

School of Computing

FACULTY OF ENGINEERING AND PHYSICAL SCIENCE



UNIVERSITY OF LEEDS

Final Report

Beacons in the Office ELDER02

Daniel Perez Cascon

**Submitted in accordance with the requirements for the degree of
BSc Computer Science**

2019/2020

40 credits

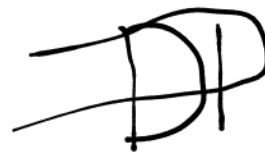
The candidate confirms that the following have been submitted:

Items	Format	Recipient(s) and Date
<i>Intermediate Report</i>	<i>Report</i>	<i>Supervisor, 18/12/19</i>
<i>External Client Form</i>	<i>Online Submission</i>	<i>Supervisor, 22/12/19</i>
<i>Application</i>	<i>Software Codebase</i>	<i>Supervisor, assessor (06/05/20)</i>
<i>Final Report</i>	<i>Report</i>	<i>Supervisor, assessor (06/05/20)</i>

Type of Project: Exploratory Software

The candidate confirms that the work submitted is their own and the appropriate credit has been given where reference has been made to the work of others.

I understand that failure to attribute material which is obtained from another source may be considered as plagiarism.



(Signature of student) _____

Summary

In many situations it is a necessity to be able to track who is within a building at a given time – be it for security or for payroll reasons. There have been countless different systems through which this has been done, but as technology advances our solutions need to advance with it too.

In this exploratory software we will investigate the viability of using Bluetooth Low Energy or BLE (in the form of beacons) and android smartphone applications to track whether a user is inside a building or not. In this scenario we will assume that the situation in which our user finds themselves in, is one in which they need to efficiently be able to notify an external system as to whether they are in a specific close off environment at a given time.

We will deliver a single smartphone application and a user guide manual about preferred configuration for the beacons.

Acknowledgements

I would like to thank my project supervisor Marc de Kamps for helping me with any questions I had thought this project.

I would also like to thank my family. My father Jose Carlos who has been an amazing role model who I aspire to be like, an amazing father and a great inspiration in regard to my pursuit for knowledge in the technical field of computing. My mother Alicia, for being the most caring mother I could have asked for and always being there to support me and encourage me to be the best person I could be.

Finally, I would like to thank my housemates Alexander Lawrence, Arun Mehan, and Mishanth Feinstein. Thank you for being very supportive while I stressed out about my beacons.

Table of Contents

Summary	iii
Acknowledgements	iv
Table of Contents.....	v
Chapter 1 Introduction.....	1
1.1 Project Justification.....	1
1.2 Aim.....	1
1.3 Project Objectives	2
1.4 Deliverables.....	2
Chapter 2 Planning and Project Management.....	3
2.1 Methodology.....	3
2.1.1 Waterfall Method.....	3
2.1.2 Agile Method.....	4
2.1.3 Decision and Justification.....	4
2.2 Project Plan.....	5
2.2.1 Research (Weeks 1-10)	5
2.2.2 Sprint 1 (Weeks 11-13)	5
2.2.3 Sprint 2 (Weeks 14-16)	6
2.2.4 Sprint 3 (Weeks 17-19)	6
2.3 Risks	6
Chapter 3 Background Research.....	7
3.1 Existing Tracking Solutions.....	7
3.1.1 RF and WLANs.....	7
3.1.2 GPS.....	7
3.1.3 Bluetooth and BLE	8
3.2 What is BLE?.....	9
3.3 Technology and Software	10
3.3.1 Beacon Communication.....	10
3.3.2 Smartphone application	10
3.3.3 Slack.....	10
3.3.4 Software Prototyping.....	11
Chapter 4 Implementation	12
4.1 Beacon Communication Service.....	12
4.1.1 Region Entrance/Exit Detection (EntranceExitDetector.java)	12

4.2 Database & Storage	13
4.3 Base Android Application.....	14
4.3.1 Status Menu (HomeFragment.java)	15
4.3.2 Beacon Search (SearchFragment.java)	15
4.3.3 Beacon Location (BeaconManagementFragment.java).....	16
4.3.4 Inter Fragment Communication.....	18
4.4 Slack API manager.....	18
4.5 Evaluation of Implementation	18
4.5.1 Sprint 1 (Weeks 11-13)	18
4.5.2 Sprint 2 (Weeks 14 - 16)	20
4.5.3 Sprint 3 (Weeks 17 - 19)	21
4.5.4 Difficulties Encountered	22
Chapter 5 Distance Calculations & Positioning/Detection.....	24
5.1 Beacon Distance Inaccuracy and Optimization.....	24
5.1.1 RunningAverageRssiFilter (Default).....	24
5.1.2 ARMA (Autoregressive Moving Average).....	26
5.1.3 Distance Algorithm conclusion	26
5.2 Beacon Positioning/Detection.....	26
5.2.1 Beacon Detection Thresholding	26
5.2.2 Beacon Positioning	26
Chapter 6 Testing.....	28
6.1 Multiple smartphones	28
6.1.1 Correct Output	28
6.1.2 Time to Output.....	29
6.2 Tests Conclusion.....	29
Chapter 7 Ethical, Social, Legal and Professional Issues.....	30
7.1 Ethical Issues	30
7.1.1 Tracking.....	30
7.1.2 Disability and Special Considerations	30
7.2 Security Issues	30
7.2.1 BLE Pairing & General BLE Maleficence	31
7.2.2 Slack Webhook Token	31
7.2.3 Relevance to our System.....	32
7.3 Social Issues	32
7.3.1 Workers Rights	32

7.3.2 Permissions	33
7.4 Legal Issues	33
Chapter 8 Conclusion & Personal Reflection.....	34
8.1 Objectives Consideration.....	34
8.1.1 Main Objectives	34
8.1.2 Additional Objectives	35
8.2 Overall Conclusion	35
8.3 Possible Further Work	36
8.3.1 Physical Lock Implementation.....	36
8.3.2 Interaction with Other Mobile Technologies	36
8.4 Self Appraisal and Personal Reflection.....	37
Appendix A: External Materials.....	39
A 1 External Materials	39
Appendix B: Installation and Expected Usage.....	40
B 1 Download and Installation	40
B 2 Example of Usage and Expected Screens	40
B 2.1 Status Screens	40
B 2.2 Search Screens.....	41
B 2.3 Map Screens	41
B 2.4 Beacon Management Screens	42
B 3 User Guide.....	43
B 3.1 Beacon Registration	43
B 3.2 Slack Status Name Change.....	43
B 3.3 Beacon Management Entrance Change	43
B 3.4 Beacon Position Management.....	43
List of References.....	45

Chapter 1

Introduction

1.1 Project Justification

The use of beacons for tracking systems has become more and more prevalent throughout the personal and professional sphere over the last couple of years. In the office, employers need to keep track of how often their employees are leaving/entering the office to be able to correctly track their wages. In warehouses, beacons can be used to quickly track inventory and to make sure that products of high value are not being taken out of their safe storage area [1]. Furthermore, the use of beacons for people tracking can help in emergency situations such as building fires or hostage situations, where it is essential to be able to check if everyone who was in the building has safely exited [2]. Overall, BLE (Bluetooth Low Energy) beacons are a fast and efficient way to track any sort of item and are much cheaper (at a 60 to 80 percent lower cost) than traditional Bluetooth based alternatives [3].

In a busy office environment there will always be a chance of the employee forgetting to log their times in office tracking systems, this could lead to greater costs for the employer/client and potential health and safety risks. Most of the time it is not possible, nor is it desirable, to have a supervisor or manager micromanaging where each employee is at any given time. Therefore, there needs to be a system in place to be able to do simple timekeeping and management. We believe that by using BLE beacon technologies, the potential damage or additional costs caused by forgetful/careless employees can be limited to a certain degree. Therefore, we will be studying the feasibility and the application of BLE beacons in relation to tracking the flow of people in and out of a closed-off environment, such as an office.

1.2 Aim

The main aim of the project is to create a smartphone-based application that will allow our users to be checked in and out of an office time tracking system with the assistance of BLE beacons.

There is an obvious ethical issue with the tracking of individuals in any environment (as explained in Section 7). Therefore, it will be an additional aim to develop the application with special consideration for the privacy and security of the data being handled.

1.3 Project Objectives

The main project will be broken down into a set of sub-objectives which will culminate in an end application that allows for the tracking of individuals.

The initial main sub-objectives will be:

1. Investigation into existing products that use BLE beacon technologies - to see what sort of concepts we can use in our designs.
2. Research into appropriate existing technologies and tools that will allow us to develop the application and integrate it with the BLE beacons. This will include SDKs, APIs, and any other code-related tools.
3. Basic application development to interact with the beacons and to allow us to test said beacon's limitations. The limitations including the range, power, and reliability of the beacons in different closed-off environments.
4. Developing a more robust application that will be able to interact with an office timekeeping service or Slack.

There will also be a set of additional sub-objectives which will only be attempted if the initial project scope is achieved in due manner. These additional objectives being:

1. Implementing the interaction of multiple beacons with multiple smartphones in a given time and a given area.
2. Testing the feasibility of using a triangulation system to deduce, with more accuracy, if the person being tracked is either entering or leaving the office environment.

1.4 Deliverables

The project deliverables will be: the source code of the smartphone application and any other pieces of software developed, a report detailing the research, testing and evaluation process, and a detailed instruction document about the setup of the BLE beacon hardware for effective use of the system.

Chapter 2

Planning and Project Management

2.1 Methodology

The development methodology will be essential to ensure that the minimum amount of time is wasted and to ensure that we reduce the risk of developing inefficient or bug-filled code. Due to this, we will discuss and compare major schools of development methodology to try and find one that suits our project best.

2.1.1 Waterfall Method

Waterfall is what is considered a 'heavyweight' development methodology. It is a methodology which requires a large amount of documentation before any development is started and it tends to not change that documentation as the development process advances. Waterfall is based on a six step sequential non-adaptive process (as seen in figure 2.1) where the development team cannot advance onto the next set of tasks before finishing the current stage [4]. Its rigidity leads to a non-adaptive development process which is not welcoming to change or influence from the client or user base. Additionally, due to its delayed testing period it encourages haphazard coding practices which can lead to an ineffective and bloated codebase.

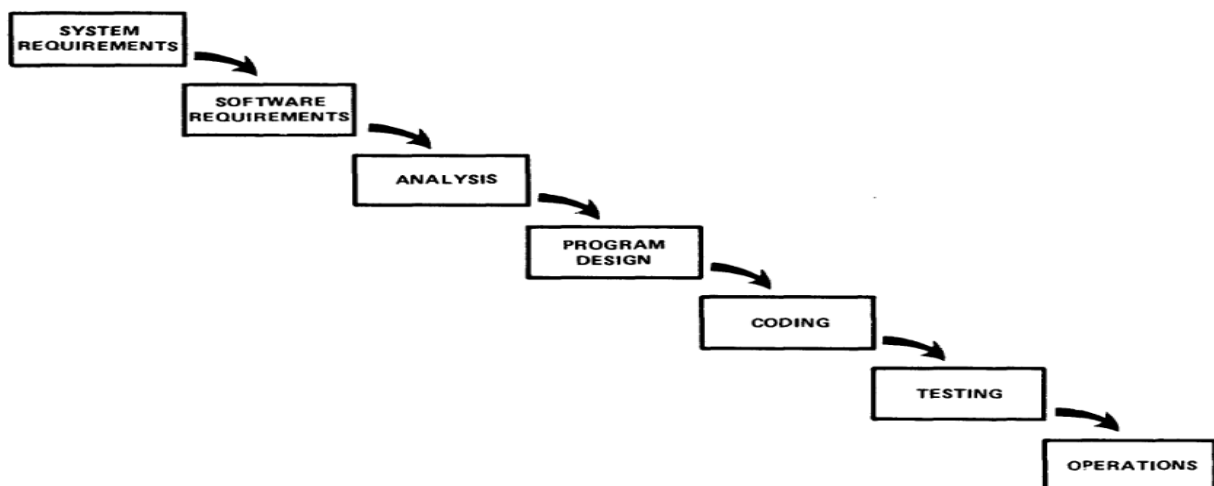


Figure 2.1 Waterfall Model [4]

2.1.2 Agile Method

Agile is a cycle-based methodology which relies on the steps: analysing, defining, designing and, testing. These steps are repeated on the product after it is deployed (as seen in figure 2.2).

The agile methodology is a flexible and adaptive software development approach. In this methodology there is a heavy emphasis on being able to change the product depending on customer collaboration and feedback - which is required at a constant rate [5]. This methodology does not require a lot of initial documentation which makes the development plan vaguer and the overall process much less predictable. However, the lack of rigidity in the planning is a trade-off which allows for rapid design change and response to feedback.

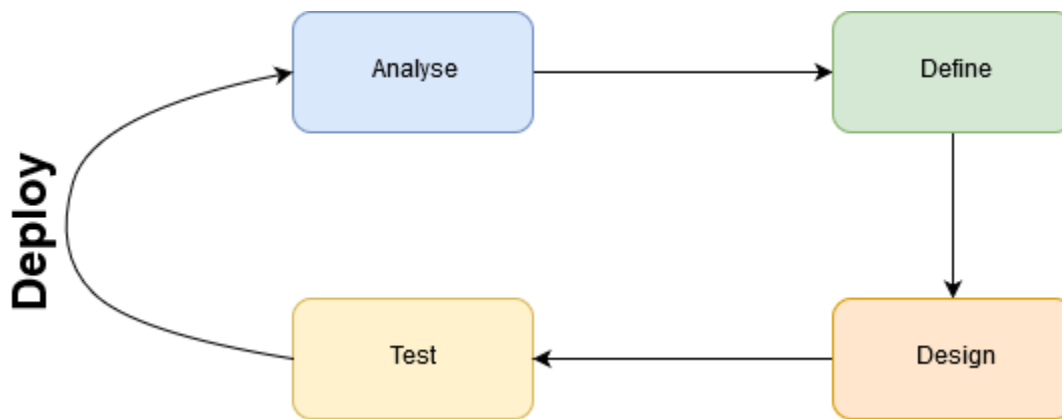


Figure 2.2 Agile Model

2.1.3 Decision and Justification

Due to the exploratory nature of this project, it is likely that we will need to make constant amendments and changes to the product specification and to the general design. Agile methodology would allow for these fast-paced design changes, while waterfall's rigid structure would provide too much resistance. Furthermore, an exploratory project will always contain a lot of errors and bugs in the early stages of the project since we are not very accustomed to the technologies we are working with. If we were to leave testing to a later stage of the production process, as is done in waterfall, we would be at risk of having to rush bug fixes due to our approaching deadlines – this might prove critical to the overall project.

Therefore, we will be using an agile methodology. We will be using a method of agile which is based on Feature Driven Development (FDD) [6]. FDD will ensure that we work on a specific set of features one at a time before moving on to the next. Our features will be assigned to a

specific 3 week sprint. The sprint breakup is represented using a Gantt chart (as seen in figure 2.3).

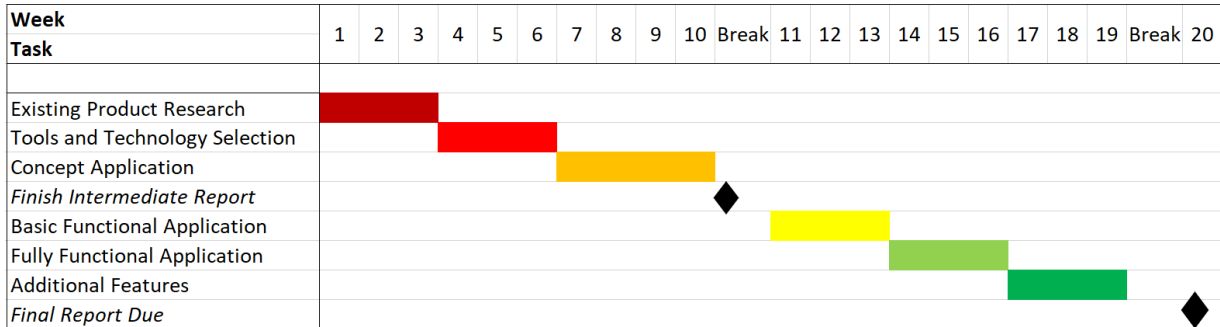


Figure 2.3 Gantt Chart

2.2 Project Plan

2.2.1 Research (Weeks 1-10)

These weeks will be allocated to the foundational research section of the project. We will try and research existing publicly accessible projects and we will look into how BLE beacons are being used in a professional environment. Furthermore, we will carry out research into tools, such as SDKs, provided by beacon manufacturers to decide what we will be using for the final product in the following weeks.

The final week of this research time will be a prototyping sprint where we will try and prove some basic concepts by developing a simple android application with limited functionality. We will also run some tests on the BLE beacons we have been provided to see what sort of limitations we might encounter in the environment where they will be deployed.

2.2.2 Sprint 1 (Weeks 11-13)

This initial development sprint will be used to create a bare bones smartphone application which will interact with a single beacon. The interaction will consist of seeing if the beacon is capable of being registered onto the application and to test if an external server will be required for data storage and management.

2.2.3 Sprint 2 (Weeks 14-16)

This secondary development sprint will seek to flesh out the application a bit more. We will try and calculate the distance of the user from the actual beacon, and we will implement an algorithm to detect whether the user is moving in or out of the office. This might require a secondary beacon to be implemented. When this is done, we will set up a Slack hook with which to update the status of the user and to record whether they are in/out of the office.

2.2.4 Sprint 3 (Weeks 17-19)

These final weeks will be used for testing and adding additional non-vital features. We will test to see how the application functions differently if multiple smartphones are in the vicinity and we will test to see if a tertiary beacon can be set up to more accurately triangulate the position of the entering/exiting user.

2.3 Risks

Some of the major risks arise since we have a relatively low understanding of the development process with BLE beacons. We will attempt to tackle this issue by starting the development process in a relatively early sprint, and by creating basic prototypes towards the end of the research sprint (as seen in Section 2.2.1). This will allow us to assess what we are capable of doing with the beacons.

Additionally, a risk is that Bluetooth technology can be widely affected by its environment. For example, an environment which is densely populated by IoT systems is likely to affect the reliability of BLE beacon readings [6] due to a lot of those systems using the same 2.4 GHz frequency band as BLE beacons [7]. We will try and mitigate this by running tests on the ideal positioning of beacons and by using pre-existing research to try and produce an optimum positioning system.

Finally, while the beacons are low energy and so can stay constantly activated for prolonged periods of time this is not the case with smartphones. Many users will prefer to keep their Bluetooth disabled to preserve battery life and so our application will not be able to communicate with the beacons. This is an issue which we cannot avoid and so the most effective way of ensuring that this risk is mitigated is by having reminders in the workplace about the application e.g. posters and other visual cues.

Chapter 3

Background Research

3.1 Existing Tracking Solutions

There are a couple of existing tracking solutions using different types of technologies. It is important to know how these previous solutions were implemented to try and avoid any of the pitfalls they encountered.

3.1.1 RF and WLANs

The use of IEEE 802.11 WLAN networks to track the location of a user is a solution which has been considered by many companies for a long time. Microsoft's RADAR [8] is a solution which implemented RF (radio frequency) technologies using the above-mentioned standard with a mobile host to allow for user tracking. The use of RF was a change made to try and counteract the inefficiencies of using IR (infra-red) as IR struggled to handle situations with multiple users in a given space. One of the issues with RADAR and using RF technologies was that the mobile host required special hardware to detect these specific RF signals and so, might not be ideal for ease of use with smartphones. Furthermore, with multiple APs (access points) the mobile host would need to scan all channels at the same time to be able to read data from all the present APs as they operate on different channels due to the classical reuse requirement in cell-based networks [9]. This will obviously create a large overhead which will affect performance and battery life.

BLE beacons would not encounter the same issues as the previously stated solution as it does not require any additional hardware to be installed in the user's smartphones. This is the case as long as the smartphones are capable of detecting 'Bluetooth 4.0' [4]. Most smartphones will be able to detect and interact with the Bluetooth 4.0 standard since it has been used in mainstream smartphone products since 2011. Research done by 'Reacon Analytics' shows that the average smartphone user upgrades their device after less than 30 months [10]. Therefore, we can assume that the time scale of 96 months (the time since Bluetooth 4.0 was released) is a long enough time period for us to assume that most smartphone users will have devices that can interact with Bluetooth 4.0.

3.1.2 GPS

Originally, GPS could not be used for indoor tracking due to its lack of accuracy when it comes to positioning [11]. Furthermore, GPS suffers from signal fading [12] in indoor

environments and so would not be a suitable technology to use for our solution. However, with the integration of other technologies, researchers have managed to use GPS location systems for indoor tracking. The integration of GPS technologies with a worldwide network of reference stations allowed for more accurate tracking than a raw GPS detection system. This system is wireless, like BLE, but is less reliable than a system using BLE Beacons. It requires a constantly powered antenna receiver to be put in place, which is not only expensive, but is also power hungry and is required to have a constant uptime for the system to work. Our solution with BLE beacons does not suffer from the fading issue that GPS systems have. Additionally, it is much more energy efficient and can be done with beacons which can be moved around and do not require a single anchor point to be plugged in from, unlike with the above-mentioned system.

3.1.3 Bluetooth and BLE

Traditional Bluetooth has a relatively long scan time which means that it does not add a lot of value to tracking systems or localization systems [13]. BLE (non-traditional Bluetooth) beacons, however, do not have that issue and so are the technology we have chosen to use in our system. BLE beacons allow for low energy usage and mobility while also giving fast readings for distance from a given position. Furthermore, BLE beacons are battery powered, this allows for an increase in mobility and so makes the testing and deploying of the system more flexible.

Additionally, we chose BLE over traditional Bluetooth due to its device listening system. Unlike traditional Bluetooth, BLE is constantly in a sleep mode and only activates once a connection is initiated. These connection setups take a relatively small amount of time (a few mS) as opposed to traditional Bluetooth which can take more than 100mS to connect [14]. This fast connection would allow us to communicate data to predict the users distance from the beacon in a more rapid manner.

3.2 What is BLE?

BLE is a wireless personal area network technology that uses a low energy system to communicate data over 40 channels, each being 2 MHz wide and around the 2.4 GHz radio band (as shown in figure 3.1).

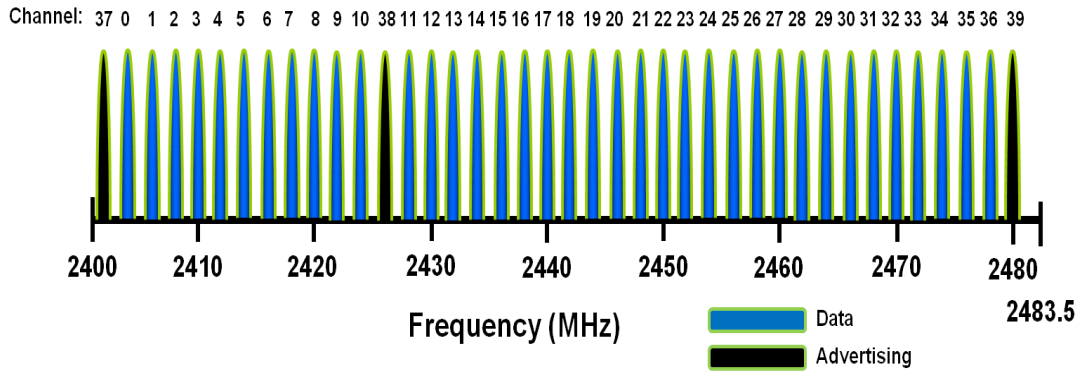


Figure 3.1 BLE Channels [14]

BLE uses two methods for data communication, one through its advertising channels, and one through its data channels. Channels 37,38 and 39 are known as advertising channels which are used to broadcast data to all devices in range and also to establish connections. The data transmitted by beacons contains a UUID (universal unique ID) which is a 128-bit value to identify which beacon is transferring data [7]. This can be used by applications to filter out data and only receive data from specific services.

The data channels are used after connection is established and communicate in a bidirectional manner [14]. At this stage a main question arises – how do we pick which data channel will be used to transfer data? The answer to that, is frequency hopping. Frequency hopping is done using a value which is set during the connection process (value hI) and the current channel to pick the next channel which will be used for event communication (as shown in equation 3.1). The hop value is a summation of the hI and a random value between 5-16.

$$f_{n+1}=(f_n+hop)mod37$$

Equation 3.1

As was previously mentioned, BLE devices will be affected by interference of other devices (such as Wi-Fi) in a similar radio frequency. However, only certain channels will be affected by this interference as all 40 channels operate in difference frequencies. To help remap packets from a channel which is being interfered with to a channel which is not, we use

adaptive frequency hopping. Adaptive frequency hopping lets us mark certain channels as 'bad channels' and to remap all the traffic for those bad channels to other 'good channels'.

3.3 Technology and Software

3.3.1 Beacon Communication

Due to our economic limitations we chose to use the PC062 Beacons [16] for our application development. These beacons came with an SDK and users guide included. However, the SDK was very poorly made and lacked basic documentation – due to this reason we chose to avoid it. After running some tests and trying to make some prototypes with their provided SDK we chose to abandon it and to change to the 'Android Beacon Library' SDK [17]. Android Beacon Library is open source and so has a lot more user made documentation and is tested more thoroughly than a closed company development kit would be.

3.3.2 Smartphone application

Our smartphone application will be developed using 'Android Studio'. This is due to our relative previous knowledge on the software at hand and also due to our understanding with the Java programming language. Furthermore, we do not own a platform to test IOS applications with and so they are outside of the scope of our project.

3.3.3 Slack

Since our client required the use of Slack as a way of notifying the office of whether a user had entered/exited the office we had to do some research into it.

Slack is described as a "workplace communication tool" which allows for instant messaging and for the deployment of bots to enhance the online messaging experience. Slack can be connected to external applications using their official API to change countless features of channels and users in a slack workplace.

Furthermore, Slack allows for webhooks to be used which allow for external applications to send messages to given API endpoints and for those messages to be posted to a specific channel. The system in place for these webhooks is quite secure as it uses authentication tokens which are required for any message to be posted.

3.3.4 Software Prototyping

We created a basic mock-up of an application interacting with two beacons we had setup to broadcast their MAC addresses using their advertising channels (as seen in figure 3.2). This was done using the opensource 'Android Beacon Library' - since it was the only library which efficiently and effectively let us do this, we chose to use that as our SDK for the rest of the project.

The mock-up listened for the advertisements from the beacons and every time it received one it would add the MAC address of the beacon to a list.

The application successfully interacts with both beacons, but we do see an issue with the consistency of communication between the smartphone application and the beacon as it does not register the advertisements in a consistent manner. This problem is very noticeable as one of the beacons was detected multiple times before the second beacon was detected, and they also seemed to have their advertisements detected at a different rate. An issue like this will need to be addressed in the real application and might not be present when interacting with connected data channels rather than with decoupled advertising channels.



Figure 3.2 Phone Prototype

Chapter 4 Implementation

4.1 Beacon Communication Service

The beacon communication service (BCS) uses the 'Android Beacon Library' to allow us to do two major functions: 1) detecting new beacons and registering them to the app using a monitoring system, 2) run a ranging feature in the background of the app to detect whether the user is in range of any registered beacons. The BCS will be constantly running in the background of the user's smartphone [17].

4.1.1 Region Entrance/Exit Detection (EntranceExitDetector.java)

Our algorithm implemented to detect if a beacon's region has been entered uses a set of LocalBroadcastManagers to communicate with the main BCS. The logic path that the detection algorithm used is described in figure 4.1.

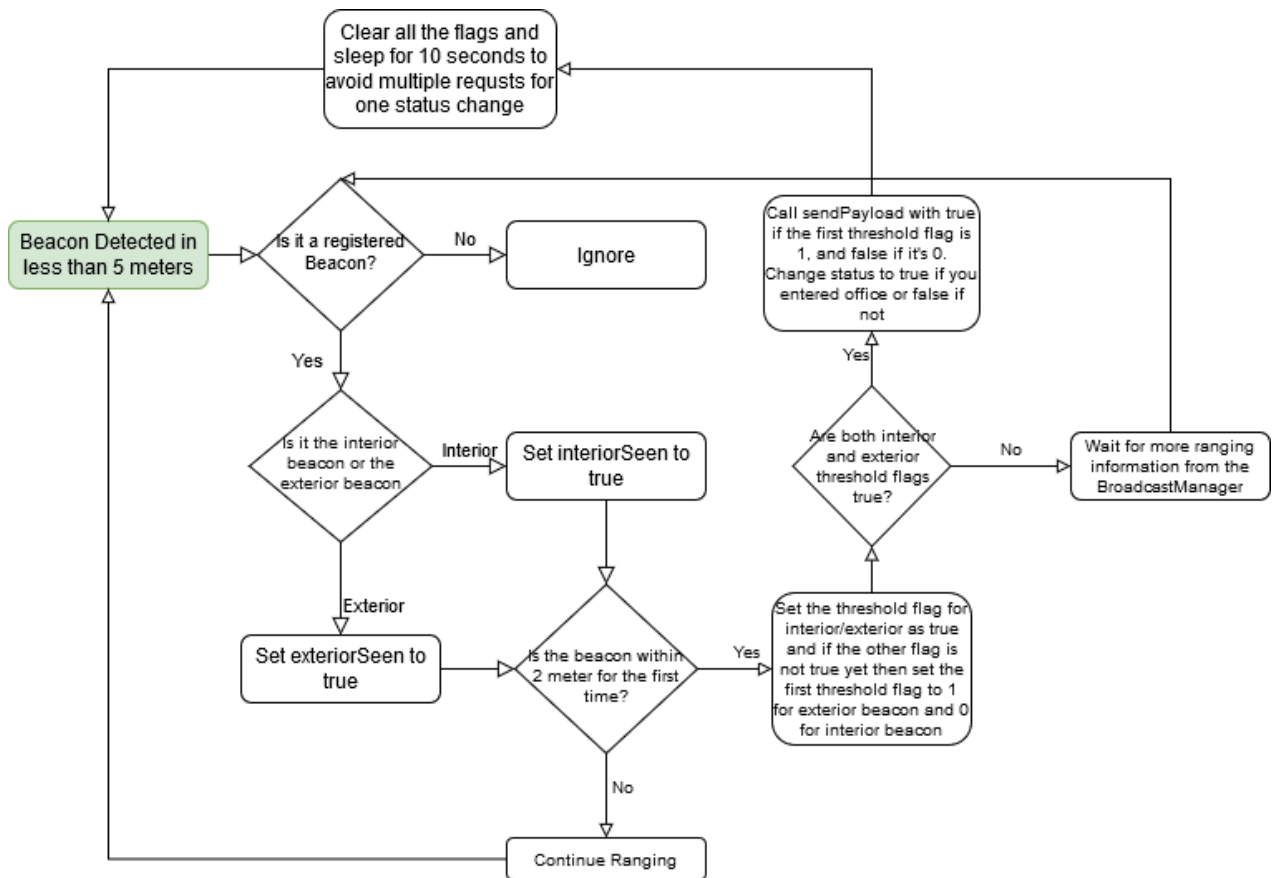


Figure 4.1 Detection Algorithm

One of the biggest challenges with this algorithm was defining a threshold for the beacon detection. We go into depth about how we defined our threshold selection approach in Sections 5 and 6.

4.2 Database & Storage

The information about registered beacons is stored in a table in an SQLite database called “beaconsRegistered” (as seen in figure 4.2). The entries are created using a DatabaseHelper object (described in the class DatabaseHelper.java) which was implemented using a singleton model. We chose to create it in that way since it is an object which would need to be used by multiple classes throughout the codebase – but should not be recreated every time it is called. To avoid repetitive dependency injection between classes we created it using a getInstance() method which allows whoever wants to use the database to acquire the current instance in use.

To facilitate the use of the database we created a BeaconEntry class (as seen in figure 4.3). This class has a set of getters and setters which allow you to access and change information which would be stored in the database. Through doing so we can easily create and pass a BeaconEntry object to the DatabaseHelper and have it be stored correctly. Furthermore, we can also expect to receive a BeaconEntry object (or a list of them) back from a DatabaseHelper after a query.

beaconsRegistered

beacons	
ID	String
name	String

Figure 4.2 Entry in beacons table

BeaconEntry.java

- beaconID	String
- beaconName	String
+ getBeaconID()	String
+ setBeaconID(String ID)	void
+ getBeaconName()	String
+ setBeaconName(String name)	void

Figure 4.3 BeaconEntry

Additionally, the BeaconIconObjects (explained in Section 4.3.3) will also be stored in this database. They will be stored in a separate table called “beaconImages” and will be stored as a set of parameters which will allow us to define each icon in the BeaconManagerFragment (as seen in figure 4.4).

beaconImagesRegistered

beaconImages	
ID	<i>String</i>
x_coords	<i>String</i>
y_coords	<i>String</i>

Figure 4.4 Entry in beaconImages table

4.3 Base Android Application

The complete application was created using a system of screen fragments which allowed us to change the utility which is displayed to the user quickly and efficiently. Fragments act as a sub activity and can each be called and changed on the click of a button from the bottom menu bar. The fragment itself “represents a behaviour or a portion of user interface” [18], therefore it allows us to change a specific section of the application to another one in a modular, reusable and efficient way. The use of fragments allowed us to add a more dynamic aspect to the application while not requiring the greater computational power of using different activities for different screens [19].

The general interaction path (as seen in figure 4.5) shows how the main activity is used as the singular backing activity and that the menu system simply updates the fragment UI.

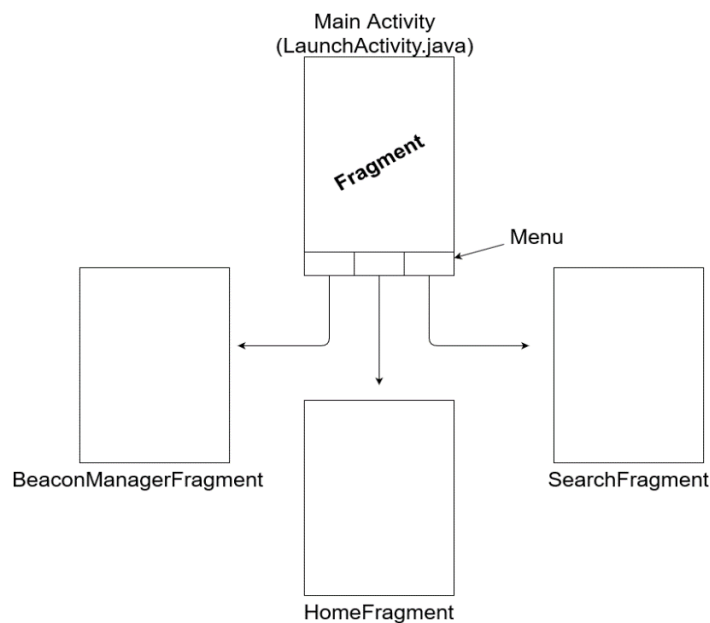


Figure 4.5 App Interaction Path

4.3.1 Status Menu (HomeFragment.java)

The status menu is a simplistic and relatively static screen which displays whether the user is in the office or out of the office. The status will be updated in the main activity in the moment in which the beacon tracker detects that the user has entered/exited the office.

This screen also offers a text input for the user to update the ID they want displayed next to their status message on slack. On clicking the “Update Name” button on the screen. The updated ID to be used will be stored in a SharedPreferences file.

4.3.2 Beacon Search (SearchFragment.java)

The search fragment contains a set of buttons that allow for the user to start seeking for beacons in their local area. This uses the BCS (explained in Section 4.1) to start monitoring for BLE advertisements. The new beacons will be displayed to the user as a list of IDs and will then give the user the option to register said beacons onto their application. If the detected beacons ID is pressed the user will see a new activity which gives more information about said beacon. If they attempt to register the beacon and it is already present in the database, the application will simply notify the user using a toast (a type of android notification) that this beacon is already registered (as seen in figure 4.6). Once the beacon is registered, we will be able to start ranging it and detecting the distance at which it is from the



Figure 4.6 Beacon Already Registered Toast

user's smartphone. All of the information about the beacons is stored in a local SQL database (explained in Section 4.2).

4.3.3 Beacon Location (BeaconManagementFragment.java)

The Beacon Location fragment is a fragment which is designed to allow the user to specify the location of their beacons on a given map (as seen in figure 4.7). The maps are custom made designs of what a standard entrance to a building would look like. They allow the system to be able to define the position of a beacon and to help accurately predict whether the user has entered the building or not. The map type can be selected with a set of radio buttons at the top of the tab and can be changed dynamically.

The beacon location system uses a dynamic graphical aspect where it allows the user to add a beacon and specify which beacon from the database it is. There is a cap to how many beacons can be added, and this is defined by the current number of registered beacons.

Once the "Add beacon" button is clicked, a new modal tab will open on top of the existing fragment and will allow the user to select the beacons by ID based on the beacons that they have previously registered.

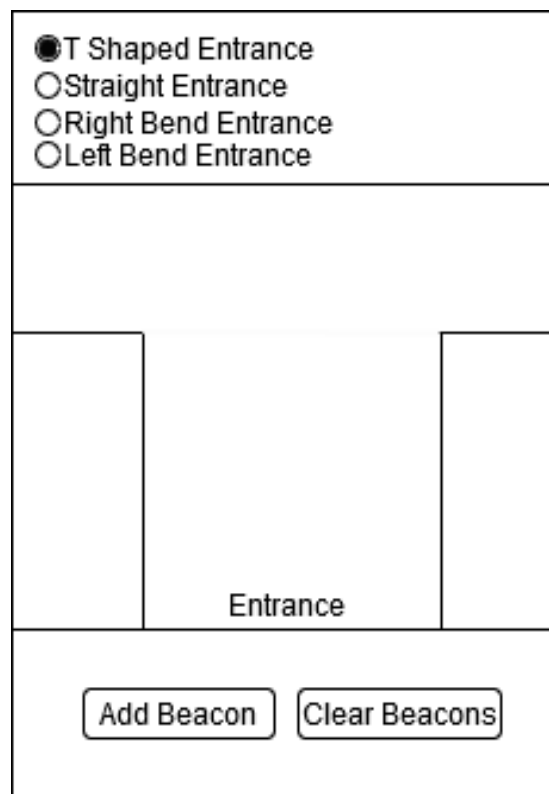


Figure 4.7 BeaconManagementFragment

To be able to manage the new beacon images added onto the fragment in a modular and efficient way we created a `BeaconIconObject` class. This class works in a similar way to the `BeaconEntry` objects but stores information which is required to display the new icons on the user's display (as seen in figure 4.8). Once a beacon is added to the screen, we will create a new instance of `BeaconIconObject` and associate it to said new beacon.

BeaconIconObject.java	
- icon	<i>ImageView</i>
- beacon	<i>BeaconEntry</i>
- coords	<i>float[]</i>
+ getBeacon()	<i>BeaconEntry</i>
+ setBeacon(BeaconEntry b)	<i>void</i>
+ getIcon()	<i>ImageView</i>
+ setIcon(ImageView i)	<i>void</i>
+ getCoords()	<i>float[]</i>
+ setCoords(float x, float y)	<i>void</i>

Figure 4.8 BeaconIconObject

The information about the `BeaconIconObjects` will be stored in the same database as the `BeaconEntries`. The preference of which entrance type is to be used will be stored in a `SharedPreferences` file. This type of file is not accessible from other applications and so is relatively secure.

The beacon location can be changed by dragging and dropping the beacons to a specific area within the map after it has been added. To be able to implement the image dragging and dropping we had to do research into the android drag/drop framework [20], this is again, something we had never worked with before and which proved to be rather complicated to get an initial understanding of. The initial selection of a beacon by starting to drag it around creates a brief shadow trail to indicate to the user where the image is being taken. When it is dropped, we update the x and y location of the beacon icon and then update the `BeaconIconObject` in the database.

4.3.4 Inter Fragment Communication

To avoid us having to rely too much on the main activity of the application we decided to use LocalBroadcastManagers to communicate information between fragments. This system uses the general concept of the observer model where each observer is notified by the subject of a specific change. By using this system, we are able to change the image for the status in the HomeFragment in real time without having to rely on variables on the main activity.

4.4 Slack API manager

The Slack API allows us to add a large amount of functionality to our application by adding a bot to our Slack workspace and then sending messages to our channel using said bot's authentication token. Thanks to the authentication token which is implemented with each POST message to the Slack channel we can ensure that it is only authorised users that are placing the requests. By doing this it ensures that even if the API's endpoints are leaked, we will not receive unauthorised messages.

The implementation of the Slack message manager (implemented in SlackApiObject.java) is done through the jSlack library [21] which allows us to utilise the webhook functionality in the Slack API.

4.5 Evaluation of Implementation

4.5.1 Sprint 1 (Weeks 11-13)

The initial sprint was delayed due to issues with the libraries at use and due to issues with android studio. The library we were using for the beacon communication required features of AndroidX which were no longer compatible with the current version. This led to constant crashes and therefore consumed a large amount of time in the development stage. The solution we found for this issue was adding backwards compatibility to older versions of AndroidX.

After the initial hurdles were solved, we worked on building up the prototype which we developed as part of our research and framework testing (as explained in Section 3.3).

Our plan stated that by the end of the sprint we should have:

1. Set up a bare bones android application
2. Implemented registration of beacons
3. Been able to interact with a single beacon

Out of those three only the first and the last were implemented fully. The first feature was implemented through a system of fragments in the application which the user could traverse by using a menu running along the bottom of the screen. The last feature had already been implemented in the prototype of the application. The second feature was only partially implemented since we chose to try and do bug fixes and delay said feature to a later sprint.

The reason why all the features were not implemented completely was due to a bug where the application required you to enter and exit the beacon's region (which is 40m of range) for the beacon to get detected by the application. This bug took up most of our research time and was only solved in the second development sprint. I attribute this issue to the vagueness in documentation which is often found in smaller open source projects.

The progress of this sprint can be seen in figure 4.9. The "Start Search" button allowed us to try and find beacons by their MAC address and these addresses would get displayed in a text box under the word "test".

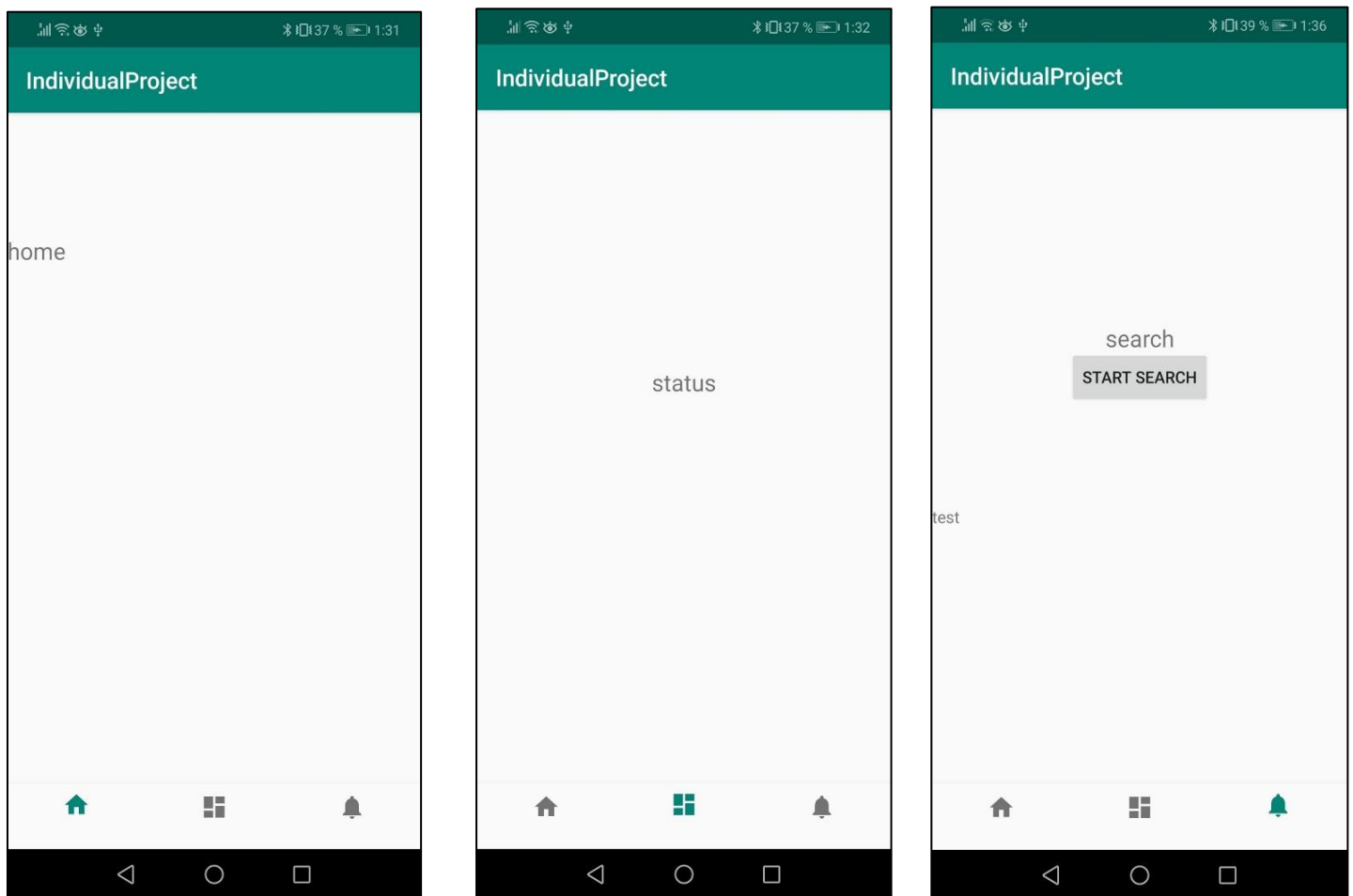


Figure 4.9 Progress After Sprint 1

4.5.2 Sprint 2 (Weeks 14 - 16)

During this sprint we hoped to handle the majority of the application's features, such as:

1. Distance calculation
2. Algorithm for entrance/exit detection
3. Possible secondary beacon implementation
4. Slack hook to communicate entrance/exit status
5. Fleshing out of app UI

Some of the features of this sprint were delayed as we still needed to implement a system which allowed us to register and store information about beacon. This was a feature which was left over from sprint one and which was fully completed in this sprint.

Additionally, we also encountered issues with the use of external images in an android project. JPG images would end up being pixelated and would get distorted when displayed on different devices, so we had to use SVG images. Our original source for SVG images produced images that were not formatted in a way in which android studio could process them – this took too long to realise and so we spent two days debugging this problem. This issue could be accredited to our lack of experience using android studio and with android app development in general.

The first two and the fourth feature were not completed during this sprint. This massive deviation from the initially defined schedule was caused by the overflow of features from the previous sprint and due to the vague definition of the fifth feature. Since feature five was defined in such a vague manner we worked on it for a large amount of time and kept pushing the implementation of other features back. We implemented a section which wasn't in the original application design (drag and drop of beacon icons onto a map) since we believed that it was required as part of this fifth feature, and so, due to this scope creep, we fell behind in the development of other features.

However, many positive changes were made to the application in this sprint. We managed to solve the issue of beacon detection that we encountered in sprint one. This issue was caused as regions were persisting from previous sessions and so by removing region persistence, we managed to solve this bug. Furthermore, we managed to implement a robust database system which allowed us to store, remove, and update entries about registered beacons. The android based database library was new to us and so becoming accustomed to it did take up a large part of this sprint's time. We additionally also improved the UI of the application and added the previously stated drag and drop feature, which we feel like added a degree of variety and challenge to the implementation, therefore adding value to the overall project.

The progress of this sprint can be seen in figure 4.10.

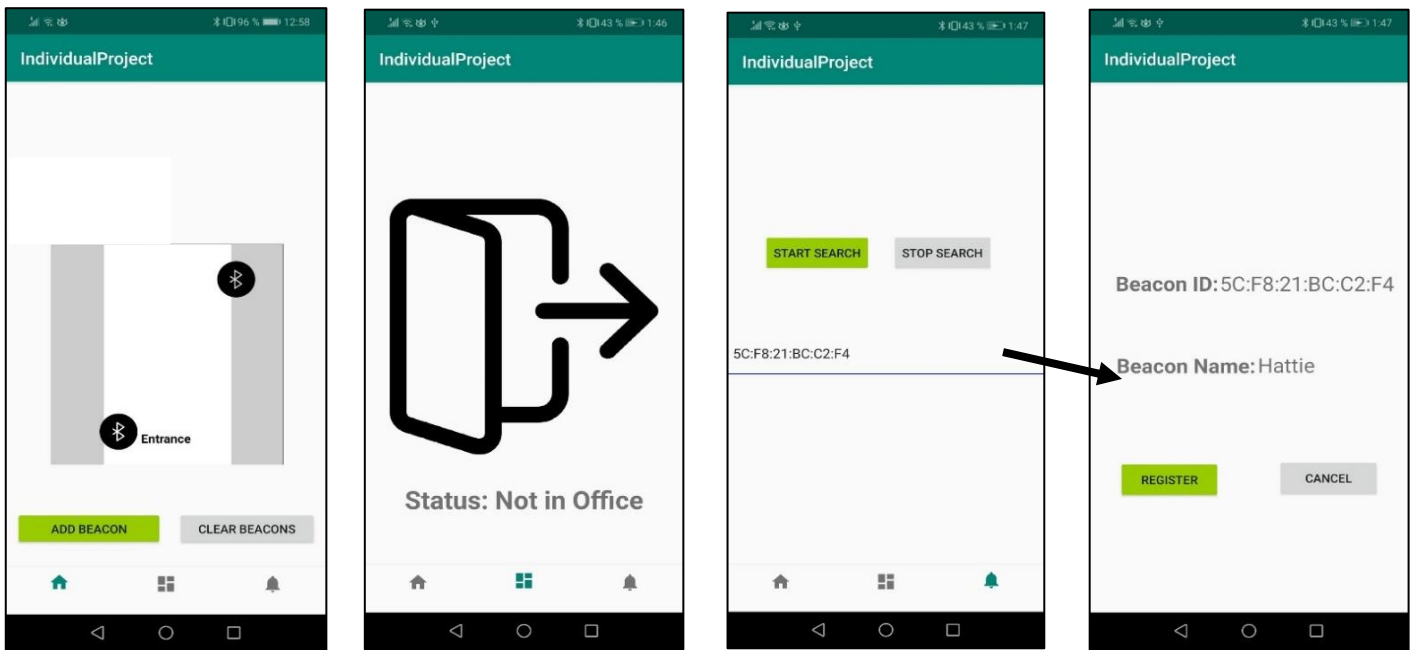


Figure 4.10 Progress After Sprint 2

4.5.3 Sprint 3 (Weeks 17 - 19)

This sprint was originally meant to be devoted to final touch-ups and testing. However, due to the issues of scope creep mentioned in the previous sprints, we had to work on main features throughout this sprint too.

In this sprint we implemented the algorithm for detecting if a user was entering/exiting an office space depending on the readings of two beacons. We also added interaction with the Slack server and added a secondary table into the database to allow us to correctly store and reinstate the user's configuration on beacon positioning. Furthermore, we kept improving the UI to make it more user friendly and logical.

To ensure that we can identify different smartphones on the Slack channel we also added in a text box to the main status fragment which allows us to be able to set a name that would be sent to the Slack channel with the main status message (as seen in figure 4.11). Along with this, we also added in the backend for the storage of the configuration using the SharedPreferences file system. The SharedPreferences system not only stores information about the name that will be associated with the smartphone on the Slack channel, but it also stores information about the preferred entrance map that the user configured.

The progress for this sprint is shown in figure 4.12.

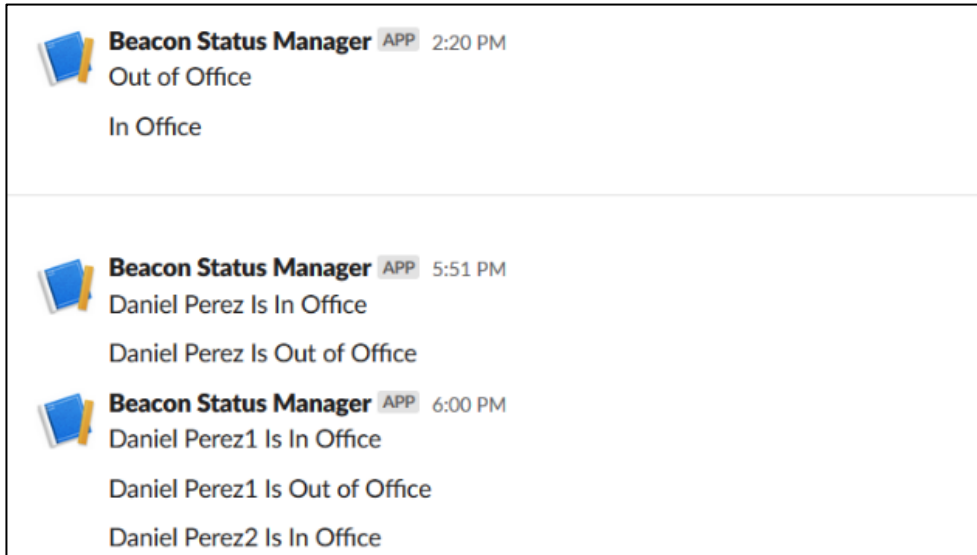


Figure 4.11 Slack messages before and after custom name option

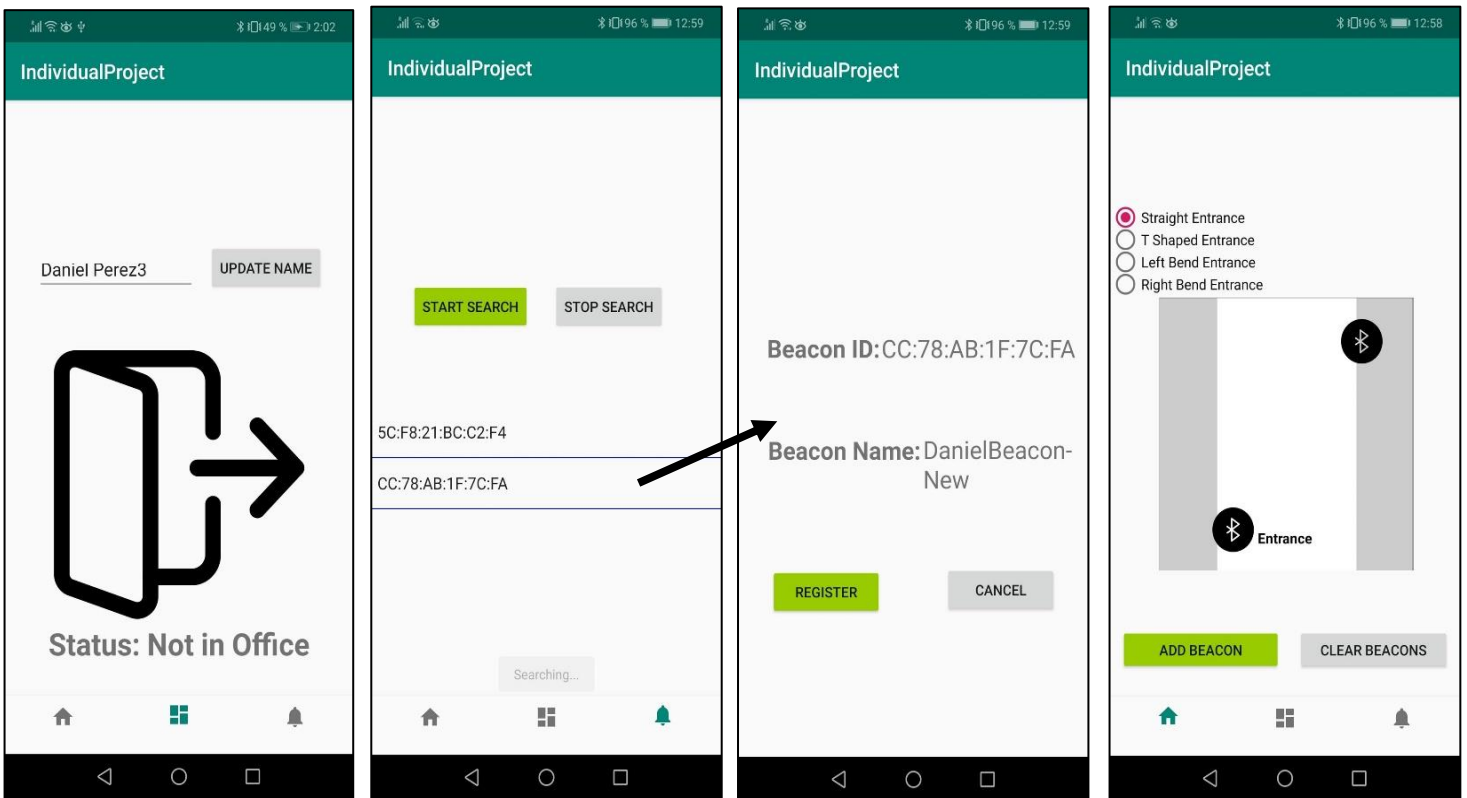


Figure 4.12 Progress After Sprint 3

4.5.4 Difficulties Encountered

Some of the main difficulties we encountered while trying to implement this application were around the use of android studio. While app development was taught in the university as a

module it is not one that we had taken. Therefore, building on my basic knowledge of android studio and how Java interacted with an android application was a very time-consuming aspect of the overall development process.

We also found BLE beacons to be extremely unreliable meaning that we could not check for changes in distance in a simple manner as we consistently had an uncertainty of at least 50cm – 150cm [22] for each measurement that came in from the beacons (as seen in Section 6.1). This uncertainty grew exponentially the further away the beacon was from the smartphone.

Furthermore, just like with android app development, working with Bluetooth in an application and specifically working with BLE was new to us. Learning about how the BLE libraries worked was very time consuming and is something that delayed the overall schedule of the first couple of sprints.

Finally, there was also a large issue with using Slack as our office tracking system. While Slack has a large amount of support for JavaScript and for C# it does not have an official API for Java. The recommended community driven API for Java had a set of conflicting modules with some of the other libraries we were using and so did not allow for compilation to be carried out as expected. This limitation meant that we had to find an alternative library which did not cause conflicts which again caused a delay in our original schedule. After testing multiple different open source libraries, we settled for jSlack. Due to jSlack not being an officially recommended library it did not implement every aspect of the Slack API and so limited our potential capability. It additionally does not have documentation which is as fleshed out as the original slack API's documentation.

Chapter 5

Distance Calculations & Positioning/Detection

5.1 Beacon Distance Inaccuracy and Optimization

Throughout some initial tests in the implementation stage we found that beacons seemed to be having a large amount of inaccuracy. After reading over the documentation of the library we found that this was expected and that “At close proximity of about 1 meter, you can expect to see distance estimates between 0.5-2 meters. At further distances you will see more variation. At 20 meters of actual distance, the estimate provided by the library may vary from 10-40 meters.” [22].

5.1.1 RunningAverageRssiFilter (Default)

The way in which applications calculate the distance at which a beacon is located from its host smartphone is through using the beacon’s service signal level (RSSI). The application will then find the distance by using a power regression against a table of known values through the formula:

$$d = A * (r/t)^B + C$$

where:

- d = distance
- r = RSSI measure
- t = known RSSI reference at 1 meter
- A, B, C being constants

While we could have calibrated the application to our specific android device, we could not access the webpage which contained the calibration models as the documentation pointed to a dead url.

To see how much of an accuracy issue we were going to be encountering we decided to carry out some first-hand tests using our own beacons (as seen in table 5.1 & 5.2). We decided to check the variation of the signal overtime, since the beacon library uses an averaging system over 20 seconds to try and get a more accurate reading. Over the 20 seconds it will throw out the top and bottom 10% of measurements and average the rest.

The disadvantage of this method of distance estimation is that it will lag behind and will tell you where the main device was relative to the beacon a set amount of time after the device has passed that location.

We believe that this testing is necessary as the different makes of beacons will inevitably mean that beacons will be made with different components of a varying quality. Since the beacons we used for testing were of rather low quality we wanted to test how uncertain their readings were.

Actual Distance (m)	App Readings at 1 second (m)			App Readings at 5 second (m)			App Readings at 10 second(m)			App Readings at 20 second (m)		
0.1	0.0	0.1	0.0	0.0	0.0	0.1	0.1	0.1	0.1	0.1	0.1	0.1
0.5	0.0	0.2	0.1	0.3	0.2	0.3	0.7	0.4	0.6	0.9	0.6	0.6
1	0.8	0.9	0.7	1.1	0.9	0.9	1.4	1.0	1.1	1.4	1.0	1.0
2	1.4	1.0	1.4	1.5	1.4	1.6	1.4	1.4	1.8	1.5	1.9	1.9

Table 5.1 Distance Readings

Actual Distance (m)	App Readings Average/Error at 1 second (m / %)	App Readings Average/Error at 5 second (m / %)	App Readings Average/Error at 10 second(m / %)	App Readings Average/Error at 20 second (m / %)	Average Error at distance (%)
0.10	0.03 / 70	0.03 / 70	0.10 / 0	0.10 / 0	35
0.50	0.10 / 80	0.27 / 46	0.57 / 14	0.70 / 40	45
1	0.80 / 10	0.97 / 3	1.17 / 17	1.13 / 13	10.8
2	1.27 / 36.5	1.40 / 30	1.53 / 22	1.77 / 11.5	25

Table 5.2 Readings Averages

For each given distance and time, we ran 3 tests. The above tests, while not being very conclusive, do show that for our beacons we would still get a large amount of uncertainty. However, we did see that the closer we got to the 20 seconds used as a maximum for the distance calculation, the lower the percentage error we got (excluding the readings from 1 meter). As seen by table 5.2, the inaccuracy of these beacons is quite noticeable and so could affect our applications accuracy and reliability.

5.1.2 ARMA (Autoregressive Moving Average)

This method of distance calculation is similar to the default running average algorithm but adds more weight onto more recent samples. This algorithm allows for less lag with distance estimates but is subject to varying performance as it is affected more heavily by radio conditions [23].

5.1.3 Distance Algorithm conclusion

After running both algorithms as the distance calculator, we found that the default algorithm seemed to work well enough. We encountered a slight issue with the time scale of the averaging period as 20 second seemed to be excessive since we only expect users to be in the beacons range for 5-10 seconds, we changed the averaging period to 10 seconds to reflect this. This change might add slight uncertainty to our distance readings, but it seems to be an appropriate change for our given solution.

5.2 Beacon Positioning/Detection

5.2.1 Beacon Detection Thresholding

Due to the issues mentioned in Section 5.1 we have to consider how the lack of accuracy for this technology is going to affect our threshold for being in range of a beacon. If we set the distance threshold as too small, we will encounter the issue of anomalous readings triggering our application to carry out an action. If we have the threshold as too large, we could encounter the issue where beacons must be placed too far apart for the application to work, and therefore will not be viable for small office spaces.

After a set of tests, we decided to go for a beacon detection threshold of 2 meters. This would account for the uncertainty of the beacon readings.

5.2.2 Beacon Positioning

The position of the beacons in a corridor is of massive importance. The application in its current state merely has 2 states for a beacon – interior, and exterior. With interior being the beacon inside the office and furthest from the exit, while the exterior one is the one closest to the door. The beacon's state is assigned depending on the Y coordinates of their icon in the beacon management fragment (see Section 4.3.3). This configuration does not reflect the reality of beacon positioning very accurately since it is only 2 dimensional and in real life we

have to place the beacons with regard to the inaccuracy we are expecting and therefore – with regard to the threshold which we discussed in Section 5.2.1.

Through some research we found that the minimum width of any hallway or landing area in a dwelling is of 900mm [24] therefore, in an ideal scenario where there was no degree of error in our beacon readings the position at which the beacon could be placed would be at hip height on any side of the entrance corridor and the threshold could be reduced to 900mm. This scenario does not take into account the uncertainty provided by the beacon distance readings which is why the threshold in 5.2.1 is so much greater than the above estimate.

However, there were other positions that were tested before we came to the above conclusion. The first possible position that came to mind for the beacons would be above doors in the entrance area. Doors in Europe range in height from 1981mm – 2040mm [25], and the average height in the UK is around 1680mm. Assuming we store our smartphones halfway up our body (be it in a bag or a pocket) we will need to have a threshold of 1200mm without considering error. This will lead to a threshold of a minimum of 2000mm when taking error into account. While this is a possibility it did not seem ideal due to reasons which are explained in the ethical issues section (Section 7.1.2).

With consideration of the expected error, and accounting for the threshold we stated before, we still think that the ideal position for a beacon to be placed in a landing area would be somewhere around hip height (~840mm off the ground). While this is the ideal position, we do have to consider that certain landing areas will be wider than the expected 900mm and so in these situations we would have to increase the beacons threshold. The increased threshold will come with other issues as mentioned in 5.2.1. A solution to this could be to place the beacon on a post in the entrance area at hip height and at a closer position to the actual door.

Chapter 6 Testing

6.1 Multiple smartphones

Due to the nature of the application we were hoping that there would be no issue with multiple smartphones interacting with the beacons at a given time. In theory there should be no issue with this interaction, as there should be a reliable connection with up to seven smartphones for android devices and ten for apple devices [26] for a given beacon at a given time.

However, as is shown by research done on the MetaSensor BLE solution [26] the best results occur when using less than three mobile applications at a given time.

To test this, we will run two subcategories of tests three times over each.

6.1.1 Correct Output

The objective of these tests is simply to see if the application gives the correct response for both users on the Slack channel. No other metric will be assessed at this stage.

1. We will test how the entrance detection works with a single smartphone, and then with two smartphones on the same person. The beacons will stay in the same position the whole time and the smartphone position on the person will also be the same.
2. We will test how the exit detection works with the same controls as the above test.

If the application takes more than two minutes to submit the status update to Slack, we will assume that the attempt has failed.

	Attempt 1 Entrance	Attempt 1 Exit	Attempt 2 Entrance	Attempt 2 Exit	Attempt 3 Entrance	Attempt 3 Exit
Single Device	Green	Green	Green	Red	Green	Red
Two Devices	Green	Purple	Green	Red	Green	Green

- Green All beacons Detected
- Red No Beacons Detected
- Purple Some Beacons Detected

Table 6.1 Device Detection Results

6.1.2 Time to Output

The objective of these tests will be to assess how fast we will be getting a response in the Slack channel from the beacons. We will start timing the moment we cross the second beacon and we will stop when the notification hits the Slack channel.

Just like in the previous test in Section 6.1.1 we will keep the controls on the smartphones and the beacon positions the same. Due to the inconsistencies we got in the initial tests of 6.1.1 we will only be testing for the time for the entrance message to be sent to the Slack channel.

	Attempt 1 (seconds)		Attempt 2 (seconds)		Attempt 3 (seconds)	
Single Device	3		2		5	
Two Devices	2	3	4	N/A	4	4

Table 6.2 Time to Output Results

6.2 Tests Conclusion

What we concluded from the tests is that the use of BLE beacons for office tracking will not lead to a reliable outcome, or at the very least that this technology becomes very unreliable as time goes by. As seen in the tests in 6.1.1 there were multiple instances where the entrance was detected correctly, while the exit was not detected correctly. This same issue came up in the second attempt in the tests carried out in 6.1.2.

Possible cause for this is the low quality of our beacons. Due to the low budget of the project we had to work with the cheapest beacons we could find. Cheaper beacons tend to have lower quality hardware within them and so might give of unreliable signals. A secondary possible cause for this failure could have been due to an unreliable beacon in use. The beacon in question had been in store by the university for 3-4 years, and so could have been running relatively low on battery affecting its performance. We believe this affected our output as the algorithm for exit detection is the same as the algorithm for entrance detection – and the only thing that changed in the tests for entrance/exit was which beacon was closer to the exit and which was closer to the entrance. However, we saw a larger amount of failures in 6.1.1 for the exit detection, this could lead us to attribute this larger rate of failure to a singular beacon out of the two used.

Overall, in a professional environment this issue could be limited by using beacons which were either of better quality or by changing the battery inside the beacon every set amount of time.

Chapter 7

Ethical, Social, Legal and Professional Issues

7.1 Ethical Issues

7.1.1 Tracking

When it comes to the tracking of individuals, we must consider their rights to privacy. Since our system can only tell whether a user is inside the range of our beacons, or whether they recently passed through the beacon's region, we can consider the positional tracking factor of our application to be relatively limited. The application does not give GPS coordinates of the user and it only sends out positional information in the form of a call to a Slack webhook to update the user's Slack status in a channel.

7.1.2 Disability and Special Considerations

Section 5.2.2 on beacon positioning brought to our attention that we might be encountering situations where our application is used by a person who is wheelchair bound, a person who is below the average height, or a person who suffers from dwarfism, we will refer to these users as special case users (SCU) from here on out. We have to be inclusive in the way in which BLE beacon detection systems work and therefore the above SCU have to be accounted for.

The SCU which are mentioned above share a distinct feature, and that is that their pockets (and therefore their smartphones) will be at a lower height than the average user and so the beacons placement will need to be done with consideration to this. Due to the above-mentioned reason we have chosen to recommend that beacons are not placed above door frames or on ceilings as this could lead to alienation of SCU (as is mentioned in Section 5.2.2.).

7.2 Security Issues

If the data of the current position of a user is leaked or accessed by non-authorised personnel, there could be a large potential risk to said user. Therefore, we must ensure that all data that is stored about a given user is done so in a safe manner, such as by using encrypted tables. Furthermore, data that is transferred over a non-secure network will need to be encrypted using an acceptable and secure encryption method.

7.2.1 BLE Pairing & General BLE Maleficence

BLE has two major categories of connection systems, “LE Legacy” and “LE Secure Connect” (LESC). LESG is the most secure of those two systems as it uses FIPS-compliant Elliptic-Curve Diffie-Hellman for key generation [27] during the connection process.

LE Legacy can be broken down by its 3 main key exchange methods:

- Just Works: A method which is considered “unauthenticated”. Is authenticate using a pin code made up of 6 0’s (usually). Is the most commonly used for devices with no interfaces.
- Passkey Entry: An authenticated method which requires users to put in a specific pin code.
- Out-Of-Bound (OOB): Uses a non-Bluetooth channel to transmit keys. Is not used much due to the additional overhead of the key exchange protocol.

The most used methods are “Just Works” and “Passkey Entry” and in our situation we can only use “Just Works” due to our lack of an interface on the actual beacon.

Due to how “Just Works” is implemented it is very easy for a bystander to eavesdrop onto the packages being exchanged between the applications host and the beacon. One method in which this may be achieved is explained by Harry O’Sullivan [28] by using a Bluetooth sniffer and carrying out a Wireshark analysis on the plaintext packets emitted by the beacon.

Furthermore, it has also been proven how a MITM attack can be carried out on any of the LE Legacy implementations through tricking the device to reauthenticate the application’s host allowing the attacker to intercept the exchanged keys and the seed for frequency hopping [29].

Apart from this, it is also possible to carry out DoS attacks where the objective is to drain the battery of the device hosting the application which communicates with the BLE beacon by sending a vast amount of malicious or incorrectly formatted requests. This specific attack is known as the “Battery-Draining-Denial-of-Service Attack” and according to research carried out by the University of Utah it was shown to reduce the battery of life of a smartphone by up to “as much as 97%” [30].

7.2.2 Slack Webhook Token

Due to our usage of Slack’s webhook feature we required a token which must be sent with every message to the API for channel identification and for authentication. The storage of this token is of relevance as a simple decompilation of the APK could lead to the exposure of the

token. As is shown by Enrique López-Mañas [31] we need to be able to have a system which clearly hides the token even when a decompilation is done on our codebase. Furthermore, we also need to ensure that the token cannot be accessed by other applications on the user's smartphone. In an ideal scenario we would want to request the token from an external source when the application is launched, and we would only want to store the token in memory so that the exfiltration of this data is harder.

7.2.3 Relevance to our System

Our application will use the "Just Works" method of key exchange and so is vulnerable to a large amount of security issues. We have mitigated the security issues at hand by only transferring the most essential amounts of data between the beacons and the smartphone application.

Furthermore, in our system the Slack token is hardcoded into the codebase since the application is merely meant to be a proof of concept for indirect communication between a timekeeping system and a BLE beacon through an android application.

7.3 Social Issues

7.3.1 Workers Rights

Our application does give the office management more tracking power over their employees. While this does exploit the hawthorn effect [31] to increase the workplaces productivity and so benefits the company, it does come with some interesting social issues regarding the workers. There have been multiple cases of offices implementing overly intrusive employee monitoring systems, such as the one seen in "The Daily Telegraph's" offices in 2016 [32] where productivity trackers were placed at worker's desks to monitor how long they worked for. Systems like these have led to discomfort for the employees and has led to protest from multiple different unions (such as the National Union of Journalists in the previous example) due to its supposed violation of workers' rights. In the above example the system implemented tracked the time which the worker spent at their desk and so could be considered as much more obtrusive than our own system, which merely tracks if a worker is in the office or not.

While our system and other office registering systems do track how long the worker is at the office for, it is rather unobtrusive as it only tracks the entrance and the exit of the worker into the office space. This sort of system has existed in offices for decades. How our system differs from traditional registration systems is that it allows for the automation of the process

rather than requiring the workers to manually sign in and sign out – therefore it does not have as many issues as the “Daily Telegraph” example.

7.3.2 Permissions

The library we are using for the communication with the beacons requires that we ask for certain permissions from the user. Firstly, we need to ask the user to activate their Bluetooth constantly, users might not want to do this as it might drain their battery and might also be a way for thieves to detect when a smartphone is left in a vehicle or in an empty office block.

Secondly, we need permissions to use their: Internet, Coarse Location, and Background Location. This gives the application access to the location given by the network provider and by the GPS system of the smartphone. If the application were to get hijacked in any way by a malicious application or malicious actor, we would risk giving them a lot of privileges to our smart phone and so the outcome could be catastrophic. This is a social issue as the tracking of individuals and the mass collection of data on individuals is something that has become very socially relevant lately and is something that could be done by malicious actors in the form of hackers for organized crime groups or for nation states.

7.4 Legal Issues

The data we store and process must comply with existing regulations and must be only collected with consent of the user [33] to be able to comply with GDPR. Since our application only collects data after the user has downloaded it there is implied consent, and so our application is not in violation of the user’s privacy. All users are free to withdraw their consent from the data collection system at any time by deleting the application from their smartphones. Since we are using local files for the storage of information about our system all of the files can be removed whenever the user wants [34]. The files that contain data about the user are the database mentioned in Section 4.2 and the SharedPreference files mentioned throughout Section 4. Both of these files get deleted upon the deinstallation of the application.

Chapter 8

Conclusion & Personal Reflection

8.1 Objectives Consideration

8.1.1 Main Objectives

The objective of the project was to develop a solution for office tracking which used mobile smartphone applications. Our solution implemented this by using BLE beacon technologies. Most of the objectives which were set initially (see Section 1.3) were met, however, some were met more successfully than others:

- Investigation of BLE technologies, including SDKs and tools for integration: We successfully managed to research existing solutions to our challenge in the context of wireless technologies (see Section 3.1). Within this research we studied the possibilities of using BLE for a new solution and how feasible this would be (see Section 3.1.3). We also managed to find suitable libraries for BLE communication and interaction (see Section 3.3.1). In our opinion this section was done thoroughly and we managed to identify some of the possible pitfalls we might encounter in the development process.
- Basic Application development: We managed to create a basic application in the earlier stages of development in a form of a prototype (see Section 3.3.4) which allowed us to test some of the limitations of our beacons. These tests allowed us to calculate appropriate thresholds and estimations for our ranging process (see Section 5) and therefore, allowed us to create a more robust and optimized application in further stages.
- Developing a robust application which can interact with Slack and a timekeeping service: While we feel like a relatively robust application was developed it still does suffer from slight performance issues. The application is sometimes slow to detect beacons and is slow to change fragments right after using the Slack webhook due to a badly optimised set of libraries used. In hindsight, the problem with the slow fragment changes could have been solved by assigning the task of managing the Slack communications to a separate worker thread. Additionally, while the application does communicate with Slack it does not interface with a timekeeping service – this is due to the company which we collaborated with for our project never providing the endpoints for the timekeeping service's API.

8.1.2 Additional Objectives

A set of additional objectives were laid out in our plan. These were to be implemented in sprint 3 after the main objectives were met. Due to sprint 3 being used to deal with the implementation of features which we failed to implement in previous sprints – we did not manage to complete all of these additional sub objectives.

One of the additional objectives met was the use of a secondary smartphone in the system. The addition of a secondary smartphone to the beacon's area did not seem to affect the system much. The results of the tests in Section 6 show how using a single smartphone and using multiple smartphones seem to have a rather similar effect.

Furthermore, we did not manage to introduce a system which would have used a third beacon to triangulate the user's exact position, this was due to multiple reasons. Firstly, due to the beacons large uncertainty while ranging - it would have been virtually impossible for us to triangulate the position of a moving target accurately. Secondly, we had economic deterrents due to the price of beacons which we had to personally acquire. Finally, we did not have enough time to implement this feature due to scope creep.

8.2 Overall Conclusion

Overall the application worked with a relative degree of success. As shown in Section 6 we mainly got the expected results, and in the situations where we did not get said expected results the fault could be attributed to the hardware (we suspect heavily). In regards to our original plan – most of the expected features were implemented to a satisfactory degree (as seen in Section 8.1) and the ones that were not implemented were also not implemented due to issues with the hardware which we used in the development of the application (see Section 8.1.2).

Overall, beacon systems can be used for many things in regard to location tracking within a specific area, but when the implementation of a solution requires multiple beacons and a certain degree of accuracy, BLE is not the best solution. Furthermore, one of the issues which we encountered while working on this solution was our lack of experience with BLE development. Even though a reasonable solution for this would be to introduce university modules around this topic, we feel that it would not be a valuable usage of student's time since this technology is not widespread enough; nor reliable enough for it to warrant mass education on the topic.

To conclude, through the development of this exploratory product we believe that we managed to show, to a reasonable degree, why the use of BLE is not one that should be considered for office tracking solutions.

8.3 Possible Further Work

8.3.1 Physical Lock Implementation

A possible further use case for this, and similar systems, could be to add a physical factor to it. In the current state of our solution there is no mention of how this system would ensure that only workers of the office using said system would be allowed in. Currently, it only serves the purpose of registering individuals once they get into the office. However, the entire concept of having a seamless entrance and registration is made redundant if the user also has to pull out a key card or input a passcode to open a front door/gate to their office.

Therefore, one possible future improvement or application of this system could be to have it interact with the entrance/exit doors in an office block. Once the application detects a set of BLE beacons a message with a unique worker ID could be sent to a server. The server would check a database of workers, and if the database contains the ID of the user that got detected - then the front door could be signalled to open. A system like this would not only improve how quickly a user can enter and leave an office at a given time, but it would also allow for the management of a company to remove the ability of former employees from accessing the office block once they have ended their employment at said company. A system like this would therefore reduce the risk of disgruntled ex-employees attempting to harm the company after their firing. Additionally, a contactless door system would avoid the risk of workers potentially spreading a virus/disease to each other by having to touch the same keypad multiple times over a given day.

8.3.2 Interaction with Other Mobile Technologies

Furthermore, the current state of the application only allows for it to be hosted on an android smartphone. This system could be implemented on other types of smartphones such as apple smartphones. This would increase the possible target audience of the system. Additionally, this application could be changed to be able to function from a smartwatch. In many situations workers could forget their smartphones at home or in the office, in a situation like this the system we described in this report becomes redundant. In the case where a worker uses a smartwatch it would be much less likely that they would leave said watch in their office or at

home since it is much more connected to them. This solution would help to make sure that the reliability of the application is higher than if just smartphones were used.

8.4 Self Appraisal and Personal Reflection

I started this project being relatively optimistic about my ability to learn new technologies. The new technologies I had to learn for this project included:

- BLE App Integration
- BLE & Traditional Bluetooth Basics/Stack
- Android Development
- Slack Integration

I had never done any of the above to any meaningful degree before and they had not been taught to me at university nor in any other course. With the lack of knowledge in these areas I encountered the issues of not knowing where to start or what tools to use for development. I quickly found this issue to be rather cumbersome since the libraries I wanted to use had conflicting dependencies and led to the compilation of the product failing (this was described in more depth in Section 4.5.4). I feel like I did not deal with these issues in an appropriate and efficient way, as I tried to modify the libraries and the dependencies of said libraries to try and make them work with each other (this is also something I had never done). These attempted changes to the libraries resulted in me spending too much time trying to solve an issue which could have been resolved by changing the library in use, which was also what I ended up doing after much stress. In future projects I will remember to consider multiple alternative libraries before settling on one and trying to make it work by force.

At the start of the development process I found myself having issues with the SDK provided by the company which made the beacons I ordered (Axaet). The SDK had documentation, but it was not written in great English and the documentation lacked technical aspects. The example code for their beacons was also badly commented and difficult to understand. To avoid plagiarism, rather than copying from the examples provided by Axaet, I chose to change the SDK from the one provided by Axaet to an open source one. In future work I will remember that opensource libraries have a lot of contribution from their user base and so have a large amount of examples and help available for them on multiple sites.

As seen in the implementation (Section 4) I did not adhere to my original plan very strictly. While I chose to use an agile methodology to try and account for this issue, I feel like I drifted from the schedule more than I expected to, this was due to three main reasons. A large part of this was due to other work throughout my second semester of university. Secondly, I believe that it was also caused due to the lack of experience I had with a lot of the

technologies I used. This lack of experience led to me having to do a relatively large amount of research in the development sprints, which is something I should have done more in depth during the first 10 weeks which were dedicated to research. The third reason links in with the second one, and it is the fact that I did not do enough in-depth research on the tools I would be using. Had I initially carried out research to a greater degree on specifically the tools I would use for the code of the project, I think I would have been able to adhere to my original plan with greater efficacy.

Furthermore, as mentioned in Sections 4.5.2 and 4.5.3, I suffered from an issue with scope creep. Some of the objectives I set out initially were written in a vague way which did not set a concrete end of said objective. This led to me spending a rather long amount of time on objectives that were not critical to the main solution which cause other objectives to be delayed and to be pushed onto later sprints. A solution to this would have been to define a specific and concrete end point for each objective I wanted to achieve in a given sprint.

The experience of developing a full android application was one that I found very valuable. I learned a lot about how users interact with an application and how phones themselves also interact with the apps. A large amount of the time spent in application development was based around creating an intuitive UI which would allow a user to quickly understand and use an app. The experience of working with android phone UIs was very valuable for me and is something that I feel like I learned a lot from.

Overall, the main points I learned from this project can be broken down into:

- Don't try to reinvent the wheel when it comes to using external SDKs or libraries
- Don't start a project unless you've done enough research into the tools and techniques you will be using
- Make sure you outline your objectives explicitly to avoid scope creep

Appendix A: External Materials

A 1 External Materials

The external materials in this project are divided up into 2 main sections. External material for graphical designs and external material for coding purposes.

Graphical:

- www.draw.io was used to make the designs for the application and the graphics to describe the database. This was used throughout section 4 of the report.

Code:

- jSlack was used to interact with the Slack client [22]
- Android Beacon Library along with its example code was used to develop the system that interacted with the beacons [17]

Appendix B: Installation and Expected Usage

B 1 Download and Installation

The download link for the APK file is:

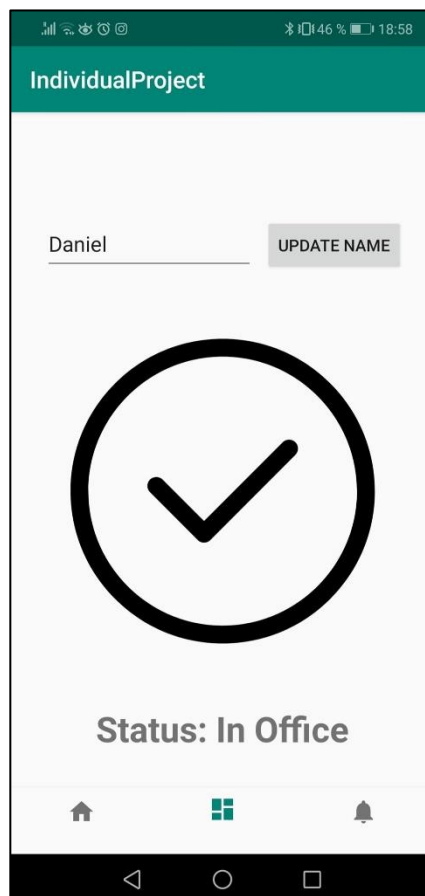
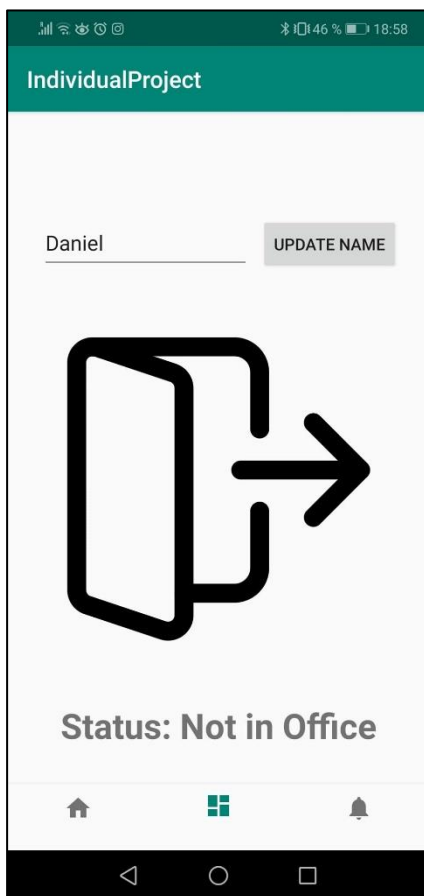
<https://drive.google.com/file/d/1tGo4yOPpUX-y94qR9i1NmZ1zPCf0xd7O/view?usp=sharing>

Once the file has been downloaded onto an android phone it will request for your Bluetooth and location services to be activated. This is necessary for the beacon library to function.

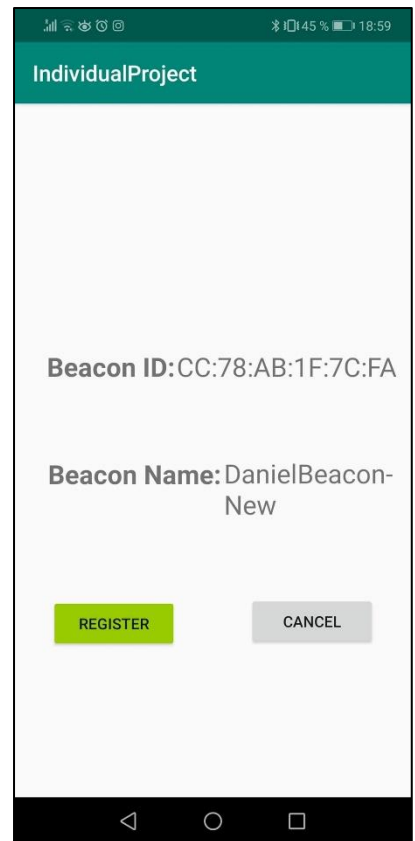
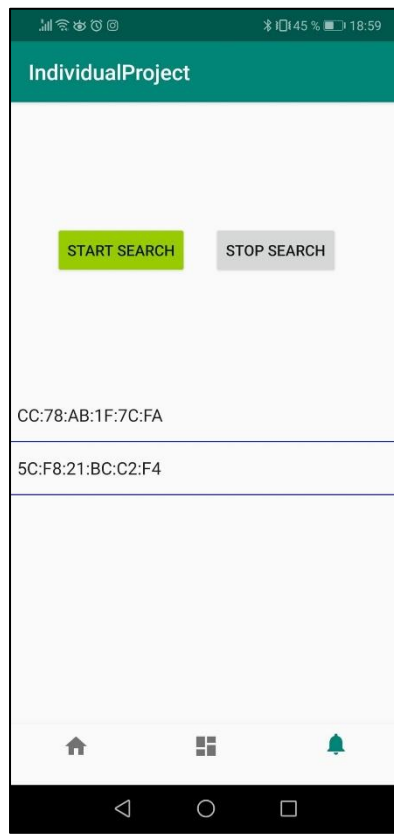
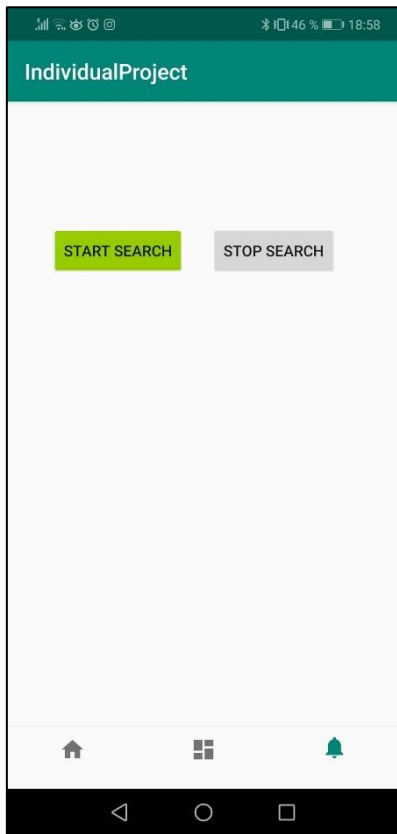
B 2 Example of Usage and Expected Screens

Two examples of the applications usage can be found in the same directory as the .apk file (https://drive.google.com/drive/folders/13JARdiCATD_jvCgA5h80lm_xpLn2wXyD?usp=sharing). One of the examples did not have any beacons present while the other example had beacons in the region.

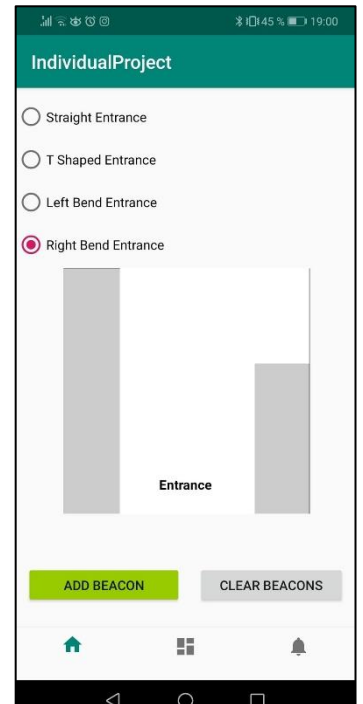
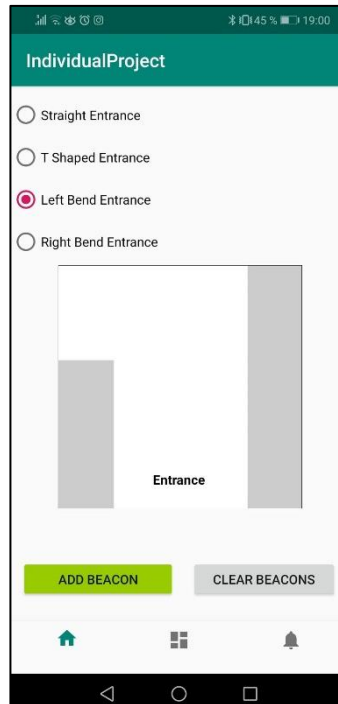
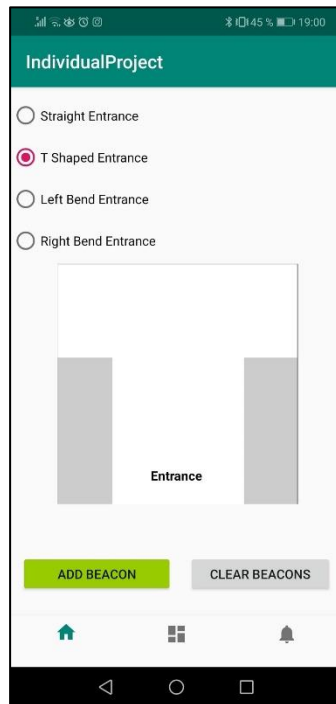
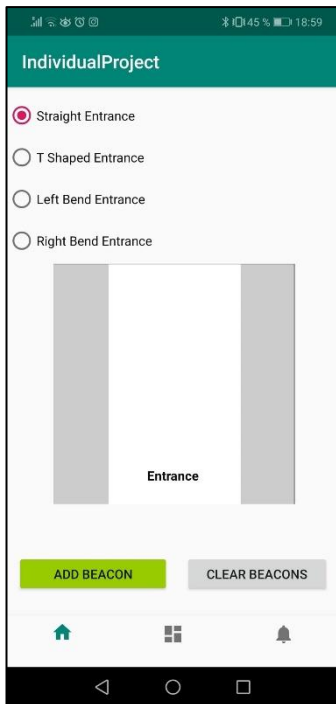
B 2.1 Status Screens



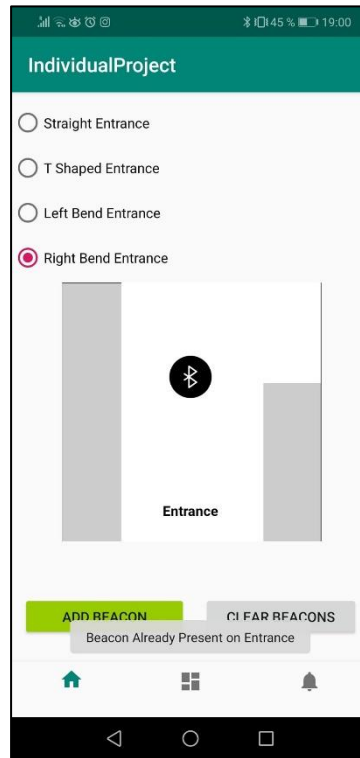
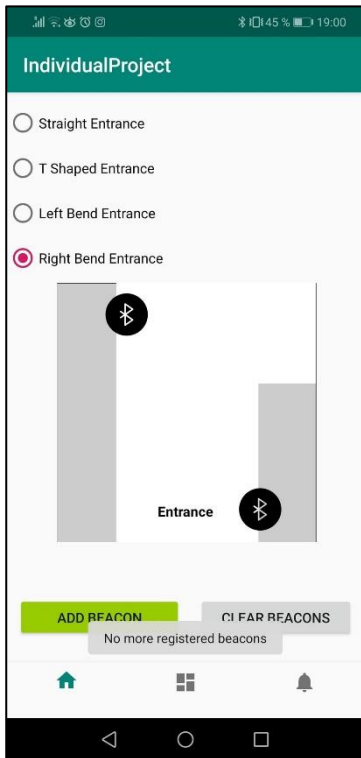
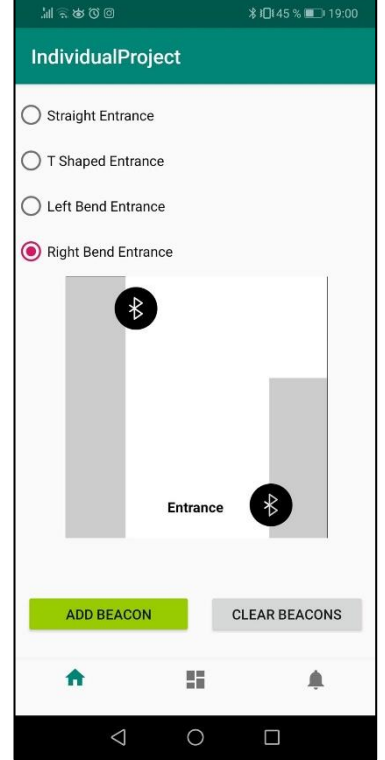
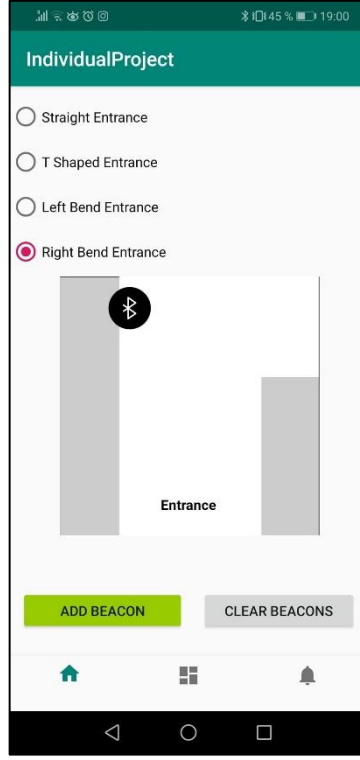
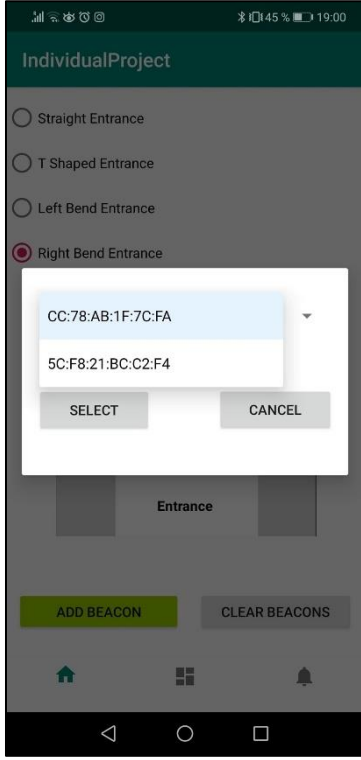
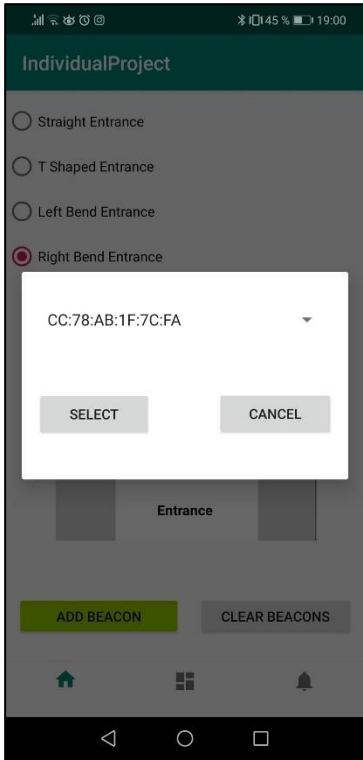
B 2.2 Search Screens



B 2.3 Map Screens



B 2.4 Beacon Management Screens



B 3 User Guide

B 3.1 Beacon Registration

By clicking on the right most menu button, you will be taken to the beacon registration screen. You will click the “Start Search” button to start the search for beacons. You will click the “Stop Search” button to stop the search for beacons.

Once beacons appear, you have the option to click on them. This will take you to the registration screen. If you choose to register them you will click “Register”, if not you will click “Cancel”. The beacon will only register if you have previously not registered the same beacon. If this beacon has been previously registered, you will receive a message stating “Beacon Already Registered”.

B 3.2 Slack Status Name Change

If at any stage you want to change the name which appears in the Slack notification once you enter/exit the office click the middle menu button. On this fragment you will see a text bar at the top, fill in the name you want to change to and click on “Update Name”. This will update your name.

B 3.3 Beacon Management Entrance Change

You have the option of setting the entrance map to one of 4 pre-sets. This can be done in the leftmost tab of the menu bar. By clicking one of the radio buttons on the top part of the screen you should see an instant change in the entrance map.

B 3.4 Beacon Position Management

In the leftmost tab of the menu you can change the position in which beacons are placed in the app’s configuration to replicate their position in real life. To add a beacon, click the “Add Beacon” button. This will open up a modal which will give you a drop down of the beacons you have registered. If you do not have any beacons registered, you will see a message reading that “No More Registered Beacons” and you will not be able to add more beacons.

Once a beacon is selected from the drop down in the modal you can click select and a beacon icon will appear at the centre of the pre-selected entrance map. If this beacon already

exists on the map you will receive a message stating that “Beacon Already Present on Entrance” and the new beacon icon will not appear. The beacon icons can be dragged and dropped to their desired position and can be cleared from the screen using the “Clear Beacons” button.

List of References

- [1] R. Ramakrishnan, L. Gaur, and G. Singh, "Feasibility and Efficacy of BLE Beacon IoT Devices in Inventory Management at the Shop Floor," vol. 6, no. 5, 2016.
- [2] A. Filippoupolitis, W. Oliff, G. Loukas, "Occupancy Detection for Building Emergency Management Using BLE Beacons," 2016.
- [3] "What Are Beacons and How Beacons Technology Works," 2017. [Online]. Available: <https://www.intellectsoft.net/blog/what-are-beacons-and-how-do-they-work/>. [Accessed 25 November 2019].
- [4] Boehm, Barry W., "Managing the Development of Large Software Systems," 1987.
- [5] "Agile Manifesto," 2001. [Online]. Available: <https://agilemanifesto.org/principles>. [Accessed 26 November 2019].
- [6] ProductPlan, "Feature Driven Development (FDD)," [Online]. Available: <https://www.productplan.com/glossary/feature-driven-development/>. [Accessed 6 April 2020].
- [7] K. Cho, W. Park, M. Hong, G. Park, W. Cho, J. Seo, K. Han, "Analysis of Latency Performance of Bluetooth Low Energy (BLE) Networks," 2015.
- [8] SIG, Bluetooth, *Bluetooth Specification Version 4.0*, Bluetooth SIG, 2010.
- [9] V. Bahl and V. Padmanabhan, "Enhancements to the RADAR User Location and Tracking System," 2000.
- [10] T. S. Rappaport, A. Rappaport, "Wireless Communications. Principles and Practice," 1995.
- [11] R. Entner. , "2014 US Mobile Phone sales fall by 15% and handset replacement cycle lengthens to historic high.," 10 February 2015. [Online]. Available: <http://reconanalytics.com/2015/02/2014-us-mobile-phone-sales-fall-by-15-and-handset-replacement-cycle-lengthens-to-historic-high/>. [Accessed 6 December 2019].
- [12] *GPS Positioning Standard (SPS) Performance Standard*, US Government, 2008.
- [13] F. van Diggelen, C. Abraham, "Indoor GPS Technology," Dallas, 2001.

- [14] B. Ray, "Bluetooth Vs. Bluetooth Low Energy: What's The Difference?," 2015. [Online]. Available: <https://www.link-labs.com/blog/bluetooth-vs-bluetooth-low-energy> . [Accessed 9 December 2019].
- [15] "Bluetooth Low Energy Channels," Microchip Technology Inc., 2019. [Online]. Available: <https://microchipdeveloper.com/wireless:ble-link-layer-channels> . [Accessed 9 December 2019].
- [16] "PC062," BeaconZone Ltd. , 2019. [Online]. Available: <https://www.beaconzone.co.uk/eddystone/PC062> . [Accessed 9 December 2019].
- [17] "Android Beacon Library," Android Beacon Library, 2014. [Online]. Available: <https://altbeacon.github.io/android-beacon-library/> . [Accessed 9 December 2019].
- [18] "Services Overview," Android Developers, 2020. [Online]. Available: <https://developer.android.com/guide/components/services> . [Accessed 5 March 2020].
- [19] Android Developers, "Fragments," Android Developers, [Online]. Available: <https://developer.android.com/guide/components/fragments>. [Accessed 31 March 2020].
- [20] "Use Android Jetpack to Accelerate Your App Development," Android Developers, 2018. [Online]. Available: <https://android-developers.googleblog.com/2018/05/use-android-jetpack-to-accelerate-your.html> . [Accessed 5 March 2020].
- [21] Android Developers, "Drag and drop," Developers, 27 December 2019. [Online]. Available: <https://developer.android.com/guide/topics/ui/drag-drop>. [Accessed 8 March 2020].
- [22] seratch, "jSlack Github," [Online]. Available: <https://github.com/seratch/jslack-maintenance-releases>. [Accessed 13 March 2020].
- [23] "Distance estimates," Android Beacon Library, 2014 - 2019. [Online]. Available: <https://altbeacon.github.io/android-beacon-library/distance-calculations.html>. [Accessed 16 March 2020].
- [24] "Distance estimates vs. time," Android Beacon Library, 2014-2019. [Online]. Available: https://altbeacon.github.io/android-beacon-library/distance_vs_time.html. [Accessed 2020 March 16].
- [25] "Internal doorways and hallways," Lifetime Homes, [Online]. Available: <http://www.lifetimehomes.org.uk/pages/6-internal-doorways-and-hallways.html>. [Accessed 17 March 2020].

- [26] "Standard Door Sizes," JBKind Doors, [Online]. Available: <https://www.jbkind.com/info-centre/standard-door-sizes>. [Accessed 17 March 2020].
- [27] MBIENTLAB INC, "About Wireless," MBIENTLAB INC, 2017. [Online]. Available: <https://mbientlab.com/tutorials/AboutWireless.html>. [Accessed 29 March 2020].
- [28] P. Sivakumaran and J. B. Alis, "A Low Energy Profile: Analysing Characteristic Security on BLE Peripherals," in *CODASPY'18*, Tempe, AZ, 2018.
- [29] H. O'Sullivan, "Security Vulnerabilities of Bluetooth Low Energy Technology (BLE)," Tufts University, Massachusetts, 2015.
- [30] K.Saravanan, L.Vijayanand and R.K.Negesh, "A Novel Bluetooth Man-In-The-Middle Attack Based On SSP using OOB Association model," Marthandam college of Engineering and Technology, Marthandam, 2010.
- [31] S. N. Premnath and S. K. Kasera, "Battery-Draining-Denial-of-Service Attack on Bluetooth Devices," University of Utah, Utah, 2008.
- [32] E. López-Mañas, "A follow-up on how to store tokens securely in Android," Google Developers Experts, 30 April 2017. [Online]. Available: <https://medium.com/google-developer-experts/a-follow-up-on-how-to-store-tokens-securely-in-android-e84ac5f15f17>. [Accessed 2 April 2020].
- [33] J. McCambridge et al, , "Systematic review of the Hawthorne effect: New concepts are needed to study research participation effects," *Journal of Clinical Epidemiology*, vol. 67, no. 3, pp. 267-277, 2014.
- [34] Jackson, Ben Quinn & Jasper, "Daily Telegraph to withdraw devices monitoring time at desk after criticism," The Guardian, 11 January 2016. [Online]. Available: <https://www.theguardian.com/media/2016/jan/11/daily-telegraph-to-withdraw-devices-monitoring-time-at-desk-after-criticism>. [Accessed 13 March 2020].
- [35] . *REGULATION (EU) 2016/679 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC*, The European Parliament and The Council of the European Union, 2016.
- [36] "Access app-specific files," Android Developers, [Online]. Available: <https://developer.android.com/training/data-storage/app-specific>. [Accessed 3 March 2020].