# ****IST105 - Introduction to Programming****

## ****Assignment #9****

## Parsing JSON with a Python Application Using OpenRouteService API

## ****Objectives:****

* Obtain an OpenRouteService API Key.
* Import necessary Python modules.
* Create API request variables and construct a JSON body for a POST request.
* Add user input functionality.
* Add a quit feature so that the user can end the application.
* Display trip information for time, distance, and fuel usage.
* Iterate through the JSON data to extract and output the directions.
* Display error messages for invalid user input.

## ****Background / Scenario****

In this lab, you will create a Python application that retrieves JSON data from the OpenRouteService Directions API, parses the data, and formats it for output to the user. You will use the POST Route request from the OpenRouteService Directions API. Review the OpenRouteService API documentation here:  
<https://openrouteservice.org/dev/#/api-docs>

**Note:** If the above link no longer works, search for “OpenRouteService API Documentation”.

## ****Required Resources****

* Computer with Python installed.
* Access to the internet.
* OpenRouteService API key (free).

## ****Instructions****

#### **Step 1: Obtain an OpenRouteService API Key**

1. Go to <https://openrouteservice.org/>.
2. Click on **Sign Up** at the top of the page.
3. Fill out the form to create a new account.
4. After signing up, navigate to the **API Keys** section in your account dashboard.
5. Generate a new API key and copy it for future use.

#### **Step 2: Import Necessary Modules**

To begin your script for parsing JSON data, you will need to import the requests and json modules.

1. Open a blank script file and save it as 09\_openroute\_parse\_json.py.
2. Import the requests and json modules:

import requests

import json

#### **Step 3: Create Variables for API Request**

The first step in creating your API request is to construct the JSON body that your application will use to make the call. The JSON body will include the following:

* coordinates – A list of start and end coordinates.

1. Create variables to build the JSON body:

directions\_api = "https://api.openrouteservice.org/v2/directions/driving-car"

geocode\_api = "https://api.openrouteservice.org/geocode/search?"

key = "your\_api\_key" # Replace with your OpenRouteService API key

#### **Step 4: Add User Input Functionality**

1. Add a while loop to allow the user to input starting and destination locations:

while True:

orig = input("Starting Location: ")

if orig == "quit" or orig == "q":

break

dest = input("Destination: ")

if dest == "quit" or dest == "q":

break

#### **Step 5: Geocode Addresses**

1. Add a function to geocode addresses:

def geocode\_address(address):

url = f"{geocode\_api}api\_key={key}&text={address}"

response = requests.get(url)

if response.status\_code == 200:

json\_data = response.json()

if json\_data["features"]:

coords = json\_data["features"][0]["geometry"]["coordinates"]

print(f"Geocoded coordinates for '{address}': {coords}") # Debugging

if -90 <= coords[1] <= 90 and -180 <= coords[0] <= 180:

return coords

else:

print(f"Error: Invalid coordinates for address '{address}'")

return None

else:

print(f"Error: No results found for address '{address}'")

return None

else:

print(f"Error: {response.status\_code} - {response.text}")

return None

#### **Step 6: Construct the JSON Body and Make the POST Request**

1. Construct the JSON body and make the POST request:

# Geocode the addresses

orig\_coords = geocode\_address(orig)

dest\_coords = geocode\_address(dest)

if not orig\_coords or not dest\_coords:

print("Unable to geocode one or both addresses. Please try again.\n")

continue

# Construct the JSON body for the POST request

body = {

"coordinates": [orig\_coords, dest\_coords]

}

# Make the POST request

headers = {

"Authorization": key, # Use your API key as the authorization token

"Content-Type": "application/json"

}

response = requests.post(directions\_api, headers=headers, json=body)

json\_data = response.json()

print(json\_data) # Debugging

#### **Step 7: Parse and Display Trip Information**

1. Parse and display trip information:

if response.status\_code == 200:

if 'routes' in json\_data and json\_data['routes']:

route = json\_data['routes'][0]

if 'segments' in route and route['segments']:

segment = route['segments'][0]

print("\nAPI Status: Successful route call.\n")

print("=============================================")

print(f"Directions from {orig} to {dest}")

# Extract trip duration and distance

duration = segment.get('duration', 'N/A')

distance = segment.get('distance', 'N/A')

print(f"Trip Duration: {duration} seconds")

print(f"Distance: {distance} meters")

print("=============================================")

# Extract and print step-by-step directions

if 'steps' in segment:

for step in segment['steps']:

instruction = step.get('instruction', 'N/A')

step\_distance = step.get('distance', 'N/A')

print(f"{instruction} ({step\_distance} meters)")

else:

print("No step-by-step directions available.")

print("=============================================\n")

else:

print("Error: No segments found in the route.")

else:

print("Error: No routes found in the response.")

else:

print(f"Error: {response.status\_code} - {response.text}")

#### **Step 8: Test the Application**

1. Run the script and test it with valid addresses:

**Starting Location: Berlin, Germany**

**Destination: Hamburg, Germany**

Expected Output:

API Status: Successful route call.

**=============================================**

**Directions from Berlin, Germany to Hamburg, Germany**

**Trip Duration: 12345 seconds**

**Distance: 56789 meters**

**=============================================**

**Head north on Friedrichstraße (500 meters)**

**Merge onto A100 (10000 meters)**

**...**

**=============================================**

1. Test with invalid addresses:

**Starting Location: Invalid Address**

**Destination: Another Invalid Address**

Expected Output:

**Error: No results found for address 'Invalid Address'**

**Unable to geocode one or both addresses. Please try again.**

### **Assignment Tasks for Students**

1. **Obtain an API Key**: Follow the instructions to get an OpenRouteService API key.
2. **Write the Script**: Implement the script step-by-step as described.
3. **Test the Application**: Test the script with valid and invalid inputs.
4. **Enhance the Application**: Add features like unit conversion (e.g., meters to kilometers) or error handling for specific API errors.

### **Submission Requirements**

1. **GitHub Repository**:
   * A GitHub repository containing:
     + The Python file (openroute\_directions.py).
   * The repository must have a main branch with the Python file committed and pushed.

[Add your repository]

1. **Deployed EC2 Instance**:
   * The Python script must be deployed and running on an AWS EC2 instance.
   * Ensure the EC2 instance is accessible via its public URL.
   * **Screenshot 1**: EC2 instance’s **Public IP** or **Public DNS** from the AWS.

[Add your screenshot]