Software Requirements Specification

Version 2.0

<<Annotated Version>>

June 18, 2020

Gym Assistant app

***Muhammad Wazier***

***Suliman Alshami***

Submitted in partial fulfillment

Of the requirements of

CS 310 Software Engineering

# Table of Contents

[Table of Contents i](#_Toc77487619)

[List of Figures ii](#_Toc77487620)

[1.0. Introduction 1](#_Toc77487621)

[1.1. Purpose 1](#_Toc77487622)

[1.2. Scope of Project 1](#_Toc77487623)

[1.3. Glossary 2](#_Toc77487624)

[1.4. References 2](#_Toc77487625)

[1.5. Overview of Document 2](#_Toc77487626)

[2.0. Overall Description 4](#_Toc77487627)

[2.1 System Environment 4](#_Toc77487628)

[2.2 Functional Requirements Specification 5](#_Toc77487629)

2.2.1 User..............................................................................................................................4 2.2.2 Create aAccount......................................................................................................4 2.2.3 Login............................................................................................................................5 2.2.4 Submit Exercise Program ..........................................................................................5 2.2.5 View Existing Programs..............................................................................................6 2.2.6 Rate a Program..........................................................................................................6 2.2.7 Save a Program to Personal List ..............................................................................7 2.2.8 Edit Personal Program ..............................................................................................7 2.2.9 Send Message...........................................................................................................7

[2.3 User Characteristics 8](#_Toc77487648)

[2.4 Non-Functional Requirements 10](#_Toc77487649)

[3.0. Requirements Specification 11](#_Toc77487650)

[3.1 External Interface Requirements 11](#_Toc77487651)

[3.2 Functional Requirements 11](#_Toc77487652)

Use Cases..........................................................................................................................11

Use Case Index ................................................................................................................11

Use Case #1 – Create Account.........................................................................................13

Use Case #2 – Login.........................................................................................................14

Use Case #3 – Submit Program .....................................................................................15

Use Case #4 – View Program..........................................................................................16

Use Case #5 – Rate a Program ..........................................................................................17

Use Case #6 – Save Program..........................................................................................19

Use Case #7 – Edit Program...........................................................................................21

Use Case #8 – Send Message.........................................................................................23

[3.3 Detailed Non-Functional Requirements 24](#_Toc77487665)

Portability..................................................................................................................24

Reliability...................................................................................................................24

Ease of Use ..............................................................................................................25

Speed........................................................................................................................25

Size.............................................................................................................................25

Class Diagram..........................................................................................................26

# List of Figures

[Figure 1 - System Environment 4](#_Toc77487669)

[Figure 2 - User Use Cases 12](#_Toc77487671)

[Figure 3 - Class Diagram ……………26](#_Toc77487672)

# 1.0. Introduction

## 1.1. Purpose

The purpose of this document is to present a detailed description of the Gym Assistant application .Content includes research, requirements, use cases, data requirements, usability. This document outlines key technical details of the Gym Assistant application.

This document is intended for both the stakeholders and the developers of the system.

## 1.2. Scope

Gym Assistant is a community based health and fitness mobile application. It aims to offer a social environment for people interested in health and fitness. The application will be fueled by user generated content, users will be able to submit and view exercise programs within the application. All programs in the application will be submitted by users of the application, so if a user finds a certain program works really well for them or finds a useful workout on the internet, they can submit it to the application for other users to try out. Users will be able to rate programs submitted to the application, this will help ensure high quality content within the application. The main objectives of our project is to create an app which enables the user to effectively lose weight, maintain weight and gain muscle in a sustainable manner with clear and concise plans

## 1.3. Glossary

|  |  |
| --- | --- |
| **Term** | **Definition** |
| Software Requirements Specification | A document that completely describes all of the functions of a proposed system and the constraints under which it must operate. For example, this document. |
| Stakeholder | Any person with an interest in the project who is not a developer. |
| User | Any person who will use this app. |
| Google Firebase | A No SQL, real-time, online database used to permanently store application data. |
| GUI | (graphical user interface) is a system of interactive visual components for computer software. A GUI displays objects that convey information, and represent actions that can be taken by the user. The objects change color, size, or visibility when the user interacts with them |
| API | An application programming interface (API) is a computing interface which defines interactions between multiple software intermediaries. |
| IOS | iOS is a mobile operating system created and developed by Apple Inc. exclusively for its hardware. It is the operating system that presently powers many of the company's mobile devices, including the iPhone, and iPod Touch; it also powered the iPad prior to the introduction of iPadOS in 2019 |

## 1.4. References

IEEE. *IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications.* IEEE Computer Society, 1998.

## 1.5. Overview of Document

The next chapter, the Overall Description section, of this document gives an overview of the functionality of the product. It describes the informal requirements and is used to establish a context for the technical requirements specification in the next chapter.

The third chapter, Requirements Specification section, of this document is written primarily for the developers and describes in technical terms the details of the functionality of the product.

Both sections of the document describe the same software product in its entirety, but are intended for different audiences and thus use different language.

# 2.0. Overall Description

## 2.1 System Environment

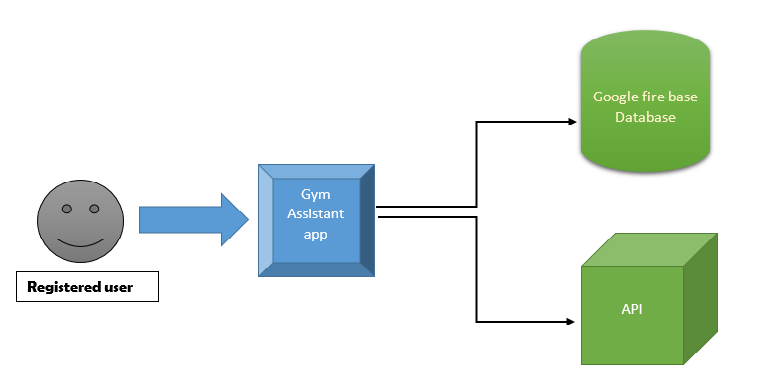


Figure 1 - System Environment

The application is designed to work on all devices that are running iOS version 8 or later. Currently this includes the iPhone.

The application will store all data in a Google Firebase database. The application will require an online connection at all times to allow for user sign on, user communication, and access to online user submitted content.

## 2.2 Assumptions & Dependencies

2.2.1 If no exercise routines are created by users, there will be no routines listed

in the application for users.

2.2.2 If no internet connection is available, users will be unable to log in or

access any functionality of the software.

2.2.3 Unregistered users will have no access to the software.

2.2.4 Users will only be permitted to register and gain access to the software

once they agree to the terms of service agreement.

## 2.**3** Functional Requirements

2.3.1 User

2.3.1.1 All users of the software shall have the ability to create an account

which is used to store user data and tie user actions to a user alias.

((User registration and login shall be mandatory)).

2.3.2 Create an Account

2.3.2.1 The system should provide the user with an easy to use GUI to facilitate

their creating an account.

2.3.2.2 The system shall ask for an email address and password.

2.3.2.3 The system shall notify the user if incorrect characters are used in the

email or password fields.

2.3.2.4 The system should notify the user if their email has already been used.

2.3.2.5 The system should notify the user if any required fields are left empty.

2.3.2.6 The system should not allow the user to create weak or unsecure

passwords.

2.3.2.6.1 The system should explain how the submitted password is

unsecure.

2.3.2.7 The system should prevent the user from completing registration if the

terms of service has not been agreed to.

2.3.3 Login

2.3.3.1 The system should provide a user friendly GUI to allow the user to login

when the application launches.

2.3.3.2 The system should prompt the user for their email address and

password.

2.3.3.2.1 The system should notify the user if submitted information is

incorrect.

2.3.4 Submit Exercise Program

2.3.4.1 The system should provide an intuitive UI for logged in users to allow

them to submit their exercise routine to the application.

2.3.4.1.1 The system should prevent the user submitting a blank or

empty routine.

2.3.4.2 The system shall add successfully submitted routines to the Google

Firebase Database.

2.3.4.3 The system should display user submitted routines from the Google

Firebase database in the appropriate section of the application.

2.3.5 View Existing Programs

2.3.5.1 The system should provide intuitive and user friendly navigation to allow

users to locate the current list of user submitted exercise routines.

2.3.5.2 Once selected, the system shall retrieve all user submitted programs

from the Google Firebase database and display them to the user.

2.3.5.3 The system shall display full details of an exercise routine once one is

selected by the user.

2.3.5.4 The system should allow quick and easy navigation between different

routines in the list.

2.3.6 Rate a Program

2.3.6.1 The system should provide a button to rate a program. Programs can be

rated up or down based on the level of success the user has with them

2.3.6.1.1 Programs will be displayed based on ratings. Lower rated

programs will be pushed to the bottom of the list before being removed.

2.3.7 Save a Program to Personal List

2.3.7.1 The system should provide a button to save a program from the public

list of user submitted programs to their own personal list.

2.3.7.1.1 Programs saved to a personal list are not visible to other

users.

2.3.8 Edit Personal Program

2.3.8.1 The system should provide an intuitive and user friendly UI to allow the

user to view and manage their personal list of workout programs.

2.3.8.2 The system should allow editing of programs saved to a user’s personal

list.

2.3.8.2.1 Multiple edits can be made and all changes must be

saved in real time

2.3.9 Send Message

2.3.9.1 The system shall provide an interface for sending messages between

users.

2.3.9.2 Messages should be sent in real time and have no delays.

***2.4* User Characteristics**

**Unregistered User**

Unregistered users will not be able to gain access to the application or use any of its features. The whole focus of the application is on user submitted content and social interactions, none of this is possible without an account alias to tie a user to. Additionally, the services offered by the application will be behind a ToS (Terms of Service) agreement, this model was chosen as users will be following programs submitted by other users completely at their own risk. No liability is accepted by the developer of the application or any parties in connection with the developer.

**Registered User**

Registered users will gain full, non admin access to the application. Once a user registers their details and agrees to the ToS they will be able to post workout routines, view workout routines, rate workout routines, save routines to their personal list for edits, chat with other users and use the gyms nearby feature.

**Administrator**

The administrator will maintain the data in the application behind the scenes. The administrator will deal with managing user accounts if any bans are levied against a user. They will also manage workout program data based on user feedback. If a program receives enough negative feedback it will need to be removed from the application by the administrator. The administrator will also carry out day to day housekeeping within the application and ensure the quality of the application content.

## 2.5 Non-Functional Requirements

Portability , Reliability , Ease of Use , Speed , Size

# 3.0 Requirements Specification

## 3.1 External Interface Requirements

## *User Interface*

## The user interface shall offer the user a logical representation of what the software is asking the user to do. Dropdown menus and buttons should be used where possible to aid the user. Input hints shall be used to aid the user when entering data.

## The application should have its logo present on each screen once a logo has been designed.

## A user friendly color scheme should be chosen , UI design should be carried out with visually impaired and color blind users in mind.

## The UI should have well defined constraints to ensure that the software displays correctly on the screens of all compatible devices. The UI should display in both portrait and landscape.

## The GUI should have continuity, all screens should have the same design and layouts should be consistent.

## *Hardware Interfaces*

The system shall be operated with a compatible Apple device using the

devices touch Screen && virtual keyboard .

## *API Interfaces*

The software must store user information and user submitted content in a

Google Firebase database using Cocoa Pod files to achieve

communications between the database and the application.

The software must show mapping information around the user’s current

location using the Google maps API.

## 3.2 Functional Requirements

**3.2.1 Use Cases**

**3.2.1.1 Use Case Index**

### 

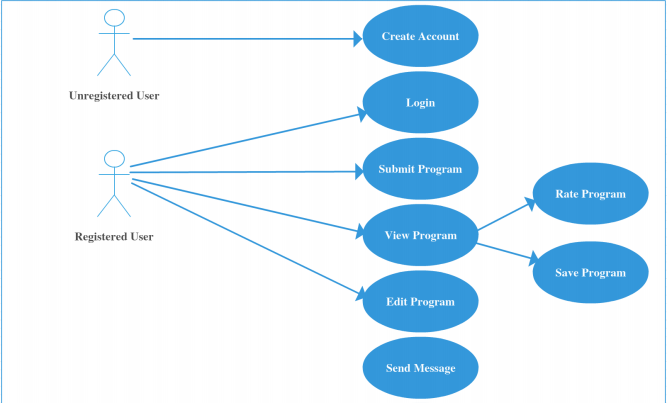
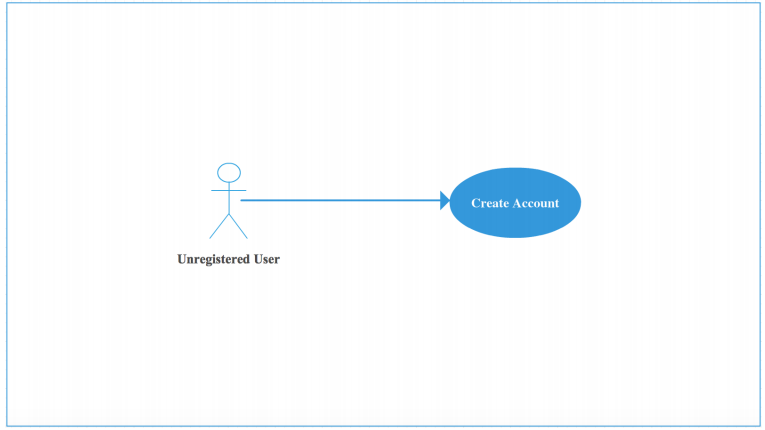
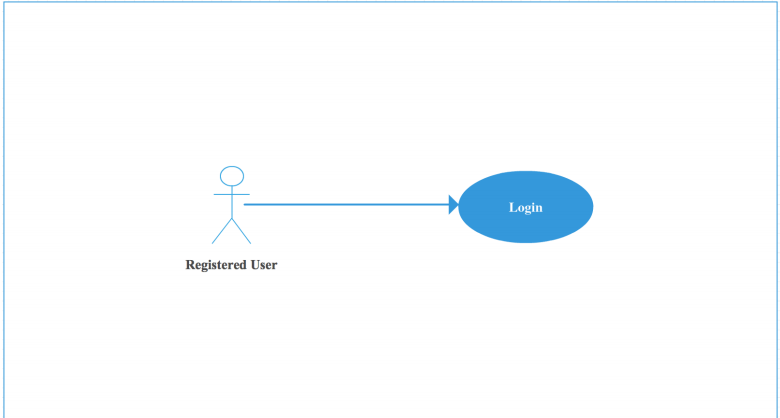


Figure 2 Use Cases Diagram

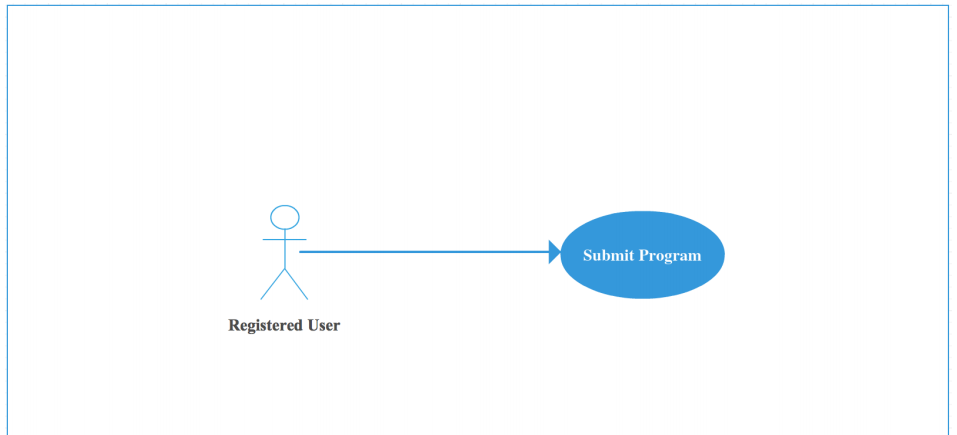
|  |  |
| --- | --- |
| Use Case Element | Description |
| Use Case Number | 1 |
| Application | Gym Assistant |
| Use Case Name | Create Account |
| Use Case Description | A user starts the application for the first time and is prompted to create an account. |
| Primary Actor | Unregistered User |
| Precondition | The application is started by a new user who does not have an account. |
| Trigger | The unregistered user taps the register button |
| Basic Flow | 1. This use case starts when a new user launches the application for the first time and opts to create an account. 2. The user inputs their email and password into the allotted text fields. 3. The application notifies the user that the account has been created and grants them access to the application. |
| Alternate Flows | 1. The user does not want to create an account and chooses to exit the application at that point. 2. The user enters invalid characters into the text field or leaves them blank. The system notifies the user of their error. |
| Termination | The flow is terminated once the user is successfully registered and their details stored in the database. |
| Post Condition | The use is returned to the Login page and the system enters a wait state. |



|  |  |
| --- | --- |
| Use Case Element | Description |
| Use Case Number | 2 |
| Application | Gym Assistant |
| Use Case Name | Login |
| Use Case Description | A registered user starts the application and is prompted to login using their personal details |
| Primary Actor | Registered User |
| Precondition | The application is started by a registered user |
| Trigger | A registered user starts the application |
| Basic Flow | 1. This use case starts when a registered user runs the application and is presented with a login GUI page. 2. The user enters their username and password and is granted access to the system. |
| Alternate Flows | 1. The user inputs an incorrect username or password and is notified by the system |
| Termination | The flow is terminated when the user is validated by the system and logged in to the application |
| Post Condition | The user is brought to the application Home page and the system enters a wait state. |



|  |  |
| --- | --- |
| Use Case Element | Description |
| Use Case Number | 3 |
| Application | Gym Assistant |
| Use Case Name | Submit Program |
| Use Case Description | A registered user creates and submits an exercise program to the application. |
| Primary Actor | Registered User |
| Precondition | A registered user must be logged in. |
| Trigger | A registered user logs in and taps the Submit Program button |
| Basic Flow | 1. This use case begins when a logged in user clicks the Submit Program Button. 2. The user proceeds to enter the program details into the supplied fields. 3. Once all details have been added the user then clicks the Submit button to add their program to the application |
| Alternate Flows | 1. The user enters invalid information or leaves some or all fields empty and the system informs the user of their error |
| Termination | The flow is terminated when the user successfully submits their program to the application and it is saved to the database |
| Post Condition | The user is returned to the All Programs page and the system enters a wait state. |

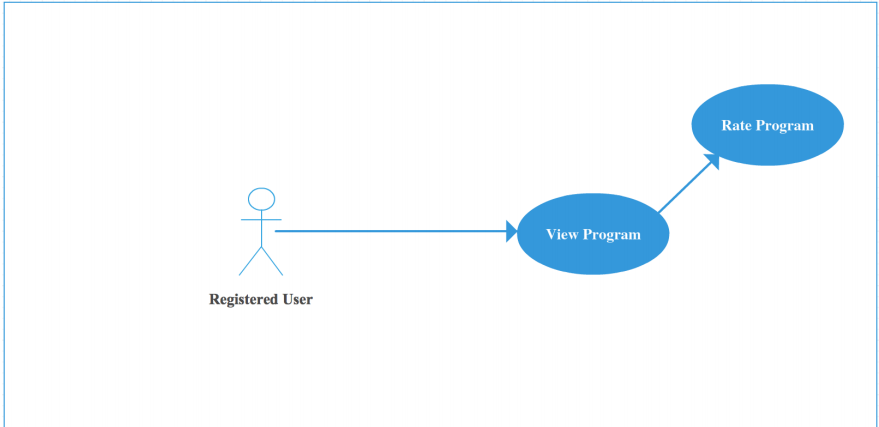


|  |  |
| --- | --- |
| Use Case Element | Description |
| Use Case Number | 4 |
| Application | Gym Assistant |
| Use Case Name | View Program |
| Use Case Description | A registered user logs in and navigates to the All Programs section of the application. |
| Primary Actor | Registered User |
| Precondition | A registered user must be logged in. Programs must exist in the application |
| Trigger | A registered user logs in and taps the All Programs button. |
| Basic Flow | 1. This use case begins when a logged in user clicks the All Programs button. 2. The user is then presented with a list of all the programs hosted by the application. |
| Alternate Flows | 1. The application has no user submitted programs and has nothing to display to the user. The system notifies the user of this and returns them to the home page |
| Termination | The flow is terminated when the user is presented with a list of all programs or is informed that there are no programs. |
| Post Condition | Depending on whether there are programs to show, the user is either brought to the All Programs page or the Home page. |

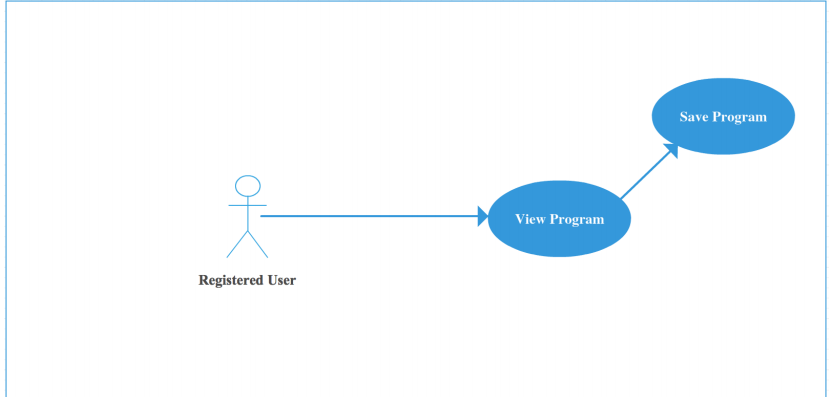
## 

## 

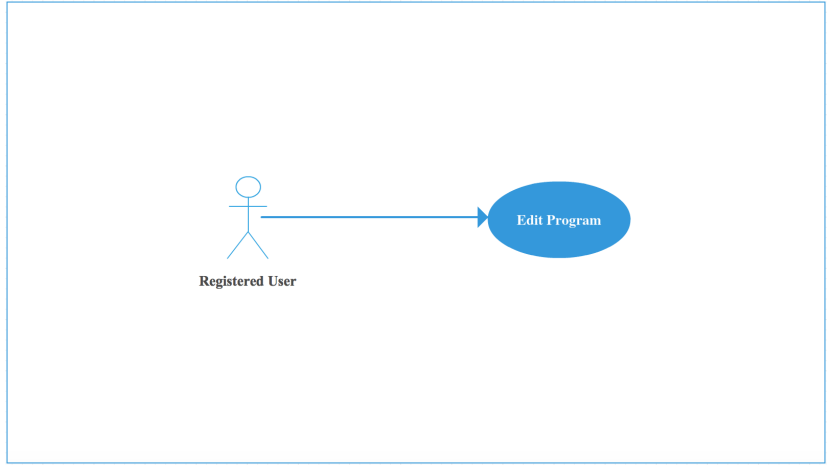
|  |  |
| --- | --- |
| Use Case Element | Description |
| Use Case Number | 5 |
| Application | Gym Assistant |
| Use Case Name | Rate Program |
| Use Case Description | A registered user logs in and navigates to the All Programs section of the application and applies a rating to a listed program |
| Primary Actor | Registered User |
| Precondition | A registered user must be logged in and on the All Programs page. Programs must exist in the application. |
| Trigger | A registered user logs in and taps the All Programs button then the Rate button. |
| Basic Flow | 1. This use case begins when a logged in user clicks the All Programs button and then the Rate Button. 2. The user is then presented with a list of all the programs hosted by the application. 3. The user then clicks the Rate button beside one of the programs to apply their positive or negative rating |
| Alternate Flows | 1. The application has no user submitted programs and has nothing to display to the user. The system notifies the user of this and returns them to the home page |
| Termination | The flow is terminated when the users rating is applied to the program and is saved to the database |
| Post Condition | The user remains on the All Programs page and the system enters a wait state. |



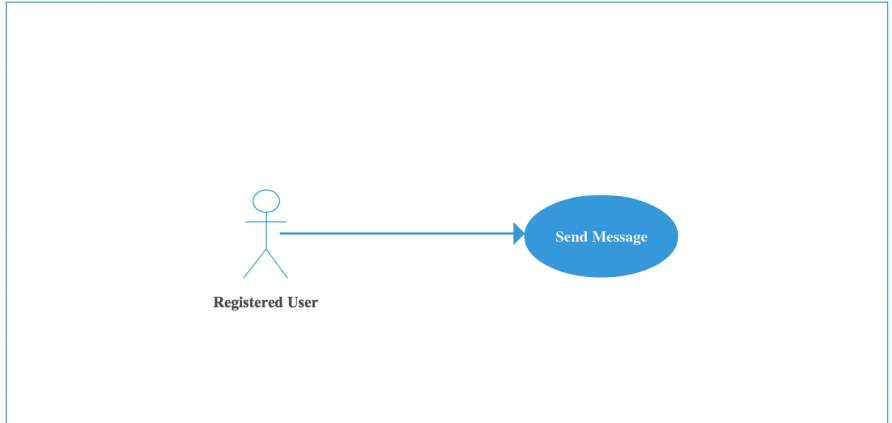
|  |  |
| --- | --- |
| Use Case Element | Description |
| Use Case Number | 6 |
| Application | Gym Assistant |
| Use Case Name | Save Program |
| Use Case Description | A registered user logs in and navigates to the All Programs section of the application and taps the Save Program button |
| Primary Actor | Registered User |
| Precondition | A registered user must be logged in and on the All Programs page. Programs must exist in the application. |
| Trigger | A registered user logs in and taps the All Programs button then the Save Program button. |
| Basic Flow | 1. This use case begins when a logged in user clicks the All Programs button and then the Save Program button. 2. When the user taps the Save Program button beside one of the programs it is saved to their personal list. |
| Alternate Flows | 1. The application has no user submitted programs and has nothing to display to the user. The system notifies the user of this and returns them to the home page. |
| Termination | The flow is terminated when the program is successfully saved to the user’s personal list and the database is updated accordingly. |
| Post Condition | The user is returned to the My Programs page and the system enters a wait state |



|  |  |
| --- | --- |
| Use Case Element | Description |
| Use Case Number | 7 |
| Application | Gym Assistant |
| Use Case Name | Edit Program |
| Use Case Description | A registered user logs in and navigates to the My Programs section of the application and taps the Edit Program button to make changes to their program. |
| Primary Actor | Registered User |
| Precondition | A registered user must be logged in and on the My Programs page. The user must have at least one program saved to their list. |
| Trigger | A registered user logs in and taps the My Programs button then the Edit Program button |
| Basic Flow | 1. This use case begins when a logged in user taps the My Programs button and then the Edit Program button.2. When the user taps the Edit Program button beside one of the programs they are presented with a GUI with edit fields allowing them to edit the program |
| Alternate Flows | 1. The user has no programs saved and the system has nothing to display to the user. The system notifies the user of this and returns them to the home page. |
| Termination | The flow is terminated when the user’s program is successfully edited and the database is updated accordingly |
| Post Condition | The user is returned to the My Programs page and the system enters a wait state. |



|  |  |
| --- | --- |
| Use Case Element | Description |
| Use Case Number | 8 |
| Application | Gym Assistant |
| Use Case Name | Send Message |
| Use Case Description | A registered user logs in and navigates to the Messages section of the application and taps the Send Message button and selects a user to send a message to. |
| Primary Actor | Registered User |
| Precondition | A registered user must be logged in and on the Messages page. The user must know the alias of the other user they wish to message. |
| Trigger | A registered user logs in and taps the Messages button and then sends a message to another user |
| Basic Flow | 1. This use case begins when a logged in user taps the Messages button and selects a user to message.2. Once a user is selected the user is presented with a UI enabling them to type their message.3. The message is sent when the user taps Send. |
| Alternate Flows | 1. The user enters an incorrect user alias and the system notifies the user of this and returns them to the messages page |
| Termination | The flow is terminated when the user’s message is sent to the recipient and the database is updated accordingly. |
| Post Condition | The user is returned to the Messages page and the system enters a wait state. |



## 3.3 Detailed Non-Functional Requirements

# Portability:

# **the application will run natively on any Apple device running iOS 8 or later.**

# Reliability:

# **The system should be extremely reliable and have an approximate up time of 99.999%.**

# **In the event of a crash or any other error, the System should inform the user of any problems and gracefully terminate.**

# Ease of Use:

# **The application should be user friendly and intuitive to use. GUIs should make their functions clear**

# Speed:

# **The application should open and be ready to use within 10 seconds of being selected && be quick to use with interaction**

# Size:

# **The size of the software in relation to storage media should be no larger than 250MB.**

**Class Diagram**

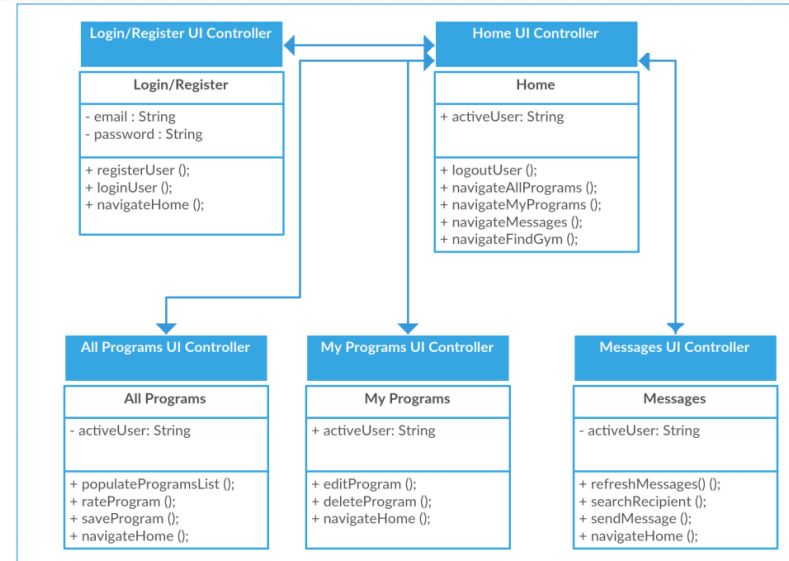
****

Figure 3 Class Diagram