

Title Page

**TeamCode: 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 through a flag system. The flag system will be a method to notify the problems of a learner to the instructor and allow the instructor to solve them with real-time editing of the code. 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 Monaco 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 thank my supervisor Mr. Akash Deo for his time and guidance 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 are required to submit their Chapter Assignments to their Supervisor.
- Students should ensure they are aware of how and when to submit their assignment.
- Students who miss the deadlines because of illness must present an original (not a photocopy) medical certificate to the lecturer on the first day back in classes.
- Students who request compassionate consideration must provide then funeral announcement, obituary, doctor's certificate or death certificate, or relevant documentation.

Student Declaration

- I declare that this project is my own work and does not involved plagiarism or collusion and shall be graded as Failed if proven.
- It does not contain my own previous work without this being stated.
- I am aware that any violation of these rules will be considered cheating and consider as Failed.
- I am aware that all material submission will be accepted after the deadline but no marks will be given.
- I have made a photocopy and electronic copy of my project, which I can produce if the original is lost for any reason.
- 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 project and the research contained in it are the result of my efforts, and that any statements or material derived from the work of others is properly referenced.

Date:

Arjan Gahatraj Sunar
041902900012

Table of contents

Title Page.....	i
Abstract.....	ii
Acknowledgement.....	iii
Approval.....	iv
Agreement letter.....	v
Declaration.....	vi
Table of contents.....	vii
Table of figures.....	ix
List of tables.....	x
List of abbreviations.....	xi
1 Chapter 1: Introduction.....	1
1.1 Introduction.....	1
1.2 Problem Statement.....	2
1.3 Objectives.....	2
1.4 Scope.....	3
1.4.1 User scope.....	3
1.4.2 System Scope.....	3
1.5 Constraints.....	3
1.6 Methodology.....	3
1.7 Development tools.....	4
1.8 Structure of Report.....	4
1.9 Conclusion.....	5
2 Chapter 2: Literature review.....	6
2.1 Background study.....	6
2.1.1 Virtual classroom and collaborative learning.....	6
2.1.2 Current solutions and challenges.....	8
2.2 Common features of similar systems.....	8
2.3 Research similar systems.....	12
2.4 Feature comparison.....	17
2.5 Conclusion.....	18
3 Chapter 3: Methodology.....	19
3.1 Introduction.....	19
3.2 Agile Methodology.....	19
3.3 Use case diagram.....	20
3.4 Written use case.....	20
3.4.1 Login.....	20
3.4.2 Create Room.....	21
3.4.3 Join Room.....	21
3.4.4 Voice/Text chat.....	21
3.4.5 Share code and files.....	22
3.4.6 Modify and execute code.....	22
3.4.7 Respond to Student.....	22

3.4.8 Notify Teacher.....	23
3.5 Activity Diagram.....	23
3.5.1 Login.....	23
3.5.2 Create Room.....	24
3.5.3 Join Room.....	24
3.5.4 Voice/Text chat.....	24
3.5.5 Share files and code.....	24
3.5.6 Modify and execute code.....	25
3.5.7 Notify Teacher.....	25
3.5.8 Respond to Student.....	26
3.6 Sequence diagram.....	27
3.6.1 Login.....	27
3.6.2 Room Creation.....	28
3.6.3 Join room.....	29
3.6.4 Text/Voice chat and share files.....	30
3.6.5 Share code.....	31
3.6.6 Modify and execute code.....	32
3.6.7 Notify Teacher.....	33
3.6.8 Respond to Student.....	34
3.7 Class diagram.....	35
3.8 Conclusion.....	35
References:.....	36

Table of figures

Figure 1.1.1: Problems faced by students of computer science during distance learning.....	1
Figure 2.1.1: Literature Framework.....	7
Figure 2.2.1: Sharing files in Microsoft teams.....	8
Figure 2.2.2: Skype's Audio and text interface.....	9
Figure 2.2.3: Creation of a meeting room in Google meet.....	9
Figure 2.2.4: Sharing files in Google Docs with different permission.....	10
Figure 2.2.5: Executing code in LeetCode.....	10
Figure 2.2.6: Integrated text editor in Codewars.....	11
Figure 2.2.7: Real-time editing in google docs.....	11
Figure 2.3.1: Screenshot of Zoom's Website.....	12
Figure 2.3.2: Screenshot of Google Meet's Website.....	12
Figure 2.3.3: Screenshot of Google Doc's Website.....	13
Figure 2.3.4: Screenshot of Skype's Website.....	13
Figure 2.3.5: Screenshot of Microsoft Teams' Website.....	14
Figure 2.3.6: Screenshot of Google Hangouts' Website.....	14
Figure 2.3.7: Screenshot of Codewars website.....	15
Figure 2.3.8: Screenshot of CodingBat's Website.....	15
Figure 2.3.9: Screenshot of LeetCode's Website.....	16
Figure 2.3.10: Screenshot of HackerRank's Website.....	16
Figure 3.3.1: Use Case diagram for TeamCode.....	20
Figure 3.5.1: Overview of authentication process to the system.....	23
Figure 3.5.2: Steps of creating a new meeting room.....	24
Figure 3.5.3: Steps to join a meeting room.....	24
Figure 3.5.4: Overview of text and voice chat in TeamCode.....	24
Figure 3.5.5: Overview of sharing files in TeamCode.....	24
Figure 3.5.6: Steps of modification and execution of code in TeamCode.....	25
Figure 3.5.7: Steps of notifying Teacher of a problem by a Student.....	25
Figure 3.5.8: Overview of a Teacher responding to notifications.....	26
Figure 3.6.1: Sequence Diagram for Login.....	27
Figure 3.6.2: Room creation sequence diagram.....	28
Figure 3.6.3: Join room sequence diagram.....	29
Figure 3.6.4: Share files and chat interface sequence diagram.....	30
Figure 3.6.5: Share code sequence diagram.....	31
Figure 3.6.6: Sequence diagram for modification and execution of code.....	32
Figure 3.6.7: Sequence diagram for notifying errors during code execution.....	33
Figure 3.6.8: Sequence diagram of responding to student error notification.....	34
Figure 3.7.1: TeamCode Class diagram.....	35

List of tables

Table 2.4.1: Comparison table.....	17
------------------------------------	----

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

1 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 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 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.

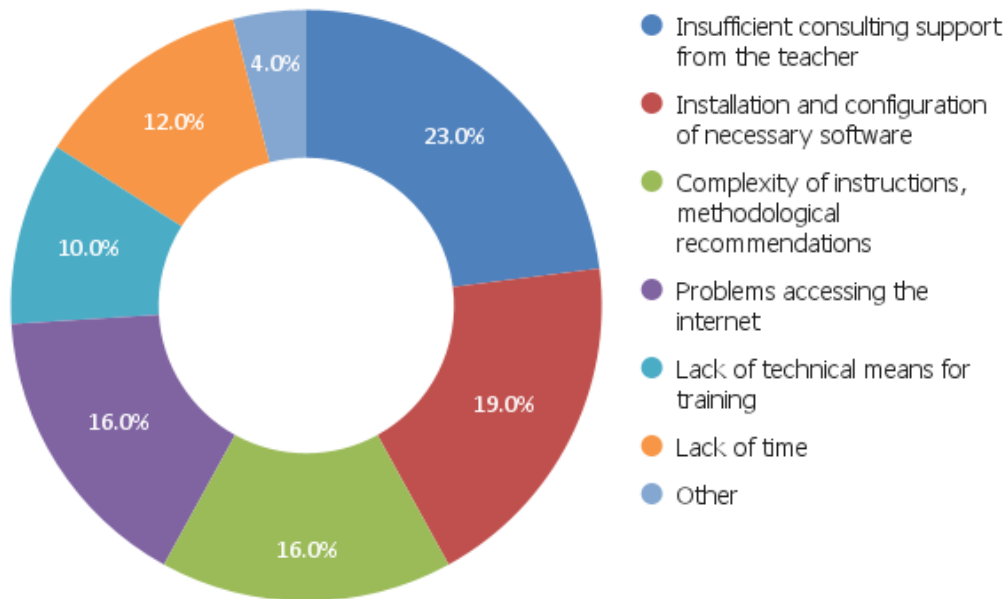


Figure 1.1.1: Problems faced by students of computer science during distance learning
(Zinovieva et al., 2021)

Online education for programming produces outcomes worse than that of regular classes. 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, online education for computer science brings along challenges that result in differences in outcomes against full-time students. Online education loses 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

- Existing applications used to learn to code online provide fewer features for real-time editing of code.
- Existing applications have fewer models for users to pass on notifications to their instructors by highlighting the point at which they encounter anomalies in their code.
- Existing applications require users to install and configure necessary software before they can learn to code online.
- Fewer systems have diverse accessibility control options such as view only and can edit.

1.3 Objectives

- To create a real-time application using WebRTC and Websockets.
- To provide the feature of real-time editing that an instructor can access to solve a learner's problem.
- To make use of flags with the help of which learners can notify any problem.
- To provide a platform for learning to code without having the need to set up and configure programming environments.
- To execute the written code with Judge0 API (Application Programming Interface) and display the output.
- To create an audio channel of communication between instructors and learners.
- To include diverse control options within the system.

1.4 Scope

1.4.1 User scope

- Users will be able to host a room where learners can join in if the user is an instructor.
- Users will be able to communicate through text and audio interfaces.
- Users will be able to execute code within the system.
- Users will be able to share their code with other users in the system with varying permissions.
- Users will be able to edit the code of their learners in real-time.

1.4.2 System Scope

- The system will provide users to communicate through text and audio interfaces.
- The system will allow users to share their code with others with permissions such as can edit and can view.
- The system will provide users to code online without any previous setup.
- The system will enable real-time editing of code.
- The system will allow instructors to create a room for learners to join in.

1.5 Constraints

- The system will use Monaco Editor as the code editor which is not supported in mobile browsers or mobile web frameworks.
- 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.
- 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*, n.d.).

1.6 Methodology

Real-time features implemented with optimized resource consumption will ensure that the communication between the instructor and learner remains stable. 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, 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.

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 Monaco 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. Monaco 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 paper. The paper consists of six 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 paper topic.
3. Chapter 3: This chapter gives a detailed description of the methodology used to achieve the paper's objective, the different software tools and the reasons behind using the said tools.

1.9 Conclusion

There exists a gap 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 problem into consideration the project will facilitate better communication between instructors and learners in the context of learning to code online. The project will provide real-time interaction using WebSockets and a communication channels with an audio interface through WebRTC.

2 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).

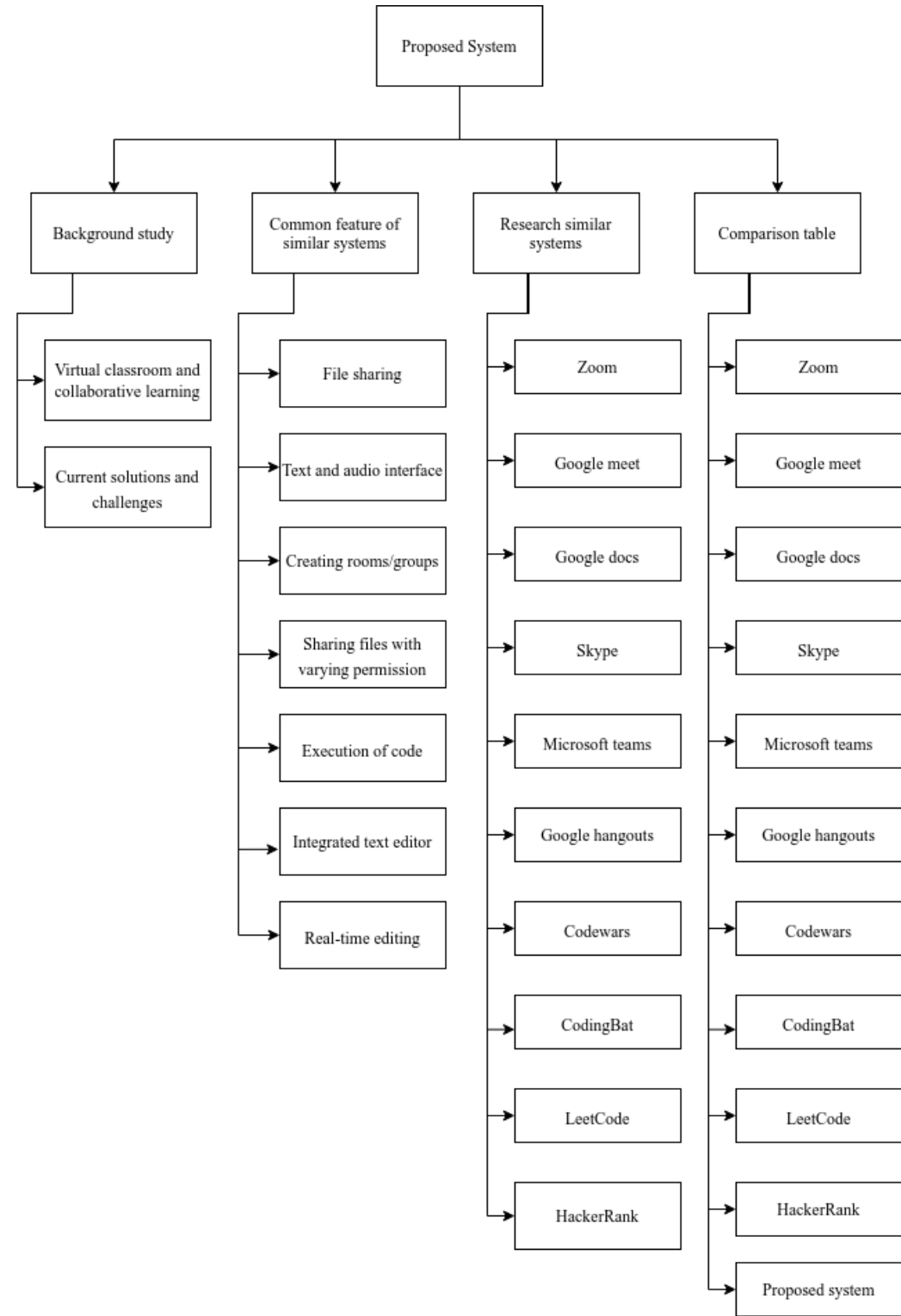


Figure 2.1.1: Literature Framework

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, codingbat. 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:

- a) 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.

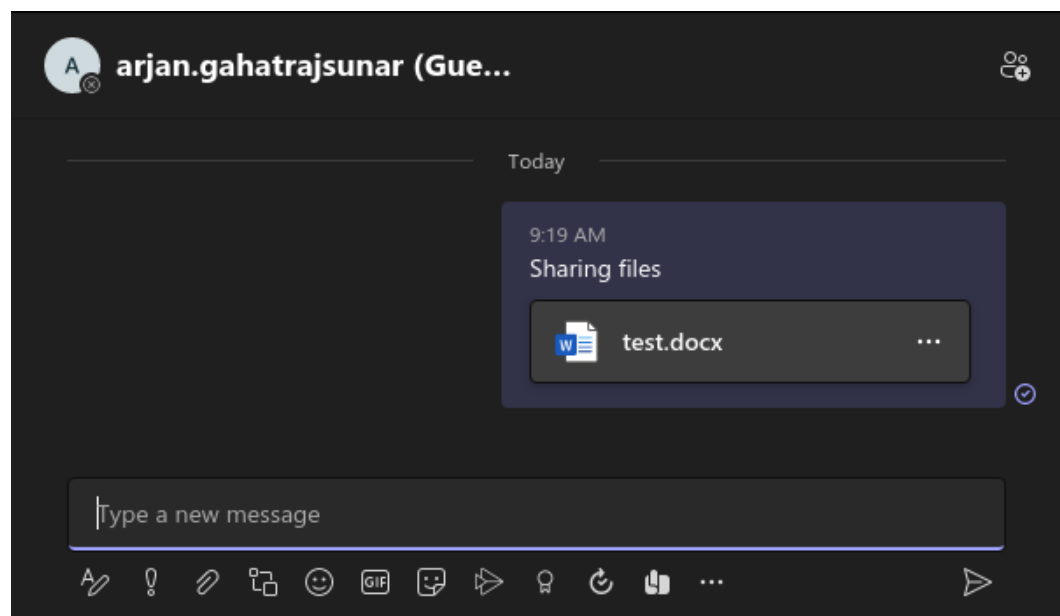


Figure 2.2.1: Sharing files in Microsoft teams

(Video Conferencing, Meetings, Calling | Microsoft Teams, n.d.)

- b) Text and audio interface: This feature enables communication and interaction between users of the system.

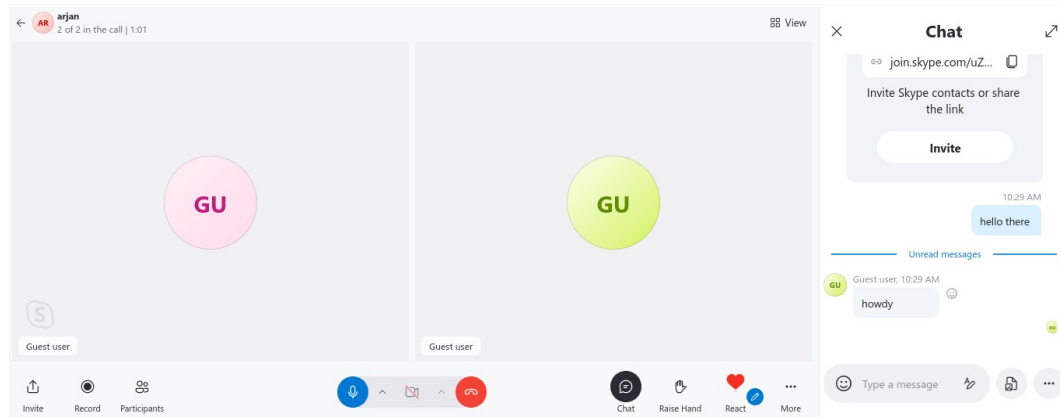


Figure 2.2.2: Skype's Audio and text interface
(Skype | Stay Connected with Free Video Calls Worldwide, n.d.)

- c) Creating rooms/groups: This feature enables users to create groups or rooms to schedule a meeting for a class.

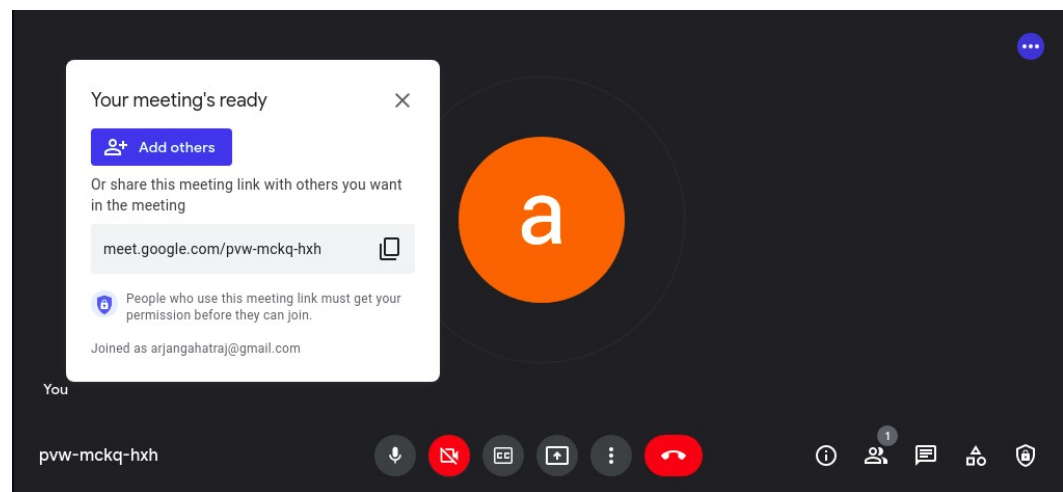


Figure 2.2.3: Creation of a meeting room in Google meet
(Google Meet, n.d.)

- d) 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.

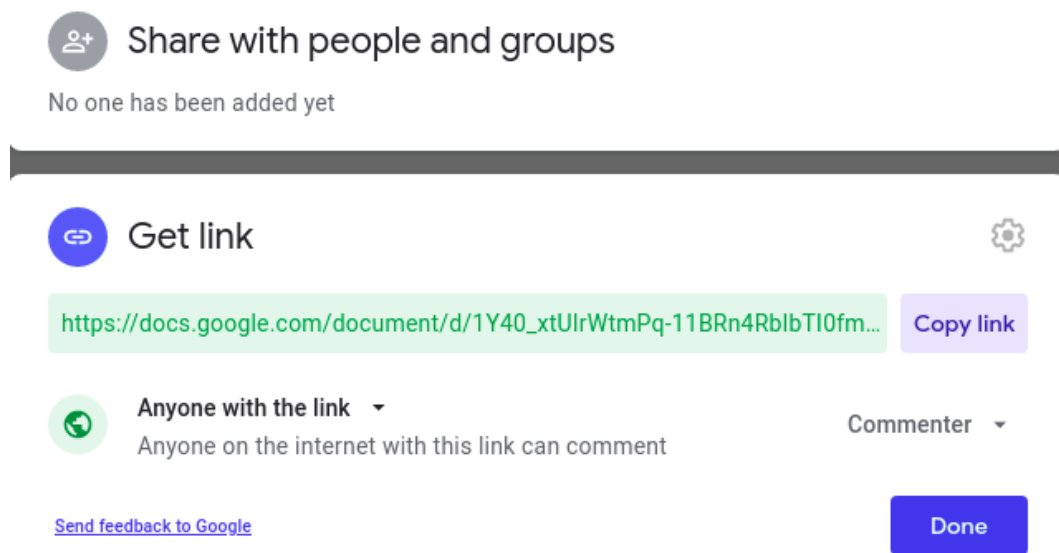


Figure 2.2.4: Sharing files in Google Docs with different permission (Google Docs, n.d.)

- e) Execution of code: This feature enables users to execute their code or solution to a given problem and view their results.

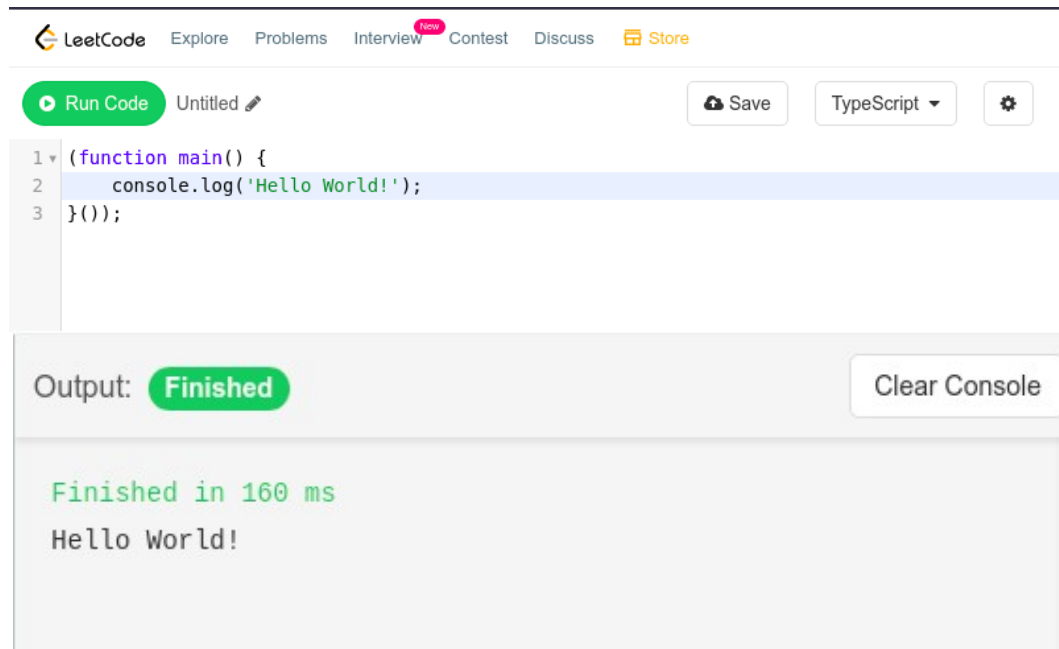
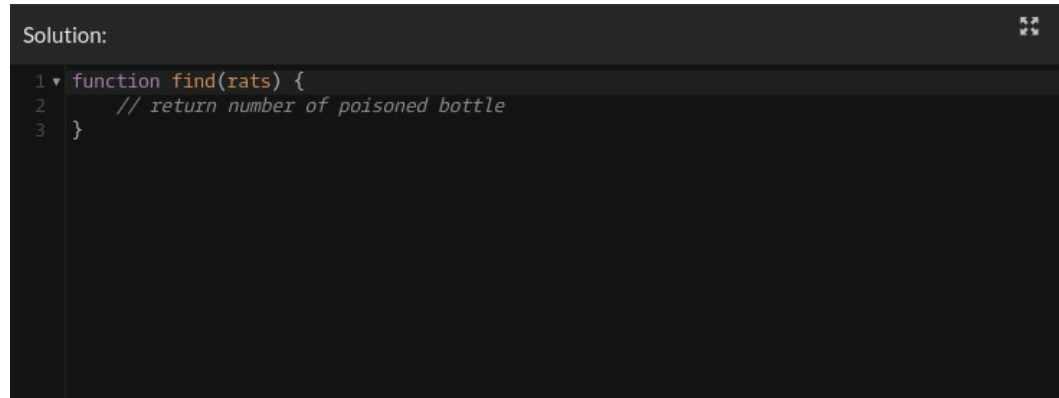


Figure 2.2.5: Executing code in LeetCode (LeetCode - The World's Leading Online Programming Learning Platform, n.d.)

- f) Integrated text editor: This feature enables users to create and modify their code or solution within the system.



```
Solution:
1 function find(rats) {
2   // return number of poisoned bottle
3 }
```

Figure 2.2.6: Integrated text editor in Codewars
(Codewars, n.d.)

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

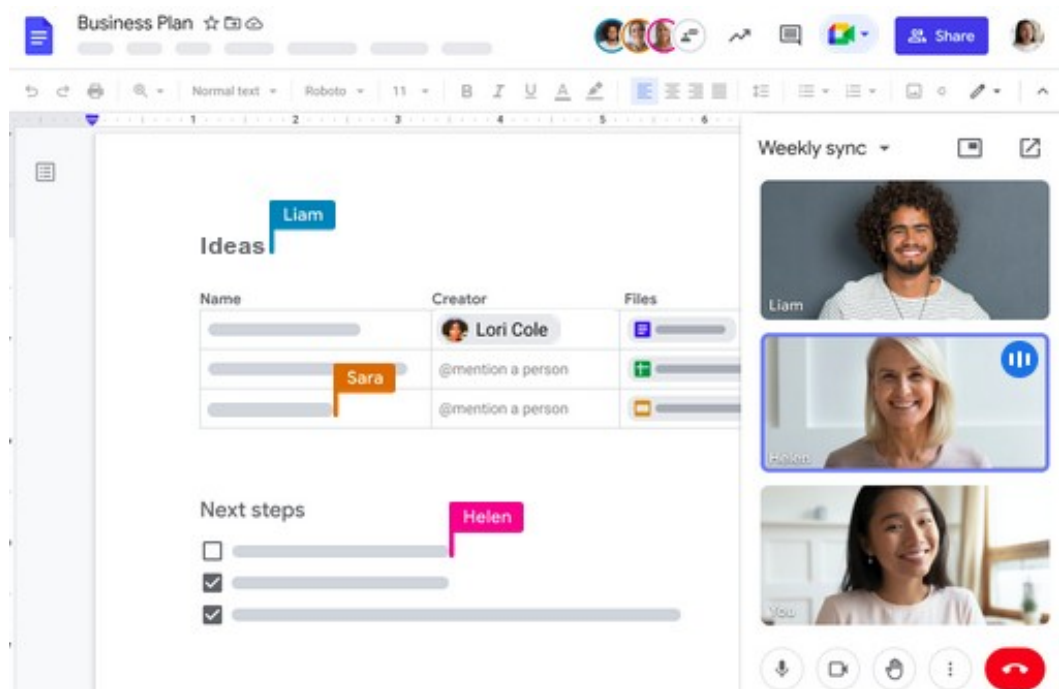


Figure 2.2.7: Real-time editing in google docs
(Google Docs, n.d.)

2.3 Research similar systems

- a) Zoom: Zoom is a secure and reliable video platform that provides services including meetings, chat, phone, webinars, and online events.

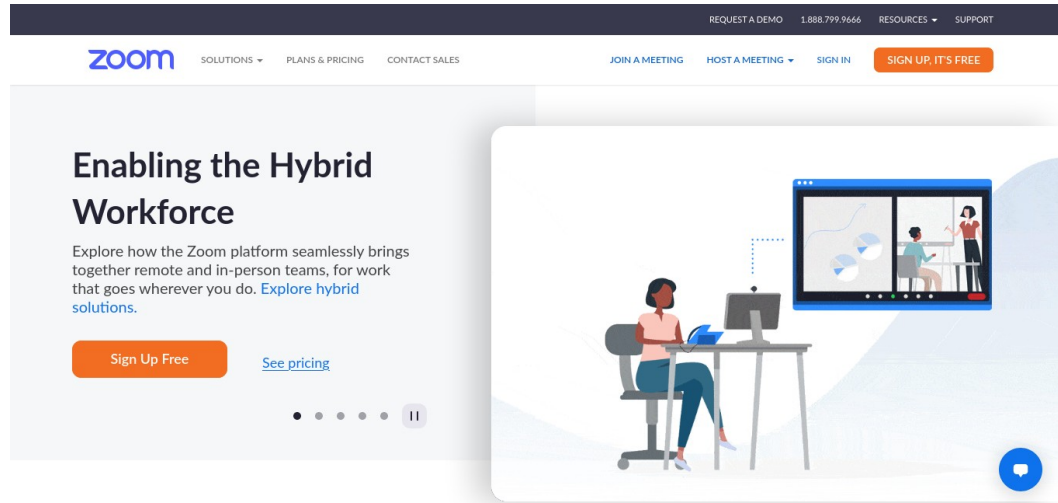


Figure 2.3.1: Screenshot of Zoom's Website

(Video Conferencing, Cloud Phone, Webinars, Chat, Virtual Events | Zoom, n.d.)

- b) Google Meet: Google meet is a real-time meeting application by Google that allows users to share video, desktop, presentations using a web browser.

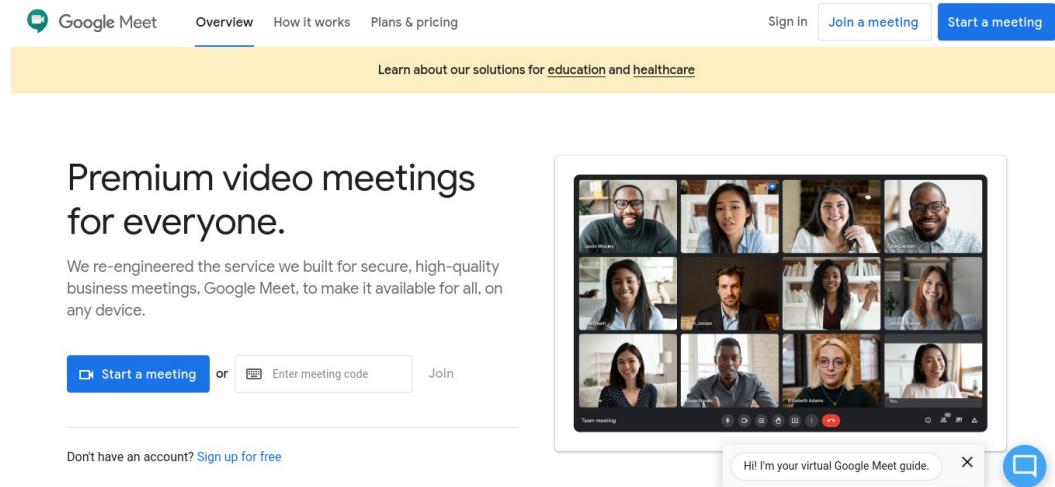


Figure 2.3.2: Screenshot of Google Meet's Website

(Google Meet, n.d.)

- c) Google Docs: Google Docs is an online word processing application that is part of Google's free, web-based Google Docs Editors package.

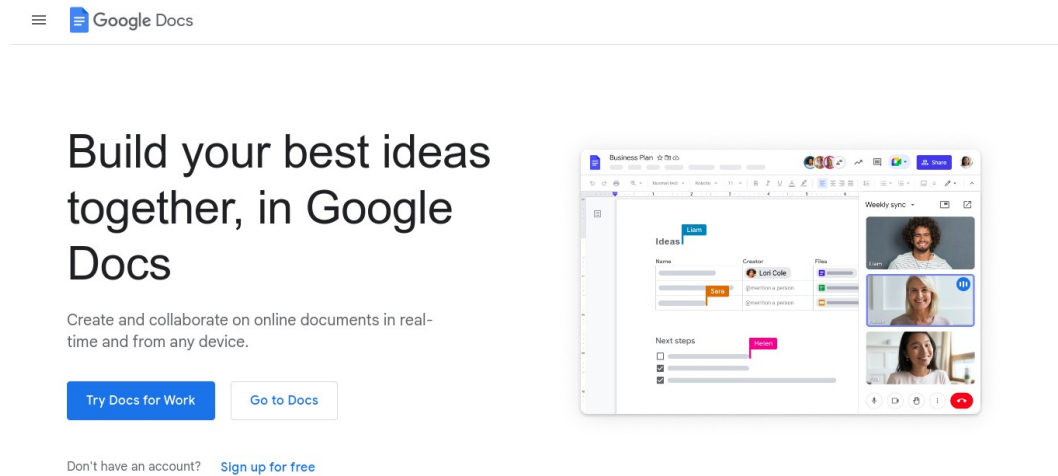


Figure 2.3.3: Screenshot of Google Doc's Website
(Google Docs, n.d.)

- d) Skype: Skype is a proprietary telecommunications service that includes VoIP-based video telephony, video and voice conferencing.

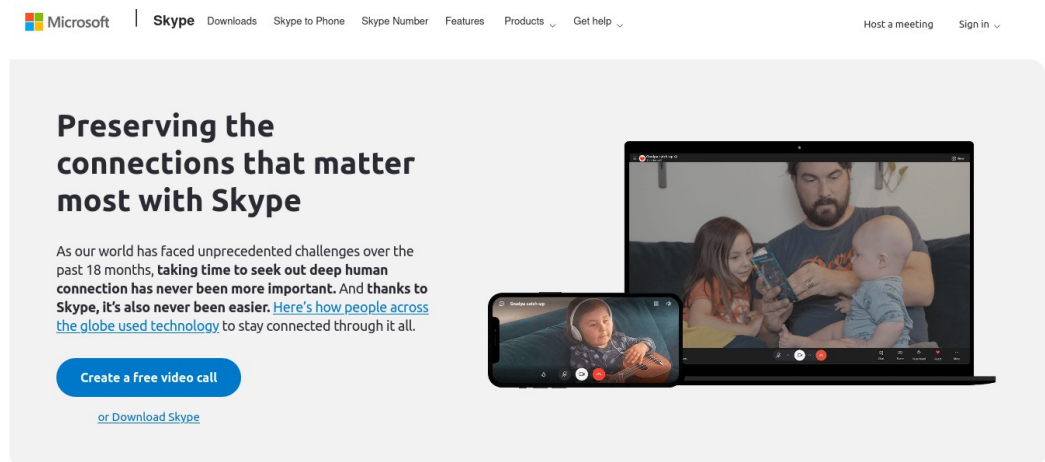


Figure 2.3.4: Screenshot of Skype's Website
(Skype | Stay Connected with Free Video Calls Worldwide, n.d.)

- e) Microsoft teams: Microsoft Teams is a specialized business communication platform that includes workplace chat and videoconferencing, file storage, and application integration.

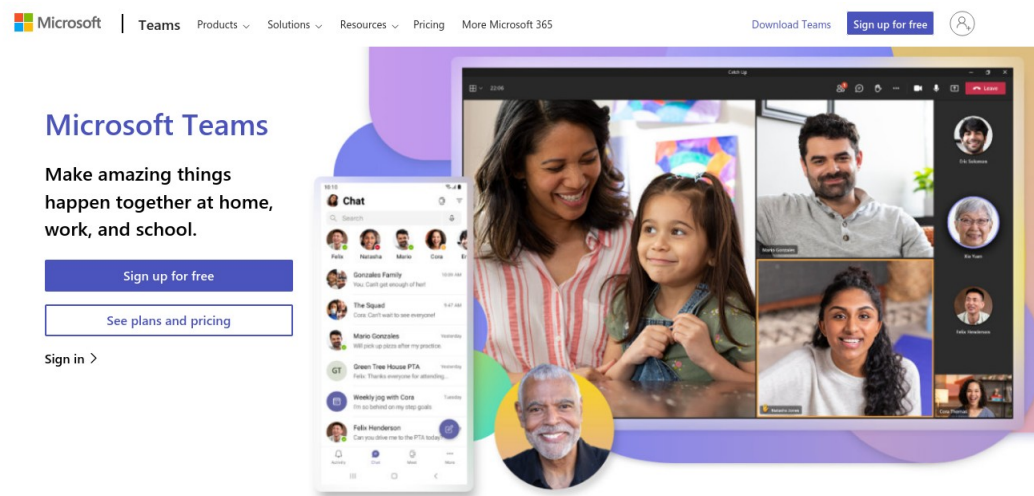


Figure 2.3.5: Screenshot of Microsoft Teams' Website
(Video Conferencing, Meetings, Calling | Microsoft Teams, n.d.)

- f) 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.

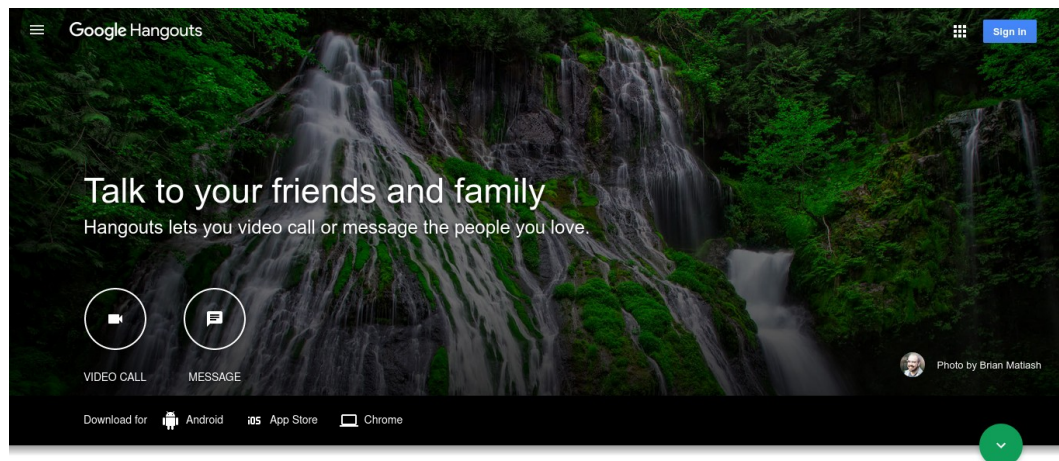


Figure 2.3.6: Screenshot of Google Hangouts' Website
(Google Hangouts, n.d.)

- g) Codewars: Codewars is a coding practice site for all programmers that assists in the learning of various programming languages.

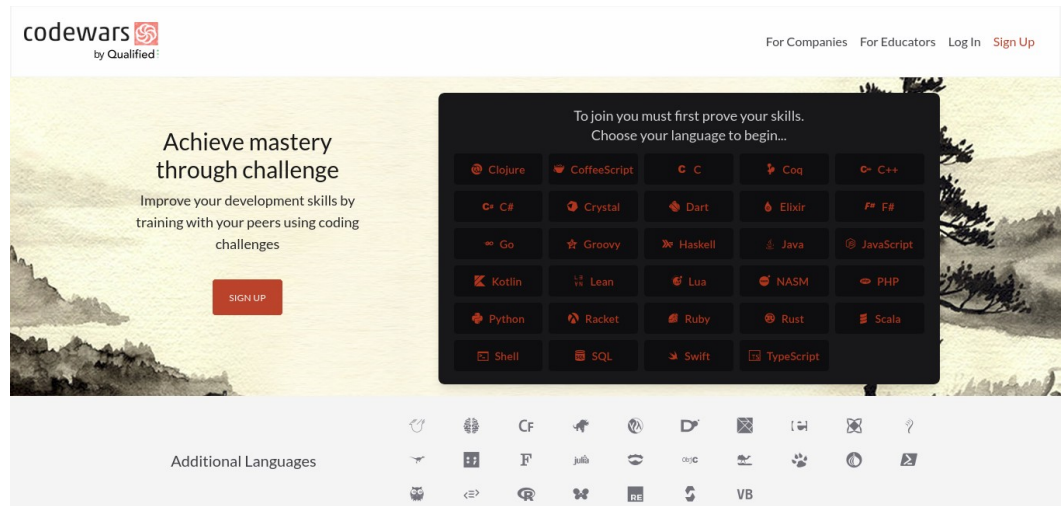


Figure 2.3.7: Screenshot of Codewars website
(Codewars, n.d.)

- h) 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.

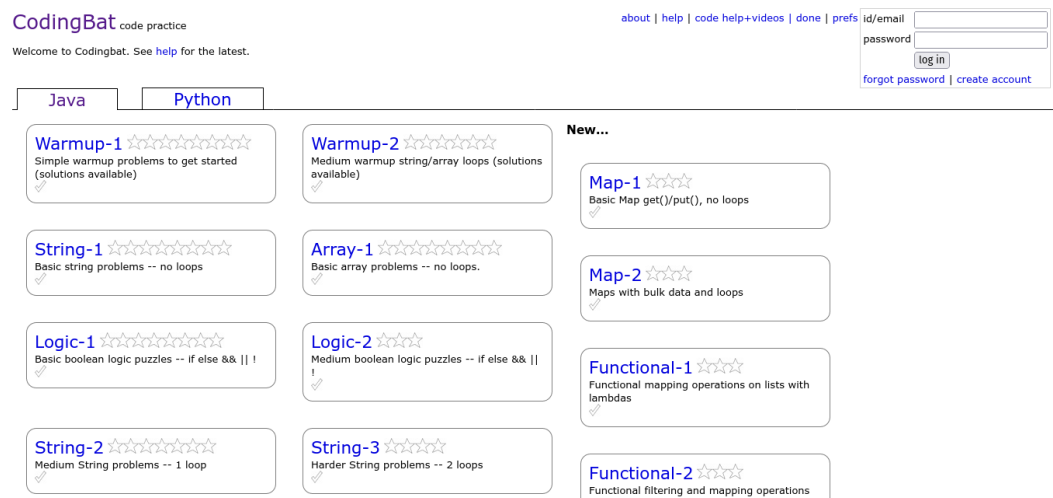


Figure 2.3.8: Screenshot of CodingBat's Website
(CodingBat Java, n.d.)

- i) LeetCode: LeetCode is a web-based platform that helps to improve coding abilities, broaden technical knowledge, and prepare for technical interviews.

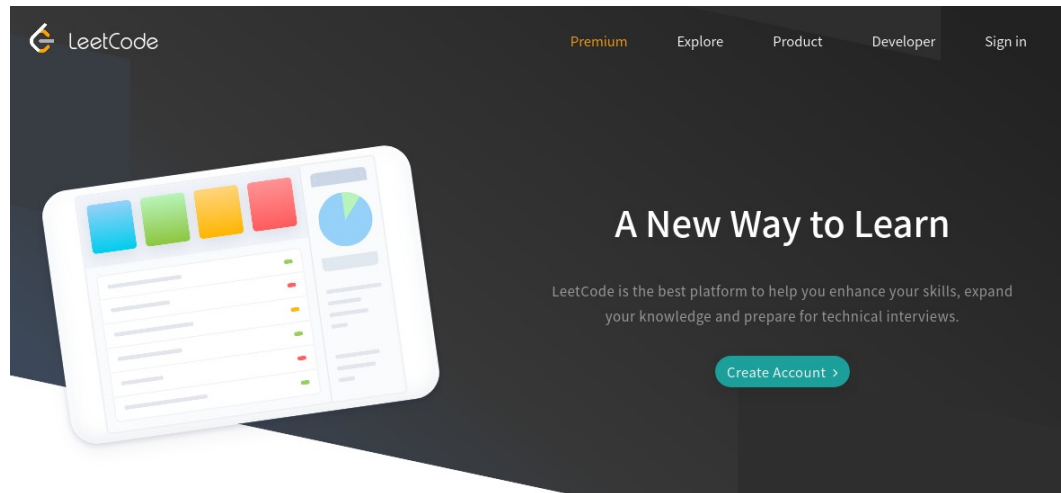


Figure 2.3.9: Screenshot of LeetCode's Website
(LeetCode - The World's Leading Online Programming Learning Platform, n.d.)

- j) 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.

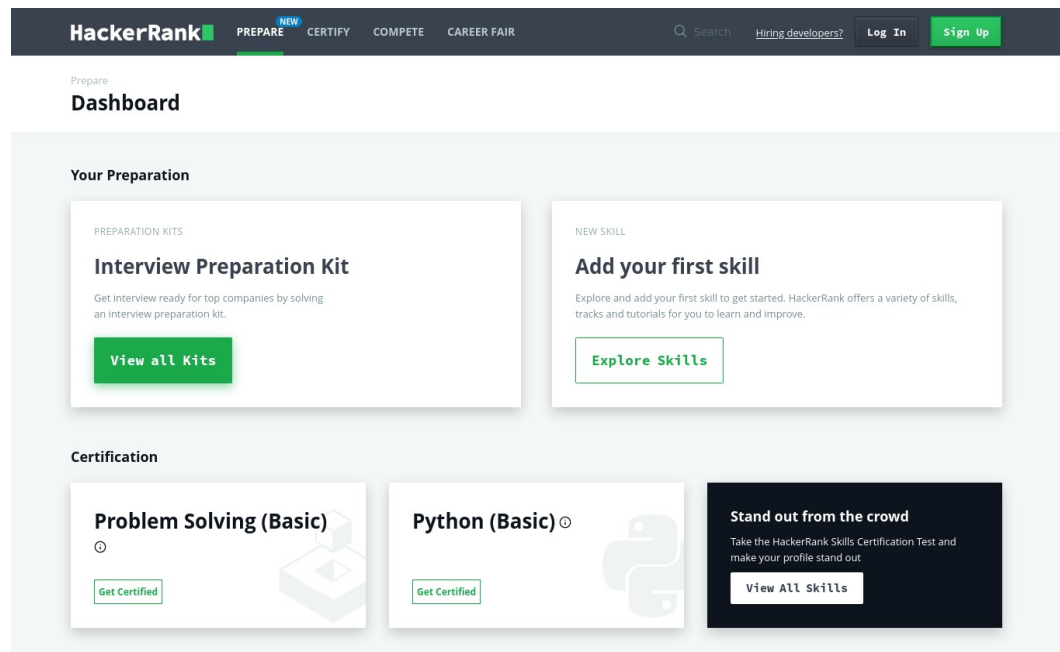


Figure 2.3.10: Screenshot of HackerRank's Website
(HackerRank, n.d.)

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	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

There are problems in online education and questions related to its effectiveness but these problems can be minimized by using proper teaching strategies. Focus on collaborative learning and interaction can be the strategy to improve online education. Synchronous meeting apps help to provide collaboration and interactive elements to online learning while online judge apps provide a platform for users to gain technical knowledge. Therefore, taking elements from both these applications will enable better technical online education in the field of computer science.

3 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.

3.3 Use case diagram

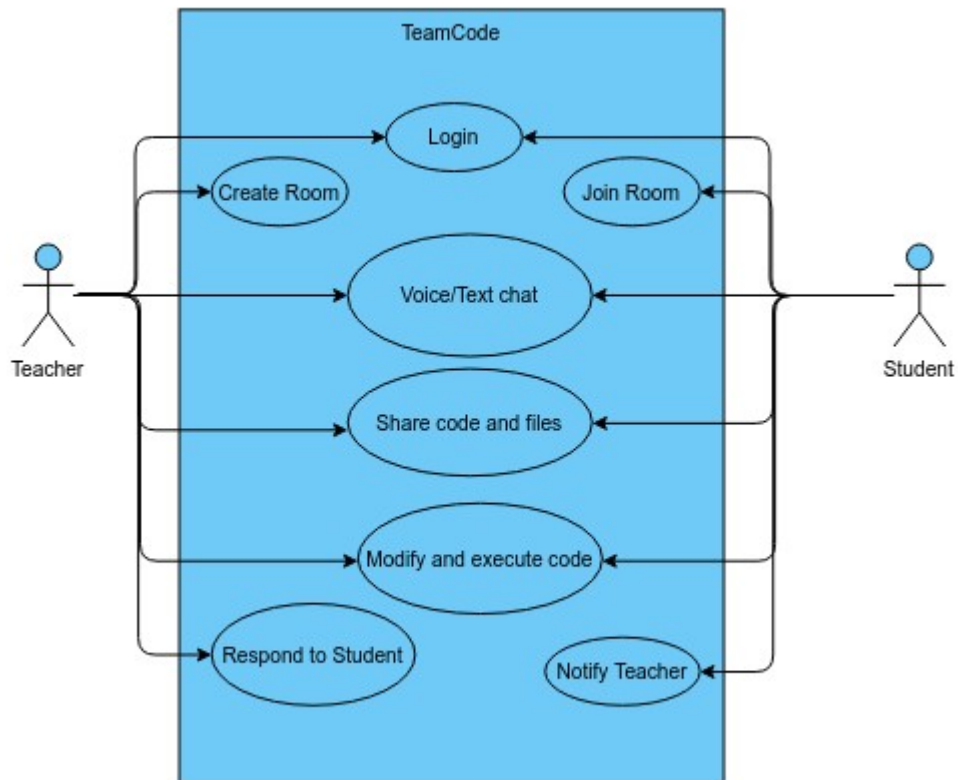


Figure 3.3.1: Use Case diagram for TeamCode

3.4 Written use case

3.4.1 Login

Use case ID:	UC-1
Use case Name:	Login
Actors:	<ul style="list-style-type: none">TeacherStudent
Preconditions:	Non
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 have successfully logged in with their GitHub account	
Exception	
1. Users do not have GitHub account	

3.4.2 Create Room

Use case ID:	UC-2
Use case Name:	Create Room
Actors:	<ul style="list-style-type: none">• 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	
<ol style="list-style-type: none">1. Teacher hosts a room2. Teacher distributes it to students	

3.4.3 Join Room

Use case ID:	UC-3
Use case Name:	Join Room
Actors:	<ul style="list-style-type: none">• 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	
<ol style="list-style-type: none">1. Both actors have joined the room	
Exception	
<ol style="list-style-type: none">1. Incorrect room id	

3.4.4 Voice/Text chat

Use case ID:	UC-4
Use case Name:	Voice/Text chat
Actors:	<ul style="list-style-type: none">• 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	
<ol style="list-style-type: none">1. Users have joined a room and started communicating	

3.4.5 Share code and files

Use case ID:	UC-5
Use case Name:	Share code and files
Actors:	<ul style="list-style-type: none">• 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 are able to share their code and files with each other	

3.4.6 Modify and execute code

Use case ID:	UC-6
Use case Name:	Modify and execute code
Actors:	<ul style="list-style-type: none">• 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 are able to use the text editor and execute the code within the system	

3.4.7 Respond to Student

Use case ID:	UC-7
Use case Name:	Respond to Student
Actors:	<ul style="list-style-type: none">• Teacher
Preconditions:	Student notifies Teacher of 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 are able to view notifications from Students and respond to it with real-time code editing	
Exception	
1. No students have joined or no notifications	

3.4.8 Notify Teacher

Use case ID:	UC-8
Use case Name:	Notify Teacher
Actors:	<ul style="list-style-type: none">• Student
Preconditions:	Student joins a room and executes code
Post conditions:	Student notifies the Teacher if errors are found when code is executed using flag feature
Trigger:	Student encounters a problem during execution of code
Main success scenario	
1. Students are able to raise error flags to notify the Teacher when problems are encountered during execution of code.	

3.5 Activity Diagram

3.5.1 Login

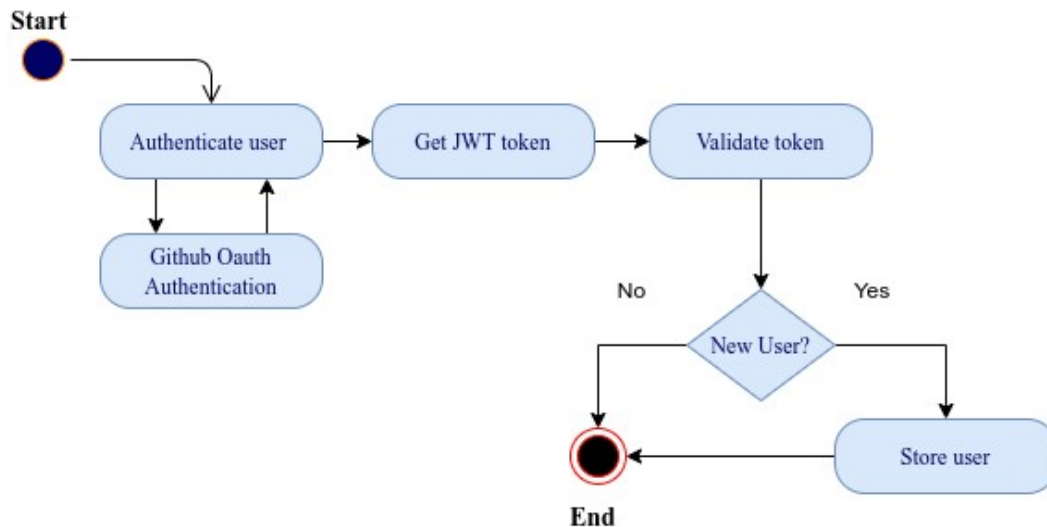


Figure 3.5.1: Overview of authentication process to the system

3.5.2 Create Room

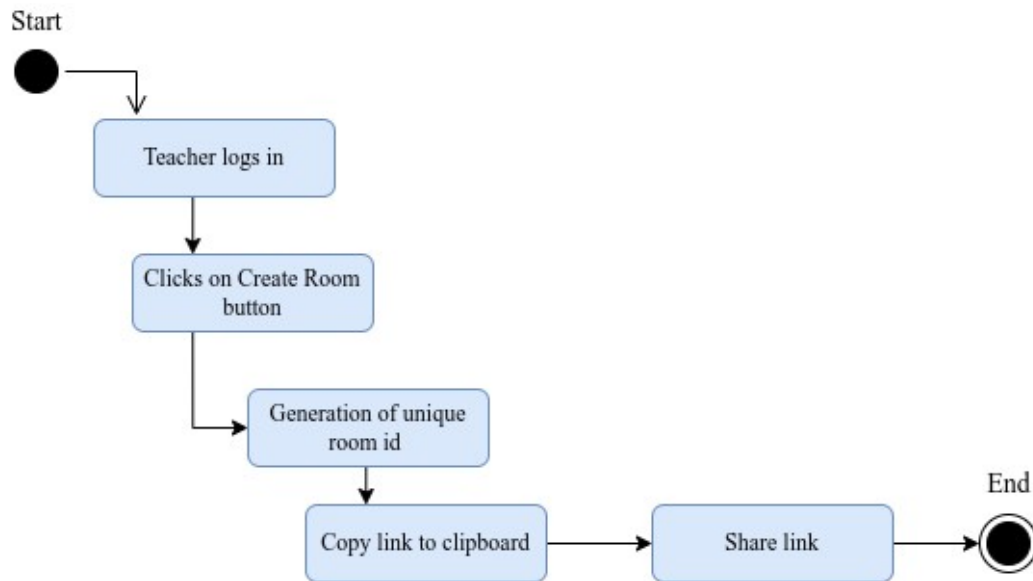


Figure 3.5.2: Steps of creating a new meeting room

3.5.3 Join Room

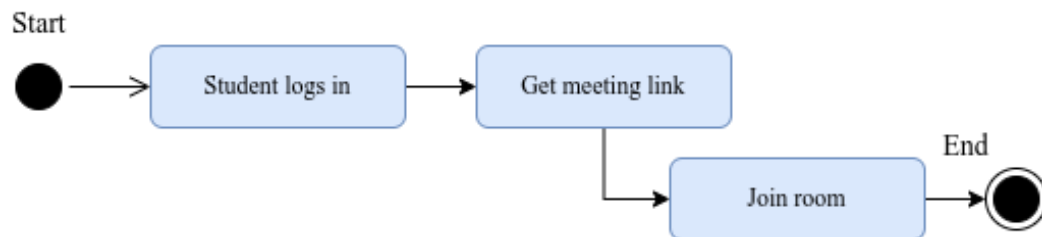


Figure 3.5.3: Steps to join a meeting room

3.5.4 Voice/Text chat

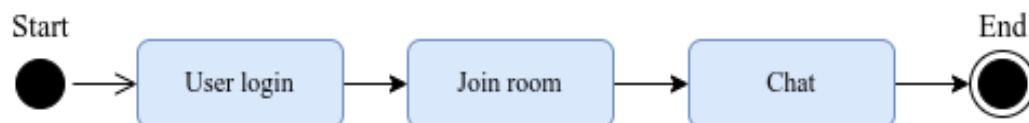


Figure 3.5.4: Overview of text and voice chat in TeamCode

3.5.5 Share files and code

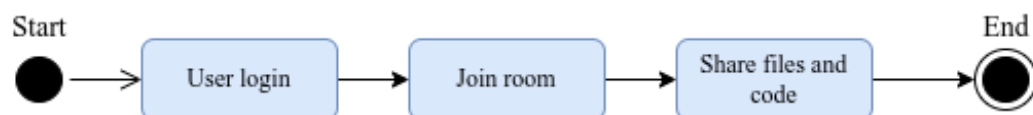


Figure 3.5.5: Overview of sharing files in TeamCode

3.5.6 Modify and execute code

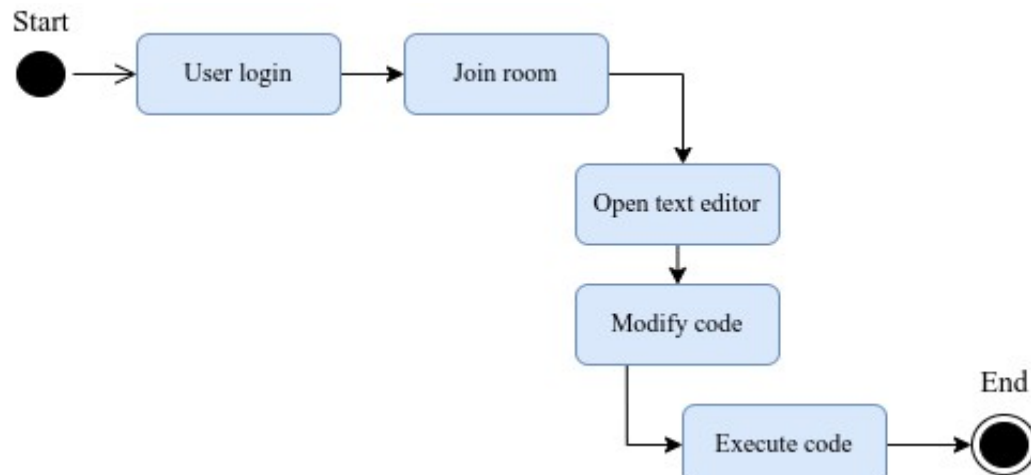


Figure 3.5.6: Steps of modification and execution of code in TeamCode

3.5.7 Notify Teacher

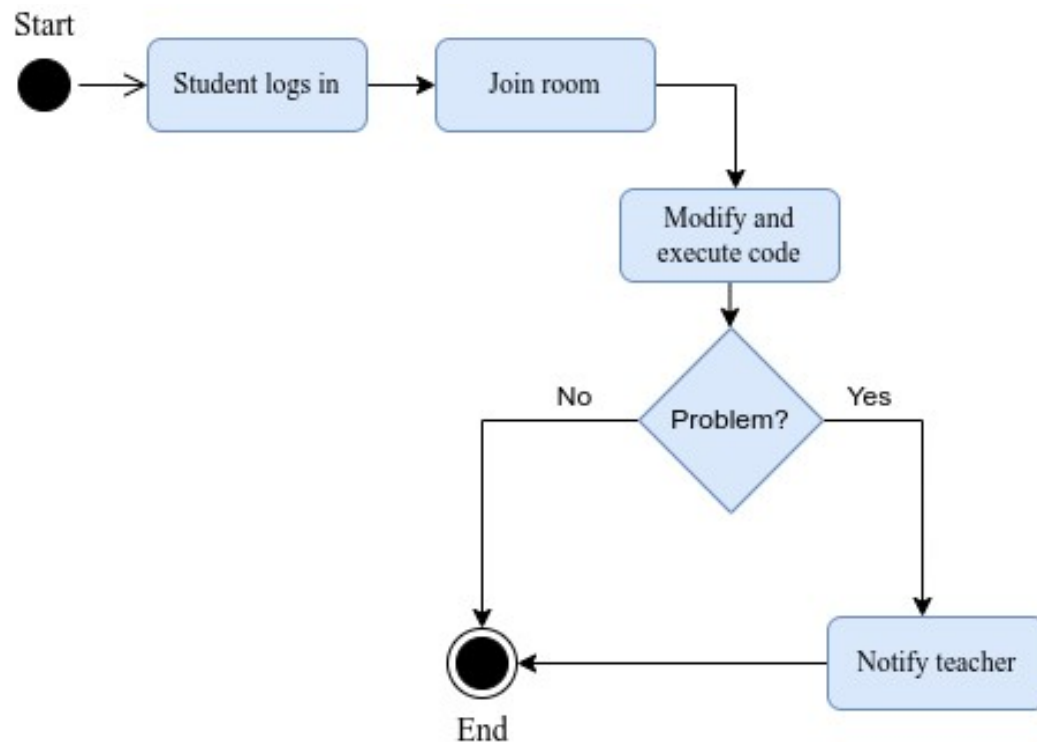


Figure 3.5.7: Steps of notifying Teacher of a problem by a Student

3.5.8 Respond to Student

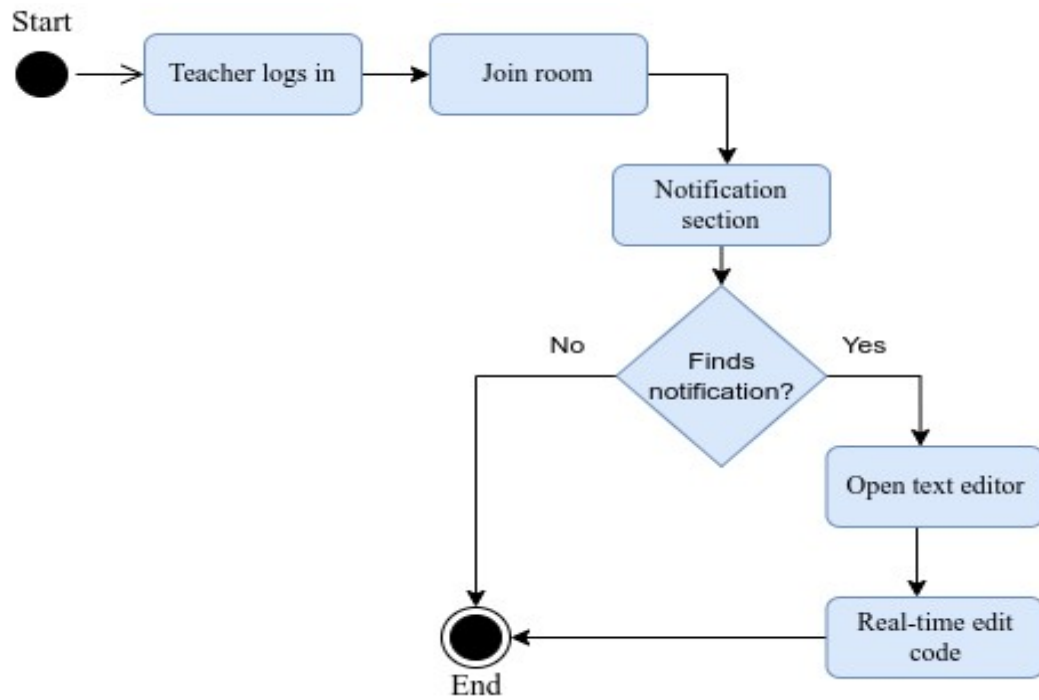


Figure 3.5.8: Overview of a Teacher responding to notifications

3.6 Sequence diagram

3.6.1 Login

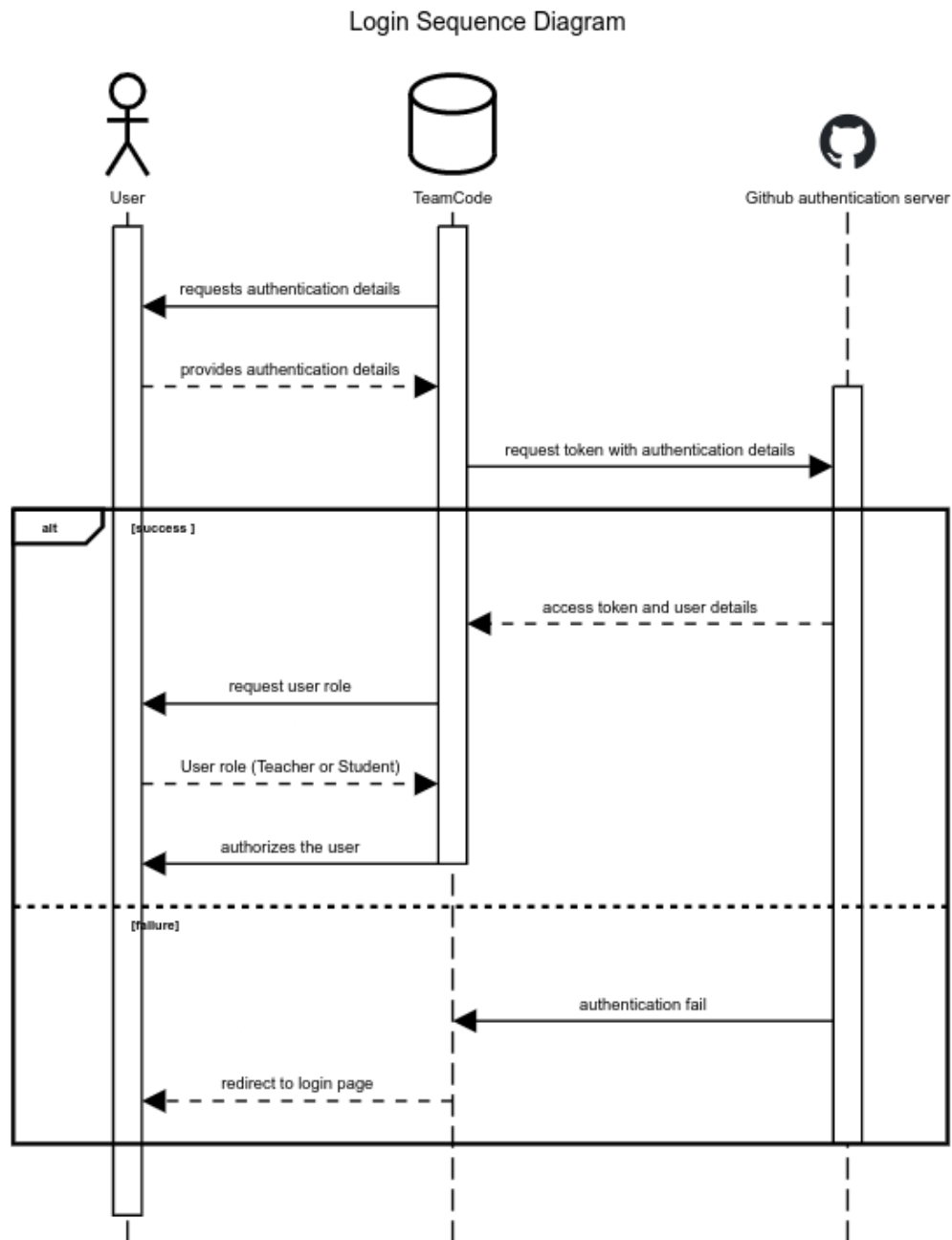


Figure 3.6.1: Sequence Diagram for Login

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

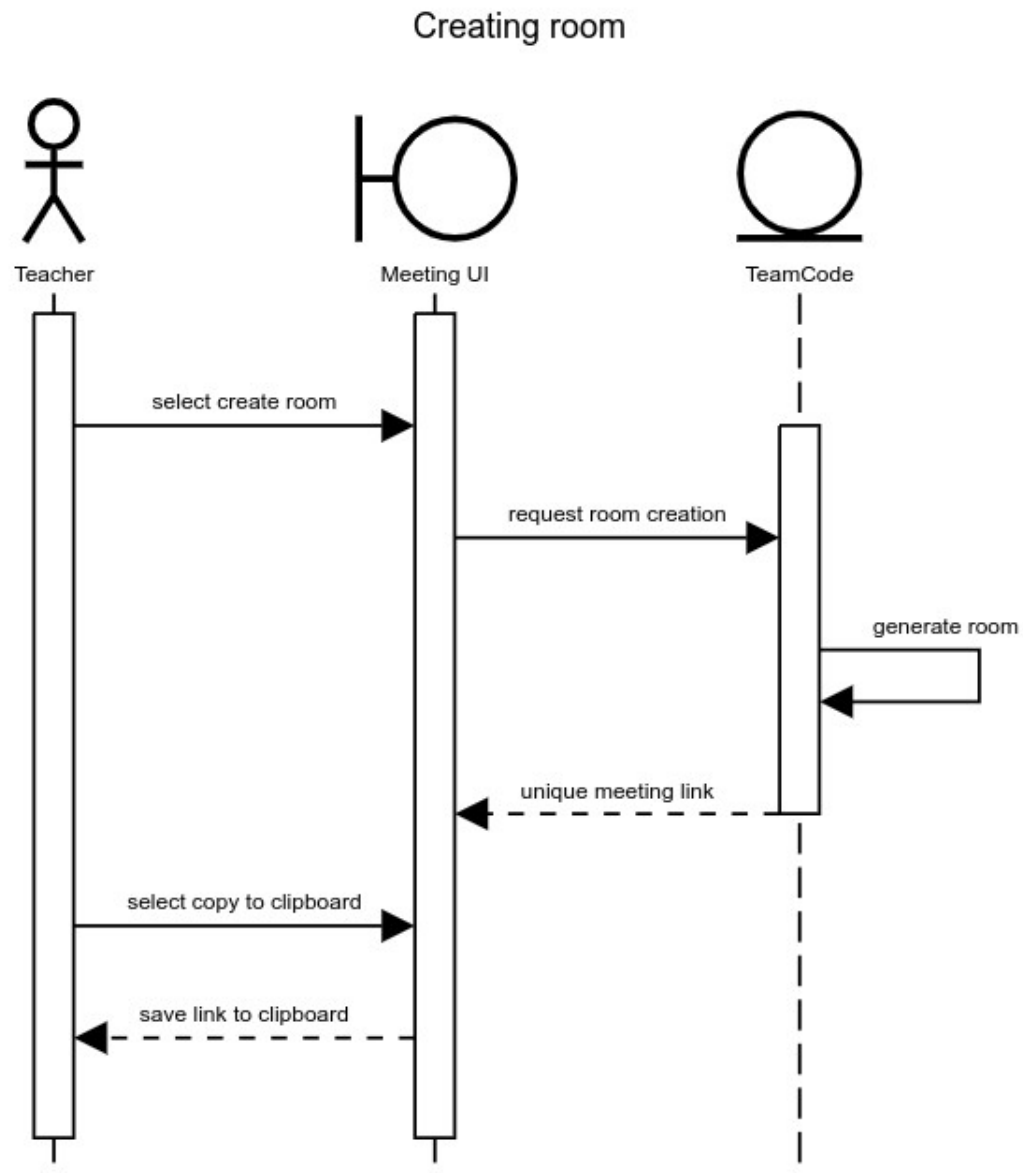


Figure 3.6.2: Room creation sequence diagram

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

3.6.3 Join room

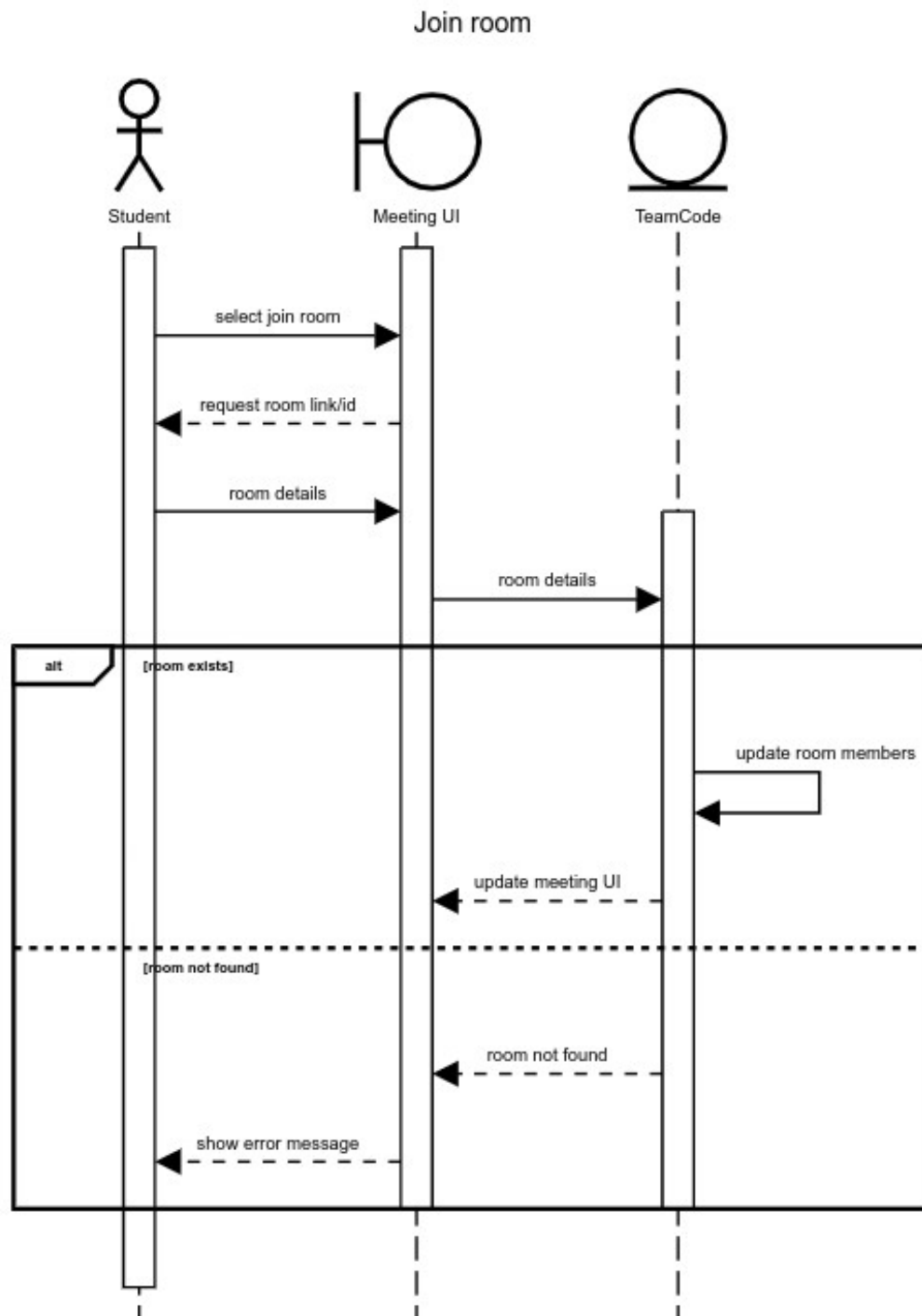


Figure 3.6.3: Join room sequence diagram

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

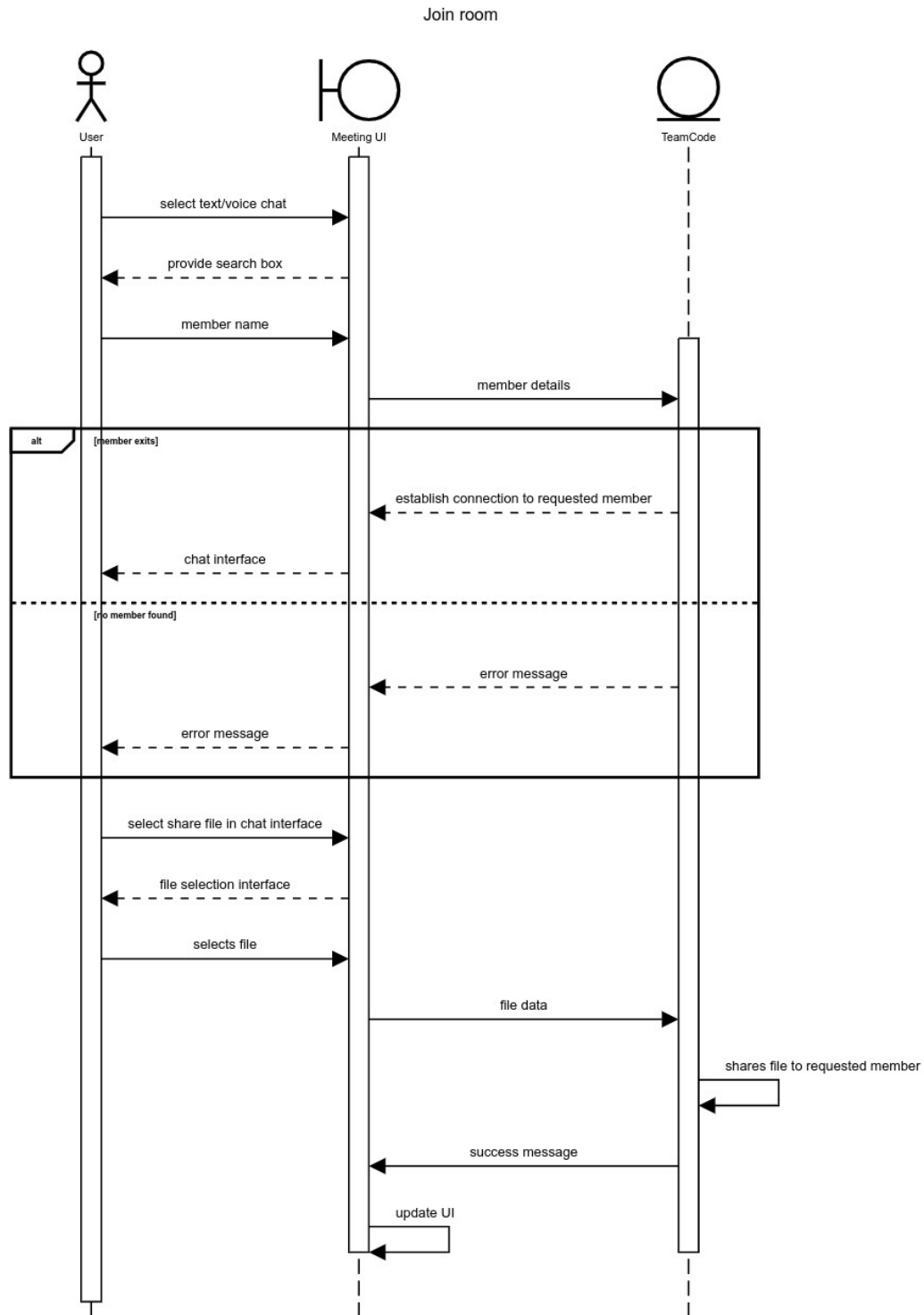


Figure 3.6.4: Share files and chat interface sequence diagram

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

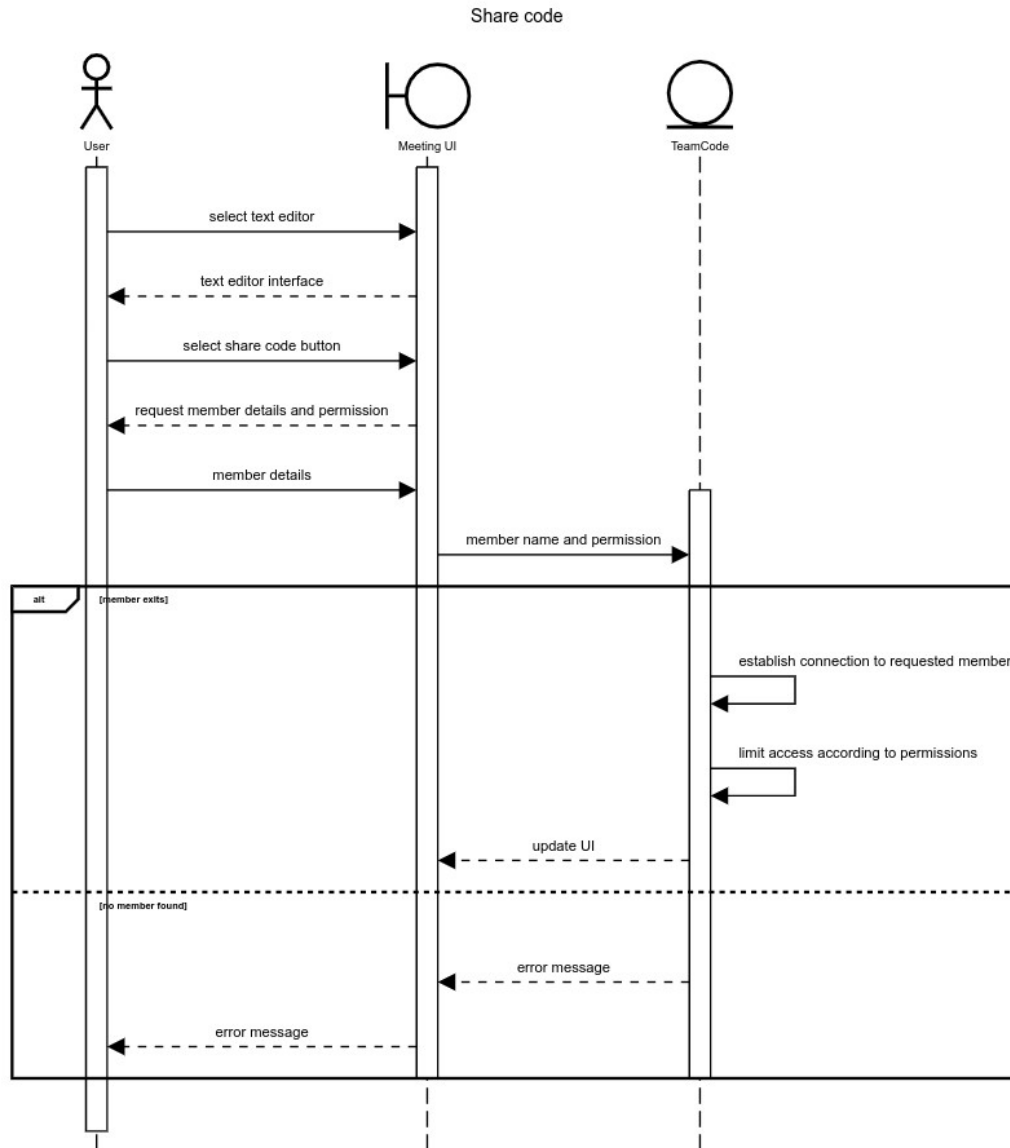


Figure 3.6.5: Share code sequence diagram

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

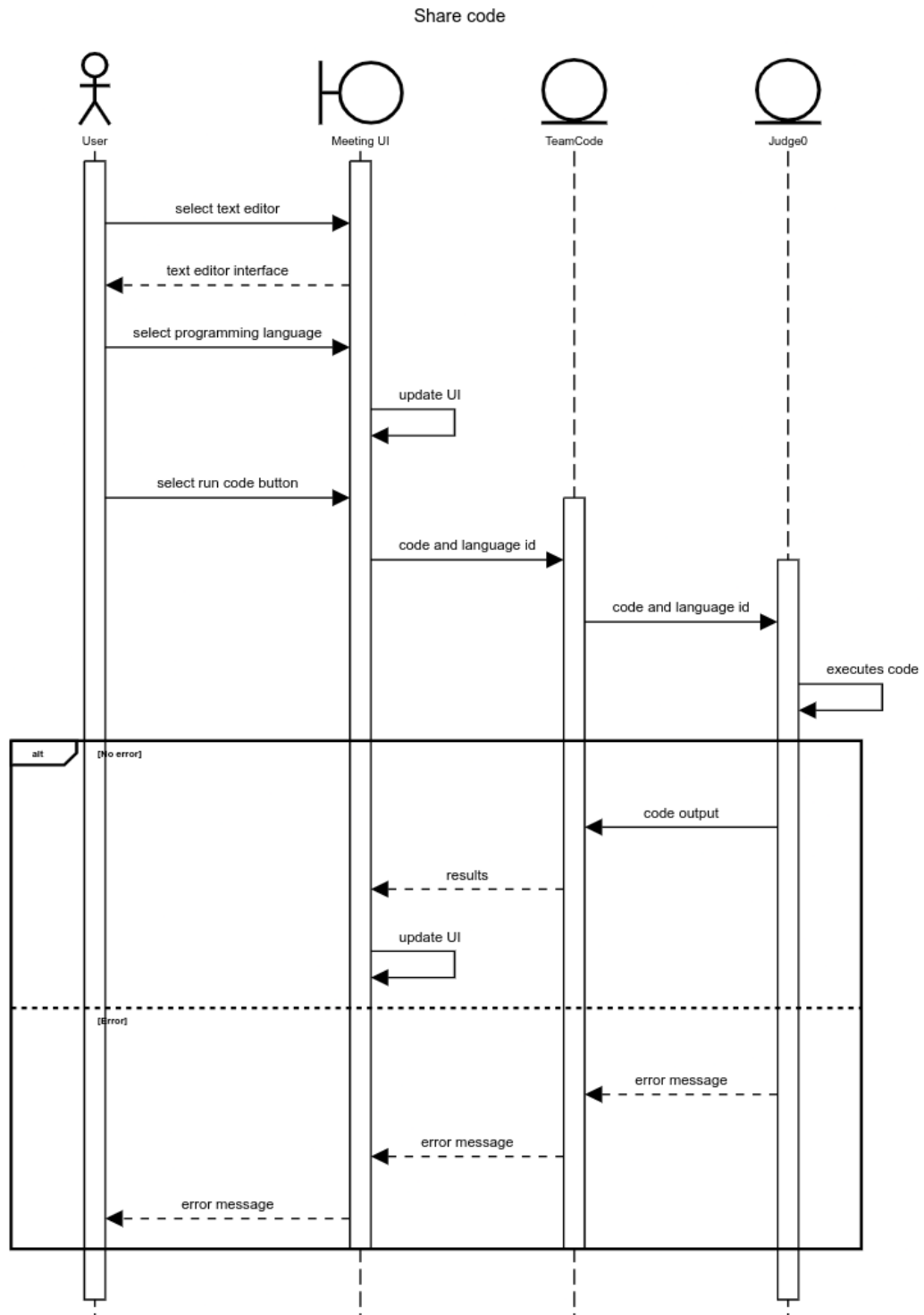


Figure 3.6.6: Sequence diagram for modification and execution of code

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

3.6.7 Notify Teacher

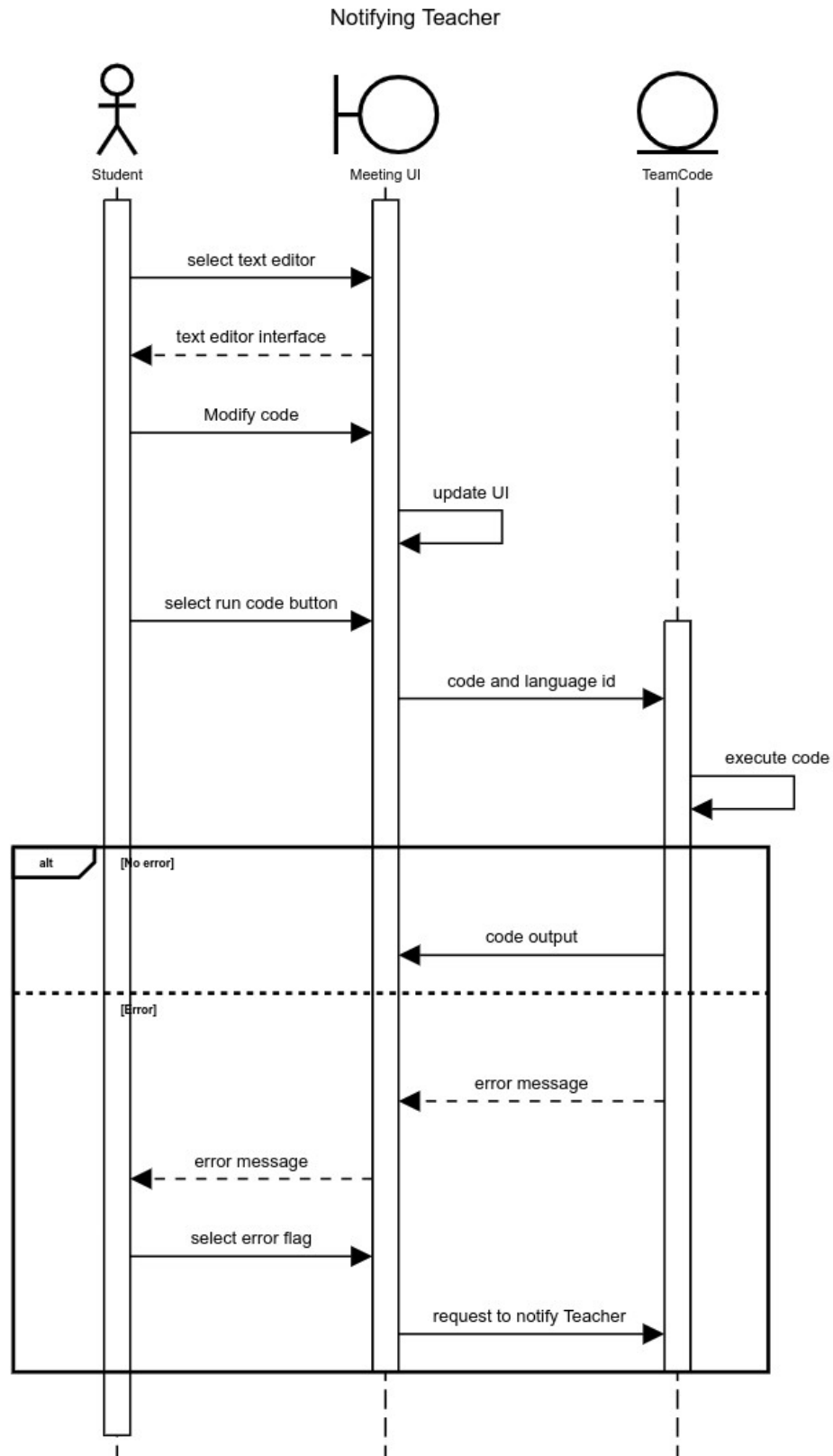


Figure 3.6.7: Sequence diagram for notifying errors during code execution

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 flag which triggers a 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

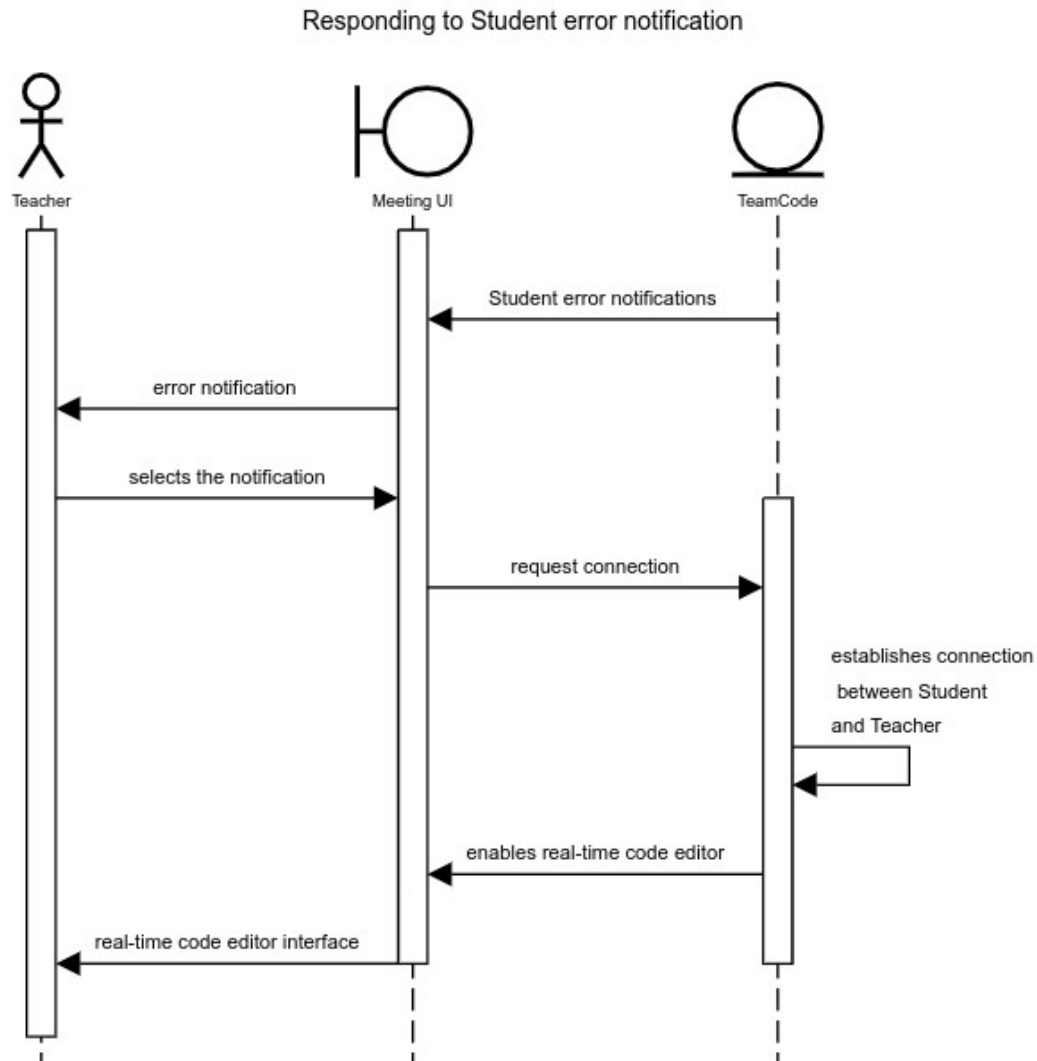


Figure 3.6.8: Sequence diagram of responding to student error notification

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

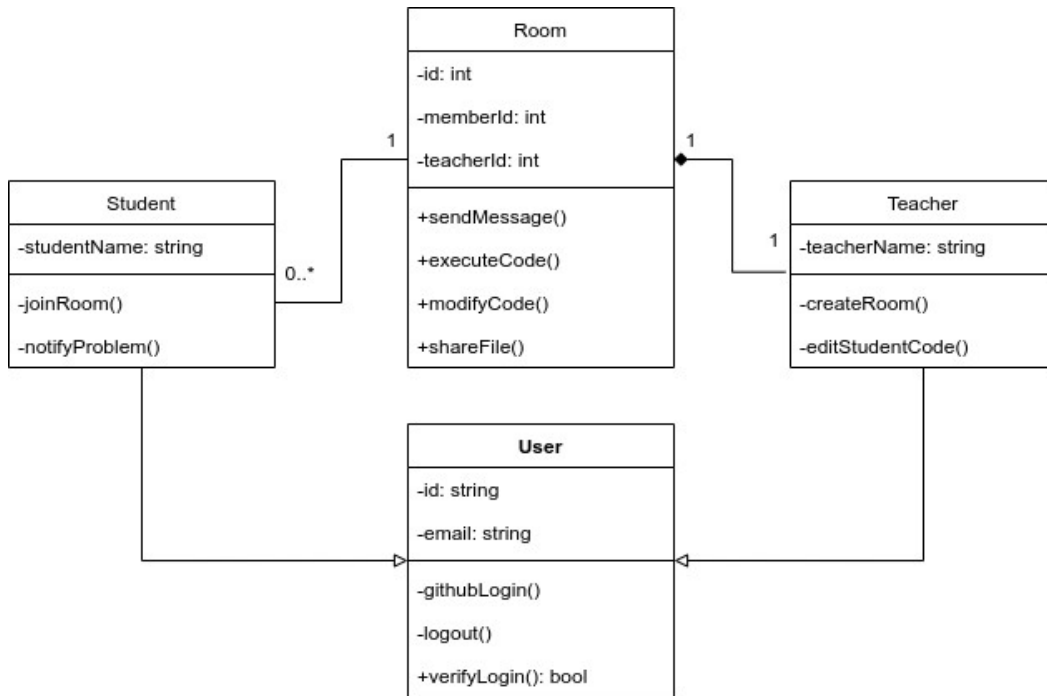


Figure 3.7.1: TeamCode Class diagram

The class diagram shows the different entities present 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.

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*. (n.d.). Codewars. Retrieved January 24, 2022, from <https://www.codewars.com>
- CodingBat Java*. (n.d.). Retrieved January 24, 2022, from <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*. (n.d.). Retrieved January 24, 2022, from <https://docs.google.com>
- Google Hangouts*. (n.d.). Retrieved January 24, 2022, from <https://hangouts.google.com/>
- Google Meet*. (n.d.). Retrieved January 24, 2022, from <https://meet.google.com/>
- HackerRank*. (n.d.). HackerRank. Retrieved January 24, 2022, from <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*. (n.d.). Retrieved January 24, 2022, from <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](https://doi.org/10.1016/S1096-7516(02)00158-6)
- Skype | Stay connected with free video calls worldwide. (n.d.). Retrieved January 24, 2022, from <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. (n.d.). Retrieved January 24, 2022, from <https://zoom.us/>
- Video Conferencing, Meetings, Calling | Microsoft Teams. (n.d.). Retrieved January 24, 2022, from <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. (n.d.). Retrieved December 17, 2021, from <https://caniuse.com/?search=web%20rtc%20>

Zinovieva, I. S., Artemchuk, V. O., Iatsyshyn, A. V., Popov, O. O., Kovach, V. O., Iatsyshyn, A. V., Romanenko, Y. O., & Radchenko, O. V. (2021). The use of online 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>