5. Emergency Safety Alert (Author: Srabonti Biswas)

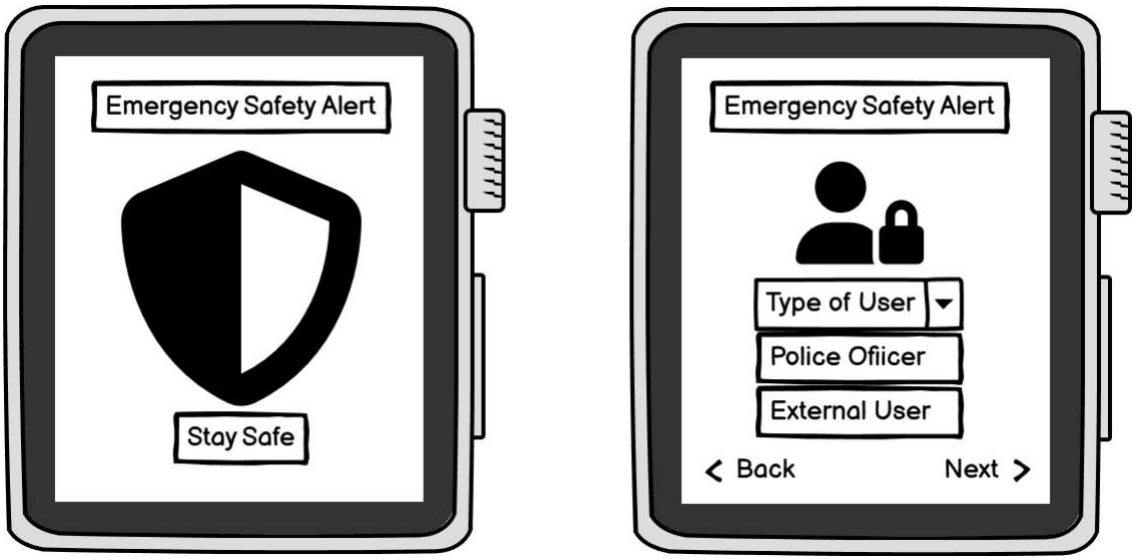

Figure79: Home screen and Lock screen Emergency Safety Alert App

5.1. Introduction (Author: Srabonti Biswas)

As part of developing smart watch application for law enforcement departments, we have observed that it is very important to initiate communication between police control center and the public (residents and visitors), when a public safety issue arises. Therefore, all rights to manage Emergency Safety Alert system are reserved by law enforcement departments.

The functionalities of Emergency Safety Alert App will change according to the type of user. For the same application the user interface will be different for police officer and the public. The application will be pre-installed on the provided smart watches for police departments, while the external user need to download the application from play store.

Police control center will use Emergency Safety Alert system to transmit information to the population during major emergencies and disasters. Considering the importance of public safety, we have decided to use push notification service to deliver information to the smart watch devices.

Our goal is to avoid big damages through a smart watch application which can eventually increase the efficiency of the law enforcement department. The application includes security, authenticity of information, push notification services, integrated mapping services, providing status dashboards. We will try to develop our application that is easy and straightforward for our customers to use Smart Watch user interface.

In this documentation, we have focused mainly on developing front-end solutions. The following system architecture, description and diagrams will provide the fundamental ideas of our application.

5.2. Preliminary Requirements for the system (Author: Srabonti Biswas)

The objective of our application is to get information from police officer at incident spot via app and send notification to the registered customer's app in form of push notification. Police control room will play as a bridge between fetching information from police officer and making ready to broadcast messages from a central server of control room. The requirements for the system architecture were analysed throughout the process of the project. Some requirements were refined and divided into several requirements, while some other requirements were summarized into one.

5.2.1. Actors and their goals

We have identified actors in our system, they are typically external objects of the system that produce/consume data. Apart from main actor, we have also identified supporting actors in our system who are engaged to provide service to our system. The different types of actors of the system and how and why they are used will be described in below:

- **Customer**

There are two types of users who can login to our system: Police personnel and External user. Both of them are customers of our application. All the functionalities of this application are developed according to the type of end-user. User interacts directly with the front-end part of the application.

- **Database admin**

Once a user registers and the admin logs in, the admin may delete the user, examine a user's information, edit information and display various app reports from within the app. The user's registration is saved to the Database of police control centre.

- **Push notification service**

Upon registration, the app receives a unique device token from the push notification service. This token is unique to the device and the app, serving as an identifier to broadcast the notification messages to all registered devices.

- **Token maintainer**

The token maintainer stores each new token in a database.

- **Push provider**

Upon getting request from server application, the push provider sends push notification requests to the push service

- **Backend server**

The app's backend server integrates with the push notification service.

- **Database**

Each registered token is stored in a database.

- **Operating System**

Tapping on a notification trigger to open the corresponding app involves the interaction between the smart watch operating system, the app and the notification system.

- **Customer Support**

Customer support team simply processes the queries and complaints of the customer related to our application.

- **Business Analyst**

A business analyst is part of the management team, takes over the responsibility of analyzing the accumulated data from each customer. His tasks include data visualization, creating dashboards to make decisions on investments in different departments.

5.2.2. Requirement Analysis

To implement the app, we have collected a few requirements. These requirements have been divided into functional and non-functional requirements. Each sub-section is sorted by the priority of the items. For the non-functional requirements, there is no business value and estimation for the implementation determined, because these requirements are implemented implicitly throughout the development process of the project.

5.2.2.1. Functional Requirements

These are the requirements that help the reader to understand the functionalities of the system. The following requirements are represented in the form of input to be given to the system, the operation performed, and the output expected. The sub-section of table will provide more details about which requirement refers to which actor in our application.

We have determined 'Priority: low/medium/high' from the perspective of developer and 'Business value: low/medium/high' from the perspective of manufacturing company. 'Estimation Implementation (A team of 5 full-time developers): d = day, w = week'.

ID	Name	Priority (low, medium, high)	Brief description	Business Value	Est. Impl.	Reviewed by
1	Login as Police Officer	High	The police officer will be able to login the system with the provided login credentials from police department	Medium	0.5w	*Liliana-Andreea-Mika Ragalie*
2	Communicate with the Control Room	High	The police officer will be able to initiate communication with police control department through their provided smart watches.	Medium	1w	*Liliana-Andreea-Mika Ragalie*

3	Share Location	High	The police officer will be able to share real-time location through the application. GPS allows devices to determine their precise location.	Medium	1w	
4	Manage Notification	High	The push notification is a short message sent by central server entity of police control centre to all registered devices	High	2w	*Liliana-Andreea-Mika Ragalie*
5	Customer Support	High	Customer support is needed to process customer query via phone number/email.	Low	0.5w	
6	Create scheduled push notification	High	Keeping track of the timestamp when each notification is sent and considering the removing time. We have decided to implement scheduled scripts on server that periodically checks the timestamps of notification and remove the expired notification from server.	High	2d	
7	User Registration/ Login as external user	High	Only an external user needs to register when using the system for the first time. The login data will be stored in the database of police control centre.	High	1w	*Liliana-Andreea-Mika Ragalie*

8	React on Push Notification	High	Pushes always pop up at the top pf the screen. The users' interaction with the notification by tapping on it triggers to open the corresponding messages	High	1w	
9	Administrator Rights	Medium	The administrator has the permission to add or delete user accounts.	High	1d	
10	Dynamic Dashboard	High	According to the type of user (Police officer and The Public), the application will be redirected to the respected dashboard. In each case, the dynamic dashboard will show real-time data. And the dashboard will be automatically updated itself at regular intervals.	High	2w	
11	App Registration	High	When a user installs and opens the app for the first time, the app registers with the push notification service (a global service) provided by the respective operating system.[1]Upon registration the app receives a unique device token from the	High	0.5w	

[1] The decision about on which operating system to use for developing the application will be made by the development team in collaboration with other stakeholders.

			push notification service.			
12	App customization	High	Users may have the ability to customize the dashboard based on their preferences. This could include choosing push notification turn on/off	Low	0.5w	

Figure 80: Functional Requirements

5.2.2.2. Non-functional Requirements

The following requirements are analysed to improve the quality and performance of our application:

ID	Name	Priority (low, medium, high)	Brief description
13	Backend Timer	High	If several push notifications are sent in a row, they may stop reaching the user. To avoid this problem, we have used backend timer to ensure that push notifications are only sent once every 20 Minutes.
14	Optimise battery usage	High	Optimizing battery usage in an app is essential to provide a positive user experience. Here are several strategies that developer may implement in the application to optimize battery usage: 1. Efficient Network Usage 2. Minimize the use of continuous location tracking 3. Utilize push notification services efficiently
15	Optimize server resources	High	Regularly cleaning up outdated notifications helps to optimize server resources by reducing unnecessary data storage
16	Rating system	Medium	User of the smart watch will be able to rate the feature of

			the application with stars and leave feedback for the improvement of the product.
17	Customer data analysis	Low	Business Analyst analyses the usage of the application and the customer satisfaction.
18	IP Adress Mapping		The device's IP address can be used to determine its approximate geographic location. This method is an alternative, when the application is unable to lock onto a GPS signal. However, this method provides less accurate location information than GPS.
19	Font readability	High	For positive user experience font choice, size, Text Alignment, Contrast are crucial.

Figure 81: Non-functional Requirements

5.3. System Design (Author: Srabonti Biswas)

Our Emergency Safety Alert application is designed to be accessed by two different types of users : Police officer and the public. And it is managed by a central server. According to that, we decided to that the police officer will get the application pre-installed on their smart watch, whereas the warning service for the public will be accessible only via the downloaded application. The system architecture with the functionalities that are provided by the application is illustrated by the diagram below:

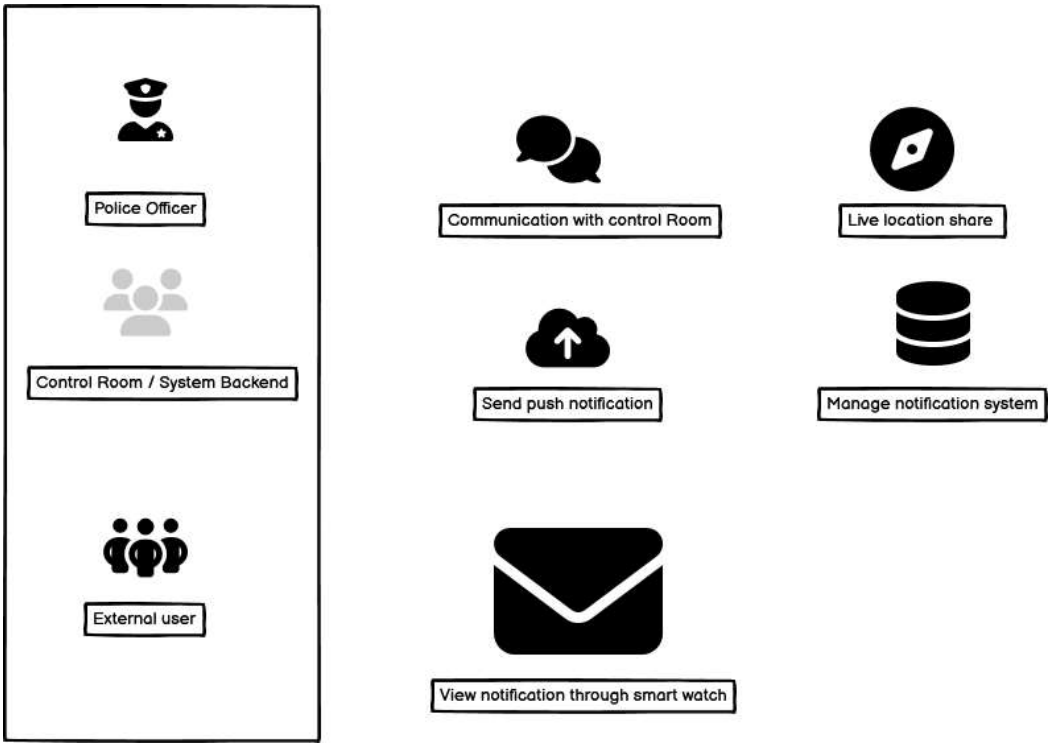

Figure 82: System design Emergency Safety Alert App

5.4. Tabular list of Backlog Items (Author: Srabonti Biswas)

In the previous chapter, we have documented several requirements required to develop the application. To get a clear glimpse of the required efforts to develop the application, we have turned the list of requirements into backlog items. In the following section, we will perform a simple user story mapping from the end-user perspective. The user story helps the whole development team to understand what targeted goals they will accomplish and how much work they can handle. We have used Trello to visualize each process as a separate column.

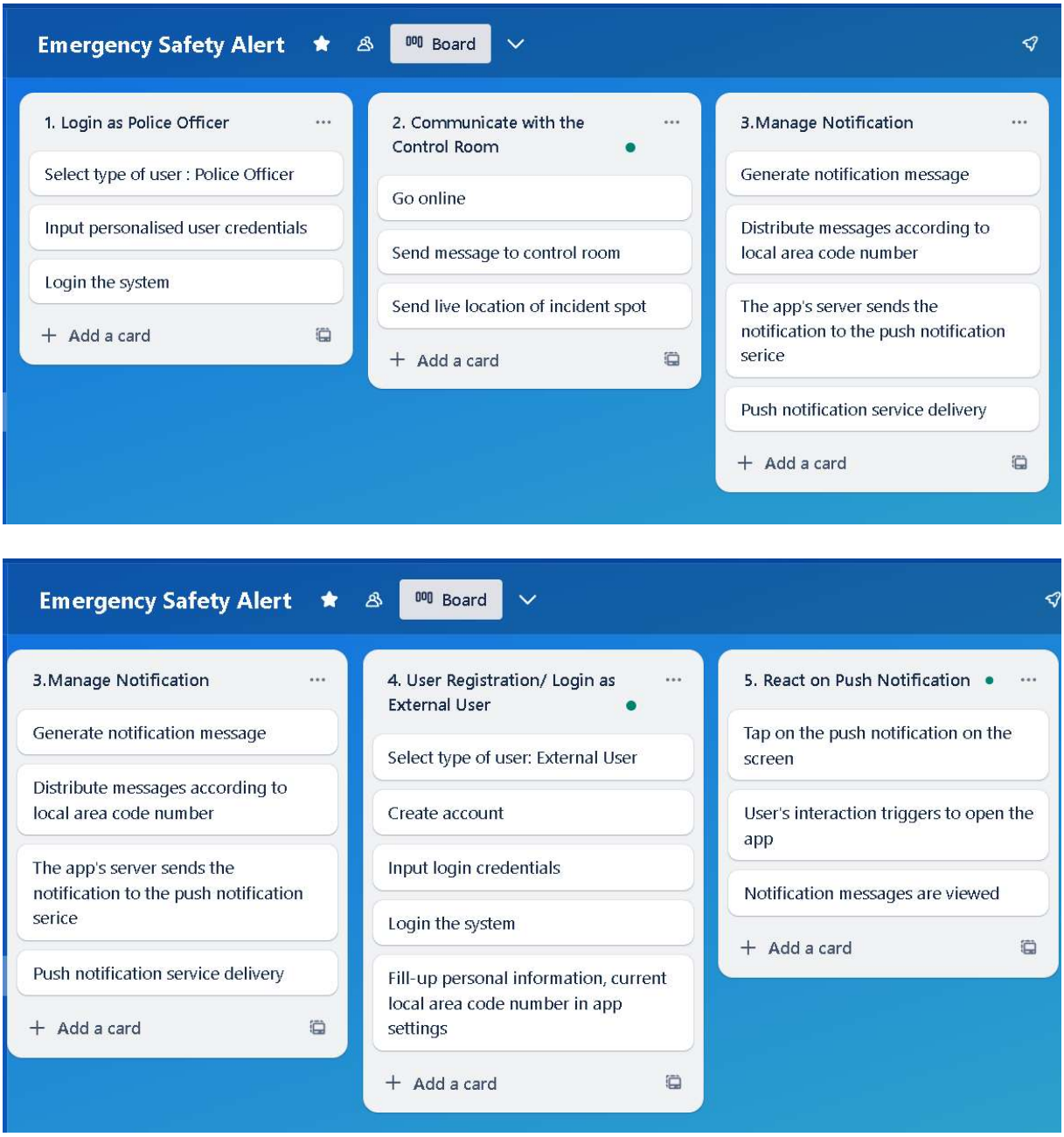

Figure 83: Simple User Story Mapping (Trello Board)

5.5. Implementation Details (Author: Srabonti Biswas)

In the previous chapter we have created a User Story mapping to prioritize the actions for each process step according to the importance for this application. In the following section, we will discuss the purpose of each backlog item in form of a use case description. To provide a better overview, several requirements have been depicted in UML Diagrams.

Before we specify all our diagram, we would like to analyse the selected requirements (in form of a use case description). In the following use case specification, we have provided the following set of information:

- Who is the primary (secondary) actor?
- What are the actor's goals?
- Which preconditions should exist?
- Which tasks are performed by the actor?
- What exceptions might occur?
- What system information will the actor acquire, produce or change?
- What variations are possible?

After the completion of requirement analysis, we will step towards modelling Unified Modelling Language (UML). Among a large variety of diagrams, we have focused on the following UML Diagrams:

- Use case diagram
- Activity diagram
- Class diagram
- Sequence diagram
- Class Diagram (Entire system)

For graphical representation, we have used MagicDraw. Depending on the type of requirement, we have omitted some diagrams with proper explanation.

For the visualization of user interface, we have used User Interface prototype Model. Our goal is to design user friendly app's interface, allowing designers, developers, and stakeholders to see how the app will look and function. In our project, we have used Balsamiq Wireframes for Desktop to design the user interface for this application.

5.5.1. ID #1 Login as Police Officer (Author: Srabonti Biswas)

5.5.1.1. Use Case Description

Name	Login as Police Officer
ID	1
Trigger	Police officer will input their personalised login credentials to login the system
Actors	Police officer
Pre-conditions	The policer officer is provided with personalised login credentials from the police department.
Post-Conditions	Police officer has successfully logged in the system.
Basic Flow	**Input correct login credentials :** 1. Each police officer is provided with a personalised login credentials from the police department. 2. No need to register the system. 3. Input username: Employee ID (To identify the police officer) 4. Input password: Password (Provided by police department)
Alternative Flow	**Input wrong login credentials.** Redirect to login screen

Figure 84: Login as Police Officer Use Case Description

5.5.1.2. Use Case Diagram

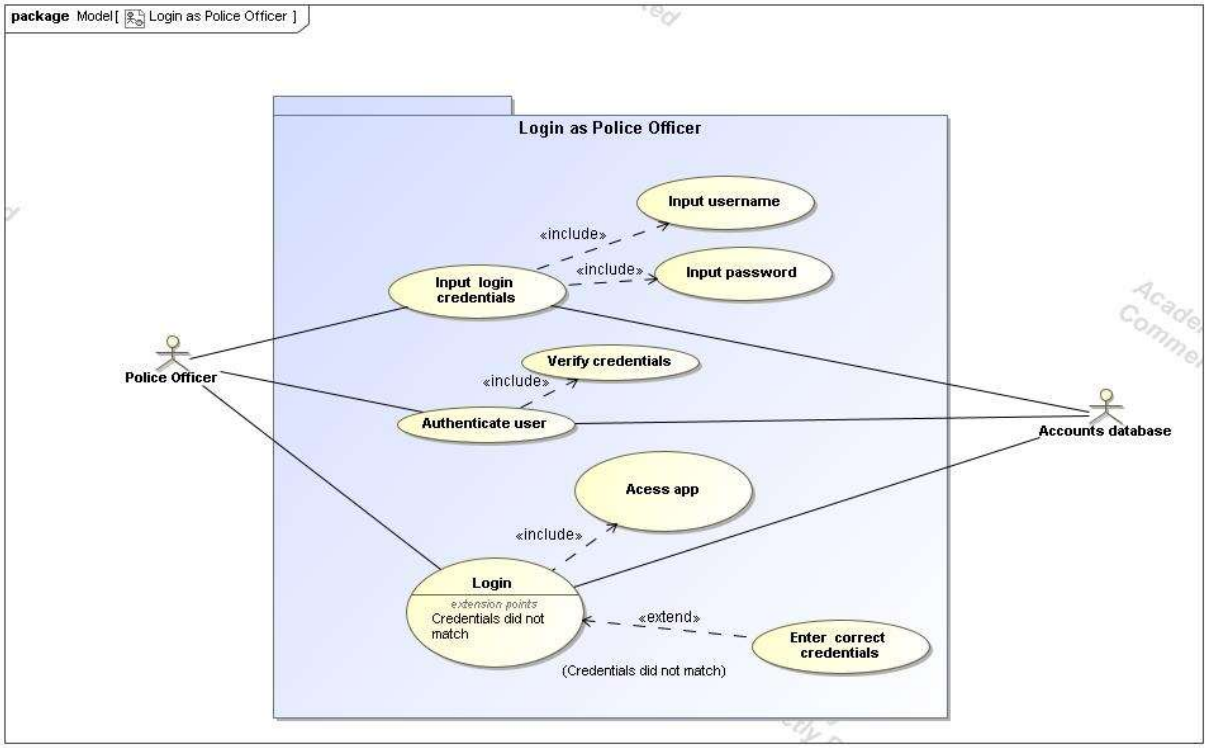

Figure 85: Login as Police Officer Use Case Diagram

5.5.1.3. Activity Diagram

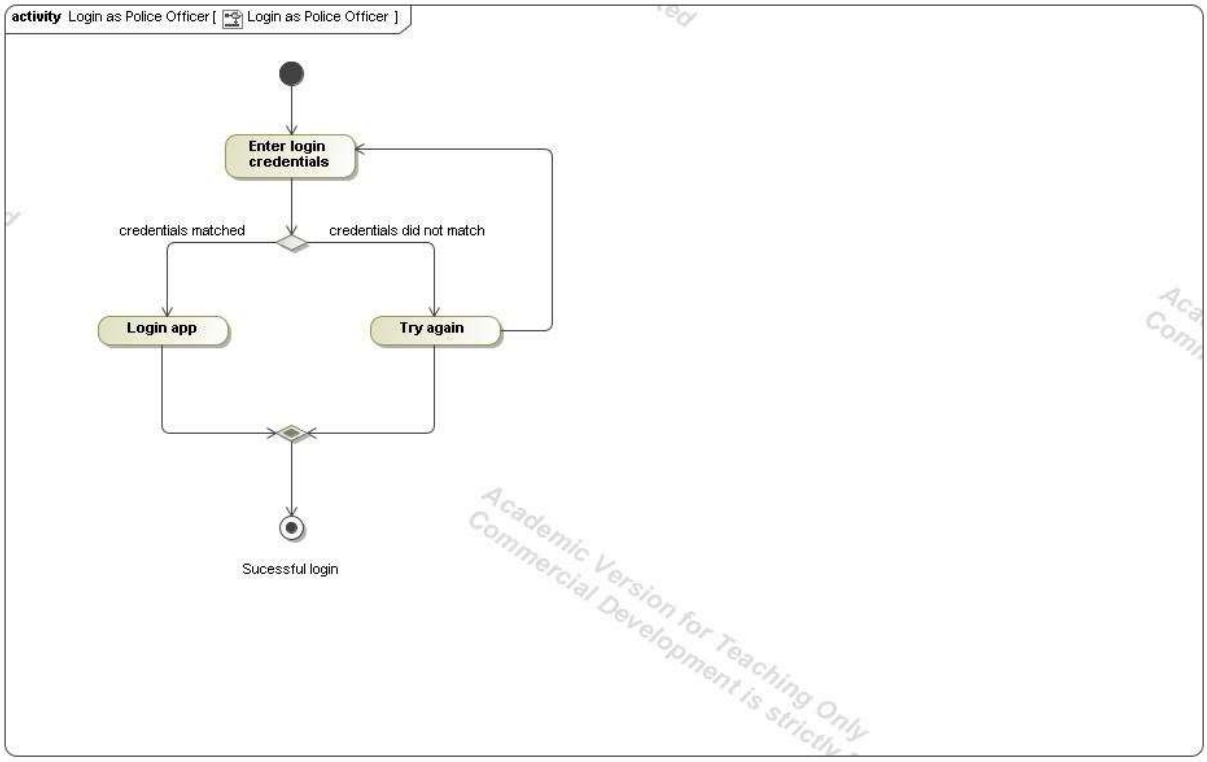

Figure 86: Login as Police Officer Activity Diagram

5.5.1.4. Sequence Diagram

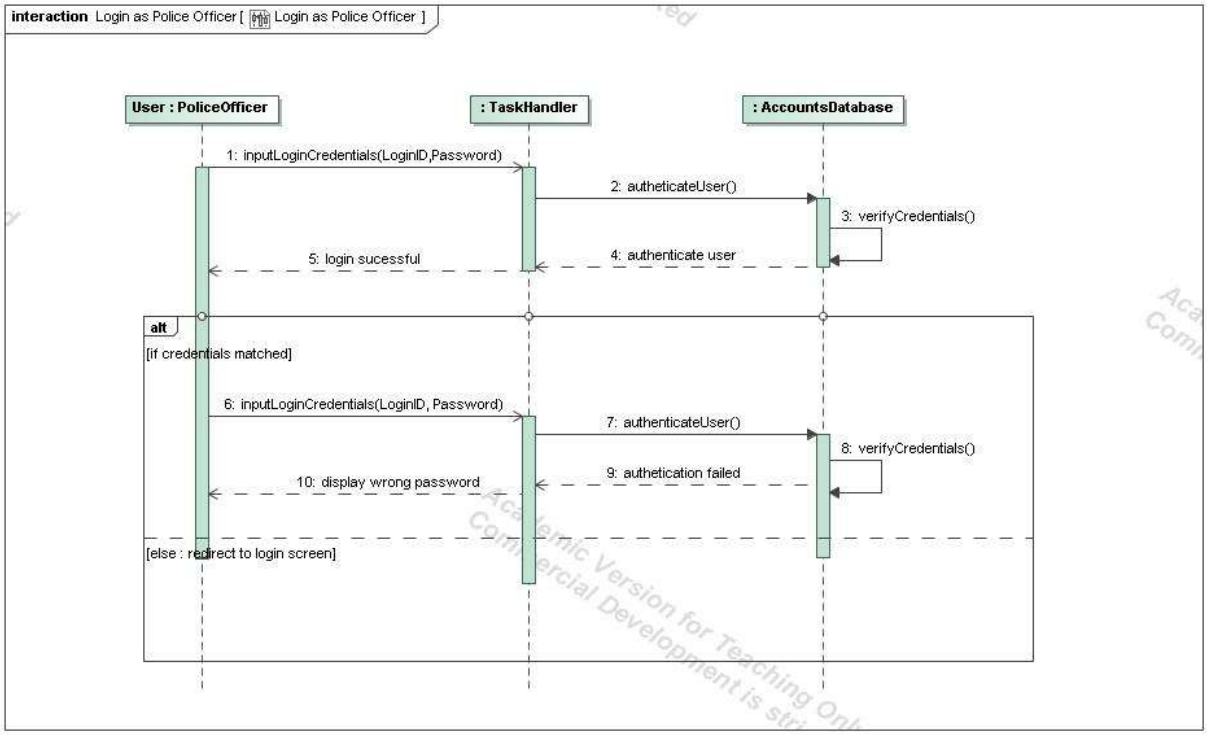

Figure 87: Login as Police Officer Sequence Diagram

5.5.1.5. UI Mockup

Figure 88: UI Mock up ID # 1 Login as Police Officer

5.5.2. ID #2 Communicate with Control Room (Author: Srabonti Biswas)

5.5.2.1. Use Case Description

In the following use case description, we will describe the interaction between the police officer at incident spot and the application to initiate communication with police control room.

Name	Communicate with the Control Room
ID	2
Description	The police officer at incident spot uses personal smart watch to communicate with the police control room.
Trigger	The police officer logs in the system and sends messages to inform the control room.
Actors	The police officer
Pre-conditions	The police officer has successfully connected with the control room
Post-conditions	A ticket is generated for each communication to manage the information in control room.
Basic Flow **Description** **Actions** **1** **2** **3** **4**	**Communicate with the control room** This scenario describes the situation where the police officer needs to communicate with the police control room to share a new incident, so that control room can activate emergency alert for the public. The police officer observes the incident spot Thepolice officer tries to connect with the control room , he logs in the system The police officer sends messages to control room to inform about the new incident. The officer will be able to share live location of the incident spot
Alternative Flow **Description** **Actions** **1** **2**	**If the connection with the control room is not successful** The scenario describes the situation where connection with the control room is not successful. The user interface will display an error message The app will redirect to home screen
Alternative Flow **Description** **Actions** **1** **2**	**If there is active GPS signal** The scenario describes the situation where the GPS signal is available The GPS-sensor update the current location and send the Police officer current location

Figure 89: Communicate with the Control room use case description

5.5.2.2. Use Case Diagram

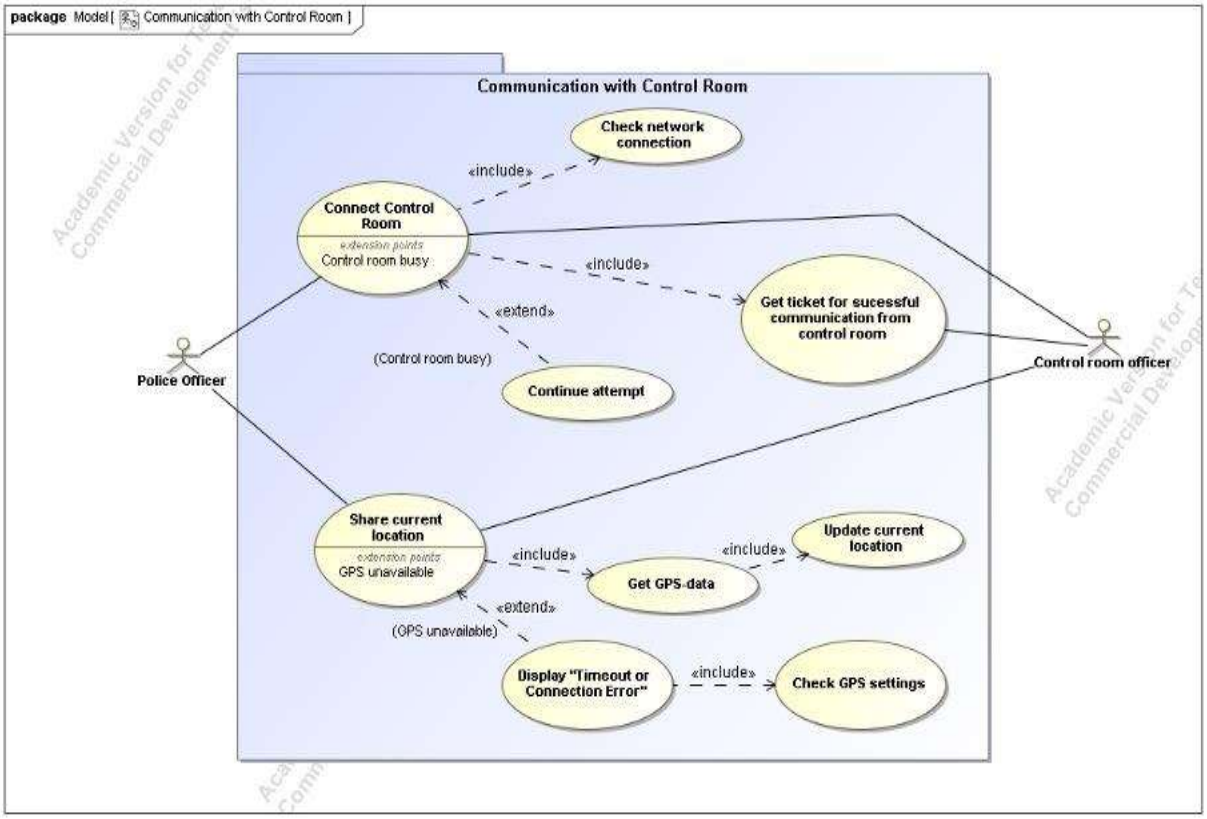

Figure 90: Communicate with the Control room use case diagram

5.5.2.3. Activity Diagram

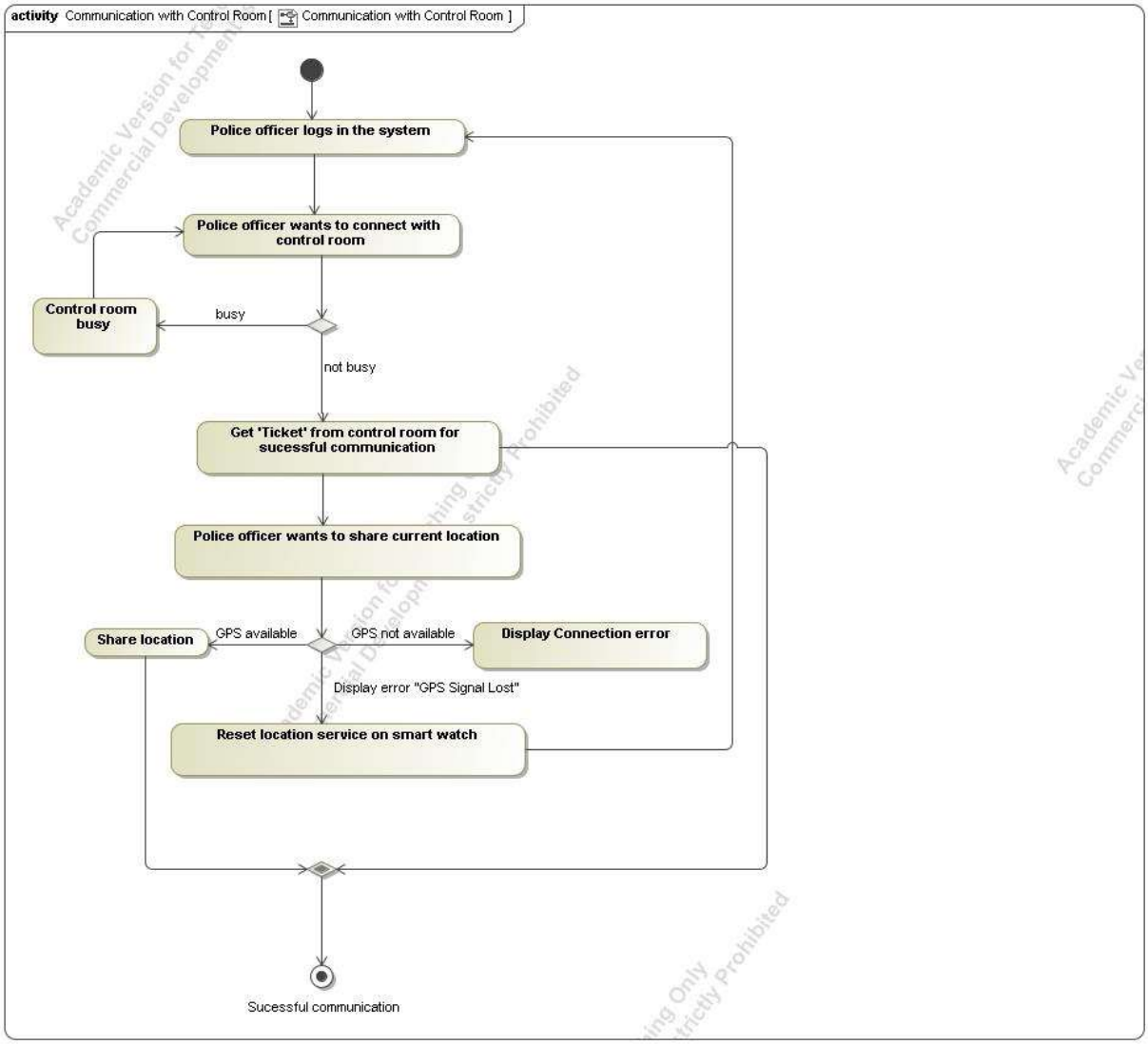

Figure 91: Communicate with the Control room activity diagram

5.5.2.4. Sequence Diagram

I have focused on abstract representation of the entire process that happens in this use case. Therefore, I have illustrated a sequence diagram of the use case:

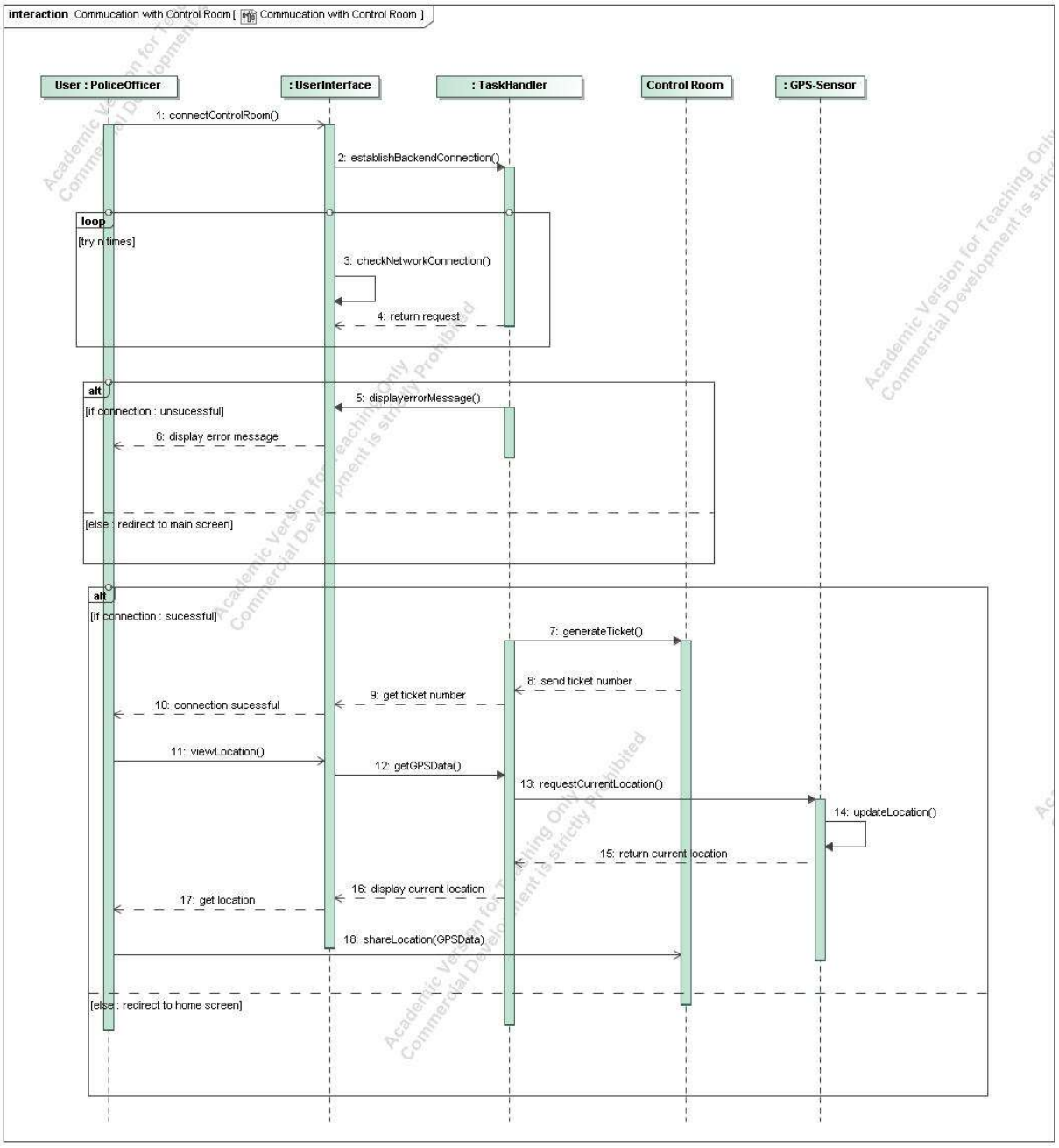

Figure 92: Sequence Diagram ID #2 Communicate with the Control Room

5.5.2.5. UI Flowchart

The flowchart of UI will illustrate the user interface how the app functions when the police officer wants to establish connection with the police control room through the application:

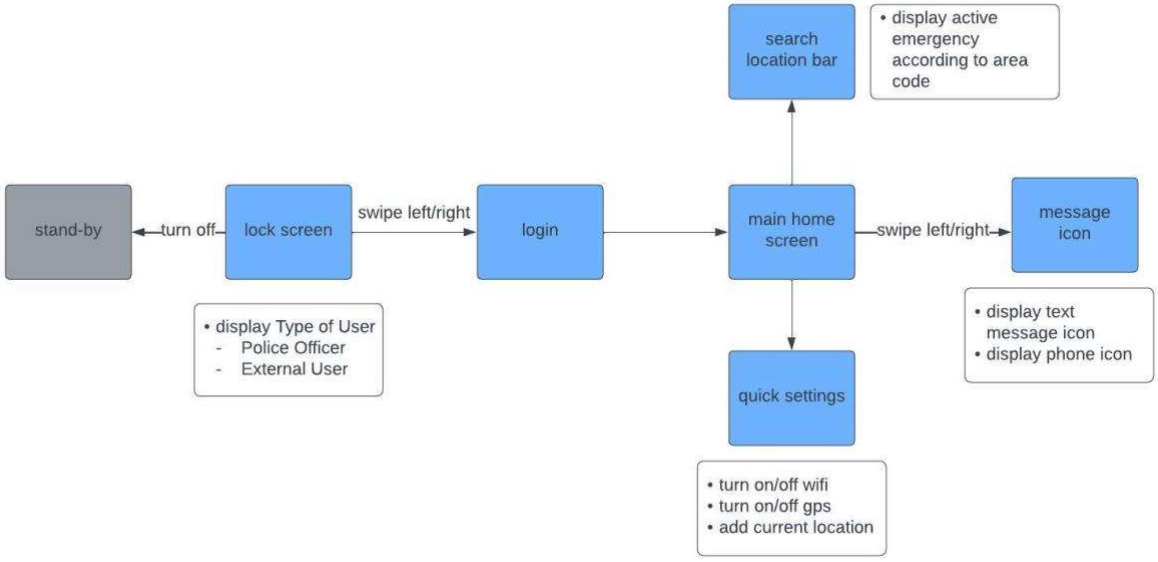

Figure 93: UI Flowchart (Communicate with the Control Room)

5.5.2.6. UI Mockup

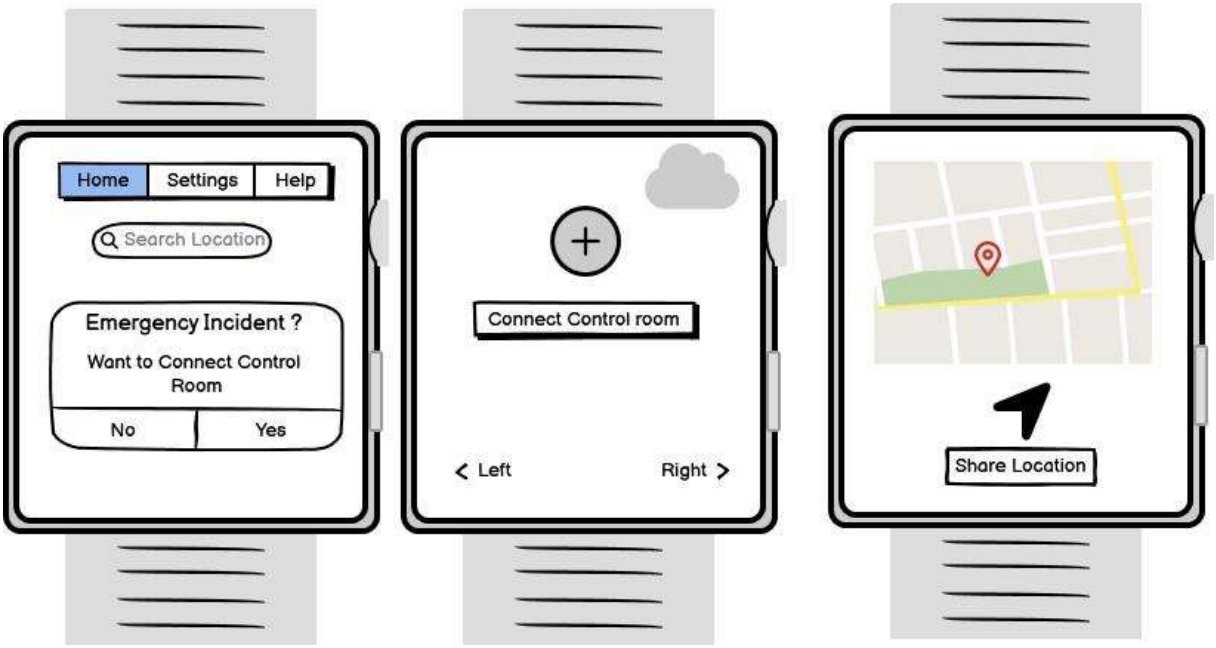

Figure 94: UI mock up ID # 2 Communicate with the Control Room

5.5.3. ID #3 Manage Notification (Author: Srabonti Biswas)

Our application is used to send push notification to remote devices. Between the user interface of police officer and the public there is an intermediate entity which is responsible to manage the transmission of notification. For our understanding, we consider the intermediate entity as the backend part of our system.

5.5.3.1. Push Notification Management Flowchart

The following flowchart provides insights into the synchronization between app's backend server with the frontend of the application:

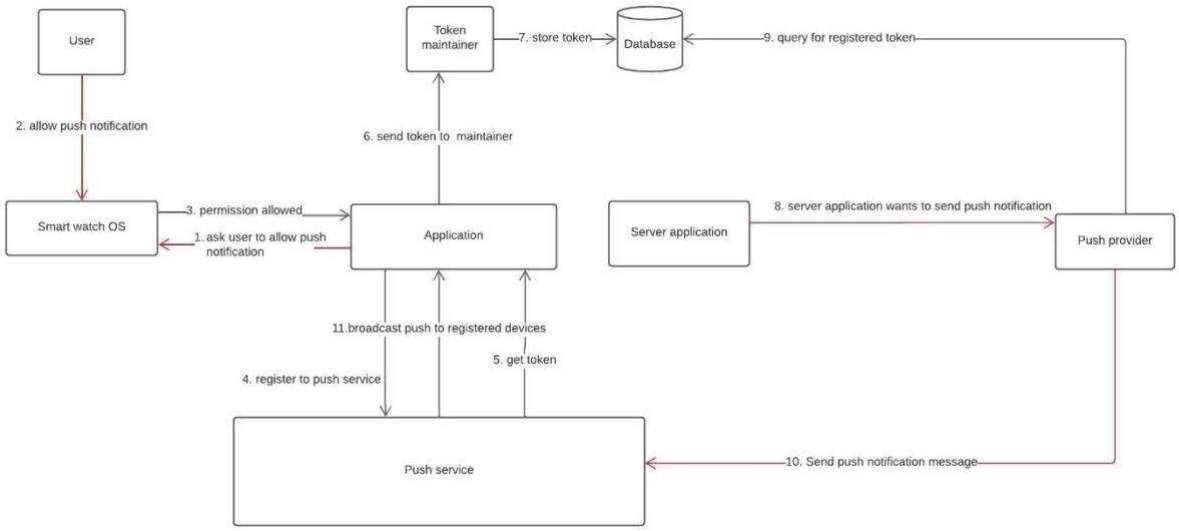

Figure 95: Push Notification Management Flowchart

5.5.3.2. Use Case Description

Name	Manage Notification
ID	3
Description	The External user allow push notification on the device. The server sends push notification with assistance with the push service.
Trigger	The external user allows push notification to initiate the process
Actors	The external user, application

Figure 96: Manage Notification Use Case Description

I have described the notification management system in the following section of this use case

5.5.3.3. Use case Diagram

In our project, notification management is the backend part of the application. In the following diagram we will show the use cases to visualize the notification management system and how the registered user, application interact with backend. Additionally, the diagram will show what the notification management system does.

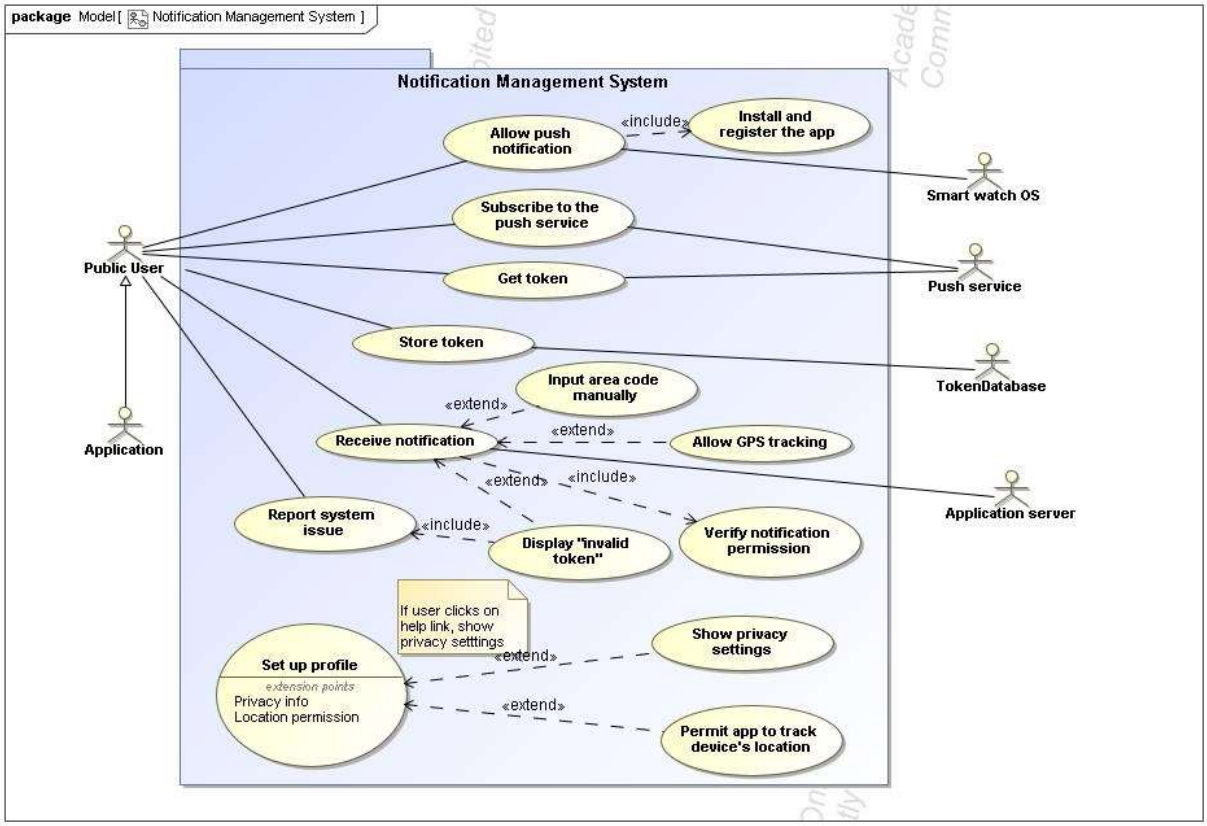

Figure 97: Use Case Diagram ID #3 Manage Notification

Anything inside the rectangle that happens within the backend part of application. In our following diagram, registered user is defined as primary actor, because the registered user initiates the use case to permit push notification service in the application, while Smart watch OS, Push service, Database are involved in this use case to achieve the goal of the primary actor.
The notation, a line with an arrow, shows a generalized relationship between the two components. Here 'User' is generalized to 'Application'.
To get the push notification service through the application, the pre-requisite is the push notification permission must be activated. Therefore, i have illustrated include relationship between 'Receive push notification' and 'Verify push notification' use cases.
The user can choose the option between Allow or Disable GPS tracking to get the location-based information. Therefore, I have illustrated extend relationship between the 'Receive push notification' and 'Allow GPS tracking' use cases.
Alternatively, user can manually input area code to get location-based notification through the app.
Display "Invalid token" this type of error shows, if the application is reinstalled or device is reset.

5.5.3.4. Activity Diagram

The following diagram will visualize a flowchart to represent the flow of control among the activities in notification management system.

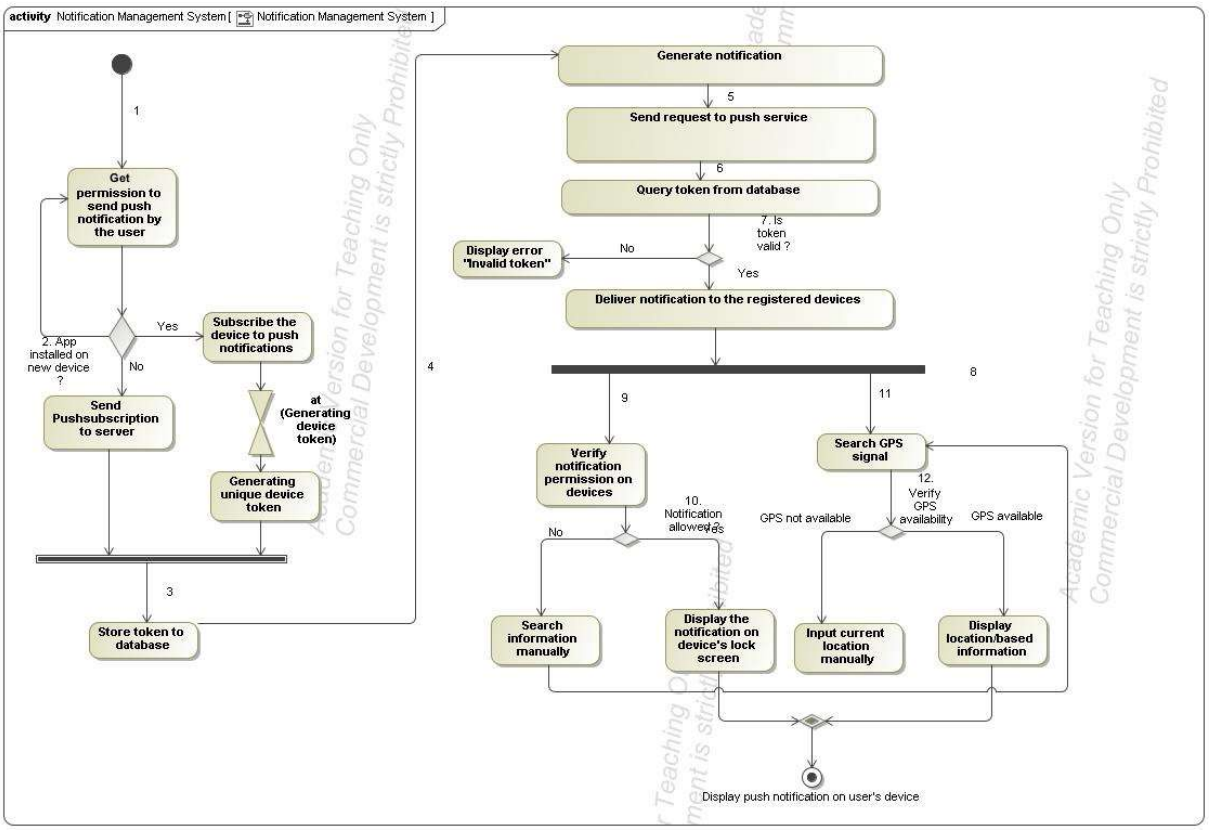

Figure 98: Activity Diagram ID #3 Manage Notification

If we analyse the above diagram, the steps are as follows:

Step 1: Get permission to send push notifications:
First, the application needs to get the users' permission to send push notifications. This should be triggered by users' gesture, such as clicking on 'Yes' button next to a prompt 'Do you want to receive push notification?' After that confirmation, the operating system on the users' smart watch will present a UI format to formally confirm that the user wants to opt in to push notifications.
Step 2: Subscribe the new user to push notifications.
After getting the permission, the application needs to initiate the process of subscribing the new user to push services. The subscription process is accomplished with assistance of the respective operating system of the smart watch. Assuming the subscription process is successful.
Step 3: Generating Unique device token
Upon successful subscription, the app receives a unique device token from the push service. This token is an identifier for delivering notifications to remote devices. The unique token needs to be stored in a database. Normally this is done by sending the information to server that controls the application.
Step 4 + Step 5 + Step 6: Push Notification service delivery:

Server application wants to send a push notification. And send request to push service. Here a database query is generated. The push notification service delivers the notification to the targeted device via the unique device token.
Step 7: If app is re-installed or device is reset, the token gets invalid. In that case, notification is not possible to send, resulting in error message.
Step 8: In this step the verification is performed concurrently, whether push notification permission is activated, and the devices have GPS signal.
Step 9 + Step 10 : If notification permission is activated, the system displays the notification on the lock screen. Alternatively, the user can turn off notification permission. In that case, the user may be able to search manually.
Step 11 + Step 12: To deliver location-specific information, active GPS signal is a pre-requisite. In our project, we have considered a special situation when GPS becomes unavailable for certain period after power on. If it happens, user can manually input their current location to update the smart watch with their current position.

5.5.3.5. Sequence Diagram

In the following diagram, i have illustrated how different parts of notification management system interact with each other to carry out the use case 'Manage Notification' and its order when the particular use case is executed.

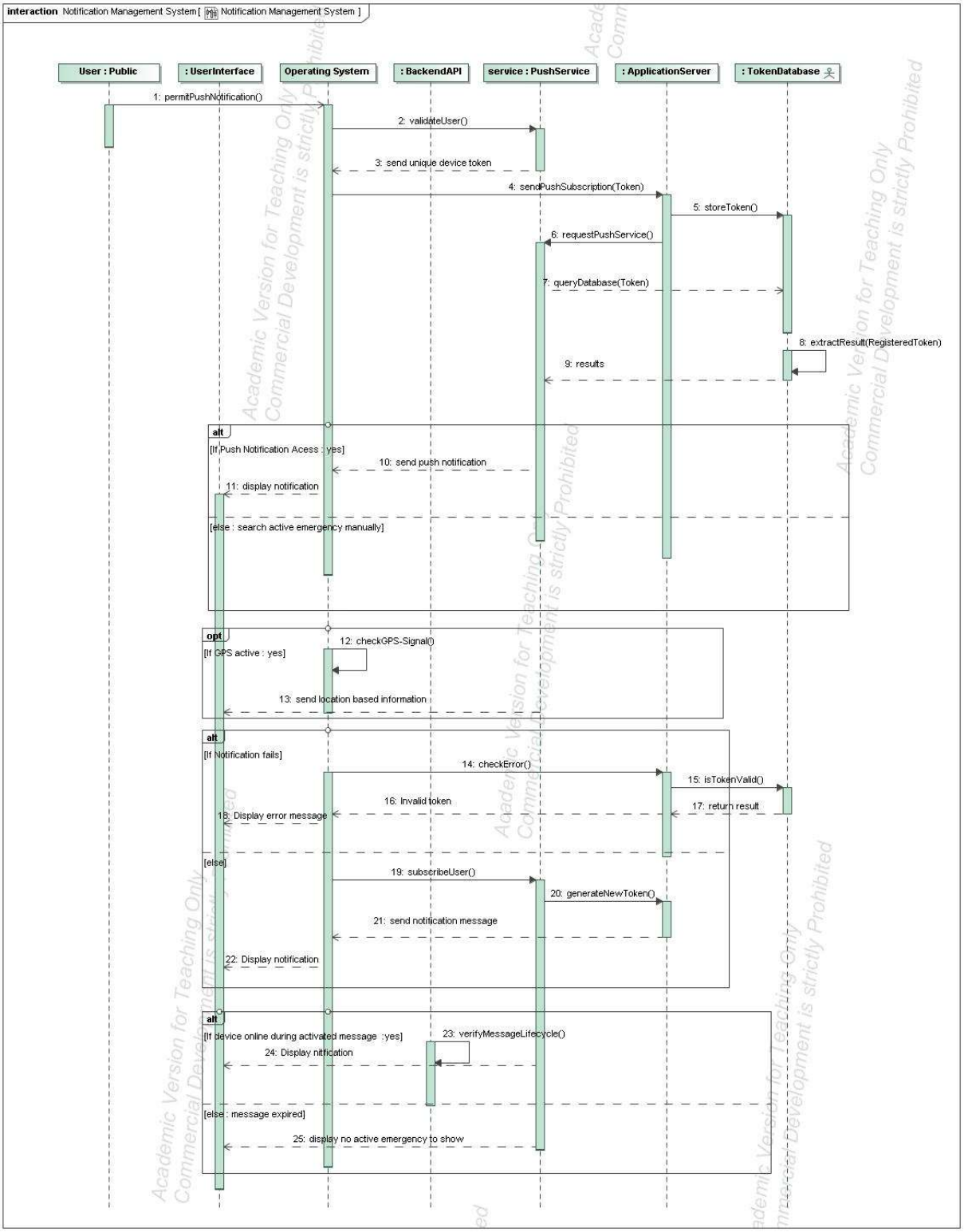

Figure 99: Sequence Diagram ID #3 Manage Notification

In above Figure, the message 6 invocation message direction showing solid arrow, because here server sent request to push service to send notification

Please note that:
I have omitted the UI mock-up for this use case because this is the backend of the system which is not visible in the user interface.

5.5.4. ID#4 User Registration/ Login as External User (Author: Srabonti Biswas)

5.5.4.1. Use Case Description

Name	User Registration / Login as External User
ID	4
Description	The external user logs in to the system. This is a necessary step before the user can use any function of the system
Trigger	The user wants to open the app
Actors	The external user
Pre-conditions	App is installed, connected to the internet
Post-conditions	The user is successfully logged in to the system
Basic Flow **Description** **Actions** **1** **2** **3**	 This is the main scenario where the user already has a valid account and password The user enters the email address The user enters the password The system verifies that the email address and password are valid. If so, the user is logged in to the system.
Alternative Flow **Description** **Actions** **1** **2**	A The user is not registered yet (unknown user) The user registers an account according to Use Case 2 The user will be redirected to the login screen again
Alternative Flow **Description** **Actions** **1** **2** **3**	B The user has an account but forgot the password The user resets the password The user receives an email for resetting the password according to Use Case 3 The user will be redirected to the login screen again

Figure 100: Use Case Description ID # 4 User Registration/ Login as External User

5.5.4.2. Use Case Diagram

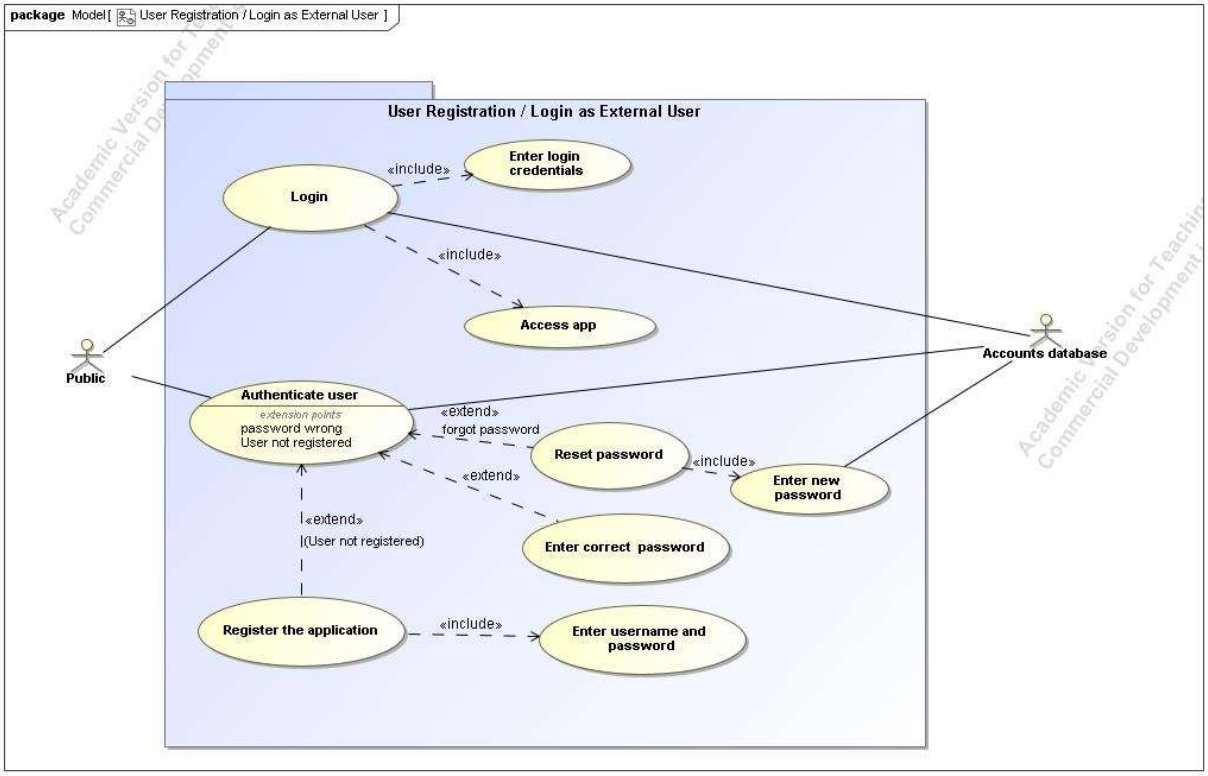

Figure 101: Use Case Description ID # 4 User Registration/ Login as External User

5.5.4.3. Activity Diagram

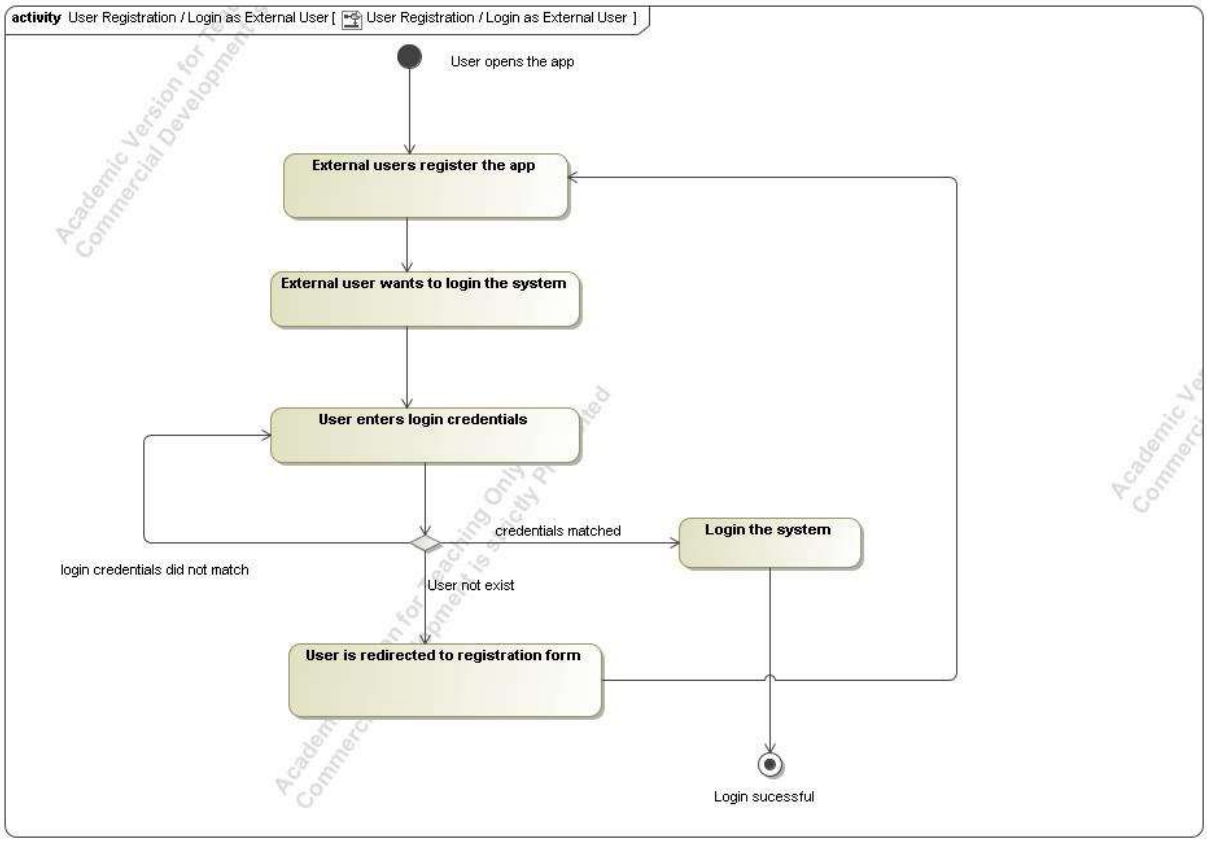

Figure 102: Activity Diagram ID # 4 User Registration/ Login as External User

5.5.4.4. Sequence Diagram

I have illustrated the use case with a sequence diagram to visualize the abstract process of the use case. The 'User registration/ Login as external user' use case is described with the help of following sequence diagram.

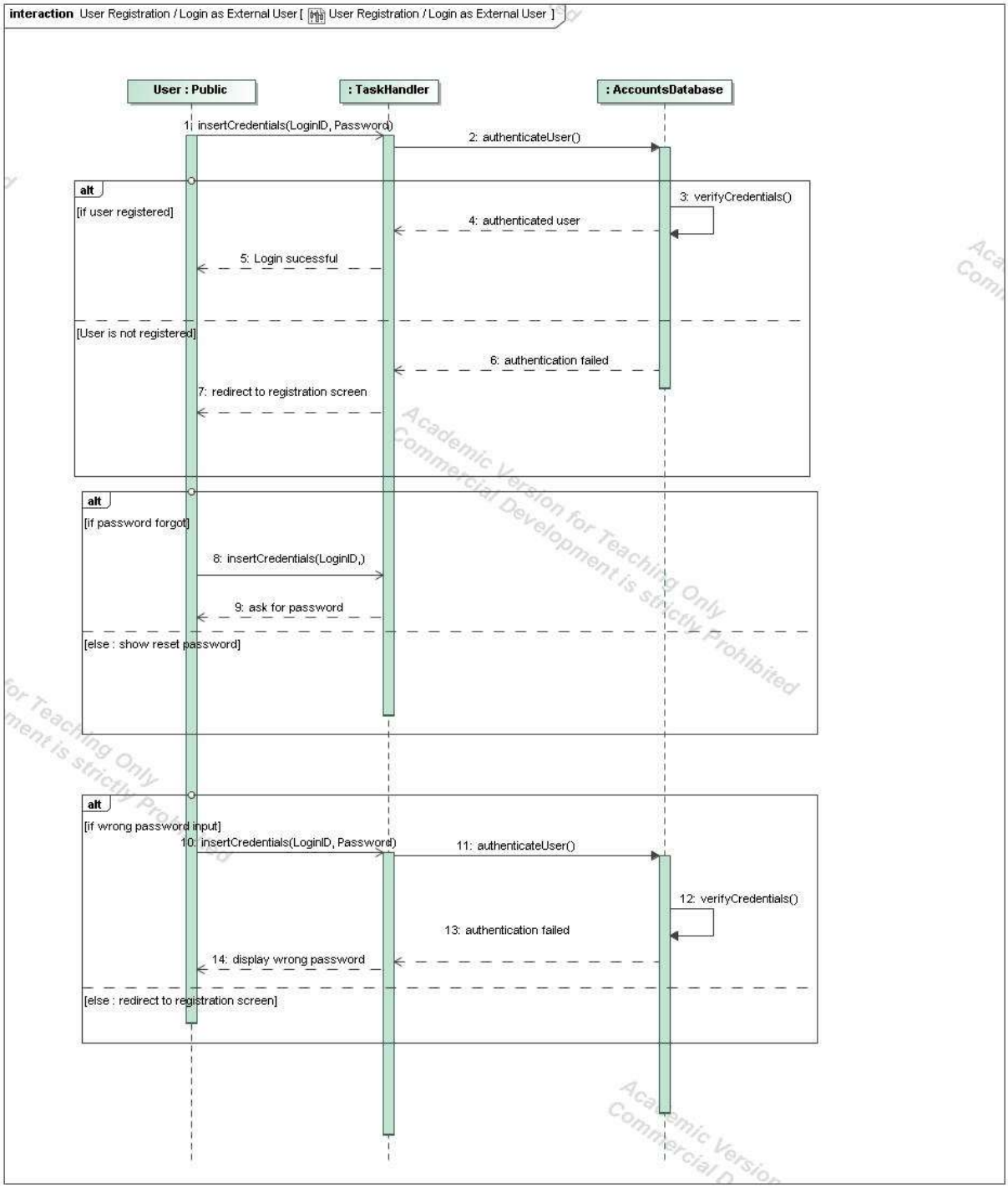

Figure 103: Sequence Diagram ID#4 User registration / login as External User

5.5.4.5. UI Mockup

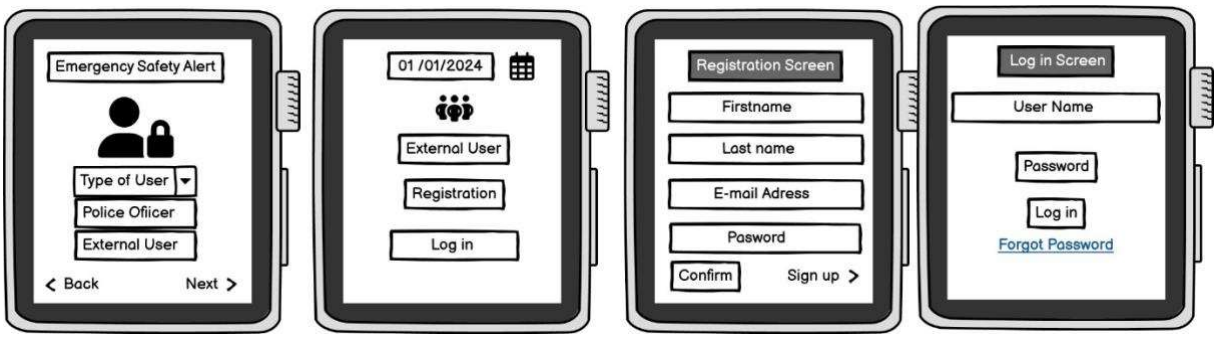

Figure 104: UI mock up User Registration / Login External User

5.6. Class Diagram of the System (Author: Srabonti Biswas)

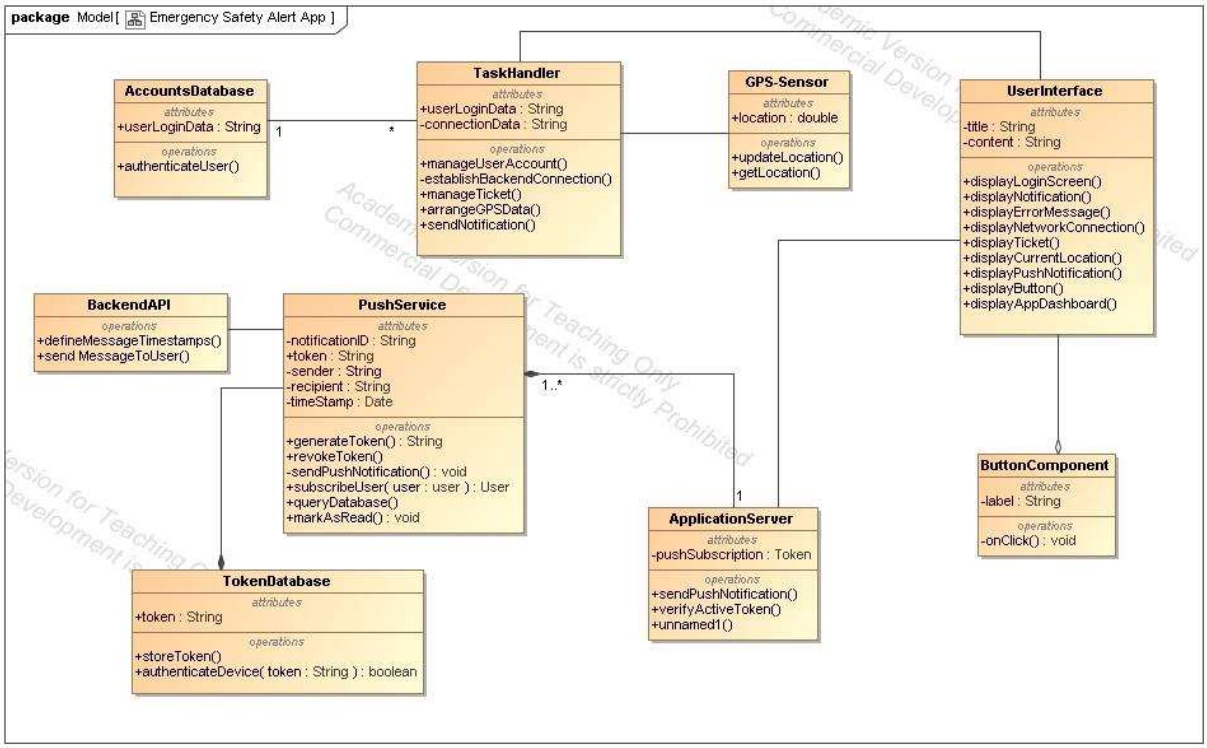

Figure 105: Class Diagram

If i analyse the above class diagram, “AccountsDatabase” class has association relationship with the class “TaskHandler”. One Accountsdatabase can store the data of multiple users.

“PushService” and “TokenDatabase” can not exist independently. To store the registered unique device token, token database is a necessity; sothat the push service can verify the regeistered device, when application server sends request to send notifications to remote devices.

“ApplicationServer” and “PushService” can not exist independently. Because to get push notification each must register with minimum one push service

“UserInterface” has a relationship with “ButtonComponent”. Although user interface displays various notification, information, but whenever user wants to command the application to call information, application requires button component to click on the user interface.