**5. PUBLIC TRANSPORT OPTIMIZATION**

**PYTHON SCRIPT FOR IOT COMPONENTS USED IN THE PROJECT**

import random  
import time  
  
# Simulate GPS data  
class GPSSensor:  
    def get\_location(self):  
        latitude = random.uniform(40, 45)  
        longitude = random.uniform(-75, -70)  
        return latitude, longitude  
  
# Simulate fuel consumption data  
class FuelSensor:  
    def get\_fuel\_level(self):  
        return random.uniform(0, 100)  
  
# Simulate temperature and humidity data  
class EnvironmentSensor:  
    def get\_temperature(self):  
        return random.uniform(15, 30)  
  
    def get\_humidity(self):  
        return random.uniform(30, 70)  
  
# Simulate ultrasonic sensor and camera  
class UltrasonicCameraSensor:  
    def detect\_obstacle(self):  
        return random.choice([True, False])  
  
# Simulate ticket validator  
class TicketValidator:  
    def is\_valid\_ticket(self):  
        return random.choice([True, False])  
  
# Simulate suspicious activity sensor  
class SuspiciousActivitySensor:  
    def detect\_suspicious\_activity(self):  
        return random.choice([True, False])  
  
# Simulate weather sensor  
class WeatherSensor:  
    def get\_weather\_conditions(self):  
        conditions = ["Sunny", "Rainy", "Cloudy", "Windy"]  
        return random.choice(conditions)  
  
# Main function for Smart Transit Optimization  
def smart\_transit\_optimization():  
    ticket\_validator = TicketValidator()  
    suspicious\_activity\_sensor = SuspiciousActivitySensor()  
    weather\_sensor = WeatherSensor()  
  
    while True:  
        # Check the ticket validator  
        if not ticket\_validator.is\_valid\_ticket():  
            print("Invalid Ticket Detected. Please resolve the issue.")  
  
        # Check for suspicious activity  
        if suspicious\_activity\_sensor.detect\_suspicious\_activity():  
            print("Suspicious Activity Detected. Please take appropriate action.")  
  
        # Get weather conditions  
        weather = weather\_sensor.get\_weather\_conditions()  
        print(f"Weather Condition: {weather}")

# Simulate noise level monitoring  
class NoiseSensor:  
    def get\_noise\_level(self):  
        return random.uniform(40, 90)  
  
# Main function to monitor and optimize smart transit  
def smart\_transit\_optimization():  
    gps = GPSSensor()  
    fuel\_sensor = FuelSensor()  
    environment\_sensor = EnvironmentSensor()  
    ultrasonic\_camera\_sensor = UltrasonicCameraSensor()  
    noise\_sensor = NoiseSensor()  
  
    while True:  
        # Get data from sensors  
        latitude, longitude = gps.get\_location()  
        fuel\_level = fuel\_sensor.get\_fuel\_level()  
        temperature = environment\_sensor.get\_temperature()  
        humidity = environment\_sensor.get\_humidity()  
        obstacle\_detected = ultrasonic\_camera\_sensor.detect\_obstacle()  
        noise\_level = noise\_sensor.get\_noise\_level()