UNIVERSITY OF SCIENCE AND TECHNOLOGY OF HANOI

UNDERGRADUATE UNIVERSITY GROUP PROJECT

REPORT

Online Academy for weight loss and healthcare Mobile Cross Platform

by DO Thanh Hieu, TRAN Trung Hieu,

PHI Nguyen Hai Minh, NGUYEN Huu Nhan, DO Dinh Phuc, NGUYEN Quang Khai

Supervisor : DOAN Nhat Quang

February 8, 2022

Table of Contents

Acknowledgement

Our report would not be completed without the help of many people to whom we wish to express the thankfulness.

We express our sincere thanks to our project guide, Dr.DOAN Nhat Quang for his enthusiastic guidance, precious feedback and constructive instruction all over the project. He gave us insightful suggestions

We would be pleased to extend our deepest gratitude to all teachers in the ICT department, who always supported and nurtured us in the past two years.

Finally, thanks should also go to all of our members, who work so hard to complete this project.

I INTRODUCTION

1.1 Context and Motivation

As we have known, Covid-19 pandemic has existed in our daily life for almost 3 years, since 2019. It has changed the way people live which was offline to mostly online. Most of our daily tasks such as working or studying are switched to online via some meeting platforms, for example Zoom or Google Meet.

We found out that there is no specific online platform that helps people to keep their body fit and balanced because in the pandemic situation when people spend most of their time at home sitting in front of the computer screen, their health will not be taken care of regularly. One of the reasons for that is fitness places are not allowed to serve in the lock-down situation.

Based on the above, we put up with an idea of an Online Academy for Weight Loss and Health Care Mobile Application. The purpose of this application is based on users' health index and metric, in association with the Health Experts to give users their own exercises.

1.2 Project Objectives

In this Group Project, we aim at developing a software for managing users and sending to users their own exercises from the collaborated health experts, our goals include:

- Develop an interface for users to keep track of their health, send messages to their Personal Trainers and Experts.
- Allow Personal trainers and Experts can reply users' message, feedback,...and keep track of their trainee's health
- The UI of the software must be user-friendly for every type of customers, Personal Trainers and Experts.

1.3 Thesis Structure

The thesis will contain all the information necessary to reproduce the result.

- Firstly in **Section I: Introduction**, we will introduce the exciting problem in managing
- Then in **Section II: Objective**, we will define expected requirements of the project, provide a brief overview of the function of the system and the reasons for its development. After that, we will describe the scenarios, use cases, object model, and dynamic models for the system
- In, **Section III: Requirement Analysis**, this section contains the complete functional specification, including navigational paths representing the sequence of screens.
- In Section **IV: Methodology**, we will list all the tools and techniques used in the project, the reasons why they are chosen and the detailed use cases implementation.
- In Section **V: Results**, we will list all the functionalities which are implemented in the system.
- Final in, **Section VI: Conclusion and Future Work**.

II OBJECTIVE

In this session, we will define expected requirements of the project, provide a brief overview of the function of the system and the reasons for its development

2.1 Desired Features

The main goal of this project is to develop a Mobile application for 3 types of users : Customers, Personal Trainers/Experts and Administrators. Customers can access our application in order to view their own bodies' stats, choose, message and call their Personal Trainers/Experts, and also they can have their own nutrition suggestions. Personal Trainers/Experts can view their customers, give them their exercises and nutritional advice. Finally, Administrators can access the system to create, delete, read and update customers' and personal trainers/experts' information.

- **GENERAL FEATURE**
    - **FEATURE 1: Authenticate**
        - **SUB-FEATURE 1.1: Register** allow users to create a new account
        - **SUB-FEATURE 1.2: Login** allow users to access into the system
        - **SUB-FEATURE 1.3: Logout** allow users to log out of the system
- **CUSTOMERS FEATURE**
    - **FEATURE 2: View Free Courses**
    - **FEATURE 3: Update bodies and health status** allows Customers to keep track of their health so that they can manage their exercise and workout.
    - **FEATURE 4: View Paid Courses** allows Customers to :
        - **SUB-FEATURE 4.1** : View advanced courses
        - **SUB-FEATURE 4.2** : Self-customized courses and nutritional
        - **SUB-FEATURE 4.3** : Private message/call with their chosen Personal Trainers/Experts
- **PERSONAL TRAINERS/EXPERTS FEATURE**
    - **FEATURE 5: View Customers Information** allows Personal Trainers/Experts to search their Customer by name and view Customer Information.
    - **FEATURE 6: Communicating With Customers**

        - **SUB-FEATURE 6.1: Private Message** allows Personal Trainers/Experts to communicate with their customers via chat
        - **SUB-FEATURE 6.2: Private Call** allows Personal Trainers/Experts to help their customers communicate via Audio/Video call
        - **SUB-FEATURE 6.3: Suggest Nutritional** allows Personal Trainers/Experts to suggest reasonable nutritional/meals to their customers
- **ADMINISTRATOR FEATURE**
    - **FEATURE 7: Manage Users' Information**
        - **SUB-FEATURE 7.1: View Users' Information** allows Administrator to create, read, update and delete users (include Customers, Personal Trainers and Experts) by name and their transaction
        - **SUB-FEATURE 7.2: View Message/Call** from customers, personal trainers and experts to ensure that they do not send invalid/harmful messages.

2.2 Expected Outcome

Our system aims at creating a connection between customers and personal trainers/experts for a better experiment of enrollment, managing and modifying information. The specific goals include:

- Develop a backend engine for managing enrollment and interview data such as Customers Admission Form,...)
- Develop a beautiful interface on the mobile with interaction between Customers, Personal Trainers/Experts or Administrators and the System – The application should have all the features mentioned in Desired Features

III REQUIREMENT ANALYSIS

In this section, we will describe a brief overview of the functions in the project, the scenarios and use cases for the system. This part contains the complete functional specification, including navigational paths representing the sequence of screens.

1. Overall System Requirements

In general, this application must satisfy the following main requirements:

– A login system with authentication

– For Customer

- Get exercise
- Get advice for nutrition
- Get advice for exercise
- Private messages/call with Personal Trainers/Experts

- For Personal Trainers/Experts

- Give advice for nutrition

- Give advice for exercise
- Private messages/call with Customers

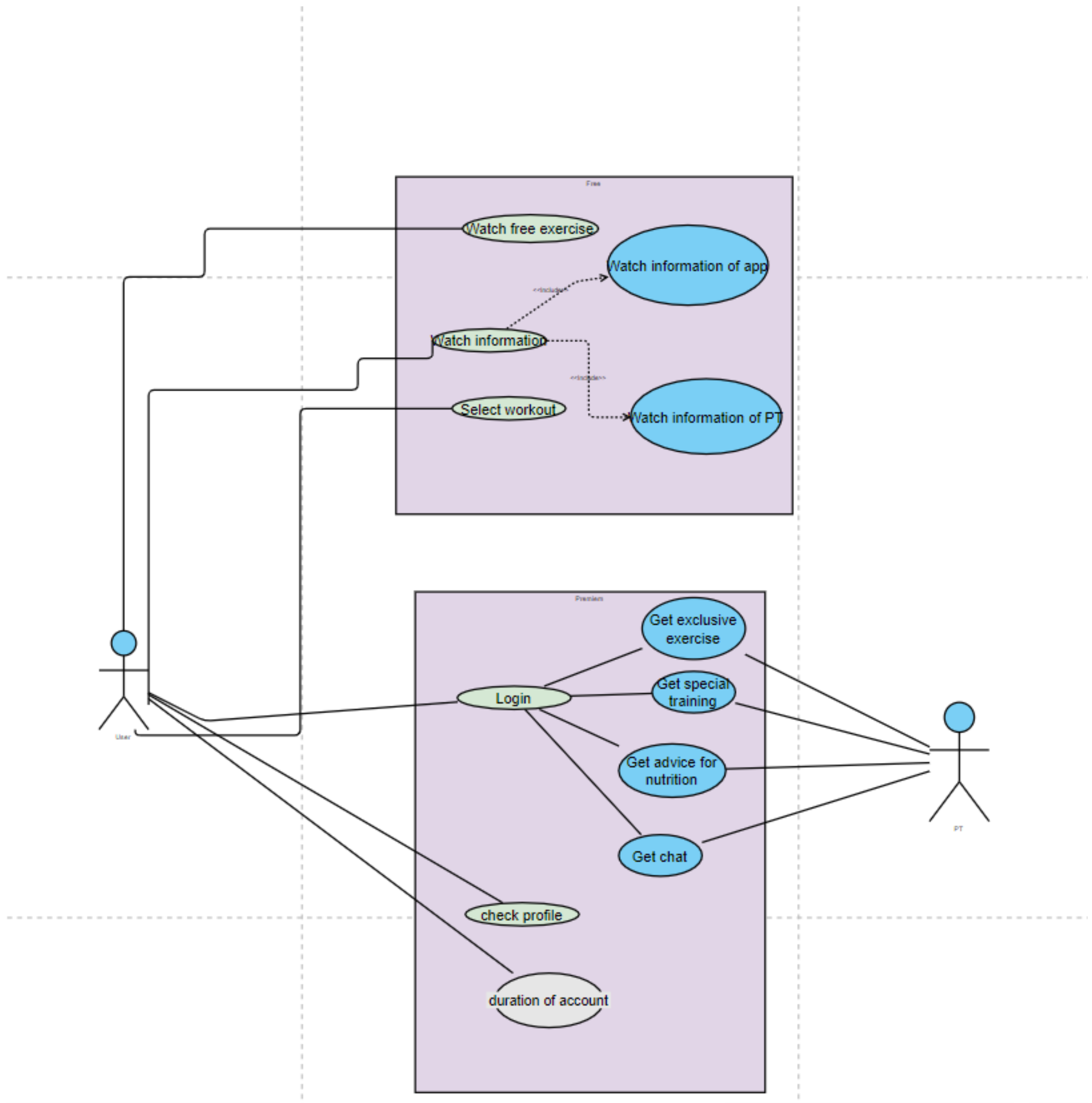
– For Administrators

- Create, read, update and delete Customers/Personal Trainers/Experts Information
- Manage customers payment transactions

2. Use Cases

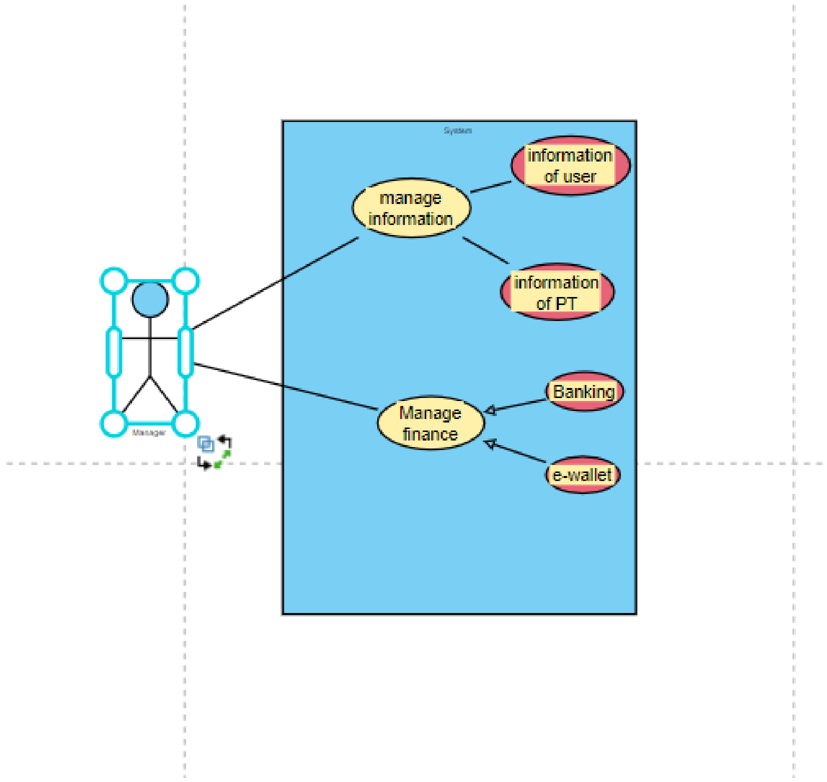
a. Use Cases Diagram

Figure : Use Case Diagram



This figure shows the Use-case diagram that consists of all features/sub-features of the system.

Figure



b. User Characteristics

There are 3 types of users that interact with the system: **Customers, Admin** and **Personal Trainers/Experts**. Each user type has different uses in the system so each of them has their own requirements:

– **Customers** : a user who uses the system to get information, send and receive message

- **Personal Trainers/Experts** : A user who uses the system to give information, send and receive message

– **Admin** : a user who uses the system to create, read, update, delete information, and also manage Customers and Personal Trainers/Experts activities

c. Use Cases and Scenario Description

i. Use case: Register

a. Brief description

This use case describes how an user registers to the system.

b. Flow of events

Basic Flow Table: Register Basic Flow

Actor Action	System Action	Data

1. The actor wishes to register into the system.	2. The system requests the actor to select user type: Customer or Personal Trainer/Expert.	
3. The actor select his/ her type	4. The system requests the actor to enter email, username and password.	
5. The actor enters the requested information.	6. The system checks if the entered email is unique from the table "user" in Database. If unique, the system saves the user information and assigns a unique id to the user.	Email Username Password

c. Alternatives Flow

- Email is already in use at step 5 in the Basic Flow:

Table : Register Alternatives Flow 1

Actor Action	System Action	Data
5. The actor enters the requested information.	6. The system announces that the email is in use and requests the user to re-enter email, username and password.	
7. The actor enters another username, email and password.	8. The system checks if that username is used or not. If not, the system saves the user information. The system assigns a unique id to the user. Else, repeat steps from step 3.	Email Username Password

- Admin wants to register at step 3 in the Basic Flow:

Table : Register Alternatives Flow 2

Actor Action	System Action	Data
3. The actor select type "User" and wishes to register into the system	4. The system sends an error message "Please contact us to register".	

d. Special Requirements

Only Customers can Register into the system. Admin have to contact Staff to register.

e. **Pre-conditions** User is not logged in.
f. **Post-conditions**

If the use case is successful, the user registered successfully. If not, the system state is unchanged.

ii. Use case: Login

a. **Brief description**

This use case describes how a student logs to the system. The actor is customer

b. **Flow of events**

Basic Flow Table : Login Basic Flow

Actor Action	System Action	Data
1. The actor wishes to enter into the system.	2. The system requests the actor to select user type: Customer or Admin.	
3. The actor select his/ her type	4. The system requests the actor to enter email, username and password.	
5. The actor enters the email and password.	6. The system validates the email, password and type entered from the user table in the Database. If valid, the system logs the actor in.	Email Username Password

- Invalid login parameters. At step 5 in the Basic Flow:

Table : Login Alternatives Flow 1

Actor Action	System Action	Data
5. The actor enters the username and password.	6. The system validates the username and password. If the username or password is invalid, the system displays an error message " User not found " and requests the	Email Username Password

	user to re-enters username and password.	

- Users select invalid type. At step 5 in the Basic Flow:

Table : Login Alternatives Flow 2

Actor Action	System Action	Data
5. The actor enters the username and password.	6. The system validates the username and password. If the username or password is valid but the type is invalid the system displays an error message " You aren't Expert" and requests the user to re-enters username and password..	Email Username Password

d. Special Requirements None.

e. Pre-conditions

The user account must exist

f. Post-conditions

If the use case is successful, the user logged into the system successfully. Otherwise, the system state is unchanged.

iii. Use case: View Free Course

a. Brief description

This use case describes how a customer accesses the template course and nutrition advice that is provided for every user. The actor is a non-paid Customer.

b. Flow of events

Table : View Free Course Basic Flow

Actor Action	System Action	Data
1. The actor wishes to access free courses through the system.	2. The system requests the actor to choose : Their bodies' metrix, their wish and type of course they want.	

3. The actor choose the provided information	4. The system validates and show the chosen information.	Type of course

iv. Use case: View Paid Course

a. Brief description

This use case describes how a customer accesses the template course and nutrition advice that is provided for paid users. The actor is a paid Customer.

b. Flow of events

- **Basic Flow**

Table : View Paid Course Basic Flow

Actor Action	System Action	Data
1. The actor wishes to access advanced courses through the system.	2. The system will have to validate if the actor have done the subscription, then return the choice of course	Actor's Transaction
3. The actor choose the provided information	4. The system validates and shows the chosen information.	Type of course

v. Use case: Communicating

a. Brief description

This use case describes how a user communicates with other users. The actor is a Customer/Personal Trainer/Expert.

b. Flow of events

- **Basic Flow**

Table : Communicating Basic Flow

Actor Action	System Action	Data
1. The actor wishes to communicate with other users so the actor accesses the message tab.	2. The system returns the message tab.	

3. The actor choose other actor who they want to communicate with each other			Communicating
	5. When actor types the text and hit send button	6. The system take the message and deliver to the receiver	
	7. When actor hit Audio Call button	8. The system makes a call between the two actors	
	9. When actor hit Video Call button	10. The system makes a video call between the two actors	

c. Special Requirements None.

d. Pre-conditions

The user account must be declared as an already-subscription account

e. Post-conditions

If the use case is successful, the user can communicate via the system. Otherwise, the system state is unchanged.

vi. Use case: Suggesting Exercise and Nutrition

a. Brief description

This use case describes how a user can add new suggestion in exercising and nutrition. The actor should be a Personal Trainer/Expert.

b. Flow of events

Table : Suggestion Basic Flow

Actor Action	System Action	Data
1. The actor wishes to add a new suggestion, then access the suggestion tab.	2. The system returns the suggestion tab.	Text input
3. The actor click on Plus button to add new suggestion	4. The system returns a form for the actor to fill in.	

5. The actor fills all the needed information and then hits on Add New Suggestion.	6. The system validates the information and then put through the system to the target user.	

c. **Special Requirements** None.

d. **Pre-conditions**

The user account must be declared as an already-subscription account

e. **Post-conditions**

If the use case is successful, the user can communicate via the system. Otherwise, the system state is unchanged.

vii. Use case: Add User

a. **Brief description**

This use case describes how an Administrator adds a new user. The actor is an Administrator.

b. **Flow of events**

Table : Add New User Basic Flow

Actor Action	System Action	Data
1. The actor wishes to add new user through the system.	2. The system requests the actor to enters new user information	
3. The actor enters the user information	4. The system get the new data from the form.	
	5. The system adds new user information, and displays a successful message.	

c. **Special Requirements** None.

d. **Pre-conditions**

The administrator must login.

e. **Post-conditions**

If the use case is successful, the administrator will add new user successfully.

viii. Use case: Edit User Information

a. **Brief description**

This use case describes how an administrator edits user information. The actor is an administrator.

b. Flow of events

Table : Edit User Information Basic Flow

Actor Action	System Action	Data
1. The actor wishes to edit user information through the system.	2. The system requests the actor to enters new user information:	
3. The actor enters new user information	4. The system get the new data from the form.	
	5. The system makes changes to the previous information and displays a successful message.	

c. **Special Requirements** None.

d. Pre-conditions

The administrator must login.

e. Post-conditions

If the use case is successful, the administrator will edit user successfully.

ix. Use case: Delete User

a. Brief description

This use case describes how an administrator deletes users. The actor is an admin.

b. Flow of events

Table : Delete user's information Basic Flow

Actor Action	System Action	Data
1. The actor wishes to delete user information through the system.	2. The system dispatch an alert message to the user to make sure that the user want to delete the user information	
3. The actor decided to delete the user information.	4. The system delete all data of the user.	
	5. The system displays a successful message.	

Alternatives Flow

The actor chooses not to delete the user's information . At step 3 in the Basic Flow:

Table : Delete user's information Alternatives Flow 1

Actor Action	System Action	Data
3. The actor chose not to delete the user.	4. The system went back to its previous state.	

c. **Special Requirements** None.

d. **Pre-conditions**

The administrator must login.

e. **Post-conditions**

If the use case is successful, the administrator will delete the user successfully.

IV METHODOLOGY

In this section, we will list all the tools and techniques used in the project, the reasons why they are chosen and the detailed use cases implementation.

1. Tools and Techniques

a. SQL

With the amount of data in our system, it is important to choose the proper Database to manage the data. There are two types of Database: using SQL and NoSQL, each database has its own advantages and disadvantages.

Table : Differences between SQL and NoSQL

	SQL	NoSQL
Type of database	Relational Database	Non-relational Database
Schema	Pre-defined Schema	Dynamic Schema
Database Categories	Table based Databases	Document-based databases, Key-value stores, graph stores, wide column stores
Complex Queries	Good for complex queries	Not a good fit for complex queries
Hierarchical Data Storage	Not the best fit	Fits better when compared to SQL
Scalability	Vertically Scalable	Horizontally Scalable

SQL is suitable for a project with a fixed schema, high transaction, low maintenance, data security with a limited budget and NoSQL is for unstable schema, high availability, cloud computing, with in-built sharding.

In this project, the data will be updated regularly and it may be expanded in size, in its structure and relation.

Our system will be flexible. Therefore, the best solution for our system is using NoSQL Database.

b. Figma

Figma is a vector graphics editor and prototyping tool which is primarily web based. It's currently (arguably) the industry's leading interface design tool, with robust features which support teams working on every phase of the design. process.

c. Firebase

Firebase is a Backend-as-a-Service (BaaS). It provides developers with a variety of tools and services to help them develop quality apps, grow their user base, and earn profit. It is built on Google's infrastructure. Firebase is categorized as a NoSQL database program, which stores data in JSON-like documents.

Firebase Authentication provides backend services, and ready-made UI libraries to authenticate users. It supports authentication using passwords, phone numbers, popular federated identity providers like Google, Facebook and Twitter, and more..

Cloud Firestore is a flexible, scalable database for mobile, web, and server development from Firebase and Google Cloud.

With various advantages, we decided to use Firebase for the Backend-Side.

d. React Native

React Native is a Javascript framework for writing real, natively rendering mobile applications for iOS and Android. It's based on React, Facebook's Javascript library for building user interfaces, but instead of targeting the browsers, it targets mobile platforms. In other words: Web developers can now write mobile applications that look and feel truly "native", all from the comfort of a Javascript library that we already know and love. Plus, because most of the code you write can be shared between platforms, React Native makes it easy to simultaneously develop for both Android and iOS.

So that, to make this application to "cross-platform", we chose React Native in order to simplify the operation and updating in the future

e. Server Side

This is a Client side rendering web application, the Back-end takes the responsibility of creating a secure connection between the web browser and system's database. Our project depends heavily on Firebase. Specifically, we mainly use three features provided by this platform:

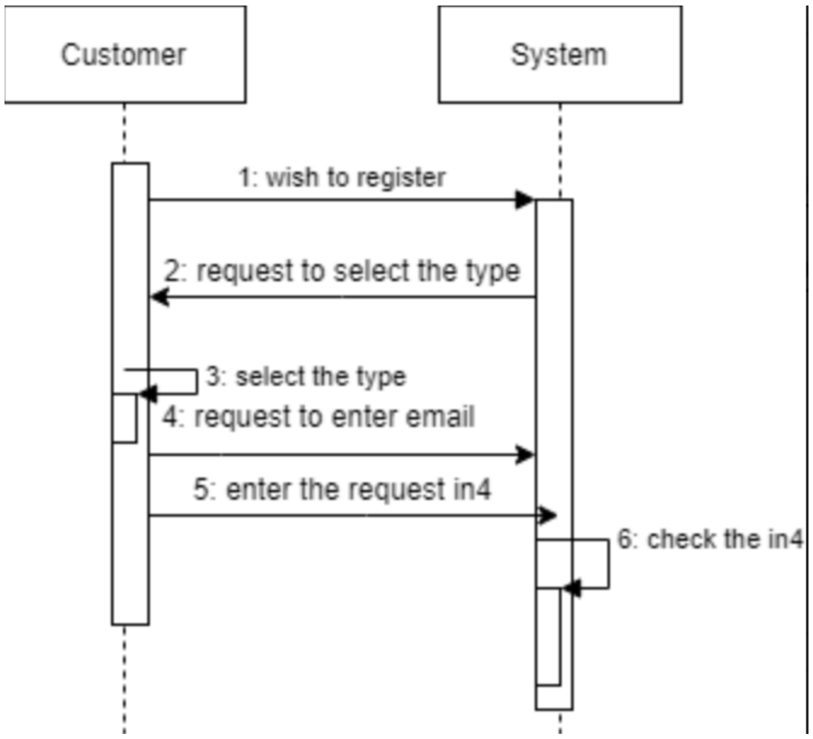
- Firebase Authentication: provides backend services and easy-to-use SDKs. It supports authentication using passwords, phone numbers, popular federated identity providers like Google, Facebook and Twitter, and more
- NodeJS for API processing (including receive requests, authenticating requests, communicating with database, return response in JSON form). Structure of an API contains:
- Define action.
- Request verification with token.

– Query from Database. - Response:
– Return a response data in JSON format if success.
– Return an error message if there is an error.

f. Use Cases Implementation

- **Login**

This use case describes how a user logs into the system.
The user can be a Customer, Personal Trainer/Expert or Admin.
When a user wants to login, the system requests the actor to select his/ her type (Customer, Personal Trainer/Expert or Admin). After the type is selected, the system displays a login screen and requests the user to enter email and password. Once the user enters the email and password, the system encrypts the password and validates the email, password and user type entered from the user table in the Database. If valid, the system logs the actor in. Otherwise, the system displays an error message and asks the user to re-enter the information.



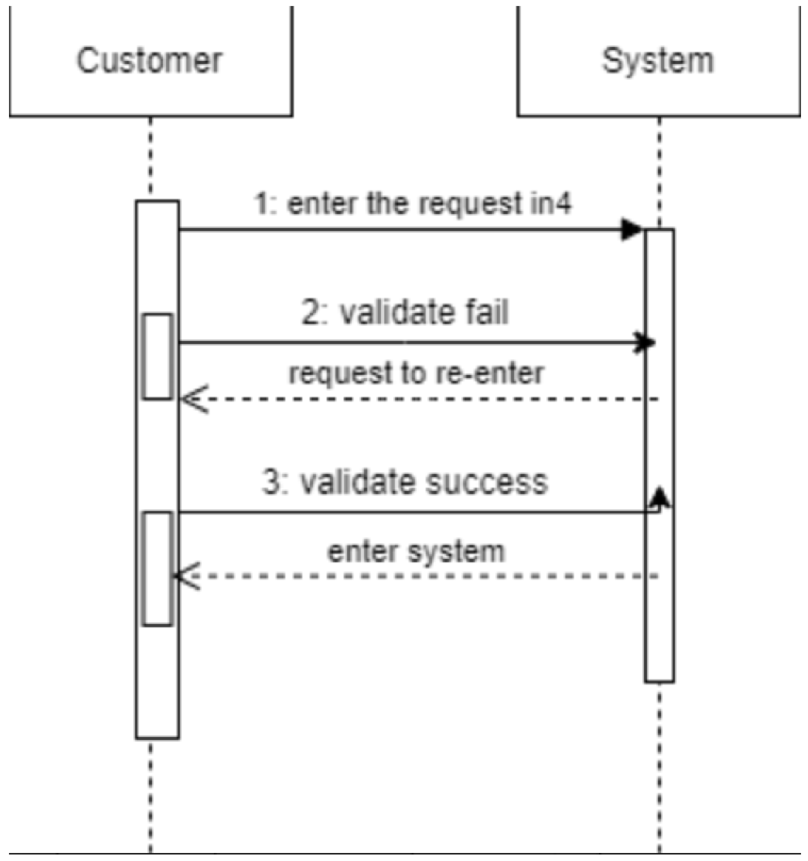


Figure: Login Sequence Diagram

- **Register**

This use case describes how a user registers into the system.
The user is a Customer.
When a user wants to register into the system (by clicking on "Create new account"), the system displays a register interface and requests the user to enter email, username and password. After the user enters the required information, the system checks if the entered email is unique from the table "user" in Database. If unique, the system encrypts the password and saves the user information, assigns a unique id to the user and displays a success message. Otherwise, the system displays an error message and asks the user to reenter the information.

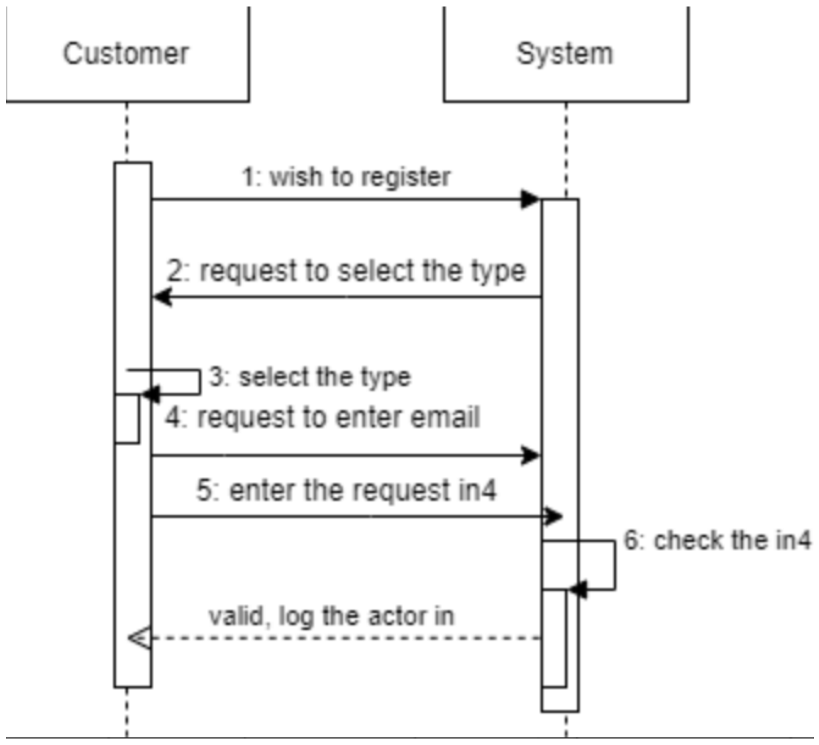


Figure: Register Sequence Diagram

- **View information (exercises, nutritional,..) :**

This use case describes how a user views their information about their exercises and nutrition advice.
The user is a Customer.
When a user wants to view the information through the system, the system displays a list which contains the type of workout they want to join (Eg: Running, jogging, gymming,...). After choosing their want, system will display the information they need to start/continue exercising, such as the process of exercising, nutritional information for the chosen type of exercise,...

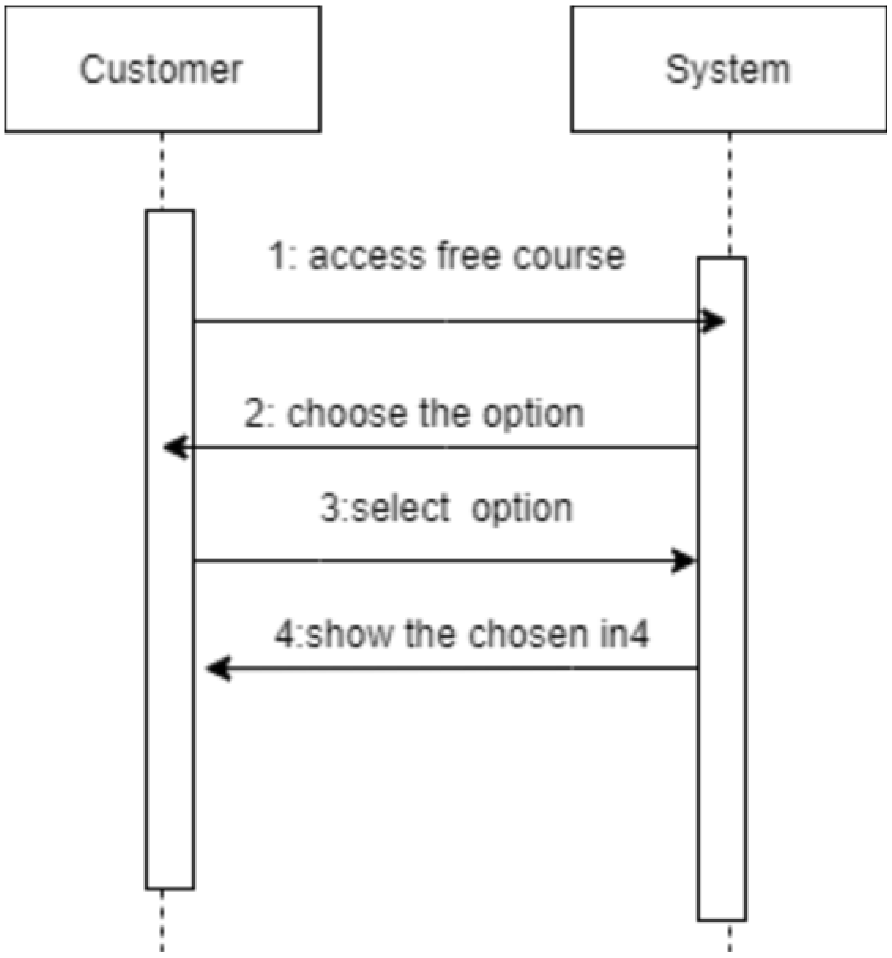


Figure : View Information Sequence Diagram

- **Communicating**

  This use case describes how a user communicates via message or audio/video call.
  The user is a Customer or a Personal Trainer/Experts.
  When a user wants to send message or audio/video call with another user via the system, they have to access the message tab (at the bottom of the screen - in the navigation bar) and get into a specific chat view - where there are a type box to type message, an audio call button and a video call button. To send a message, the user needs to input the text they want to send into the type box and hit the send button. The text will send to the system and then delivered to the one whose message belongs to.

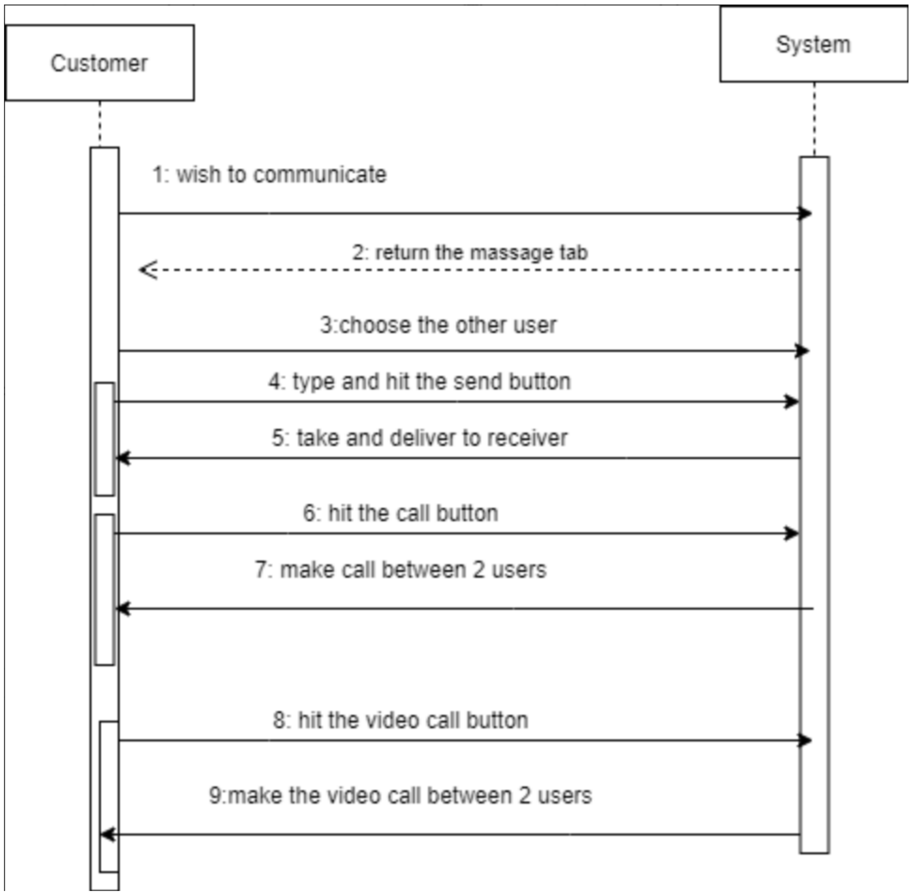

Customer
System
1: wish to communicate
2: return the massage tab
3:choose the other user
4: type and hit the send button
5: take and deliver to receiver
6: hit the call button
7: make call between 2 users
8: hit the video call button
9:make the video call between 2 users

Figure: Communicating Diagram

- **Suggesting Exercise and Nutrition**

This use case describes how a user gives suggestions and advice about exercise process, method and nutrition.

The user is a Personal Trainer/Expert.

When a user wants to send suggestions and advice about exercise process, exercise method and reasonable nutrition recommendations, they need to access the advice tab (at the bottom of the screen - in the navigation bar) and hit on the Plus button to add new advice. Finish filling all the needed information and hitting the update button, the system will receive and transfer to the Customers which chose that Personal Trainer/Expert mobile application.

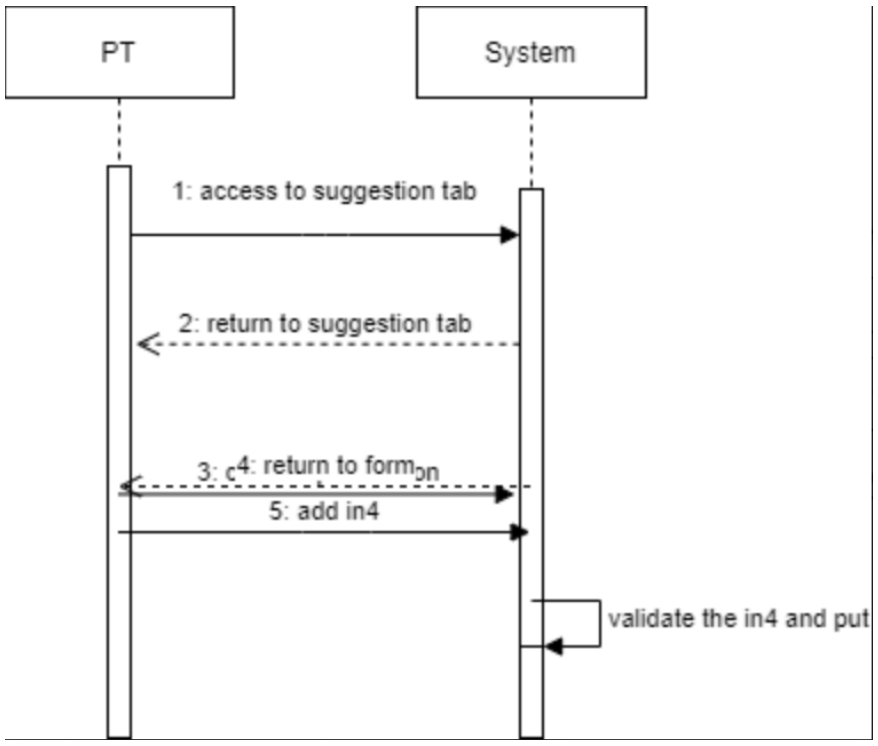


Figure: Suggesting Exercise and Nutrition Diagram

- **Create New User**

This use case describes how the admin creates a user.
The user is the admin.
When the admin wants to create a user, they have to fill in all the information of the user they want to create. After that, the system will send a create request to the server, now the server will check if the user exist or not, if the user existed, the system notice the admin and require them to refill the information, and if not, the system will save the new user into the database and send a success image.

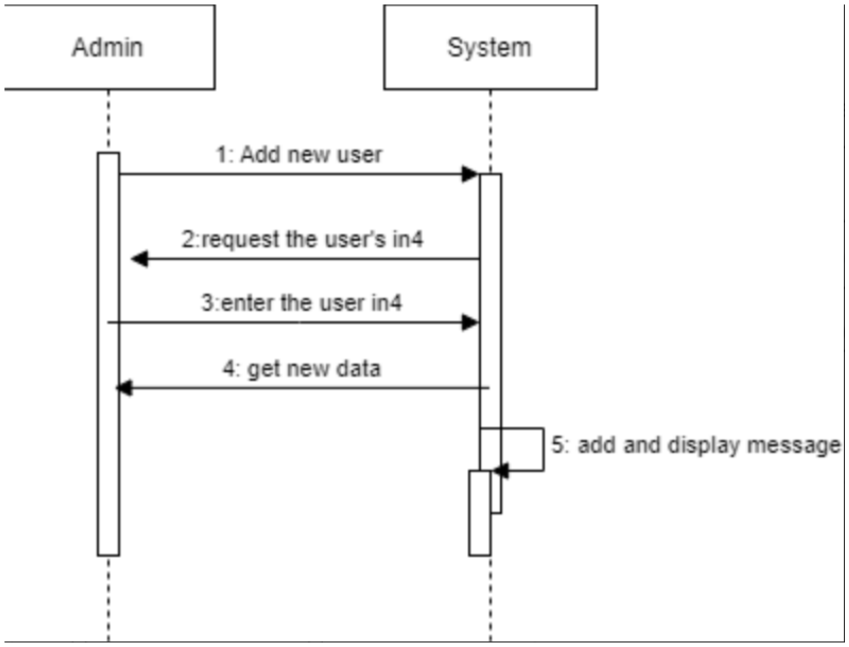


Figure: Create User Diagram

- **Read Users Information**

This use case describes how the admin read the users list.
The user is the admin.
When the admin wants to read the list of users. They can send a read request, after that, the system will send a read request to the server, then the server will send back the users list data to the system, and the system will display the users list data in the UI.

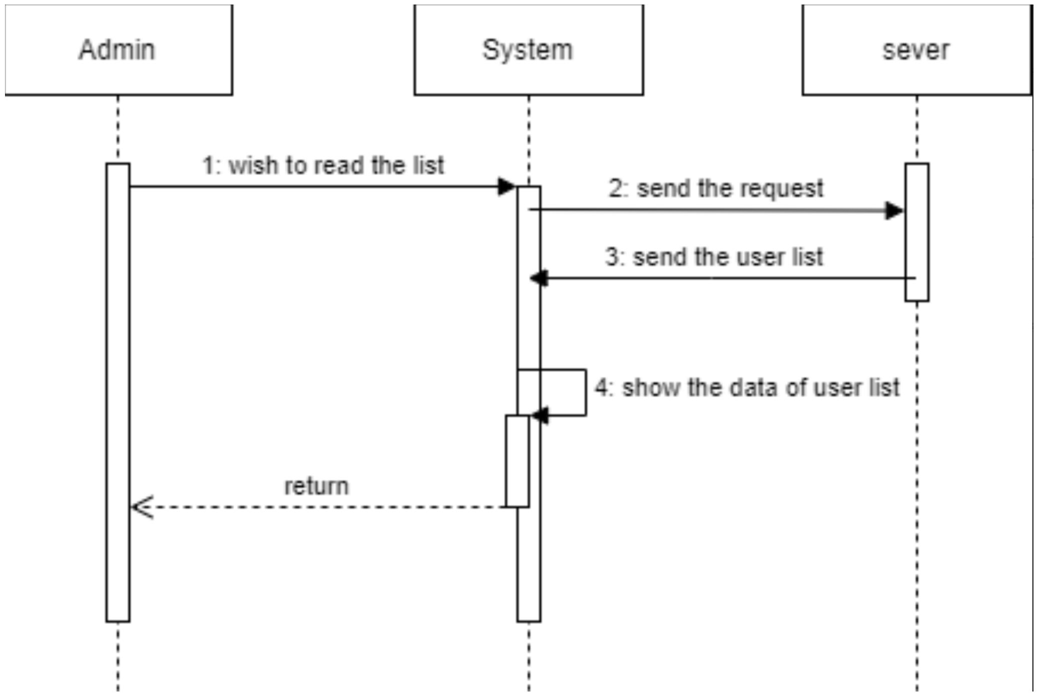


Figure: Read User Diagram

- **Update Users Information**

This use case describes how the admin updates users.
The user is the admin.
When the admin want to update users in the database, they will have to send a update the request, after that the system will send a update request to the server, the server will update the users list as the admin wanted, and send back the updated version of the users list after the admin require a read request.

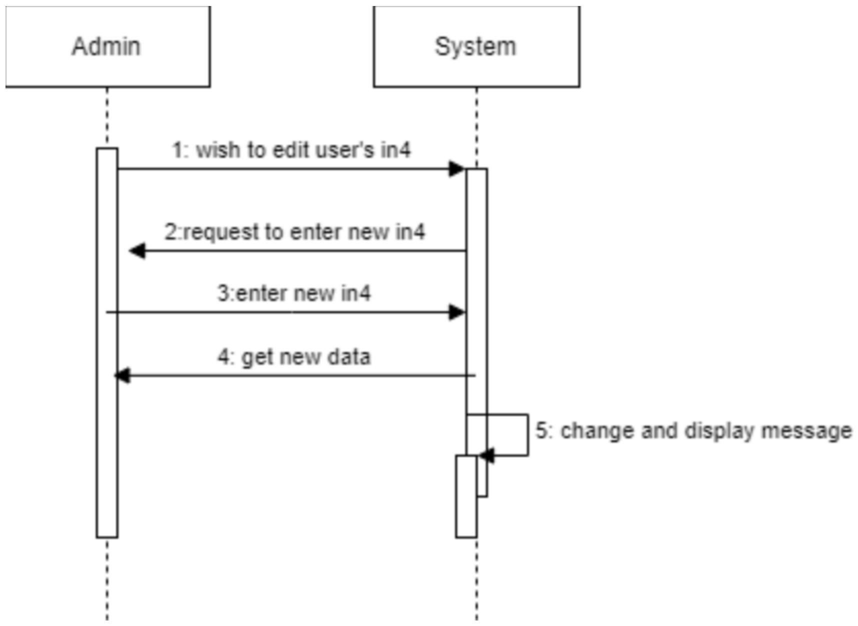

Admin
System
1: wish to edit user's in4
2:request to enter new in4
3:enter new in4
4: get new data
5: change and display message

Figure : Update Users Information Diagram

- **Delete Users**

This use case describes how the admin deletes Users.
The user is the admin.
When the admin wants to delete users from the database, they will have to make a delete request, after that, the server will send a delete request to the server, the server will delete the users as the admin wanted and send back a success message.

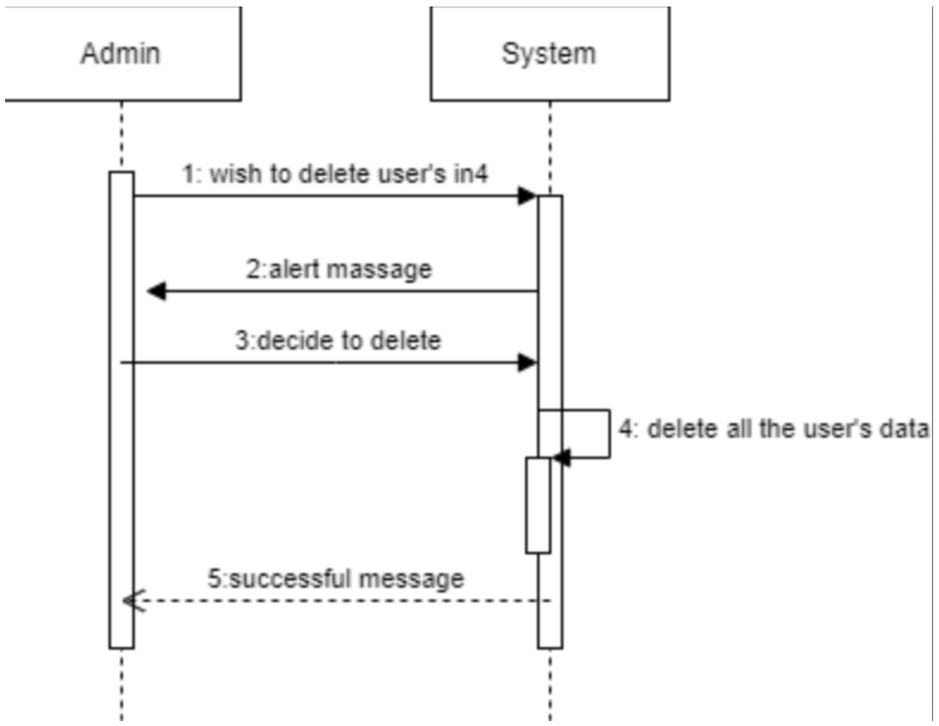


Figure 28: Delete User Diagram

V RESULT AND DISCUSSION

In this section, we will list all the finished functions in the system.

1. Result

In this system, these main functions have implemented completely:

– **User Interface:** The system's User Interface for Android's user is almost complete.

– **Register and Login:** Users can Create a new account and Access into the system.

– **Information, Stats, Message & Communication:** Users can see the template of the application (as we haven't deployed it on any store platforms).

2. Discussion

Despite being implemented completely, the main functions in this project have some existing problems:

– The system may not be secure.

– The system is not yet optimal.

– There are still some incomplete functions.

- There is just the Android version for this application (while we need to develop and deploy the web version and the iOS version)

VI CONCLUSION AND FUTURE WORKS

1. Conclusion

In conclusion, the purposes and objectives of the Online Academy for Weight Loss and Healthcare are mostly achieved. By providing a user-friendly interface on the web, the interaction between Customer, Personal Trainer/Expert and Administrator becomes more convenient. Communicating, giving advice and administration becomes simpler and more effective.

The main functions of the Online Admission Application and Interview System have already been implemented:

– Users can register and log into the system.

– Users can search and view information about exercises and nutrition advice.

– Administrator can read, create, update and delete information of Customers/Personal Trainer/Expert

2. Difficulties

Throughout the project, we have encountered the following obstacles:

Human

– Members of the group lack experience in application-oriented coding, thus coding style is not adequately solid. The limitation of knowledge and skill relating to app-building also set the group some difficulties when the members have to learn and do at the same time.

Time Constraint

– The project was carried out when the schedule at the university came at a busy time, which made it hard to concentrate and spend much time on the project.

Time Management

– The primary obstacle for us was the lack of time spent for the project.

– This lack of time dedicated to the project was because of poor time management. Moreover, there had been other time-consuming projects and labworks in other courses, which make the management harder.

3. Future Work

To further improve this application in the future, the following tasks need to be done:

– Fulfill all the functions.

– Support iOS and Web platforms.

– Create a Guideline for using the system.

- Develop AI recommendations.
- Add a discount function.
- Add payment method.
- Handle multi-request (E.g: Two users book the same seat at a time).
- Recommender system (E.g: generate showtime automatically).

Reference

[1] ReactNative Documentation. Referrend from https://reactnative.dev/
[2] Firebase Documentation. Referred from https://firebase.google.com/docs
[3] NodeJS Documentation. Referred from https://nodejs.org/en/docs/
[4] JSON Web Token documentation.. Referred from https://jwt.io/introduction//
[5] NoSQL documentation.. Referred from https://www.mongodb.com/nosql-explained