4th year project
Automated Drone Air Traffic Control System

Final Report

Institute of Technology Carlow

Supervisor: Dr. Oisin Cawley
Author: James Hall
Submission Date: 26 April 2021

Abstract

This document is the final report on the development of the Automated Drone Air Traffic Control system. It will outline the details of the project, including the technologies used and the goals achieved, as well the learning outcome of the project.

Table of Contents

Table of figures

1. Introduction

This document will detail the process of creating the Automated Drone Air Traffic Control system. It will explain the scope of the project, and the technologies used to create it. I will also explain the difficulties faced during the development of the application. The goals achieved and missed as well as what had to change in the design in order to realise the current application.

2. Project Description

The purpose of this project is to create an application that is capable of automatically controlling a drone or fleet of drones without the need for constant human input. The application should be able to have drones connected to the system and their details stored. It should be capable of storing the geo-locations of a number of real life locations that would act as a transport network for the numerous drones to travel to and from. The system should also be able to allow the user to create and begin automated flights and for the details of these flights to be stored on the system.

The combination of these functionalities should allow for the user to begin a flight with touch of a button and for the drone chosen for the flight to automatically take off from its origin, travel to and land at its destination safely. This requires the inclusion of collision detection mechanisms to ensure that drones do not collide with one another in flight. It also requires the need to be able to cancel any flights in progress and have the drone return safely to their origin or the nearest safe landing zone.

The application should be able to display all of the required information to the user in the form of a GUI that would ideally show the real time locations of all drones on the system as well as all locations on the system.

2.1 Language

The language used for this project was Python, specifically Python 3.6.9. Python is a high level object oriented programming language that is considered to be beginner friendly. It has a large standard library and supports a wide range of functions and operations. As it is mostly an interpreted language, it is highly portable as its bytecode may run on any compatible virtual machine.

2.2 Libraries/APIs

2.2.1 Olympe

Parrot's Olympe SDK provides a Python based controller programming interface. It allows for the use of both real and simulated drone flight, with the simulation being visualised through their Sphinx software running on Gazebo, a 3d visualisation tool. Olympe is available on the Linux platform and has been tested by the developers on Ubuntu 18.04 and works on Debian 9 or higher. The software is written in the Python programming language and allows the control of Parrot drones through Python scripts. The integration of the Olympe SDK with the Sphinx virtualisation software allows potential developers to use a virtual version of a number of Parrot drone types without the need for a physical drone on hand.

2.2.2 Kivy

Kivy is an open source Python framework for developing mobile apps and other multitouch application software with a natural user interface (NUI). It is capable of being run on Android, iOS, Linux, macOS, and Windows. Of particular note for this project is Kivy's Garden MapView library which allows for the creation of map based widgets using OpenStreetMap data.

2.3 Operating System Linux Ubuntu

Linux is an open source and community-developed operating system for a wide range of device types like computers, servers, mainframes, mobile devices and embedded devices. It is supported on most major computer platforms such as x86 and ARM. It comes in a variety of types or "distributions" including Ubuntu, which was chosen for this project. Ubuntu is based on the Debian GNU/Linux distribution. Ubuntu 18.04 was used for the creation of this project due to its compatibility with Parrot Olympe SDK.

2.4 Hardware

The hardware used in the creation was the Drone, a Bebop 2 drone [Fig 1] created by Parrot Drones. This is a now deprecated drone model from parrot and as such its features are somewhat dated. It has a flight time of about 25 minutes and a range of approximately 300 meters which may be extended to roughly 500 meters with the use of the included "Sky controller". It's main advantage for this project is the ability to interface with the drone through Parrot's Olympe API, allowing for the drone to be controlled independently of the included controller.

Figure 1. Bepop 2 drone

2.5 The Application

The application consists of a single screen containing a map that is movable and has zoom functionality. Placed above the map are 3 sets of colour coded buttons relating to the main operations of the application. These allow for the addition of new drones, locations and flights to the system. They also allow the user to remove these objects from the system. and for the user to initiate flights, the progress of which may be viewed on the map itself with the drone markers activated. Below the details of each functional aspect of the application with be detailed.

2.5.1 The Main Screen

The entirety of the application can be controlled through the main map screen [Fig 2]. This contains all the necessary functionality to perform CRUD operations for Drones, Locations and Flights through the utilisation of popup menus.

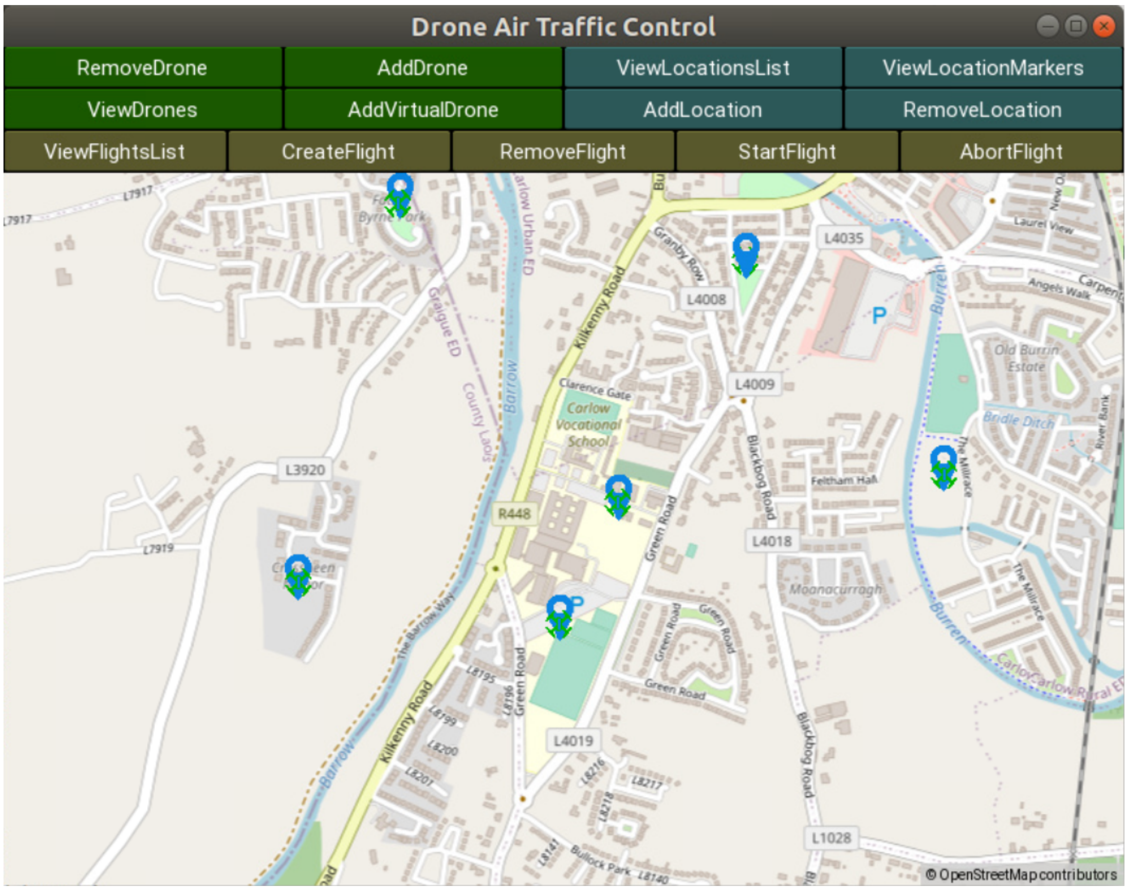
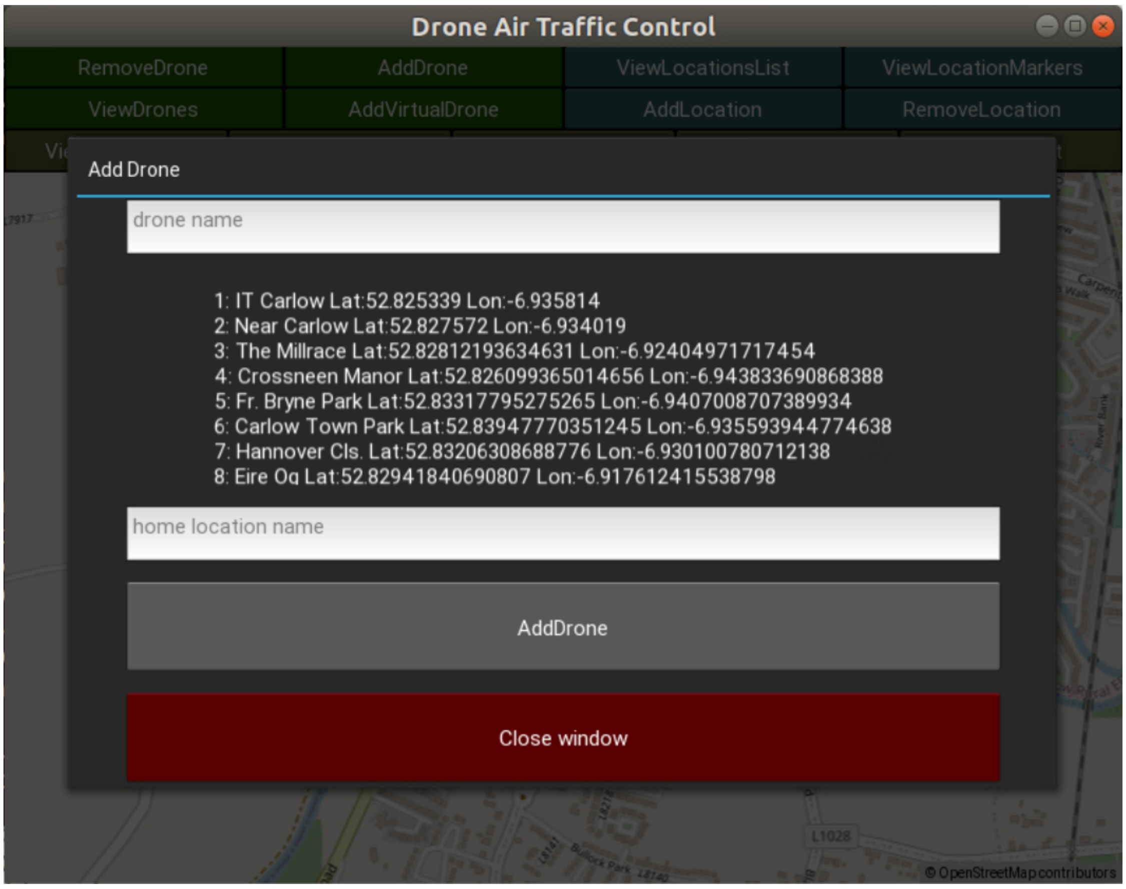


Figure 2. The main map screen

2.5.2 Add Virtual Drone

The Add Virtual Drone Popup [Fig 3] consists of a text input box to enter the drone's name and a scrollable list of stored locations from which the user may enter the corresponding number into the numerical input box below in order to choose the drone's home location. Upon clicking the "Add Drone" button the drone is stored on the system. As in all Kivy popups, the user may exit the popup with no action being taken by clicking outside the popup, they may also click the "Close Window" button.

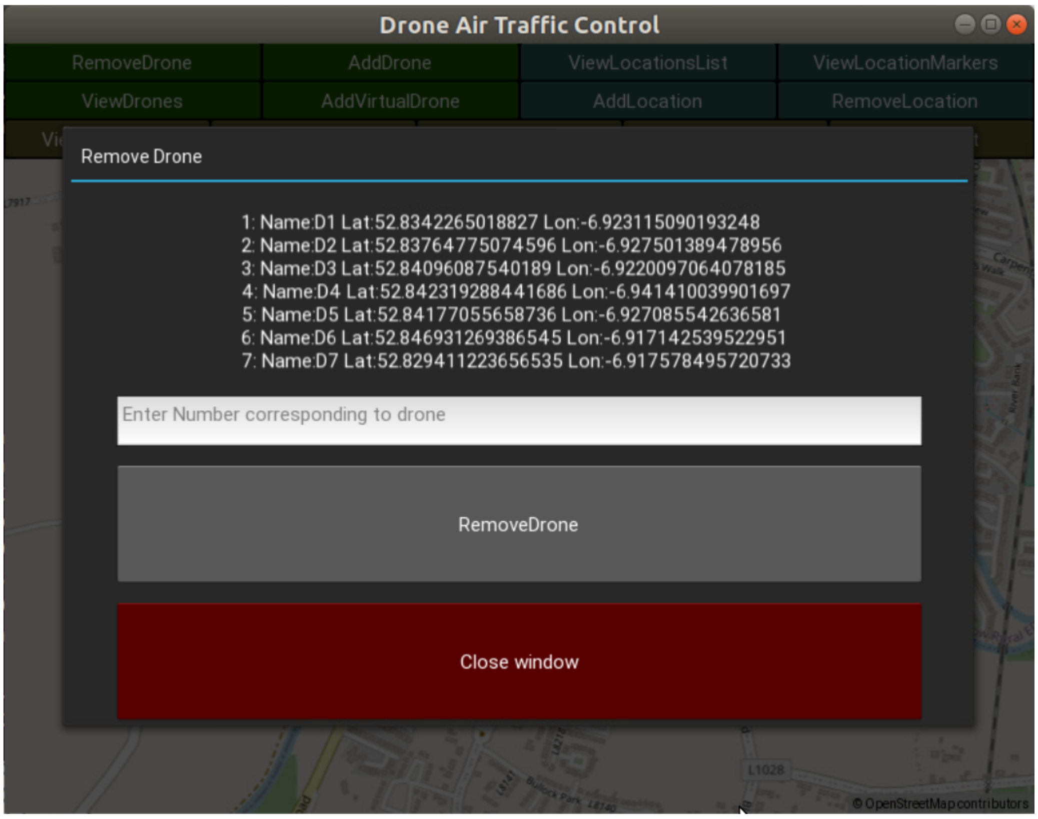


```
1: IT Carlow Lat:52.825339 Lon:-6.935814
2: Near Carlow Lat:52.827572 Lon:-6.934019
3: The Millrace Lat:52.82812193634631 Lon:-6.92404971717454
4: Crossneen Manor Lat:52.826099365014656 Lon:-6.943833690868388
5: Fr. Bryne Park Lat:52.83317795275265 Lon:-6.9407008707389934
6: Carlow Town Park Lat:52.83947770351245 Lon:-6.935593944774638
7: Hannover Cls. Lat:52.83206308688776 Lon:-6.930100780712138
8: Eire Og Lat:52.82941840690807 Lon:-6.917612415538798
```

Figure 3. The Add Virtual Drone Popup

2.5.3 Remove Drone Button

Upon clicking the “Remove Drone” button, the user is presented with a popup displaying a list of all drones on the system [Fig 4]. The user may enter the number corresponding to the drone they wish to remove from the system and then click the popup’s “Remove Drone” button to delete the drone from the system.



```
1: Name:D1 Lat:52.8342265018827 Lon:-6.923115090193248
2: Name:D2 Lat:52.83764775074596 Lon:-6.927501389478956
3: Name:D3 Lat:52.84096087540189 Lon:-6.9220097064078185
4: Name:D4 Lat:52.842319288441686 Lon:-6.941410039901697
5: Name:D5 Lat:52.84177055658736 Lon:-6.927085542636581
6: Name:D6 Lat:52.846931269386545 Lon:-6.917142539522951
7: Name:D7 Lat:52.829411223656535 Lon:-6.917578495720733
```

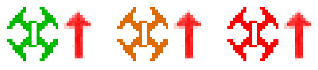
Figure 4. The Remove Drone Popup

2.5.4 View Drones

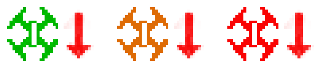
When the "View Drones" button is pressed, the system's drones are drawn on screen at their correct locations [Fig 6]. Clicking the button again will hide them [Fig 5].

There are several different images for the drones, each indicating a different state or battery level.

The drone will appear green when above 50% battery, orange when below, and red when there in <10% battery left.



When the drone is increasing in height, it will display an up arrow beside it



When the drone is decreasing in height, it will display an down arrow beside it

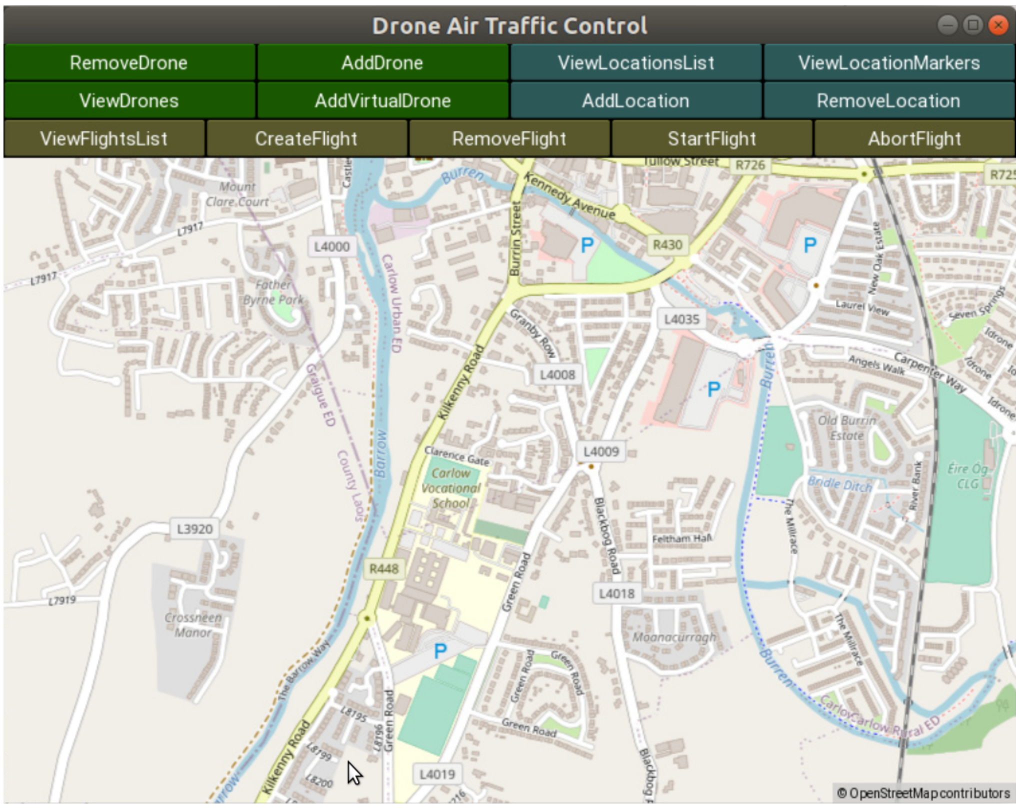


Figure 5. The Map with no drones drawn on screen

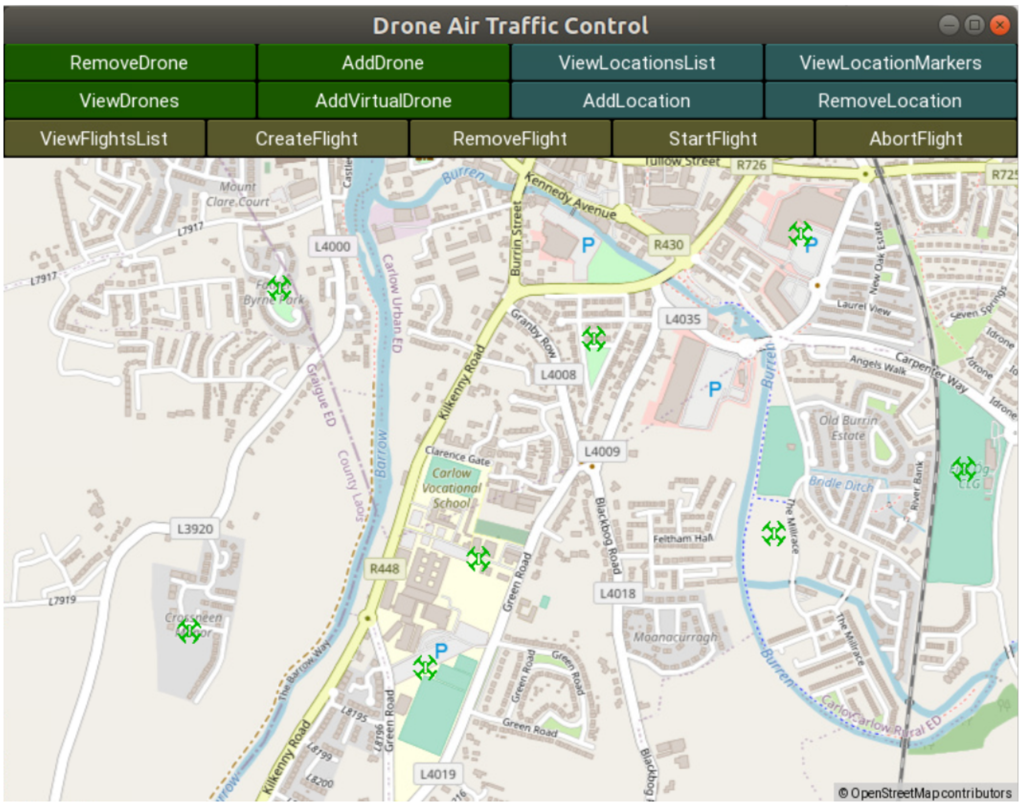


Figure 6. The map once the view drones button is pressed

2.5.5 Add Location

When the user clicks the “Add Location” button, they are presented with a popup [Fig 7] containing a text input where they can name their location. When they click the “Add Location Name” button the popup will close and they may then click on the map and the new location will be added at the coordinates corresponding to the location where they click. They can click the “Close Window” button to exit the window without creating a new location.

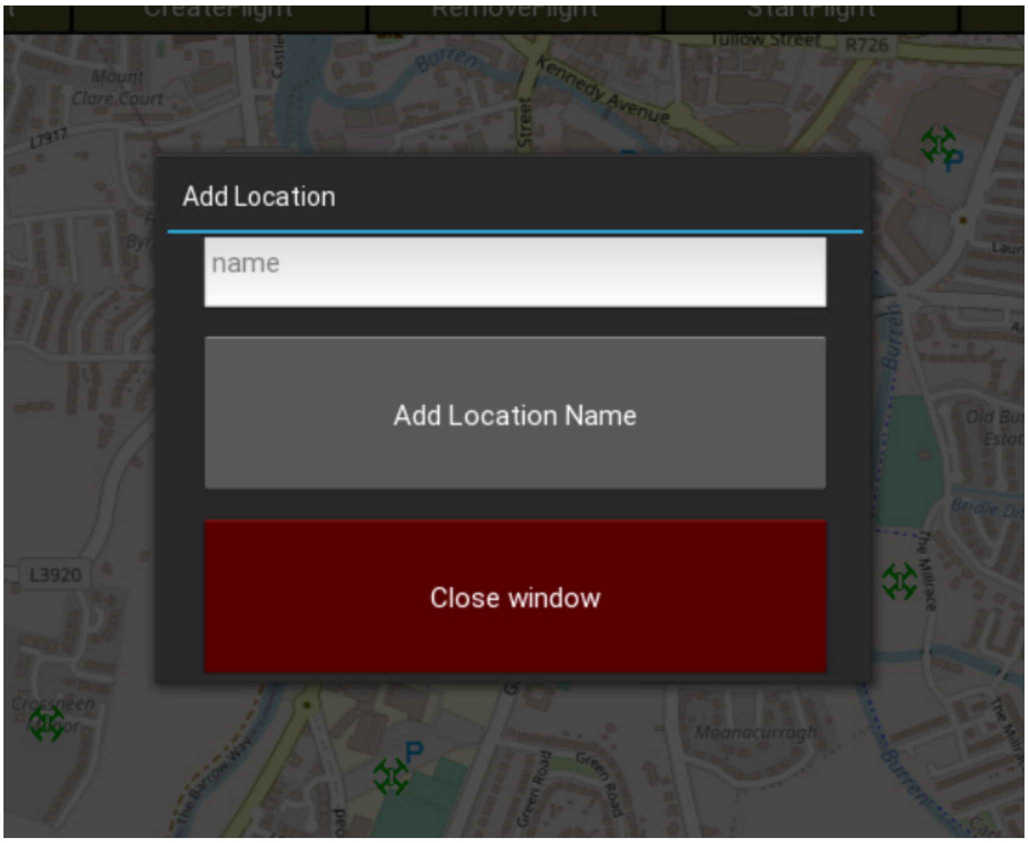
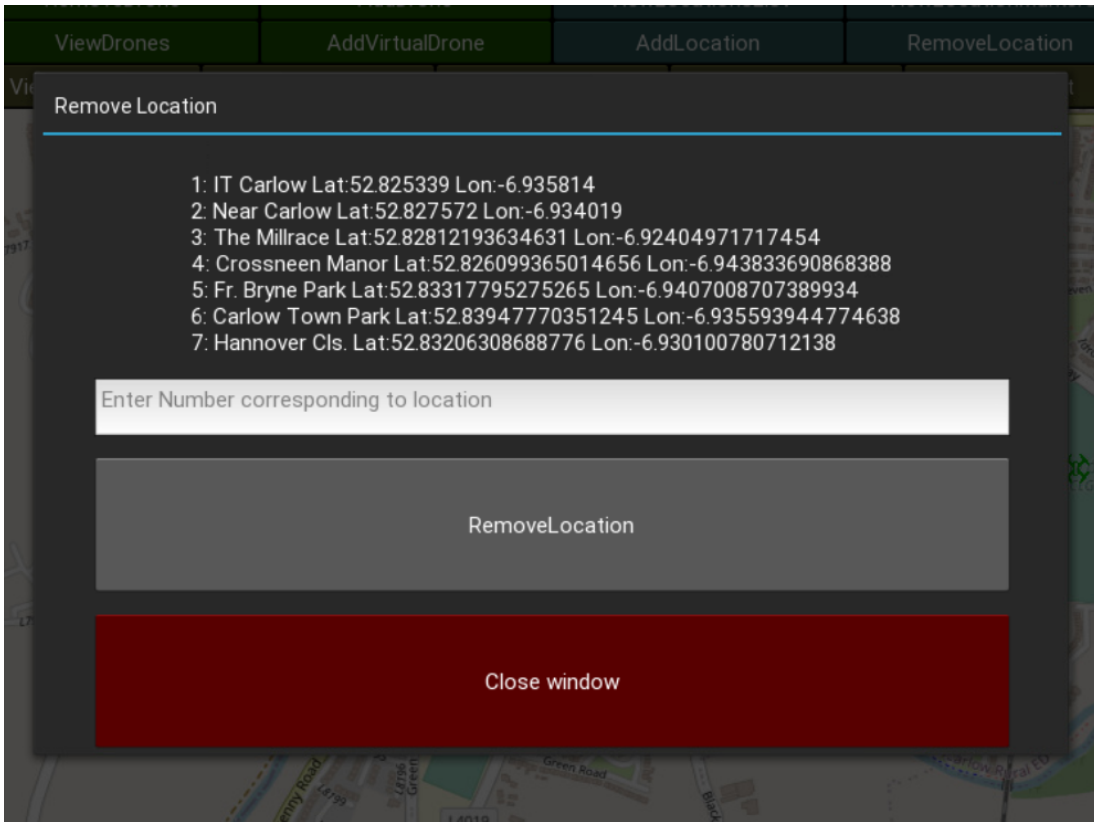


Figure 7. Add Location Popup

2.5.6 Remove Location

Upon clicking the “Remove Location” button the user will be presented with a popup displaying a list of all locations on the system [Fig 8]. They may enter a number corresponding to the location they intend to remove and when the “Remove Location” button is pressed, the location will be deleted from the system. The user may exit the popup without any action being taken by pressing the “Close Window” button.

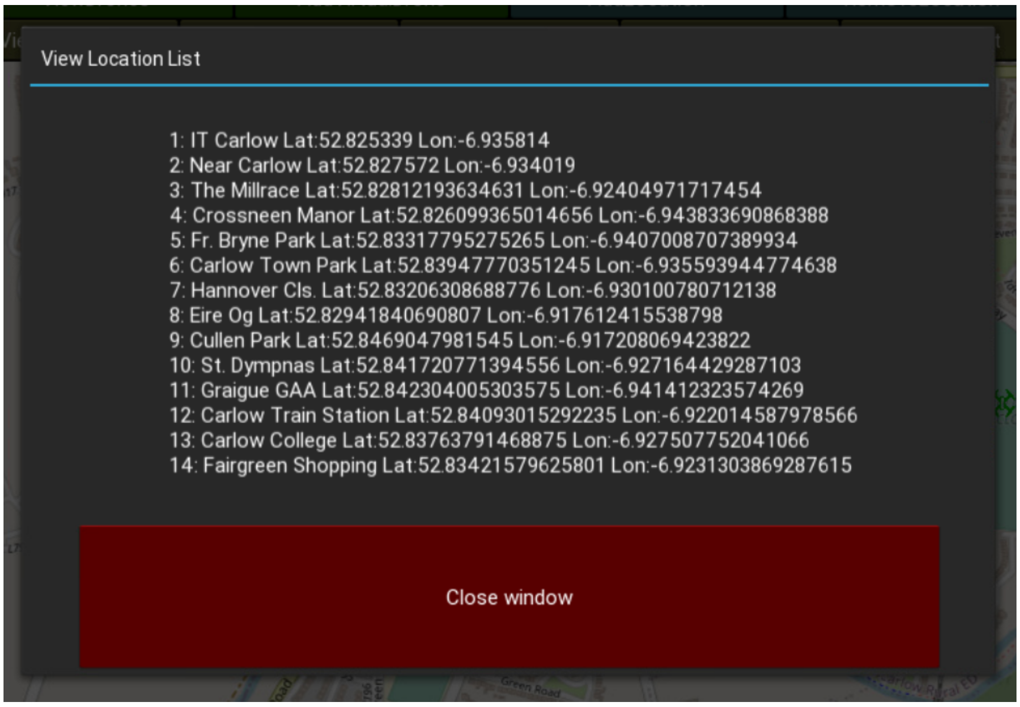


1: IT Carlow Lat:52.825339 Lon:-6.935814
2: Near Carlow Lat:52.827572 Lon:-6.934019
3: The Millrace Lat:52.82812193634631 Lon:-6.92404971717454
4: Crossneen Manor Lat:52.826099365014656 Lon:-6.943833690868388
5: Fr. Bryne Park Lat:52.83317795275265 Lon:-6.9407008707389934
6: Carlow Town Park Lat:52.83947770351245 Lon:-6.935593944774638
7: Hannover Cls. Lat:52.83206308688776 Lon:-6.930100780712138

Figure 8. Remove Location Popup

2.5.7 View Location List

If the user wishes to see a list of all locations on the system, they can click the “View Locations List” button to be presented with a popup displaying a list of all locations [Fig 9]. The user may exit the popup by pressing the “Close Window” button.



1: IT Carlow Lat:52.825339 Lon:-6.935814
2: Near Carlow Lat:52.827572 Lon:-6.934019
3: The Millrace Lat:52.82812193634631 Lon:-6.92404971717454
4: Crossneen Manor Lat:52.826099365014656 Lon:-6.943833690868388
5: Fr. Bryne Park Lat:52.83317795275265 Lon:-6.9407008707389934
6: Carlow Town Park Lat:52.83947770351245 Lon:-6.935593944774638
7: Hannover Cls. Lat:52.83206308688776 Lon:-6.930100780712138
8: Eire Og Lat:52.82941840690807 Lon:-6.917612415538798
9: Cullen Park Lat:52.8469047981545 Lon:-6.917208069423822
10: St. Dympnas Lat:52.841720771394556 Lon:-6.927164429287103
11: Graigue GAA Lat:52.842304005303575 Lon:-6.941412323574269
12: Carlow Train Station Lat:52.84093015292235 Lon:-6.922014587978566
13: Carlow College Lat:52.83763791468875 Lon:-6.927507752041066
14: Fairgreen Shopping Lat:52.83421579625801 Lon:-6.9231303869287615

Figure 9. The View Location List Popup

2.5.8 View Location Markers

When the user clicks the “View Location Markers” button, all the locations will be drawn on the map at their correct coordinates [Fig 11}. Clicking the button again will hide the markers [Fig 10].

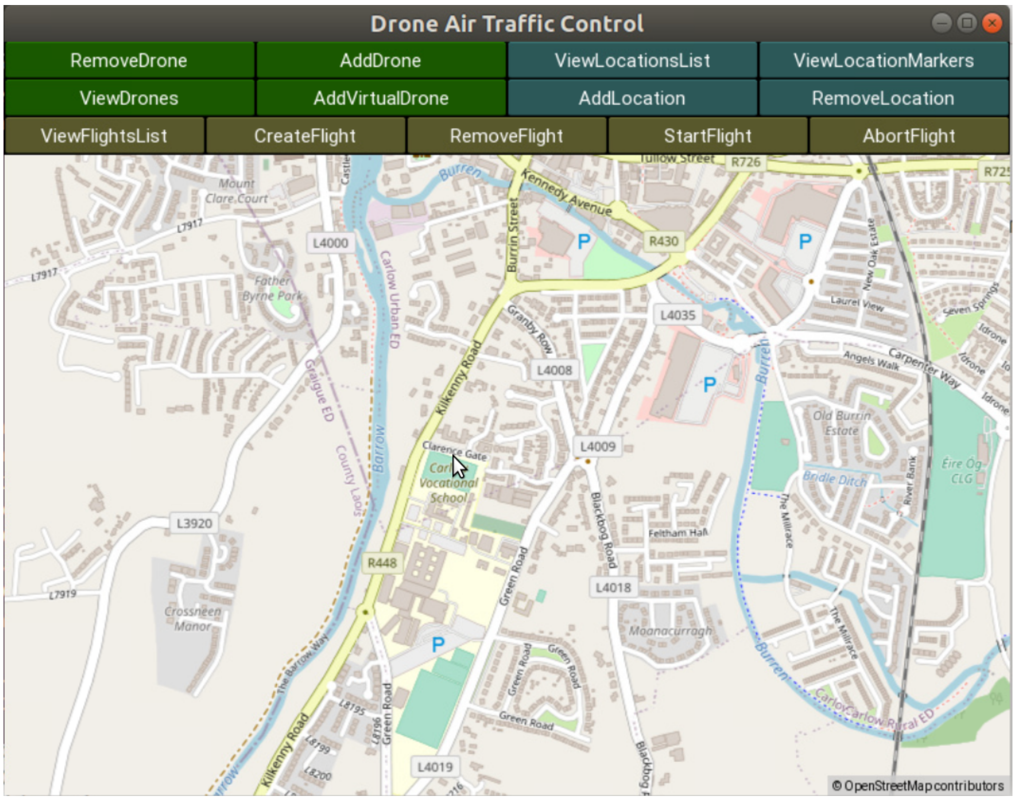


Figure 10. The map without Location Markers

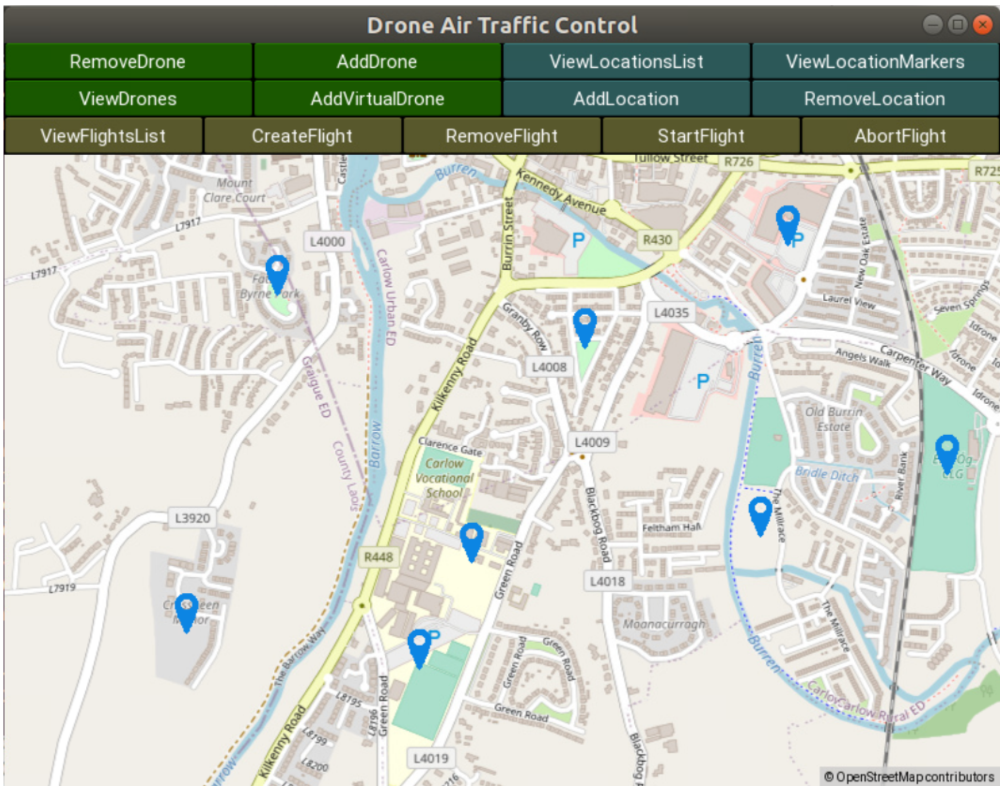
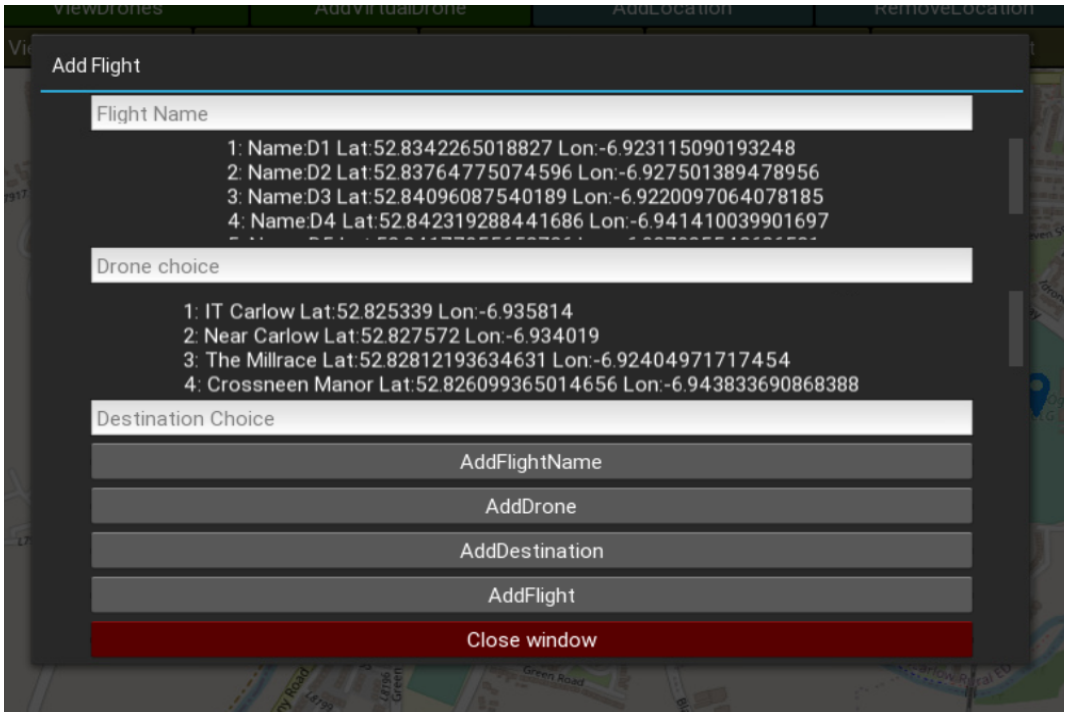


Figure 11. The map with Location Markers displayed

2.5.9 Create Flight

When a user wishes to add a new flight to the system, by pressing the "Create Flight" button they are presented with a popup displaying a list of all drones and locations on the system [Fig 12]. They can enter a name for the flight here as well a numerical value corresponding to the drone they wish to use for the flight and the location they wish for the drone to fly to. Upon clicking the "Add Flight" button, the flight is added to the system and if all fields are successfully filled in the popup closes.



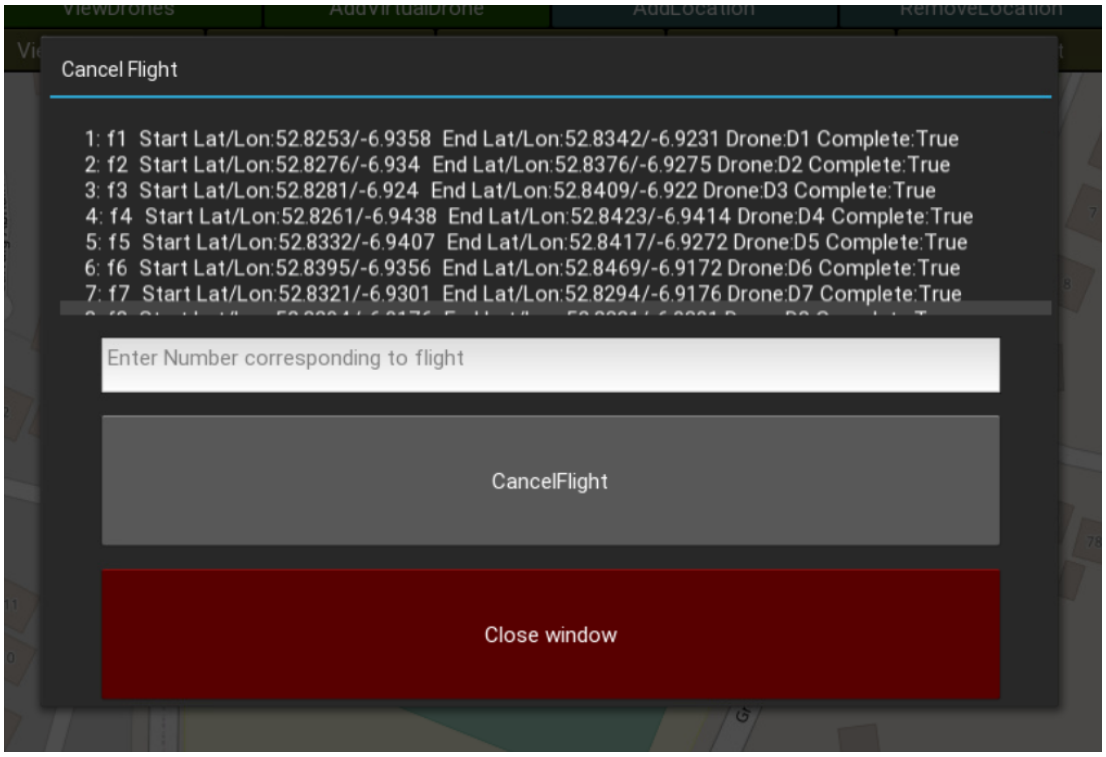
```
1: Name:D1 Lat:52.8342265018827 Lon:-6.923115090193248
2: Name:D2 Lat:52.83764775074596 Lon:-6.927501389478956
3: Name:D3 Lat:52.84096087540189 Lon:-6.9220097064078185
4: Name:D4 Lat:52.842319288441686 Lon:-6.941410039901697
```

```
1: IT Carlow Lat:52.825339 Lon:-6.935814
2: Near Carlow Lat:52.827572 Lon:-6.934019
3: The Millrace Lat:52.82812193634631 Lon:-6.92404971717454
4: Crossneen Manor Lat:52.826099365014656 Lon:-6.943833690868388
```

Figure 12. The Create Flight Popup

2.5.10 Remove Flight

When the user wishes to remove a flight from the system, they click the “Remove Flight” button and are presented with a popup showing them a list of all flights on the system [Fig 13]. They may enter the number corresponding to the flight they wish to remove and upon hitting the “Cancel Flight” button the flight is removed from the system. The user may exit the popup without any action being taken by pressing the “Close Window” button.

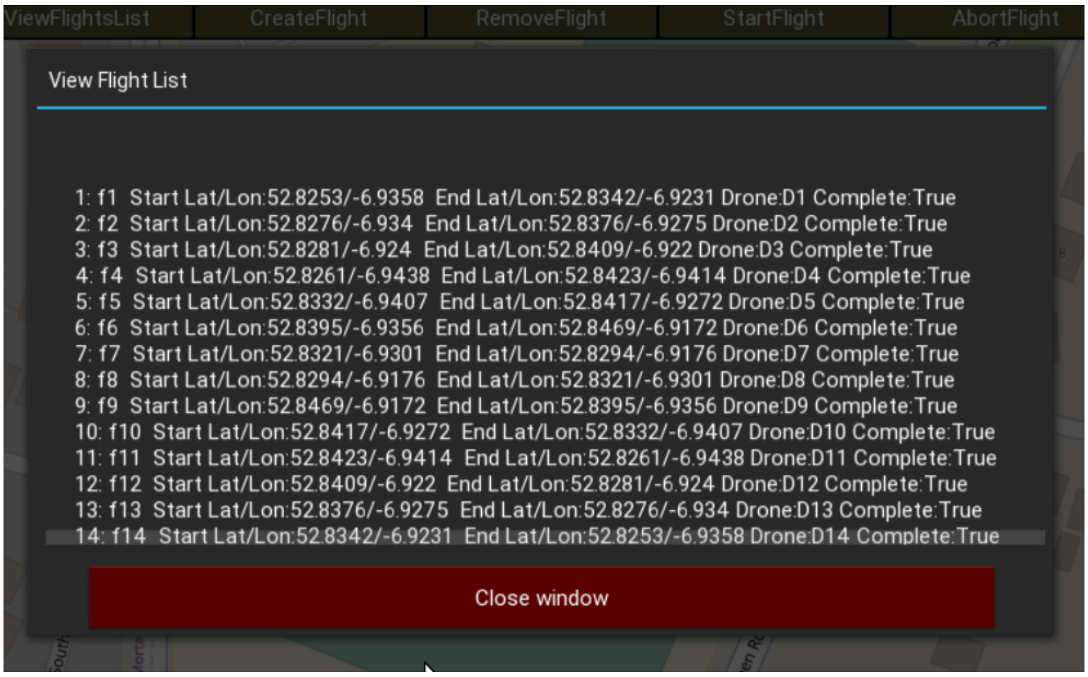


```
1: f1  Start Lat/Lon:52.8253/-6.9358  End Lat/Lon:52.8342/-6.9231 Drone:D1 Complete:True
2: f2  Start Lat/Lon:52.8276/-6.934  End Lat/Lon:52.8376/-6.9275 Drone:D2 Complete:True
3: f3  Start Lat/Lon:52.8281/-6.924  End Lat/Lon:52.8409/-6.922 Drone:D3 Complete:True
4: f4  Start Lat/Lon:52.8261/-6.9438  End Lat/Lon:52.8423/-6.9414 Drone:D4 Complete:True
5: f5  Start Lat/Lon:52.8332/-6.9407  End Lat/Lon:52.8417/-6.9272 Drone:D5 Complete:True
6: f6  Start Lat/Lon:52.8395/-6.9356  End Lat/Lon:52.8469/-6.9172 Drone:D6 Complete:True
7: f7  Start Lat/Lon:52.8321/-6.9301  End Lat/Lon:52.8294/-6.9176 Drone:D7 Complete:True
```

Figure 13. The Cancel Flight Popup

2.5.11 View Flights List

The user may view a popup [Fig 14] displaying a list of all flights on the system including their chosen drone, start and end location and whether or not the flight is complete or not. This is done by pressing the “View Flight List” button. They may exit the list by clicking the “Close Window” button.

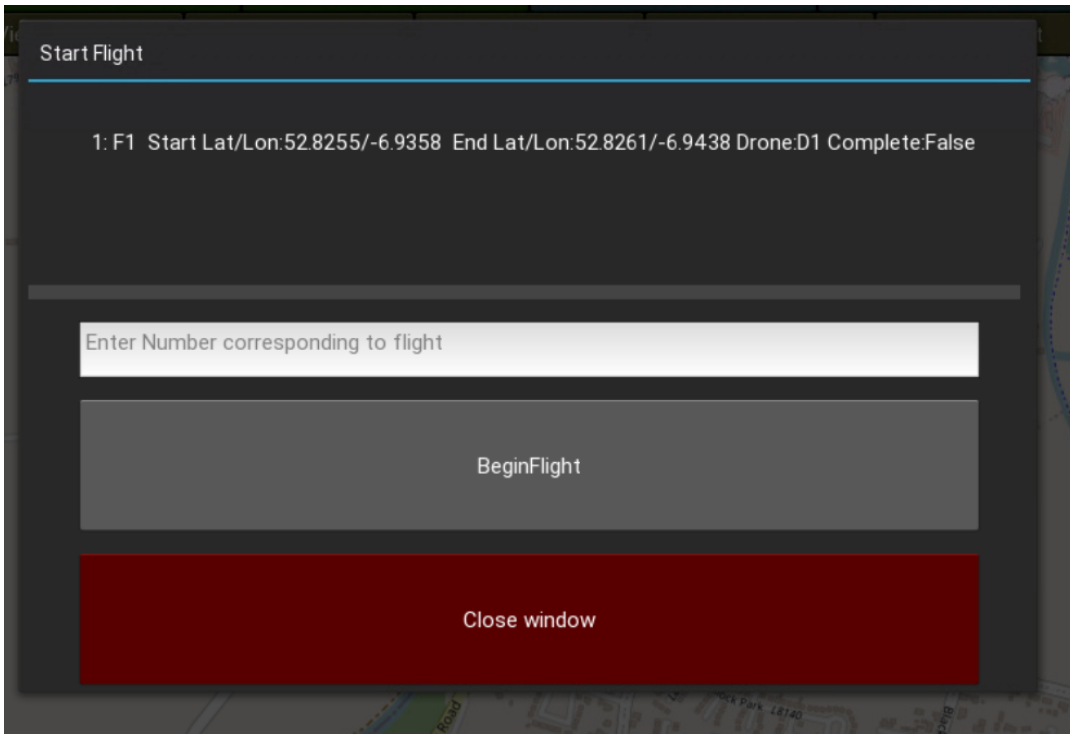


```
1: f1  Start Lat/Lon:52.8253/-6.9358  End Lat/Lon:52.8342/-6.9231 Drone:D1 Complete:True
2: f2  Start Lat/Lon:52.8276/-6.934  End Lat/Lon:52.8376/-6.9275 Drone:D2 Complete:True
3: f3  Start Lat/Lon:52.8281/-6.924  End Lat/Lon:52.8409/-6.922 Drone:D3 Complete:True
4: f4  Start Lat/Lon:52.8261/-6.9438  End Lat/Lon:52.8423/-6.9414 Drone:D4 Complete:True
5: f5  Start Lat/Lon:52.8332/-6.9407  End Lat/Lon:52.8417/-6.9272 Drone:D5 Complete:True
6: f6  Start Lat/Lon:52.8395/-6.9356  End Lat/Lon:52.8469/-6.9172 Drone:D6 Complete:True
7: f7  Start Lat/Lon:52.8321/-6.9301  End Lat/Lon:52.8294/-6.9176 Drone:D7 Complete:True
8: f8  Start Lat/Lon:52.8294/-6.9176  End Lat/Lon:52.8321/-6.9301 Drone:D8 Complete:True
9: f9  Start Lat/Lon:52.8469/-6.9172  End Lat/Lon:52.8395/-6.9356 Drone:D9 Complete:True
10: f10  Start Lat/Lon:52.8417/-6.9272  End Lat/Lon:52.8332/-6.9407 Drone:D10 Complete:True
11: f11  Start Lat/Lon:52.8423/-6.9414  End Lat/Lon:52.8261/-6.9438 Drone:D11 Complete:True
12: f12  Start Lat/Lon:52.8409/-6.922  End Lat/Lon:52.8281/-6.924 Drone:D12 Complete:True
13: f13  Start Lat/Lon:52.8376/-6.9275  End Lat/Lon:52.8276/-6.934 Drone:D13 Complete:True
14: f14  Start Lat/Lon:52.8342/-6.9231  End Lat/Lon:52.8253/-6.9358 Drone:D14 Complete:True
```

Figure 14. The View Flight List Popup

2.5.12 Start Flight

When the user wishes to begin a flight they may press the “Start Flight” button. They will be presented with a popup displaying a list of all available flights on the system [Fig 15]. By entering into the numerical input field the number corresponding to the flight they wish to begin and clicking the “BeginFlight" button, the flight will be started. They may then return to the map view to watch any flights in progress on the map [Fig 16].


Start Flight

1: F1 Start Lat/Lon:52.8255/-6.9358 End Lat/Lon:52.8261/-6.9438 Drone:D1 Complete:False

Enter Number corresponding to flight

Figure 15. The Start Flight Popup

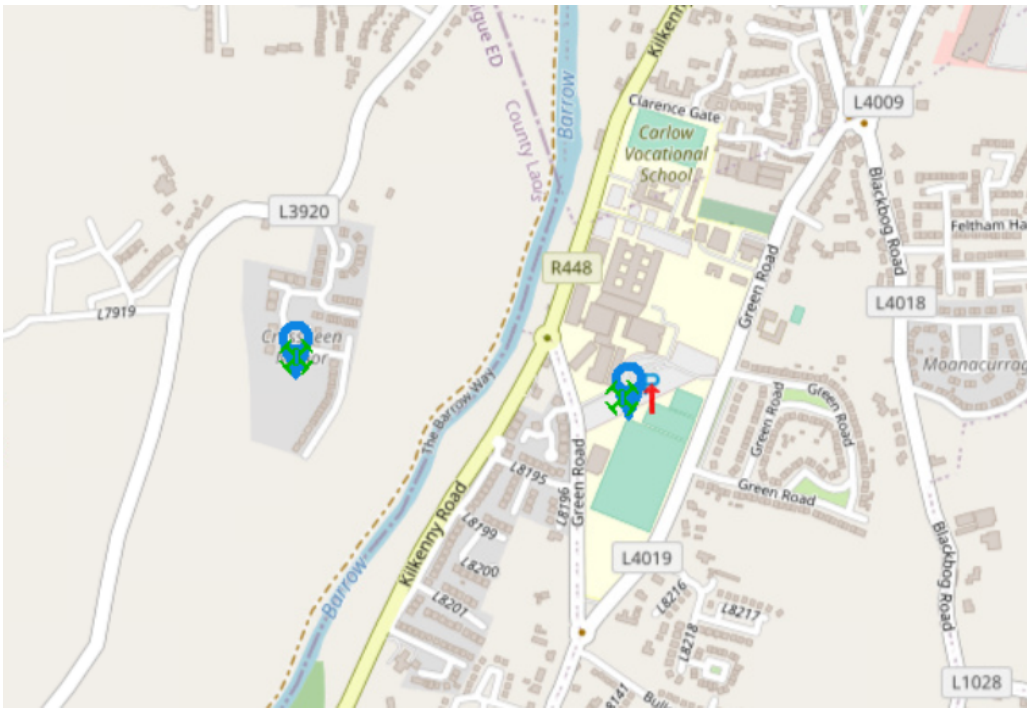
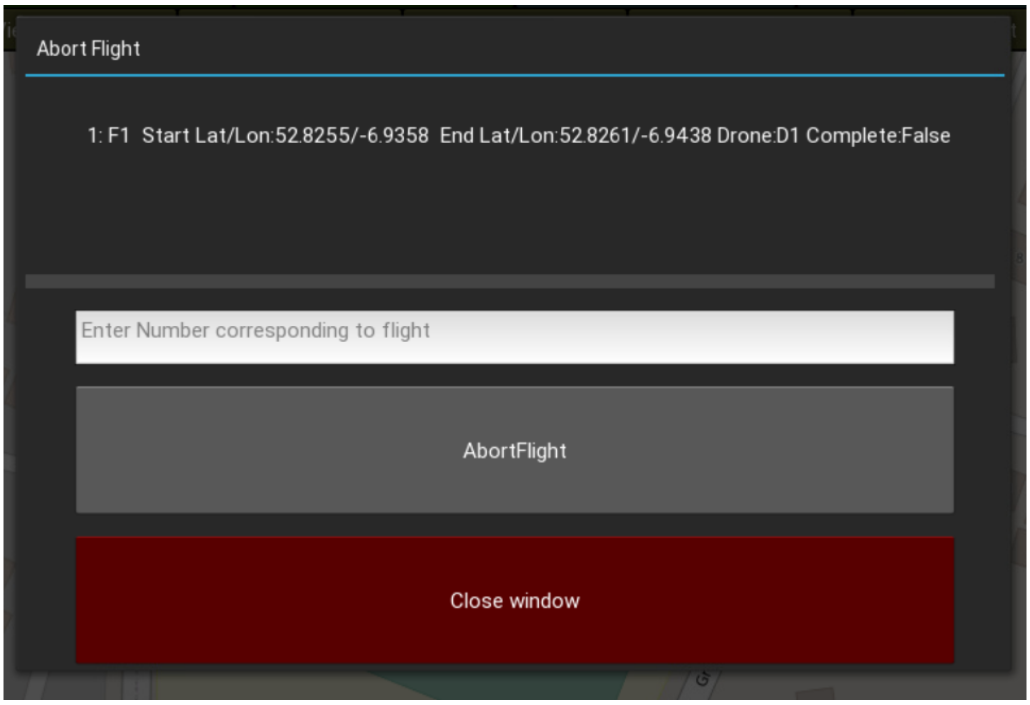


Figure 16. The drone on the right is beginning its ascent at the start of its flight

2.5.13 Abort Flight

If for some reason the user needs to quickly abort a flight, if the battery is running low or the weather suddenly becomes dangerous to drone flight. The user may click the “Abort Flight” button and be presented with the abort flight popup [Fig 17]. This will display a list of flights to the user and they may enter the number corresponding to the flight in the numeric input box and click the “Abort Flight” button. This will cause the drone to change course if necessary and move to the nearest safe landing location on the system and land there. The user may leave this popup without action by pressing the “Close Window” button.



1: F1 Start Lat/Lon:52.8255/-6.9358 End Lat/Lon:52.8261/-6.9438 Drone:D1 Complete:False

Figure 17. The Abort Flight Popup

3. Analysis of Project Achievements

During the course of this project, the majority of the desired deliverables were achieved. A functional GUI application was created using a new GUI creation tool in Python. The system is able to accept CRUD operations on Drones, Locations, and Flight plans. There is a system in place to detect when drones are in close proximity to one another, and for them to take measures to avoid collisions. The system is capable of handling multiple flights simultaneously and for all drones in flight to be correctly guided to their required destinations. As well as this a script allowing for connection to a real drone was tested in a basic script.

3.1 Proof of Concept Test

The purpose of this test was to ascertain whether it was feasible to gain control of a real drone using the application without relying on the Olympe shell.

During the course of this test a connection to the Bebop 2 drone was established and a short flight was carried out, involving the drone taking off moving forward, changing direction moving further and then landing. This served as a proof of concept that the drone could indeed be controlled through the API. This flight was carried out from the standard python virtual environment and did not require the controlling script to be run from with the Olympe shell. This was an important milestone as it further served as a proof of concept that the system would be able to control drones independently of the Parrot ecosystem.

3.2 Creation of Graphical User Interface Application

As detailed in section 2, Kivy user interface tool was chosen for the development of the user interface for this project.The task of creating a functioning graphical user interface using Kivy and its Mapview widget proved to be one of the more difficult and time consuming tasks. While at first glance, the Mapview tool appeared to be the ideal tool for the job, as it allowed for easy map integration and simple marker placement, it's functionality was incredibly limited beyond this.

The ability to add and remove markers dynamically had to be created. Also the ability to add markers to the map and their geo locations to the system by simply clicking on the map, took up a large amount of time and involved in the end extending Kivy's on_touch functionality.

3.3 Drone and Flight Management Functionality

The majority of drone control algorithms from the initial design specifications for the project were achieved. CRUD operations for any number of drones, locations, and flights are present on the system. As a proof of concept Python's Pickle was used to provide persistent data and the information was stored in a pickled format within the project files.

The system is capable of managing the flight of a drone or drones from their starting point to their intended destination, and is capable or relaying that destination to the user in the form of a list showing drone positions and also visually via the map, where each drone geo coordinates are translated to the correct pixel locations on the MapView. There is a collision detection system in place whereby when two or more drones approach within a certain distance of one another, they execute an avoidance algorithm where the one with the lowest battery takes precedence, followed by a simple check of which drone is further North and then further West of the other.

4. Issues Encountered

While a majority of the required functionalities were achieved, there were some that could not be completed due to various issues such as time constraints, choice of technology and the current situation with Covid-19. These range from GUI based issues to integration with the Bebop 2 drone.

4.1 User interface clarity and functionality

As Kivy's Mapview widget is quite basic in its implementation, it provides no inbuilt functionality for displaying text on the map itself or next to markers. This made it incredibly difficult to display the names and details of the locations and drones displayed on the map. Ultimately this was not achieved in the project to date. However some information about the various drones was able to be displayed to the user visually by the use of different drone images depending on whether the drone is ascending/descending or when the battery is running low. The author's limited previous experience with creation of GUI applications also led to some aspects of the UI not being as user friendly as they should be. The scrollable lists while functional, do not always work as intended and suffer from some formatting issues. Overall the look of the application could do with some improvement.

4.2 Flight functionality

Although the majority of functionality for the desired flight algorithms were achieved, the ability to change the destination of a drone to any desired one mid flight was not implemented. The ability to simply pause a flight and have the drone hover was also not implemented and while not a strict requirement would have been a useful addition. The reasoning behind these missed goals were simple time constraints on the project.

4.3 Account creation

The ability to perform CRUD operations for a user account was not implemented in the final project due to time limitations. This would have been ideal to have implemented in the final project to ensure the security of the system. Though is was included in the original design specification for the project

4.4 Bebop 2 integration

Full integration of the Bebop 2 drone was not realised on this project. While a proof of concept drone flight was achieved using a basic script that imported the project's BebopDrone.py module, the ability to fly the drone outside was hampered by restrictions due to Covid-19. As the ability to test the drone outdoors was limited, it was decided to concentrate on the virtual aspects of the system to ensure the collision detection and flight algorithms were completed.

4.5 Unit testing

Due to time constraints, unit testing was not able to be achieved for this project. A set of unit tests to check the stability and functionality of the codebase would have been desirable for a finished product. This however was not possible as other deliverables needed to be prioritised in order to get maximum functionality in the required timeframe.

4.6 Project management issues

4.6.1 Covid-19

The outbreak of Covid-19 and the intermittent usage of lockdowns in Ireland to combat its spread caused severe issues with the ability to test the drone in an outdoor environment. As it is illegal to fly a drone in Ireland without the proper drone pilot license and all drones must be under direct control of a human operator at all times. It was decided that the virtual aspects of the project and the generic functionality should be concentrated on, as the author did not want to risk breaking both covid restrictions and drone regulations at the same time, and a suitably remote location to test the system could not be reach within the 5km travel restrictions in place for much of the project's duration.

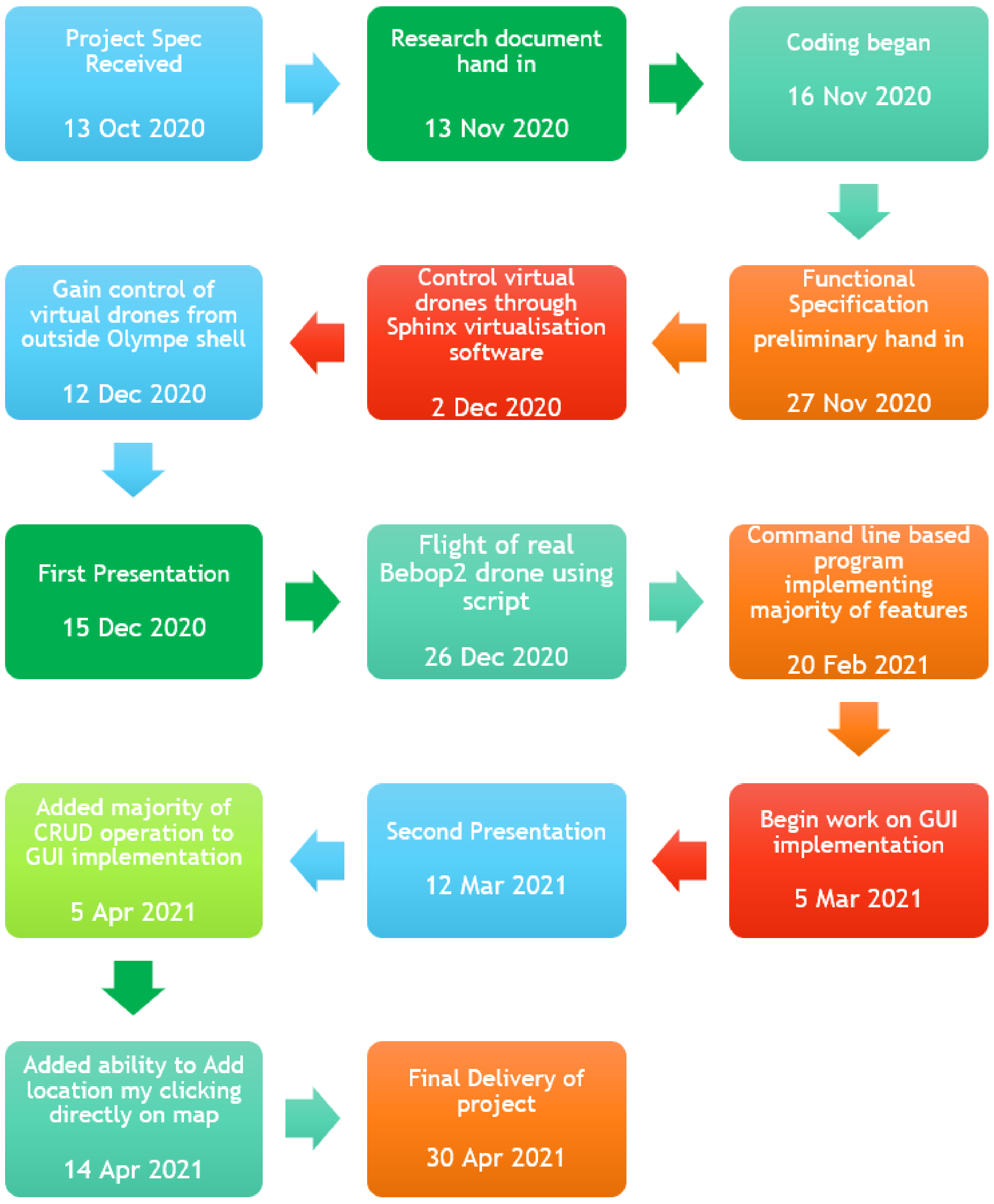
4.6.2 Kivy MapView

While Kivy's MapView widget at first seemed like the perfect solution to map creation for the project, it's implementation and functionality turned out to be very basic. Though the map window itself can be created with only a few lines of code and markers may be added to the map with similar ease, the functionality of the addon ends there. The online documentation is sparse and as such creative but time consuming solutions need to be worked out in order to utilise the addon.

An example of this is that although markers may be added to the map with ease, there is no easy way to remove them by type once added. As a result of this, when a user wants to view drone markers on the map and while location markers are visible. All markers need to be deleted and then redrawn with the required image for each marker type added on creation.

The ability to add new locations to the map by clicking on the map itself also required extending the functionality of Kivy's on_touch function. Overall the lack of functionality and online resources relating to several issues encountered with Kivy's functionality lead to time consuming reading of the addon source code in order to understand how it could be coaxed into providing the required results.

5. Project Timeline/Milestones



Project Spec Received
13 Oct 2020
Research document hand in
13 Nov 2020
Coding began
16 Nov 2020
Functional Specification preliminary hand in
27 Nov 2020
Control virtual drones through Sphinx virtualisation software
2 Dec 2020
Gain control of virtual drones from outside Olympe shell
12 Dec 2020
First Presentation
15 Dec 2020
Flight of real Bebop2 drone using script
26 Dec 2020
Command line based program implementing majority of features
20 Feb 2021
Begin work on GUI implementation
5 Mar 2021
Second Presentation
12 Mar 2021
Added majority of CRUD operation to GUI implementation
5 Apr 2021
Added ability to Add location my clicking directly on map
14 Apr 2021
Final Delivery of project
30 Apr 2021

Figure 18. Project milestone timeline

The project milestones can be categorised by the delivery of documentation for the project, the presentations of works to date and by significant resolution of tasks and blockers. In this respect the milestones can be categorised as follows.

1. 13/10/2020: Project Specification received, research begun.

2. 13/11/2020: Research document handed in, Design phase begun

3. 16/11/2020: Coding for project begun.

4. 27/11/2020: Functional Specification hand in.

5. 02/11/2020: Control of virtual parrot drones achieved through Sphinx virtualization software.

6. 12/12/2020: Gain control of virtual Parrot drones from outside the Olympe shell.

7. 15/12/2020: First project presentation of work to date.

8. 26/12/2020: Flight of real Bebop 2 drone from outside Olympe shell using BebopDrone.py class with script.

9. 20/02/2021: Command line version of program implementing majority of required functionality completed.

10. 05/03/2021: Beginning of conversion of functionality to GUI based application.

11. 12/03.2021: Second presentation of work to date.

12. 05/04/2021: Added majority of CRUD operations to GUI implementation.

13. 14/04/2021: Added ability to add new location by directly clicking on the map. No longer need to enter in latitude and longitude as long floating point numbers.

14. 30/04/2021: Final delivery of Project.

6. Testing

While ideally a full set of black and white box unit tests would have been ideal for this project, it was not possible to achieve this given other problems faced in delivering core functionality for the project. However basic ad-hoc and beta testing was carried out to ensure that each of the included functions of the end product were in a working condition. This included trying to access out of bounds list addresses and ensuring that the program still functioned sufficiently with large numbers of drones and flights in operation simultaneously.

In addition to this there was the proof of concept flight of the Bebop 2 drone in order to test the ability of the program to fly a real drone from outside of the Olympe virtual environment.

7. Further Work

In addition to the current functionality of the application, it would also be beneficial to add some extra functionality and QOL aspects to the application in the future.

7.1 Multi-route flights

It would be a worthwhile addition to the application for the user to be able to create flight plans with multiple landing and take off events. This would have the advantage of being able to make many stops along a more varied and complex route system. The addition of such a feature would not be an enormous step beyond the current functionality of the system.

7.2 Addition of cloud based data storage

The addition of a cloud based data storage system would allow for users to access the fleet of drones from multiple devices and allow for each location on the system to be used as an active control hub for the drones. In this way a user 1 at location 1 could send a drone to location 2 and user 2 at this location would be able to set up a return flight on the system and execute it. Ideally this could be implemented using either a SQL relational database or MongoDB non-relational database.

7.3 Adding compatibility for other drone types

As the system is currently, in order to be used with Parrot drones, the application must be installed on Ubuntu 18.04 or older. This due to the need for the Olympe API to communicate with Parrot drones. It would be advantageous to develop compatibility for other drone type also. This would be no small feat, as for example DJI’s API is only compatible with Windows devices currently. So solving the challenge of control both types of drone simultaneously would be an interesting challenge to undertake.

8. Learning Outcomes

During the course of this project a new understanding of the requirements of creating a large software project was achieved. This included the depth of research required to understand what is needed to realise a working application, as well learning to persevere and find ways to overcome the challenges faced along the way.

I gained a better understanding of the project life cycle during the course of this project. From research for the requirements (planning, designing) to writing the code and iterating through various working states (building, testing). Though the results are not perfect, I also gained a better knowledge of documentation through use of the Sphinx auto documentation tool. I feel I have a better understanding of the scope of the process than previously. My development process was far from perfect, but the shortcomings have given me a better understanding of what is required for future projects.

Through the use of various tools and platforms in the course of development, I have gained a better working knowledge of Python, the project’s chosen language, and of how to build functional applications using it. I have also gained a better understanding of the Linux operating system and specifically Ubuntu. Where once the terminal interface was foreboding, I am now more comfortable in it’s usage. My knowledge of git usage has increased and so has my understanding of the need for proper documentation.

I feel that though many mistakes were made in terms of time management and allocation, I would be better prepared in future to direct my time towards more important aspects of the development process. This was most apparent in the early stages of the project when too much time was given to trying to get multiple virtual drones running in Parrot’s Sphinx virtualisation software instead of working on implementing my own visualisation.

While I believe I managed to achieve the majority of the project requirements, I know that with proper time management and a better focus on the specifications, I could have achieved more of the goals I set out to achieve.

During the course of this project the presentations proved invaluable for getting direct feedback on the work I had done. They helped me to better understand the external aspects of a project as the feedback could be likened to meetings with stakeholders in a real world scenario. The presentations also served to help me hone my own presentation skills and will be valuable additions to my skill set moving forward.

Acknowledgements

Firstly, I would like to extend my gratitude to Dr. Oisin Cawley, my project supervisor, for his direction and knowledge during the course of this project. I would also like to thank all the lecturers I have had the pleasure of learning from over the past four years at IT Carlow, as well as all members of the faculty and staff that have made my time there a pleasurable learning experience.

I would also like to thank my wonderful and helpful classmates from the last four years, many of whom I have had the honour of getting to know as friends, and who have been invaluable in their support and advice.

Finally I would like to thank Parents, my sister, my partner Magda and our dog Bailey for the profound emotional support they have given throughout my studies and my life in general.

Plagiarism Declaration

Declaration

- I declare that all material in this submission e.g. thesis/essay/project/assignment is entirely my/our own work except where duly acknowledged.
- I have cited the sources of all quotations, paraphrases, summaries of information, tables, diagrams or other material; including software and other electronic media in which intellectual property rights may reside.
- I have provided a complete bibliography of all works and sources used in the preparation of this submission.
- I understand that failure to comply with the Institute's regulations governing plagiarism constitutes a serious offense.

Student Name: James Hall

Student Number: C00007006

Signature: James Hall