CMPE 273 – Enterprise Distributed Systems
Lab1: Splitwise Assignment

YouTube Link for Splitwise Application: https://youtu.be/vWbV0Tqy1TU

AWS Link for Splitwise Application: http://54.234.226.150:3000/

Github Link : https://github.com/Swes-sjsu/273-Lab-1-Splitwise.git

Name: Swetha Singi Reddy
SJSU ID: 015354015

Introduction

Splitwise is a widely used application that facilitates bill management where the bills are split among the members of a group. It also keeps track of bills paid and who owes how much. In this Lab1 assignment we will be building a prototype similar to Splitwise application where bills are split equally among the members of a group once they become part of the group.

Goal

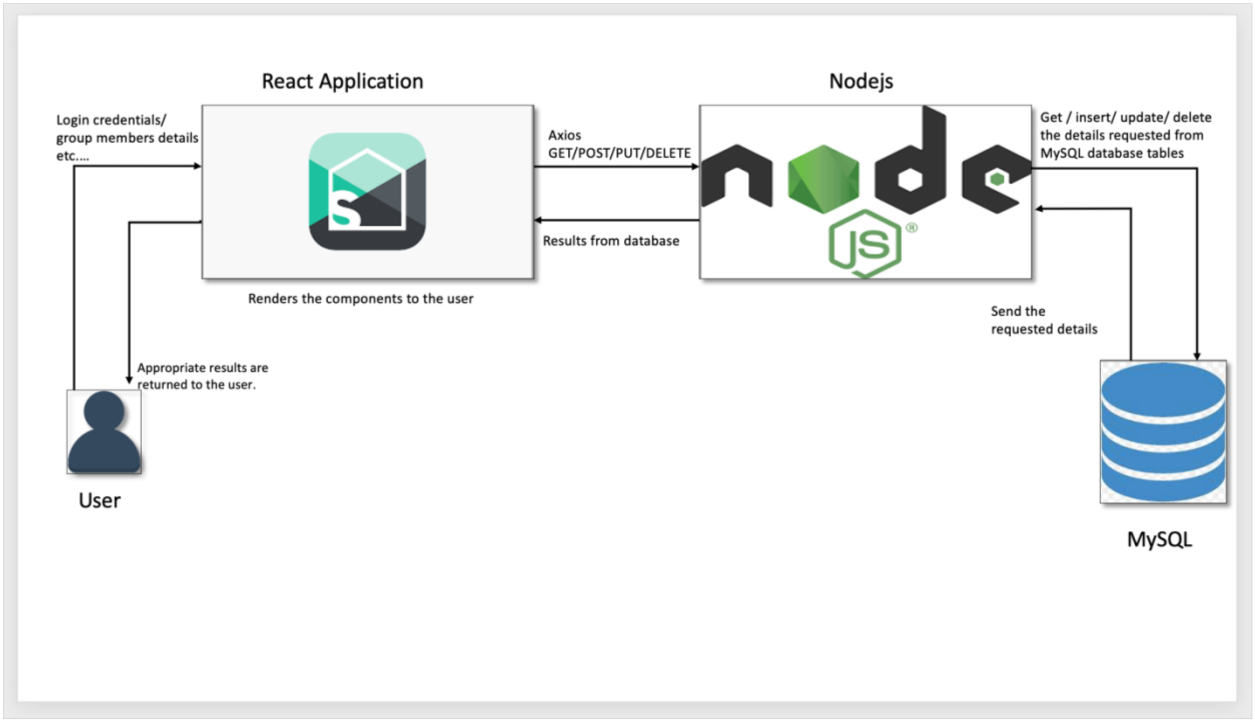
The goal of this project is to build a prototype like Splitwise using Nodejs, React and MySQL.

The application will have the following features:

- Users will be able to signup to the application by providing their full name, email address and password.
- Users then will be able to login to the application.
- Users can update their account details.
- Users will have access to a dashboard page as soon they are logged in which provides users with a summary of "who owes who" in total and across the groups.
- Users can create a group where they can add a group member by selecting from a list of dropdown members who are registered users.
- Users will be sending an invitation to other users to join the group.
- Users can either accept or deny the invitation. If they accept, they become part of the group.
- "My Groups" page is an addition to the original application where it displays the list of groups, they are part of and the list of groups they are invited to. They can navigate to the groups they are part of or create a new group. They can also accept or reject an invitation.
- Once the invite is accepted the users will be navigated to the group page which displays the list of transaction in the group along with the summary of "who owes who" in the group.
- Multiple users will be able to add a bill that gets split equally.
- Recent activity page will give you all the details of the transactions for that user.

- From dashboard the users can settle up the balances between them.
- Users will be able to log out of the application.

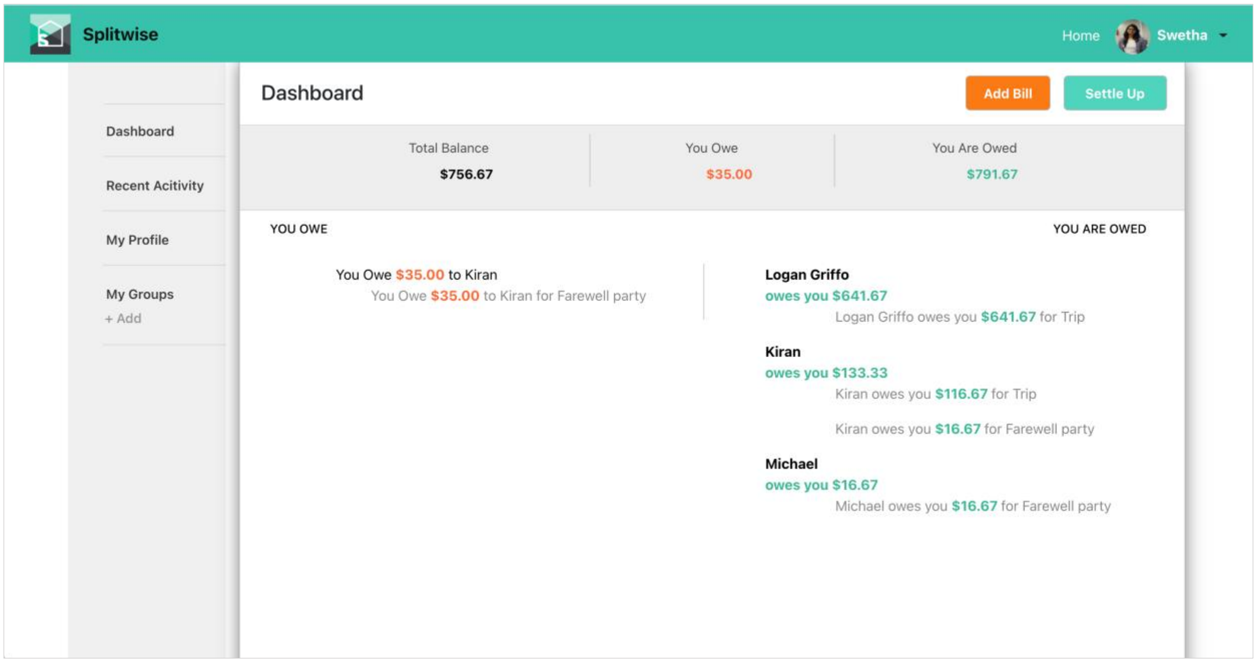
System Design


React Application
Nodejs
Login credentials/ group members details etc....
Axios GET/POST/PUT/DELETE
Results from database
Get / insert/ update/ delete the details requested from MySQL database tables
Renders the components to the user
Send the requested details
Appropriate results are returned to the user.
User
MySQL

We are using React, Nodejs and MySQL database.

Pages in the application are as follows:

Dashboard Page



Total Balance	You Owe	You Are Owed
$756.67	$35.00	$791.67

YOU OWE

You Owe $35.00 to Kiran
You Owe $35.00 to Kiran for Farewell party

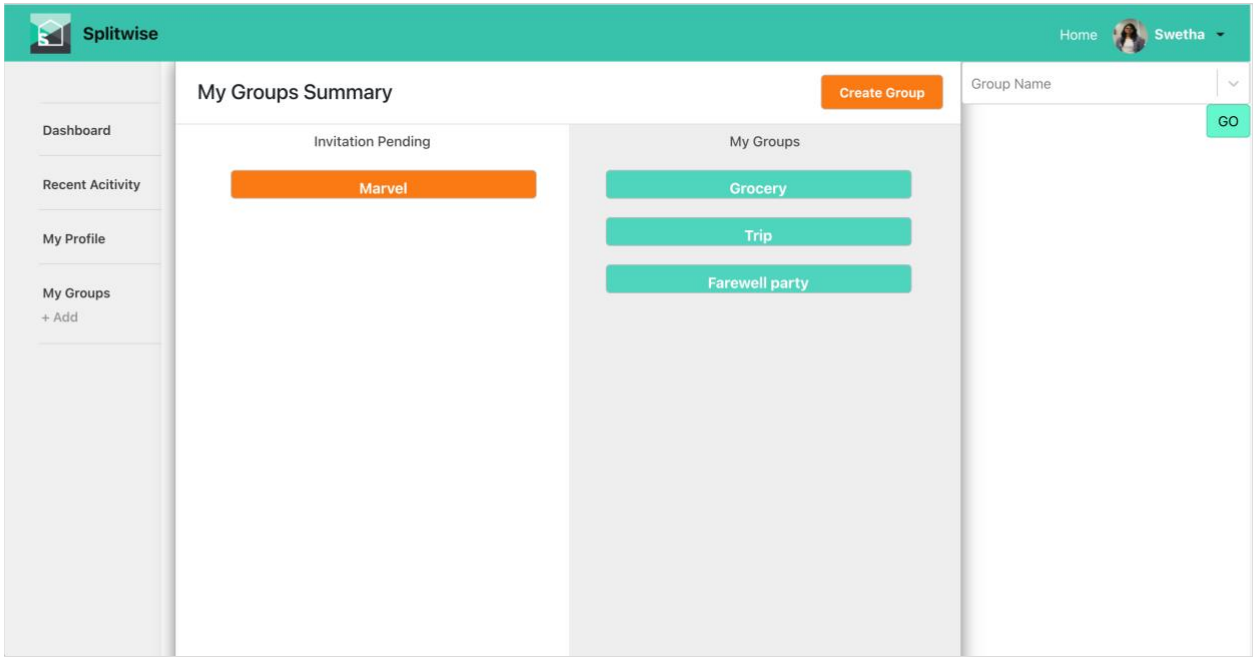
YOU ARE OWED

Logan Griffo
owes you $641.67
Logan Griffo owes you $641.67 for Trip

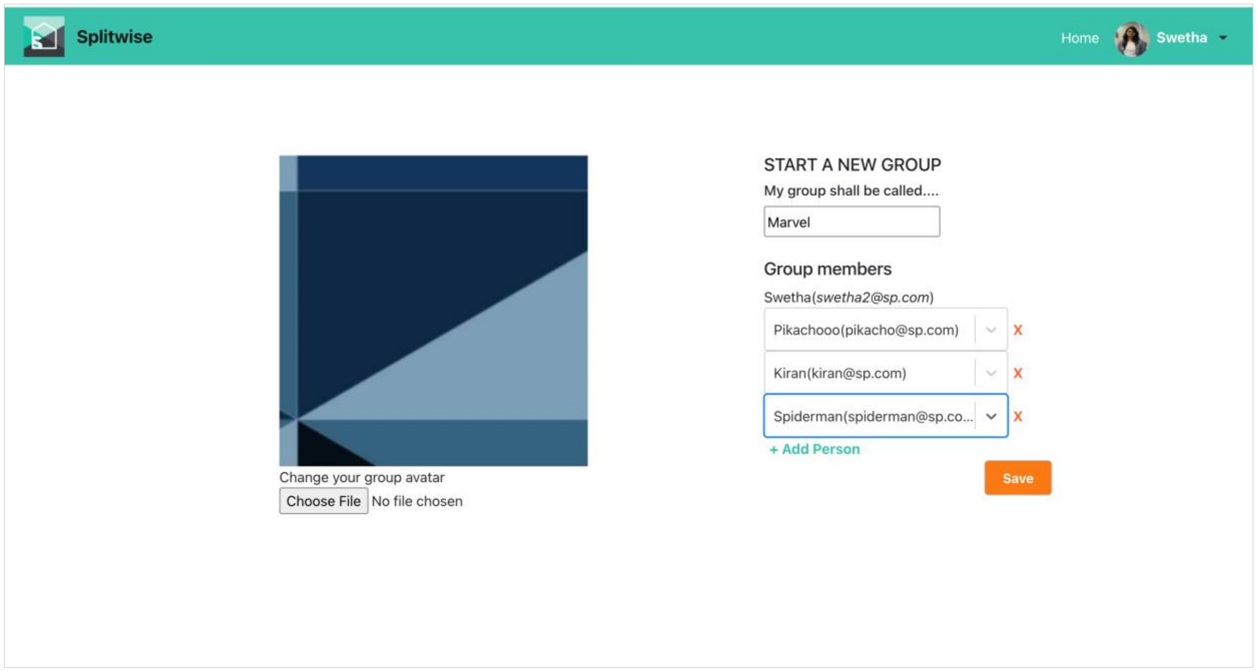
Kiran
owes you $133.33
Kiran owes you $116.67 for Trip
Kiran owes you $16.67 for Farewell party

Michael
owes you $16.67
Michael owes you $16.67 for Farewell party

My Groups



Create group

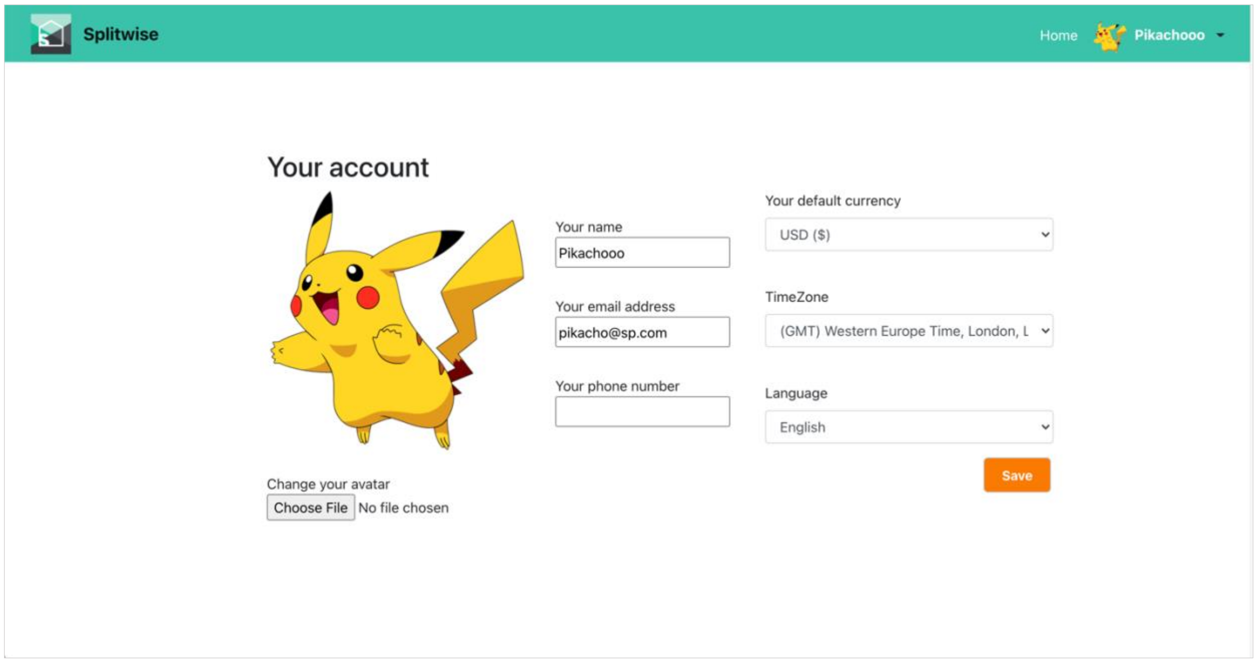


START A NEW GROUP
My group shall be called....
Marvel

Group members
Swetha(*swetha2@sp.com*)
Pikachooo(pikacho@sp.com) X
Kiran(kiran@sp.com) X
Spiderman(spiderman@sp.co... X
+ Add Person
Save

Change your group avatar
Choose File No file chosen

Update profile



Your account

Your name
Pikachooo

Your email address
pikacho@sp.com

Your phone number

Your default currency
USD ($)

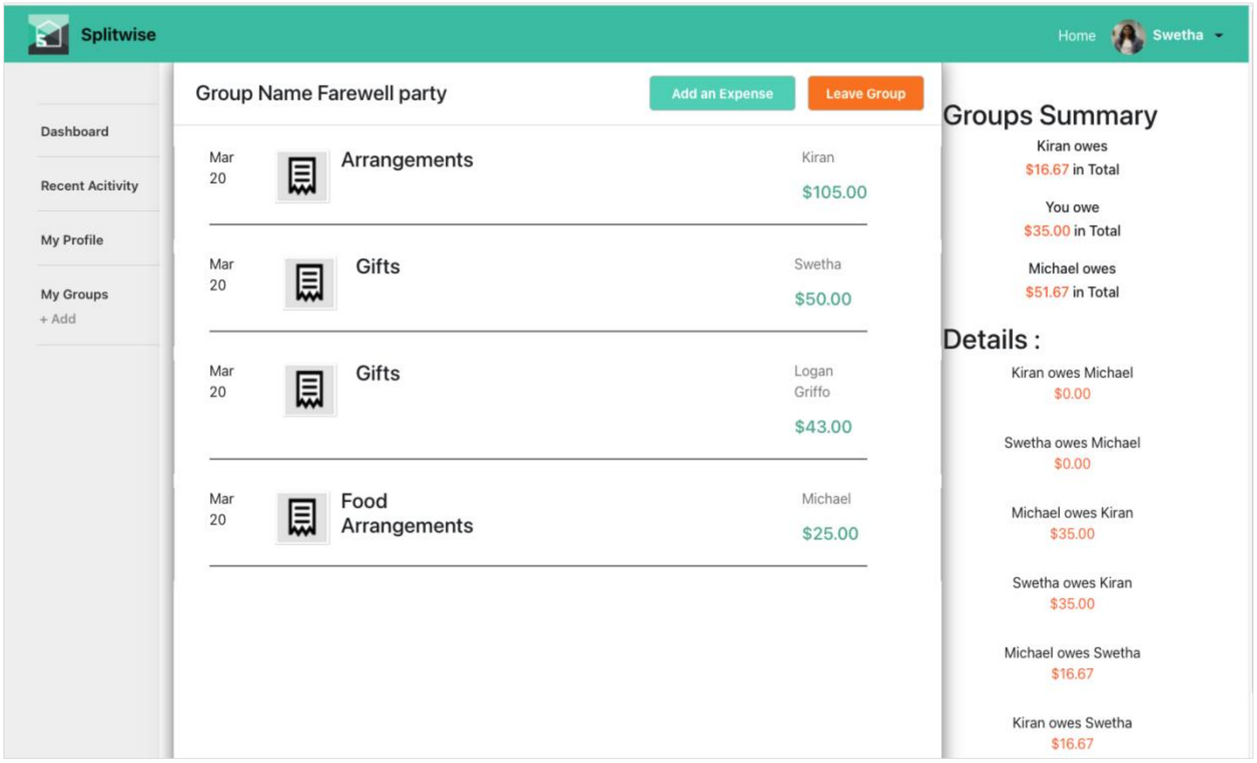
TimeZone
(GMT) Western Europe Time, London, L

Language
English

Save

Change your avatar
Choose File No file chosen

Group page



Group Name Farewell party

Mar 20	Arrangements	Kiran	$105.00
Mar 20	Gifts	Swetha	$50.00
Mar 20	Gifts	Logan Griffo	$43.00
Mar 20	Food Arrangements	Michael	$25.00

Groups Summary

Kiran owes
$16.67 in Total

You owe
$35.00 in Total

Michael owes
$51.67 in Total

Details :

Kiran owes Michael
$0.00

Swetha owes Michael
$0.00

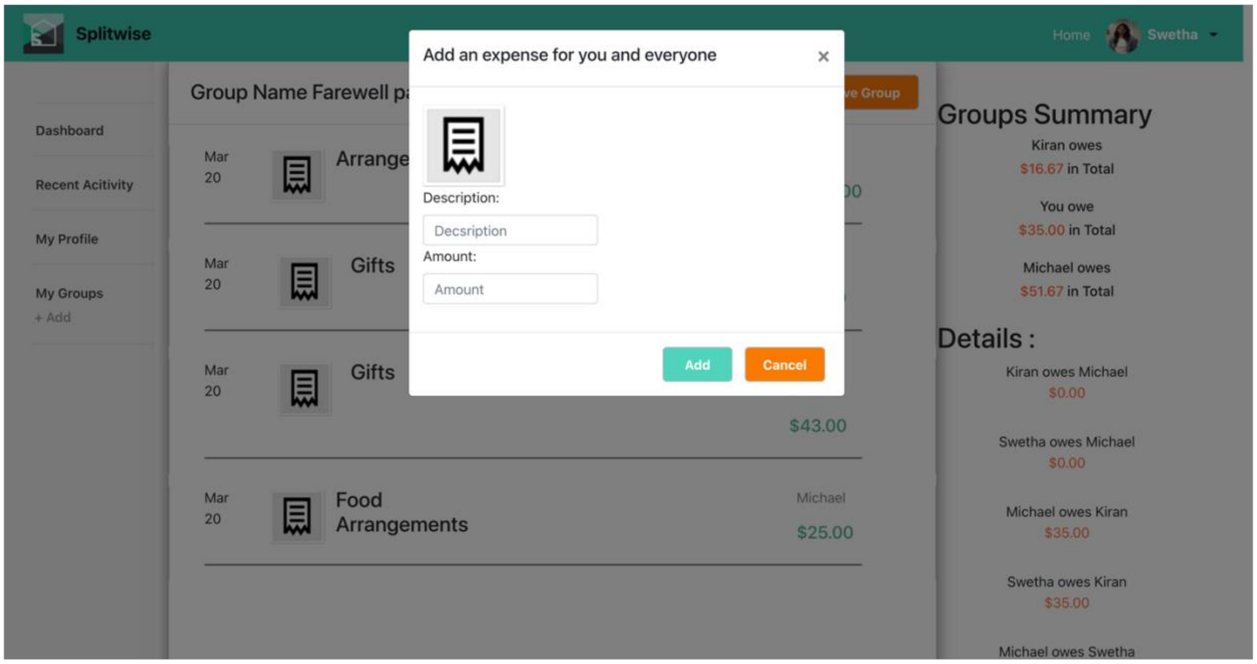
Michael owes Kiran
$35.00

Swetha owes Kiran
$35.00

Michael owes Swetha
$16.67

Kiran owes Swetha
$16.67

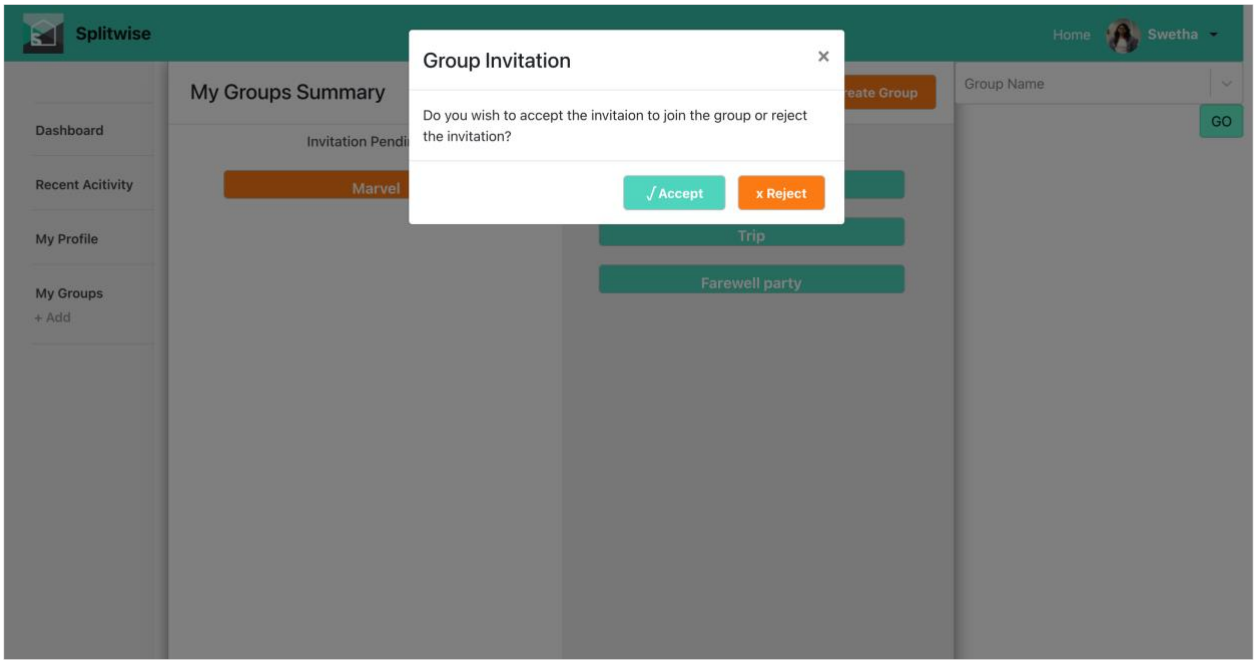
Add an expense



Add an expense for you and everyone

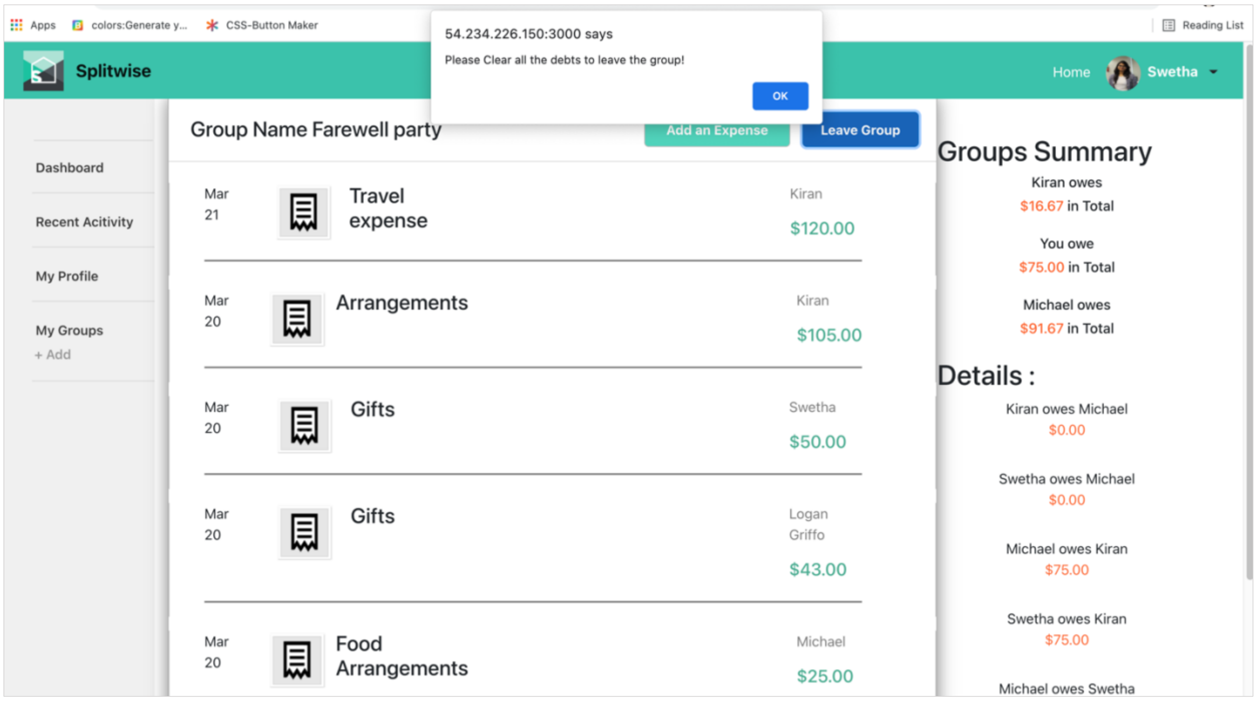
Description:
Decsription
Amount:
Amount

Accept/deny invitation



Do you wish to accept the invitaion to join the group or reject the invitation?

Leave Group



54.234.226.150:3000 says
Please Clear all the debts to leave the group!

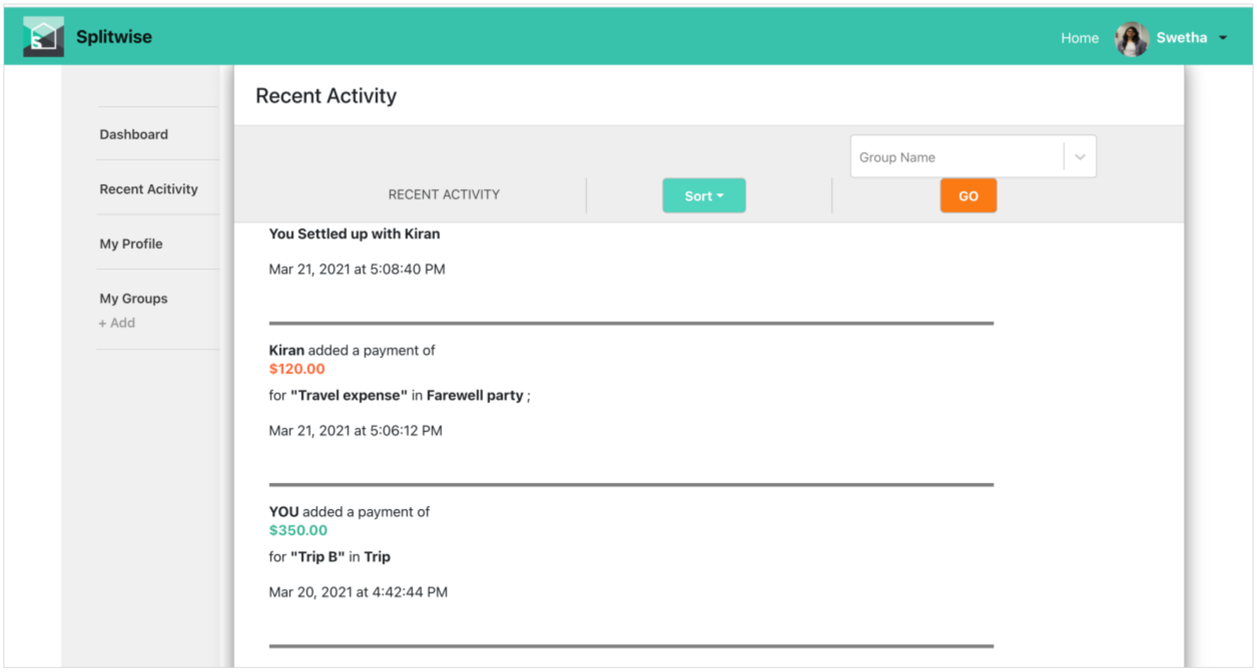
Group Name Farewell party

Mar 21	Travel expense	Kiran	$120.00
Mar 20	Arrangements	Kiran	$105.00
Mar 20	Gifts	Swetha	$50.00
Mar 20	Gifts	Logan Griffo	$43.00
Mar 20	Food Arrangements	Michael	$25.00

Groups Summary
Kiran owes
$16.67 in Total
You owe
$75.00 in Total
Michael owes
$91.67 in Total

Details :
Kiran owes Michael
$0.00
Swetha owes Michael
$0.00
Michael owes Kiran
$75.00
Swetha owes Kiran
$75.00
Michael owes Swetha

Recent activity



Recent Activity

RECENT ACTIVITY

You Settled up with Kiran

Mar 21, 2021 at 5:08:40 PM

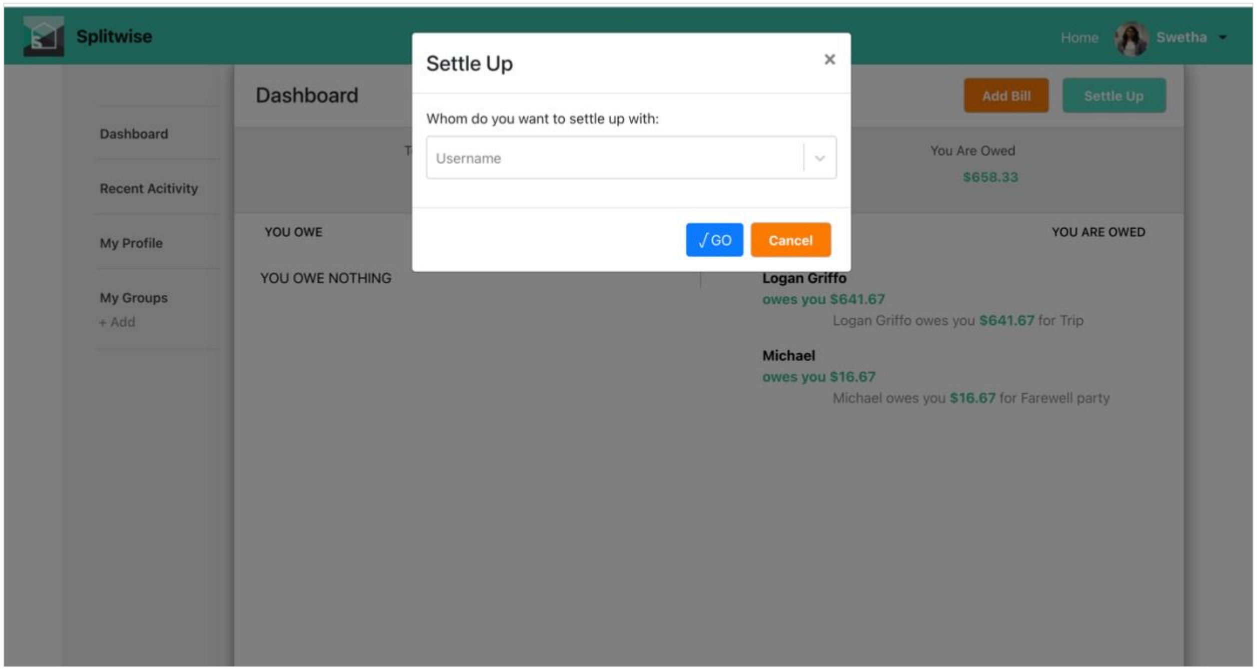
Kiran added a payment of
$120.00
for **"Travel expense"** in **Farewell party** ;

Mar 21, 2021 at 5:06:12 PM

YOU added a payment of
$350.00
for **"Trip B"** in **Trip**

Mar 20, 2021 at 4:42:44 PM

Settle up



Settle Up

Whom do you want to settle up with:

Dashboard

You Are Owed
$658.33

YOU OWE

YOU OWE NOTHING

YOU ARE OWED

Logan Griffo
owes you $641.67
Logan Griffo owes you **$641.67** for Trip

Michael
owes you $16.67
Michael owes you **$16.67** for Farewell party

Testing

Backend services were tested using Jmeter and Mocha.

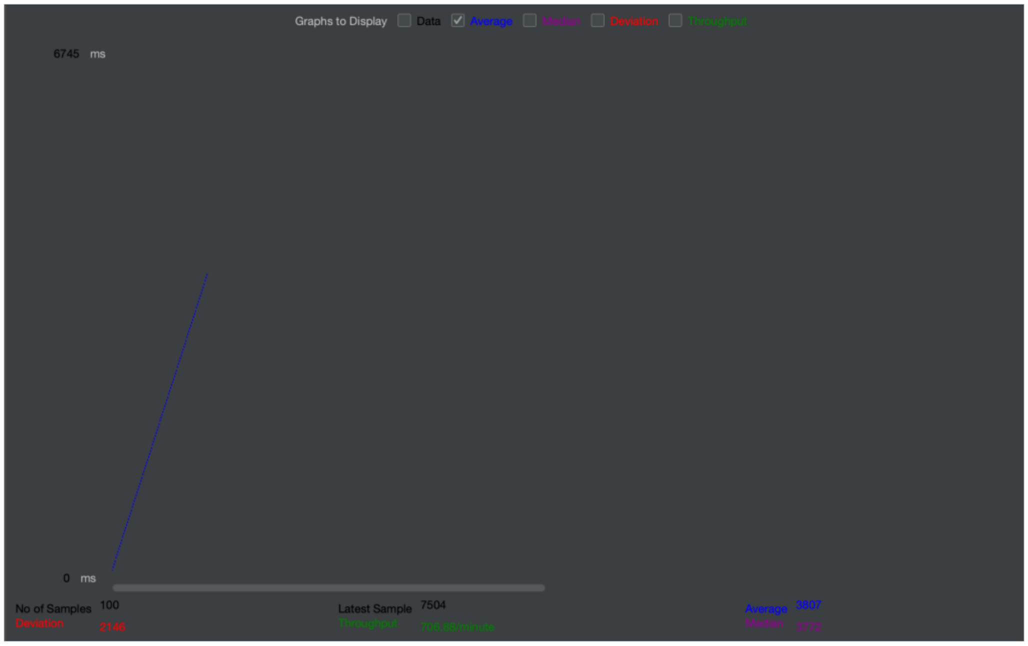
Below are the test results.

Jmeter Testing:

Without connection pooling

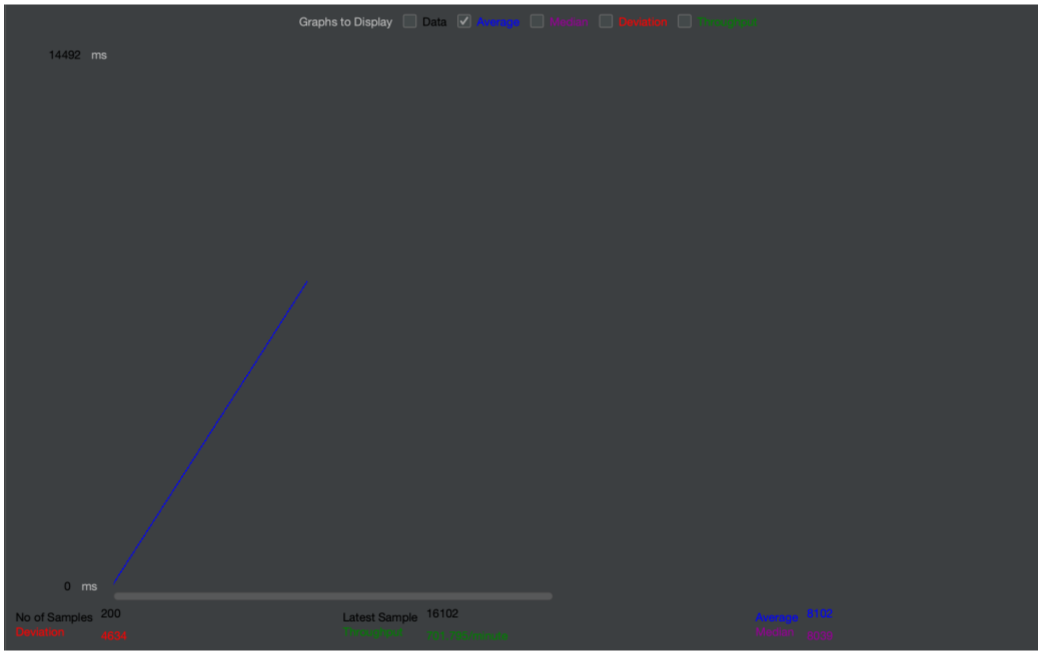
Post request for login

» 100 Concurrent Users:



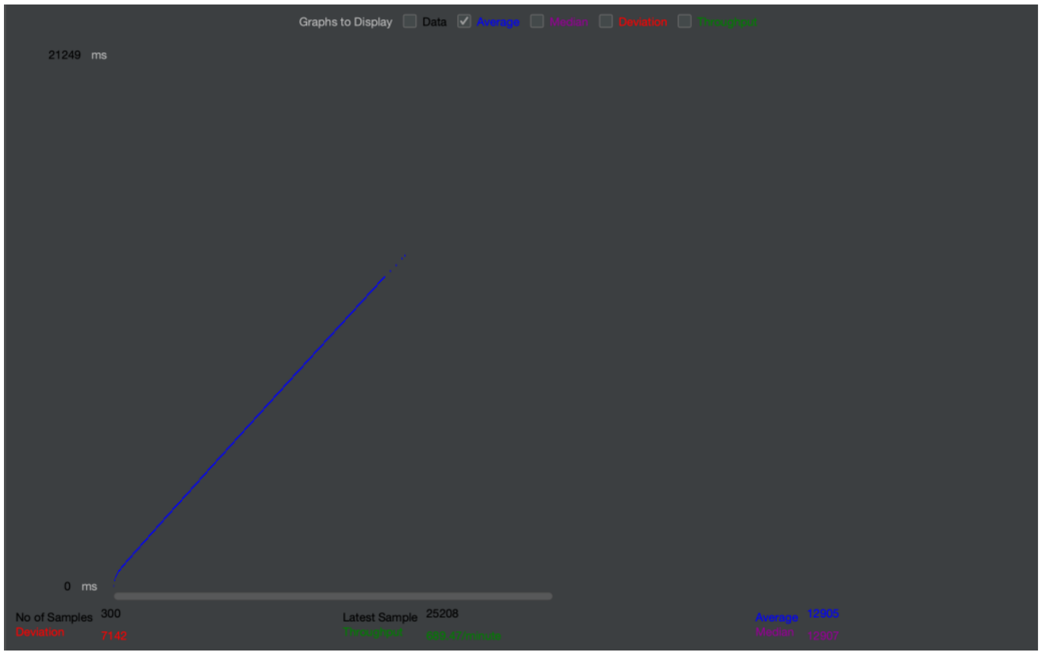
No of Samples 100
Deviation 2146
Latest Sample 7504
Throughput 706.88/minute
Average 3807
Median 3772

» 200 Concurrent Users:



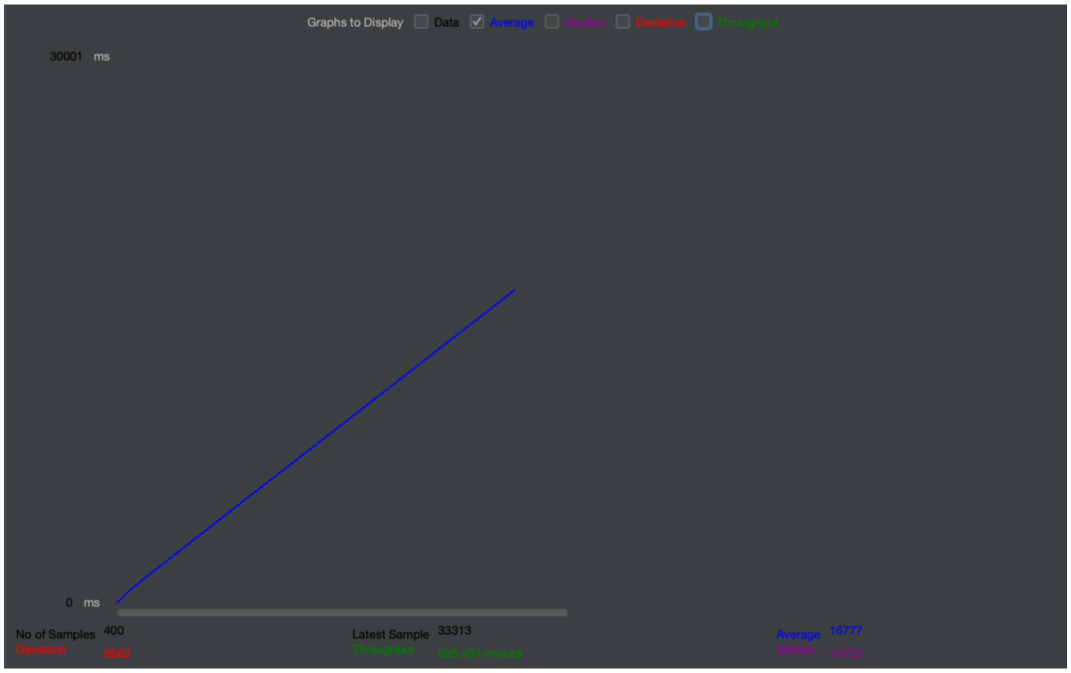
No of Samples 200
Deviation 4634
Latest Sample 16102
Throughput 701.795/minute
Average 8102
Median 8039

» 300 Concurrent Users:



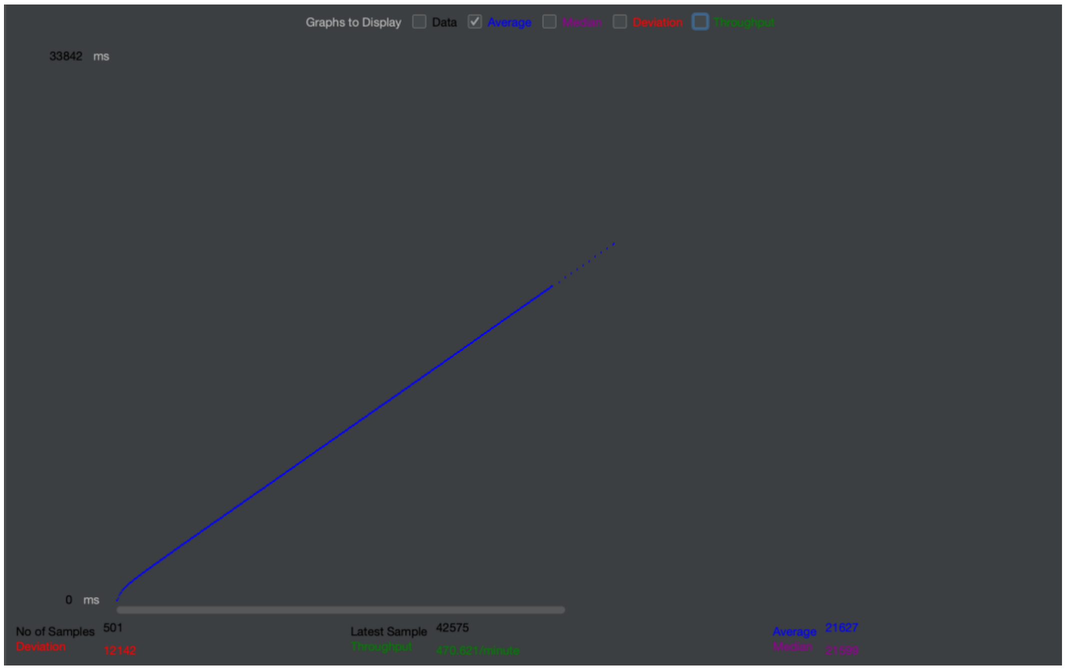
No of Samples 300
Deviation 7142
Latest Sample 25208
Throughput 689.47/minute
Average 12905
Median 12907

» 400 Concurrent Users:



```
No of Samples  400        Latest Sample  33313              Average  16777
Deviation      9562       Throughput     695.451/minute     Median   16722
```

» 500 Concurrent Users:



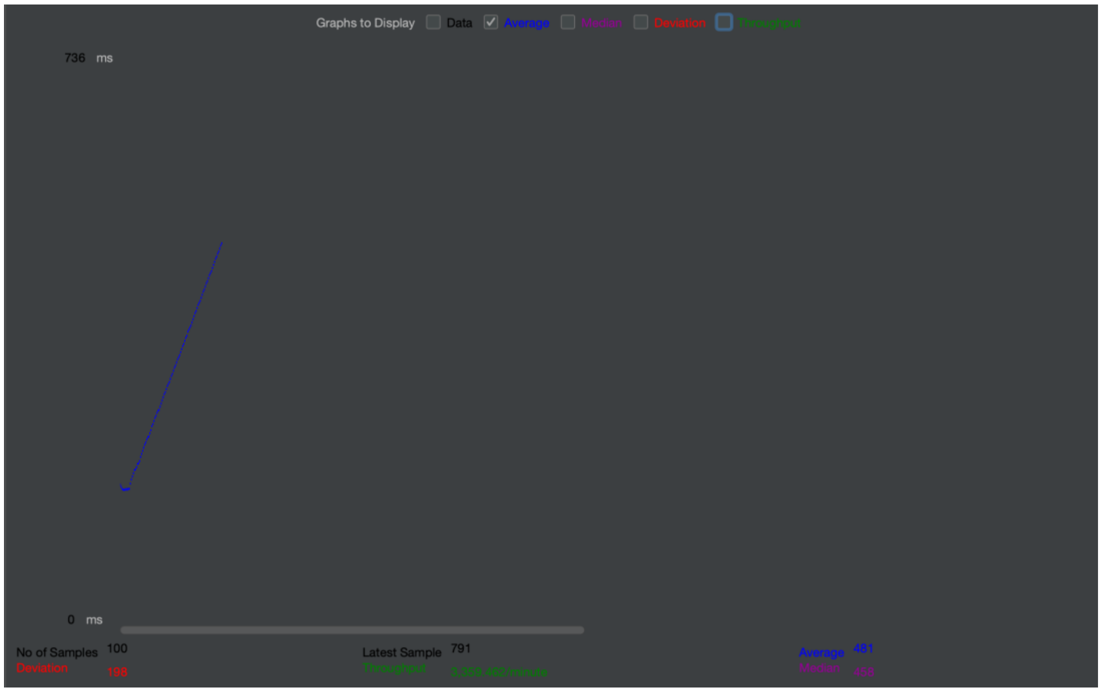
```
No of Samples  501        Latest Sample  42575              Average  21627
Deviation      12142      Throughput     470.621/minute     Median   21599
```

Without connection pooling, we see that the average time taken by 100, 200, 300, 400, 500 concurrent users to execute the GET request on getuserdetails API is 3807ms, 8102ms, 12905ms, 16777ms, 21627ms respectively. The average time increases with the increase in number of samples.

With connection pooling

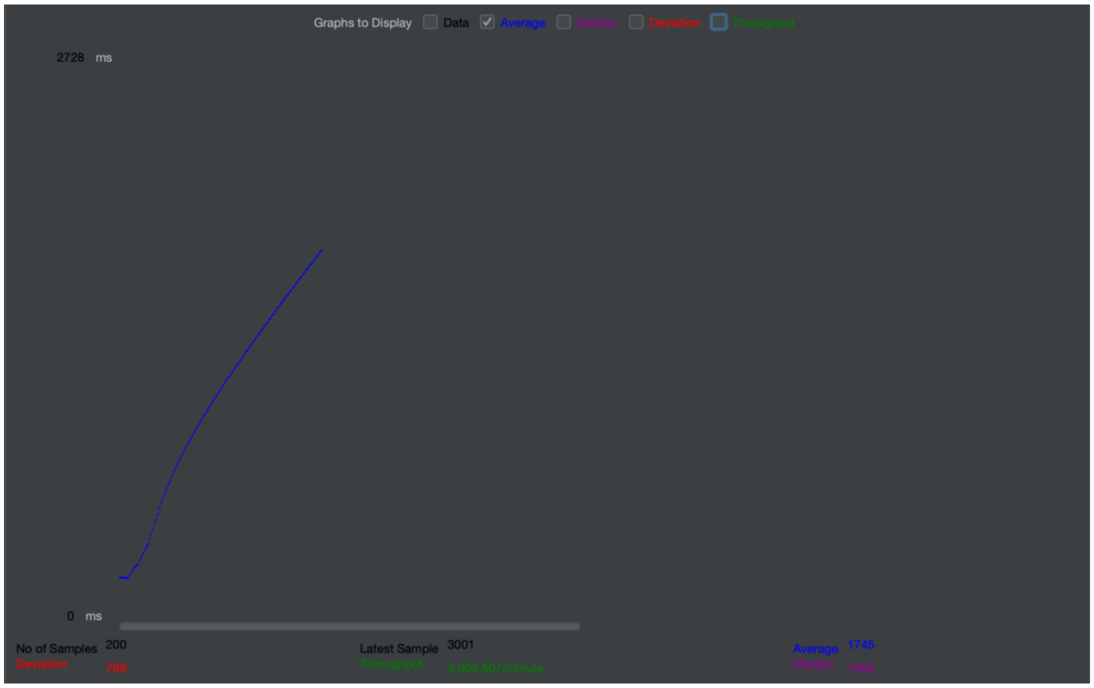
Post request for login

» 100 Concurrent Users:



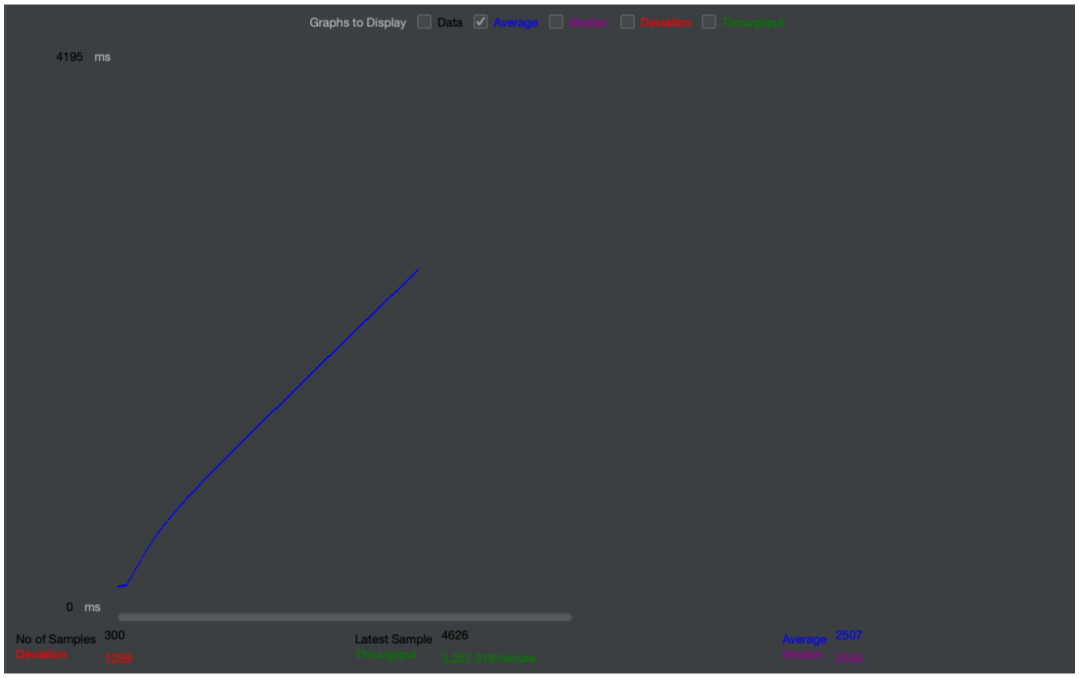
No of Samples 100
Deviation 198
Latest Sample 791
Throughput 3,359.462/minute
Average 481
Median 458

» 200 Concurrent Users:



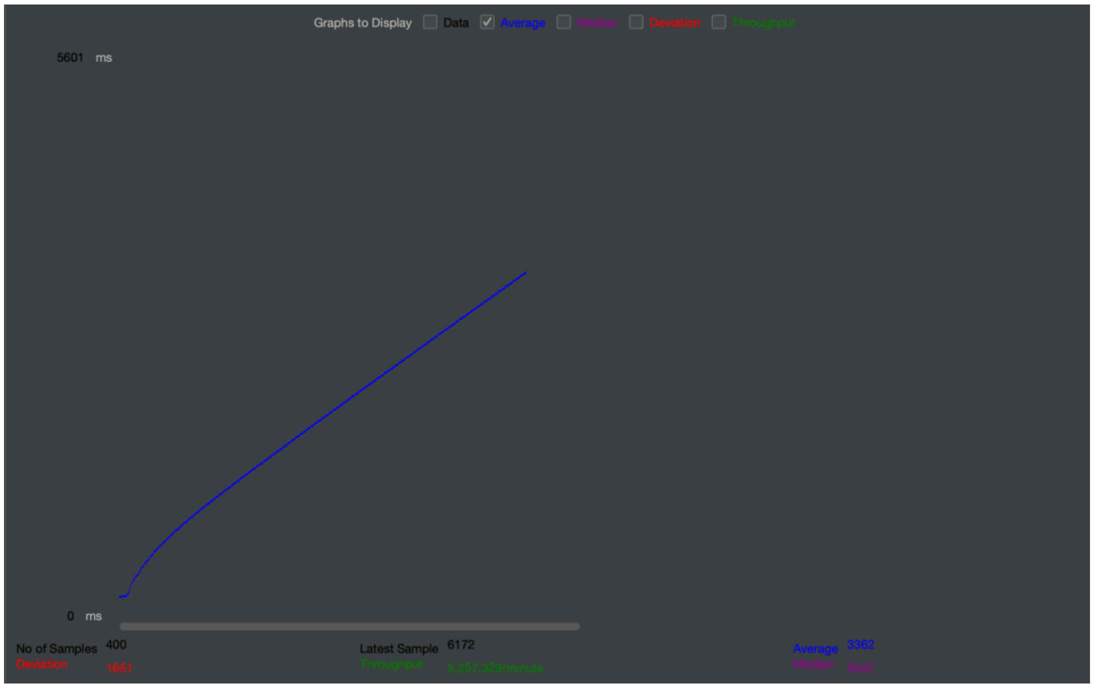
No of Samples 200
Deviation 798
Latest Sample 3001
Throughput 3,004.507/minute
Average 1745
Median 1802

» 300 Concurrent Users:



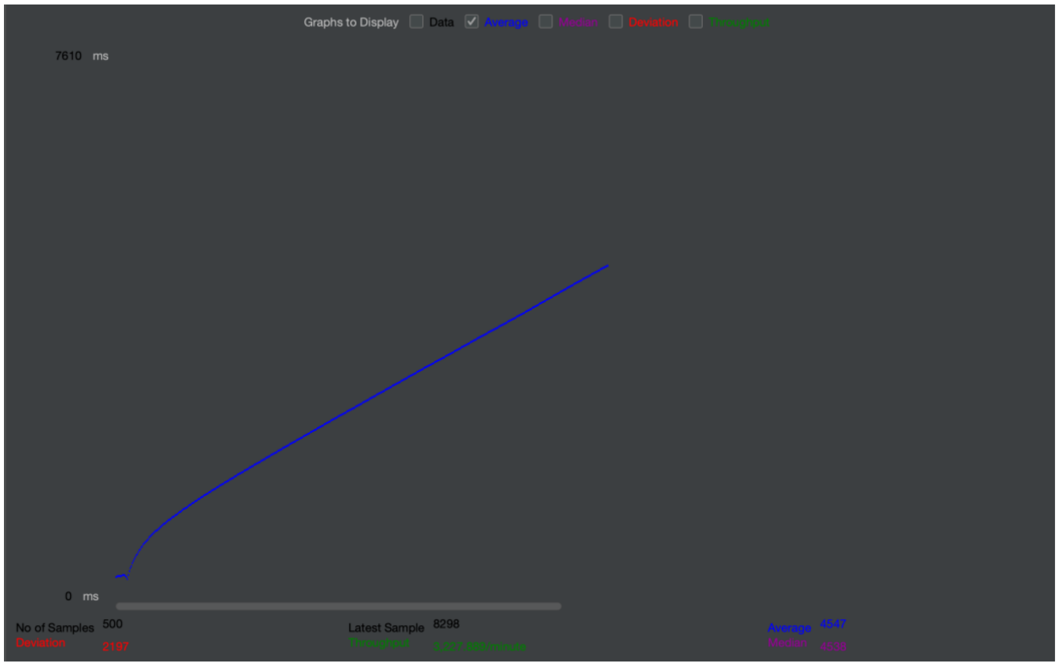
No of Samples 300
Deviation 1259
Latest Sample 4626
Throughput 3,257.919/minute
Average 2507
Median 2490

» 400 Concurrent Users:



No of Samples 400
Deviation 1651
Latest Sample 6172
Throughput 3,257.329/minute
Average 3362
Median 3347

» 500 Concurrent Users:

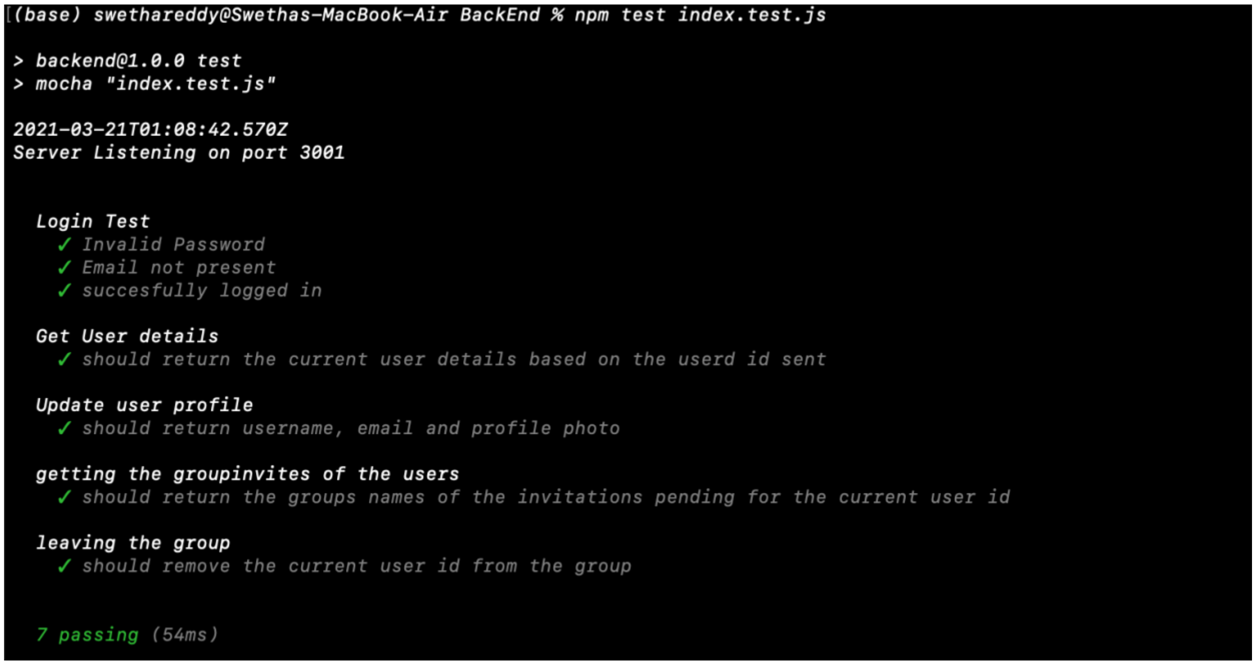


No of Samples 500
Deviation 2197
Latest Sample 8298
Throughput 3,227.889/minute
Average 4547
Median 4538

Without connection pooling, we see that the average time taken by 100, 200, 300, 400, 500 concurrent requests to execute the GET request on getuserdetails is 481ms, 1745ms, 2507ms, 3362ms, 4547ms respectively. The average time increases with the increase in number of samples but do not drastically increase as in case of "without connection pooling". The connection pool gives a pool of connections to be reused thus reducing the overload of creating new connection for each and every request.

Mocha Testing:

Mocha testing is used to test the backend services.



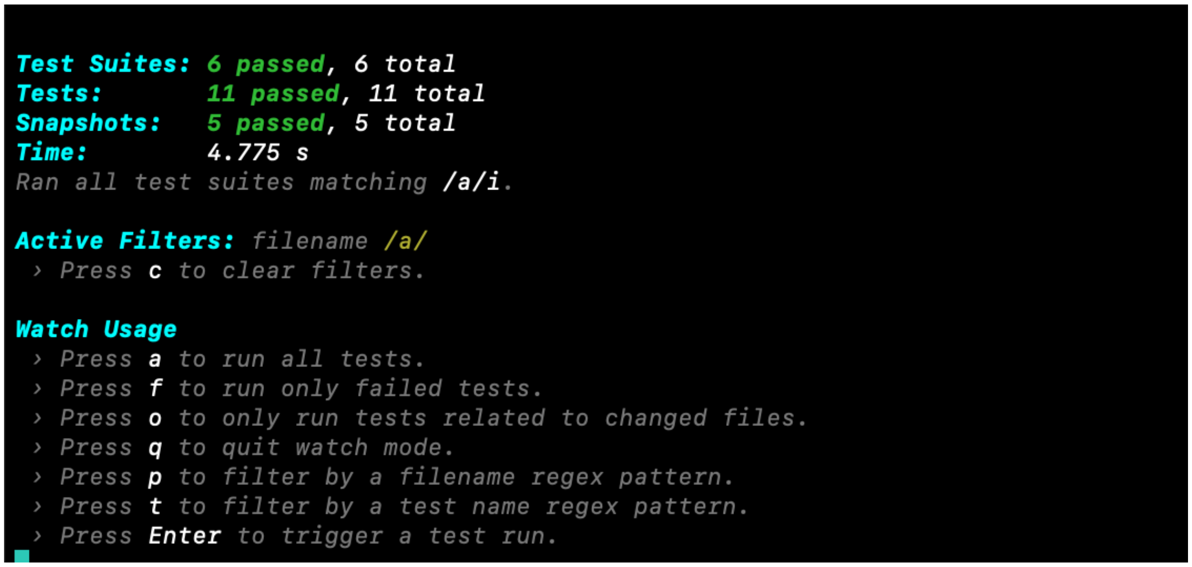
```
[(base) swethareddy@Swethas-MacBook-Air BackEnd % npm test index.test.js

> backend@1.0.0 test
> mocha "index.test.js"

2021-03-21T01:08:42.570Z
Server Listening on port 3001


  Login Test
    ✓ Invalid Password
    ✓ Email not present
    ✓ succesfully logged in

  Get User details
    ✓ should return the current user details based on the userd id sent

  Update user profile
    ✓ should return username, email and profile photo

  getting the groupinvites of the users
    ✓ should return the groups names of the invitations pending for the current user id

  leaving the group
    ✓ should remove the current user id from the group


  7 passing (54ms)
```

Backend services were tested for /POST login, /POST updating user details in MyProfile page, /POST leave group functionalities along with /GET user details and /GET group invites for a user. The functionalities were tested, and test cases passed for all 5 API calls.

- Login Test checked for invalid password, email not present and successful login for the /POST login API.
- Get User Details returned the details to be displayed on the profile page for the /GET API.
- Update User Profile updates the details provide by the user in MySQl database.
- To display the list of groups a user is part of /getgroupinvites API was tested.
- /POST leave group request was tested when users exit the group.

Frontend services were tested using React testing library.

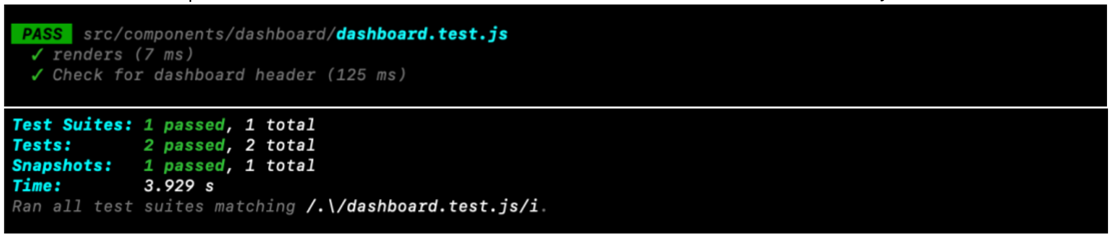
React testing library is a light weight library for testing the react components and ensuring maintainable components throughout the life cycle of development.



```
Test Suites: 6 passed, 6 total
Tests:       11 passed, 11 total
Snapshots:   5 passed, 5 total
Time:        4.775 s
Ran all test suites matching /a/i.

Active Filters: filename /a/
 › Press c to clear filters.

Watch Usage
 › Press a to run all tests.
 › Press f to run only failed tests.
 › Press o to only run tests related to changed files.
 › Press q to quit watch mode.
 › Press p to filter by a filename regex pattern.
 › Press t to filter by a test name regex pattern.
 › Press Enter to trigger a test run.
```

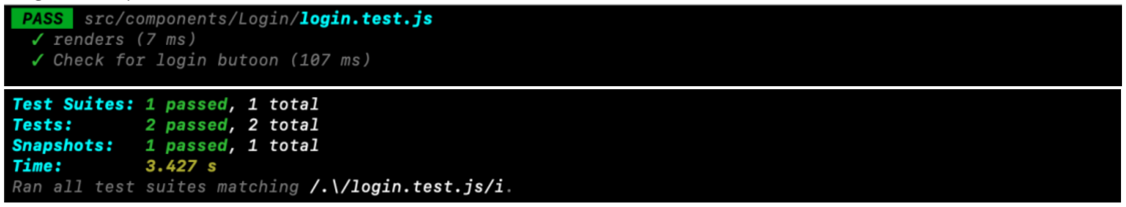
Below are the test results for 5 components:

» Dashboard component: Checked if the dashboard can render and is rendered correctly.



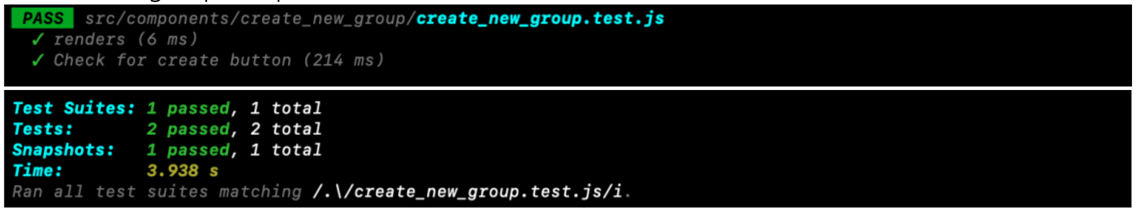
```
 PASS  src/components/dashboard/dashboard.test.js
  ✓ renders (7 ms)
  ✓ Check for dashboard header (125 ms)

Test Suites: 1 passed, 1 total
Tests:       2 passed, 2 total
Snapshots:   1 passed, 1 total
Time:        3.929 s
Ran all test suites matching /.\/dashboard.test.js/i.
```

» Login component: Checked if the login can render correctly.



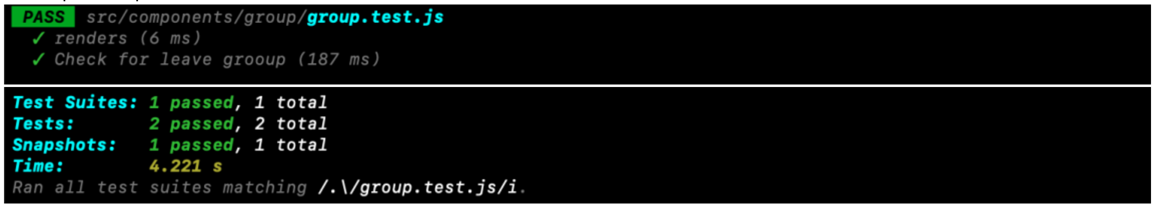
```
 PASS  src/components/Login/login.test.js
  ✓ renders (7 ms)
  ✓ Check for login butoon (107 ms)

Test Suites: 1 passed, 1 total
Tests:       2 passed, 2 total
Snapshots:   1 passed, 1 total
Time:        3.427 s
Ran all test suites matching /.\/login.test.js/i.
```

» Create_new_group component:



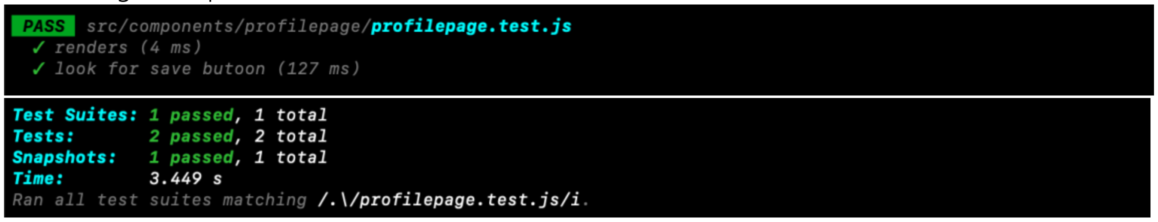
```
 PASS  src/components/create_new_group/create_new_group.test.js
  ✓ renders (6 ms)
  ✓ Check for create button (214 ms)

Test Suites: 1 passed, 1 total
Tests:       2 passed, 2 total
Snapshots:   1 passed, 1 total
Time:        3.938 s
Ran all test suites matching /.\/create_new_group.test.js/i.
```

» Group component:



```
 PASS  src/components/group/group.test.js
  ✓ renders (6 ms)
  ✓ Check for leave grooup (187 ms)

Test Suites: 1 passed, 1 total
Tests:       2 passed, 2 total
Snapshots:   1 passed, 1 total
Time:        4.221 s
Ran all test suites matching /.\/group.test.js/i.
```

» Profile_Page Component:

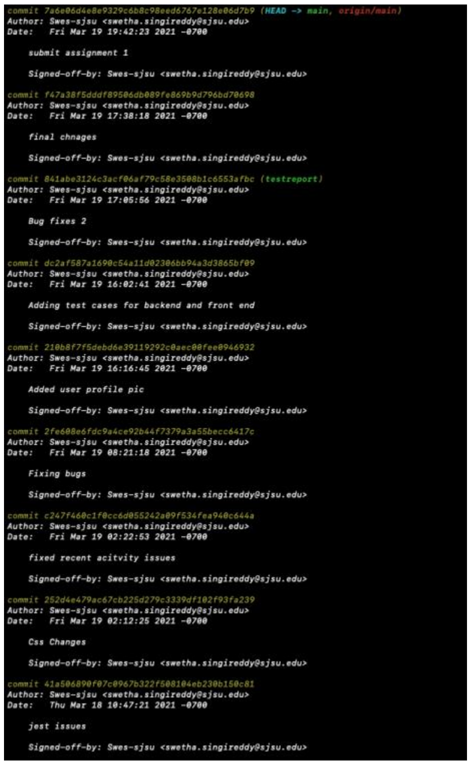


```
 PASS  src/components/profilepage/profilepage.test.js
  ✓ renders (4 ms)
  ✓ look for save butoon (127 ms)

Test Suites: 1 passed, 1 total
Tests:       2 passed, 2 total
Snapshots:   1 passed, 1 total
Time:        3.449 s
Ran all test suites matching /.\/profilepage.test.js/i.
```

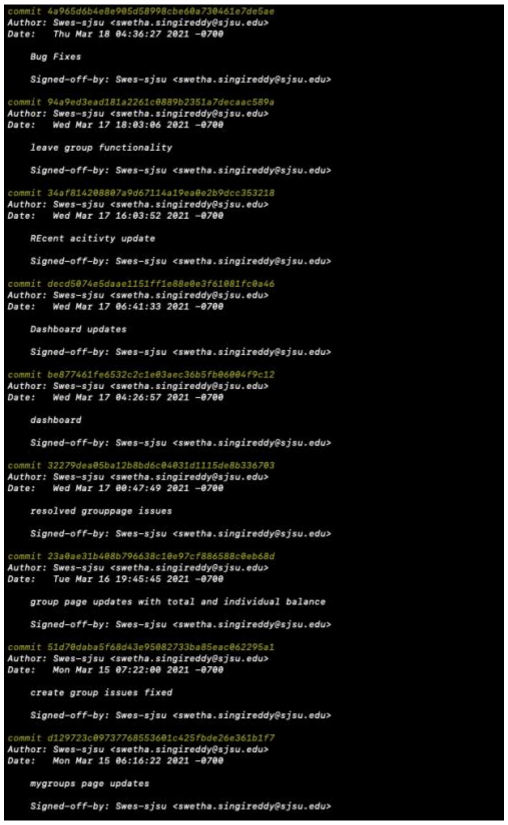
Current Limitations of the Application

- Currently Add a bill in dashboard in not implemented.
- The bills are split equally among the users as of now.

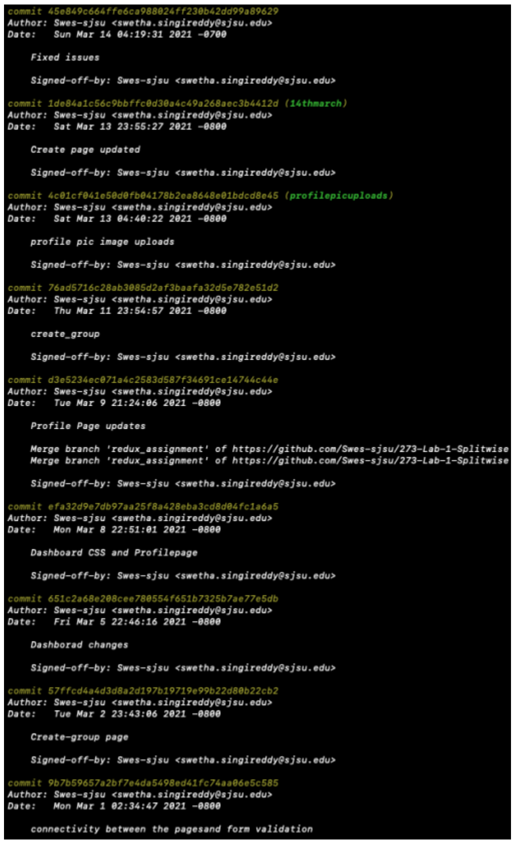
Git commit history



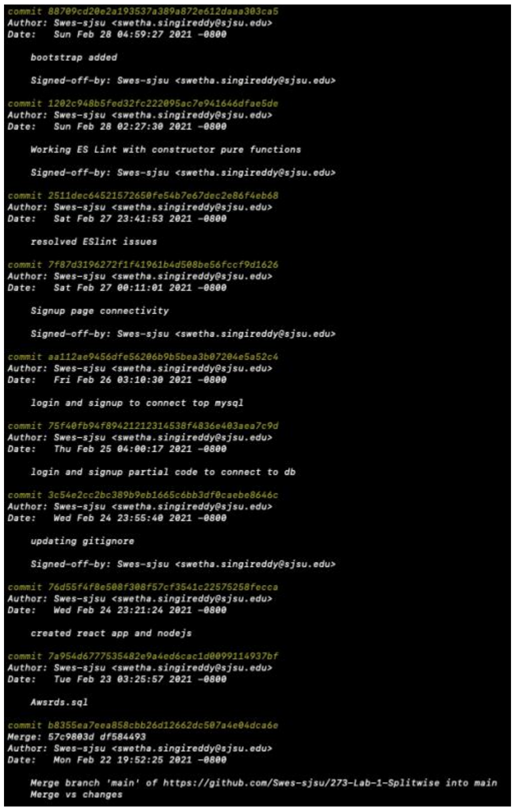
```
commit 7a6e06d4e8e9329c6b8c98eed6767e128e06d7b9 (HEAD -> main, origin/main)
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Fri Mar 19 19:42:23 2021 -0700

    submit assignment 1

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit f47a38f5dddf89506db089fe869b9d796bd70698
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Fri Mar 19 17:38:18 2021 -0700

    final chnages

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit 841abe3124c3acf06af79c58e3508b1c6553afbc (testreport)
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Fri Mar 19 17:05:56 2021 -0700

    Bug fixes 2

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit dc2af587a1690c54a11d02306bb94a3d3865bf09
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Fri Mar 19 16:02:41 2021 -0700

    Adding test cases for backend and front end

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit 210b8f7f5debd6e39119292c0aec00fee0946932
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Fri Mar 19 16:16:45 2021 -0700

    Added user profile pic

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit 2fe608e6fdc9a4ce92b44f7379a3a55becc6417c
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Fri Mar 19 08:21:18 2021 -0700

    Fixing bugs

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit c247f460c1f0cc6d055242a09f534fea940c644a
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Fri Mar 19 02:22:53 2021 -0700

    fixed recent acitvity issues

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit 252d4e479ac67cb225d279c3339df102f93fa239
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Fri Mar 19 02:12:25 2021 -0700

    Css Changes

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit 41a506890f07c0967b322f508104eb230b150c81
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Thu Mar 18 10:47:21 2021 -0700

    jest issues

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>
```



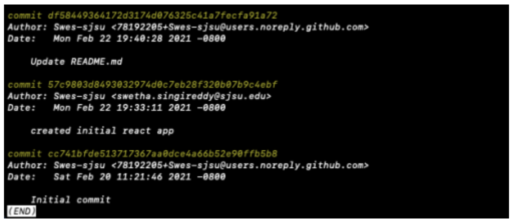
```
commit 4a965d6b4e8e905d58998cbe60a730461e7de5ae
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Thu Mar 18 04:36:27 2021 -0700

    Bug Fixes

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit 94a9ed3ead181a2261c0889b2351a7decaac589a
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Wed Mar 17 18:03:06 2021 -0700

    leave group functionality

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit 34af814208807a9d67114a19ea0e2b9dcc353218
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Wed Mar 17 16:03:52 2021 -0700

    REcent acitivty update

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit decd5074e5daae1151ff1e88e0e3f61081fc0a46
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Wed Mar 17 06:41:33 2021 -0700

    Dashboard updates

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit be877461fe6532c2c1e03aec36b5fb06004f9c12
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Wed Mar 17 04:26:57 2021 -0700

    dashboard

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit 32279dea05ba12b8bd6c04031d1115de8b336703
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Wed Mar 17 00:47:49 2021 -0700

    resolved grouppage issues

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit 23a0ae31b408b796638c10e97cf886588c0eb68d
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Tue Mar 16 19:45:45 2021 -0700

    group page updates with total and individual balance

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit 51d70daba5f68d43e95082733ba85eac062295a1
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Mon Mar 15 07:22:00 2021 -0700

    create group issues fixed

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit d129723c09737768553601c425fbde26e361b1f7
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Mon Mar 15 06:16:22 2021 -0700

    mygroups page updates

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>
```



```
commit 45e849c664ffe6ca988024ff230b42dd99a89629
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Sun Mar 14 04:19:31 2021 -0700

    Fixed issues

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit 1de84a1c56c9bbffc0d30a4c49a268aec3b4412d (14thmarch)
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Sat Mar 13 23:55:27 2021 -0800

    Create page updated

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit 4c01cf041e50d0fb04178b2ea8648e01bdcd8e45 (profilepicuploads)
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Sat Mar 13 04:40:22 2021 -0800

    profile pic image uploads

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit 76ad5716c28ab3085d2af3baafa32d5e782e51d2
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Thu Mar 11 23:54:57 2021 -0800

    create_group

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit d3e5234ec071a4c2583d587f34691ce14744c44e
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Tue Mar 9 21:24:06 2021 -0800

    Profile Page updates

    Merge branch 'redux_assignment' of https://github.com/Swes-sjsu/273-Lab-1-Splitwise
    Merge branch 'redux_assignment' of https://github.com/Swes-sjsu/273-Lab-1-Splitwise

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit efa32d9e7db97aa25f8a428eba3cd8d04fc1a6a5
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Mon Mar 8 22:51:01 2021 -0800

    Dashboard CSS and Profilepage

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit 651c2a68e208cee780554f651b7325b7ae77e5db
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Fri Mar 5 22:46:16 2021 -0800

    Dashborad changes

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit 57ffcd4a4d3d8a2d197b19719e99b22d80b22cb2
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Tue Mar 2 23:43:06 2021 -0800

    Create-group page

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit 9b7b59657a2bf7e4da5498ed41fc74aa06e5c585
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Mon Mar 1 02:34:47 2021 -0800

    connectivity between the pagesand form validation
```



```
commit 88709cd20e2a193537a389a872e612daaa303ca5
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Sun Feb 28 04:59:27 2021 -0800

    bootstrap added

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit 1202c948b5fed32fc222095ac7e941646dfae5de
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Sun Feb 28 02:27:30 2021 -0800

    Working ES Lint with constructor pure functions

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit 2511dec64521572650fe54b7e67dec2e86f4eb68
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Sat Feb 27 23:41:53 2021 -0800

    resolved ESlint issues

commit 7f87d3196272f1f41961b4d508be56fccf9d1626
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Sat Feb 27 00:11:01 2021 -0800

    Signup page connectivity

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit aa112ae9456dfe56206b9b5bea3b07204e5a52c4
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Fri Feb 26 03:10:30 2021 -0800

    login and signup to connect top mysql

commit 75f40fb94f89421212314538f4836e403aea7c9d
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Thu Feb 25 04:00:17 2021 -0800

    login and signup partial code to connect to db

commit 3c54e2cc2bc389b9eb1665c6bb3df0caebe8646c
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Wed Feb 24 23:55:40 2021 -0800

    updating gitignore

    Signed-off-by: Swes-sjsu <swetha.singireddy@sjsu.edu>

commit 76d55f4f8e508f308f57cf3541c22575258fecca
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Wed Feb 24 23:21:24 2021 -0800

    created react app and nodejs

commit 7a954d6777535482e9a4ed6cac1d0099114937bf
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Tue Feb 23 03:25:57 2021 -0800

    Awsrds.sql

commit b8355ea7eea858cbb26d12662dc507a4e04dca6e
Merge: 57c9803d df584493
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Mon Feb 22 19:52:25 2021 -0800

    Merge branch 'main' of https://github.com/Swes-sjsu/273-Lab-1-Splitwise into main
    Merge vs changes
```



```
commit df58449364172d3174d076325c41a7fecfa91a72
Author: Swes-sjsu <78192205+Swes-sjsu@users.noreply.github.com>
Date:   Mon Feb 22 19:40:28 2021 -0800

    Update README.md

commit 57c9803d8493032974d0c7eb28f320b07b9c4ebf
Author: Swes-sjsu <swetha.singireddy@sjsu.edu>
Date:   Mon Feb 22 19:33:11 2021 -0800

    created initial react app

commit cc741bfde513717367aa0dce4a66b52e90ffb5b8
Author: Swes-sjsu <78192205+Swes-sjsu@users.noreply.github.com>
Date:   Sat Feb 20 11:21:46 2021 -0800

    Initial commit
(END)
```

Questions

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.

Looking at the resultant graphs for the connections with and without pooling, we see that the average time increase drastically in case of there is no pool. As previous mentioned, without connection pooling, the average time taken by 100, 200, 300, 400, 500 concurrent requests to execute the GET request on getuserdetails is 481ms, 1745ms, 2507ms, 3362ms, 4547ms respectively. The average time increases with the increase in number of samples but do not drastically increase as in case of "without connection pooling". The connection pool gives a pool of connections to be reused thus reducing the overload of creating new connection for each and every request. The connections from the pool are reused to process the request. The connection pool will create new connections when there are no available connections in the pool until the maximum size is not reached.

Connection Pooling Algorithm:

There will be a pool of connection created to process the requests. Once a connection request comes in, it will check if there are any available connections in pool to process that request. If the connections are available, it will use the connection from the pool and return it back to the pool. If the connection is not available, the request has to either wait for connection to be available or create a new connection. New connections can only be created if no connections are available in the pool and the max size of the pool (maximum number of connections that can be created in that pool) is not reached. The size of the pool will be incremented until the maximum size of the pool is reached. If the maximum size is reached, it will wait for any of the request to be available to process it and it will retry to get the connection. If none of the connections are available within the time out period, the request will fail.

What are the ways to improve SQL Performance? List at least 3 strategies. Explain why those strategies improve the performance.

- As seen previously, one of the strategies would definitely be using connection pooling. When there are concurrent requests connection pooling will help balance the resources thus not overwhelming the database.
- Once the connection is made to the SQL Server, the table design plays an important role. Having appropriate keys, constraints and indexes on the tables reduces the time it takes for the query to execute and provide results faster. Having clustered and non-clustered indexes will help in faster retrieval of the data. Further using stored procedures makes the queries efficient by describing the set of instructions it needs to be execute in one batch. Thus, its beneficial in case of deletes of updates on the tables. Deleting or updating in batches will be faster as it need not query the table for every row to row, instead can look for a batch of rows to be deleted/updated. Writing efficient queries to return the minimal amount of information that is required is key to increase the performance.
- Further making sure that the SQL server has been provided with sufficient resources on the machine it is running on is important as the queries might get slower if the resources are not available in the machine. if the server/machine is shared but multiple applications, SQL might be starving for resources and causing timeouts.