Software Requirements Specification

for

CRIMEWATCH

Version 1.6 approved

Prepared by

AGARWAL DHRUVIKAA

DOMMARAJU PRANATI

KHOR HAOJUN

TAN CHUEN KEAT

ZAYD SHABAZ ALI

Nanyang Technological University

2025-04-13

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
KHOR HAOJUN	2025-03-04	Initial write-up.	1.0
PRANATI	2025-03-13	Included section *1*.	1.1
DHRUVIKAA	2025-03-18	Updated Appendix A.	1.2
TAN CHUEN KEAT	2025-03-24	Initial write-up on section *2*.	1.3
ZAYD SHABAZ ALI	2025-03-25	Included section *3*.	1.4
DHRUVIKAA	2025-04-04	Included Supplementary Materials	1.5
PRANATI	2025-04-10	Modified System Architecture Diagram	1.5.1
KHOR HAOJUN	2025-04-13	Included section *5*, finalizing documentation.	1.6

1. Introduction

1.1 Purpose

This Software Requirement Specification (SRS) document is intended for the *CrimeWatch* mobile application. The purpose of this SRS document is to describe the requirements specifications for the *CrimeWatch* mobile application to facilitate the development and production process for all users. All aspects of the mobile application, which includes but not limited to, the system features, the limitations, the non-functional and interface requirements, are documented within this SRS document.

1.2 Document Conventions

This section describes the conventional standards used throughout this document. It is imperative that all readers pay attention to the standards listed in this section.

Font: Times New Roman

Heading: Bold, Size 18

Sub-heading: Bold, Size 14

Content: Italic, Size 12

Technical Standards: IEEE 830-1998

Refer *Appendix A: Data Dictionary* for the definitions of special terms used throughout this documentation.

1.3 Intended Audience and Reading Suggestions

This document is intended for all stakeholders, which include the users of *CrimeWatch* mobile
This document begins by stating the purpose of the mobile application and several conventions used throughout the document. Next, a high-level overview of the application functionalities is introduced, followed by several design constraints and assumptions of the application. Then, the interface requirements of the application are stated. Finally, the document includes a detailed write-up of the system features and non-functional requirements of the application.

All stakeholders are advised to begin by reading section *1.1 Purpose*, *1.2 Document Conventions* and *Appendix A: Data Dictionary* to be familiarized with the purpose of the web application, as well as the documentation standards and technical terms definition used throughout this document.

The *CrimeWatch* development team is strongly encouraged to proceed with section *2. Overall Description* to have a high-level understanding of the application functionalities, design, and constraints. Then, section *4. System Features* follows, where the developers will gain a low-level understanding of each system features to be included in the application. Finally, the developers should read section *3. External Interfaces Requirements* and *5. Other Nonfunctional Requirements* to understand the requirements specified for the application to function as desired.

On the other hand, the users of *CrimeWatch* mobile application, the *CrimeWatch* testing team, the project managers and the *CrimeWatch* marketing team are encouraged to proceed reading this document in sequential order.

1.4 Product Scope

Citizens facing emergencies often don't know how to directly contact the nearest police center. Currently, reports go through multiple communication layers via the Police Operations Command Centre (POCC)This process can be time-consuming, especially when users are required to describe their location and situation verbally. These delays can lead to slower response times and increased stress in already critical situations.

The CrimeWatch application allows citizens to report crimes or suspicious activity directly to the nearest police station using their mobile device. It leverages geolocation to identify the nearest station and securely sends report data, potentially including media attachments. The goal is to facilitate quicker and more accessible crime reporting, promoting safety and public trust in law enforcement.

1.5 References

i. *IEEE 830-1998 - IEEE Recommended Practice for Software Requirements Specifications. IEEE Standards Association. (n.d.). Retrieved April 13, 2025, from https://standards.ieee.org/ieee/830/1222/*

ii. *React Native Documentation. React Native. (n.d.). Retrieved April 13, 2025, from https://reactnative.dev/docs/getting-started*

iii. *FastAPI Documentation. FastAPI. (n.d.). Retrieved April 13, 2025, from https://fastapi.tiangolo.com/*

iv. *MySQL 8.0 Documentation. MySQL. (n.d.). Retrieved April 13, 2025, from https://dev.mysql.com/doc/refman/8.0/en/*

v. *Google Maps Platform Documentation. Google Developers. (n.d.). Retrieved April 13, 2025, from https://developers.google.com/maps/documentation*

vi. *Twilio API Documentation. Twilio. (n.d.). Retrieved April 13, 2025, from https://www.twilio.com/docs/usage/api*

vii. *Representational State Transfer (REST). Service Architecture. Barry, D. K. (n.d.). Retrieved April 13, 2025, from https://www.service-architecture.com/articles/web-services/representational-state-transfer-rest.html*

2. Overall Description

2.1 Product Perspective

The *CrimeWatch* mobile application is a new, standalone, and self-contained mobile application. An overall system diagram depicting the operation of *CrimeWatch* mobile application is as follows:

2.2 Product Functions

The *CrimeWatch* mobile application's system features can be broken down into three main sub-categories —— Users and Reports and Side Functions. This section provides a high-level view of the system features provided by the mobile application. Specifics regarding each feature, such as the activity flows, conditions, assumptions, and functional requirements achieved can be found under section *4. System Features*.

2.2.1 Users

i. Create User Account
ii. Login to User Account
iii. Edit User's Profile

2.2.2 Reports

i. View Crime Offenses
ii. Display Most Common Crime Types
iii. Make A Report
iv. Send SMS Notification to NPC
v. Find the Nearest Police Station
vi. Access User's Location Data

2.2.3 Side Functions

i. Submit Feedback
ii. View History

2.3 User Classes and Characteristics

The *CrimeWatch* mobile application anticipates its demographic audience to be the following, ranked by importance:

2.3.1 General Public (Civilians/Residents)

Attributes	**Description**
Frequency of use	High
Subset of functions used	All
Technical expertise	Low
Characteristics	Users in this class are everyday citizens who may not have advanced technical knowledge. They rely on the app to report crimes, access nearby police support, and stay informed about common crimes in their area. Usability and clear UI/UX are essential to ensure effective engagement.

2.4 Operating Environment

This section breaks down the operating environment into two sub-categories — production environment and development environment.

All stakeholders except the CrimeWatch development team and the CrimeWatch testing team are not required to be familiarized with the specifics of the development environment. However, both teams must be familiar with both environments.

2.4.1 Production Environment of *CrimeWatch*

This sub-section describes the setting of which the mobile application is put into operation.

Setting	**Description**
Mobile Platform Support	The mobile application is designed to run on both Android and iOS platforms via React Native and the Expo Go environment.
Internet Connectivity	Requires active internet connection to send reports, access geolocation services, and interact with APIs such as Google Maps and Twilio.
API Integration Support	Must support RESTful API interactions with the backend hosted on Railway, and external services like Google Maps API and Twilio API.
Location Services	Requires device-level permission to access GPS for finding the nearest police station and tagging the user's current location in reports.
SMS Capability	Utilizes Twilio API to send SMS alerts to the National Police Command (NPC) or emergency contacts when necessary

2.4.2 Development Environment of *CrimeWatch*

This sub-section describes the setting of which the mobile application is built and tested on during development phase.

Setting	**Description**
Front-end development using React Native	React Native is an open-source framework for building cross-platform mobile apps using JavaScript and React. Used with Expo for rapid development and testing. Edition: React Native (0.73.x) with Expo SDK (v50)
Back-end development using FastAPI	FastAPI is a modern, high-performance web framework for building APIs with Python 3. It is used to manage user authentication, report processing, and data logic. Edition: FastAPI (0.110.x)
Database using MySQL	MySQL is an open-source relational database system used to store user profiles, crime reports, and user feedbacks. Hosted using Railway cloud platform. Edition: MySQL (v8.0)
Application Hosting using Railway	Railway is a cloud platform used to host the MySQL database. It offers CI/CD integration and simple deployment for backend services.
Mapping Services	Google Maps API is used for real-time geolocation and to find the nearest police stations.
SMS Services	

2.5 Design and Implementation Constraints

This section covers all constraints which have limited, or will be limiting, options available to the *CrimeWatch* development team, both during development stage and post-production maintenance stage. Additionally, this section also documents all design standards of the mobile application.

2.5.1 Limitations

This sub-section covers all limitations which may or may not affect certain functionalities of the mobile application.

Limitations	Details
Google Maps API Limitations	The Google Maps API operates under a freemium model. The free tier has daily usage limits for geolocation and map loads. Exceeding this quota may result in temporary unavailability of location-based features until the monthly billing cycle resets or additional quota is purchased.
Twilio API Limitations	The Twilio API used for SMS alerts operates on a pay-as-you-go model. In the free trial version, messages are limited and may include a Twilio branding prefix. This can affect the delivery of emergency messages if the credit is depleted.
Cloud Hosting Limitations	The CrimeWatch backend and database are hosted on Railway under its free tier. This tier has limited resources (e.g., monthly usage hours, storage) and may lead to temporary downtime or limited performance if exceeded. Upgrading to a paid tier may be necessary for production-scale use.

Mobile Permissions Dependency	Core features like reporting a crime, locating nearby stations, and sending alerts require access to device GPS, camera, and SMS functionality. The app cannot perform these actions if users deny the relevant permissions.

2.5.2 Design Standards

This sub-section covers all design standards adopted by the *CrimeWatch* mobile application.

Design Standards	**Details**
Programming Standards	All non-class variables must adopt the camelCase naming conventions. All class variables must adopt the Pascal Case naming conventions.
User Interface Standards	All User Interface designs must adhere to the stakeholders' pre-approved colour scheme.

2.6 User Documentation

2.6.1 *CrimeWatch* API Documentations

The *CrimeWatch* testing team is strongly advised to go through each API endpoint's documentation to be familiarized with the endpoints provided by the *CrimeWatch* mobile application back-end server.

- Ali, Z. S.(n.d.). *CrimeWatch API Documentation (Authentication & User Management)*. CrimeWatch API. Retrieved March 14, 2025, from https://documenter.getpostman.com/view/24005937/2s84DmwjBR
- Ali, Z. S. (n.d.). *CrimeWatch API Documentation (Police Reporting & Emergency Contacting)*. CrimeWatch API. Retrieved March 20, 2025, from https://documenter.getpostman.com/view/24005937/2s84Dmx3yZ
- Ali, Z. S. (n.d.). *CrimeWatch API Documentation (Profile & Feedback Management)*. CrimeWatch API. Retrieved March 27, 2025, from https://documenter.getpostman.com/view/24005937/2s84Dmx3yb

2.7 Assumptions and Dependencies

The *CrimeWatch* development team developed the *CrimeWatch* mobile application with the following assumptions:

- **API Subscription Assumption**

  The full-scale version of the CrimeWatch mobile application is expected to receive funding for full API subscriptions (e.g., Google Maps API, Twilio API) as detailed in section *4. Limitations* once the prototype is approved by all stakeholders.

- **Cloud Hosting & Database Assumption**

  It is assumed that the production version will be granted access to unrestricted cloud hosting and database services, either through paid plans or sponsored infrastructure (e.g., Railway, MySQL) to ensure stable uptime and scalability.

- **Mobile-First Design Assumption**

  The CrimeWatch mobile application is built using React Native, with the assumption that it will be primarily accessed on mobile devices (Android and iOS). It is not optimized for desktop browsers, although basic web access through Expo or a web wrapper may be possible during development.

- **Authentication Behavior Assumption**

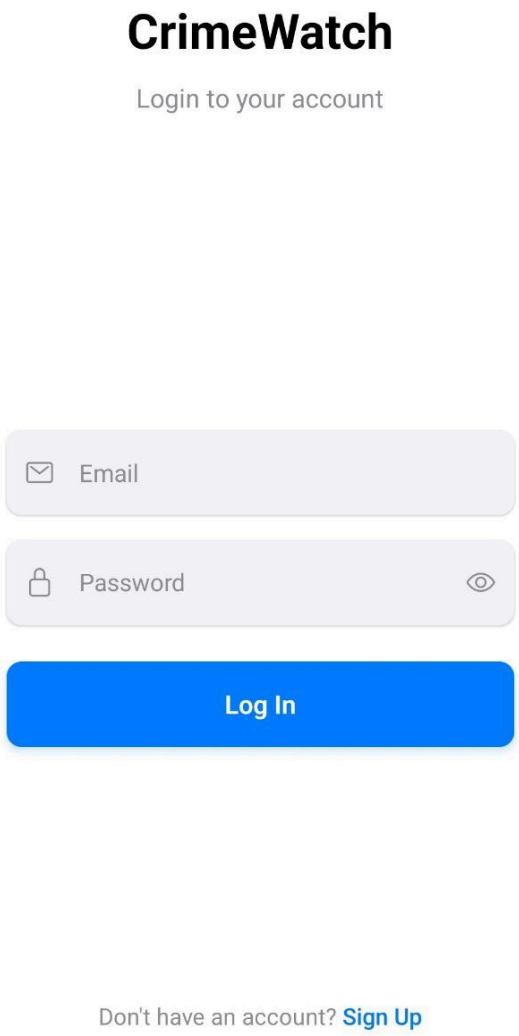
  The current implementation does not include OAuth or automatic session expiration. Therefore, users remain logged in unless they manually log out. It is assumed that users will log out responsibly after each session to maintain security.

3. External Interface Requirements

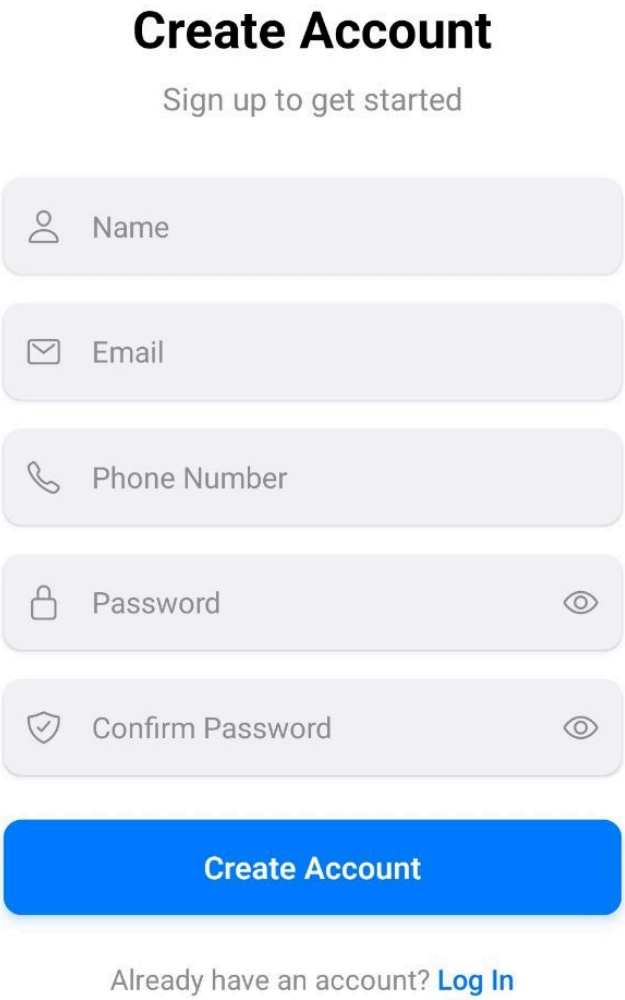
3.1 User Interfaces

The *CrimeWatch* mobile application is designed to be viewed with an aspect ratio which matches a mobile. On top of that, the mobile must satisfy the requirements as indicated within section *2.4.1 Production Environment of CrimeWatch*.

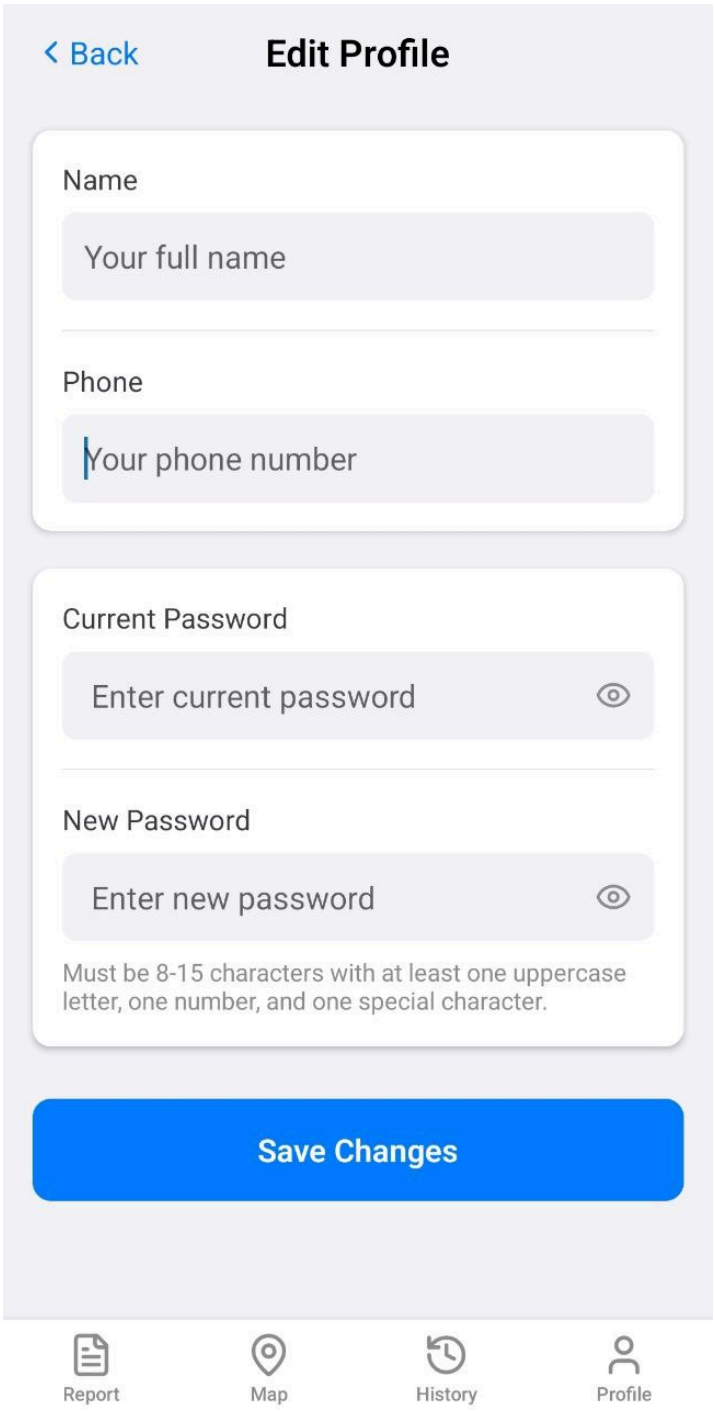
3.1.1 Login Page



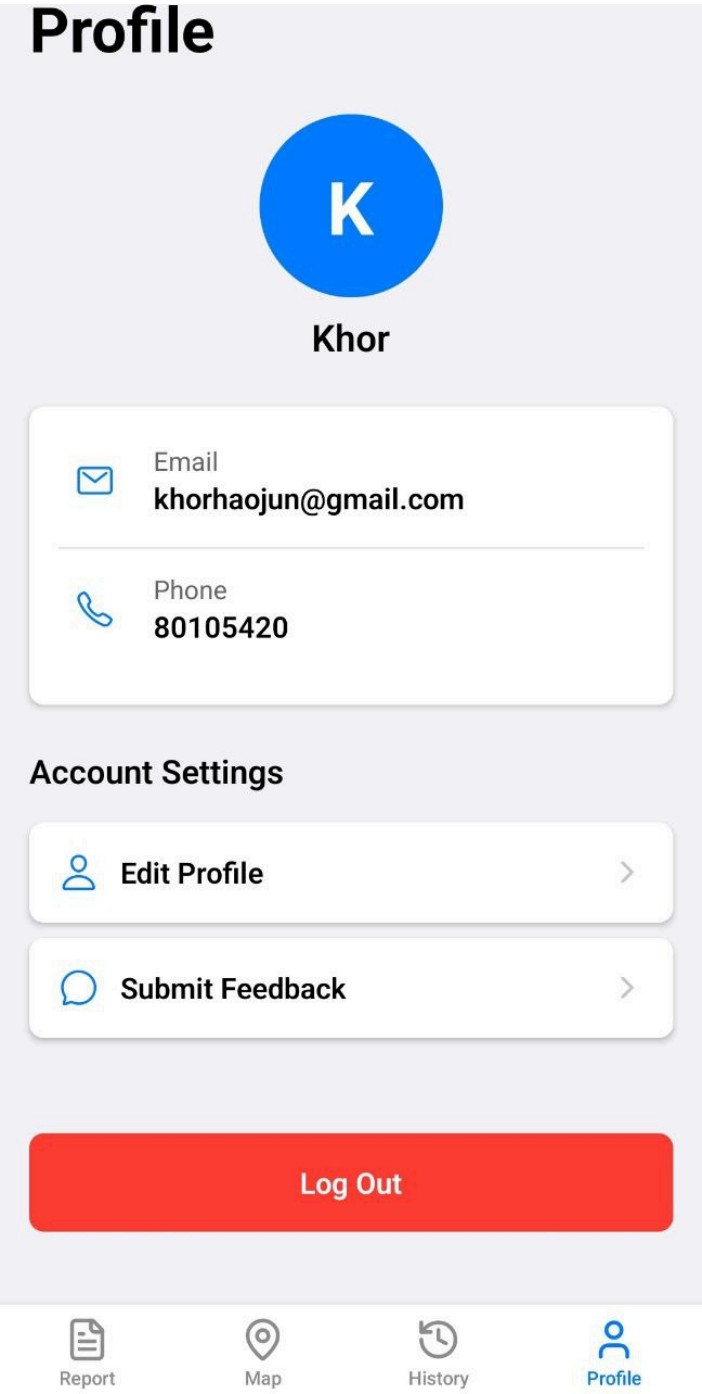
3.1.2 Signup Page



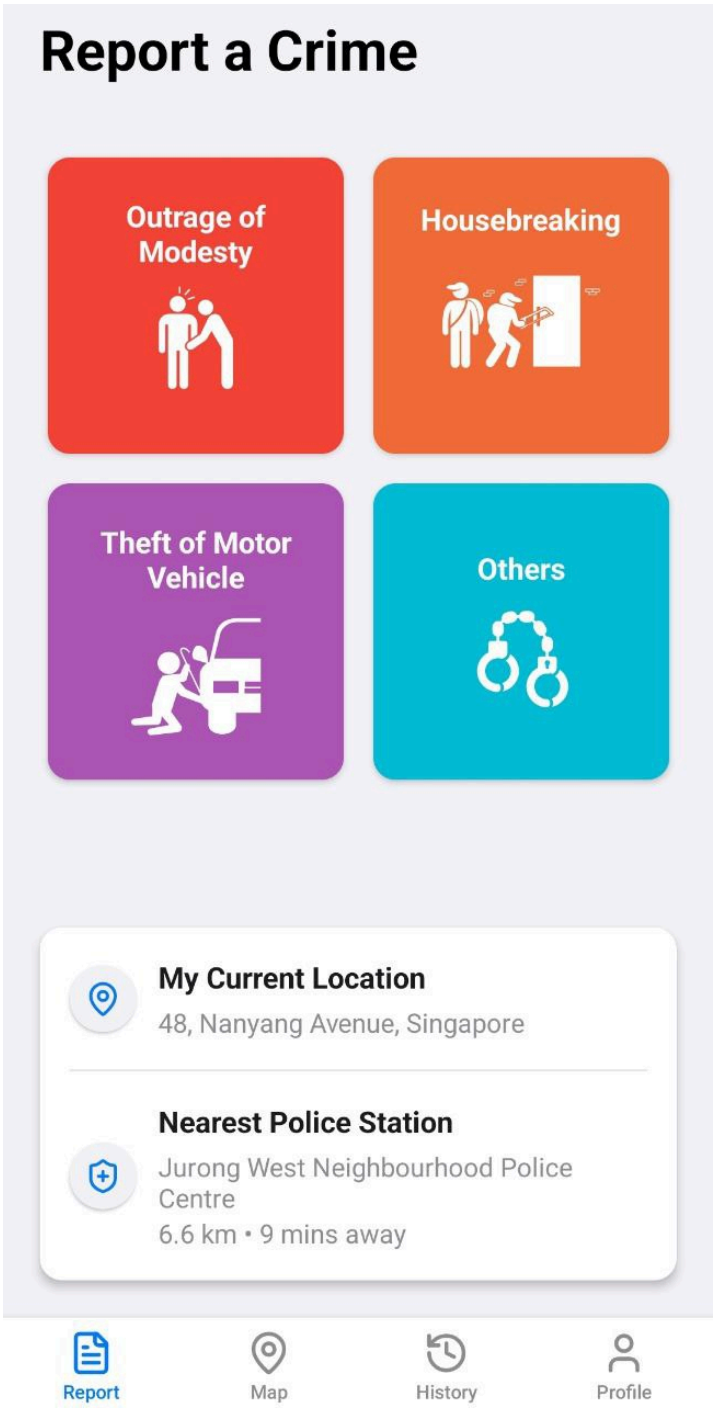
3.1.3 Edit Profile Page



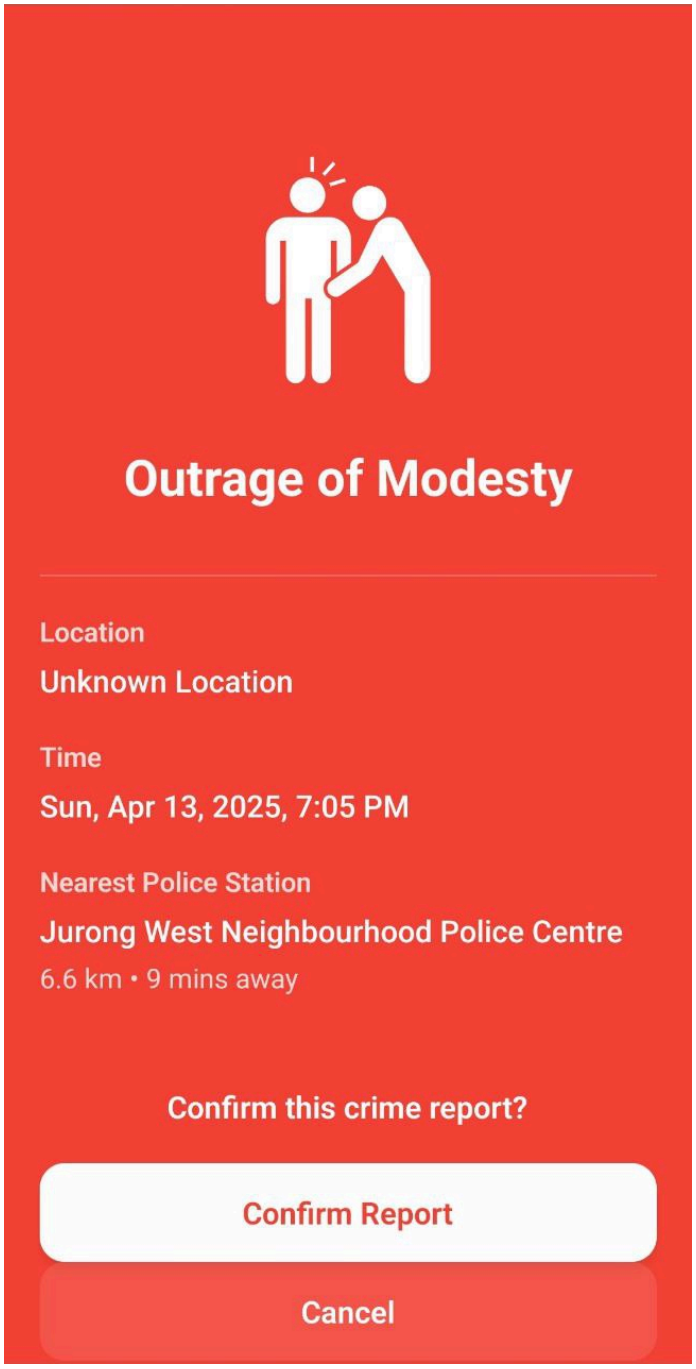
3.1.4 User Profile Page



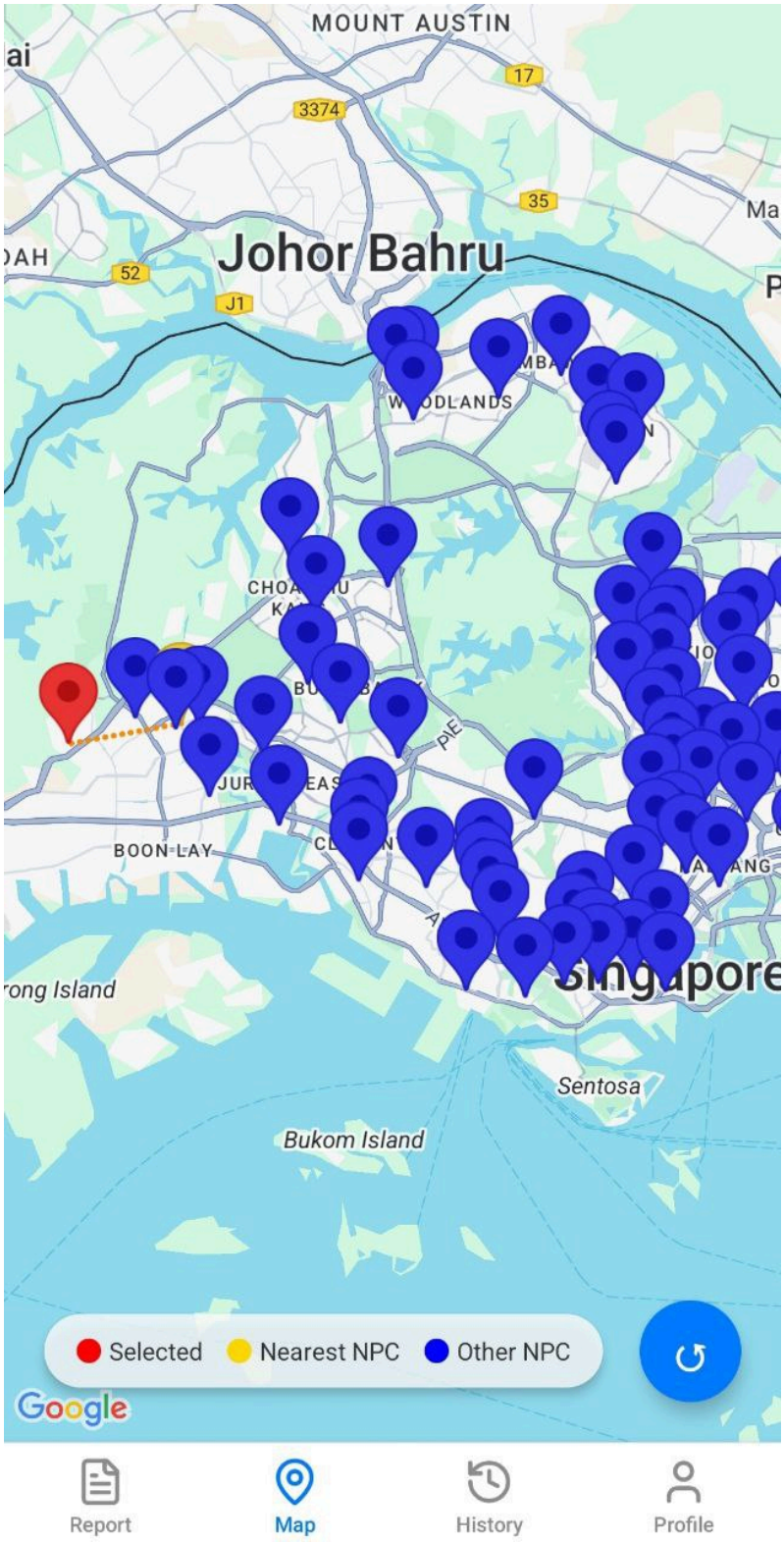
3.1.5 View Most Common Crime Types Page (Home Page)



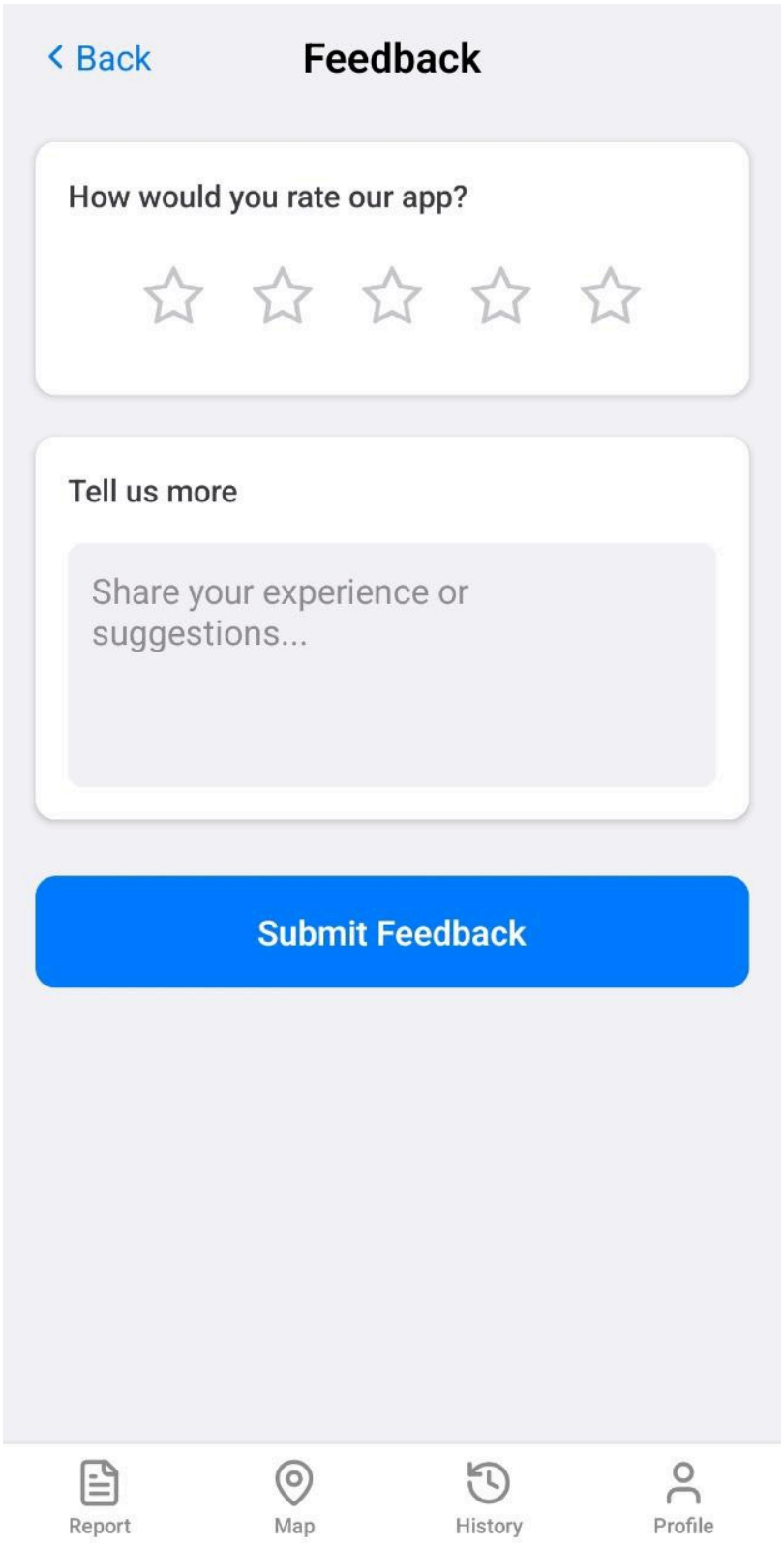
3.1.6 Submit Report Page



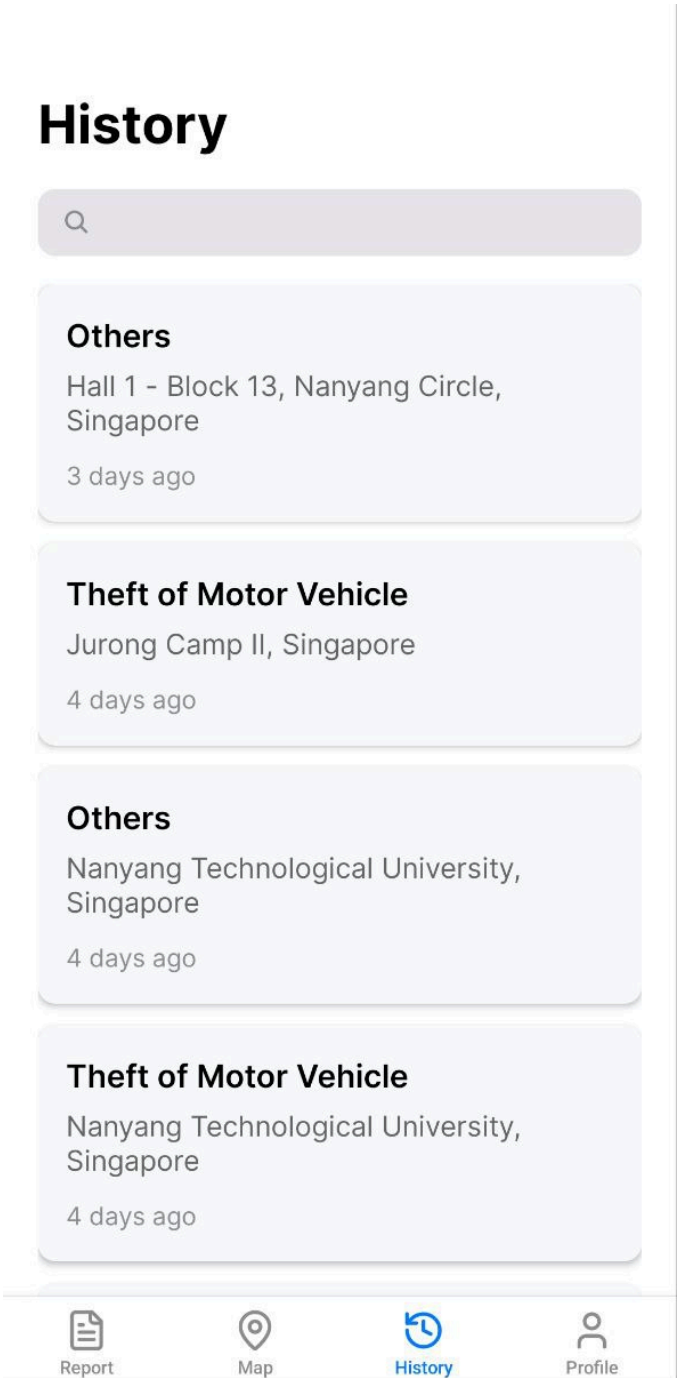
3.1.7 View Nearest Police Station Page

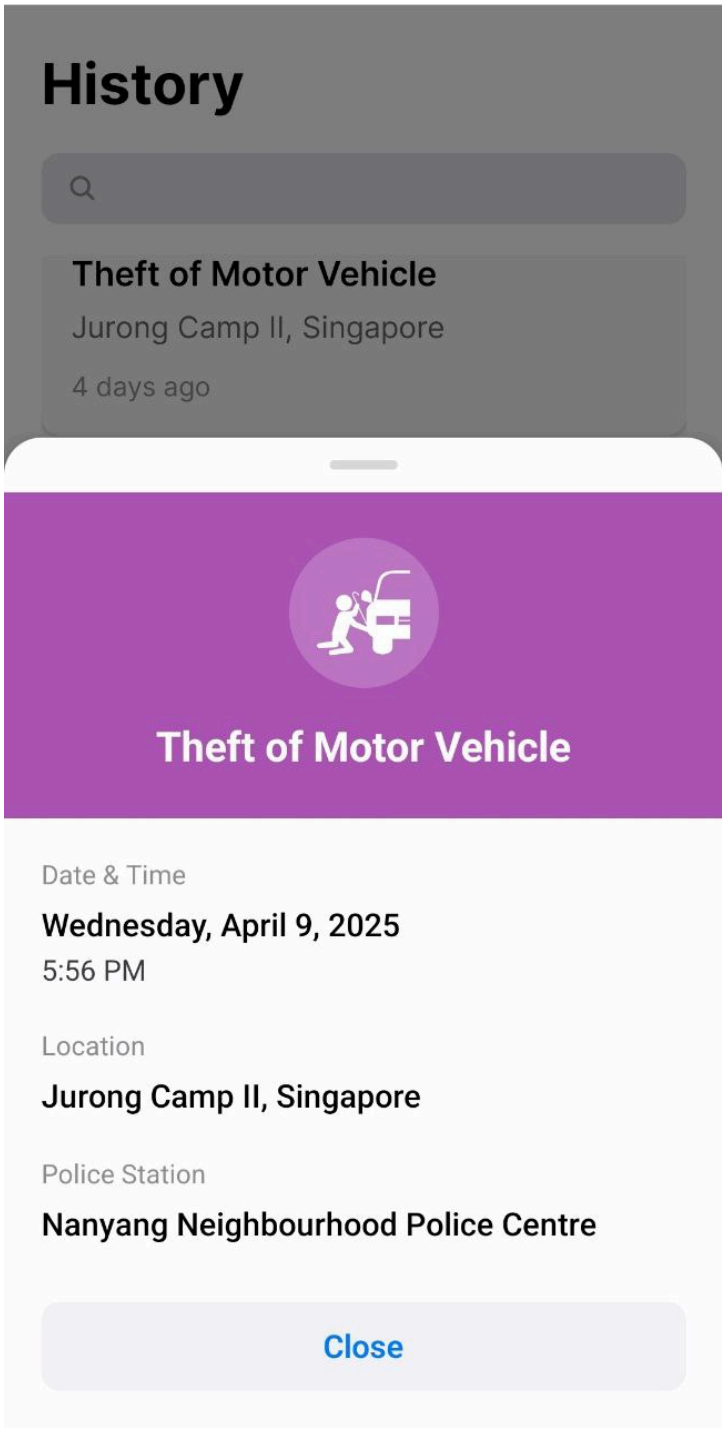


3.1.8 Submit Feedback Page



3.1.9 View History Page


History
Others
Hall 1 - Block 13, Nanyang Circle, Singapore
3 days ago
Theft of Motor Vehicle
Jurong Camp II, Singapore
4 days ago
Others
Nanyang Technological University, Singapore
4 days ago
Theft of Motor Vehicle
Nanyang Technological University, Singapore
4 days ago
Report
Map
History
Profile


History
Theft of Motor Vehicle
Jurong Camp II, Singapore
4 days ago
Theft of Motor Vehicle
Date & Time
Wednesday, April 9, 2025
5:56 PM
Location
Jurong Camp II, Singapore
Police Station
Nanyang Neighbourhood Police Centre
Close

3.2 Hardware Interfaces

This section outlines the hardware interface requirements necessary for the CrimeWatch mobile application to function as intended. These requirements are divided into client-side and server-side categories for clarity.

3.2.1 Client-side Requirements

The CrimeWatch mobile application is designed primarily for mobile devices. The following are the minimum client hardware requirements:

Device Type: Smartphone (Android or iOS)

Hardware Features Required:

GPS Module – to access the user's real-time location and find the nearest police station

Camera – for capturing photos related to incidents (if report image support is implemented)

Network Access (Wi-Fi/Cellular) – required to send reports, retrieve data from APIs, and send SMS via the backend

SMS Capability – for sending emergency alerts via Twilio integration (initiated through backend)

Optional Testing Environment: While the app can run in mobile simulators or browsers via Expo Go, it is not optimized for desktop browser use.

3.2.2 Server-side Requirements

The backend system powering CrimeWatch is hosted on cloud infrastructure. The following hardware and hosting environment requirements apply:

- Cloud Hosting Platform: Railway (or equivalent cloud provider)
- Server-Side Application:
    - Must support FastAPI to handle backend logic and RESTful APIs
    - Requires a server capable of running a Python environment
- Database Server:
    - Must support MySQL and allow cloud-based remote access by the backend server
- Third-party Integrations:
    - Access to external APIs like Google Maps and Twilio must be supported with active internet connectivity from the hosting server

3.3 Software Interfaces

3.3.1 Software Components and Versions

The *CrimeWatch* mobile application utilizes a back-end server and a database to handle all the system features implementation. The back-end server is built and implemented using the Python FastAPI framework. The *CrimeWatch* development team adopts MySQL, as the database for the mobile application.

3.3.2 Software Architecture

The *CrimeWatch* mobile application software architecture must follow the Model-View-Controller design pattern. The interface must be able to connect to a database to store persistent data in SQL format.

3.4 Communications Interfaces

The *CrimeWatch* mobile application communication architecture must follow a client-server model. Each communication must go through a REST-styled Application Programming Interface (API) provided by the back-end server. Each request must also be served over HypterText Transfer Protocol Secure (HTTPS).

Communication from client to server must invoke GET and POST requests. Communication from server to client must serve data standardized in JavaScript Object Notation (JSON) format.

4. System Features

4.1 Account Registration

4.1.1 Description and Priority

New user of *CrimeWatch* can register for an account. Upon registration, a record of the user's email address, phone, username and password is stored in the database. Any subsequent updates made by the user, such as make a report, edit profile, or submit , will be using this registered email address as reference.

Overall Priority	High

4.1.2 Stimulus/Response Sequences

Use Case ID:	UC1
Use Case Name:	Create Account

Actor:	User (Initiating actor)
Description:	The user can create an account using their name, email, and phone number. Once the account is created, the user can log in to the app to access its features.
Preconditions:	1. The device has a usable phone number and access to the Internet 2. The user has not created an account before

Postconditions:	1. The user is registered. 2. The user is logged in and redirected to the home screen.
Priority:	High
Frequency of Use:	Once for every user
Flow of Events:	1. The user selects the **"Sign Up"** option on 2. The user enters the required information, including: a) Full name (8-20 characters) b) Phone number c) Email d) Password (8-20 characters) e) Confirm Password 3. The system validates input (format, length, match) 4. Upon successful validation, the system creates the user account and logs the user into the app with a welcome message.
Alternative Flows:	1. The user submits the form with missing or incorrect details. a) The system displays an error message. b) The user corrects the errors and resubmits the form.

Exceptions:	1. The user has registered an account before. Solution: The system prompts “An account has been created” message
Includes:	
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

4.1.3 Functional Requirements

FR-1
The application shall provide the ability to create new users

FR-1.1
The application shall allow for users to enter their user details

FR-1.1.1
The application shall allow for users to enter their name

FR-1.1.1.1
The username must be between 8 and 20 alphanumeric characters

FR-1.1.2
The application shall allow for users to enter their password

FR-1.1.2.1
The password type data must be text of at least 8 characters and less than 20 characters

FR-1.1.2.2
The password character type must include at least one number

FR-1.1.2.3
The password character type must include at least one symbol

FR-1.1.2.4
The password character type must include at least one lowercase letter

FR-1.1.2.5
The password character type must include at least one uppercase letter

FR-1.1.3
The application shall allow for users to enter their mobile number

FR-1.1.3.1
The mobile number type data must be exactly 8 digits

FR-1.1.3.2
The mobile number must contain only numeric characters

FR-1.1.3.3
The mobile number must only be applicable within the area code of +65

4.2 Account Login

4.2.1 Description and Priority

Existing user of *CrimeWatch* can login to their account. App User is required to input their username and password as verification during the login process. App User must login prior to accessing all the features provided by the App.

Overall Priority	High

4.2.2 Stimulus/Response Sequences

Use Case ID:	UC2

Use Case Name:	Login

Actor:	User
Description:	Users can log in using their email and password to access personalized features.
Preconditions:	1. The user has an existing, validated account. 2. The app is installed and has internet access.
Postconditions:	1. If successful, the user is authenticated and gains access to the app.
Priority:	High
Frequency of Use:	Every time the user enters the app.
Flow of Events:	1. The user opens the app and selects the **"Log In"** option. 2. The user enters their registered email and password. 3. If the credentials are correct, the user is logged in successfully. 4. The system redirects the user to the home screen.
Alternative Flows:	1. Email and password do not match 2. The application displays an "Incorrect password" message and prompts the user to retry 3. 2. The account does not exist 4. The application displays an “Invalid account” message and prompts the user to create an account
Exceptions:	-

Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-
Actor:	User

4.2.3 Functional Requirements

FR-2
The application shall provide the ability for users to login

FR-2.1
The application shall allow for users to enter their login details

FR-2.1.1
The application shall allow a user to enter their username

FR-2.1.2
The application shall allow a user to enter their password

FR-2.2
The application shall verify the user's login details against the registered users database

FR-2.2.1
The application shall verify the user's email

FR-2.2.2
The application shall verify the user's password

FR-2.2.2.1
The application will display an error message upon three consecutive incorrect login attempts

FR-2.2.2.2

After three consecutive incorrect password attempts, the application shall implement a 30-second cooldown period before allowing another login attempt

FR-2.2.2.3

The application shall reset the cooldown upon a successful login attempt

FR-2.3

The application shall allow a user to login via facial recognition

FR-2.3.1

The application shall allow a user to scan their face

FR-2.3.2

The application shall verify a user's facial scan against the one stored in the registered user database

4.3 Logout

4.3.1 Description and Priority

App User can log out and terminate their session.

Overall Priority	Low

4.3.2 Stimulus/Response Sequences

Use Case ID:	UC3
Use Case Name:	Log out

Actor:	User

Description:	The user logs out of the application. The application ends the user session and redirects them to the login page.
Preconditions:	1. The user is logged in.
Postconditions:	The user session is terminated, and the user is redirected to the login page.
Priority:	Low
Frequency of Use:	-
Flow of Events:	1. User selects the “Logout” button. 2. The application ends the user session. 3. The system redirects the user to the login page.
Alternative Flows:	-
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

4.3.3 Functional Requirements

FR-3

The application shall provide the ability for users to log out of their accounts.

FR-3.1

The application shall allow a user to log out from their current session.

FR-3.1.1

The user shall be logged out when they click the "Log Out" button from the profile or settings screen.

FR-3.1.2

Upon log out, the user’s session shall be terminated, and they shall be redirected to the login screen.

FR-3.2

The application shall ensure that the user’s credentials are removed from the device upon log out.

FR-3.2.1

All authentication tokens (e.g., JWT) shall be cleared from local storage or device memory upon log out.

FR-3.3

The application shall automatically log out a user after a specified period of inactivity.

FR-3.3.1

The application shall log out a user after 60 minutes of inactivity, with an optional warning notification displayed 1 minute before the automatic log out.

4.4 Edit profile

4.4.1 Description and Priority

App User can edit their profile name, phone or password

Overall Priority	Low

4.4.2 Stimulus/Response Sequences

Use Case ID:	UC4
Use Case Name:	Edit profile

Actor:	User
Description:	The user edits their profile, including changing their name, phone number, and password.
Preconditions:	1. The user is logged in.
Postconditions:	The changes made are instantly reflected on the profile page
Priority:	-
Frequency of Use:	-
Flow of Events:	1. User selects the “Edit profile” button. 2. User inputs the changes they want to make.

	3. The successfully changed message is displayed upon changing.
Alternative Flows:	1. Any of the fields is empty. - Prompts the user not to leave any field empty. 2. The current password is incorrectly input. - Prompts the user to input the correct current password so that they can change their password.
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

4.4.3 Functional Requirements

FR-4
The application shall provide the ability for users to edit their profile information.

FR-4.1
The application shall allow users to update their personal details from the profile screen.

FR-4.1.1
The user shall be able to edit their name.

FR-4.1.2
The user shall be able to update their phone number.

FR-4.2
The application shall validate the input fields when the user submits profile updates.

FR-4.2.1
The application shall check that the **email** entered is in a valid email format.

FR-4.2.2

The application shall ensure the **mobile number** entered is valid, with 8 digits and adhering to the correct area code (e.g., +65 for Singapore).

FR-4.3

The application shall allow users to update their **password**.

FR-4.3.1

The user shall enter their **current password** and a **new password**.

FR-4.3.2

The new password shall follow the same validation rules as the initial password (e.g., at least 8 characters, one number, one symbol, etc.).

FR-4.4

The application shall display a success message when the profile is updated successfully.

FR-4.4.1

If the update fails, the application shall display an appropriate error message with the reason (e.g., invalid email, weak password).

FR-4.5

The application shall provide the option for users to **cancel** the profile edit operation.

FR-4.5.1

If the user cancels the edit operation, the application shall revert any unsaved changes to the original profile data.

4.5 View and Select a Crime

4.5.1 Description and Priority

User can view the crimes and select which crime that they witnessed

Overall Priority	High

4.5.2 Stimulus/Response Sequences

Use Case ID:	UC5
Use Case Name:	View and select a crime type

Actor:	User
Description:	Users can choose from the three crimes that are most common in their location and an “others” option and make a crime report.
Preconditions:	1. The user has internet access to load the crime offenses data. 2. The user is logged into the app.
Postconditions:	1. The user has successfully viewed the list of available crime offenses.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. The user taps on the "Home Page" section in the app's navigation menu. 2. The app displays a list of categorized crime offenses. 3. User selects a crime. 4. The app displays descriptions like location and time. 5. The user can navigate back to the main crime offenses list or other parts of the app.
Alternative Flows:	1. -
Exceptions:	-

Includes:	UC8: Display the most common crime type
Special Requirements:	-
Assumptions:	-
Notes and Issues:	1. -

4.5.3 Functional Requirements

FR-5

The application shall allow users to view the list of common crime offenses and select a crime to report.

FR-5.1

The application shall display the most common crimes based on the user's location.

FR-5.1.1

The list shall include the three most common crime types in the user's location and an "Others" option for crimes not listed.

FR-5.1.2

The list shall include the crime offense name, location, and time for each offense.

FR-5.2

The application shall allow users to select a crime from the list to make a report.

FR-5.2.1

When a user selects a crime, the app shall display more details about that crime, including its description, location, and time.

FR-5.2.2

The user shall be able to report the selected crime by filling out a form or selecting a predefined option for reporting.

FR-5.3

The application shall allow the user to navigate back to the crime offenses list or other sections of the app.

FR-5.3.1

The user shall be able to go back to the crime offense categories list after viewing crime details.

FR-5.3.2

The user shall be able to navigate to other parts of the app (e.g., main menu or settings) after viewing crime details.

FR-5.4

The application shall ensure that the user is logged in before accessing the crime offenses list.

FR-5.4.1

The application shall prompt the user to log in if they are not authenticated when trying to view crime offenses.

FR-5.5

The application shall display an error message if it fails to retrieve the crime offenses due to connectivity issues.

4.6 Make a Report

4.6.1 Description and Priority

App User can make a report by just one tap.

Overall Priority	High

4.6.2 Stimulus/Response Sequences

Use Case ID:	UC6
Use Case Name:	Make a report

Actor:	User, Neighbourhood Police Center (NPC)
Description:	The user can report a crime via the application, which will be sent to the nearest police station.
Preconditions:	1. The user is logged in. 2. The device has an active internet connection.
Postconditions:	1. The report is logged in the system. 2. The nearest **Neighbourhood Police Centre (NPC)** receives the report and dispatches resources if necessary.
Priority:	High
Frequency of Use:	As needed

Flow of Events:	1. The user views the crime offenses and selects a crime offense. 2. The user selects the "Make a Report" option. 3. The report is submitted to the Neighbourhood Police Centre. 4. The system notifies the user that the report has been received.
Alternative Flows:	1. If the crime type does not match the listed options, the user selects "Other" and manually describes the incident.
Exceptions:	Report Submission Fails ● If the report fails to send, the system notifies the user and allows them to retry.
Includes:	1. UC7: Send SMS notification to NPC 2. UC8: Find the nearest police station
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

4.6.3 Functional Requirements

FR-6

The application shall allow users to report a crime to the nearest Neighbourhood Police Centre (NPC).

FR-6.1

The user shall view the list of crime offenses and select one to report.

FR-6.1.1

The user shall be able to select a crime offense from the list of available crimes.

FR-6.1.2

If the selected crime offense is not listed, the user shall be able to choose the "Other" option and manually describe the crime incident.

FR-6.2

The application shall provide an option for the user to submit a report for the selected crime offense.

FR-6.2.1

The user shall have a “Make a Report” button after selecting a crime offense.

FR-6.2.2

The report shall be sent to the nearest Neighbourhood Police Centre (NPC) based on the user's location.

FR-6.3

The application shall notify the user when the crime report has been successfully submitted.

FR-6.3.1

The system shall display a confirmation message to the user indicating that the report has been received by the NPC.

FR-6.3.2

If the report submission fails, the system shall notify the user of the failure and provide an option to retry.

FR-6.4

The application shall send an SMS notification to the nearest NPC when a report is submitted.

FR-6.4.1

The system shall automatically send the SMS notification to the NPC upon receiving a report.

FR-6.4.2

The SMS shall contain the details of the reported crime (e.g., type, location, and time).

FR-6.5

The application shall find the nearest police station (NPC) based on the user's location.

FR-6.5.1

The application shall use the user’s GPS or location data to determine the nearest NPC.

FR-6.5.2

The system shall check the nearest NPC's availability to dispatch resources when a report is submitted.

4.7 Send SMS Notification to Nearest NPC

4.7.1 Description and Priority

An SMS notification is sent to the nearest Police Station (NPC) to inform them about the crime witnessed.

Overall Priority	High

4.7.2 Stimulus/Response Sequences

Use Case ID:	UC7
Use Case Name:	Send SMS notification to NPC

Actor:	User, Neighbourhood Police Center (NPC)
Description:	The user can report a crime via the application, which will be sent as an SMS notification to the nearest NPC.
Preconditions:	1. The user is logged in. 2. The user submitted a crime report
Postconditions:	1. The report is logged in the system. 2. The nearest **Neighbourhood Police Centre (NPC)** receives the report and dispatches resources if necessary.

Priority:	High
Frequency of Use:	As needed
Flow of Events:	1. The user views the crime offenses and selects a crime offense. 2. The user selects the "Make a Report" option. 3. The report is submitted to the Neighbourhood Police Centre. 4. The Neighnourhood Police Centre will receive an SMS notification regarding the crime type, location of the crime, coordinates, reporter email, and the time when the crime happened.
Alternative Flows:	-
Exceptions:	-
Includes:	UC9: Find the nearest police station
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

4.7.3 Functional Requirements

FR-7

The application shall send an SMS notification to the nearest Neighbourhood Police Centre (NPC) upon the submission of a crime report.

FR-7.1

The application shall send a detailed SMS notification to the nearest NPC when a user submits a crime report.

FR-7.1.1

The SMS notification shall include the following details:

- Type of crime.
- Location of the crime (address or nearby landmark).
- Coordinates (latitude and longitude) of the crime scene.
- The email address of the report's author (user).
- Time when the crime occurred.

FR-7.2

The application shall use the user’s location to determine the nearest Neighbourhood Police Centre (NPC).

FR-7.2.1

The system shall automatically detect the user’s location through GPS or location services.

FR-7.2.2

The system shall identify and contact the nearest NPC based on the user’s current location.

FR-7.3

The application shall ensure that the crime report is logged into the system once the SMS notification is successfully sent.

FR-7.3.1

The system shall log the report in the database with the status "Report Sent to NPC" after the SMS is dispatched.

FR-7.4

The application shall notify the user once the SMS notification is successfully sent to the nearest NPC.

4.8 Display The Most Common Crime Types

4.8.1 Description and Priority

The app shows three of the most common crime types around their location for the user to lodge a report.

Overall Priority	High

4.8.2 Stimulus/Response Sequences

Use Case ID:	UC8
Use Case Name:	Display the most common crime type

Actor:	User
Description:	The system displays the most common major offenses in that area for the user to choose.
Preconditions:	1. The user wants to report a crime.

Postconditions:	The system will reconfirm with the user if the user wishes to report the crime.
Priority:	High
Frequency of Use:	Whenever the user wants to report a crime
Flow of Events:	1. The user wants to report a crime. 2. The system accesses the Cases Recorded For Selected Major Offences Annual database to determine some popular major offences. 3. The system displays some of the most common crime types in the user area by using the **Access User's Location Data (UC9)**.
Alternative Flows:	The user selects "others" to report other offenses not available in the menu.
Exceptions:	-
Includes:	UC9: Access User's Location Data
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

4.8.3 Functional Requirements

FR-8

The application shall display the most common major offenses in the user's area when the user wants to report a crime.

FR-8.1

The system shall access the Cases Recorded For Selected Major Offences Annual database to identify the most common crime types in the user's location.

FR-8.2

The system shall use the user's location data to retrieve the most common major offenses in the area.

FR-8.2.1

The system shall use the Access User's Location Data (UC9) to determine the user's current location.

FR-8.3

The system shall display the most common crime types based on the retrieved data in the user's area.

FR-8.3.1

The list of displayed offenses shall be sorted by the frequency of occurrence in the user's area.

FR-8.3.2

The system shall allow the user to choose from the displayed list of offenses.

FR-8.4

The system shall provide an option for the user to select "Others" to report a crime type not listed among the displayed offenses.

FR-8.4.1

The user shall be able to manually describe the crime in the "Others" option.

FR-8.5

The system shall reconfirm with the user whether they want to proceed with the selected crime report.

4.9 Find Nearest Police Station

4.9.1 Description and Priority

The app will determine which is the nearest police station based on the user's location or the location that the user selected

Overall Priority	High

4.9.2 Stimulus/Response Sequences

Use Case ID:	UC9
Use Case Name:	Find nearest police station

Actor:	Google Maps API
Description:	The system finds the nearest police station to the user from a static dataset using basic distance logic.
Preconditions:	1. The app has access to the user's real-time location. 2. The system has access to police station location datasets.
Postconditions:	1. If successful, the nearest police station is informed and can respond to the incident. 2. If unsuccessful, the user may need to report the crime manually or retry.
Priority:	High

Frequency of Use:	Whenever crime reporting
Flow of Events:	1. The user selects a crime type from the automatic contact grid. 2. The system includes **Access User's Location Data (UC10)** to refine crime selection. 3. The system checks the police station dataset and identifies the nearest station. 4. The police station receives the alert and acknowledges the request. 5. The system notifies the user that the report has been sent successfully.
Alternative Flows:	1. The user manually finds the police stations with the location specified.
Exceptions:	
Includes:	UC10: Access User's Location Data
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

4.9.3 Functional Requirements

FR-9

The application shall find the nearest police station based on the user's real-time location.

FR-9.1

The system shall access the user's real-time location to determine their position.

FR-9.1.1

The system shall use the Access User’s Location Data (UC10) to retrieve the user’s current location.

FR-9.2

The system shall access a dataset of police station locations.

FR-9.3

The system shall calculate the nearest police station based on the user’s location and the police station dataset.

FR-9.4

The system shall notify the user when the nearest police station has been successfully identified.

FR-9.5

The system shall send the report to the identified police station once it's determined to be the nearest.

FR-9.6

The system shall notify the user that the report has been successfully sent to the nearest police station.

4.10 Access User's Location Data

4.10.1 Description and Priority

The app will get the user's current location data including coordinates with latitude and longitude.

Overall Priority	High

4.10.2 Stimulus/Response Sequences

Use Case ID:	UC10
Use Case Name:	Access User's Location Data

Actor:	Google Maps API
Description:	The system retrieves the user's location for crime reporting and emergency response.
Preconditions:	1. The user has granted location permissions. 2. The device's GPS is enabled.
Postconditions:	1. The location is retrieved and included in reports or emergency calls.
Priority:	High
Frequency of Use:	Whenever a crime is reported

Flow of Events:	1. The application requests access to the user's location. 2. The user grants permission. 3. The system retrieves the location and updates the report.
Alternative Flows:	1. If GPS is not enabled, the system prompts the user to enable GPS. 2. If the user denies location access, the system prompts manual entry of location details.
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

4.10.3 Functional Requirements

FR-10

The application shall retrieve the user's location for crime reporting and emergency response.

FR-10.1

The system shall request access to the user's location.

FR-10.1.1

The system shall prompt the user to grant location permissions for the app.

FR-10.2

The system shall retrieve the user's real-time location once permission is granted and the GPS is enabled.

FR-10.3

The system shall include the retrieved location in the crime report or emergency call.

FR-10.4

The system shall notify the user if the location retrieval was unsuccessful and provide alternative options.

4.11 Submit Feedback

4.11.1 Description and Priority

App User can submit feedback with ratings and messages.

Overall Priority	Low

4.11.2 Stimulus/Response Sequences

Use Case ID:	UC11
Use Case Name:	Submit Feedback

Actor:	User

Description:	Allows users to provide feedback about police response or the app.
Preconditions:	1. The user has previously used the app's services.
Postconditions:	The feedback is acknowledged by the system and sent to the Neighbourhood Police Center.
Priority:	Low
Frequency of Use:	Occasionally
Flow of Events:	1. The user selects "Submit Feedback" from the menu. 2. The system prompts for ratings and comments. 3. The user submits feedback. 1. The feedback is logged and sent to law enforcement. 2. The system confirms successful submission.
Alternative Flows:	**User submits feedback without rating** ● The system prompts the user to provide a rating before submission. **User submits feedback without message** ● The system prompts the user to provide a message before submission.
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-

Notes and Issues:	-

4.11.3 Functional Requirements

FR-11

The application shall allow users to provide feedback about police response or the app.

FR-11.1

The system shall provide an option for the user to access the feedback feature from the app's menu.

FR-11.2

The system shall prompt the user to provide a rating and a comment for their feedback.

FR-11.3

The system shall allow the user to submit their feedback once both the rating and comment are entered.

4.12 View History

4.12.1 Description and Priority

App User can view the past reports that they made in the history page

Overall Priority	Medium

4.12.2 Stimulus/Response Sequences

Use Case ID:	UC12
Use Case Name:	View History
Actor:	User
Description:	The user can view their past reports and interactions with law enforcement.
Preconditions:	1. The user logged in.
Postconditions:	1. The user can access their report history.
Priority:	Medium
Frequency of Use:	Occasionally
Flow of Events:	1. The user selects "View History" from the menu. 2. The system retrieves previous reports from the **User Database**.

	3. The user can browse past reports and police responses.
Alternative Flows:	**No previous reports exist** ● The system displays a message: “No report history found.”
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

4.12.3 Functional Requirements

FR-12

The application shall allow users to view their past reports and interactions with law enforcement.

FR-12.1

The system shall provide an option for the user to access the "View History" feature from the app's menu.

FR-12.2

The system shall retrieve previous reports and interactions from the User Database.

FR-12.3

The system shall display the user's past reports.

5. Other Nonfunctional Requirements

5.1. Performance Requirements

- **Response Time:** The application must provide a response to user actions (such as submitting a crime report or loading the crime offenses list) within 2 seconds under normal load conditions.
- **Throughput:** The system should handle up to 1000 concurrent users without significant degradation in performance.
- **Data Retrieval Time:** Crime reports or history should be retrievable within 5 seconds after the user requests it.

5.2. Usability Requirements

- **User Interface Design:** The mobile application should have an intuitive, user-friendly interface that requires minimal training for first-time users.

5.3. Reliability Requirements

- **Availability:** The system must be available 99.9% of the time, excluding planned maintenance windows.

- **Error Rate:** The error rate should not exceed 0.1% of total user interactions.

- **Backup and Recovery:** In case of system failure, data must be recoverable within 30 minutes.

5.4. Security Requirements

- **Data Encryption:** All user data, including personal information and crime reports, must be encrypted both at rest and in transit using strong encryption methods (e.g., AES-256, TLS).

- **Authentication:** The system must support two-factor authentication (2FA) for user logins.

- **Authorization:** Users must have role-based access control (RBAC), ensuring that only authorized personnel (such as law enforcement) can access sensitive data.

5.5. Scalability Requirements

- **Horizontal Scalability:** The system should support horizontal scaling to accommodate increasing users or requests. Additional servers should be able to be added to handle higher loads without significant downtime.

- **Database Scalability:** The database should support sharding or partitioning to ensure high availability and fast access as the amount of data grows.

5.6. Maintainability Requirements

- **Code Modularity:** The codebase should be modular, following established coding standards, making it easy for developers to maintain, update, or debug.

- **Logging:** The application must provide detailed logs of critical events for debugging and auditing purposes.

- **Documentation:** The application's source code and architecture must be well-documented for future development and maintenance.

Appendix A: Data Dictionary

Glossary for CrimeWatch	
Term	**Definition**
Account	A registered user's profile including personal information like name, email, and phone.
API	Backend endpoints that handle client requests such as login, signup, report submission, etc.
Application	The mobile app used to report crimes, view locations, and interact with police data.
AsyncStorage	Local storage mechanism in React Native used to save tokens securely on the device.
authContext	React context that stores and provides authentication state throughout the frontend.
authService	Frontend service that manages user authentication (login/signup/token validation).
AlertsScreen	Displays a list of past crime reports submitted by the logged-in user.

Boundary Class	UI-facing class or screen that interacts directly with the user.
CrimeTypeGrid	UI component displaying a grid of crime categories that users can report.
CrimeReportModal	A modal screen that allows users to confirm report submission details.
CrimeType	An entity representing the title, ID, color, and icon of a crime category.
crimeReportService	Frontend service that submits crime reports, fetches top crimes, and gets nearest station and user email.
Control Class	Coordinates business logic and interacts with entities and UI (boundaries).
DB Connection	Refers to get_db_connection() used to connect to the MySQL database.
EditProfileScreen	UI screen allowing users to change their name, phone number, or password.
Entity Class	Core data structures used across the app (e.g., Report, Location, CrimeType).

FeedbackScreen	UI screen where users submit ratings and feedback messages.
FeedbackRequest	JSON object sent to the backend containing rating and message for user feedback.
findNearestPoliceStation	Use case that returns the closest NPC based on the user's coordinates.
GeoJSON	Geospatial file format used in the app to store and parse police station location data.
index.tsx (Report Screen)	Main screen where user selects a crime, views location, and submits report.
Location	Entity holding GPS information (latitude, longitude, and human-readable address).
locationContext	React context that fetches current device location and converts to a readable address.
LoginResponse	Object returned from successful login API call. Contains token and user info.
LoginScreen	Initial screen where user enters email and password to access the app.

MapScreen	Displays a map with the user's location and police stations. Allows user to manually change location.
Modal	A popup overlay UI element used to get confirmation from user (e.g., for submitting reports).
Most Common Crime	A statistical result showing which crimes occur most frequently in a given area.
MySQLDatabase	Backend SQL database used to store persistent data such as users, reports, and feedback.
NearestStation	JSON object returned by backend with NPC name and division code closest to the user.
NPCGeoJSONdata	The police station dataset file used in both backend and frontend for distance calculations and map display.
policeDataService	Frontend utility to extract name, phone, and coordinates from raw police station GeoJSON data.
policeStationsData	Parsed data object used to display all NPCs on the map.
ProfileScreen	Displays name, email, phone, and options to logout, give feedback, or go to edit profile.

RankingController	Backend logic that queries top crime types based on location.
Report	Main entity for a submitted report — includes type, time, coordinates, and station.
ReportScreen	Common alias for index.tsx, where the reporting process begins.
Reverse Geocoding	Converting coordinates (latitude, longitude) into a human-readable address.
SMSController	Backend service that formats and sends SMS notifications to the nearest NPC upon successful crime report.
SignupScreen	UI screen to register a new user with email, name, phone number, and password.
Submit Feedback	The process of sending user ratings and comments to the backend.
useLocation()	Custom hook to retrieve and access the locationContext.
ValidationResult	API response returned when validating a user's email and password.
View Crime History	Allows users to see a timeline or list of their past crime reports.
View Crime Offenses	Feature to see common crime types in a region based on division.

A. App Architecture Terms	
Term	**Definition**
Boundary Class	A user interface component that interacts directly with the user (e.g., screens like LoginScreen, FeedbackScreen).
Control Class	Handles business logic and coordination between UI and data (e.g., authService, crimeReportService).
Entity Class	Represents core data used across the system (e.g., CrimeType, Report, FeedbackRequest).
BCE Model	A design pattern dividing the app into Boundary, Control, and Entity layers based on use case behavior.
AsyncStorage	External local storage on the device used for persisting user tokens.
API	A backend endpoint that handles frontend requests for login, signup, crime reporting, etc.
B. Frontend Context & Services	
Term	**Definition**
authContext	React Context that manages authentication state and user token storage.
locationContext	React Context that fetches and stores the current GPS coordinates and reverse geocoded address of the user.
authService	Frontend service that interacts with user login, signup, and token validation APIs.

crimeReportService	Frontend service that handles report submission, fetches top crimes, and nearest police station.
policeDataService	Extracts name, tel, type, and coordinates from GeoJSON police station entries.

C. Frontend Screens / Components

Term	Definition
LoginScreen	UI screen for user login. Interacts with authService.
SignupScreen	UI screen for account creation. Interacts with authService.
EditProfileScreen	UI screen for editing name, phone, and password.
FeedbackScreen	Screen that allows users to submit app-related feedback.
AlertsScreen	Displays a user's past crime report history.
MapScreen	Shows user location and nearest police stations on a map.
Index	Report screen file. Handles report UI logic and integrates CrimeTypeGrid, LocationInfo, and CrimeReportModal.
CrimeTypeGrid	Grid component showing selectable crime types.
LocationInfo	UI component showing current user location and nearest police station.
CrimeReportModal	Modal popup shown after selecting a crime type to confirm report.

D. Data Structures / Entities

Term	Definition

CrimeType	Defines a type of crime with id, title, icon, and color.
Report	A crime report including type, location, police station, and timestamp.
Location	Represents a user's GPS data with latitude, longitude, and name.
NearestStation	Nearest police station info fetched from backend (name, divcode).
FeedbackRequest	JSON object containing user feedback message and rating.
LoginResponse / ValidationResult / CheckUserResponse	Responses returned from authentication APIs.

E. Backend Services & Controllers

Term	Definition
auth.py	Backend controller handling login, signup, user validation.
location.py	Returns nearest police station based on GPS coordinates.
ranking.py	Backend controller to fetch most reported crimes in division.
sms.py	Sends SMS to nearest NPC on report confirmation.

F. External Resources

Term	Definition
NPCGeoJSONdata	GeoJSON file of police stations (SingaporePoliceForceEstablishments2018GEOJSON.geojson). Used to locate and extract NPC coordinates and metadata.

MySQLDatabase	SQL database used to store user info, crime reports, and feedback. Connection managed by get_db_connection() from db.py.
policeStationsData	Parsed police station dataset from the GeoJSON file for frontend map display.
extractPoliceStationInfo()	Parses a police station feature and extracts name, coordinates, tel, and type.

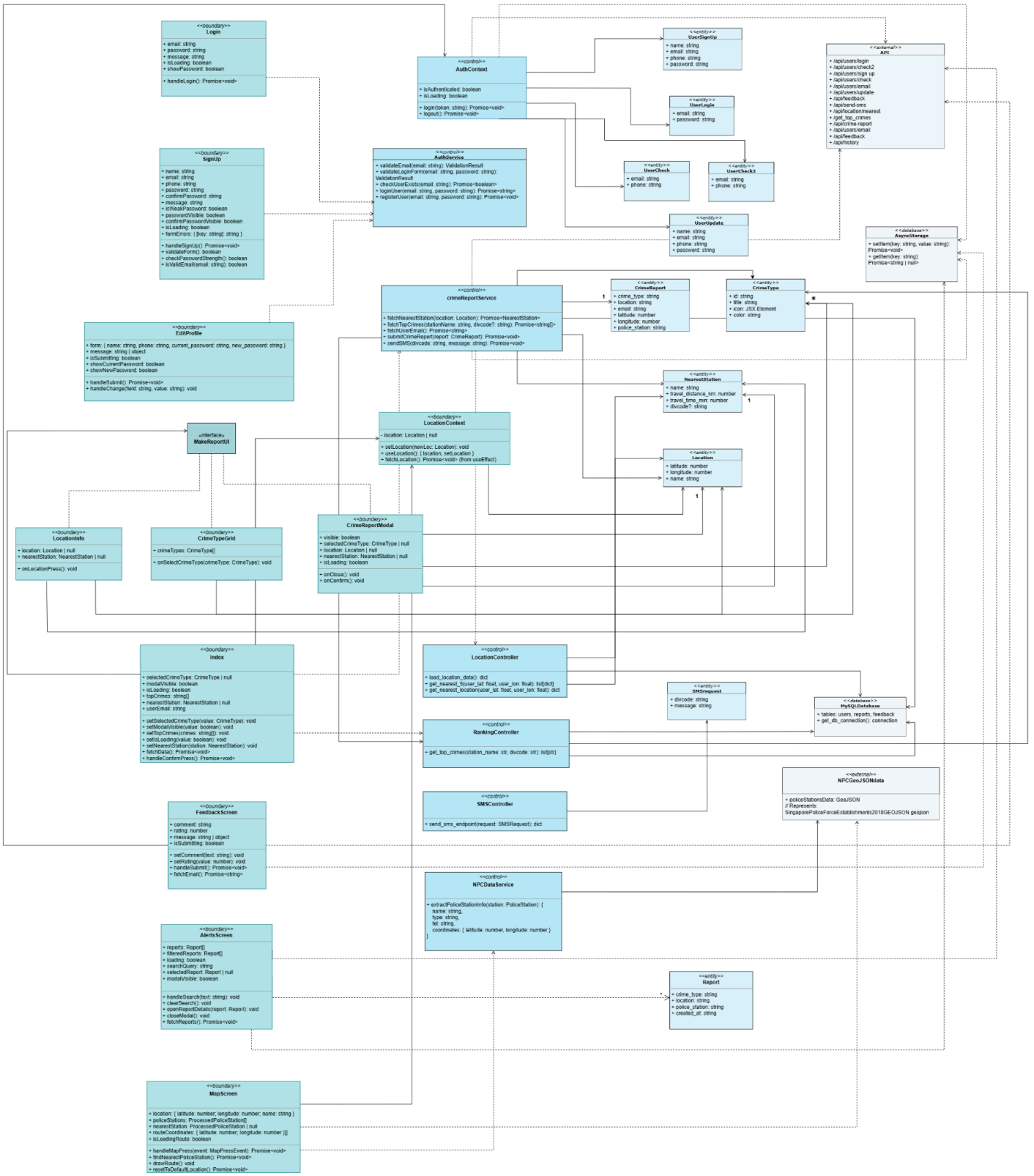
G. App Logic Concepts

Term	**Definition**
Find Nearest Police Station	Calculates which police station is geographically closest to the user. Uses backend and policeDataService.
Display Most Common Crime Type	Shows the top reported crimes in the user's area. Uses ranking controller and crimeReportService.
Send SMS Notification	Sends SMS with location and crime details to the NPC. Uses sms controller.
User Feedback Submission	Allows users to submit textual and rating feedback to backend.
History Retrieval	Loads previous reports submitted by user. Displays in AlertsScreen.

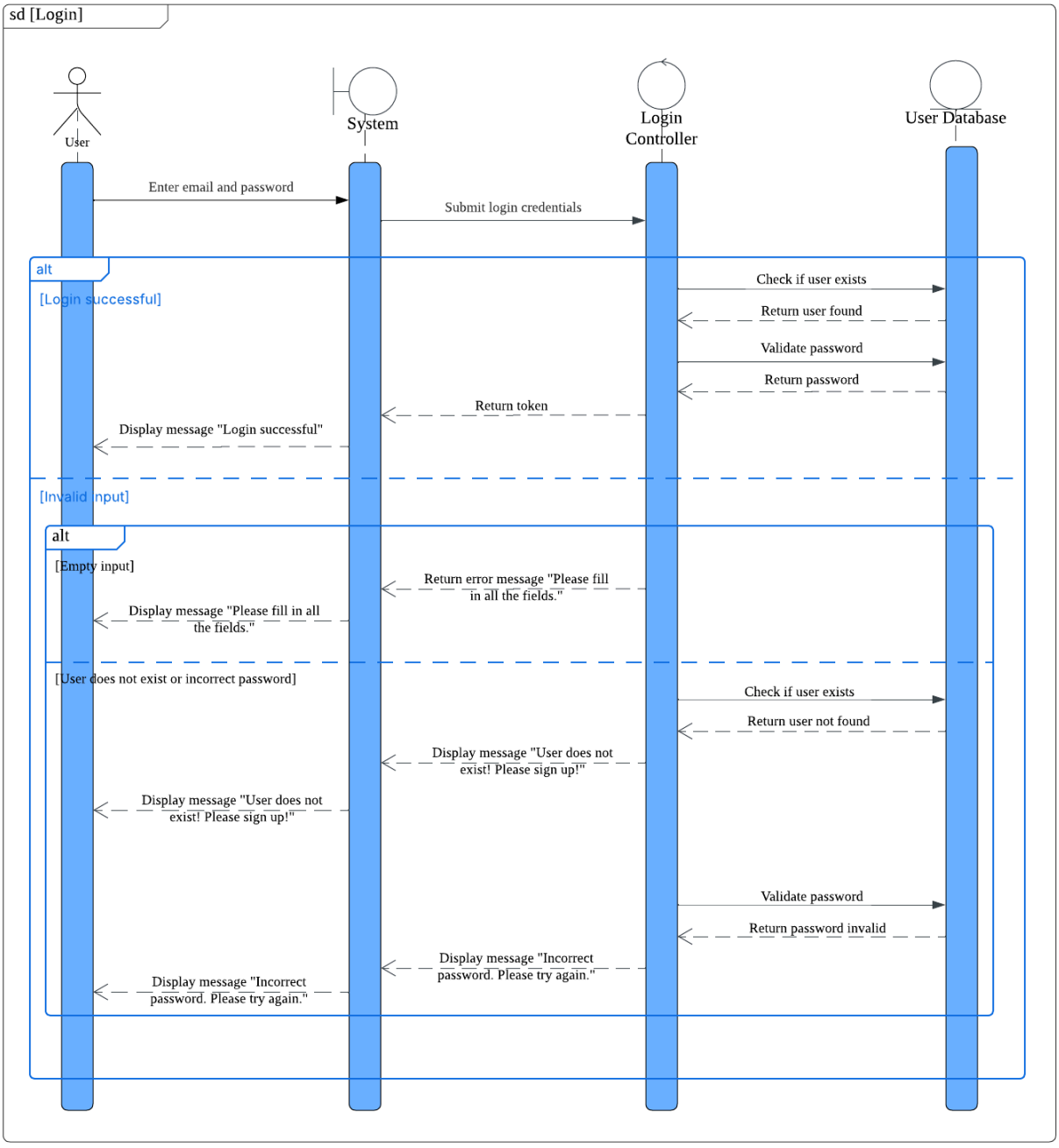
Appendix B: Analysis Models

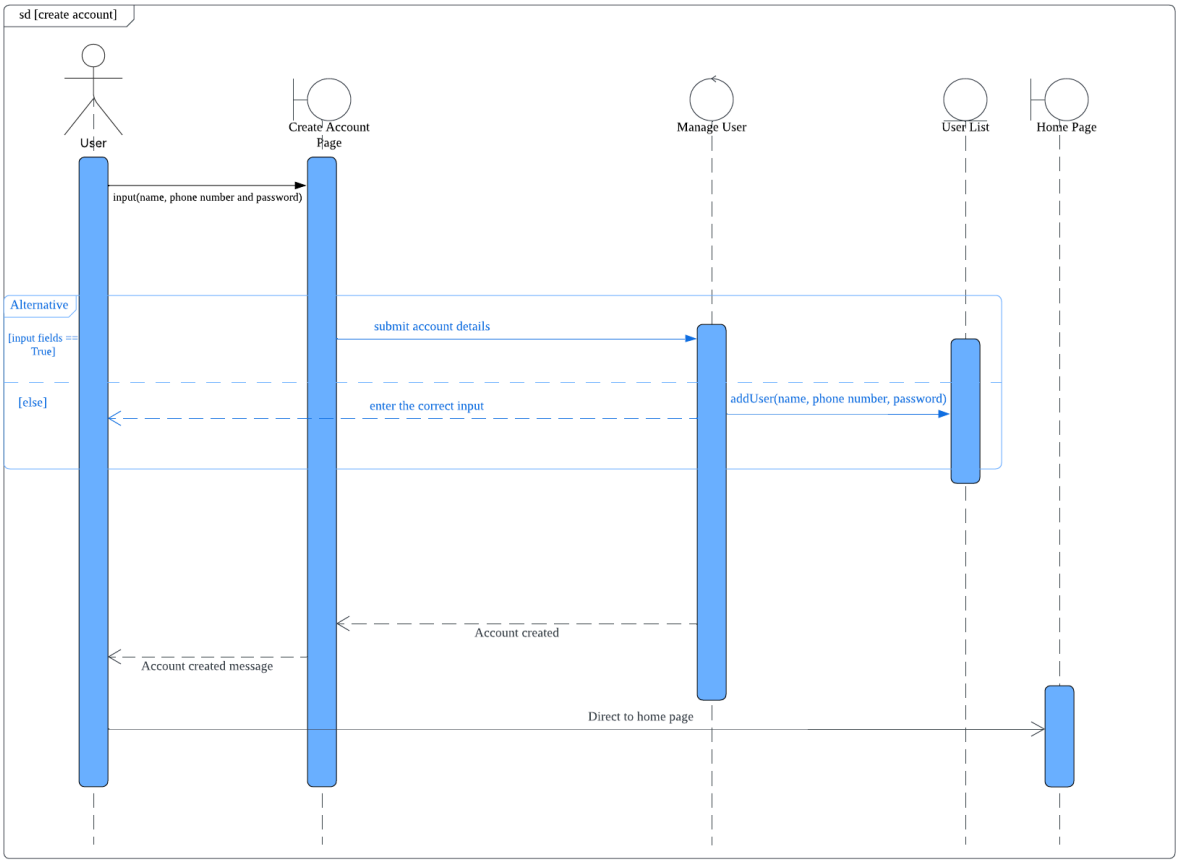
B.1 Use Case Model

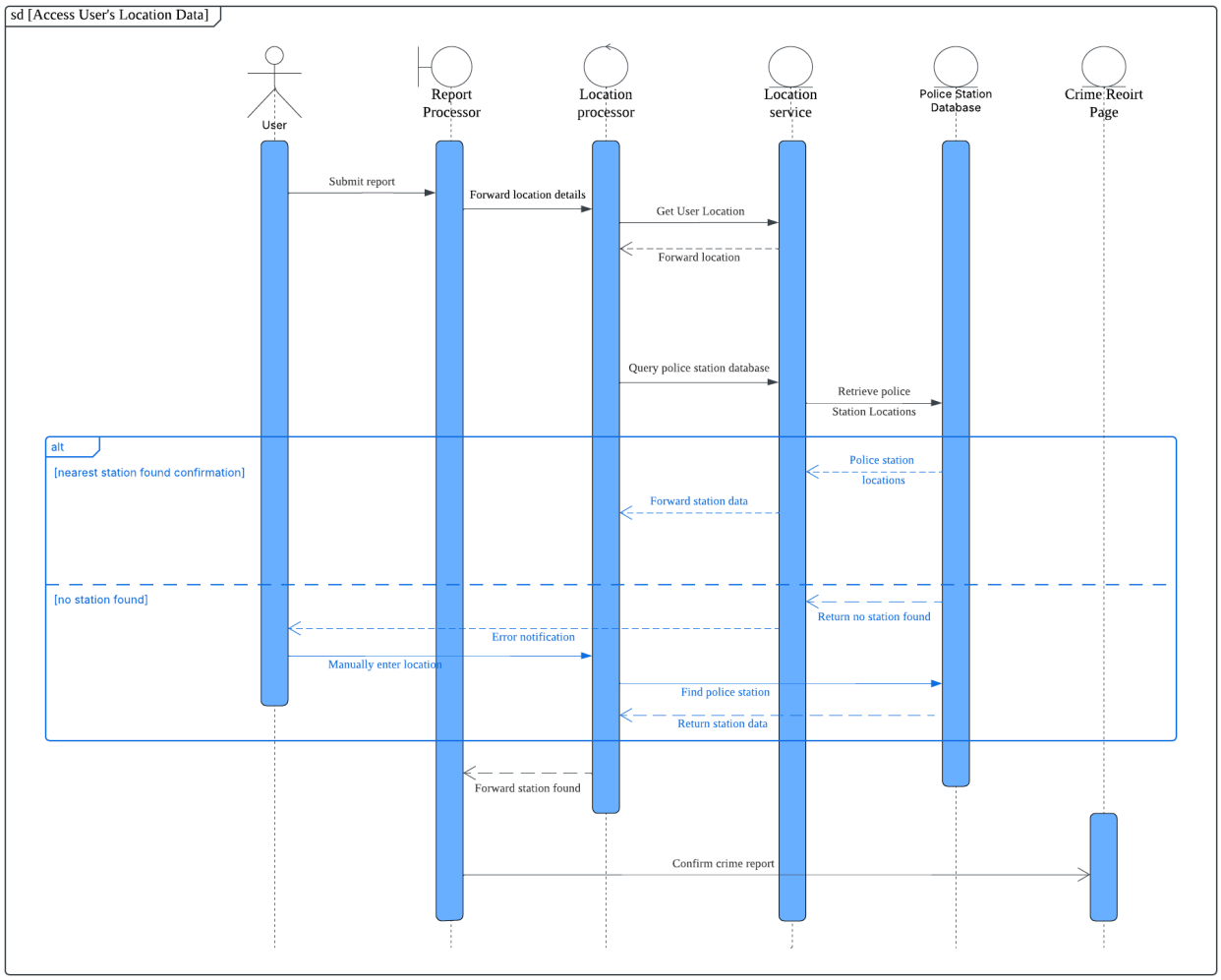
B.2 Class Diagram

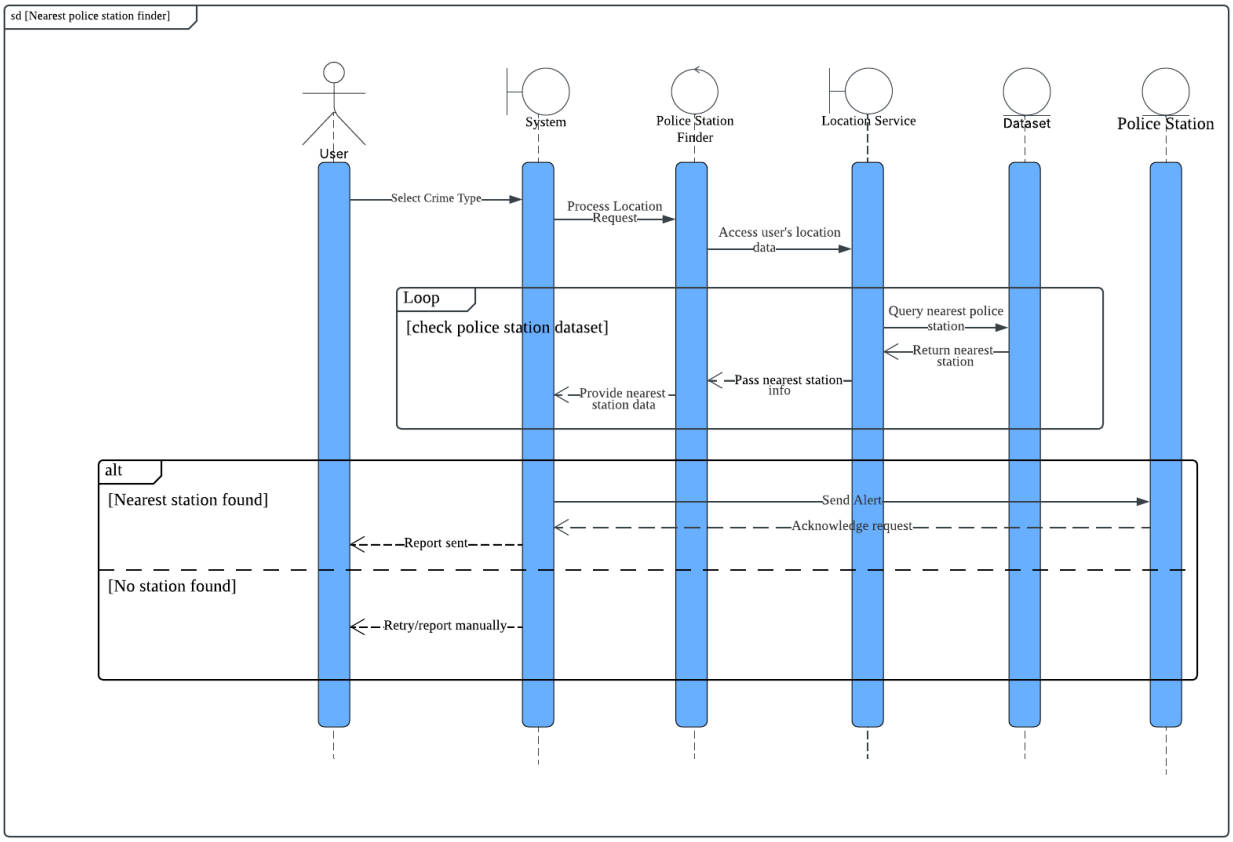


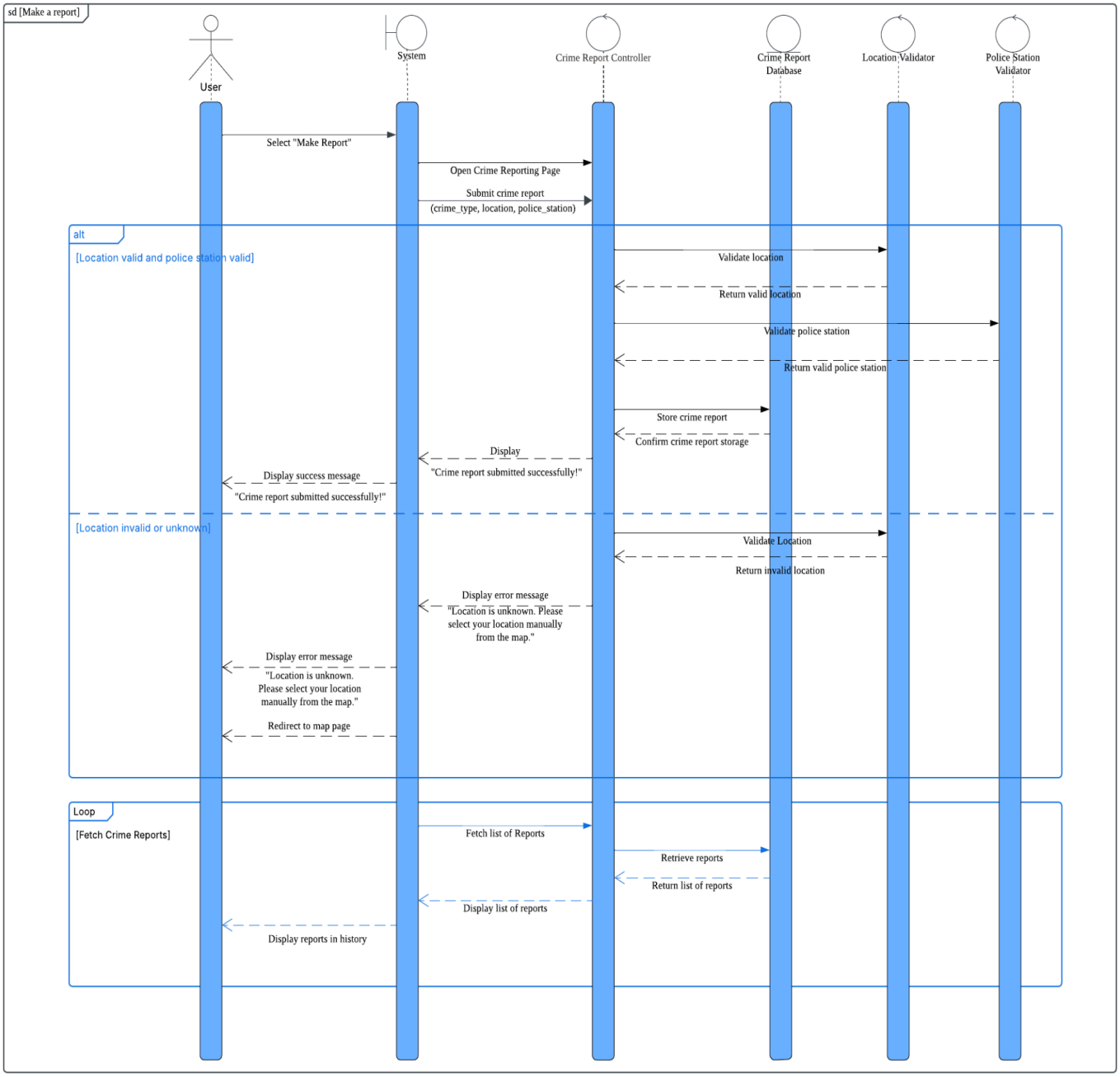
B.4 Sequence Diagrams

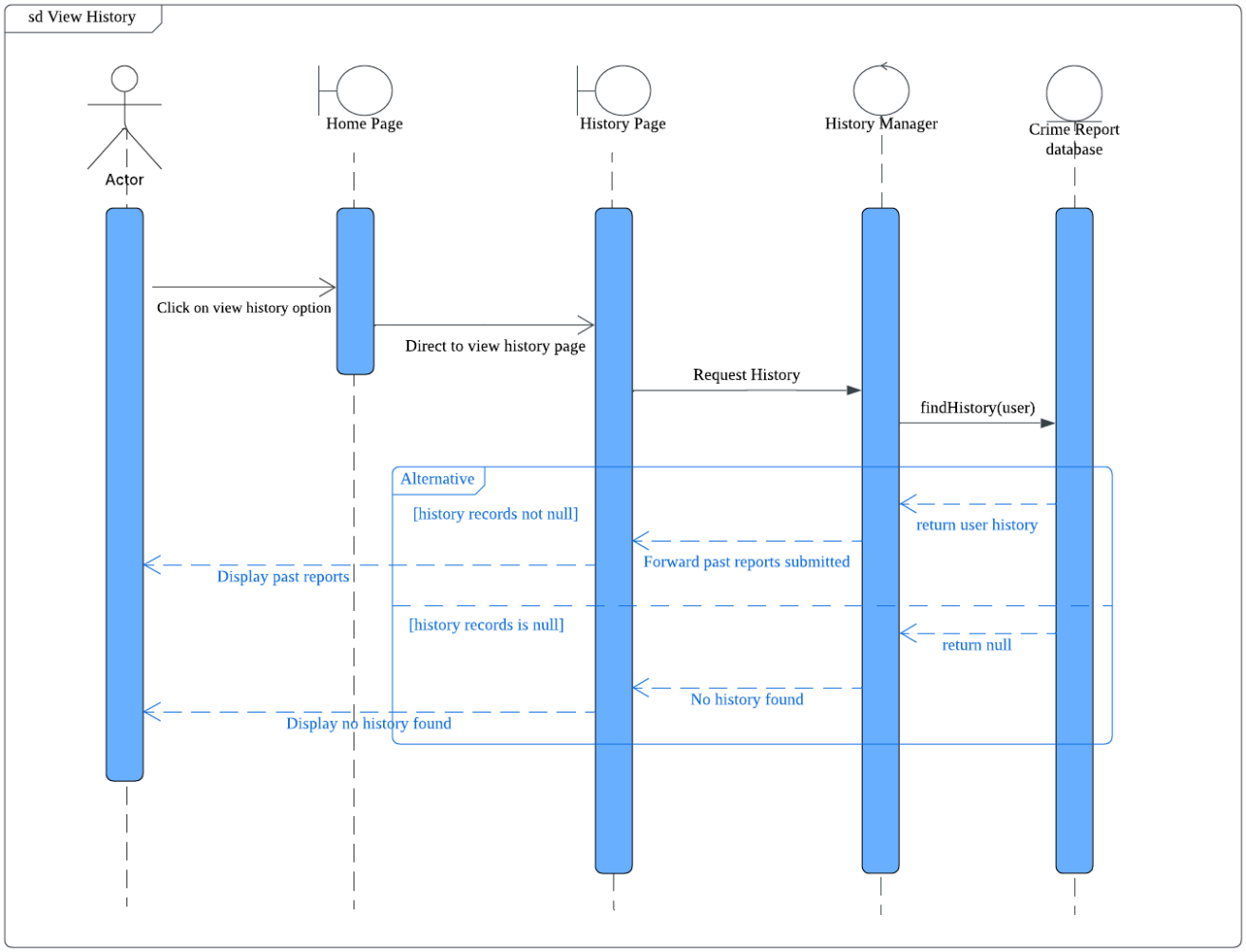


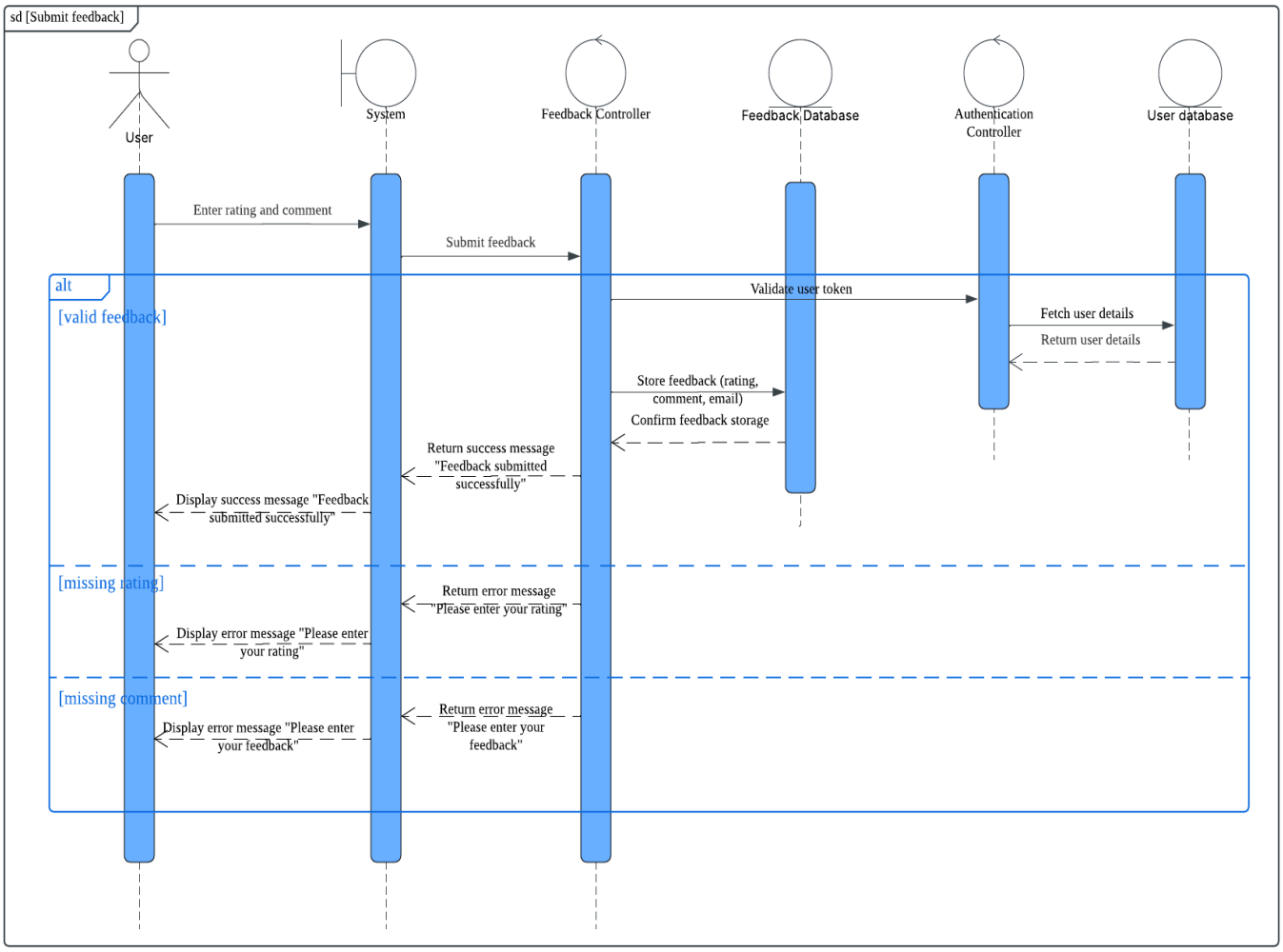

sd [create account]
User
Create Account Page
Manage User
User List
Home Page
input(name, phone number and password)
Alternative
[input fields == True]
submit account details
[else]
enter the correct input
addUser(name, phone number, password)
Account created
Account created message
Direct to home page


sd [Access User's Location Data]
User
Report Processor
Location processor
Location service
Police Station Database
Crime Reoirt Page
Submit report
Forward location details
Get User Location
Forward location
Query police station database
Retrieve police Station Locations
alt
[nearest station found confirmation]
Police station locations
Forward station data
[no station found]
Return no station found
Error notification
Manually enter location
Find police station
Return station data
Forward station found
Confirm crime report

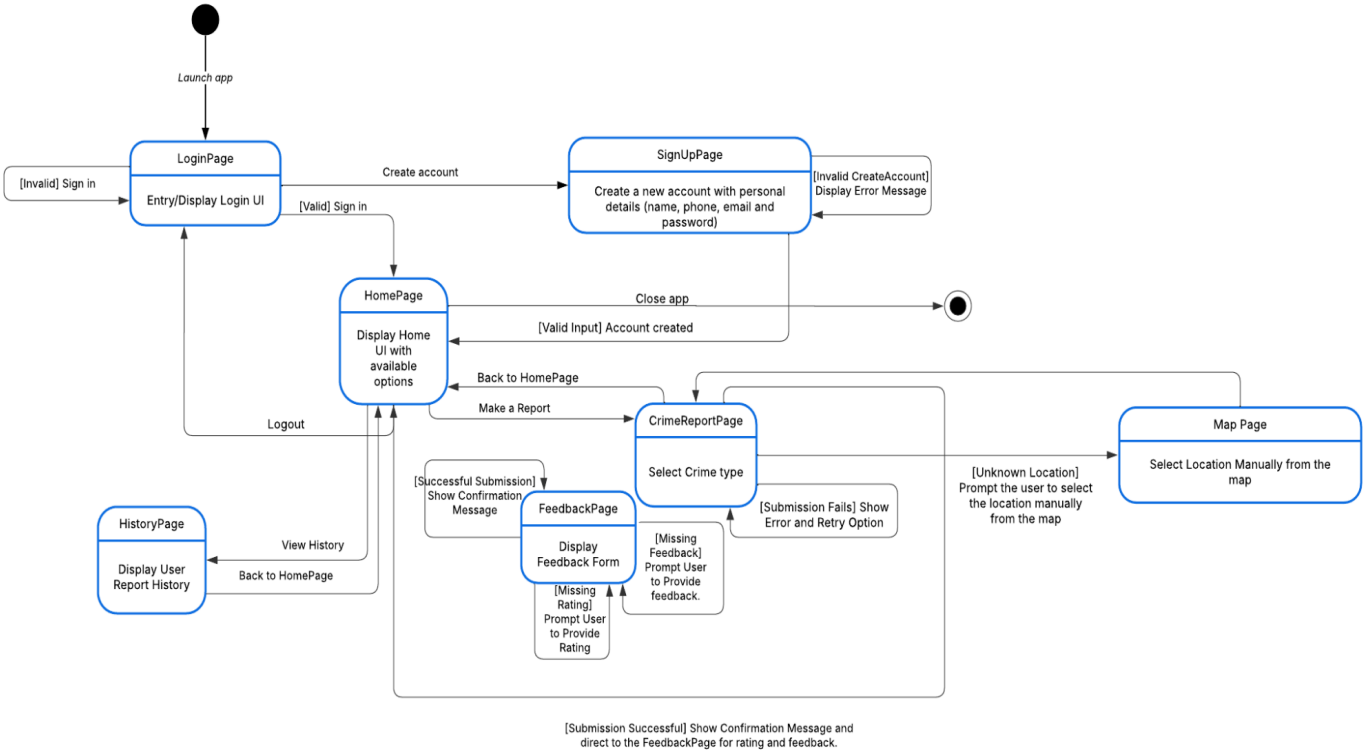

sd [Nearest police station finder]
User
System
Police Station Finder
Location Service
Dataset
Police Station
Select Crime Type
Process Location Request
Access user's location data
Loop
[check police station dataset]
Query nearest police station
Return nearest station
Pass nearest station info
Provide nearest station data
alt
[Nearest station found]
Send Alert
Acknowledge request
Report sent
[No station found]
Retry/report manually


sd [Make a report]
User
System
Crime Report Controller
Crime Report Database
Location Validator
Police Station Validator
Select "Make Report"
Open Crime Reporting Page
Submit crime report
(crime_type, location, police_station)
alt
[Location valid and police station valid]
Validate location
Return valid location
Validate police station
Return valid police station
Store crime report
Confirm crime report storage
Display
"Crime report submitted successfully!"
Display success message
"Crime report submitted successfully!"
[Location invalid or unknown]
Validate Location
Return invalid location
Display error message
"Location is unknown. Please select your location manually from the map."
Display error message
"Location is unknown. Please select your location manually from the map."
Redirect to map page
Loop
[Fetch Crime Reports]
Fetch list of Reports
Retrieve reports
Return list of reports
Display list of reports
Display reports in history

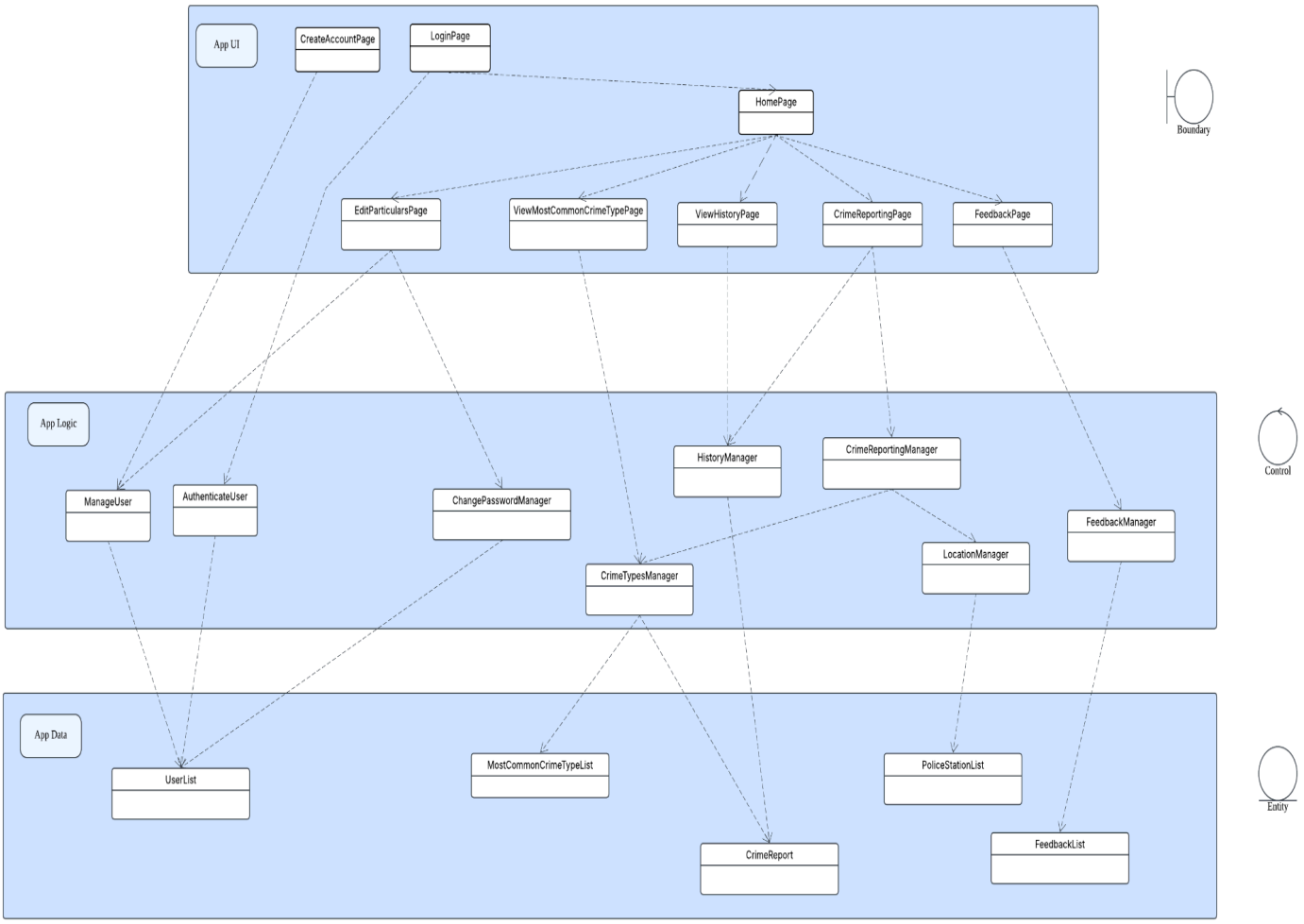

sd View History
Actor
Home Page
History Page
History Manager
Crime Report database
Click on view history option
Direct to view history page
Request History
findHistory(user)
Alternative
[history records not null]
return user history
Forward past reports submitted
Display past reports
[history records is null]
return null
No history found
Display no history found


sd [Submit feedback]
User
System
Feedback Controller
Feedback Database
Authentication Controller
User database
Enter rating and comment
Submit feedback
alt
[valid feedback]
Validate user token
Fetch user details
Return user details
Store feedback (rating, comment, email)
Confirm feedback storage
Return success message "Feedback submitted successfully"
Display success message "Feedback submitted successfully"
[missing rating]
Return error message "Please enter your rating"
Display error message "Please enter your rating"
[missing comment]
Return error message "Please enter your feedback"
Display error message "Please enter your feedback"

B.5 Dialog Map



B.6 System Architecture Model



Appendix C: Supplementary Materials

In the CrimeWatch mobile application, we integrate data from various external sources, including Google Maps API, Twilio API and also Five Preventable Crime Cases Recorded By Neighbourhood Police Centre (NPC), and Singapore Police Force Establishments (2018) (GEOJSON), datasets from data.gov.sg. As a result, different tools are required for web scraping each of these platforms to extract relevant crime data as well as police station data.

C.1 Mobile App Data Retrieval

Since **CrimeWatch** is a mobile application and not a web scraper, the primary method of data retrieval involves integrating with APIs and fetching data in real time. Instead of dealing with static or dynamic websites, we rely on external APIs to gather data related to crime reports, user locations, and nearby police stations.

- **Fetching Data from APIs:**
    - **API Requests:** The app communicates with the backend server (FastAPI) using **HTTP requests** to interact with data sources. This can include reporting a crime, retrieving the most common crimes in a given location, or finding the nearest police stations.
    - **Libraries Used:**
        - **Axios** (for making HTTP requests): Axios is a promise-based HTTP client that works in both the browser and Node.js. It's used in React Native apps for handling API requests and responses.
        - **Fetch API**: This is a built-in API in JavaScript that is also used for making HTTP requests. In **CrimeWatch**, it can be used to send and receive data from backend servers or third-party services.

- **Real-Time Data and Location Services:**

    - The app leverages **Expo Location** to fetch the user's current location in real time, which helps in displaying nearby crime hotspots or police stations

In the **CrimeWatch** mobile app, we do not rely on static or dynamic website scraping. Instead, we integrate directly with external APIs for data retrieval, real-time location services for user location tracking, and use efficient libraries like **Axios** and **Expo Location** to interact with the backend and other third-party services. By focusing on real-time data fetching and API interactions, we ensure a seamless and responsive experience for users reporting crimes and accessing crime-related information.