[CS350] 2021 Fall Team Project
Project #3. Software Design Description (SDD)

Smart Tracker

Team 12
20160509 이정재
20180155 김준범
20180265 박윤정
Submission Date: 2021-11-21

Table of Contents

1. System Overview

A. Requirements

Functional requirements		
R. ID	**Requirement Description**	**Dependencies/Assumptions**
R.F.A.1	Bluetooth communication should be available between the tracker device and the smartphone application	Bluetooth connection will be available
R.F.A.2	A smartphone application should always run in the background and should receive a strength signal from the tracker device for distance measurement.	-
R.F.B.1	The tracker device should be able to be charged with C-type battery charger	The battery charger will meet the size of the tracker and be compatible to enable the charging function
R.F.C.1	The tracker device can ring buzzer to inform the exact position to the user	-
R.F.C.2	The smartphone application should offer functions for helping the user to find the exact position of the connected tracker devices	-
R.F.C.3	The smartphone application should save the most recent position of the connected tracker devices periodically	There should be enough memory for the application to save information (tracker information, GPS location)
R.F.D.1	The smartphone application shall show the remaining battery amount of the connected tracker devices to the user	-
R.F.D.2	The smartphone application shall turn on/off the connected tracker devices by the user's intention	-
R.F.E.1	The tracker device should be easily attached/detached from the user's property by utilizing tag or ring form	-
R.F.F.1	The smartphone application and the tracker device shall be paired by bluetooth connection	Bluetooth connection will be available
R.F.F.2	The tracker device should remember the paired smartphone application and communicates with it	-

Non-functional requirements	
R. ID	**Requirement Description**
R.N.1	Tracker device attaching process should be simple so users can easily attach/detach tracker devices to their belongings.
R.N.2	GUIs should be intuitive so that they are easily understandable for users.
R.N.3	All critical errors should be notified to users well, so that users can recognize and deal with errors (try again).
R.N.4	Size of the tracker device should be small enough so that they will not border users when they are attached to belongings.
R.N.5	Batteries in a tracker device should last at least several hours when they are fully charged.
R.N.6	Tracker devices should not be easily broken.
R.N.7	Smartphone device should have enough storage for installing the designated smartphone application and storing recent GPS positions
R.N.8	Bluetooth communication between the smartphone device and the tracker device should be available at distance
R.N.9	The smartphone application should be compatible with major smartphone OSs (iOS, AOS)
R.N.10	Smartphone applications should not track or manage unregistered tracker devices and the tracker devices should not receive messages from unregistered smartphone applications.

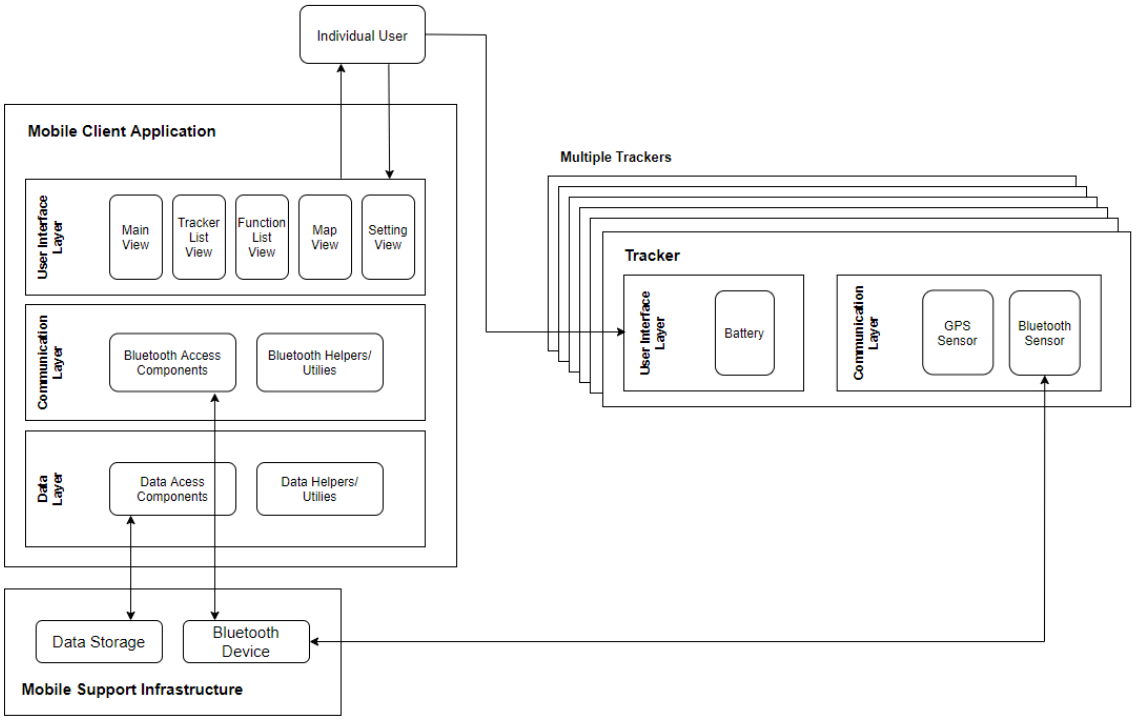
B. Tasks

Task Model		
Task ID	**Task description**	**Related Req(s).**
T.1	Build Tracker device	R.F.E.1, R.N.1, R.N.4, R.N.6
T.1.1	Attach GPS, Bluetooth, distance sensor, and Buzzer	R.F.C.1
T.1.2	Attach rechargeable battery and capacity sensor	R.F.B.1, R.F.D.1, R.N.5
T.1.3	Build tracker inner software	R.F.C.1, R.F.D.2
T.1.3.1	Implement ‘Tracker’ class	R.F.E.1, R.N.1, R.N.4, R.N.6
T.1.3.2	Implement ‘TrackerInfo’ class	R.F.C.2

T.1.3.3	Implement ‘SmartphoneDeviceInfo’ class	R.F.F.2
T.1.3.4	Implement Interfaces for distance, battery, GPS sensors.	R.F.A.1, R.F.A.2, R.F.C.2, R.F.D.1
T.1.4	Combine hardware components with wires and combinational logics	R.N.4, R.N.6
T.2	Build smartphone application	R.N.9
T.2.1	Create and Design UI	R.N.2
T.2.1.1	Implement Main, Tracker List, Function List, Map, Setting View	R.N.2, R.N.9
T.2.1.2	Implement transitions of above views	R.N.2
T.2.2	Implement functionalities of the application	R.F.C.2, R.N.3
T.2.2.1	Implement ‘TrackerOperations’ class	R.F.C.2, R.F.D.2
T.2.2.2	Implement ‘TrackerInfoList’ and ‘TrackerInfo’ class	R.F.D.1, R.N.10
T.2.2.3	Implement ‘TrackerUpdate’ class	R.F.A.2, R.F.C.3
T.2.3	Implement ‘BluetoothInterface’ class	R.F.A.1, R.F.F.1, R.F.F.2, R.N.8, R.N.9, R.N.10

2. System Design

A. System Architecture



[Figure 1] System architecture diagram

Layers
We identified the following 3 layers:

- User Interface Layer
- Communication layer
- Data layer

User Interface Layer
The User Interface Layer contains components that can directly interact with the users. In mobile application, User Interface Layer contains frontend parts of the app, which are the following views.

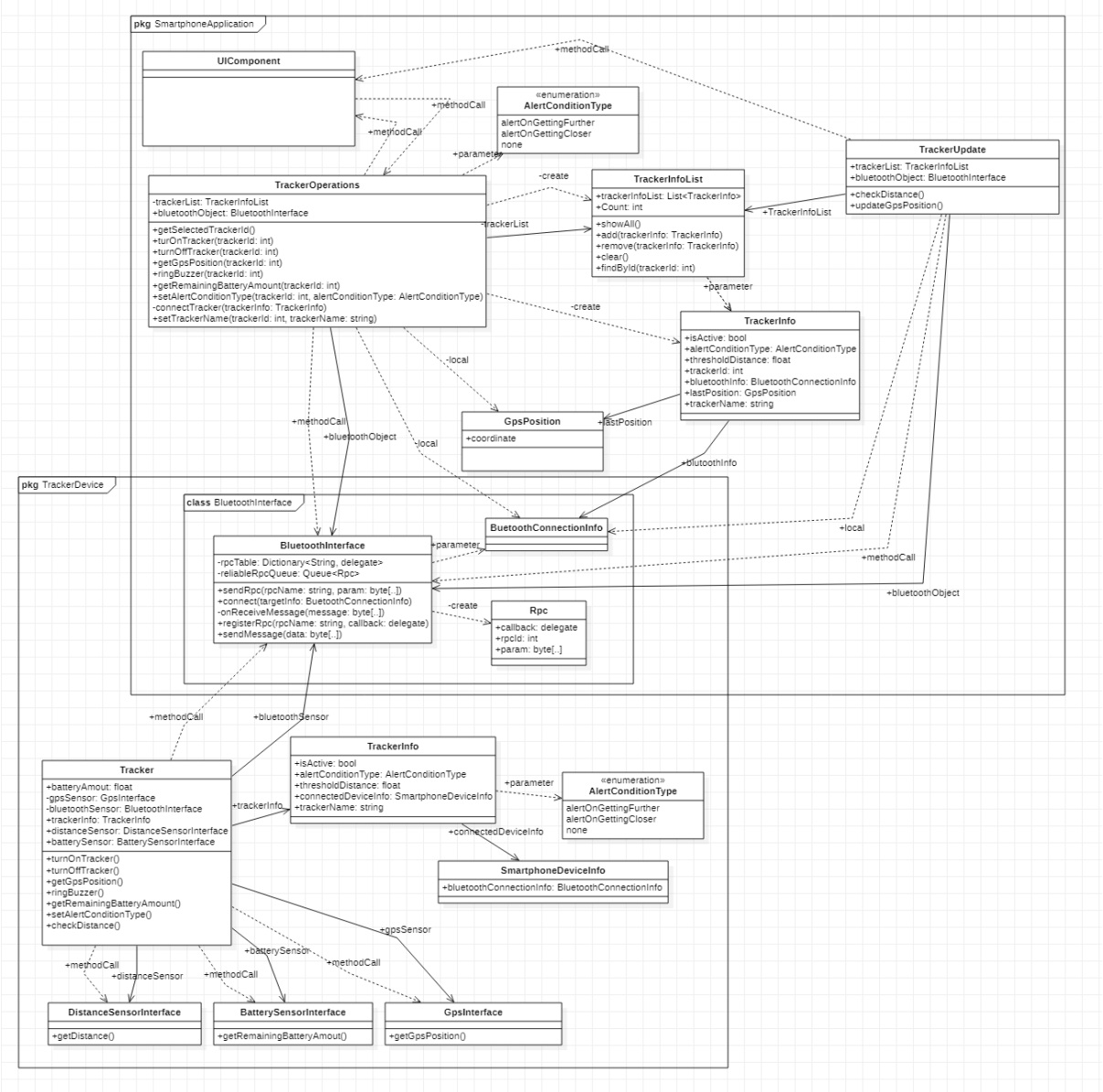
- Main view : The main view where the app begins.
- Tracker list view: Shows the list of the trackers that users have in connection.
- Function list view: Shows the list of functions users can utilize for a selected tracker.
- Map view: A map that shows the GPS location of the tracker
- Setting view: A view that can change various conditions of the tracker.

On the tracker side, the User Interface Layer contains the battery part of the tracker. Users can directly interact with it by using a C-type charger.

Communication Layer
The Communication Layer contains components that are used for Bluetooth and GPS connection. In the mobile application, Communication Layer only contains components and helper functions related to Bluetooth access. It is because the mobile application only interacts with trackers via Bluetooth. But the Communication Layer on the tracker needs to contain components related to GPS. It is because trackers need to calculate their GPS location sometimes.

Data Layer
The Data Layer contains components that are used for storing some data and accessing them.

B. Class Diagram



[Figure 2] Class diagram

UIComponent

Set of classes for UI framework

TrackerOperations

Set of tracker functions which can be utilized by users.

Related classes

- **TrackerInfoList**: The list of informations for trackers managed by application
- **TrackerInfo**: The set of informations of a certain tracker device
- <enum> **AlertConditionType**: Types of alert conditions. Users can choose from “alert when getting close”, “alert when getting further”, and “do not alert”.

TrackerUpdate

Update tracker device information periodically.

BluetoothInterface

The interface for bluetooth communication. It offers some functions including remote procedure calls and sending messages.

Related classes

- **BluetoothConnectionInfo**: A set of informations for bluetooth connection
- **Rpc**: A class for storing callback information of already sent remote procedure calls waiting for response message

Tracker

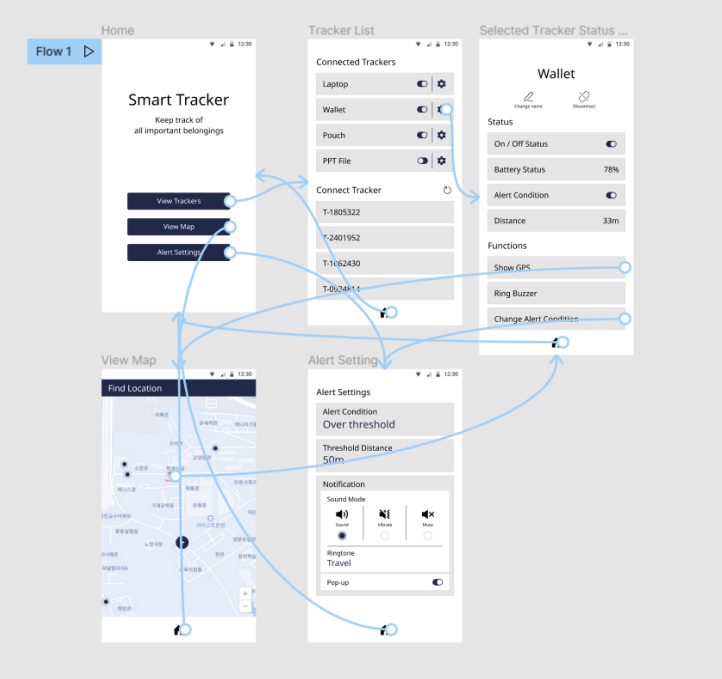
A class for a tracker device. It conducts operations requested from the connected smartphone application and sends a response message.

Related classes

- **TrackerInfo**: A set of information about the tracker device.
- **SmartphoneDeviceInfo**: A set of information required to bluetooth connection and communication with the smartphone device.
- <<enum>>**AlertConditionType**: The same with the enum in smartphone application.
- **DistanceSensorInterface**: An interface for receiving data from the distance sensor in the tracker device
- **BatterySensorInterface**: An interface for receiving data from the battery sensor in the tracker device
- **GpsInterface**: An interface for receiving GPS position data

C. User Interface Design

We have designed a simple prototype for the software application of the Smart Tracker using Figma. Below is the link to access to the figma file.
https://bit.ly/2Z8zJs4
Five main views were designed: the main page, tracker list page, status and function page, map view, and the alert setting page. Each page will be transitioned to each other by tapping particular buttons. [Figure 3] shows the rough outline of how the actions are structured.



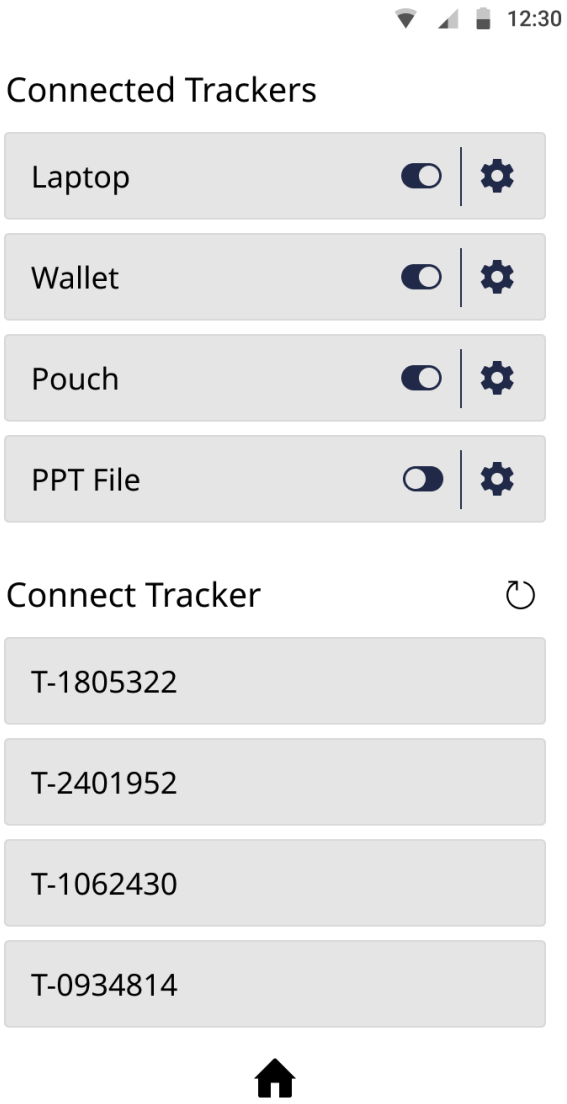
[Figure 3] The overall picture of transitions

a. Main Page

[Figure 4] Main Page View

Main page is the screen the users will face as they enter the application or click the home button at the bottom of all other pages. It has mainly three buttons that are connected to each page of its name: the tracker list, map view, and the general alert settings.

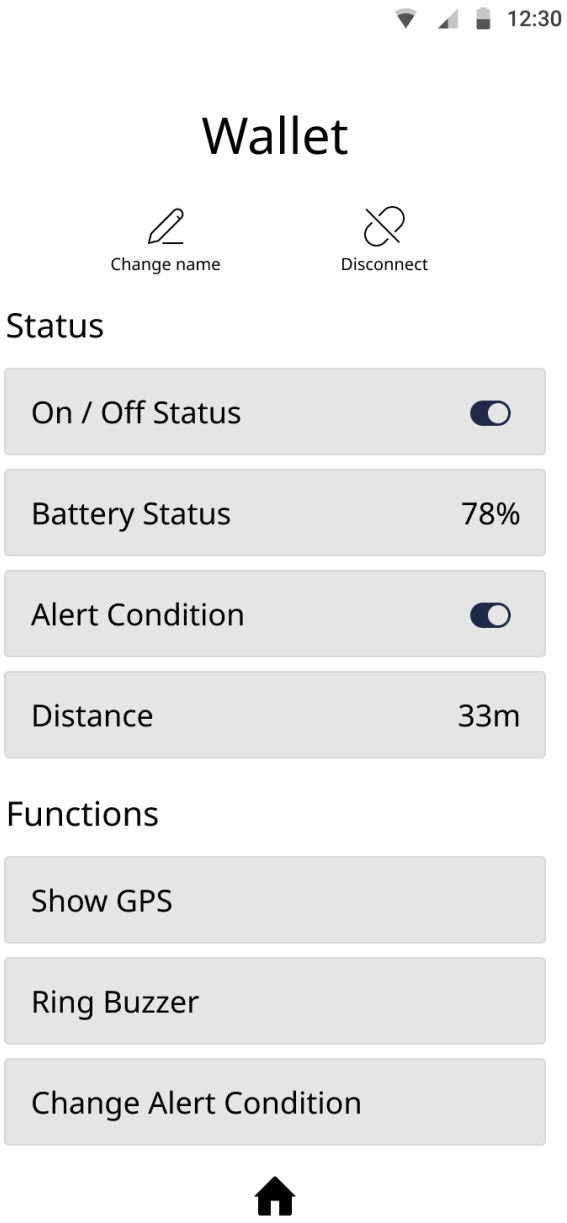
b. Tracker List Page



[Figure 5] Tracker List

When the user clicks “View Trackers” in the main page, this page is shown. The list of trackers already connected are listed first with the on/off button and settings button assigned for each. Users can also connect new trackers to the smartphone application via bluetooth, and they would be able to choose the appropriate tracker by the id. Refresh function will be supported.

c. Status and Function Page



[Figure 6] Status and Function Page

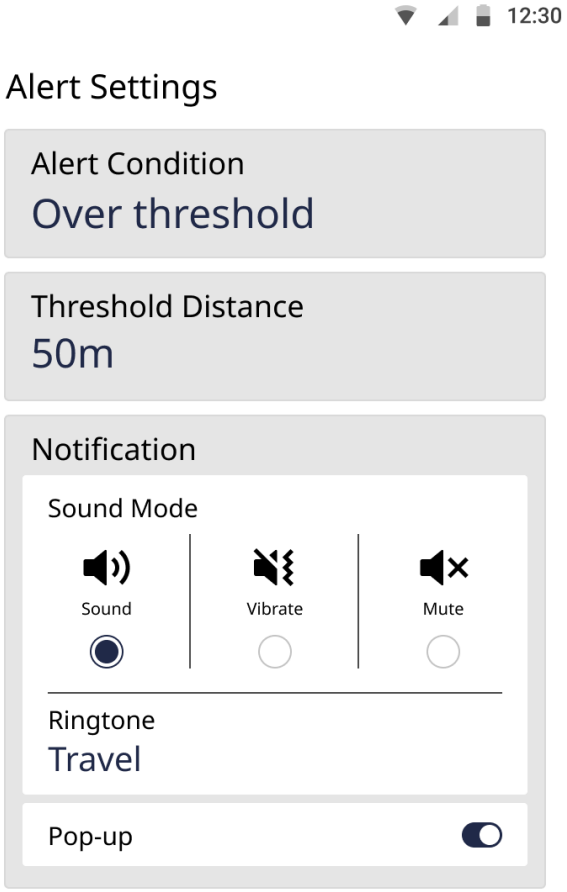
[Figure 6] shows the status and function for each selected tracker. This page is shown when clicking the settings button in the screen of [Figure 4]. On the top, users can change the name of the tracker or disconnect the tracker. Status and functions follow next. Under the status category users can check the on/off status, battery status, alert condition (whether the tracker will give notification when meeting the condition), and the distance. Under functions category users can check the location on the map via GPS, ring the buzzer, or change alert conditions more specifically.

d. Map View

[Figure 7] Map View Page

After clicking the “View Map” button at the main page or the “Show GPS” button at the status and function page, users can check the location of themselves and all the trackers. The selected tracker would be emphasized in red, and by clicking the buttons of the not-selected tracker, users can select other trackers. By clicking the selected tracker once again users can check the status and function.

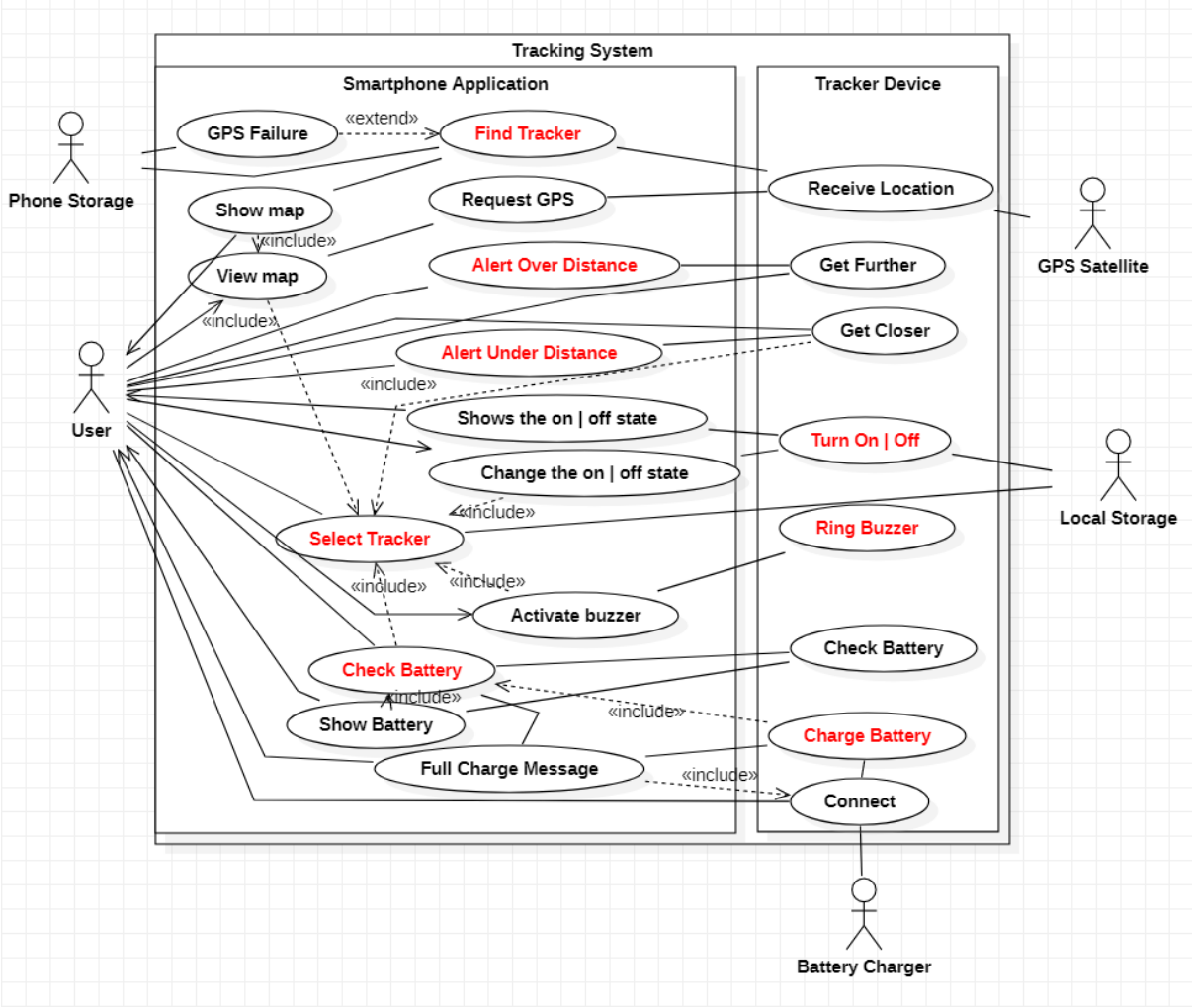
e. Alert Setting Page



[Function 8] Alert Setting Page

Alert setting page would exist for the default case and for the specific tracker each. Clicking the “Alert Settings” button at the main page would show the default alert settings, and clicking the “Change Alert Condition” button would show the specific alert settings for each tracker. Users can change the condition of which the tracker would alert the users in which way. “Alert condition” enables users to choose when the tracker would ring: when the tracker is over the threshold distance or under the threshold distance. Users can also choose the threshold distance, which should not be too far for the sensor to operate. Finally users can choose how the tracker would alert them: by sound, vibration, or no sound. Also users can choose if the app would show a pop-up notification message.

D. Use Case Diagram & Description



[Figure 9] Use Case Diagram

Use case name	Select Tracker
Related Requirements	R.F.C.1, R.F.C.2, R.F.C.3, R.F.D.1, R.F.D.2
Goal in Context	User chooses which tracker to utilize using the smartphone application.
Preconditions	Tracker must be in bluetooth connection range and have enough power.
Successful End Condition	Users operate features on a selected tracker or get information on its status.
Failed End Condition	The connection of the tracker and smartphone does not occur and the user gets an error message
Primary Actors	User
Secondary	None

Actors		
Trigger	User asks the smartphone application to select a certain tracker.	
Main Flow	Step	Action
	1 2 3	The user selects which tracker they will utilize on the smartphone application. From the internal storage, the application chooses which tracker to operate. The application informs the users that the connection has been established.
Extension		

Use case name	**Find Tracker**	
Related Requirements	R.F.C.2	
Goal in Context	User receives the precise location of the tracker using GPS.	
Preconditions		
Successful End Condition	User receives information about the location of the tracker on the map.	
Failed End Condition	Tracker device does not send the tracker's GPS information.	
Primary Actors	User	
Secondary Actors	GPS Satellite, Phone Storage	
Trigger	User asks the smartphone application to load a map that shows where the tracker is.	
Main Flow	Step	Action
	1 2 3 4 5 6 7	The user selects the tracker to utilize. The user chooses to view the tracker on the map. The application sends a signal to the tracker for request. The tracker receives GPS location from the GPS satellite. Tracker sends the information of the location to the application. The application saves it in the smartphone storage. The application shows the map with the tracker location.
Extension	5.1 5.2	Smartphone application does not receive any bluetooth signal from the tracker Smartphone application utilizes the most recently saved location of the tracker to draw a map.

Use case name	**Ring Buzzer**	
Related Requirements	R.F.C.1, R.F.C.2	
Goal in Context	User receives the location of the tracker by making tracker buzz	
Preconditions	None	
Successful End Condition	The buzzer inside the tracker rings.	
Failed End Condition	The connection of the tracker and smartphone does not occur and the user gets an error message.	
Primary Actors	User	
Secondary Actors	None	
Trigger	User asks the smartphone application to make the tracker's buzzer ring.	
Main Flow	Step	Action
	1 2 3 4	The user selects the tracker to utilize. The user chooses to activate the tracker's buzzer. The tracker receives the signal. The internal buzzer rings.
Extension		

Use case name	**Alert Over Distance**
Related Requirements	R.F.A.1, R.F.A.2
Goal in Context	Smartphone application sends alert to user when the signal from the device gets weaker, expecting the user might lose the target belonging.
Preconditions	The device should be attached to the target belonging. The tracker should be set up to give alarm when over threshold distance.
Successful End Condition	User notices that he/she left his/her target belongings behind.
Failed End Condition	The user cannot receive an alert from the smartphone application.
Primary Actors	User
Secondary Actors	None

Trigger	The power of the signal between the device and smartphone is under a certain threshold.	
Main Flow	Step	Action
	1 2 3	The user, with the smartphone, gets further away from the target belonging. The application detects the signal from the tracker's internal transmitter gets weaker until a certain threshold. The application alerts the user in a predetermined way.
Extension		

Use case name	**Alert Under Distance**	
Related Requirements	R.F.A.2	
Goal in Context	When the tracker gets closer than a certain distance, the smartphone application gives a signal.	
Preconditions	The smartphone application should be connected to the tracker. The tracker should be set up to give alarm when under threshold distance.	
Successful End Condition	Signal appears in the application.	
Failed End Condition	The signal does not appear even though the tracker is right next to the user.	
Primary Actors	User	
Secondary Actors	None	
Trigger	The tracker gets closer to the user (and his/her smartphone)	
Main Flow	1 2 3 4	The user selects the tracker to keep track of. The tracker gets close. The application detects the signal from the tracker's internal transmitter gets stronger until a certain threshold. The application alerts the user in a predetermined way.
Extensions	1.1	If the connection fails the application shows the error message to the user.

Use case name	**Turn On | Off**
Related Requirements	R.F.D.2
Goal in Context	Users should be able to manage multiple devices. Users can choose

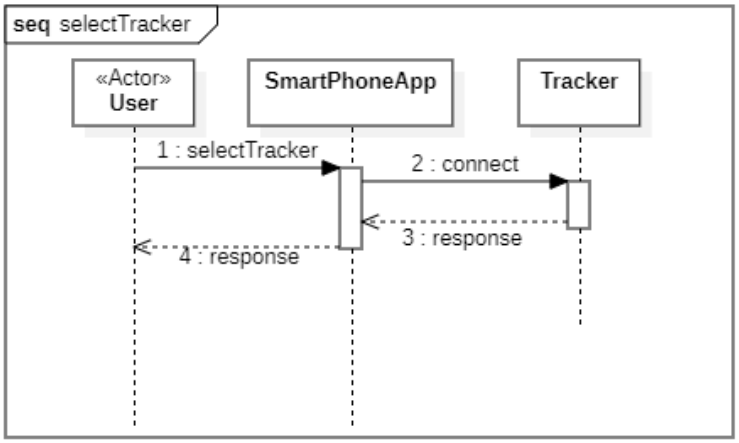
	which device to activate to prevent unnecessary alerts from not using devices.	
Preconditions	Smart tracker devices should be properly registered on the smartphone application.	
Successful End Condition	The smartphone application will not alert for smart tracker devices which are turned off, but still keep tracking on all devices. The smartphone application marks the target device as “turned off” in its device list. The target device changes its state to standby mode.	
Failed End Condition	The smartphone application should not mark the target device as “turned off” and alert if the distance from the target device is over threshold. The target device should not change its state.	
Primary Actors	User	
Secondary Actors	None	
Trigger	The user executes the smartphone application and requests to turn on/off a certain smart tracker device.	
Main Flow	Step	Action
	1 2 3 4 5	The user selects the tracker to utilize. The user changes the on/off state using the application. The tracker receives the signal and turns on / off. The application receives the changed state from the tracker. The application shows the changed state to the user.
Extension	4.1 4.2	The smartphone application does not receive any message from the target device. The smartphone application shows a connection loss error message to the user.

Use case name	**Check Battery**
Related Requirements	R.F.D.1
Goal in Context	User requests the battery leftover percentage from the smartphone application.
Preconditions	The tracker should be connected to the user’s smartphone application.
Successful End Condition	The smartphone application sends the current percentage of the remaining battery successfully to the user.
Failed End Condition	The application for the current battery is rejected.
Primary Actors	User

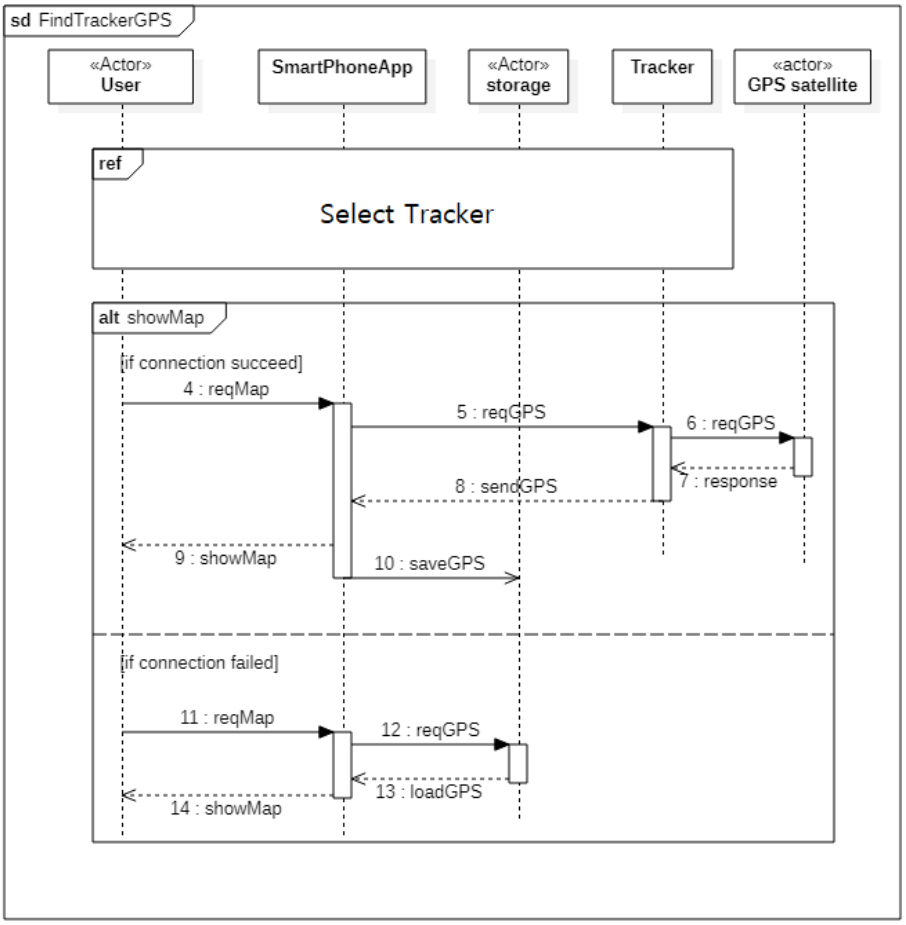
Secondary Actors	None	
Trigger	The user asks for the remaining battery of the tracker.	
Main Flow	Step	Action
	1 2 3 4	The user selects the tracker to utilize. The user chooses to check the battery. The tracker gets the signal, checks the battery and sends it back. The application shows the remaining battery to the user.
Extensions	3.1 3.2	The application fails to get connected to the tracker. The application sends an error message back to the user.

Use case name	**Charge Battery**	
Related Requirements	R.F.B.1	
Goal in Context	User charges the battery of the tracker.	
Preconditions		
Successful End Condition	The tracker gets fully charged.	
Failed End Condition	The battery percentage of the tracker does not change after charging.	
Primary Actors	User, Battery Charger	
Secondary Actors	None	
Trigger	The user connects the charger to the charging port of the tracker.	
Main Flow	Step	Action
	1 2 3 4	The user connects the battery charger to the tracker. The embedded battery of the tracker starts to be charged. After the battery is charged 100%, the application receives the full charge signal. The application shows the full charge message to the user.
Extensions	2.1 2.2 2.3	The connection between the charger and the tracker is unstable. The charger sends an error signal to the smartphone application. The application shows an error message to the user.

E. Sequence Diagram

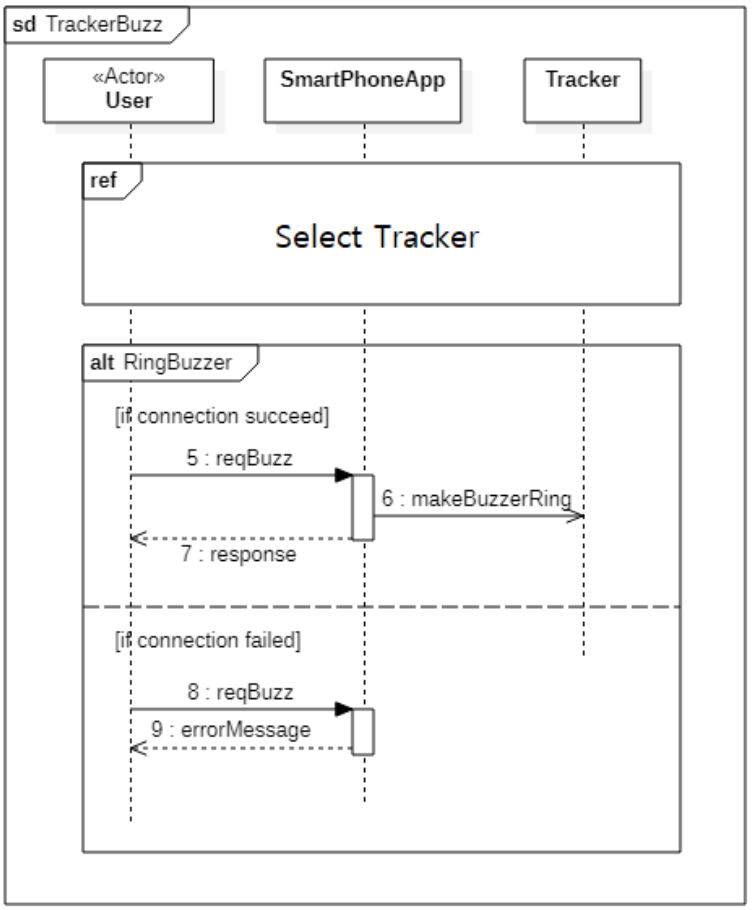


[Figure 10] Sequence diagram of 'Select Tracker' use case



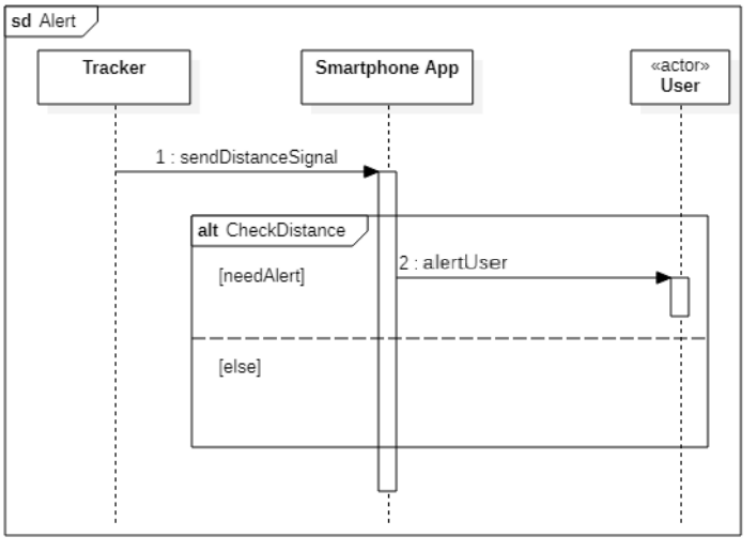
[Figure 11] Sequence diagram of 'Find Tracker' use case

The saveGPS function does not require a response message as the storage simply receives the GPS information and stores it in its memory.



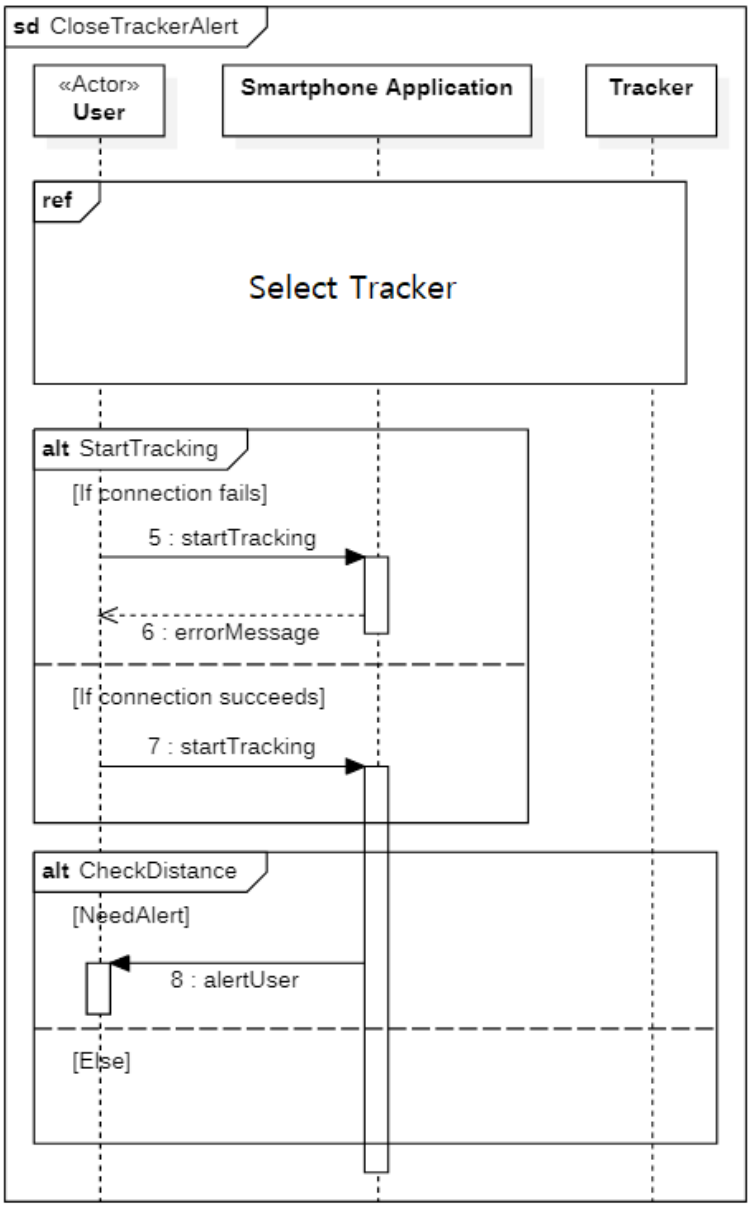
[Figure 12] Sequence diagram of 'Ring Buzzer' use case

The makeBuzzerRing function does not require a response message because the application simply rings the tracker to help the user.



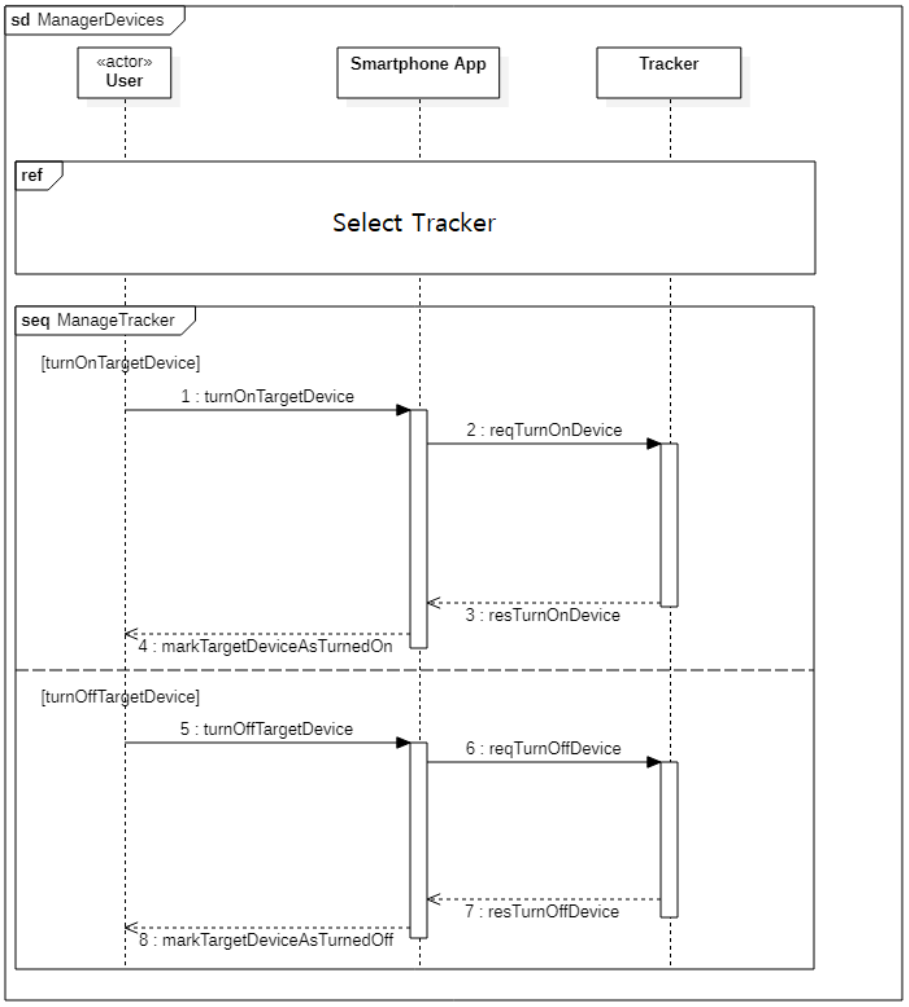
[Figure 13] Sequence diagram of 'Alert Over Distance' use case

The alertUser function does not require a response message because the application simply alerts the user when meeting conditions.

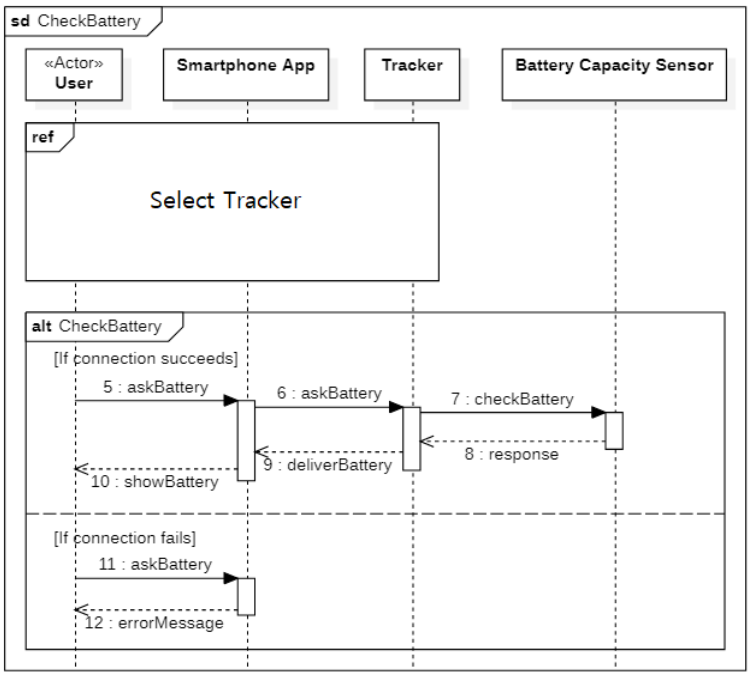


[Figure 14] Sequence diagram of 'Alert Under Distance' use case

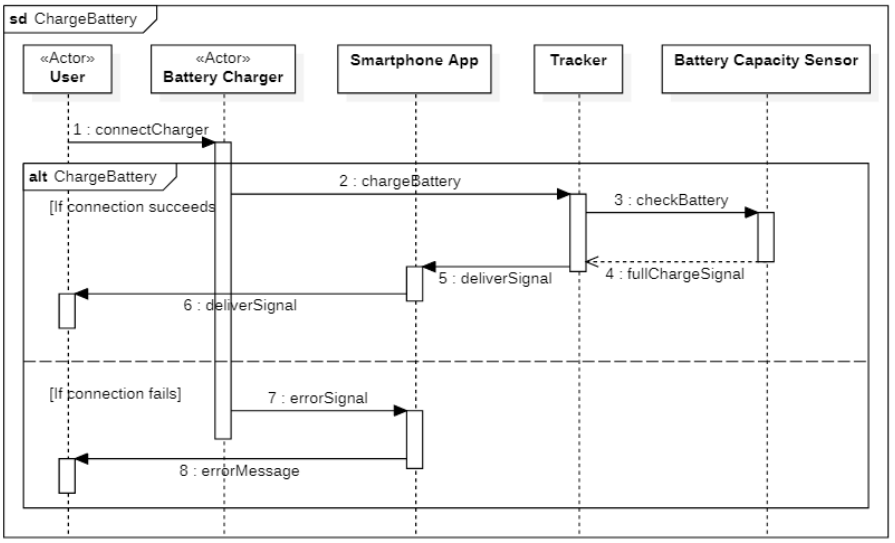
The alertUser function does not require a response message because the application simply alerts the user when meeting conditions.



[Figure 15] Sequence diagram of 'Turn On | Off' use case



[Figure 16] Sequence diagram of 'Check Battery ' use case



[Figure 17] Sequence diagram of ‘Charge Battery’ use case

3. Acknowledgement

System Overview

- Requirements - 이정재
- Tasks - 김준범

System Design

- System Architecture - 김준범
- Class Diagram - 이정재
- User Interface Design - 박윤정
- Use case Diagram & Description - 박윤정
- Sequence Diagram - 박윤정