PUBLIC TRANSPORT AUTOMATION

CODE 1:

communication protocol like MQTT or HTTP to transmit data from the IoT devices to the central server.

```
import paho.mqtt.client as mqtt
def on_connect(client, userdata, flags, rc):
    print("Connected with result code " + str(rc))
    client.subscribe("bus_data")

def on_message(client, userdata, msg):
    # Process the received data
    print(msg.topic + " " + str(msg.payload))

client = mqtt.Client()
client.on_connect = on_connect
client.on_message = on_message

client.connect("mqtt.eclipse.org", 1883, 60)

client.loop_forever()
```

CODE 2:

```
import paho.mqtt.client as mqtt

def on_connect(client, userdata, flags, rc):
    print("Connected with result code " + str(rc))
    client.subscribe("bus_data")

def on_message(client, userdata, msg):
    # Process and store the received data
    print("Received message '" + str(msg.payload) + "' on topic '" + msg.topic + "'")

client = mqtt.Client()
client.on_connect = on_connect
client.on_message = on_message

client.connect("mqtt.eclipse.org", 1883, 60)

client.loop_forever()
```

CODE 3:

```
import sqlite3
```

```
conn = sqlite3.connect('bus_data.db')

cursor = conn.cursor()


# Create a table if it doesn't exist

cursor.execute('''CREATE TABLE IF NOT EXISTS bus_data

                (bus_id text, timestamp text, latitude real, longitude real,
passenger_count integer)''')


# Insert data into the table

cursor.execute("INSERT INTO bus_data VALUES (?, ?, ?, ?, ?)", ('Bus001', '2023-10-
18 12:00:00', 40.7128, -74.0060, 25))


conn.commit()

conn.close()
```

CONCEPT OF CODE:

Sensors and Devices: You'll need IoT devices on buses, stops, and central servers. Each device should be able to communicate data like GPS location, passenger count, and bus status. The code for these devices will depend on the hardware you're using, but typically you'll

need to work with microcontrollers (e.g., Arduino or Raspberry Pi) and sensors.

Data Communication: Use a communication protocol like MQTT or HTTP to transmit data from the IoT devices to the central server.

Data Processing: On the central server, you need code to receive, process, and store the data.

Data Storage: You'll need a database to store the data for further analysis.

Central Server: This is the core of your system. It collects data from all IoT devices, processes it, and manages various aspects of public transport.

Data Communication: You'll need to establish a protocol for communication between devices and the central server.

Real-time Tracking: Implement code to display real-time bus locations and other information.

User Interface: Develop a user-friendly interface for passengers and administrators, typically through a web application or mobile app.

Security: Ensure that your code and data transmission are secure, especially when dealing with real-time tracking and passenger information.

This is a high-level overview, and in reality, each of these components will be more complex. The exact code will depend on your

chosen platforms, languages, and hardware. Make sure to plan the project thoroughly and consider scalability and security from the outset.

BY:

Muthu Tamilarasan.V

950321104031