# DROPBOX Lab 3

**(GitHub Repository : https://github.com/nsjethani/CMPE273/tree/master/Lab3)**

***Goals of the system:***

The main goal of this system is to develop a dropbox like application which provides users functionalities like file upload, download file, share files via email id, star mark important files using MySQL and ReactJS.

***Purpose of developing the system:***

The purpose behind making dropbox is to provide user an online storage system to maintain his important files on the cloud.

***System Design:***

The system design follows client-server architecture. Server functionalities are implemented in **SpringBoot**. SpringBoot server interacts with MySQL database and returns necessary data to front-end. To manage dependencies, Apache Maven is used.

**MySQL:**

MySQL includes solid data security layers that protect sensitive data from intruders. Rights can be set to allow some or all privileges to individuals. Passwords are encrypted.

MySQL can handle almost any amount of data, up to as much as 50 million rows or more.

**React JS:**

The front-end of the system is developed using ReactJS, HTML5, CSS and React-Bootstrap. The great benefit of using react is that we need the components are automatically rendered whenever a small change occur in the state but it does not re-render all the components. Only those components which are changes are re-rendered e.g. in our system, list of files is displayed via state so whenever any file is uploaded, only this part is re-rendered and rest of the page remains as is. Additional functionalities such of react such as material-ui provides a good UI pattern for react components.

**SpringBoot (Spring + Hibernate):**

* Spring Boot is a Framework from “The Spring Team” to ease the bootstrapping and development of new Spring Applications. It provides defaults for code and annotation configuration to quick start new Spring projects within no time. It completely avoids XML configuration. It also provides lots of plugins to work with embedded and in-memory Databases very easily.

So, Springboot is combination of Spring Framework + Embedded HTTP servers - XML beans. It provides Opinionated Development approach.

**Other Important Design features:**

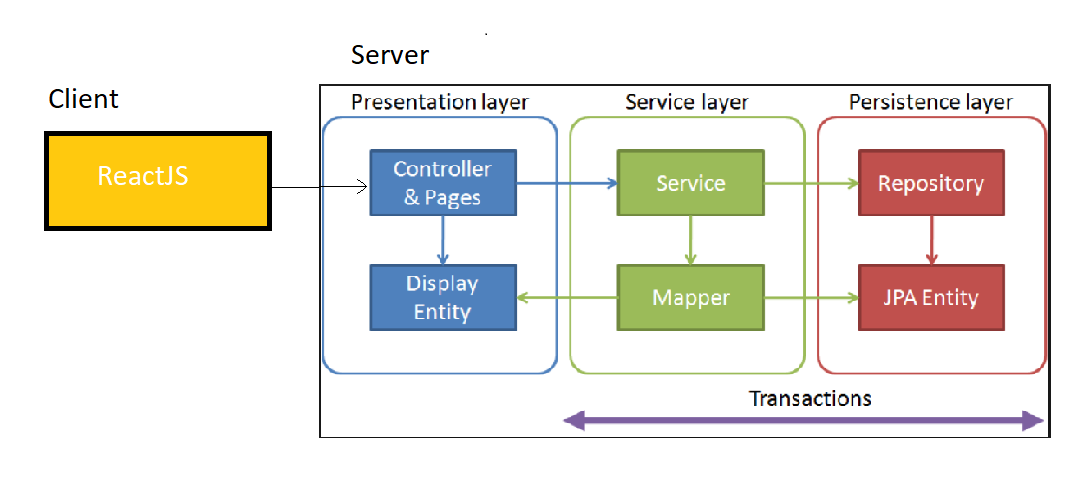
* To connect SpringBoot and React, CORs functionality is used so that clients and servers which are running on different ports are connected and parameters can be asked from React side to server side.
* SpringBoot in-built session functionality is used.
* SHA-1 : Also, SHA-1 encryption algorithm is used for storing passwords in hashed form in the database which provides additional security mechanism and prevents the system from attacks.

Code of SHA-1 in Spring:

MessageDigest sha = MessageDigest.*getInstance*(**"SHA-1"**);  
**byte**[] hashedBytes = sha.digest(input.getBytes());  
**char**[] digits = { **'0'**, **'1'**, **'2'**, **'3'**, **'4'**, **'5'**, **'6'**, **'7'**, **'8'**, **'9'**,  
 **'a'**, **'b'**, **'c'**, **'d'**, **'e'**, **'f'** };  
**for** (**int** idx = 0; idx < hashedBytes.**length**;idx++) {  
 **byte** b = hashedBytes[idx];  
 hash.append(digits[(b & 0xf0) >> 4]);  
 hash.append(digits[b & 0x0f]);  
}

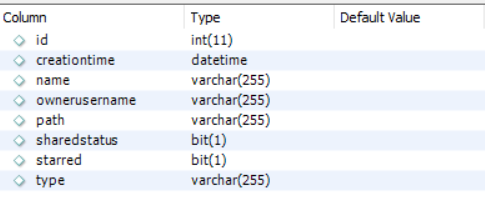
**System Architecture Diagram:**

* Spring is used to implement the controller layer which accepts request from .
* Service layer is implemented to write business logics and communicate with database layers.
* Hibernate is used to make repository layer and Entity layers.
* Repository layer is communicating with MySQLl Database using JPA.

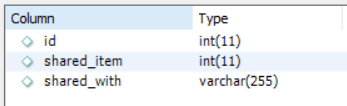


**Database Design:**

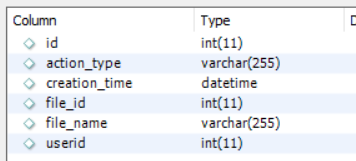
1. File\_details : to store file/folder details



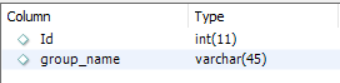
1. Share\_data : to store which file is shared with which user



1. File\_operation\_logs : to store user activity such as file upload,delete,share

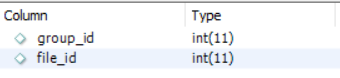


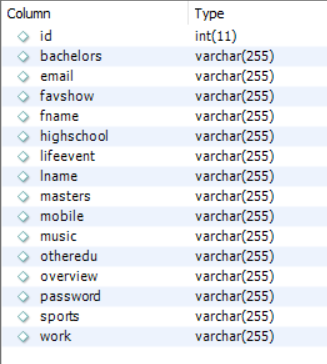
1. Grouping : to store group details



1. Group\_members : to store member details of group



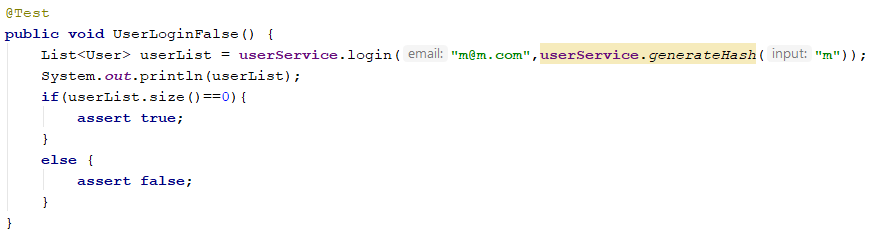
1. Group\_data : to store which file is shared in
2. User : to store user details and profile



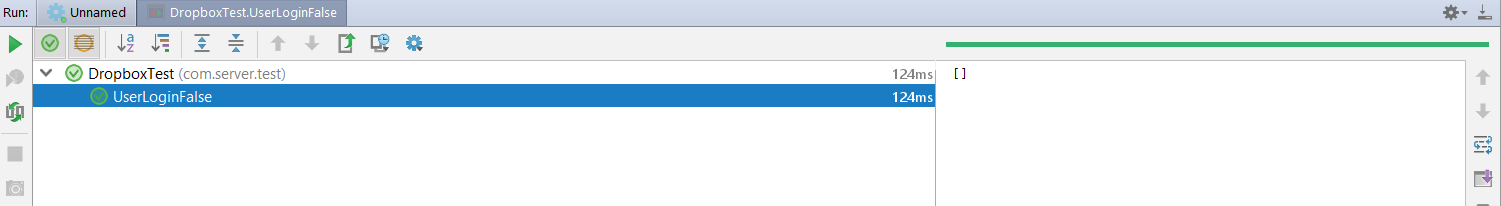
**Junit Tests:**

1. Invalid username and password doesn’t return any user data.

Test Case:



Test Result :

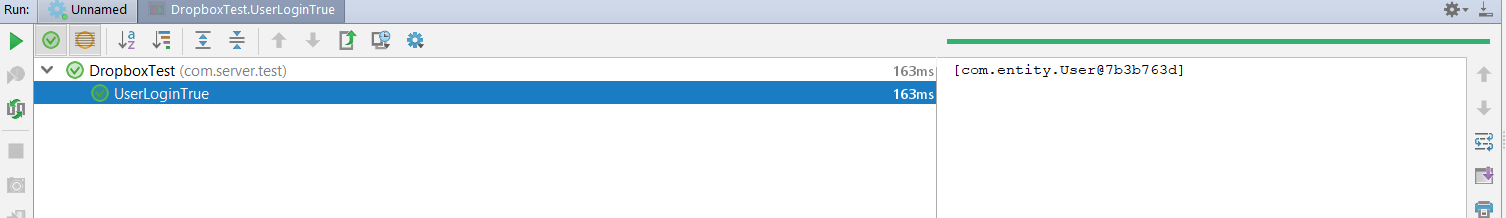


1. Valid username and password returns user object.

Test case:

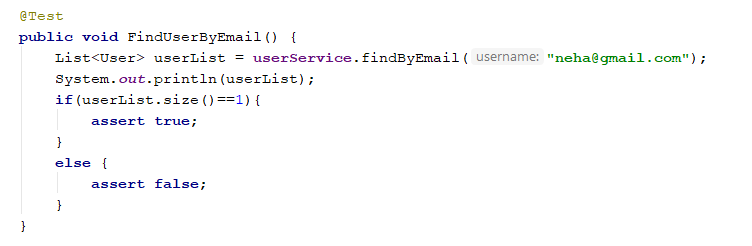


Test Result:

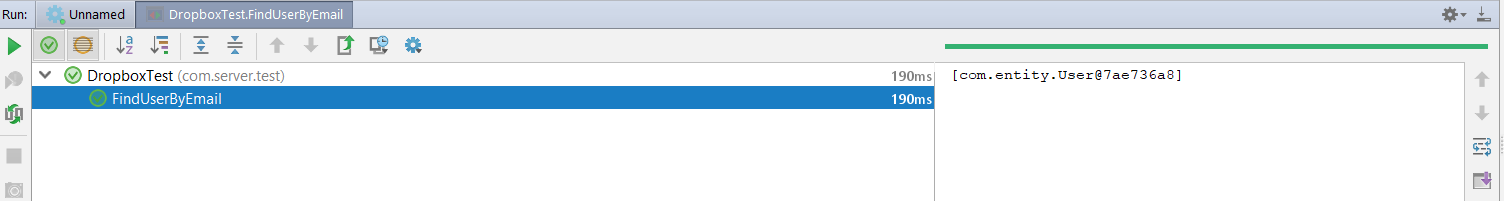


1. Find user by given email id :

Test Case:

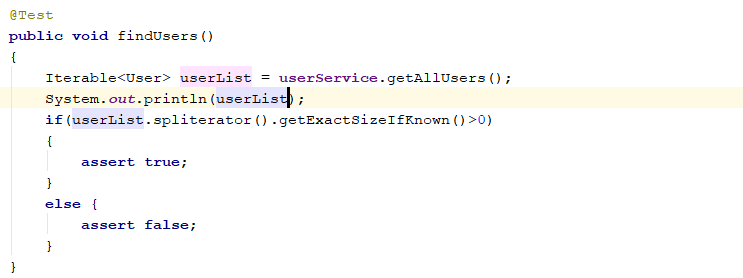


Test Result:

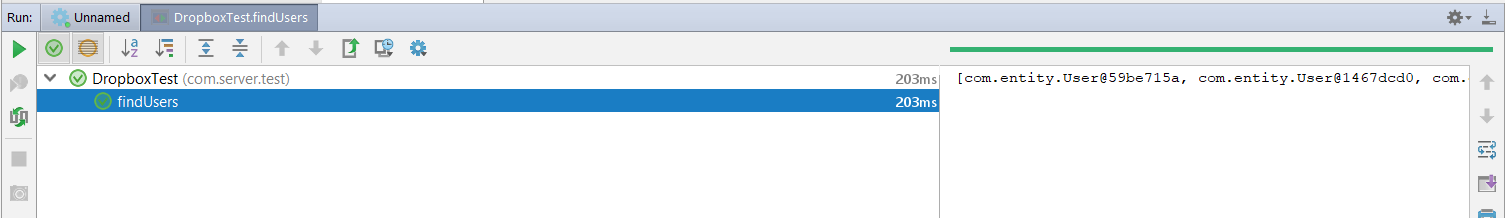


1. Find list of users:

Test Case:

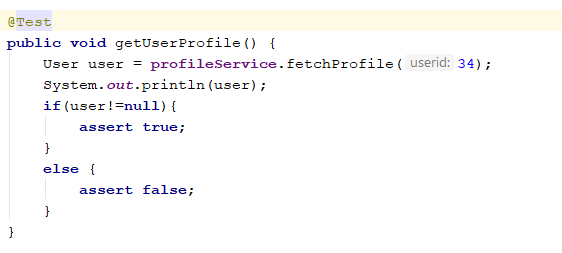


Test Result:

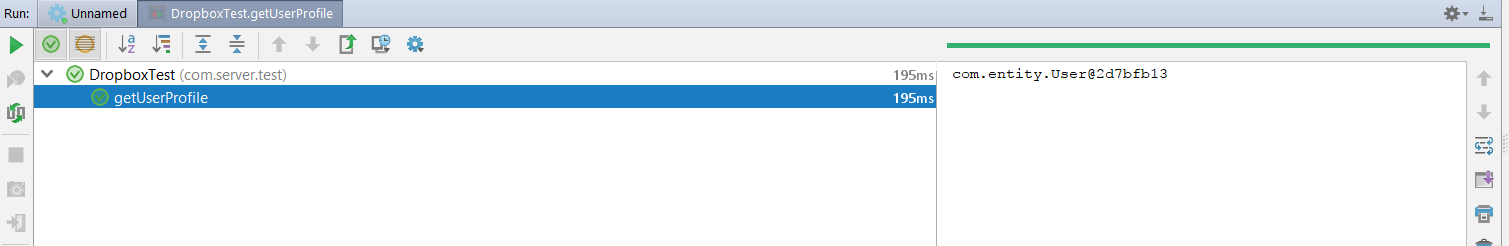


1. To fetch user profile from given useris.

Test Case:

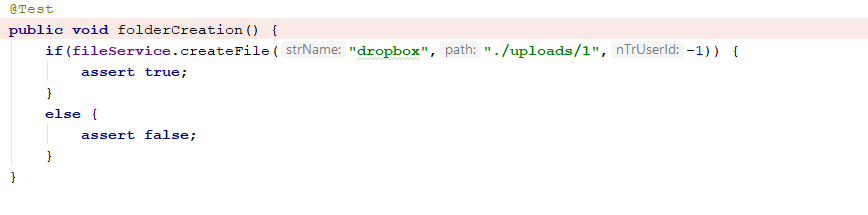


Test Result:

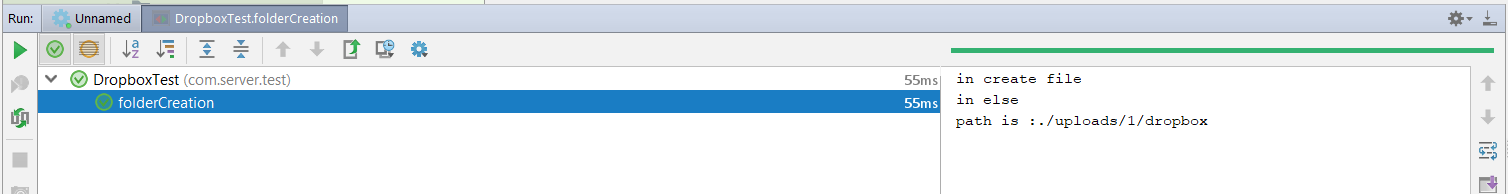


1. To create folder on given path:

Test Case:

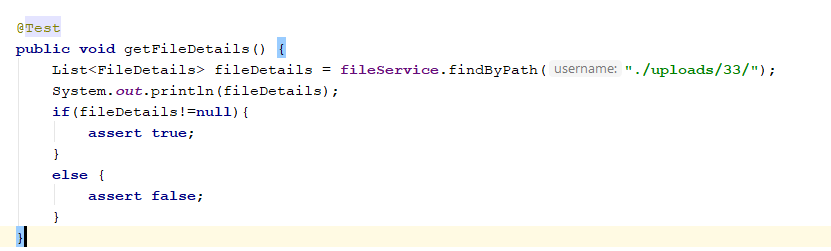


Test Result:

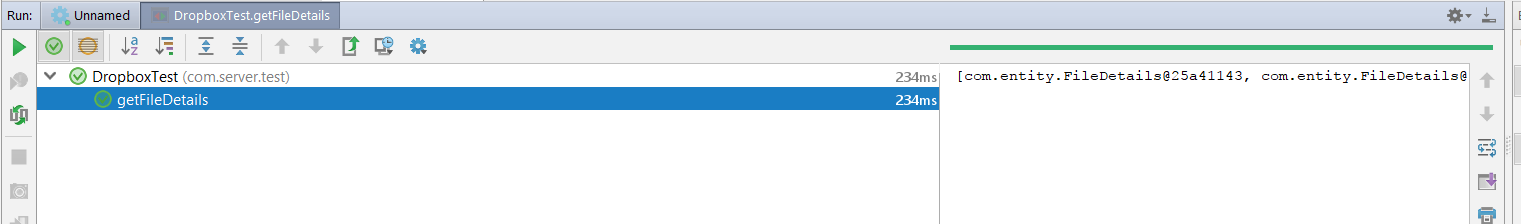


1. To get file details from given path

Test Case:

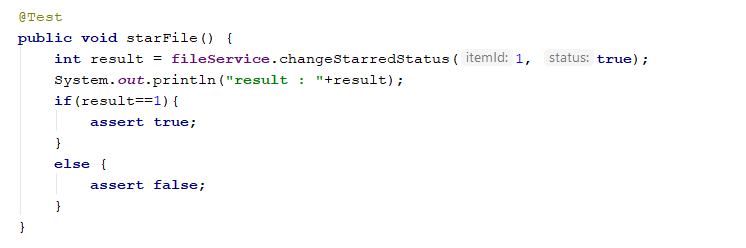


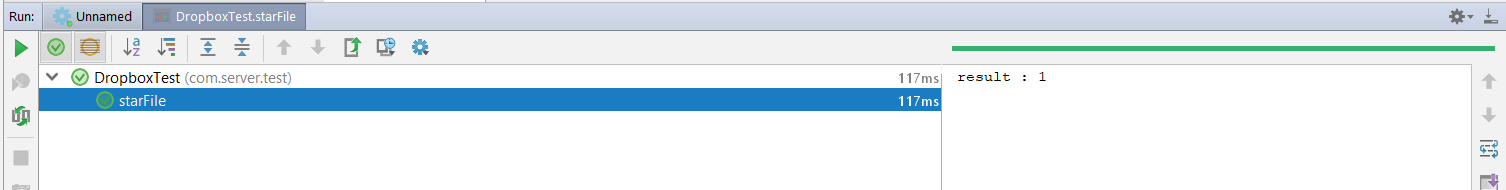
Test Result:



1. To star mark given file id

Test Case :

Test Result:

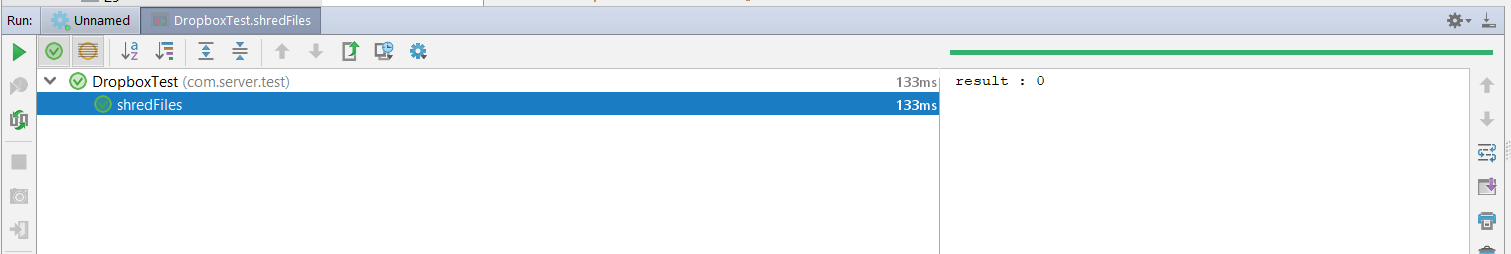


1. To get shared files with particular user:

Test Case:

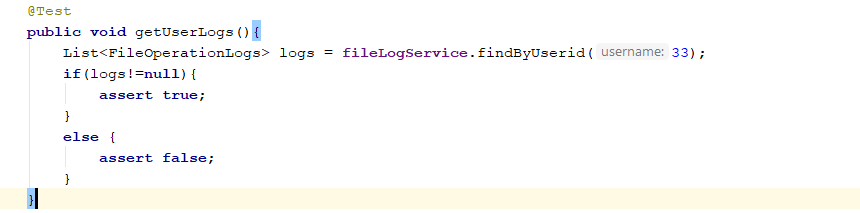


Test Result:

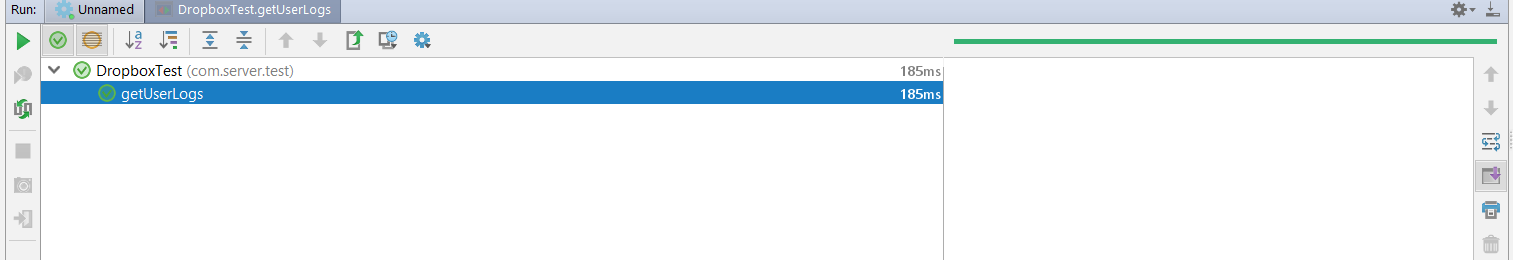


1. Get User activity logs:

Test Case :

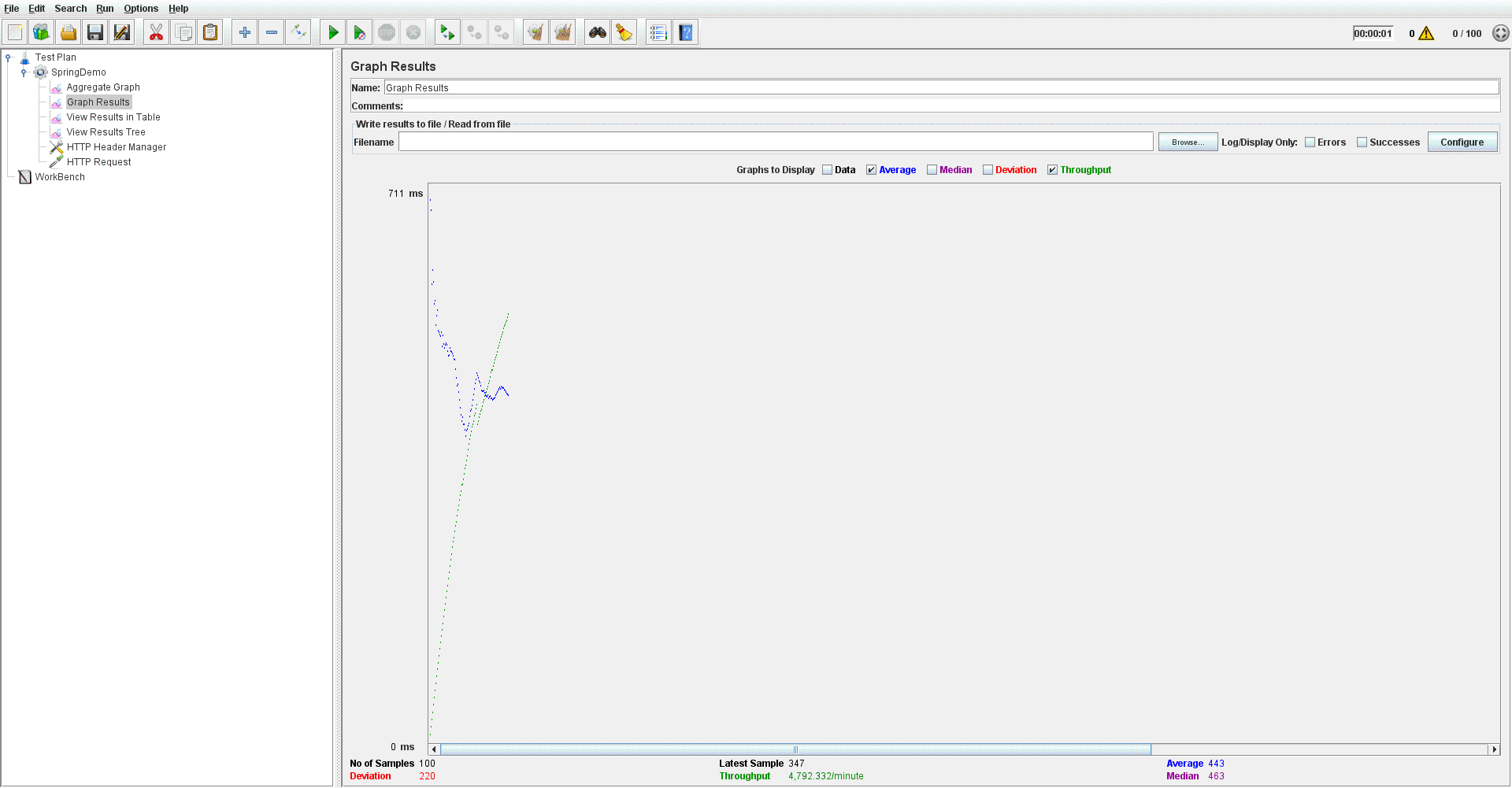


Test Result:

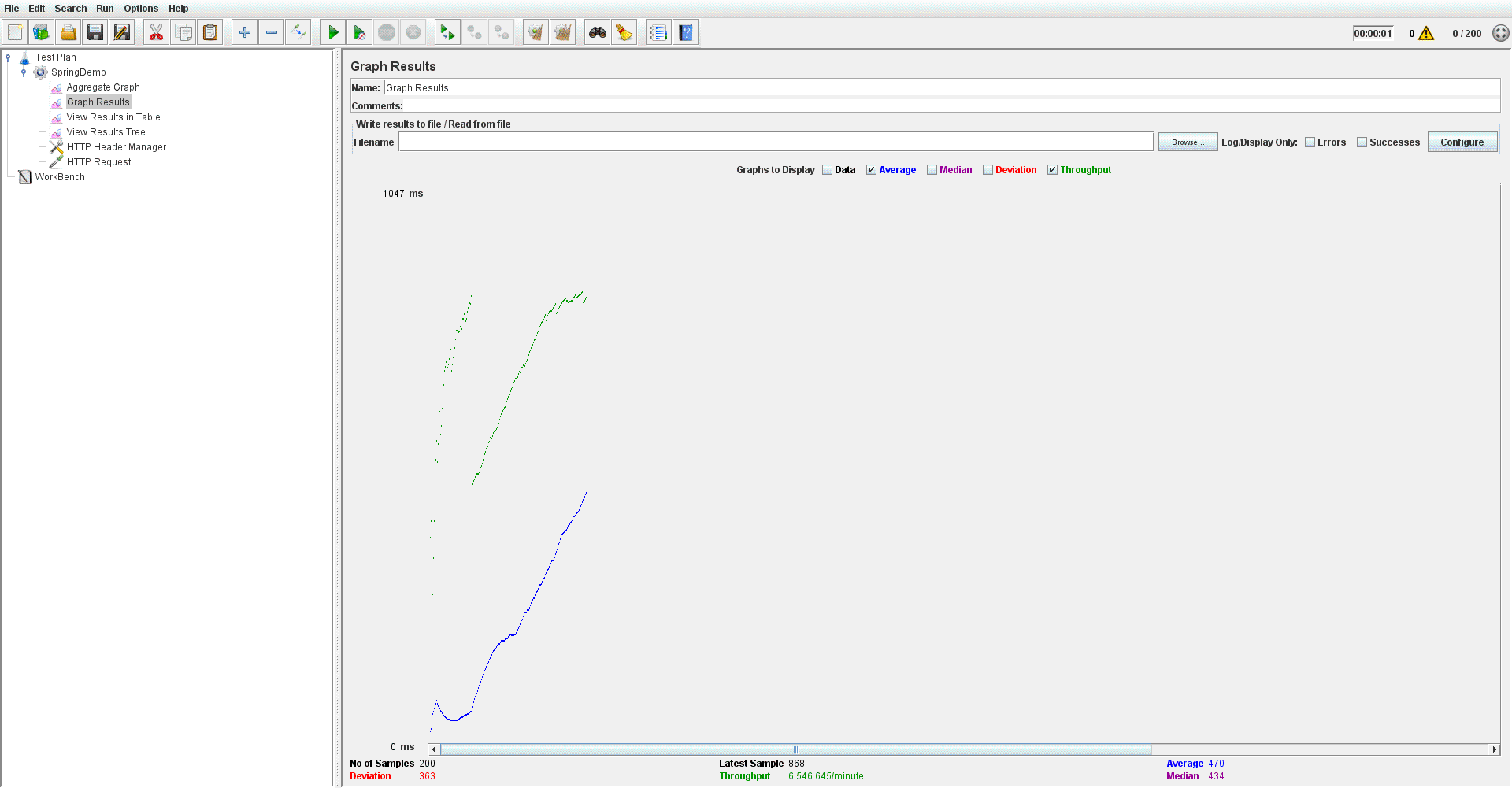


**JMeter Tests:**

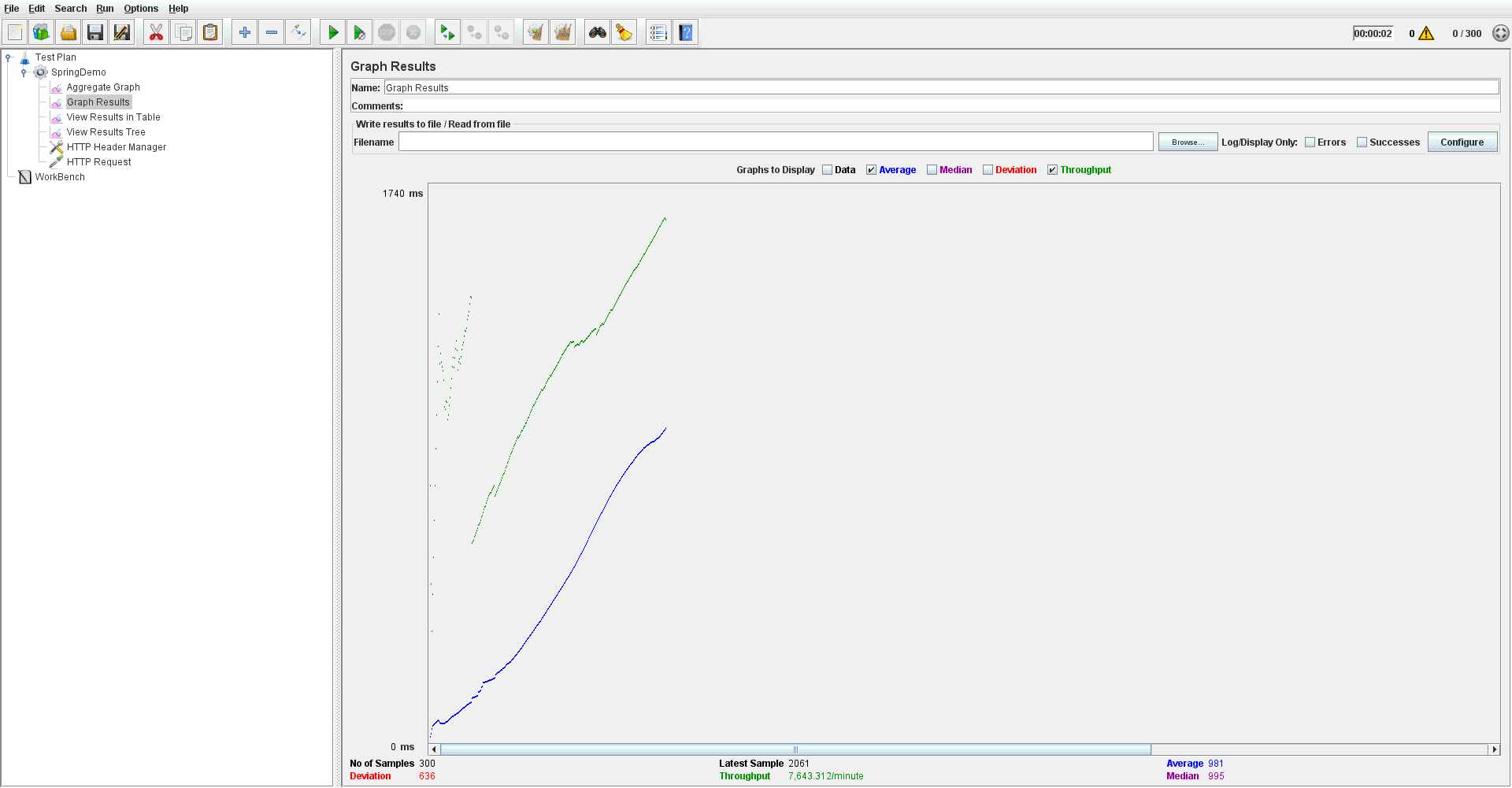
1. Graph Results :
2. 100 Concurrent users



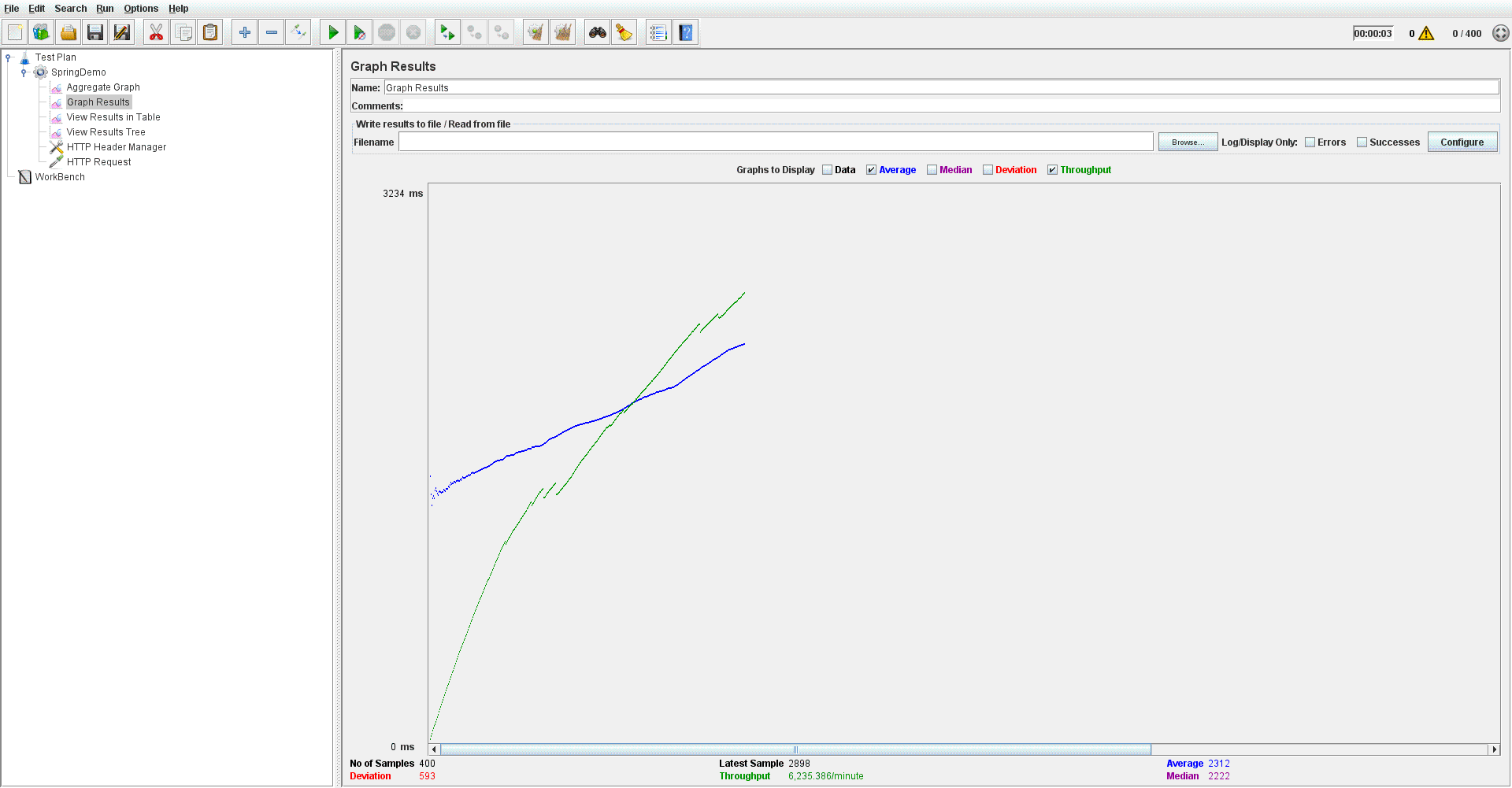
1. 200 Concurrent users



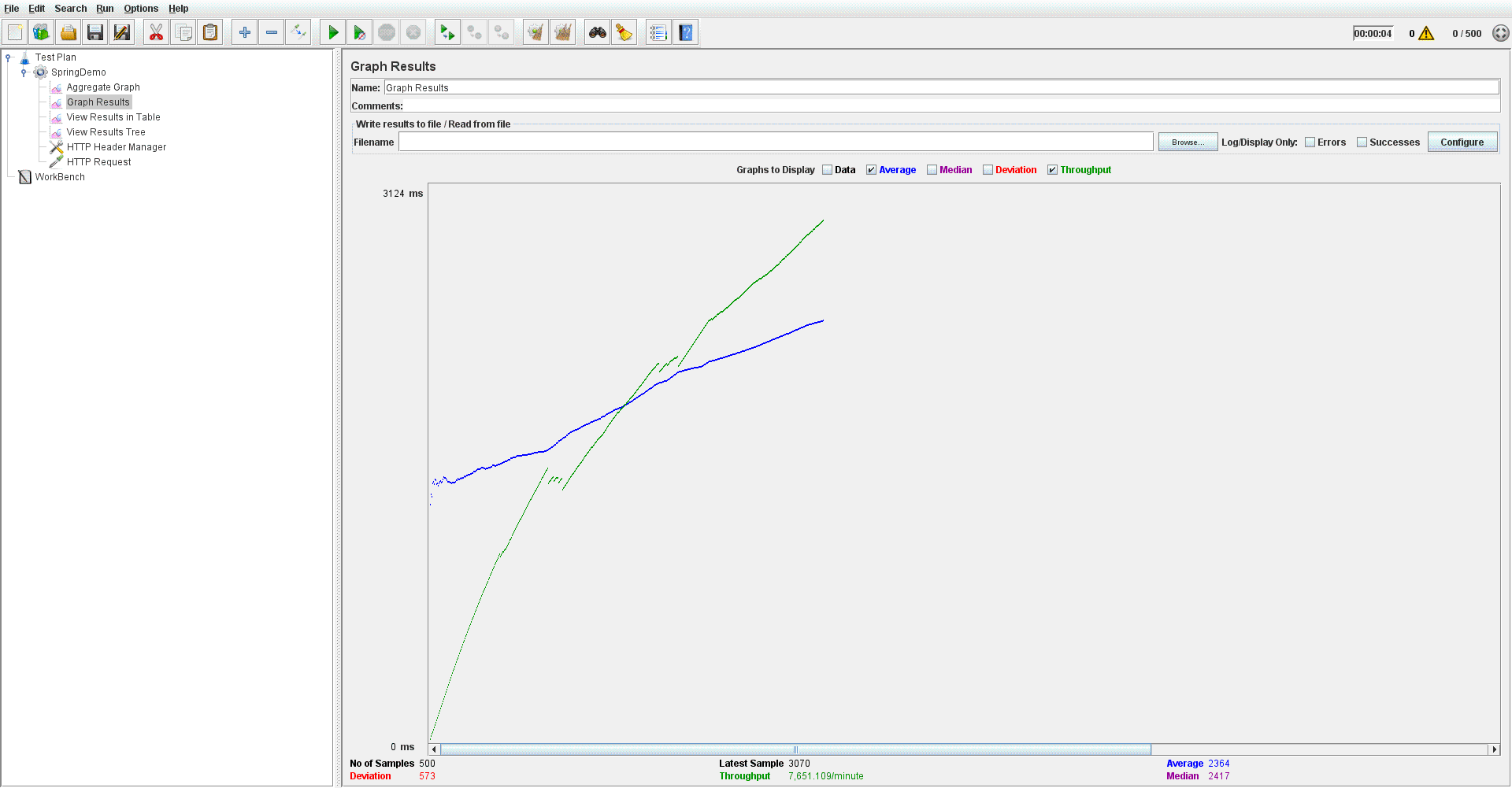
1. 300 Concurrent users



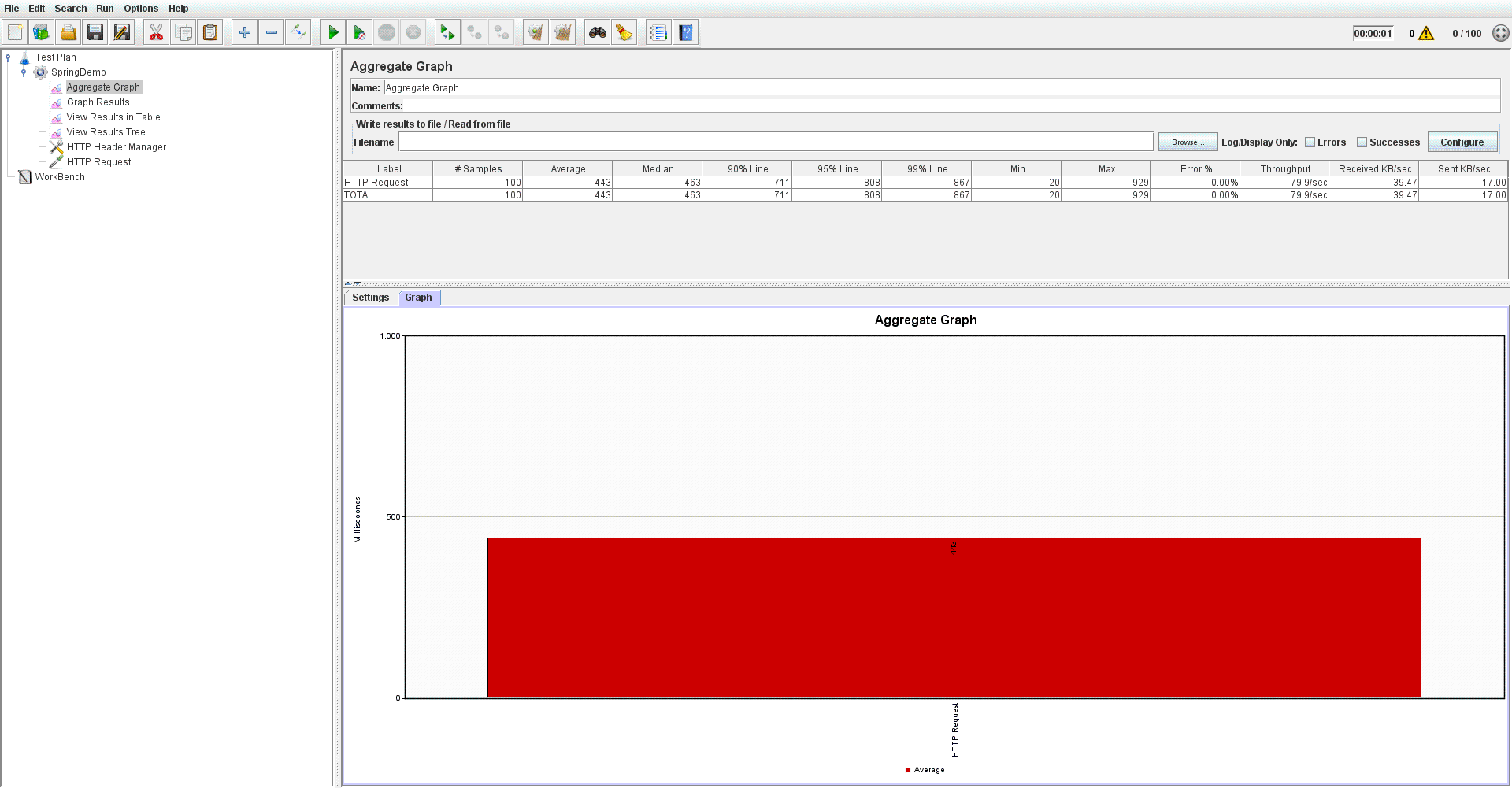
1. 400 Concurrent users



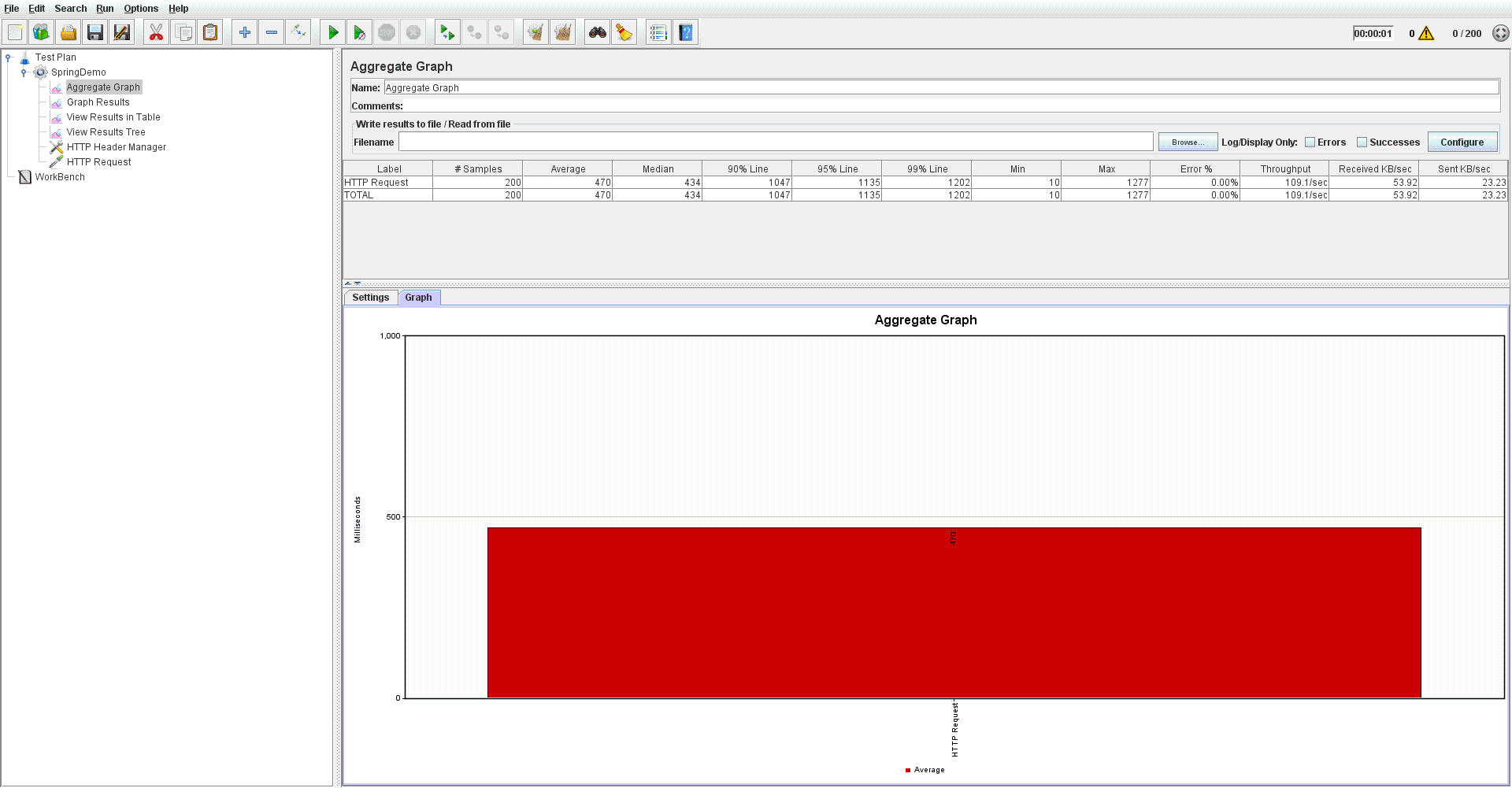
1. 500 Concurrent users



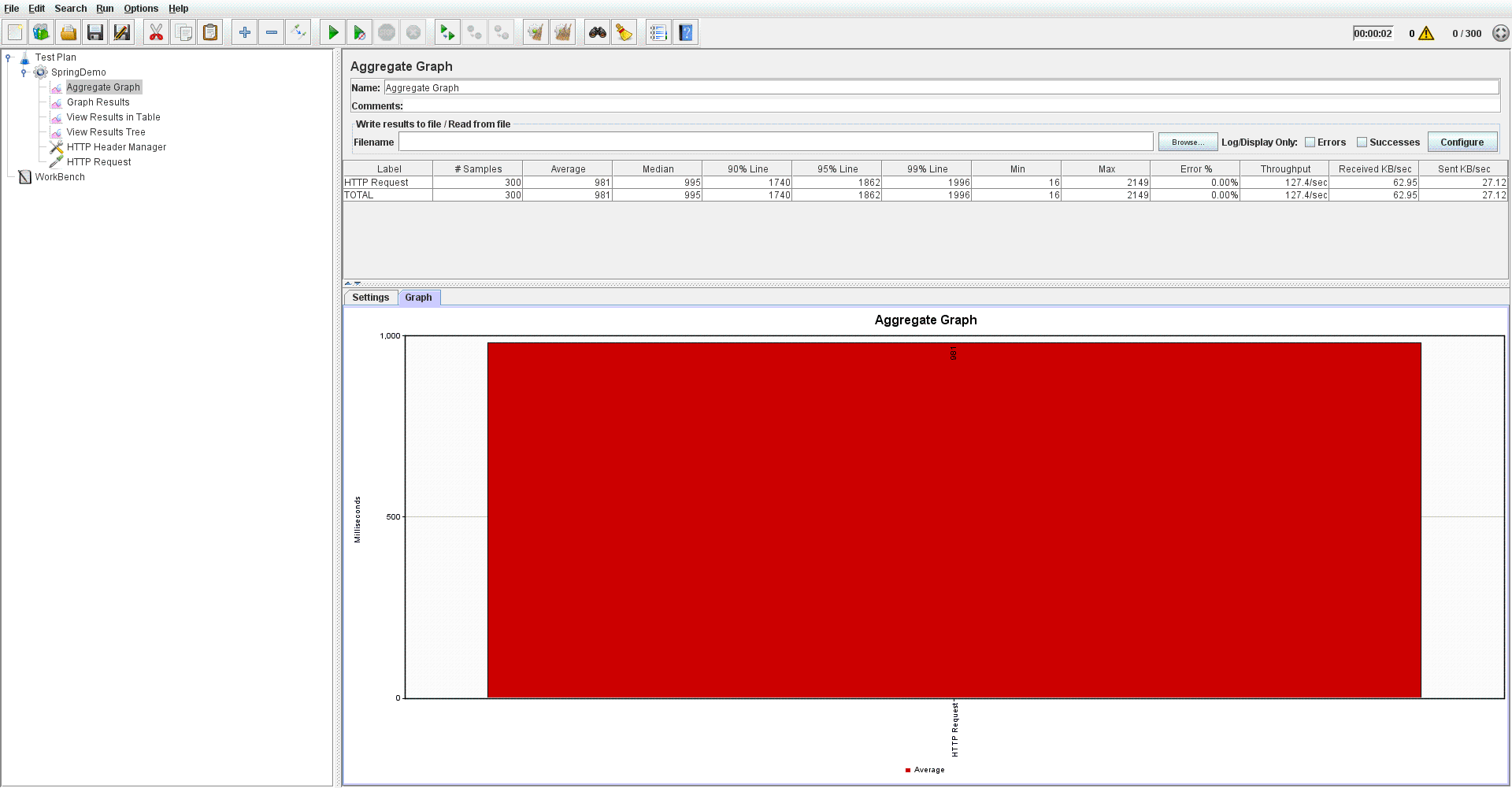
1. Aggregate Graph:
2. 100 Concurrent Users



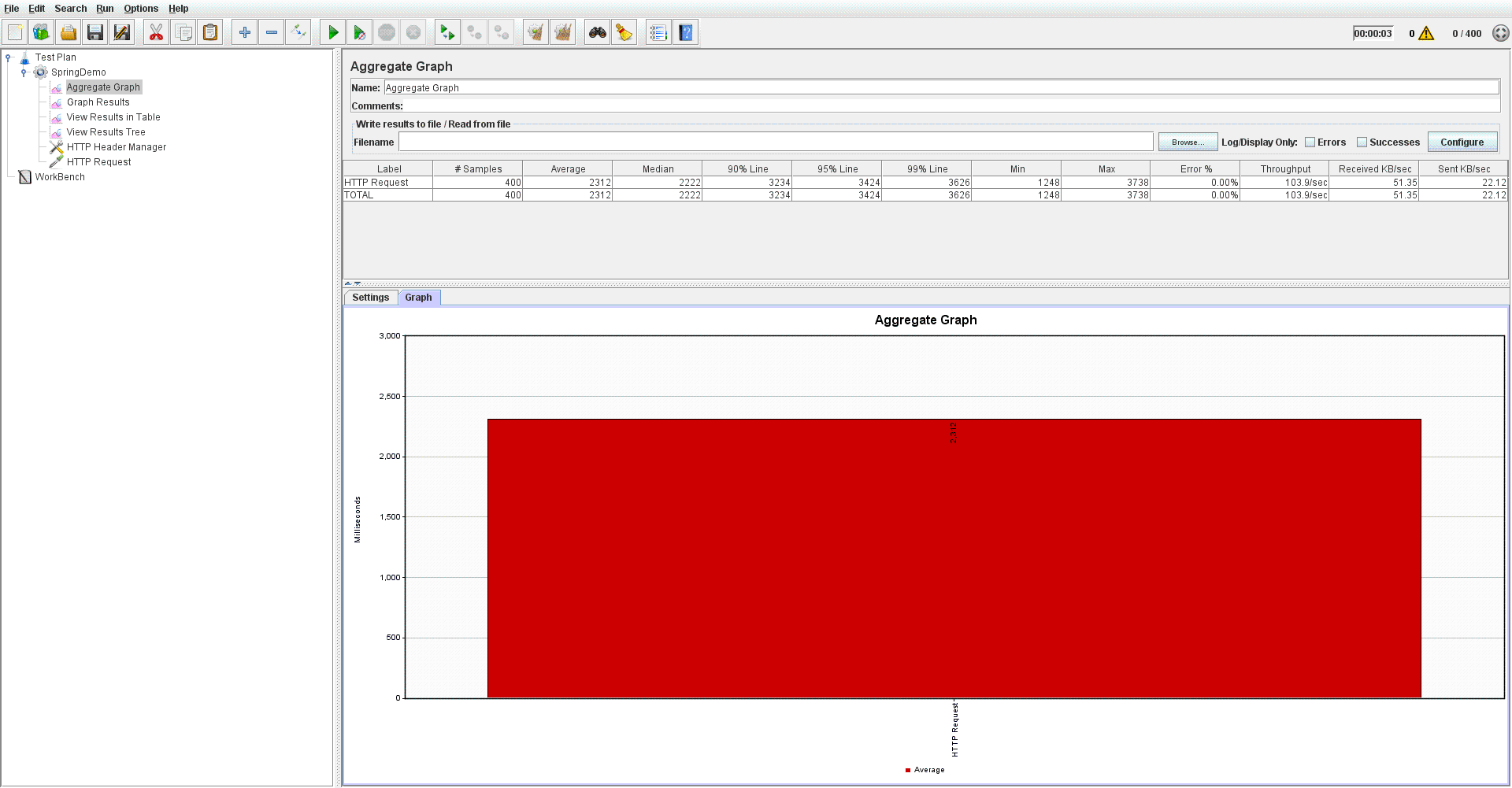
1. 200 Concurrent users



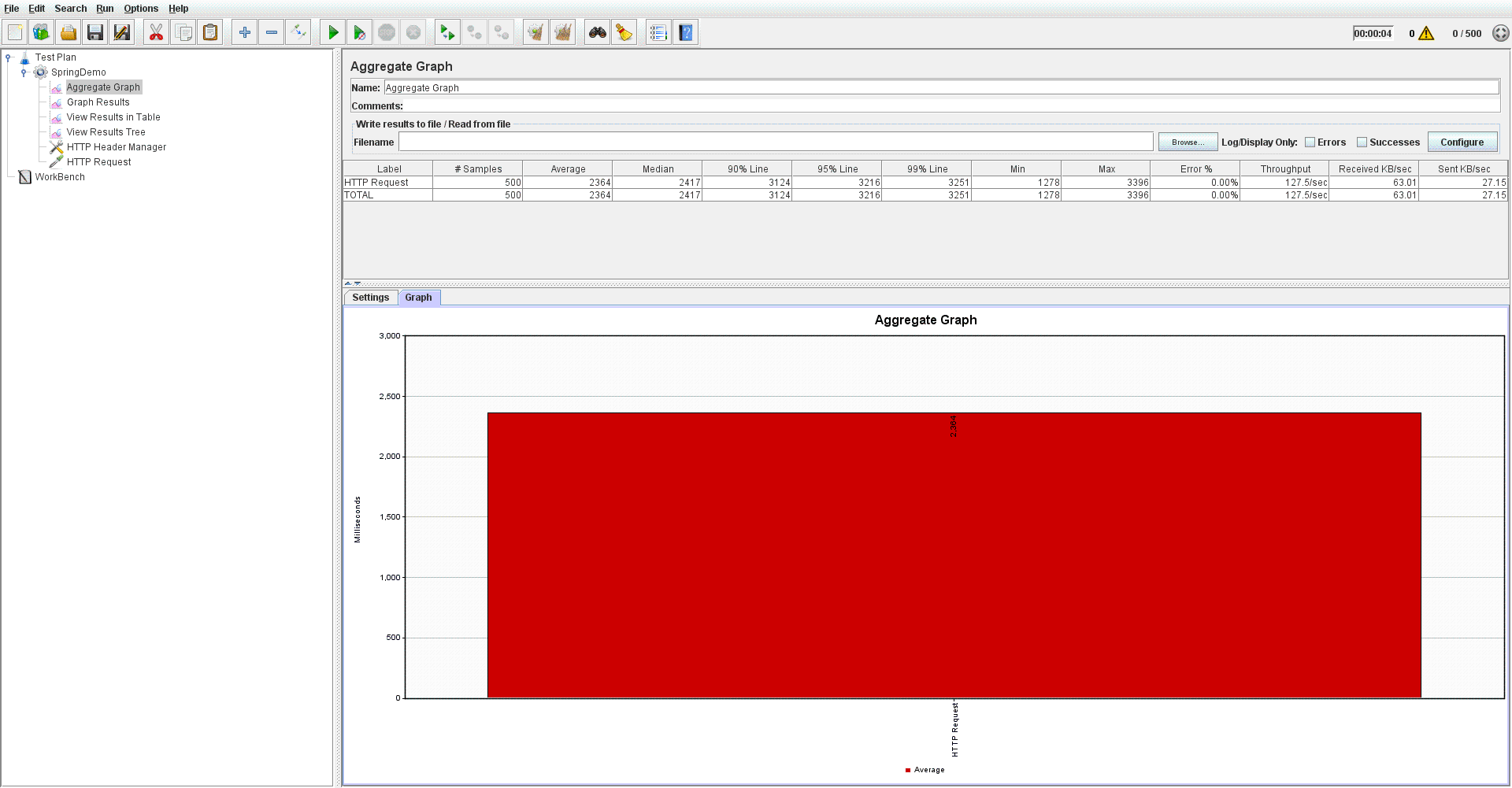
1. 300 Concurrent users



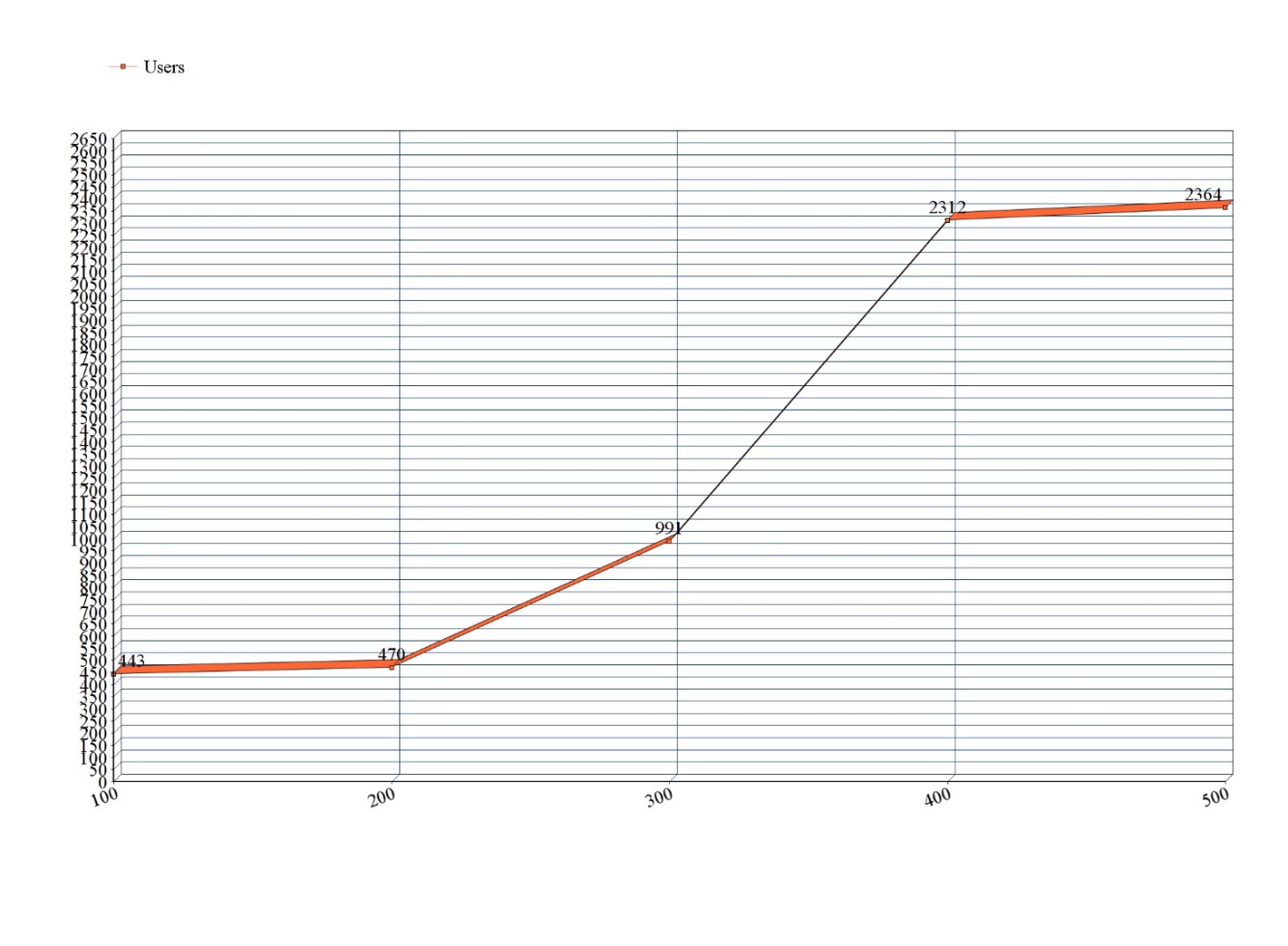
1. 400 Concurrent users



1. 500 Concurrent users

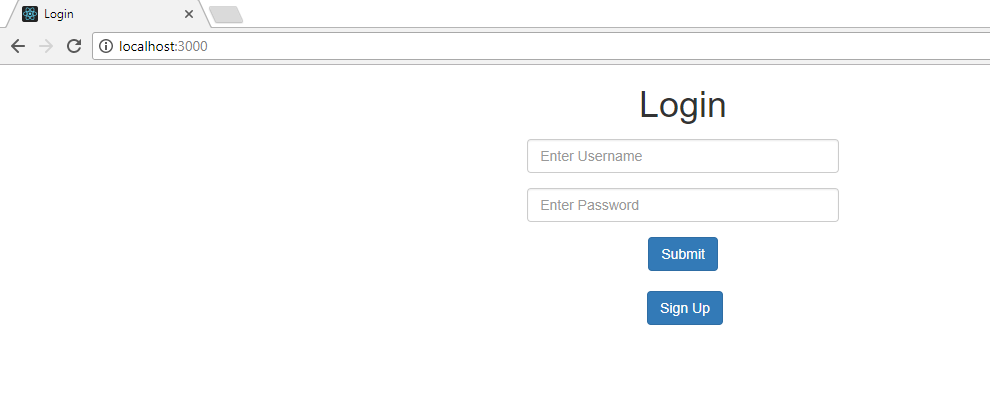


Comparison for average time taken for all the users :

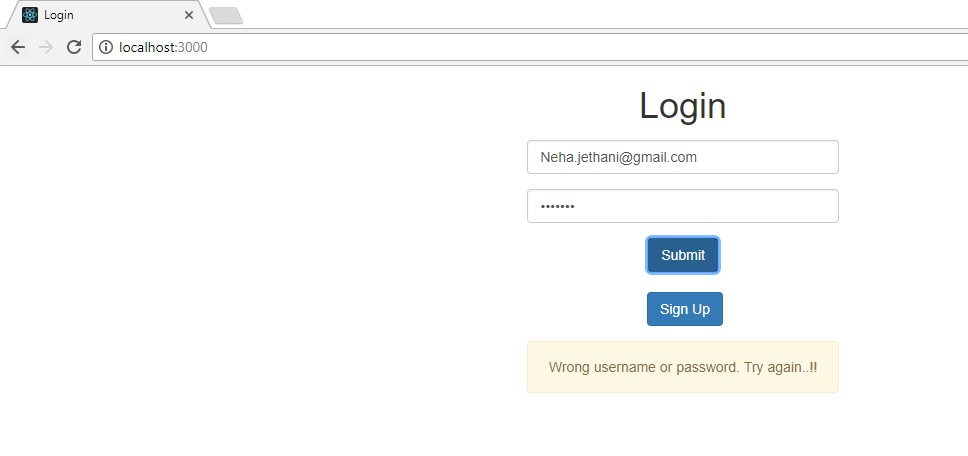


ScreenShots:

1. Main Landing Page is login page:



1. A user cannot login without valid credentials:



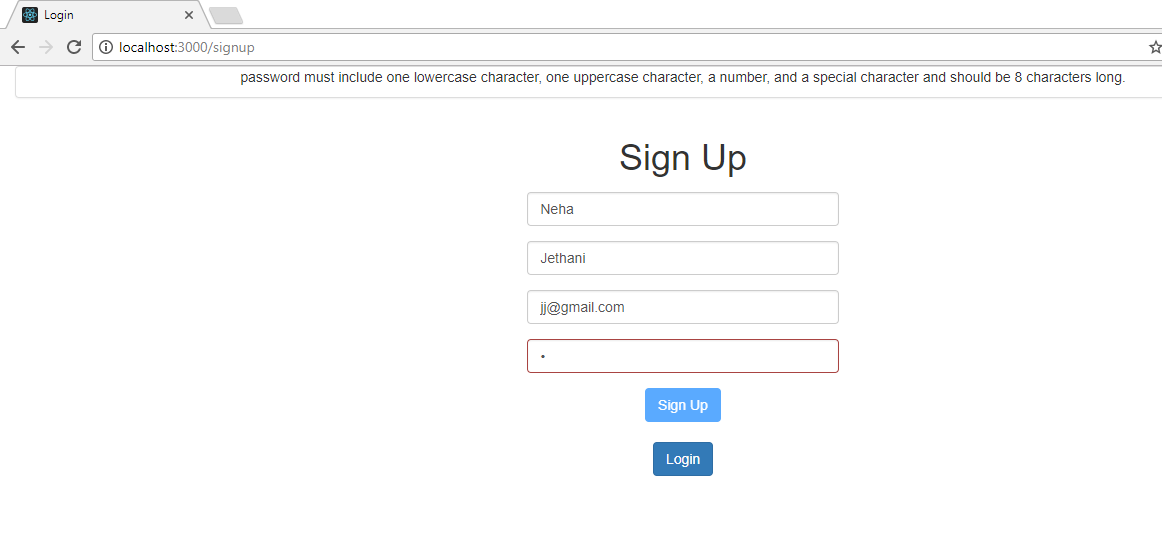
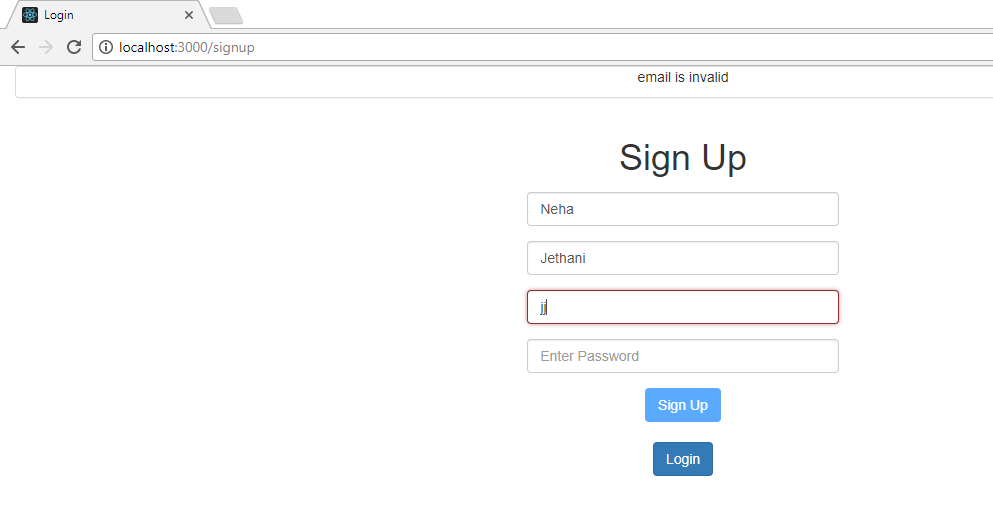
1. Sign Up page has all the validations

First Name and last name cannot be empty

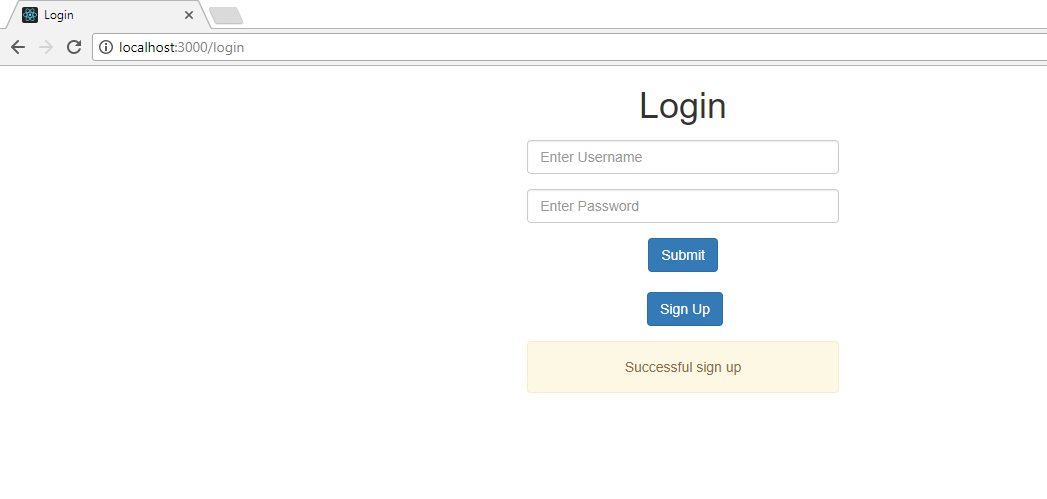
Email id should be in proper email format.

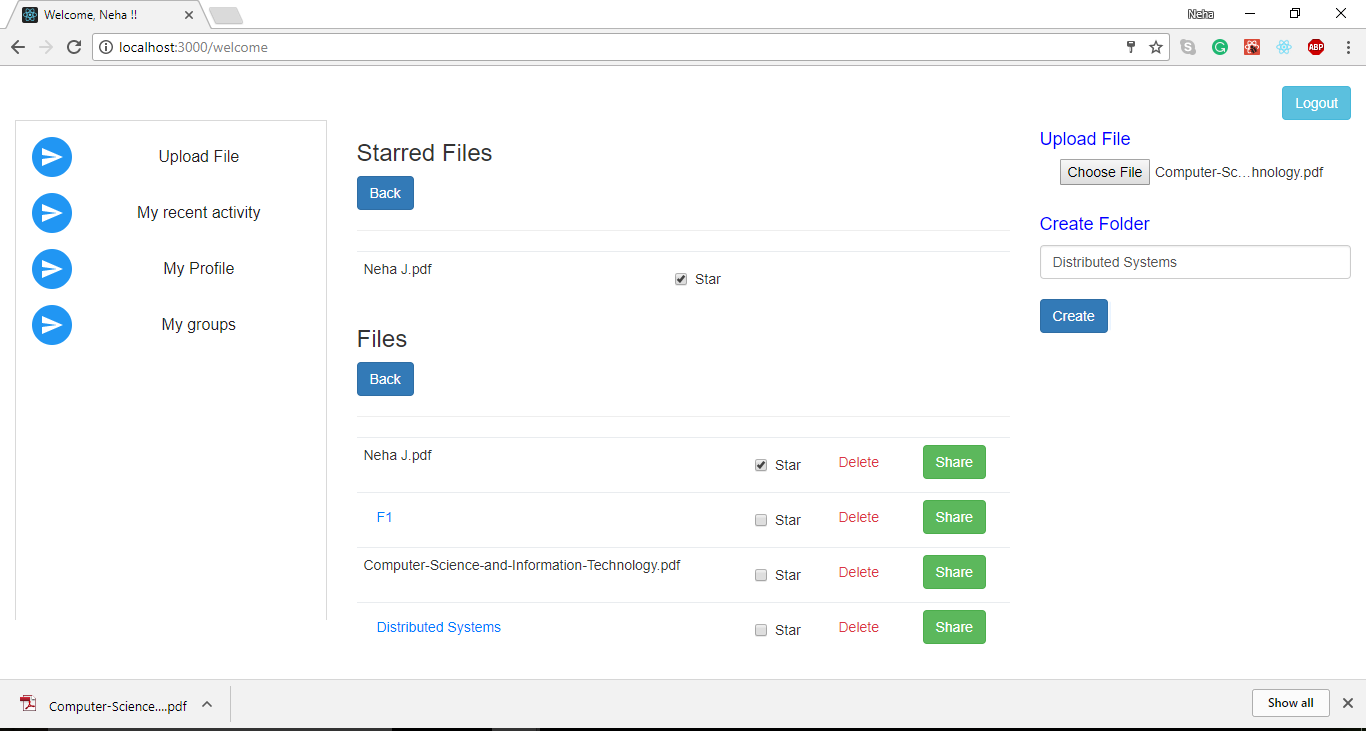
Password must contain at least one small case letter, one upper case letter, one special case letter and one numeric. Password should be of at least 8 characters

:



After successful sign up, user gets login page.





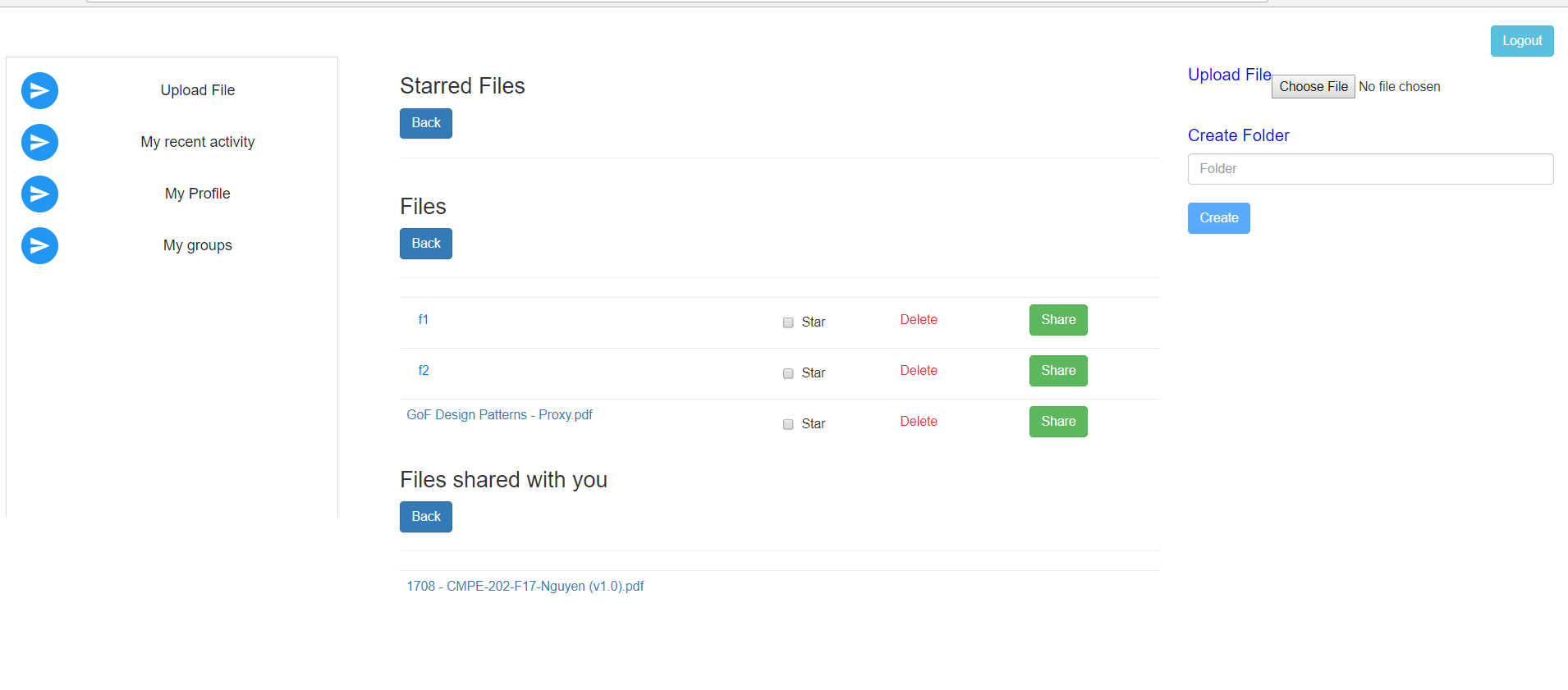
Successful login on server side

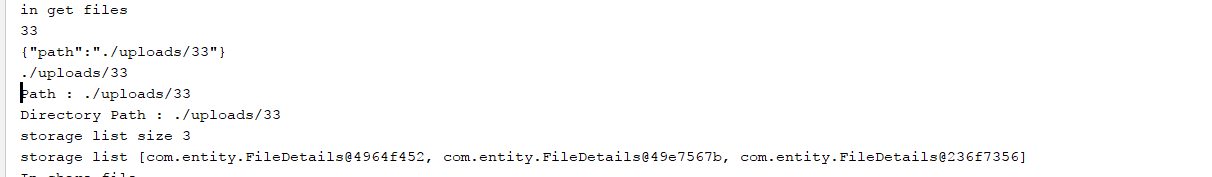


Existing user cannot sign up again with same email id.

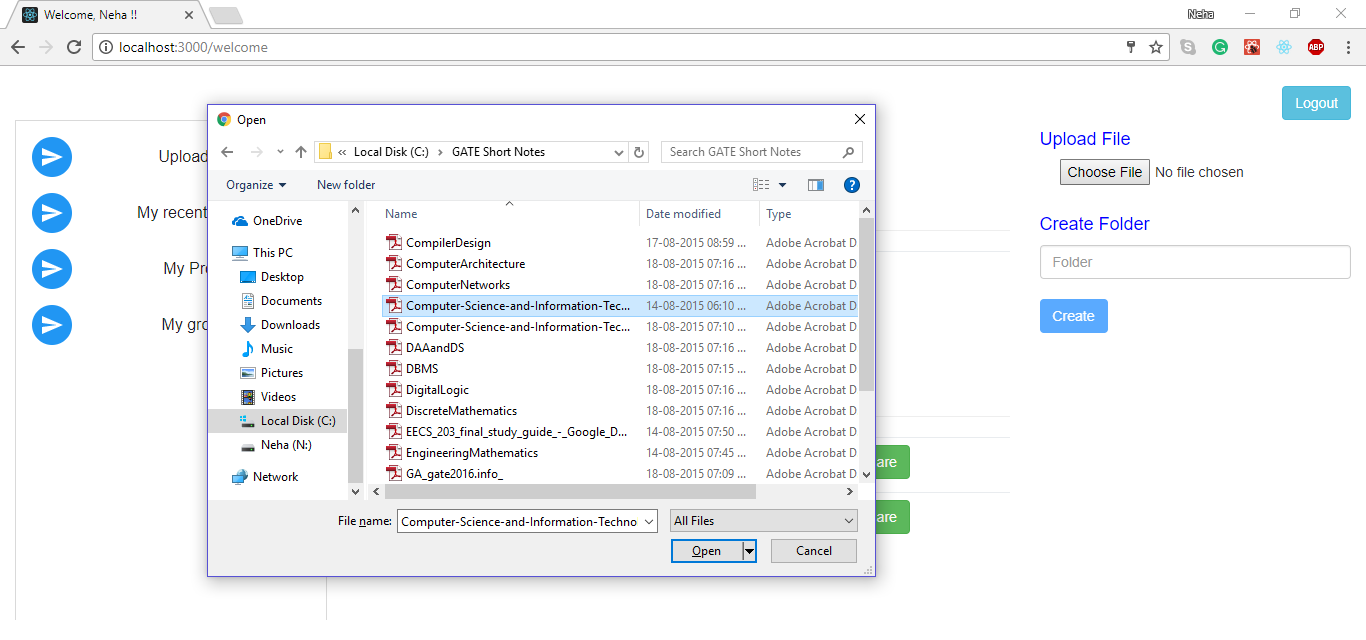
1. Home page after successful login:

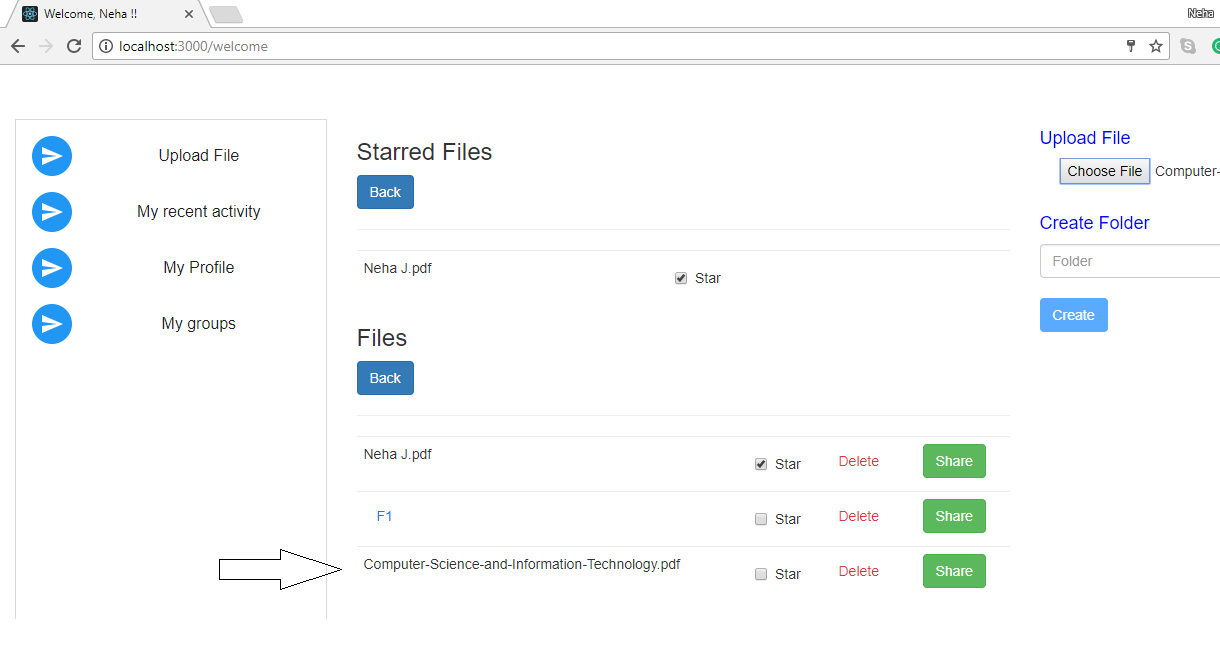
(User can view all the files and starred files on home page. User can also view files shared by others with that user.)





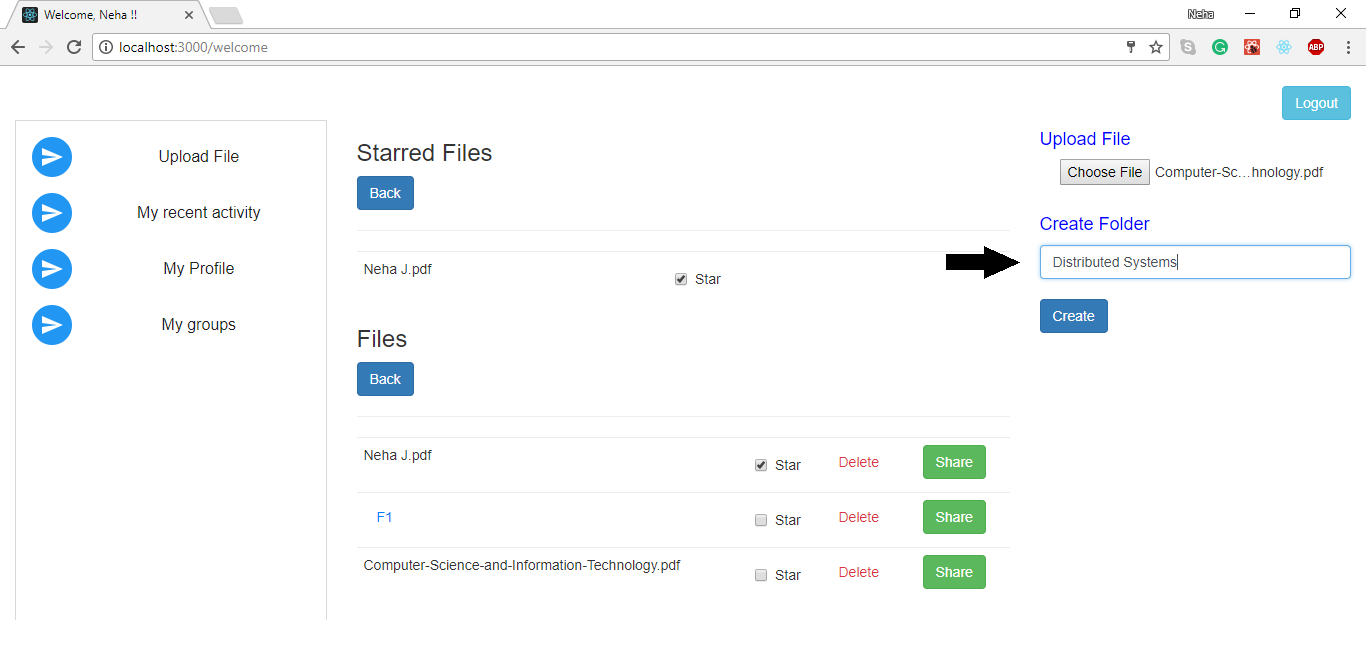
1. User can upload file and it will be displayed to the page.

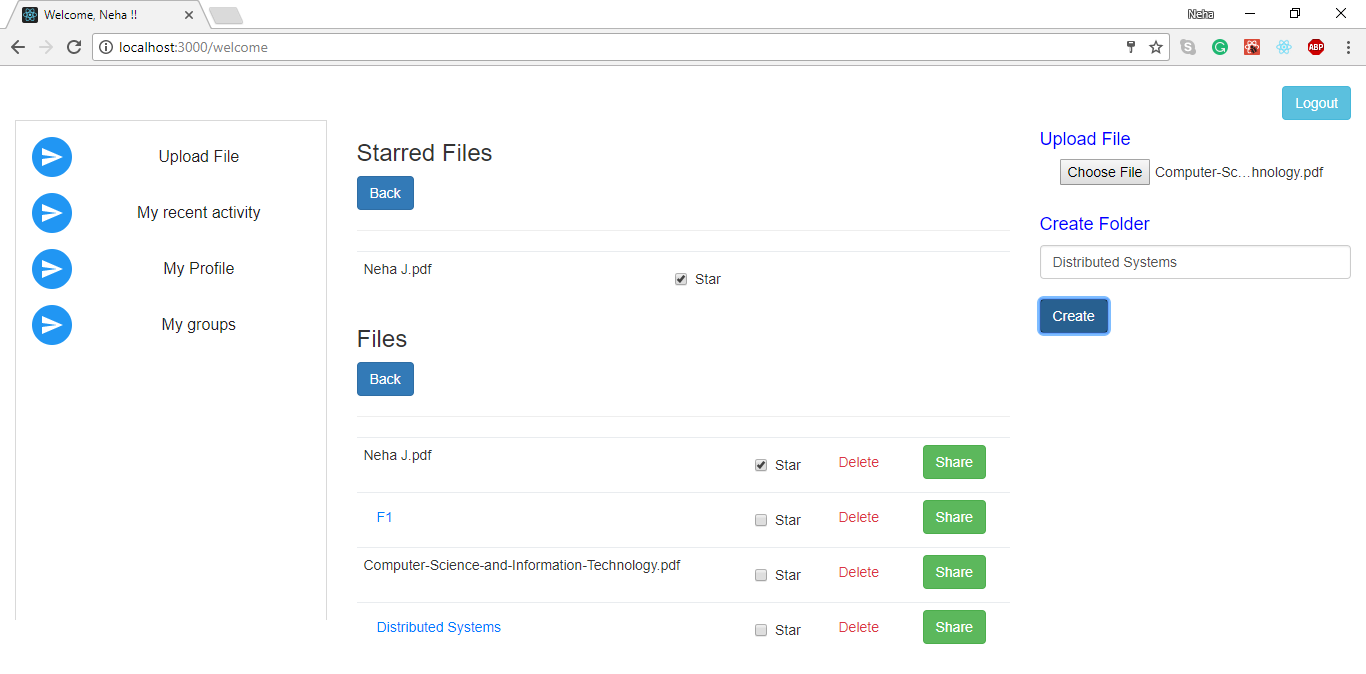




1. User can create directory.

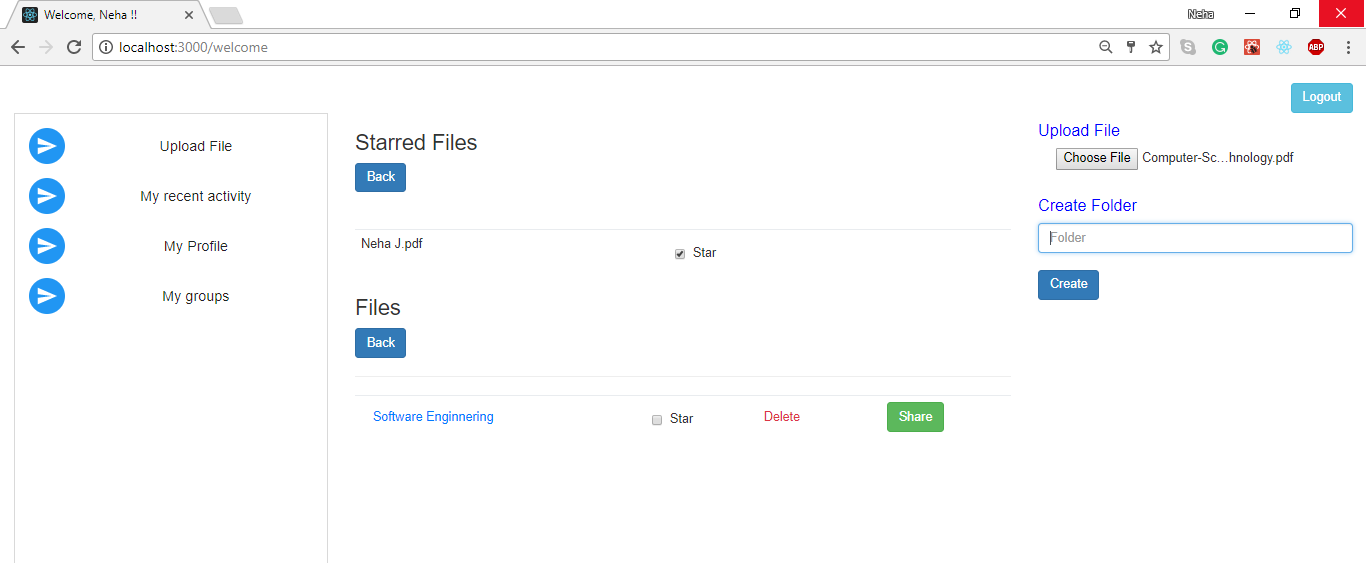
Before folder creation:



After Folder Creation:

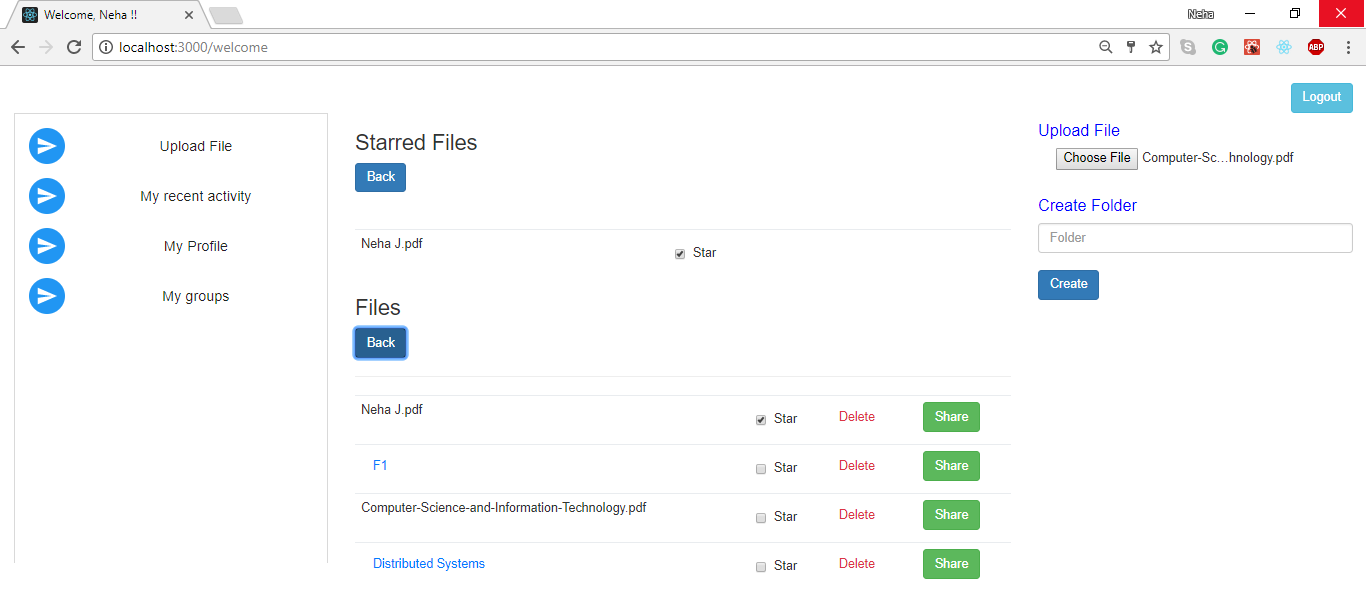
1. When user clicks on file, file is automatically downloaded.
2. Files can be uploaded and folders can be created inside folder.

(Inside ‘distributed systems’ folder)

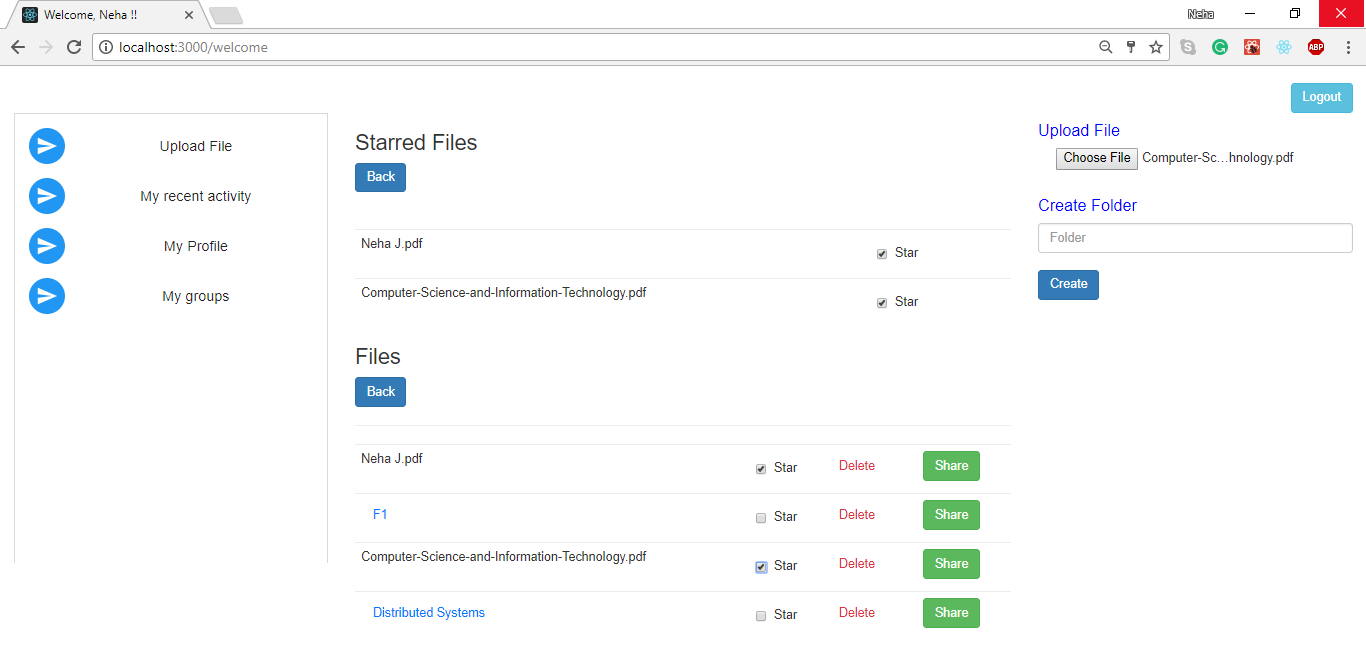


1. A user can star mark any file or folder and it will be displayed to starred file list on top.

Before Starring(File:Computer-Science-And-Information-Technology.pdf):

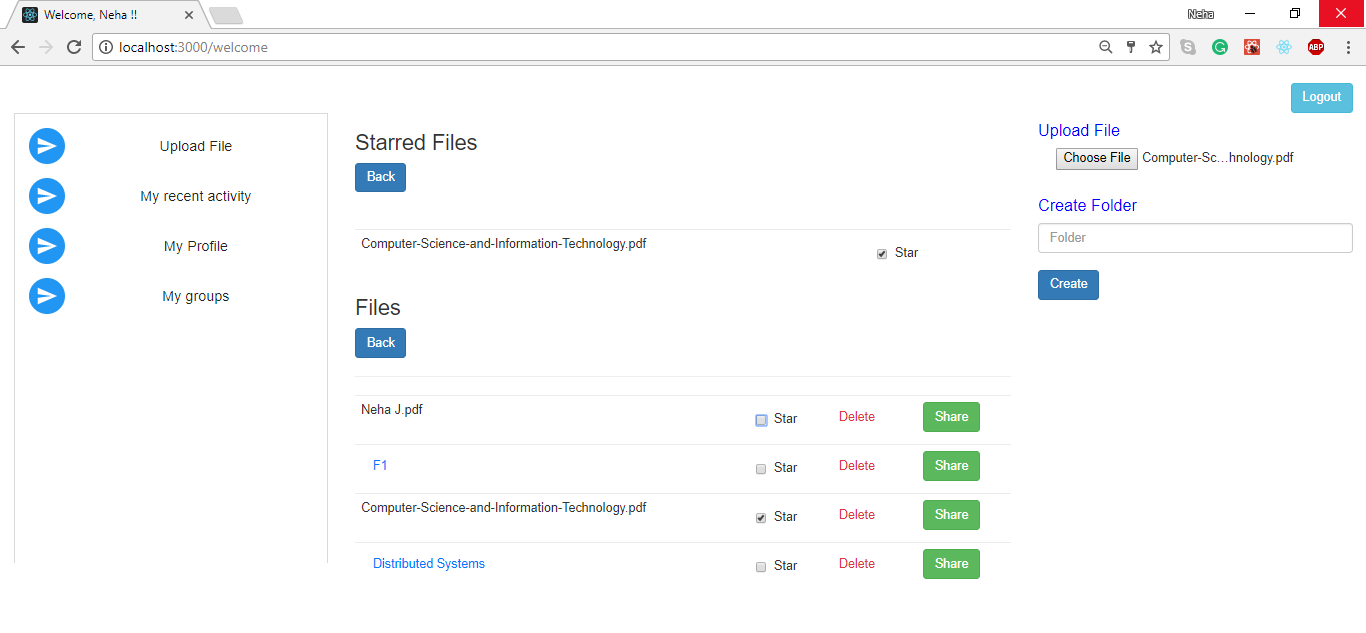


After Starring(File:Computer-Science-And-Information-Technology.pdf):



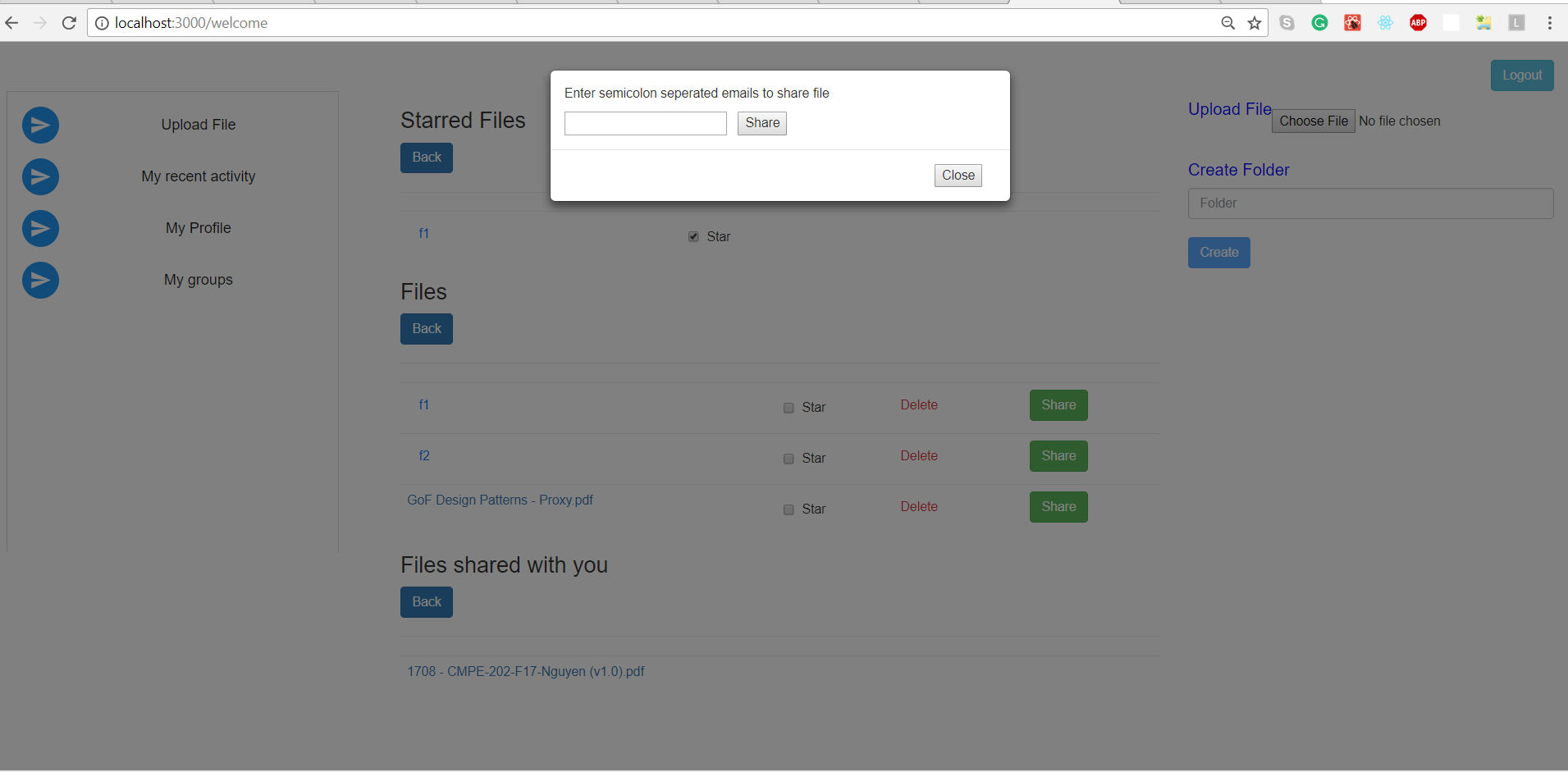
1. User can unstar the file/folder from starred file list or from uploaded file list and it will be removed form star marked files.

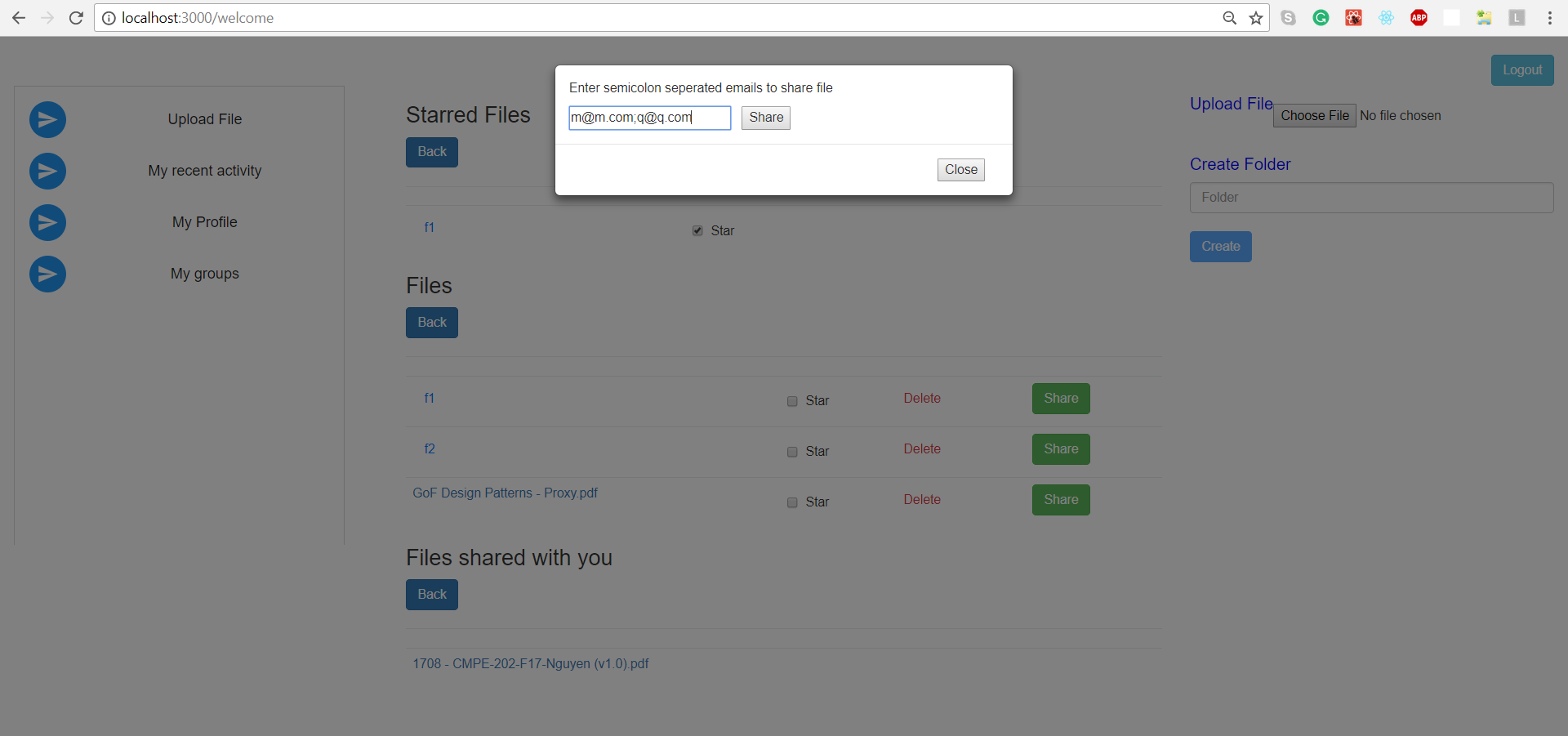
(After unstarring Neha J.pdf file)



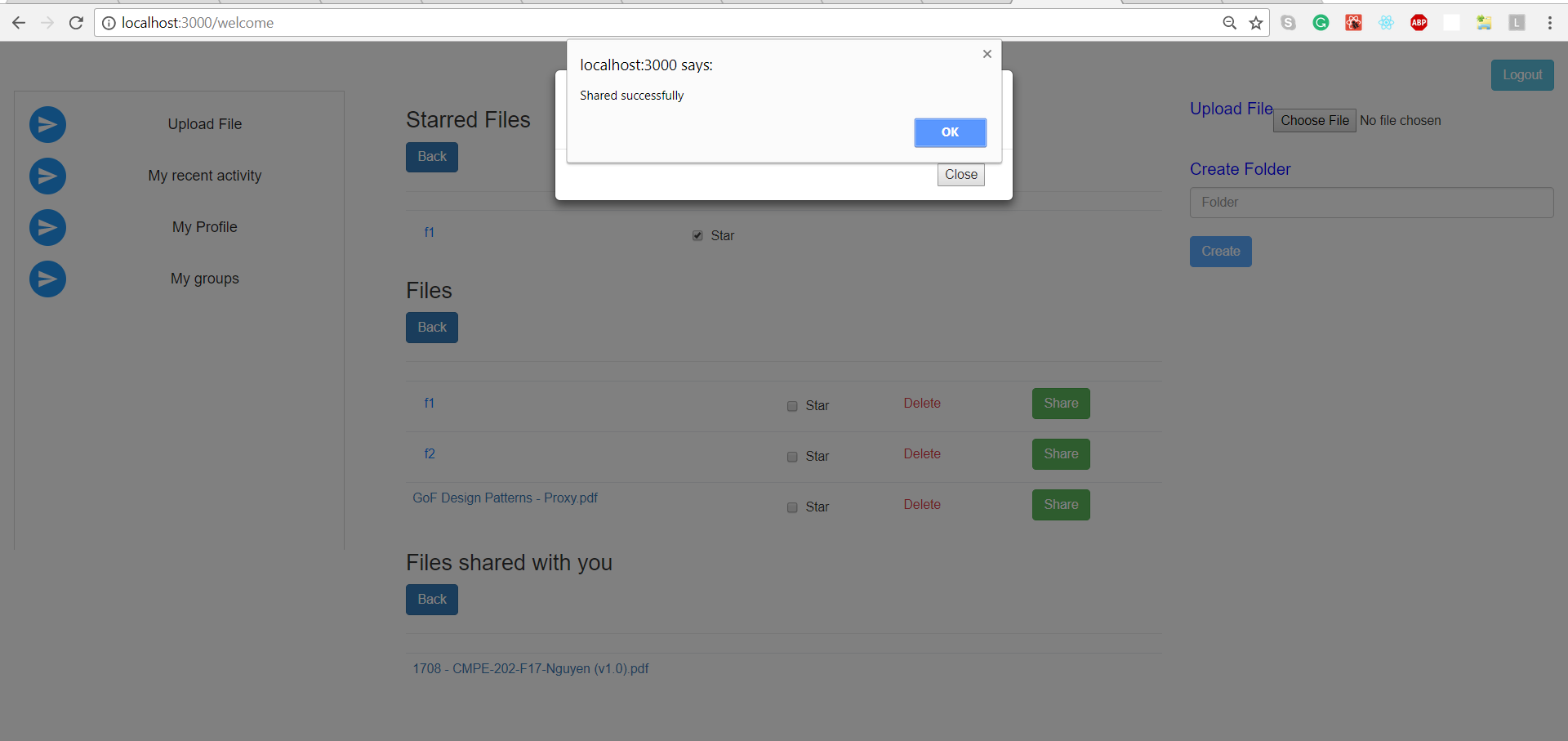
1. User can share files/folders via email ids.

If a user shares a file, it can be seen in dropbox login of all the users with whom it was shared.





The mentioned email ids receives the shared file/folder.

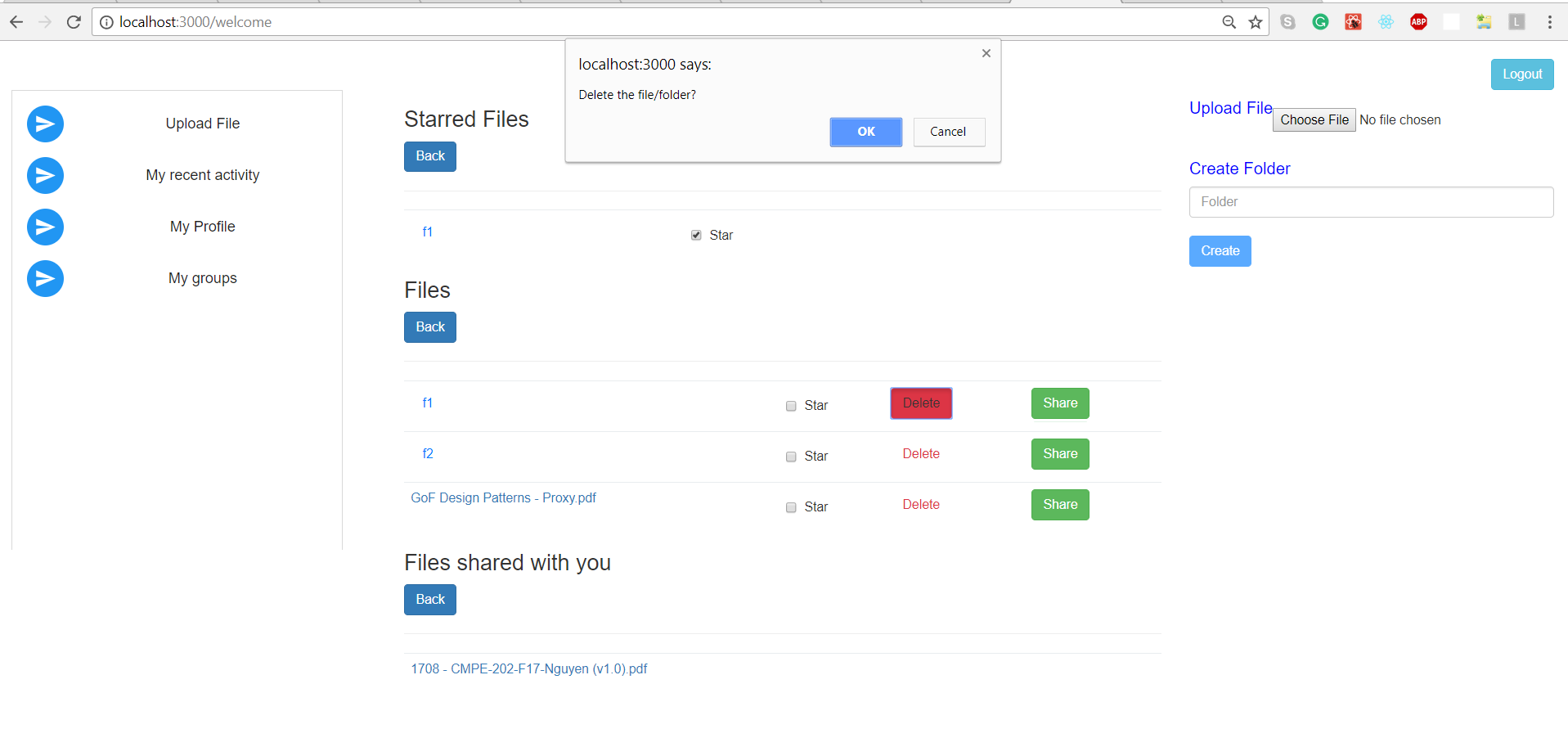


1. User can delete file/folder. And if the file/folder is present in starred list, it will be deleted from both the lists.

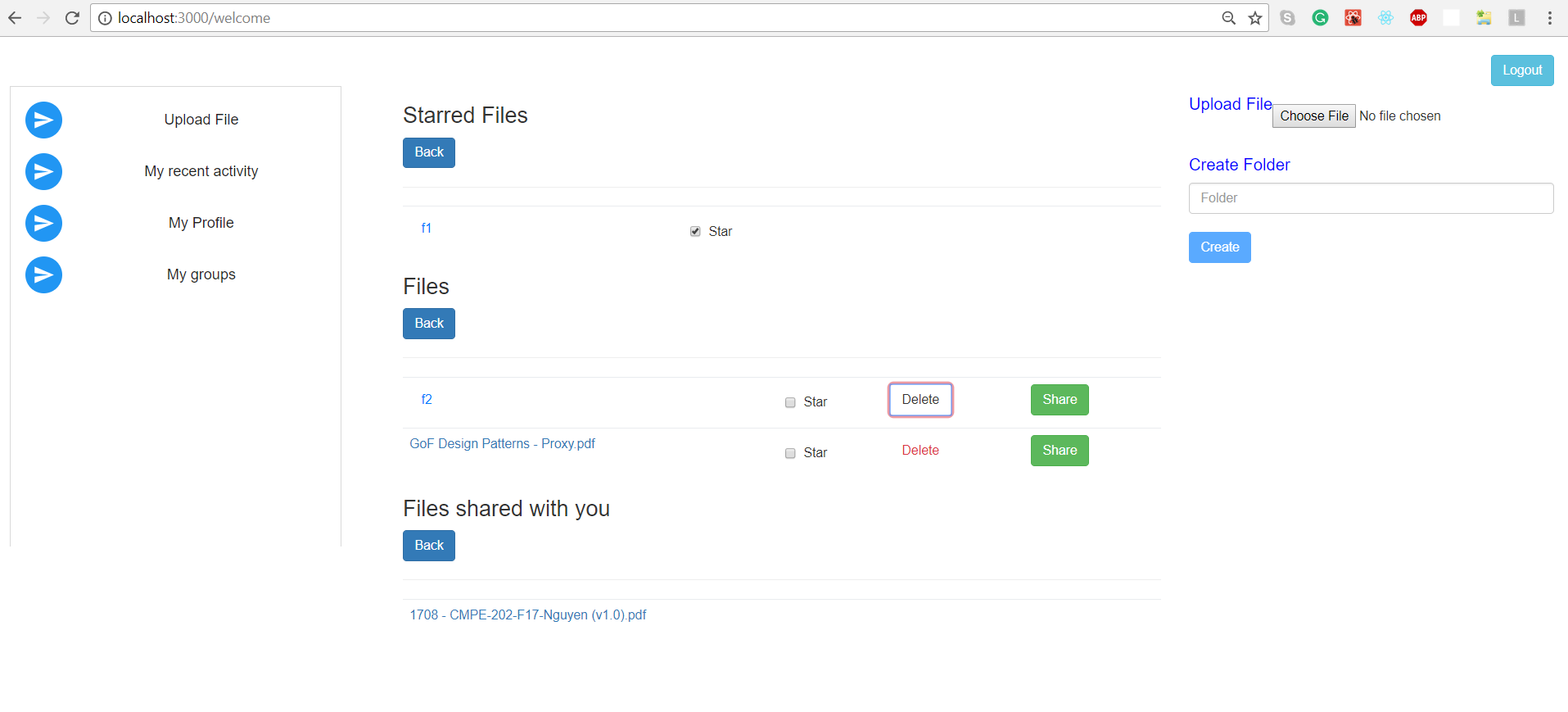
For folder creation, all the folder contents are recursively deleted.

**public void** deleteRecur(File file){  
 **for** (File childFile : file.listFiles()) {  
 **if** (childFile.isDirectory()) {  
 deleteRecur(childFile);  
 } **else** {  
 **if** (!childFile.delete()) {  
 System.***out***.println(**"error in recursion"**);  
 }  
 }  
 }  
  
 **if** (!file.delete()) {  
 System.***out***.println(**"error in recursion 1"**);  
  
 }  
}

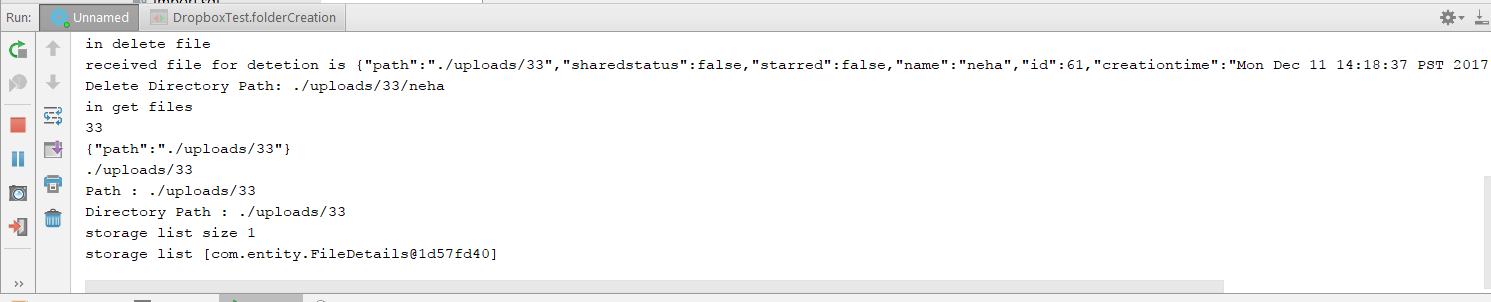
Before Deleting file:



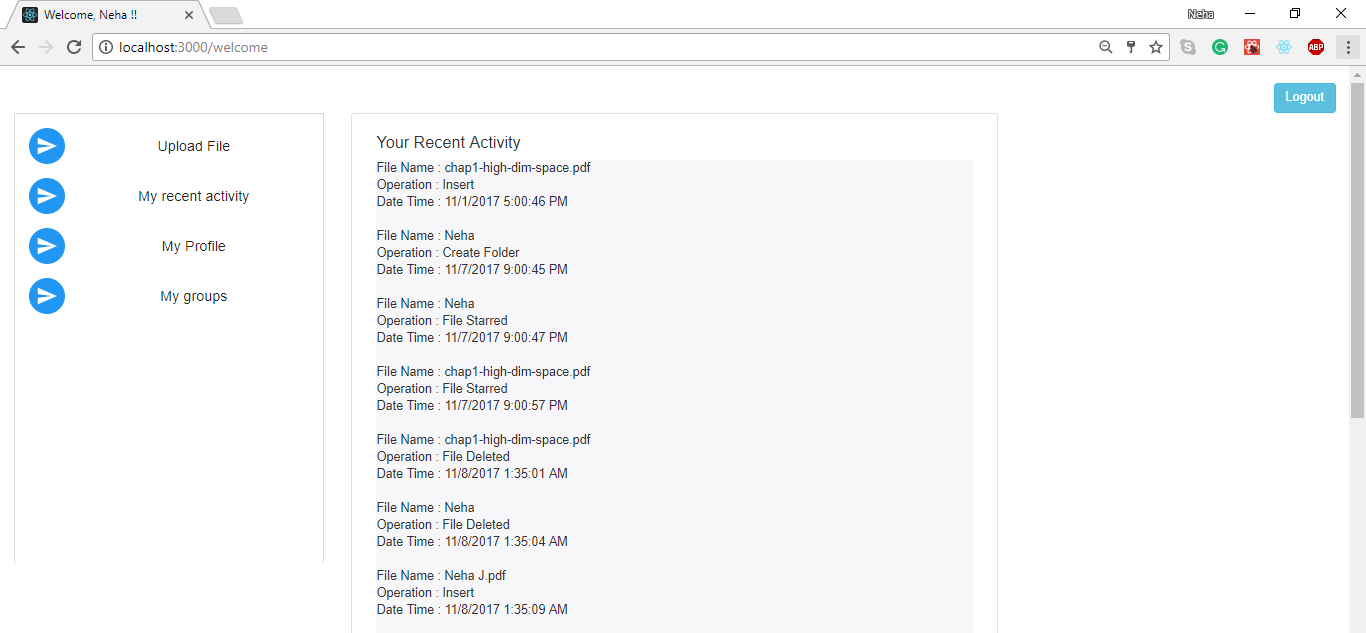
After file deletion:

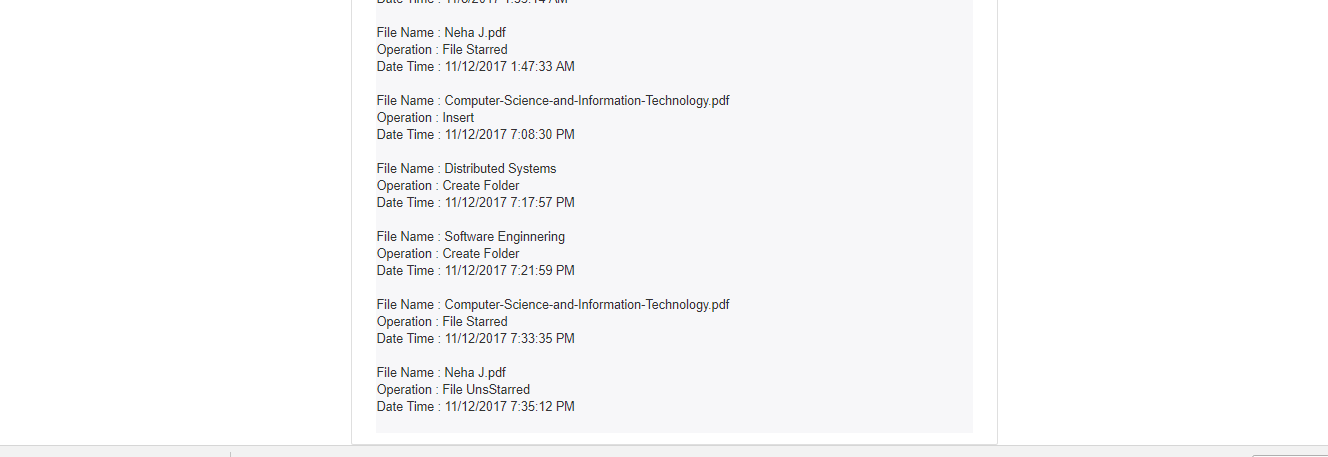


Server Logs for deletion :



1. User can view his recent activities(with activity date and time).



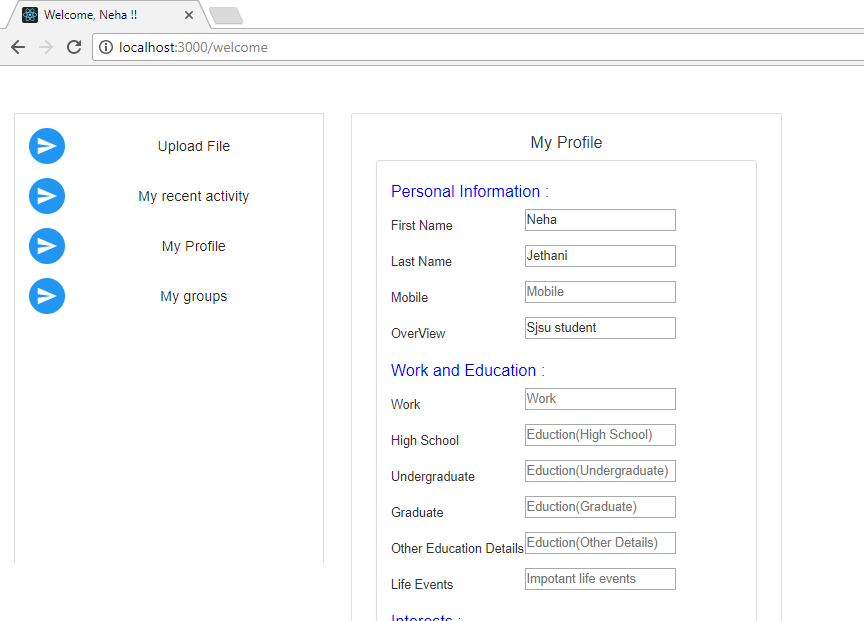


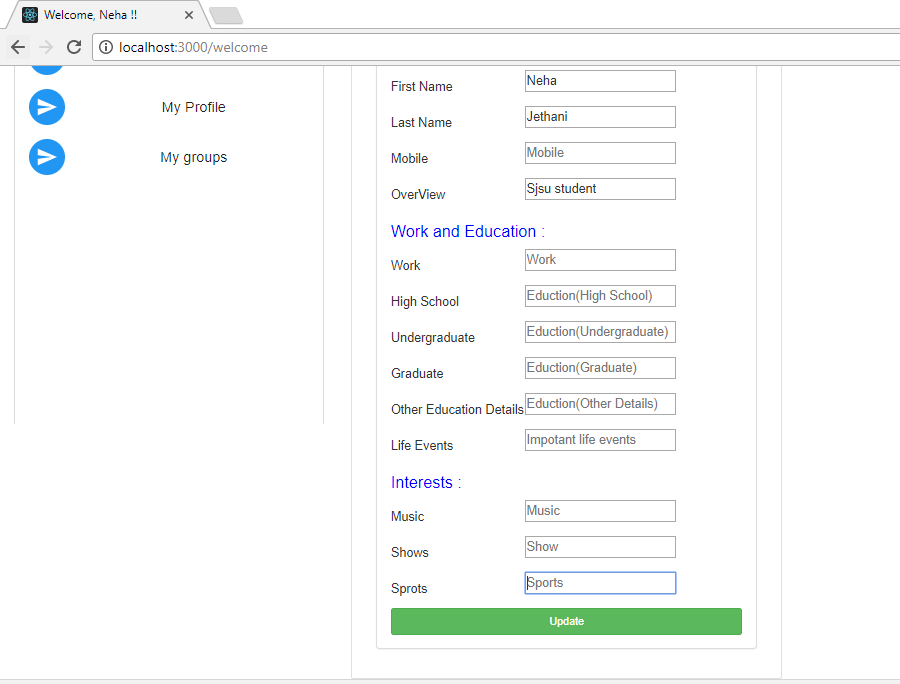
Server Logs for fetching user activity :



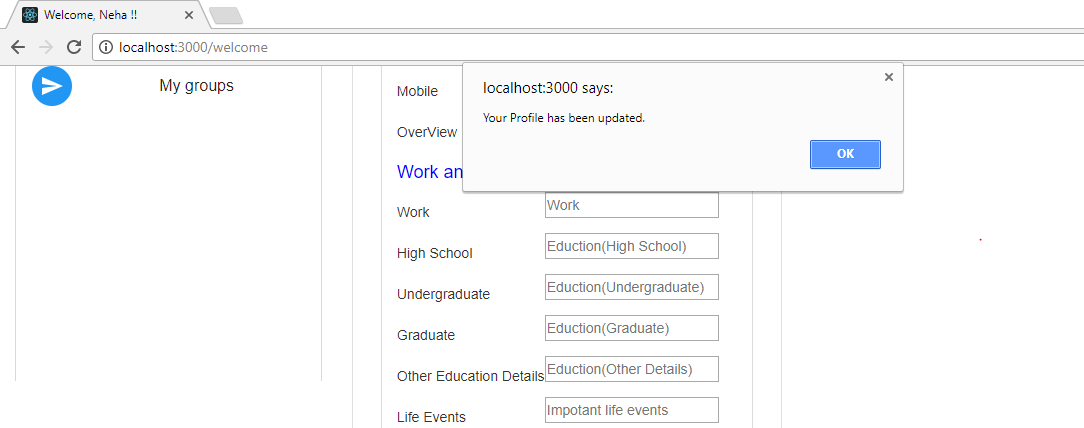
Activities done in previous screenshots(file upload, folder creation, file star/unstar) has been successfully noted.

1. User can view and edit his Personal information, Work and Education details and interests.

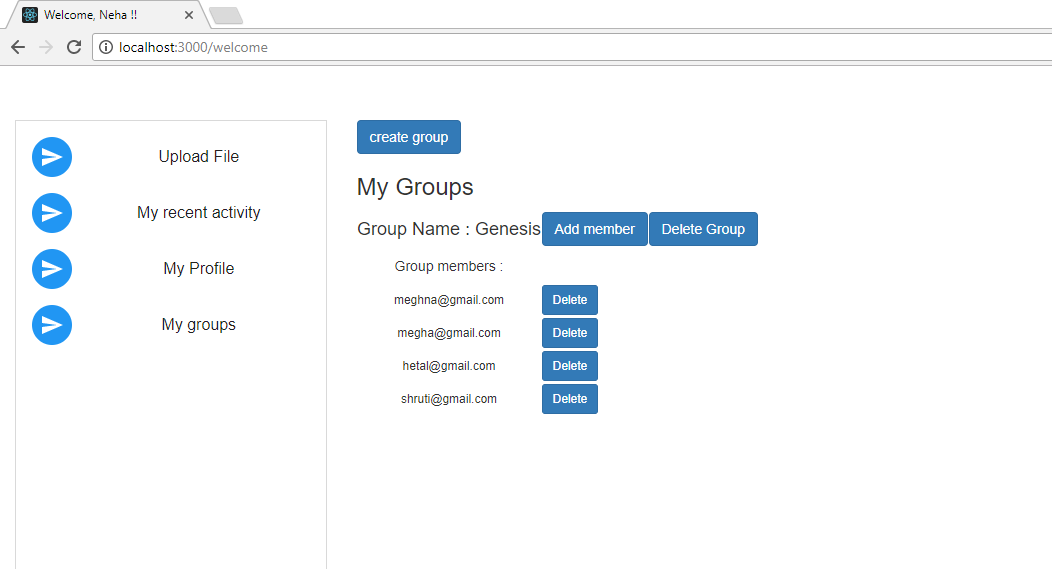


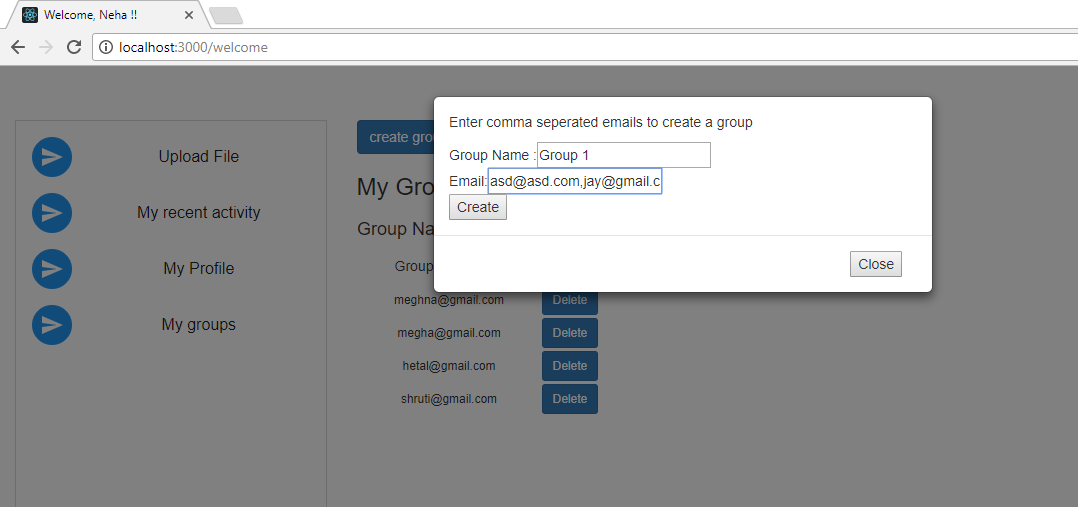


Profile Updation:

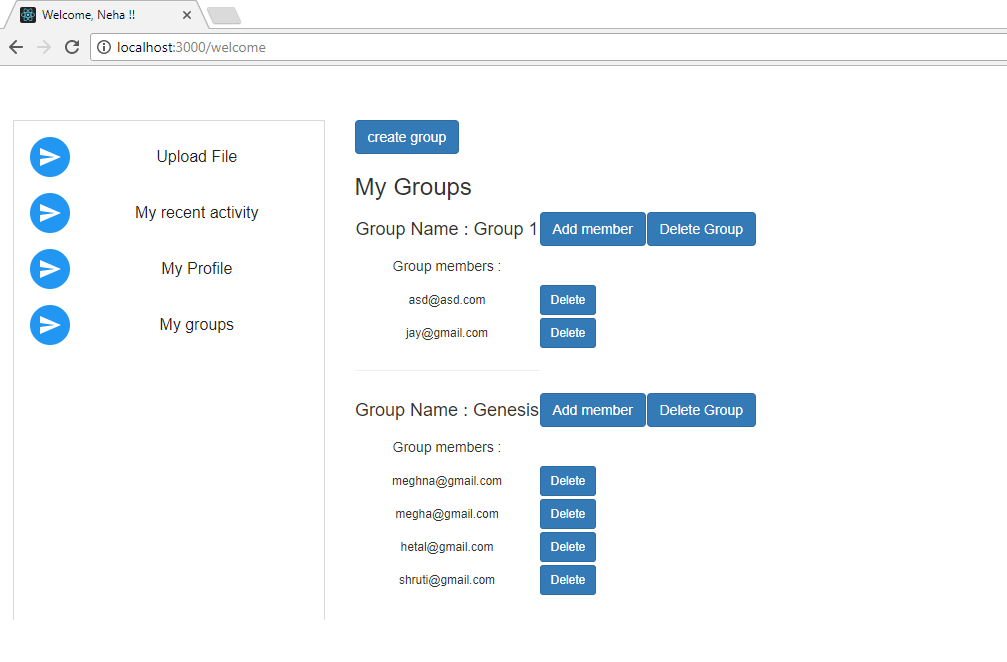


1. User can create group, add members to group, delete members from group and delete the groups.





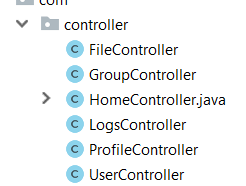
Group created:



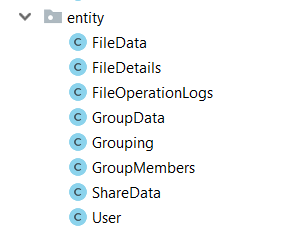
1. On logout, the session is destroyed and user gets back to landing page of application.

**Server Code Listing :**

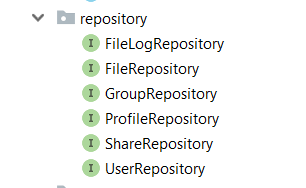
1. Controller



1. Entities



1. Repository



1. Service

