Software Requirements Specification

for

AnotherChat

Version 1.0 approved

Prepared by:

Joshua Alkins 190908

University of the West Indies

21/06/2020

Contents

[1. Introduction 3](#_Toc43655983)

[1.1 Purpose 3](#_Toc43655984)

[1.2 Document Conventions 3](#_Toc43655985)

[1.3 Intended Audience and Reading Suggestions 3](#_Toc43655986)

[1.4 Product Scope 3](#_Toc43655987)

[1.5 References 3](#_Toc43655988)

[2. Overall Description 4](#_Toc43655989)

[2.1 Product Perspective 4](#_Toc43655990)

[2.2 Product Functions 4](#_Toc43655991)

[2.3 User Classes and Characteristics 5](#_Toc43655992)

[2.4 Operating Environment 5](#_Toc43655993)

[2.5 Design and Implementation Constraints 5](#_Toc43655994)

[2.6 Assumptions and Dependencies 5](#_Toc43655995)

[3. External Interface Requirements 6](#_Toc43655996)

[3.1 User Interfaces 6](#_Toc43655997)

[3.1.1. Login Screen 6](#_Toc43655998)

[3.1.2. Register Screen 6](#_Toc43655999)

[3.1.3 Chat Screen 6](#_Toc43656000)

[3.1.4 Map Screen 6](#_Toc43656001)

[3.2 Software Interfaces 6](#_Toc43656002)

[4. System Features 7](#_Toc43656003)

[4.1 Change Theme 8](#_Toc43656004)

[4.1.1 Description and Priority 8](#_Toc43656005)

[4.1.2 Stimulus/Response 9](#_Toc43656006)

[4.1.3 Functional Requirements 9](#_Toc43656007)

[4.2 Login 9](#_Toc43656008)

[4.2.1 Description and Priority 10](#_Toc43656009)

[4.2.2 Stimulus/Response 10](#_Toc43656010)

[4.2.3 Functional Requirements 10](#_Toc43656011)

[4.3 Register Account 10](#_Toc43656012)

[4.3.1 Description and Priority 11](#_Toc43656013)

[4.3.2 Stimulus/Response 11](#_Toc43656014)

[4.3.3 Functional Requirements 11](#_Toc43656015)

[4.4 View Chat 12](#_Toc43656016)

[4.5.1 Description and Priority 13](#_Toc43656017)

[4.5.2 Stimulus/Response 13](#_Toc43656018)

[4.5.3 Functional Requirements 13](#_Toc43656019)

[4.6 Send Messages 13](#_Toc43656020)

[4.6.1 Description and Priority 13](#_Toc43656021)

[4.6.2 Stimulus/Response 13](#_Toc43656022)

[4.6.3 Functional Requirements 13](#_Toc43656023)

[4.7 View Location 13](#_Toc43656024)

[4.8.1 Description and Priority 13](#_Toc43656025)

[4.8.2 Stimulus/Response 13](#_Toc43656026)

[4.8.3 Functional Requirements 14](#_Toc43656027)

[5. Other Nonfunctional Requirement 14](#_Toc43656028)

[5.1 Performance Requirements 14](#_Toc43656029)

[5.2 Security Requirements 14](#_Toc43656030)

[5.3 Other Software Quality Attributes 14](#_Toc43656031)

[5.3.1 Learnability 14](#_Toc43656032)

[5.3.2 Availability (Network Failure) 15](#_Toc43656033)

[Appendix A: Glossary 16](#_Toc43656034)

[Appendix C: To Be Determined 17](#_Toc43656035)

[Additional Features 17](#_Toc43656036)

# 1. Introduction

## 1.1 Purpose

The purpose of this document is to give a description of the “AnotherChat” application for iOS. It will act as an overview of the entire system, explaining the purpose and features of the application and interfaces of the application.

## 1.2 Document Conventions

This document was created based on the IEEE template for System Requirements Specification Documents.

The document also incorporates Software Modeling techniques taught in the SWEN program.

## 1.3 Intended Audience and Reading Suggestions

This document is intended for:

* Programmers and developers working on the project.
* Dr. Xu, SWEN 3005 lecturer.
* Any individual involved in the assessment of SWEN courses.

While reading this document the project should be considered with the context of the SWEN 3005 iOS Development course syllabus in mind. This product was developed primarily to meet the requirements of the course and to incorporate as many features taught in the class into the project as reasonably possible. These features include:

1. View Controller
2. Text Input
3. CoCoa Design
4. Views & View Hierarchy
5. Maps/GPS
6. Notifications
7. Login
8. Database integration
9. HTTP Asynchronous tasks
10. Sounds

## 1.4 Product Scope

AnotherChat is a basic chat room application created to allow users to communicate remotly. Users can use the application to both interact with anyone else who joins the chat room. The application intends to allow users to share and view locations to all users to organize meet ups.

## 1.5 References

IEEE template for System Requirements Specification Documents: <https://goo.gl/nsUFwy>

# 2. Overall Description

## 2.1 Product Perspective

This application was designed to satisfy the requirements of the SWEN 3004 end of semester project and incorporate various features taught throughout the semester. This application, All-Chat Room, was inspired by applications such as WeChat and WhatsApp, and is designed to allow users to socialize over the internet via online chat rooms in a similar fashion.

## 2.2 Product Functions

The application aims to allow users to connect and chat in large public and private chat rooms to facilitate socializing.

The application primarily allows users to:

* Send Text Messages
* Share Media
* Share Locations

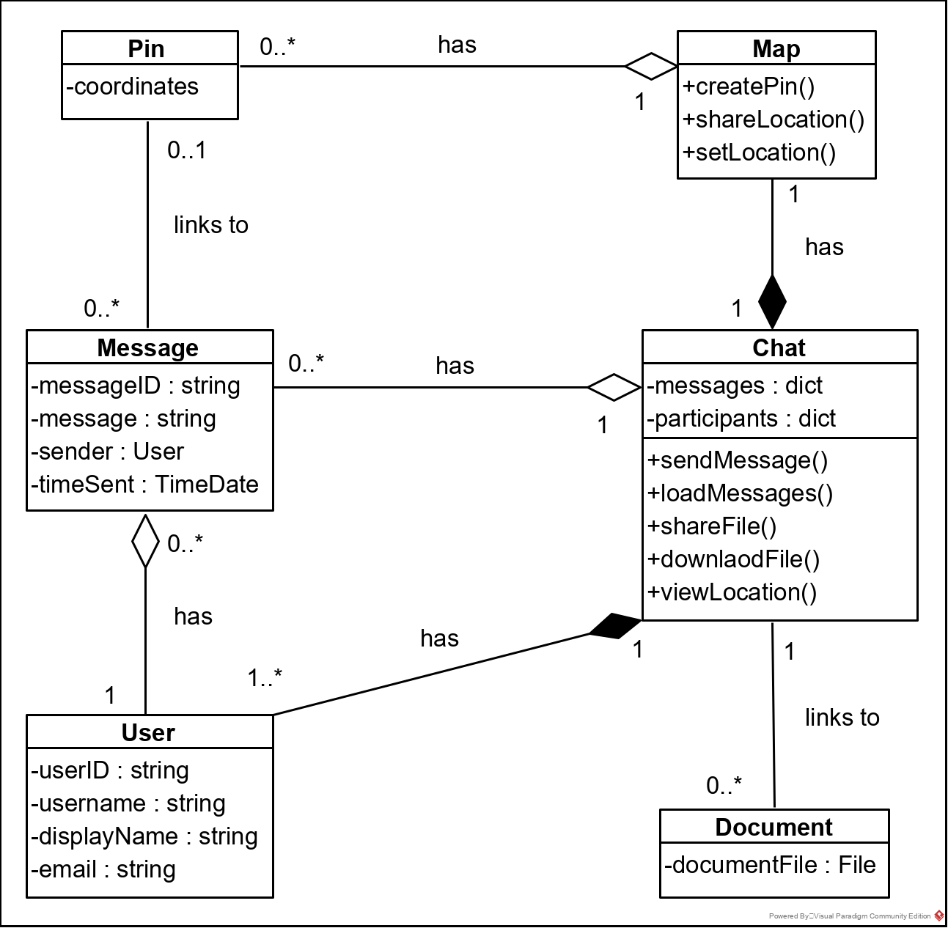


Figure 2.2.1. Object Class diagram containing methods relating to primary functionality.

## 2.3 User Classes and Characteristics

The application only has one user who has access to all functionality provided by the system.

## 2.4 Operating Environment

The system is intended for Devices running iOS 10 and above.

## 2.5 Design and Implementation Constraints

**CON-1** The application must use Google Firebase Firestore Database.

**CON-2** The application must use Apple Map Services.

**CON-3** The application must be designed to run on iOS devices.

## 2.6 Assumptions and Dependencies

Google Firebase Firestore Database – acting as the primary database and backend of the system.

# 3. External Interface Requirements

## 3.1 User Interfaces

The application currently consists of four screens: the Login Screen, the Register Screen, the Chat Screen and the Map Screen. The interface was constructed using a Storyboard with a Navigation Controller to perform transitions between the different views, and UIKit to form the appearance and functionality of the interface.

### 3.1.1. Login Screen

The Login Screen consists of a Label at the top to display the application name, two Text Fields to collect the user’s email and password, two Buttons (one to activate the login sequence and the other to navigate to the Register Screen). Additionally, there is a hidden label just bellow the Text Fields to display error messages when login information is invalid.

### 3.1.2. Register Screen

The Register Screen is very similar to the Login Screen with the exception that there are four Text Fields to allow the user to enter all information they need in order to register for an account.

### 3.1.3 Chat Screen

The Chat Screen consists primarily of a table view to display message sent to and received from the chat. Messages with display names matching the user’s are displayed with a white background and are aligned to the left, while messages with display names not matching the user’s are displayed on an orange background aligned to the right with an additional label to display the author’s name. At the bottom of the screen there is a Text Filed to enter messages the user wishes to send. Next to the Text Field are two Buttons, the done to the far left allows the user to send the message entered in the Text Field, the other just to the left of the Text Field navigates the user to the Map Screen. From this screen the user can logout by selection the back button in the top right hand side of the screen.

### 3.1.4 Map Screen

The Map Screen simply displays a map to the user using the Apple Map Kit and displays the user’s current location using CoreLocation, updating the user’s position on the map as the user moves.

## 3.2 Software Interfaces

* Firebase Firestore Version 17.4.3
* Firebase Auth
* Apple Map Services

# 4. System Features

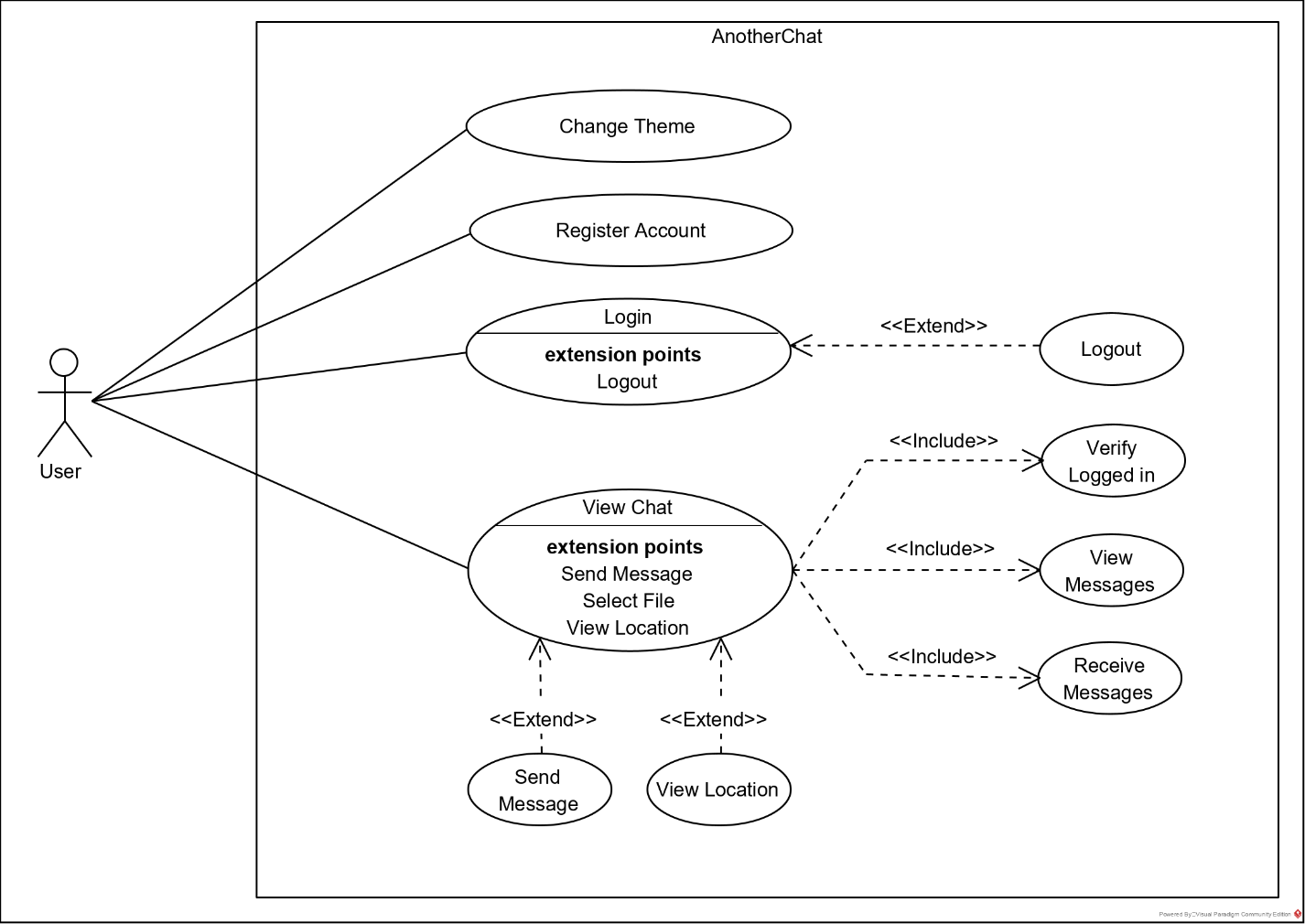


Figure 4.1. Use Case Context Diagram.

## 4.1 Change Theme

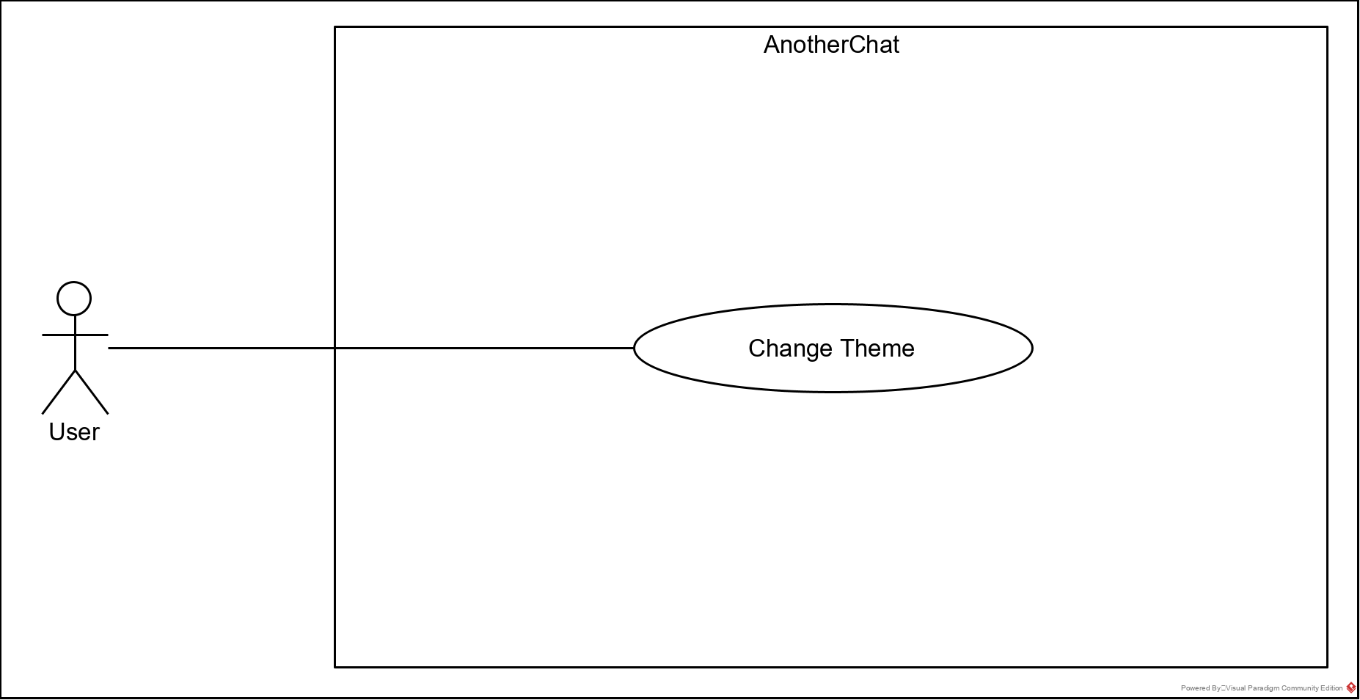


Figure 4.1.1. Change Theme Use Case Diagram.

Figure 4.1.2 Change Theme Use Case.

|  |  |
| --- | --- |
| Title | Change Theme |
| Use Case ID | UC-1 |
| Description | The application changes the theme either from the “light theme” to “dark theme” or vice versa to match the setting on the user’s device. |
| Primary Actors | User |
| Preconditions | Application is open. |
| Postconditions | Theme changes from the current theme to the alternate theme. |
| Main Flow | 1. The application checks the active theme selected in iOS. 2. The application changes the theme settings from current theme to alternate theme. 3. The application changes the displayed colors of objects in the application to suit the new theme. |
| Alternative Flows | - |
| Non-Functional Requirements | **QA-1** Theme must be set appropriately before the application becomes visible to the user. |

### 4.1.1 Description and Priority

The application is designed to respond to the user’s device theme settings.

Priority: **Medium**

### 4.1.2 Stimulus/Response

On opening the application, the application automatically responds to the user’s device settings and changes the theme to match.

### 4.1.3 Functional Requirements

**REQ-1** The application must match the theme settings on the user’s device.

## 4.2 Login

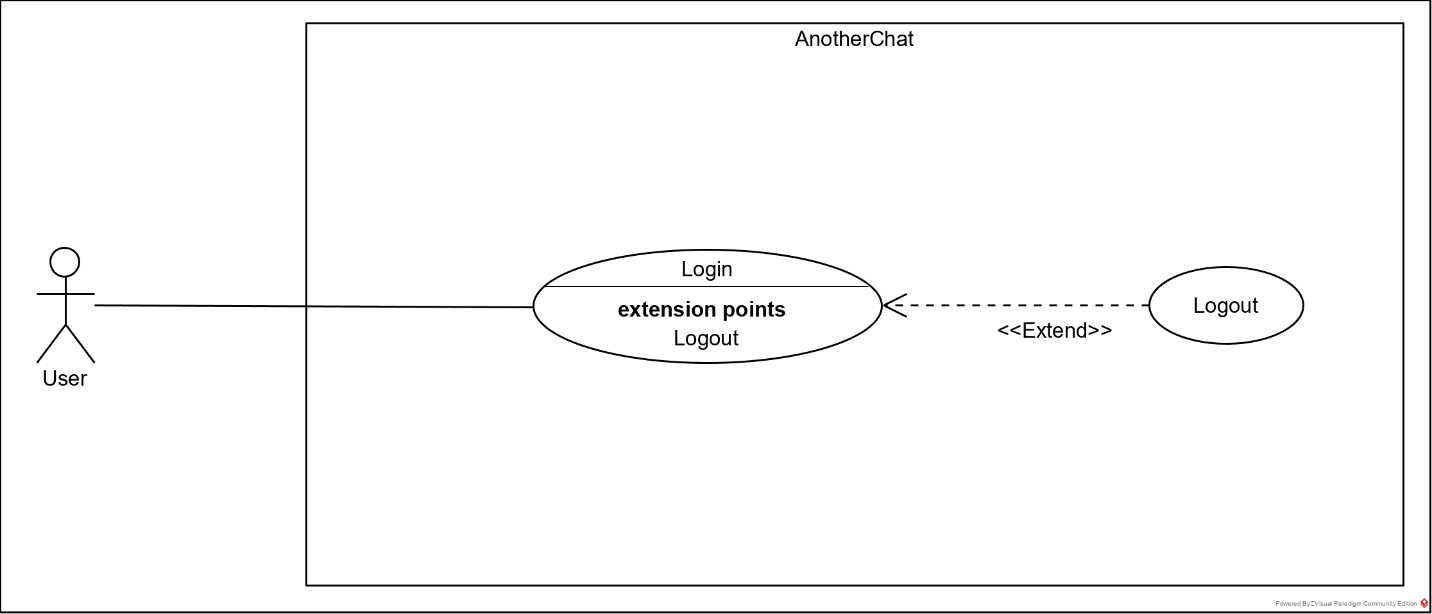


Figure 4.2.1. Login Use Case Diagram.

Figure 4.2.2. Login Use Case.

|  |  |
| --- | --- |
| Title | Login |
| Use Case ID | UC-2 |
| Description | The user selects their username and the application grants access to the system to the user. |
| Primary Actor | User |
| Preconditions | Application is open. |
| Postconditions | The user granted access to the system. |
| Main Flow | 1. The user selects their username. 2. The user selects “Login”. 3. The application gives the user access to the system. |
| Alternative Flows | 1. The user selects “Logout”    1. The application removes the user’s information from the application.    2. The application returns to the “Login Screen” |
| Non-Functional Requirements | **QA-2** The application should log the user in within 5 seconds of the user selecting login.  **QA-3** The application should let the user select the “Login” button if they have not selected their username.  **QA-4** The application should open the Chat within 5 seconds the user being logged in.  **QA-5** The application should log the user out and navigate to the Login Screen within 1 second of the user selecting logout. |

### 4.2.1 Description and Priority

The user selects their username and the application then grants the user access to the systems functionality and the user’s account information

Priority: **High**

### 4.2.2 Stimulus/Response

The user enters their email and password into the available fields and selects login. The system then uses Firebase Auth API to verify the user’s information and logs the user into the application.

### 4.2.3 Functional Requirements

**REQ-2** The user must be able to login to the system using email and password.

**REQ-3** The application must play a sound on login to notify the user.

**REQ-4** The user must be able to logout of the system.

## 4.3 Register Account

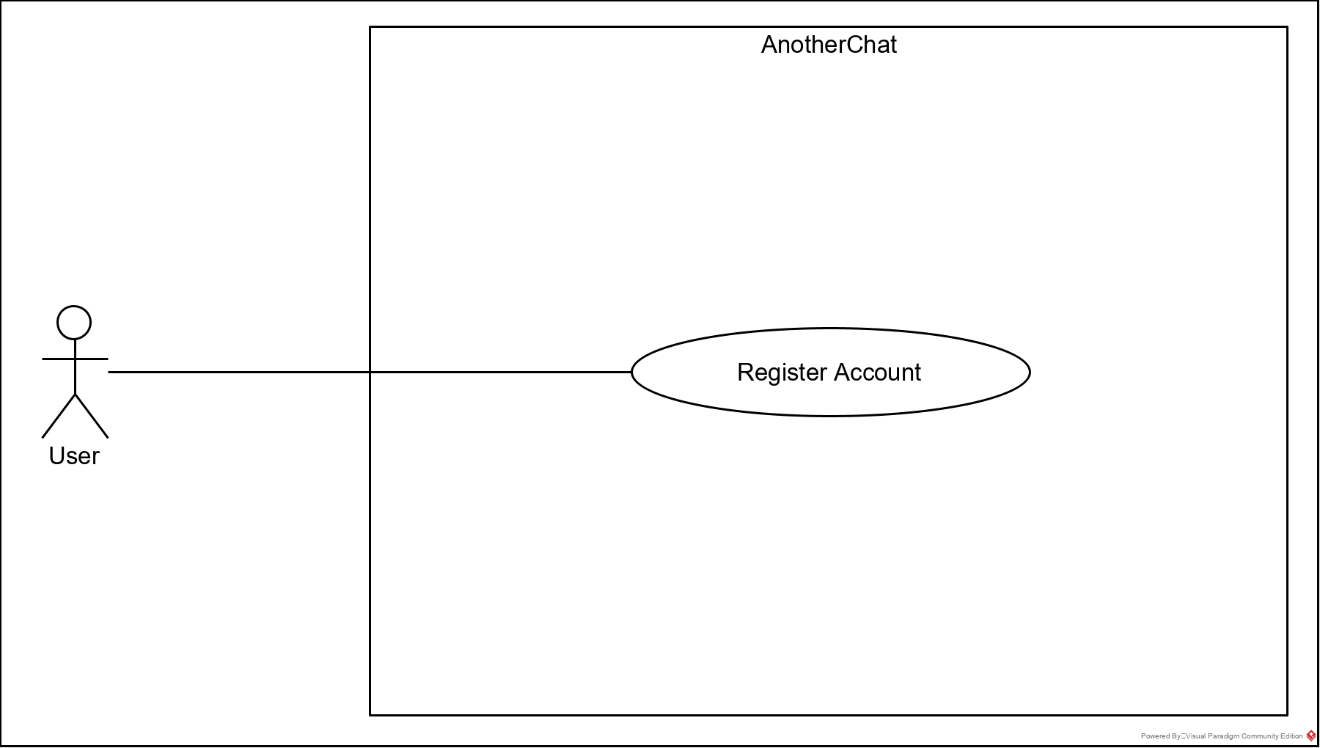


Figure 4.3.1. Register Account Use Case Diagram.

Figure 3.3.2 Register Use Case.

|  |  |
| --- | --- |
| Title | Register Account |
| Use Case ID | UC-3 |
| Description | The user registers a new account with the system |
| Primary Actor | User |
| Preconditions | Application is open |
| Postconditions | A new account is added to the system |
| Main Flow | 1. The user enters their information 2. The user selects “Register” 3. The application adds the new account to the database. 4. The user is logged into the application. |
| Alternative Flows | - |
| Non-Functional Requirements | **QA-6** The application should register the new user within 5 seconds from the user selecting the “Register” button.  **QA-7** The application should the user select the “Register” button if the user is missing information. |

### 4.3.1 Description and Priority

The user registers a new account with the system, allowing the user to gain access to the system’s functionality in the future. Using Firebase Auth taking the user’s email and password to generate the new account.

Priority: **High**

### 4.3.2 Stimulus/Response

The user enters their display name, email and password into the available fields, confirms their password and selects “Register”. The application then adds the new account to the database. The user is then automatically logged in and sent to the application.

### 4.3.3 Functional Requirements

**REQ-5** The user must be able to create a new account to access the application.

**REQ-6** The application must take a display name, email and password to register the user.

## 4.4 View Chat

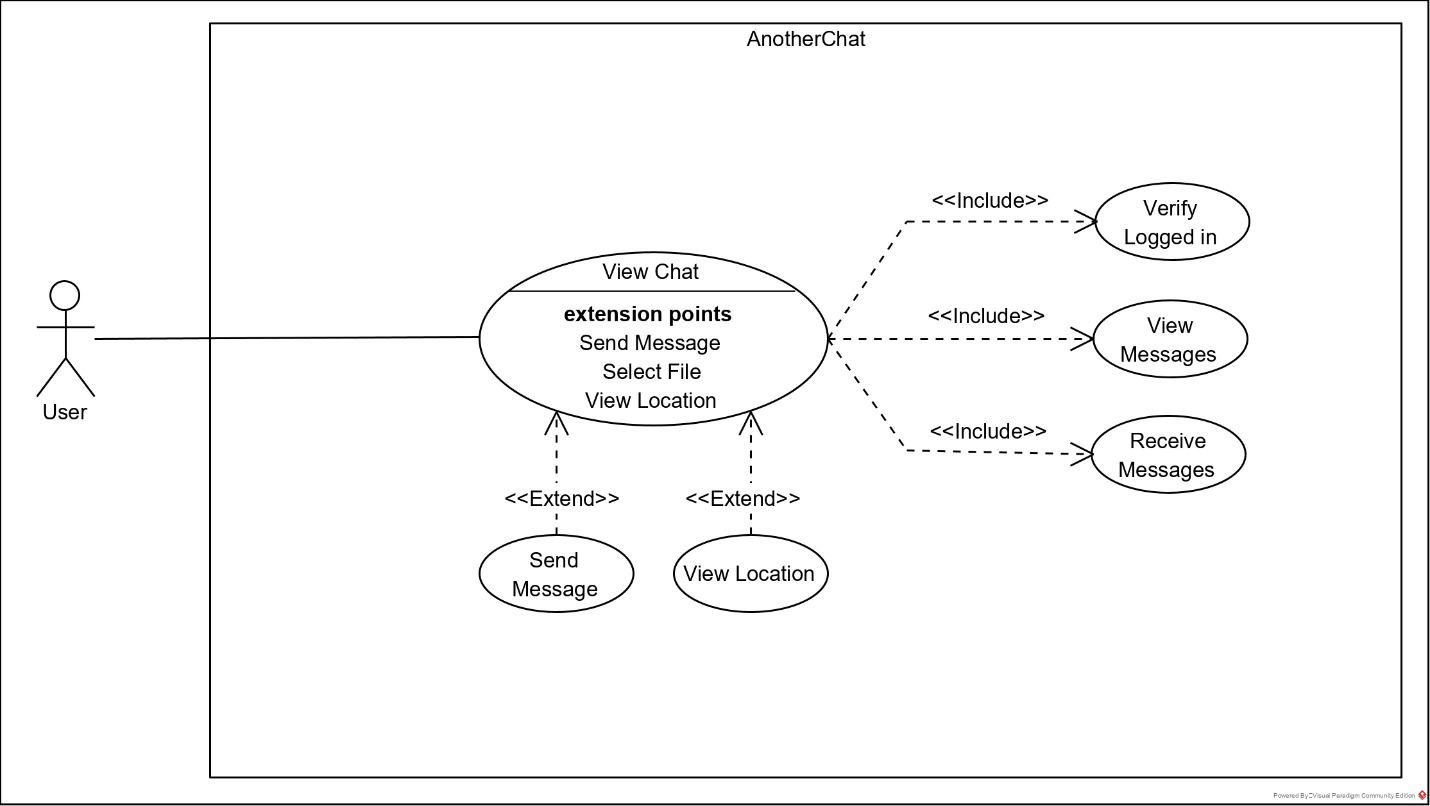


Figure 4.4.1. View Chat Use Case Diagram. change

Figure 4.4.2. View Chat Use Case.

|  |  |
| --- | --- |
| Title | View Chat |
| Use Case ID | UC-4 |
| Description | The user is presented with the chat history associated with the chat they have entered. |
| Primary Actor | User |
| Preconditions | User is logged in |
| Postconditions | - |
| Main Flow | 1. The application retrieves all messages previously sent in the chat from the database. 2. The application displays the messages to the user in chronological order. 3. The application listens for new messages sent in the chat room. |
| Alternative Flows | 1. Send Message    1. The user enters text    2. The user selects send    3. The application sends message to database 2. View Location    1. The application navigates to the map |
| Non-Functional Requirements | **QA-8** All messages should be displayed within 3 seconds of opening the chat.  **QA-9** The application should upload a message to the database within 1 second of the user selecting send.  **QA-10** The application should display a newly sent message within 1 second of being notified by the database that a new message has been sent.  **QA-11** Sent messages should be displayed in order of time sent. |

### 4.5.1 Description and Priority

The application loads the previous 50 messages sent in the chat and displays the m chronologically, with the most recent message at the bottom.

Priority: **High**

### 4.5.2 Stimulus/Response

The application retrieves all messages previously sent in the chat from the database and then displays the messages to the user in ascending chronological order. The application then listens for new messages sent in the chat room and displays any messages that are sent.

### 4.5.3 Functional Requirements

**REQ-7** The user must be able to view chat history.

**REQ-8** The application must receive messages for an active chat room in real time.

## 4.6 Send Messages

### 4.6.1 Description and Priority

The application sends a message created by the current user, to all other users in the chat.

Priority: **High**

### 4.6.2 Stimulus/Response

The user enters a message into the available field and selects send. The application then uploads the message to the Firestore database. All other user applications are then notified of the upload.

### 4.6.3 Functional Requirements

**REQ-9** Users must be able to send custom text messages in a chat room.

## 4.7 View Location

### 4.8.1 Description and Priority

The user opens the map service which displays the user’s current location.

Priority: **Medium**

### 4.8.2 Stimulus/Response

The user selects the map service and the application navigates to the map screen. The map screen then displays the user’s location on the map using a combination of Apple Maps Services and CoreLocation.

### 4.8.3 Functional Requirements

**REQ-10** The user must be able to open a map service and view their current location.

# 5. Other Nonfunctional Requirement

## 5.1 Performance Requirements

**QA-1** Theme must be set appropriately before the application becomes visible to the user.

**QA-2** The application should log the user in within 5 seconds of the user selecting login.

**QA-4** The application should open the Chat within 5 seconds the user being logged in.

**QA-5** The application should log the user out and navigate to the Login Screen within 1 second of the user selecting logout.

**QA-6** The application should register the new user within 5 seconds from the user selecting the “Register” button.

**QA-8** All messages should be displayed within 3 seconds of opening the chat

**QA-9** The application should upload a message to the database within 1 second of the user selecting send.

**QA-10** The application should display a newly sent message within 1 second of being notified by the database that a new message has been sent.

**QA-11** Sent messages should be displayed in order of time sent.

**QA-12** The system should be able to handle 1 million concurrent users each sending 1 message per second.

## 5.2 Security Requirements

**QA-13** Users should be required to (register an account and) login to the system before accessing any of the services provided by the app.

**QA-14** Users should not be able to view or access any personal or account information about any other user in the system.

## 5.3 Other Software Quality Attributes

### 5.3.1 Learnability

**QA-3** The application should let the user select the “Login” button if they have not selected their username.

**QA-7** The application should the user select the “Register” button if the user is missing information.

**QA-15** The app should only have no more than 5 selectable buttons present on any given screen.

**QA-16** All buttons should give a clear indication to their function, either via text on the button or text in the form of a toast shown on long click, describing the button’s function.

### 5.3.2 Availability (Network Failure)

**QA-17** The application should display all data that was prevented from being displayed due to a network failure once the app has reconnected with the network.

# Appendix A: Glossary

**Firebase** – A mobile application development platform from Google that provides several useful services.

**Firestore** – A database service provided by Firebase. A notable feature of which is ability to notify clients using the database of updates in real time.

**Message** – A collection of information including a string of text to be displayed (usually entered by users), the ID of the user who created the message, and the time and date the message was created.

**Chat** – A collection of messages and users who are communicating or sharing information via those messages.

**Chat Room** – A digital space that facilitates a chat.

**Chat History** – A collection of messages previously sent in a chat.

# 

# Appendix C: To Be Determined

## Additional Features

Chat Room Selection – The ability to select between multiple chat rooms, each housing a different chat.

Chat Room Creation – The ability for users to create new chat rooms.

Change able display name – A feature to allow users to alter the name that is displayed on messages to identify the user as the author of the message.

Alternate color pallets – A feature to allow the user to select different color pallets for the application, changing the color of components of the app to better fit the user’s preferences.

Colored display names – A feature to color each users name when displayed as the author of a message to help identify the author more easily.

Settings Page - A dedicated page to allow the user alter setting. Currently not implemented due to the lack of adjustable settings currently implemented.

Private chat rooms – The ability to add a custom password to a chat room, to restrict access to a chat to only certain users.

Administrators – Add a new user class for trusted individuals that can remove chats or users from the system for misconduct.

Moderators – Add a user class that can remove users from a specific chat for mis conduct.

Location Sharing – Allow the user to share a location in the chat for others to view on the map screen.

File Sharing – Allowing users to share images and other files from their device in chat.