Linnovate Project

Project Description

The project main task will be to create a fully functional taxi company.
We're going to display the taxi company overview and movements as a google map.

The actions a taxi company should complete are:

1. Check the request queue for the next request.
2. Handle Request for a taxi
    a. Find nearest taxi available
    b. Dispatch a taxi
3. Drive customer to destination

Check the request queue for the next request

Description

As we're working with a queue to manage our requests and we know that sometimes there would be no available taxi, we should first check our queue for the next request.

Instructions

Work with the KUE api in order to get the next request in the queue.
If the queue is empty wait for **30** seconds before checking the request queue again.

Handle request for a taxi

Description

Get a request for a taxi at a certain address.

Instructions

implement an endpoint on the server side - **“/taxi/request”**.
when the client side calls this endpoint the function that handles the logic of “requesting a taxi” will be called.
The logic will consist of 2 actions.

Find nearest taxi available

1. Find the closest **available** taxi to send to that address.
   This means that each taxi object has an “available” property that can be either TRUE or FALSE.
   There are 3 cases here:
   - In case there is at least one available taxi we need to find which one is the closest.
   - In case there is only one taxi available we return it.
   - In case there are no available taxi we queue the request again with a **high** priority so that it will be handled as soon as possible.
     If this is the **third** time this request is being handled we need to push this request with a **low** priority to give a chance for the other requests to be handled.
2. Calculate path from two points.
   Find the shortest path from the 2 points, the location of the taxi and the location of the request.
   Since we’re using google maps we can use the directions api to find the nearest path.
   https://developers.google.com/maps/documentation/javascript/directions?hl=tr-TR.
   Here’s a working example made by google you can learn:
   https://google-developers.appspot.com/maps/documentation/javascript/examples/directions-simple?hl=tr-TR
3.

Dispatch a taxi

The main focus of this section is to draw on the map the **marker** of the taxi moving towards the customer. (adding markers to the google map - https://developers.google.com/maps/documentation/javascript/markers)
In order to present a true to life movement we can move the taxi through the points of the google map path (the one we calculated), we will move the taxi image every **5** seconds (the taxi image is provided in the files that came with this project).

Drive customer to destination

The main focus of this section is to draw on the map the **marker** of the taxi moving towards the address that the client requested.
This section includes finding the nearest path to the requested address and drawing the marker moving.

Technologies

The client side will be build using:

- Angular - https://angularjs.org
- SASS (using the SCSS syntax) - http://sass-lang.com/guide, and using the node-sass implementation https://www.npmjs.com/package/node-sass
- Bootstrap 3 - http://getbootstrap.com

The server side will be build using:

- Nodejs - https://nodejs.org
- MongoDB - https://www.mongodb.org, and using mongoose for nodejs http://mongoosejs.com
- Kue (Queue manager) - http://automattic.github.io/kue

*** The project should be hosted on github on the personal student account.**

Tutorials

1. Git - https://try.github.io/levels/1/challenges/1
2. html/css - http://www.codecademy.com/en/tracks/web
3. javascript - http://www.codecademy.com/en/tracks/javascript
4. Angular - https://www.codeschool.com/courses/shaping-up-with-angular-js
5. SASS - http://sass-lang.com/guide
6. Bootstrap - http://www.w3schools.com/bootstrap/bootstrap_get_started.asp
7. Kue - http://www.screenr.com/oyNs, https://vimeo.com/26963384