# Requirements Analysis

## Introduction

The purpose of this project is to produce a system for scheduling delivery via drone. The system will have a GUI for the user where they can add or remove orders, adjust the number of drones being used and see the resulting routes in a visual way. The orders will first be placed into groups with similar locations. The system will then apply an algorithm to find good routes to allow drones to serve customers efficiently. This requirements analysis will lay out all the functional and non-functional requirements of the project.

## Functional Requirements

### Must:

* Have a GUI for the user to interact with:
  + The main screen will have a real map of a city. A fictional depot will be marked on, as well as customer locations, groups that customers have been assigned and the route that each drone will take. This is needed to make the application usable and the results understandable to the average user.
  + The user will be able to input orders from the GUI. They will have the option of clicking a point on the map to select that as the location, and then entering details of the order. Alternatively, they will be able to input everything manually.
  + The user will be able to control some parameters in the system. These will include the number of drones and maximum flight distance per drone. This is essential as it allows a user to create a scenario and figure out how many drones and of what power they would need for their business.
* Divide all the customer locations into smaller groups. For this a clustering method should be used. The locations are to be grouped by proximity so that each drone only serves a tight area. This will result in more efficient delivery as flight time is reduced.
* Routes must then be created for drones to take. There are several algorithms available to solve this problem, at least one will be available for the user. This is required as without it; the problem cannot be solved.

### Should:

* Give the user several algorithms to choose from for path finding. This will be useful for comparison of algorithms and to ensure that we find the best one for the job. Several algorithms were reviewed in the literature review so we will select from there.
* Consider weather conditions. As the user interface will have a real location, we may take the weather into account when planning deliveries. The main factors taken into consideration will be wind speed and direction. If it is too windy a drone will take a long time, or even be unable to complete its route.
* Allow for interactive user input. Ideally the application should be running constantly, waiting for orders to be placed. The user should be able to add, remove or edit an order while the system is running, and it will adapt its current groups and routes to accommodate the new order.

### Could:

* Display the current state of routes as they are created. It would be an interesting way of showing how the different path finding algorithms work and how quickly they converge on a good route.
* When running, show the location of each drone as they move through their route.

## Non-Functional Requirements

* Must use python for development. The language has been chosen as there are several libraries available to aid development. Additionally, I am familiar with the language.
  + Could use scikit-learn library for clustering. The library provides a range of algorithms such as k-means and affinity propagation. Developing a clustering algorithm for ourselves would take a long time and is not the focus of this project thus the library.
  + Could use pyeasyga for the genetic algorithm. As with scikit-learn for clustering, the purpose of this project is not to explore implementing a genetic algorithm. While it would be interesting to do, it is not essential.
  + Could use pyqt to produce a GUI. Pyqt is a powerful tool for creating GUIs that will allow a clean user interface to be developed.
* Should ship as a .exe file. This will allow portability over a range of hardware and make it easy to use.
* Should use an appropriate API for weather data if this is implemented
* Should have a fast runtime. As this is designed for delivering items to customers, a long wait time is not acceptable.

## Summary

|  |  |
| --- | --- |
| **Concept** | **Reason for choice** |
| **Must have:** | |
| A GUI | This is essential to make the end product usable for anyone. It would be extremely difficult to show the user what is happening without doing so visually. |
| Real city map | A map of a real city makes the system more applicable to the real world and shows a realistic outlook on how drones may operate |
| User input from GUI | Without this it becomes a difficult tool for the average user to use and understand |
| User controlled parameters from GUI | A final item of the GUI that will make the system intuitive to use. The user can directly see the effect that different parameters have on the system |
| Clustering technique | For a delivery system, speed is essential. The use of clustering allows the system to more quickly find suitable routes for drones. It also ensures that drones are not flying large distances in between customers where they can avoid it |
| Path finding algorithm | Possibly the most crucial step of the whole process is finding the routes. With no pathfinding algorithm the end product would not solve the problem of scheduling delivery. |
| **Should have:** | |
| Solution produced in a reasonable timeframe | The data should be analysed quickly, and a solution provided. May not be possible if there is a large amount of orders placed at once. |
| Simple to use GUI | It is important that anybody should be able to use the software without guidance. |
| Explore alternative path finding algorithms | Differing algorithms would be an interesting feature so that users can see how they vary in quality and speed of solution |
| Weather conditions considered | Makes the solution more realistic and interesting |
| Interactive user input | It is possible to restart the process every time a new order is added. However, this is far from ideal, as the routes would all need to be recalculated every time a new order was placed. |
| Delivered as a .exe file | This allows portability across windows machines. Additional formats could be created for macOS and linux. |
| **Could have:** | |
| Display current state of routes | It would be interesting for the user to see how routes are generated with different algorithms. |
| Show where drones are on their route | Provides a more interactive experience for the user. |