Title: Restaurant Finder

Introduction

A command line application which takes in the current location of the user (City name) then gets the exact coordinates for the city name defined. Then uses the Zomato API to find out a restaurant in that city (picked dynamically not hard coded), and then to find out the cab fare prices between your current location and the restaurant location, and display it on the screen.

Programming Software: Python 3.8

What is an API?

An **A**pplication **P**rogram **I**nterface (**API**) is a set of [routines](https://www.webopedia.com/TERM/R/routine.html), [protocols](https://www.webopedia.com/TERM/P/protocol.html), and tools for building [software applications](https://www.webopedia.com/TERM/A/application.html). Basically, an API specifies how software components should interact. Additionally, APIs are used when programming graphical user interface ([GUI](https://www.webopedia.com/TERM/G/Graphical_User_Interface_GUI.html)) components. A good API makes it easier to develop a [program](https://www.webopedia.com/TERM/P/program.html) by providing all the building blocks. A [programmer](https://www.webopedia.com/TERM/P/programmer.html) then puts the blocks together. The API used in our project is the ZOMATO API.

* ZOMATO API:

Zomato APIs give you access to the freshest and most exhaustive information for over 1.5 million restaurants across 10,000 cities globally. With the Zomato APIs, you can:

1. Search for restaurants by name, cuisine, or location.
2. Display detailed information including ratings, location and cuisine.
3. Use the Zomato Foodie Index to show great areas to dine in a city.

In this particular project we used the API to return us the name of a restaurant using the Zomato API. This API to be activated needs an API key which can be obtained through their developer website and can be used to make 1000 calls per day.

GEOPY

* geopy is a Python client for several popular geocoding web services.
* geopy makes it easy for Python developers to locate the coordinates of addresses, cities, countries, and landmarks across the globe using third-party geocoders and other data sources.
* geopy includes geocoder classes for the [OpenStreetMap Nominatim](https://nominatim.org/), [Google Geocoding API (V3)](https://developers.google.com/maps/documentation/geocoding/), and many other geocoding services. The full list is available on the [Geocoders doc section](https://geopy.readthedocs.io/en/latest/#geocoders). Geocoder classes are located in [geopy.geocoders](https://github.com/geopy/geopy/tree/master/geopy/geocoders).
* geopy is mainly used for Geocoding that is returning Laitutude and Longitude for a particular address vice versa and for measuring distance between two locations.

Requests and JSON

Requests is a Python HTTP library, released under the Apache License 2.0. The goal of the project is to make HTTP requests simpler and more human-friendly. By importing the request library working with API becomes much simpler.

[JSON(JavaScript Object Notation)](http://json.org/), specified by [**RFC 7159**](https://tools.ietf.org/html/rfc7159.html) (which obsoletes [**RFC 4627**](https://tools.ietf.org/html/rfc4627.html)) and by [ECMA-404](http://www.ecma-international.org/publications/standards/Ecma-404.htm), is a lightweight data interchange format inspired by [JavaScript](https://en.wikipedia.org/wiki/JavaScript) object literal syntax. JSON exposes an API familiar to users of the standard library [marshal](https://docs.python.org/3/library/marshal.html#module-marshal) and [pickle](https://docs.python.org/3/library/pickle.html#module-pickle) modules.

Algorithm

* Import libraries such as requests, json and random.
* Setting the headers and parameters for the Zomato API in the main function.
* Initiating only two inputs from the user i.e Current city and Destination city.
* Calling for the cityid from the cities search of Zomato API.
* Defining the Search Restaurant function and and setting up the parameters while calling the cityid from main.
* Converting the JSON response into a Python List.
* Creating a for loop specifically for dynamically calling of restaurant and printing it.
* Creating a function for getting coordinates and an estimate travel price.
* Importing geopy for coordinates of the cities.
* Getting the coordinates for current location by using geocoder.
* Calculating the distance between the two and using conditions for calculating the estimate travel and printing it.

Code:

import requests

import json

import random

def SearchRestaurant() :

params = {

'entity\_id' : city\_id,

'entity\_type' : 'city',

}

response = requests.get('https://developers.zomato.com/api/v2.1/search', headers=headers, params=params)

data = json.loads(response.text) #converting json file into python list

n = data['results\_shown']

i= 0

v=[]

k=[data['restaurants'][i]['restaurant']['name']]

for i in range (i,n): #for loop for getting all the restaurants in one list

k=data['restaurants'][i]['restaurant']['name']

v.append(k)

restaurant= v

print("Your Restaurant: %s "%random.choice(restaurant))

def LocationAndEstimate() :

from geopy.geocoders import Nominatim

from geopy.distance import geodesic

geolocator = Nominatim(user\_agent="google maps", timeout=None)

x = geolocator.geocode((a))

c =((x.latitude,x.longitude)) #Coordinates of current city

print ("Current destination coordinates: ")

print(c)

city\_2 = geolocator.geocode((city2))

d = ((city\_2.latitude,city\_2.longitude)) #Coordinates of destination city

e =(geodesic(c,d).km) #Calculate the Distance between the two cities

#Fare estimate w.r.t to Uber

if ( e <= 20 ):

fare = 60 + (e \* 6)

print("Your estimate fare for ride will be = %d Rs"%(fare))

elif (e > 20) :

fare = 60 + (20 \* 6 ) + ((e-20)\*12)

print("Your estimate fare for ride will be = %d Rs"%(fare))

else:

print("No fare")

if \_\_name\_\_ == "\_\_main\_\_":

print("Welcome to Restaurant Finder")

a=input("Enter current city : ")

city2=input("Enter destination city: ")

headers = {

'Accept': 'application/json',

'user-key': '8f43e5828484c5a3673c0ed6b14583b5',

}

params = {

'q': city2,

}

response = requests.get('https://developers.zomato.com/api/v2.1/cities', headers=headers, params=params)

data=response.json()

city\_id = data['location\_suggestions'][0]['id']

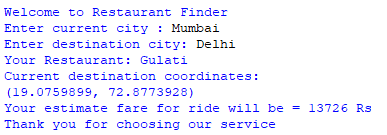
SearchRestaurant()

LocationAndEstimate()

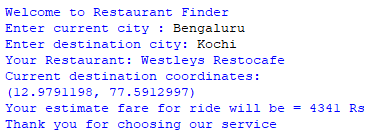
print("Thank you for choosing our service")

Outputs:

1.



2.

. 

Conclusion:

The results obtained were the same as specified in the problem statement. During this it was also noticed that few APIs which were supposed to be free before are asking money, so in a few years’ time maybe developers will not be left with free APIs.