Title Page

An Interactive Web Application to Learn Coding Online with Real-Time Data Exchange

By

Arjan Gahatraj Sunar

Dissertation Submitted to the Faculty of Engineering, Science and Technology, Infrastructure University Kuala Lumpur, in Fulfilment of the Requirements for the Bachelor in Computer Science (Hons)

February 2022

Abstract

TeamCode will be a web application that focuses on online education for programming. With this application, the instructor will be able to have real-time communication with learners and vice-versa to solve any problems encountered. Online Learning, when used for practical subjects like programming, produces results inferior to learning the subject in full-time classes thereby failing to fully replicate the classroom interaction. The project aims to bridge this gap through real-time code editing and communication through text and audio interfaces. The real-time editing feature will be implemented using WebSockets through Socket.IO library and the communication channel with the audio interface will be implemented through WebRTC (Web Real-Time Communication) using the UDP protocol (User Datagram Protocol). This application will make use of an event-driven programming paradigm as every change made in a user's editor will trigger an event that can be broadcasted with the help of WebSockets using Socket.IO. The project will be using a signalling server to initiate a connection between the users while the real-time data exchange for communication will be done through peer-to-peer communication through WebRTC. The web application will be built using JavaScript (JS), Node js and Nest js will be used to create the API (Application Programming Interface), and React js will be used to create the frontend of the application where Ace editor will be used to create the text editing interface of the application. If this application gets implemented then instructors will be able to see their learner's code in real-time and provide individualized and immediate support. This application could facilitate better communication between instructors and learners in the context of learning to code online.

Acknowledgement

I would like to express my appreciation to Mr Akash Deo and Mr Prakash Gautam for his time to make this work successful.

I also extend my gratitudes towards my family and friends that have helped and supported me to finish this project on time. Their contribution has been imperative.

Approval

We have examined this dissertation and verified that it meets the program and school requirements for the Bachelor of Computer Science (HONs).

Signature : …….
Supervisor : …Mr. Akash Deo
Date: ……...…...

Agreement letter

Valid Reasons for Late Submission

- Students must present assignments to supervisor.
- Students should be mindful of how and when to submit assignments.
- For unmet deadlines due to illness, students must present an original medical certificate to lecturer.
- Students applying for consideration must provide funeral announcement, obituary, doctor's certificate or death certificate, or relevant documentation.

Student Declaration

- This is an output of my own work and isn’t plagiarised. If proven, it can be graded as Failed.
- This work doesn’t include my own previous work without this being stated.
- Any violation of these rules will be considered cheating and Failed.
- I realize that assignments will be accepted after the deadline but with no marks.
- I own a soft copy of this work in case the primary one gets misplaced.
- This declaration will be automatically withdrawn once I drop the IT Project subject.

Student Name: Arjan Gahatraj Sunar
Student ID: 041902900012

Student signature: _____
Date:

Declaration

I guarantee that this report is the result of my own efforts. All statements and material derived from others is properly referenced.

Date:

Arjan Gahatraj Sunar
041902900012

Table of contents

Table of figures

List of tables

List of abbreviations

WebRTC	Web Real-Time Communication
API	Application Programming Interface
UDP	User Datagram Protocol
IE	Internet Explorer
AJAX	Asynchronous JavaScript and XML
SSR	Server Side Rendered
SSG	Static Site Generation
P2P	Peer to Peer
UI	User Interface
JS	JavaScript
HTML	Hyper Text Markup Language
CSS	Cascading Style Sheet
URL	Uniform Resource Locator

Chapter 1
Introduction

1.1 Introduction

Through the last ten years, the definition of education has been constantly changing. With emerging technology, education has been able to take various forms. Due to the unforeseen events in early 2020, all forms of education had to adapt to the online medium using distance learning platforms such as Zoom, Google Meet and Microsoft Teams (Pal et al., 2021). The transition from full-time learning to distance learning is shown to produce a certain “stress” that can hamper the students’ learning outcomes (Zinovieva et al., 2021). Good teaching itself doesn’t vary as the mediums change but translating these teachings over to an online medium brings unique challenges (Driscoll et al., 2012).

According to a survey at Kyiv National Economic University, of bachelor level computer science students, it was discovered that 73% of the students had problems with distance learning (Zinovieva et al., 2021). Figure 1.1.1 shows the most common problems the students faced. Interaction has been deemed as an essential component for online education to succeed (Driscoll et al., 2012). Figure 1.1.1 highlights support and consulting from teachers as the main problem among students of online education which supports the importance of interaction in an online setting.

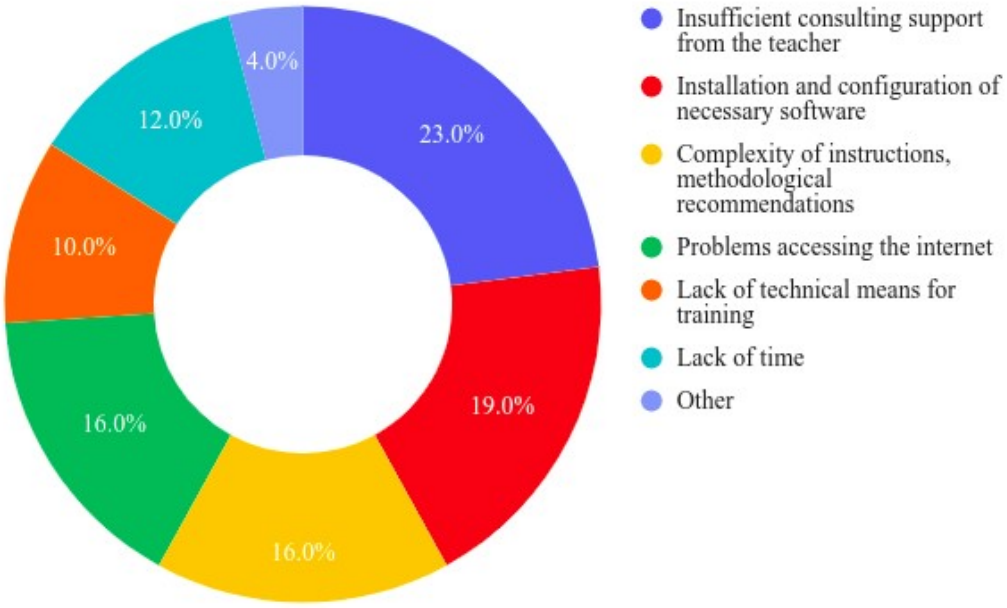

Figure 1.1.1: Problems faced by students of computer science during distance learning

(Zinovieva et al., 2021)

It has been proven that students that learn programming through online education or E-learning have outcomes that lag behind students learning the same course in regular classes (Fojtik, 2017). As shown in Figure 1.1.1, online education for computer science brings along challenges that result in differences in outcomes against full-time students. Online education looses the spontaneous feedback between instructors and learners (Martin, 2019)

The main scope of the system will be in the online education field targeting programming courses. There will be two users facilitated by the system, instructors and learners. It has been found that online courses when designed with maximizing interaction between instructors and learners in mind can be as effective as face to face classes (Driscoll et al., 2012). Therefore, this system aims to provide a platform to learn coding online with a focus on interaction and communication between instructors and learners.

1.2 Problem Statement

1. Meeting applications often used for distance learning like Zoom and Teams have proven to show inefficient results for online coding lectures.
2. Meeting applications provide less features for learning to code online like built-in text editor.
3. Online coding platforms like Hacker Rank and Coding Bat provide less features for real-time code editing.
4. Existing applications require users to install and configure necessary software before they can learn to code online.
5. Fewer systems have diverse accessibility control options such as view only and can edit.

1.3 Objectives

1. To create a real-time meeting web application for learning to code online.
2. To create an interface to write, modify and execute code.
3. To provide a platform for learning to code without having the need to set up and configure programming environments.
4. To include diverse control options like can edit and can view within the system.
5. To develop a web application with notification functionality such that learners can notify any problem.
6. To create an audio channel of communication between instructors and learners.

1.4 Scope

1.4.1 User scope

1. Teachers will be able to host a room where Students can join in.

2. Users will be able to communicate through text and audio interfaces.
3. Users will be able to execute code within the system.
4. Users will be able to share their code with other users in the system with varying permissions.
5. Teachers will be able to edit the code of their Students in real-time.

1.4.2 System Scope

1. The system will allow teachers to create a room for learners to join in.
2. The system will enable text and audio communication.
3. The system will enable execution and modification of code.
4. The system will allow users to share their code with others with permissions such as can edit and can view.
5. The system will enable real-time editing of code.

1.5 Constraints

1. The system will use Ace Editor as the code editor which is not fully supported in mobile browsers or mobile web frameworks.
2. Access to the internet itself can be a limitation to the system as it is a web application and uses UDP protocols and sockets to form the communication bridge between instructors and learners.
3. The support for WebRTC can be a limitation to the system as it is not supported in browsers such as Internet Explorer (IE), Opera Mini and UC browser for Android ("web Rtc " | Can I Use... Support Tables for HTML5, CSS3, Etc, 2021).

1.6 Methodology

The main objective of TeamCode is to combine tools for coding within an online meeting application for better programming education. Within the tools for coding, features such as real time code editing will be implemented in the system through WebSockets, as in comparision to AJAX (Asynchronous JavaScript and XML) it is known to work with better network performance and greater throughput (Puranik et al., 2013). WebSockets will be used with the Socket.IO library. For achieving synchronous meeting, the communication channel will be implemented through WebRTC (Web Real-Time Communication). The project will be using a signalling server to initiate a connection between the users while the real-time data exchange for communication will be done through peer-to-peer communication.

1.7 Development tools

The development tools to be used in the system are as follows:

1. Socket.IO library: This library provides a higher-level API (Application Programming Interface) to work with WebSockets. This library will be used to facilitate real-time editing of code by broadcasting the changes made in the Ace editor.
2. NestJS: It is a framework for building scalable server-side Node.js applications. The project will use this framework to create the back-end and its API.
3. React.js: It is a library used to build reactive user interfaces for the web. The project will use this framework for creating the user interface of the system.
4. Ace editor: This library provides a text editor similar to that of Visual Studio Code for the web. The project will use this library to create a text editor for users to write code in and the library also provides events for when the code is written which can be used to broadcast the changes for the real-time editing feature.
5. PeerJS: This library is a higher-level wrapper over the WebRTC API that simplifies peer-to-peer (P2P) data, video and audio calls. The project will use this library to establish the P2P connection between the users for text and audio communication.

1.8 Structure of Report

The structure of the report outlines the entire format of the report. The report consists of three chapters and the contents of the chapters are described below:

1. Chapter 1: This chapter gives a brief outline of the methodology used in the system, and gives a description of the problem and objective the project is trying to achieve.
2. Chapter 2: This chapter provides a review of the literature that has already been written in the related field of the report topic.
3. Chapter 3: This chapter gives a detailed description of the methodology used to achieve the report's objective, the different software tools and the reasons behind using the said tools.

1.9 Conclusion

There exists a discrepancy between distance learners and face-to-face learners especially in terms of programming subjects. It has also been proven that interaction plays a major role in the success of distance learning. Taking this into consideration the project- TeamCode will be a meeting web application made specifically for coding with added features like execution of code, integrated text editor and real time editing.

Chapter 2
Literature review

2.1 Background study

Education in a virtual classroom requires higher effort to remain motivated to learn when compared to a face-to-face classroom (Mihai, 2014). Due to the lack of real presence of learners, instructors in virtual classrooms require more effort to gather the attention of learners (Mihai, 2014). This problem is present in the online education of every study but it is amplified in the field of studies that emphasises more on practical aspects like computer science (Zinovieva et al., 2021). The author also presents that the transition to online education from a face-to-face lecture further distances students from the course, their peers and teachers ultimately introducing a certain “stress” that affects their learning. It has been noted that the presence of a community and its support is essential to maintain students’ engagement in virtual education programs (Berry, 2017; Rovai, 2003). Past literature has defined a community as a social group that is credited to managing stress and decreasing isolation (Pyhältö et al., 2009; Stubb et al., 2011; Berry, 2017). Emphasis on building a community gives rise to a collaborative learning process. It has been proven that collaborative learning provides benefits over individual learning (Nokes-Malach et al., 2015). Collaborative learning enables the communication between peers, discussion of ideas and the opportunity to question and exchange ideas which motivate learners to be active (Laal, 2013). Collaborative learning will be focused on the system through the use of audio and text communication with real-time editing features that will encourage students to take direct feedback from their peers and their teachers.

2.1.1 Virtual classroom and collaborative learning

Past literature in the field of education comparing virtual and face-to-face classrooms have shown dissimilarities. Some literature suggests virtual classrooms cannot be considered equal to face-to-face classrooms due to their lack of non-verbal communication and real presence that affects motivation levels (Mihai, 2014). In contrast, others suggest that with the proper teaching strategies online learning can be very effective (Berry, 2019). It has been agreed that a collaborative approach to learning is the key to success in both online and face-to-face classrooms (Laal, 2013; Laal & Laal, 2012). In both environments, students try to seek out entertainment and social interaction which can directly affect their motivation to learn in the said environment (Fisher & Coleman, 2001). While designing a platform to enable distance learning, the aspect of social interaction must be considered and can be facilitated through e-mail attachments, instant messaging, newsgroups, synchronous real-time environments, and personal Web pages (Fisher & Coleman, 2001). Instant messaging and audio/video conferencing can be the tools needed for replicating verbal interaction and allow better collaborative work (Repman et al., 2005). Due to the

importance of audio and video conferencing it has been deemed as a functional requirement of every virtual classroom (Rehman & Khan, 2016).

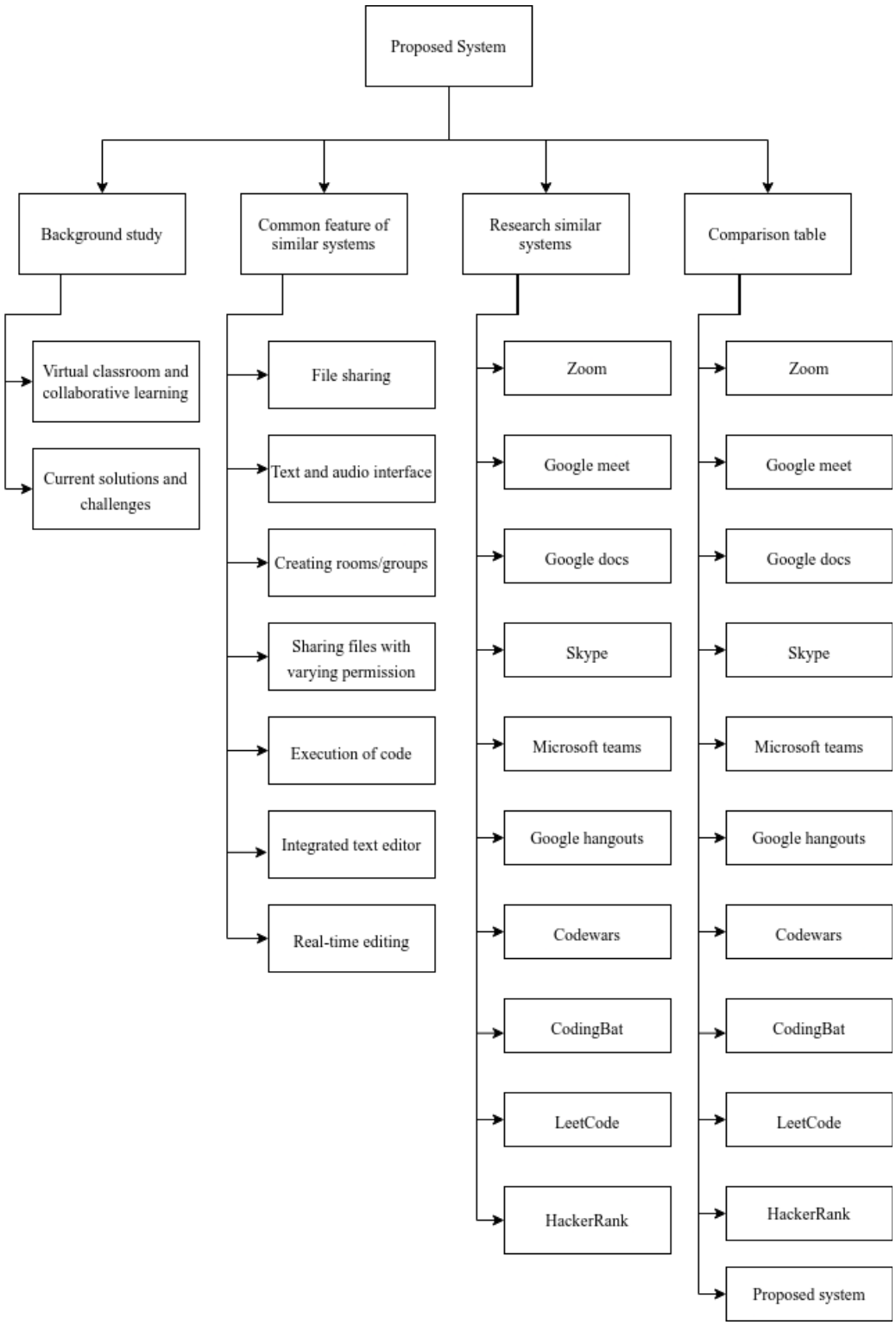

Figure 2.1.1: Literature Framework

2.1.2 Current solutions and challenges

The synchronous meeting applications such as Zoom, Google Meet, Cisco Webex, Skype, ClickMeeting, Adobe Connect, Free Conference, Big Blue Button, Microsoft teams, VEDAMO virtual classroom, Google Hangouts are being used for virtual classrooms (Deepika et al., 2021). During the process of adapting to the virtual environment several challenges emerged. From the side of students it was discovered that they had a tendency for isolation and lack of motivation and from the side of teachers they experienced lack of control over learners and couldn't use their communication skills (Deepika et al., 2021). Also during the use of Zoom for distance learning, students were less willing to nominate themselves to respond to questions and it was hard to monitor learners engagement with larger classes (Moorhouse, 2020). This method for distance learning had potential security issues. In case of Zoom, 'Zoombombing' where a live class gets intentionally hacked was a major security issue (Kohnke & Moorhouse, 2020). While meeting applications have been the main substitute for classrooms, judge applications such as codewars, codehunt, coding-bat. Codeboard, pythonchallenge, Leetcode, Hackerrank have been advantageous for online learning at one's own pace (Wasik et al., 2018).

2.2 Common features of similar systems

The similar features of existing systems used to educate programming online are follows:

2.2.1 File sharing

Systems that support online education enable users to share files with other users allowing users to share their solution and promote collaborative learning.

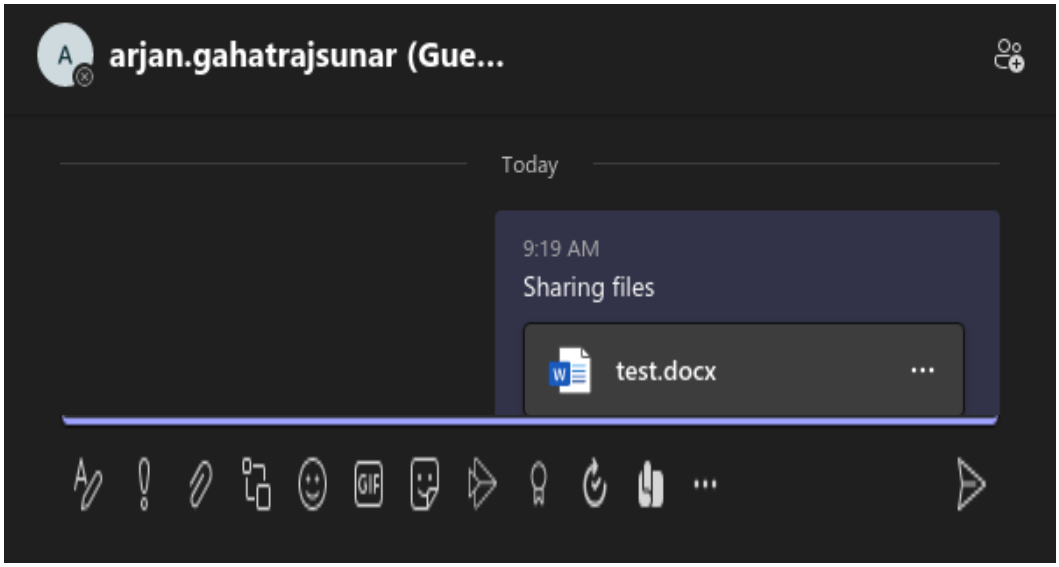

Figure 2.2.1: Sharing files in Microsoft teams
(Video Conferencing, Meetings, Calling | Microsoft Teams, 2022)

2.2.2 Text and audio interface

This feature enables communication and interaction between users of the system.

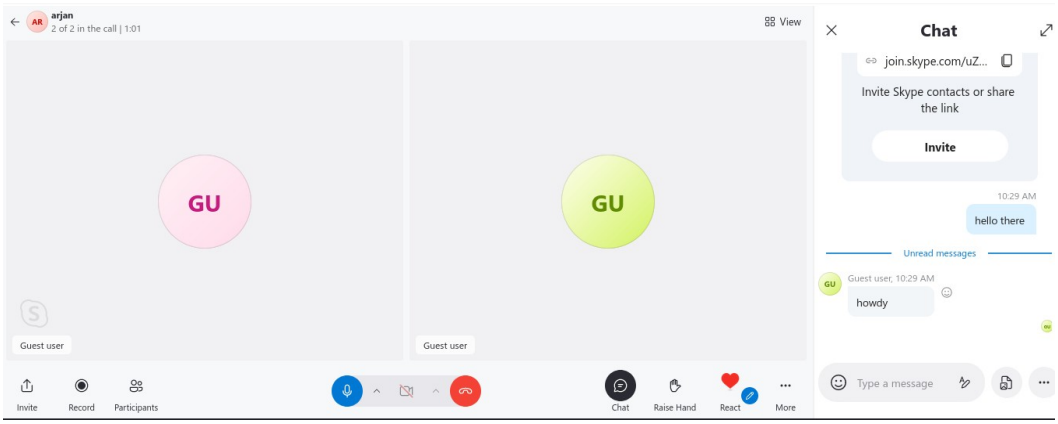

Figure 2.2.2: Skype's Audio and text interface
(Skype | Stay Connected with Free Video Calls Worldwide, 2022)

2.2.3 Creating rooms/groups

This feature enables users to create groups or rooms to schedule a meeting for a class.

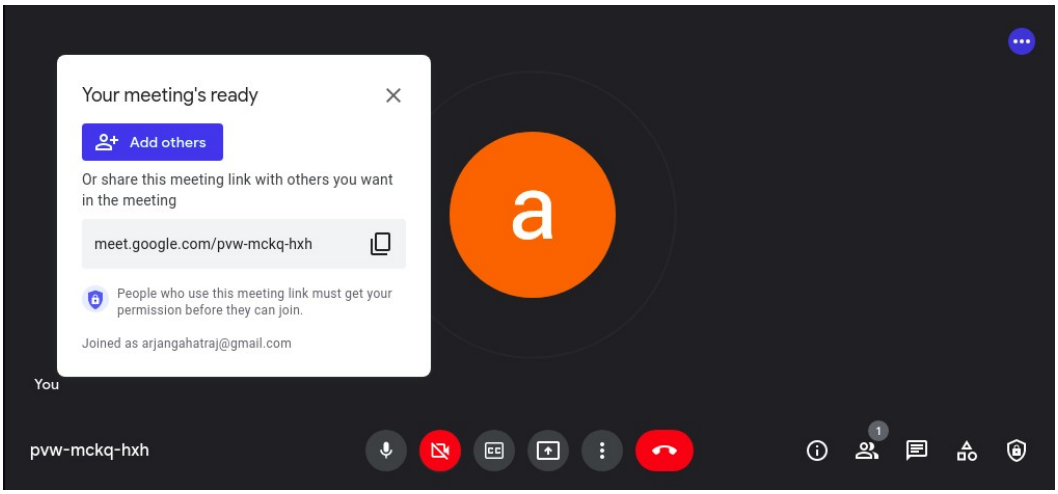

Figure 2.2.3: Creation of a meeting room in Google meet
(Google Meet, 2022)

2.2.4 Sharing files with varying permission

Systems allow users to share files with varying permissions. Allowing certain users full rights to edit, view or share while limiting other users.

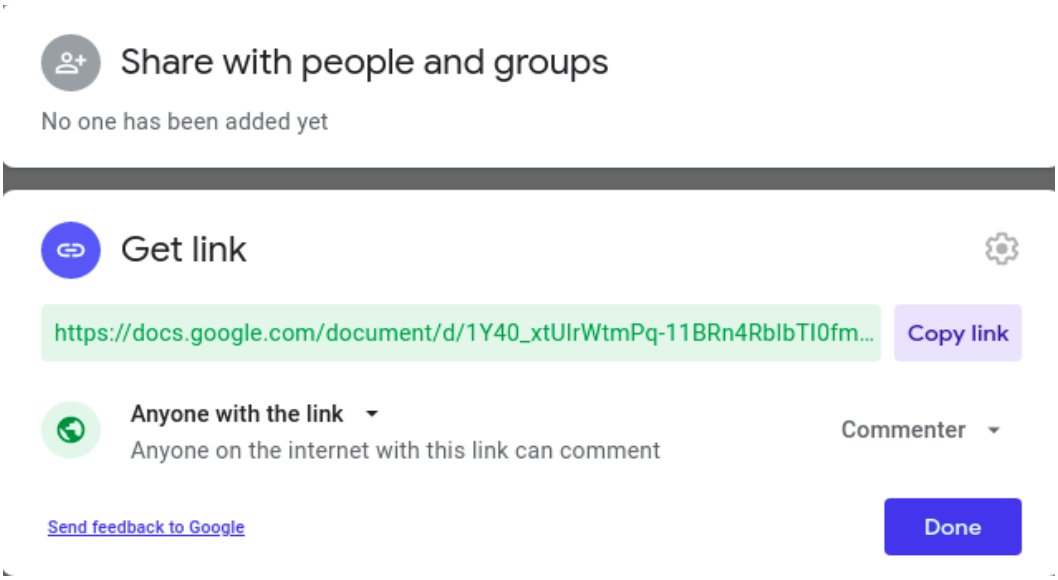

Figure 2.2.4: Sharing files in Google Docs with different permission
(Google Docs, 2022)

2.2.5 Execution of code

This feature enables users to execute their code or solution to a given problem and view their results.

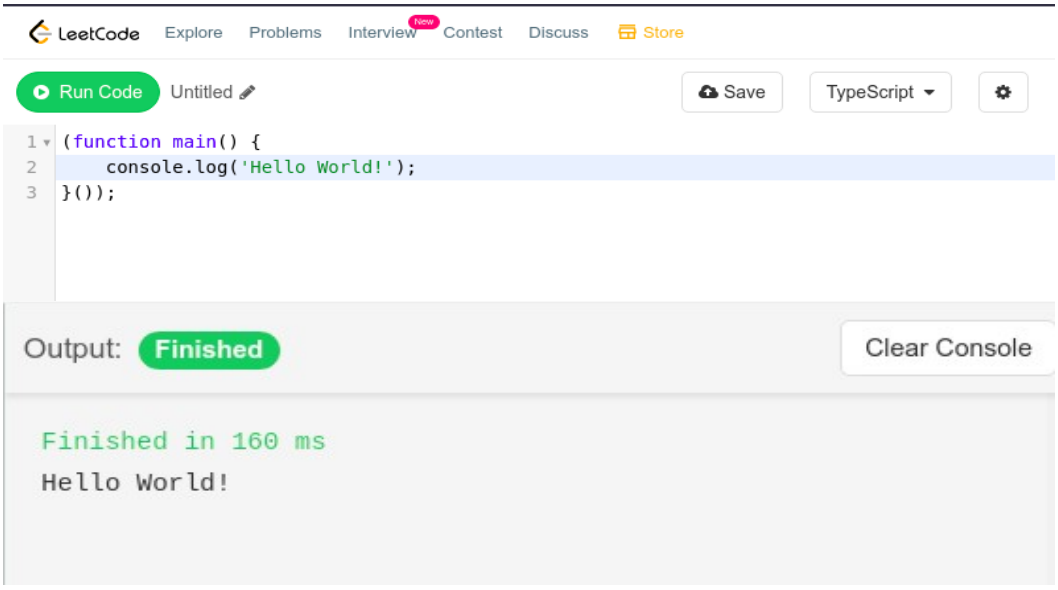

Figure 2.2.5: Executing code in LeetCode
(LeetCode - The World's Leading Online Programming Learning Platform, 2022)

2.2.6 Integrated text editor

This feature enables users to create and modify their code or solution within the system.

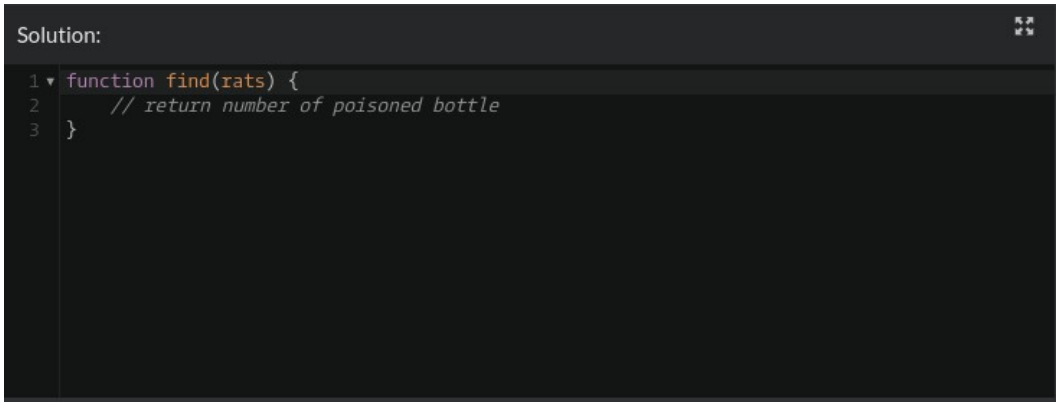

Figure 2.2.6: Integrated text editor in Codewars
(Codewars, 2022)

2.2.7 Real-time editing

This feature enables users to share their solutions with other users and get feedback and help through real-time modifications.

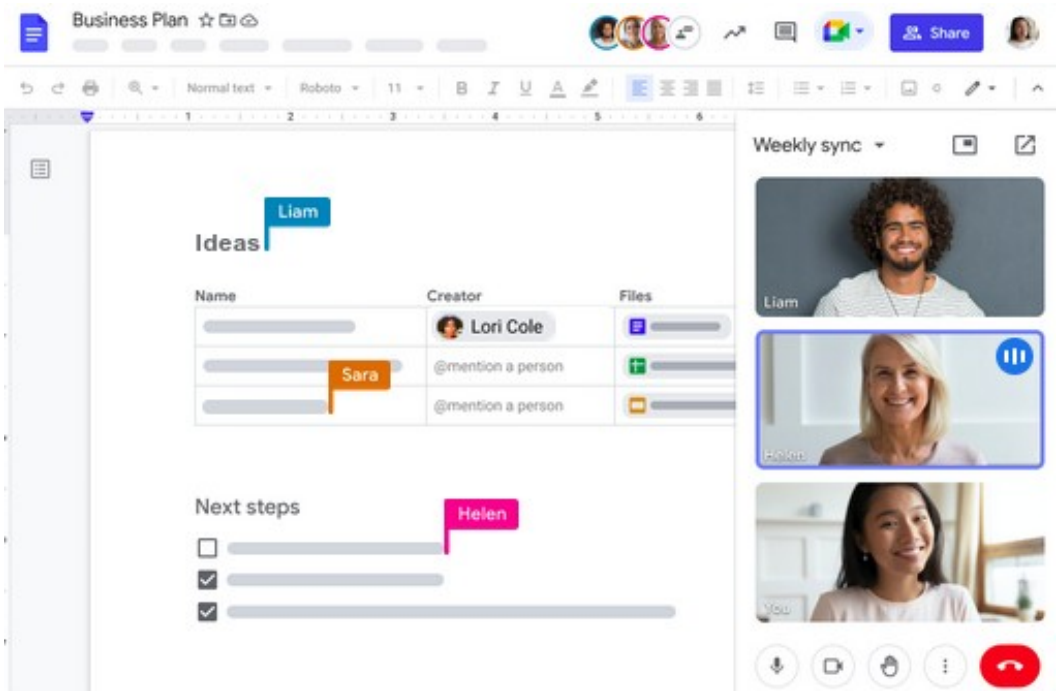

Figure 2.2.7: Real-time editing in google docs
(Google Docs, 2022)

2.3 Research similar systems

2.3.1 Zoom

Zoom is a secure and reliable video platform that provides services including meetings, chat, phone, webinars, and online events. Zoom enables ad hoc and scheduled meetings for both individuals and groups with simple access using a web link. It also incudes tools to emulate interaction such as screen share, emoticons and chat which helps to emulate the non-verbal cues. It enables users to share file, communicate with text and audio and create groups which are being used to facilitate distance learning.

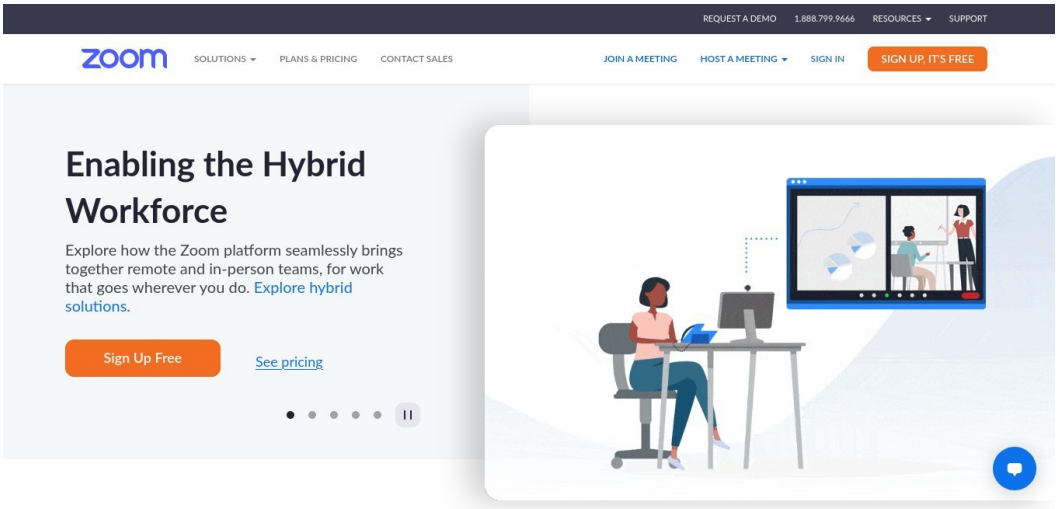

Figure 2.3.1: Screenshot of Zoom's Website
(Video Conferencing, Cloud Phone, Webinars, Chat, Virtual Events | Zoom, 2022)

2.3.2 Google Meet

Google meet is a real-time meeting application by Google that allows users to share video, desktop, presentations using a web browser. It is a web application that allows users to quickly set up instant meetings or schedule a meeting beforehand. It allows users to create rooms and communicate through audio and video.

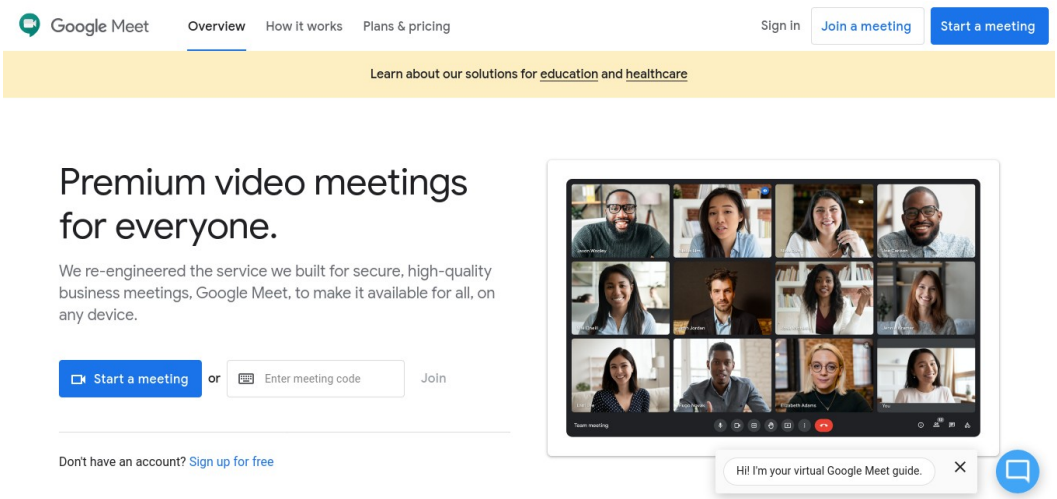

Figure 2.3.2: Screenshot of Google Meet's Website
(Google Meet, 2022)

2.3.3 Google Docs

Google Docs is an online word processing application that is part of Google's free, web-based Google Docs Editors package. It enables users to create a group brainstorm sessions, shared notes, group assignments and share resources. It is used in conjunction with other Google products like Google Meet for distance learning. It enables users to share files with varying permissions, edit documents, and also features real-time editing for boosting collaboration.

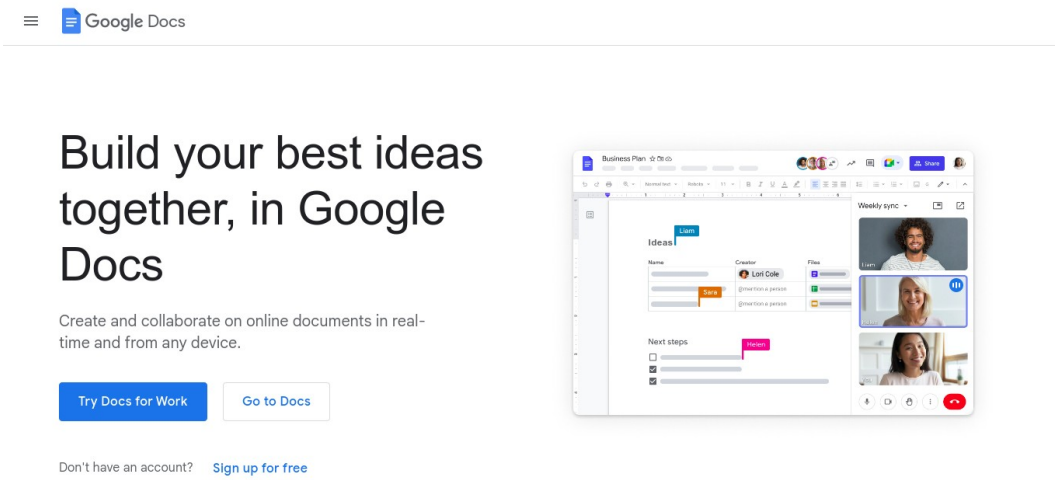

Figure 2.3.3: Screenshot of Google Doc's Website
(Google Docs, 2022)

2.3.4 Skype

Skype is a proprietary telecommunications service that includes VoIP-based video telephony, video and voice conferencing. Skype enables users to communicate online, record meetings and create learning resources for future use. It has an integrated voice and text chat with file sharing features where users can create separate groups for particular subjects.

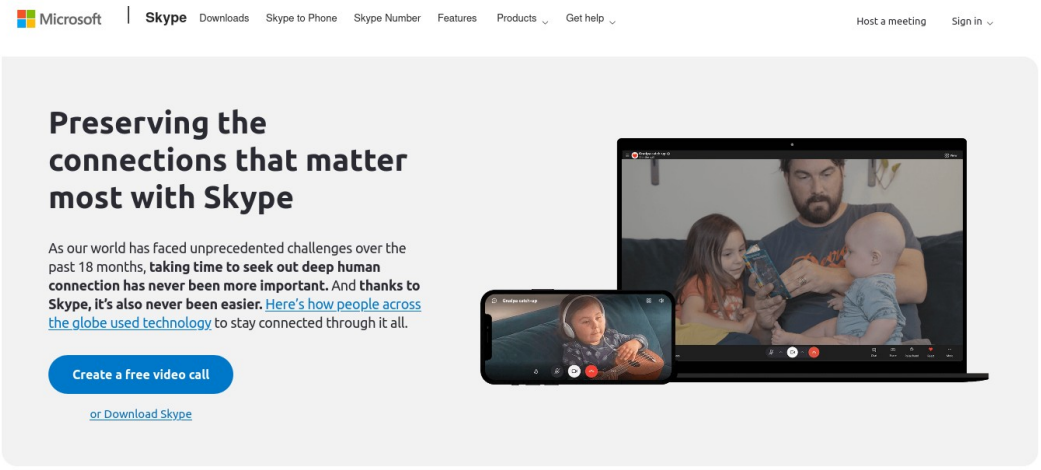

Figure 2.3.4: Screenshot of Skype's Website
(Skype | Stay Connected with Free Video Calls Worldwide, 2022)

2.3.5 Microsoft teams

Microsoft Teams is a specialized business communication platform that includes workplace chat and videoconferencing, file storage, and application integration. Microsoft Teams is also used for distance learning with its collaborative features like screen sharing, audio and video communications, share files and edit files with other users in real-time.

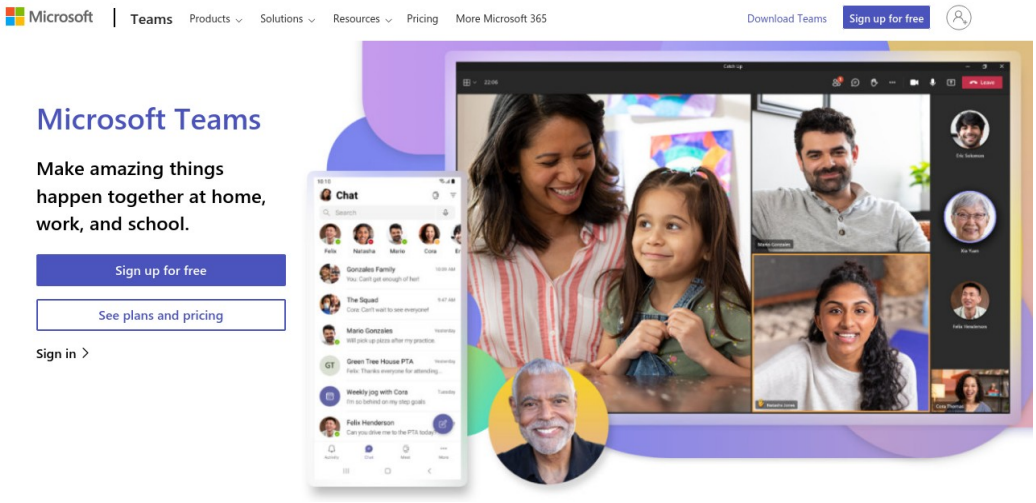

Figure 2.3.5: Screenshot of Microsoft Teams' Website
(Video Conferencing, Meetings, Calling | Microsoft Teams, 2022)

2.3.6 Google Hangouts

Google Hangouts is a cross-platform instant messaging application created by Google that allows users to form groups and hold meetings through video/audio conferencing. It enables users in a online learning setting to ask questions, collaborate in group chats and create virtual rooms for classes or projects.

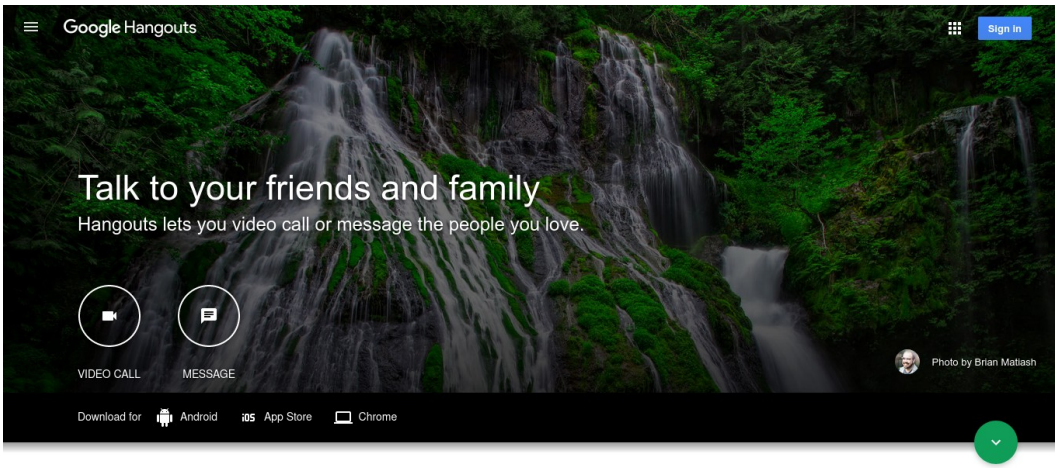

Figure 2.3.6: Screenshot of Google Hangouts' Website
(Google Hangouts, 2022)

2.3.7 Codewars

Codewars is a coding practice site for all programmers that assists in the learning of various programming languages. It enables users to edit, modify and execute code with its integrated code editor. It is primarily used for practising coding and allows users to participate in a forum and get help from other users.

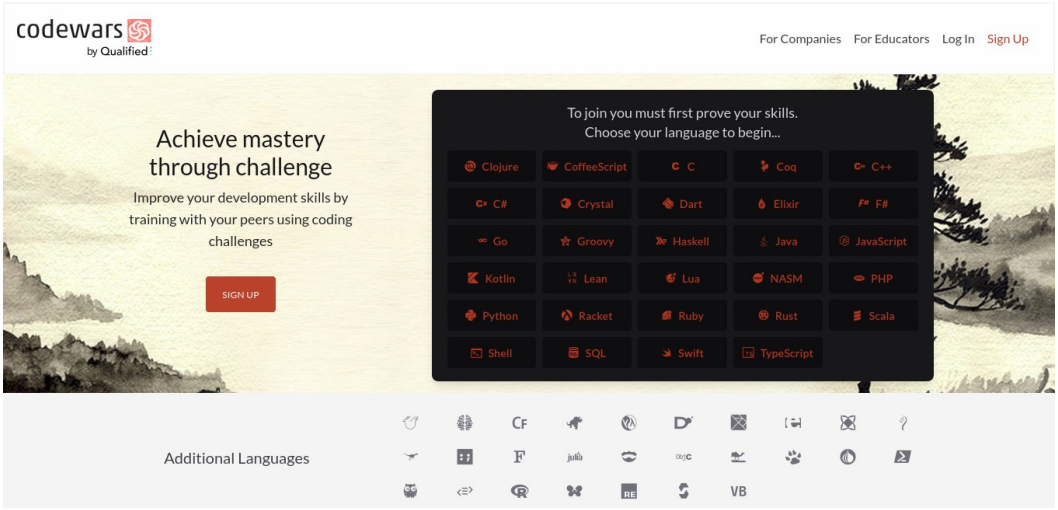

Figure 2.3.7: Screenshot of Codewars website
(Codewars, 2022)

2.3.8 Coding-bat

CodingBat is a free website with a large number of live coding problems. The site's goal is to help people learn to code in Python and Java. It has problems created specifically for users to learn Java and Python and allows users to practice and learn programming fundamentals.

Figure 2.3.8: Screenshot of CodingBat's Website
(CodingBat Java, 2022)

2.3.9 LeetCode

LeetCode is a web-based platform that helps to improve coding abilities, broaden technical knowledge, and prepare for technical interviews. It allows users to practice coding problems and practice for interviews. It provides multiple coding problems and courses for users to learn coding online. It has an integrated code editor, allows users to execute code and allows users to share their solutions to coding problems.

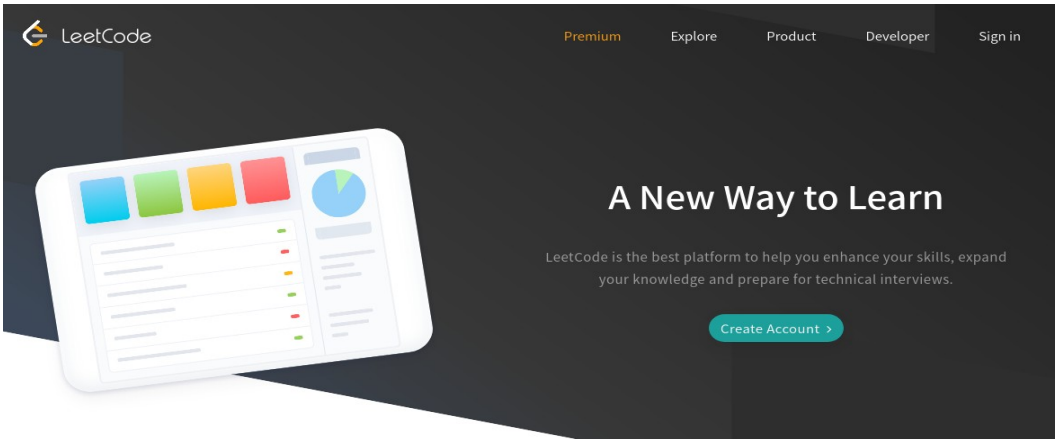

Figure 2.3.9: Screenshot of LeetCode's Website
(LeetCode - The World's Leading Online Programming Learning Platform, 2022)

2.3.10 HackerRank

HackerRank is a digital company that focuses on competitive programming challenges for both individuals and corporations, in which programmers compete by attempting to program according to given parameters. It enables users to solve and practice coding problems and has courses with problems designed to teach how to code. It allows users to modify, execute and test their solutions to coding problems. It also allows users to communicate with its text chat feature.

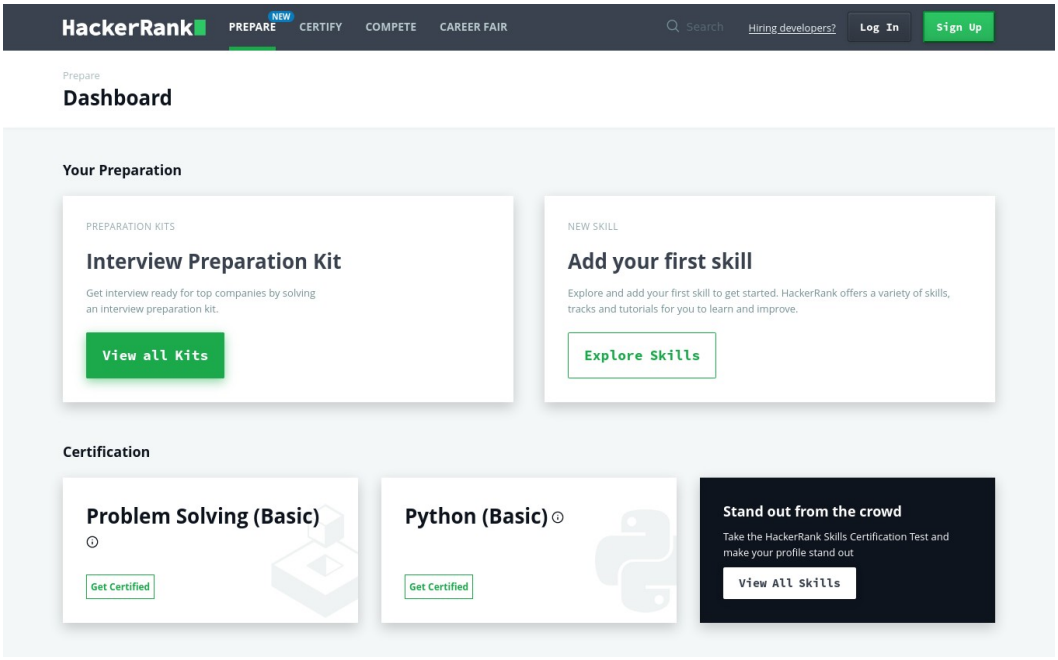

Figure 2.3.10: Screenshot of HackerRank's Website
(HackerRank, 2022)

2.4 Feature comparison

Table 2.4.1: Comparison table

Feature	Zoom	Google Meet/ Docs	Skype	Micros-oft teams	Google Hangou-ts	Code wars	Coding Bat	Leet Code	Hacker Rank	Propos-ed system (Team Code)
File sharing	✓	✓	✓	✓	✓	✗	✗	✗	✗	✓
Voice chat	✓	✓	✓	✓	✓	✗	✗	✗	✗	✓
Text chat	✓	✓	✓	✓	✓	✗	✗	✗	✓	✓
Creatin-g rooms/ groups	✓	✓	✓	✓	✓	✗	✗	✗	✗	✓
Sharin-g files with varying permis sion	✗	✓	✗	✓	✗	✗	✗	✗	✗	✓
Executi-on of code	✗	✗	✗	✗	✗	✓	✓	✓	✓	✓
Integra-ted text editor	✗	✓	✗	✓	✗	✓	✓	✓	✓	✓
Real-time editing	✗	✓	✗	✓	✗	✗	✗	✗	✗	✓

As shown in Table 2.4.1, the proposed system will include all the features combining features from synchronous online meeting applications and online judge systems to enable a platform to learn programming online with integrated meeting features and text editing features. The proposed system will include the functionality to send notifications to instructors when encountering errors.

2.5 Conclusion

Research supports that one of the setback of online education is lack of real presence of learners and instructors in virtual classrooms. This setback is further emphasized while learning practical subjects like programming. However with focus on collaborative learning and interaction, this lack of presence can be minimized. Therefore, TeamCode will employ collaborative features like audio and text communication interfaces. Further since synchronous meeting apps help to provide collaboration and online judge applications provide a platform for users to gain technical knowledge, aspects from both these applications will be used in TeamCode. This will enable better technical online education for programming.

Chapter 3

Methodology

3.1 Introduction

Online education has been stated to be optimal when conducted with proper learning strategies (Berry, 2019). Collaborative learning is said to be one of these learning strategies that has been proven in both online and face-to-face scenarios (Laal, 2013; Laal & Laal, 2012). While designing a platform to enable distance learning, the aspect of social interaction must be considered and can be facilitated through e-mail attachments, instant messaging, newsgroups, synchronous real-time environments, and personal Web pages (Fisher & Coleman, 2001). Instant messaging and audio/video conferencing can be the tools needed for replicating verbal interaction and allow better collaborative work (Repman et al., 2005). Therefore, TeamCode will implement elements of real-time communication through text and voice communication and real-time code editing features. Real-time features and streaming voice can be heavy resource consuming task. This limits the methods that can be used to achieve a stable connection between an instructor and learner for TeamCode.

WebSockets are known to work with better network performance and greater throughput when compared against AJAX (Asynchronous JavaScript and XML) (Puranik et al., 2013). Therefore, TeamCode will make use of WebSockets to enable real-time editing of code between the users. The WebSockets implementation will be done through Socket.IO library which provides an abstraction over the native WebSocket API and provides fallback methods in case WebSockets are not supported. The communication channel with the audio interface will be implemented through WebRTC (Web Real-Time Communication) using the UDP protocol (User Datagram Protocol). This application will employ an event-driven programming paradigm, with any change made in the user's editor triggering an event that can be broadcasted via WebSockets and Socket.IO. The project will deploy a signaling server to establish a connection between users, while real-time data transmission will be accomplished by WebRTC peer-to-peer communication.

3.2 Agile Methodology

Agile methodology refers to a set of software development approaches that are iterative and incremental in nature. Every agile methodology focus on being flexible and adaptive to new cases. In every iteration the phases of Software Development Life Cycle-planning, defining, designing, building, testing and deployment; is carried out.

For TeamCode, Scrum will be the development methodology. Scrum refers to an agile development style that consists of small iteration time frames called sprints involving completion of defined tasks through planning, development, review and launch phases. The entire project will comprise of small tasks focussing on particular feature. Each of this tasks will involve a sprint time frame where the particular task will go from planning to

development and integrating with the system. With the development and integration of each of this task will incrementally build the system for TeamCode.

3.3 Use case diagram

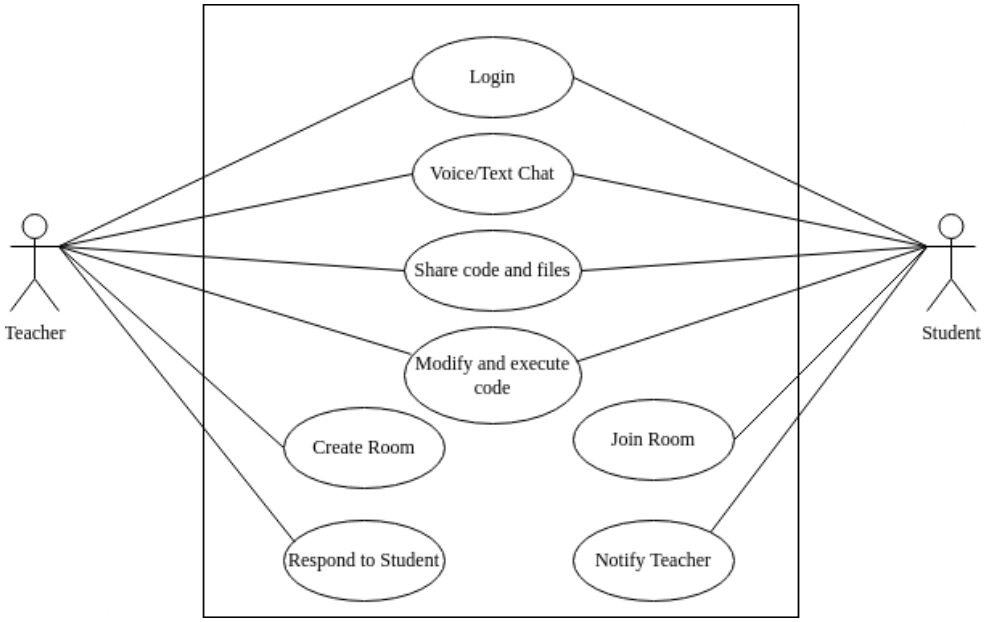

Figure 3.3.1: Proposed system use case diagram (TeamCode)

3.4 Written use case

3.4.1 Login

Use case ID:	UC-1
Use case Name:	Login
Actors:	• Teacher • Student
Preconditions:	None
Post conditions:	The users are logged into the system and assigned roles of teacher and student respectively
Trigger:	User wants to join or create a class
Main success scenario	
1. Users provide GitHub username	

2. Users provide GitHub password 3. Users log in to the system
Exception
1. Users do not have GitHub account 2. Users provided invalid credentials

3.4.2 Create Room

Use case ID:	UC-2
Use case Name:	Create Room
Actors:	• Teacher
Preconditions:	Teacher has logged in
Post conditions:	Teacher creates a room with unique id and distributes it to students
Trigger:	Teacher wants to start a class
Main success scenario	
1. Teacher generates room id 2. Teacher hosts a room 3. Teacher distributes it to students	

3.4.3 Join Room

Use case ID:	UC-3
Use case Name:	Join Room
Actors:	• Student
Preconditions:	Teacher has created the room and Student has logged in
Post conditions:	Student joins a room with unique id created by a Teacher
Trigger:	Teacher has started a class and Student wants to join
Main success scenario	
1. Teacher joins in 2. Student joins in	
Exception	
1. Incorrect room id	

3.4.4 Voice/Text chat

Use case ID:	UC-4
Use case Name:	Voice/Text chat
Actors:	• Teacher • Student
Preconditions:	Users have logged in and joined a room
Post conditions:	Users communicate with each other
Trigger:	Teacher has started a class
Main success scenario	
1. Users can send messages 2. Users can use voice communication 3. Users can communicate with other users	

3.4.5 Share code and files

Use case ID:	UC-5
Use case Name:	Share code and files
Actors:	• Teacher • Student
Preconditions:	Users have logged in and joined a room
Post conditions:	Users share code and files with each other
Trigger:	Teacher started a class
Main success scenario	
1. Users can share code 2. Users can share files 3. Users can learn from codes of other users	

3.4.6 Modify and execute code

Use case ID:	UC-6
Use case Name:	Modify and execute code
Actors:	• Teacher • Student

Preconditions:	Users have logged in and joined a room
Post conditions:	Users can write, modify and execute code
Trigger:	Teacher started a class
Main success scenario	
1. Users can use text editor 2. Users can modify the existing code 3. Users can execute their code	

3.4.7 Respond to Student

Use case ID:	UC-7
Use case Name:	Respond to Student
Actors:	• Teacher
Preconditions:	Student notifies Teacher with a problem
Post conditions:	Teacher solves a student's problem by editing their code in real-time
Trigger:	Student encounters a problem and notifies the teacher
Main success scenario	
1. Teachers can view notifications from Student 2. Teachers can respond to student queries 3. Teacher can solve student's problem	
Exception	
1. No students have joined 2. No notifications	

3.4.8 Notify Teacher

Use case ID:	UC-8
Use case Name:	Notify Teacher
Actors:	• Student
Preconditions:	Student joins a room and executes code
Post conditions:	Student notifies the Teacher if errors are found when code is executed
Trigger:	Student encounters a problem during execution of code
Main success scenario	
1. Students can notify teacher 2. Student can get feedback	

3.5 Activity Diagram

3.5.1 Login

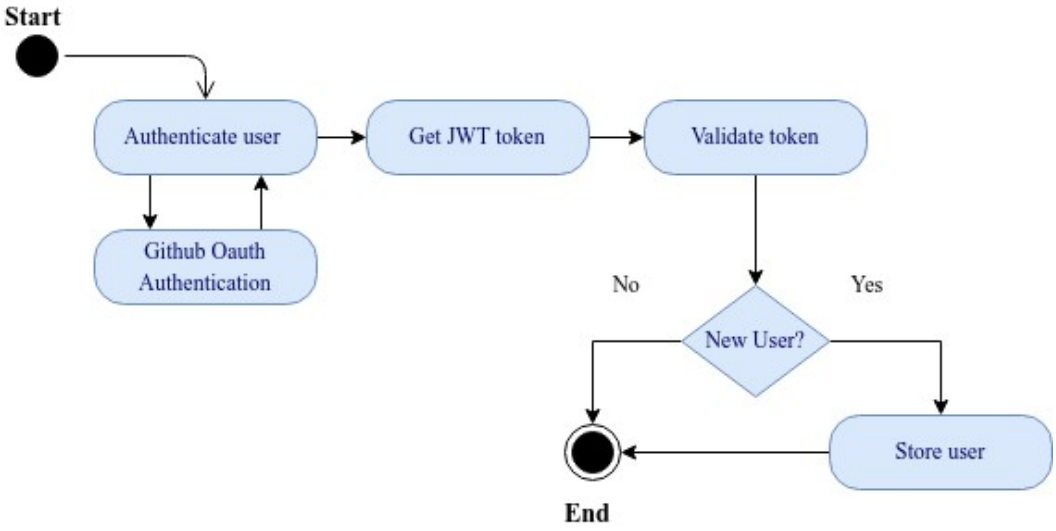

Figure 3.5.1: Overview of authentication process to the system

3.5.2 Create Room

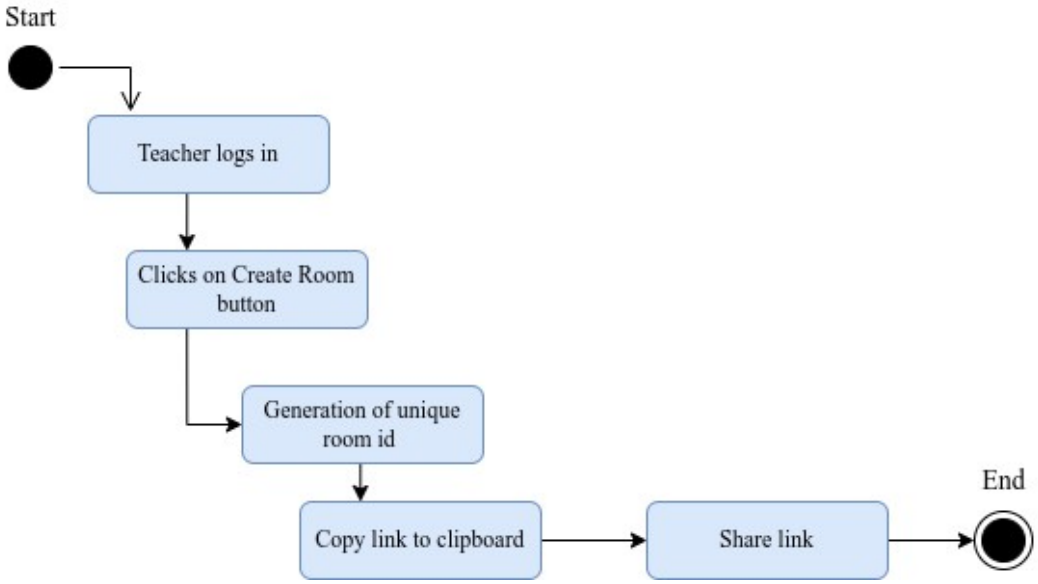

Figure 3.5.2: Steps of creating a new meeting room

3.5.3 Join Room

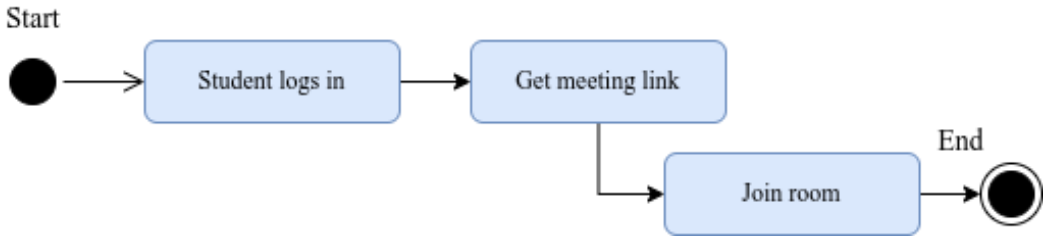

Figure 3.5.3: Steps to join a meeting room

3.5.4 Voice/Text chat

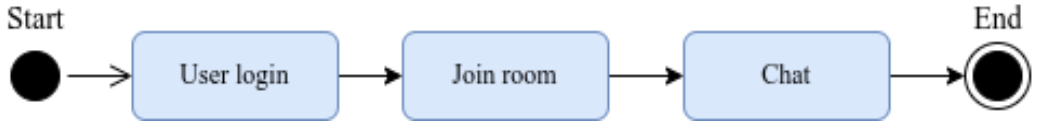

Figure 3.5.4: Overview of text and voice chat in TeamCode

3.5.5 Share files and code

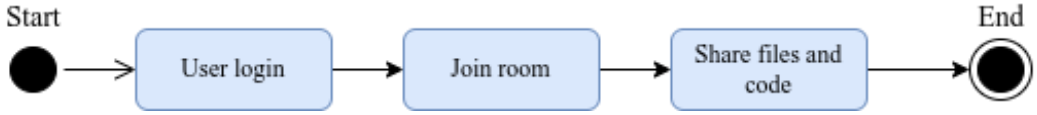

Figure 3.5.5: Overview of sharing files in TeamCode

3.5.6 Modify and execute code

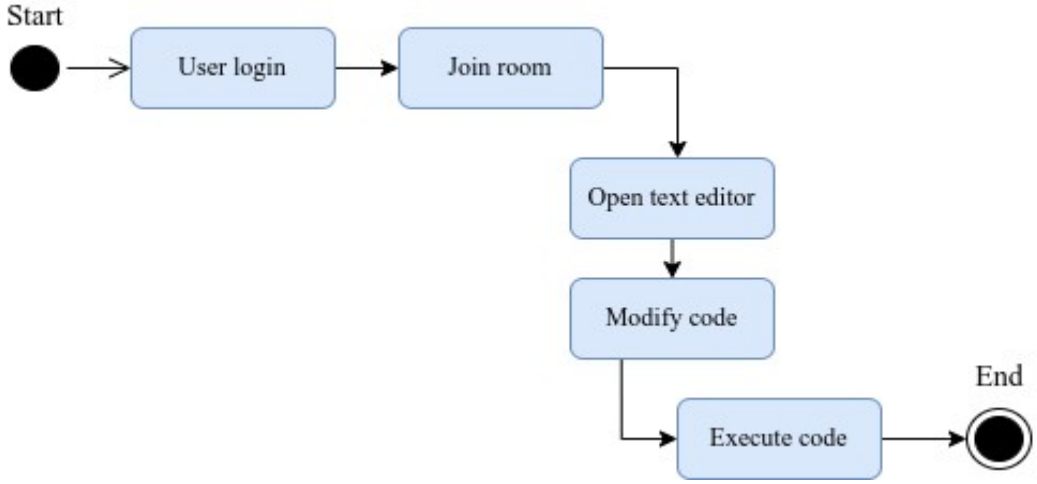

Figure 3.5.6: Steps of modification and execution of code in TeamCode

3.5.7 Notify Teacher

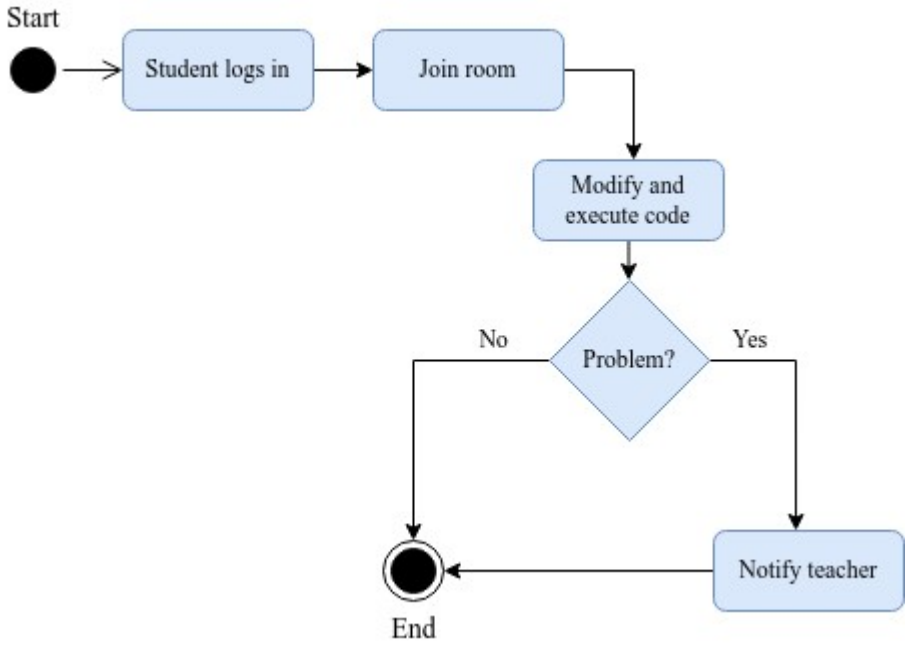

Figure 3.5.7: Steps of notifying Teacher of a problem by a Student

3.5.8 Respond to Student

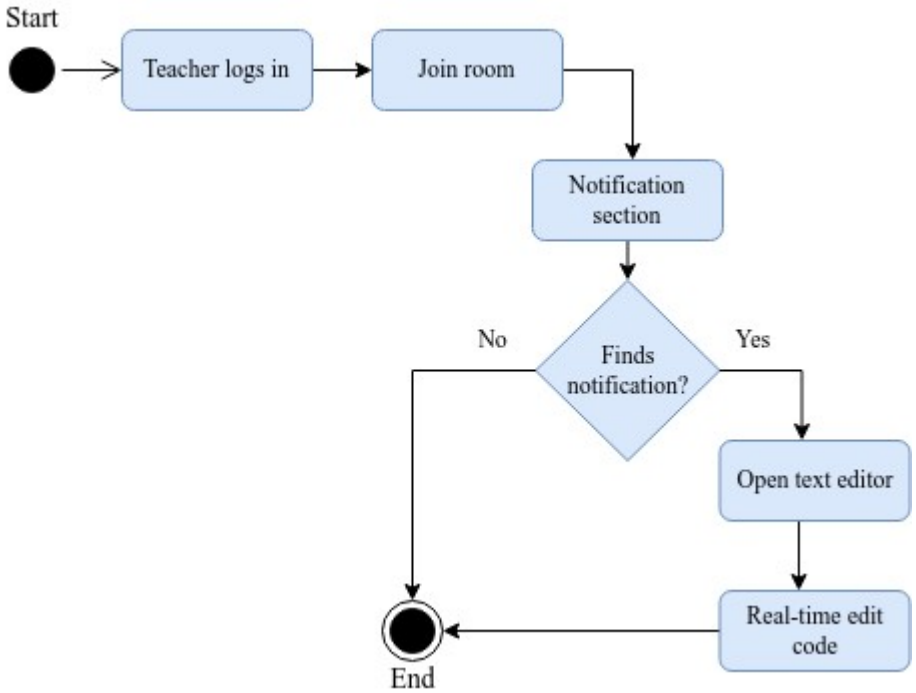

Figure 3.5.8: Overview of a Teacher responding to notifications

3.6 Sequence diagram

3.6.1 Login

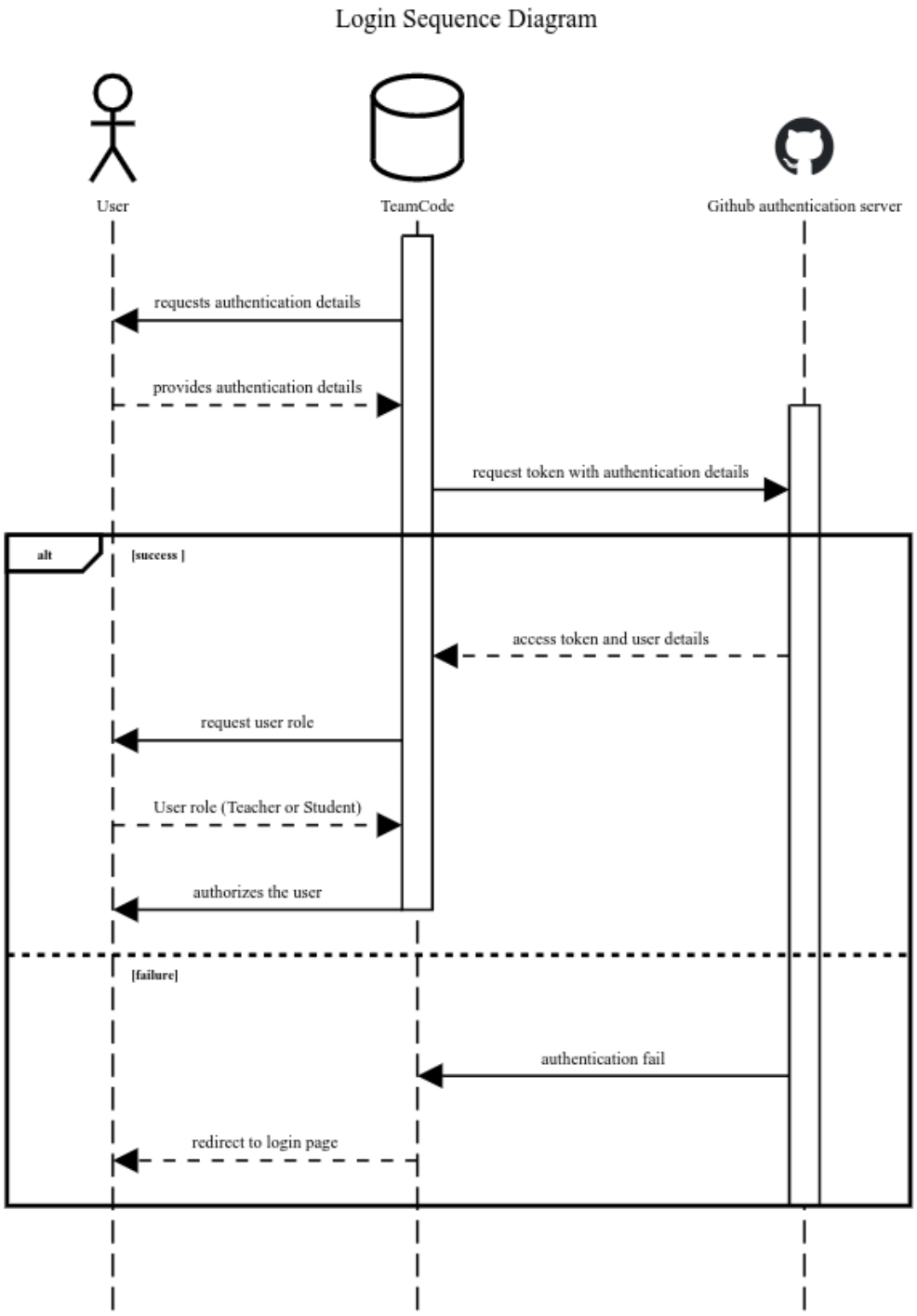

Figure 3.6.1: Login sequence diagram

The figure above shows the login process of TeamCode system using GitHub authentication strategy. The figure shows that a user needs to authenticate with their GitHub account

which creates an access token used to authenticate the user into TeamCode's system. The user then provides the role- either Teacher or Student, to the system.

3.6.2 Room Creation

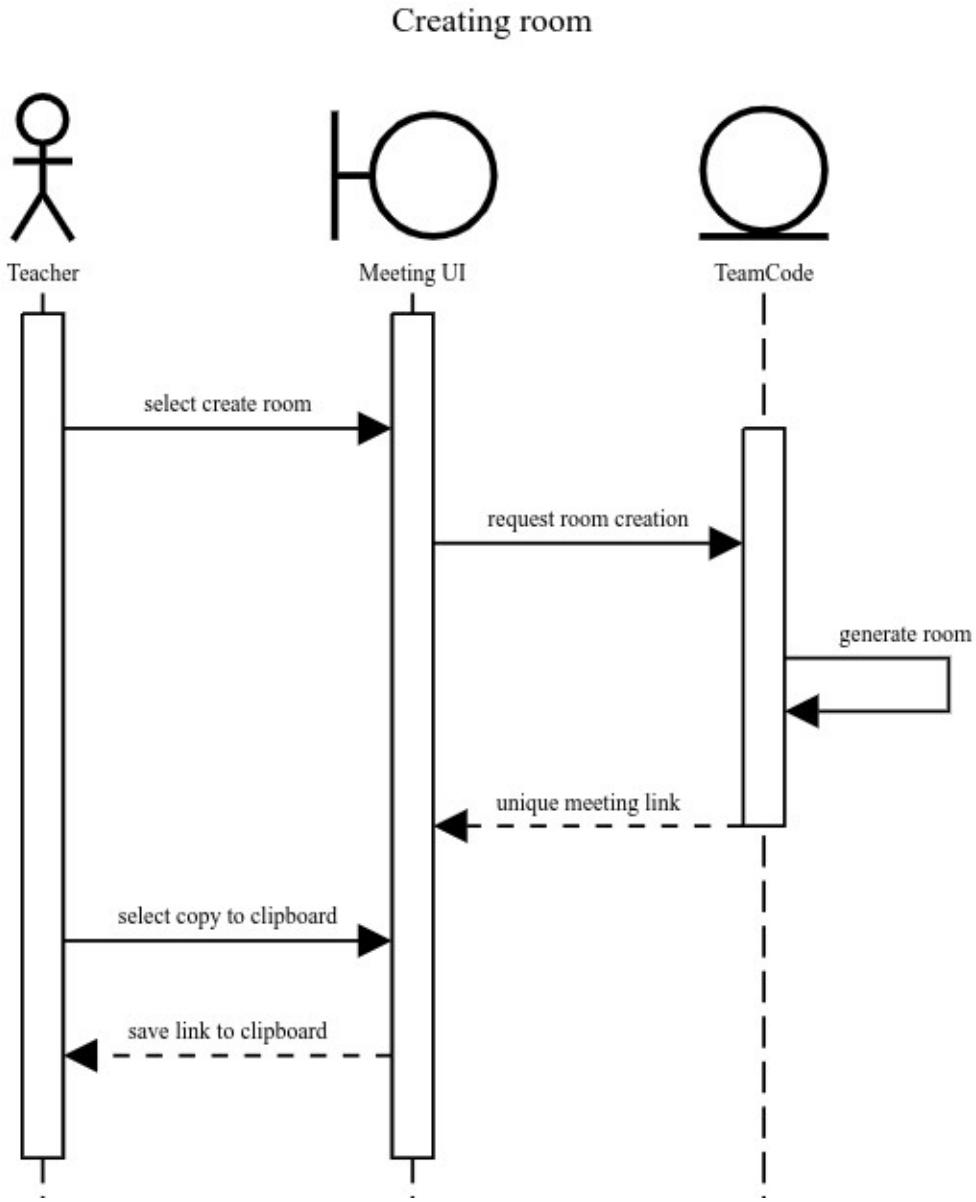

Figure 3.6.2: Room creation sequence diagram

The figure above shows the steps a Teacher takes to create a room in TeamCode.

3.6.3 Join room

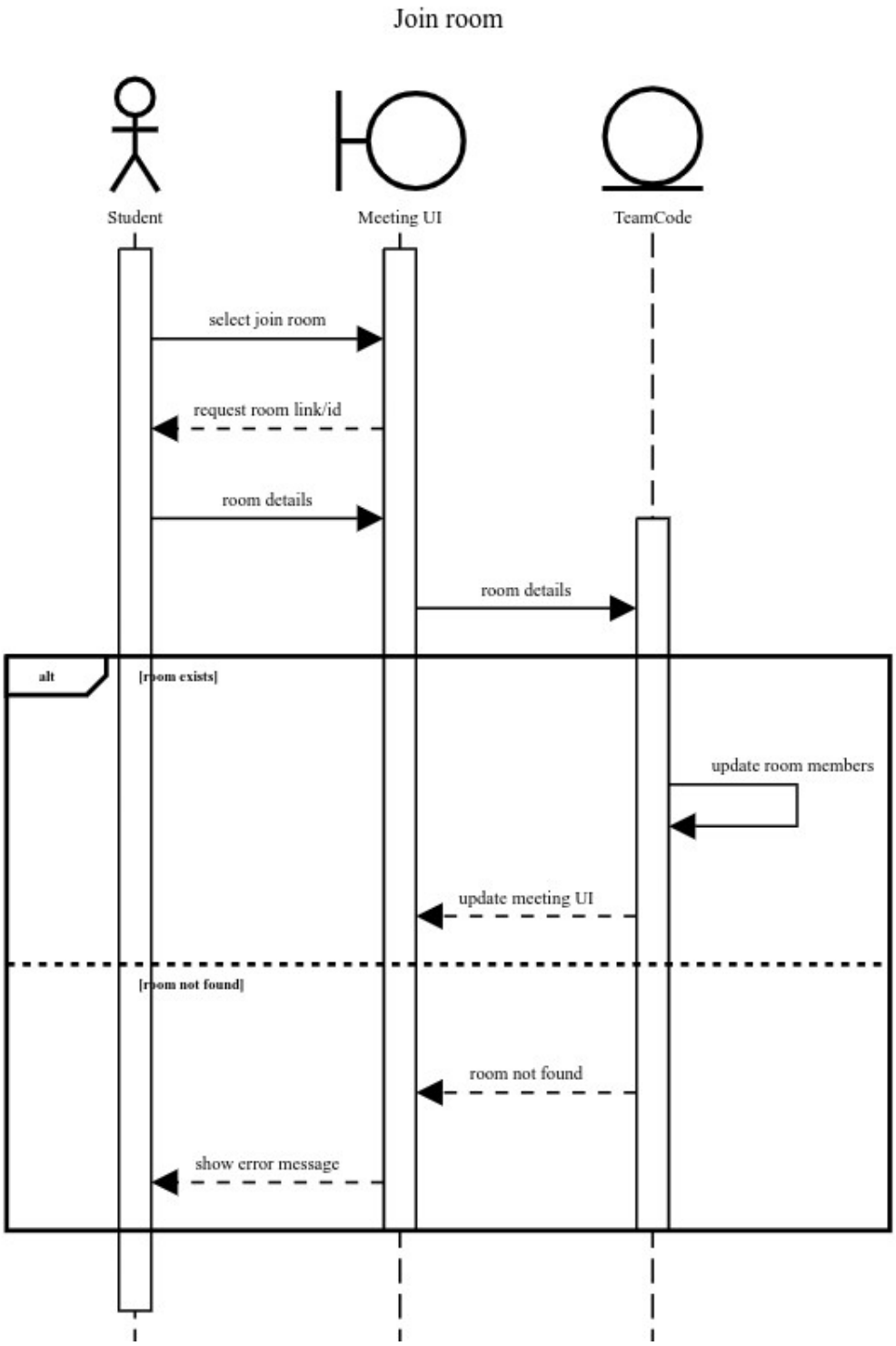

Figure 3.6.3: Join room sequence diagram

The figure above shows the steps a Student takes to join a room in TeamCode. A Student can join a room with only correct room id else is presented with error message.

3.6.4 Text/Voice chat and share files

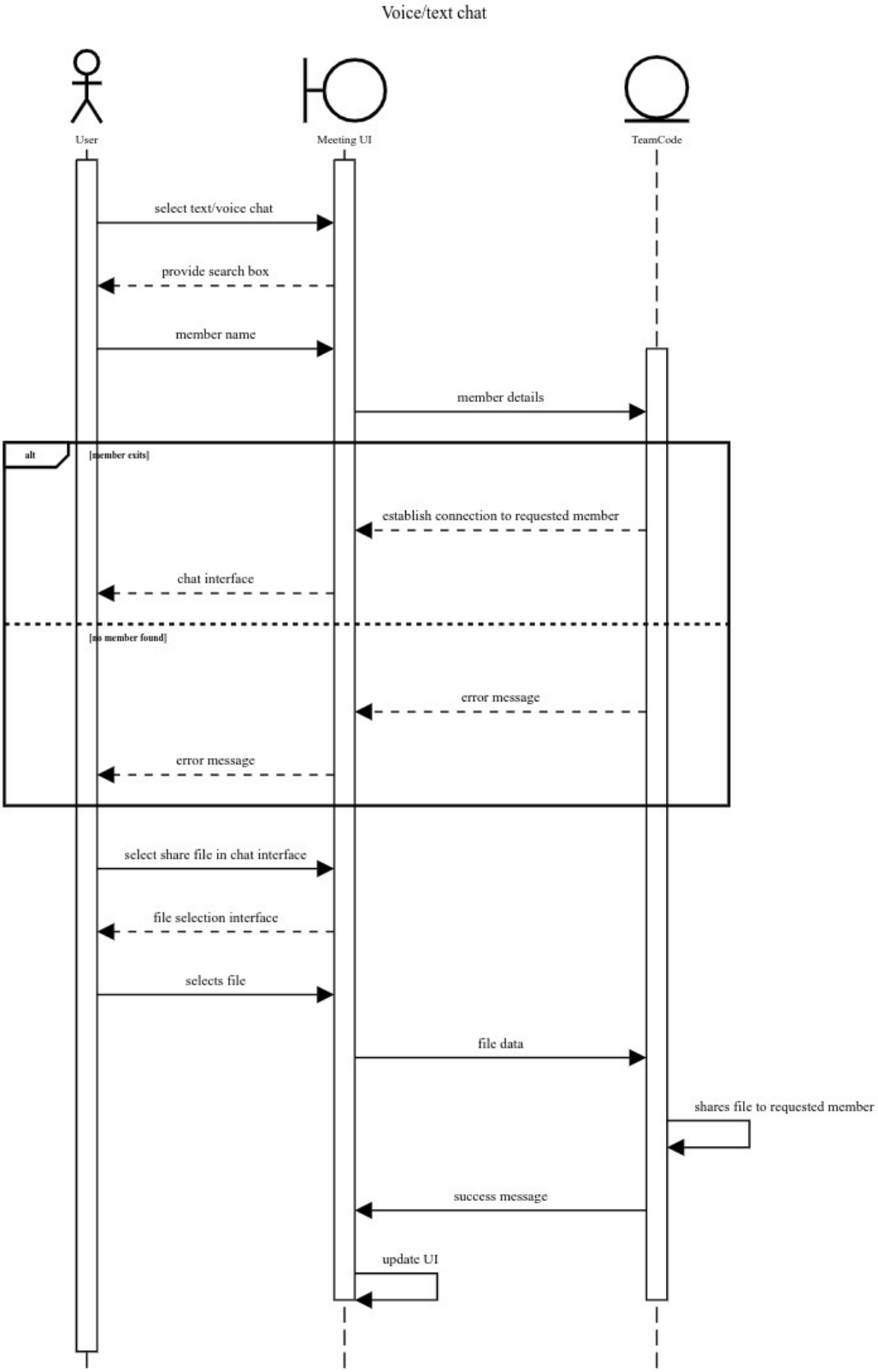

Figure 3.6.4: Share files and chat interface sequence diagram

The figure above shows the steps a user will take to share files and chat in TeamCode. After a user receives the chat interface then the user will be able to share files through the interface.

3.6.5 Share code

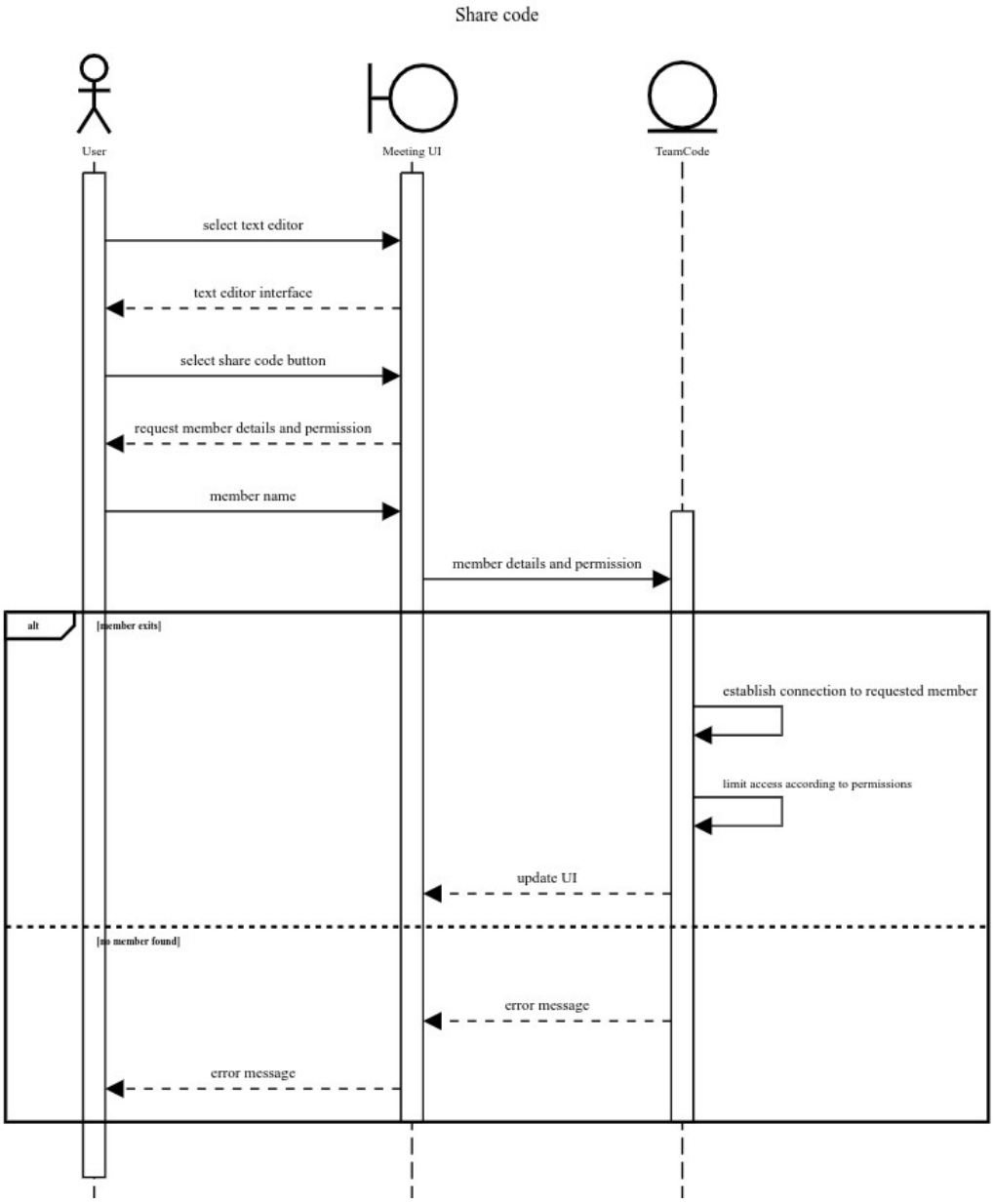

Figure 3.6.5: Share code sequence diagram

The figure above shows steps that a user will take to share code in TeamCode. In the text editor a user will be able to share code with other members within the room. The user will be able to assign permission to edit or view the code while sharing.

3.6.6 Modify and execute code

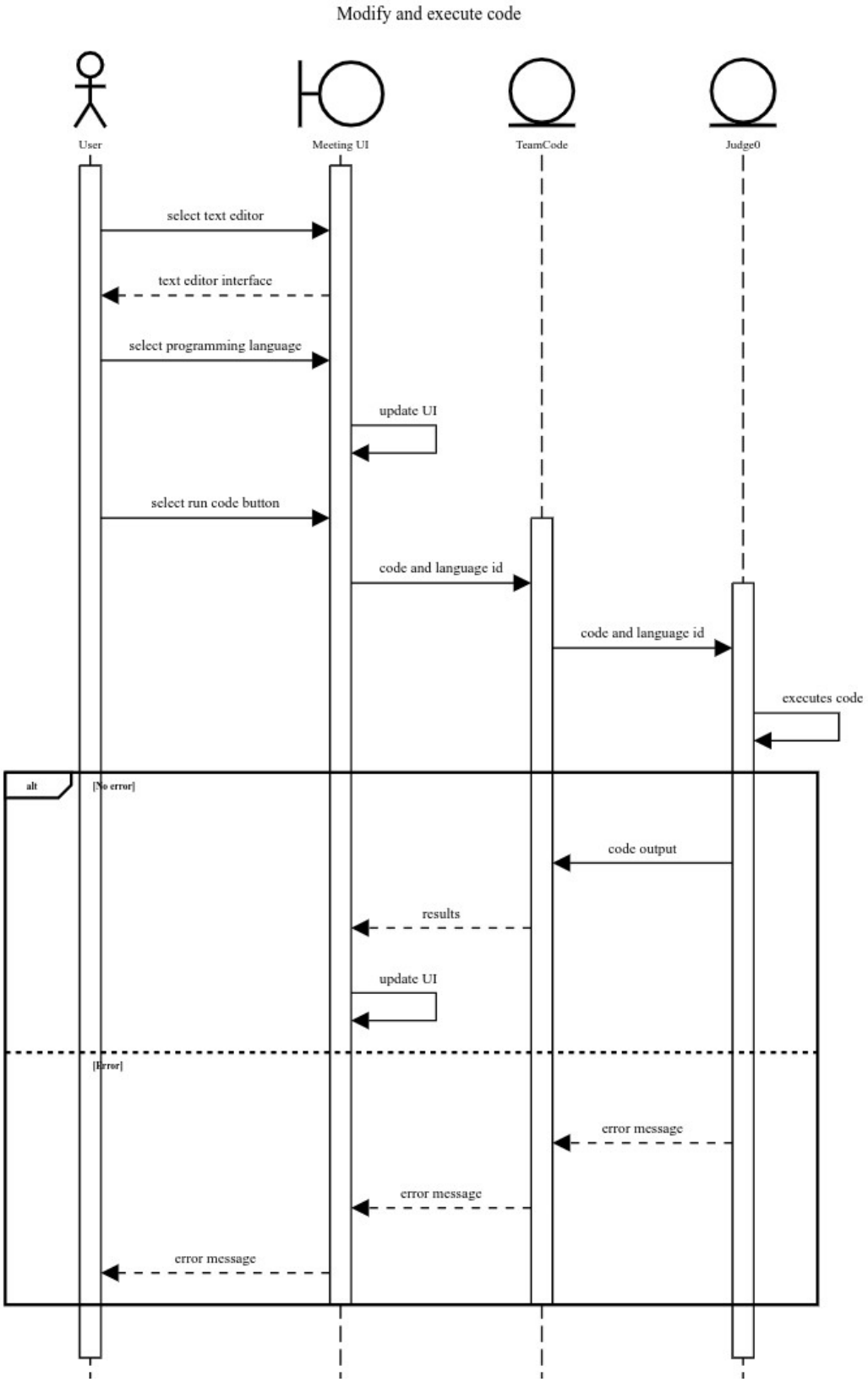

Figure 3.6.6: Sequence diagram for modification and execution of code

The figure above shows the steps a user will take to modify and execute code. The code is modified though an integrated text-editor while it is executed via Judge0 API.

3.6.7 Notify Teacher

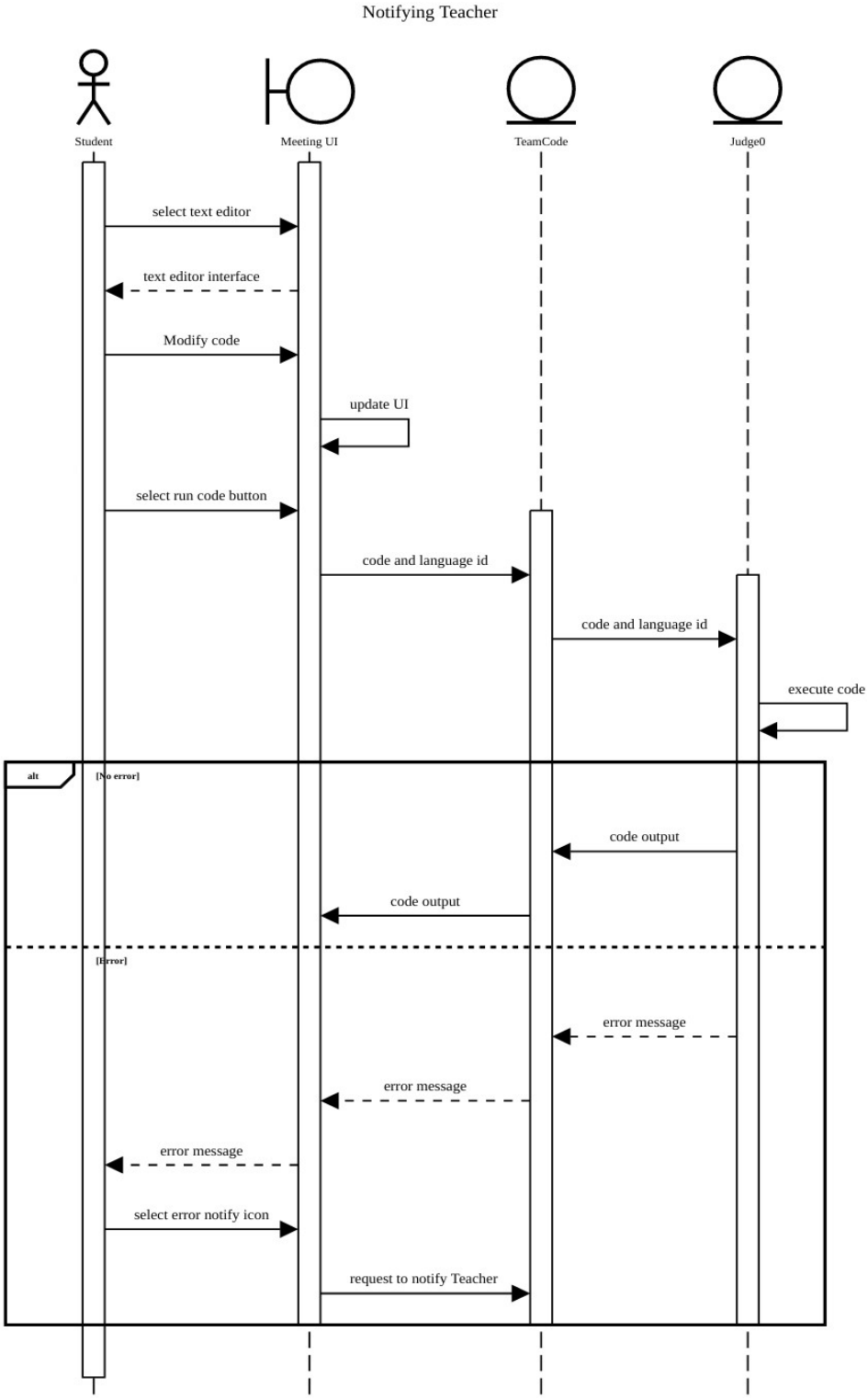

Figure 3.6.7: Sequence diagram for notifying errors during code execution

The steps through which a Student will be able to send notifications to the Teacher in TeamCode is shown in Figure 3.6.7. If an error occurs during execution of code, then the Student can raise an error notification to the Teacher in the particular room. Through notifications the Teacher will be able to give feedback to the Student and also correct the problem in real-time.

3.6.8 Respond to Student

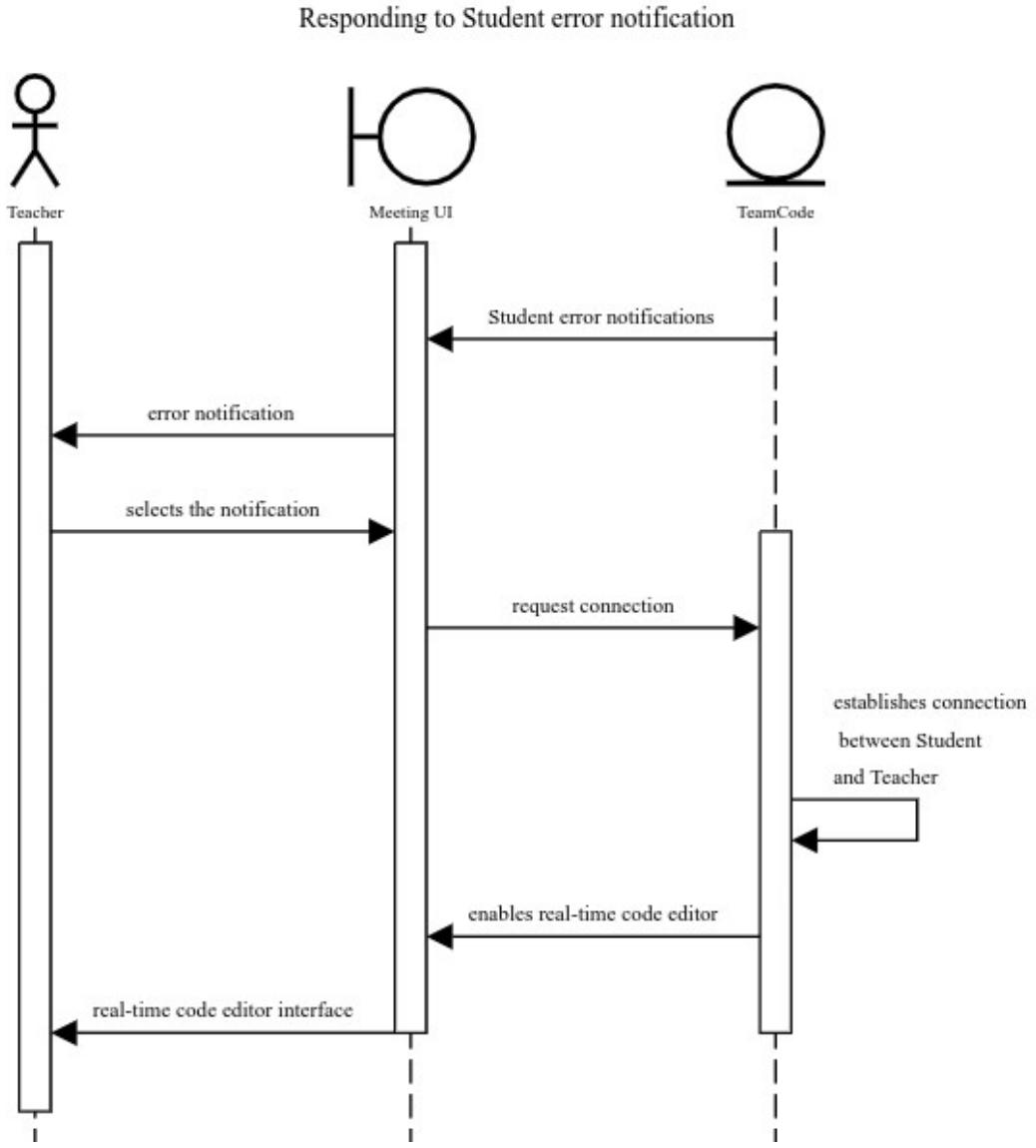

Figure 3.6.8: Sequence diagram of responding to student error notification

The figure above shows the steps a Teacher will take to respond to error notifications sent by Student in the room. The system will provide an interface to view notifications where the Teacher can select a notification to respond to and start a connection to the particular Student's text editor; allowing real-time edits.

3.7 Class diagram

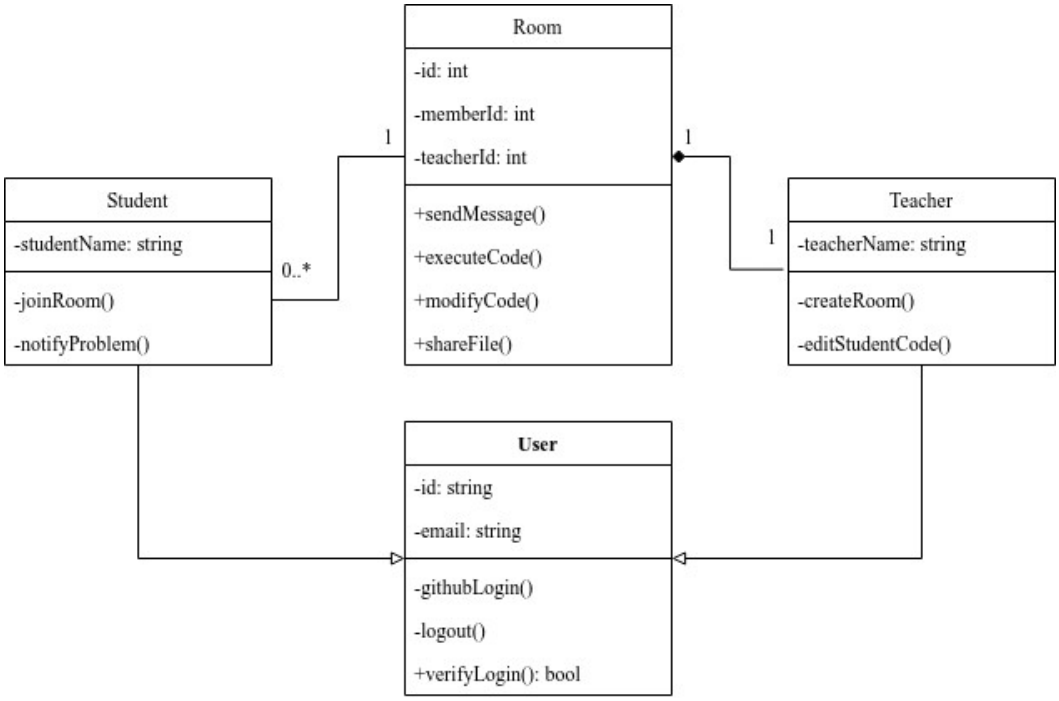

Figure 3.7.1: TeamCode Class diagram

The diagram presents different class entities in the system. The User entity is a generalised entity showing the common attributes and methods of Teacher and Student entities. The Room entity describes the meeting room which can be created by a Teacher so has a composition relation to the Teacher entity and can have zero or many Students in it. The Room entity provides methods to share code and files, send messages, modify code, and execute code.

3.8 Conclusion

The focus on collaboration in an online learning platform can be one of the best learning strategies. The use case diagrams have highlighted features that boosts collaboration among the users of TeamCode. The activity diagrams and sequence diagrams further explain the steps planned to achieve the said features of TeamCode whereas the class diagram provides insights to the structural model of the system.

Chapter 4
Interface

4.1 Introduction

This chapter explains the entire User Interface of the TeamCode web application along with a chronological order of the functionalities from the perspective of the two users – Teacher and Student.

4.2 Interface explain

4.2.1 Login page

Login page is the initial page of the application which allows users to login via their GitHub account information. After adding their GitHub credentials they are redirected to select their role of either a Student or Teacher.

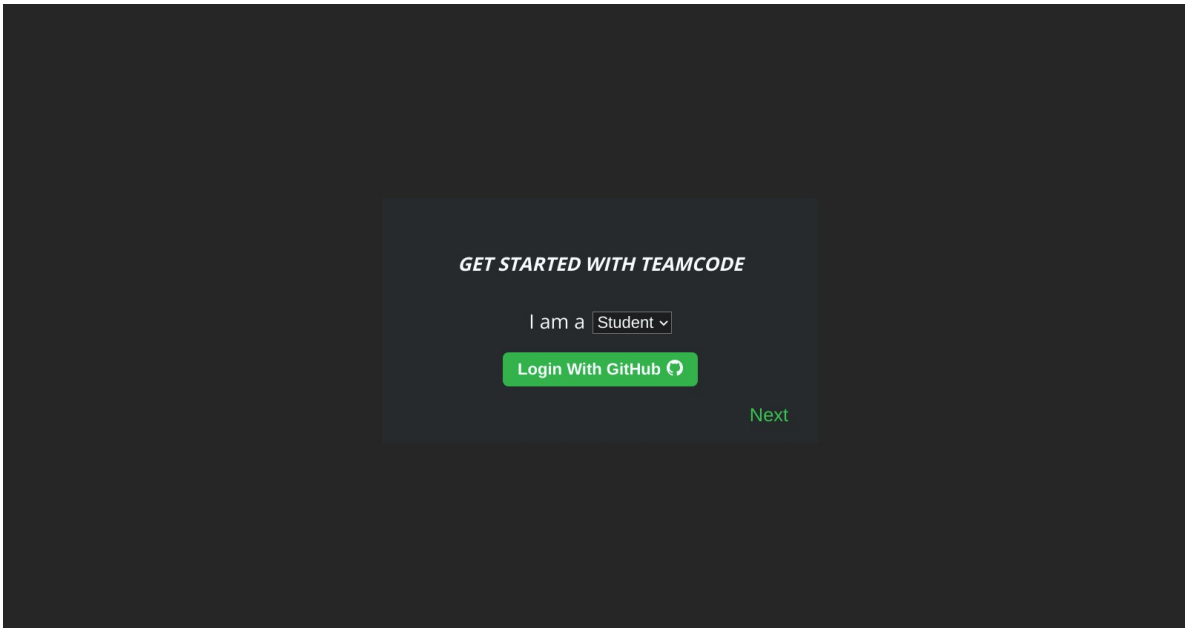

Figure 4.2.1: Login page

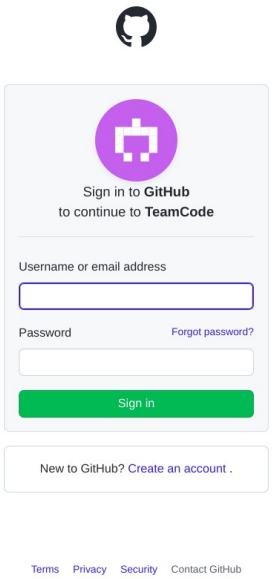

Figure 4.2.2: GitHub account verification page

4.2.2 Meeting details page

Meeting detail page is page after the login process. It shows the user's information and their respective roles. According to the user's role, the meeting details page showcases function of starting and ending meetings for Teachers, a join meeting with meeting URL instruction for Student and a logout button that is common for both roles.

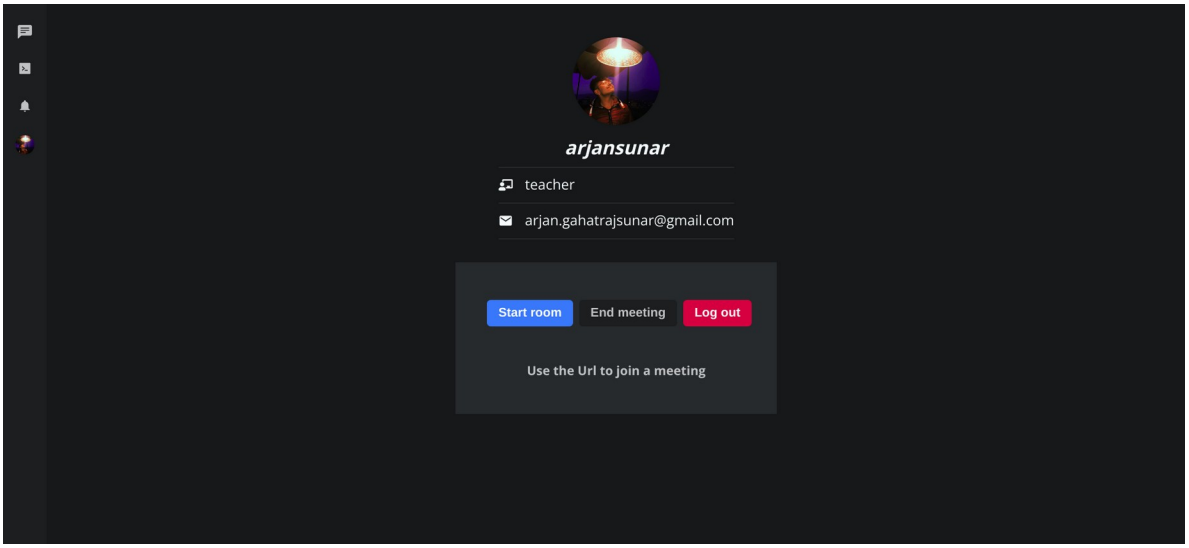

Figure 4.2.3: Meeting detail page for Teacher

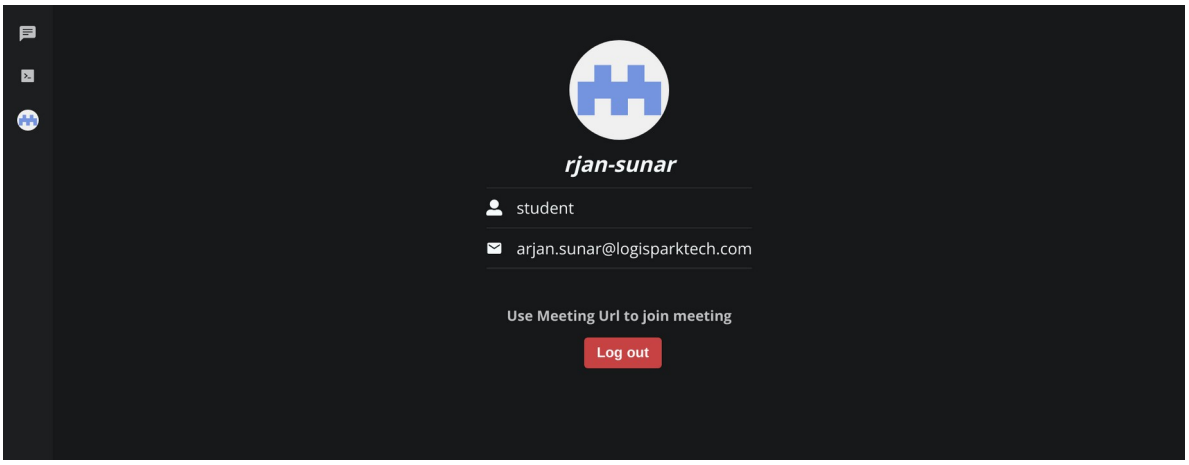

Figure 4.2.4: Meeting detail page for Student

4.2.3 Starting a meeting

Teacher is able to start a meeting after clicking the start a room button. After the action, a meeting URL gets added below the button group which enables user to copy the meeting link. Subsequently, end meeting button ends the meeting created by the Teacher and logout button logs the user out.

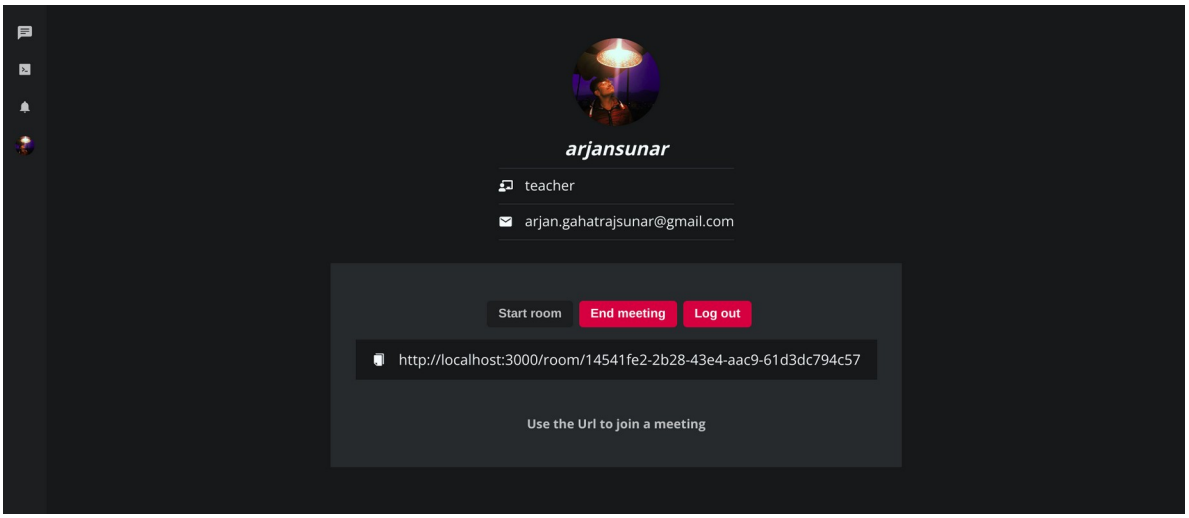

Figure 4.2.5: Starting a meeting

4.2.4 Meeting members page

Meeting members page shows all the users except the current user that have joined the current meeting. This page is shown after the meeting link is used.

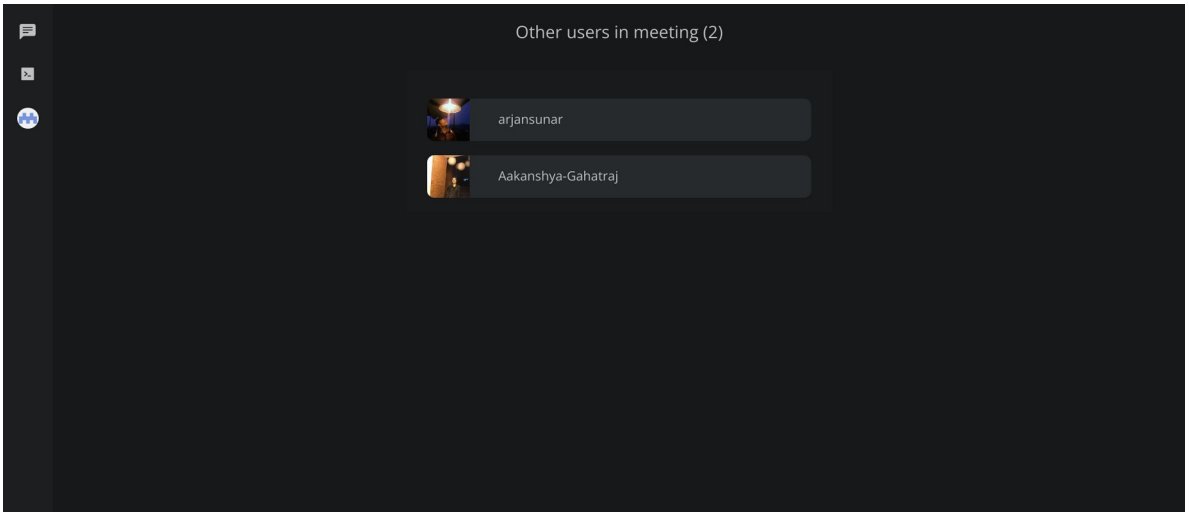

Figure 4.2.6: Meeting members page

4.2.5 Meeting page joining a meeting

After joining a meeting through the meeting URL, the meeting page gets additional UI elements for joining a video call.

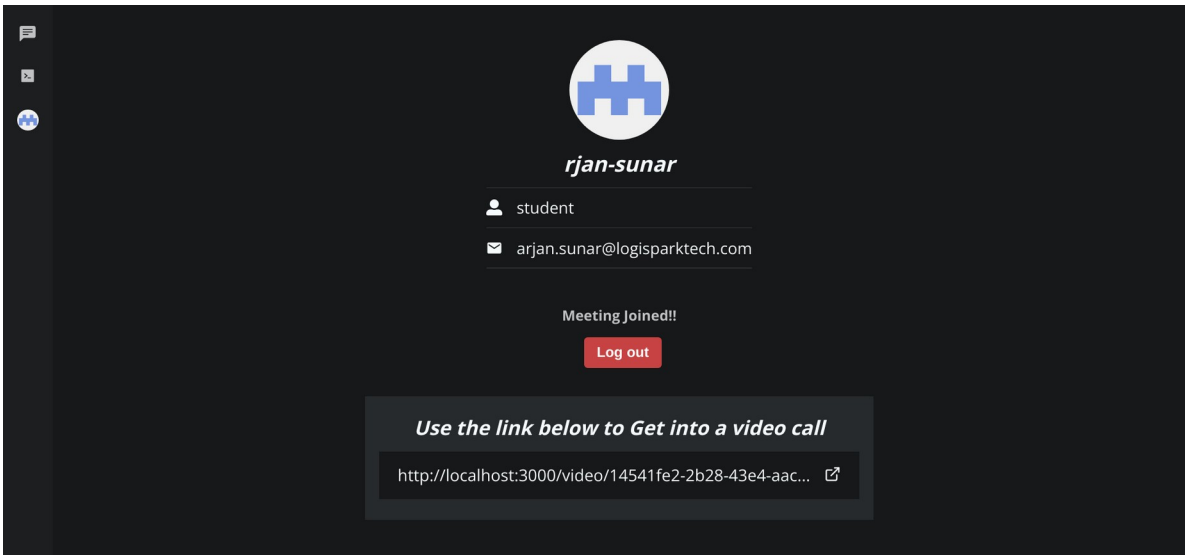

Figure 4.2.7: Meeting detail page after joining a meeting (Student view)

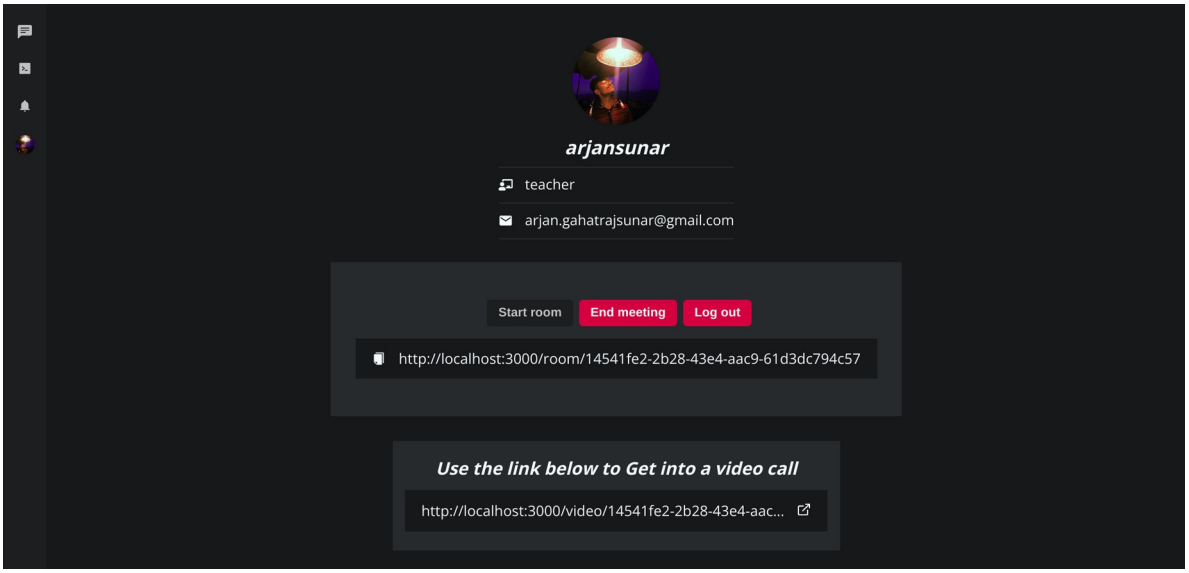

Figure 4.2.8: Meeting detail page after joining a meeting (Teacher view)

4.2.6 Group chat

This page provides a group chat interface for all the users in the meeting to communicate via text messages and also allows sharing of files in the form of pictures.

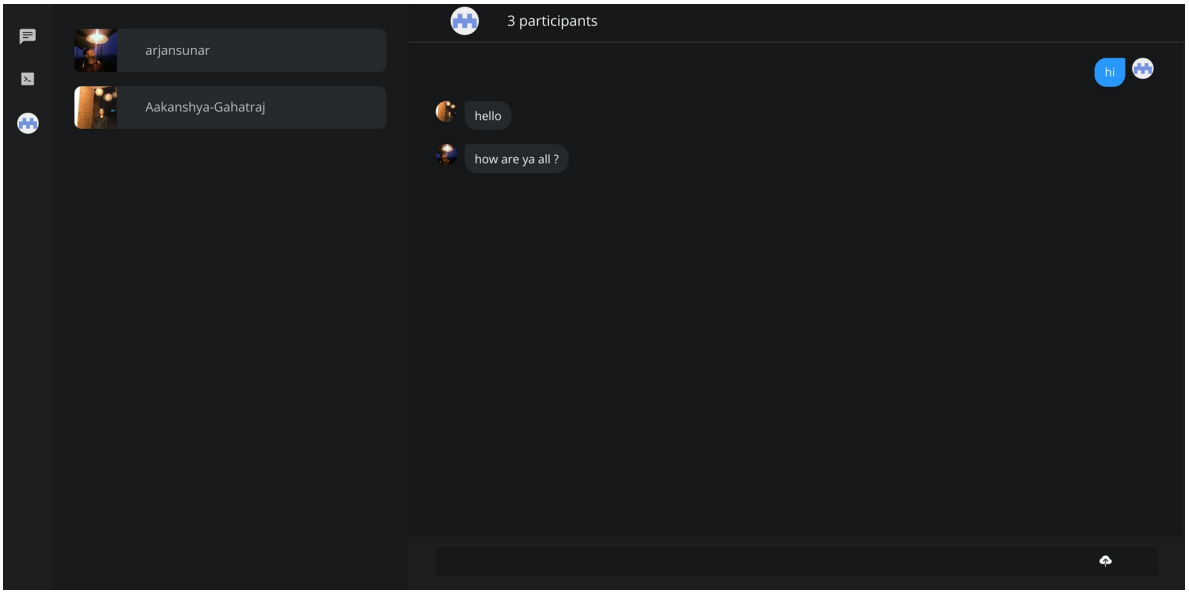

Figure 4.2.9: Group chat page

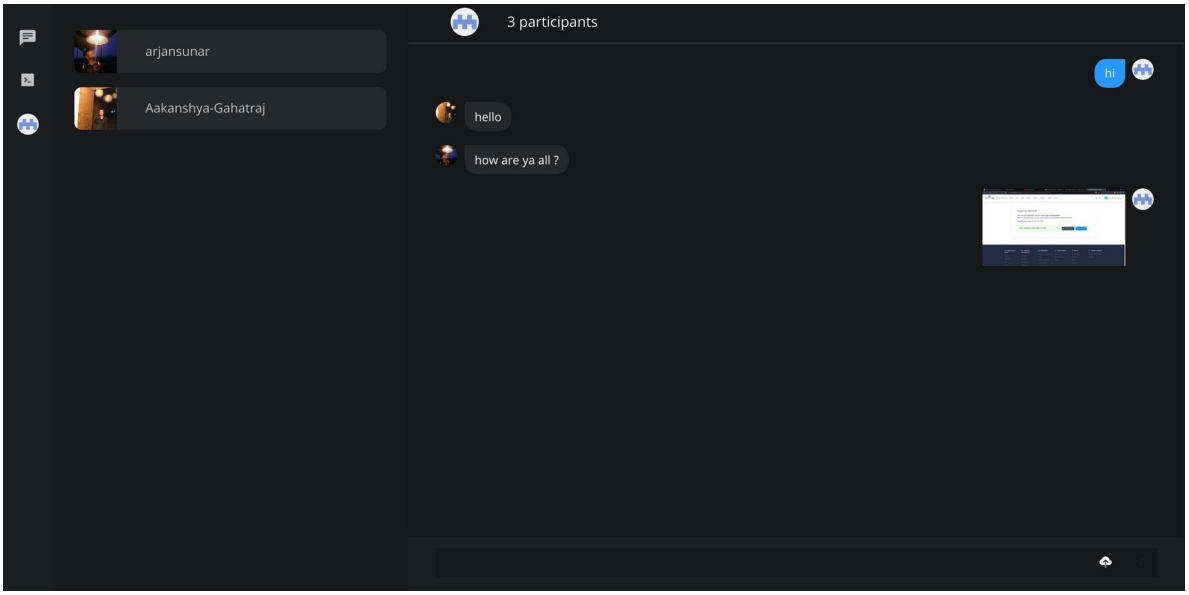

Figure 4.2.10: Sharing pictures in group chat

4.2.7 Code editor page

Code editor page provides a code editing interface to edit and execute code. The output of the code written gets shown at the bottom of the page. The page consists of a group of buttons with run button to execute the code, share to share the code to other users, read only button to share a read only version to other users and notify button to notify the Teacher of the meeting regarding any errors or problems while coding.

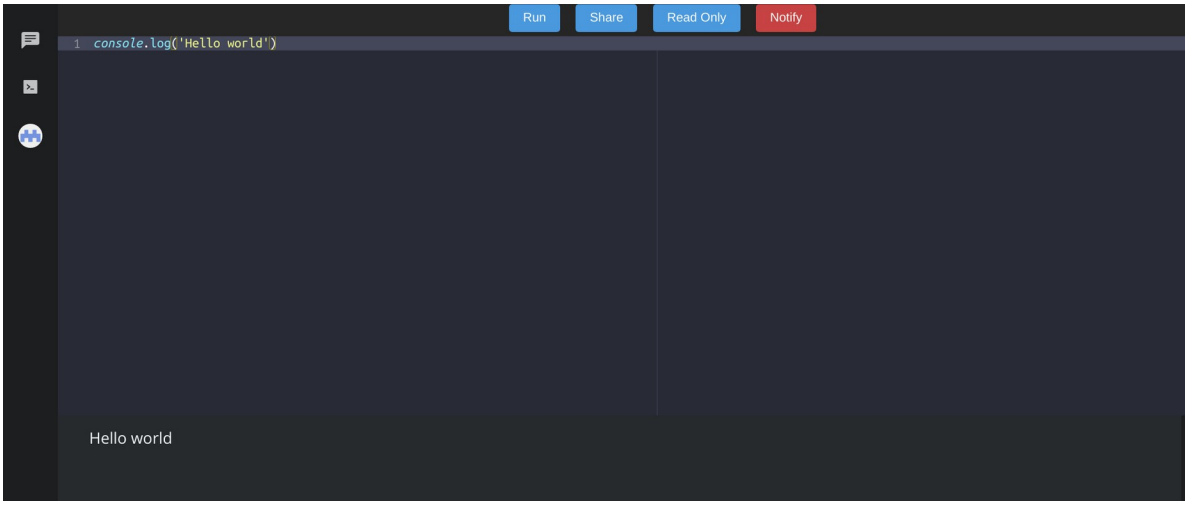

Figure 4.2.11: Code editor page

4.2.8 Share code from editor and choose users

After clicking on the share code button from the code editor page, the user is able to select the participants to whom he/she wants to share their code to. If the read only button is clicked then the exact same interface is created but the link created will not allow any edits to be made to the code shared.

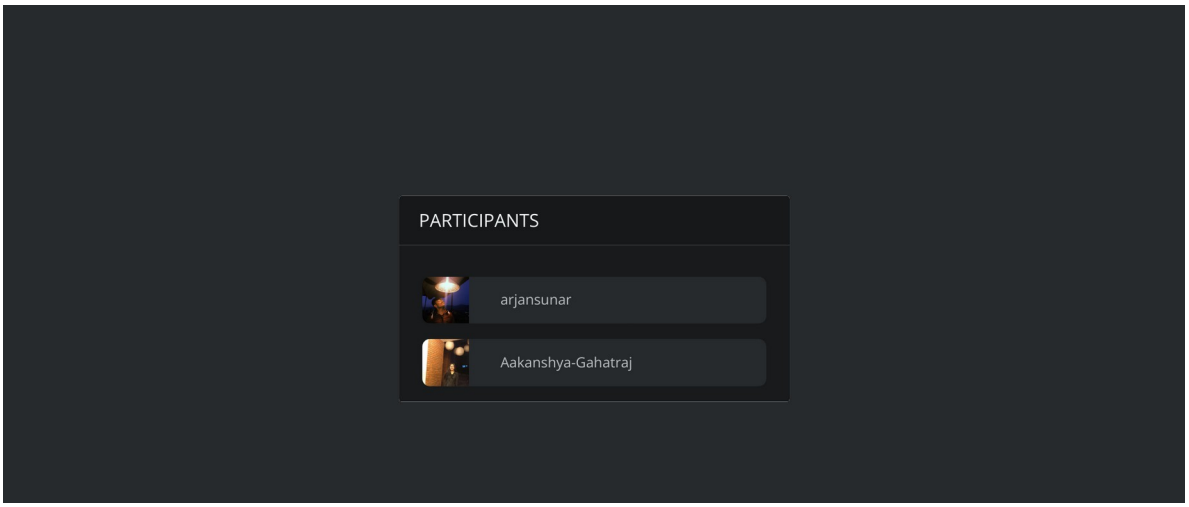

Figure 4.2.12: List of possible participants to share code with

After selecting the participant a share code link gets generated in the bottom which can be sent to the selected user and establish a real time code editing room.

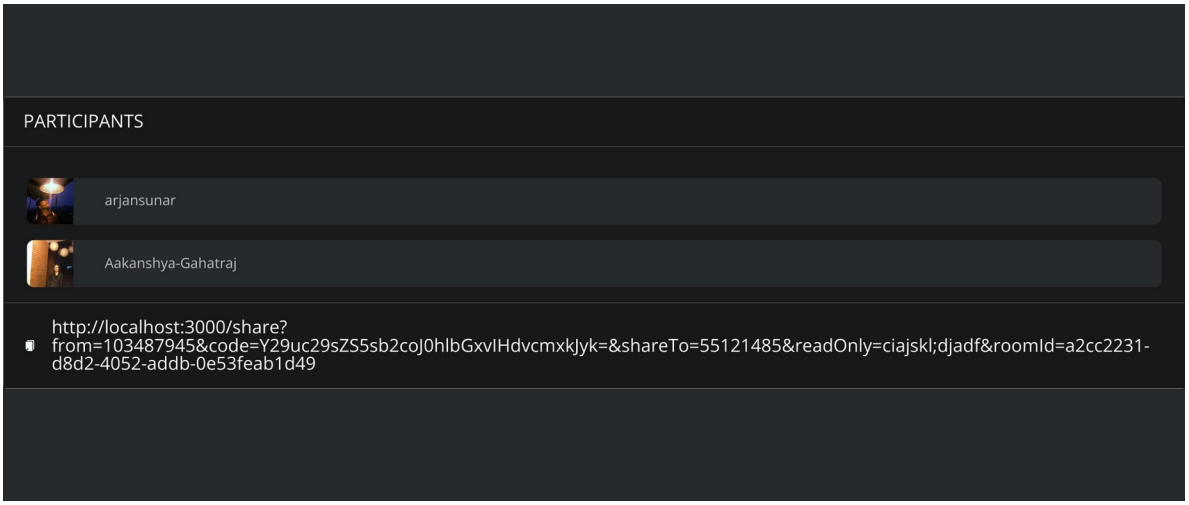

Figure 4.2.13: Share code link after selecting a participant

4.2.9 Notifying Teacher of a problem

Students can notify the Teacher of a problem by clicking the notify button in red colour. This brings in a alert box where students can add their message to the Teacher.

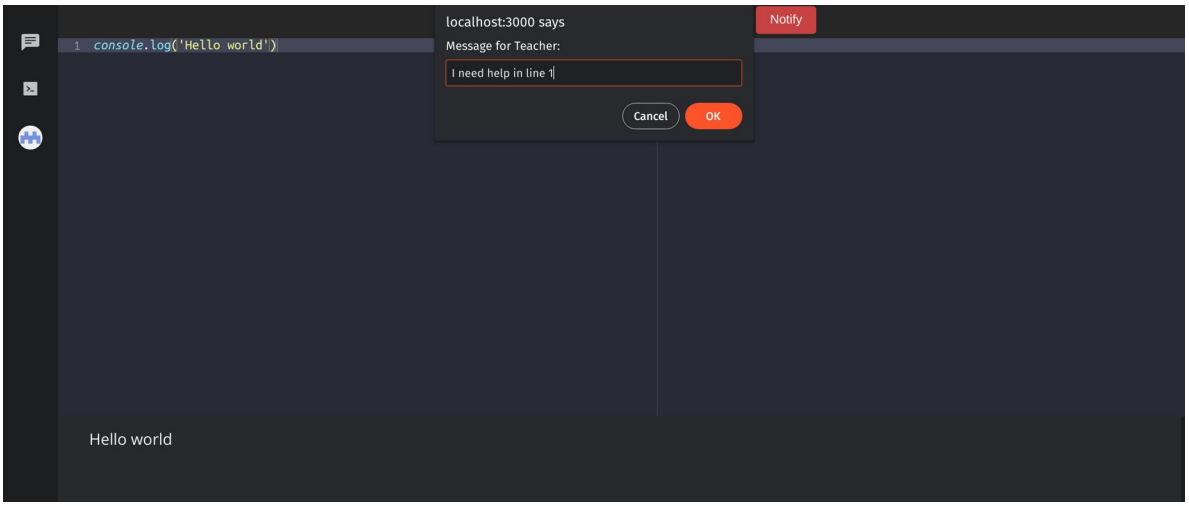

Figure 4.2.14: Notifying Teacher from code editor page

4.2.10 Notification page

Notification page shows all the requests from students regarding their problems in code. This page can only be seen by the Teacher of a meeting and only the Teacher of a particular meeting can access the notifications sent by Students of that meeting. Each notification consists of a message from the student and his/her code state in the form of a code share link. This link leads to a different page with real-time code editing enabled.

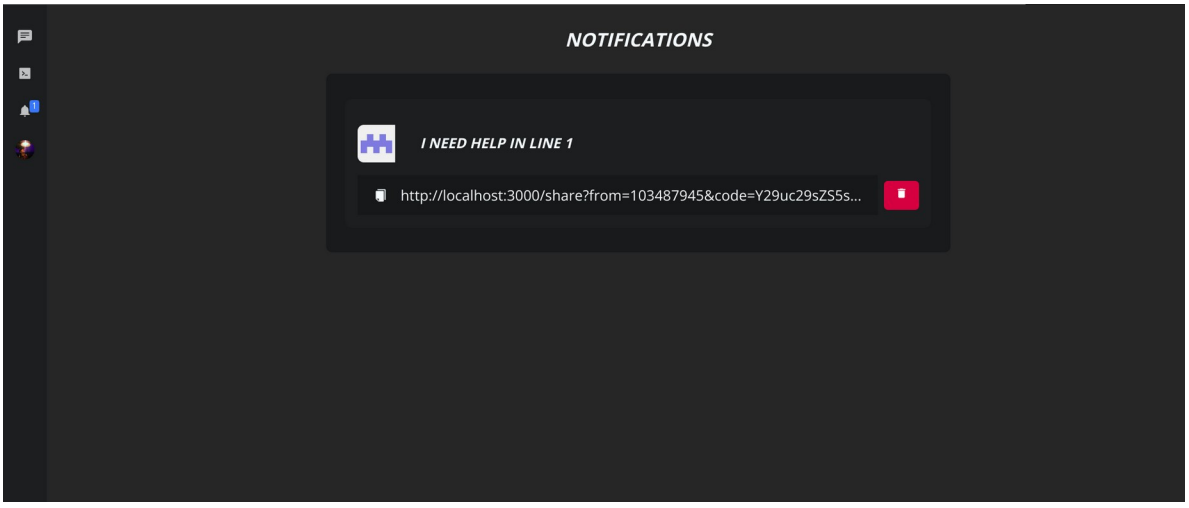

Figure 4.2.15: Notification page

4.2.11 Share code page

Share code page consists of a difference editor with the original code snippet of the left editor and a copy of the snippet on the right editor. The editor on the right enables editing and the edits made is broadcasted to users with access to the share code link and permission.

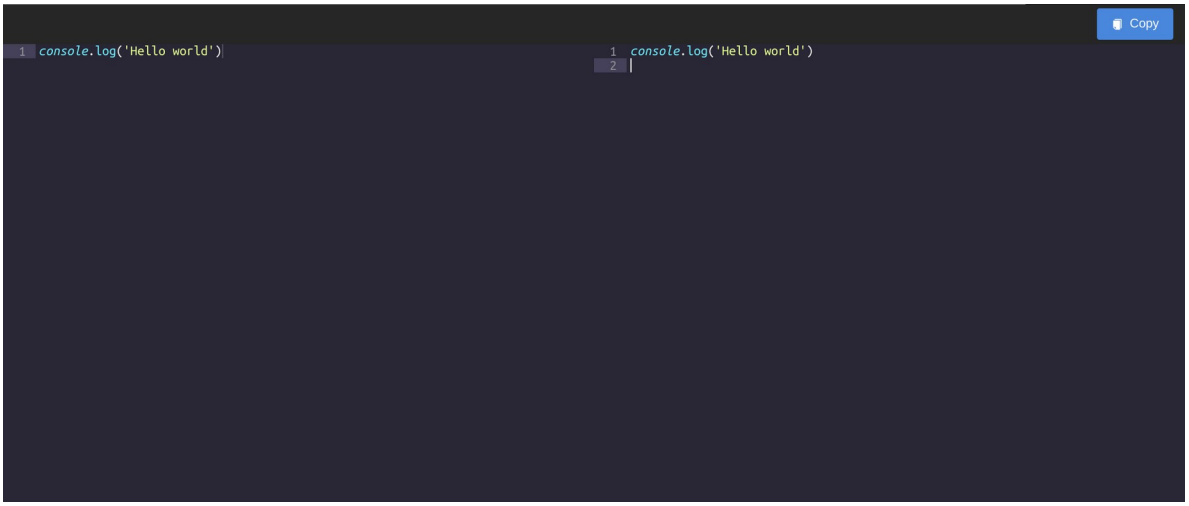

Figure 4.2.16: Share code page

4.2.12 Video call page

Video call page creates a group video call room where video stream is broadcasted from one user to another. This page is accessed via the group call link provided in the meeting detail page.

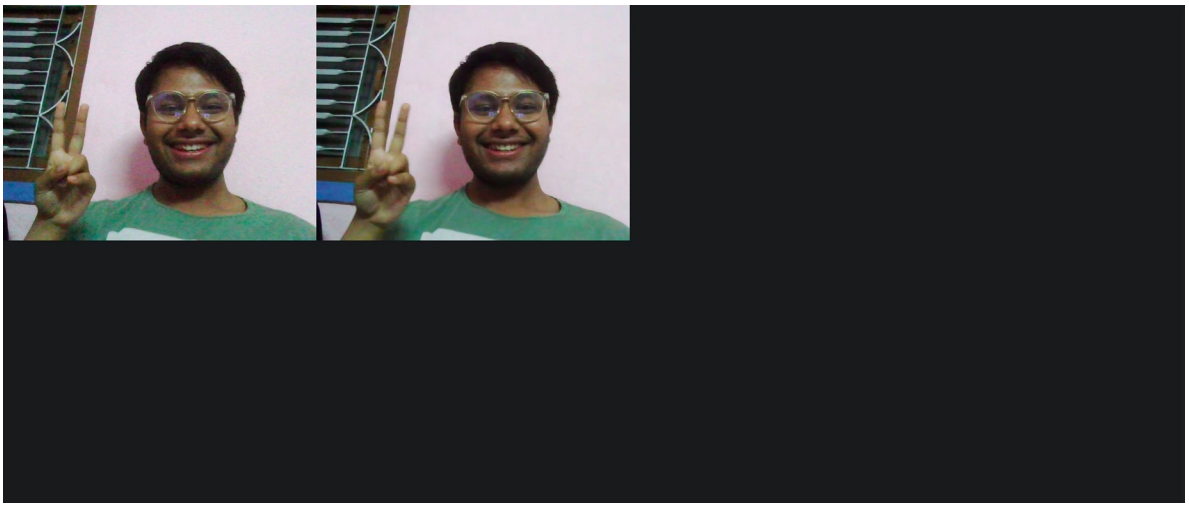

Figure 4.2.17: Video call page

4.3 Conclusion

The application interface includes elements for every objectives as mentioned in Chapter 1. The application has a simple design with dark theme with uniform design cues. The changes between Teacher and Student role has minimal UI changes and follows the same design elements within those changes.

Chapter 5

Testing and results

5.1 Introduction

Testing ensures that an application meets its requirements through a series of processes. The application is subjected to black box testing in order to validate that the application meets its objectives.

Software testing is an important step during the lifecycle of any software as it ensures that the software is ready to be distributed and also helps prevent frequent software crashes, bugs that lead to loss of confidence in the software.

- Black box testing

  Black box testing is a method of testing application without knowing about the internal workings of the application therefore referred to as a “Black box”. The testing is done by observing the outputs given by the application to particular inputs based on the requirements of the application.

5.2 Decision table

5.2.1 GitHub Login module

Test objective	Test data	Expected result	Actual result	Test status
Test if a user can access the system	Valid GitHub credentials	User must log into the system	User can login	Success

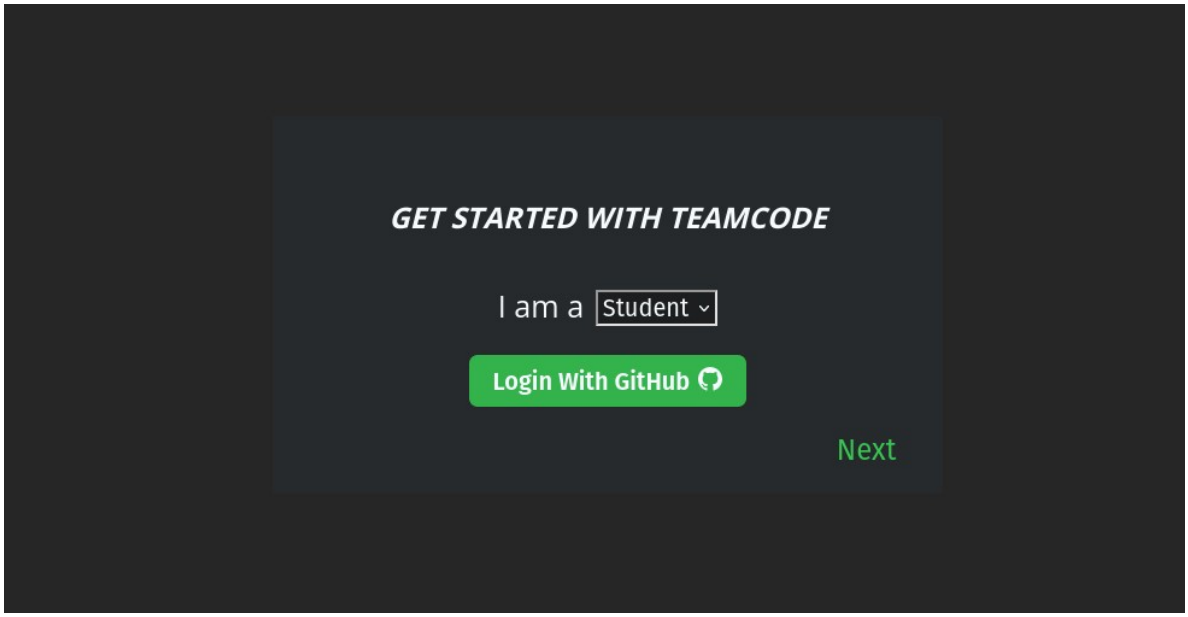

Figure 5.2.1: Login page

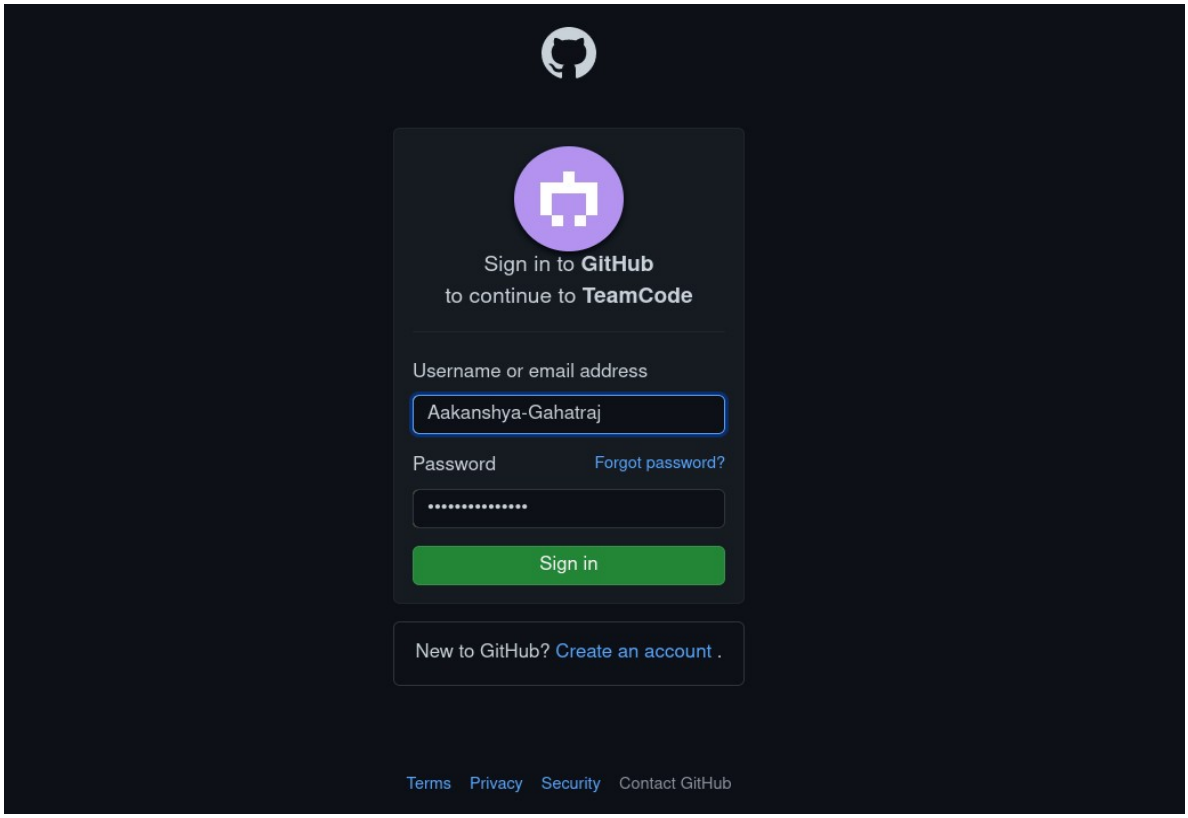

Figure 5.2.2: Redirect to GitHub user verification page

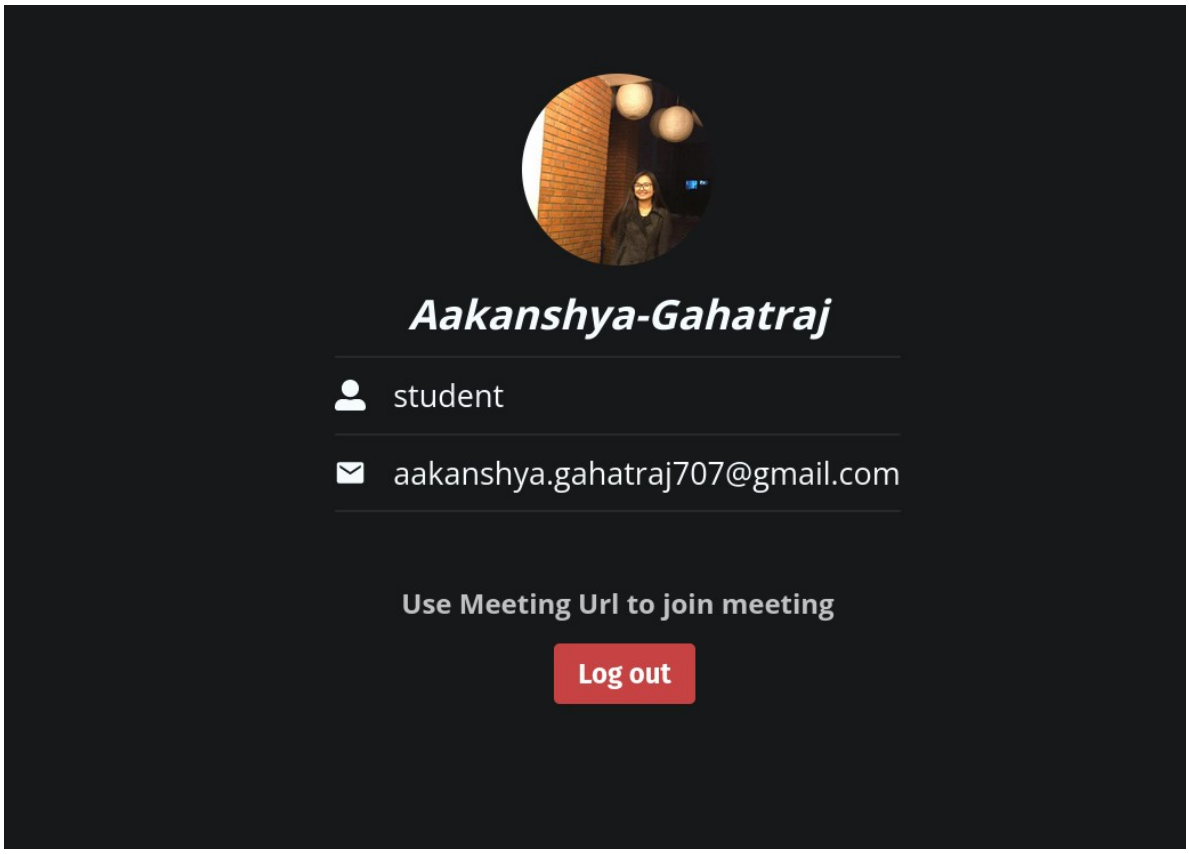

Figure 5.2.3: Successful login to the system

5.2.2 Execute code

Test objective	Test data	Expected result	Actual result	Test status
Test if users can execute the code they have written	User code in code editor page code: console.log(‘Hello world’)	Output must be the string Hello world	Hello world is shown as output	Success

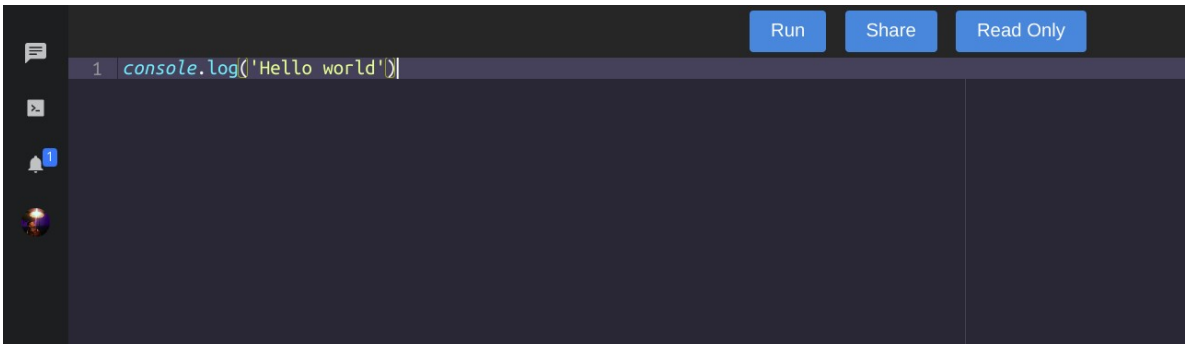

Figure 5.2.4: Input to test code execution

Figure 5.2.5: Successful execution of code

5.2.3 Share code with only chosen users

Test objective	Test data	Expected result	Actual result	Test status
Test if only chosen users can access shared code	Users clicks share code and selects a participant	Only the chosen participant should access the shared code	Only the chosen participant accesses the code	Success

Figure 5.2.6: Code to share

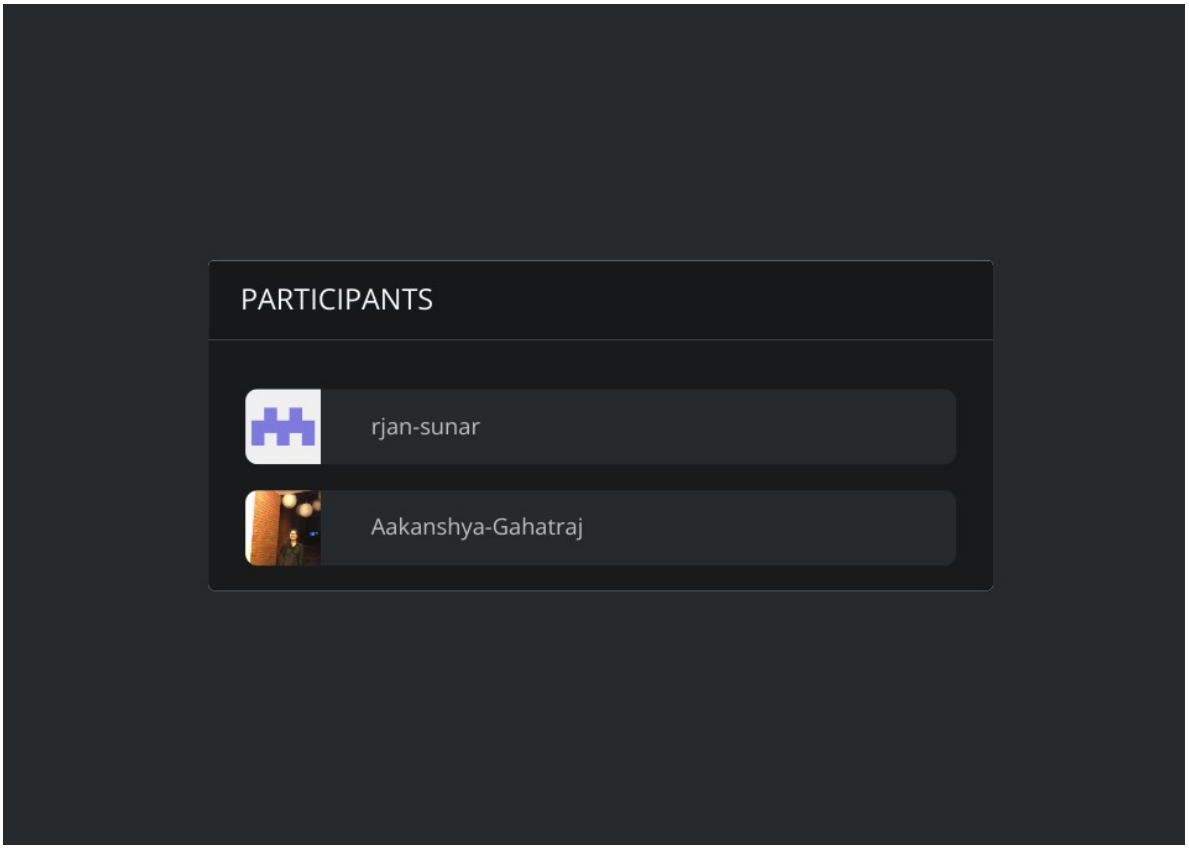

Figure 5.2.7: Participants list after user clicks share button

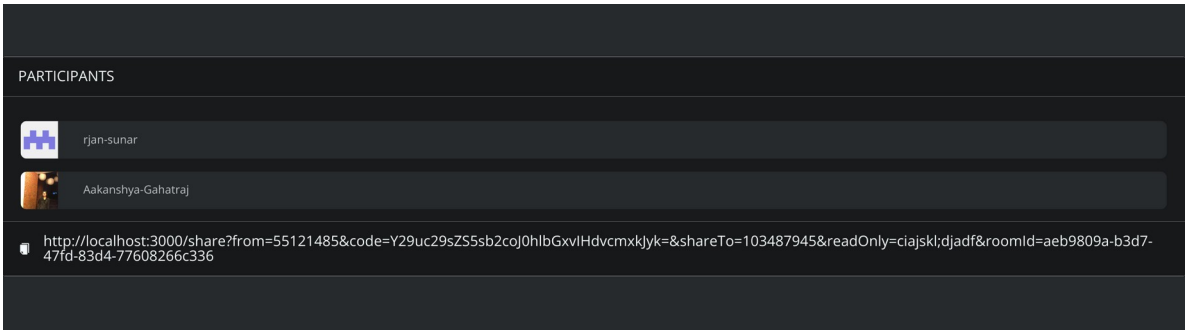

Figure 5.2.8: Share link generated according to selected participant

5.2.4 Broadcast code changes

Test objective	Test data	Expected result	Actual result	Test status
Check if code changes are shared for real-time code editing	A user sends share code link to other with permission	Changes made by other user and the user is reflected on both user's difference editors	The changes are updated in both user's difference editor	Success

Figure 5.2.9: Selected participant can access the shared code

Figure 5.2.10: Changes made by the user

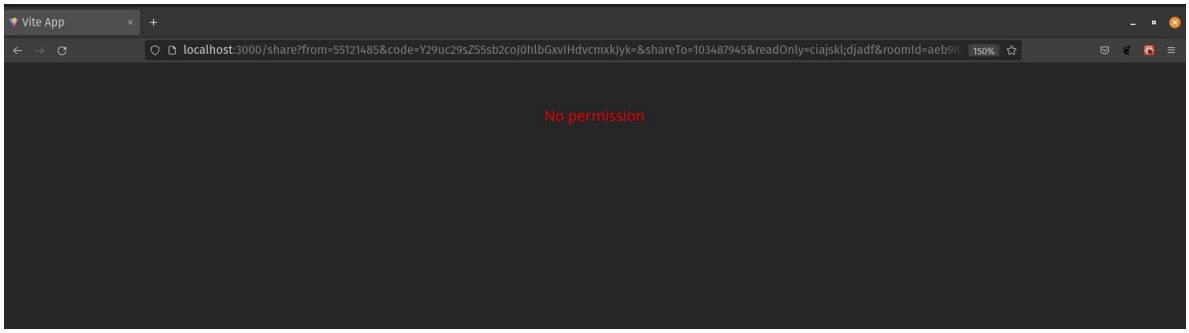

Figure 5.2.11: Error page for users except selected participant

Figure 5.2.12: Changes received by the chosen receiver

Figure 5.2.13: Changes made by the receiver is reflected on user's difference editor

5.2.5 Read only code share

Test objective	Test data	Expected result	Actual result	Test status
Test if user can share code to others with read only permission	User click read only button in code editor page	Should generate a link with difference editor that can't be edited	Generates a link with difference editor that can't be edited	Success

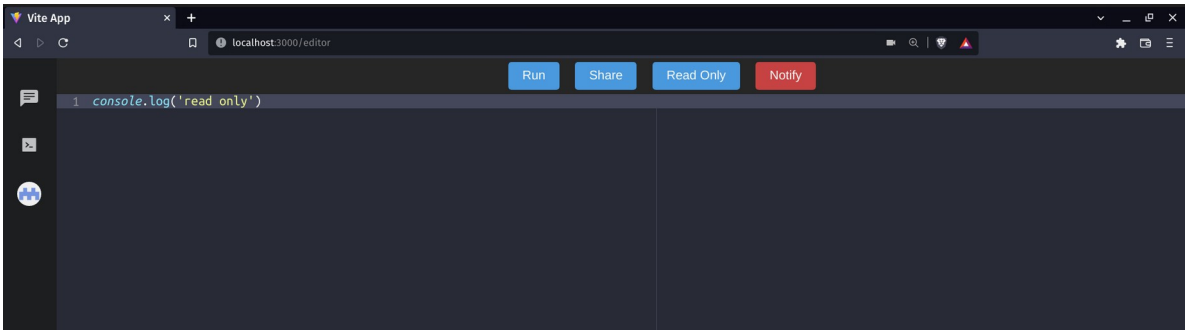

Figure 5.2.14: Code to share

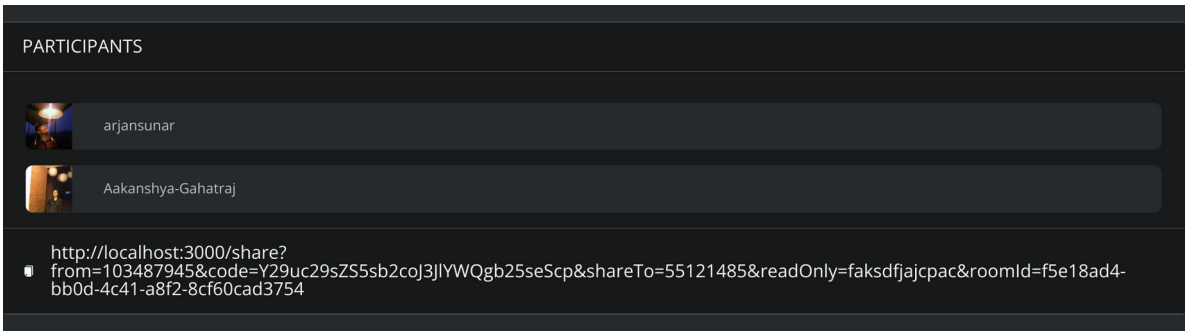

Figure 5.2.15: Read only link

Figure 5.2.16: Unable to make changes to read only shared code

5.2.6 Notify teacher

Test objective	Test data	Expected result	Actual result	Test status
Test if Student's notification is reached to Teacher	Student clicks notify button in code editor page	Teacher should receive a notification	Teacher receives a notification	Success

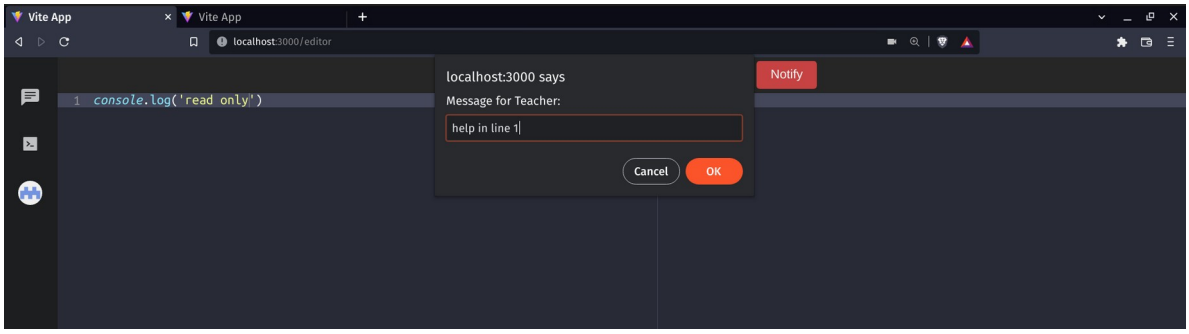

Figure 5.2.17: User clicks notify button and adds a message

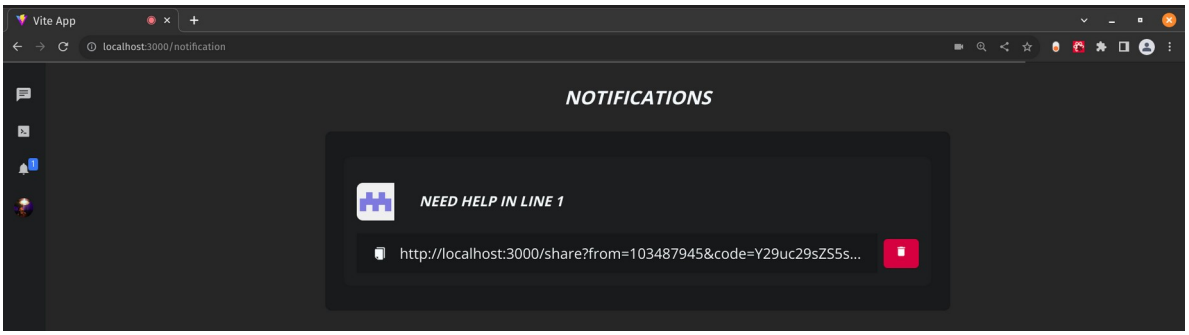

Figure 5.2.18: Teacher successfully receives the notification

5.2.7 Send chat message and pictures

Test objective	Test data	Expected result	Actual result	Test status
Test if users can send messages and pictures	Users message and a screenshot as a picture	Share message and picture to all the users in the group chat	Message and picture is broadcasted	Success

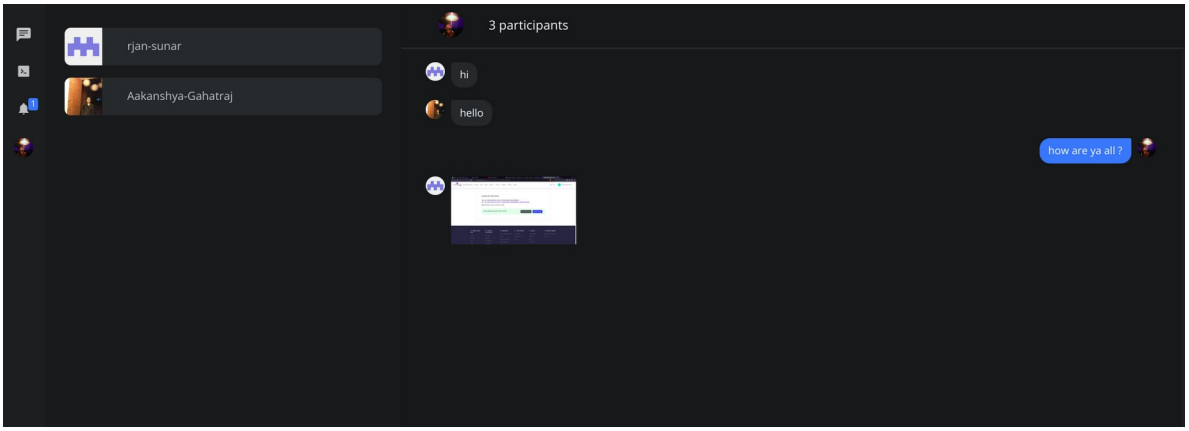

Figure 5.2.19: Messages and pictures sent in chat

5.2.8 Video call

Test objective	Test data	Expected result	Actual result	Test status
Conducting a video call with different users	Opening video call link from different browsers to emulate different users	Users must be able to stream each others video data	Multiple browser aren't able to access a singular web cam	Failure

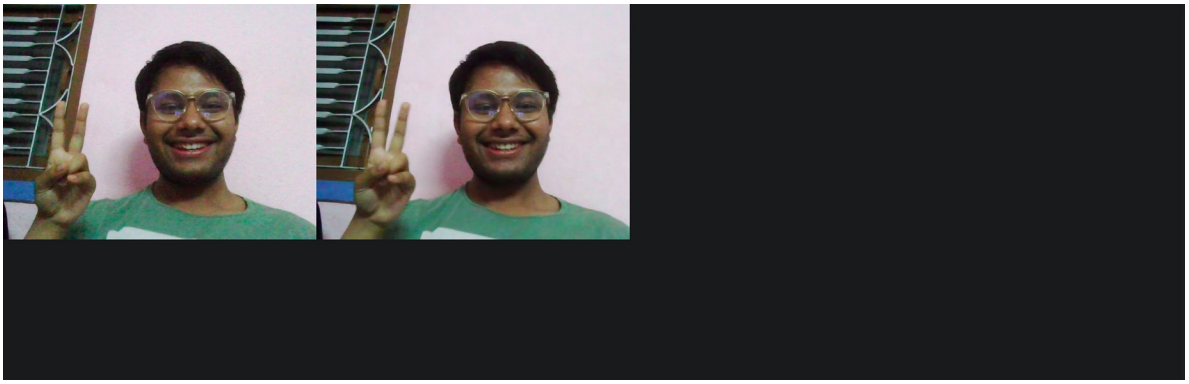

Figure 5.2.20: Video call made by duplicating browser tab to emulate multiple users

5.2.9 End meeting by Teacher

Test objective	Test data	Expected result	Actual result	Test status
Check if a Teacher is able to end a meeting	Teacher clicks end meeting button	Meeting must end for all participants of the meeting	Meeting ends for all participants	Success

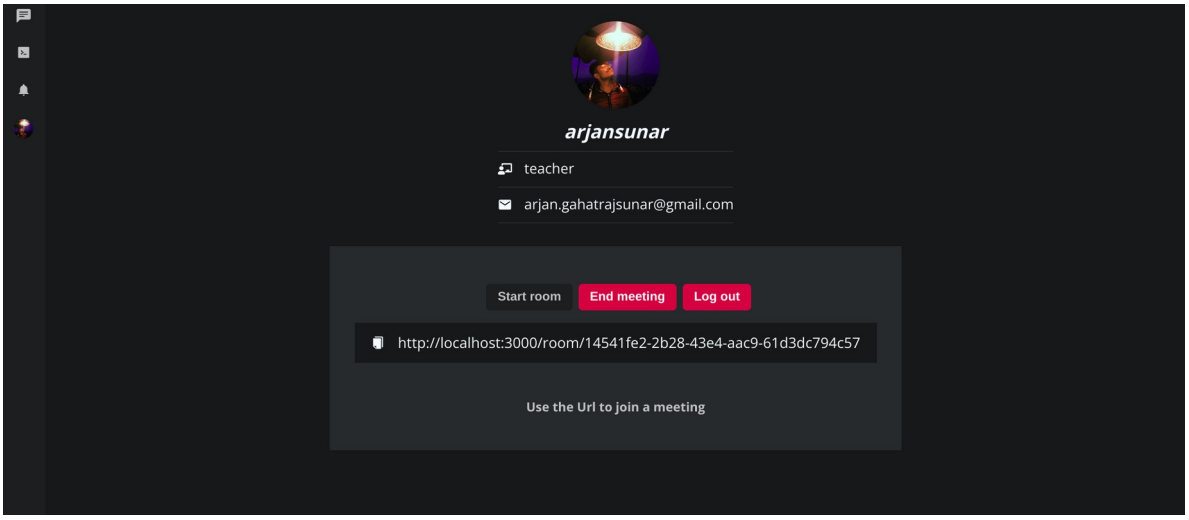

Figure 5.2.21: Meeting page with end meeting button

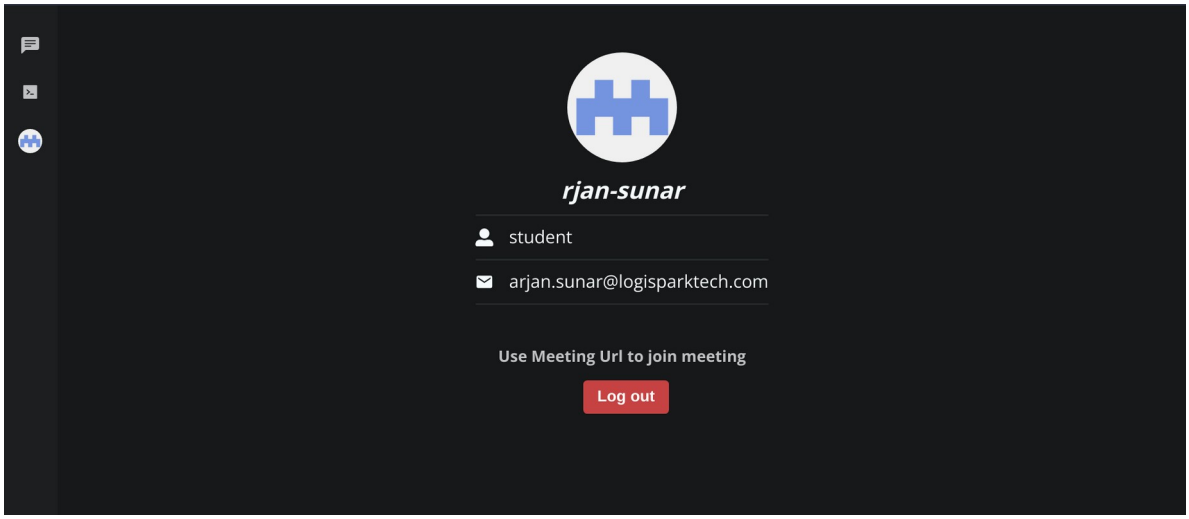

Figure 5.2.22: Student is removed from the meeting

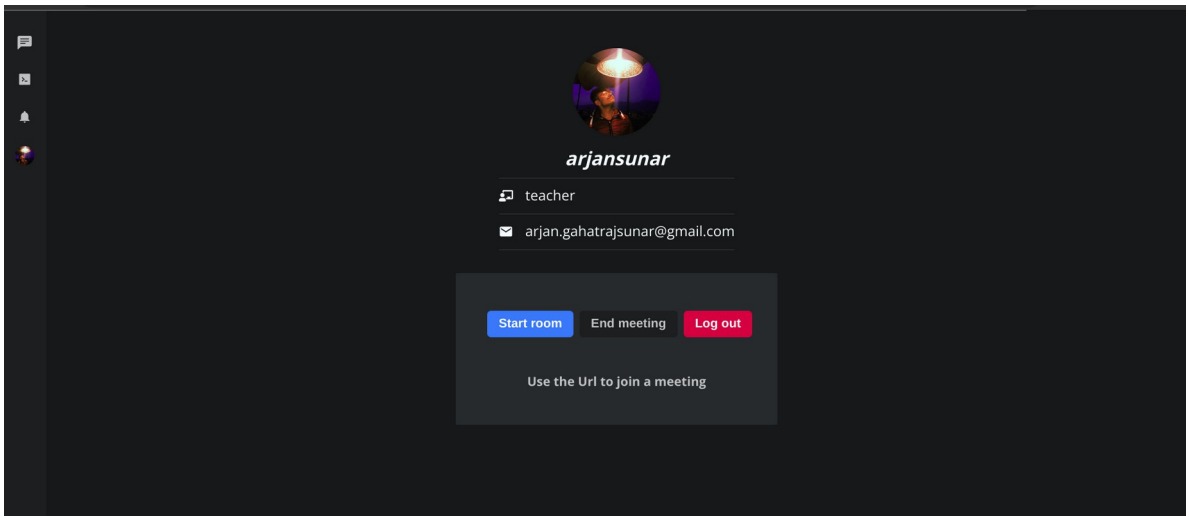

Figure 5.2.23: Teacher's meeting page UI changes as meeting has ended

5.3 Conclusion

The web application is tested to meet all its stated objectives. Most modules perform as expected as shown by black box testing but however some bugs might have been not yet been found and will have to be solved through future improvements. The video call module cannot be showcased according to expected result but is a case that can be solved through deployment of the application. The testing process with its outcome have been mentioned and system breaking errors have been solved.

Chapter 6
Conclusion and Suggestions

6.1 Introduction

After the development and testing of an application it goes to the maintenance phase where the application is constantly improved by addressing issues and bugs. The errors and bugs found during the testing phase make the application more secure by solving them then and there. Through solution of bugs can the application be more resilient and uphold to the expectation of the developer and the end user.

6.2 Challenges

- Documentation phase: Lack of proficiency using LibreOffice writer as the documentation writing tool made it difficult to match the report guidelines. The solution was to find documentation and online forms for help.
- Implementation process:
  - Lack of compatibility between the libraries initially used made it harder to implement features. The solution was to find other alternatives that better worked with the libraries intended to be used.
  - Lack of knowledge in building applications with the programming language (TypeScript) made it harder to develop features faster. The solution was to read through the documentation and tutorials videos to get familiar with the language.

6.3 Advantages of the system

- The system is easy to use and facilitates multiple users.
- The system helps to increase interactivity between instructors and learners.
- The system allows to learn and execute code without the need for setting up an environment.

6.4 Limitations

- The application User Interface is not mobile friendly.
- Internet access is a limitation to the system as it is a web application and uses UDP protocols and sockets to form the communication bridge between instructors and learners.

- WebRTC is not supported in browsers such as Internet Explorer (IE), Opera Mini and UC browser for Android.

6.5 Future scope

- Support for mobile users.
- Addition of multiple programming languages.
- Addition of meeting scheduler.
- Support for web development classes with HTML, CSS and JS editors.

6.6 Conclusion

This web application will enable teachers to start programming classes online with an emphasis on interaction with students. It will be subjected to future enhancement and provide additional features. Lastly, the web application fulfils the stated objectives of the project.

References:

Berry, S. (2017). *Exploring Community in an Online Doctoral Program: A Digital Case Study* [Doctoral Dissertation]. University of Southern California.

Berry, S. (2019). Teaching to Connect: Community-Building Strategies for the Virtual Classroom. *Online Learning*, *23*(1), 164–183.

Codewars: Achieve mastery through coding challenge. (2022, January 24). Codewars. https://www.codewars.com

CodingBat Java. (2022, January 24). https://codingbat.com/java

Deepika, V., Soundariya, K., Karthikeyan, K., & Kalaiselvan, G. (2021). 'Learning from home': Role of e-learning methodologies and tools during novel coronavirus pandemic outbreak. *Postgraduate Medical Journal*, *97*(1151), 590. https://doi.org/10.1136/postgradmedj-2020-137989

Driscoll, A., Jicha, K., Hunt, A. N., Tichavsky, L., & Thompson, G. (2012). Can Online Courses Deliver In-class Results?: A Comparison of Student Performance and Satisfaction in an Online versus a Face-to-face Introductory Sociology Course. *Teaching Sociology*, *40*(4), 312–331. https://doi.org/10.1177/0092055X12446624

Fisher, M., & Coleman, B. (2001). Collaborative online learning in virtual discussions. *Journal of Educational Technology Systems*, *30*(1), 3–17.

Fojtik, R. (2017). Issues in Distance Learning of Programming. *New Trends and Issues Proceedings on Humanities and Social Sciences*, *3*, 48–54. https://doi.org/10.18844/gjhss.v3i3.1522

Google Docs. (2022, January 24). https://docs.google.com

Google Hangouts. (2022, January 24). https://hangouts.google.com/

Google Meet. (2022, January 24). https://meet.google.com/

HackerRank. (2022, January 24). HackerRank. https://www.hackerrank.com/dashboard

Kohnke, L., & Moorhouse, B. L. (2020). Facilitating Synchronous Online Language Learning through Zoom. *RELC Journal*, *0033688220937235*.

Laal, M. (2013). Collaborative Learning; Elements. *Procedia - Social and Behavioral Sciences*, *83*, 814–818. https://doi.org/10.1016/j.sbspro.2013.06.153

Laal, M., & Laal, M. (2012). Collaborative learning: What is it? *Procedia - Social and Behavioral Sciences*, *31*, 491–495. https://doi.org/10.1016/j.sbspro.2011.12.092

LeetCode—The World's Leading Online Programming Learning Platform. (2022, January 24). https://leetcode.com/

Martin, J. (2019). Building Relationships and Increasing Engagement in the Virtual Classroom: Practical Tools for the Online Instructor. *Journal of Educators Online*, *16*(1), 8.

Mihai, A. (2014). The Virtual Classroom: Teaching European Studies Through Webinars. *European Political Science*, *13*(1), 4–11. https://doi.org/10.1057/eps.2013.31

Moorhouse, B. L. (2020). Adaptations to a face-to-face initial teacher education course 'forced' online due to the COVID-19 pandemic. *Journal of Education for Teaching*, *46*(4), 609–611. https://doi.org/10.1080/02607476.2020.1755205

Nokes-Malach, T. J., Richey, J. E., & Gadgil, S. (2015). When Is It Better to Learn Together? Insights from Research on Collaborative Learning. *Educational Psychology Review*, *27*(4), 645–656. https://doi.org/10.1007/s10648-015-9312-8

Pal, K. B., Basnet, B. B., Pant, R. R., Bishwakarma, K., Kafle, K., Dhami, N., Sharma, M. L., Thapa, L. B., Bhattarai, B., & Bhatta, Y. R. (2021). Education system of Nepal: Impacts and future perspectives of COVID-19 pandemic. *Heliyon*, *7*(9), e08014. https://doi.org/10.1016/j.heliyon.2021.e08014

Puranik, D. G., Feiock, D. C., & Hill, J. H. (2013). Real-Time Monitoring using AJAX and WebSockets. *2013 20th IEEE International Conference and Workshops on Engineering of Computer Based Systems (ECBS)*. https://doi.org/10.1109/ECBS.2013.10

Pyhältö, K., Stubb, J., & Lonka, K. (2009). Developing scholarly communities as learning environments for doctoral students. *International Journal for Academic Development*, *14*(3), 221–232. https://doi.org/10.1080/13601440903106551

Rehman, S. ur, & Khan, M. U. (2016). Security and Reliability Requirements for a Virtual Classroom. *Procedia Computer Science*, *94*, 447–452. https://doi.org/10.1016/j.procs.2016.08.069

Repman, J., Zinskie, C., & Carlson, R. D. (2005). Effective Use of CMC Tools in Interactive Online Learning. *Computers in the Schools*, *22*(1–2), 57–69. https://doi.org/10.1300/J025v22n01_06

Rovai, A. P. (2003). In search of higher persistence rates in distance education online programs. *The Internet and Higher Education*, *6*(1), 1–16. https://doi.org/10.1016/S1096-7516(02)00158-6

Skype | Stay connected with free video calls worldwide. (2022, January 24). https://www.skype.com/en//

Stubb, J., Pyhältö, K., & Lonka, K. (2011). Balancing between inspiration and exhaustion: PhD students' experienced socio-psychological well-being. *Studies in Continuing Education*, *33*(1), 33–50. https://doi.org/10.1080/0158037X.2010.515572

Video Conferencing, Cloud Phone, Webinars, Chat, Virtual Events | Zoom. (2022, January 24). https://zoom.us/

Video Conferencing, Meetings, Calling | Microsoft Teams. (2022, January 24). https://www.microsoft.com/en-us/microsoft-teams/group-chat-software

Wasik, S., Antczak, M., Badura, J., Laskowski, A., & Sternal, T. (2018). A Survey on Online Judge Systems and Their Applications. *ACM Computing Surveys (CSUR)*, *51*(1), 1–34.

"web rtc " | Can I use... Support tables for HTML5, CSS3, etc. (2021, December 17). https://caniuse.com/?search=web%20rtc%20

coding platforms as additional distance tools in programming education. *Journal of Physics: Conference Series*, *1840*(1), 16. https://doi.org/10.1088/1742-6596/1840/1/012029