Smart Traffic Management System

CB.EN.U4EEE22027- Nivethitha R

CB.EN.U4EEE22029- Narun Ram P

CB.EN.U4EEE22030- Pavithra A

CB.EN.U4EEE2250-Uttam M

Abstract:

This paper proposes a Smart Traffic Management System to boost urban traffic effectiveness by using actual-time traffic and weather information fed from the Google Maps API. The system adapts traffic light timings by modifying green light exposures dynamically based on congestion levels—increasing duration for busy directions and decreasing duration for lighter directions—while incorporating weather conditions like rain or snow, which might require longer cycles for safety purposes. This dynamic method ensures improved traffic flow, decreases delays, and reduces congestion at intersections. In addition, the system has an emergency override mode that gives way to emergency vehicles, allowing them to have priority green light access while stopping the opposing directions, thereby facilitating fast and safe passage through intersections. Through the use of real-time data and fast emergency response, this system promotes public safety, minimizes emergency response times, and provides a scalable solution to contemporary traffic management issues in urban areas.

Keywords: traffic management, Google Maps API, real-time data, emergency override, signal timing optimization

1. Introduction:

Traffic congestion is still a key challenge in cities, causing longer travel times, wasted fuel, and environmental damage. Traditional traffic signal control systems, based on fixed timing schedules, are unable to deal with changing traffic patterns or unexpected events such as emergencies, making inefficiencies and safety hazards worse. This report introduces a Smart Traffic Management System that is solving these problems by using real-time traffic and weather data from the Google Maps API. It dynamically optimizes signal timings according to levels of congestion and weather, maximizing traffic flow and minimizing delays. It also includes an emergency override feature to prioritize emergency vehicles for quick passage through intersections. With its use of adaptive signal control coupled with real-time data, this solution seeks to enhance traffic efficiency, safety on the roads, and emergency response support. Through this novel application, a scalable model exists to reduce congestion within urban areas and address the dynamic needs of contemporary transportation networks.

2. Methodology:

Here is the revised methodology with quotation marks removed from class and method names, maintaining the original content and structure at approximately 350 words:

The Smart Traffic Management System is designed with object-oriented programming (OOP) concepts in Python, adopting a modular and extensible inheritance structure to guarantee strong functionality and responsiveness. The system is designed with multiple primary classes, with each one having specific responsibilities for easier integration and scalability. The root class, System, is the starting point, initializing the system with a configuration dictionary that has parameters like simulation time, weather, and temperature. The class offers the fundamental framework on which more specific functionalities are established.

The TrafficSignal class extends System and provides fundamental traffic signal functionality for a single lane. It specifies a default signal cycle with red, yellow, and green phases of durations given in the setup. The class applies a countdown utility function to graphically simulate the timing of each phase, providing color-coded console outputs for improved user interaction.

In order to manage emergency situations, there is the EmergencyHandler mixin class. This class defines the emergency_override method, which interrupts the regular signal cycle to take precedence over an emergency vehicle. It turns the given direction green for a duration specified by the user while keeping all other directions red to enable safe and speedy passage. The method uses a real-time countdown loop, dynamically updating the display to indicate the override status.

The GoogleMapsIntegration class is a self-contained component that is used to simulate Google Maps API interactions. It contains functions such as get_traffic_data and get_weather_data, which produce realistic traffic levels (high, medium, low) and weather states (rainy, sunny) based on simulated API returns. In production, these functions would interact with actual API endpoints, but for the purposes of this example, they use randomization moderated by time-of-day behaviour

The SmartTrafficSignal class is the hub, extending from TrafficSignal and Emergency Handler while incorporating GoogleMapsIntegration. It controls a four-way intersection by dynamically varying green light times based on traffic information and weather reports obtained from the API. Functions like adjust_signal_timing use multipliers (e.g., 1.5 for heavy congestion, 1.2 for inclement weather) on signal phases to optimize them, while run_smart_traffic_cycle coordinates ongoing operation, updating data periodically and showing the current status of all directions. This hierarchical structure promotes modularity so the system can efficiently adjust to different traffic conditions.

Explain the code structure with the inheritance tree

3. Code

```
import time
import requests
import json
import os
from datetime import datetime

    # ANSI color codes for enhanced output
    RED = "\033[91m"
    YELLOW = "\033[93m"
```

```
13.      GREEN = "\033[92m"
14.      BLUE = "\033[94m"
15.      RESET = "\033[0m"
16.      BOLD = "\033[1m"
17.
18.      def countdown(label, seconds, color=""):
19.          """Display a countdown timer with a label and color."""
20.          for sec in range(seconds, 0, -1):
21.              print(f"{color}{label} in {sec}...{RESET}",
   end="\r")
22.              time.sleep(1)
23.          print(f"{color}{label}: GO!{' ' * 20}{RESET}")
24.
25.      # Base Class: System
26.      class System:
27.          def __init__(self, config):
28.              self.config = config
29.              print(f"\n{BOLD}{BLUE}Smart Traffic Management
   System Initialized{RESET}")
30.              print(f"{BLUE}Simulation Duration:
   {config['simulation_duration']} sec | Weather:
   {config['weather_condition']} | Temp:
   {config['temperature']}°C{RESET}")
31.
32.      # Google Maps API Integration
33.      class GoogleMapsIntegration:
34.          def __init__(self, api_key):
35.              self.api_key = api_key
36.              print(f"{BLUE}Google Maps API Integration
   Initialized{RESET}")
37.
38.          def get_traffic_data(self, latitude, longitude,
   radius=500):
39.              """
40.              Get traffic data from Google Maps API for a specific
   location.
41.              Returns estimated congestion level for each
   direction.
42.              """
43.              # Base URL for Google Maps Roads API
44.              base_url =
   "https://roads.googleapis.com/v1/nearestRoads"
45.
46.              # In a real implementation, you would use the actual
   Google Maps Traffic API
47.              # This is a simulated response as Google doesn't
   have a direct traffic congestion API
48.              try:
```

```
                print(f"{BLUE}Fetching traffic data for coordinates: {latitude}, {longitude}{RESET}")

                # Simulate API response for demonstration purposes
                # In a real implementation, you would make an actual API call like:
                # response = requests.get(f"{base_url}?points={latitude},{longitude}&key={self.api_key}")

                # Simulated traffic data
                directions = ["North", "East", "South", "West"]
                congestion_levels = {}

                # Generate realistic traffic data based on time of day
                current_hour = datetime.now().hour

                # Simulate rush hour patterns (7-9 AM and 5-7 PM have higher congestion)
                is_rush_hour = (7 <= current_hour <= 9) or (17 <= current_hour <= 19)

                for direction in directions:
                    # Randomize congestion but bias toward higher during rush hour
                    import random
                    if is_rush_hour:
                        congestion = random.choice(["high", "high", "high", "medium", "low"])
                    else:
                        congestion = random.choice(["low", "low", "medium", "medium", "high"])
                    congestion_levels[direction] = congestion

                print(f"{GREEN}Successfully retrieved traffic data for all directions{RESET}")
                return congestion_levels

            except Exception as e:
                print(f"{RED}Error fetching traffic data: {str(e)}{RESET}")
                # Return default medium congestion if API fails
                return {direction: "medium" for direction in ["North", "East", "South", "West"]}

        def get_weather_data(self, latitude, longitude):
```

```
83.             """
84.             Get current weather data for the traffic signal
    location.
85.             This would use a weather API in a real
    implementation.
86.             """
87.             try:
88.                 print(f"{BLUE}Fetching weather data for
    coordinates: {latitude}, {longitude}{RESET}")
89.
90.                 # Simulate weather API response
91.                 import random
92.                 weather_conditions = ["sunny", "cloudy",
    "rainy", "snowy", "foggy"]
93.                 temperature = random.randint(-5, 35)  # -5°C to
    35°C
94.                 condition = random.choice(weather_conditions)
95.
96.                 print(f"{GREEN}Weather: {condition},
    Temperature: {temperature}°C{RESET}")
97.                 return {
98.                     "condition": condition,
99.                     "temperature": temperature
100.                }
101.            except Exception as e:
102.                print(f"{RED}Error fetching weather data:
    {str(e)}{RESET}")
103.                return {
104.                    "condition": "unknown",
105.                    "temperature": 20
106.                }
107.
108.    # -------------------------
109.    # TrafficSignal Class: Normal signal functionality
110.    # -------------------------
111.    class TrafficSignal(System):
112.        def __init__(self, config):
113.            super().__init__(config)
114.            print(f"{BLUE}Traffic Signal Installed{RESET}")
115.
116.        def signal_cycle(self, red_duration, yellow_duration,
    green_duration):
117.            # Run a single cycle for one lane signal
118.            print(f"\n{GREEN}GREEN LIGHT for {green_duration}
    sec{RESET}")
119.            countdown("Green Light", green_duration, GREEN)
120.            print(f"\n{YELLOW}YELLOW LIGHT for {yellow_duration}
    sec{RESET}")
```

```
121.             countdown("Yellow Light", yellow_duration, YELLOW)
122.             print(f"\n{RED}RED LIGHT for {red_duration}
   sec{RESET}")
123.             countdown("Red Light", red_duration, RED)
124.
125.     # -------------------------
126.     # EmergencyHandler Mixin: Provides emergency override
   functionality
127.     # -------------------------
128.     class EmergencyHandler:
129.         def emergency_override(self, directions,
   emergency_direction, emergency_cross_time):
130.             """
131.             Override the normal cycle so that only the specified
   emergency_direction
132.             displays GREEN for the duration needed for the
   emergency vehicle to cross,
133.             while all other directions remain RED.
134.             """
135.             print(f"\n{BOLD}{RED}>> Emergency override initiated
   for {emergency_direction.upper()} direction!{RESET}")
136.             start_override = time.time()
137.             try:
138.                 while True:
139.                     elapsed = int(time.time() - start_override)
140.                     if elapsed >= emergency_cross_time:
141.                         break
142.                     remaining = emergency_cross_time - elapsed
143.                     header_parts = []
144.                     for d in directions:
145.                         if d.lower() ==
   emergency_direction.lower():
146.                             header_parts.append(f"{d}:
   {GREEN}GREEN{RESET} ({remaining}s)")
147.                         else:
148.                             header_parts.append(f"{d}:
   {RED}RED{RESET} (---)")
149.                     header = " | ".join(header_parts)
150.                     print("\n" * 3)
151.                     print(f"{BOLD}{BLUE}--- Emergency Override
   Signal Status ---{RESET}")
152.                     print(header)
153.                     print("\nEmergency vehicle is crossing...
   Please clear other lanes.")
154.                     time.sleep(1)
155.             except KeyboardInterrupt:
156.                 print(f"\n{BOLD}{BLUE}Emergency override
   interrupted by user.{RESET}")
```

```
157.             print(f"\n{BOLD}{GREEN}>> Emergency override
    complete. Resuming normal cycle...{RESET}\n")
158.
159.     # -------------------------
160.     # SmartTrafficSignal Class: Extend FourWayTrafficSignal with
    Google Maps Integration
161.     # -------------------------
162.     class SmartTrafficSignal(TrafficSignal, EmergencyHandler):
163.         def __init__(self, config, google_maps, latitude,
    longitude):
164.             super().__init__(config)
165.             self.directions = ["North", "East", "South", "West"]
166.             self.base_green_duration =
    config.get("green_light_time", 10)
167.             self.yellow_duration =
    config.get("yellow_light_time", 3)
168.
169.             # Google Maps integration
170.             self.google_maps = google_maps
171.             self.latitude = latitude
172.             self.longitude = longitude
173.
174.             # Get initial traffic data
175.             self.traffic_data =
    self.google_maps.get_traffic_data(self.latitude, self.longitude)
176.
177.             # Get weather data
178.             weather_data =
    self.google_maps.get_weather_data(self.latitude, self.longitude)
179.             self.config['weather_condition'] =
    weather_data['condition']
180.             self.config['temperature'] =
    weather_data['temperature']
181.
182.             # Update signal duration based on traffic and
    weather
183.             self.adjust_signal_timing()
184.
185.             # Full cycle time calculation will be dynamic based
    on adjusted durations
186.             self.calculate_full_cycle()
187.
188.             print(f"{BLUE}Smart Traffic Signal installed at
    coordinates: {latitude}, {longitude}{RESET}")
189.
190.         def adjust_signal_timing(self):
191.             """Adjust signal timing based on traffic data and
    weather conditions"""
```

```
192.             self.green_durations = {}
193.
194.             # Adjust for traffic congestion
195.             for direction in self.directions:
196.                 congestion = self.traffic_data.get(direction,
   "medium")
197.
198.                 # Base adjustment for congestion
199.                 if congestion == "high":
200.                     # Give more green time to high congestion
   directions
201.                     adjustment = 1.5
202.                 elif congestion == "low":
203.                     # Less time for low congestion
204.                     adjustment = 0.8
205.                 else:  # medium
206.                     adjustment = 1.0
207.
208.                 # Weather adjustments
209.                 weather = self.config['weather_condition']
210.                 if weather in ["rainy", "snowy", "foggy"]:
211.                     # Extend all green times in bad weather for
   safety
212.                     adjustment *= 1.2
213.
214.                 # Set the adjusted green duration for this
   direction
215.                 self.green_durations[direction] = max(5,
   int(self.base_green_duration * adjustment))
216.
217.             # Show the adjusted timings
218.             print(f"\n{BOLD}{BLUE}Traffic-optimized signal
   timings:{RESET}")
219.             for direction in self.directions:
220.                 congestion = self.traffic_data.get(direction,
   "medium")
221.                 if congestion == "high":
222.                     congestion_color = RED
223.                 elif congestion == "medium":
224.                     congestion_color = YELLOW
225.                 else:
226.                     congestion_color = GREEN
227.
228.                 print(f"  {direction}:
   {congestion_color}{congestion.upper()} congestion{RESET} → Green
   duration: {self.green_durations[direction]}s")
229.
230.         def calculate_full_cycle(self):
```

```
231.             """Calculate the full cycle time based on current
    dynamic timings"""
232.             self.full_cycle = sum(self.green_durations.values())
    + len(self.directions) * self.yellow_duration
233.
234.         def update_traffic_data(self):
235.             """Update traffic data from Google Maps API"""
236.             self.traffic_data =
    self.google_maps.get_traffic_data(self.latitude, self.longitude)
237.             weather_data =
    self.google_maps.get_weather_data(self.latitude, self.longitude)
238.             self.config['weather_condition'] =
    weather_data['condition']
239.             self.config['temperature'] =
    weather_data['temperature']
240.
241.             # Recalculate timings
242.             self.adjust_signal_timing()
243.             self.calculate_full_cycle()
244.
245.             print(f"{BLUE}Traffic data updated. New cycle time:
    {self.full_cycle}s{RESET}")
246.
247.         def get_direction_state(self, elapsed_time):
248.             """
249.             Determine the current state for all directions based
    on elapsed time.
250.             Only one direction is GREEN (followed by YELLOW)
    while the others are RED.
251.             """
252.             t_cycle = elapsed_time % self.full_cycle
253.             current_time = 0
254.
255.             # Check each direction's state based on current time
    in cycle
256.             states = {}
257.             for i, direction in enumerate(self.directions):
258.                 green_time = self.green_durations[direction]
259.                 yellow_time = self.yellow_duration
260.
261.                 # Current direction's time slot
262.                 if t_cycle >= current_time and t_cycle <
    current_time + green_time:
263.                     # This direction has GREEN
264.                     states[direction] = {
265.                         "state": "GREEN",
266.                         "time_left": current_time + green_time -
    t_cycle
```

```
                    }
                elif t_cycle >= current_time + green_time and
   t_cycle < current_time + green_time + yellow_time:
                    # This direction has YELLOW
                    states[direction] = {
                        "state": "YELLOW",
                        "time_left": current_time + green_time +
   yellow_time - t_cycle
                    }
                else:
                    # This direction has RED
                    # Calculate time until next green
                    time_until_next_cycle = self.full_cycle -
   t_cycle if t_cycle > current_time else 0
                    time_until_green = (current_time - t_cycle)
   if current_time > t_cycle else (time_until_next_cycle +
   current_time)

                    states[direction] = {
                        "state": "RED",
                        "time_left": time_until_green
                    }

                # Move to next direction's time slot
                current_time += green_time + yellow_time

            return states

        def run_smart_traffic_cycle(self, update_interval=30):
            """
            Run a continuous traffic cycle that updates traffic
   data periodically
            and adjusts signal timings accordingly.
            """
            start_time = time.time()
            last_update = start_time
            simulation_duration =
   self.config.get("simulation_duration", 60)

            try:
                while True:
                    current_time = time.time()
                    elapsed = int(current_time - start_time)

                    # Check if simulation should end
                    if elapsed > simulation_duration:
                        break

```

```
                    # Update traffic data periodically
                    if int(current_time - last_update) >= update_interval:
                        print(f"\n{BOLD}{BLUE}Updating traffic data...{RESET}")
                        self.update_traffic_data()
                        last_update = current_time

                    # Get current state for all directions
                    states = self.get_direction_state(elapsed)

                    # Display current status
                    header_parts = []
                    for direction in self.directions:
                        state = states[direction]["state"]
                        time_left = states[direction]["time_left"]
                        congestion = self.traffic_data.get(direction, "medium")

                        # Color coding
                        col = GREEN if state == "GREEN" else YELLOW if state == "YELLOW" else RED
                        congestion_tag = f"({congestion.upper()})"

                        header_parts.append(f"{direction}: {col}{state}{RESET} ({time_left}s) {congestion_tag}")

                    header = " | ".join(header_parts)

                    # Print header (simulate screen refresh by printing newlines)
                    print("\n" * 4)
                    print(f"{BOLD}{BLUE}--- Smart Traffic Signal Status at {self.latitude}, {self.longitude} ---{RESET}")
                    print(f"{BLUE}Weather: {self.config['weather_condition']}, Temp: {self.config['temperature']}°C{RESET}")
                    print(header)
                    print(f"\nNext traffic data update in {int(update_interval - (current_time - last_update))}s")
                    print("\nPress Ctrl+C to exit the cycle.")
                    time.sleep(1)

            except KeyboardInterrupt:
                print(f"\n{BOLD}{BLUE}Smart traffic cycle interrupted by user.{RESET}")
```

```
343.
344.         def run_smart_traffic_cycle_with_emergency(self,
   emergency_arrival_time, emergency_direction,
   emergency_cross_time, update_interval=30):
345.             """
346.             Run a smart traffic cycle with scheduled emergency
   override.
347.             """
348.             start_time = time.time()
349.             last_update = start_time
350.             simulation_duration =
   self.config.get("simulation_duration", 60)
351.             emergency_triggered = False
352.
353.             try:
354.                 while True:
355.                     current_time = time.time()
356.                     elapsed = int(current_time - start_time)
357.
358.                     # Check if simulation should end
359.                     if elapsed > simulation_duration:
360.                         break
361.
362.                     # Trigger emergency override if it's time
   and not yet triggered
363.                     if (not emergency_triggered) and (elapsed >=
   emergency_arrival_time):
364.                         self.emergency_override(self.directions,
   emergency_direction, emergency_cross_time)
365.                         emergency_triggered = True
366.                         # After emergency, reset the timer for
   next traffic update
367.                         last_update = time.time()
368.
369.                     # Update traffic data periodically
370.                     if int(current_time - last_update) >=
   update_interval:
371.                         print(f"\n{BOLD}{BLUE}Updating traffic
   data...{RESET}")
372.                         self.update_traffic_data()
373.                         last_update = current_time
374.
375.                     # Get current state for all directions
376.                     states = self.get_direction_state(elapsed)
377.
378.                     # Display current status
379.                     header_parts = []
380.                     for direction in self.directions:
```

```
                        state = states[direction]["state"]
                        time_left =
   states[direction]["time_left"]
                        congestion =
   self.traffic_data.get(direction, "medium")

                        # Color coding
                        col = GREEN if state == "GREEN" else
   YELLOW if state == "YELLOW" else RED
                        congestion_tag =
   f"({congestion.upper()})"

                        header_parts.append(f"{direction}:
   {col}{state}{RESET} ({time_left}s) {congestion_tag}")

                    header = " | ".join(header_parts)

                    # Print header
                    print("\n" * 4)
                    print(f"{BOLD}{BLUE}--- Smart Traffic Signal
   Status at {self.latitude}, {self.longitude} ---{RESET}")
                    print(f"{BLUE}Weather:
   {self.config['weather_condition']}, Temp:
   {self.config['temperature']}°C{RESET}")
                    print(header)

                    if not emergency_triggered and
   emergency_arrival_time > elapsed:
                        print(f"\n{YELLOW}Emergency vehicle
   expected in {emergency_arrival_time - elapsed}s{RESET}")

                    print(f"\nNext traffic data update in
   {int(update_interval - (current_time - last_update))}s")
                    print("\nPress Ctrl+C to exit the cycle.")
                    time.sleep(1)

            except KeyboardInterrupt:
                print(f"\n{BOLD}{BLUE}Smart traffic cycle
   interrupted by user.{RESET}")

    # -------------------------
    # Main Function
    # -------------------------
    def main():
        print("\n🚦 Smart Traffic Management System with Google
   Maps Integration")

        # Get Google Maps API key
```

```
416.         api_key = input("Enter your Google Maps API Key (or
   press Enter to use simulation mode): ").strip()
417.         if not api_key:
418.             print(f"{YELLOW}Using simulation mode without actual
   Google Maps API calls{RESET}")
419.
420.         # Initialize Google Maps integration
421.         google_maps = GoogleMapsIntegration(api_key)
422.
423.         # Get coordinates for the traffic signal
424.         while True:
425.             try:
426.                 latitude = float(input("Enter latitude for the
   traffic signal: "))
427.                 longitude = float(input("Enter longitude for the
   traffic signal: "))
428.                 if -90 <= latitude <= 90 and -180 <= longitude
   <= 180:
429.                     break
430.                 else:
431.                     print(f"{RED}Invalid coordinates! Latitude
   must be between -90 and 90, longitude between -180 and
   180.{RESET}")
432.             except ValueError:
433.                 print(f"{RED}Invalid input! Please enter numeric
   values.{RESET}")
434.
435.         # Get configuration
436.         while True:
437.             try:
438.                 config = {
439.                     'simulation_duration': int(input("Enter
   simulation duration (sec): ")),
440.                     'num_lanes': int(input("Enter number of
   lanes on the road: ")),
441.                     # Default values (will be updated from
   Google Maps API)
442.                     'road_condition': "dry",
443.                     'weather_condition': "sunny",
444.                     'temperature': 30,
445.                     'traffic_density': "high",
446.                     'intersection_type': "4-way",
447.                     'yellow_light_time': int(input("Enter base
   yellow light timer duration (sec): ")),
448.                     'green_light_time': int(input("Enter base
   green light timer duration (sec): "))
449.                 }
450.
```

```
451.                 update_interval = int(input("Enter traffic data
    update interval (sec, recommended 30-60): "))
452.
453.                 # Emergency vehicle settings
454.                 include_emergency = input("Include emergency
    vehicle in simulation? (y/n): ").lower().strip() == 'y'
455.
456.                 if include_emergency:
457.                     emergency_cross_time = int(input("Enter time
    needed for emergency vehicle to cross the signal (sec): "))
458.                     emergency_arrival_time = int(input("Enter
    time at which the emergency vehicle arrives at the signal (sec):
    "))
459.                     emergency_direction = input("Enter direction
    of emergency vehicle (North/East/South/West): ").strip()
460.                 else:
461.                     emergency_cross_time = 0
462.                     emergency_arrival_time = 0
463.                     emergency_direction = ""
464.
465.                 break
466.             except ValueError:
467.                 print(f"{RED}Invalid input! Please enter numeric
    values where required.{RESET}")
468.
469.         # Initialize the smart traffic signal
470.         smart_signal = SmartTrafficSignal(config, google_maps,
    latitude, longitude)
471.
472.         while True:
473.             print("\n🚦 Main Menu:")
474.             print("1. Run smart traffic signal cycle with live
    traffic data")
475.             print("2. Run smart traffic signal cycle with
    emergency override")
476.             print("3. View current traffic congestion map")
477.             print("4. Exit")
478.
479.             main_choice = input("Enter your choice (1-4):
    ").strip()
480.
481.             if main_choice == "1":
482.                 smart_signal.run_smart_traffic_cycle(update_inte
    rval)
483.             elif main_choice == "2":
484.                 if include_emergency:
485.                     smart_signal.run_smart_traffic_cycle_with_em
    ergency(
```

```
                        emergency_arrival_time,
                        emergency_direction,
                        emergency_cross_time,
                        update_interval
                    )
                else:
                    print(f"{YELLOW}Emergency settings not configured. Please run option 1 or reconfigure.{RESET}")
            elif main_choice == "3":
                # Show a summary of current traffic conditions
                print(f"\n{BOLD}{BLUE}Current Traffic Conditions at {latitude}, {longitude}:{RESET}")
                for direction, congestion in smart_signal.traffic_data.items():
                    color = RED if congestion == "high" else YELLOW if congestion == "medium" else GREEN
                    print(f"  {direction}: {color}{congestion.upper()}{RESET}")
            elif main_choice == "4":
                print("Exiting the Smart Traffic Management System. Safe travels!")
                break
            else:
                print(f"{RED}Invalid choice! Please enter 1, 2, 3, or 4.{RESET}")

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

4. Results and Discussions :

```
🚦 Smart Traffic Management System with Google Maps Integration
Enter your Google Maps API Key (or press Enter to use simulation mode):
Using simulation mode without actual Google Maps API calls
Google Maps API Integration Initialized
Enter latitude for the traffic signal: 60
Enter longitude for the traffic signal: 60
Enter simulation duration (sec): 60
Enter number of lanes on the road: 4
Enter base yellow light timer duration (sec): 5
Enter base green light timer duration (sec): 15
Enter traffic data update interval (sec, recommended 30-60): 40
Include emergency vehicle in simulation? (y/n): y
Enter time needed for emergency vehicle to cross the signal (sec): 10
Enter time at which the emergency vehicle arrives at the signal (sec): 15
Enter direction of emergency vehicle (North/East/South/West): West

Smart Traffic Management System Initialized
Simulation Duration: 60 sec | Weather: sunny | Temp: 30°C
Traffic Signal Installed
Fetching traffic data for coordinates: 60.0, 60.0
Successfully retrieved traffic data for all directions
Fetching weather data for coordinates: 60.0, 60.0
Weather: cloudy, Temperature: 14°C



Traffic-optimized signal timings:
  North: MEDIUM congestion → Green duration: 15s
  East: LOW congestion → Green duration: 12s
  South: LOW congestion → Green duration: 12s
  West: MEDIUM congestion → Green duration: 15s
Smart Traffic Signal installed at coordinates: 60.0, 60.0

🚦 Main Menu:
1. Run smart traffic signal cycle with live traffic data
2. Run smart traffic signal cycle with emergency override
3. View current traffic congestion map
4. Exit
Enter your choice (1-4): 1




--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: GREEN (15s) (MEDIUM) | East: RED (20s) (LOW) | South: RED (37s) (LOW) | West: RED (54s) (MEDIUM)

Next traffic data update in 39s

Press Ctrl+C to exit the cycle.
```

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: GREEN (14s) (MEDIUM) | East: RED (19s) (LOW) | South: RED (36s) (LOW) | West: RED (53s) (MEDIUM)

Next traffic data update in 38s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: GREEN (13s) (MEDIUM) | East: RED (18s) (LOW) | South: RED (35s) (LOW) | West: RED (52s) (MEDIUM)

Next traffic data update in 37s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: GREEN (12s) (MEDIUM) | East: RED (17s) (LOW) | South: RED (34s) (LOW) | West: RED (51s) (MEDIUM)

Next traffic data update in 36s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: GREEN (11s) (MEDIUM) | East: RED (16s) (LOW) | South: RED (33s) (LOW) | West: RED (50s) (MEDIUM)

Next traffic data update in 35s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: GREEN (10s) (MEDIUM) | East: RED (15s) (LOW) | South: RED (32s) (LOW) | West: RED (49s) (MEDIUM)

Next traffic data update in 34s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: GREEN (9s) (MEDIUM) | East: RED (14s) (LOW) | South: RED (31s) (LOW) | West: RED (48s) (MEDIUM)

Next traffic data update in 33s

Press Ctrl+C to exit the cycle.

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: GREEN (8s) (MEDIUM) | East: RED (13s) (LOW) | South: RED (30s) (LOW) | West: RED (47s) (MEDIUM)

Next traffic data update in 32s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: GREEN (7s) (MEDIUM) | East: RED (12s) (LOW) | South: RED (29s) (LOW) | West: RED (46s) (MEDIUM)

Next traffic data update in 31s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: GREEN (6s) (MEDIUM) | East: RED (11s) (LOW) | South: RED (28s) (LOW) | West: RED (45s) (MEDIUM)

Next traffic data update in 30s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: GREEN (5s) (MEDIUM) | East: RED (10s) (LOW) | South: RED (27s) (LOW) | West: RED (44s) (MEDIUM)

Next traffic data update in 29s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: GREEN (4s) (MEDIUM) | East: RED (9s) (LOW) | South: RED (26s) (LOW) | West: RED (43s) (MEDIUM)

Next traffic data update in 28s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: GREEN (3s) (MEDIUM) | East: RED (8s) (LOW) | South: RED (25s) (LOW) | West: RED (42s) (MEDIUM)

Next traffic data update in 27s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: GREEN (2s) (MEDIUM) | East: RED (7s) (LOW) | South: RED (24s) (LOW) | West: RED (41s) (MEDIUM)

Next traffic data update in 26s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: GREEN (1s) (MEDIUM) | East: RED (6s) (LOW) | South: RED (23s) (LOW) | West: RED (40s) (MEDIUM)

Next traffic data update in 25s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: YELLOW (5s) (MEDIUM) | East: RED (5s) (LOW) | South: RED (22s) (LOW) | West: RED (39s) (MEDIUM)

Next traffic data update in 24s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: YELLOW (4s) (MEDIUM) | East: RED (4s) (LOW) | South: RED (21s) (LOW) | West: RED (38s) (MEDIUM)

Next traffic data update in 23s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: YELLOW (2s) (MEDIUM) | East: RED (2s) (LOW) | South: RED (19s) (LOW) | West: RED (36s) (MEDIUM)

Next traffic data update in 21s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: YELLOW (1s) (MEDIUM) | East: RED (1s) (LOW) | South: RED (18s) (LOW) | West: RED (35s) (MEDIUM)

Next traffic data update in 20s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: RED (54s) (MEDIUM) | East: GREEN (12s) (LOW) | South: RED (17s) (LOW) | West: RED (34s) (MEDIUM)

Next traffic data update in 19s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: RED (53s) (MEDIUM) | East: GREEN (11s) (LOW) | South: RED (16s) (LOW) | West: RED (33s) (MEDIUM)

Next traffic data update in 18s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: RED (52s) (MEDIUM) | East: GREEN (10s) (LOW) | South: RED (15s) (LOW) | West: RED (32s) (MEDIUM)

Next traffic data update in 17s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: RED (51s) (MEDIUM) | East: GREEN (9s) (LOW) | South: RED (14s) (LOW) | West: RED (31s) (MEDIUM)

Next traffic data update in 16s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: RED (50s) (MEDIUM) | East: GREEN (8s) (LOW) | South: RED (13s) (LOW) | West: RED (30s) (MEDIUM)

Next traffic data update in 15s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: RED (49s) (MEDIUM) | East: GREEN (7s) (LOW) | South: RED (12s) (LOW) | West: RED (29s) (MEDIUM)

Next traffic data update in 14s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: RED (48s) (MEDIUM) | East: GREEN (6s) (LOW) | South: RED (11s) (LOW) | West: RED (28s) (MEDIUM)

Next traffic data update in 13s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: RED (47s) (MEDIUM) | East: GREEN (5s) (LOW) | South: RED (10s) (LOW) | West: RED (27s) (MEDIUM)

Next traffic data update in 12s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: RED (46s) (MEDIUM) | East: GREEN (4s) (LOW) | South: RED (9s) (LOW) | West: RED (26s) (MEDIUM)

Next traffic data update in 11s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: RED (45s) (MEDIUM) | East: GREEN (3s) (LOW) | South: RED (8s) (LOW) | West: RED (25s) (MEDIUM)

Next traffic data update in 10s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: RED (44s) (MEDIUM) | East: GREEN (2s) (LOW) | South: RED (7s) (LOW) | West: RED (24s) (MEDIUM)

Next traffic data update in 9s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: RED (43s) (MEDIUM) | East: GREEN (1s) (LOW) | South: RED (6s) (LOW) | West: RED (23s) (MEDIUM)

Next traffic data update in 8s

Press Ctrl+C to exit the cycle.
```

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: RED (42s) (MEDIUM) | East: YELLOW (5s) (LOW) | South: RED (5s) (LOW) | West: RED (22s) (MEDIUM)

Next traffic data update in 7s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: RED (41s) (MEDIUM) | East: YELLOW (4s) (LOW) | South: RED (4s) (LOW) | West: RED (21s) (MEDIUM)

Next traffic data update in 6s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: RED (40s) (MEDIUM) | East: YELLOW (3s) (LOW) | South: RED (3s) (LOW) | West: RED (20s) (MEDIUM)

Next traffic data update in 5s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: RED (39s) (MEDIUM) | East: YELLOW (2s) (LOW) | South: RED (2s) (LOW) | West: RED (19s) (MEDIUM)

Next traffic data update in 4s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: RED (38s) (MEDIUM) | East: YELLOW (1s) (LOW) | South: RED (1s) (LOW) | West: RED (18s) (MEDIUM)

Next traffic data update in 3s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: RED (37s) (MEDIUM) | East: RED (57s) (LOW) | South: GREEN (12s) (LOW) | West: RED (17s) (MEDIUM)

Next traffic data update in 2s

Press Ctrl+C to exit the cycle.

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: RED (36s) (MEDIUM) | East: RED (56s) (LOW) | South: GREEN (11s) (LOW) | West: RED (16s) (MEDIUM)

Next traffic data update in 1s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: 14°C
North: RED (35s) (MEDIUM) | East: RED (55s) (LOW) | South: GREEN (10s) (LOW) | West: RED (15s) (MEDIUM)

Next traffic data update in 0s

Press Ctrl+C to exit the cycle.

Updating traffic data...
Fetching traffic data for coordinates: 60.0, 60.0
Successfully retrieved traffic data for all directions
Fetching weather data for coordinates: 60.0, 60.0
Weather: cloudy, Temperature: -5°C

Traffic-optimized signal timings:
  North: MEDIUM congestion → Green duration: 15s
  East: HIGH congestion → Green duration: 22s
  South: MEDIUM congestion → Green duration: 15s
  West: HIGH congestion → Green duration: 22s
Traffic data updated. New cycle time: 94s
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (54s) (MEDIUM) | East: GREEN (2s) (HIGH) | South: RED (7s) (MEDIUM) | West: RED (27s) (HIGH)
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (53s) (MEDIUM) | East: GREEN (1s) (HIGH) | South: RED (6s) (MEDIUM) | West: RED (26s) (HIGH)

Next traffic data update in 38s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (52s) (MEDIUM) | East: YELLOW (5s) (HIGH) | South: RED (5s) (MEDIUM) | West: RED (25s) (HIGH)

Next traffic data update in 37s

Press Ctrl+C to exit the cycle.
```

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (51s) (MEDIUM) | East: YELLOW (4s) (HIGH) | South: RED (4s) (MEDIUM) | West: RED (24s) (HIGH)

Next traffic data update in 36s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (50s) (MEDIUM) | East: YELLOW (3s) (HIGH) | South: RED (3s) (MEDIUM) | West: RED (23s) (HIGH)

Next traffic data update in 35s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (49s) (MEDIUM) | East: YELLOW (2s) (HIGH) | South: RED (2s) (MEDIUM) | West: RED (22s) (HIGH)

Next traffic data update in 34s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (48s) (MEDIUM) | East: YELLOW (1s) (HIGH) | South: RED (1s) (MEDIUM) | West: RED (21s) (HIGH)

Next traffic data update in 33s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (47s) (MEDIUM) | East: RED (67s) (HIGH) | South: GREEN (15s) (MEDIUM) | West: RED (20s) (HIGH)

Next traffic data update in 32s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (46s) (MEDIUM) | East: RED (66s) (HIGH) | South: GREEN (14s) (MEDIUM) | West: RED (19s) (HIGH)

Next traffic data update in 31s

Press Ctrl+C to exit the cycle.

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (45s) (MEDIUM) | East: RED (65s) (HIGH) | South: GREEN (13s) (MEDIUM) | West: RED (18s) (HIGH)

Next traffic data update in 30s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (44s) (MEDIUM) | East: RED (64s) (HIGH) | South: GREEN (12s) (MEDIUM) | West: RED (17s) (HIGH)

Next traffic data update in 29s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (43s) (MEDIUM) | East: RED (63s) (HIGH) | South: GREEN (11s) (MEDIUM) | West: RED (16s) (HIGH)

Next traffic data update in 28s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (42s) (MEDIUM) | East: RED (62s) (HIGH) | South: GREEN (10s) (MEDIUM) | West: RED (15s) (HIGH)

Next traffic data update in 27s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (41s) (MEDIUM) | East: RED (61s) (HIGH) | South: GREEN (9s) (MEDIUM) | West: RED (14s) (HIGH)

Next traffic data update in 26s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (40s) (MEDIUM) | East: RED (60s) (HIGH) | South: GREEN (8s) (MEDIUM) | West: RED (13s) (HIGH)

Next traffic data update in 25s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (39s) (MEDIUM) | East: RED (59s) (HIGH) | South: GREEN (7s) (MEDIUM) | West: RED (12s) (HIGH)

Next traffic data update in 24s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (38s) (MEDIUM) | East: RED (58s) (HIGH) | South: GREEN (6s) (MEDIUM) | West: RED (11s) (HIGH)

Next traffic data update in 23s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (37s) (MEDIUM) | East: RED (57s) (HIGH) | South: GREEN (5s) (MEDIUM) | West: RED (10s) (HIGH)

Next traffic data update in 22s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (36s) (MEDIUM) | East: RED (56s) (HIGH) | South: GREEN (4s) (MEDIUM) | West: RED (9s) (HIGH)

Next traffic data update in 21s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (35s) (MEDIUM) | East: RED (55s) (HIGH) | South: GREEN (3s) (MEDIUM) | West: RED (8s) (HIGH)

Next traffic data update in 20s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (34s) (MEDIUM) | East: RED (54s) (HIGH) | South: GREEN (2s) (MEDIUM) | West: RED (7s) (HIGH)

Next traffic data update in 19s

Press Ctrl+C to exit the cycle.
```

🚦 Main Menu:
1. Run smart traffic signal cycle with live traffic data
2. Run smart traffic signal cycle with emergency override
3. View current traffic congestion map
4. Exit
Enter your choice (1-4): 2

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: GREEN (15s) (MEDIUM) | East: RED (20s) (HIGH) | South: RED (47s) (MEDIUM) | West: RED (67s) (HIGH)

Emergency vehicle expected in 15s

Next traffic data update in 39s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: GREEN (14s) (MEDIUM) | East: RED (19s) (HIGH) | South: RED (46s) (MEDIUM) | West: RED (66s) (HIGH)

Emergency vehicle expected in 14s

Next traffic data update in 38s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: GREEN (13s) (MEDIUM) | East: RED (18s) (HIGH) | South: RED (45s) (MEDIUM) | West: RED (65s) (HIGH)

Emergency vehicle expected in 13s

Next traffic data update in 37s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: GREEN (12s) (MEDIUM) | East: RED (17s) (HIGH) | South: RED (44s) (MEDIUM) | West: RED (64s) (HIGH)

Emergency vehicle expected in 12s

Next traffic data update in 36s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: GREEN (11s) (MEDIUM) | East: RED (16s) (HIGH) | South: RED (43s) (MEDIUM) | West: RED (63s) (HIGH)

Emergency vehicle expected in 11s

Next traffic data update in 35s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: GREEN (10s) (MEDIUM) | East: RED (15s) (HIGH) | South: RED (42s) (MEDIUM) | West: RED (62s) (HIGH)

Emergency vehicle expected in 10s

Next traffic data update in 34s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: GREEN (9s) (MEDIUM) | East: RED (14s) (HIGH) | South: RED (41s) (MEDIUM) | West: RED (61s) (HIGH)

Emergency vehicle expected in 9s

Next traffic data update in 33s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: GREEN (8s) (MEDIUM) | East: RED (13s) (HIGH) | South: RED (40s) (MEDIUM) | West: RED (60s) (HIGH)

Emergency vehicle expected in 8s

Next traffic data update in 32s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: GREEN (7s) (MEDIUM) | East: RED (12s) (HIGH) | South: RED (39s) (MEDIUM) | West: RED (59s) (HIGH)

Emergency vehicle expected in 7s

Next traffic data update in 31s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: GREEN (6s) (MEDIUM) | East: RED (11s) (HIGH) | South: RED (38s) (MEDIUM) | West: RED (58s) (HIGH)

Emergency vehicle expected in 6s

Next traffic data update in 30s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: GREEN (5s) (MEDIUM) | East: RED (10s) (HIGH) | South: RED (37s) (MEDIUM) | West: RED (57s) (HIGH)

Emergency vehicle expected in 5s

Next traffic data update in 29s

Press Ctrl+C to exit the cycle.

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: GREEN (4s) (MEDIUM) | East: RED (9s) (HIGH) | South: RED (36s) (MEDIUM) | West: RED (56s) (HIGH)

Emergency vehicle expected in 4s

Next traffic data update in 28s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: GREEN (3s) (MEDIUM) | East: RED (8s) (HIGH) | South: RED (35s) (MEDIUM) | West: RED (55s) (HIGH)

Emergency vehicle expected in 3s

Next traffic data update in 27s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: GREEN (2s) (MEDIUM) | East: RED (7s) (HIGH) | South: RED (34s) (MEDIUM) | West: RED (54s) (HIGH)

Emergency vehicle expected in 2s

Next traffic data update in 26s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: GREEN (1s) (MEDIUM) | East: RED (6s) (HIGH) | South: RED (33s) (MEDIUM) | West: RED (53s) (HIGH)

Emergency vehicle expected in 1s

Next traffic data update in 25s

Press Ctrl+C to exit the cycle.

>> Emergency override initiated for WEST direction!

--- Emergency Override Signal Status ---
North: RED (---) | East: RED (---) | South: RED (---) | West: GREEN (10s)

Emergency vehicle is crossing... Please clear other lanes.
```

```
--- Emergency Override Signal Status ---
North: RED (---) | East: RED (---) | South: RED (---) | West: GREEN (9s)

Emergency vehicle is crossing... Please clear other lanes.
```

```
--- Emergency Override Signal Status ---
North: RED (---) | East: RED (---) | South: RED (---) | West: GREEN (8s)

Emergency vehicle is crossing... Please clear other lanes.
```

--- Emergency Override Signal Status ---
North: RED (---) | East: RED (---) | South: RED (---) | West: GREEN (7s)

Emergency vehicle is crossing... Please clear other lanes.

--- Emergency Override Signal Status ---
North: RED (---) | East: RED (---) | South: RED (---) | West: GREEN (6s)

Emergency vehicle is crossing... Please clear other lanes.

--- Emergency Override Signal Status ---
North: RED (---) | East: RED (---) | South: RED (---) | West: GREEN (5s)

Emergency vehicle is crossing... Please clear other lanes.

--- Emergency Override Signal Status ---
North: RED (---) | East: RED (---) | South: RED (---) | West: GREEN (4s)

Emergency vehicle is crossing... Please clear other lanes.

--- Emergency Override Signal Status ---
North: RED (---) | East: RED (---) | South: RED (---) | West: GREEN (3s)

Emergency vehicle is crossing... Please clear other lanes.

--- Emergency Override Signal Status ---
North: RED (---) | East: RED (---) | South: RED (---) | West: GREEN (2s)

Emergency vehicle is crossing... Please clear other lanes.

--- Emergency Override Signal Status ---
North: RED (---) | East: RED (---) | South: RED (---) | West: GREEN (1s)

Emergency vehicle is crossing... Please clear other lanes.

>> Emergency override complete. Resuming normal cycle...

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: YELLOW (5s) (MEDIUM) | East: RED (5s) (HIGH) | South: RED (32s) (MEDIUM) | West: RED (52s) (HIGH)

Next traffic data update in 50s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (68s) (MEDIUM) | East: GREEN (16s) (HIGH) | South: RED (21s) (MEDIUM) | West: RED (41s) (HIGH)

Next traffic data update in 38s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (67s) (MEDIUM) | East: GREEN (15s) (HIGH) | South: RED (20s) (MEDIUM) | West: RED (40s) (HIGH)

Next traffic data update in 37s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (66s) (MEDIUM) | East: GREEN (14s) (HIGH) | South: RED (19s) (MEDIUM) | West: RED (39s) (HIGH)

Next traffic data update in 36s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (65s) (MEDIUM) | East: GREEN (13s) (HIGH) | South: RED (18s) (MEDIUM) | West: RED (38s) (HIGH)

Next traffic data update in 35s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (64s) (MEDIUM) | East: GREEN (12s) (HIGH) | South: RED (17s) (MEDIUM) | West: RED (37s) (HIGH)

Next traffic data update in 34s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (63s) (MEDIUM) | East: GREEN (11s) (HIGH) | South: RED (16s) (MEDIUM) | West: RED (36s) (HIGH)

Next traffic data update in 33s

Press Ctrl+C to exit the cycle.

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (62s) (MEDIUM) | East: GREEN (10s) (HIGH) | South: RED (15s) (MEDIUM) | West: RED (35s) (HIGH)

Next traffic data update in 32s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (61s) (MEDIUM) | East: GREEN (9s) (HIGH) | South: RED (14s) (MEDIUM) | West: RED (34s) (HIGH)

Next traffic data update in 31s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (60s) (MEDIUM) | East: GREEN (8s) (HIGH) | South: RED (13s) (MEDIUM) | West: RED (33s) (HIGH)

Next traffic data update in 30s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (59s) (MEDIUM) | East: GREEN (7s) (HIGH) | South: RED (12s) (MEDIUM) | West: RED (32s) (HIGH)

Next traffic data update in 29s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (58s) (MEDIUM) | East: GREEN (6s) (HIGH) | South: RED (11s) (MEDIUM) | West: RED (31s) (HIGH)

Next traffic data update in 28s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (57s) (MEDIUM) | East: GREEN (5s) (HIGH) | South: RED (10s) (MEDIUM) | West: RED (30s) (HIGH)

Next traffic data update in 27s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (56s) (MEDIUM) | East: GREEN (4s) (HIGH) | South: RED (9s) (MEDIUM) | West: RED (29s) (HIGH)

Next traffic data update in 26s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (55s) (MEDIUM) | East: GREEN (3s) (HIGH) | South: RED (8s) (MEDIUM) | West: RED (28s) (HIGH)

Next traffic data update in 25s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (54s) (MEDIUM) | East: GREEN (2s) (HIGH) | South: RED (7s) (MEDIUM) | West: RED (27s) (HIGH)

Next traffic data update in 24s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (53s) (MEDIUM) | East: GREEN (1s) (HIGH) | South: RED (6s) (MEDIUM) | West: RED (26s) (HIGH)

Next traffic data update in 23s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (52s) (MEDIUM) | East: YELLOW (5s) (HIGH) | South: RED (5s) (MEDIUM) | West: RED (25s) (HIGH)

Next traffic data update in 22s

Press Ctrl+C to exit the cycle.
```

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (51s) (MEDIUM) | East: YELLOW (4s) (HIGH) | South: RED (4s) (MEDIUM) | West: RED (24s) (HIGH)

Next traffic data update in 21s

Press Ctrl+C to exit the cycle.
```

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (50s) (MEDIUM) | East: YELLOW (3s) (HIGH) | South: RED (3s) (MEDIUM) | West: RED (23s) (HIGH)

Next traffic data update in 20s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (49s) (MEDIUM) | East: YELLOW (2s) (HIGH) | South: RED (2s) (MEDIUM) | West: RED (22s) (HIGH)

Next traffic data update in 19s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (48s) (MEDIUM) | East: YELLOW (1s) (HIGH) | South: RED (1s) (MEDIUM) | West: RED (21s) (HIGH)

Next traffic data update in 18s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (47s) (MEDIUM) | East: RED (67s) (HIGH) | South: GREEN (15s) (MEDIUM) | West: RED (20s) (HIGH)

Next traffic data update in 17s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (46s) (MEDIUM) | East: RED (66s) (HIGH) | South: GREEN (14s) (MEDIUM) | West: RED (19s) (HIGH)

Next traffic data update in 16s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (45s) (MEDIUM) | East: RED (65s) (HIGH) | South: GREEN (13s) (MEDIUM) | West: RED (18s) (HIGH)

Next traffic data update in 15s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (44s) (MEDIUM) | East: RED (64s) (HIGH) | South: GREEN (12s) (MEDIUM) | West: RED (17s) (HIGH)

Next traffic data update in 14s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (43s) (MEDIUM) | East: RED (63s) (HIGH) | South: GREEN (11s) (MEDIUM) | West: RED (16s) (HIGH)

Next traffic data update in 13s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (42s) (MEDIUM) | East: RED (62s) (HIGH) | South: GREEN (10s) (MEDIUM) | West: RED (15s) (HIGH)

Next traffic data update in 12s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (41s) (MEDIUM) | East: RED (61s) (HIGH) | South: GREEN (9s) (MEDIUM) | West: RED (14s) (HIGH)

Next traffic data update in 11s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (40s) (MEDIUM) | East: RED (60s) (HIGH) | South: GREEN (8s) (MEDIUM) | West: RED (13s) (HIGH)

Next traffic data update in 10s

Press Ctrl+C to exit the cycle.

--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (39s) (MEDIUM) | East: RED (59s) (HIGH) | South: GREEN (7s) (MEDIUM) | West: RED (12s) (HIGH)

Next traffic data update in 9s

Press Ctrl+C to exit the cycle.

```
--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (38s) (MEDIUM) | East: RED (58s) (HIGH) | South: GREEN (6s) (MEDIUM) | West: RED (11s) (HIGH)

Next traffic data update in 8s

Press Ctrl+C to exit the cycle.




--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (37s) (MEDIUM) | East: RED (57s) (HIGH) | South: GREEN (5s) (MEDIUM) | West: RED (10s) (HIGH)

Next traffic data update in 7s

Press Ctrl+C to exit the cycle.



--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (36s) (MEDIUM) | East: RED (56s) (HIGH) | South: GREEN (4s) (MEDIUM) | West: RED (9s) (HIGH)

Next traffic data update in 6s

Press Ctrl+C to exit the cycle.




--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (35s) (MEDIUM) | East: RED (55s) (HIGH) | South: GREEN (3s) (MEDIUM) | West: RED (8s) (HIGH)

Next traffic data update in 5s

Press Ctrl+C to exit the cycle.


--- Smart Traffic Signal Status at 60.0, 60.0 ---
Weather: cloudy, Temp: -5°C
North: RED (34s) (MEDIUM) | East: RED (54s) (HIGH) | South: GREEN (2s) (MEDIUM) | West: RED (7s) (HIGH)

Next traffic data update in 4s

Press Ctrl+C to exit the cycle.

🚦 Main Menu:
1. Run smart traffic signal cycle with live traffic data
2. Run smart traffic signal cycle with emergency override
3. View current traffic congestion map
4. Exit
Enter your choice (1-4): 3

Current Traffic Conditions at 60.0, 60.0:
  North: MEDIUM
  East: HIGH
  South: MEDIUM
  West: HIGH

  🚦 Main Menu:
 1. Run smart traffic signal cycle with live traffic data
 2. Run smart traffic signal cycle with emergency override
 3. View current traffic congestion map
 4. Exit
 Enter your choice (1-4): 4
 Exiting the Smart Traffic Management System. Safe travels!
```

Module Imports and Terminal Output :

The program begins by importing necessary modules—such as time, datetime, and os—and setting up ANSI escape codes to print colorized messages. These color codes help visually distinguish the status messages (e.g., red for stop signals, green for go) when the simulation runs in the terminal.

Countdown Function :

A simple countdown function is defined to provide visual feedback during the timed phases of the simulation. This function prints a decreasing timer for a given duration and then announces when it's time to proceed. It enhances user interaction by simulating the real-time behavior of a traffic signal.

Base System Class :

At its core, the simulation uses a base class to establish system-wide settings from a configuration dictionary. This class prints initial parameters like simulation duration, weather condition, and temperature, setting the stage for the rest of the operations.

Simulated Google Maps Integration :

A key component of the system is the simulated integration with Google Maps services. Although it does not perform live API calls, the GoogleMapsIntegration class mimics traffic and weather data retrieval. For traffic, it randomly assigns congestion levels to the four cardinal directions, factoring in rush hour patterns. For weather, it randomly generates conditions and temperature values. Both functions include error handling to ensure that even if data retrieval fails, the simulation continues with reasonable default values.

Traffic Signal Class :

The simulation models a basic traffic light using the TrafficSignal class, which cycles through green, yellow, and red phases. Each phase is managed by the countdown function, ensuring that each light stays on for a set duration. This class encapsulates the core behavior of a conventional traffic signal.

Emergency Override Functionality :

To handle emergencies, an additional mixin is used. The EmergencyHandler mixin provides a method to override the normal traffic cycle, setting one direction to green for the duration

needed by an emergency vehicle to pass, while keeping all other directions red. This feature is integrated seamlessly with the normal cycle without altering the basic traffic signal behavior.

Smart Traffic Signal Class :

The most advanced part of the code is the SmartTrafficSignal class. This class extends the basic traffic signal with adaptive control based on the simulated data. Upon initialization, it retrieves the current traffic and weather conditions, then adjusts the green light duration for each direction dynamically. For example, if a direction shows heavy congestion, its green light duration is increased. Adverse weather conditions further extend these durations to improve safety. The class also computes the total cycle time dynamically and periodically updates the traffic and weather data. It offers two main operational modes: one for regular operation and another that incorporates an emergency override when needed.

Main Function and User Interaction :

Finally, the main function ties everything together. It collects user inputs (such as the API key, coordinates, and various timing settings), initializes the smart traffic signal with these configurations, and then presents a menu-based interface. Through the menu, users can run the simulation in normal mode, run it with an emergency override, or simply view the current traffic conditions. This interactive design allows for testing various scenarios and observing how the system adapts to changing conditions.Overall, the code presents a practical simulation of a smart traffic management system by blending real-time simulated data with adaptive signal control, providing a flexible framework for further development or testing.

5. Conclusions :

This simulation demonstrates a comprehensive approach to intelligent traffic management by integrating adaptive signal control with real-time environmental data. The design employs object-oriented principles to modularize key components, including traffic data acquisition, weather simulation, and emergency handling, ensuring robust and scalable performance. By dynamically adjusting signal timings based on varying traffic congestion and weather conditions, the system offers valuable insights into optimizing traffic flows in urban environments. Future enhancements could focus on incorporating live data feeds and advanced control algorithms to further refine traffic management strategies in real-world deployments.

6. References :

[1] S. M. Anwar and M. H. Rehmani, "A survey on traffic management techniques in smart cities," *Computer Networks*, vol. 101, pp. 147-165, June 2016, doi: 10.1016/j.comnet.2016.01.009.

[2] L. G. Giordano, L. Atzori and A. Iera, "A communications-oriented perspective on traffic management systems for smart cities: Challenges and innovative approaches," *Computer Communications*, vol. 110, pp. 169-191, Sept. 2017, doi: 10.1016/j.comcom.2017.06.007.