Student Networking Application

Hesha Sheth
Master of Applied Computing
University of Windsor
shethh@uwindsor.ca

Mounika Vannawada
Master of Applied Computing
University of Windsor
vannawa@uwindsor.ca

Nizar Kadri
Master of Applied Computing
University of Windsor
kadrin@uwindsor.ca

Adaobi Aniuchi
Master of Applied Computing
University of Windsor
aniuchi@uwindsor.ca

Abstract: Effective communication is key to one's success. As we are all studying online, communication is happening virtually. Sometimes, it becomes difficult to communicate because of time barriers. So we thought of a Web application, where students can connect to their fellow students and classmates by creating their events known here as virtual tables. Students can join the tables and communicate via messaging, audio, or video call. The campus events happening very shortly are listed and are connected to the official events page link. This application was developed only for University of Windsor students. When a student faces a problem in understanding a concept, he /she shall create a Table on the discussion board about the topic, and other fellow students can join the table and help the student to understand that topic via video call, audio call, or messaging. One can also create tables to discuss their project or research on new technology. It's not limited to projects or academics discussions but can also be used to create tables about video games and have fun activities.

Keywords: Virtual Tables, Discussion Board, Campus Events, Calls, and Messaging.

I. ACKNOWLEDGEMENT

We feel that it's a great opportunity to sharpen our skills and knowledge by developing this project. It would not be possible without the support of Dr. Ziad Kobti. So, We would like to express our sincere appreciation to him for his guidance, continuous encouragement, and for helping us in deciding the technology we used in our project. Also, we have to thank all the community members of Stack Overflow for saving our time by providing us with solutions to some critical errors.

We are glad to have such a talented member like Hesha, whose expertise in design helped us to make our website look more appealing and easy to use. Mounika, whose experience in backend development helped us to achieve some critical features like a conference call and chat. Nizar, whose experience with cloud and APIs helped us to handle database, signup, and feedback related functionalities. Adaobi's deep understanding of APIs has helped us to develop critical features like authentication, dashboard, and event handling. We are immensely grateful to all the team members involved in this project. Without their inspiration, valuable suggestions, and hard work it would not have been possible to develop the project within the prescribed time.

II. INTRODUCTION

Student Networking application is a web application that will allow the University of Windsor students to communicate virtually.

A. Problem Statement

Due to COVID-19, networking between students has become highly difficult. Even though there are many free communication applications to form groups and discuss, it is causing difficulties in tracking the information and sometimes messages are left unanswered in between different messages. There exists no application that helps to keep track of the activities taking place in the university. So, there is a necessity for an application that provides students with an opportunity to network. So, they can easily share and acquire knowledge from others.

B. Motivation

With the ongoing pandemic, most of the universities are shifting to online learning because

of which students are missing out on the opportunity of creating new networks and experiencing the culture of the university.
One would say, students can exchange their social media accounts and get to know each other. BUT!

- Breaking the ice isn't that easy, especially when everyone is new to the university
- Also, it becomes awkward sometime to message someone randomly out of nowhere

To overcome issues like this, we have thought of a platform where all students can openly share their ideas and get to know people having similar thoughts.

C. Background

There are many online applications like WhatsApp, Facebook, Zoom, and Teams which have pretty much included all the features we were seeking. However, these applications were not only for the University of Windsor's Students but also for other people. Also, it is not useful to manage both social and school projects in one application.

D. Objectives

- Implement a discussion board that helps the users of the application to create a table and join existing tables to participate in the discussion or activity.
- Develop a chat feature that supports the participants to share their views if their audio or video is not functioning.
- Display an events page that lists all the upcoming events happening in and around the University of Windsor's campus.
- Develop a Login feature that allows only University of Windsor students to access the application.
- Provide feedback form for the users to address their issues.

E. Risks and Mitigation plans

Development Complexity: Shifted from Completely Node.js application to Node.js with Express.js

- The main reason to shift from PHP to Node.js was because of the additional advantages offered by Node.js compared to PHP.
- As per our research, we analyzed that Node.js is the best option for real-time applications and as our web app includes chat and video call features, we want to offer the best experience to our users without any delay or lag.
- Additionally, by using node.js, we got a consistent environment of js in both front end and back end environments, which makes the development simple.
- Disguised Signups: Restricted access through the university mail id login.
- To make our web app more secure and safe for students, we have limited access to only the University of Windsor.
- Data Confidentiality Issues: Used IBM DB2 on IBM cloud front to secure the data efficiently.
- Misconduct issues: Added privacy policies and Terms of conditions and Usage to prevent misconduct behaviour.
- Performance Issues: Maintain web application standards to prevent applications from hacking and crashing.

F. Report Breakdown Description

There are three phases for this project: Inception, Elaboration I, and Elaboration II.

1) Inception phase

In this phase, we determined the feasibility of the project considering the other alternatives. We described and understood the project scope, vision, and objectives. After identifying all the major risks of the project, we listed all the high-level use cases and developed the prototype for the project with the base features. We also developed the use case diagram for a better understanding of the project.

2) Elaboration Phase I

In this phase, we had listed all the tasks in the project management tool, which is Redmine and assigned tasks ourselves during the team meeting. We had created the templates and designed API points for the project features, which makes it easy to work as we finalized the look of the project and its core APIs.
We also finalized the architecture for developing the project and also decided on languages and tools to use. We also set the standards and guidelines for project development. Further, we worked on the database design for the features to include.
Features Developed: We have developed the login, signup, and events pages and their functionalities.

3) Elaboration phase II

In this phase, we created a complete database using the IBM Cloud DB2 service. We started developing the core features of the project, which include a discussion board to create and manage tables and Tables for a virtual meetup. And we managed to distribute the tasks using Redmine. Branching in Gitlab helped us to work independently and integrate our work.
Features Developed: Developed discussion board features such as video call, audio call, and chat. Even implemented operations to create and join the tables. We also developed the wiki pages and deployed the application on IBM Cloud.

III. RELATED WORK

There are several existing communication systems, such as Slack, Teams, and WhatsApp. This section will take a look at those applications and their features in comparison with ours.

A. Slack

Slack is a cloud-based business communication platform that provides digital workspace and information management and is used to manage productivity and improve team efficiency [1]. It is a freemium product with a free tier layer that has a 10000 message limit after which we cannot access older messages. We use the term workspace to describe the shared space that brings together all the channels and chats of a company, where all communication, file sharing, and conference calls take place.[2]

Content is organized in a Slack workspace by channels which are chat rooms dedicated to a specific topic or project. Access to content in certain channels can be limited to some team members while others can be accessed by all members. It provides file sharing, video, and audio call features as well and integrates with over 600 native and external applications for file management, analytics, project management, social tools, etc. [1] It also provides one-on-one messaging capabilities for individual conversations. It is web-based and has a desktop application as well as a Mobile App for the two popular mobile platforms - Android and iOS.

B. Teams

Teams is a business chat application created by Microsoft as part of their Office 365 offering. An organization or a team can set up a Team and within that team, there are subsections known as Channels. Channels are used to group conversations or spaces into different projects or departments. Within each Channel, there is a 'conversations' tab which is the chatroom for communication, a file tab for files shared within the channel, and a notes tab for taking notes during meetings. It also has video and audio call capabilities with an additional feature to allow dial-ins to regular phone numbers if the user is not available on Teams. Lastly, just like Slack, it has a one-on-one chat feature for individual discussions and integrates with Office 365 services for file management such as Word, Excel, PowerPoint, and OneNote. It is available on the web as well as a desktop application and a mobile application.

C. WhatsApp

WhatsApp is another similar product but is more suited for informal conversations. It is a free, cross-platform messaging and Voice over IP application owned by Facebook [3]. It provides one-on-one messaging as well as group chats that can take up to 256 members. It also provides calls and video options for both the one-on-one conversations as well as group discussions.
It is available primarily as a mobile application across Android and iOS devices and also has a web interface for users who want to access their chats from the browser.

D. Comparison of the Platforms with Our Use Case

Both Slack and Teams are business communication platforms and might not be appropriate for the use case we are trying to address in our project. The slack platform would require setting up a workspace and sharing invites to people to join the workspace. For Teams, even though the school currently has a Teams offering, creating separate channels for discussions would require admin privileges which might not easily be available to students.

WhatsApp, on the other hand, is informal, but the risk of distractions can be present when using a platform like it. There is the tendency to receive messages that are not about the discussion at hand and lose attention. Our proposed platform puts all school-related conversations in one place, and people can join as they deem fit without distractions from other real-life events.

IV. APPROACH

A. Assumptions

We made some assumptions in the course of the project. One is that the users of the platform will only be University of Windsor students, and hence they should all have official email addresses. We also assumed that they would have active email inboxes where they can receive verification mails to verify their account.

B. Environment

Our application is built on the NodeJS runtime environment. NodeJS is a JavaScript runtime environment that allows developers to write command-line tools and server-side scripts outside of a browser. [3] Our code is developed in the JavaScript language supported by the NodeJS runtime. We also used a cloud-based database - IBM's db2 for our database needs.

C. Application Setup

The project was built in the NodeJS environment. NodeJS has a package manager service that is used to install necessary packages to use within a project. However before a node project can work on any system, you have to install the node run time locally. So before development, part of the set up was to install the NodeJS runtime which comes with the npm package manager on our systems. Next, we initialized a NodeJS project by running `npm init`. This adds a package.json file to our project. The package.json file stores all necessary information regarding our application including but not limited to the project dependencies, scripts to run and test the project, and deployment information.

Our project architecture found in Figure 1 of Appendix B consists of 3 layers: the data layer, the business layer, and the client layer. For our data layer, we used the IBM db2 service for our database. Our business layer consists of an application server. The application server is set up using the ExpressJS npm package. It is a popular, open-source server framework for NodeJS with features such as routing and template serving abilities that allow developers to create servers.
To connect our data layer to the application server we set up our DB connection variables in a .env file and using the node_ibm-db2 npm package we connected our application to the cloud database. The client layer displays data to the user through templates generated using the Embedded JavaScript EJS templating language. We set up the server to use this templating language to display any web pages to be viewed by the client.
We also set up an automated rebuild of our project in development as changes are made and saved using Nodemon - an npm package that restarts the node application anytime changes in the file directory are detected.

We set up testing in our project using Mocha and Chai with Mocha being the test framework and Chai as the assertion library. Mocha is a JavaScript test framework that runs tests written and works together with any assertion library. Chai is a test-driven development assertion library that can be used with any testing framework.
Our tool for version control is the Gitlab tool. We used the Cloud Foundry service from IBM Cloud for the hosting of the application. Part of our setup included downloading and setting up IBM CLI, the command-line tool for interacting with IBM Cloud instances. We added a manifest.yml file that contains our configuration for deploying the application from Gitlab to the cloud.

D. Database Setup

As mentioned above, we used DB2 from IBM for our implementation. The tables and data types of the columns are shown in the logical model captured in Figure 2 in Appendix B.

V. DEMONSTRATION

When the home page of the application is visited, you navigate to sign up to create a new account. A user has to sign up with their UWindsor email address. If the email address does not contain the domain 'uwindsor.ca' an error message is displayed to the user. On successful account creation, a verification link with a timeframe is sent to the user's email address. Clicking the verification link within the specified time frame will activate the account, and the user can proceed to login.

On successful login, the user is redirected to the events page with a list of events separated into 'Upcoming Events' and 'Past Events'. Clicking on the My Tables link in the navigation bar will direct the user to the discussion tables page where all discussion boards in the system are listed. Clicking on any of the discussion boards will

redirect the user to the individual board page where the user can see details about the board and join if they want to. Joining a board is as simple as clicking on the 'Join' button on the board details page. The user will be added to the board as a member. The prototype also allows users to create their board and exchange messages through chat, audio, and video communication.

A. Testing

Tests are contained in the tests folder of the project. To run the tests, you have to run the command `npm test`. The Mocha framework then runs all the tests in the files contained in that folder. We have written tests to check the status codes of responses from the API endpoints we have created in the project. We have also tested special functions like the events scraper function and the function for generating a code for the verification link to be sent to the user on signup. Figure 3 shows what the output is when the test cases are run.

VI. DISCUSSION

During the inception phase, we determined the critical use cases and subsequently designed a prototype of the application in the web prototyping tool called Figma. We deliberated and chose to use PHP and Laravel for development. In the Elaboration phase I, we changed to NodeJS as it is a popular framework for web development and has cloud support in many cloud platforms. We also wanted to expand our skillset by using something new.

When we moved from elaboration phase I to phase II we changed our server implementation. We were previously using the default HTTP server that comes with NodeJS for development but it lacked routing abilities and we had to add those routing features by ourselves and it was leading to additional complexities in our code. We switched to ExpressJS as it provided those features out of the box and allowed us to render HTML dynamically through a templating engine. The results were that we produced our features within time and were able to run our tests easily.

VII. CONCLUSION

The major objective of this project is to fill the gap in informal communication and university culture by providing an online networking platform.

To achieve this, we had divided the project into smaller objectives as listed:

- Implement a discussion board that helps the users of the application to create a table and join existing tables to participate in the discussion or activity.
- Develop a chat feature that helps the participants to share their views if their audio or video is not functioning.
- Display an events page that lists all the upcoming events happening in and around the University of Windsor's campus.
- Develop a Login feature that allows only University of Windsor students to access the application.
- Provide feedback form for the users to address their issues.

So to achieve these objectives we started our research on how each feature can be developed in the best possible way. During our first phase of the project, we started working with PHP, and later on, realized that our app would perform better if we switch to Node. So we switched to Node and soon realized that things are getting messy because of multiple routes and API calls. So on researching more we realized that express would be perfect to manage our app most efficiently because of its various features like Routing, middleware, and many more. It was a great decision and that helped us to simplify our development with cleaner code.

One of the challenging tasks was to decide which API to use for our conference call feature as most of the available APIs are paid and require a subscription. We tested some of the available APIs like jits and zoom which were free for a limited period and with consistent trials and hard work we were able to create a prototype but with some visual inaccuracy. To solve those we learned about the EJS(Embedded Javascript Templates) and completed the feature successfully.

One of the key concerns for us was the security of our users, and to achieve the best possible encryption for the passwords, we have used a readily available library, Bcrypt. It hashes the password to make it secure and also it provides flexibility to set up the number of rounds we want to use for hashing.

In the limited time assigned for this project, we have successfully developed a working prototype

achieving all the objectives mentioned above. Although there is a limitation on how many users can register themselves as we are using the student account of IBM Db2, we have a limited amount of storage.

VIII. FUTURE WORK

There are some features listed below, which can be added in the upcoming iteration.

- Users will be able to upload a profile picture.
- Improving the user interface of conference calls and chat features.
- Admin interface to manage all the features.
- Cron job for deleting unused tables in the discussion board after some days.
- Add Host privileges for the tables.
- Tracking feature for checking feedback reported.

IX. REFERENCES

[1] H. A. Johnson, "Slack," Journal of the Medical Library Association, 02-Jan-2018 [Online] Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5764588/ [Accessed: 05-Dec-2020].

[2] Standuply, "Learn How to Use Slack Effectively in 2019", Standuply, 19-Jun-2019. [Online]. Available: https://standuply.com/how-to-use-slack [Accessed: 05-Dec-2020].

[3] C. Metz, "Forget Apple vs. the FBI: WhatsApp Just Switched on Encryption for a Billion People", WIRED, 05-Apr-2016. [Online]. Available: https://www.wired.com/2016/04/forget-apple-vs-fbi-whatsapp-just-switched-encryption-billion-people/ [Accessed: 05-Dec-2020].

[4] D. Arrachequesne, "Get started," Socket.IO, 24-Sep-2020. [Online]. Available: https://socket.io/get-started/chat/. [Accessed: 05-Dec-2020].

[5] "Simple peer-to-peer with WebRTC," PeerJS. [Online]. Available: https://peerjs.com/. [Accessed: 05-Dec-2020].

[6] "Express/Node introduction," *MDN Web Docs*. [Online]. Available: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/Introduction. [Accessed: 05-Dec-2020].

[7] J. C. Martinez, "Testing in Node.js Using Mocha and Chai [1/2]," *Live Code Stream*, 17-Aug-2020. [Online]. Available: https://livecodestream.dev/post/testing-in-nodejs-using-mocha-and-chai-part-1/. [Accessed: 05-Dec-2020].

[8] "axios," *npm*. [Online]. Available: https://www.npmjs.com/package/axios. [Accessed: 05-Dec-2020].

[9] "node-cron," *npm*. [Online]. Available: https://www.npmjs.com/package/node-cron. [Accessed: 05-Dec-2020].

[10] "Node.js - Express Framework," *Tutorialspoint*. [Online]. Available: https://www.tutorialspoint.com/nodejs/nodejs_express_framework.htm. [Accessed: 05-Dec-2020].

[11] Cloud.ibm.com. 2020. IBM Cloud Docs. [online] Available at: <https://cloud.ibm.com/docs/key-protect?topic=key-protect-retrieve-access-token> [Accessed: 05-Dec-2020].

[12] Ibm.com. 2020. What Is Cloud Computing?. [online] Available at: <https://www.ibm.com/cloud/learn/cloud-computing> [Accessed: 05-Dec-2020].

[13] "ibm_db," npm. [Online]. Available: https://www.npmjs.com/package/ibm_db. [Accessed: 05-Dec-2020].

X. APPENDIX A

Project Management Approach

A. Team Collaboration

Our team used google docs to work parallelly in order to optimize the time required for integrating the documents and suggest edits. We have used WhatsApp for communicating about the project progress and meeting schedule. We used Teams for meetings and finalized tasks are listed in its chatbox. TeamViewer and Any Desk for collaborative development activities.

B. Task Distribution Criteria:

Firstly, we noted down the areas of expertise of each team member.

In our team, all the members have diverse skills and experiences. We have efficiently utilized those skills by disturbing the tasks as per the expertise level.

C. Project Management

In our project, we used the Redmine tool for project management and issue tracking. The tasks for the phase are listed before each phase and all are allocated to individual team members after team meetings. The tasks are assigned priorities to

acknowledge the importance of their completion.

Link:
https://redmine.cs.uwindsor.ca/projects/ase-team1/

Figure 4 shows the tasks and their attributes like the status, priority, and also the assignee name.

D. Version Control Software

In our project, we have used GitLab for maintaining the code versions. We developed features in individual branches and integrated them into the master branch after testing the stability of the feature.

Link:
https://gitlab.cs.uwindsor.ca/ase_team1/student-networking-application/-/tree/master

Figure 5 shows how we used different branches for developing different features.
Figure 7 shows the code repository for the project.

E. Documentation

The README.md file in the GitLab code repository is used to document the steps of using the project code. We also developed wiki pages in GitLab to document the features of the project.
Link:
https://gitlab.cs.uwindsor.ca/ase_team1/student-networking-application/-/wikis/

F. Implementation

We have used agile methodology to develop the code. In each phase, we have adapted changes and included them in the project.

G. Testing

We have developed the features using test-driven development by using Mocha and Chai testing tools. We even performed performance, maintainability, and acceptance testing to ensure the quality of the application developed. Figure 6 shows the screenshots for some of our test cases for our features. While Figure 2 shows all test cases output.

XI. APPENDIX B

Figures

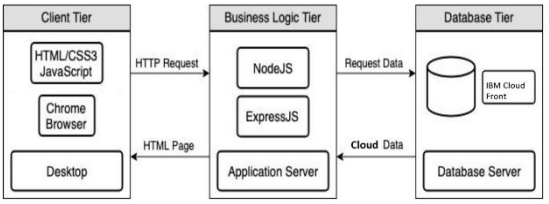

Figure 1: System Architecture

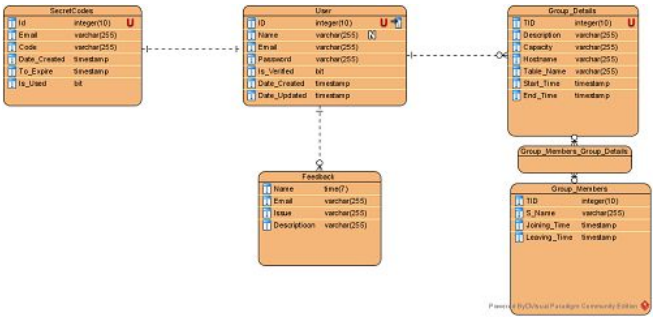

Figure 2: ERD for Database

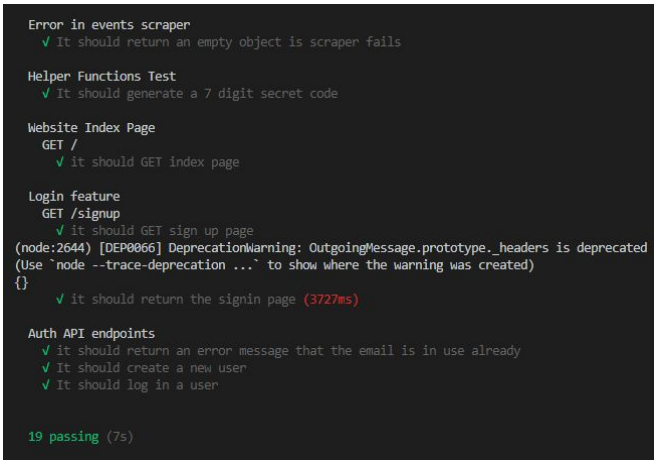

Figure 3: All Unit test cases output

#	Status	Assignee	Subject
3881	Closed	Mounika Vannawada	Chat feature for the discussion board tables
3880	Closed	Mounika Vannawada	Video feature for the discussion board table
3879	Closed	Mounika Vannawada	Audio feature for the discussion board tables
3876	Closed	Adaobi Aniuchi	CRUD operations for discussion tables
3875	Closed	Adaobi Aniuchi	Discussion Board for the application
3874	Closed	Nizar Juned Kadri	Feedback pages
3872	Closed	Nizar Juned Kadri	Create Form to report issues
3871	Closed	Nizar Juned Kadri	Report feature for the application
3822	Closed	Nizar Juned Kadri	Develop the registration feature
3821	Closed	Nizar Juned Kadri	Develop the login related pages
3819	Closed	Adaobi Aniuchi	Develop the Events Display Feature

Figure 4: Redmine Task Division

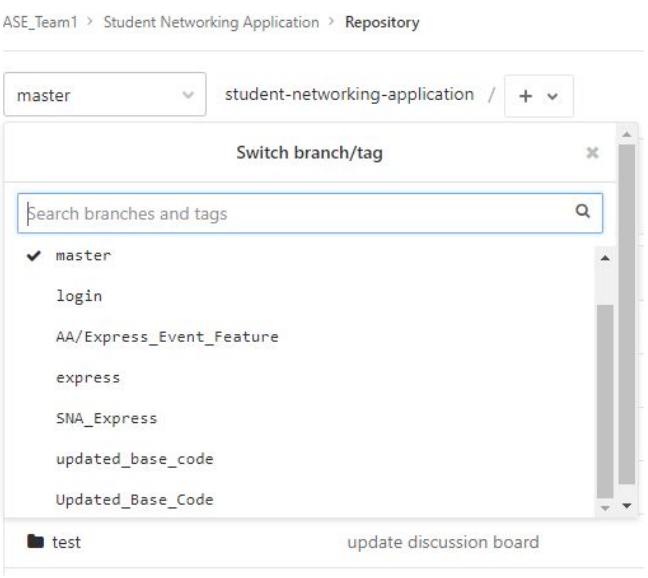

Figure 5: GitLab branching

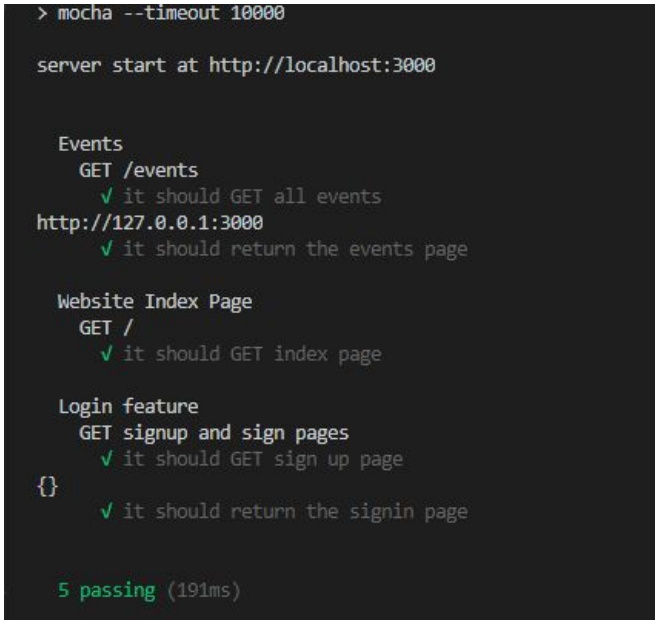

Figure 6: Unit test Cases for Login and Sign Up

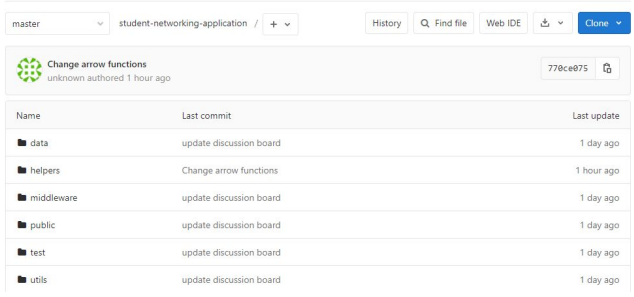

Figure 7: GitLab Code Repository

XII. SPECIAL ACKNOWLEDGEMENT

We would like to express our gratitude to IBM for providing this wonderful opportunity to access IBM Cloud resources to experiment and use them through our academics. This facility really helped us understand the services provided by IBM Cloud and learn practical skills using some of those services.