Tracking in Building Using Bluetooth Beacons

COMP8047 – Major Project

Kevin Lo – A00952922
4-17-2023

Table of Contents

1. Introduction

1.1. Student Background

I am a student in the Bachelor of Technology in Network Security Applications Development program at BCIT. Since taking the courses at BCIT, I have learned the importance of network security and techniques to prevent hackers from reading information. This project will use Bluetooth and TCP communications to facilitate communication between devices.

1.2. Project Description

The aim of this project is to create a system that could track the approximate locations of people by using a tracker, beacons, and a server to communicate all the information to. This project will have a tracking application which reads Bluetooth RSSI dBm information received from Raspberry Pi beacons listening for Bluetooth devices in the area. The information from the beacons is sent to the server which calculates the approximate location of the tracker and displays it on a map. By using this, the approximate location of someone can be found quickly.

1.2.1. Essential Problems

The essential problem that this project will be solving is to allow the server operator to know where and when people are in a location. If the device has the tracker installed and enabled on their phone, the server operator will be able to know where and when someone was in a location with Raspberry Pi beacons scanning for the trackers.

Using a minimum of three beacons, the approximate location of a tracker can be calculated. This is possible by using RSSI dBm signal strength and converting it to distance. Using this distance, a radius can be established around each beacon and the location of the device can be found by finding the point where the area of the three beacons intersects.

This method of calculation is called trilateration which uses distance instead of angles which is triangulation.

1.2.2. Goals and Objectives

The goal of this project is to create a program that can approximate the location of someone to allow the server operator to easily know where someone is. The implementation used to calculate location is the “Three-border positioning” trilateration method which uses the coordinates of three reference nodes and distances. By solving the system of three equations, the X and Y coordinates of the device. The objective of this project is to test and experiment with the results of the location tracking feature and to see if its accurate or not.

2. Body

2.1. Background

There is no company to be my sponsor. I chose this topic because it is interesting and may have potential for future use for companies interested in surveillance. The main idea came from how people are always tracked these days such as methods like Apple AirTag and similar. I wanted to create something similar which became a project using Bluetooth as the transmission method to get devices in the area.

The nature of this project is to learn to develop a program that can track a device and display the data in a simple to understand way to the user. This process will scan the nearby area for Bluetooth devices, send the RSSI dBm and device information to the SQL database server, process the RSSI dBm data to produce a X and Y coordinate and record the result onto the database for future use. The user can use a program to view and interact with the data on the SQL database server to add or remove beacons, devices, maps, and alert rules or just view the current tracking results or the past logs of where devices have been detected.

2.2. Project Statement

Tracking users has become the utmost importance in the modern-day era with government surveillance and companies trying to scrape data to sell to the highest bidder. This project will demonstrate the effectiveness of using Bluetooth beacons as a medium to scan for devices used in a tracking program. The program will use trilateration as the method to determine location.

2.3. Possible Alternative Solutions

There are many different alternate solutions that could be used for this project. This can range from switching out the medium used to scan for devices such as using Bluetooth Low Energy, WIFI or any other form of software defined radio. The method used to calculate location can be also changed such as using triangulation instead of trilateration.

2.4. Chosen Solution

The solution I chose is to use regular Bluetooth as the scanning medium and trilateration for the method to calculate location. I used regular Bluetooth because I was having many issues with getting Bluetooth Low Energy for multiple weeks when I already had a basic version of regular Bluetooth beacon working during that time. I used trilateration as it was more straightforward for me to understand as it only uses distance unlike triangulation which utilizes angles which I do not understand as well and couldn’t find as much research papers. Initially the many papers I read were stating method as triangulation. There was some confusion between having triangulation in the title however, the method used to calculate location was trilateration which uses distances instead of angles.

2.5. Details of Design and Development

2.5.1 System/Software Architecture Diagram

The following Diagram shows the system architecture of this project and shows overall components and their relationship between each other.

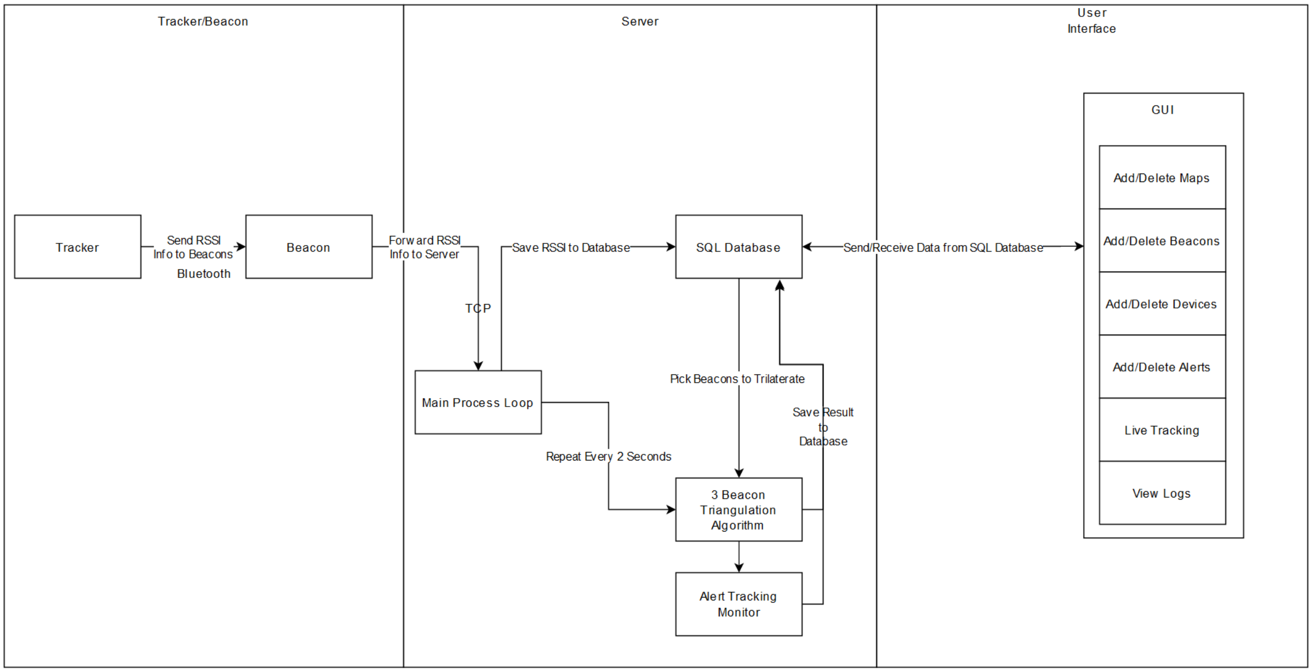


Figure 1: Diagram representing the Bluetooth Beacon Tracking Project.

The following diagram represents the communication network between each device in the tracking network. The tracking device represents the phone installed with the tracking application. Beacons are Raspberry PIs running an application to detect Bluetooth broadcasting signals and relay the RSSI dBm and Bluetooth MAC information to the server. The server saves RSSI dBm information to later process all the information received by the beacons into a X and Y coordinate for every tracked device registered to the server. This allows the GUI program to easily send information to add new entries to the database such as devices, beacons, maps and, alerts or do the opposite such as viewing the information. An easy-to-understand live map will be available for the user to view by constantly retrieving the most recent calculated device location.

The tracker enables Bluetooth broadcasting which allows the beacons to detect and measure RSSI dBm power. The beacon then forwards the tracker information with RSSI dBm and the beacon information to the server SQL database where it can be stored together with tracker, beacon, and RSSI dBm as an entry in the database. For each device to be tracked in the database, the three most recent and highest RSSI dBm beacons will be selected to trilateration the location of the tracker.

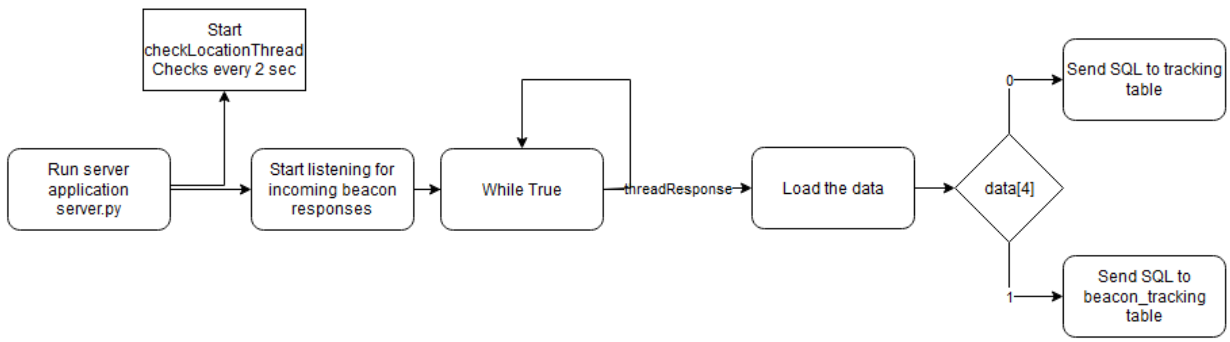


Figure 2 Server.py Diagram

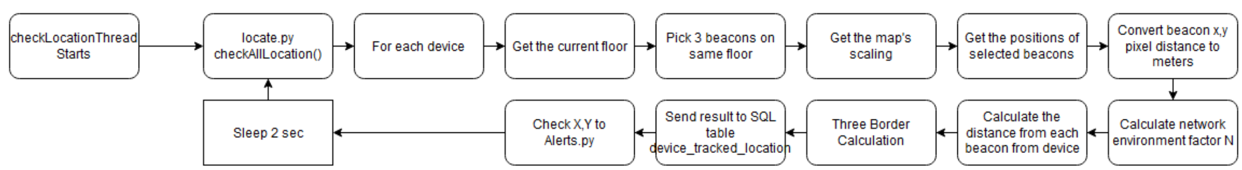


Figure 3 locate.py Diagram

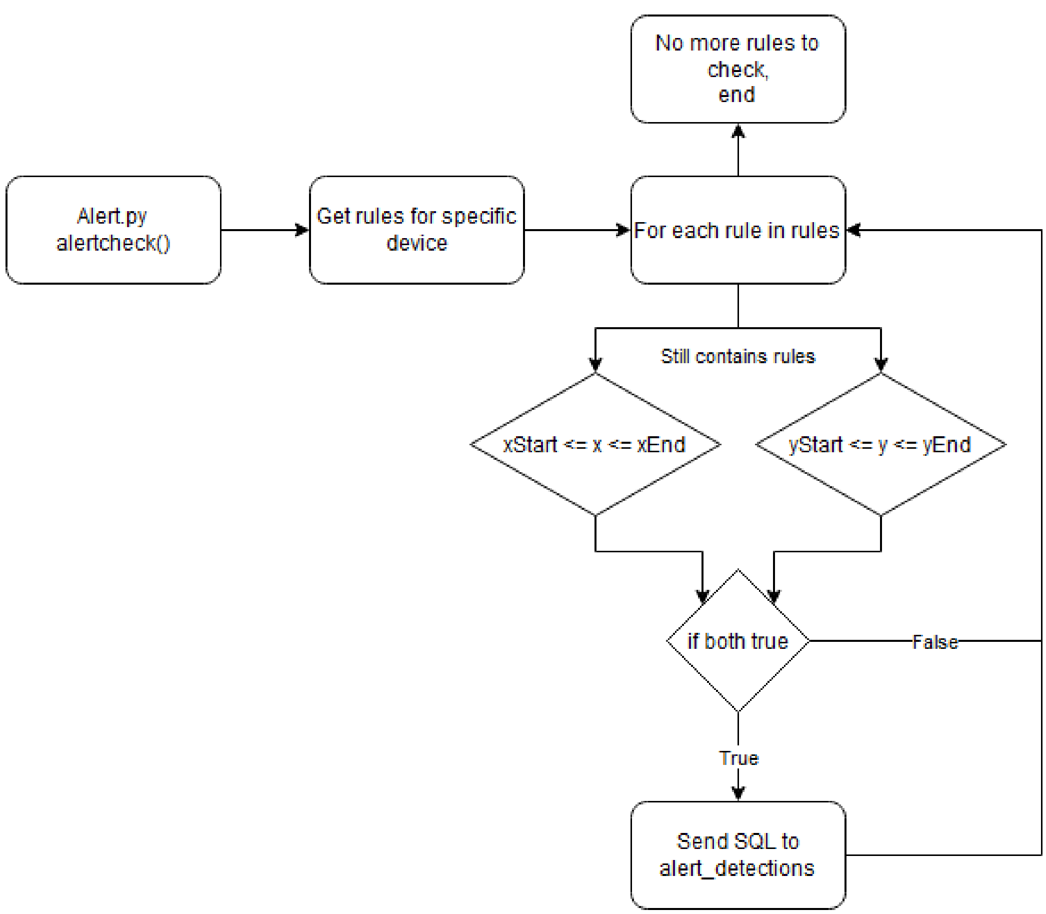


Figure 4 alerts.py diagram

The Python server program being developed will be broken down into multiple Python files to simplify the implementation of the code. The server starts server.py which runs the necessary code to listen for beacon response packets for the server. The Python file server.py calls the relevant functions from locate.py and alerts.py to find the location of the tracked devices and checks if it has broken any rules and records the data into the SQL database.

2.5.2 Pseudocode

The following pseudocode represents the project in a simplified pseudocode form. The program is broken down into multiple parts. The Tracker app is written in Android Studio. The Beacon, Location, Alert and Server are written in Python. The GUI is written in C# Visual Studio.

```
Tracker App
    Enable Bluetooth broadcasting

Beacon
    Start broadcasting thread
    Scan for Bluetooth broadcasts
    For each Bluetooth device detected
        if not a beacon device
            send information to SQL database in "tracking" database
        if a beacon device
            send information to SQL database in "beacon_tracking" database

Location Module
    for each device_mac + reference RSSI pair
        Get most recent floor the device_mac was on
        Get the 3 most recent beacons on the same floor
        Get the map scaling pixels to meters
        Calculate the environmental factor N
        Calculate the distance d for each beacon to device in meters
        Calculate the X,Y with Three Border Calculation
        Save x,y,device_mac,date to SQL database in "device_tracked_location"
        AlertModule(x,y,device_mac,floor)

Alert Module
    rules = using device_mac and floor get list of rules
    for each rule in rules
        if(xRuleStart <= x =< xRuleEnd)
            xInArea = True
        if(yRuleStart <= y =< yRuleEnd)
            yInArea = True

        if(xInArea && yInArea)
            save x,y,device_mac,date to SQL database in "alert_detections"

Server
    Start Location Module thread

    while true
        start listening thread to get beacon/device RSSI results
        if not a beacon device
            save to SQL database in "tracking" database
        if a beacon device
            save to SQL database in "beacon_tracking" database

GUI
    Ask for SQL server login information
    Display the main menu with the multiple menu options
    Allow the user to add/delete maps
    Allow the user to add/delete devices
    Allow the user to add/delete beacons
    Allow the user to add/delete rules
    Allow the user to view the logs
    Allow the user to view live tracking
```

Figure 5 Pseudocode representing the program.

2.5.3 SQL Database Layout

The following image is the layout of the database server. The tables ***map_image, beacon_location, devices,*** and ***alert_rules*** are the settings of the program. The tables ***tracking, beacon_tracking, device_tracked_location,*** *and* ***alert_detections*** are used for storing recorded information. The table *tracking* is used to store the raw capture of beacon_mac, device_mac, RSSI dBm and date from a beacon used for trilateration. The table ***beacon_tracking*** is used to store the raw capture of beacon_mac1, beacon_mac2, RSSI dBm, and date used for finding the environmental factor for trilateration. The tables ***device_tracked_location*** and ***alert_detections*** are used to store the results from trilateration and checking if the device location breaks any of the rules.

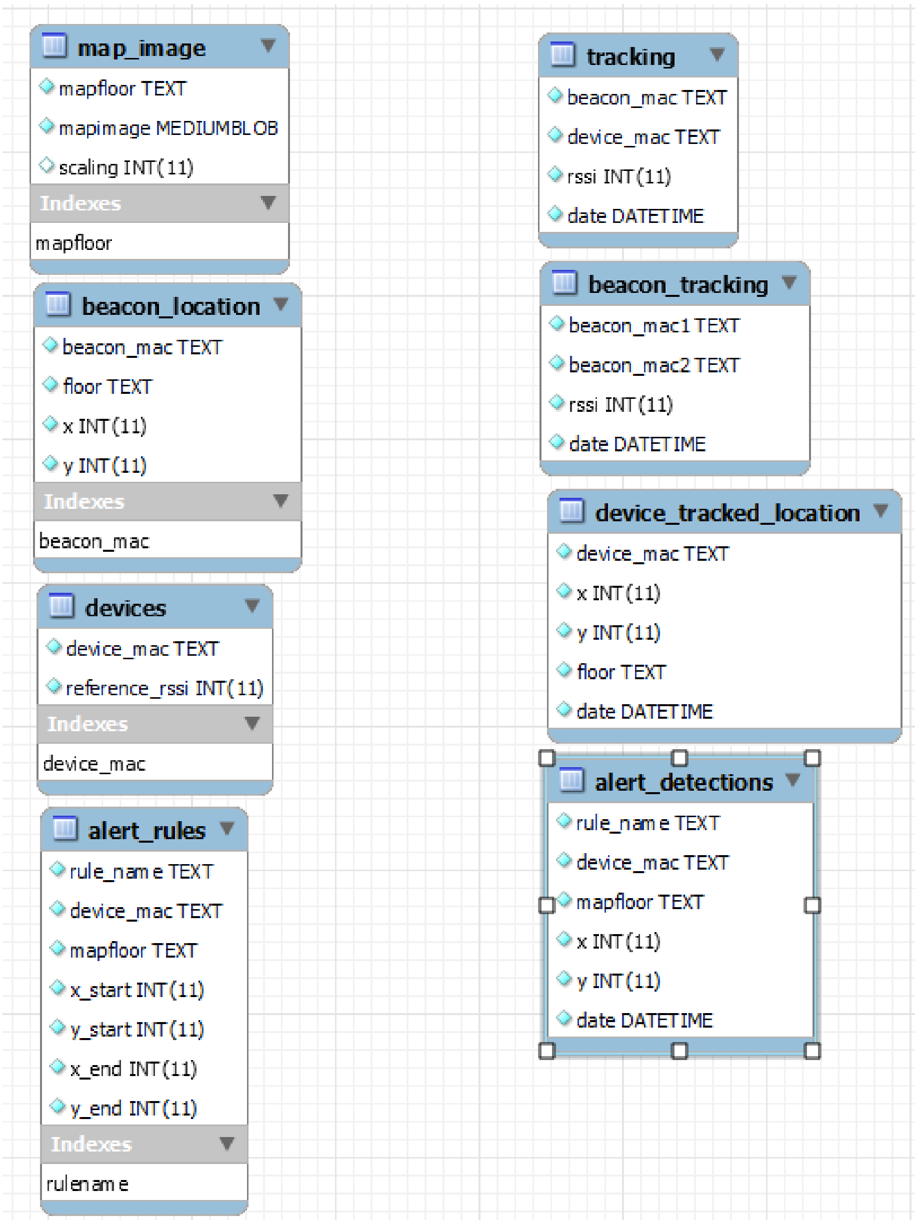


Figure 6 Layout of the SQL server.

2.6. Testing Details and Results

The testing for this project mainly consists of manual and acceptance testing. For example, placing the tracker in between two or more beacons and how the server responds. Testing the server features of adding and displaying information about devices and trackers in the network to ensure everything is working properly.

To run the program various dependencies, need to be installed onto the hardware.

Server (Raspberry Pi) –
python -m pip install mysql-connector-python
python -m pip install numpy
Install MySQL **https://www.youtube.com/watch?v=OBbTkBy2blI**
Allow remote access to MySQL

Beacons (Raspberry Pi) –
python -m pip install mysql-connector-python
sudo apt install python3-bluez
enable Bluetooth broadcasting
nano /lib/systemd/system/bluetooth.service
edit ***ExecStart=/usr/libexec/bluetooth/bluetoothd***
to
edit ***ExecStart=/usr/libexec/bluetooth/bluetoothd --compat --noplugin=sap***
systemctl daemon-reload
systemctl restart bluetooth

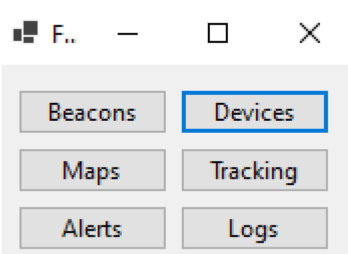
Android (Phone Tracker) –
Download and install the Bluetooth tracking APK to enable Bluetooth broadcasting

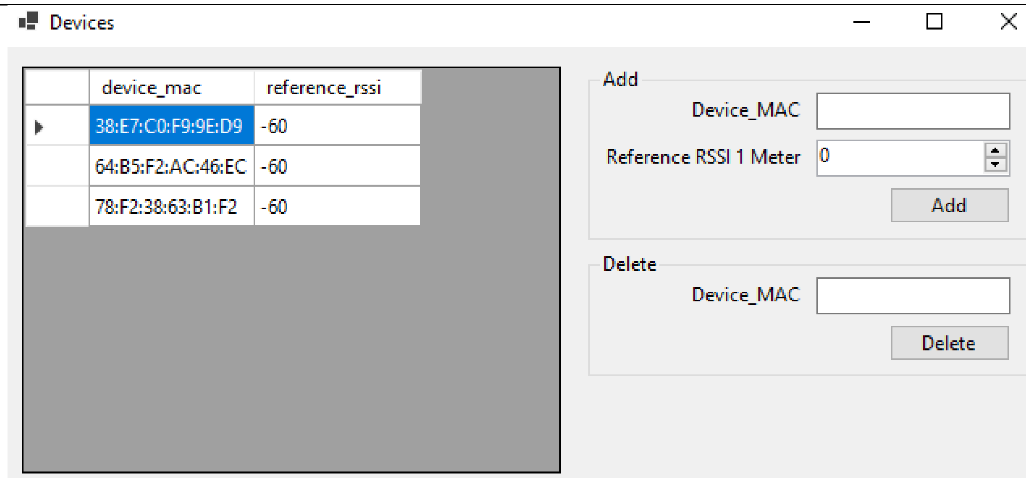
Windows (GUI) –
Download the Windows Forms program to easily display database info

No.	Draft Requirement	Test	Passing Criteria
1	Adding a new device to the server	Add a device to the tracking network	The device is successfully added
2	Deleting a device on the server	Delete a device on the tracking network	The device is successfully deleted
3	Adding a new map to the server	Add a map of a building	The map is successfully added
4	Deleting a map on the server	Delete a map on the tracking network	The map is successfully deleted
5	Adding a new beacon to the server	Add a beacon to the server	The beacon is successfully added
6	Deleting a beacon on the server	Delete a beacon on the tracking network	The beacon is successfully deleted
7	Viewing the alerts on the server	Opening the alerts	Alerts are able to be seen

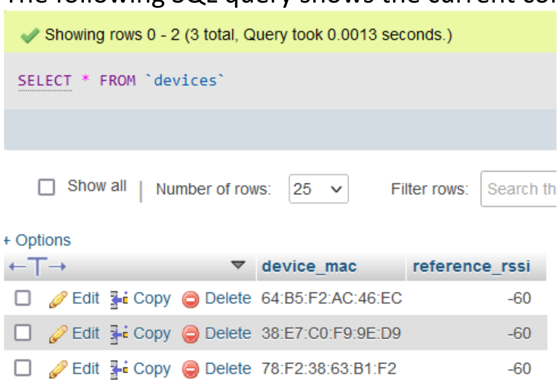
8	Adding a new rule to the server	Add a new rule to the server	The rule is successfully added
9	Deleting a rule on the server	Delete a rule on the tracking network	The rule is successfully deleted
10	Area rule broken will log to alert_detections	Make a rule where tracker will violate	The violation is logged to the database
11	Able to view the tracking logs	Open the log menu for a specific tracker	The history of the tracker is shown with the timeline showing with the map
12	Able to view the live tracking	Open the live tracking menu	The live tracking is showing positions
13	Tracking is room accurate	Viewing the live tracking to visually verify the tracker dot is in the correct map and area	The tracking is in the correct map and area.
14	Tracking is pinpoint accurate	Viewing the live tracking to visually verify the tracker dot is in the correct map and is less than 0.2 meters wrong	The tracking is in the correct map and is less than 0.2 meters wrong consistently

2.6.1 Manual Test Results

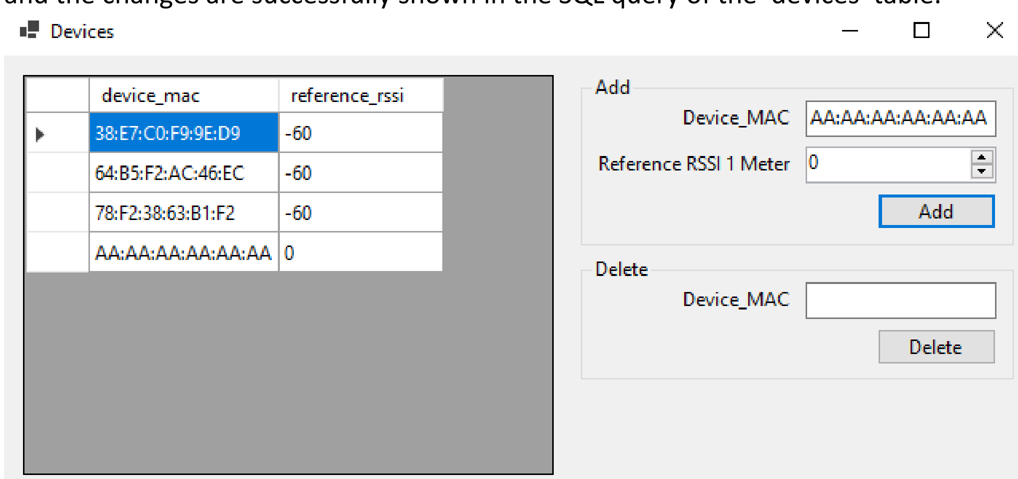
No.	Results	Verdict
1	**Adding A Device** This ability to add a device will be shown in the GUI application. The following two screenshots show the MainProgram form opening the Devices form. A device can be added by entering information into the respective fields and pressing add. It will fail if there is already a duplicate device_mac in the table.  	Pass

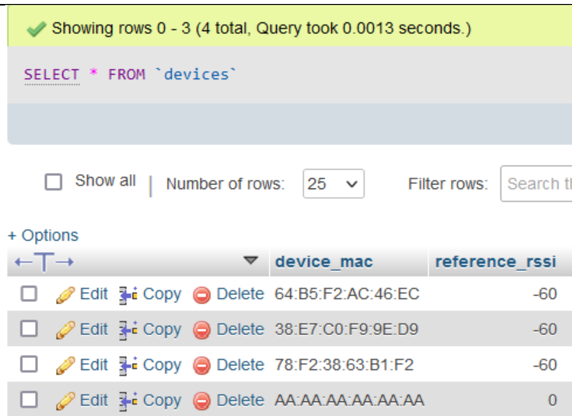


The following SQL query shows the current contents of the 'devices' table.



The device_mac AA:AA:AA:AA:AA:AA is successfully added through the GUI menu and the changes are successfully shown in the SQL query of the 'devices' table.

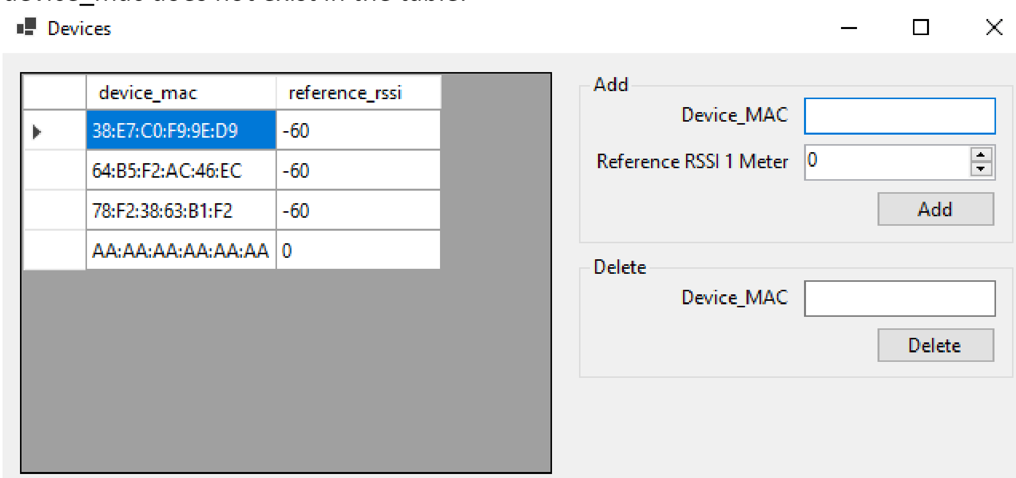


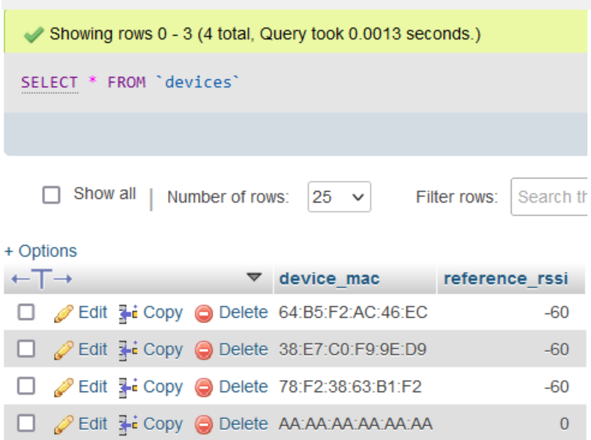


2

Removing A Device

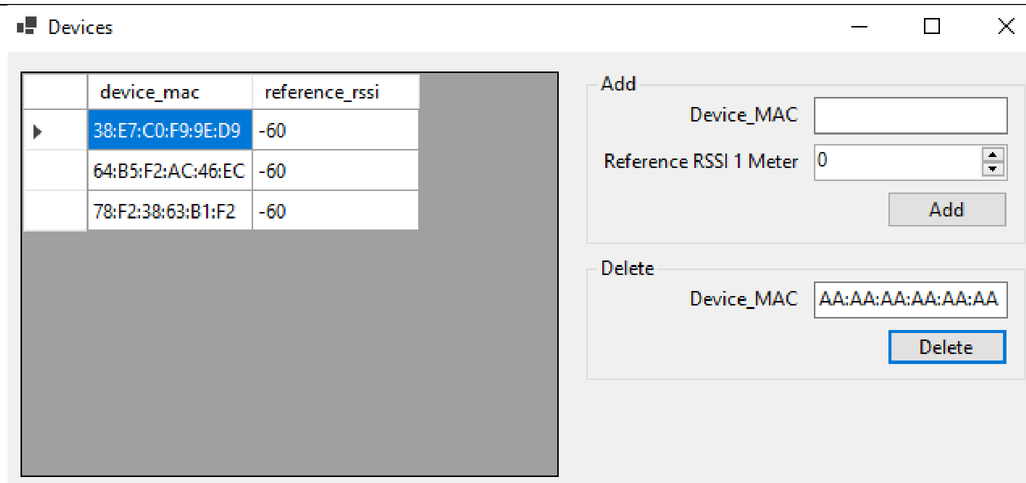
The following two screenshots show the shows the current state of the 'devices' table from both the GUI program and the SQL database. A device can be deleted by entering information into the respective fields and pressing delete. It will fail if the device_mac does not exist in the table.

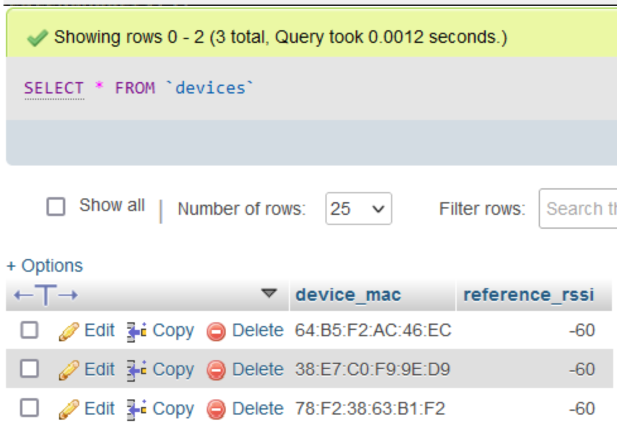
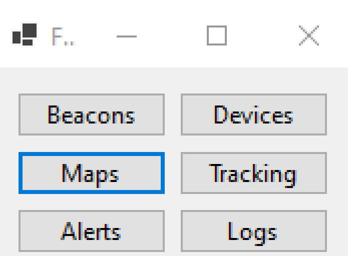


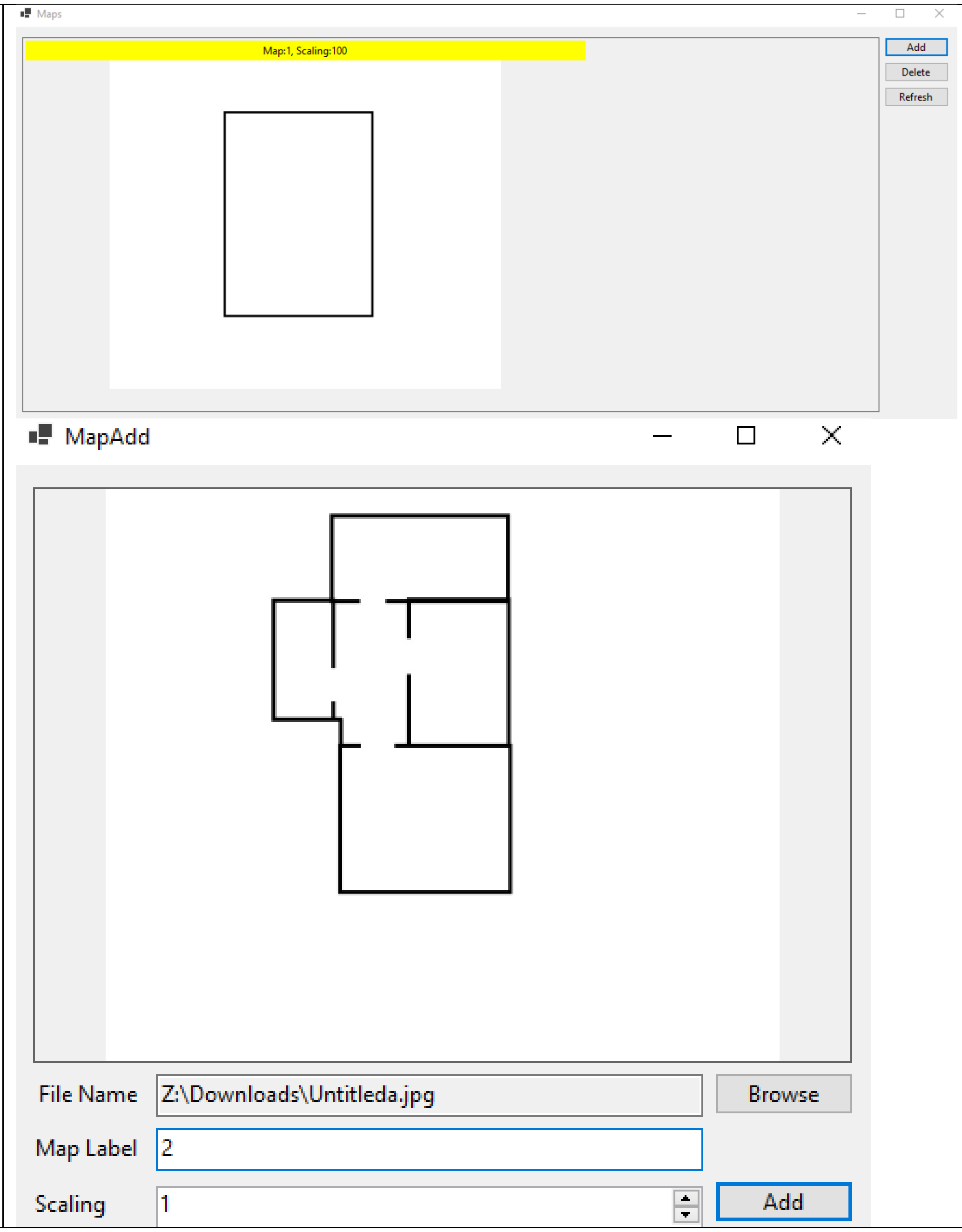


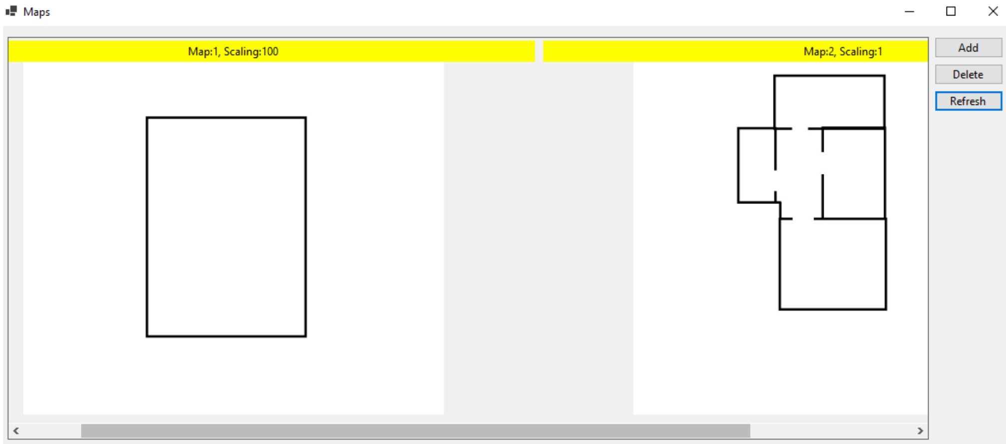
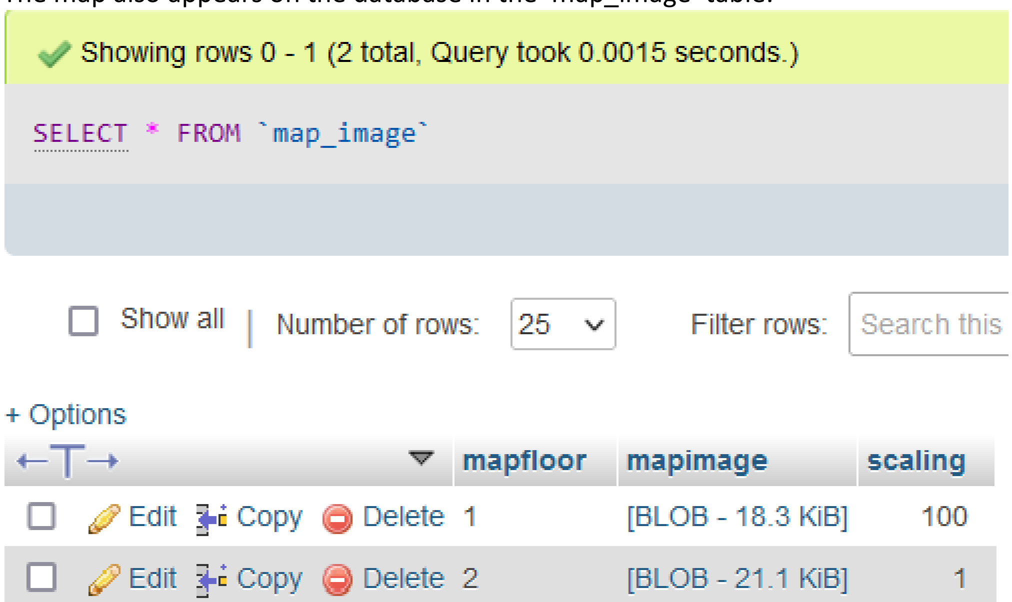
The device_mac AA:AA:AA:AA:AA:AA is successfully deleted through the GUI menu and the changes are successfully shown in the SQL query of the 'devices' table.

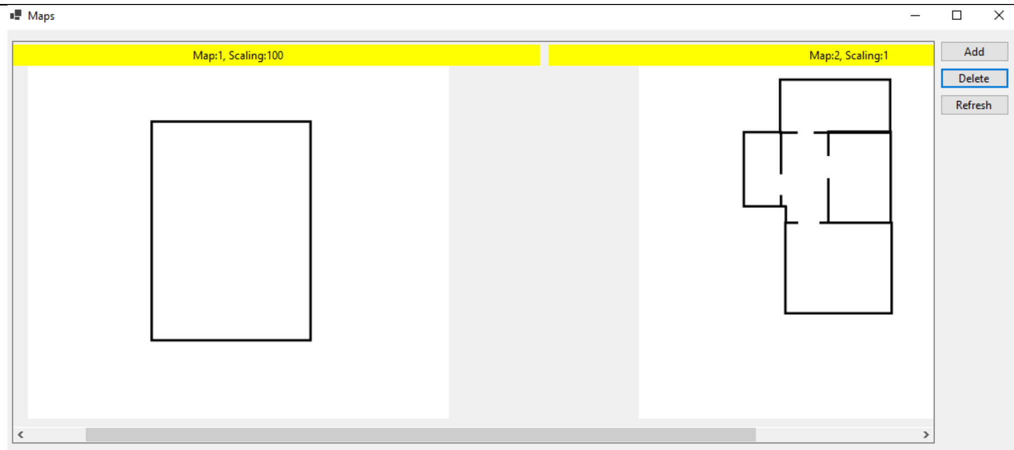
Pass

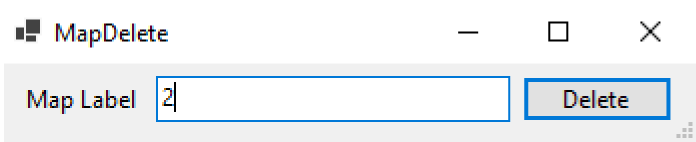
<table>
<tr>
<td></td>
<td>

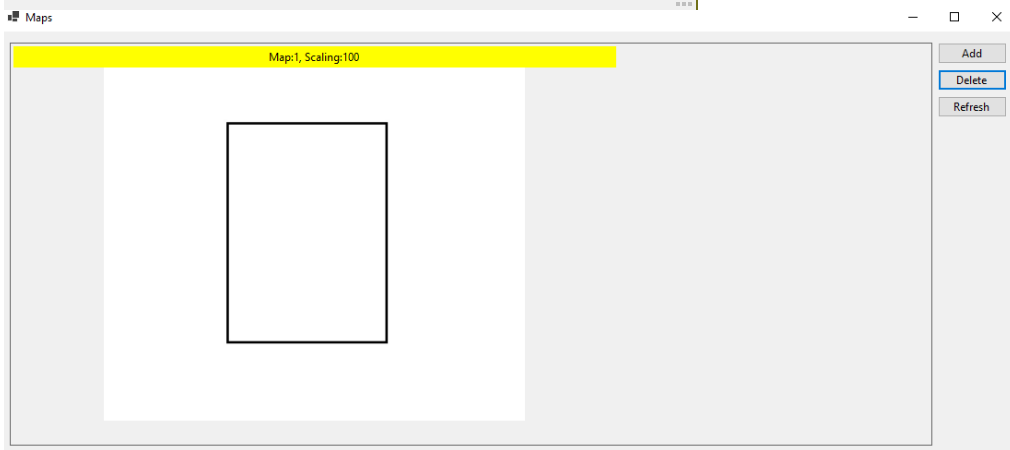

</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Adding a new map to the server
The ability to add a map will be shown in the GUI application. The following two screenshots show the MainProgram form opening the Map form. A map can be added by clicking add and opening the MapAdd form and entering the relevant information.

</td>
<td>Pass</td>
</tr>
</table>

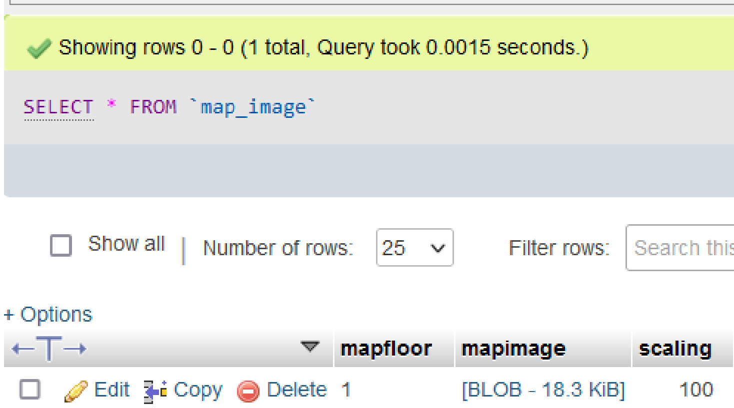

Maps
Map:1, Scaling:100
Add
Delete
Refresh
MapAdd
File Name
Z:\Downloads\Untitleda.jpg
Browse
Map Label
2
Scaling
1
Add

<table>
<tr><td></td><td>The map will then appear in the Map form after pressing “Refresh”.

The map also appears on the database in the ‘map_image’ table.
</td><td></td></tr>
<tr><td>4</td><td>Deleting a map on the server
The ability to add a map will be shown in the GUI application. A map can be deleted by clicking delete and opening the MapDelete form and entering the relevant information and clicking delete.</td><td>Pass</td></tr>
</table>

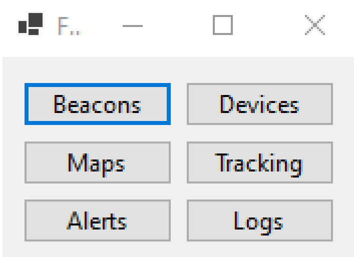
<table>
<tr>
<td></td>
<td>


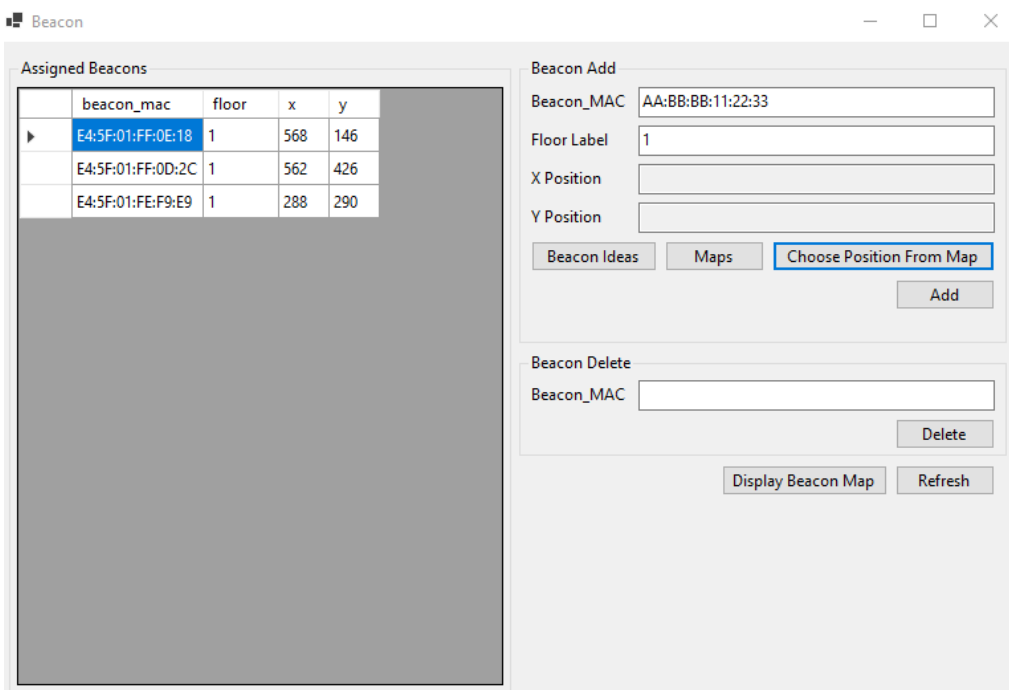


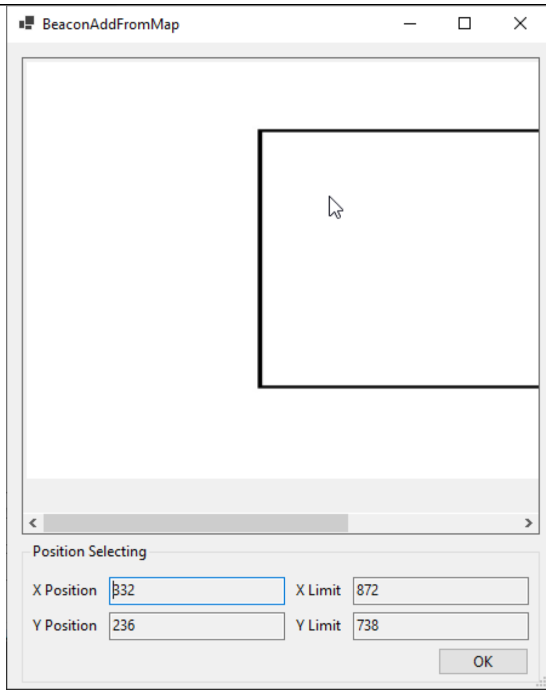


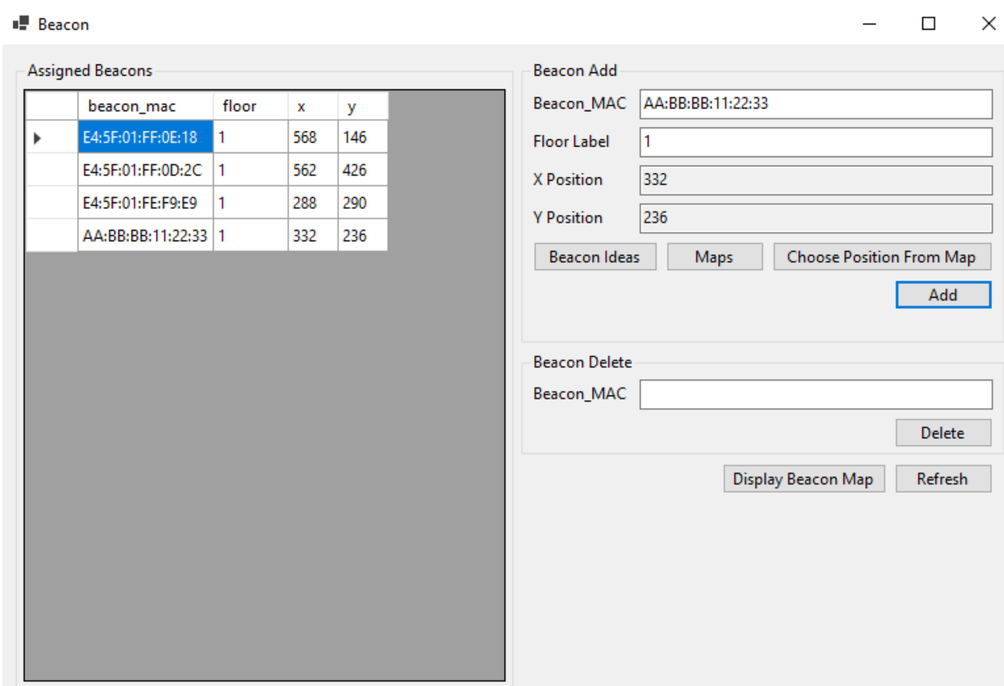


</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Adding a new beacon to the server
The ability to add a beacon will be shown in the GUI application. The following two screenshots show the MainProgram form opening the Beacon form. A beacon can be added by filling in the Beacon_MAC and Floor Label fields and clicking the “Choose Position From Map” to select the X, Y positions where the beacon is located. Finally clicking “Add” will then add the beacon to the database.</td>
<td>Pass</td>
</tr>
</table>

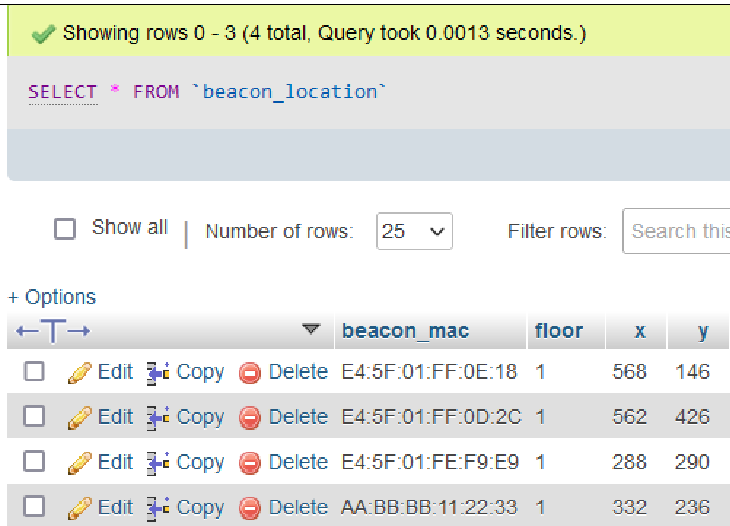

F..
Beacons
Devices
Maps
Tracking
Alerts
Logs


Beacon
Assigned Beacons
beacon_mac
floor
x
y
E4:5F:01:FF:0E:18 1 568 146
E4:5F:01:FF:0D:2C 1 562 426
E4:5F:01:FE:F9:E9 1 288 290
Beacon Add
Beacon_MAC
AA:BB:BB:11:22:33
Floor Label
1
X Position
Y Position
Beacon Ideas
Maps
Choose Position From Map
Add
Beacon Delete
Beacon_MAC
Delete
Display Beacon Map
Refresh





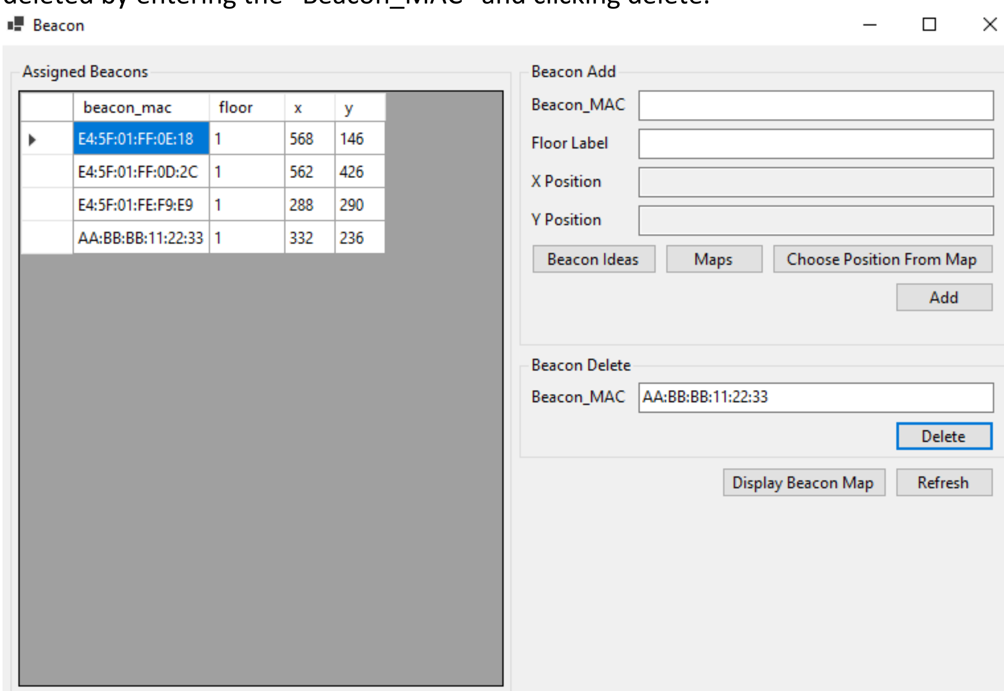
The database result shows that the new beacon has been successfully added to the server.



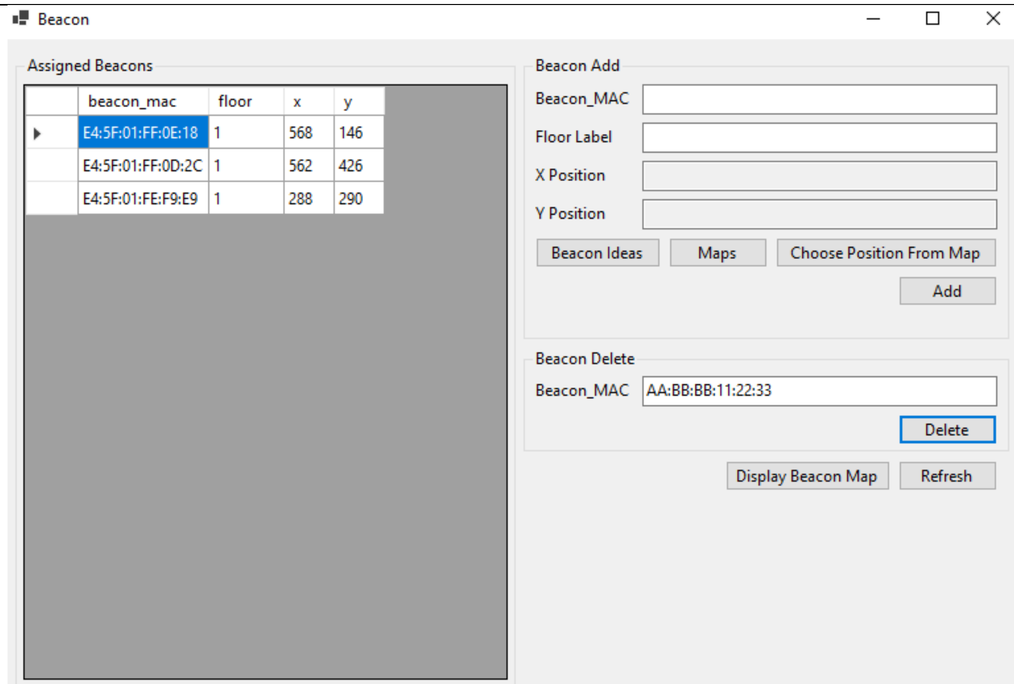
6

Deleting a beacon on the server

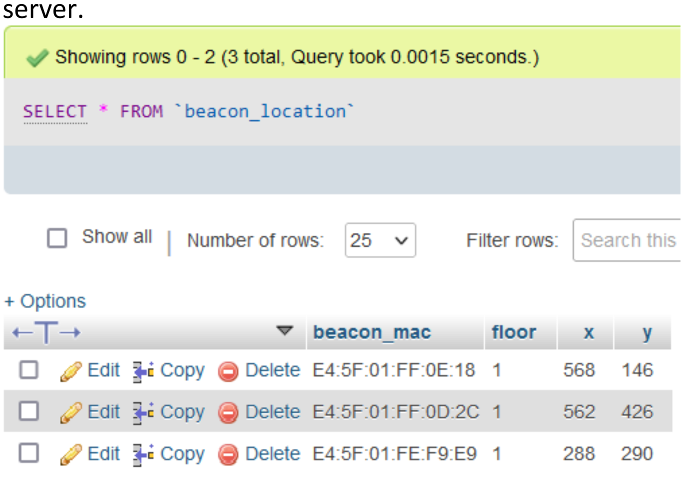
The ability to delete a beacon will be shown in the GUI application. A beacon can be deleted by entering the "Beacon_MAC" and clicking delete.



Pass

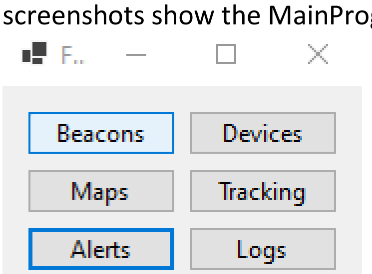


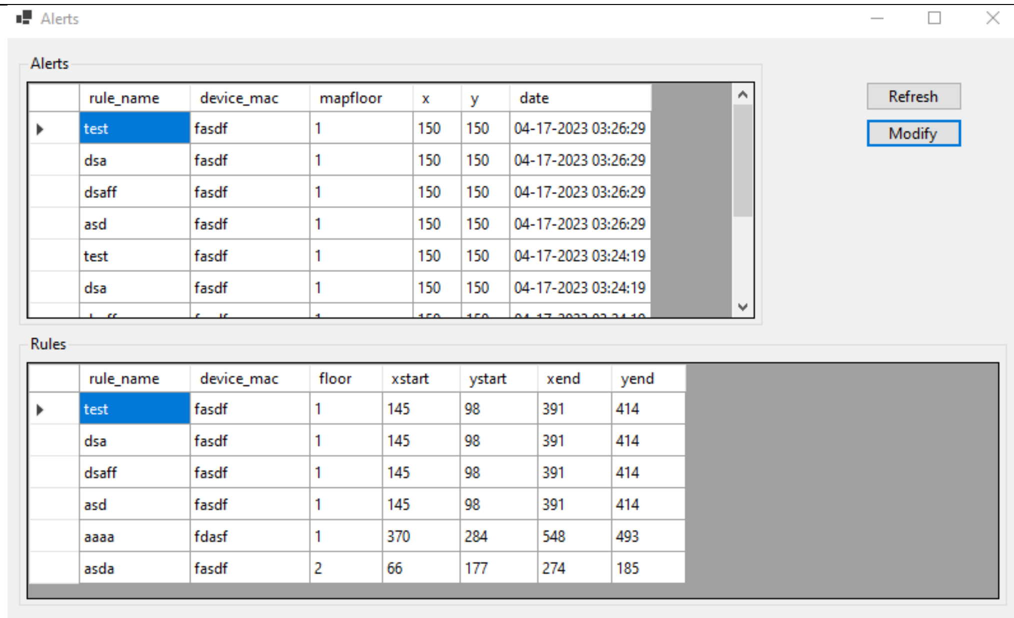
The database result shows that the beacon has been successfully deleted from the server.



7 **Viewing the alerts on the server**

The ability to view alerts will be shown in the GUI application. The following two screenshots show the MainProgram form opening the Alerts form.





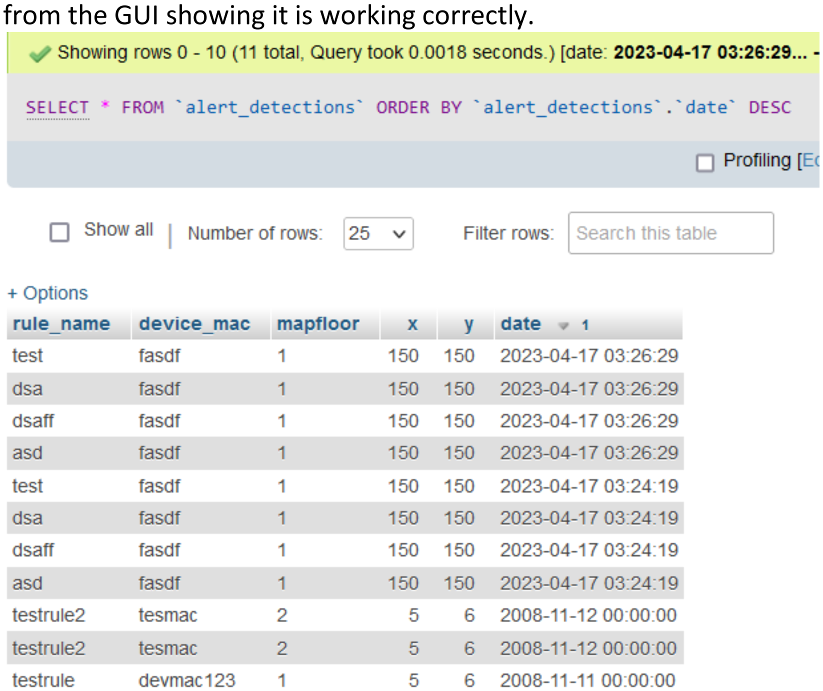
Alerts

rule_name	device_mac	mapfloor	x	y	date
test	fasdf	1	150	150	04-17-2023 03:26:29
dsa	fasdf	1	150	150	04-17-2023 03:26:29
dsaff	fasdf	1	150	150	04-17-2023 03:26:29
asd	fasdf	1	150	150	04-17-2023 03:26:29
test	fasdf	1	150	150	04-17-2023 03:24:19
dsa	fasdf	1	150	150	04-17-2023 03:24:19

Rules

rule_name	device_mac	floor	xstart	ystart	xend	yend
test	fasdf	1	145	98	391	414
dsa	fasdf	1	145	98	391	414
dsaff	fasdf	1	145	98	391	414
asd	fasdf	1	145	98	391	414
aaaa	fdasf	1	370	284	548	493
asda	fasdf	2	66	177	274	185

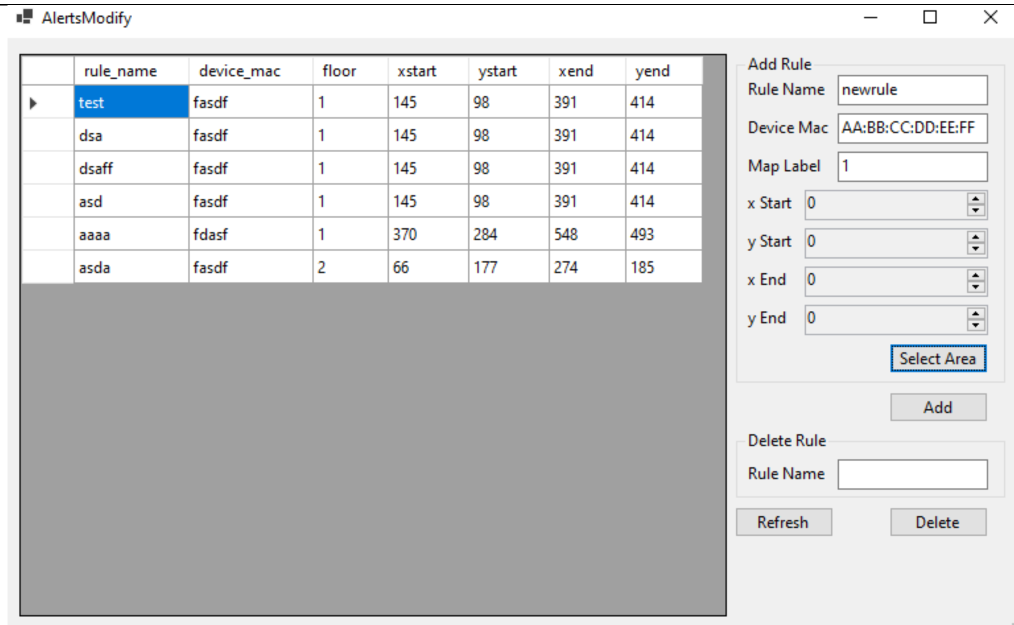
The table above shows the detections found based on the table of rules. The SQL query from the 'alert_detections' table matches with the results in the Alerts table from the GUI showing it is working correctly.



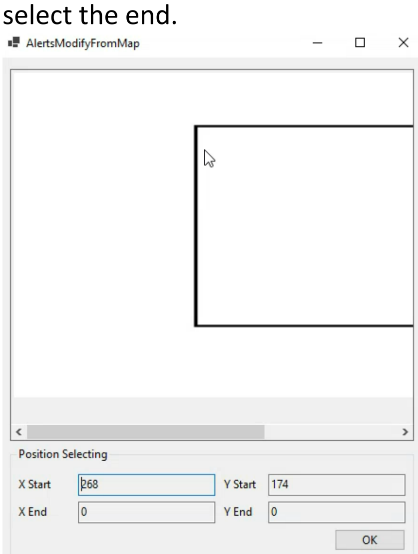

```
SELECT * FROM `alert_detections` ORDER BY `alert_detections`.`date` DESC
```

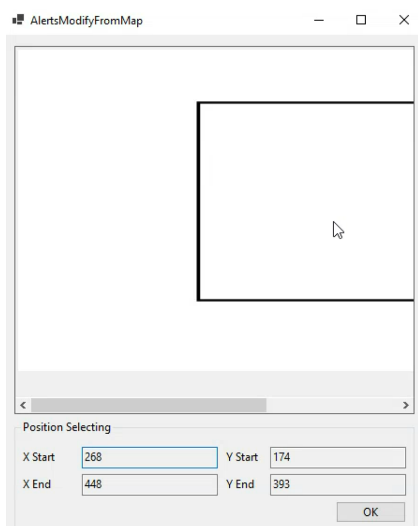
rule_name	device_mac	mapfloor	x	y	date
test	fasdf	1	150	150	2023-04-17 03:26:29
dsa	fasdf	1	150	150	2023-04-17 03:26:29
dsaff	fasdf	1	150	150	2023-04-17 03:26:29
asd	fasdf	1	150	150	2023-04-17 03:26:29
test	fasdf	1	150	150	2023-04-17 03:24:19
dsa	fasdf	1	150	150	2023-04-17 03:24:19
dsaff	fasdf	1	150	150	2023-04-17 03:24:19
asd	fasdf	1	150	150	2023-04-17 03:24:19
testrule2	tesmac	2	5	6	2008-11-12 00:00:00
testrule2	tesmac	2	5	6	2008-11-12 00:00:00
testrule	devmac123	1	5	6	2008-11-11 00:00:00

8	**Adding a new rule to the server** Adding a new rule can be done by clicking the "Modify" button from the Alerts window. This will open the AlertsModify form. Fill in the rule name, device MAC, and map label.	Pass

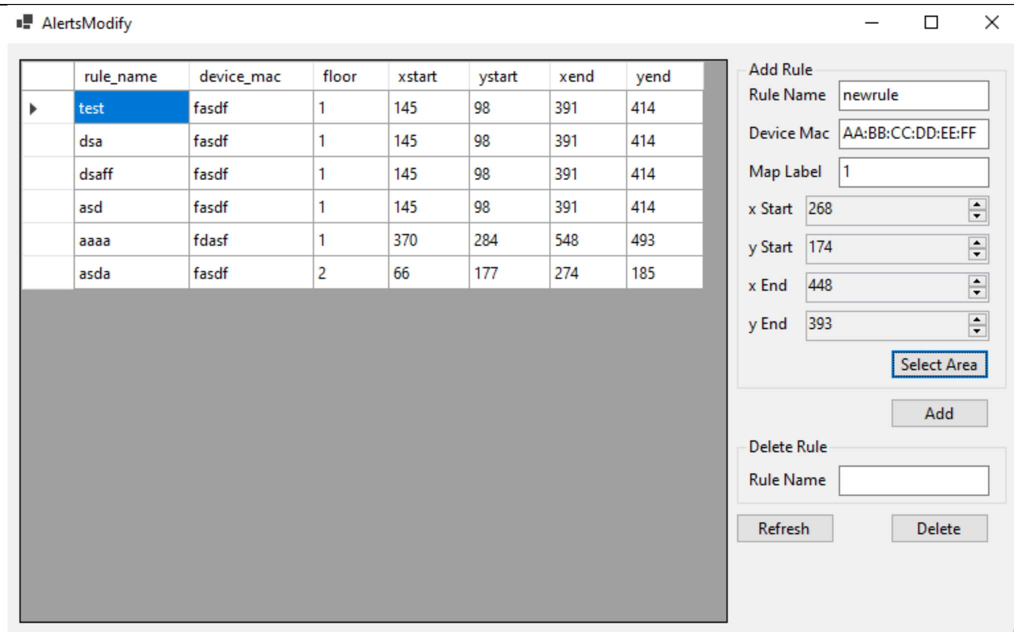


To select the X and Y rectangle area, click the "Select Area" button to open the AlertsModifyFromMap form where the rectangle can be selected by clicking and dragging the mouse. Left click down selects the beginning, releasing left click will select the end.

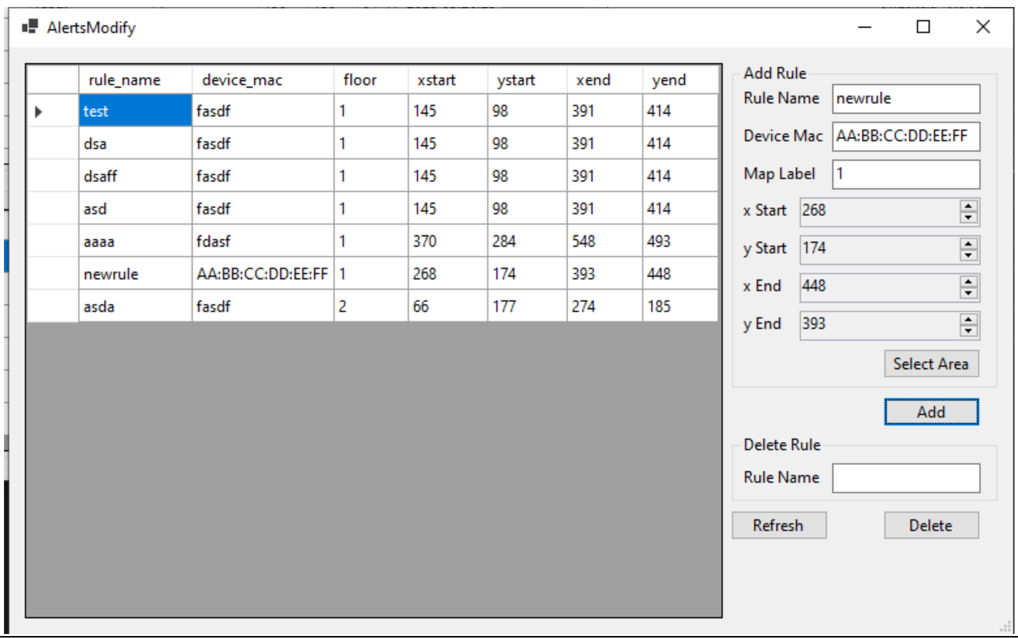


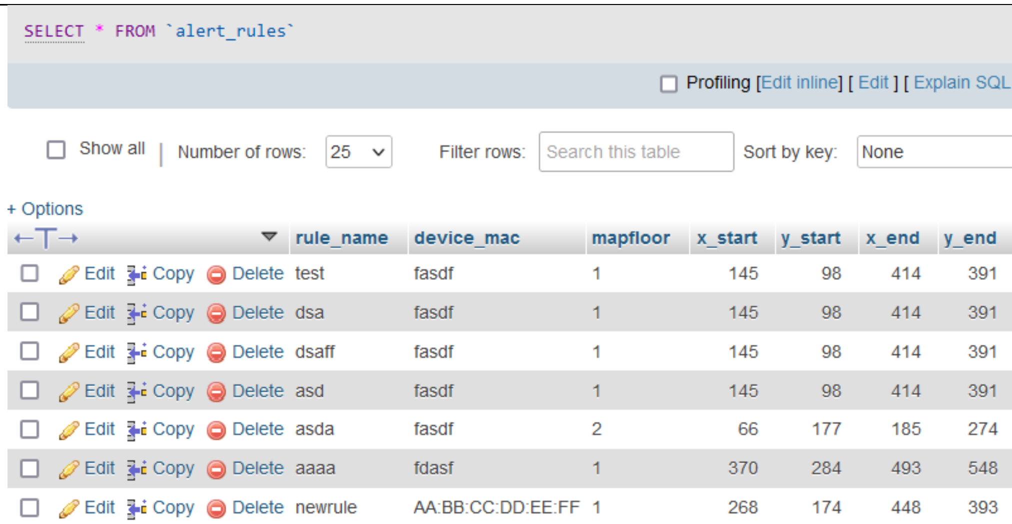


Clicking "OK" will then transfer the X and Y coordinates to the previous form.



Clicking add will finally add the new rule to the database which is also shown in the GUI.





```
SELECT * FROM `alert_rules`
```

Profiling [Edit inline] [Edit] [Explain SQL]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

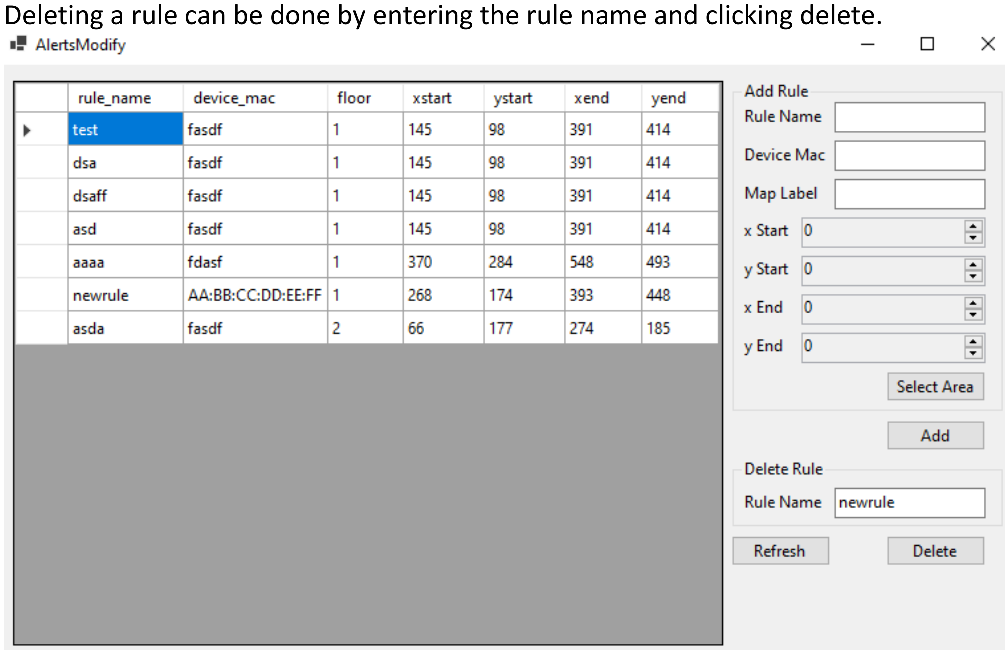
+ Options

	rule_name	device_mac	mapfloor	x_start	y_start	x_end	y_end
Edit Copy Delete	test	fasdf	1	145	98	414	391
Edit Copy Delete	dsa	fasdf	1	145	98	414	391
Edit Copy Delete	dsaff	fasdf	1	145	98	414	391
Edit Copy Delete	asd	fasdf	1	145	98	414	391
Edit Copy Delete	asda	fasdf	2	66	177	185	274
Edit Copy Delete	aaaa	fdasf	1	370	284	493	548
Edit Copy Delete	newrule	AA:BB:CC:DD:EE:FF	1	268	174	448	393

The new rule is successfully added as shown by both the SQL database and GUI images.

9 **Deleting a rule on the server**

Deleting a rule can be done by entering the rule name and clicking delete.

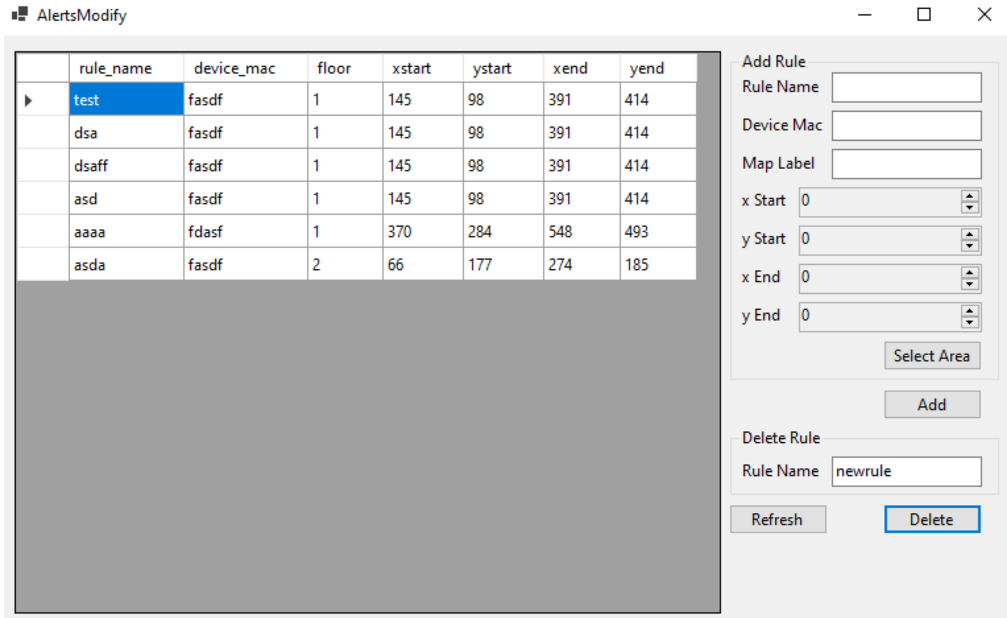


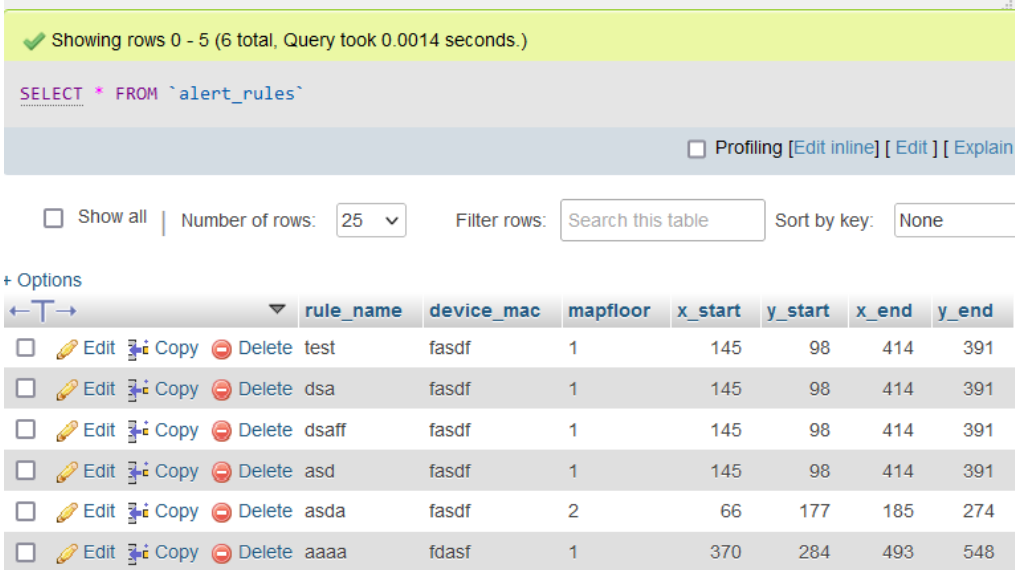
AlertsModify

rule_name	device_mac	floor	xstart	ystart	xend	yend
test	fasdf	1	145	98	391	414
dsa	fasdf	1	145	98	391	414
dsaff	fasdf	1	145	98	391	414
asd	fasdf	1	145	98	391	414
aaaa	fdasf	1	370	284	548	493
newrule	AA:BB:CC:DD:EE:FF	1	268	174	393	448
asda	fasdf	2	66	177	274	185

Pass

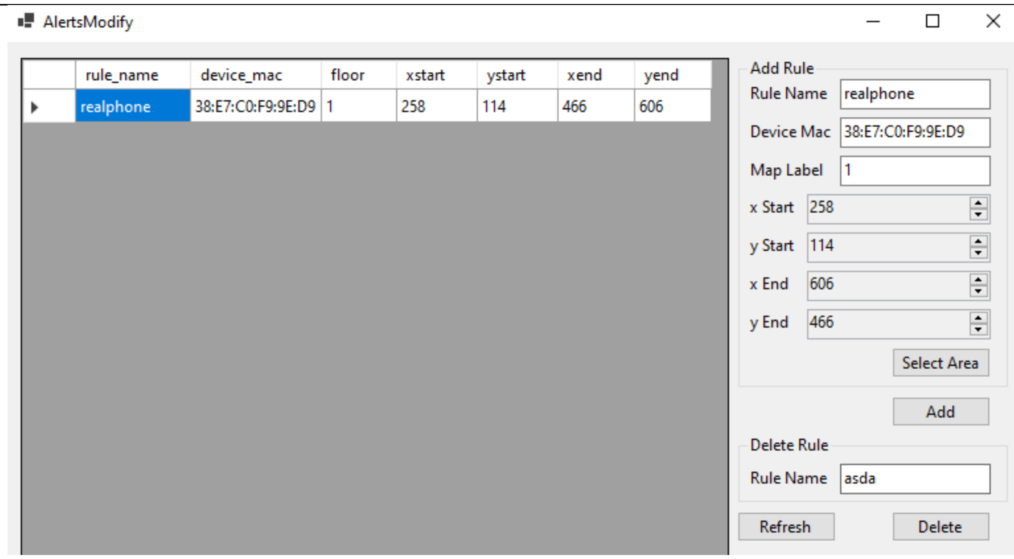
After clicking delete, the rule will be gone from both the table in the GUI and the database.



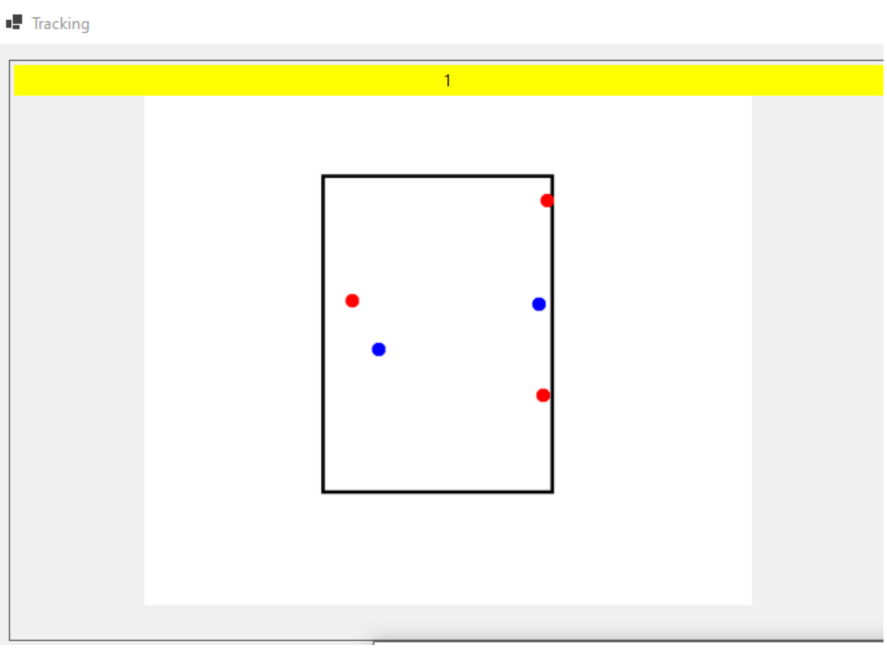


This shows that the rule has successfully been deleted.

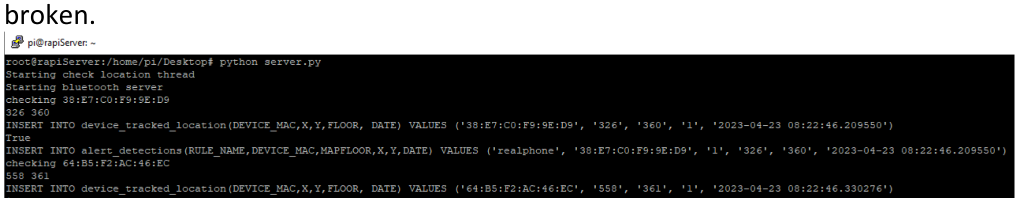
10	**Area rule broken will log to alert_detections** The rule ‘realphone’ is made for the 38:E7:C0:F9:9E:D9 device_mac.	Pass



38:E7:C0:F9:9E:D9 is the blue dot on the right in the tracking view, and is in the area specified by the ‘realphone’ rule.

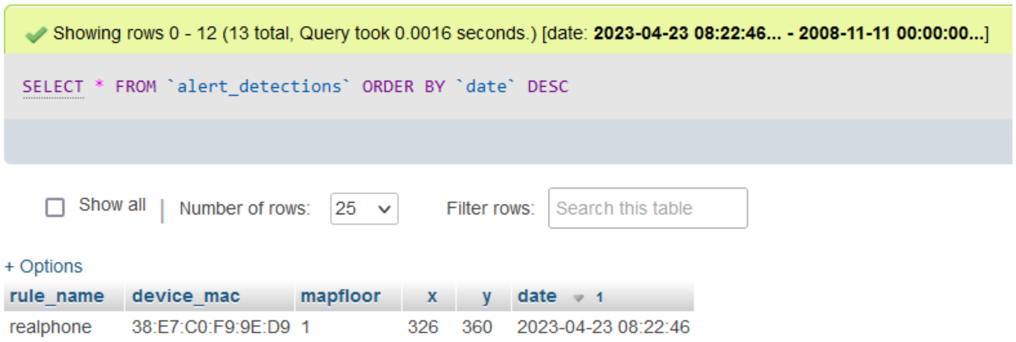


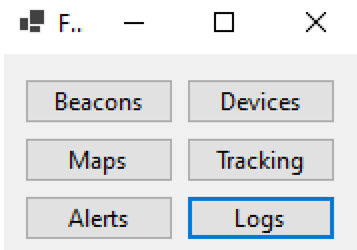
While the server is operating, it will check the locations of each device and check if it has broken any rules. The console shows SQL command that it will execute to add the new alert_detections entry to the table because it detected the rule being broken.



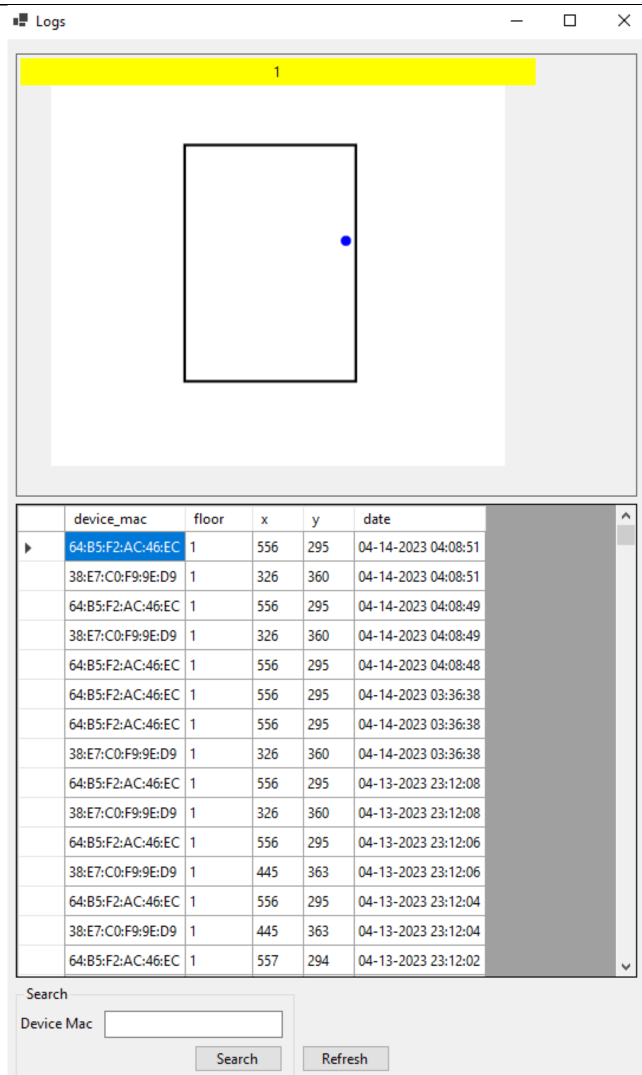
```
pi@rapiServer: ~
root@rapiServer:/home/pi/Desktop# python server.py
Starting check location thread
Starting bluetooth server
checking 38:E7:C0:F9:9E:D9
326 360
INSERT INTO device_tracked_location(DEVICE_MAC,X,Y,FLOOR, DATE) VALUES ('38:E7:C0:F9:9E:D9', '326', '360', '1', '2023-04-23 08:22:46.209550')
True
INSERT INTO alert_detections(RULE_NAME,DEVICE_MAC,MAPFLOOR,X,Y,DATE) VALUES ('realphone', '38:E7:C0:F9:9E:D9', '1', '326', '360', '2023-04-23 08:22:46.209550')
checking 64:B5:F2:AC:46:EC
558 361
INSERT INTO device_tracked_location(DEVICE_MAC,X,Y,FLOOR, DATE) VALUES ('64:B5:F2:AC:46:EC', '558', '361', '1', '2023-04-23 08:22:46.330276')
```

The ‘realphone’ rule was broken and is logged to the SQL database.

<table>
<tr>
<td></td>
<td>


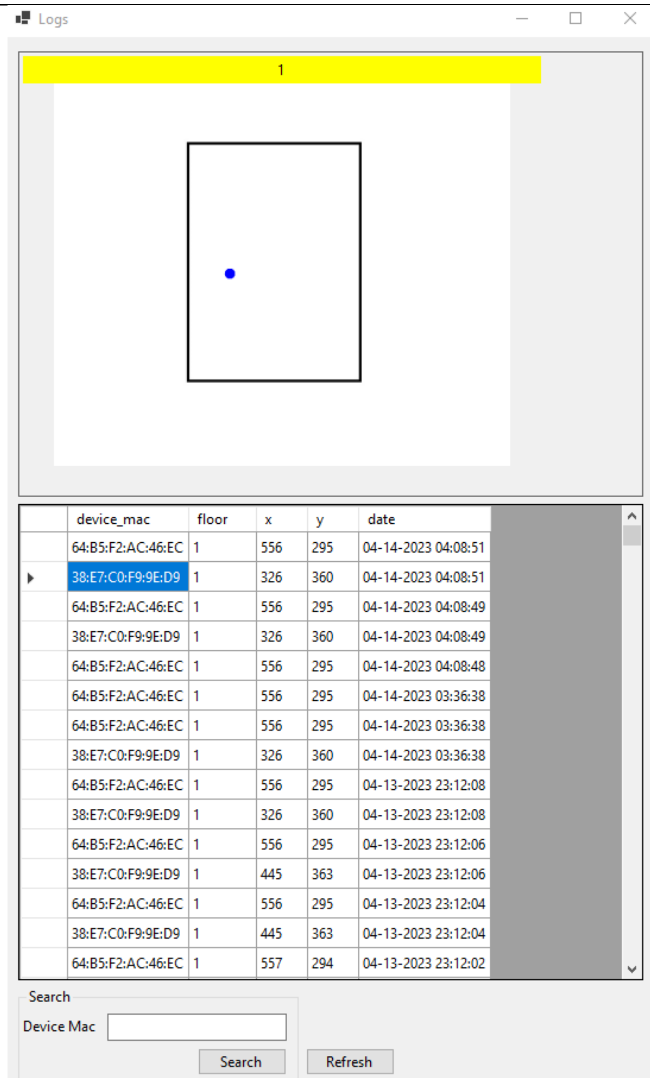
This shows that it was successful in detecting a rule being broken and logging the result to the SQL database.
</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>
Able to view tracking logs
The ability to view logs will be shown in the GUI application. The following two screenshots show the MainProgram form opening the Logs form.


</td>
<td>Pass</td>
</tr>
</table>

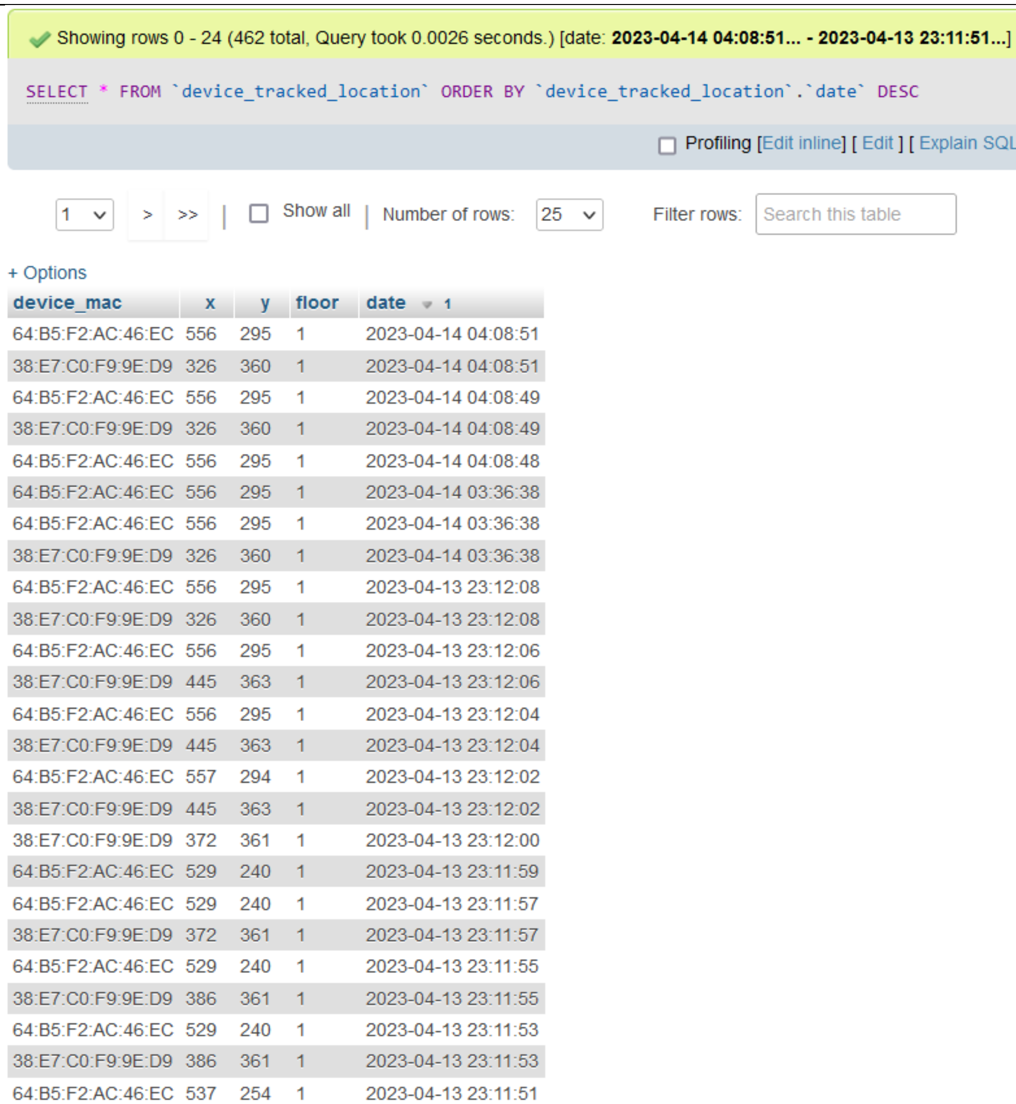


device_mac	floor	x	y	date
64:B5:F2:AC:46:EC	1	556	295	04-14-2023 04:08:51
38:E7:C0:F9:9E:D9	1	326	360	04-14-2023 04:08:51
64:B5:F2:AC:46:EC	1	556	295	04-14-2023 04:08:49
38:E7:C0:F9:9E:D9	1	326	360	04-14-2023 04:08:49
64:B5:F2:AC:46:EC	1	556	295	04-14-2023 04:08:48
64:B5:F2:AC:46:EC	1	556	295	04-14-2023 03:36:38
64:B5:F2:AC:46:EC	1	556	295	04-14-2023 03:36:38
38:E7:C0:F9:9E:D9	1	326	360	04-14-2023 03:36:38
64:B5:F2:AC:46:EC	1	556	295	04-13-2023 23:12:08
38:E7:C0:F9:9E:D9	1	326	360	04-13-2023 23:12:08
64:B5:F2:AC:46:EC	1	556	295	04-13-2023 23:12:06
38:E7:C0:F9:9E:D9	1	445	363	04-13-2023 23:12:06
64:B5:F2:AC:46:EC	1	556	295	04-13-2023 23:12:04
38:E7:C0:F9:9E:D9	1	445	363	04-13-2023 23:12:04
64:B5:F2:AC:46:EC	1	557	294	04-13-2023 23:12:02

When clicking any of the entries in the table of past detections, the window above will show the calculated position of the device on the relevant map.



	device_mac	floor	x	y	date
	64:B5:F2:AC:46:EC	1	556	295	04-14-2023 04:08:51
▸	38:E7:C0:F9:9E:D9	1	326	360	04-14-2023 04:08:51
	64:B5:F2:AC:46:EC	1	556	295	04-14-2023 04:08:49
	38:E7:C0:F9:9E:D9	1	326	360	04-14-2023 04:08:49
	64:B5:F2:AC:46:EC	1	556	295	04-14-2023 04:08:48
	64:B5:F2:AC:46:EC	1	556	295	04-14-2023 03:36:38
	64:B5:F2:AC:46:EC	1	556	295	04-14-2023 03:36:38
	38:E7:C0:F9:9E:D9	1	326	360	04-14-2023 03:36:38
	64:B5:F2:AC:46:EC	1	556	295	04-13-2023 23:12:08
	38:E7:C0:F9:9E:D9	1	326	360	04-13-2023 23:12:08
	64:B5:F2:AC:46:EC	1	556	295	04-13-2023 23:12:06
	38:E7:C0:F9:9E:D9	1	445	363	04-13-2023 23:12:06
	64:B5:F2:AC:46:EC	1	556	295	04-13-2023 23:12:04
	38:E7:C0:F9:9E:D9	1	445	363	04-13-2023 23:12:04
	64:B5:F2:AC:46:EC	1	557	294	04-13-2023 23:12:02



Showing rows 0 - 24 (462 total, Query took 0.0026 seconds.) [date: **2023-04-14 04:08:51... - 2023-04-13 23:11:51...**]

```
SELECT * FROM `device_tracked_location` ORDER BY `device_tracked_location`.`date` DESC
```

Profiling [Edit inline] [Edit] [Explain SQL

1 > >> | Show all | Number of rows: 25 Filter rows: Search this table

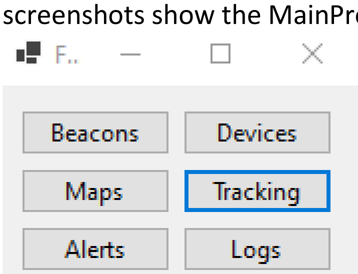
+ Options

device_mac	x	y	floor	date 1
64:B5:F2:AC:46:EC	556	295	1	2023-04-14 04:08:51
38:E7:C0:F9:9E:D9	326	360	1	2023-04-14 04:08:51
64:B5:F2:AC:46:EC	556	295	1	2023-04-14 04:08:49
38:E7:C0:F9:9E:D9	326	360	1	2023-04-14 04:08:49
64:B5:F2:AC:46:EC	556	295	1	2023-04-14 04:08:48
64:B5:F2:AC:46:EC	556	295	1	2023-04-14 03:36:38
64:B5:F2:AC:46:EC	556	295	1	2023-04-14 03:36:38
38:E7:C0:F9:9E:D9	326	360	1	2023-04-14 03:36:38
64:B5:F2:AC:46:EC	556	295	1	2023-04-13 23:12:08
38:E7:C0:F9:9E:D9	326	360	1	2023-04-13 23:12:08
64:B5:F2:AC:46:EC	556	295	1	2023-04-13 23:12:06
38:E7:C0:F9:9E:D9	445	363	1	2023-04-13 23:12:06
64:B5:F2:AC:46:EC	556	295	1	2023-04-13 23:12:04
38:E7:C0:F9:9E:D9	445	363	1	2023-04-13 23:12:04
64:B5:F2:AC:46:EC	557	294	1	2023-04-13 23:12:02
38:E7:C0:F9:9E:D9	445	363	1	2023-04-13 23:12:02
38:E7:C0:F9:9E:D9	372	361	1	2023-04-13 23:12:00
64:B5:F2:AC:46:EC	529	240	1	2023-04-13 23:11:59
64:B5:F2:AC:46:EC	529	240	1	2023-04-13 23:11:57
38:E7:C0:F9:9E:D9	372	361	1	2023-04-13 23:11:57
64:B5:F2:AC:46:EC	529	240	1	2023-04-13 23:11:55
38:E7:C0:F9:9E:D9	386	361	1	2023-04-13 23:11:55
64:B5:F2:AC:46:EC	529	240	1	2023-04-13 23:11:53
38:E7:C0:F9:9E:D9	386	361	1	2023-04-13 23:11:53
64:B5:F2:AC:46:EC	537	254	1	2023-04-13 23:11:51

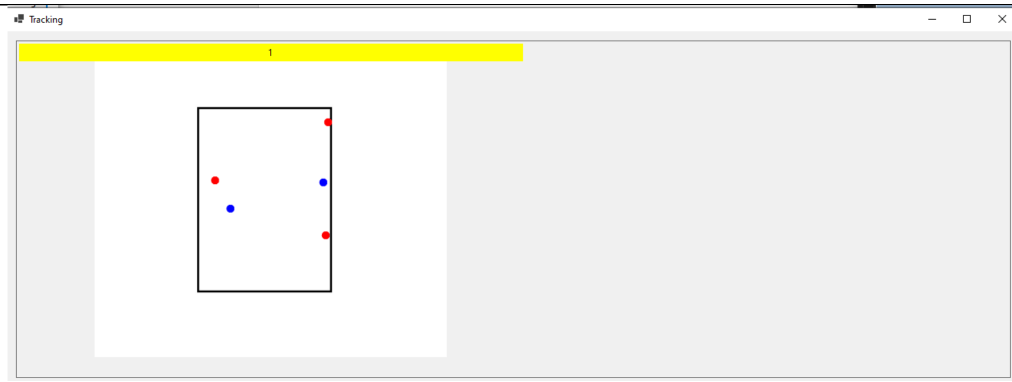
This shows that the logs are correctly working by displaying the same results as the database and is effective at showing the user the log entry.

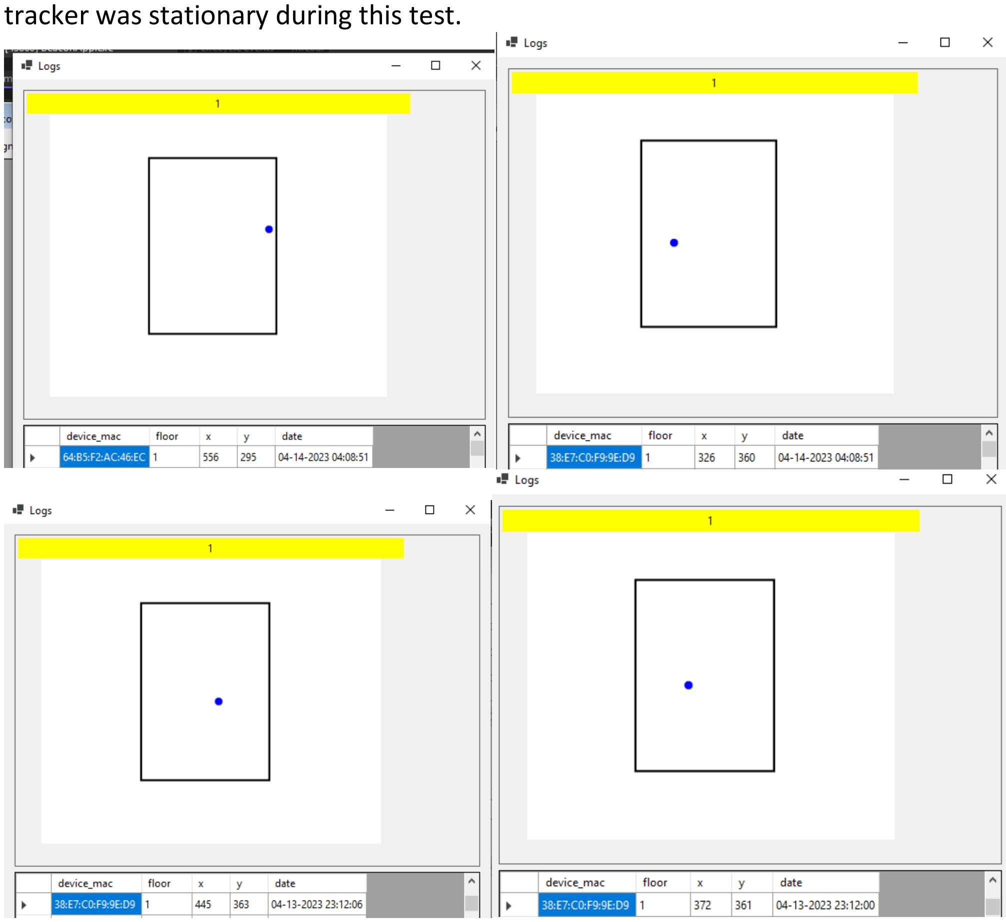
12 **Able to view live tracking**

The ability to view alerts will be shown in the GUI application. The following two screenshots show the MainProgram form opening the Alerts form.

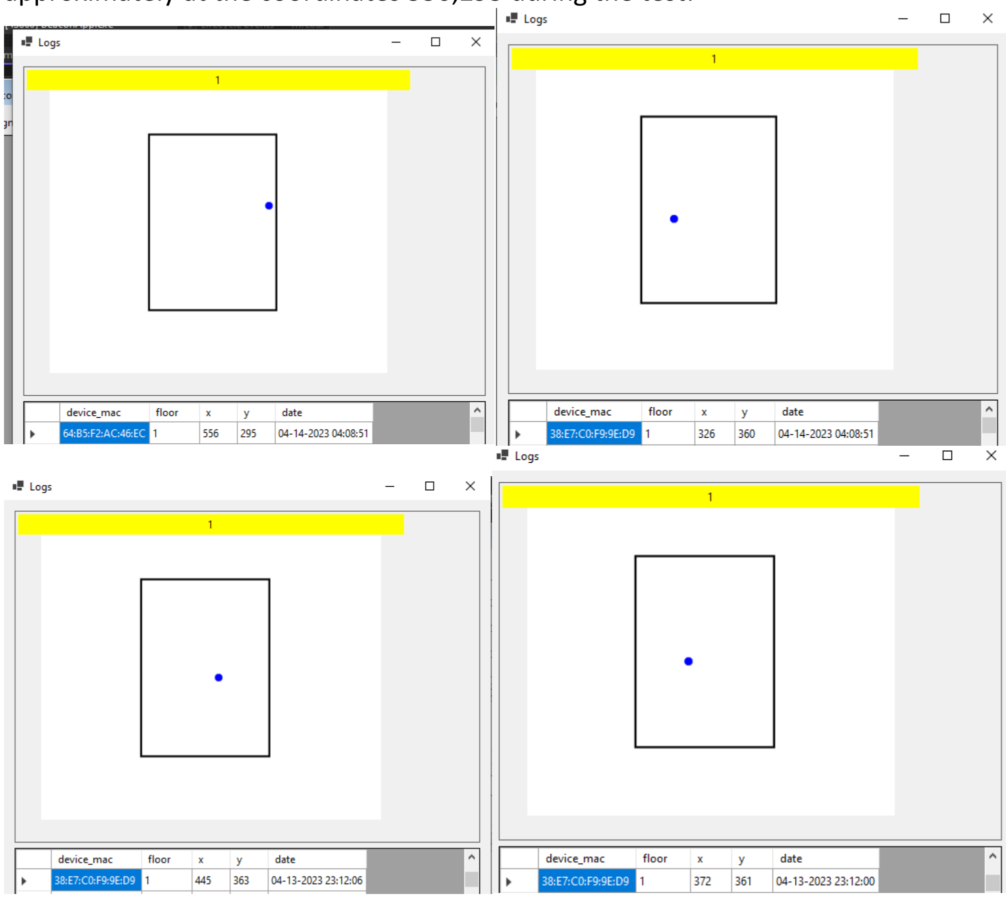


Pass

<table>
<tr><td></td><td>

The GUI correctly shows the current devices being tracked and the beacons in the area.</td><td></td></tr>
<tr><td>13</td><td>Tracking is room accurate
Based on the results seen in the logs, the tracker is consistently in the rectangle representing my testing area of 2.5m x 4m walled room as shown by my results. The tracker was stationary during this test.


The tracker being in my rectangular area shows that my trilateration algorithm can trilateration to the correct room, in this case it stays within the rectangular area.</td><td>Pass</td></tr>
<tr><td>14</td><td>Tracking is pinpoint accurate
Based on the results seen in the logs, the tracker cannot precisely pinpoint the location of my device accurately. My testing area is a 2.5m x 4m walled room as</td><td>Fail</td></tr>
</table>

	shown by my room. The tracker was stationary during this test and was approximately at the coordinates 556,295 during the test.  Over the test, the calculated location will drift around the rectangular area and sometimes be in the correct spot but will usually be about 1-2m inaccurate. Because of this drifting despite the tracker not moving, it would be fair to say that it failed at being pinpoint accurate.	

2.7. Implications of Implementation

In my implementation, I changed some things from what was stated in my initial proposal because of some issues that appeared during the development of this project.

Initially I wanted to use BLE as the scanning medium, however there were some issues with implementing BLE on the Android platform. This is because Android forces randomized device address which constantly changes when broadcasting; however, when using BluePy on Python for broadcasting BLE on beacons, the actual unchanging Bluetooth MAC address is broadcasted. This is an issue because I needed the actual Bluetooth MAC address to log into my database as it is a lot easier to identify.

I opted to use regular Bluetooth because after testing, the broadcasted address from the Android device is the same as the actual Bluetooth MAC address. I switched to PyBluez for scanning regular Bluetooth for beacons to match with the Android devices using regular Bluetooth.

For the graphical user interface, I opted to use C# Windows Forms instead of my proposed Python for GUI. This is because I tried creating a basic interface with Python using Tkinter however, it was very time intensive positioning elements. I researched for more popular languages for creating GUI and decided to use C# Windows Forms because it is well known for its ease of use because of the drag and drop nature of placing GUI elements. The main implication of using Windows Forms for the GUI means it can only run on Windows based operating systems.

2.8. Innovation

The innovation of this project is the usage of beacons to track multiple people in a building. This is different from conventional tracking which uses GPS which is weak in buildings due to problems such as bad signal and being unable to track what floor the tracker is on. By using beacons, it can track what floor a tracker is at because the beacon on the floor will detect the tracker and report it back to the server and triangulate its location. This would work if there were enough beacons around the building. There are no applications that would allow multiple devices to be tracked and continuously monitored, which would be useful in some situations.

This project is different from other tracking tools because it is only meant to track many people in a specific building or area where this system is installed. Other tracking systems such as Apple AirTag which can track around the world however, it only tracks one device at a time.

2.9. Complexity

The complexity for this project lies in calculating the location of where the tracker is by using trilateration of RSSI dBm from three beacons. The program combines all the information received by the beacons to be processed by the server and displays it all in an easy to digest manner for the user through a map. The most complex part of the project is writing the trilateration algorithm because of many factors such as walls, differing signal strength from other devices, and other interference that may be present in the area.

For a diploma student, it may present a challenge as unknown concepts such as how to connect all the devices together and communicate to the server. The major roadblock to diploma students would be implementing the algorithm for trilateration. Students in the diploma program usually write code for only one device and not multiple devices communicating together.

The real problem that is not trivial to solve is the way to process all the data together and display it on a map of the building the beacons are located at. The customizability of the server software allows a user to place beacons markers, map out the building, add devices to track, and notifications if a tracker is detected in a specific area.

The areas of specialty knowledge that I would need to solve the problem would be to design a server interface that communicates with multiple beacons and can be adjusted whenever the layout of the building changes, more beacons are added, or more trackers are added.

The knowledge I am lacking to research to explore to solve this problem is location trilateration with RSSI dBm, scanning for nearby Bluetooth devices, determining how to distinguish which beacon is closest to the tracker, GUI interface for the server to map and operate the program.

2.10. Research in New Technologies

This project uses research on location detection algorithms to determine the location of a device by using wireless signals as a measurement tool. The following are the libraries and technologies used in this project.

- Raspberry PI(s)
- Bluetooth
- SQL and SQL Server
- Python
- C#
- Kotlin
- Android Studio
- Visual Studio

Raspberry PI is a popular minicomputer platform that is popular for developing projects meant for small low power devices that need little computation power. Bluetooth is used in many mobile devices for convenient low range and power communications. SQL and SQL server is used as the data storage for logging detected devices by the beacons and retrieval solution to easily insert and search for relevant data needed for the project.

Python is a popular versatile language because of its many libraries and frameworks that are used by developers. Python is frequently used for creating prototypes because creating the same program in lower-level languages such as C would be more difficult as it is more barebone.

C# is a language developed by Microsoft which allows drag and drop creation of a graphical user interface. C# was chosen to create the GUI because creating the GUI in another language such as Python would require lots of testing to move objects into the correct positions.

Kotlin is a language used in Android Studio for development of Android applications.

2.11. Future Enhancements

There are many features I would like to include as future enhancements which I discovered over the development of this project.

- More than Three Beacons Used: Allow the location algorithm to have the ability to use more than three beacons to calculate location for higher accuracy.
- Beacons from Other Floors Assist: Beacons from other floors/maps being able to be used in the location algorithm to increase the number of beacons used in the location algorithm.
- Implement Bluetooth Low Energy to allow testing of both regular Bluetooth and Bluetooth Low Energy.
- RSSI Smoothing Algorithm: It will smooth out the RSSI dBm received from beacons as I had RSSI dBm fluctuations while scanning for devices such as going from -60 RSSI dBm to -10 RSSI dBm alternating despite the device not moving at all. An example of a smoothing algorithm is a weighted or average filter.

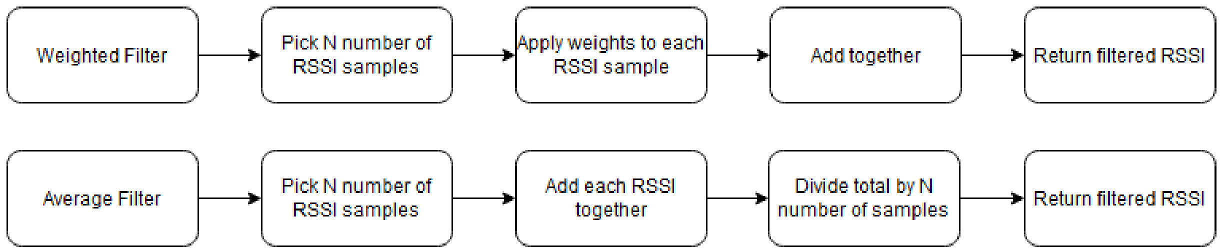


Figure 7 Diagram of basic smoothing filters

```
Weighted Filter
    Pick n number of RSSI samples of specific device
    Multiply each sample by corresponding weight
    n1 = rssiN1 * 0.65
    n2 = rssiN2 * 0.20
    n3 = rssiN3 * 0.10
    n4 = rssiN4 * 0.05
    filterRSSI = n1 + n2 + n3 + n4
    return filterRSSI

Average Filter
    Pick n number of RSSI samples of specific device
    total = rssiN1 + rssiN2 + rssiN3 + rssiN4
    filterRSSI = total / n
    return filterRSSI
```

Figure 8 Pseudocode of examples of smoothing filters

2.12. Timeline and Milestones

Task	Time Estimate Required (h)/(Total)	Time Actual Required (h)/(Total)	Start Period
Phase 1: Setup of Devices			
Order more Raspberry PIs for beacons	2(2)	2(2)	Week 1
Download required libraries and dependencies	2(4)	4(6)	Week 1
Create databases	2(6)	4(10)	Week 1
Learn the libraries required for program	20(26)	22(32)	Week 1
Create basic tracking program for Android tracker	10(36)	12(44)	Week 2
Create basic beacon program for Raspberry PI	10(46)	12(56)	Week 2
Create basic server program for Windows	10(56)	12(68)	Week 2
Test basic program	5(61)	8(76)	Week 2
Phase 2: Refining Basic Programs			
Improve user interface and program for Android tracker	10(71)	6(82)	Week 3

Improve program of Raspberry Pi beacon	10(81)	20(102)	Week 3
Implement user interface for Windows server	10(91)	12(114)	Week 3
Add database to Windows server	5(96)	5(119)	Week 4
Learn database calls required to get the information required	10(106)	16(135)	Week 4
Phase 3: Implement Interface Tracking Feature			
Implement user interface option to upload PNG map to read	5(111)	8(143)	Week 5
Implement user interface to add beacons to map	10(121)	10(153)	Week 5
Implement user interface to assign beacon names	5(126)	7(160)	Week 5
Implement user interface option to select a tracker to follow	5(131)	6(166)	Week 5
Implement user interface option to show log to a tracker	10(141)	12(178)	Week 6
Implement database call to retrieve required information	20(161)	20(198)	Week 6
Test base tracking feature	10 (171)	14(212)	Week 6
Phase 4: Implement Interface to Show Approximate Location			
Implement user interface to show approximate location	10(181)	16(228)	Week 7
Implement database call to retrieve required information	10(191)	12(240)	Week 7
Implement scrolling through tracker log timeline	20(211)	18(258)	Week 7
Implement display of approximate location by lightning up beacon on map	10(221)	16(274)	Week 8
Test approximate location feature	10(231)	14(286)	Week 8
Phase 5: Implement alert system			
Implement user interface to enable and set alerts	20(251)	16(304)	Week 9
Implement backend loop to check logs that a tracker has been detected by a specified area	30(281)	20(324)	Week 10
Implement function to send alert to the server operator and tracker	20(301)	10(334)	Week 11
Test alert system feature	10(311)	10(344)	Week 12
Phase 6: Wrap up and report			
Gather data and write report analysis	20(331)	20(364)	Week 13
Gather data and write documentation	20(351)	24(388)	Week 14
Extra time for report changes	TBD	8(396)	Week 15
Total Time	**351**	**396**	

3. Conclusion

In conclusion, this project focused on tracking devices by utilizing features that are typically overlooked such as RSSI dBm being used to measure distance and calculating the position by using three of these results. Other tracking products such as Apple AirTag utilize similar technologies of converting RSSI. This gives me a glimpse of how other technologies function because of having the experience of developing it.

In the CST and bachelor program, I rarely had any interactions with using SQL besides the very first year of CST where it was mostly executing SQL commands from phpMyAdmin. In this project I learned to integrate a SQL database with a program in both Python and C# Windows Forms by using SQL as the backbone of all the data the program requires to use. This experience is crucial as all projects will need some sort of database to interact in the modern day always connected world.

3.1. Lessons Learned

This project taught me that I should be open to changes when experiencing problems when using different languages, libraries, or technologies. Such changes are switching from trying to use Bluetooth Low Energy to using standard Bluetooth because of issues with detecting the correct addresses. Using C# Windows Forms instead of Python Tkinter greatly sped up my GUI creation process but, there were still many windows to make, and it restricted the GUI program to only work on Windows.

It was very easy to feel despair that my project wasn't in a working state yet because there is a lot of groundwork to set up before the main work of using trilateration to determine location could be done. Once the groundwork was completed, which was setting up the databases and coding the beacons and basic server, I felt a slight ease because I could finally start seeing actual progress on my project.

I learned that some platforms may have unexpected behavior such as the address randomization when using Bluetooth Low Energy on the Android platform which doesn't happen on Python.

I occasionally encountered issues of not knowing how to do something. A trick that helped is to take a step away from it and slowly think about it instead of trying to hammer away by trying bits of code and hoping it works which would only make me more frustrated.

There were many lessons to be learned from experiencing pitfalls in the creation of this project.

3.2. Closing Remarks

Throughout the project, I constantly felt lost in this project. I was very lost in the beginning with trying to implement Bluetooth to the project because of the difficulties of trying to use Bluetooth Low Energy on an Android device. I lost lots of time trying to do this and had to put my foot down to use regular Bluetooth because I was starting to fall over two weeks behind my milestones.

As the project is now completed, I wish I didn't opt to make the mobile Android application to use as a tracking device. This is because when I tried testing Bluetooth Low Energy broadcasts on the beacons, they were using the actual Bluetooth address instead of the randomized one like on Android.

If I had more time, I would try to add Bluetooth Low Energy as an option for this program so I could test the performance difference between regular and low energy Bluetooth.

This project was great in helping me understand how other similar products such as Apple AirTag function in its location sensing properties. The main difference between my project and a commercial product is the scale and finely tuned algorithms.

From completing this project, I can imagine how AirTags function. The well-known detail is AirTags can use every iPhone in the area to act as a beacon. The RSSI detected from the AirTag is sent back to Apple along with the GPS location of the iPhone. Finally, this information is then used with trilateration and or triangulation to greatly increase the accuracy of the location algorithm.

4. Appendix

4.1. Approved Proposal

Tracking in Building Using Bluetooth Beacons

COMP8047 – Major Project

Table of Contents

5. Student Background

I am a student in the Bachelor of Technology in Network Security Applications Development program at BCIT. Since taking the courses at BCIT, I have learned the importance of network security and techniques to prevent hackers from reading information. This project will use encryption and security techniques to communicate between trackers, beacons, and the server.

5.1. Education

I learned some basic programming in high school which led to an interest to pursue programming in post-secondary education. I completed my two years for the Computer Science Technology diploma at BCIT. Since I looked at many job postings seem to desire a bachelor's degree or higher, I decided to pursue the BTech program.

British Columbia Institute of Technology (BCIT)

Computer Systems Technology Diploma	***Graduated 2019***
Computer Systems Network Security Applications Development (BTech)	***Graduating May 2023***

5.2. Work Experience

After graduating from CST in 2019, I wasn't too lucky in looking for any tech or programming work. I decided to work a minimum wage job for about a year to save for tuition before applying for the Network Security Applications Development Bachelor of Technology program.

Tim Hortons **2016 – 2017**
Cashier – Take the customer's orders and make their order

Panago Pizza **2016 – 2018**
Pizza Maker – Make pizzas according to the customer's order

Sushi Go **2018 – 2019**
Cashier – Take the customer's orders on cashier and phone
Cook – Cook the dishes that customers have ordered

Impark **2019 – Present**
Parking Enforcement – Write parking tickets when parking has not been paid for

2. Project Description

The aim of this project is to create a system that could track the approximate locations of people by using a tracker, beacons, and a server to communicate all the information to. This application will have a tracking application which sends out a Bluetooth packet in the area which is read by Raspberry Pi beacons listening for trackers. The information from the beacons is then sent to the server which calculates the approximate location of the tracker and displays it on a map. By using this, the approximate location of someone can be found quickly.

3. Problem Statement and Background

The problem my application will solve is being able to able to find someone quickly in buildings such as a hospital. By knowing the general location someone may be at, the security personnel can quickly head to that area or check that area's cameras instead of trying to search every camera in the building.

I hear stories of people escaping hospitals or being taken out without permission such as the recent amber alert of the 3-year-old child being taken out of the hospital. By being able to tell where someone is supposed to be at a certain time, this system could be used when someone is moved without permission and security can take the appropriate actions.

I notice many large organizations or buildings have mesh Wi-Fi networks set up. This preexisting wireless network will used for the beacons to connect to and send data to the server. This is a low-cost solution because only the beacons can be low-cost devices such as Raspberry PIs that have Wi-Fi and Bluetooth connectivity.

The locations shown by the application will be calculated using triangulation using at least three beacons. Bluetooth triangulation has been shown to be fairly accurate at determining the location of a tracker based on multiple studies. BLE will be used as the protocol between the beacons and the trackers to get information such as the RSSI strength, TCP will be used as the protocol between the beacons and the server to forward the information to the server. BLE is commonly used in many Bluetooth tracking applications such as tracking sensors and tags. TCP is used as the backbone of communication between the beacon and server because of the preexisting Wi-Fi in many places.

The main software development effort is the Bluetooth triangulation algorithm to triangulate the location of the tracker by calculating the approximate area where the tracker is by the RSSI strength. GUI and the backend are additional efforts to allow a user to get the information they want, displayed in a simple to understand fashion. This is because I have no experience with Bluetooth triangulation and little experience with databases and graphical user interface creation.

4. Scope and Depth

The scope of this project is to develop beacon tracking system using Bluetooth with triangulation to detect tracked devices in the area and display the information in an easy-to-use user interface on the server. The tracking application will be a basic application that could be installed onto any Android phone which only sends out BLE Bluetooth packets to the beacons nearby of its RSSI value of each beacon it can detect. The beacons will receive the RSSI information from the tracker with the application installed and send the information to the server for processing.

Server features would include a triangulation algorithm and an easy to configure user interface. The user should be able to add new devices to the tracking network easily such as implementing new beacons or phones with the application installed. The user can upload floor maps and assign points where the beacons are on the floor map by clicking on the location where the beacon should be on the floor map and confirming the information. The information about the trackers and beacons will be displayed on the floor map the user has uploaded. A timeline of the position of the tracked device should be viewable on the server displaying where the device is at any point in time by maintaining the logged data in a database.

Outside of scope is developing a program that works across different network/subnets and using triangulation to find precise location of a device.

The feasibility of the project should be high. There are APIs that allow monitoring and sniffing of Bluetooth packets using Scapy and GUI libraries, and there is experimentation with Bluetooth tracking papers proving that it is possible to triangulate location using RSSI. The use of libraries for Bluetooth sniffing and GUI greatly reduces the length of development to be an obtainable goal of the 360 hours requirement which includes research and testing. Most of the development is programming the server to deal with edge cases such as when the tracker is detected by multiple beacons and GUI.

5. Test Plan

The testing for this project mainly consists of manual and acceptance testing. For example, placing the tracker in between two or more beacons and how the server responds. Testing the server features of adding and displaying information about devices and trackers in the network to ensure everything is working properly.

Draft Requirement	**Test**	**Passing Criteria**
A beacon should send back a response when a tracker device is nearby.	Place tracker device near a beacon	The tracker and location of beacon is logged by the server
The server should be able to handle multiple beacons detecting the same tracker	Place tracker device near multiple beacons	The tracker and location of the beacons are logged by the server along with signal strength.
Adding a new device to the server	Add a device to the tracking network	The new device gets added successfully
Adding a new map to the server	Add a map of a building	The new map gets added successfully
Ability to view the tracking logs	Open the log menu for a specific tracker	The history of the tracker is shown with the timeline showing

6. Methodology

The methodology for this project will be iterative and incremental. Iterative and incremental development allows the developer to create the basic foundations of the program using iterative development and then slowly build up each feature of the program with incremental development. This methodology is most suitable for this project because of the ability to test if everything works before advancing further for each portion of the program.

The Bluetooth sniffing will be handled by using the Scapy library and the GUI will be created using a Python Library. The first milestone is getting a tracker device to send a Bluetooth signal to a beacon and relay this information to the server. The second milestone will focus on building up the server by creating the GUI to add or remove devices in the tracking network. The third milestone is refining how the server responds when it receives information about a tracking device being detected by multiple beacons.

The first version of the software will be iterative as all the base components of the program will be created first. The server should be able to receive tracker data from the beacons in this first version. Every other version of the software will be incremental as it builds up the pieces of each part of the program such as adding one more feature each time such as viewing logs, improving the GUI.

7. System/Software Architecture Diagram

The following diagram shows the system architecture of this project and shows overall components and relationship between each other.

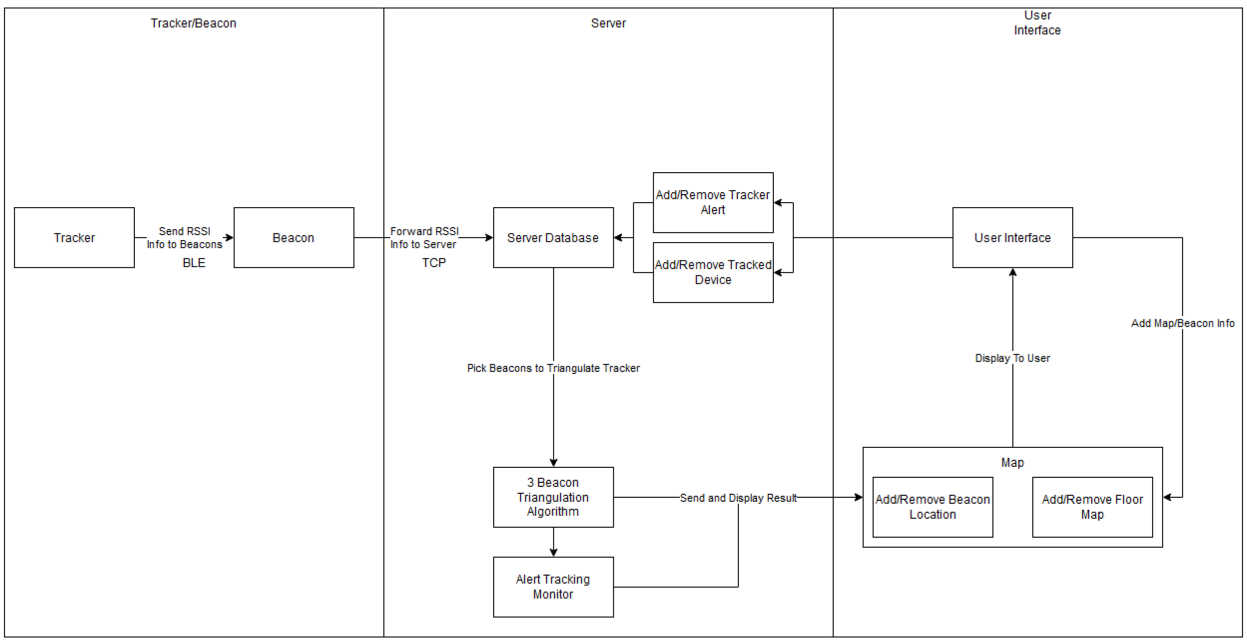


Figure 9: Diagram representing the Bluetooth Beacon Tracking Project

The following diagram represents the communication network between each device in the tracking network. The tracking device represents the phone installed with the tracking application. Beacons are Raspberry PIs running an application to detect specific Bluetooth signals and relay the information to the server. The server processes all the information received by the beacons into a simple to understand map for the user to view and allows adding of new beacons and devices.

The following diagram represents the software architecture diagram of my program and its interactions between each part of the program. The tracker waits for a special broadcast packet before sending the RSSI information back to the beacon that performed the broadcast. The beacon then forwards the tracker information with RSSI and the beacon information to the server database where it can be stored as together with tracker, beacon, and RSSI as an entry in the database. For each device to be tracked in the database, the three most recent and highest RSSI beacons will be selected to triangulate the location of the tracker.

```
while true
      Detect beacon broadcast BLE packet with unique identifier
      Send BLE packet to beacon containing RSSI information

Beacon
thread 1 while true
      Send broadcast BLE packet with unique identifier
      Repeat every 5 seconds
thread 2 while true
      Receive BLE packet from tracker containing RSSI information
      Send TCP packet containing RSSI information of the tracker and
beacon information

Server
while true
      Receive TCP packet containing RSSI information of the tracker and
beacon information
      Insert the packet information of tracker, beacon, and RSSI into the
database

While true
      For each tracker to be tracked
            pick 3 beacons with highest RSSI to tracker from database
            triangulate location of tracker using RSSI and known beacon
locations
            display triangulated location of beacon on map
            if tracker in specific area
                  send alert to user

GUI
Add tracker alert to database to watch for
specify tracker and area to alert the user
if tracker enters area, alert sounds and displays the tracker in question

Add tracked device
specify what device to track (only these devices will be added to
database)
when server receives packet, only add devices in the tracked device list

Map
add floor map
upload image files of building layout to use
add beacon location
specify coordinates, physical distance between beacons
switch to other floor with appropriate beacon locations when requested
```

The main software development will be the triangulation algorithm using RSSI to determine location, user interface and the interaction with the database that stores all the beacon tracking information. This effort would include the user interfaces to add new beacons and trackers, add alerts when a user is detected in a specified location, approximate the location of a user, and display the detection log of a tracked device. The appropriate database call would be required to get all the history of a specific device or add a device.

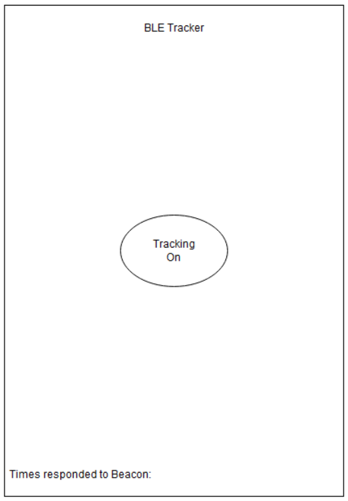


Figure 10 Basic tracker app for android

The Android application to track a device will be a simple application that can be turned on and run in the background of the device. The main function of the tracking app is to respond to beacons' broadcast searching for other tracker devices and responds back to the beacons with the RSSI strength of the link between each beacon.

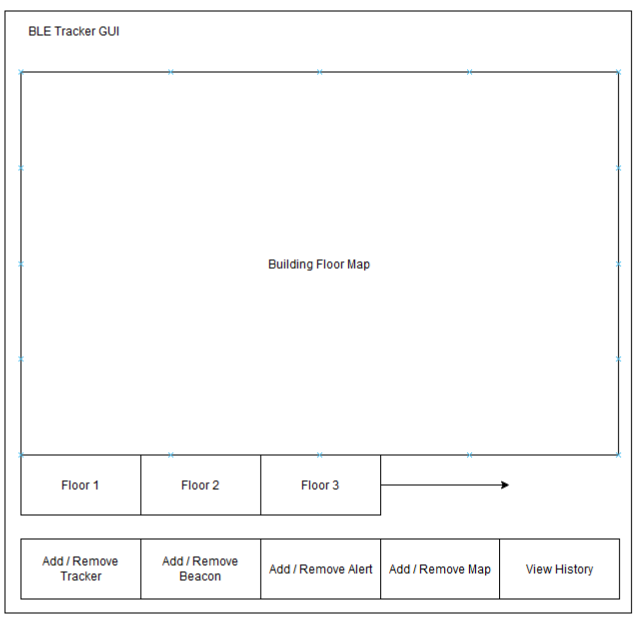


Figure 11 Application to view tracked devices

The desktop GUI application would allow a user to add and remove trackers, beacons, alerts, and maps from the program. By adding the tracker to the server, it will allow the application to enter valid tracked devices' the RSSI and beacon information into the database. Adding beacons to the map will allow the user to enter information about the beacon such as where it is in the building and distance from other beacons for triangulation calculations. Adding alerts will notify the user when a tracker enters a specific area on the floor map in the specified coordinates. Adding a map is a visual for the user to understand the information better as it is a representation of the specified floor.

8. Innovation

The innovation of this project is the usage of beacons to track multiple people in a building. This is different from conventional tracking which uses GPS which is weak in buildings due to problems such as bad signal and being unable to track what floor the tracker is on. By using beacons, it can track what floor a tracker is at because the beacon on the floor will detect the tracker and report it back to the server and triangulate its location. This would work if there were enough beacons around the building. There are no applications that would allow multiple devices to be tracked and continuously monitored which would be useful in some situations.

This project is different from other tracking tools because it is only meant to track many people in a specific building or area where this system is installed. Other tracking systems such as Apple Airtags can track around the world however, it only tracks one device at a time.

9. Complexity

The complexity for this project lies in calculating the location of where the tracker is using triangulation of RSSI from three beacons. The program combines all the information received by the beacons to be processed by the server and displays it all in an easy to digest manner for the user through the map. The most complex part of the project is writing the triangulation algorithm because of many factors such as walls, differing signal strength from other devices, and other interference that may be present in the area.

For a diploma student, it may present a challenge as unknown concepts such as how to connect all the devices together and communicate to the server. The major roadblock to diploma students would be implementing the algorithm for triangulation. Students in the diploma program usually write code for only one device and not multiple devices communicating together.

The real problem that is not trivial to solve is to process all the data together and displaying it in a map of the building it operates in. Mostly the customizability of the server software to allow a user to place beacons markers, map out the building, adding devices to track, and notifications if a tracker is detected by specific beacons.

The areas of specialty knowledge that I would need to solve the problem would be to design a server interface that communicates with multiple beacons and can be adjusted whenever the layout of the building changes, more beacons are added, or more trackers are added.

The knowledge I am lacking to research to explore to solve this problem is location triangulation with RSSI, sniffing for specific crafted Bluetooth packets, determining how to distinguish which beacon is closest to the tracker, UI interface for the server to map and operate the program.

10. Technical Challenges

This project is technically challenging because of the calculations required to perform the triangulation using RSI to find the location of the tracker. Additionally, I was never required to implement a GUI as most courses use simple text prompts or run from a config file. These are important because the triangulation is the cornerstone of this project and displaying the information in a GUI will easily prove that the tracker is functioning correctly.

11. Development Schedule and Milestones

Task	**Time Required(h)/(Total)**	**Start Period**
Phase 1: Setup of Devices		
Order more Raspberry PIs for beacons	2 (2)	Week 1
Download required libraries and dependencies	2 (4)	Week 1
Create databases	2 (6)	Week 1
Learn the libraries required for program	20 (26)	Week 1
Create basic tracking program for Android tracker	10 (36)	Week 2
Create basic beacon program for Raspberry PI	10 (46)	Week 2
Create basic server program for Windows	10 (56)	Week 2
Test basic program	5 (61)	Week 2
Phase 2: Refining Basic Programs		
Improve user interface and program for Android tracker	10 (71)	Week 3
Improve program of Raspberry Pi beacon	10 (81)	Week 3
Implement user interface for Windows server	10 (91)	Week 3
Add database to Windows server	5 (96)	Week 4
Learn database calls required to get the information required	10 (106)	Week 4
Phase 3: Implement Interface Tracking Feature		
Implement user interface option to upload PNG map to read	5 (111)	Week 5
Implement user interface to add beacons to map	10 (121)	Week 5
Implement user interface to assign beacon names	5 (126)	Week 5
Implement user interface option to select a tracker to follow	5 (131)	Week 5
Implement user interface option to show log to a tracker	10 (141)	Week 6
Implement database call to retrieve required information	20 (161)	Week 6
Test base tracking feature	10 (171)	Week 6
Phase 4: Implement Interface to Show Approximate Location		
Implement user interface to show approximate location	10(181)	Week 7
Implement database call to retrieve required information	10(191)	Week 7
Implement scrolling through tracker log timeline	20(211)	Week 7
Implement display of approximate location by lightning up beacon on map	10(221)	Week 8
Test approximate location feature	10(231)	Week 8
Phase 5: Implement alert system		
Implement user interface to enable and set alerts	20(251)	Week 9

Implement backend loop to check logs that a tracker has been detected by a specified area	30(281)	Week 10
Implement function to send alert to the server operator and tracker	20(301)	Week 11
Test alert system feature	10(311)	Week 12
Phase 6: Wrap up and report		
Gather data and write report analysis	20(331)	Week 13
Gather data and write documentation	20(351)	Week 14
Extra time for report changes	TBD	Week 15
Total Time	**351**	

12. Deliverables

Deliverables:

- Program for Bluetooth tracking for Android
- Program for beacons to detect the Bluetooth tracker
- Program for server to process data from the beacons
- Documentation on how to use the Bluetooth beacon tracking program
- A report detailing the project development process, research results and analysis

13. Conclusion and Expertise Development

This project will further my experience in my specialization by using Bluetooth communications. This is because I only learned surface knowledge of Bluetooth in the courses. I will further my experience about graphical user interfaces because most programs made in the program are command line based. Databases are another concept I will be learning about from this project as I have don't have much experience with it.

The experience gained from being able to develop a program using Bluetooth will be a good experience because I will be able to understand the limitations of Bluetooth. Bluetooth is constantly used in everyday electronics and by developing a product I learn the ins and outs of the communication such range, obstructions, and responding to the same tracking device. Facing these difficulties, it will give me the experience to work around them in future projects and improve.

14. References

Surian, D., Kim, V., Menon, R., Dunn, A. G., Sintchenko, V., & Coiera, E. (2019, September 9). *Tracking a moving user in indoor environments using bluetooth low energy beacons*. Journal of Biomedical Informatics. Retrieved November 2, 2022, from https://www.sciencedirect.com/science/article/pii/S1532046419302072

Mansell, G., & Curran, K. (n.d.). *Location Aware Tracking with beacons - UAH*. UAH. Retrieved November 2, 2022, from https://ipin2016.web.uah.es/usb/app/descargas/176_WIP.pdf

Sakphrom, S., Suwannarat, K., Haiges, R., & Funsian, K. (2021, September 25). *A simplified and high accuracy algorithm of RSSI-based localization zoning for children tracking in-out the school buses using Bluetooth Low Energy Beacon*. MDPI. Retrieved November 7, 2022, from https://www.mdpi.com/2227-9709/8/4/65/htm

Almaula, V., & Cheng, D. (n.d.). *Bluetooth triangulator - university of California, San Diego*. UC San Diego. Retrieved November 21, 2022, from https://cseweb.ucsd.edu/classes/fa06/cse237a/finalproj/almula.pdf

Inpixon. (n.d.). *Ble (bluetooth) Location Tracking & Positioning*. Inpixon. Retrieved November 21, 2022, from https://www.inpixon.com/technology/standards/bluetooth-low-energy

Wang, Y., Liu, Y., Yang, X., Cuthbert, L., & Zhao, Y. (2018, May 31). *Bluetooth positioning using RSSI and triangulation methods - researchgate*. ResearchGate. Retrieved November 21, 2022, from https://www.researchgate.net/publication/261056426_Bluetooth_positioning_using_RSSI_and_triangulation_methods

15. Change Log

November 2, 2022: Version 1.0

November 21, 2022: Version 2.0

- Added mention use of triangulation, and BLE/TCP protocol usage (Page 3, Section 3)
- Added mention use of triangulation, and clarifying floor map scope (Page 3, Section 4)
- Added software architecture diagram (Page 5, Section 7)
- Added pseudocode (Page 6, Section 7)
- Added application diagram (Page 7, Section 7)
- Added mention use of triangulation (Page 8, Section 9)
- Added mention use of triangulation (Page 9, Section 10)
- Added more references (Page 10, 11, Section 14)

5.3. Project Supervisor Approvals

[Include written approvals from the project supervisor indicating that they've approved both the proposal and report. Also include any changes that have been approved by the project supervisor. Please note that without written approvals from the project supervisor, the committee may not review the final report.]

6. References

Sakphrom, S., Suwannarat, K., Haiges, R., & Funsian, K. (2021, September 25). *A simplified and high accuracy algorithm of RSSI-based localization zoning for children tracking in-out the school buses using Bluetooth Low Energy Beacon*. MDPI. Retrieved November 7, 2022, from https://www.mdpi.com/2227-9709/8/4/65/htm

Mansell, G., & Curran, K. (n.d.). *Location Aware Tracking with beacons - UAH*. UAH. Retrieved November 2, 2022, from https://ipin2016.web.uah.es/usb/app/descargas/176_WIP.pdf

Sakphrom, S., Suwannarat, K., Haiges, R., & Funsian, K. (2021, September 25). *A simplified and high accuracy algorithm of RSSI-based localization zoning for children tracking in-out the school buses using Bluetooth Low Energy Beacon*. MDPI. Retrieved November 7, 2022, from https://www.mdpi.com/2227-9709/8/4/65/htm

Almaula, V., & Cheng, D. (n.d.). *Bluetooth triangulator - university of California, San Diego*. UC San Diego. Retrieved November 21, 2022, from https://cseweb.ucsd.edu/classes/fa06/cse237a/finalproj/almula.pdf

Inpixon. (n.d.). *Ble (bluetooth) Location Tracking & Positioning*. Inpixon. Retrieved November 21, 2022, from https://www.inpixon.com/technology/standards/bluetooth-low-energy

Wang, Y., Liu, Y., Yang, X., Cuthbert, L., & Zhao, Y. (2018, May 31). *Bluetooth positioning using RSSI and triangulation methods - researchgate*. ResearchGate. Retrieved November 21, 2022, from https://www.researchgate.net/publication/261056426_Bluetooth_positioning_using_RSSI_and_triangulation_methods

N. Pakanon, M. Chamchoy and P. Supanakoon, "Study on Accuracy of Trilateration Method for Indoor Positioning with BLE Beacons," *2020 6th International Conference on Engineering, Applied Sciences and Technology (ICEAST)*, Chiang Mai, Thailand, 2020, pp. 1-4, doi: 10.1109/ICEAST50382.2020.9165464.

7. Change Log

April 17, 2023: Version 1.0

- Filled in cover page
- Filled in sections 1 to 2.4

April 18, 2023: Version 1.1

- Filled in sections 2.8 to 2.12

April 19, 2023: Version 1.2

- Filled in sections 2.5 and 2.7

April 22, 2023: Version 1.3

- Filled in section 2.6

April 23, 2023: Version 1.4

- Added more software architecture diagrams to 2.5.1 System/Software Architecture Diagram
- Filled in section 5 References
- Filled in section 3 Conclusion
- Added section 4.1 Approved Proposal

April 27, 2023: Version 2.0

- Supervisor Aman Abdulla revision remarks
    - Added RSSI units of dBm to all mentions of RSSI
    - Added examples of pseudocode and finite state machines for RSSI smoothing algorithm and elaborated under section 2.11 Future Enhancements