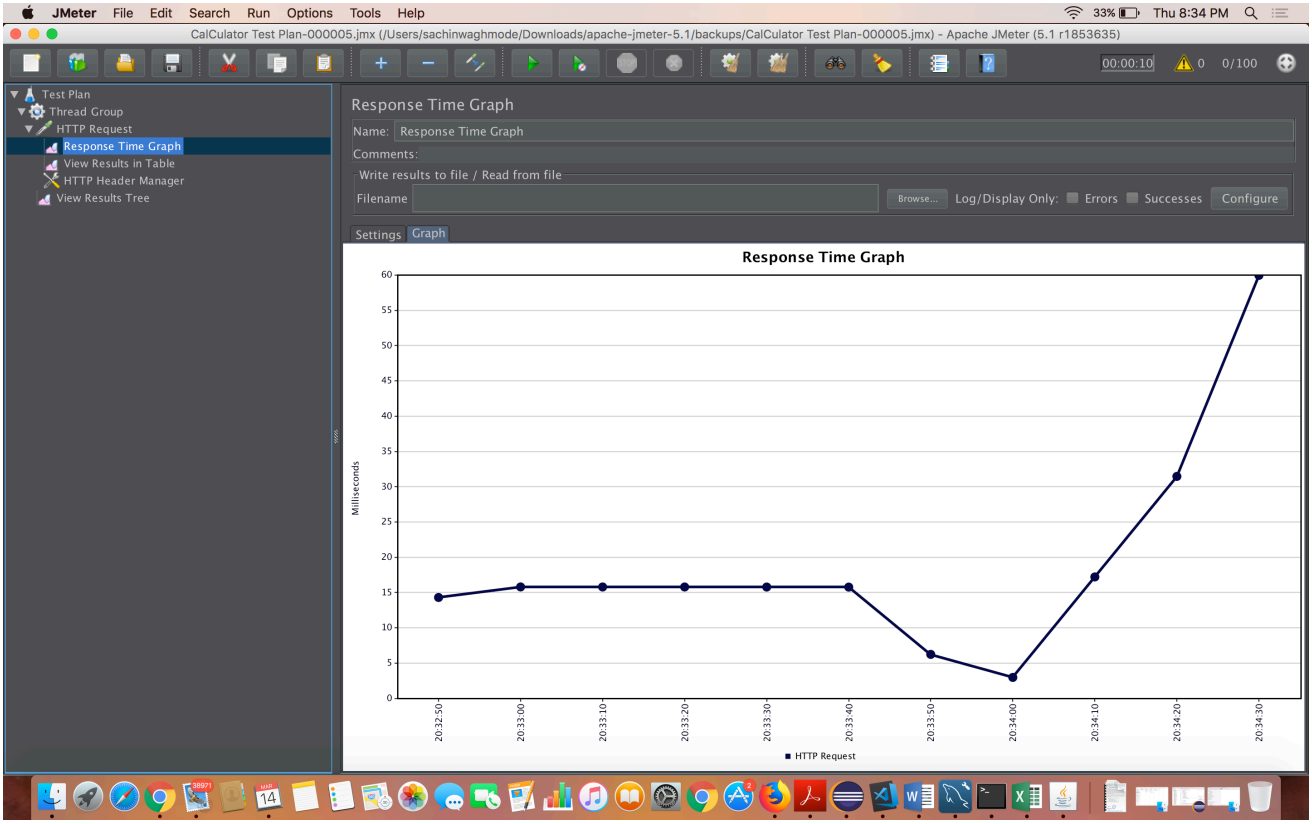


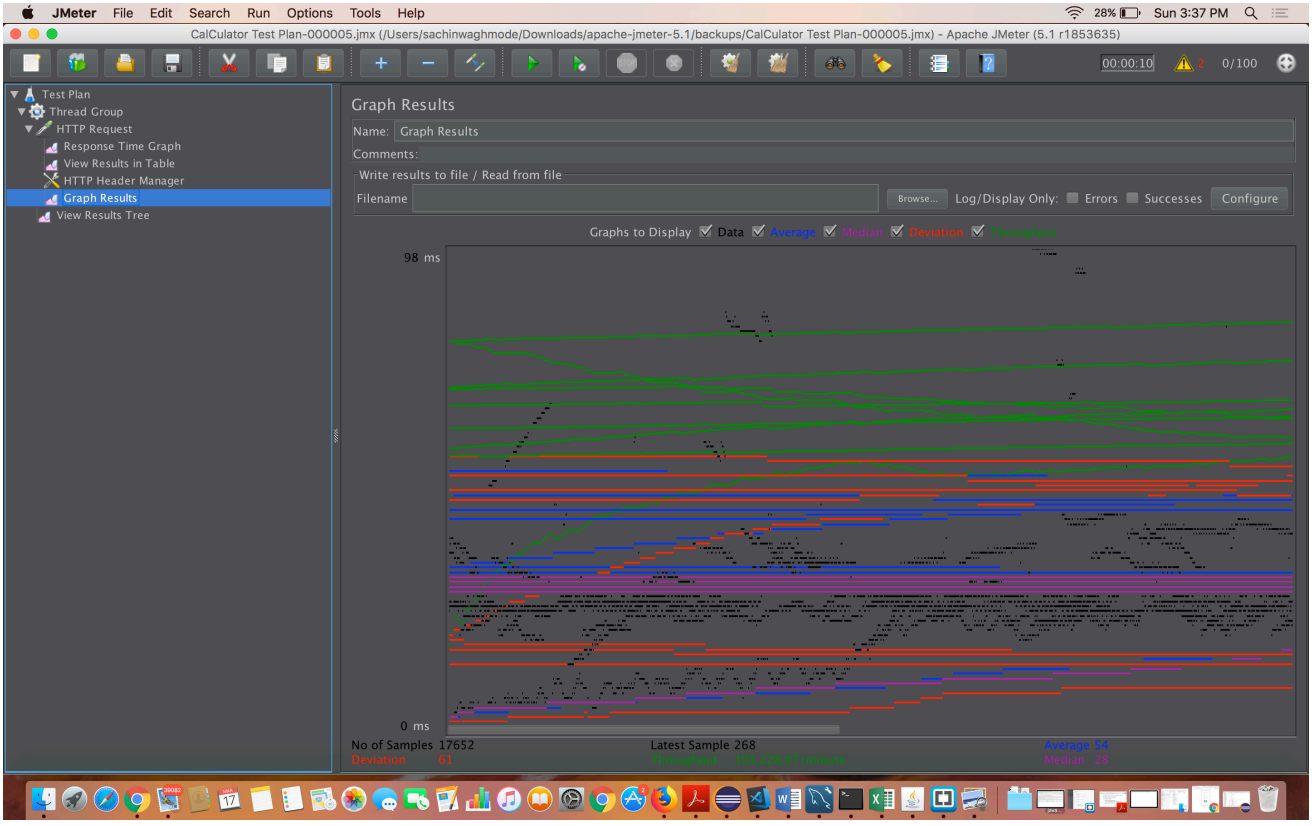
It took 4 seconds for complete 5000 calculator calls.

It means, 0.8 milliseconds for each Calculator call.

3) 100 users with 1000 calls

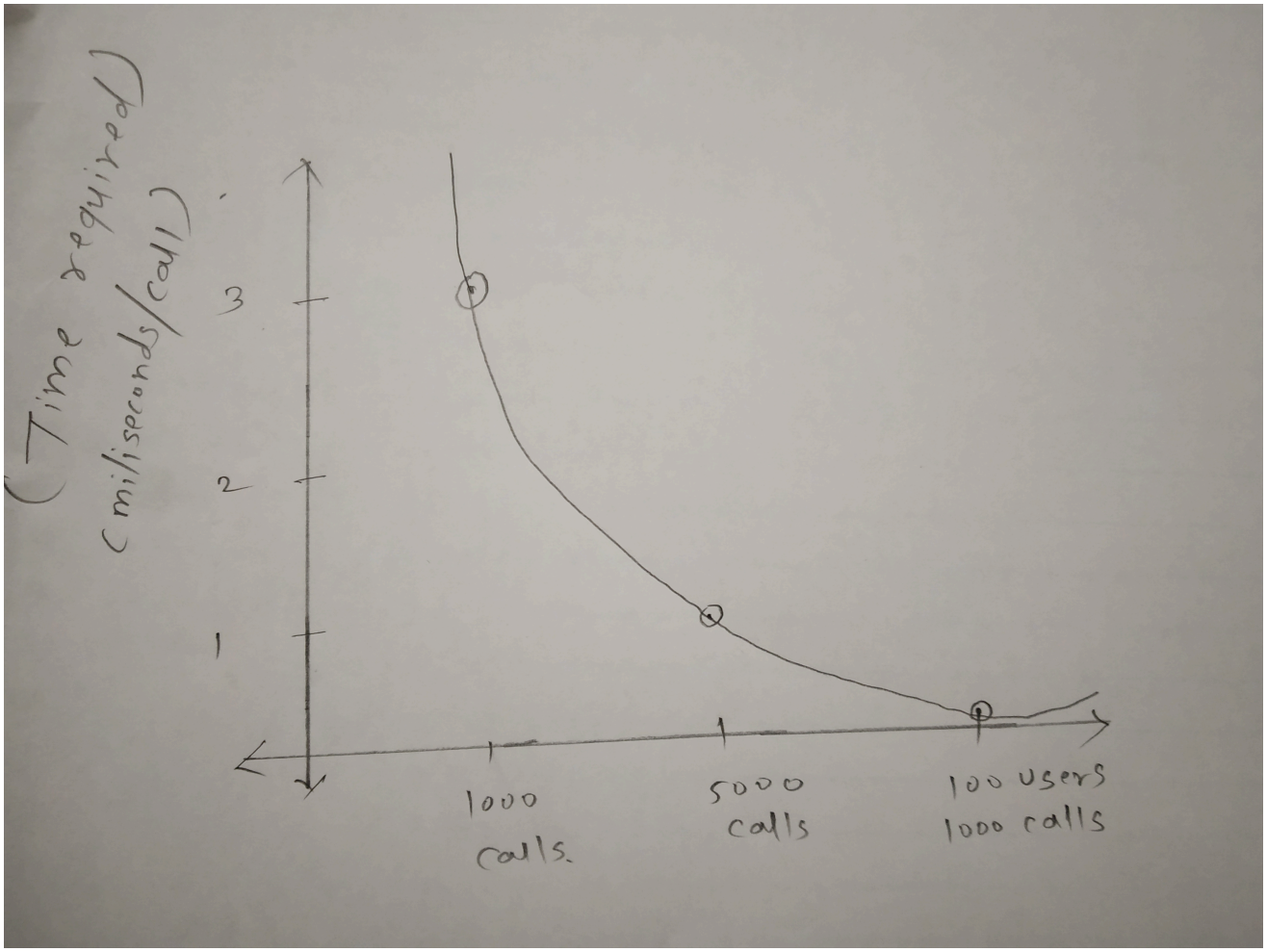






It took almost 10 seconds for completion of this process.

It means, 0.1 millisecond for each call.



Performance comparison:

Conclusion in above testing is that, the number of calls to the service increase the speed of the application got increased.

1000 calls → 3 milliseconds per call.

5000 calls → 0.8 milliseconds per call.

100 user and 1000 calls → 0.1 milliseconds per call.

Questions:

1. **Explain the encryption algorithm used in your application. Mention different encryption algorithms available and the reason for your selection of the algorithm used.**

→ Encryption is nothing but the way of hiding the sensitive data from the attacker. Through this process, the sensitive data will have encrypted by using some algorithms and stored at the backend side. Like, password in our application which is very sensitive data through this someone can access your profile.

Below are the algorithms used in the industry now days.

a. MD5

b. RSA

c. AES

d. SHA

e. Blowfish

In this Canvas project, I have utilized AES algorithm in CTR mode. This algorithm uses block cipher to provide data security. AES stands for Advanced Encryption Standard. This algorithm is based on substitution-permutation network. It performs series of operations in which some of the data will be replaced that is substitutions and some of the shuffled that is permutations.

The CTR mode used in Canvas is have high speed in all the subparts of AES algorithms (CBC, OFB, CFB). CTR created pseudo random stream that is not dependent of plain data. The different pseudo random streams are multiplied by maximum data length. In this, decryption and encryption done with parallelizable. This is more useful because it creates stream cipher from block cipher.

The rule of this algorithm is that never use same key with IV every time. Due to which, this algorithm provides more security with faster speed. It provides security as the initial counter value will be unpredictable. AES-128 CTR with 64-bit unpredictable counter provides more strengths against precomputation attacks.

2. **Compare the results of graphs with and without in-built MySQL connection pooling of database. Explain the result in detail and describe the connection pooling algorithm if you need to implement connection pooling on your own.**

→ I have tried with and without connection pooling process for the login page and attached screen-prints in the Jmeter testing section. It is nothing but re-use of connection created with database.

Connection pooling is mainly used to optimize the performance of the system while fetching data from the backend tables in a high load environment where connection maybe delayed due to this process. It is simple a cache of an database. In web environment, the connection pooling opens the database and after completion of process it will closes the connection.

1. Start with the connection with the database which is requested by the user.
2. Then data source will acquire the pool.
3. Further, pool get connected with the database requested by the user.
4. Entire requested data got passed to the front end or requested service so that user can able to see the data.
5. When the next time, the connection is requested, the pooling data source use the available connections pool to acquire new connection.
6. The pool close() method only return connection to the pool. It does not close the connection with the database.
7. In this way, connection pooling algorithm works.

The connection pooling provides speed to the application. Also, connection pool is safer as compare to the normal connection. But, in this process the process needs to handle maximum & minimum size, the total time it takes for operation, acquire timeout.

3. **What is SQL caching? What all types of SQL caching are available, and which suits your code the most. You don't need to implement the caching, write pseudocode or explain in detail**.

→ SQL catching is used to speed up the application. The canvas application contains lot of processes where Catching can be implemented. During the storing of files, images or even fetching the data from SQL tables is the slower process. The catching process can be used to retrieve them with high speed. Cache stores file into the memory and when user request the service it will provide it immediately.

The initial select of data from the table will be stored in the cache memory. And the next identical selects will be faster as we have data already stored in the cache memory.

Pseudocode for SQL catching →

a. Initially, consider the memory of your local system.

b. Hard disks are slow but the cache of your laptop is fast but it has less memory length.

c. In this process, most of the data got stored on the system and get called whenever it is needed and requested by the user. But, this process is very slow and requires more time for small data fetch which user uses frequently.

d. To speed-up your process, the data which requires more frequently to the user was accessed via some buffers. Some amount of memory allocated to these buffers so that it speed-up the entire user request.

e. In SQL catching, to save the time and speed up the application SQL catching method is used. In this, this technique is used to improve the delay due to transit time in the application. The SQL catching operation speed up the data read and write operations.

f. So, when user request the data then SQL process stores this data in SQL cache, it will be stored in buffer so that next time user can access it faster as compare to the initial fetch.

g. Due to this, the operations which requires many database fetch calls will be performed well as data is already part of cache. No need to hit the database due to which time required for application will be reduced.

h. But, there are certain limitations for it. It has only 8k pages. We cannot store more than that in Cache memory.

i. Once the memory gets full then the initial data will be replaced by using page replacement algorithm.

j. Memory cache is the best technique to improve the performance of any of the application.

4. **Is your session strategy horizontally scalable? If YES, explain your session handling strategy. If NO, then explain how you can achieve it.**

→ Yes, the Canvas system is horizontally scalable. The scalability is nothing but resize the resource usage to match the needs. The application needs to have scalability during the time of heavy loads. The primary reason of scalable is loose coupling. If the system is tightly coupled then it is not scalable. Vertical scaling is nothing but add the resources in a single node system. And in other side, horizontal scaling adds multiple resources.

I have used sessions library in the node based server.

Due to this, the data is available across the all servers and client-server processes.

Due to this, it can be accesses from anywhere and it request and check the cookie.

Whenever user logs into the system, I have set the flag that will tell that the current user is already logged in. That the main reason, due to which user can open the multiple tabs and access his profile in the canvas application process.

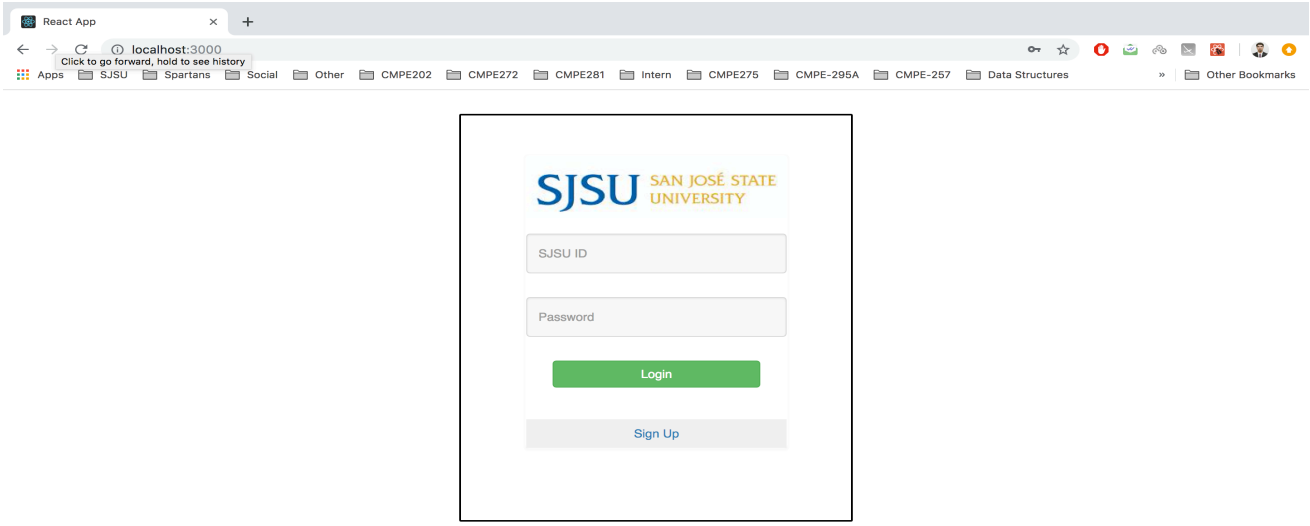
In Canvas application, once the user logs into the system the flag got set for that respective user and it was maintained using sessions and due to this reason only the Canvas application is horizontally scalable.

Canvas →

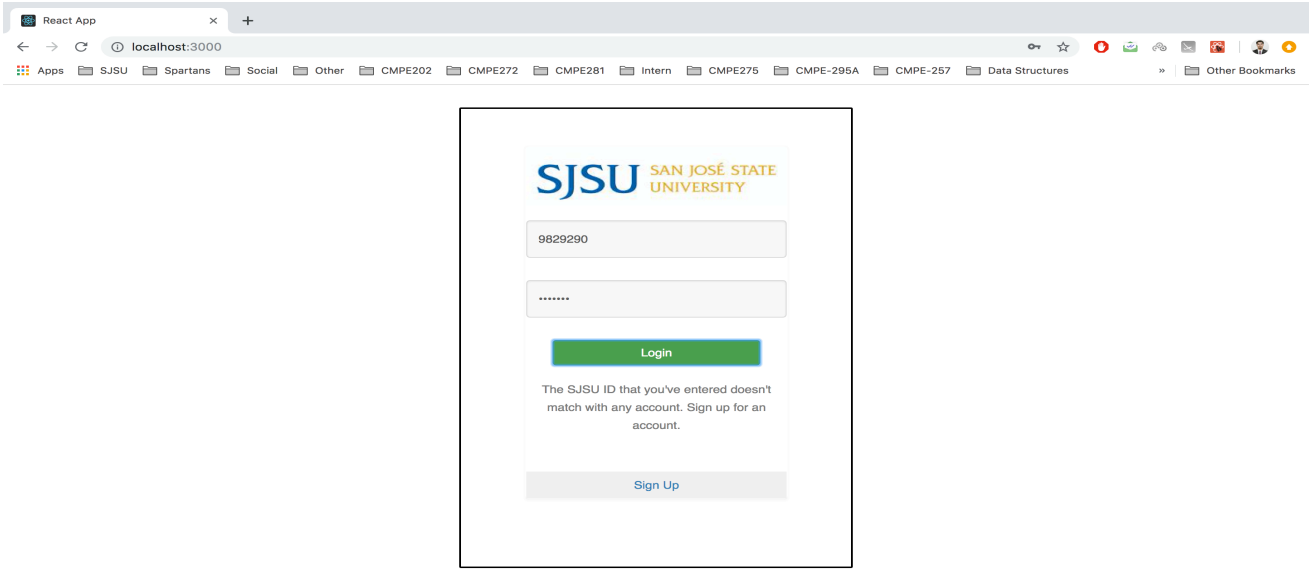
Login Page →

Initially, if user student the wrong username and password then appropriate message will be and if student enters valid password then it got diverted to the Dashboard page.

The page when React app ran for the first time.

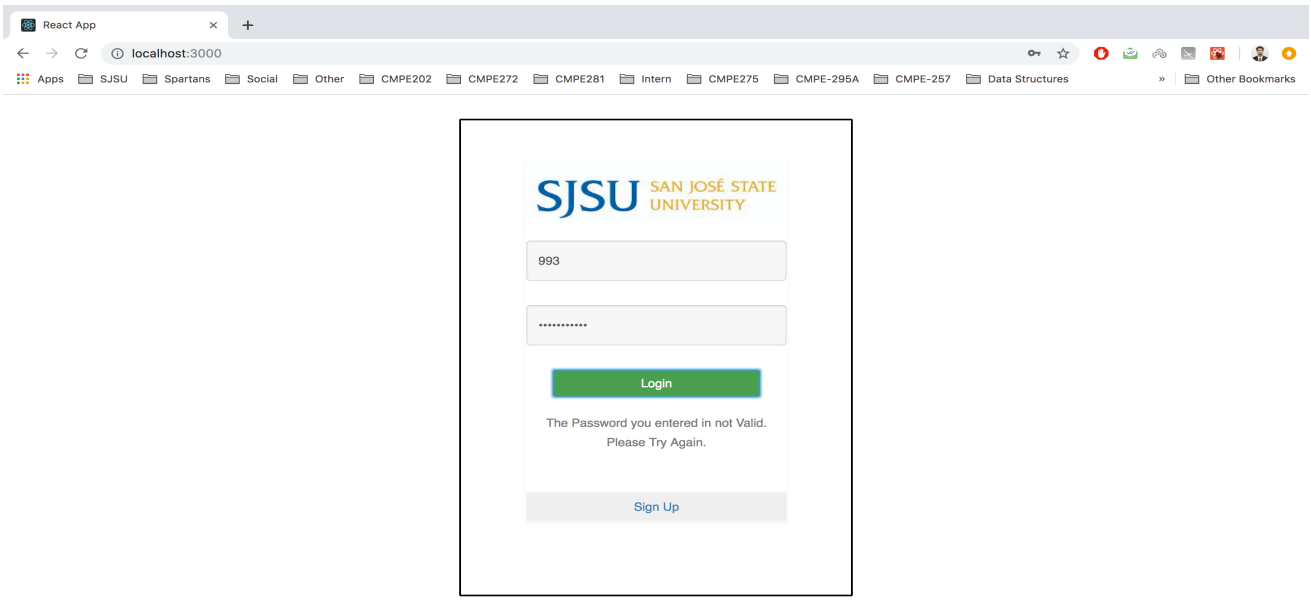


a. If user enters invalid user name.



```
Sachins-MacBook-Pro:Backend-Code sachinwaghmode$ node index.js
Server Listening on port 3000
Inside Login Post Request
Req Body :  { sjsuId: '29921', password: 'ejdsk' }
29921
val1sjsuid 29921
val1pwd ejdsk
ecrypted 0184d0832a
undefined
(node:64697) [DEP0106] DeprecationWarning: crypto.createCipher is deprecated.
(node:64697) Warning: Use Cipheriv for counter mode of aes-256-ctr
Row not Found
[{"finalstatus":false,"facultyfnd":false,"pwdvalidity":false}]
```

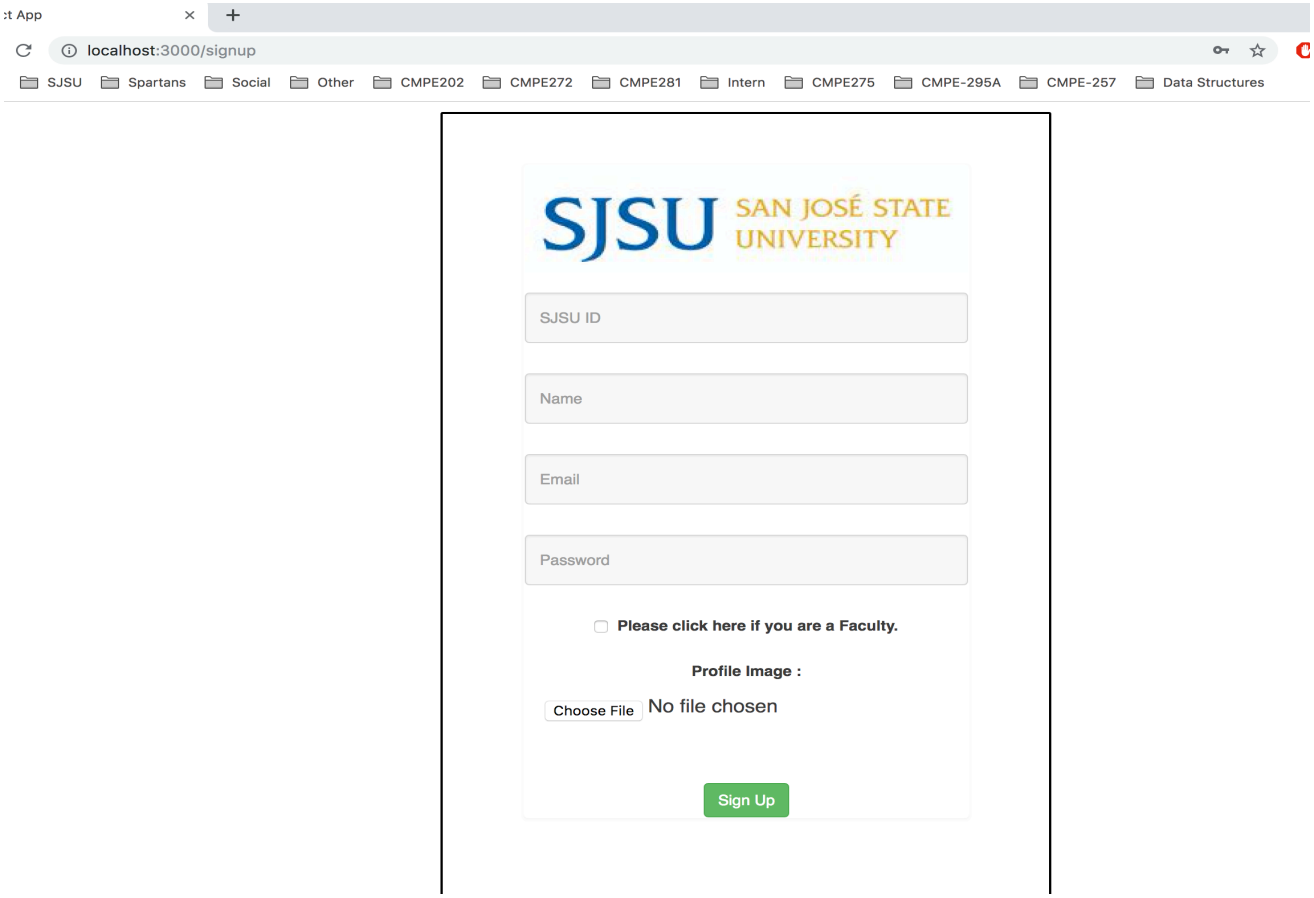
b. If the student enters invalid password with correct SJSU ID.



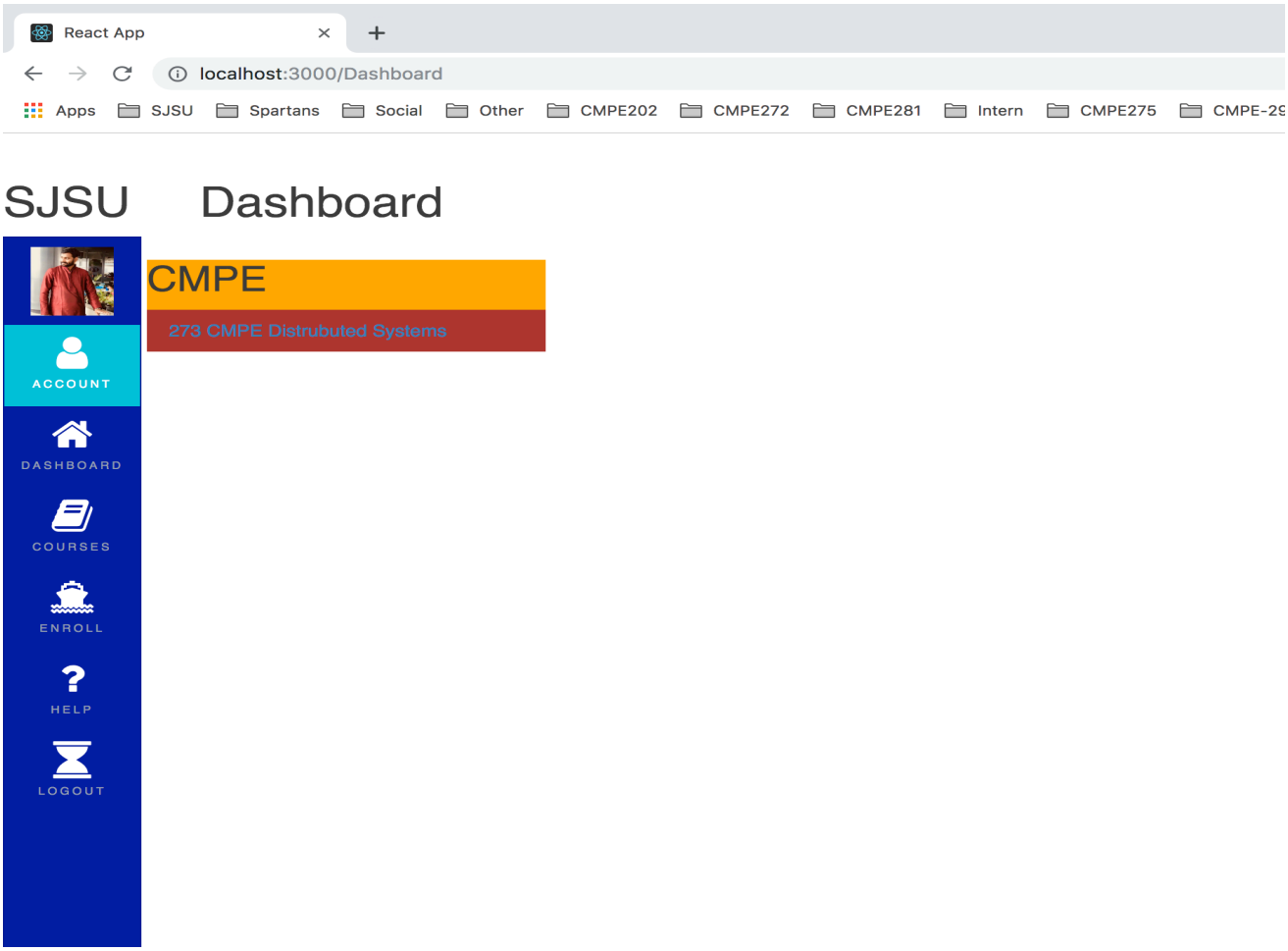

```
[{"finalstatus":false,"facultyfnd":false,"pwdvalidity":false}]
Inside Login Post Request
Req Body :  { sjsuId: '992', password: 'dsjk' }
992
val1sjsuid 992
val1pwd dsjk
ecrypted 009dde9b
undefined
(node:64697) Warning: Use Cipheriv for counter mode of aes-256-ctr
res:  [ RowDataPacket { user_flag: 'Y', password: '5dd786' } ]
[ { finalstatus: true, facultyfnd: false, pwdvalidity: false } ]
results[0].user_flag Y
Invalid Pwd
[{"finalstatus":true,"facultyfnd":true,"pwdvalidity":false}]
```

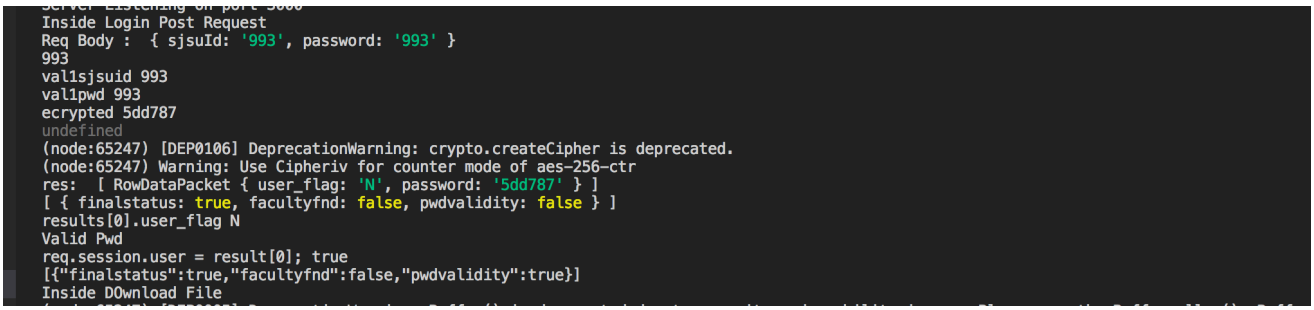
Ln 406, Col 17 Spaces: 2 UTF-8 LF Javasc

c. If the User clicks on Signup Page.



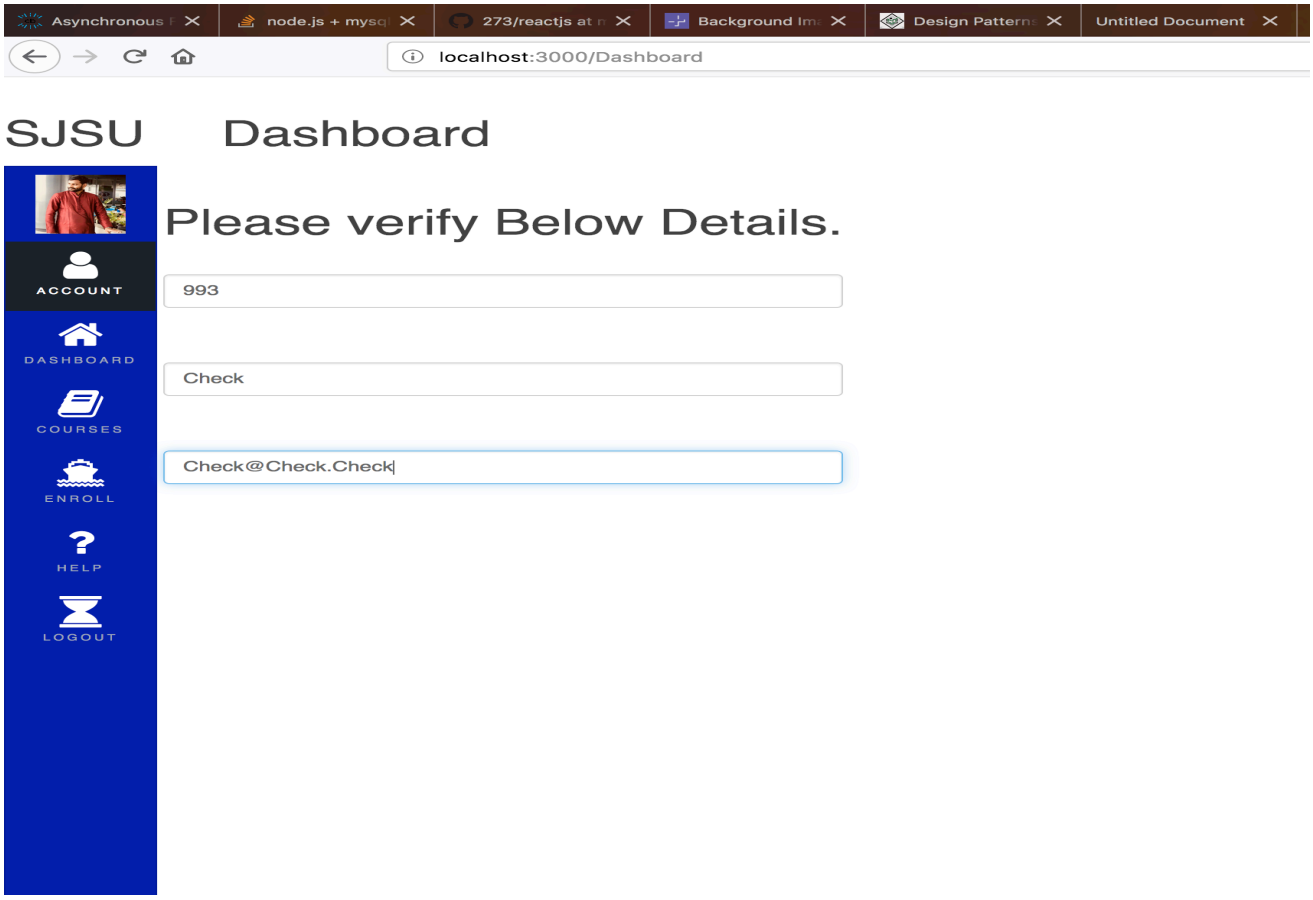
d. If Student enters valid details then it got diverted to the Dashboard page.



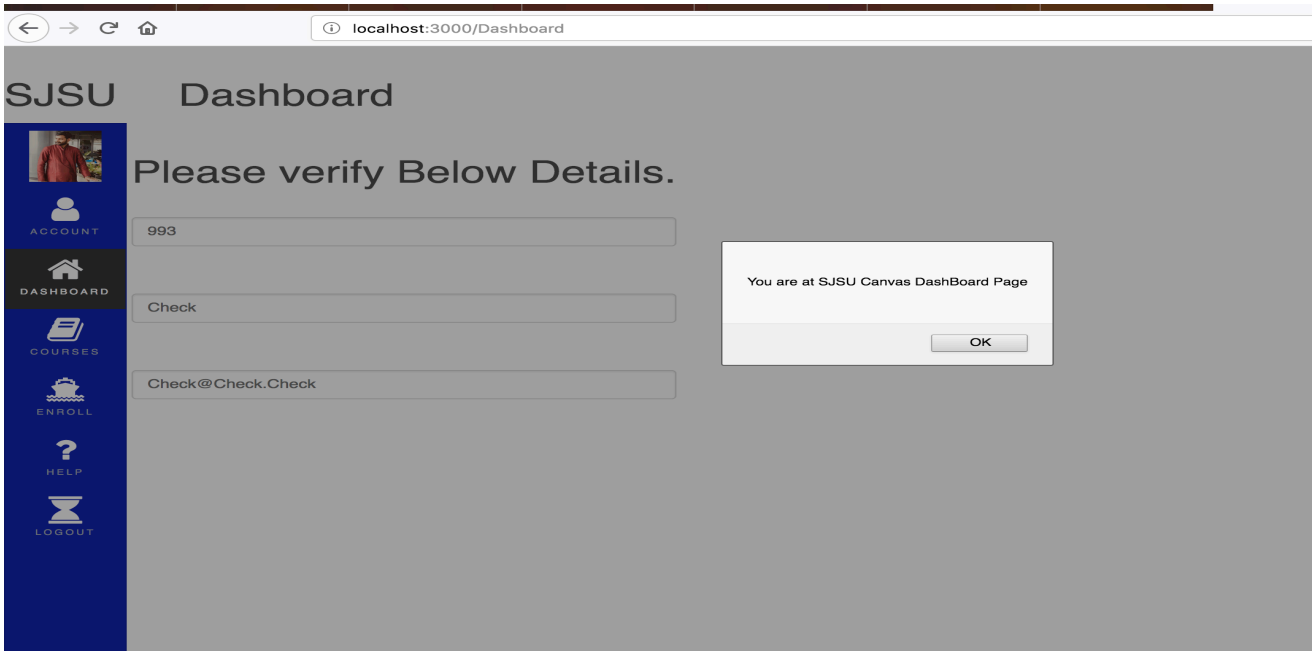


On the Dashboard Page, the Logout page will destroy the session and Student/faculty will be back to login page. At the top, the image of the student got downloaded and added as a profile of an Student.

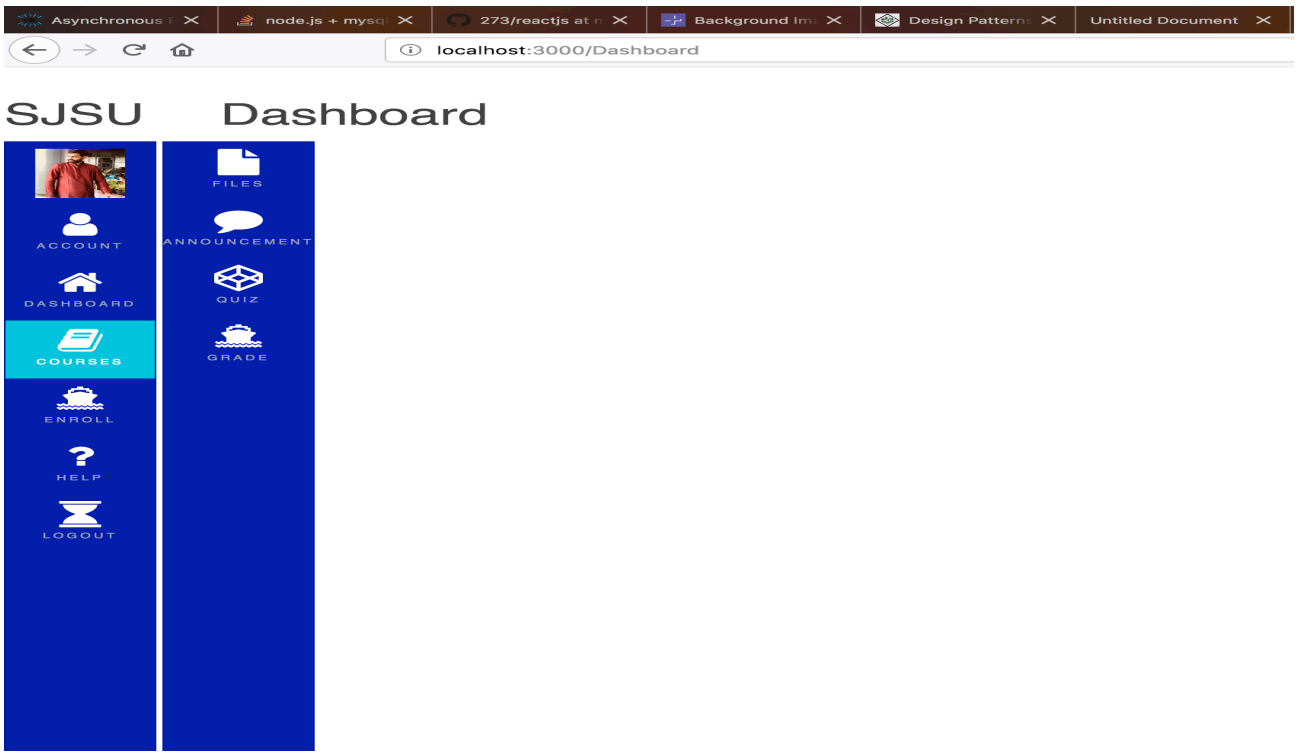
2. Once Student Click on Account, it will display the details of the user.

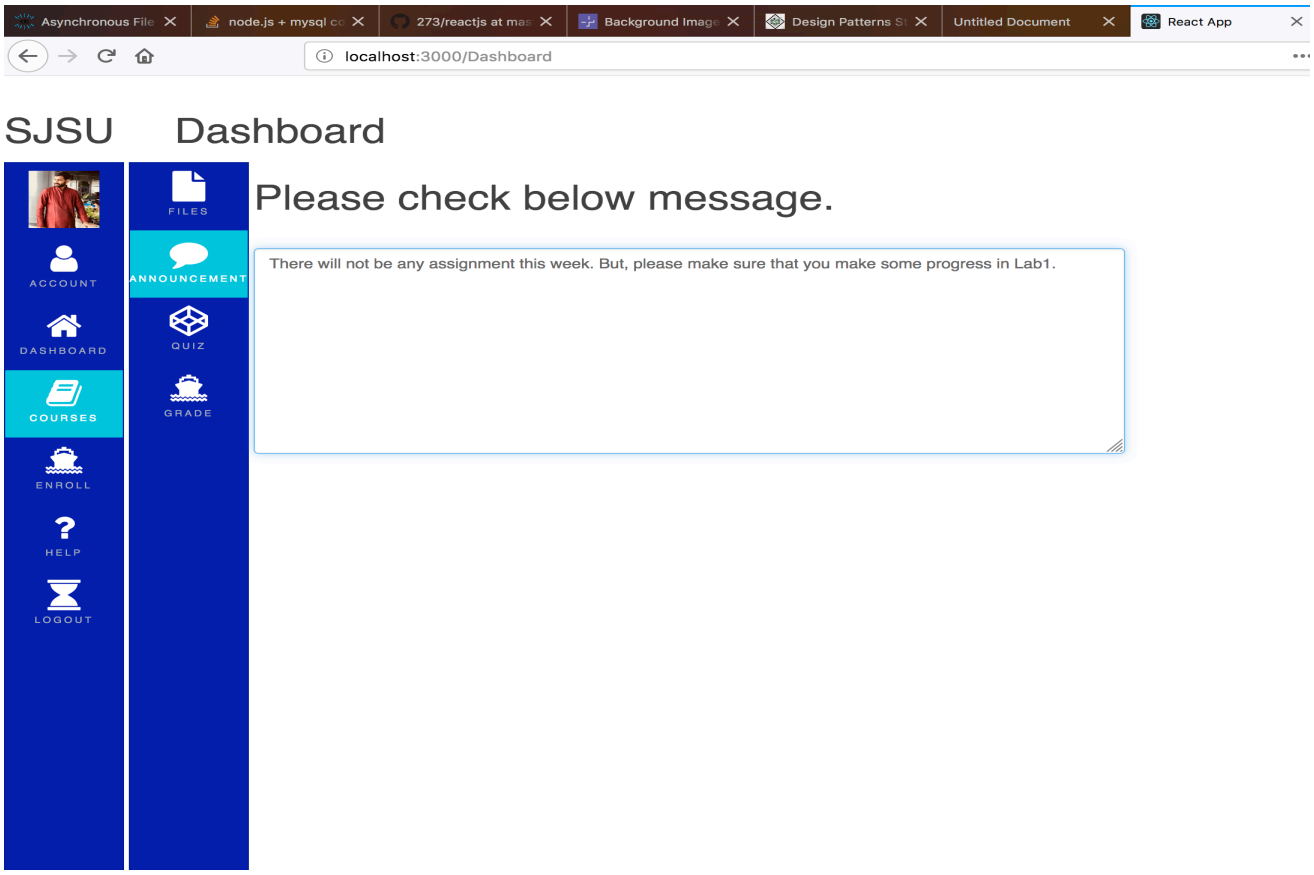


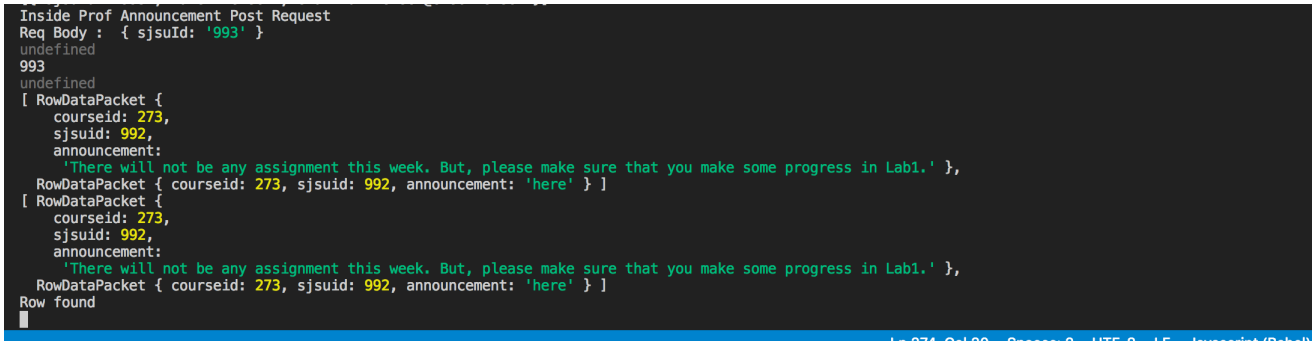
If User Clicks on Dashboard then it will be moved to same page



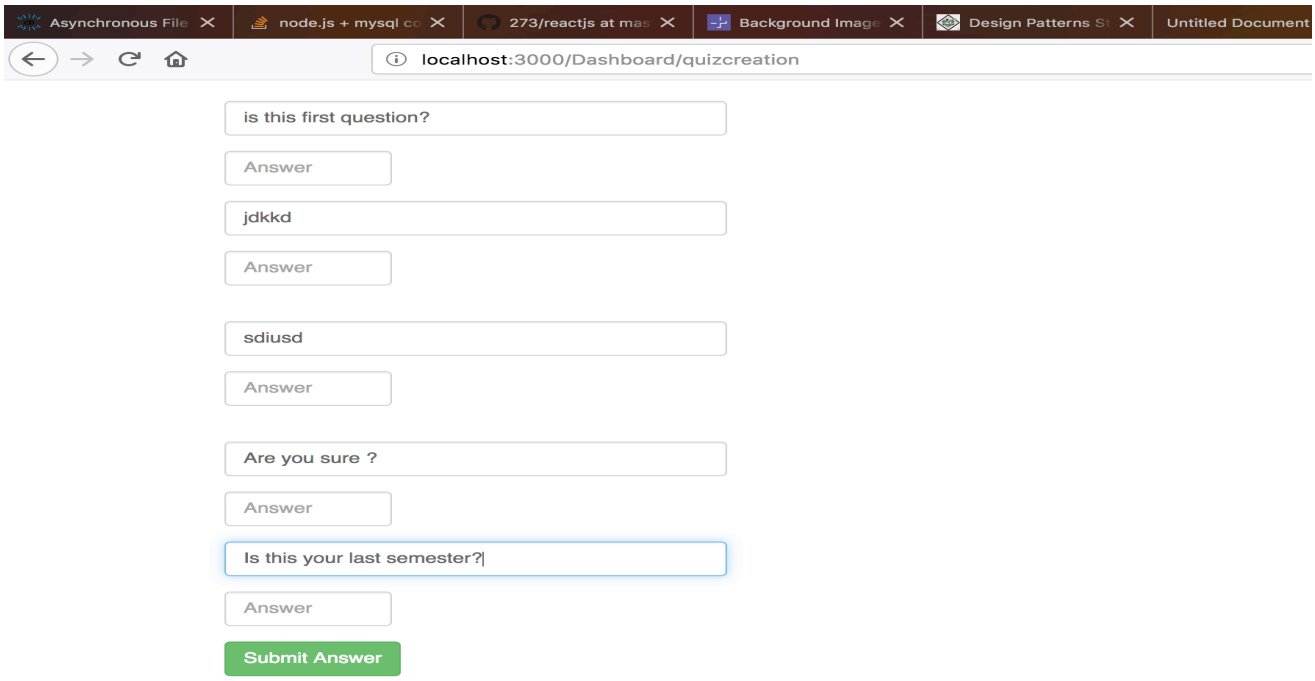
Once Student Clicks on Dashboard it will display the options Student wants to perform.



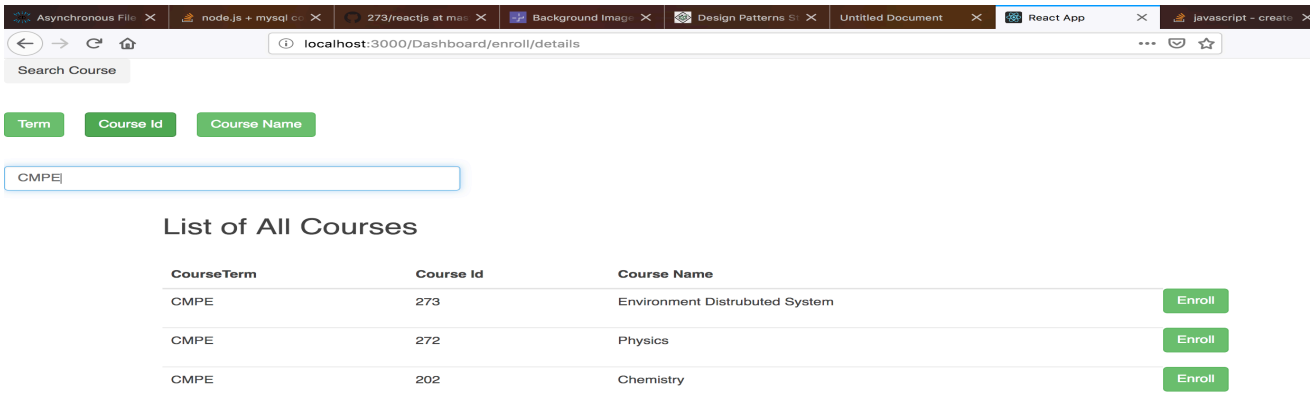




Once Student Clicks on Quiz then it will be diverted to new page and Student will be able to give the Quiz.



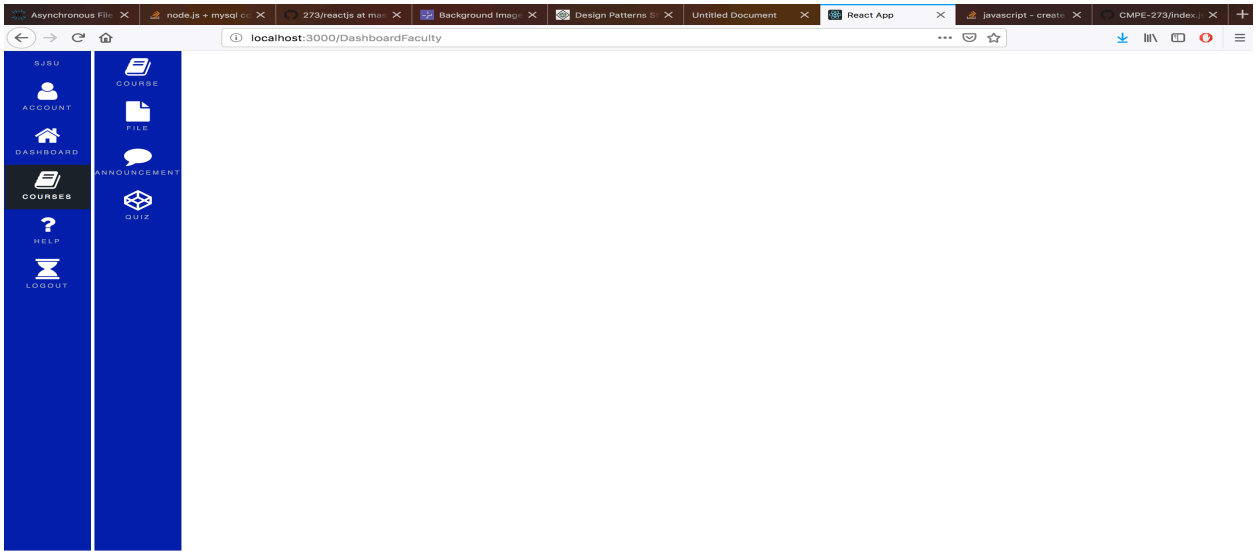
Student Enrollment process. Once Student clicks on Enroll then the respective subject gets added with SJSU Id of Student



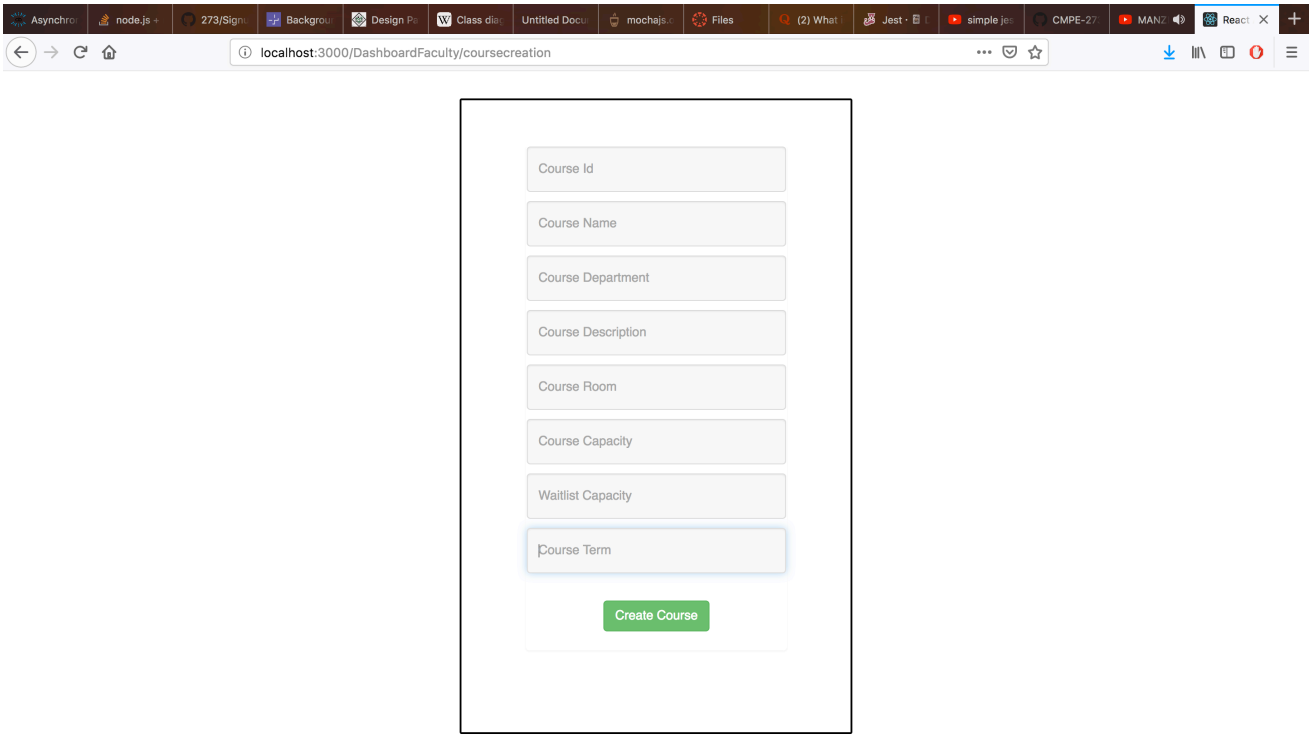
```
Inside Download File
Inside Enroll Login get Request
Req Body :  {}
undefined
Inside details Login Post Request
Req Body :  { courseInfo: 'CMPE' }
courses :  [ { courseid: '273',
    coursename: 'Environment Distrubuted System',
    coursedept: 'CMPE',
    coursedesc: 'Environment Distrubuted System',
    courseroom: '189',
    coursecapacity: 40,
    waitlistcapacity: 40,
    coursetemid: 'add' },
  { courseid: '272',
    coursename: 'Physics',
    coursedept: 'CMPE',
    coursedesc: 'Environment Distrubuted System',
    courseroom: '189',
    coursecapacity: 40,
    waitlistcapacity: 40,
    coursetemid: 'add' },
  { courseid: '202',
    coursename: 'Chemistry',
    coursedept: 'CMPE',
    coursedesc: 'Environment Distrubuted System',
    courseroom: '189',
    coursecapacity: 40,
    waitlistcapacity: 40,
    coursetemid: 'add' } ]
undefined
Courses :  [{"courseid":"273","coursename":"Environment Distrubuted System","coursedept":"CMPE","coursedesc":"Environment Distrubuted System","courseroom":"189",
"coursecapacity":40,"waitlistcapacity":40,"coursetemid":"add"},{"courseid":"272","coursename":"Physics","coursedept":"CMPE","coursedesc":"Environment Distrubuted
 System","courseroom":"189","coursecapacity":40,"waitlistcapacity":40,"coursetemid":"add"},{"courseid":"202","coursename":"Chemistry","coursedept":"CMPE","course
desc":"Environment Distrubuted System","courseroom":"189","coursecapacity":40,"waitlistcapacity":40,"coursetemid":"add"}]
```

Now, the pages and actions which was added for Faculty.

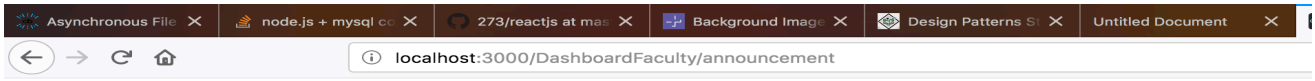
```
[{ finalstatus :true, facultyfnd :true, pwdvalidity :false}]
Inside Login Post Request
Req Body :  { sjsuId: '992', password: '992' }
992
val1sjsuid 992
val1pwd 992
ecrypted 5dd786
undefined
(node:64697) Warning: Use Cipheriv for counter mode of aes-256-ctr
res:  [ RowDataPacket { user_flag: 'Y', password: '5dd786' } ]
[ { finalstatus: true, facultyfnd: false, pwdvalidity: false } ]
results[0].user_flag Y
Valid Pwd
req.session.user = result[0]; true
[{"finalstatus":true,"facultyfnd":true,"pwdvalidity":true}]
```



Creation of Course Page. Faculty can create a Course with below details.



Below are the more options performed by the Faculty. (Announcement, Quiz, file upload)



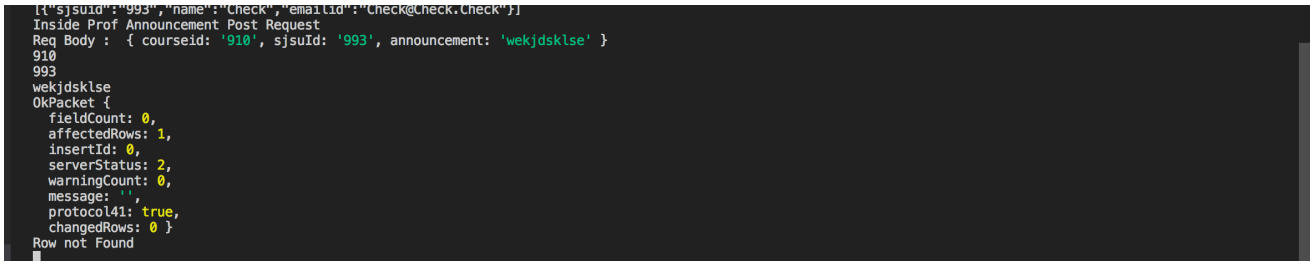
Add Announcement in Below Area :

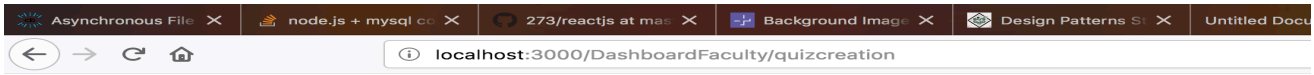
Enter Course Id

Your Announcement Here

Please fill out this field.

Announce





Question No 1

Answer

Answer

Answer

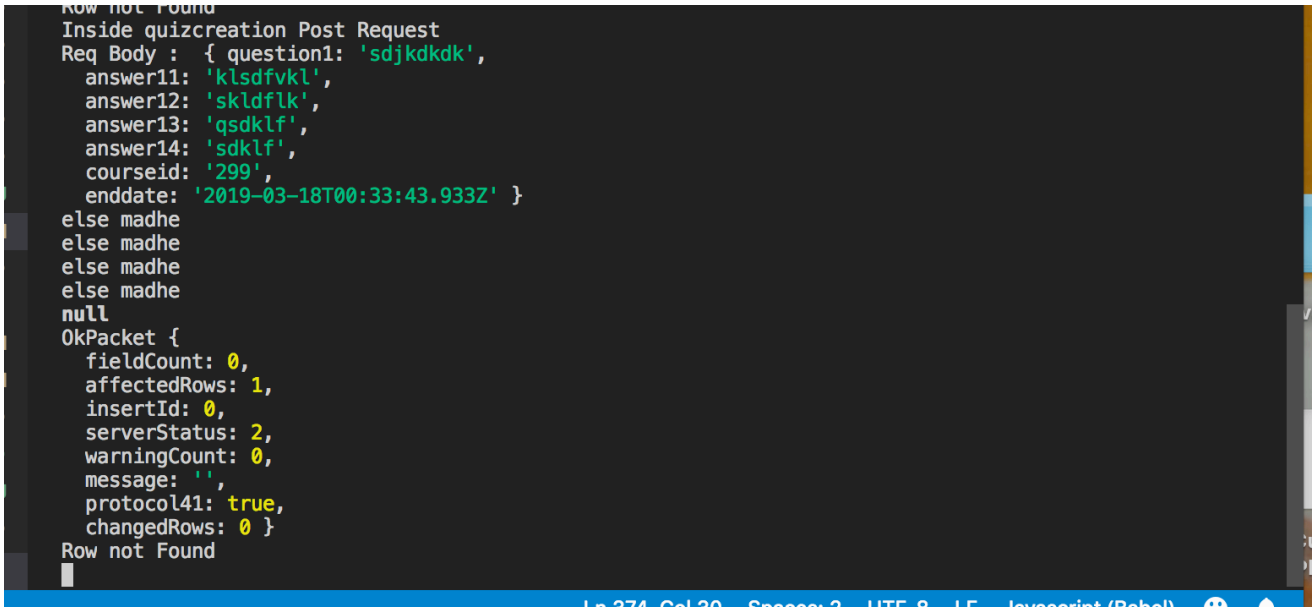
Answer

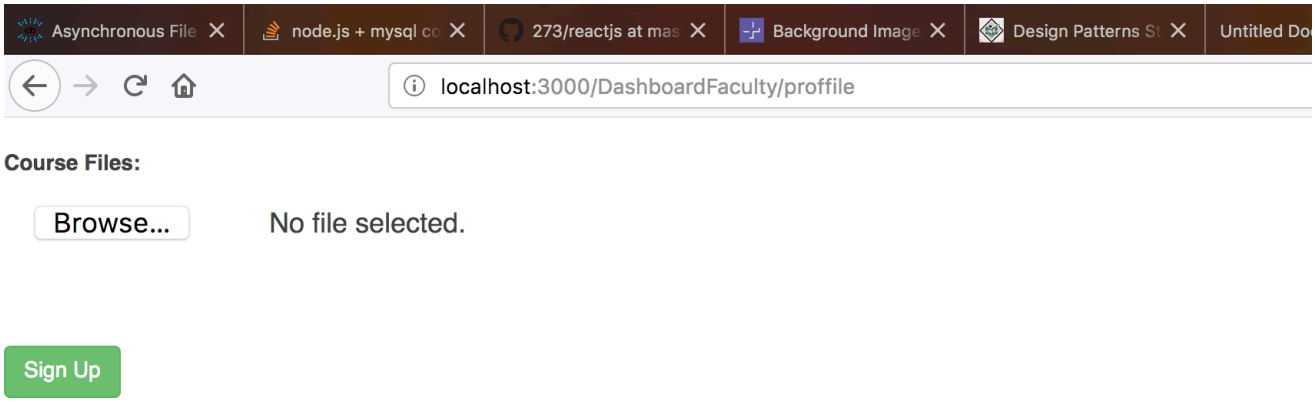
Course Id

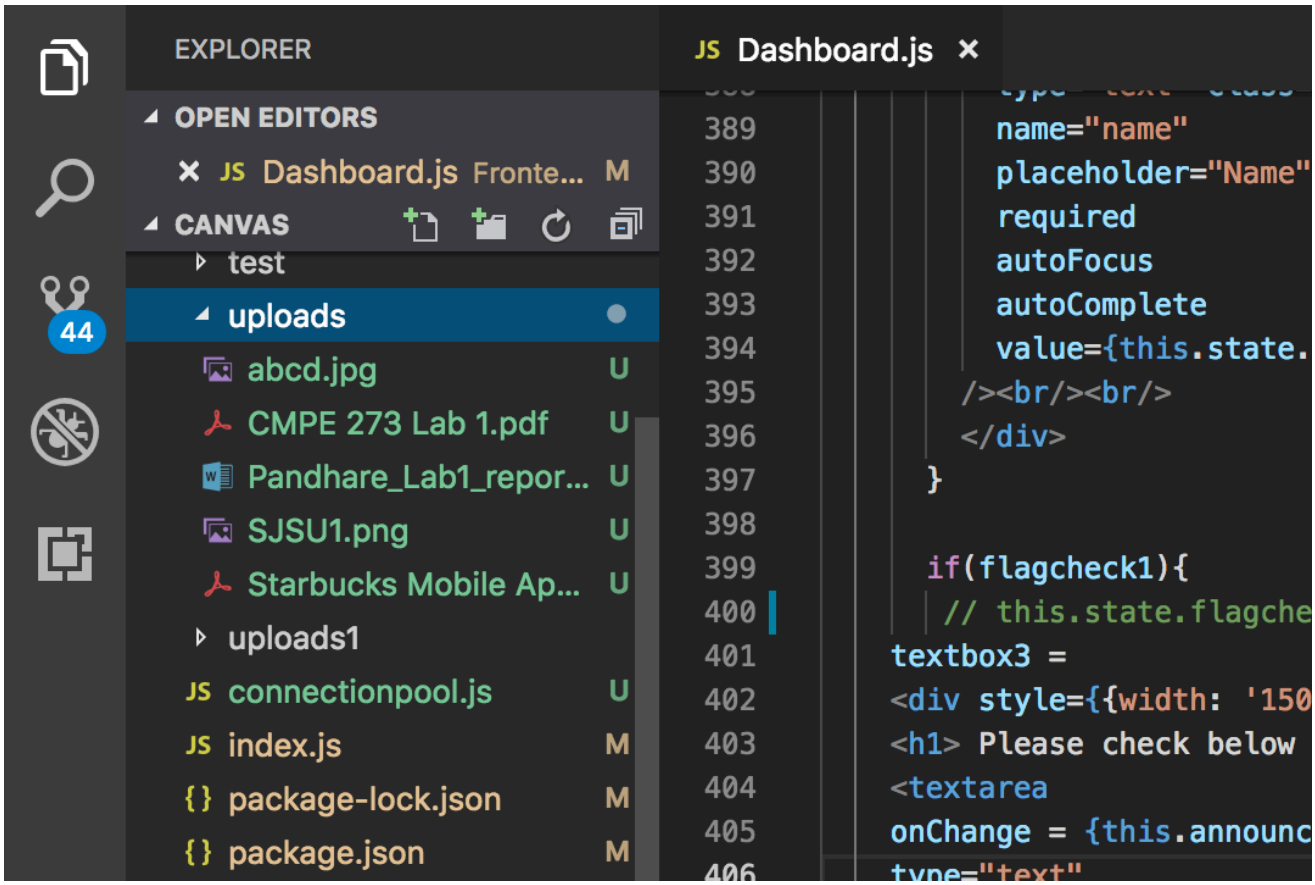
More Questions

Final Submit

2019-03-76


Row not Found
Inside quizcreation Post Request
Req Body : { question1: 'sdjkdkdk',
answer11: 'klsdfvkl',
answer12: 'skldflk',
answer13: 'qsdklf',
answer14: 'sdklf',
courseid: '299',
enddate: '2019-03-18T00:33:43.933Z' }
else madhe
else madhe
else madhe
else madhe
null
OkPacket {
fieldCount: 0,
affectedRows: 1,
insertId: 0,
serverStatus: 2,
warningCount: 0,
message: '',
protocol41: true,
changedRows: 0 }
Row not Found


Asynchronous File
node.js + mysql co
273/reactjs at mas
Background Image
Design Patterns St
Untitled Do
localhost:3000/DashboardFaculty/proffile
Course Files:
Browse...
No file selected.
Sign Up



Mocha Testing for Canvas: (Test Cases added on the GITHUB Repo)

```
Sachins-MacBook-Pro:~ sachinwaghmode$ mocha


  Mocha unit test
test done
    ✓ should hit dashboard (70ms)

  Mocha unit test
test done
    ✓ should display Signup

  Mocha unit test
test done
    ✓ should display profile details

  Mocha unit test
test done
    ✓ should display announcement

  Mocha unit test
test done
    ✓ should display quizcreation


  5 passing (115ms)

Sachins-MacBook-Pro:~ sachinwaghmode$
```

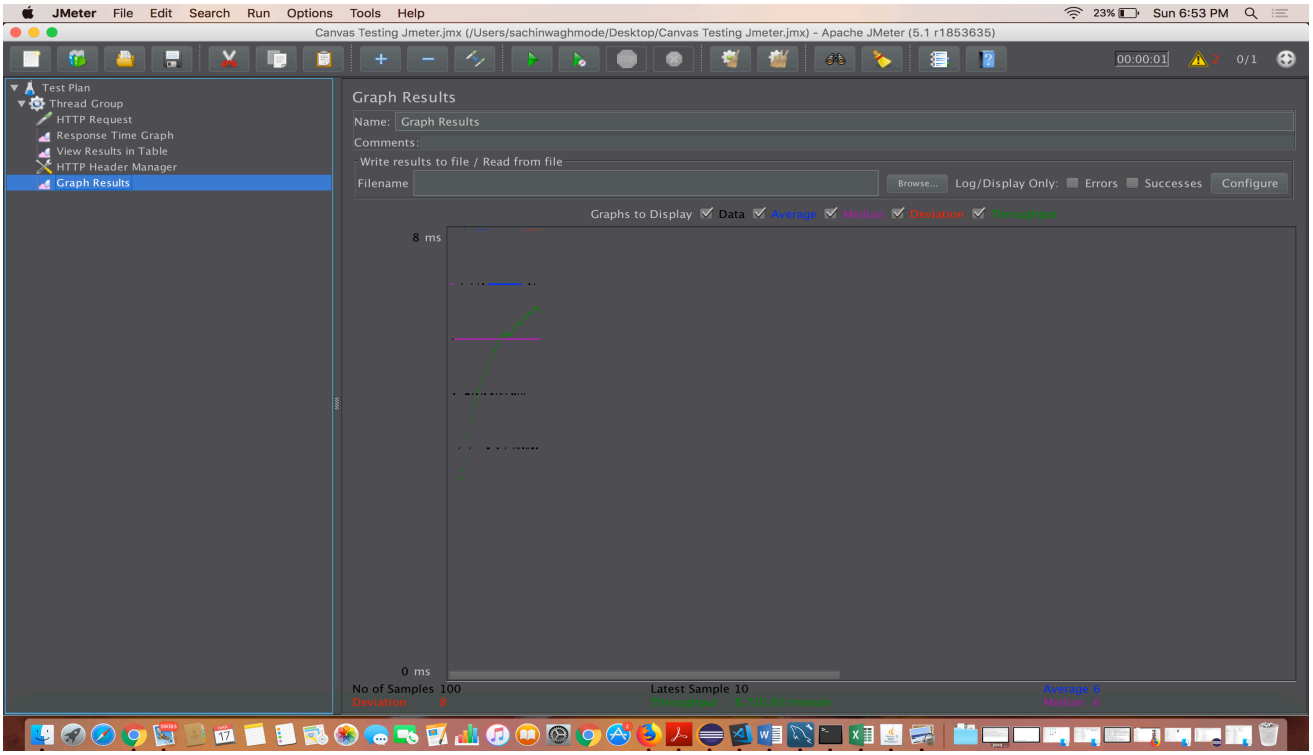
```
Sachins-MacBook-Pro:Canvas sachinwaghmode$ cd Backend-Code
Sachins-MacBook-Pro:Backend-Code sachinwaghmode$ node index.js
Server Listening on port 3000
Inside Login Post Request
Req Body :  { sjsuId: '992', password: '992' }
992
val1sjsuid 992
val1pwd 992
ecrypted 5dd786
undefined
(node:29039) [DEP0106] DeprecationWarning: crypto.createCipher is deprecated.
(node:29039) Warning: Use Cipheriv for counter mode of aes-256-ctr
res:  [ RowDataPacket { user_flag: 'Y', password: '5dd786' } ]
[ { finalstatus: true, facultyfnd: false, pwdvalidity: false } ]
results[0].user_flag Y
Valid Pwd
req.session.user = result[0]; true
[{"finalstatus":true,"facultyfnd":true,"pwdvalidity":true}]
Inside Login signup post Request
Req Body :  {}
```

```
Inside account profile Post Request
Req Body :  { sjsuId: '993' }
val1 NaN
results[object Object]
[ RowDataPacket { sjsuid: '993', name: 'Check', emailid: 'Check@Check.Check' } ]
Row found
[{"sjsuid":"993","name":"Check","emailid":"Check@Check.Check"}]
Inside Prof Announcement Post Request
Req Body :  { sjsuId: '993' }
undefined
993
undefined
[ RowDataPacket {
    courseid: 273,
    sjsuid: 992,
    announcement:
     'There will not be any assignment this week. But, please make sure that you make some progress in Lab1.' },
  RowDataPacket { courseid: 273, sjsuid: 992, announcement: 'here' } ]
[ RowDataPacket {
    courseid: 273,
    sjsuid: 992,
    announcement:
     'There will not be any assignment this week. But, please make sure that you make some progress in Lab1.' },
  RowDataPacket { courseid: 273, sjsuid: 992, announcement: 'here' } ]
Row found
Inside quiz student creation Post Request
Req Body :  {}
null
coursedatata undefined
han ni hay nakhara  undefined
null
[ RowDataPacket { question1: 'is this first question?' },
  RowDataPacket { question1: 'jdkkd' },
  RowDataPacket { question1: 'sdiusd' },
  RowDataPacket { question1: 'Are you sure ?' },
  RowDataPacket { question1: 'Is this your last semester?' } ]
Row found
```

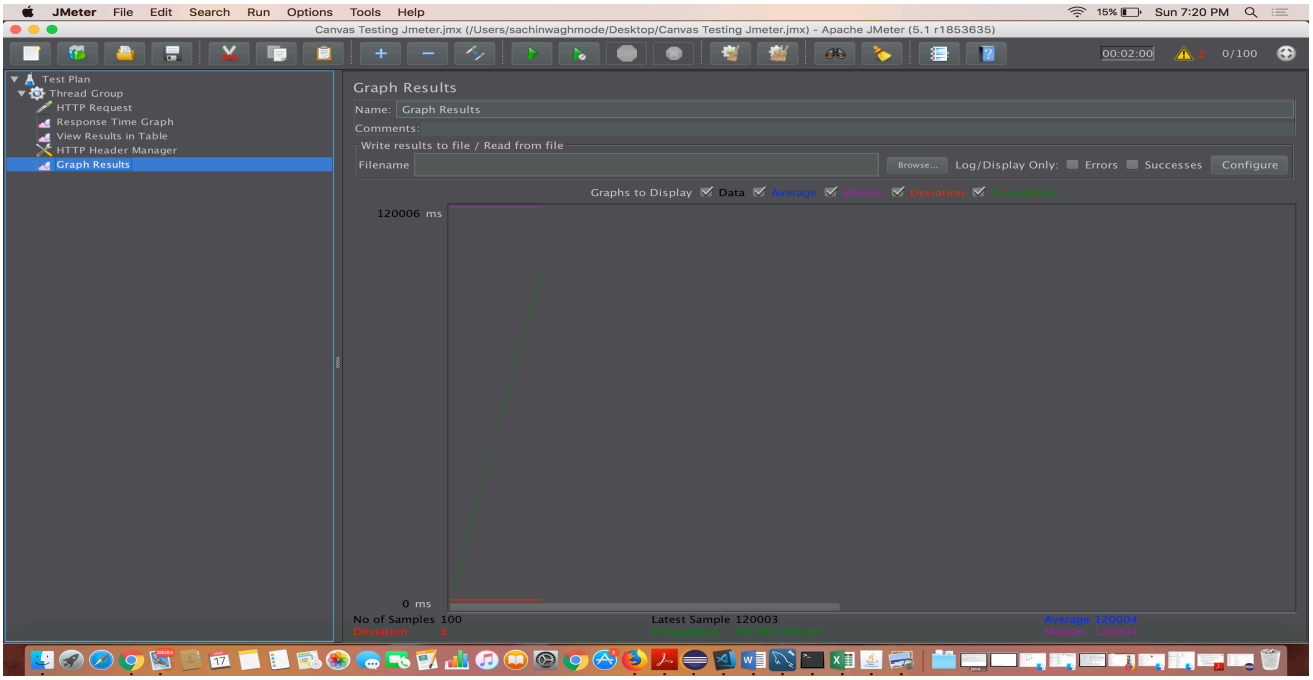
Jmeter Testing for Canvas → (with and without connection pooling)

I have tried with multiple concurrent user with and without connection pooling. I have noticed that, when I triggered multiple users with connection pooling is faster than without connection pooling.

100 Concurrent Users: Without connection pooling

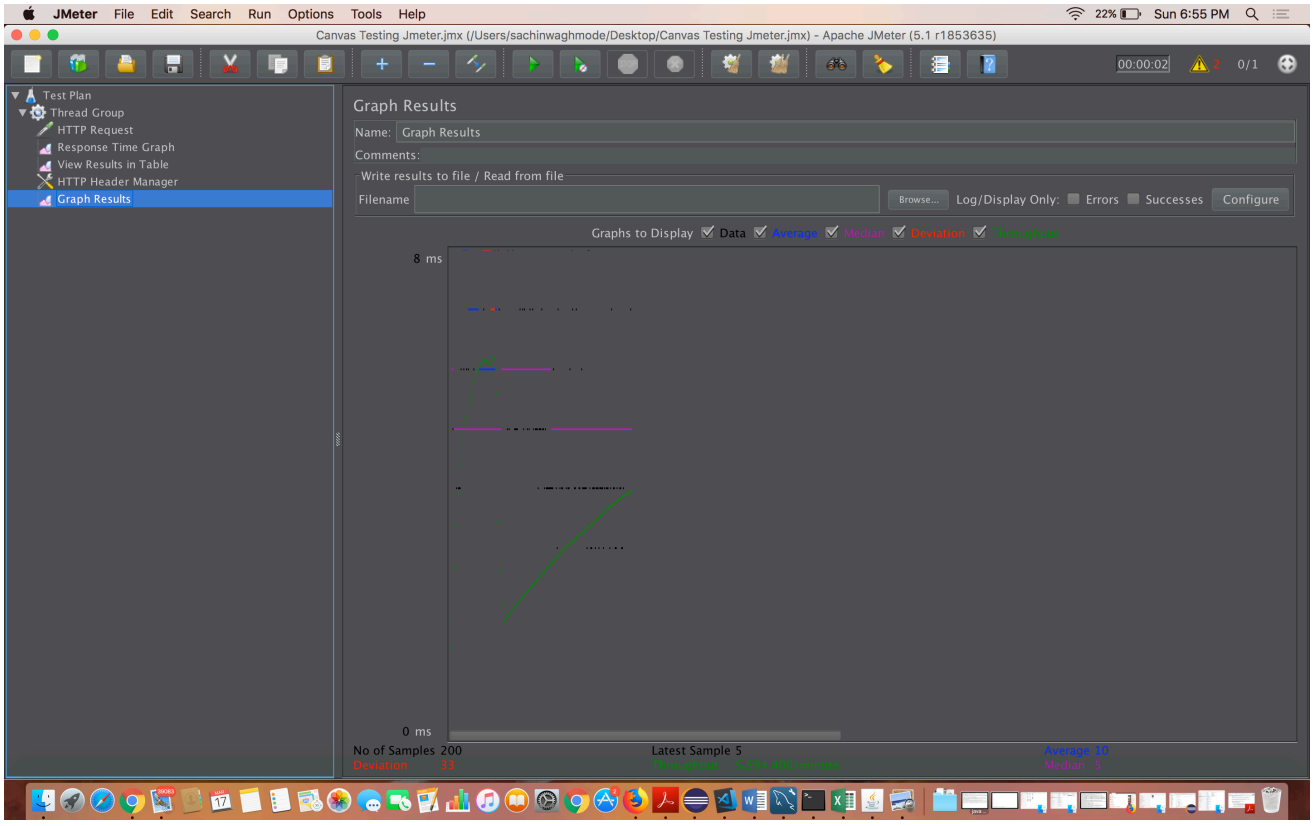


With Connection pooling:

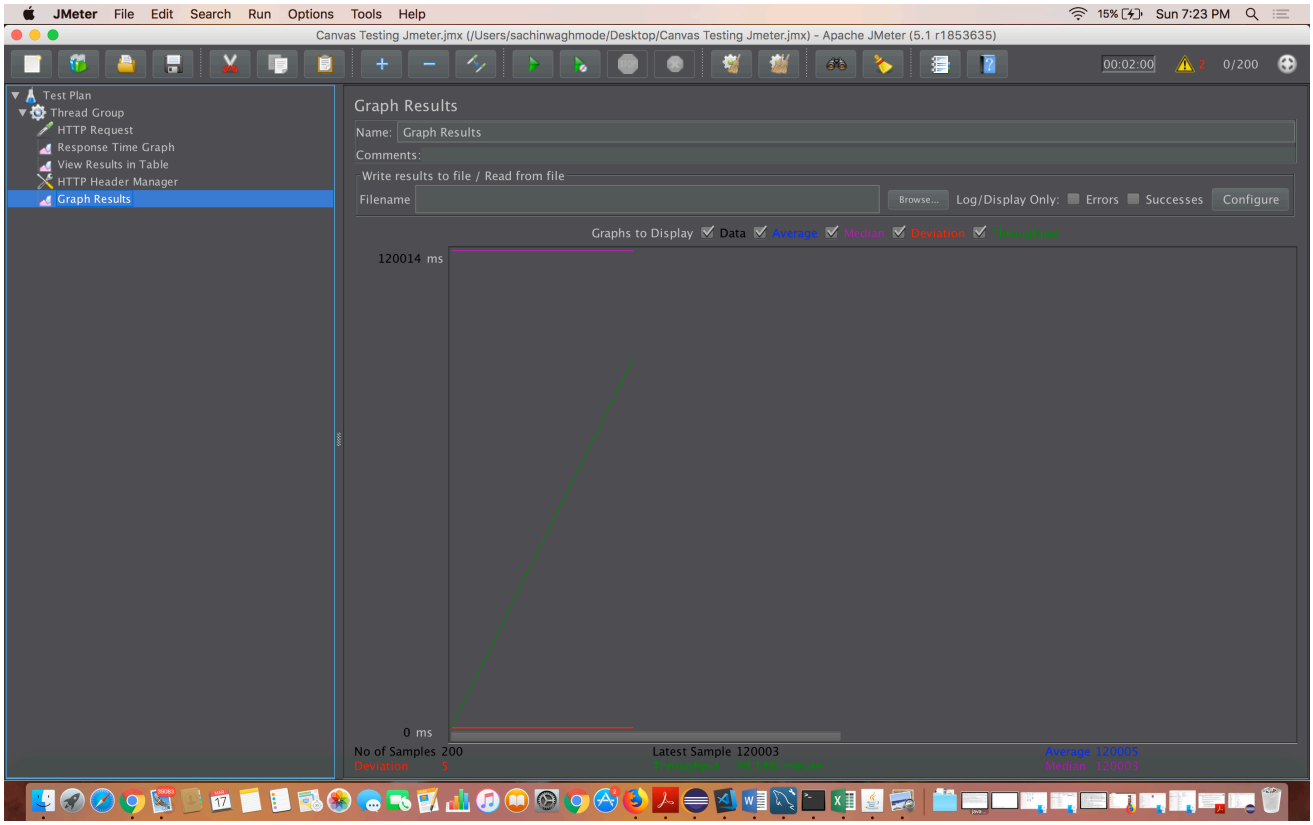


200 Concurrent Users.

Without Connection Pooling:

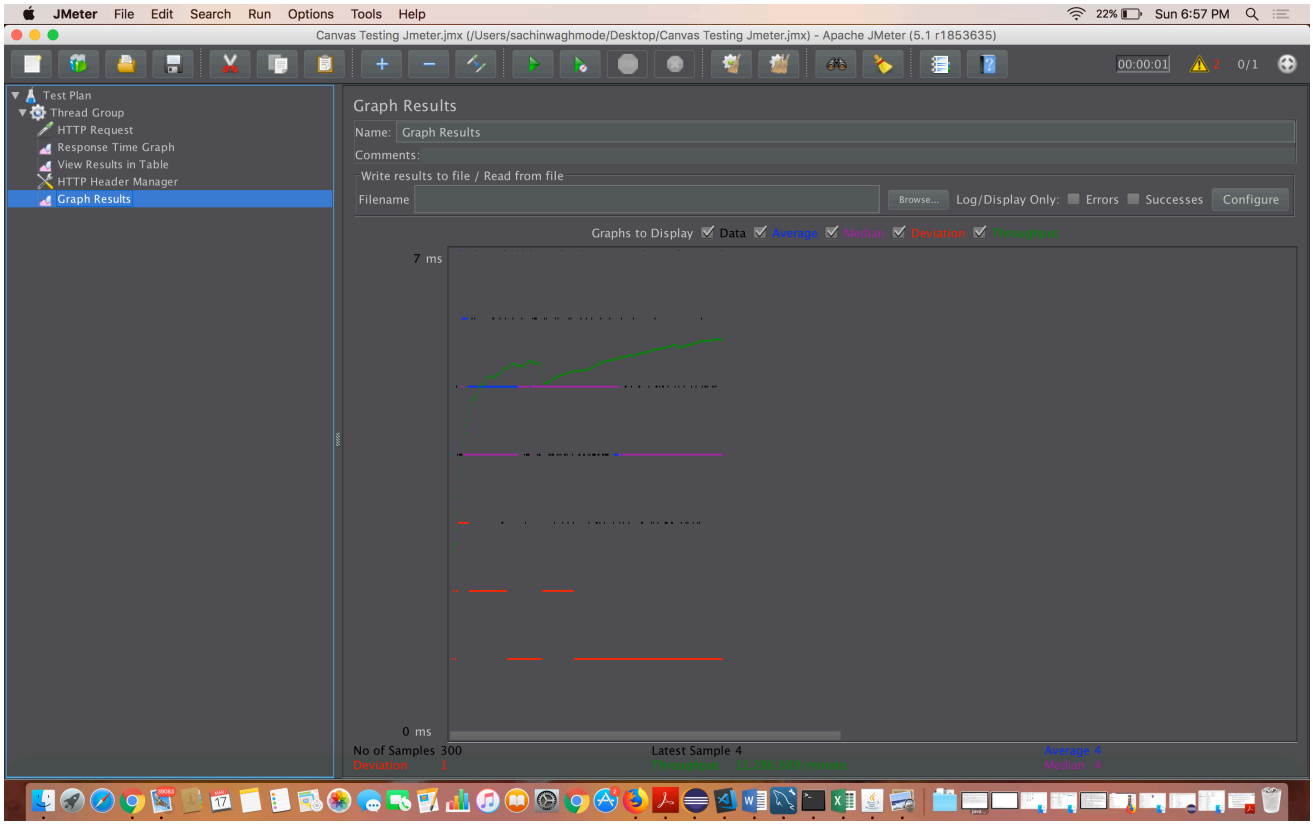


With Connection Pooling:

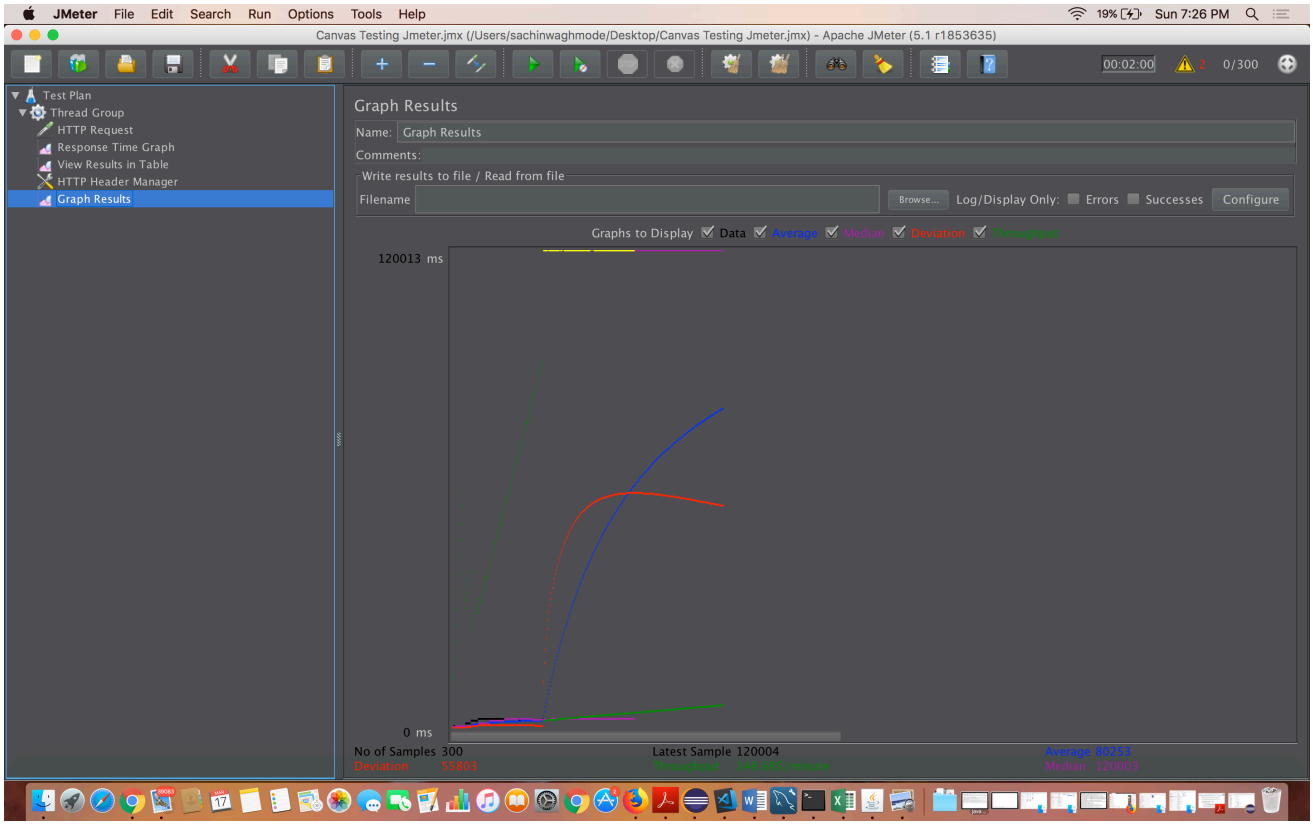


300 Concurrent Users:

Without Pooling

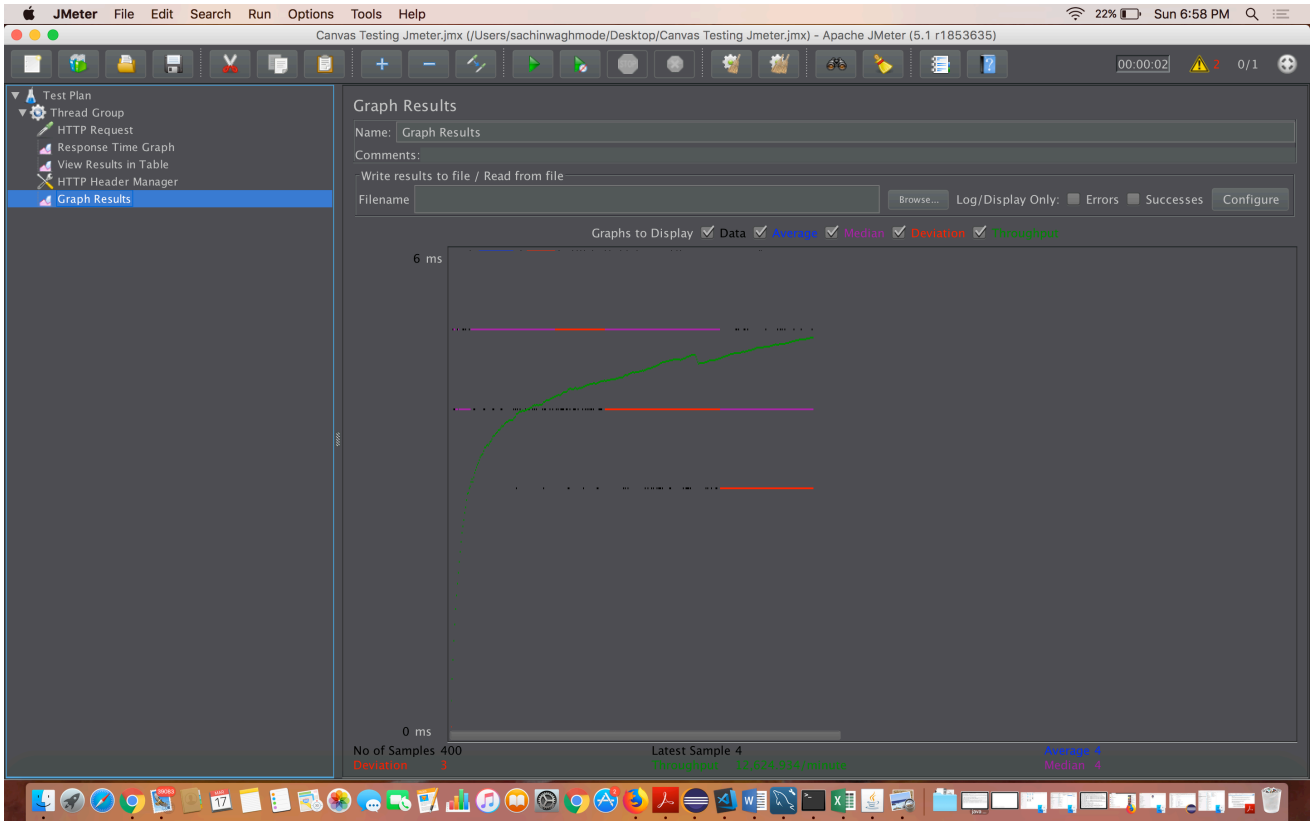


With Pooling

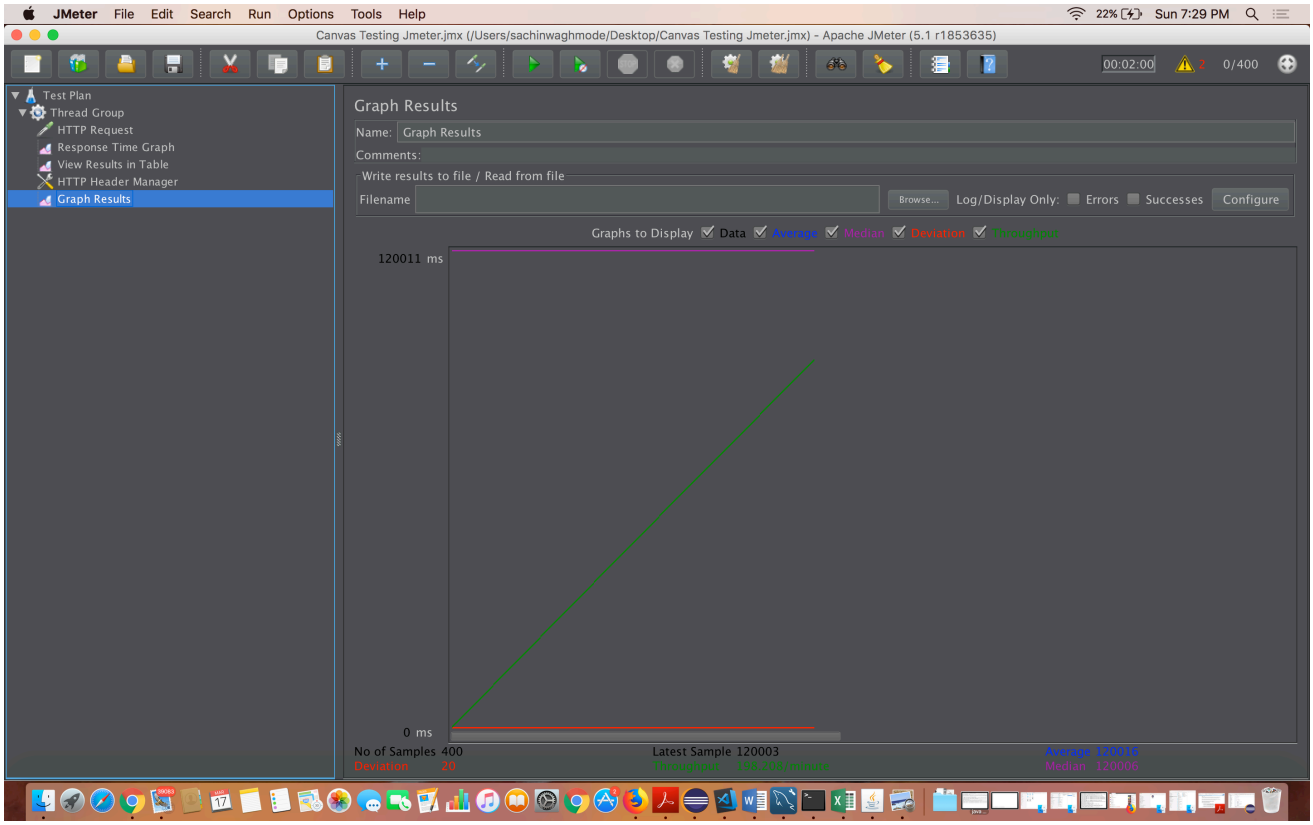


400 Concurrent Users:

Without Pooling

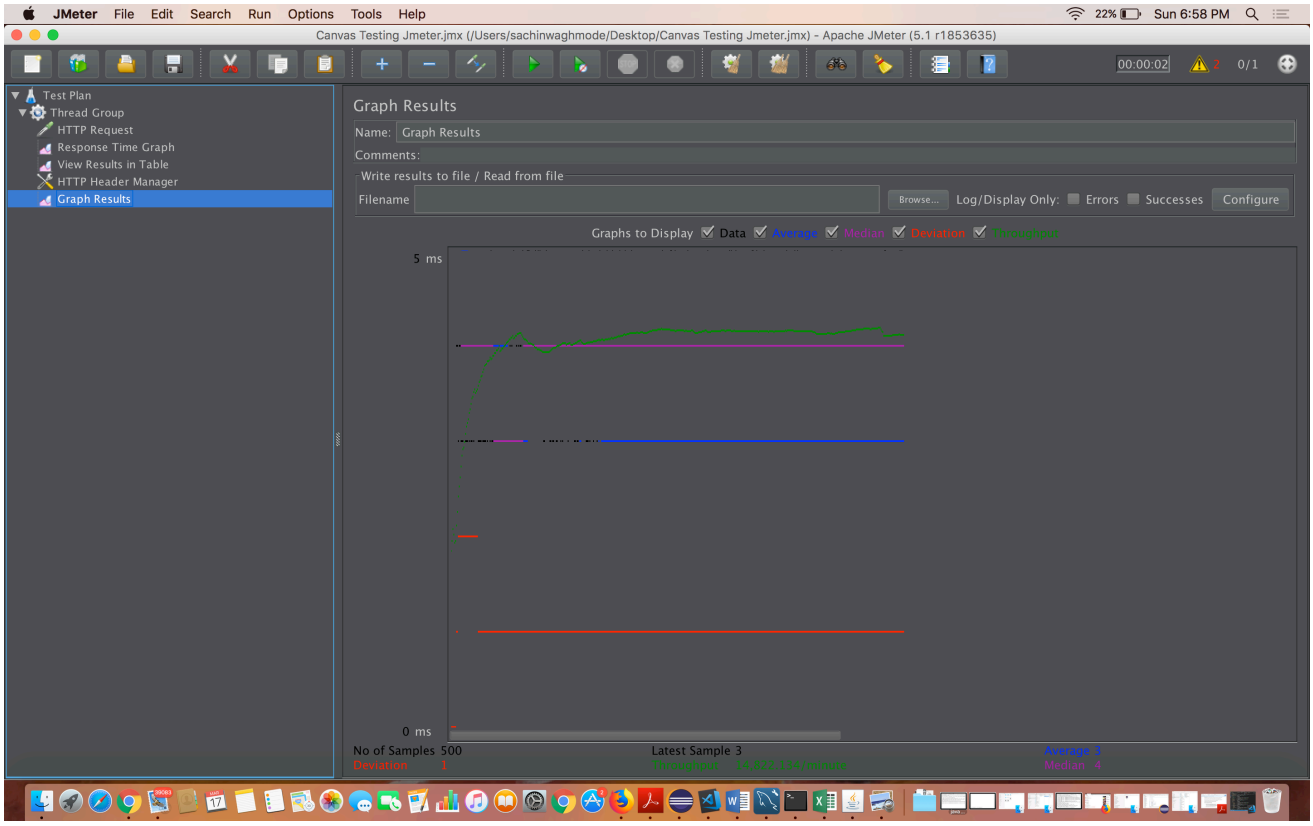


With Pooling:

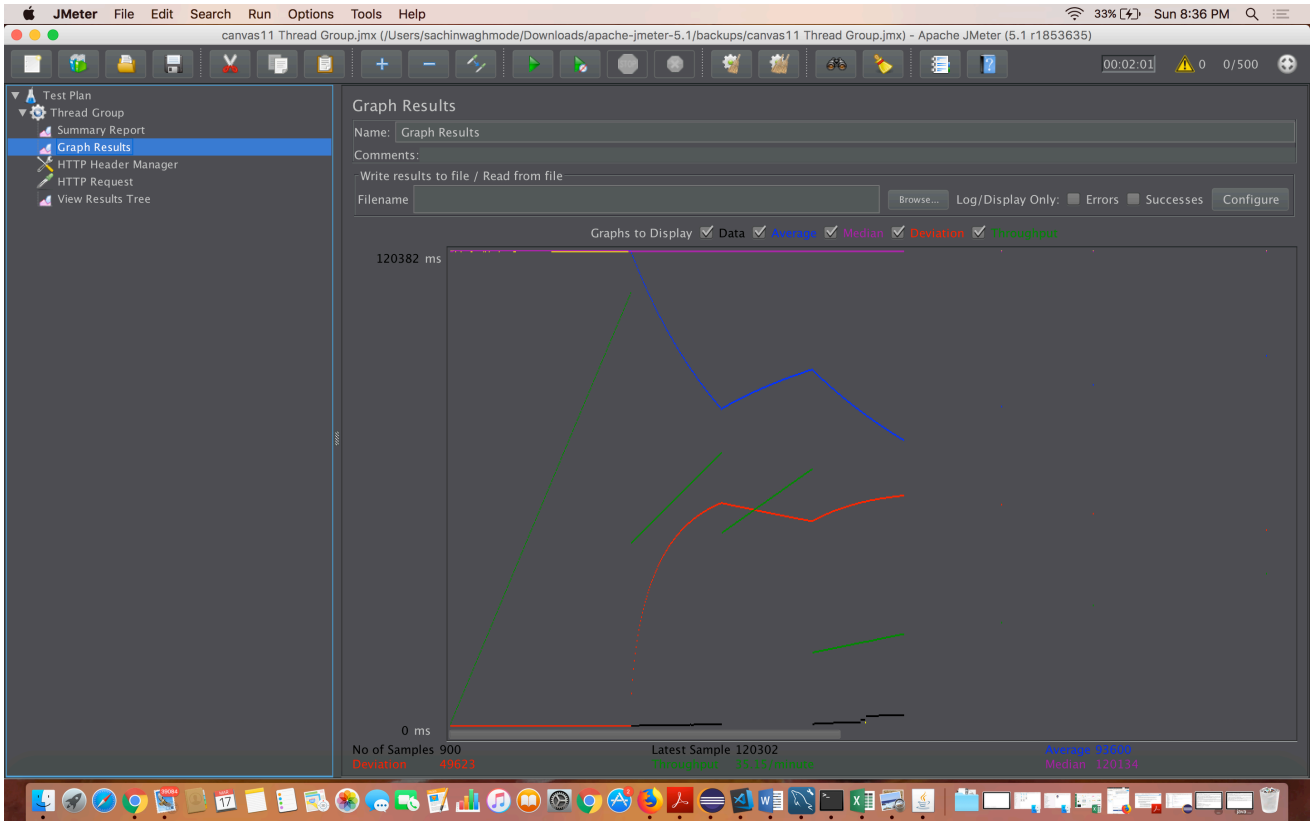


500 Concurrent Users:

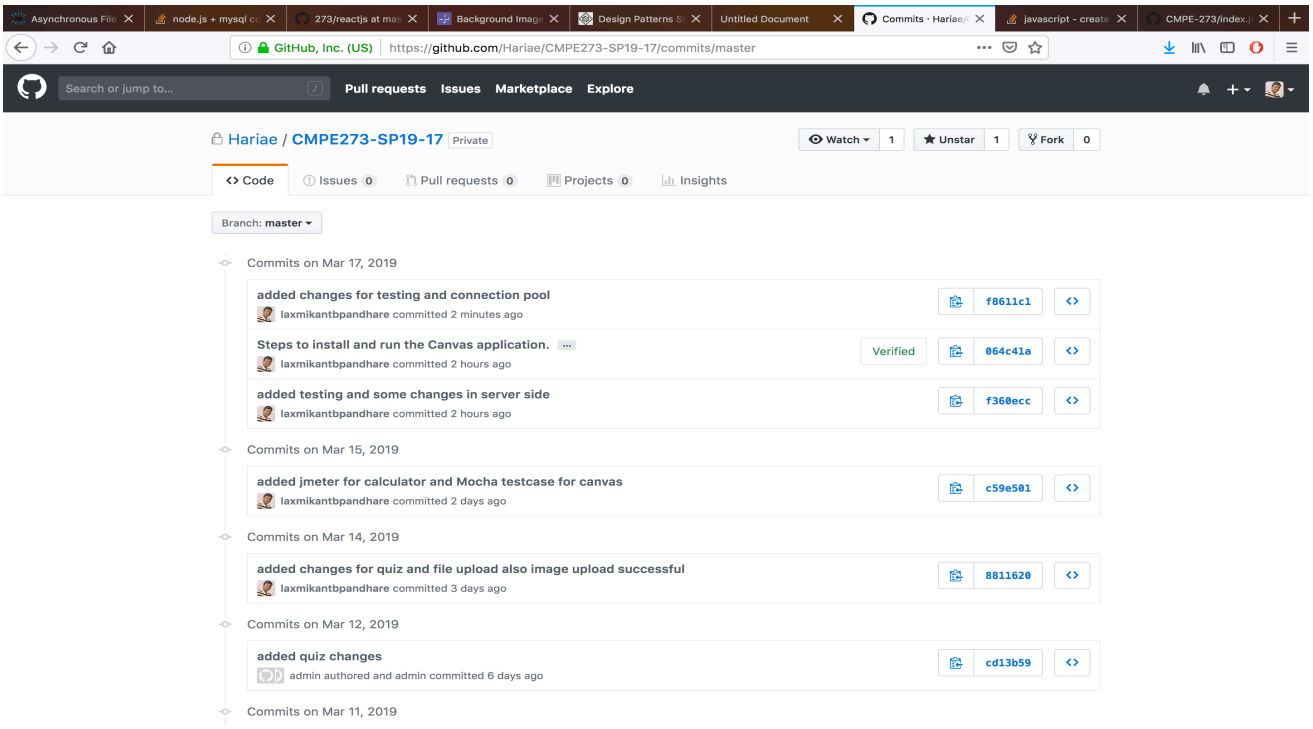
Without Pooling:

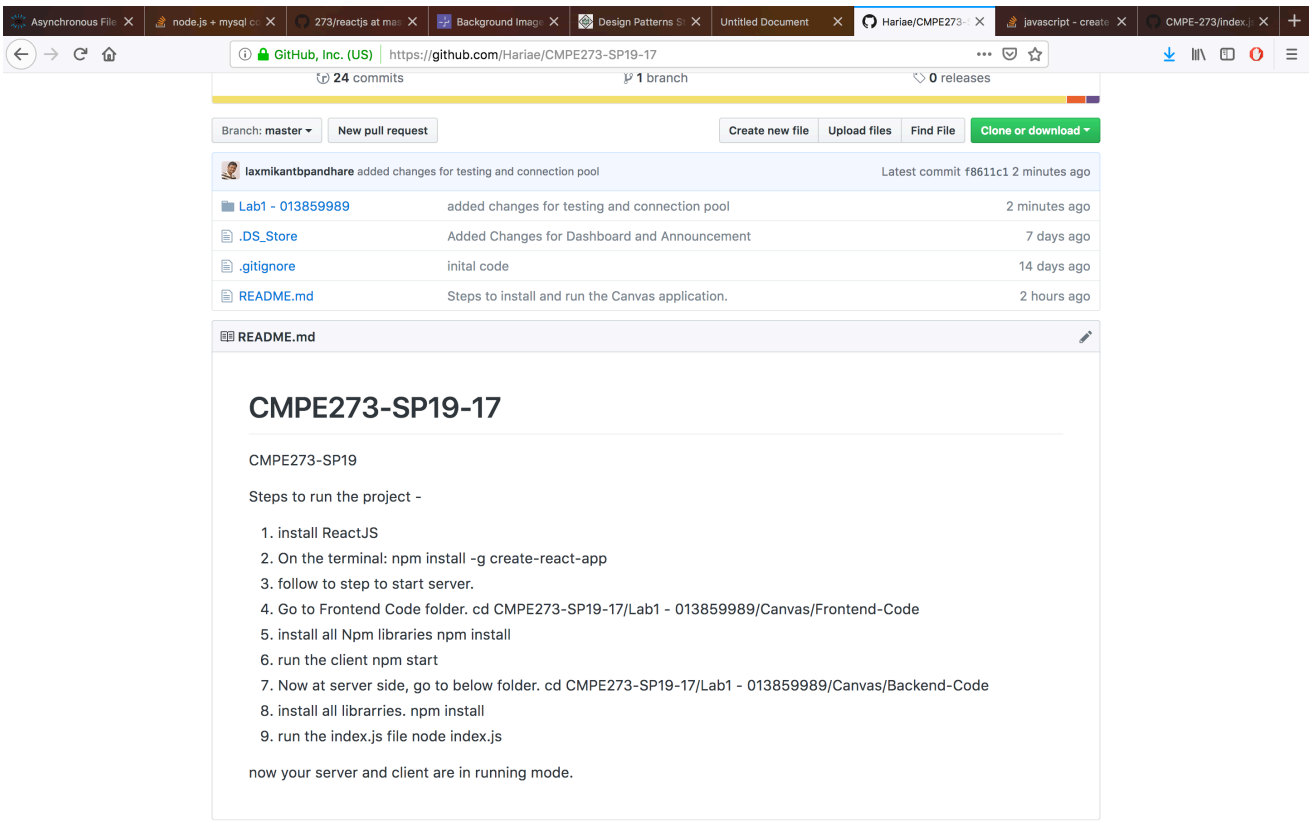


With Pooling:



GITHUB commits and steps added in README.md →





GitHub link → **https://github.com/Hariae/CMPE273-SP19-17**