Team Rocket

CS252A

No-frills Cab Locator

September 30, 2018

Overview

Our project will involve an app that matches a customer with cab drivers with no additional features. The app will serve the needs of both customers and drivers. It simply matches the customer with drivers, nothing else. No data about the customer whatsoever will be taken, and only the driver's name and phone number will be obtained. The driver's details will be presented to the customer, and then these details will be forgotten. There will be a server that will do the job of matching the customer with a driver. It will also handle storage, updating and deletion of driver details.

Highlights

1. **Simple and Easy-to-Use:** A customer is simply connected to the nearest drivers and given their details. The driver is simply asked for his/her name and phone number, and the update for his/her location is up to him/her. A notification for the driver informs him/her that he/she will be contacted.When an arrangement is later made, the driver will notify the service and opt out.

2. **Thin Client:** The app will simply transmit the GPS location. For the driver, it will transmit his/her details only once, and opting out will remove those details. A single notification for both parties informs them of the match.

3. **Very Low Server Load:** The server only has to query the database for drivers in a grid and for inserting, updating and deleting a driver's details. The server then has to find the closest driver and return the query. No complex processing is involved at any stage, resulting in a quick response for a request.

Specifications

For customers, the app will send their location obtained through GPS to the server, which will send them the name and phone number of the 3 closest drivers, after which it is up to the customer to call any driver and make the arrangements. No details will be extracted from the customer.

For drivers, the app will first ask them for their name and phone number only, which will be forgotten when the driver exits the app. It will then record their current location. The driver will be presented with a button to update their location via GPS, avoiding continuous polling of the driver's location. When a customer receives this driver's info, the driver will be notified he/she should be expecting a call. On a successful arrangement, the driver should opt out, after which his/her details will be deleted.

The server will house a MongoDB database which will contain driver details and location according to a grid-wise demarcation. On receiving a request from the customer, it will locate the grid of the customer, and then return the closest driver in that grid. Failing this, the server will search nearby grids for the closest driver. On receiving a driver's details, it will store them along with the driver's location in the appropriate grid-wise collection in the database. On receiving an update for a driver's location, it will change that entry to the appropriate grid. When the driver opts out, that driver's details will be deleted.

Milestones

1. App

   Create an app according to the specifications listed above using the Ionic framework.

2. Server

   Create a backend using the Django framework in Python that uses a MongoDB database for driver details according to the specifications listed.

3. Website (future plans)

   If time permits create a website for customers only that serves the functionality of the app.