Wolt

Internship Pre-Assignment

Position

Backend Summer 2020 Engineering Internships

Author(s)

Dung Ho

+358 449865555

dung.ho@edu.turkuamk.fi

Date of the report

28.01.2020

I. Option 2: Back-End

1. Problem description:

- Create a REST API endpoint that allows searching restaurants.
- The APIs should include 3 parameters:
  - *q*: query string. Full or partial match for the string is searched from *name, description* and *tags* fields.
  - *lat*: latitude coordinate (customer's location)
  - *lon*: longtitue coordinate (customer's location)

2. Problem solutions:

- Define a REST API endpoints with Python Flask.

```
import math
from flask import Flask, request, json, jsonify

#Create the Flask application
app = Flask(__name__)
```

- Define a function that calculate the distance between customer's location and the restaurants.

```python
#Function calculate distance between two locations by applying Harvesine fomurla
def haversine(origin, destination):
    org_lat, org_lon = origin
    des_lat, des_lon = destination
    radius = 6371 # Km -  Radius of the Earth

    #Distance between latitudes and longtitudes (in radians)
    d_lat = math.radians(des_lat - org_lat)
    d_lon = math.radians(des_lon - org_lon)

    #Convert latitudes to radians
    r_lat1 = math.radians(org_lat)
    r_lat2 = math.radians(des_lat)

    #Apply Haversine fomurla to calculate distance between 2 locations
    a = pow(math.sin(d_lat/2), 2) + math.cos(r_lat1)*math.cos(r_lat2)*pow(math.sin(d_lon/2), 2)

    #Distance
    d = 2 * radius * math.asin(math.sqrt(a))

    return d
```

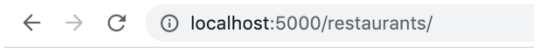
- Define a function that allows searching from the keyword.

```python
#Function returns the restaurats that offer the food's keyword
def get_restaurant(food_name, lat, lon, restaurants):
    #Create empty list to store the restaurants after search
    list_restaurants = []
    for restaurant in restaurants:
        if food_name in restaurant['name'] or food_name in restaurant['description'] or food_name in restaurant['tags']:
            distance = haversine([lat, lon], restaurant['location'][::-1])
            #If the restaurant closer than 3km from the customer's location then add to the list
            if distance <= 3:
                list_restaurants.append(restaurant)
            else:
                return "No restaurant found!!!"

    return list_restaurants
```

- Create the API endpoint.

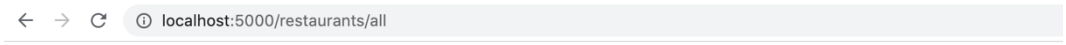
```python
#Handling error
@app.errorhandler(404)
def page_not_found(e):
    return "<h1>404</h1><p>The resource could not be found.</p>", 404


#Show Homepage
@app.route('/', methods=['GET'])
def home():
    return """<h1> WOLT SUMMER INTERNSHIP 2020</h1>
    ···········<p> Search Restaurant base on the food types and the customer's location. </p>"""


#API endpoint - Show all the restaurants in Helsinki
@app.route('/restaurants/all', methods = ['GET'])
def res():
    return jsonify(restaurants)


# API Endpoint - Perform search the restaurant base on food's name and customer's location
@app.route('/restaurants/search', methods = ['GET'])
def search():
    # Check if a query was provided as part of the URL.
    # If the query is provided, assign it to a variable.
    # If no query field is provided, display an error in the browser.
    query_parameters = request.args
    q = query_parameters.get('q')
    lat = query_parameters.get('lat', type = float)
    lon = query_parameters.get('lon', type = float)

    #Return the list of restaurants that matched the query
    l_restaurants = get_restaurant(q, lat, lon, restaurants)

    if len(q) < 1:
        return "Error: No query field provided. Please specify a query."
    else:
        return jsonify(l_restaurants)
```

- The results:

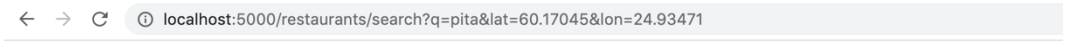


404

The resource could not be found.




```
[
  {
    "blurhash": "j3I8lGUw;;Pu00X;4ygP0Jci:j;I",
    "city": "Helsinki",
    "currency": "EUR",
    "delivery_price": 390,
    "description": "Asenneburgeri",
    "image": "https://prod-wolt-venue-images-cdn.wolt.com/5b348b31fe8992000bbec771/2be8c7738b220df2f9a0974da5c90d90",
    "location": [
      24.941325187683105,
      60.169938852212965
    ],
    "name": "Social Burgerjoint Citycenter",
    "online": false,
    "tags": [
      "hamburger",
      "fries"
    ]
  },
  {
    "blurhash": "j2DUFG8jbu8AXuLIT5Tt0B01R2;;",
    "city": "Helsinki",
    "currency": "EUR",
    "delivery_price": 390,
    "description": "Japanilaista ramenia parhaimmillaan",
    "image": "https://prod-wolt-venue-images-cdn.wolt.com/5d108aa82e757db3f4946ca9/d88ebd36611a5e56bfc6a60264fe3f81",
    "location": [
      24.941786527633663,
      60.169934599421396
    ],
    "name": "Momotoko Citycenter",
    "online": false,
    "tags": [
      "ramen",
      "risotto"
    ]
  },
```


```
[
  {
    "blurhash": "j5Ip9Y;;YgLtdsndTuTt;;Db4hPt",
    "city": "Helsinki",
    "currency": "EUR",
    "delivery_price": 390,
    "description": "Maukkaat ja reilut pitat, salaatit ja falafelit",
    "image": "https://prod-wolt-venue-images-cdn.wolt.com/5de0e07c8995dc9c25304c9f/7680cf2cffae35b7302e3a91b65ce8f5",
    "location": [
      24.93900239467621,
      60.1707249837842
    ],
    "name": "Fafa's Sokos",
    "online": true,
    "tags": [
      "street food",
      "pita"
    ]
  },
  {
    "blurhash": "j5Ip9Y;;YgLtdsndTuTt;;Db4hPt",
    "city": "Helsinki",
    "currency": "EUR",
    "delivery_price": 390,
    "description": "Herkulliset pitaleiv\u00e4t ja falafelit",
    "image": "https://prod-wolt-venue-images-cdn.wolt.com/5996b207ac09660afae98373/7680cf2cffae35b7302e3a91b65ce8f5",
    "location": [
      24.9417033791542,
      60.169590913672
    ],
    "name": "Fafa's Cityk\u00e4yt\u00e4v\u00e4",
    "online": false,
    "tags": [
      "pita",
      "falafel"
    ]
  },
```