Epic Project

Vehicle Hiring App

Deadline:

20/12/2023

1 Introduction

In this project you are asked to implement the codebase for a Taxi hiring company. Taxi hiring companies such as Uber and FreeNow rely on the fast localization of available vehicles near a customer on demand. When a customer requests a taxi, the app should contact nearby taxies for the customer and assign one in a fast and efficient way.

2 Challenges

2.1 Storing and Processing Vehicles Information

As the taxi hiring company contains thousands of active registered vehicles, you will need to find an efficient way to search, update and register a vehicle on the app. Due to the nature of these type of applications, such operations should be implemented in fast and efficient way. You must consider an efficient data structure for storing and processing a vehicle's information. A vehicle can be represented using its registration number but can also include driver information (which includes rating) and car information. In addition, you need to consider in your implementation the different types of vehicles (normal, large, etc).

2.2 Storing and Processing Locations

Vehicles and customers are localized using GPS devices available in mobile phones, and these devices determine the longitude and latitude coordinates as floating-point numbers. To simplify the task, however, we will divide the geographical area under consideration into small unit cells and use integers to identify each cell on the map (see Figure 1). Due to the limited size of each cell it is highly unexpected to have more than 10 vehicles at the same cell.

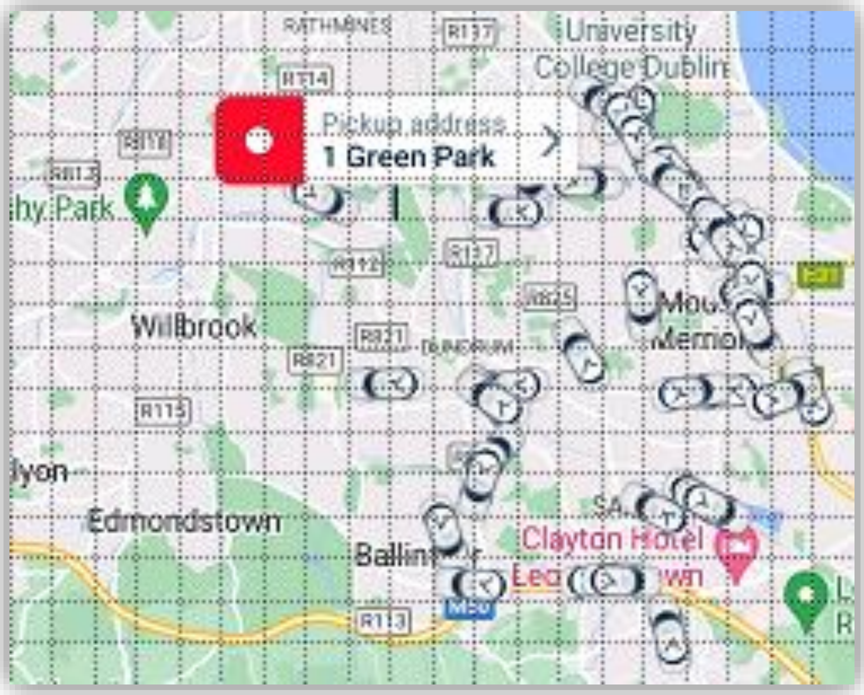

Figure 1: The city is divided into small cells so that a location is identified by two integer coordinates x and y. Note that a single cell may contain several vehicles.

When a customer requests a taxi, only vehicles within a certain range of the customer will be contacted (see Figure2). The customer request specifies the type of the vehicle (normal, large, etc). Once a customer is assigned a taxi, the taxi should no longer be visible on map. After completing the trip, the driver marks it as complete so vehicle will be visible on map for next orders, and the customer will be given an opportunity to rate the driver/vehicle. To store and process locations in fast and efficient manner, you will need to design a suitable data structure. It must be considered that available vehicles are constantly reporting to the system to update their location. For simplicity, we assume that the hiring company is managing vehicles in a small hypothetical city. Expanding the app to manage all Irish cities is left for discussion in your report.

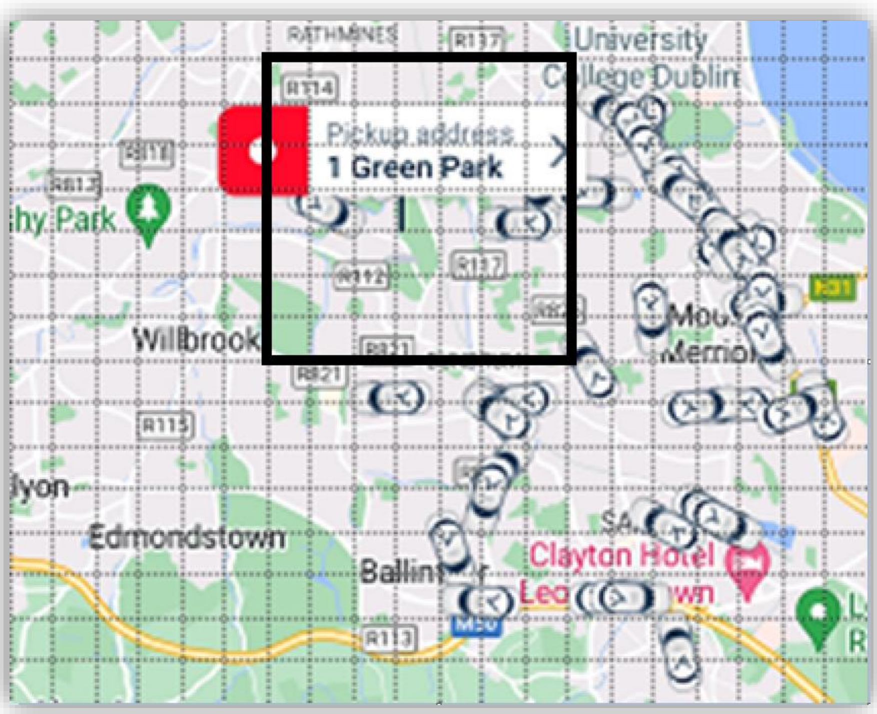

Figure 2: Only vehicles within a square of side length $2r$ centered at the customer cell are contacted.

2.3 Main App Class

The main app class must implement the following interface for testing purposes, where Location is a basic class that stores the map cell coordinates. Note that these interface methods are only for testing purposes, and they should not affect the actual system OOP design and functionality.

```
public interface VehicleHiringTest {

        // Inserts the vehicle with registration number reg to the map at location loc if it has not been already added to map.
        // It should return false if the vehicle is not registered or is already on map
        public abstract boolean testAddToMap(String reg, Location loc);

        // Update the location of the vehicle with the specified reg number to location loc if vehicle exists and return true.
        //Return false if vehicle not registered or has not been added to the map
        public abstract boolean testMoveVehicle(String reg, Location loc);

        // Remove the vehicle with the specified reg number from the map if it is registered and return true.
         // If vehicle is not registered or is not on map the method returns false.
        public abstract boolean testRemoveVehicle(String reg);

        // Return the location of vehicle specified by the reg number if it is registered and added to the map, null otherwise.
        public abstract Location testGetVehicleLoc(String reg);

        // Return a list of all vehicles registration numbers located within a square of side 2*r centered at location loc (inclusive
        //of the boundaries).
        public abstract List<String> testGetVehiclesInRange(Location loc, int r);
}
```

3 Requirements

In addition to considering the challenges mentioned above, the following requirements must be met in the submission:

- You will work in teams of 2 members each to complete the project. It is expected that each member will have a clear understanding of the full implementation at source code level. You can select a partner, whom you have not been partnered with before. There should be equal efforts (contribution) to the code development.
- All data structured used must be implemented by students.
- In compliance with Agile Scrum development, you will have to submit/publish a first sprint release and report on Thursday Week 5, which should include the requirements report for the app and development progress. This should also include guidelines in terms of planning for the final version. This will be followed by a progress check-in with instructors. **(5 marks)**
- On the submission date, you need to submit a report outlining your class design, implementation, discussion on scaling your app to expand to all Irish cities. You must consider the challenges mentioned in section 2 in your implementation and how your solution has addressed such challenges. The code should comply with OOP best practices, in addition to the requirements mentioned in section 2. The report should also outline the efficiency of the proposed solution in terms of complexity analysis. This will be followed by a project interview, where you are expected to show a clear understanding of OOP concepts and how they were considered in your implementation. **(30 marks)**
- You must keep up-to-date codebase in a GitHub repo and add the lecturers GitHub account (iseteaching) to the repo. The readme file must include the names of the team members.
- Your code must be clear and well-commented.
- You need to apply the CI/CD pipeline for automating tests and publishing app releases in your GitHub repo. **(10 marks)**
- The interface of the app is a command-line interface, and it is not required to implement a GUI (you won't get extra marks for implementing a GUI).
- You are not required to implement a database in your project (no extra marks will be rewarded for that).
- Innovation Track Requirements: **(15 marks)**

  Please identify a specific user group (i.e. primary target market) for your concept solution. Describe your justification for selection of this user group and discuss the grouping using existing secondary research sources. For your chosen group, break the segment according to the diffusion of innovation theory (or another salient segmentation typology) and develop 1 clear and customised persona, and 1 visual marketing artefact (website, app, billboard, Instagram post) as relevant. Marketing artefacts do not need to be functioning prototypes but should be specific in brand, tone and messaging to engage the user segment identified.

  Key Outputs:

  1. What broad user group did you choose and why? [50words]
  2. Describe this broad user group, using relevant secondary data/research [300words] [max two visuals (tables/figures)]
  3. Describe your chosen segmentation method and key segment focused on [50 words]
  4. Presentation of persona [1 visual – PowerPoint slide or similar]
  5. Marketing artefact/image [1 visual – image or video]

  Submission:

  All elements to be submitted using the .pptx template provided by the instructor. This should be submitted with the portfolio on Brightspace. This section will be graded by the Innovation Track leader, and will not be examined within the EPIC interview.