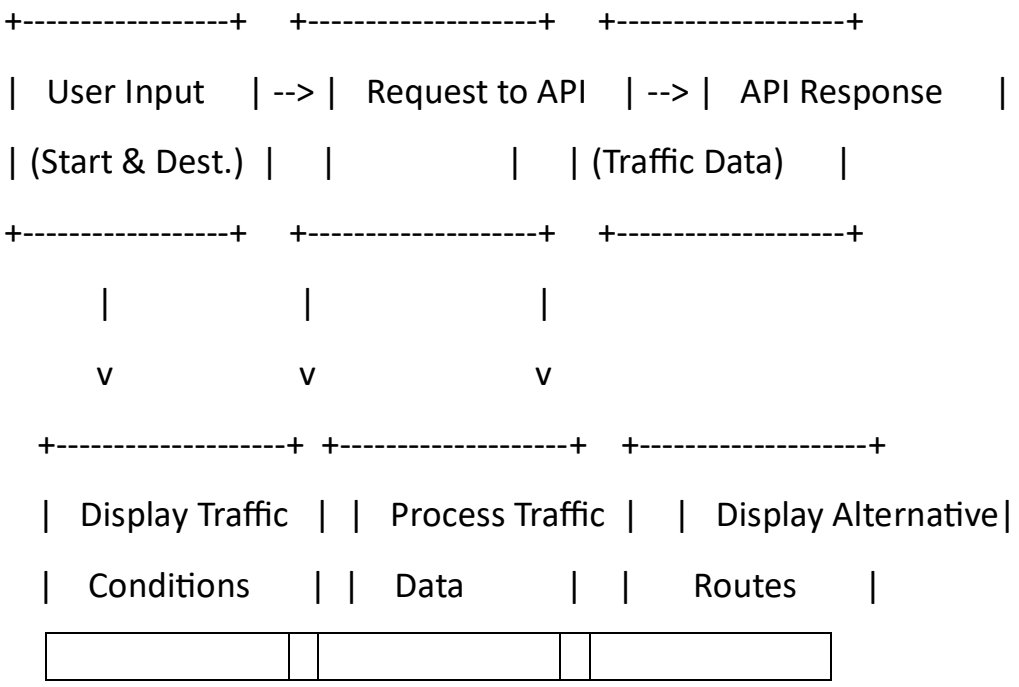
QUESTION : 3

1. Data Flow Diagram

The data flow for a real-time traffic monitoring system can be broken down into the following components:

- **User Input:** The user inputs their starting point and destination.
- **Request to Traffic API:** The application sends a request to the traffic monitoring API with the user's input data.
- **API Response:** The API returns real-time traffic data, including current traffic conditions, estimated travel times, and any incidents or delays.
- **Processing:** The application processes the data to determine alternative routes and filter relevant traffic information.
- **Display:** The processed information is displayed to the user, showing current traffic conditions, travel time, incidents, and suggested alternative routes.



2. Pseudocode :

START

1. Initialize the application.

2. Prompt the user to input the starting point and destination.

3. Send a request to the Traffic API with the user inputs.

4. Receive real-time traffic data from the API.

5. Process the data:

a. Extract current traffic conditions.

b. Calculate estimated travel time.

c. Identify any incidents or delays.

d. Determine possible alternative routes.

6. Display the traffic conditions, travel time, and incidents to the user.

7. Display the alternative routes, if available.

END

3. Python Implementation :

```
import requests


def get_traffic_data(start_point, destination, api_key):
    # Example URL for a traffic monitoring API (e.g., Google Maps Traffic API)
    url = 
f"https://maps.googleapis.com/maps/api/directions/json?origin={start_point}&destination={destination}&key={api_key}&departure_time=now"


    response = requests.get(url)
    data = response.json()


    return data


def process_traffic_data(data):
    # Extract relevant traffic information
    traffic_conditions = data['routes'][0]['legs'][0]['traffic_speed_entry']
    travel_time = data['routes'][0]['legs'][0]['duration_in_traffic']['text']
    incidents = data['routes'][0].get('warnings', [])
```

```
    # Identify alternative routes
    alternative_routes = data['routes'][1:] if len(data['routes']) > 1 else []

    return traffic_conditions, travel_time, incidents, alternative_routes

def display_traffic_info(traffic_conditions, travel_time, incidents, alternative_routes):
    print(f"Current Traffic Conditions: {traffic_conditions}")
    print(f"Estimated Travel Time: {travel_time}")

    if incidents:
        print(f"Incidents: {incidents}")

    if alternative_routes:
        print("Alternative Routes Available:")
        for route in alternative_routes:
            print(route['summary'])

def main():
    # Example inputs
    start_point = "Start_Address"
    destination = "Destination_Address"
    api_key = "Your_Traffic_API_Key"

    # Fetch traffic data
    data = get_traffic_data(start_point, destination, api_key)

    # Process and display the traffic information
    traffic_conditions, travel_time, incidents, alternative_routes = process_traffic_data(data)
    display_traffic_info(traffic_conditions, travel_time, incidents, alternative_routes)
```

```
if __name__ == "__main__":
    main()
```

4. Documentation

- **API Integration: The implementation uses a traffic monitoring API like Google Maps Traffic API. The API request is constructed using the user's starting point, destination, and an API key. The data is fetched in JSON format.**
- **Methods:**
    - **get_traffic_data: Fetches traffic data from the API.**
    - **process_traffic_data: Processes the JSON response to extract relevant information.**
    - **display_traffic_info: Displays the processed information to the user.**
- **Assumptions:**
    - **The user inputs are accurate and valid.**
    - **The API key is correct and has sufficient privileges.**
    - **The API response structure is consistent.**
- **Potential Improvements:**
    - **Implement error handling for invalid user inputs or API failures.**
    - **Add a graphical user interface (GUI) for better user interaction.**
    - **Cache API responses to reduce redundant API calls and improve performance.**
    - **Incorporate user preferences (e.g., avoiding toll roads) in the route suggestions.**