**Assignment: Python Programming for GUI Development**

Name: T. Yogananda Reddy

Register Number:192365058

Department: Computer Science & Engineering-(CS)

Date of Submission: 24-08-2024

**Problem 3:** **Real-Time Traffic Monitoring System**

**Scenario:**

You are tasked with developing a real-time traffic monitoring system as part of a smart city initiative. This system should provide live traffic updates and recommend alternative routes to users.

**Tasks:**

1. **Model the Data Flow**: Develop a data flow diagram that illustrates how the system will retrieve real-time traffic data from an external API and present it to the user. This diagram should depict the flow of information between the user, the system, and the API.
2. **Implement a Python Application:** Create a Python application that interfaces with a traffic monitoring API, such as the Google Maps Traffic API. The application should be capable of fetching real-time traffic data based on user inputs.
3. **Display Traffic Information**: Implement functionality to present current traffic conditions, estimated travel times, and any traffic incidents or delays to the user. Ensure that the information is clear and easy to understand.
4. **User Input:** Develop features that allow users to input their starting location and destination. The system should then display real-time traffic updates and suggest alternative routes based on the latest traffic information.

**Deliverables:**

* **Data Flow Diagram**: Provide a detailed diagram showing how the application interacts with the API to retrieve and display traffic data. The diagram should illustrate the flow of data and the key components involved.
* **Pseudocode and Implementation**: Write the pseudocode that outlines the logic of the traffic monitoring system and implement the system in Python. Ensure the code integrates with the traffic monitoring API and performs the required functions.
* **Documentation**: Document the process of integrating with the API, including the steps for fetching and displaying traffic data. Explain how the application processes and presents the traffic information to the user.
* **Explanation**: Provide a summary of any assumptions made during the development of the system, such as the validity of user input and API key configuration. Also, suggest potential improvements to enhance the system, such as adding real-time updates, notifications for traffic alerts, or integrating with a map visualization tool.

# Solution:

# Real-Time Traffic Monitoring System

# 1.Data Flow Diagram

**User**

Start and end point

**API Key and Configuration**

Location Data API key & config Data

**Fetch Traffic Data**

API Request

**Traffic API**

**Display Traffic Information**

# 2. Implementation

|  |
| --- |
| import requests  def get\_traffic\_data(api\_key, start\_point, destination):  """  Retrieves real-time traffic data from the Google Maps Directions API.    Parameters:  - api\_key (str): The API key for accessing Google Maps Directions API.  - start\_point (str): The starting location of the journey.  - destination (str): The destination location of the journey.    Returns:  - dict: A dictionary containing traffic details such as addresses, distance, and travel times.  """  base\_url = "https://maps.googleapis.com/maps/api/directions/json"    params = {  "origin": start\_point,  "destination": destination,  "key": api\_key,  "departure\_time": "now"  }    response = requests.get(base\_url, params=params)  data = response.json()    if "routes" in data and len(data["routes"]) > 0:  route = data["routes"][0]  legs = route["legs"][0]  traffic\_info = {  "start\_address": legs["start\_address"],  "end\_address": legs["end\_address"],  "distance": legs["distance"]["text"],  "duration": legs["duration"]["text"],  "duration\_in\_traffic": legs.get("duration\_in\_traffic", {}).get("text", "N/A"),  "traffic\_conditions": legs.get("traffic\_speed\_entry", [])  }  return traffic\_info  else:  return None  def display\_traffic\_info(traffic\_info):  """  Displays traffic information to the user.    Parameters:  - traffic\_info (dict): Contains traffic details such as addresses, distance, and travel times.  """  if traffic\_info:  print(f"Start Address: {traffic\_info['start\_address']}")  print(f"End Address: {traffic\_info['end\_address']}")  print(f"Distance: {traffic\_info['distance']}")  print(f"Estimated Travel Time: {traffic\_info['duration']}")  print(f"Travel Time in Traffic: {traffic\_info['duration\_in\_traffic']}")  print("Traffic Conditions:")  for condition in traffic\_info["traffic\_conditions"]:  print(f"- {condition['description']}")  else:  print("No traffic information is available for the specified route.")  # Example usage:  api\_key = "YOUR\_API\_KEY" # Replace with your actual API key  start\_point = "Kurnool"  destination = "Hyderabad"  traffic\_info = get\_traffic\_data(api\_key, start\_point, destination)  display\_traffic\_info(traffic\_info) |

# 3.Display the Current traffic information

Start Address: Kurnool, Andhra Pradesh, India

End Address: Hyderabad, Telangana, India

Distance: 213 km

Estimated Travel Time: 4 hours 30 minutes

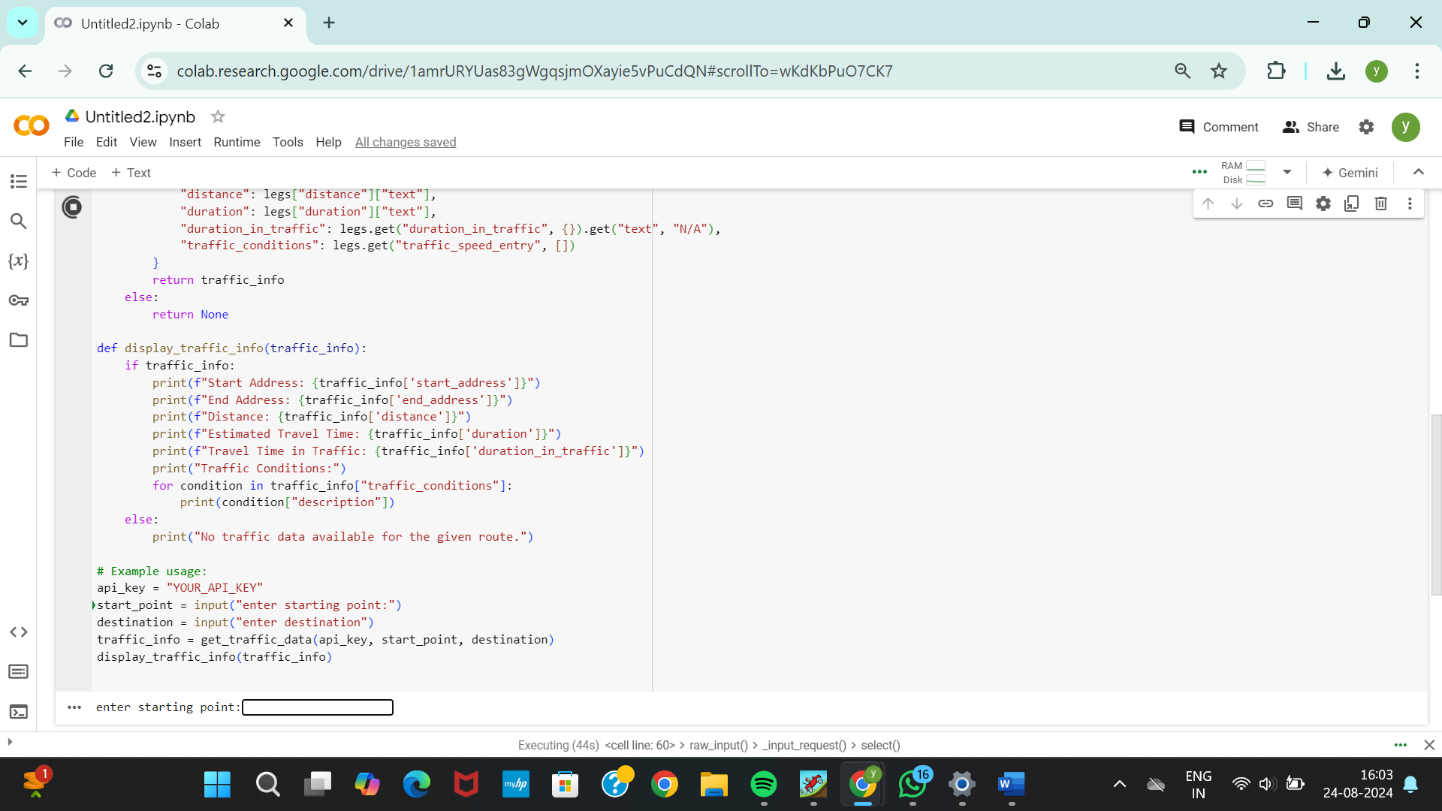
Travel Time in Traffic: 5 hours 10 minutes

Traffic Conditions:

- Slow traffic near Kurnool

- Moderate traffic near Hyderabad

# 4.User Input



**5.Documentation**

# API Integration: This system utilizes the Google Maps Directions API to obtain real-time traffic data.

#  The API request includes parameters for the starting and ending locations, API key, and departure time.

# Methods Used: The traffic data is fetched using the requests library in Python. The API response is processed to extract relevant traffic information, which is then displayed to the user.

# Assumptions: It is assumed that the user provides valid start and end locations, and that the API key is correctly set up with the necessary permissions.

# Potential Improvements: The system could be enhanced by adding features like real-time updates, push notifications for traffic alerts, and integration with a map interface for visualizing the route.

# ­­­­­