Software Requirements Specification

for

SickGoWhere

Version 1.1 approved

Prepared by Team 1

Ernest Ang Cheng HAN

Goh Tse Yinn, Sheryl

Ernest Tan Yang Heng

David Tay Ang Peng

Heng Jiu Xiao

Kim Heonjung

Nanyang Technological University
School of Computer Science and Engineering

09 Apr 2021

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Kim Heonjung	09/04/21	Added ‘Functional Requirements’	1.1

1. Introduction

1.1 Purpose

The purpose of this SRS document is to provide the specification of our mobile application, SickGoWhere. This document covers the overall description of our application and the overview of the software system design including the functional requirements, non-functional requirements and user interfaces.

1.2 Document Conventions

All requirements are documented in detail in this document. Priorities for the requirements will be indicated and they are not implied by the order of requirements.

In this document, we will use ‘healthcare facilities’ and ‘clinics’ interchangeably.

1.3 Intended Audience and Reading Suggestions

This document is intended for developers, testers, project managers, and the users of SickGoWhere. It includes a scope of project and a detailed description of the project, but is not limited to user-interface design, UML diagrams, and other analysis models.

Developers, project managers and testers are recommended to read the entirety of this document to get a clear understanding of the application and its requirements, so they will be able to implement appropriate testing and get an easier way to update and develop the application. Users of SickGoWhere are recommended to read the overall description and system features of the application to ensure the application serves their purpose.

1.4 Product Scope

The objective of SickGoWhere is for users to allow for greater accessibility to healthcare. It targets anyone without a regular family doctor that they consult with, which accounts for more than 50% of people in Singapore. Users will be able to get the five nearest clinics and their directions based on their current location. If users have a specific clinic in mind, they can also search for it by name and get the directions as well. Additionally, users will be able to book appointments at the selected clinics through the use of our application.

1.5 References

- *https://developer.android.com/*
- *https://reactnative.dev/docs/environment-setup*
- *https://docs.expo.io/workflow/android-studio-emulator/*
- *https://www.chas.sg/*
- *https://console.cloud.google.com/apis/dashboard*
- *https://chrome.google.com/webstore/detail/rester/eejfoncpjfgmeleakejdcanedmefagga*
- *https://dev.to/gabrieleloy/testing-react-native-with-testing-library-1caa*

2. Overall Description

2.1 Product Perspective

SickGoWhere is a newly developed mobile application based on requirements by the ZEA to elicit innovative applications by using government data. It allows:

- Government Data to get clinics name and their location
- Enabling users to search the healthcare facilities based on their location
- Provide the directions from users' location to healthcare facilities with three transportation options - driving, walking and public transport
- Users can select the fastest option by comparing the travelling time between various clinics
- Users can book an appointment to the healthcare facilities they want and edit the appointment details
- Provide real-time updates on the booking slot so that users will be informed the available time slots
- Confirmation dialog to users while they make, edit and cancel an appointment.

To get a more detailed view of the product, please refer to the diagrams in Appendix B.

2.2 Product Functions

This section provides the summary of the major function of the product that SickGoWhere must perform and must allow the user to perform.

2.2.1 Major function for Users

- Users must be able to input information for account registration
- Users must be able to input information for logging in into system
- Users must be able to enter clinic name in the search bar on the main page
- Users must be able to schedule an appointment if they do not already have an appointment

- Users must be able to view details of their scheduled appointment
- Users must be able to edit the details of their scheduled appointment
- Users must be able to delete their scheduled appointment
- Users must be able to select how the top 5 clinics are ranked
- Users must be able to click on the clinics to access the shortest route to the selected clinic
- Users must be able to click on the booking button to create an appointment for the selected clinic.

2.2.2 Major functions of System

- System must be able to store registration information
- System must be able to verify user input login credentials
- System must be able to retrieve list of clinics
- System must be able to display list of clinics that matches the search criteria
- System must be able to allow users to schedule appointments, edit their scheduled appointments.
- System must be able to store all details of scheduled appointments
- System must be able to show details of scheduled appointments
- System must be able to compute travel time for the clinics
- System must be able to display the 5 clinics with the least travel time and their respective travel times
- System must show the shortest route to the selected clinic on the map.
- System must show the various routes to the selected clinic based on three types of transportation - driving, walking and public transport

2.3 User Classes and Characteristics

User Class		
Users	Frequency of Use	High
	Level of Access	User Level
	Subset of Product Functions	• Allowed access to all user-level functions. (refer to section 2.2.1)

2.4 Operating Environment

SickGoWhere is a mobile application suitable for use on Android OS. The application will be developed on the React-Native using javascript.

The operating environment of the application is as follows:

- Operating System
    - Android
- Technology Stack
    - Node.js
    - React-Native
    - Android SDK
- Database
- MongoDB Atlas

2.5 Design and Implementation Constraints

- SickGoWhere shall operate on Android 11.0 or higher operating system.
- The storage of the hosting database will only be available to 512mb.

2.6 User Documentation

The user documentation including use case diagram, use case table and dialog map will be delivered along with this document in Appendix B and C.

2.7 Assumptions and Dependencies

SickGoWhere is developed on Android API 30 as the stable version of the application requires it to be installed on the device. It is assumed that the user's device is connected to the Internet through Wi-Fi and Mobile Data. It is also assumed the user's location to be known with GPS is turned on the user's device and is accessible by the application as the user has to give the application permission to access the phone's location services.

The system depends on the data from publicly available government data from data.gov.sg, so our business logic heavily depends on government policies. If any rules and regulations are changed by the Singapore government, the application will have to be updated.

Also, as the distances and routes are dependent on the Google Maps API, the users must have Google Maps installed (installed by default in most phones) and have signed in using their google account.

3. External Interface Requirements

3.1 User Interfaces

The user interface of our application is designed with respect to Schneiderman's Eight Golden Rules.

3.1.1 Strive for Consistency

- The main interface of the application provides a standardized visual layout and style throughout the screens
- The application follows a consistent color scheme throughout the entire user interface
- The booking / editing UI for appointments are similar as they are similar operations
- The steps taken to book an appointment is the same even when the users are booking through different routes such as from the dashboard or from the maps screen

3.1.2 Enable frequent users to use shortcuts

- The system saves the user login information once they log into the system with the 'Remember me' button checked
- The system also saves the user's permissions so that once the user have given permission to access their location, they will not be prompted again unless they manually revoke the permissions given

3.1.3 Offer informative feedback

- The user views the loading screen while the application is loading
- Red markers are placed on the maps to give users a grasp of the locations of the clinics relative to their position which is marked with a blue marker
- When searching clinics, the clinics that fulfils the search phrase are refreshed after every letter is entered to show users what clinics are still available

3.1.4 Design dialog to yield closure

- Confirmation pages are provided when the user books / edit / delete an appointment to confirm that they have accomplished said operation

3.1.5 Offer simple error handling

- Users are prompted to enter the correct password and usernames when they have inputted erroneous or no information
- Timings that are unavailable are removed from the dropdown options so that users cannot choose them
- When searching clinics, the clinics that fulfils the search phrase are refreshed after every letter so that users know when they have made a wrong input

3.1.6 Permit easy reversal of actions

- There is an undo button to allow easy traversal to the previous page

3.1.7 Support internal locus of control

- There are no spontaneous operations that occur naturally and all operations are triggered by the user's actions

3.1.8 Reduce short-term memory to load

- Information are being passed to the next screen without having the user have to remember it such as when users have selected a clinic to book from the maps screen, the clinic is auto-inputted in the booking screen so that users need to search for the selected clinic to book it

3.2 Hardware Interfaces

Hardware Requirements:

- Android devices with Android 11.0+
- RAM: 4GB+
- Device supports cellular networks
- Device supports location services

3.3 Software Interfaces

Software used	Category	Description
React-Native	Mobile Application Framework	React-native is chosen for our development due to its strong performance in mobile environments. While developing, we can immediately see the result of the latest code.
MongoDB Atlas	Database	MongoDB Atlas is a document-based open source database, which we used to store the records on users, clinics and appointments.
Android 11.0	Operating System	Android 11.0 was chosen for its currency and its wide range of developer features and options

3.4 Communications Interfaces

In this application, it requires the internet connection to load the clinic's data and users' data including their login information and appointment details. Also, it gets information from GoogleMap API. These communications are done through the Hypertext Transfer Protocol(HTTP).

4. System Features

4.1 User Registration

4.1.1 Description and Priority

The application allows users to register for an account and requests users to input their information. The system will save the information of the user in the database. It has a high priority as it requires to login with an account to book an appointment.

4.1.2 Stimulus/Response Sequences

	User action	System Response
1	Click the 'Login' text button	The login page is displayed
2	Click the 'Sign Up' text button	The registration page is displayed
3	Enter the personal information	The user inputs are displayed.
4	Click the 'Create' button	● User account is created ● The system saves the information into the database ● The main page is displayed

4.1.3 Functional Requirements

1. User must be able to input information for account registration
   - 1.1. User must be able to enter name
   - 1.2. User must be able to enter email address
   - 1.3. User must be able to enter password
   - 1.4. User must be able to enter password confirmation
2. System must verify whether email address already exists in system
3. System must check for password validity
   - 3.1. System must check that password and password confirmation is identical
   - 3.2. System must check that password is at least 8 characters long
   - 3.3. System must check that password is at most 15 characters long

4. System must be able to encrypt password
5. System must store registered information
6. System must inform the user that the registration is successful
7. System must inform the user that the registration is unsuccessful
8. System must be able to ensure all field are entered

4.2 User Login

4.2.1 Description and Priority

The application allows users to login into the system. It has a high priority as it requires to login with an account to book an appointment

4.2.2 Stimulus/Response Sequences

	User action	System Response
1	Click the 'Login' text button	The login page is displayed
2	Enter the email address and password	The user inputs are displayed
3	Check the checkbox 'Remember me'	System saves the login information
4	Click the 'Login' Text button	• System verifies if email address exists in the database • System verifies if password is correct for the tagged email address • The main page is displayed

4.2.3 Functional Requirements

1. User must be able to input information for logging in into system
   1.1. User must be able to enter email address
   1.2. User must be able to enter password
2. System must verify user input login credentials
   2.1. System must check that email exists in system
   2.2. System must retrieve the password tagged to the email address from system
   2.3. System must be able to compare input credentials with retrieved mobile number and password
3. System must inform user on whether login is successful

3.1. System must be able to inform the user that the login is successful if input credentials match with system

3.2. System must inform the user that login has failed if input credentials does not match with system

4.3 Search for clinic by name

4.3.1 Description and Priority

The application returns a list of clinics that names matches the user's search condition and users can query for the route to the clinic and have estimated travelling times via foot, car and public transport

4.3.2 Stimulus/Response Sequences

	User action	System Response
1	Click the magnifying glass on the search bar	The clinic search list page is displayed
2	Enter parts of the clinics name in the search bar	Clinics that satisfies the search phrase are shown
3	Click on the clinic desired	The route to the clinic is displayed with the estimated travelling time in the 3 modes displayed

4.3.3 Functional Requirements

1. User must be able to enter clinic name in the search bar on the main page
2. System must be able to retrieve list of clinics from Google Maps API
3. System must be able to display list of clinics that matches the search criteria
4. System must be able to show travelling time for each clinic on the list
   4.1. System must be able to retrieve the user's current location using the Geolocation API
   4.2. System must be able to retrieve the shortest travelling time required to travel from user's location to clinics using Google Maps API
   4.3. System must display the clinics and their respective travelling time

4.4 Appointment System

4.4.1 Description and Priority

The application allows users to book / edit / delete appointments to a specific clinic

4.4.2 Stimulus/Response Sequences

	User action	System Response
1	Click ‘Book Appointment’ button	The appointment page is displayed
2	Enter a part of the clinic’s name	List of clinics satisfying the condition is displayed
3	Click on desired clinic	Desired clinic is selected and displayed in the bar
4	Click on date	Calendar with unavailable dates greyed out is displayed
5	Click on desired date	Desired date is selected and displayed and calendar closes
6	Click on time	Dropdown bar with available timings are displayed
7	Click on desired time	Dropdown bar closes and desired timing is selected
8	Click on book	Booking is made and the appointment confirmation screen is displayed with the appointment details

4.4.3 Functional Requirements

1. User must be able to schedule an appointment if they do not already have an appointment
   1.1. User must be able to choose appointment date
   1.2. User must be able to choose appointment time
2. User must be able to view details of their scheduled appointment
   2.1. User must be able to view date of appointment
   2.2. User must be able to view time of appointment
   2.3. User must be able to view location of appointment
3. User must be able to edit the details of their scheduled appointment

3.1. User must be able to change date
3.2. User must be able to change time
4. User must be able to delete their scheduled appointment
5. System must be able to allow users to schedule appointment
5.1. System must be able to show the available time slots only
5.2. System must allow user to choose appointment date
5.3. System must allow user to choose appointment time
6. System must be able to allow users to edit their scheduled appointment
6.1. System must be able to show the available time slots only
6.2. System must allow user to choose appointment date
6.3. System must allow user to choose appointment time
7. System must be able to store all details of scheduled appointments
8. System must be able to show details of scheduled appointments
8.1. System must be able to show date
8.2. System must be able to show time
8.3. System must be able to show location

4.5 Find nearest clinics

4.5.1 Description and Priority

The application allows users to find the nearest clinics to them and displays the relative position as well as distance to each of the five clinics

4.5.2 Stimulus/Response Sequences

	User action	System Response
1	Click the 'Find Nearest Clinics' button	The map with markers of the user's location and 5 nearest clinics are displayed alongside a list of the clinics and their respective distance
2	User selects the desired clinic	The route to the selected clinic as well as the estimated travelling time for the 3 modes are displayed

4.5.3 Functional Requirements

1. User must be able to select how the top 5 clinics ranked by travelling time
2. System must be able to retrieve the travelling time for the clinics
2.1. System must be able to retrieve list of clinics using the API

2.2. System must be able to retrieve the user's current location using the API
2.3. System must be able to retrieve the shortest travelling time required to travel from user's location to clinics using API

3. System must display the 5 clinics with least travel time and their respective travel times
   3.1. System must filter out the top 5 clinics with least travel time
   3.2. System must display the names of the clinic
   3.3. System must display the travel time to the respective clinics
4. User must be able to click on the clinics to access the shortest route to the selected clinic

4.6 Show the shortest route to the selected clinic

4.6.1 Description and Priority

The application allows users to find the shortest route to a selected clinic for three different modes of transportation - driving, public transit and walking

4.6.2 Stimulus/Response Sequences

	User action	System Response
1	User selects clinic either via 'Find Nearest Clinics' or 'Search'	Shortest route to clinic is shown with the estimated travelling time for the 3 modes
3	User clicks on desired mode of transportation	Clinic location, user location and mode of transport is passed to Google Maps which displays the detailed instructions on the route

4.6.3 Functional Requirements

1. System must detect the user's current position using API
2. System must show the shortest route to the selected clinic on the map
3. System must show the various routes to the selected clinic based on type of transport
   3.1. System must be able to show various types of transportation mode below the map
   3.2. System must show the estimated time of each route
4. User must be able to click on the booking button to create an appointment for the selected clinic

5. Other Nonfunctional Requirements

5.1 Performance Requirements

5.1.1 The system should take no more than 3 seconds to load after every action by the user
5.1.2 The system must offer informative feedback
    5.1.2.1 System must display an error message that allows the users to know what went wrong when an error occurs
5.1.3 The system must allow easy reversal of the user's actions

5.2 Usability Requirements

5.2.1 The system must strive to be universally usable
    5.2.1.1 The system must support different languages
5.2.2 The system must try to reduce memory load
    5.2.2.1 The UI must be kept simple while allowing users to find desired clinic
    5.2.2.2 The users require minimal instructions to use the system
5.2.3 The system must strive for consistency
    5.2.3.1 A consistent series of actions to perform similar types of tasks such as booking a consultation through the three different access points
    5.2.3.2 A consistent design of UI must be achieved (Colors, Buttons, Fonts etc.)

5.3 Security Requirements

5.3.1 The system must ensure password is secure
    5.3.1.1 To mask the input password characters during login
    5.3.1.2 To mask password inputs during password creation at registration
    5.3.1.3 To ensure password meets stringent criteria (i.e minimum length 8, one special character etc.)
5.3.2 The system must ensure accounts' information is inaccessible by unauthorised personnel
    5.3.2.1 The account information is only accessible by the system administrators
    5.3.2.2 The account information is encrypted in database

5.4 Reliability Requirements

5.4.1 The system must not crash when user opens the application
5.4.2 The system must maintain information consistency
5.4.3 The system must be able to restart to full functionality within 10 minutes of crashing
5.4.4 The system must be able to work at any time of the day

5.4.5 The system must not lose any information
- 5.4.5.1 The system must not lose appointment information
- 5.4.5.2 The system must not lose user information

5.5 Supportability Requirements

5.5.1 The system must have ease of maintainability
- 5.5.1.1 Object-Oriented Design to ensure that debugging or maintenance is affordable and efficient
- 5.5.1.2 Test-cases must be comprehensive but succinct

5.5.2 The system must work across different platforms
- 5.5.2.1 Built on hybrid app development so that it can work across different mobile operating systems

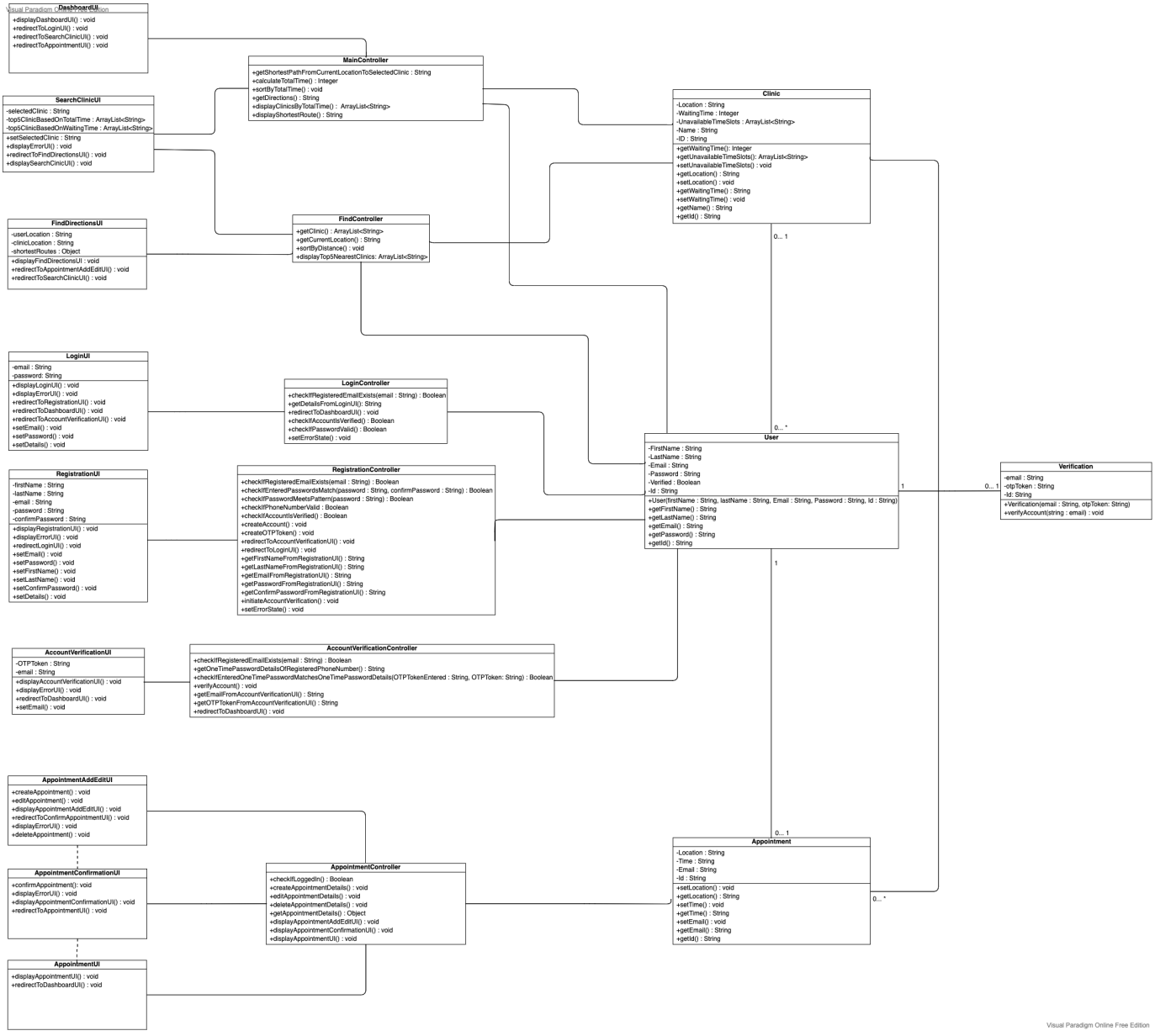
5.5.3 The system must be portable
- 5.5.3.1 The app can be migrated from legacy system to a new platform in accordance to technology changes/updates

Appendix A: Data Dictionary

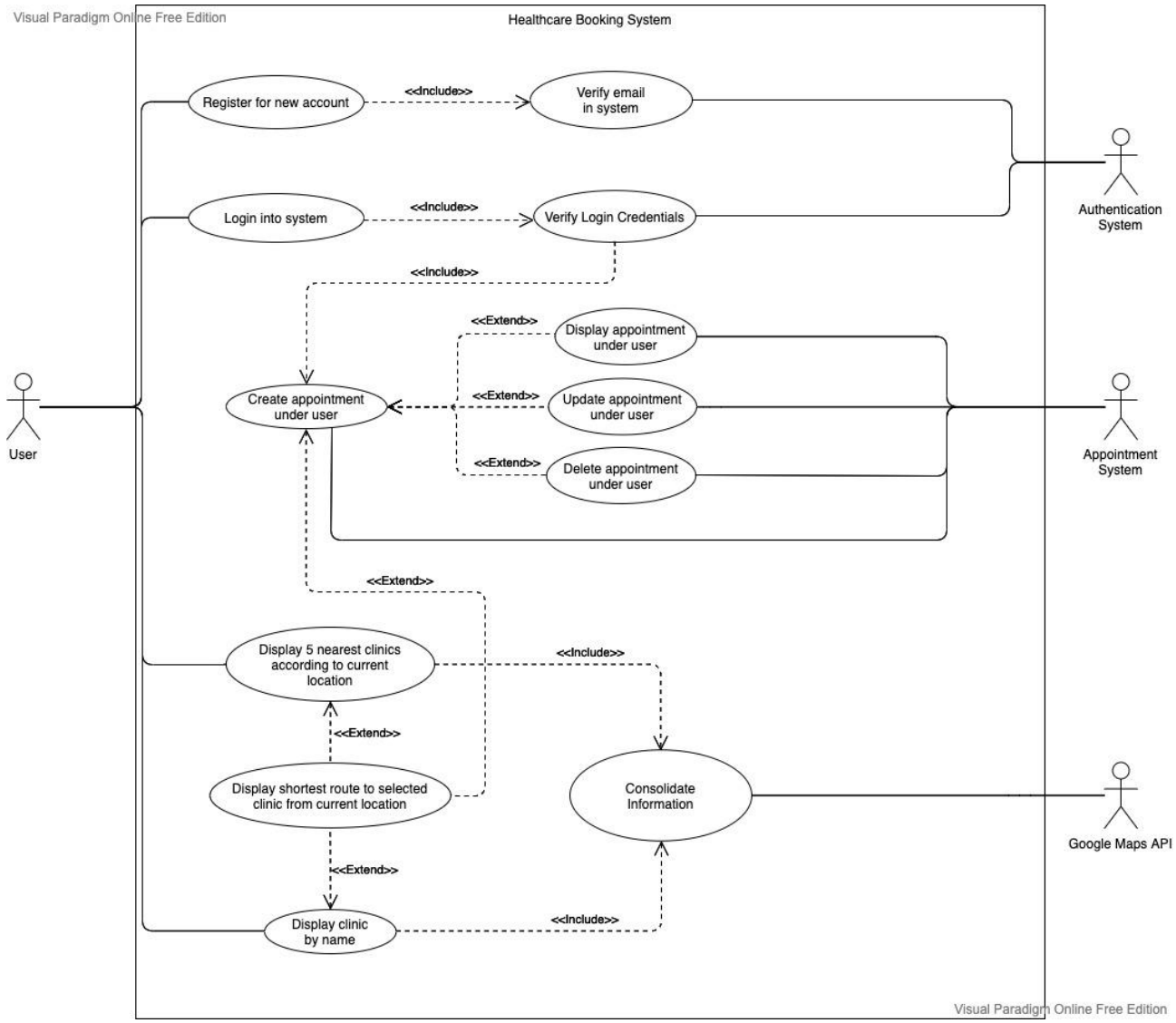
Term	**Definition**
User	User refers to a person who is using the application to search for clinics and schedule appointments
System	System refers to the mobile application
Appointment	Appointment refers to a medical consultation at selected clinic
Search	Search is a feature that allows users to find their desired clinic based on certain filters
Travel Time	Travel Time refers to the amount of time taken for the user to travel from their current location to the selected clinic
GPS	Global Positioning System which is used to pinpoint the user's current location
Geolocation	Geolocation refers to the exact geographic location of the user, found using GPS

Appendix B: Analysis Models

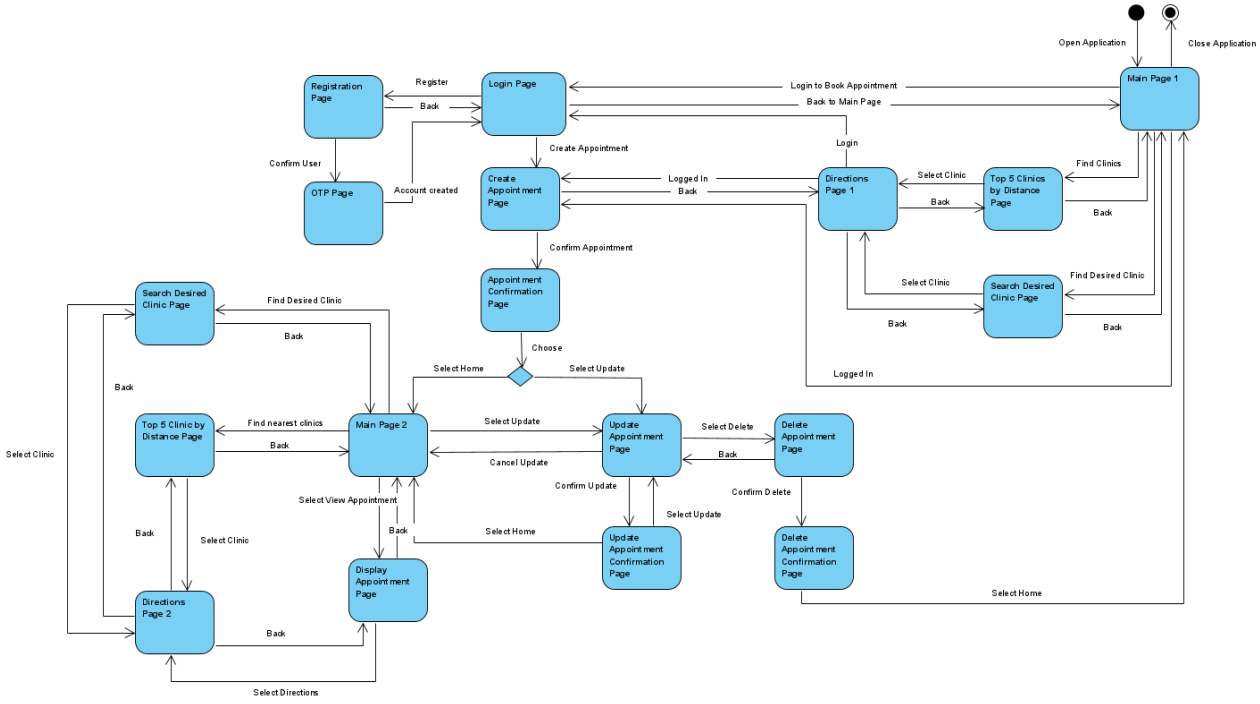
Class Diagram



Use-Case Diagram



Dialog Map



Appendix C: Use Case Table

Use Case ID:	U1		
Use Case Name:	Register for new account		
Created By:	Heng Jiu Xiao	Last Updated By:	Heng Jiu Xiao
Date Created:	28/01/2021	Date Last Updated:	01/04/2021

Actor:	User
Description:	Allow users to register for an account
Preconditions:	1. User's device must be connected to the Internet through Wi-Fi / Mobile Data
Postconditions:	1. User's account is created

	2. User's personal and login information are stored in the database 3. Existing users are denied repeated registration
Priority:	High
Frequency of Use:	Once in entire lifetime
Flow of Events:	1. User selects "Register for new account" option 2. User will enter their email 3. User will enter password 4. User will enter confirm password 5. User will enter First Name 6. User will enter Last Name 7. User will press on the "Create" button 8. System will check against database to verify if email already exists 9. System will save login information into database 10. System will create a verification object with the User's Id and verification code 11. System will send an email with the verification code to the user's email 12. User will enter the verification code 13.User will press the "verify" button 14. User will be redirected back to Verify page
Alternative Flows:	AF-S9: If the email already exists in database 1. User would be prompted that account already exists with a warning message 2. User will be redirected back to the account creation page
Exceptions:	-
Includes:	1. Verify email in system
Extends:	-
Special Requirements:	-
Assumptions:	1. User inputs **accurate** information during registration process in steps 2 to 6 of Flow of Events 2. User will enter proper and identical passwords 3. User will enter a valid email address
Notes and Issues:	-

Use Case ID:	U2		
Use Case Name:	Verify email in system		
Created By:	Heng Jiu Xiao	Last Updated By:	Heng Jiu Xiao
Date Created:	28/01/2021	Date Last Updated:	01/04/2021

Actor:	Authentication System
Description:	Check whether the email already exists in the database
Preconditions:	1. System must be connected to the authentication system 2. System must be connected to Internet through Wi-Fi
Postconditions:	1. Authentication system updates the verification status of the user
Priority:	High
Frequency of Use:	Depends on number of account creation
Flow of Events:	1. User will enter 6 character code sent to email address 2. Authentication system will verify if the email is in the database and that the entered 6 character code corresponds to the unverified email 3. Authentication system will update the verification status of the account
Alternative Flows:	-
Exceptions:	-
Includes:	-
Extends:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	U3
Use Case Name:	Login into system

Created By:	Heng Jiu Xiao	Last Updated By:	Heng Jiu Xiao
Date Created:	28/01/2021	Date Last Updated:	01/04/2021

Actor:	User
Description:	Login into system
Preconditions:	1. User's device must be connected to the Internet through Wi-Fi / Mobile Data 2. User is logged into the application
Postconditions:	1. User is brought forward to home page
Priority:	High
Frequency of Use:	1-2 times a month
Flow of Events:	1. User will input email address 2. User will input password 3. User selects the "Login" button 4. System will verify if email exists in the database 5. System will verify if password is correct for the tagged email address 6. System will display "Login successful" 7. User will press the "Home" button 8. User will be brought to home page
Alternative Flows:	AF-S6: Email address does not exist in database 1. System will display the following warning message "Email doesn't exist!" 2. User will click the "Dismiss" button 3. System returns user to login page AF-S6: Password does not tally with database's password 1. System will display the following warning message "Email address doesn't exist!" 2. User will click the "Dismiss" button 3. System returns user to login page AF-S6: Account is not verified 1. System will display account is not verified 2. System will redirect to the verification page
Exceptions:	EX-S6: If email field is left blank 1. System displays "Please enter valid email" 2. User will click the "Dismiss" button

	3. User is returned to login page EX-S6: If password is left blank, 1. System displays “Please enter valid password” 2. User will click the “Dismiss” button 3. User is returned to login page
Includes:	1. Verify login credentials
Extends:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	U4		
Use Case Name:	Verify login credentials		
Created By:	Heng Jiu Xiao	Last Updated By:	Heng Jiu Xiao
Date Created:	28/01/2021	Date Last Updated:	01/04/2021

Actor:	Authentication System
Description:	Check whether the entered password corresponds to the tagged password for a email address in the database
Preconditions:	1. System must be connected to the authentication system 2. System must be connected to Internet through Wi-Fi
Postconditions:	1. Authentication system returns a binary value to indicate if password corresponds to tagged email address
Priority:	High
Frequency of Use:	Depends on number of login attempts

Flow of Events:	1. System will send an email address and a password to authentication system 2. Authentication system will verify if the password entered corresponds to the tagged password in the database 3. Authentication system sends a binary value to the system to indicate whether the login information tallies
Alternative Flows:	-
Exceptions:	-
Includes:	-
Extends:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	U5		
Use Case Name:	Create appointment under user		
Created By:	Ernest Tan	Last Updated By:	Ernest Tan
Date Created:	27/01/2021	Date Last Updated:	01/04/2021

Actor:	User
Description:	To create an appointment at clinic of choice
Preconditions:	1. User is connected to the Internet via Wi-Fi / Mobile Data 2. User does not have an existing appointment in the system
Postconditions:	1. Database is updated with the new booking 2. User will not be able to create a new appointment
Priority:	High
Frequency of Use:	1-3 times per month

Flow of Events:	1. User selects the “Book” option on the Directions page 2. System will display the appointment page 3. User must select the date of appointment 4. System will return available time slots based on location and date input 5. User selects one time slot from the available time slots 6. System displays appointment details consisting of date, location, and time 7. User selects the “Confirm” button 8. System displays “Appointment confirmed” page 9. User is sent to the “View Appointment” page 11. System will send the details of the appointment back to the database
Alternative Flows:	AF-S3: User inputs a date with no available time slots 1. System displays message “There are no more available time slots! Please change your time or date!” 2. Return to Step 3
Exceptions:	-
Includes:	-
Extends:	1. Display appointment under user 2. Update appointment under user 3. Delete appointment under user 4. Display shortest route to selected clinic from current location
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	U6		
Use Case Name:	Display appointment under user		
Created By:	Ernest Tan	Last Updated By:	Ernest Tan
Date Created:	27/01/2021	Date Last Updated:	01/04/2021

Actor:	User

Description:	To view the details of the current appointment
Preconditions:	1. User is connected to the Internet via Wi-Fi / Mobile Data 2. User must have a pre-existing appointment within the app 3. User must be logged in to the app
Postconditions:	1. User is able to view the location, date, time and comments of their existing appointment
Priority:	Medium
Frequency of Use:	3-5 times per month
Flow of Events:	1. User selects the “View Appointment” button 2. System will check the user’s email address for a pre-existing appointment in the database 3. System returns details of the user’s appointment, including location, date and time
Alternative Flows:	AF-S3: System cannot find a pre-existing appointment tagged to user 1. System will return an error message “There are no prior appointments!” 2. User will be returned to the application homepage
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	U7		
Use Case Name:	Update appointment under user		
Created By:	Ernest Tan	Last Updated By:	Ernest Tan
Date Created:	27/01/2021	Date Last Updated:	01/04/2021

Actor:	User
Description:	To change the date or time of the current booking
Preconditions:	1. User is connected to the Internet via Wi-Fi / Mobile Data 2. User must have a pre-existing appointment within the app 3. User must be logged in to the app
Postconditions:	1. User's booking details will be successfully updated and saved in the database
Priority:	Medium
Frequency of Use:	1-3 times per month
Flow of Events:	1. User selects the "Update Appointment" button on homepage 2. System will check the user's r for a pre-existing appointment in the database 3. System will display the appointment page 4. User must select the new date of appointment 5. System will return available time slots based on location and date input 6. User selects one time slot from the available time slots 7. System displays appointment details consisting of date, location, and time 8. User selects the "Confirm" button 9. System displays "Appointment updated" 10. User clicks on the "Dismiss" button 11. User is sent to the "View Appointment" page 12. System will send the details of the appointment back to the database
Alternative Flows:	AF-S1: User selects "Edit" button on "View Appointment" page 1. Return to step 2 AF-S3: System cannot find a pre-existing appointment tagged to user 1. System will return an error message "There are no prior appointments!" 2. User will be returned to the homepage AF-S4: User inputs a date with no available time slots 1. System returns an error message stating that there are no more available time slots, and that user must change either the location or date of booking 2. Return to Step 3
Exceptions:	-
Includes:	-

Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	U8		
Use Case Name:	Delete appointment under user		
Created By:	Ernest Tan	Last Updated By:	Ernest Tan
Date Created:	27/01/2021	Date Last Updated:	01/04/2021

Actor:	User
Description:	To delete the current appointment
Preconditions:	1. User must be logged in to the application 2. User is connected to the Internet via Wi-Fi / Mobile Data 3. User must have pre-existing appointment
Postconditions:	1. User's appointment is successfully deleted from the system's database 2. User will be able to create a new appointment
Priority:	Medium
Frequency of Use:	1-3 times per month
Flow of Events:	1. User selects the "Delete" button 2. System will check the user's email address for a pre-existing appointment in the database 3. System will display the message, "Do you want to delete your current appointment?" 4. User selects "Confirm" button 5. System will delete the user's appointment from the system's database 6. System displays "Appointment deleted" 7. User clicks on the "Dismiss" button 8. User is sent to the Home page
Alternative Flows:	AF-S3: User does not have a pre-existing appointment in the database 1. System will display the message "You do not have an appointment!"

	2. User will be returned to the application homepage
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	U9		
Use Case Name:	Display 5 nearest clinics according to current location		
Created By:	David Tay	Last Updated By:	David Tay
Date Created:	27/01/2021	Date Last Updated:	01/04/2021

Actor:	User
Description:	To display to the user the 5 nearest clinics by distance
Preconditions:	1. User is connected to the Internet via Wi-Fi / Mobile Data 2. User's device must have location services turned on on their device 3. User must be logged into application
Postconditions:	1. User will be shown the 5 nearest clinics with the respective distances from the current location
Priority:	High
Frequency of Use:	1-3 times a month
Flow of Events:	1. User selects the "Find nearest clinics" option on the application homepage 2. System will compute the distances of the 5 nearest clinics based on the user's current location 3. System will display a list of 5 nearest clinics based on distance, sorted in descending order

Alternative Flows:	-
Exceptions:	-
Includes:	1. Consolidate information
Extends:	1. Display shortest route to selected clinic from current location
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	U10		
Use Case Name:	Display clinic by name		
Created By:	David Tay	Last Updated By:	David Tay
Date Created:	28/01/2021	Date Last Updated:	01/04/2021

Actor:	User
Description:	Returns a list of clinics that names matches the user's search condition
Preconditions:	1. User is connected to the Internet via Wi-Fi / Mobile Data 2. User's device must have location services turned on on their device 3. User must be logged into application
Postconditions:	1. User will be given a list of clinics that names matches the search condition
Priority:	High
Frequency of Use:	1-3 times per month
Flow of Events:	1. User will input a search phrase in the search bar 2. User will select button to find clinics from input 3. System will retrieve travel time to the clinics 4. System will display a list of clinics that fulfill the search requirement and their respective travelling time

Alternative Flows:	-
Exceptions:	-
Includes:	1. Consolidate information
Extends:	1. Display shortest route to selected clinic from current location
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	U11		
Use Case Name:	Display shortest route to selected clinic from current location		
Created By:	David Tay	Last Updated By:	David Tay
Date Created:	28/01/2021	Date Last Updated:	01/04/2021

Actor:	User
Description:	Displays the shortest route to the selected clinic from current location
Preconditions:	1. User is connected to the Internet via Wi-Fi / Mobile Data 2. User's device must have location services turned on on their device 3. User must be logged into application
Postconditions:	1. User will be shown the shortest route to the selected clinic from current location
Priority:	High
Frequency of Use:	1-3 times a month
Flow of Events:	1. User will select a clinic from the list of clinics provided 2. System will display a map with the selected clinic and current location marked with a pin and a line displaying the shortest route
Alternative Flows:	-

Exceptions:	-
Includes:	-
Extends:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

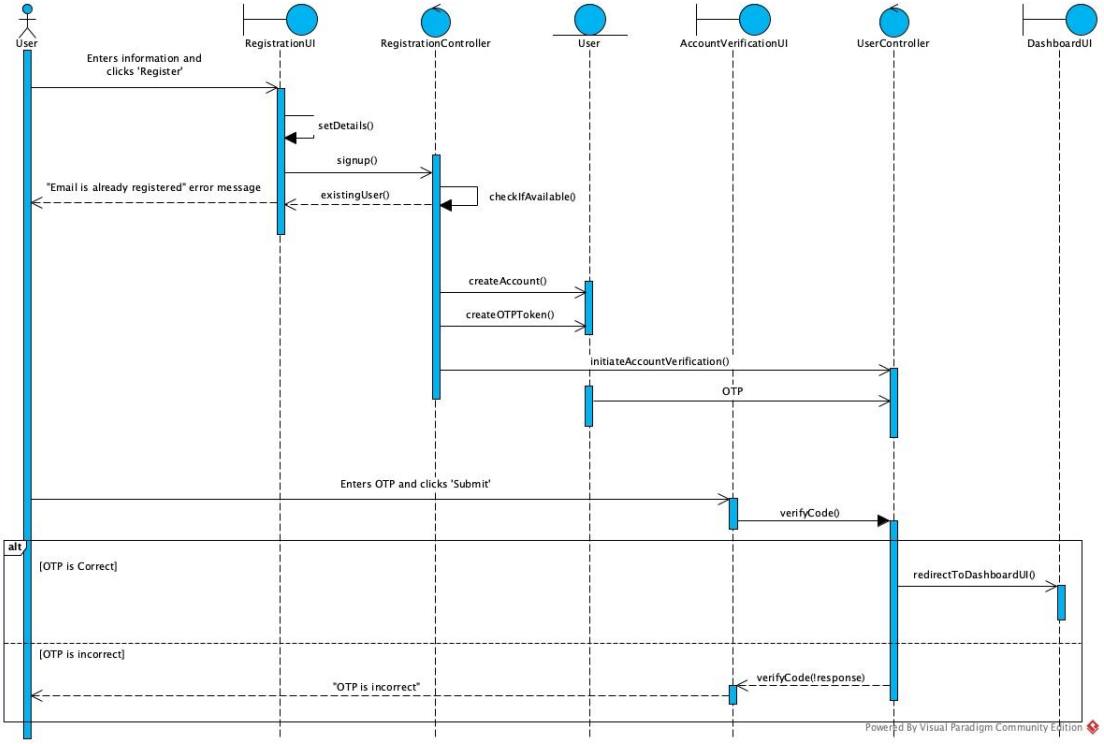
Use Case ID:	U12		
Use Case Name:	Consolidate Information		
Created By:	David Tay	Last Updated By:	David Tay
Date Created:	28/01/2021	Date Last Updated:	01/04/2021

Actor:	Google Maps API
Description:	Retrieve list of clinics, location and distance from Google Maps API
Preconditions:	1. User's device is connected to the Internet via Wi-Fi / Mobile Data 2. User's device must have location services turned on on their device
Postconditions:	1. Application will receive list of clinics and their details from Google Maps API 2. Application will receive user's current location from Google Maps API
Priority:	High
Frequency of Use:	Dependent on number of application users
Flow of Events:	1. System will retrieve list of clinics and their details from Google Maps API 2. System will receive user's current location from Google Maps API 3. System will store all information within the database for further use
Alternative Flows:	-
Exceptions:	-
Includes:	-

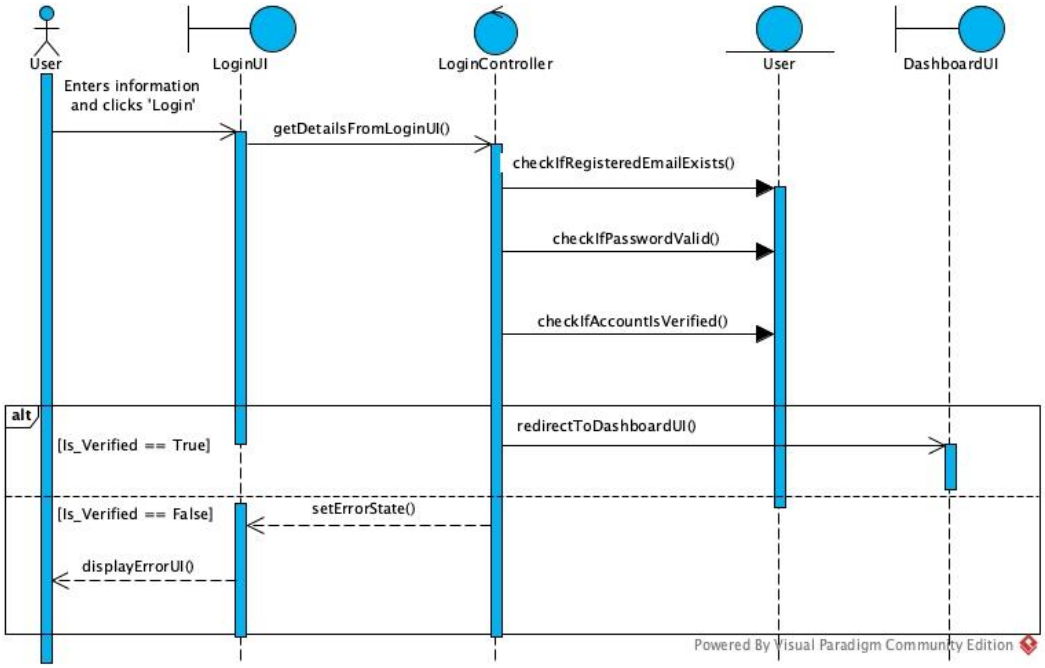
Extends:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Appendix D: Sequence Diagrams

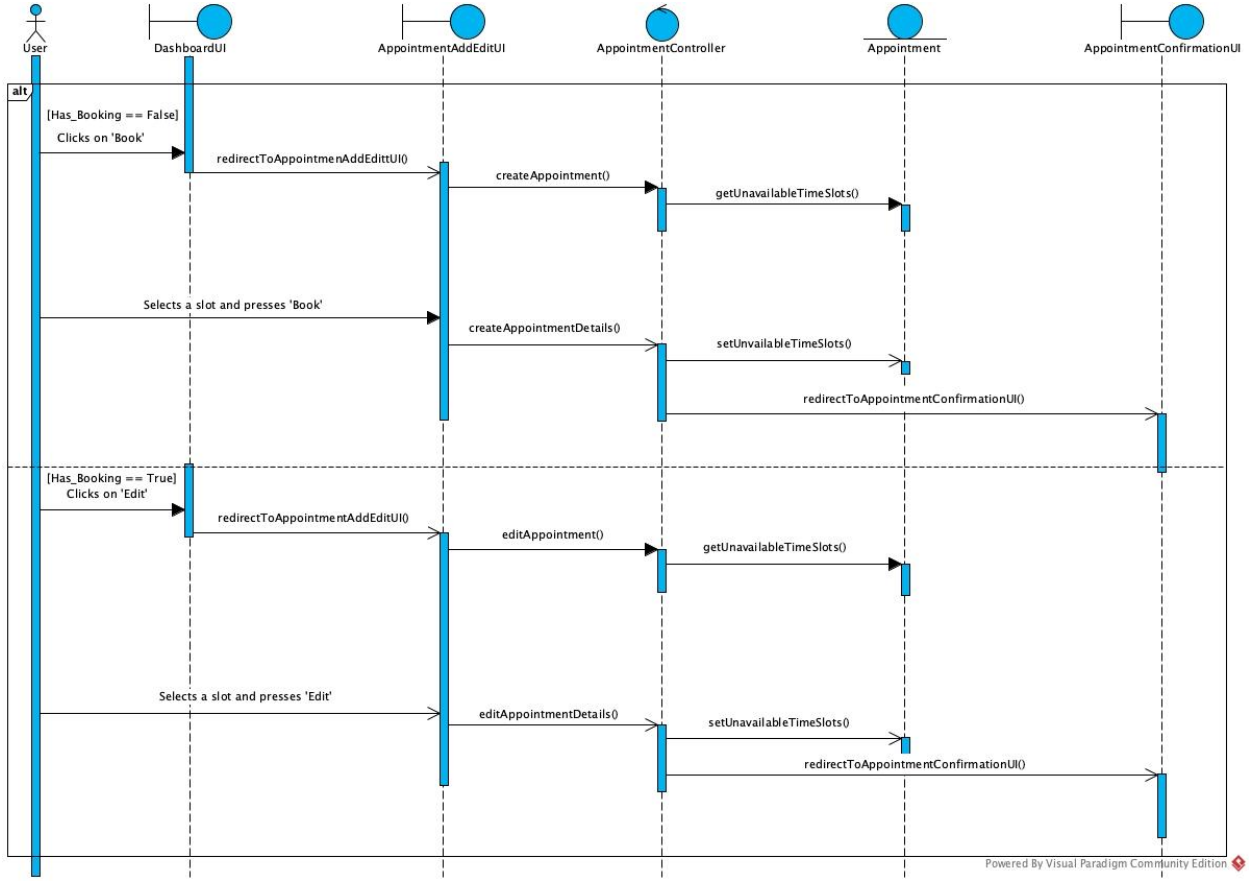
2.4.1 User Registration Sequence Diagram



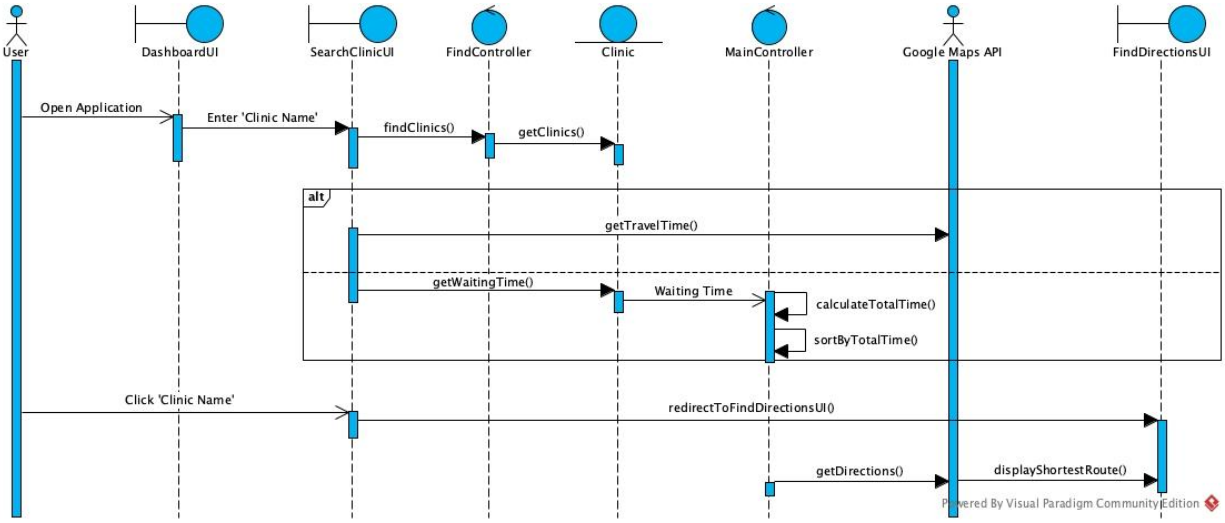
User Login Sequence Diagram



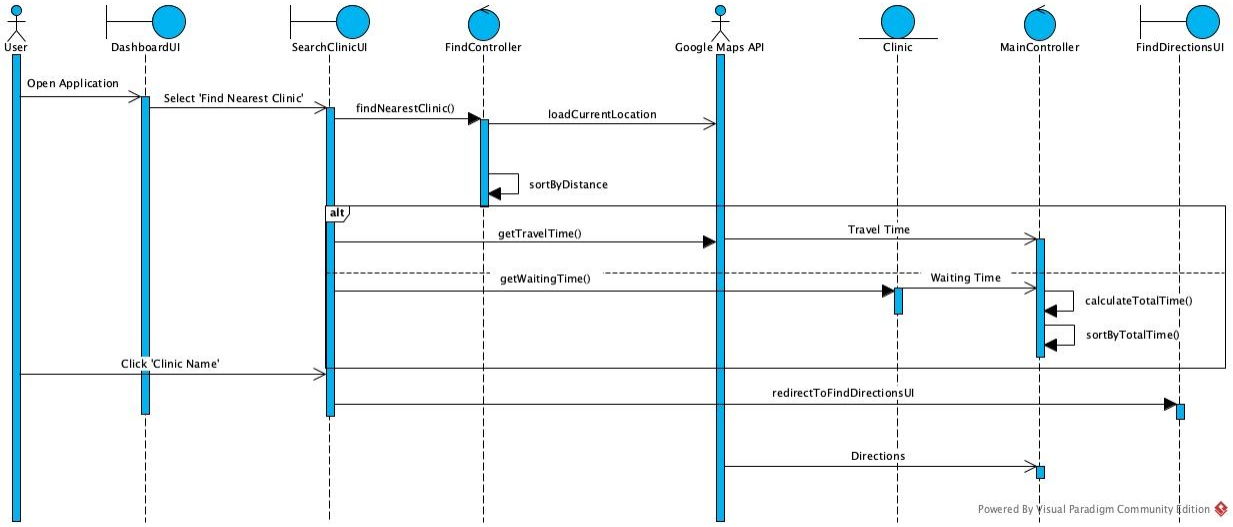
Create/Edit Appointment Sequence Diagram



Search by Clinic name Sequence Diagram



Find Nearest Clinic Sequence Diagram



Source: http://www.frontiernet.net/~kwiegers/process_assets/srs_template.doc