



The University of Hong Kong
Department of Computer Science
2023 - 2024
COMP4801 Final Year Project
Final Report

“Instant Messaging Software for Academic Consultation”

Fung Ka Ho (3035902635)

Supervised by Dr. T.W. Chim

Abstract

Instant messaging (IM) software has become one of the most common communication tools nowadays due to its convenience and ease of use. As long as people hold an electronic device like a mobile phone or a computer, they can communicate with others at any time and anywhere in real-time. Although IM software has been widely used in daily communication, it is not a common practice for students to use IM software to contact their teachers.

Nowadays, students mainly use face-to-face communication or email to contact teachers. Compared to IM software, these methods are more time-consuming and inconvenient. Taking this into consideration, this project aims to develop an IM software for students to contact their teachers. Additionally, to make students more engaged in lessons and hence enhance their learning efficiency, a system with interactive quizzes creating and responding functionalities will be added to this IM software.

As of the time of writing this final report, all graphical user interface (GUI) implementations as well as all planned functionalities in the app have been finished.

Acknowledgment

I would like to take this opportunity to express my deep and sincere gratitude to Dr. Chim, Tat Wing for providing me with lots of valuable guidance throughout this final year project.

Table of Contents

Abstract.....	1
Acknowledgment.....	2
Table of Contents.....	3
List of Figures.....	5
List of Tables.....	7
Abbreviations.....	8
1. Introduction.....	9
1.1 Background.....	9
1.2 Motivation.....	10
1.3 Project Overview and Objectives.....	11
1.4 Report Outline.....	12
2. Methodology.....	13
2.1 Software and Tools Concerned.....	13
2.1.1 Database.....	13
2.1.2 Frontend.....	14
2.1.3 Backend.....	15
2.1.4 Deployment.....	16
2.1.5 Document and Media Files Storage.....	17
2.1.6 Content Delivery Network.....	17
2.1.7 Push Notification Service.....	18
2.2 Designs.....	18
2.2.1 Database Design.....	18
2.2.2 System Architecture Design.....	23
2.2.3 Push Notification.....	23
2.2.4 Amazon CloudFront CDN and S3 Bucket.....	24
3. Result.....	26
3.1 Login.....	26
3.2 Registration.....	30
3.3 Home Page.....	33
3.4 Profile Page.....	37
3.5 Create/Join Chat Group.....	39
3.6 Real-time Chatting System.....	41
3.7 Chat Group Management.....	42
3.8 Quiz Page Access.....	44
3.9 Quiz Creation.....	45
3.10 Quiz Answering and Updating.....	46
3.11 Quiz Information.....	49
3.12 Push Notification.....	52

4. Future Plan.....	54
5. Conclusion.....	55
Reference.....	56

List of Figures

- Figure 1: Leading mobile apps worldwide in 2022, by downloads [P.9]
- Figure 2: Number of mobile phone messaging app users worldwide from 2019 to 2025 [P.10]
- Figure 3: Hierarchical structure of document-oriented database [P.13]
- Figure 4: Relationship among service worker, user's device web browser and server [P.15]
- Figure 5: Content Delivery Network (CDN) implementation [P.18]
- Figure 6: Database data model diagram for this project [P.22]
- Figure 7: Documents in User collection, ChatRoom collection, and OTP collection [P.22]
- Figure 8: System Architecture [P.23]
- Figure 9: Sequence diagram for push notification feature [P.24]
- Figure 10: Accessing documents and image files from Amazon CloudFront CDN and S3 Bucket [P.25]
- Figure 11: Login page activated user identity select drop-down menu [P.26]
- Figure 12: Login page pre-populated email domain (for student) [P.27]
- Figure 13: Login page pre-populated email domain (for teacher) [P.28]
- Figure 14: Requesting one-time password [P.29]
- Figure 15: Retrieve an OTP generated by the system from the app user's mailbox [P.29]
- Figure 16: Providing the received OTP to the designated field [P.30]
- Figure 17: Registration page OTP validation [P.31]
- Figure 18: Required personal information for registration (for student) [P.32]
- Figure 19: Required personal information for registration (for teacher) [P.32]
- Figure 20: File selector for uploading profile image [P.33]
- Figure 21: Home page layout [P.34]
- Figure 22: Example of home page chat group list [P.35]
- Figure 23: Example of home page quiz group list [P.35]
- Figure 24: Sorted chat group list creation [P.36]
- Figure 25: Sorted quiz group list creation [P.36]
- Figure 26: Chat group searching example [P.37]
- Figure 27: Quiz group searching example [P.37]
- Figure 28: Profile page (for student) [P.38]
- Figure 29: Profile page (for teacher) [P.39]
- Figure 30: Add new group page layout [P.40]
- Figure 31: Example of joining an existing chat group [P.41]

Figure 32: Example of sending text messages and sharing documents to the chat group [P.42]

Figure 33: Chat group information page (for group administrators) [P.43]

Figure 34: Chat group information page (for regular group members) [P.44]

Figure 35: Quiz page layout for teachers (left) and students (right) [P.45]

Figure 36: Example of quiz creation [P.46]

Figure 37: Example of quiz submission [P.47]

Figure 38: Notification after quiz submission [P.48]

Figure 39: Quiz model answer reveal [P.48]

Figure 40: Existing quiz update example [P.49]

Figure 41: Layout for the quiz information page [P.50]

Figure 42: List of Submit and List of Unsubmit Example [P.51]

Figure 43: Charts for displaying the correctness proportion and choices distribution [P.52]

Figure 44: Algorithm for generating a list of unique colors [P.52]

Figure 45: Notification message display on mobile phone [P.53]

Figure 46: Notification message display on computer [P.53]

List of Tables

Table 1: Database OTP collection structure [P.18]

Table 2: Database User collection structure [P.19]

Table 3: Database charRoomMember List Document structure [P.19]

Table 4: Database ChatRoom collection structure [P.20]

Table 5: Database message List Document structure [P.20]

Table 6: Database messageStatus List Document structure [P.21]

Table 7: Database quiz List Document structure [P.21]

Table 8: Database studentQuizResponse List Document structure [P.21]

Abbreviations

Instant Messaging	(IM)
Graphical User Interface	(GUI)
Progressive web app	(PWA)
HyperText Markup Language	(HTML)
Cascading Sheet	(CSS)
JavaScript Object Notation	(JSON)
Uniform Resource Locator	(URL)
User Interface	(UI)
Document Object Model	(DOM)
Server-Sent Events	(SSE)
Hypertext Transfer Protocol	(HTTP)
Amazon Elastic Compute Cloud	(Amazon EC2)
Database as a Service	(DBaaS)
Amazon Web Services	(AWS)
Content Delivery Network	(CDN)
Firebase Cloud Messaging	(FCM)
One-time password	(OTP)
The University of Hong Kong	(HKU)

1. Introduction

1.1 Background

Instant messaging (IM) can be simply defined by its goals in checking the online status of users and messaging [1]. Yet, despite its intended usage within the U.S. government for emergencies dated back to 1971, as the concept has gradually traversed into the general public, such messaging apps nowadays offer a much larger variety of features, which allow users to conduct real-time communication through text transmission and multimedia elements such as emotion stickers, photos, audio clips, videos and recently, even by creating polls [1].

With its multi-functionality and the high accessibility thanks to the increasing commonness of mobile phones and the Internet, IM software has been popularly received as shown by TikTok, Instagram and Whatsapp being the top three most downloaded mobile apps worldwide and six out of the top ten most downloaded apps either being IM applications or provided with IM functions in 2022 (see Figure 1) [2]. Also, the use of IM applications is predicted to be a consistent rise as the number of instant messaging software users worldwide is expected to reach 3.51 billion by 2025 (see Figure 2), which would be 37.1% higher than the 2.45 billion recorded in 2019 [3].

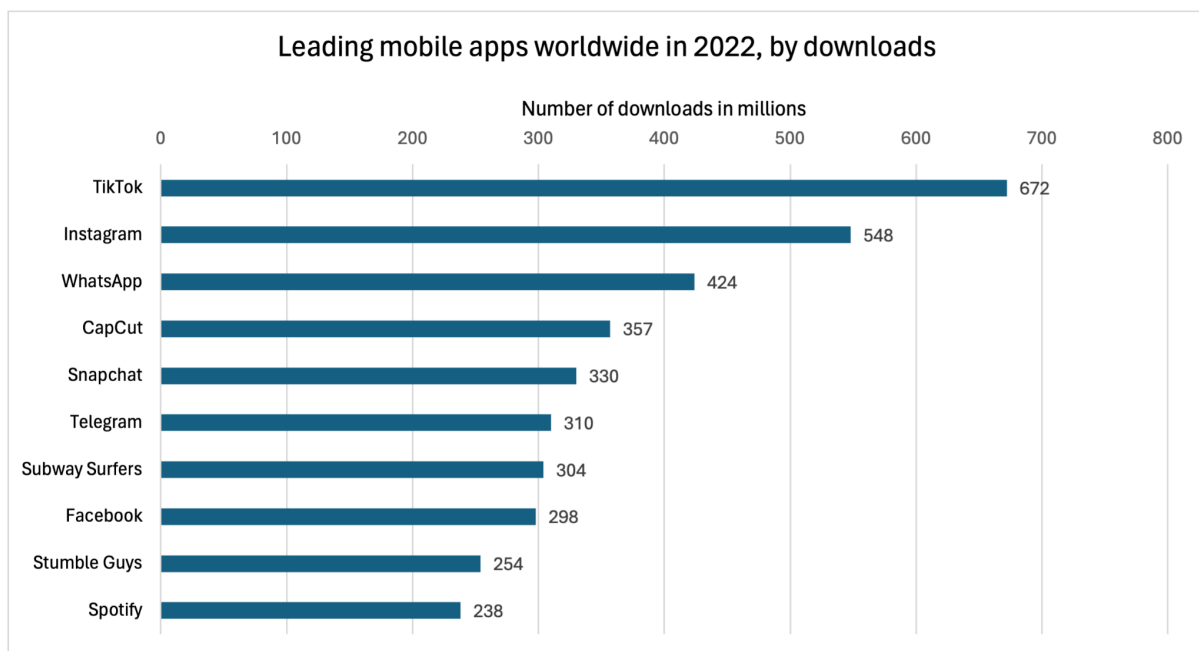


Figure 1: Leading mobile apps worldwide in 2022, by downloads [2]

The vertical axis of the figure represents the list of the top 10 most popular mobile apps while the horizontal axis represents the number of downloads (in millions) for each popular mobile app

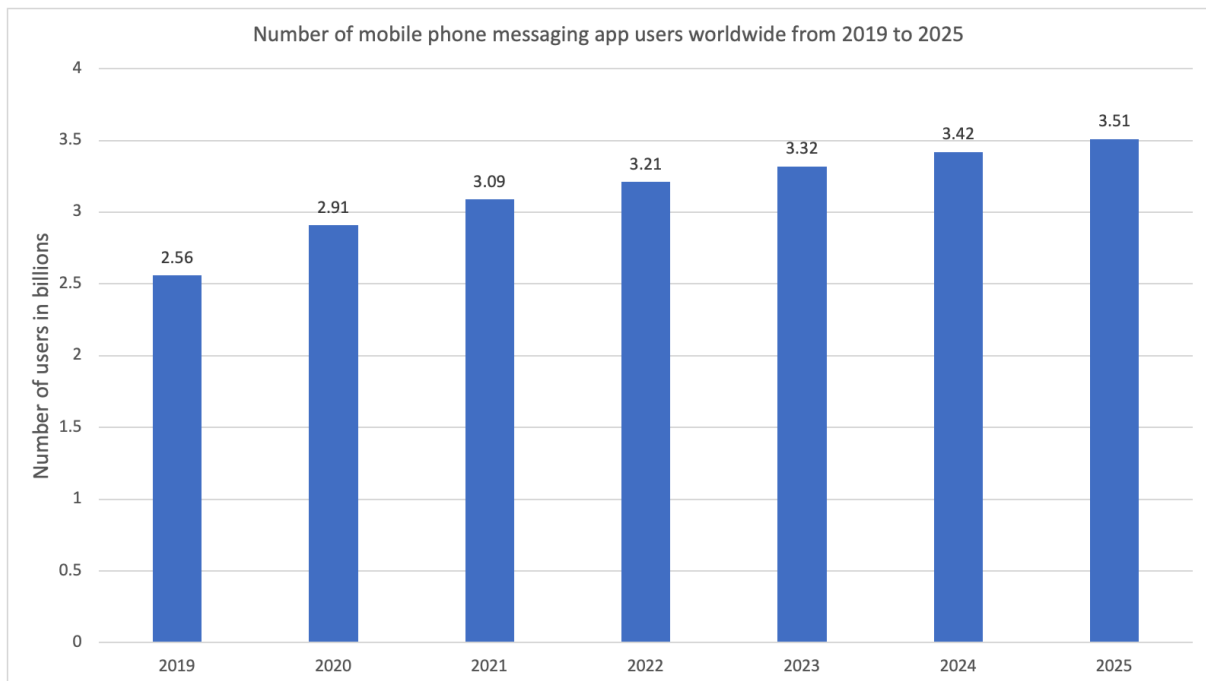


Figure 2: Number of mobile phone messaging app users worldwide from 2019 to 2025 [3]

The horizontal axis of the figure represents the year while the vertical axis represents the number of mobile phone messaging app users worldwide in billions.

1.2 Motivation

The need to connect and communicate with others is embedded in human nature, and effective communication is of the utmost importance in education to improve learning quality and education outcomes. For tertiary education in particular, the delivery of messages should be in a clear, succinct, and intelligible manner so that students are motivated and capable of understanding [4]. Such an approach happens to match the context of using IM software, which is often in a direct, casual manner. Compared to using emails, students can reduce the amount of time spent on writing a defined structure, struggling with addressing or being concise with the issue they are trying to propose to the instructor, which would foster a more efficient exchange of academic ideas even after class.

Also, by including the younger generation's preferred communication tools, it can be seen as an accommodation effort that would encourage student-instructor interaction during and after lessons. Aside from face-to-face meetings and emails, IM software has become the major virtual communication approach. Generally speaking, it is noticeable that young people tend to utilize real-time collaborative or IM applications such as Google Docs, Zoom, iMessage

rather than conventional email applications like Gmail [5]. Particularly amid the COVID-19 pandemic, as online teaching has been offered in response to the suspension of face-to-face classes, applications that support real-time virtual meetings such as Zoom and Microsoft Teams have even been popularized.

Moreover, the common acceptance of incorporating emojis in IM can also allow a deeper layer of emotional exchange between teacher and student, which could potentially assist in fostering academic connections for the future. In the 2021 study by Adobe, over half of the respondents reflected that they are more at ease in showing how they feel via emojis than talking on the phone or face-to-face [6]. With the congruent visual cues with text, it is expected to facilitate better student-teacher communication and even make a positive impression in digital conversations [7].

Hence, developing an IM software that combines standard messaging as well as new interactive functions such as quizzes for both students and educators, would assist traditional one-way teaching by adding multi-directional interaction.

1.3 Project Overview and Objectives

This project aims to develop an instant messaging software for both students and teachers to assist in creating a much more effective learning environment and prompting productivity. During the COVID-19 pandemic, online learning had become a common practice that both teachers and students had gotten used to interacting virtually via IM software like Zoom, Microsoft Teams, and Kahoot. Despite the return to normalcy that lessons are conducted face-to-face again, the functions of such applications have become indispensable particularly as many struggle with physical interaction. Although students and teachers can communicate via existing IM software like WhatsApp, while creating and responding to quizzes via some existing game-based learning platform like Kahoot, it is not a convenient and effective approach for them to conduct those academic activities as they are required to install multiple apps into their device in order to use all of those functions. Therefore, it is essential to design and develop a software that carries the functions of real-time chatting, creating and responding to interactive quizzes during class. Here are the project objectives:

- To combine the real-time chatting as well as the quizzes creating and responding functionalities into a single app
- To provide provide instructors with a suitable graphical interface which enables designing, editing and distributing interactive tests during lessons to encourage student-teacher interaction
- To function as a consultation platform for students for academic inquiry
- To allow students and teachers to have academic discussion privately or in groups
- To support message archiving that app users can check their past chats
- To allow teachers to access information regarding quiz questions to understand students' learning progress

1.4 Report Outline

There are four remaining sections in this report. For section 2, the content will focus on the methodology, which describes the software and tools used in this project. Moreover, all the system design and architecture will be discussed. For section 3, the content will focus on the results and discussion, which describe the result obtained throughout this project. For section 4, the content will focus on the future work. Finally, for section 4, the content will focus on the conclusion.

2. Methodology

In this chapter, the content will focus on the methodology of this project, which is primarily categorized into two sections. The first section will discuss the tools or software used in this project, while the second section will discuss the project's design aspects.

2.1 Software and Tools Concerned

2.1.1 Database

MongoDB is used in this project as it allows users to manage large amounts of data in a more flexible and efficient manner when compared to relational databases. Different from relational databases which store data in tabular form, MongoDB is a document-oriented NoSQL database that stores information in a key-value pair format inside documents where each document is regarded as a record in a particular collection. For each document inside a particular collection (see Figure 3), the schema is flexible, in other words, there is no requirement that all documents inside a particular collection must have the same field [8].

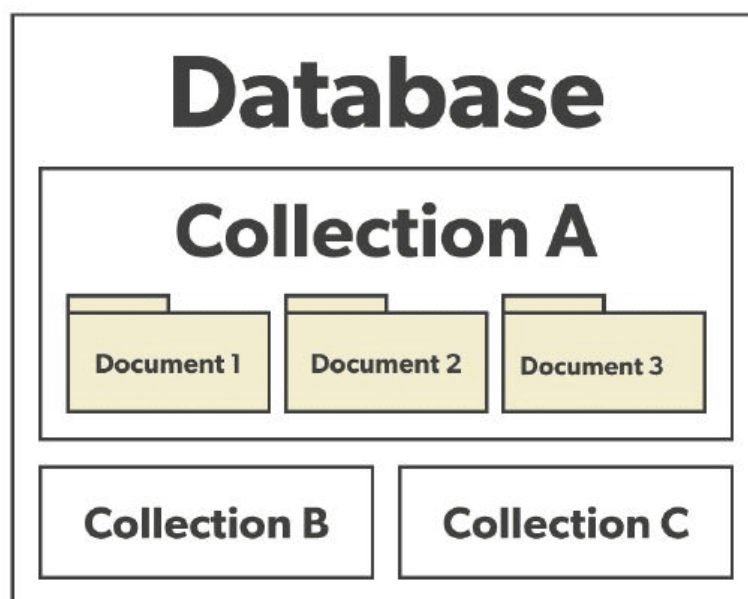


Figure 3: Hierarchical structure of document-oriented database [9]

More than one collection (same as table in relational database) can be defined inside a particular database. In each collection, it is possible to have multiple documents (same as records in relational database) and each document is regarded as a record. For documents inside a collection, their schema need not be the same.

2.1.2 Frontend

The instant messaging software in this project is built in the form of progressive web app (PWA). The PWA is an "advanced" version of the traditional web application that is developed through web platform technologies like JavaScript, HyperText Markup Language (HTML), and Cascading Sheet (CSS) [10]. Benefitting from developing through web platform technologies, the PWA offers cross-platform running without making any platform-specific adjustments. In other words, the PWA simply needs a single codebase to function across all platforms and devices. Moreover, different from a traditional web app, PWA is able to provide user experience like a native app which allows users to install the app into their device locally and offers the ability to keep on operating while it is in the background or even offline [10].

Manifest file and service worker are major components for building a PWA. The manifest file is a JavaScript Object Notation (JSON) file that informs the browser how the PWA ought to operate when it is installed on a user's computer or mobile device [11]. In other words, whenever the PWA is added to the user's device home screen or launched from the user's device home screen, the manifest file will be fully functional to manage the PWA appearance and behavior. For the manifest file in particular, it mainly consists of the name of the PWA, the icon of the PWA, the start uniform resource locator (URL) of the PWA, the theme color of the PWA as well as the orientation of the PWA [11]. On the other hand, for the service worker, it is the critical component that allows the PWA to keep on operating while it is in the background or even offline. It acts as an intermediary between the browser and server to handle different network requests (see Figure 4), including push notifications, browser cache data storage, and retrieval [12]. Benefitting from the push notifications feature given by the service worker, the instant messaging software user is allowed to get messages even when their device's browser is not opened.

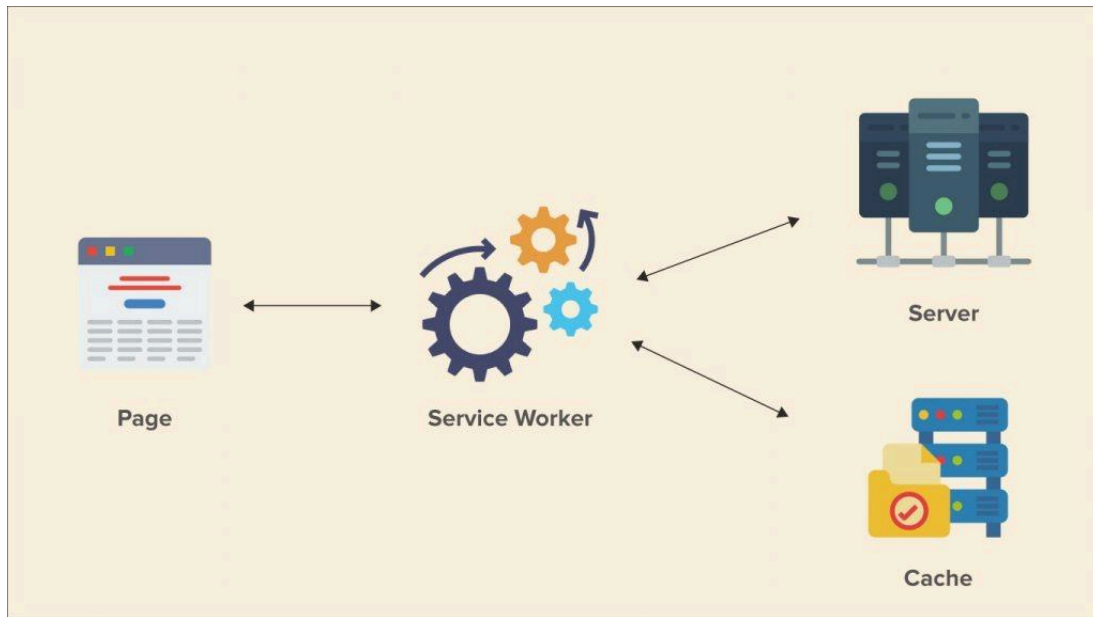


Figure 4: Relationship among service worker, user's device web browser and server [13]

For user interface (UI) designing, Figma is used as it is a free-of-charge web-based design tool that supports real-time collaboration and requires no programming background knowledge. Benefiting from the real-time collaboration capabilities and no programming background knowledge required, the efficiency of the design process is significantly enhanced. For UI building, React is used. React is a popular JavaScript library for building interactive UI of web applications. It makes use of the concept of virtual Document Object Model (DOM) to render the UI, where DOM is a representation of a set of objects that exist in a web document. With the help of virtual DOM, the UI of the app can be rendered much faster. In addition, in order to shorten the development cycle, the Vite build tool is utilized in conjunction with React as it makes the development of the server faster and allows for real-time source code updates.

2.1.3 Backend

The server of this web application is developed by Node.js. Node.js is an open-source cross-platform JavaScript runtime environment that is mainly used for building web servers [14]. JavaScript runtime environments can be simply classified into two different categories, which include the browser's runtime environment and node runtime environment. Browser's runtime environment refers to JavaScript code being executed on the web browser while node runtime environment refers to the JavaScript code being executed without the web browser [15]. For Node.js, it is the latter. Moreover, Node.js makes use of a non-blocking,

event-driven I/O model to handle multiple connections and requests, which means that Node.js has the ability to immediately identify and respond to events without waiting for the previous process to finish [16]. In addition, with the help of npm, which is a large-scale library that offers different functionalities and tools to Node.js, the server development process can be effectively facilitated.

Routing is a crucial concept in web applications as it is a key technique to perform page switching within the app. It allows developers to define how an application should react to various URLs, and determines what content to display to the user at any given time. In other words, routing directly impacts user experience and the overall functionality of the web application. In order to effectively address routing and request-related challenges, Express Framework is used together with Node.js in this project as Express Framework is able to provide features that allow developers to create APIs in a more simplified and convenient manner, making network requests easier to be handled in the back-end server.

The original plan to implement the instant messaging feature was to utilize the WebSocket protocol, but due to its complexity and lack of network issue reconnection support [17], using Server-Sent Events (SSE) in conjunction with traditional Hypertext Transfer Protocol (HTTP) requests was chosen as a more suitable alternative. Similar to the WebSocket protocol, SSE is designed for applications needing real-time updates. However, SSE implementation is simpler than that of WebSocket. Unlike WebSocket, which supports two-way communication, SSE only offers one-way communication from the server to the client. As a result, SSE is employed for sending messages from the server to the client, while traditional HTTP requests are used for communication from the client to the server.

2.1.4 Deployment

The server of this web application is hosted on Amazon Elastic Compute Cloud (Amazon EC2) as Amazon EC2 is able to offer flexible and scalable virtual computers that users can rent based on their needs. Due to no extra hardware or computers required, users only need to complete the payment in order to immediately access the service, it is possible to reduce maintenance costs and boost the app development process.

The database of this web application is hosted on a Database as a Service (DBaaS) which is called MongoDB Atlas. DBaaS is a cloud-based software that allows users to access cloud databases without requiring the users to set up and maintain the supporting infrastructure [18]. In order to simplify the database management process, we deploy the MongoDB Atlas to a cloud service provider. Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform are cloud service providers that are supported by MongoDB Atlas. Since the server is deployed to AWS, MongoDB Atlas is deployed to AWS as well.

2.1.5 Document and Media Files Storage

The ability to share document files and images is one of this instant messaging software's primary features. To efficiently manage and store these shared files from users, this IM software uses S3 bucket, a cloud storage solution provided by AWS, rather than storing them directly in the MongoDB database. Databases are generally used for smaller size data due to their limited capability in handling large binary files like images or documents. By making use of S3 bucket, the software enhances the storage and retrieval process of these larger files, as S3 is tailored for such purposes. This method not only boosts the performance of the MongoDB database but also enhances the file-sharing and accessing experience for users of the IM application.

2.1.6 Content Delivery Network

To ensure consistent and optimized internet content loading times regardless of the app user's geographical location in relation to the server, Amazon CloudFront CDN has been utilized in this project. Amazon CloudFront CDN is a Content Delivery Network (CDN) service provided by AWS for efficient data distribution [19]. By leveraging its global deployment and caching capabilities, the CDN serves internet content from servers located in close proximity to app users, resulting in faster data retrieval and enhanced app user experience. App users situated closer to the server experience shorter round-trip times, while those further away benefit from reduced latency and improved performance. This strategic implementation of Amazon CloudFront CDN enables faster and more reliable delivery of internet content, contributing to a seamless and efficient user experience within the app.

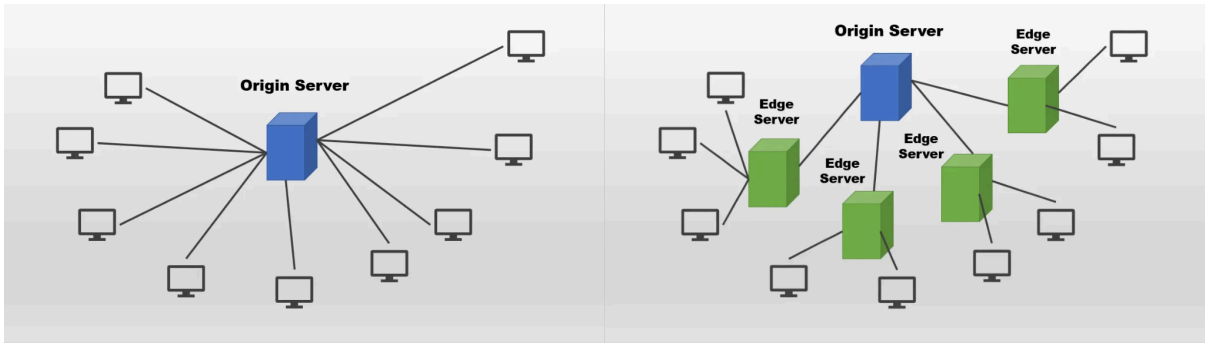


Figure 5: Content Delivery Network (CDN) implementation [20]

Without a Content Delivery Network (CDN), clients may experience varying round-trip times based on the distance between the server and the client. Clients located far from the server typically endure higher latency. However, with a CDN in place, clients can access internet content through the nearest edge server, resulting in significantly reduced round-trip times.

2.1.7 Push Notification Service

Firebase Cloud Messaging (FCM) technology is adopted in this project for implementing the push notification feature. Benefitting from the FCM technology, users of the app will receive notifications whenever new chat messages are posted in a specific group, even if the app is running in the background or offline. This functionality reminds app users to check the latest chat messages and ensures that users stay updated and connected at all times.

2.2 Designs

2.2.1 Database Design

To create the database design, we focus on easy updating and retrieval of data without repeatedly storing data into the database. To achieve this effect, we decided to make use of the DBRefs concept in MongoDB. DBRefs allows us to reference a particular document that is stored in another collection. This helps us to group all the similar documents into a collection and manage the data in a tidy manner.

The database contains 3 collections (User, ChatRoom, and OTP) and its design is as follows (see Figure 6 and Figure 7):

For OTP collection,

Field	Description
email	store the email address of the user who are logging into the

	system or registering for an account
opt	store the one-time password of the user who are logging into the system or registering for an account
createdAt	store the time that the one-time password is generated; control the one-time password expiry time

Table 1: Database OTP collection structure

For User collection,

Field	Description
userId	store the unique ID of the app user
nickname	store the nickname of the app user
firstname	store the first name of the app user
lastname	store the last name of the app user
role	indicate whether the app user is student or teacher
email	store the email address of the app user
firstMajor	store the first major program of the student user (not available for teacher)
secondMajor	store the second major program of the student user (not available for teacher)
minor	store the minor program of the student user (not available for teacher)
date_of_birth	store the date of birth of the app user
profile_image	store the profile image object of the app user
year_of_study	store the year of study of the student user (not available for teacher)
createdAt	store the registration date of the app user
registrationToken	store the unique registration token of the app user
chatRoomMember	store a list of documents and each document provide information about the app user in a chat room (refer to Table 3)

Table 2: Database User collection structure

For documents in charRoomMember List,

Field	Description
chatRoomId	store the unique ID of a particular chat room; in reference to a document in ChatRoom collection
join_date	store the date that the app user joined a particular chat room
left_date	store the date that the app user left a particular chat room
isAdmin	indicate whether the app user is a group administrator

Table 3: Database chatRoomMember List Document structure

For ChatRoom collection,

Field	Description
chatRoomId	store the unique ID of the chat room
chatRoomImage	store the profile image object of the chat room
chatRoomName	store the name of the chat room
privacy	indicate whether the chat room is public or private
createAt	store the creation date of the chat room
message	store a list of documents and each document provide information about a chat message in a chat room (refer to Table 5)
quiz	store a list of documents and each document provide information about a quiz in a quiz room (refer to Table 7)

Table 4: Database ChatRoom collection structure

For documents in message List,

Field	Description
messageId	store the unique ID of a particular message
messageFrom	store the unique user ID to indicate who send out the message; in reference to a document in the User collection
sent_date	store the time that the chat message is sent to a group
messageType	store the type of the message
messageData	store the chat message
messageStatus	store a list of documents and each document provide information about the message status for a group member (refer to Table 6)

Table 5: Database message List Document structure

For documents in messageStatus List,

Field	Description
userId	store the unique ID of the app user; in reference to a document in the User collection
read_date	store the time that the app user read the chat message
deliver_date	store the time that the chat message is delivered to the app user

Table 6: Database messageStatus List Document structure

For documents in quiz List,

Field	Description
quizId	store the unique ID of a particular quiz
quizQuestion	store the quiz question of a particular quiz
quizTopics	store a list of related topics of a quiz
quizOptions	store a list of multiple choice options of a quiz
quizModelAnswer	store the quiz model answer
quizFrom	store the quiz creator user Id
sent_date	store the quiz creation date
end_date	store the quiz expiry time
quizImage	store the image object for quiz question enrichment
studentQuizResponse	store a list of documents and each document provide information about a quiz submission given by a student (refer to Table 8)

Table 7: Database quiz List Document structure

For documents in studentQuizResponse List,

Field	Description
userId	store the unique ID of a student who has submitted a quiz; in reference to a document in User collection
quizAnswer	store the multiple choice option selected by the student

quizResponse_date	store the time that the student submit the quiz
-------------------	---

Table 8: Database studentQuizResponse List Document structure

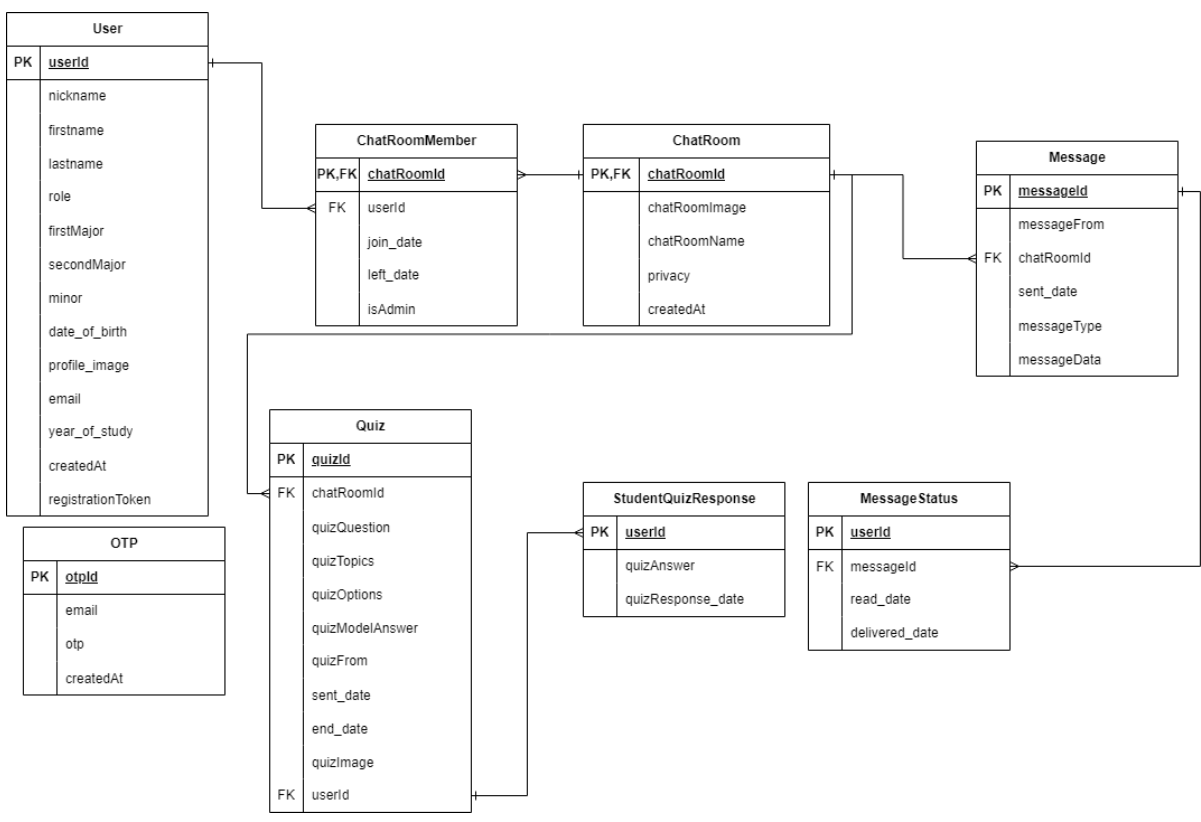


Figure 6: Database data model diagram for this project

User	ChatRoom	OTP
<pre> { "userid": ObjectID, "nickname": String, "firstname": String, "lastname": String "role": String, "email": String, "firstMajor": String, "secondMajor": String, "minor": String, "date_of_birth": Date, "profile_image": String, "year_of_study": String, "createdAt": Date, "registrationToken": String, "chatRoomMember": [{ "chatRoomId": ObjectID, "join_date": Date, "left_date": Date, "isAdmin": Boolean }] } </pre>	<pre> { "chatRoomId": ObjectID, "chatRoomImage": String, "chatRoomName": String, "privacy": String, "createdAt": Date, "message": [{ "messageId": ObjectID, "messageFrom": ObjectID, "sent_date": Date, "messageType": String, "messageData": String, "messageStatus": [{ "userid": ObjectID, "read_date": Date, "delivered_date": Date }] }] "quiz": [{ "quizId": ObjectID, "quizQuestion": String, "quizTopics": [String], "quizOptions": [String], "quizModelAnswer": String, "quizFrom": ObjectID, "sent_date": Date, "end_date": Date, "quizImage": String, "end_date": Date, "studentQuizResponse": [{ "userid": ObjectID, "quizAnswer": String, "quizResponse_date": Date }] }] } </pre>	<pre> { "email": String, "otp": String, "createdAt": Date } </pre>

Figure 7: Documents in User collection, ChatRoom collection, and OTP collection

2.2.2 System Architecture Design

The system architecture design of this project can be divided into two key components: front-end and back-end.

In the front-end, crucial resources like HTML, CSS, and JavaScript files are fetched upon launching the application and cached in the browser for rapid access. Moreover, the front-end will communicate with the back-end server via REST API calls. Additionally, to enable push notification features, Firebase Cloud Messaging will collaborate with service worker to deliver real-time notifications to users.

On the back-end side, the server will interact with the MongoDB database using the Mongoose API to execute data operations effectively. Whenever there is file transferring, files will be stored in an S3 bucket using the aws-sdk API.

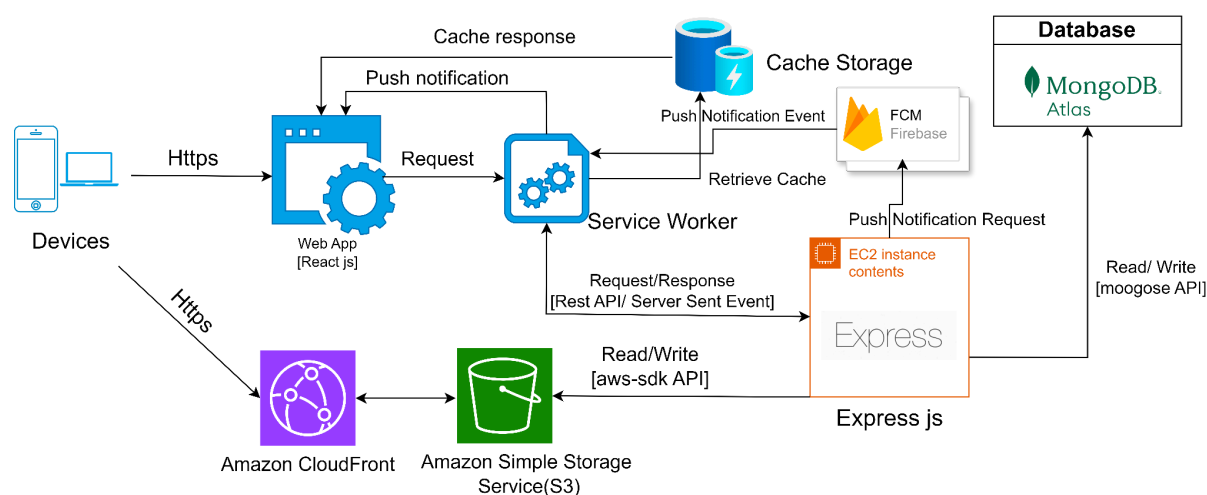


Figure 8: System Architecture

2.2.3 Push Notification

The service worker and Firebase Cloud Messaging are the major components for implementing the push notification feature in this IM software. They act as an intermediary between the client's web browser and server to handle the push notification process. The workflow of the push notification feature is as follows (see Figure 9).

For the first time the app users access the system, a pop-up message will be displayed, asking them to grant permission for the notification. Once the permission is granted, the service worker, as well as the push service will be automatically registered. Upon registration, the client's web browser will generate a unique token, which is then sent to the server. The server will then store the token in the MongoDB database for subsequent usage. Every time the server needs to send notifications to the client's browser, the server will first get the registration token from the MongoDB database and send this registration token to Firebase Cloud Messaging. After that, the service worker will work together with Firebase Cloud Messaging to deliver the notification message to the client.

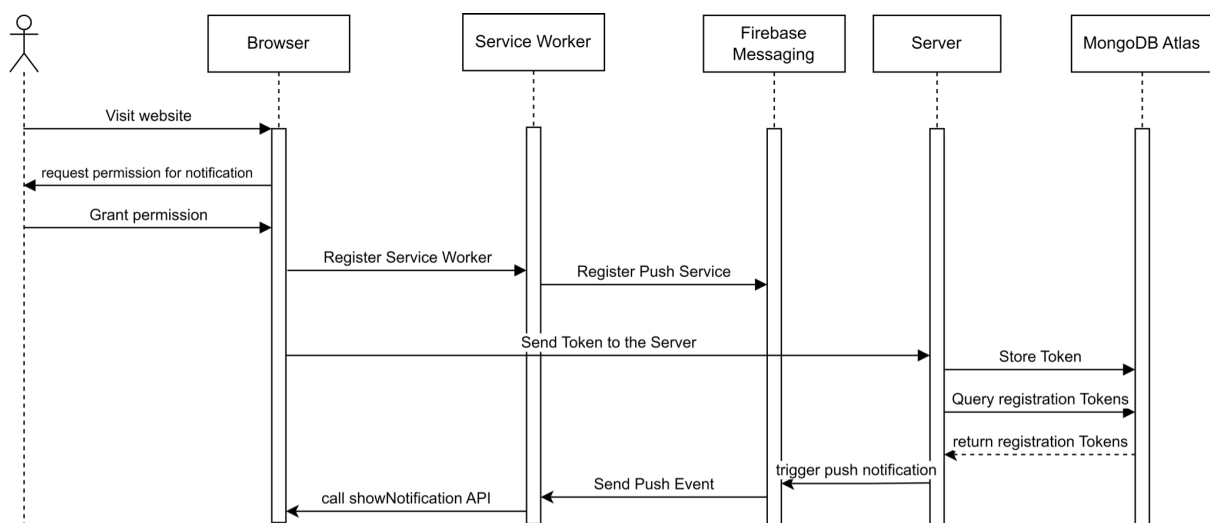


Figure 9: Sequence diagram for push notification feature

2.2.4 Amazon CloudFront CDN and S3 Bucket

To access the content stored in the S3 bucket, app users must first request access from the server. The server will retrieve the content key from the MongoDB database upon receiving the request. Using this key and its private key, the server will generate a signed URL for temporary and secure access, which will be sent to the user. The user can then use this signed URL to request the content from Amazon CloudFront CDN. The CDN will verify the signed URL and check its cache for the requested content. If the signed URL is valid and the content is in the cache, it will be returned to the user quickly, reducing data loading time. However, if the content is not in the cache, the CDN will retrieve it from the S3 bucket, resulting in a longer data loading time for the user.

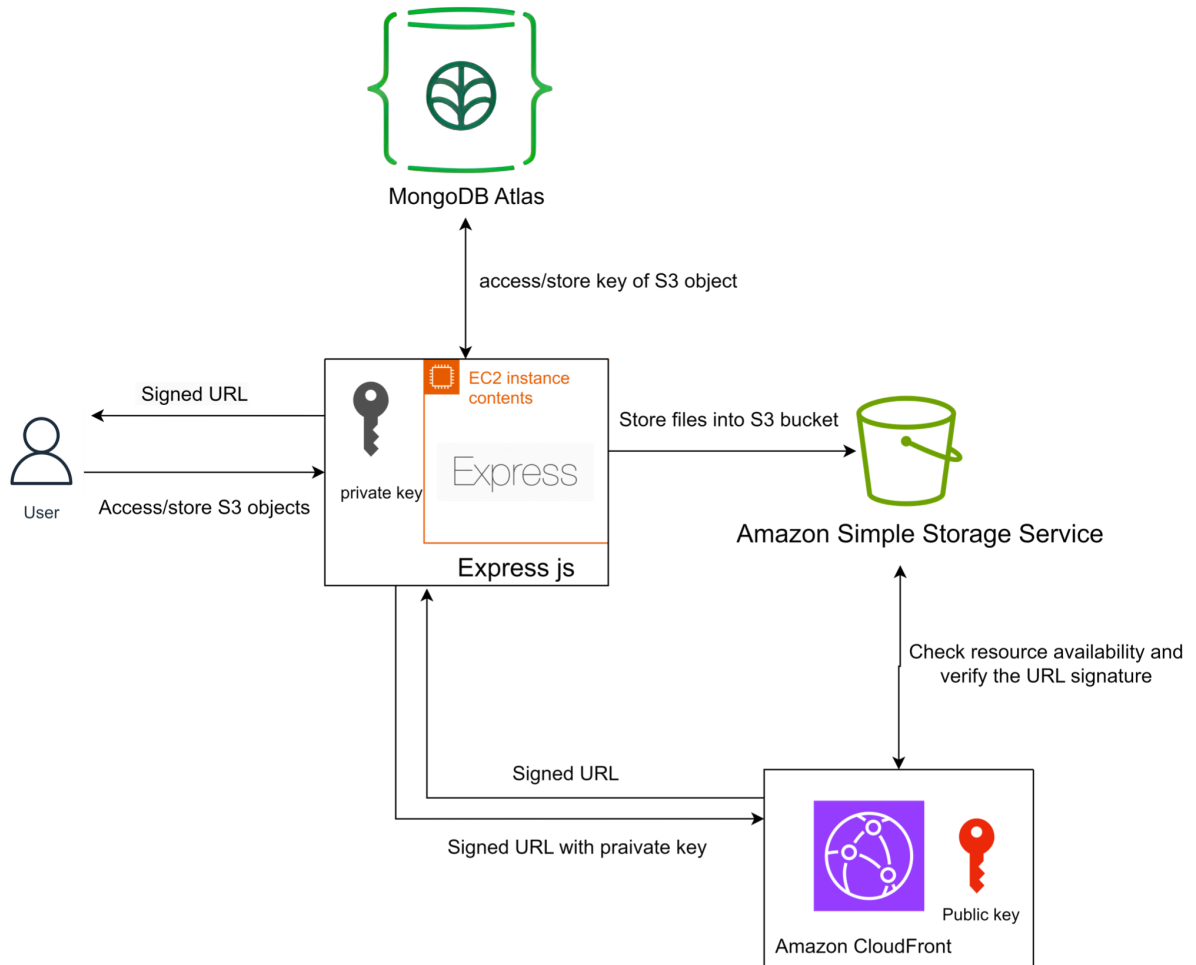


Figure 10: Accessing documents and image files from Amazon CloudFront CDN and S3 Bucket

3. Result

This chapter will focus on the result obtained through this project. There are 12 sections in this chapter and each of them will provide a detailed description of a system feature.

3.1 Login

To log into the system, the app users are required to specify their user identity. In order to simplify the information input process and enhance control over user inputs, a drop-down menu with predefined options has been provided on the login page. By simply clicking the designated button on the login page, the drop-down menu will be opened, and a list of predefined options will be shown to the user. The app users can choose between student or teacher to specify their user identity.

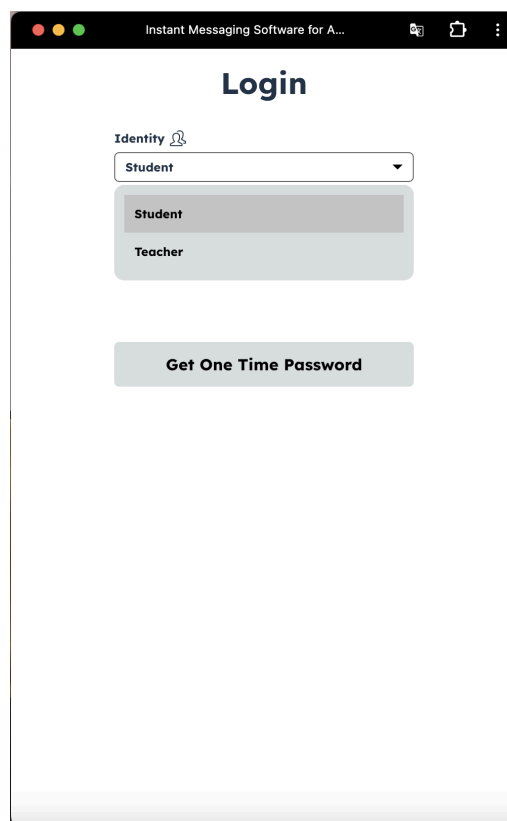
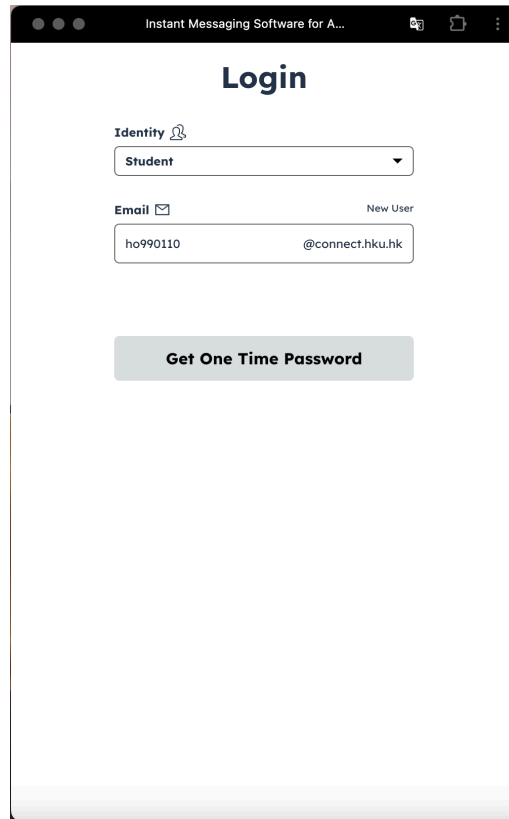


Figure 11: Login page activated user identity select drop-down menu


After indicating the user identity, the app users must enter their email address in the designated space provided. Depending on the user identity selected, the email address domain will be pre-populated by the system for the user. For students, the pre-populated email

domain will be “@connect.hku.hk” (see Figure 12), while for teachers, the pre-populated email domain will be “@hku.hk” (see Figure 13).




Instant Messaging Software for A...

Login

Identity 

Student

Email  New User

ho990110 @connect.hku.hk

Get One Time Password

Figure 12: Login page pre-populated email domain (for student)

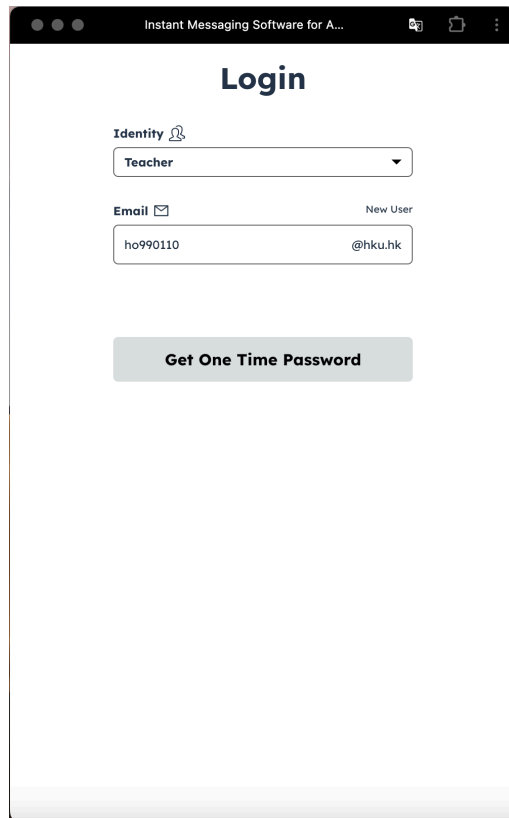


Figure 13: Login page pre-populated email domain (for teacher)

After entering the email address to the designated space provided, the app users can click the "Get One Time Password" button, located at the bottom of the page, to request a one-time password (OTP). Upon clicking this button, the OTP input field will appear on the page. Simultaneously, a unique OTP will be automatically generated by the system and sent directly to the app user's email inbox (see Figure 14 and Figure 15). Users must then retrieve the OTP from their mailbox and enter it into the provided spaces to complete the login process (see Figure 16). This verification mechanism guarantees that only HKU students or teachers can access the system. Moreover, it is important to note that the OTP will expire after 5 minutes, the app user must finish the login process within this short period, otherwise, they have to press the "Resend" label provided on the page to get a new OTP and repeat the login process.

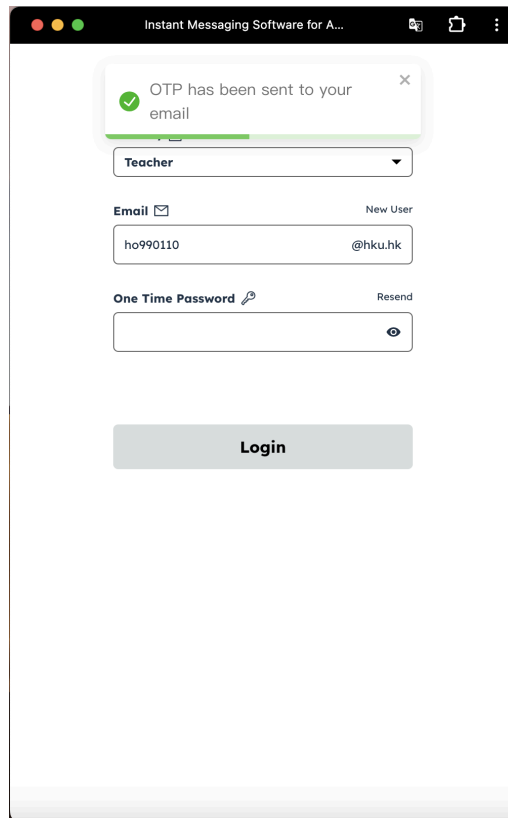


Figure 14: Requesting one-time password

After clicking the “Get One Time Password” button, a notification will be displayed to notify the app user. Moreover, the one-time password input field will appear on the screen to allow the app user to provide the OTP they received.

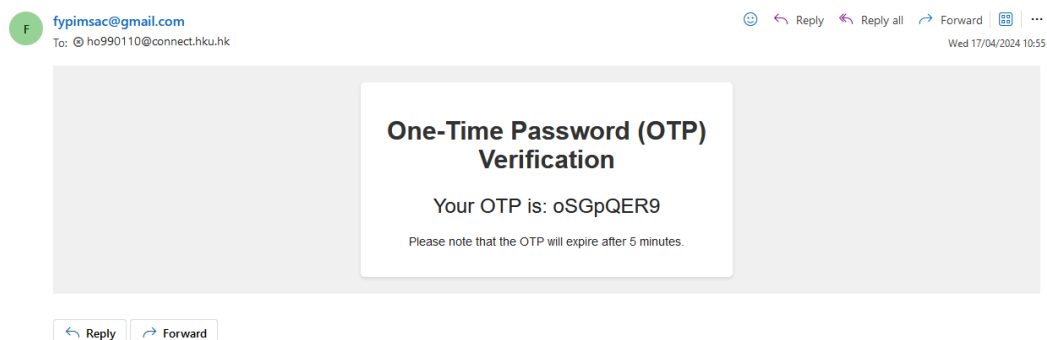


Figure 15: Retrieve an OTP generated by the system from the app user’s mailbox

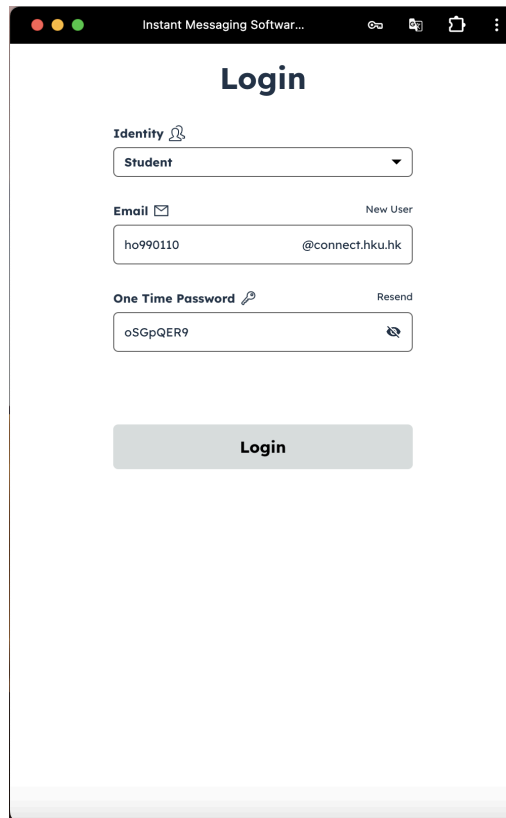


Figure 16: Providing the received OTP to the designated field

In the OTP input field, a button with an “eye” symbol is provided to allow the app user to optionally show or hide the OTP that they have entered.

3.2 Registration

For new users, they have to press the “new user” button in the login page to register for an account. Upon clicking the “new user” button, the new user will be directed to the registration page.

Similar to the login process, to register for an account, the new app users have to specify their user identity and enter their email address in the designated field to get an OTP generated by the system. Upon retrieving the OTP from the mailbox and providing it to the OTP input field, the app new user can press the “Verify One Time Password” button, which is located at the bottom of the page, to allow the system to validate the OTP (see Figure 17). This mechanism ensures that only HKU students or teachers can register for an account.

The screenshot shows a mobile application interface for registration. At the top, the title 'Registration' is centered. Below it, there are three input sections: 'Identity' with a dropdown menu showing 'Student', 'Email' with a text field containing 'ho990110' and '@connect.hku.hk', and 'One Time Password' with a text field and a 'Resend' link. At the bottom, there are two buttons: 'Verify One Time Password' and 'Cancel'.

Figure 17: Registration page OTP validation

Once the OTP is successfully validated by the system, the new app users will be asked to provide more detailed personal information to complete the registration process. For students, they are required to provide their name (separated by first name, last name, and nickname), program of study (separated by first major program, second major program, and minor program), year of study, personal profile image, as well as the date of birth (see Figure 18). While for teachers, they will not be asked to provide their program of study as well as the year of study, they are only required to provide their name, personal profile image, and date of birth (see Figure 19).

The figure shows two side-by-side screenshots of a web application's registration form. Both screenshots are titled "Registration" and are part of a window labeled "Instant Messaging Software for A...".

Left Screenshot (Student Registration):

- Identity:** A dropdown menu with "Student" selected.
- Email:** A text input field containing "ho990110" and a suffix "@connect.hku.hk".
- One Time Password:** A text input field with masked characters "....." and a "Resend" link.
- Name:** Three stacked text input fields for "FirstName", "LastName", and "NickName".
- Program:** Three stacked text input fields for "FirstMajor", "SecondMajor Optional", and "Minor Optional".
- Year of Study:** A dropdown menu with "Year 1" selected.
- Date of Birth:** A date picker field.
- Profile Image:** A field with a "Remove" link and an upload icon.
- Buttons:** "Submit" and "Cancel" buttons at the bottom.

Right Screenshot (Teacher Registration):

- Identity:** A dropdown menu with "Teacher" selected.
- Email:** A text input field containing "ho990110" and a suffix "@hku.hk".
- One Time Password:** A text input field with masked characters "....." and a "Resend" link.
- Name:** Three stacked text input fields for "FirstName", "LastName", and "NickName".
- Date of Birth:** A date picker field.
- Profile Image:** A field with a "Remove" link and an upload icon.
- Buttons:** "Submit" and "Cancel" buttons at the bottom.

Figure 18: Required personal information for registration (for student)

The figure shows two side-by-side screenshots of a web application's registration form. Both screenshots are titled "Registration" and are part of a window labeled "Instant Messaging Software for A...".

Left Screenshot (Teacher Registration):

- Identity:** A dropdown menu with "Teacher" selected.
- Email:** A text input field containing "ho990110" and a suffix "@hku.hk".
- One Time Password:** A text input field with masked characters "....." and a "Resend" link.
- Name:** Three stacked text input fields for "FirstName", "LastName", and "NickName".
- Date of Birth:** A date picker field.
- Profile Image:** A field with a "Remove" link and an upload icon.
- Buttons:** "Submit" and "Cancel" buttons at the bottom.

Right Screenshot (Teacher Registration):

- Identity:** A dropdown menu with "Teacher" selected.
- Email:** A text input field containing "ho990110" and a suffix "@hku.hk".
- One Time Password:** A text input field with masked characters "....." and a "Resend" link.
- Name:** Three stacked text input fields for "FirstName", "LastName", and "NickName".
- Date of Birth:** A date picker field.
- Profile Image:** A field with a "Remove" link and an upload icon.
- Buttons:** "Submit" and "Cancel" buttons at the bottom.

Figure 19: Required personal information for registration (for teacher)

For students, although the second major program input field, as well as the minor program input field, have been provided on the registration page, it is not a must for students to specify that information. To more clearly guide students to fill in the information concerning the program of study, placeholder text has been added to both the second major and minor program input fields, indicating that providing this information is not mandatory. Additionally, for specifying the year of study, a dropdown menu with predefined options is available. Students are only required to select their current year from this menu to complete the relevant section of the form.

Regarding the profile image upload section, both students and teachers can click on the designated upload icon to select an appropriate image. Upon clicking the icon, a file selector will appear, allowing the user to choose and upload their desired image to the system. It is important to note that the system only accepts image files with specific extensions such as ".jpg," ".png," or ".svg." Files with other non-image extensions will be disabled in the file selector (see Figure 20). After selecting a valid image, the name of the image file will be displayed. If no profile image is provided, a default image will be automatically assigned to the user. Additionally, users can delete their uploaded image by clicking the “remove” button, after which the image file name will disappear from the display.

```
<input
  type="file"
  id="registrationProfileImage"
  className="registrationProfileImageSelector"
  accept="image/*"
  onChange={(e) => handleUploadedProfileImage(e)}
/>
```

Figure 20: File selector for uploading profile image

The “accept” attribute is set to “image/*” indicates the file input should only accept files that are images.

3.3 Home Page

Upon logging into the system, the app users will be directed to the home page. At the top of this page, the user's personal information is displayed, including their nickname, profile image, and identity. Just below the app user's personal information, two buttons labeled

“Chat” and “Quiz” are provided for granting access to the app’s major systems: the real-time chatting system and the quiz-creating and responding system. Clicking on either button will display the respective group list below (see Figure 21).

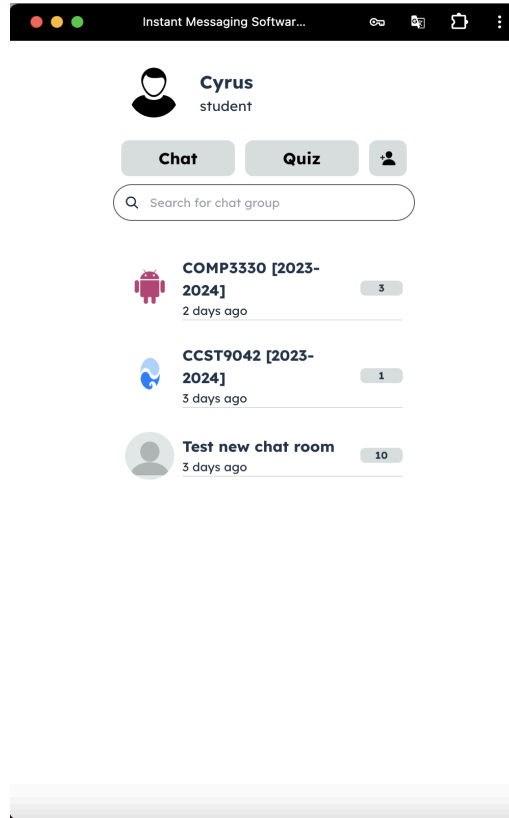


Figure 21: Home page layout

For each chat group in the chat group list, the system will display the chat group icon, the chat group name, the number of unseen chat messages, as well as the relative timestamp of the latest chat message to the user (see Figure 22). If the chat group does not contain any chat messages, the chat group creation date will be displayed.

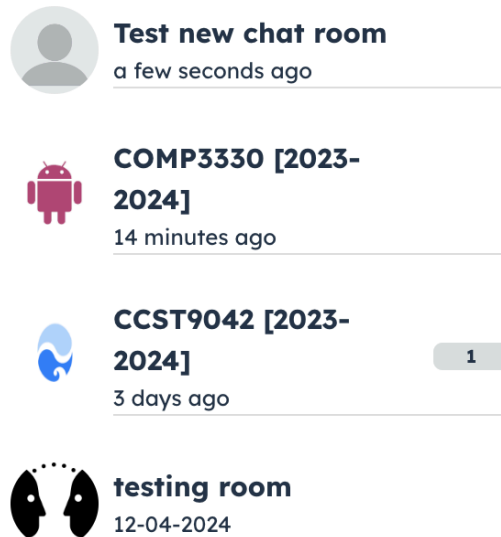


Figure 22: Example of home page chat group list

On the other hand, for each quiz group in the quiz group list, the system will display the quiz group icon, the quiz group name, as well as the relative timestamp of the latest quiz to the user (see Figure 23).

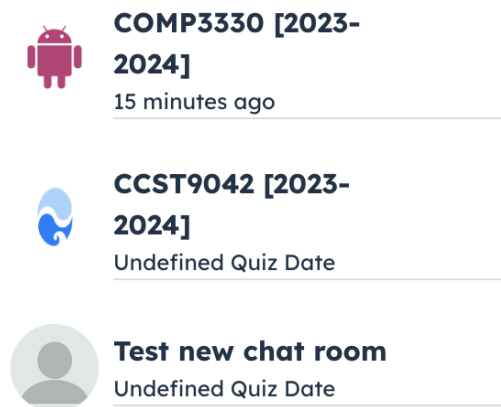


Figure 23: Example of home page quiz group list

To make it easier and more convenient for the app users to access the frequently used chat groups or quiz groups, the system will arrange the groups in both the chat group list and quiz group list based on the group's relative timestamps, placing the groups with larger timestamps at the top of the list (see Figure 24 and Figure 25).

```

//retrieve joined chat room list for chat system
const chatRoom = useGetChatRooms();
const allChatRooms = useMemo(() => {
  return (Array.isArray(chatRoom?.data) ? chatRoom?.data : [])
    .map((data_element) => ({
      chatRoomName: data_element.chatRoomName,
      chatRoomId: data_element.chatRoomId,
      chatRoomImage: data_element.chatRoomImage,
      chatRoomSentDate: data_element.sentDate,
      chatRoomCreatedAt: data_element.createdAt,
      chatRoomDateCombine:
        Number(
          moment(data_element.sentDate).format("YYYYMMDD") +
          moment(data_element.sentDate).format("HH:mm:ss").split(":").join("")
        ) ||
        Number(
          moment(data_element.createdAt).format("YYYYMMDD") +
          moment(data_element.createdAt).format("HH:mm:ss").split(":").join("")
        ),
      unseenMessageNumber: data_element.unseenMessageNumber,
    }))
    .sort(
      (compareObj1, compareObj2) =>
        compareObj2.chatRoomDateCombine - compareObj1.chatRoomDateCombine
    );
}, [chatRoom.data]);

```

Figure 24: Sorted chat group list creation

After retrieving a list of chat groups from the backend server, the system generates a corresponding list of chat group objects. Each of these newly created objects includes a property called “chatRoomDateCombine”. The system converts the timestamp associated with each chat group into an integer, which is then assigned to the “chatRoomDateCombine” property. Subsequently, the list of chat group objects is sorted based on the “chatRoomDateCombine” property.

```

//retrieve joined quiz room list for quiz system
const quizRoom = useGetQuizRoom();
const allQuizRooms = useMemo(() => {
  return (Array.isArray(quizRoom?.data) ? quizRoom?.data : [])
    .map((data_element) => ({
      quizRoomName: data_element.chatRoomName,
      quizRoomId: data_element._id,
      quizRoomImage: data_element.chatRoomImage,
      quizRoomSentDate: data_element.lastQuizSentDate,
      quizRoomDateCombine:
        Number(
          moment(data_element.lastQuizSentDate).format("YYYYMMDD") +
          moment(data_element.lastQuizSentDate).format("HH:mm:ss").split(":").join("")
        ) || 0
    }))
    .sort(
      (compareObj1, compareObj2) =>
        compareObj2.quizRoomDateCombine - compareObj1.quizRoomDateCombine
    );
}, [quizRoom.data]);

```

Figure 25: Sorted quiz group list creation

Similar to the chat group list, the system will convert the timestamp associated with each quiz group into an integer, which is then assigned to the “quizRoomDateCombine” property. The list of quiz group objects will then be sorted according to the “chatRoomDateCombine” property.

Moreover, a search bar is available on the main page to allow the app users to search for their desired group. Whenever a user enters a character into the search bar, the searching algorithm will be triggered instantly. Therefore, the app users simply need to input the subset of characters from the group name, regardless of whether they are uppercase or lowercase, the system will immediately display the search result (see Figure 26 and Figure 27).

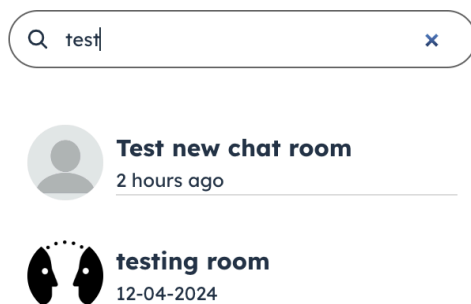


Figure 26: Chat group searching example

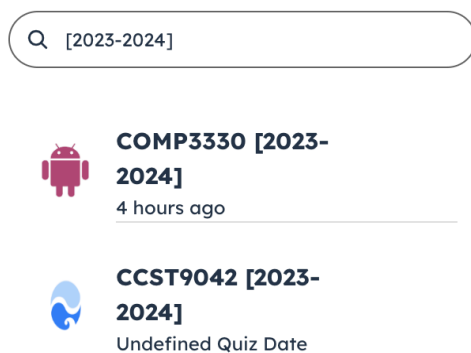


Figure 27: Quiz group searching example

3.4 Profile Page

Clicking on the app user's nickname on the home page will redirect the user to their personal profile page, where more detailed personal information is available. Apart from viewing all detailed personal information, app users can also make modifications to certain aspects. For students, they are allowed to update their nickname, second major program, as well as the minor program (see Figure 28). For teachers, they are permitted to update only their nickname (see Figure 29). It is important to note that the user's first name, last name, email

address, as well as the user ID cannot be updated as they are crucial for user identification and ensure that the system is accessed via HKU email accounts.

Instant Messaging Software for A... Instant Messaging Software for A...

Cancel Profile Save

Change profile picture

Change profile picture

First name Last name
Ka Ho Fung

Nickname ↗
Cyrus

User ID
661fc03a281bb51b2d3099b8

Email Address
ho990110@connect.hku.hk

First Major Program
BEng Computer Science

Second Major Program ↗

Minor Program ↗

Change profile picture

First name Last name
Ka Ho Fung

Nickname ↗
Cyrus

User ID
661fc03a281bb51b2d3099b8

Email Address
ho990110@connect.hku.hk

First Major Program
BEng Computer Science

Second Major Program ↗

Minor Program ↗

Logout
Logout

Figure 28: Profile page (for student)

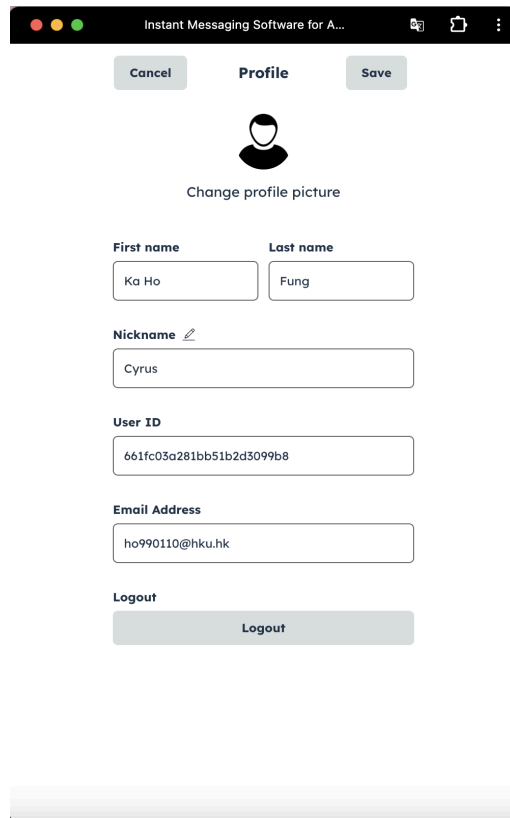


Figure 29: Profile page (for teacher)

3.5 Create/Join Chat Group

To create a new chat group or join an existing one, the app user can click the button with the "Add Person" symbol located next to the "Quiz" button on the home page. After clicking this button, the app user will be directed to the add new group page. On this page, two buttons labeled "Create new" and "Join existing" are provided for app users to access the functionalities of creating a new chat group or joining an existing chat group (see Figure 30).

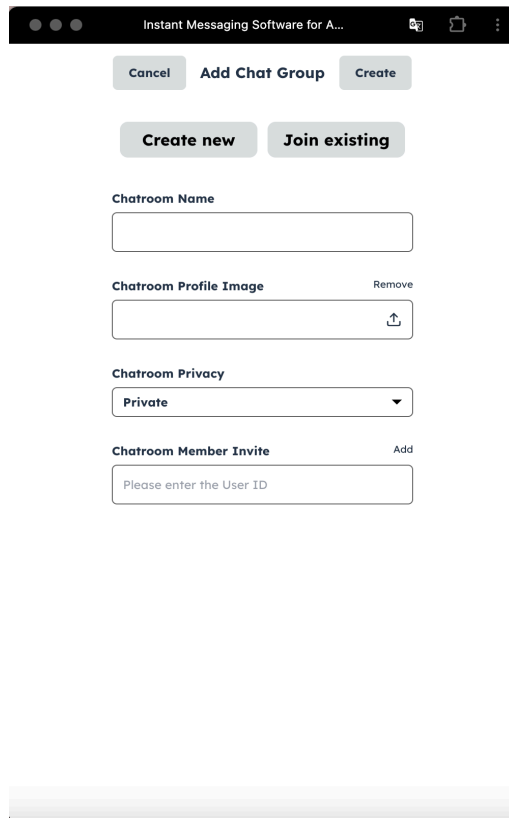


Figure 30: Add new group page layout

When the "Create new" button is clicked, app users are prompted to provide essential details to establish a new chat group. These details include specifying the chat room name and selecting the chat room privacy setting. By default, the privacy setting is configured to "private", which means that the chat group is closed and other app users cannot join the group without an invitation. Alternatively, the app user can optionally set the privacy to "public", which allows any other app users to join the chat group without being invited.

To join an existing chat group, the app user can press the "join existing" button. After that, the user will be presented with a list of all the public chat groups that the user has not yet joined. The app user can then select their preferred group from the list and click the "Join" button in the upper right corner of the page to join the group (see Figure 31).

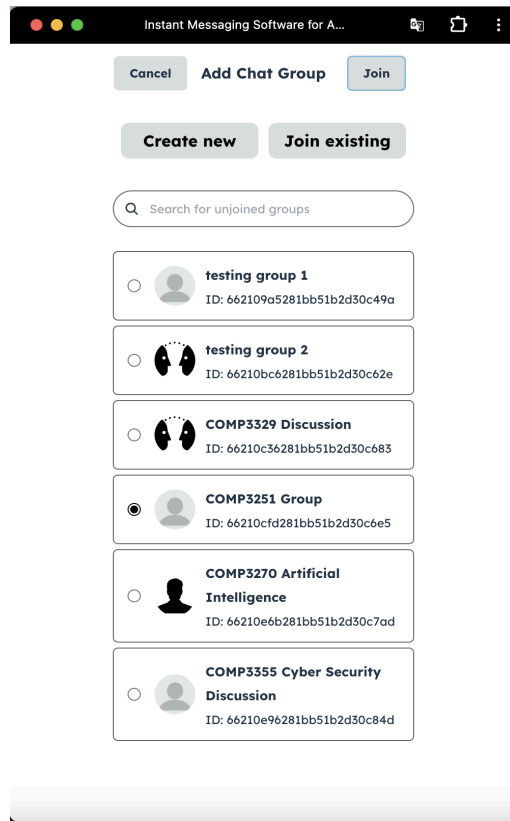


Figure 31: Example of joining an existing chat group

Additionally, to ensure that only teachers can create quizzes for their students, it is important to note that all the app users, including both teachers and students, cannot create the quiz group manually. The system will only create a quiz group for a particular chat group when there is a teacher in that chat group.

3.6 Real-time Chatting System

Moving on to the real-time chatting feature. To access this feature in the app, the app users can select one of the chat groups from the chat group list on the home page. Upon selecting a chat group, the app users will be directed to the chat page corresponding to that chat group, where they can immediately start engaging in live discussions.

The layout of the chat page mainly consists of three sections, which include the top section, the middle section, and the bottom section (see Figure 32). In the top section, the chat group name and the chat group icon are displayed to the app user, which provides clear identification of the current group. Additionally, a return button is provided to the users, which allows users to easily navigate back to the home page. In the middle section, the chat

messages that are sent by all group members are shown. The app users can scroll up to review past conversations and scroll down to see the latest messages within the chat group. In the bottom section, a text input box is provided to the users to compose and send messages to the chat group. Adjacent to the text input box, there is an emoji icon which users can click to add emojis to their messages. Furthermore, the app users can optionally share documents with the group by clicking on a button marked with a "Plus" symbol.

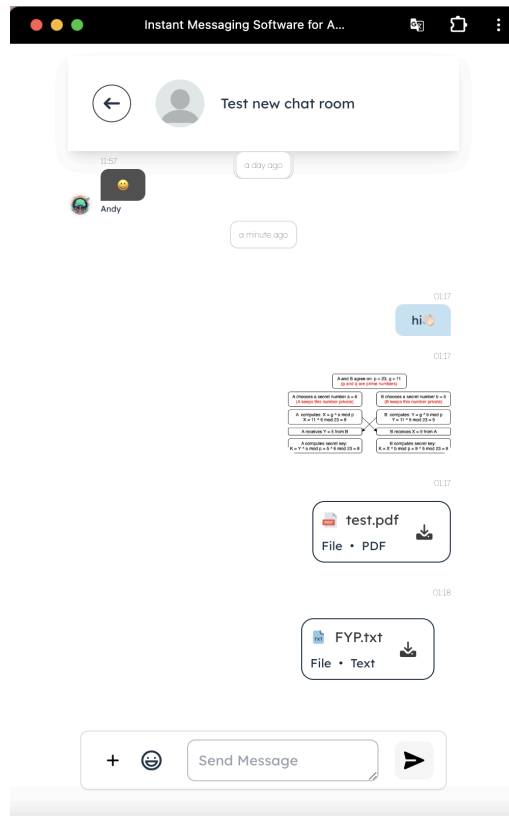


Figure 32: Example of sending text messages and sharing documents to the chat group

3.7 Chat Group Management

By clicking on the chat group name in the top section of the chat page, users will be directed to the chat room information page. In the chat group information page, users are allowed to access detailed information about the chat group.

For chat group administrators, apart from viewing all the detailed information about the chat group, they can modify various aspects of the chat group's settings, which include changing the chat group icon, updating the chat group name, altering chat group privacy, adding or

removing group members, and assigning or withdrawing chat group administrators (see Figure 33).

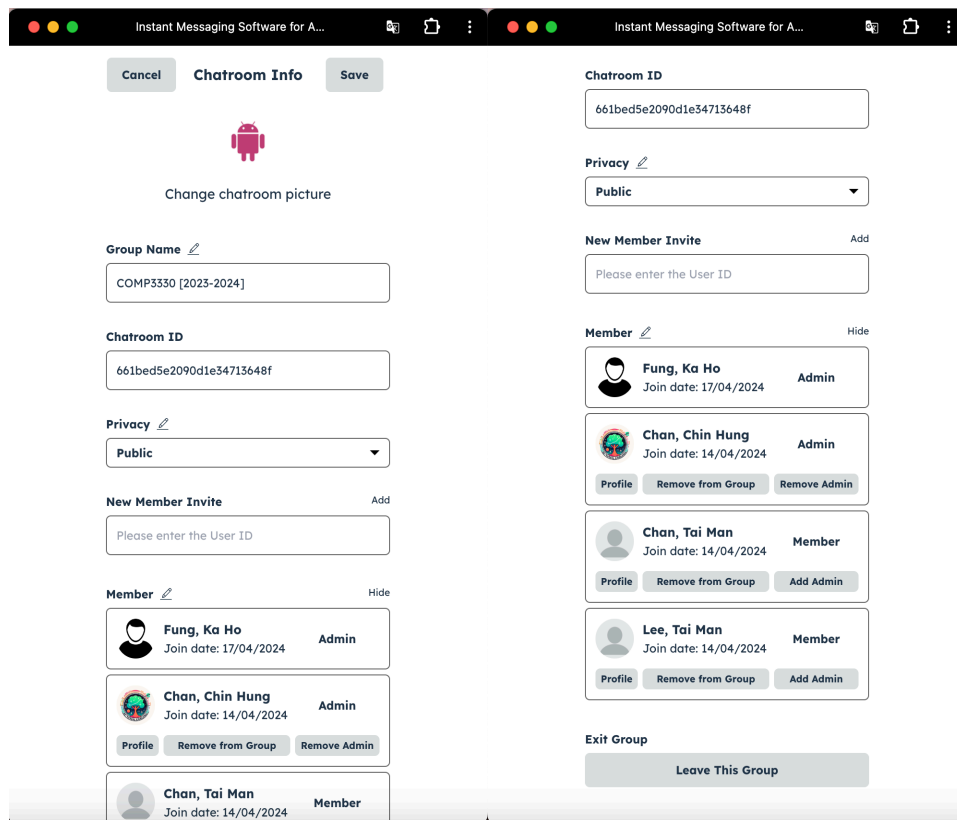


Figure 33: Chat group information page (for group administrators)

For regular chat group members, they are unable to make any modifications to chat room settings. They are only allowed to view all the chat room information (see Figure 34).

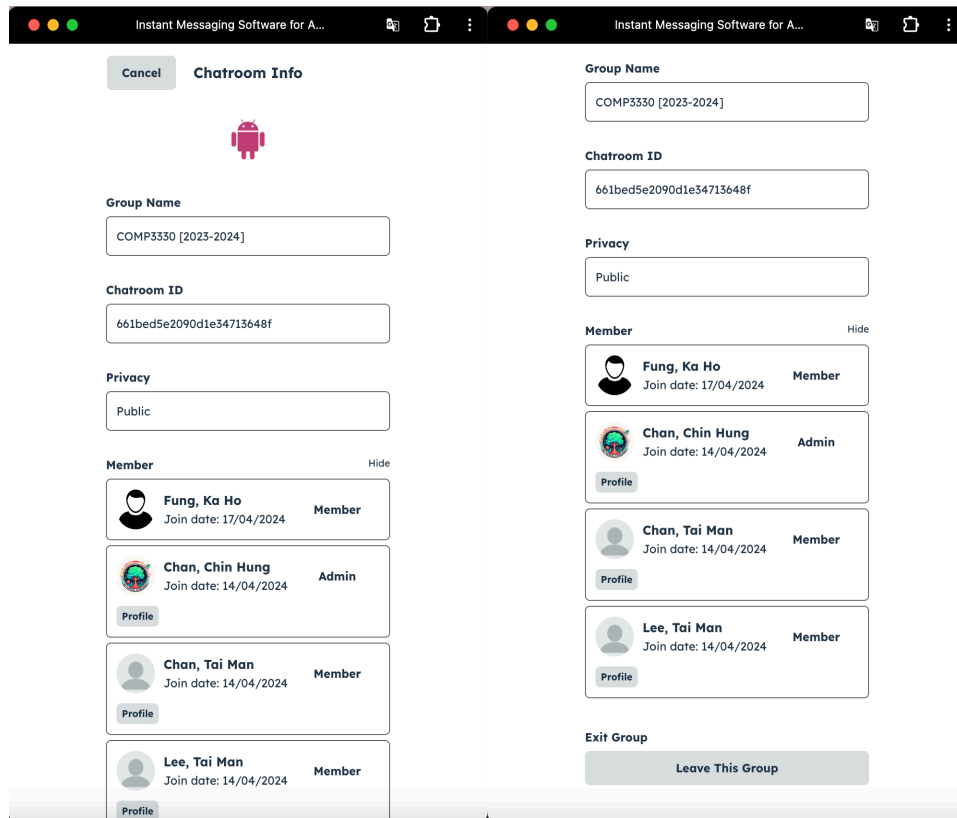


Figure 34: Chat group information page (for regular group members)

In view of the large capacity of the chat room, the number of members in the chat group member list will continue to increase, a button labeled either “Show” or “Hide” is positioned next to the “member” label to allow the app user optionally choose whether to display or conceal the list of chat group members, enhancing usability and allowing for a more customized viewing experience.

3.8 Quiz Page Access

To access the quiz creating and responding system, the app users can simply click on the “Quiz” button located on the home page. Similar to the real-time chatting system, a list of available quiz groups will then be displayed. Upon selecting a desired quiz group from the list, the app user will be directed to the quiz page where a list of quizzes that belong to that quiz group will be shown (see Figure 35). In this list, the quizzes are sorted based on their creation timestamp, the most recently created quiz will be placed at the top of the list. Additionally, each quiz entry in the list will include the date it was created and the topics related to that quiz. This feature not only helps student users easily identify quizzes by topic but also makes it more convenient for them to review and revisit relevant material as needed.

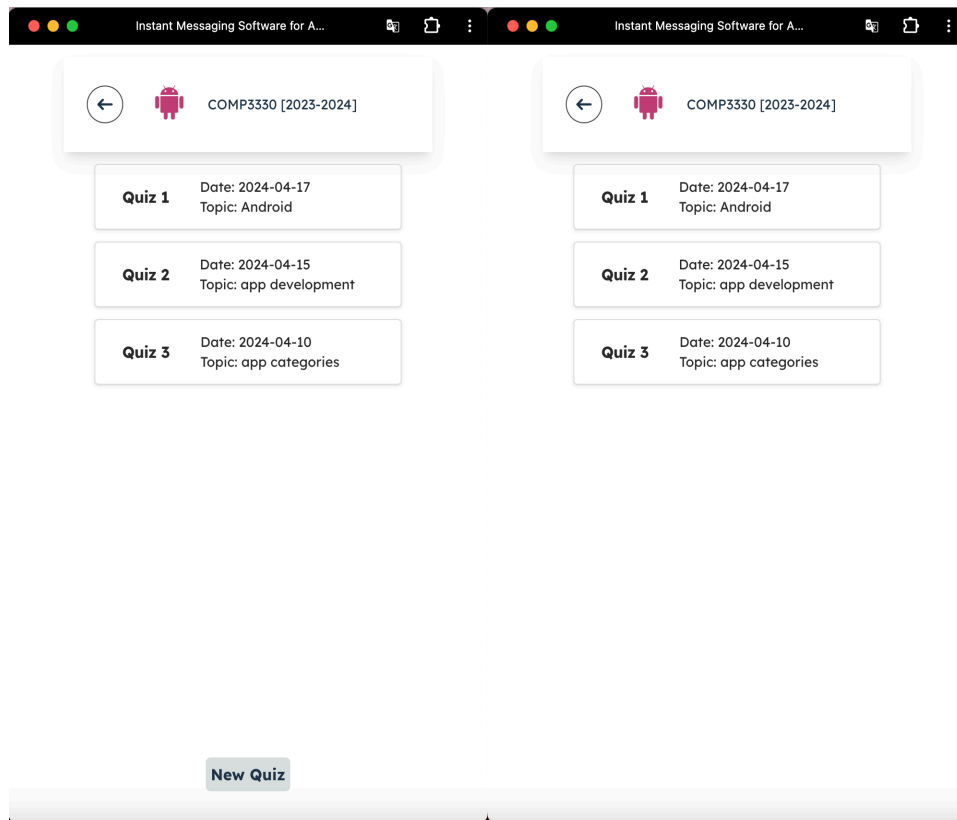


Figure 35: Quiz page layout for teachers (left) and students (right)

3.9 Quiz Creation

For teachers, they can create a new quiz by clicking on the “New Quiz” button at the bottom of the quiz page. After the designated button is clicked, the teacher will be directed to the quiz creation page. On the quiz creation page, teachers are prompted to perform several tasks to create a quiz, which includes specifying the quiz topics, formulating quiz questions, uploading figures to enrich quiz content, and defining answer choices (see Figure 36). As our focus is on encouraging student-teacher academic discussion through simple quizzes, and it does not make sense for the students to spend too much time answering quizzes during lessons, we only focus on multiple-choice questions in this project. For entering quiz questions, an automatic height adjustment text area is provided to ensure all text is clearly visible without the need for scrolling. To upload a quiz figure, teachers can simply click the “add” button which triggers a file selector that only accepts image files. If the figure is successfully uploaded to the system, the uploaded quiz figure will be shown on the page. To define multiple-choice options, teachers must provide a minimum of two choices, with the ability to add more by clicking the “add” button next to the “options” label. Upon filling in all

the information, teachers can set an expiration time, with the default being 10 minutes from the time of creation.

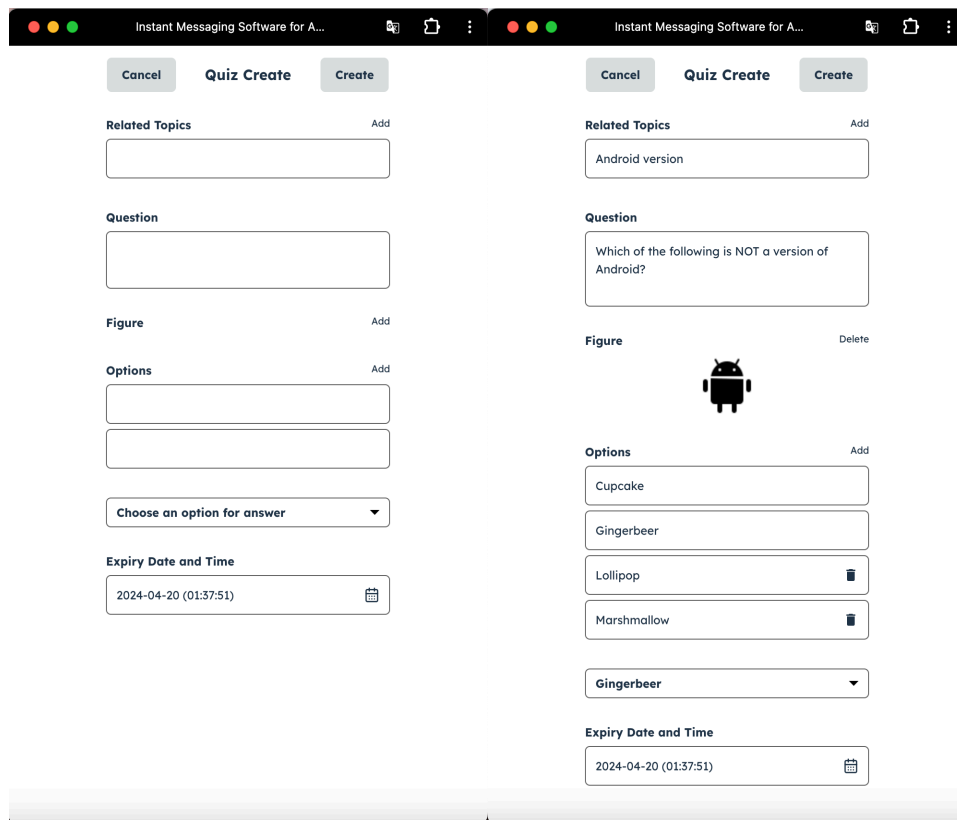


Figure 36: Example of quiz creation

The figure on the left shows the default layout of the quiz creation page. Teachers are prompted to provide the quiz-related topics, quiz questions, as well as at least two multiple-choice answer options. It is optional to upload a figure for quiz question content enrichment. The figure on the right shows a completed example of the necessary information for quiz creation. Teachers are allowed to delete the uploaded figure by clicking the “Delete” button adjacent to the “Figure” label. Moreover, teachers can also remove the extra answer options by clicking the removal button next to each additional choice.

3.10 Quiz Answering and Updating

On the quiz page, both students and teachers can choose a created quiz from the quiz list to proceed to the quiz question page.

Upon a desired quiz has been chosen, for students, they can then answer the quiz question.

On the quiz question page, the quiz question, the figure concerned, as well as the quiz answer options will be displayed to students. Furthermore, a countdown timer is displayed at the top,

indicating the remaining time available for students to submit their answers (see Figure 37). To respond to the quiz, students can select their answers from the list of available options and press the “Submit” button, which is located in the top right corner of the page. It is important to note that each student is allowed only one submission per quiz, once the submission is successful, a notification will be shown to students (see Figure 38). On the other hand, if the quiz has expired, students are unable to make any quiz submission and the model answer will be revealed to them (see Figure 39).

Instant Messaging Software for A...


Cancel Quiz 1 Submit

Remaining Time

00:00:11:55

Question

Which of the following is NOT a version of Android?



Response

Cupcake

Gingerbeer

Lollipop

Marshmallow

Figure 37: Example of quiz submission

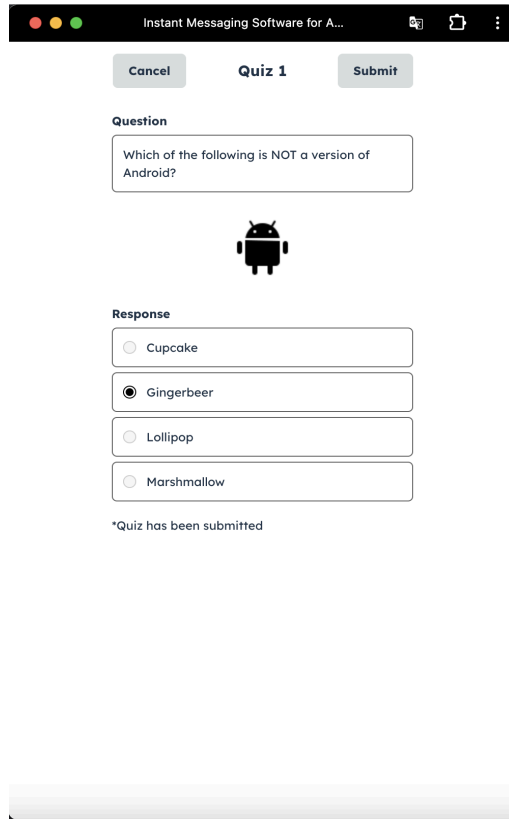


Figure 38: Notification after quiz submission

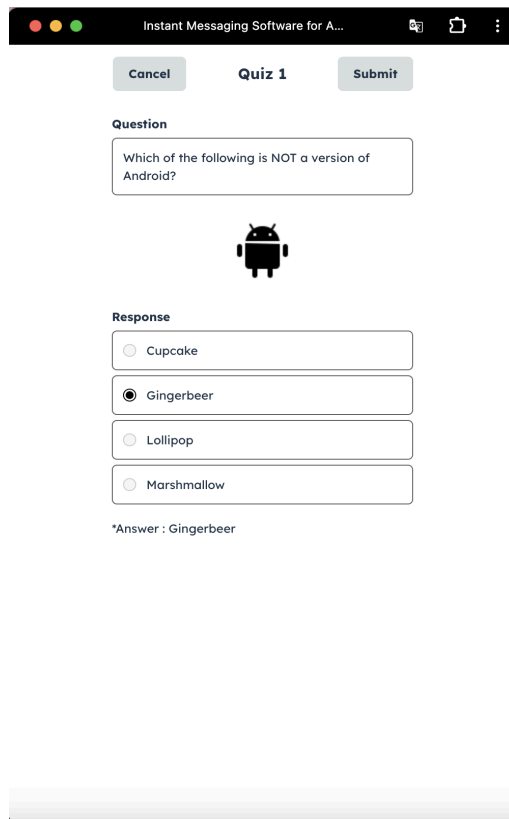


Figure 39: Quiz model answer reveal

For teachers, the quiz question page allows them to update existing quiz questions. The layout of this page for teachers is similar to the quiz creation page, except that the previous quiz question settings will be displayed to teachers and teachers can modify the quiz settings accordingly (see Figure 40).

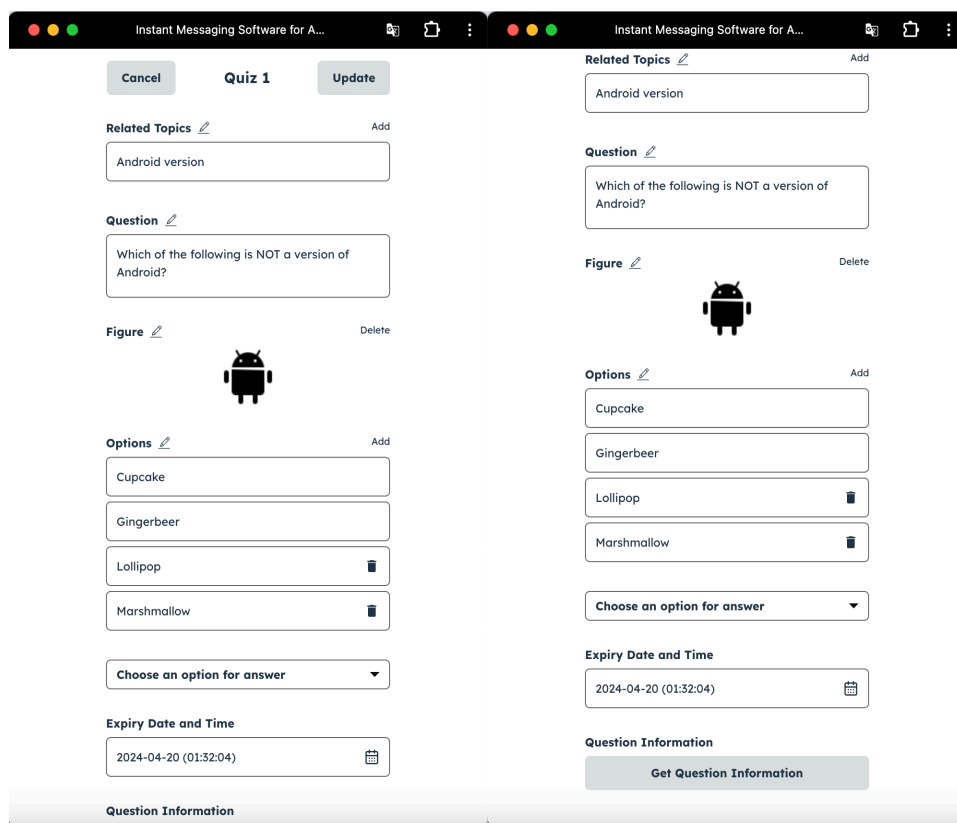


Figure 40: Existing quiz update example

3.11 Quiz Information

At the bottom of the quiz question page, a button labeled “Get Question Information” is provided for teachers to access information regarding quiz questions. This system aims to help teachers analyze and understand students' learning progress through some simple statistics. Upon clicking the designated button, the teacher will be directed to the quiz information page, where the basic information of the quiz question (including the quiz group concerned and quiz ID) as well as its statistics will be shown (see Figure 41).

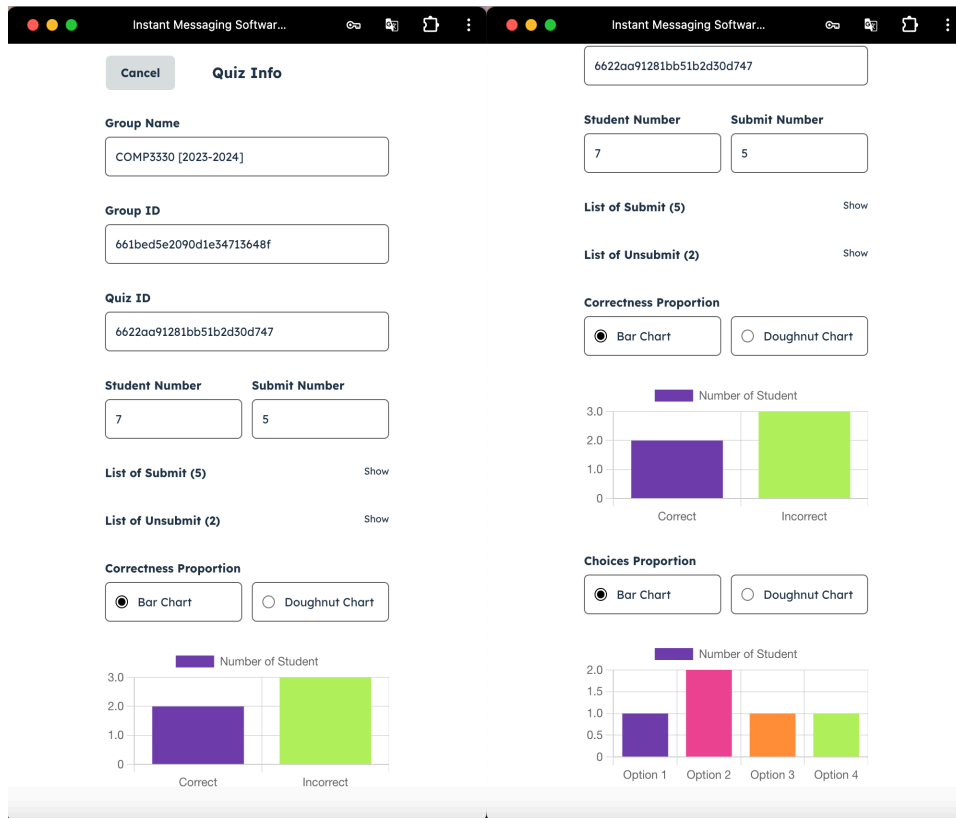



Figure 41: Layout for the quiz information page


On this page, the system will count the number of students in the quiz group and the number of submissions received. Additionally, teachers will have access to lists indicating which students have submitted the quiz and which students have not. Each quiz submission will be visibly marked with either a "tick" indicating a correct answer or a "cross" indicating an incorrect answer. Because the quiz group may contain a large number of students, it is impractical to display all the submission information of all students at once. To solve this problem, a button labeled either "Show" or "Hide" is positioned next to the "List of submit" and "List of unsubmit" labels to allow teachers to optionally choose whether to display or conceal the lists (see Figure 42).

Student Number	Submit Number
7	5


List of Submit (5) Hide




Cheng, Kai Hong
 ID: 66235c0b281bb51b2d30da5d ✓
 Date: 2024-04-20 (14:11:50)




Chan, Siu Ming
 ID: 6623601e281bb51b2d30e13e ✗
 Date: 2024-04-20 (14:27:10)



Cheung, Ka Ho
 ID: 6623619f281bb51b2d30e3bf ✓
 Date: 2024-04-20 (14:33:35)




Leung, Ho Hei
 ID: 66236483281bb51b2d30e4a9 ✗
 Date: 2024-04-20 (14:45:35)




Chan, Ho Chun
 ID: 66239fc5281bb51b2d30eb6c ✗
 Date: 2024-04-20 (18:58:24)

List of Unsubmit (2) Hide



Chan, Chin Hung
 ID: 661b73e6a36831f0cc8d098d
 Date: Undefined



Chan, Tai Man
 ID: 660b6d3edf1ae1c83956843a
 Date: Undefined

Figure 42: List of Submit and List of Unsubmit Example

Below these lists, the systems will generate two charts displaying the proportion of correctness and distribution of choices for the quiz. Teachers are allowed to choose between a bar chart or a doughnut chart to visualize the data (see Figure 43). Moreover, in order to enhance clarity in the data visualization, the system will dynamically generate a list of unique colors and assign them to each data represented in the charts (see Figure 44).

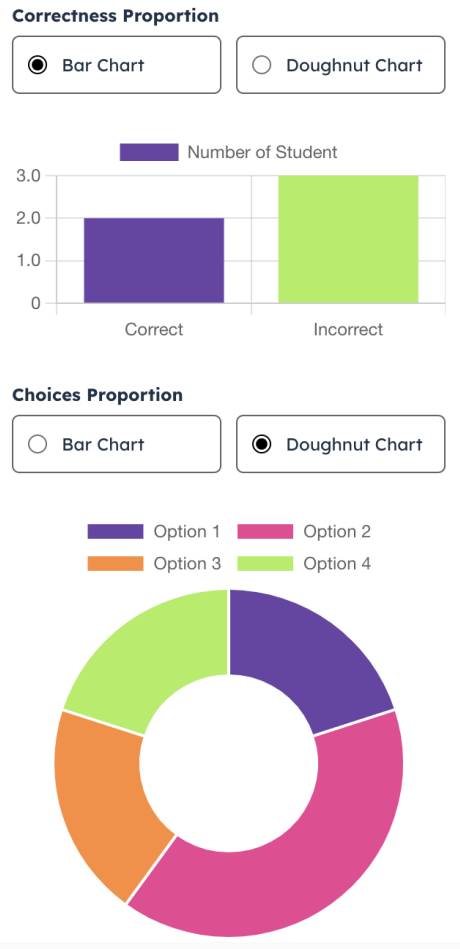


Figure 43: Charts for displaying the correctness proportion and choices distribution

```
const colorListGenerator = (numberOfColor, colorRangeLowerBound, colorRangeUpperBound) => {
  return Array(numberOfColor)
    .fill()
    .map((colorElement, idx) =>
      d3.interpolateWarm(colorRangeLowerBound + idx * (colorRangeUpperBound - colorRangeLowerBound) / (numberOfColor - 1))
    );
}
```

Figure 44: Algorithm for generating a list of unique colors

To generate a list of unique colors, this function accepts three parameters: the number of colors that are being generated, the lower bound of the color range, and the upper bound of the color range. By making use of the concept of linear interpolation, the color range will be divided into a number of sections. The starting point and the end point for each section will then be used as a value that represents a particular color.

3.12 Push Notification

The push notification feature has been implemented in this IM software. It is used to grab app user's attention and inform the app users that someone has posted new chat messages to a

particular chat group. Whenever this IM software loses focus or even online and someone posts new messages to a chat group, the notification message will be displayed to the user (see Figure 45 and Figure 46).

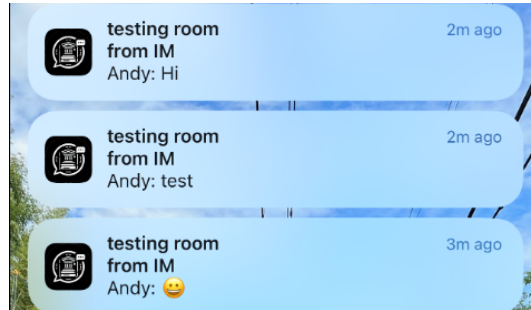


Figure 45: Notification message display on mobile phone

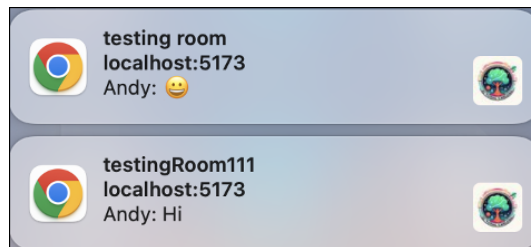


Figure 46: Notification message display on computer

4. Future Plan

While all intended features for this IM software have been completed, there are still areas for further development and enhancements.

First of all, in terms of data security, the data stored in the database should be encrypted. Unfortunately, due to lack of time, this feature has not yet been implemented in this project. Therefore, this function should be implemented in order to prevent and reduce the risk of information leakage.

Furthermore, the initial goal of this project was to create IM software in the form of PWA for facilitating academic communication between teachers and students. During its development, the focus was primarily on optimizing the software for mobile phone users. However, given that the software was built using React, users can access it through their computer's web browser. While the current features are tailored towards mobile devices, there is a recognition of the need to develop a version specifically for computer users, to better cater to the needs of both students and teachers.

Last but not least, regarding the quiz creation and answering functions of this software, there is room for further enhancement to make the system more comprehensive. For instance, additional functions could be implemented to allow teachers optionally randomize the order of quiz answers when creating a quiz. Moreover, the current system restricts question creation to multiple-choice questions, as it is designed for creating simple quizzes during lessons. To make the system more versatile and useful for a wider range of assessments, future development could involve adding other types of question formats into the quiz system.

5. Conclusion

This project aims to develop an instant messaging software specifically tailored for teachers and students at The University of Hong Kong (HKU) to enhance academic discussion during and after lessons. The IM software contains two key functionalities: a real-time chatting system and a quiz creation and response system. Users of the app will be able to engage in live academic discussions through the chatting system and utilize the quiz feature to create or respond to quizzes.

In order to accomplish this purpose, the IM software is developed in the form of progressive web app. This design allows users to access the system through their mobile phones or personal computer web browsers, and provides a user experience similar to a native app.

As of the time of writing this final report, all planned functionalities have been finished. Users at HKU can now take advantage of this IM software to enhance their academic discussions and learning experiences.

Reference

- [1] G. W. Larson, “Instant messaging” Encyclopædia Britannica, <https://www.britannica.com/topic/instant-messaging> (accessed Mar. 10, 2024).
- [2] L. Ceci, “Leading mobile apps worldwide in 2022, by downloads” Statista, <https://www.statista.com/statistics/1285960/top-downloaded-mobile-apps-worldwide/> (accessed Mar. 10, 2024).
- [3] L. Ceci, “Number of mobile phone messaging app users worldwide from 2019 to 2025” Statista, <https://www.statista.com/statistics/483255/number-of-mobile-messaging-users-worldwide/#:~:text=Global%20number%20of%20mobile%20messaging%20users%202019%2D2025&tex%20t=In%202021%2C%20an%20estimate%20of,popular%20mobile%20messenger%20app%20%20worldwide> (accessed Mar. 10, 2024).
- [4] U. Ghosh, “Role of communication in education” IJRTI, <https://www.ijrti.org/papers/IJRTI2206057.pdf> (accessed Mar. 10, 2024).
- [5] B. Bajarin, “In the new age of remote work, people under 30 might finally kill email” Fast Company, <https://www.fastcompany.com/90509588/in-the-new-age-of-remote-work-people-under-30-might-finally-kill-email> (accessed Mar. 10, 2024).
- [6] P. D. Hunt, “World emoji day 2021: How emoji can help create a more empathetic world, for all of US” Adobe Blog, <https://blog.adobe.com/en/publish/2021/07/15/global-emoji-trend-report-2021> (accessed Mar. 10, 2024).
- [7] I. Boutet, M. LeBlanc, J. A. Chamberland, and C. A. Collin, “Emojis influence emotional communication, social attributions, and information processing” ScienceDirect, <https://www.sciencedirect.com/science/article/abs/pii/S0747563221000443> (accessed Mar. 10, 2024).

- [8] “What is a Document database?” MongoDB,
<https://www.mongodb.com/document-databases> (accessed Mar. 20, 2024).
- [9] Narendra, “Cheatsheet To Document Oriented Database: Detailed breakdown,”
RedSwitches, <https://www.redswitches.com/blog/document-oriented-database/> (accessed
Mar. 20, 2024).
- [10] “Progressive web apps” MDN,
https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps (accessed Mar. 20,
2024).
- [11] F. Beaufort, P. LePage, T. Steiner, and A. Rodionov, “Add a web app manifest” web.dev,
[https://web.dev/articles/add-manifest#:~:text=The%20web%20app%20manifest%20is,user%
27s%20desktop%20or%20mobile%20device.](https://web.dev/articles/add-manifest#:~:text=The%20web%20app%20manifest%20is,user%27s%20desktop%20or%20mobile%20device.) (accessed Mar. 20, 2024).
- [12] “Service workers” web.dev,
[https://web.dev/learn/pwa/service-workers/#:~:text=Service%20workers%20are%20a%20fun
damental,a%20service%20worker%20is%20registered](https://web.dev/learn/pwa/service-workers/#:~:text=Service%20workers%20are%20a%20fundamental,a%20service%20worker%20is%20registered) (accessed Mar. 20, 2024).
- [13] L. Vu, “PWA Service Worker for Dummies” SimiCart Blog,
<https://www.simicart.com/blog/pwa-service-worker/> (accessed Mar. 20, 2024).
- [14] T. Sufiyan, “What is Node.js: A Comprehensive Guide” Simplilearn,
<https://www.simplilearn.com/tutorials/nodejs-tutorial/what-is-nodejs> (accessed Mar. 20,
2024).
- [15] “Introduction to JavaScript runtime environments” Codecademy,
<https://www.codecademy.com/article/introduction-to-javascript-runtime-environments>
(accessed Mar. 20, 2024).
- [16] O. Romanyuk, “Node.js is a great runtime environment - and here’s why you should use
it” freeCodeCamp, <https://www.freecodecamp.org/news/what-are-the-advantages-of-node-js/>
(accessed Mar. 20, 2024).

[17] E. Martin, “WebSockets vs server-sent events: Key differences and which to use in 2024” Ably Realtime, <https://ably.com/blog/websockets-vs-sse> (accessed Mar. 20, 2024).

[18] “What is database as a Service (DBaaS)?” Oracle, [https://www.oracle.com/database/what-is-a-cloud-database/dbaas/#:~:text=Database%20as%20a%20service%20\(DBaaS\)%20is%20a%20cloud%20database%20offering,and%20backing%20up%20the%20database](https://www.oracle.com/database/what-is-a-cloud-database/dbaas/#:~:text=Database%20as%20a%20service%20(DBaaS)%20is%20a%20cloud%20database%20offering,and%20backing%20up%20the%20database) (accessed Mar. 20, 2024).

[19] “Getting started with CDN - amazon Cloudfront” Amazon Web Services, <https://aws.amazon.com/cloudfront/getting-started/> (accessed Mar. 20, 2024).

[20] N. W. Foong, “Content delivery network: What you need to know” Medium, <https://levelup.gitconnected.com/content-delivery-network-what-you-need-to-know-c404c7330991> (accessed Mar. 20, 2024).