



Princess Sumaya University for Technology  
King Hussein School for Computing Sciences

# PSUT Chatroom

**Prepared By:**

Abdullah Al-Omari  
Ahmad Al-Sarraj  
Nader Barham

**Supervised By:**

Dr. Mustafa Al Fayoumi

Project Submitted in partial fulfillment for the degree of Bachelor of Science in  
Computer Science

First Semester-(2021-2022)

# Declaration of Originality

This document has been written entirely by the undersigned team members of the project. The source of every quoted text is clearly cited and there is no ambiguity in where the quoted text begins and ends. The source of any illustration, image or table that is not the work of the team members is also clearly cited. We are aware that using non-original text or material or paraphrasing or modifying it without proper citation is a violation of the university's regulations and is subject to legal actions.

**Names and Signatures of team members:**

**Abdullah Al-Omari**

**Ahmad Al-Sarraj**

**Nader Barham**

# Acknowledgments

We would like to thank Princess Sumaya University for Technology for providing all necessary facilities for this project and for their supervision.

Secondly, we'd like to thank Dr. Mustafa Al Fayoumi for his cooperation, guidance, and the support provided.

Finally, we'd like to thank our family and friends for their support and encouragement.

# Summary

Since the start of the pandemic the demand for an easy and private chat application is increasing since students need to communicate with professors without the hassle of manually creating group chats or sharing numbers.

In addition, the number of registering students' increases every year, so the need for an official chat room gradually increases with it. As a result, we intended to make an application that solves the issues of manually creating groups such as; creating groups with the professor without having all students in it, having subjects without groups because students either forget to create one or don't know how or they are lazy to go through the long process of creating and inviting all students in the mutual lectures.

PSUT Chatroom will give the students and professors a very easy experience to have each registered subject in a separate group with the professor and students auto-enrolled in it.

# List of Abbreviations

Abbreviation	Meaning
UML	Unified Modeling Language
ERD	Entity Relationship Diagram
UI	User Interface

# Table of Contents

## Contents

Chapter 1 Introduction .....	11
1.1 Overview .....	11
1.2 Problem Statement .....	11
1.3 Related Work .....	12
1.4 Document Outline .....	13
Chapter 2 Project Plan .....	14
2.1 Project Deliverables .....	14
2.2 Project Tasks .....	14
2.3 Roles and Responsibilities .....	15
2.4 Risk Assessment .....	16
2.5 Cost Estimation .....	17
2.6 Project Management Tools .....	17
Chapter 3 Requirements Specification .....	18
3.1 Stakeholders .....	18
3.2 Platform Requirements .....	18
3.3 Functional Requirements .....	20
3.4 Non-Functional Requirements .....	21
3.5 Other Requirements .....	21
Chapter 4 System Design .....	22
4.1 Logical Model Design .....	22
4.1.1 Use Case Diagrams .....	22
4.1.2 Class Diagrams .....	28
4.1.3 Object Diagram .....	29
4.1.4 Component Diagram .....	30
4.1.5 Activity Diagram .....	31
4.1.5.1 Main Activity Diagram .....	31
4.1.5.2 Settings Activity Diagram .....	32
4.1.6 Sequence Diagram .....	33
4.1.6.1 Login Sequence Diagram .....	33
4.1.6.2 Auto Enrollment Sequence Diagram .....	34

4.1.6.3 Send Message Sequence Diagram .....	35
4.1.6.4 Creating Subgroup Sequence Diagram .....	36
4.1.6.5 Add Student Sequence Diagram .....	37
4.1.7 State Transition Diagram .....	38
4.1.7.1 Student State Transition Diagram.....	38
4.1.7.2 Professor State Transition Diagram .....	39
4.1.7.3 Super Admin State Transition Diagram.....	40
4.1.7.4 Manage Chat State Transition Diagram .....	41
4.2 Physical Model Design .....	42
4.2.1 User Interface Design.....	42
4.2.2 Database Design.....	44
4.2.2.1 Database Entity Relationship Diagram .....	44
4.2.2.2 Database Schema.....	45
4.2.3 User Reports .....	46
Chapter 5 Implementation.....	47
5.1 Programming Languages.....	47
5.1.1 Front-End .....	47
5.1.2 Back-End.....	47
5.1.3 Database .....	47
5.2 Tools, API's and libraries .....	48
5.3 Main Algorithms and Coding Conventions .....	49
5.3.1 General User .....	49
5.3.1.1 Email Validation .....	49
5.3.1.2 Get Conversation .....	49
5.3.1.3 Create Message.....	50
Chapter 6 Testing .....	51
6.1 Testing Approach .....	51
6.2 Testing Tools .....	51
6.3 Tested Features.....	51
6.4 Discussion.....	52
Chapter 7 Conclusions and Future Work .....	53
Appendix A Users' Manual.....	54
Appendix B Document Changes.....	58

Appendix C Code Documentation.....	59
------------------------------------	----



# Table of Figures

Figure 1: Gantt Chart .....	15
Figure 2: General Functionalities Use Case Diagram .....	23
Figure 3: Class Diagram for PSUT Chatroom.....	28
Figure 4: Object Diagram for PSUT Chatroom .....	29
Figure 5: Component Diagram for PSUT Chatroom .....	30
Figure 6: Main Activity Diagram for PSUT Chatroom .....	31
Figure 7: Settings Activity Diagram for PSUT Chatroom .....	32
Figure 8: Login Sequence Diagram for PSUT Chatroom .....	33
Figure 9: Auto Enrollment Sequence Diagram for PSUT Chatroom.....	34
Figure 10: Sending Messages Sequence Diagram for PSUT Chatroom .....	35
Figure 11: Creating Subgroup Sequence Diagram for PSUT Chatroom .....	36
Figure 12: Add Student Sequence Diagram for PSUT Chatroom .....	37
Figure 13: Student State Transition Diagram for PSUT Chatroom .....	38
Figure 14: Professor State Transition Diagram for PSUT Chatroom .....	39
Figure 15: Super Admin State Transition Diagram for PSUT Chatroom .....	40
Figure 16: Manage Chat State Transition Diagram for PSUT Chatroom.....	41
Figure 17: Logo	Figure 18: User Interface for Login .....
Figure 19: User Interface for Main Menu.	Figure 20: User Interface for Chat. ....
Figure 21: Entity Relationship Diagram for PSUT Chatroom .....	44
Figure 22: Database Schema for PSUT Chatroom .....	45
Figure 23: Email Validation.....	49
Figure 24: Get Conversation .....	49
Figure 25: Create Message.....	50
Figure 26: Test Results .....	52
Figure 27: Test Details .....	52
Figure 28: Login Page.....	54
Figure 29: Main Menu .....	55
Figure 30: Chat.....	56
Figure 31: Voice Notes .....	57
Figure 32: Swagger Sample.....	59

# Table of Tables

Table 1: Document Outline.....	13
Table 2: Roles and Responsibilities .....	15
Table 3: Risk Assessment.....	16
Table 4: Mobile Requirements .....	18
Table 5: Server-Side Requirements .....	19
Table 6: Functional Requirements .....	20
Table 7: Non-Functional Requirements .....	21
Table 8: Tools, APIs and Libraries.....	48
Table 9: Document Changes.....	58

# Chapter 1

## Introduction

### 1.1 Overview

Due to the current circumstances that we live in (Covid-19), online teaching has become the main way of teaching, it has many advantages as well as some disadvantages.

One major disadvantage of online teaching is the lack of easy communication between tutors and students.

We aim to eliminate this disadvantage without the tutor having to share personal information that he/she might not be comfortable sharing, as well as making the whole learning experience better for the students by automatically enrolling them into each class they're taking.

### 1.2 Problem Statement

Since the pandemic happened, the communication between students and tutors has decreased significantly, students now only have two ways of communication with tutors, either via zoom meetings or emails.

The professor using zoom meetings will not give sufficient time for each student to ask their questions.

Emails can be sent any time but may take a while till the professor opens, reads and replies to all his emails, some urgent emails can be left unseen.

PSUTCHAT eliminates the lack of synchronization between sender and receiver, students will be encouraged to ask without writing a formal email and professors can reply quickly if the question is urgent, more students asking will lead to better understanding of the subject, and the professor's email inbox will be more focused on work related topics.

## 1.3 Related Work

There are many systems that are related to our work such as and not limited to;

-**WhatsApp:** The most popular chatting application now a days which is owned by Facebook, it offers free messaging and voiceover IP that connects users in group chats and gives them the functionality to share messages, photos, videos and even files while maintaining end-to-end encryption. You can also name your group, mute or customize notifications. This is similar to our application however WhatsApp uses phone numbers to connect users, and the group chats have to be created manually, unlike in our application that automatically creates and enrolls students in a chatroom for each of his/her enrolled courses with their professor, moreover our application uses official university Gmail accounts instead of numbers which help protect everyone's privacy.

- **Microsoft Teams:** It is a chat application that focuses on meetings and document sharing. You can create group chats, online meetings, audio and video calls, and web-conferencing for up to 10,000 people. And many more features such as; scheduling assistance, note-taking, screen sharing, recording meetings and instant messaging, This is also similar of what we need in our application however users who participate in conversations that are part of the chat list in Teams must have an Exchange Online (cloud-based) mailbox for an admin to search chat conversations this restricts the capability of searching students chat with the professors because content from these conversations isn't searchable and can't be placed on hold because students don't have cloud-based mailboxes. In PSUT Chatroom we just need 1 requirement for using chatrooms which is having a Gmail account that is already created once you entered PSUT University, this intern reduces the hassle of creating anything new or unofficial to use PSUT Chatroom. In addition, students/professors don't have to create groups manually since PSUT Chatroom automatically creates and enrolls students in a chatroom for each of his/her enrolled courses with their professor. Moreover, Microsoft Teams is Free for up to 300 users after which you need Premium plans that start at \$5 per month. Whereas PSUT Chatroom is free of charge.

-**Telegram:** It is similar to WhatsApp but focuses on security and speed, it adds the functionality of setting up bots for specific tasks, adding to that it helps in sharing large videos and documents of any type and private messages are encrypted and can self-destruct. However, it also lacks the functionality of auto enrollment.

- **WeChat:** It is also a famous messaging and social media application from Tencent that offers group chats and group video calls but has a strict size of sending files including PDFs, documents, spreadsheets, slideshows and more without the use of your email or file sharing

applications, adding to that it also offers WeChat pay and wallet for transactions. However, it also lacks the functionality of auto enrollment.

- **Hangouts:** It is a group chat application owned by Google for group chats and free group video call with up to 10 friends, you can also send status messages, photos, emoji, stickers and animated GIFs and share your current location with the map integration. Adding to that you can connect your Google Voice account for phone number, SMS, and voicemail integration. However, it also lacks the functionality of auto enrollment.

- **Google Chat:** A newly developed fresh application from Google that is still in early access which has everything from sharing chat, files and tasks with teams and businesses to the integration of seeing all your messages in one place and collaborating in dedicated chat room that can be easily created to share chat including threaded conversations plus shared files and tasks. It supports external users, 28 languages and 8,000 members per room. However, it's missing the capability to auto enroll students to their rooms and professors should manually add each student to his/her room.

## 1.4 Document Outline

The structure of our project is as follows:

Chapter Title	Description
<b>Chapter 1: Introduction</b>	Provides a brief overview of our idea and the problems our project solves, related work and targeted audience.
<b>Chapter 2: Project Plan</b>	Detailed description of the project's flow, task and roles distribution among the team, and cost assessment.
<b>Chapter 3: Requirements Specifications</b>	The functional and non-functional requirements and required software and hardware specifications with the declaration of stakeholders.
<b>Chapter 4: System Design</b>	Detailed analysis for the system components for both the high- and low-level design using UML diagrams and such.

*Table 1: Document Outline*

## Chapter 2

# Project Plan

### 2.1 Project Deliverables

Our research provides the following deliverables:

- Surveys: data will be collected from students and professors.
- Main Database: stores chat messages (text or files) and privilege assigning.
- React native framework that does the required tasks.
- Graphical user interface
- A mobile application available for both Android and IOS.

### 2.2 Project Tasks

Project tasks include:

- Understanding the main problem we're trying to solve by creating surveys and analyzing feedback to set the requirements.
- Requirement identification and analysis
- Setting privilege levels for basic users, professors, and admins.
- Developing a chat system using react native framework.
- Designing a user-friendly graphical interface.
- Creating a mobile application that includes all the above.

## 2.3 Roles and Responsibilities

Task Number	Team Member	Task/Responsibility
1	Abdullah, Ahmad	Creating surveys and analyzing feedback.
2	Nader, Abdullah	Requirement identification and analysis.
3	Nader	Setting privileges.
4	All of us	Developing a chat system.
5	Ahmad	Designing user-friendly graphical interface.
6	All of us	Creating a mobile app.

Table 2: Roles and Responsibilities



Figure 1: Gantt Chart

## 2.4 Risk Assessment

Risk ID	Task ID	Risk Description	Probability	Impact	Response
R1	T1	Low percentage of students fill the survey	High	Some requirements may be undefined	Sending the survey to group chats and professors
R2	T3	Wrong privileges identification	Low	Students getting high privileges or tutors' privileges aren't enough to moderate the chat room	Simulate a chat room and identify each user's needs
R3	T4	The application development takes more time than expected because of lack of experience	Medium	Late delivery	#
R4	T5	The design doesn't satisfy users' needs	Medium	Hard to use	Ask for feedback and redesign the interface

Table 3: Risk Assessment



## 2.5 Cost Estimation

Estimated starting cost is 134\$

This includes;

- Google Development Account which is a one-time 25\$ fee.
- IOS Development Account which is an annual 99\$ fee
- Lucidchart Account which is a monthly 10\$ fee.

## 2.6 Project Management Tools

- **Microsoft Word:** A documentation tool used to make, edit and format our project documentation.
- **LucidChart:** A website we used to design and edit most of our UML diagrams.
- **Draw.io:** An online diagram software we used to make the Gantt chart.
- **Adobe Photoshop:** Image editing and designing software we used to make our user interfaces and logo.
- **GoodNotes:** An application on iPad we used to draw the Object diagram.
- **Google drive:** A cloud storage service we used to share files with each other.

## Chapter 3

# Requirements Specification

### 3.1 Stakeholders

Users:

- Students: Uses PSUT Chat rooms application to ask a question, check what the professors uploaded.
- Professors: Uses PSUT Chat rooms to upload assignments, projects and answer students' questions.

Moderators: Uses PSUT Chat rooms to check for technical errors, assure that the application is only used for educational issues and create or delete chat rooms if needed.

### 3.2 Platform Requirements

Mobile users:

Requirement	Description	Priority
PRM1	Access to the Internet	Essential
PRM2	Access to Google Services	Essential
PRM3	Android OS/iOS	Essential
PRM4	Camera/Mic	Recommended

*Table 4: Mobile Requirements*

Servers:

Requirement	Description	Priority
PRS1	Access to the Internet	Essential
PRS2	Storage Units/Database	Essential
PRS3	High-end Components	Recommended
PRS4	Host Software	Essential
PRS5	Back-end services	Essential

*Table 5: Server-Side Requirements*

### 3.3 Functional Requirements

ID	Name	Constraint	Description
FR1	Login	Credentials must exist in the database.	The user will enter their email address and password to access the application.
FR2	Forgot Password	New password must be strong and must not be the same as the old password.	The user will be able to change his password from the application.
FR3	Send Messages	Display an error if the message was not uploaded properly.	The user will be able to send text, files, voice messages and images to the chat room.
FR4	Create Subgroup	New group must have less students than the original group.	The professor will be able to create a subgroup of the original group if he wishes to do so.
FR5	Remove Student	None.	A person working in the registration department will be able to remove students from a chat room.
FR6	Delete Chat Room	None.	A person working in the registration department will be able to delete a certain chat room.
FR7	Generate Statistics	None.	A person working in the registration department will be able to see statistics for which chat room is the most active and compare it to other sections or courses.
FR8	Automatic Course Enrollment	Registration must provide the application with sufficient data such as semester sections and the students enrolled in them.	The user will automatically be enrolled in a chat room for each of his enrolled courses.
FR9	Notification System	None.	Students will be notified about upcoming events such as; exams, homework, etc....
FR10	Calendar	None.	Students will have a calendar to check for future events.
FR11	Create Chat Room	New chat room must not exist already.	A person working in the registration department will be able to manually create a chat room.
FR12	Add Student	None.	A person working in the registration department will be able to add students to a chat room.

Table 6: Functional Requirements

### 3.4 Non-Functional Requirements

ID	Requirement	Description
NFR1	Privacy and Security	Messages shared between users should be encrypted to maintain privacy. Users have the right to control their personal information and how it's used. Protection against the unauthorized access of data or malicious attacks and exploitation of data. Personal information is protected.
NFR2	Robustness	To be able to deal with errors, in case user's device crashes, a backup of their chat history must be stored on remote database servers to enable
NFR3	Performance	Application performance must be lightweight, accomplish tasks in a short time or fast and must send messages instantly.
NFR4	Usability	User interface must be easy to use and understand, even for new users or for users with limited technological experience.
NFR5	Reliability	Application must be able to handle errors and user requests on time, and be trustworthy for users.
NFR6	Portability	Application must be able to run in different mobile operating systems.

*Table 7: Non-Functional Requirements*

### 3.5 Other Requirements

Other requirements are going to be based on these contributors' rules:

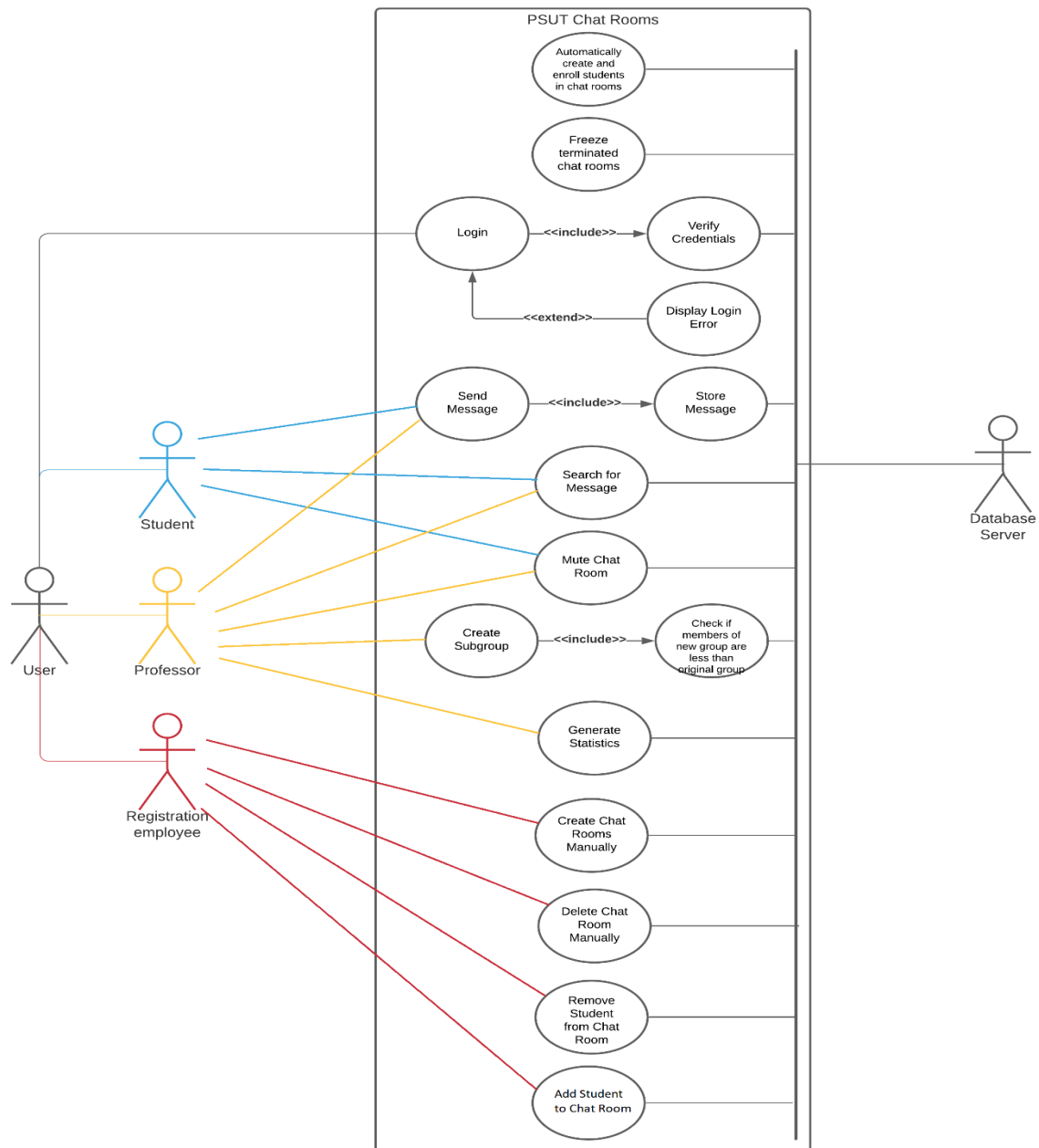
- Princess Sumaya University for Technology (Registration Department).
- Google.
- Royal Scientific Society.

# Chapter 4

## System Design

### 4.1 Logical Model Design

#### 4.1.1 Use Case Diagrams



*Figure 2: General Functionalities Use Case Diagram*

*Figure represents the general functionalities in the system, including all user types:*

- Logging in into the system which includes credential verification.
- Automatic creation and freezing for chat rooms at the beginning and end of each semester.
- Sending messages.
- Searching for a specific message in the chat.
- Muting the notifications of a chat room.
- Creating a subgroup of the original group.
- Generating statistics about the number of messages sent in a chat room.
- Manually creating or deleting chat rooms.
- Manually adding or removing students to or from chat rooms.

Below are expanded use cases derived from the above figure:

#### Expanded Use Case 1:

<b>Use Case</b>	Login
<b>Actors</b>	Student, Professor, Registration Employee
<b>Purpose</b>	To allow users to access the system.
<b>Success Scenario</b>	The user would be able to access the system successfully.
<b>Description</b>	The user would enter their credentials, the database would verify them, and upon success the user would access the system.
<b>Type</b>	Primary
<b>Reference</b>	FR1

#### Typical course of actions:

Actor Action	System Response
1. The user enters existing credentials.	2. The system verifies credentials through the database.
3. the user will be granted with the main menu and be able to use the application.	

#### Alternative:

Line 1: The user enters non-existing credentials.
Line 2: The system would display an error message which states that the entered credentials are incorrect.
Line 3: The user would not be granted access to the system.



## Expanded Use Case 2:

<b>Use Case</b>	Create Subgroup
<b>Actors</b>	Professor
<b>Purpose</b>	To give the professor more freedom to be able to subdivide his section into smaller group/groups for any specific reason that involves that smaller group if he/she wishes to.
<b>Success Scenario</b>	The professor creates a subgroup for one of his sections.
<b>Description</b>	The professor can create a smaller group of one of his sections, and be able to message them separately.
<b>Type</b>	Secondary
<b>Reference</b>	FR4

## Typical course of actions:

Actor Action	System Response
1. The professor attempts to create a subgroup.	2. The system would check if the number of members in the subgroup are less than the number of members of the original group.
	3. The system would create the subgroup and enroll the chosen students and their professor in it.

## Alternative:

Line 1: The professor would attempt to create a subgroup with the same members as the original group.
Line 2: The system would check if the members are less than in the original group.
Line 3: The system would show a message stating the issue and that the group was not created.

### Expanded Use Case 3:

<b>Use Case</b>	Create or Delete chat room
<b>Actors</b>	Registration Employee
<b>Purpose</b>	To be able to create or delete a chat room after the semester has started.
<b>Success Scenario</b>	The registration employee creates or deletes a chat room.
<b>Description</b>	The registration employee would be able to create or delete a chat room.
<b>Type</b>	Primary
<b>Reference</b>	FR11 and FR6

### Typical course of actions:

Actor Action	System Response
1. The registration employee attempts to create or delete the chat room.	2. The system creates or deletes the chat room.

### Expanded Use Case 4:

<b>Use Case</b>	Automatically enroll students in chat rooms for each of their sections
<b>Actors</b>	Database/Server
<b>Purpose</b>	To automatically generate chat rooms corresponding to the data provided by the university (sections, students and professors).
<b>Success Scenario</b>	A chat room for each section with students and professors in them would be created.
<b>Description</b>	The system creates a chat room for each course and section with students and professors enrolled in them.
<b>Type</b>	Primary
<b>Reference</b>	FR8

### Typical course of actions:

Actor Action	System Response
	1. The system creates chat rooms based on the data provided by the university.

### Expanded Use Case 5:

<b>Use Case</b>	Add or Remove student from a chat room
<b>Actors</b>	Registration Employee
<b>Purpose</b>	To allow the registration department to manually add or remove a student from chat rooms if they are no longer enrolled in that section.
<b>Success Scenario</b>	A student would be added or removed from a chat room.
<b>Description</b>	The registration employee would be able to add or remove students from chat rooms.
<b>Type</b>	Primary
<b>Reference</b>	FR12 and FR5

### Typical course of actions:

Actor Action	System Response
1. The registration employee adds or removes a student from a chat room.	2. The system enrolls or unenrolls the student from that chat room.

## 4.1.2 Class Diagrams

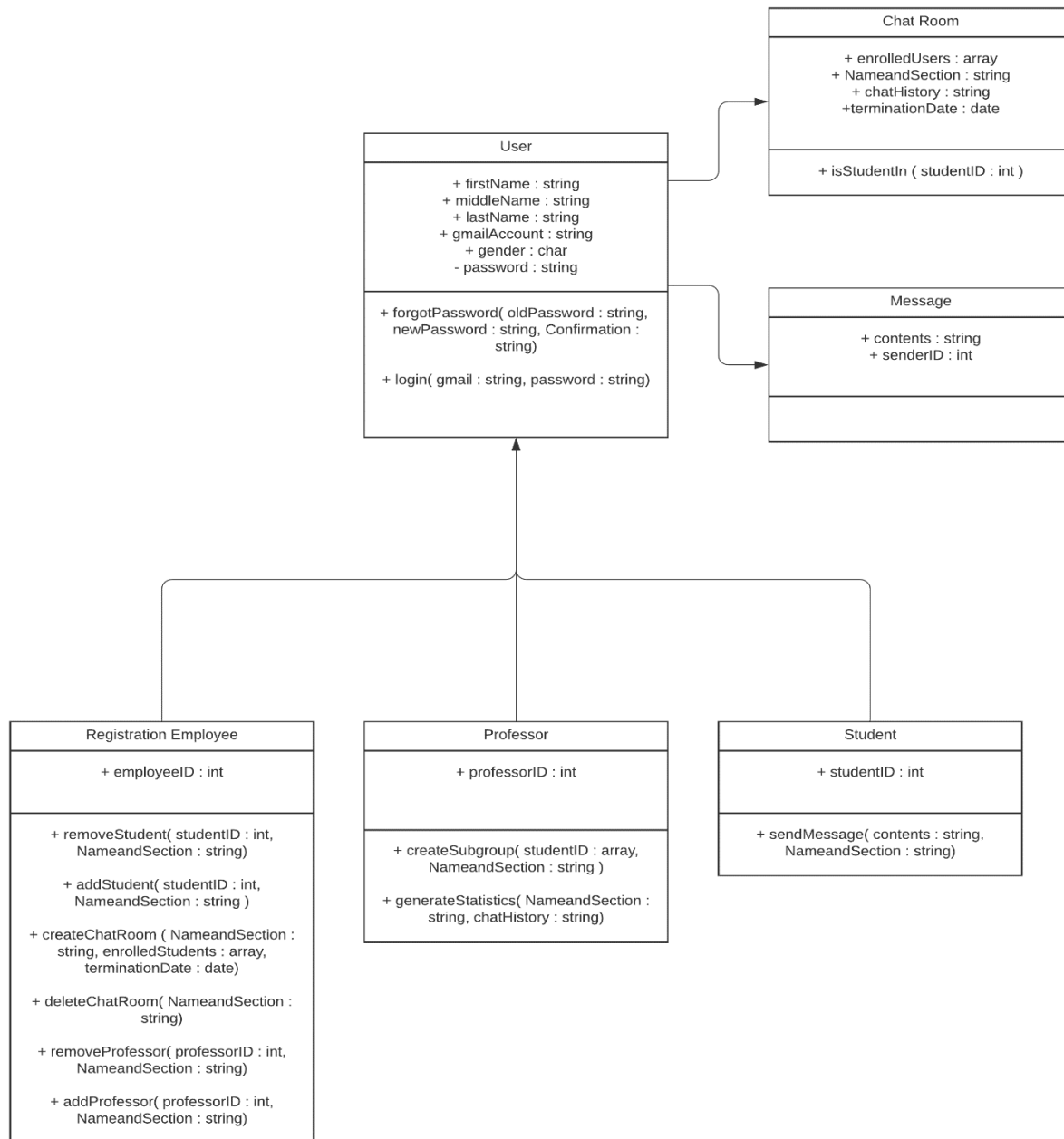


Figure 3: Class Diagram for PSUT Chatroom

Figure models the hierarchy and structure of the classes found in PSUT Chatrooms, and shows their attributes and methods.

PSUT Chatrooms has 3 types of users, each has a different set of privileges, which are:

- Student.
- Professors.

- Registration Employee.

### 4.1.3 Object Diagram

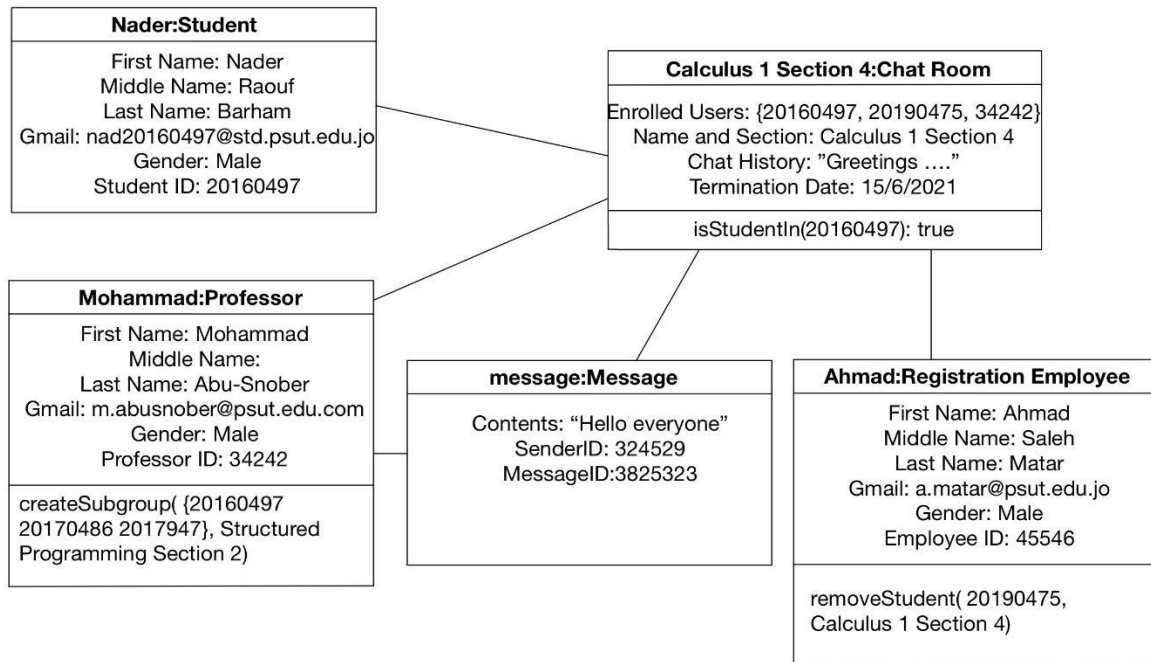


Figure 4: Object Diagram for PSUT Chatroom

Figure shows an instance of our class diagram, and how classes interact with each other.

#### 4.1.4 Component Diagram

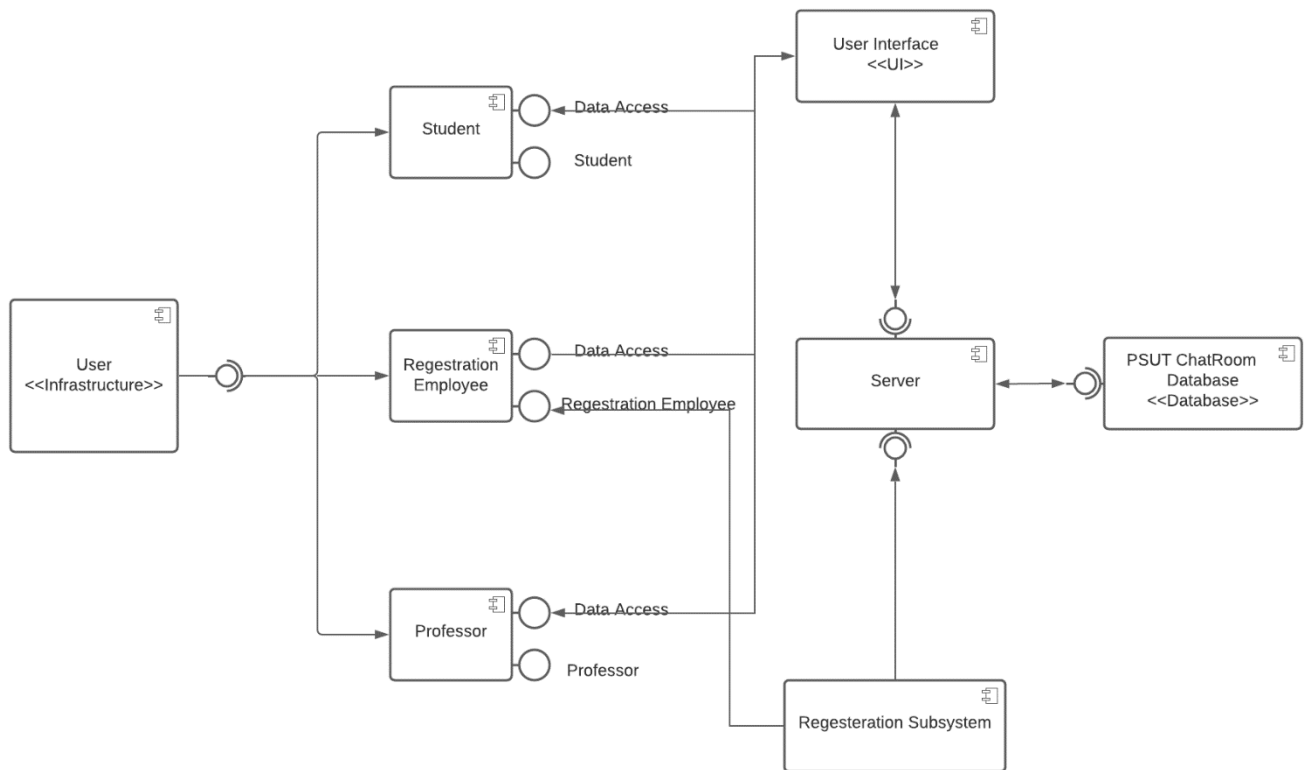


Figure 5: Component Diagram for PSUT Chatroom

## 4.1.5 Activity Diagram

### 4.1.5.1 Main Activity Diagram

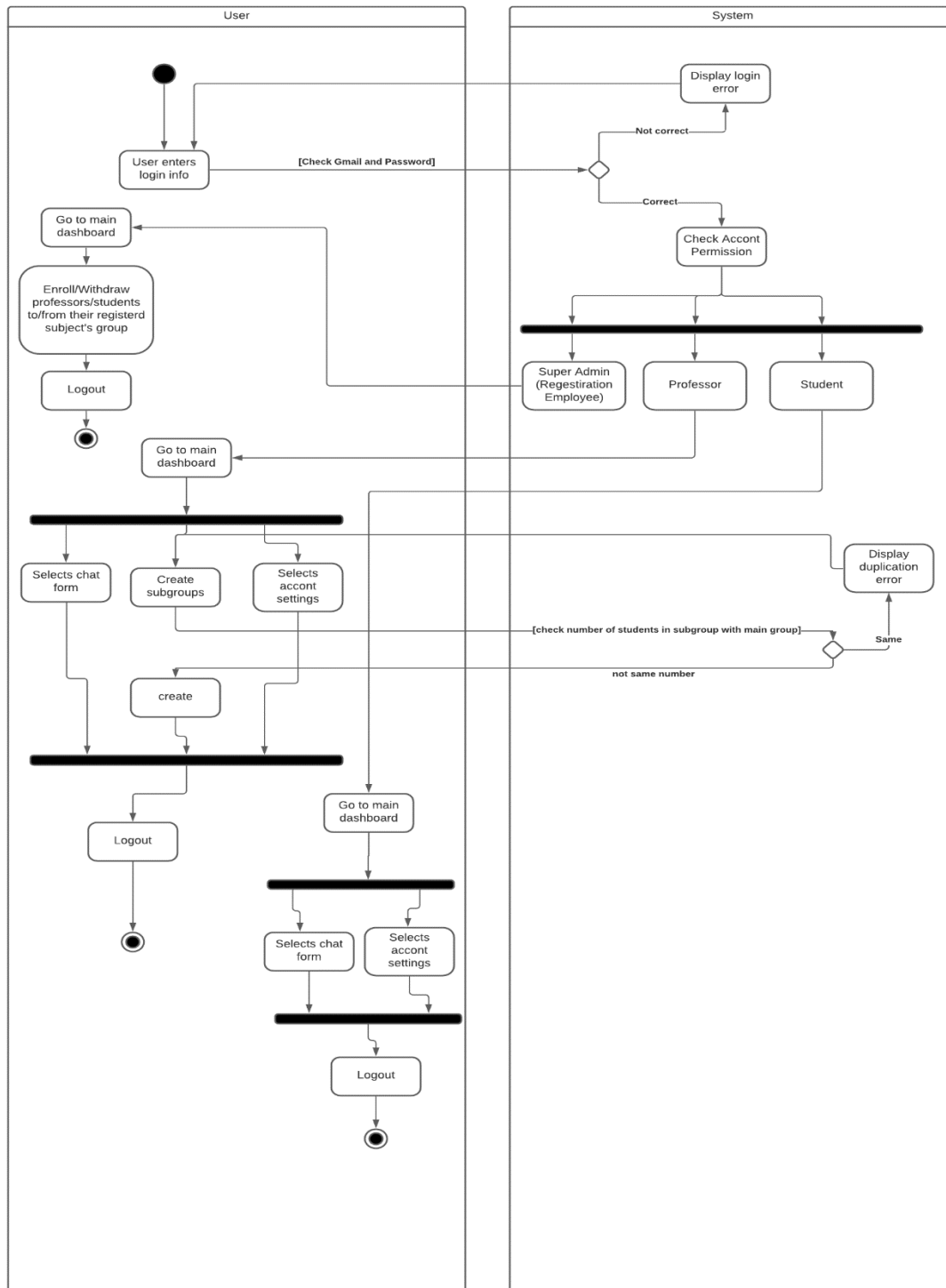


Figure 6: Main Activity Diagram for PSUT Chatroom

Main Activity Diagram: Explains the user and its integration with the system to check account permission and display login error if found and then how each user based on permission can access their settings and chat if available.

#### 4.1.5.2 Settings Activity Diagram

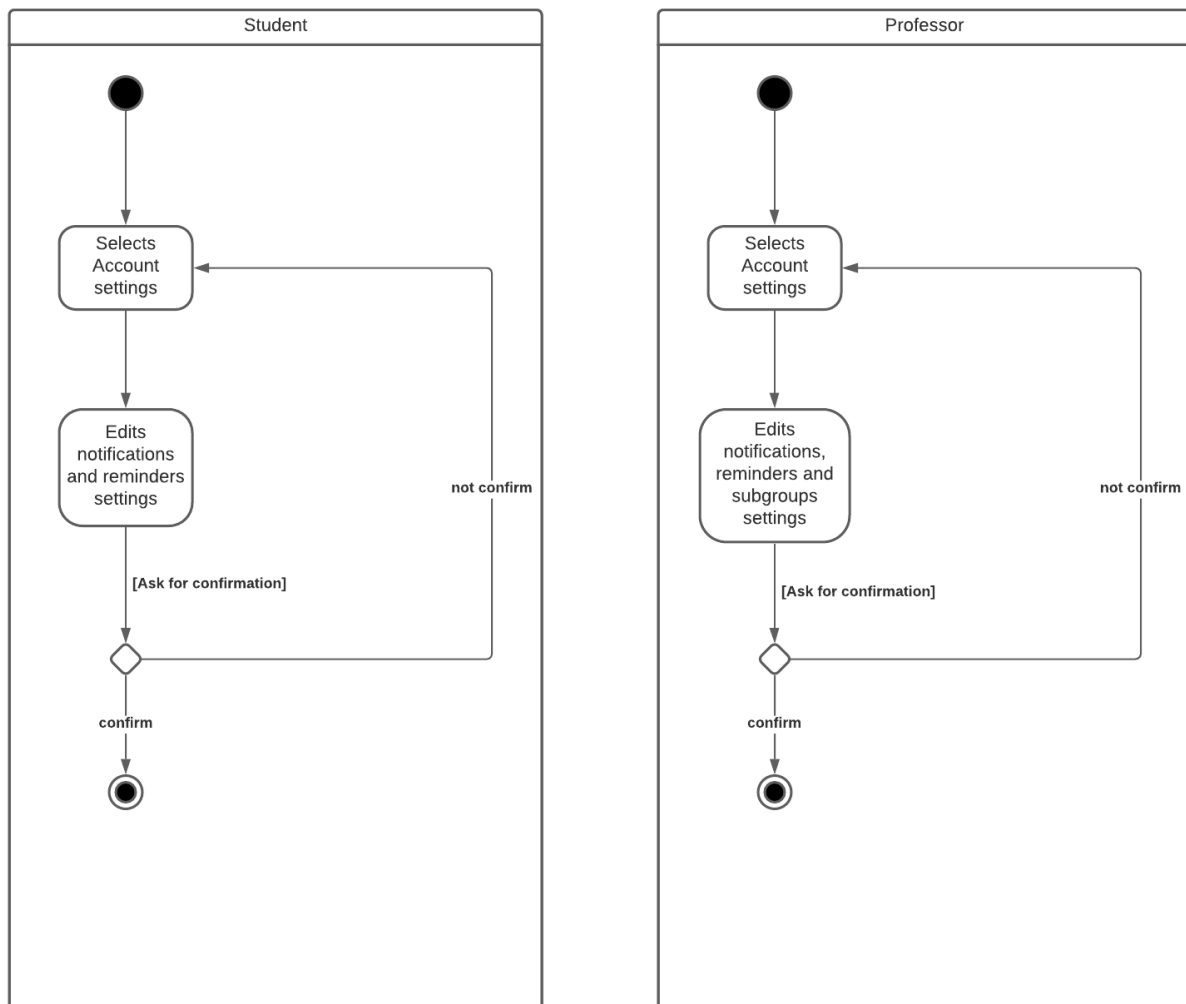


Figure 7: Settings Activity Diagram for PSUT Chatroom

Settings Diagram: Shows how different the settings are from Student's point of view and Professor's point of view since Professor can create subgroups and student can't.



## 4.1.6 Sequence Diagram

### 4.1.6.1 Login Sequence Diagram

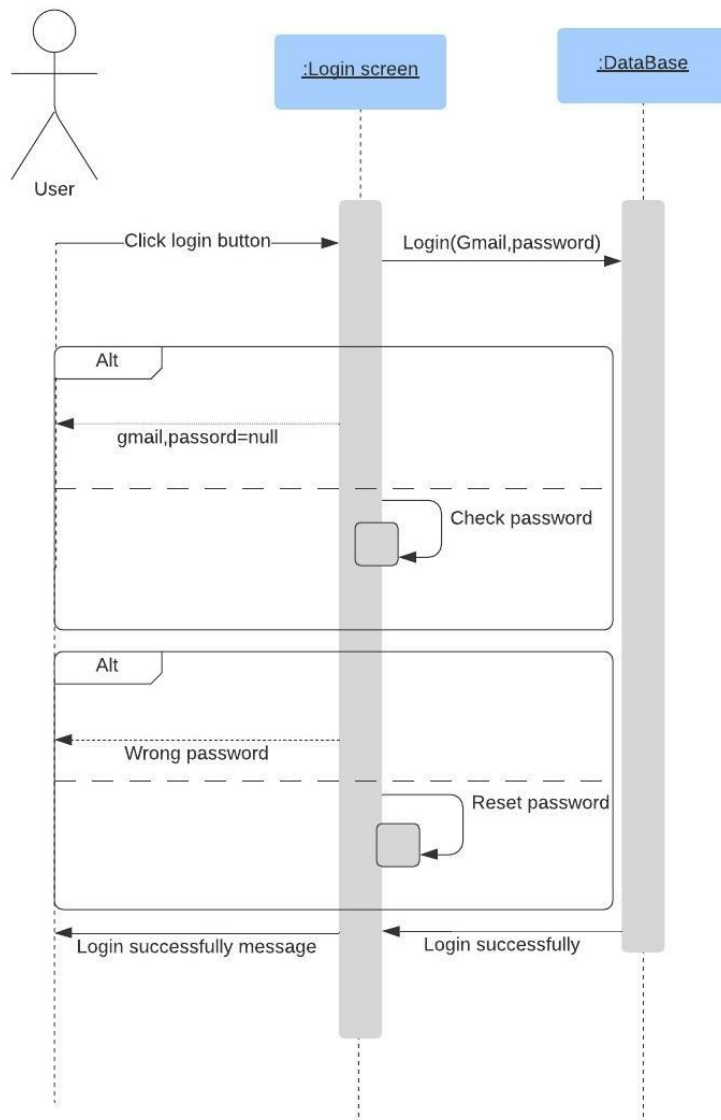


Figure 8: Login Sequence Diagram for PSUT Chatroom

Figure represents The User visits Login page enter his Gmail, password waiting for the system to validate its credentials by requesting database server to find matching user Gmail and password, if they were correct then a message with content Login successfully will appear to the user Otherwise they have to reenter their data or reset their passwords in case they forget them.

#### 4.1.6.2 Auto Enrollment Sequence Diagram

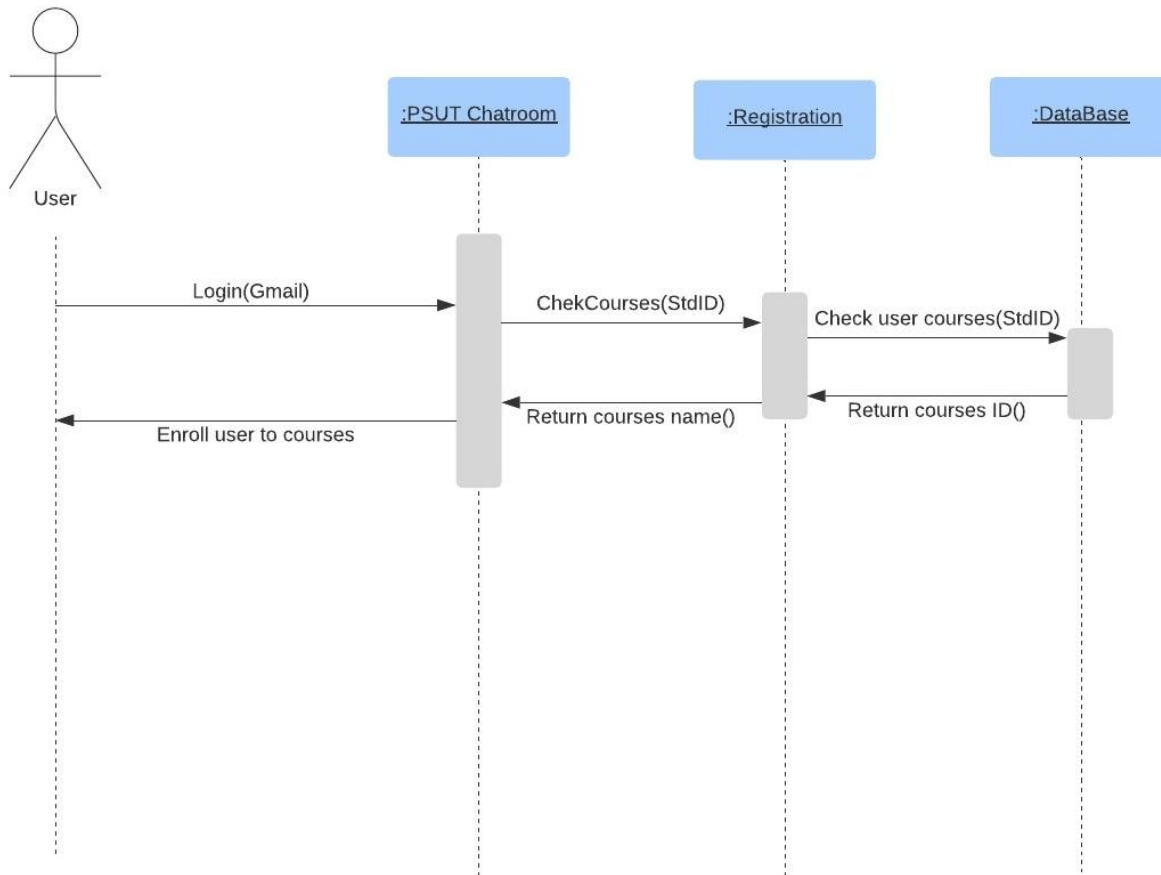


Figure 9: Auto Enrollment Sequence Diagram for PSUT Chatroom

The user login to the application then the application checks the student current enrolled courses from the registration database, the database returns the courses ID then the application enrolls the student to the enrolled courses chat rooms.

#### 4.1.6.3 Send Message Sequence Diagram

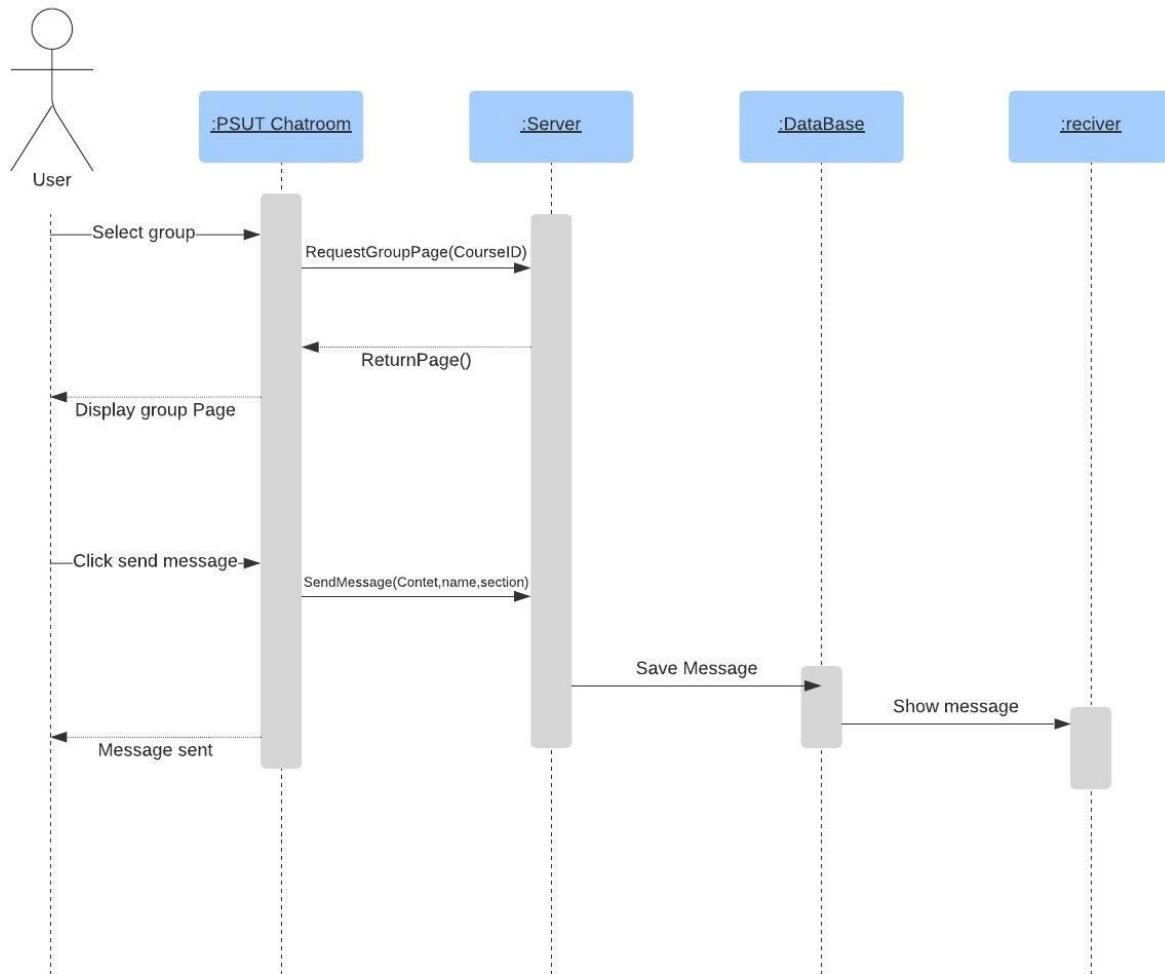


Figure 10: Sending Messages Sequence Diagram for PSUT Chatroom

The user selects a group and the server returns the group page, the user then clicks send message, the application sends message content, name of sender and section, then the message is stored in the database, the receiver sees the message and a confirmation is sent to the sender.

#### 4.1.6.4 Creating Subgroup Sequence Diagram

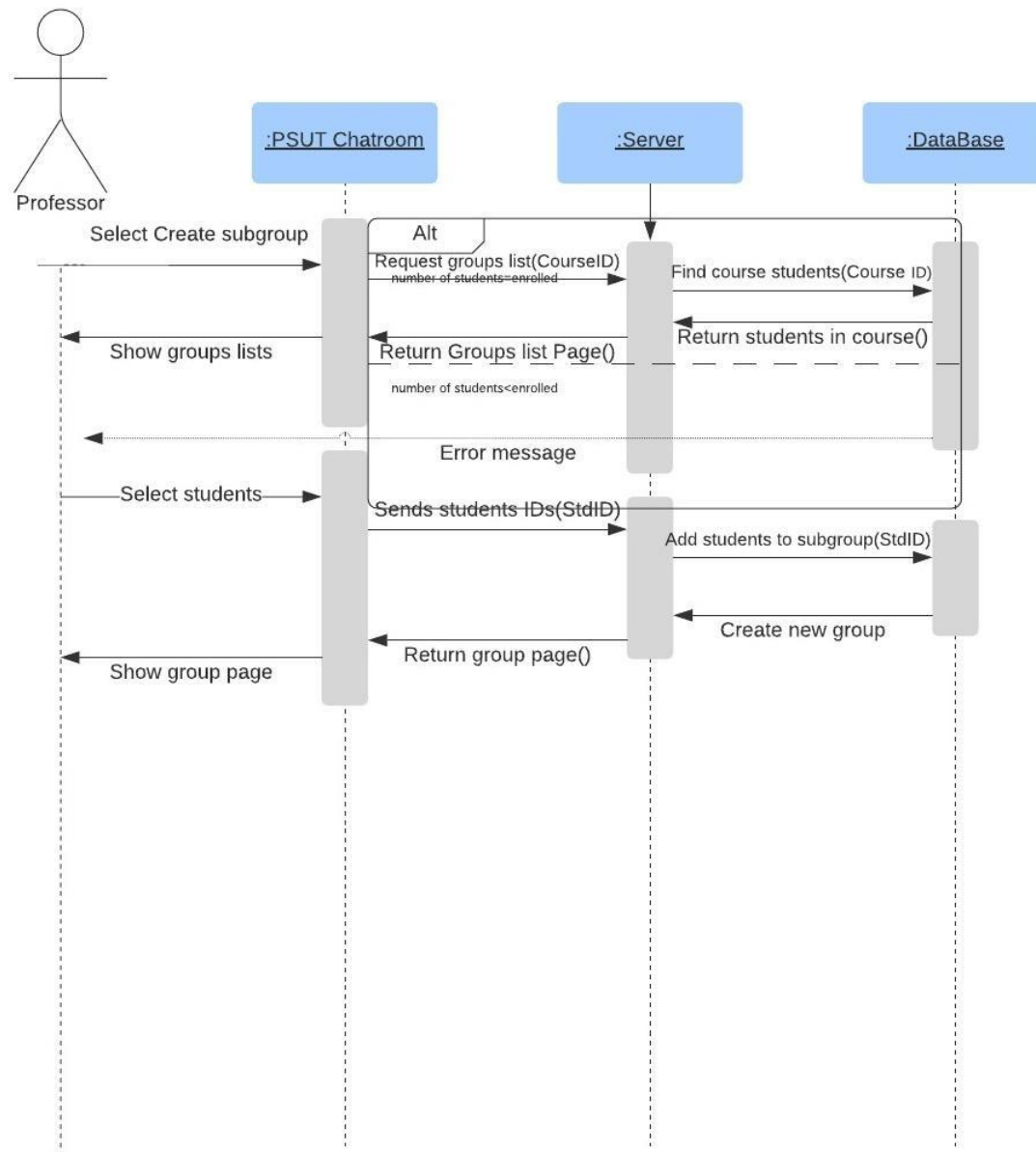


Figure 11: Creating Subgroup Sequence Diagram for PSUT Chatroom

The professor selects a group, the server returns current enrolled students, then the professor selects specific students, the application sends the selected students ID, the database enrolls the selected students into the created sub group and returns the group page to the users.

#### 4.1.6.5 Add Student Sequence Diagram

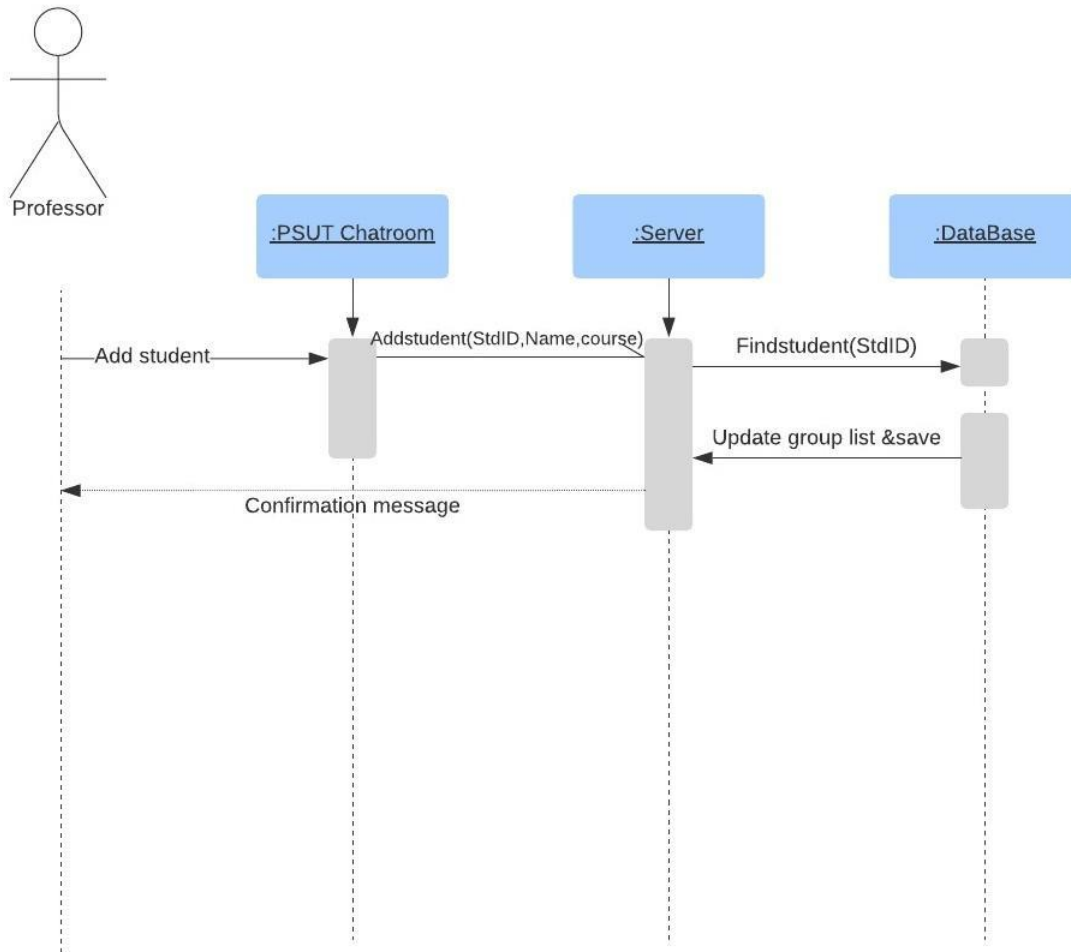


Figure 12: Add Student Sequence Diagram for PSUT Chatroom

The admin clicks on add student, insert student ID, name and course ID, the server search for the user in the database and add the student in the course and a confirmation message is sent to the admin.

## 4.1.7 State Transition Diagram

### 4.1.7.1 Student State Transition Diagram

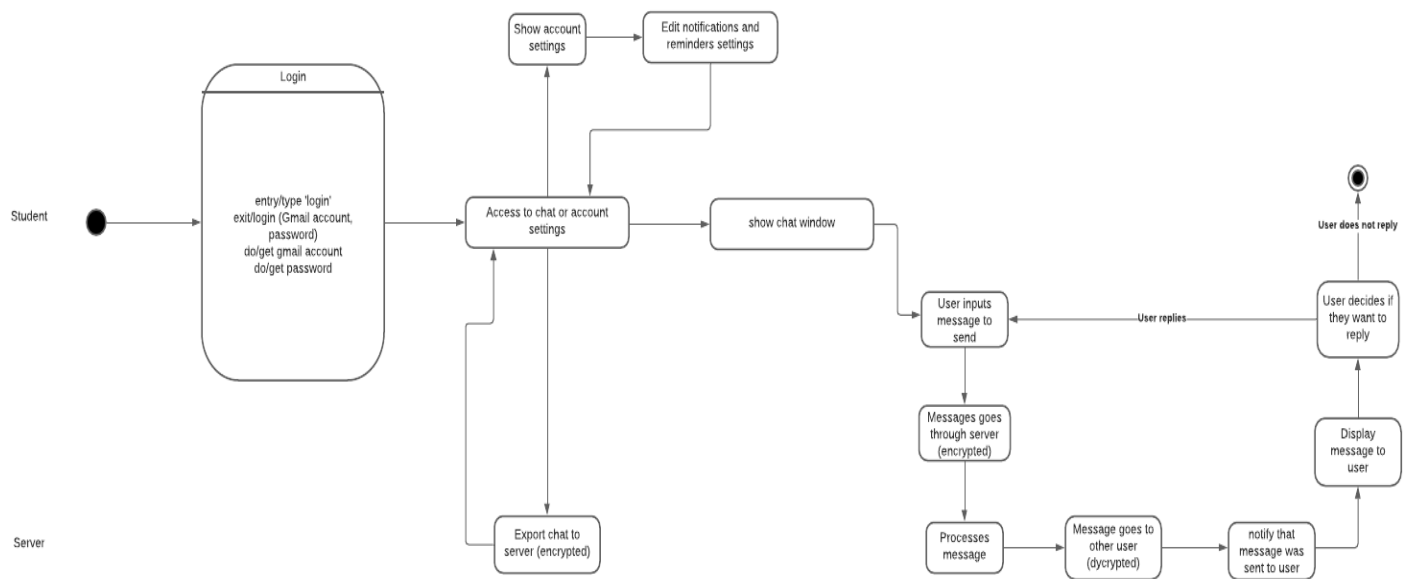


Figure 13: Student State Transition Diagram for PSUT Chatroom

State Transition Diagram Student: Explains the process that students go through to access settings and send/receive chat and the synchronization with the server.

#### 4.1.7.2 Professor State Transition Diagram

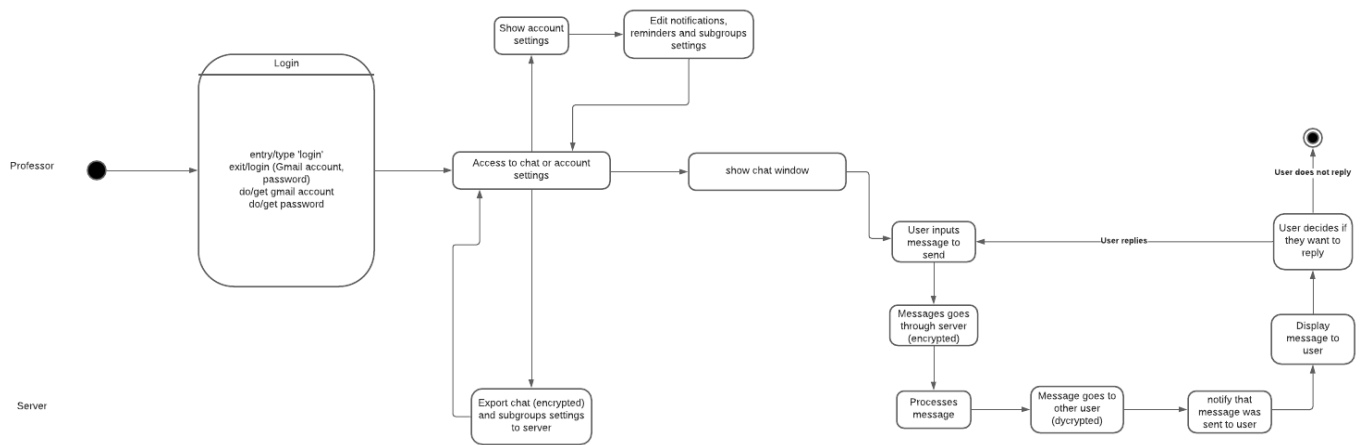


Figure 14: Professor State Transition Diagram for PSUT Chatroom

State Transition Diagram Professor: Explains the process that professors go through to access settings and send/receive chat and the synchronization with the server.

#### 4.1.7.3 Super Admin State Transition Diagram

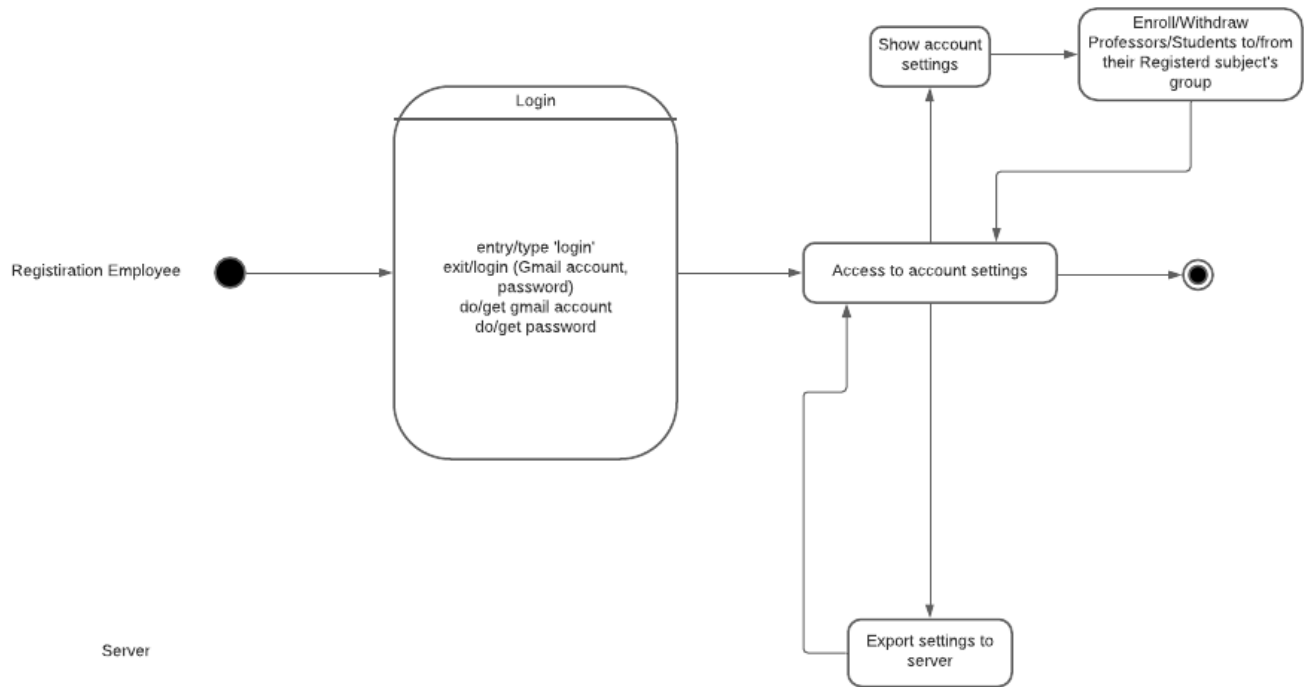


Figure 15: Super Admin State Transition Diagram for PSUT Chatroom

State Transition Diagram Super Admin: Explains the process that Registration Employee goes through to enroll or withdraw students and professors to/from their registered subject's group and then synchronize it with the server.



#### 4.1.7.4 Manage Chat State Transition Diagram

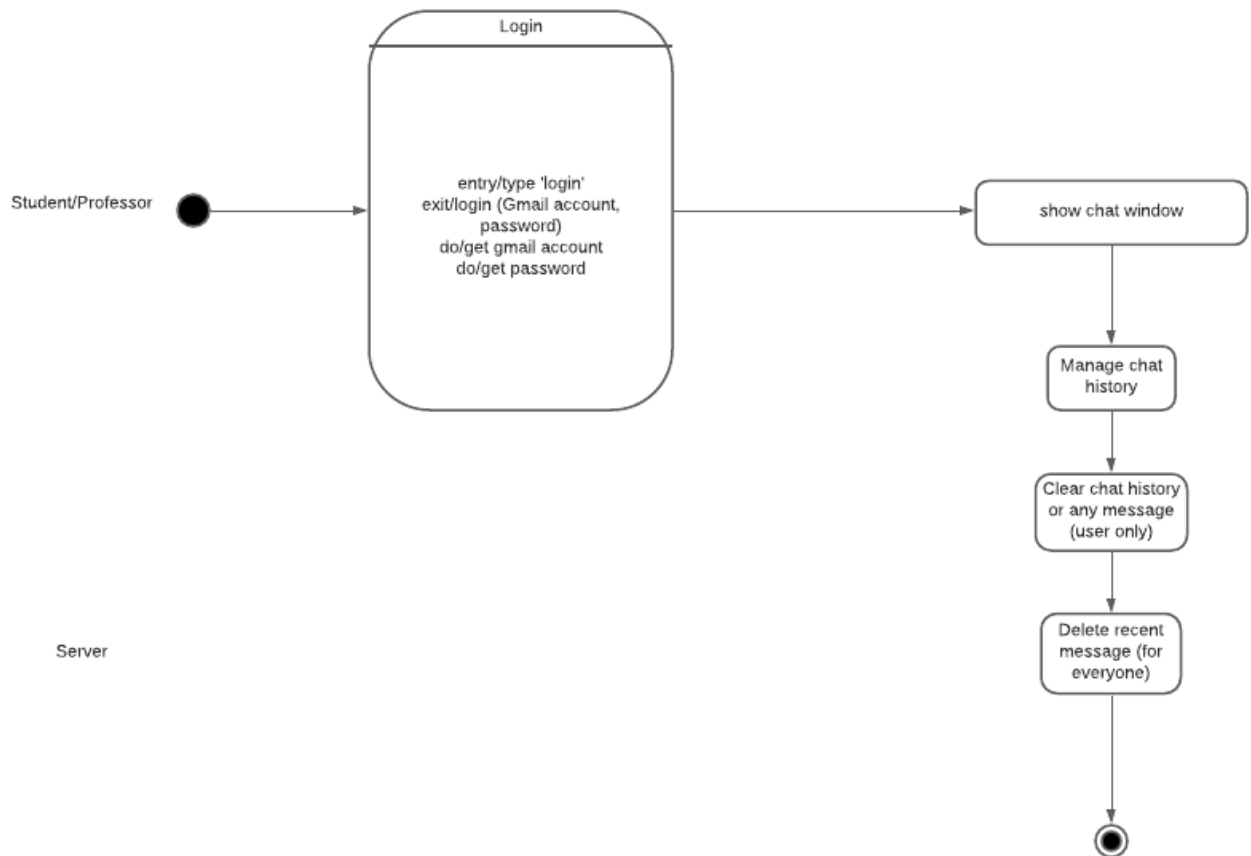


Figure 16: Manage Chat State Transition Diagram for PSUT Chatroom

State Transition Diagram Manage Chat: Explains the process that both students and professors go through to manage chat history and clear chat or any message for them or for everyone if it's a recent message and the synchronization with the server.

## 4.2 Physical Model Design

### 4.2.1 User Interface Design



Figure 17: Logo



Figure 18: User Interface for Login

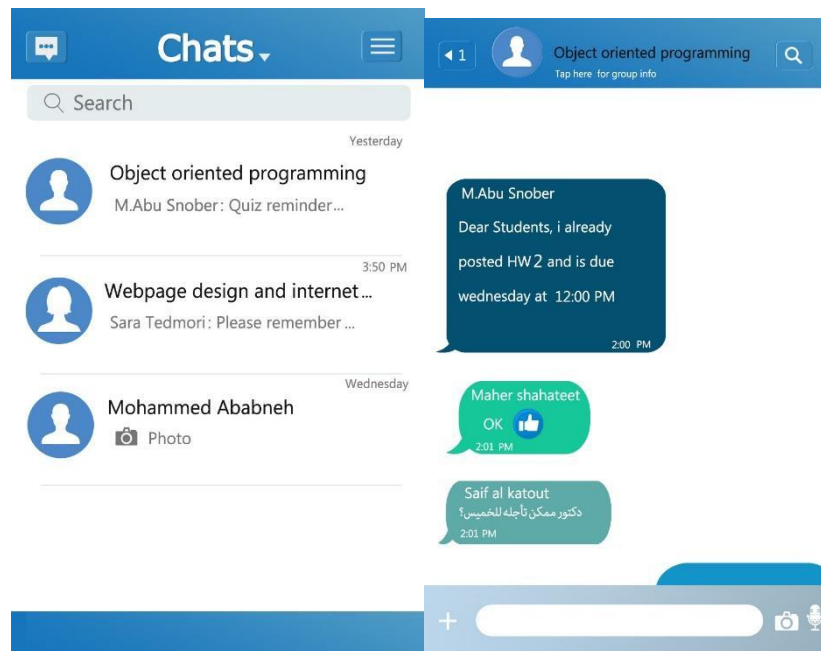


Figure 19: User Interface for Main Menu.

Figure 20: User Interface for Chat.

## 4.2.2 Database Design

### 4.2.2.1 Database Entity Relationship Diagram

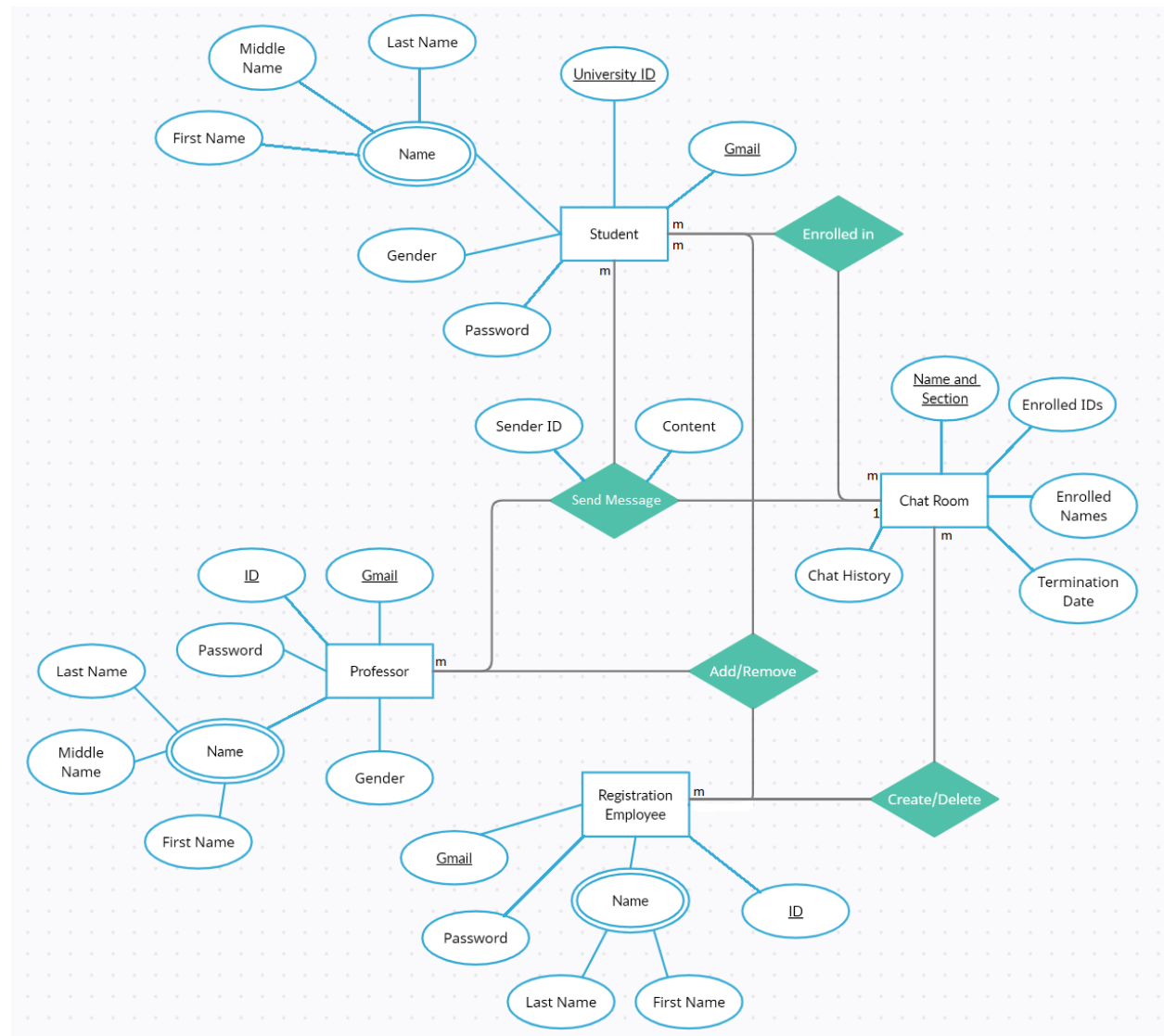


Figure 21: Entity Relationship Diagram for PSUT Chatroom

Figure 19 represents the ERD showing entities with their attributes and relations in the database.

#### 4.2.2.2 Database Schema

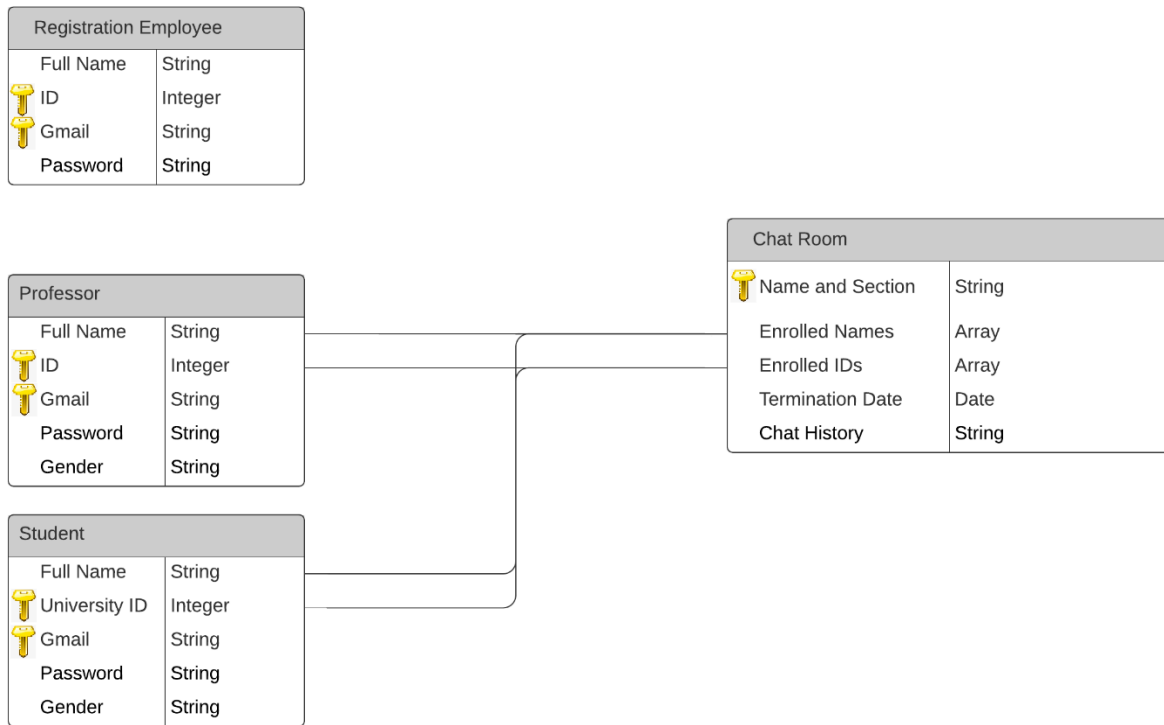


Figure 22: Database Schema for PSUT Chatroom

### 4.2.3 User Reports

Name and Section	Enrolled Names	Enrolled IDs	Number of Sent Messages

## Chapter 5

# Implementation

### 5.1 Programming Languages

The application is developed for mobile users so we have chosen the most suitable languages to use the needed resources to achieve the best results.

#### 5.1.1 Front-End

For mobile development, we used react-native to develop a native application for different mobile operating systems.

#### 5.1.2 Back-End

We used ASP.NET Core for cross-platform and performance, “ASP.NET Core is designed to allow runtime components, APIs, compilers, and languages to evolve quickly, while still providing a stable and supported platform to keep apps running.”

#### 5.1.3 Database

We used PostgreSQL (Postgres) a free and open-source relational database management system that works on all platforms, has bindings with all languages, robust and reliable.

## 5.2 Tools, APIs and Libraries

Tools/API's	Description
Entity framework	Object-relational mapper to work with a database using .NET objects, it manages entities and stores/retrieves data most efficiently.
Docker	Docker is a tool that uses OS-level virtualization to wrap the software in packages (containers).
Fluent Validation	A validation library for .NET is used to put validation rules for objects.
Auto Mapper	A library that transforms one object type to another type.
Swagger	An interface for description of RESTful APIs using JSON.

*Table 8: Tools, APIs and Libraries*



## 5.3 Main Algorithms and Coding Conventions

### 5.3.1 General User

#### 5.3.1.1 Email Validation

```
public class LoginDtoValidator : AbstractValidator<LoginDto>
{
    public LoginDtoValidator(AppDbContext dbContext, IOptions<AppOptions> appOptions)
    {
        RuleFor(d => d.Email)
            .EmailAddress()
            .Must(e => e.EndsWith(appOptions.Value.UniversityEmailDomain, StringComparison.OrdinalIgnoreCase))
            .WithMessage("The email doesn't belong to the university domain.")
            .WithMessage("Non instructor can only update himself.");
        RuleFor(d => d.GoogleToken)
            .NotEmpty();
    }
}
```

Figure 23: Email Validation

#### 5.3.1.2 Get Conversation

```
public async Task<IActionResult> Get([FromBody] int[] conversationsIds, bool metadata = false)
{
    var conversations = GetPreparedQueryable(metadata);

    var existingConversations = conversations.Where(g => conversationsIds.Contains(g.Id));
    var nonExistingConversations = conversationsIds.Except(existingConversations.Select(g => g.Id)).ToArray();

    if (nonExistingConversations.Length > 0)
    {
        return StatusCode(StatusCodes.Status404NotFound,
            new ErrorDto
            {
                Description = "The following conversations don't exist.",
                Data = new() { ["NonExistingConversations"] = nonExistingConversations }
            });
    }
}
```

Figure 24: Get Conversation

### 5.3.1.3 Create Message

```
public async Task<ActionResult<MessageMetadataDto>> Create([FromBody] CreateMessageDto dto)
{
    var user = this.GetUser(!);
    Message message = new()
    {
        Content = dto.Content,
        ConversationId = dto.ConversationId,
        ReferencedMessageId = dto.ReferencedMessageId,
        SenderId = user.Id,
        SendingTime = dto.SendingTime,
        EncryptionSalt = new byte[] { }
    };
};
```

Figure 25: Create Message

# Chapter 6

## Testing

### 6.1 Testing Approach

We have used Integration Testing to determine if all of our system components can work correctly with each other.

### 6.2 Testing Tools

We have used xUnit to run each test and generate a testing report.

Also, we have used RestSharp to communicate with the server while testing.

### 6.3 Tested Features

- 1- Backup: Testing if the system will backup all tables correctly in json format to a Zip archive.
- 2- Encryption: If the system will encrypt files before storing them and it will decrypt them before returning files to the client.
- 3- Real time updates: If the system will notify each client when an update happens using Web Sockets.
- 4- Authorization: The system will check if the caller has the required permissions to invoke the action.

## 6.4 Discussion

```
Starting test execution, please wait...
A total of 1 test files matched the specified pattern.
/home/abdallah/Desktop/UniChatApplication/Backend/Server.Tests/bin/Debug/net6.0/Server.Tests.dll
[xUnit.net 00:00:00.00] xUnit.net VSTest Adapter v2.4.3+1b45f5407b (64-bit .NET 6.0.0-rtm.21564.1)
[xUnit.net 00:00:00.78]   Discovering: Server.Tests
[xUnit.net 00:00:00.82]   Discovered: Server.Tests
[xUnit.net 00:00:00.83]   Starting: Server.Tests
    Passed Server.Tests.IntegrationTests.TestRealtimeUpdates [1 s]
[xUnit.net 00:00:03.20]   Finished: Server.Tests
    Passed Server.Tests.IntegrationTests.TestBackup [321 ms]
    Passed Server.Tests.IntegrationTests.TestAuthorization [24 ms]
    Passed Server.Tests.IntegrationTests.TestEncryption [32 ms]

Test Run Successful.
Total tests: 4
    Passed: 4
Total time: 3.8542 Seconds
```

Figure 26: Test Results

## Test run details

Total tests	Passed : 4	Pass percentage	Run duration
4	Failed : 0	100 %	3s 830ms
	Skipped : 0		

## All Results

/home/abdallah/Desktop/UniChatApplication/Backend/Server.Tests/bin/Debug/net6.0/Server.Tests.dll

## Informational messages

```
[xUnit.net 00:00:00.00] xUnit.net VSTest Adapter v2.4.3+1b45f5407b (64-bit .NET 6.0.0-rtm.21564.1)
[xUnit.net 00:00:00.78] Discovering: Server.Tests
[xUnit.net 00:00:00.82] Discovered: Server.Tests
[xUnit.net 00:00:00.83] Starting: Server.Tests
[xUnit.net 00:00:03.22] Finished: Server.Tests
```

Figure 27: Test Details

## Chapter 7

# Conclusions and Future Work

Because PSUT Chatroom implements the idea of auto-enrolling students into their own courses' groups with their professors and signing them up automatically it will reduce the hassle on both the professors and students of doing it manually. Adding to that it helps in increasing the communication between professors and students.

Since we are in an era of development and improvement of the online learning experience PSUT Chatroom will come in handy in the communication and file-sharing part making the experience smoother and easier, thus complementing the online learning experience.

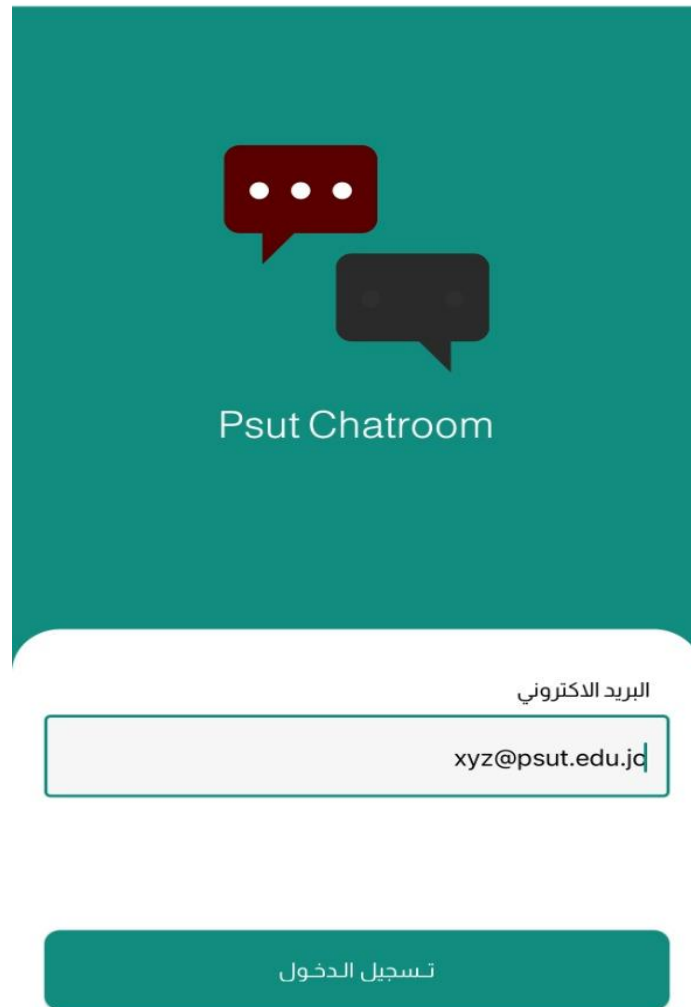
Furthermore, every application will always need improvements and updates to provide a smooth experience and implement new ideas that keep the users interested and keep the application bug-free. Thus, new features must be implemented as future work to improve the application.

Some of these features are:

- Disk level encryption: PSUT Chatroom has end-to-end chat encryption but the media like; files, photos, and videos are not encrypted. This feature will further improve security.
- Threads: It is important to have an organized way to communicate and message threads make it easier to group messages and make the conversation much smoother and neater.
- Cross-platform support: PSUT Chatroom is not available on iOS devices because it requires a MacBook.
- Generate statistics: The admin will be able to generate statistics whenever needed at both the student and group levels.
- Calendar: Integrate the calendar in the PSUT Chatroom application in the future to set reminders.
- Web Application: Develop a web application to increase the compatibility of PSUT Chatroom, and add it as a feature in the Regnew.

## Appendix A

# Users' Manual



The image shows a login page for 'Psut Chatroom'. The background is a teal color with a white rounded rectangle at the bottom. At the top, there are two speech bubble icons, one red and one dark grey. Below them, the text 'Psut Chatroom' is written in white. In the white rounded rectangle, there is a label 'البريد الالكتروني' (Electronic Mail) in Arabic. Below the label is a text input field containing the email address 'xyz@psut.edu.jd'. Below the input field is a teal button with the text 'تسجيل الدخول' (Login) in Arabic.

Figure 28: Login Page

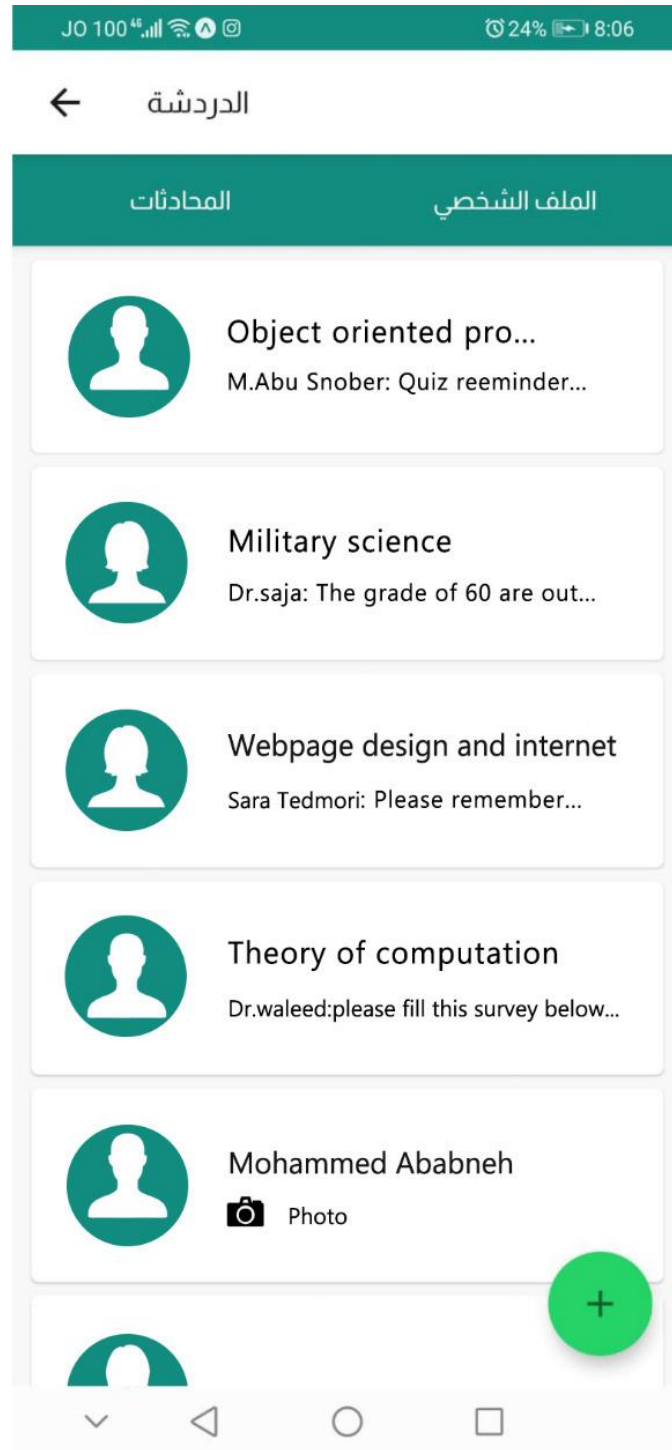


Figure 29: Main Menu

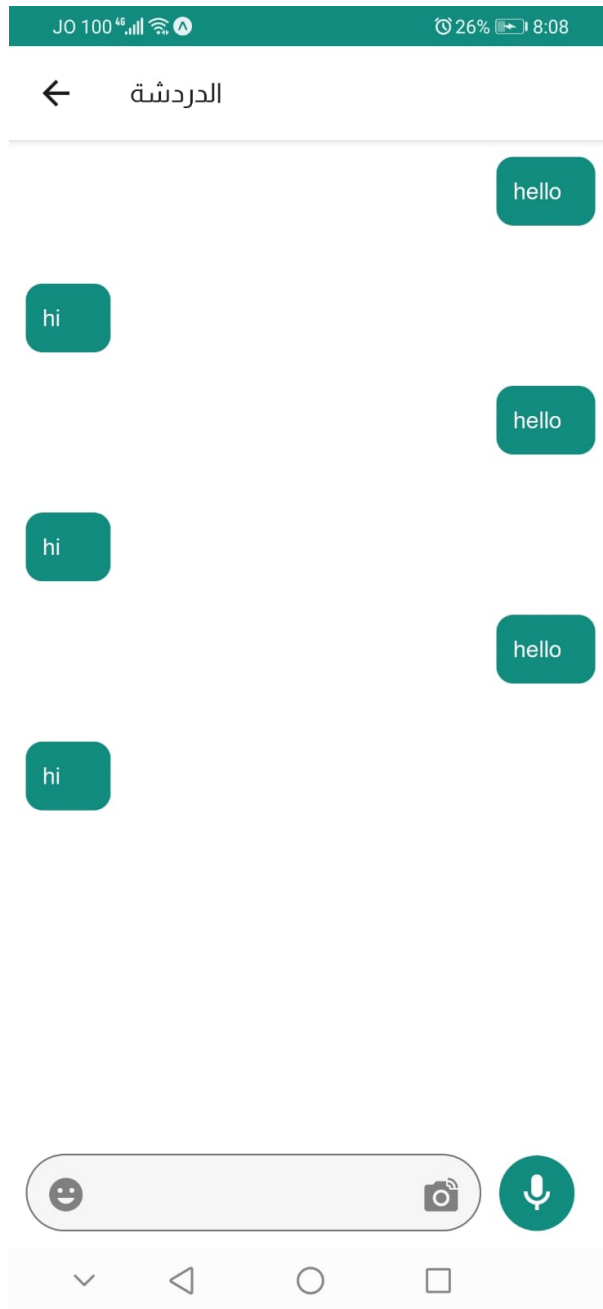


Figure 30: Chat



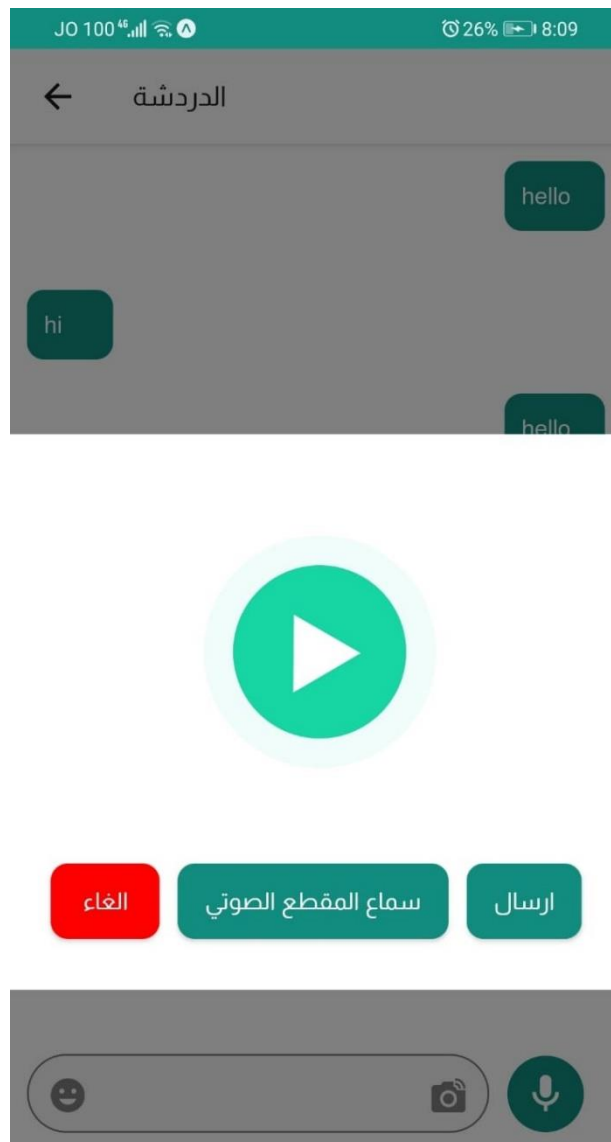


Figure 31: Voice Notes

## Appendix B

# Document Changes

ID	Name	Is implemented	Reason
DC1	Login	Yes	-
DC2	Forgot Password	Yes	-
DC3	Send Messages	Yes	-
DC4	Create Subgroup	No	Instead, we gave the professor the ability to create a whole new group, so that they would be able to merge students from multiple sections.
DC5	Remove Student	Yes	-
DC6	Delete Chat Room	Yes	-
DC7	Generate Statistics	No	Time constraint.
DC8	Automatic Course Enrollment	Yes	-
DC9	Notification System	Yes	-
DC10	Calendar	No	Time constraint.
DC11	Create Chat Room	Yes	-
DC12	Add Student	Yes	-
DC13	Ping	Yes	We added a feature that allows the student to request a one-to-one group with a professor.

Table 9: Document Changes

## Appendix C

# Code Documentation

We used swagger which provides user interfaces to our RESTful API's, it made the front-end development much easier for us.

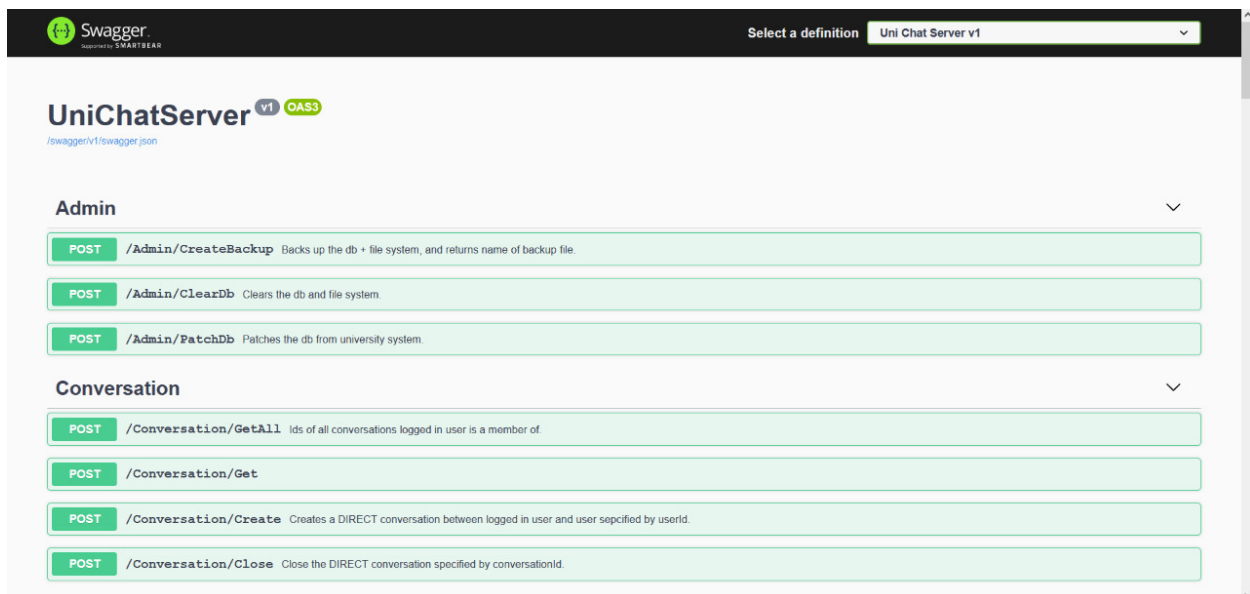


Figure 32: Swagger Sample