List of IoT Sensors:

- GPS Sensor: To track the real-time location of the vehicle.
- Passenger Counting Sensor: To count the number of passengers entering and exiting the vehicle.
- Internet Connectivity Module: Cellular or Wi-Fi module to enable communication with the transit information platform.

Arduino Code

- The Arduino code is written in C++ and is uploaded to the Arduino board.
- The code is responsible for reading the data from the sensors and sending it from the Arduino board via the serial port.
- This code is not complete for obtaining the real-time location of the vehicle and the number of passengers entering and exiting the vehicle.
- The GPS and Passenger Counting sensors are not initialized in the code.

```
#include<ArduinoJson.h>

StaticJsonDocument<200> doc;
GPS gps;
PassengerCounter pc;
void setup() {
  Serial.begin(9600);
  gps.begin();
  pc.begin();
}
void loop() {
    float lat = gps.getLatitude();
    float lon = gps.getLongitude();
    int pasin = pc.getPassengerIn();
    int pasout = pc.getPassengerOut();
    if(Serial.available() > 0) {
        char req = Serial.read();
        if(req == 'L') {
            doc.clear();
            JsonObject loc =
doc.createNestedObject("location");
            loc["latitude"] = lat;
            loc["longitude"] = lon;
            serializeJson(doc, Serial);
            Serial.println();
        }
        if(req == 'R') {
            doc.clear();
            JsonObject ridership =
doc.createNestedObject("ridership");
            ridership["in"] = pasin;
            ridership["out"] = pasout;
            serializeJson(doc, Serial);
```

```
                Serial.println();
            }
        }
    }
```

Python Script:

- The python script is used to simulate the IoT sensors and generate the data.

To connect to the MQTT Broker:

- Start the MQTT Broker on the local machine.
- Run the python script to connect to the MQTT Broker.
- The python script will publish the data to the MQTT Broker.
- Subscribe to the MQTT Broker to receive the data.
- Command : `mosquitto_sub -t pto_data -d -h localhost`

Packages:

In [ ]:
```
%pip install pyserial
```

Python Script:

In [ ]:
```
import time
import json
import paho.mqtt.client as mqtt
import serial

# Function to get real-time GPS data
def get_location_data():
    try:
        # location_data = {"latitude": 1.3521, "longitude": 103.8198} # S
        # Open serial port in Arduino
        ser = serial.Serial('/dev/cu.usbserial-1440', 9600, timeout=1)
        ser.flush()

        # Send request to get GPS data
        ser.write(b"L")
        print("Requesting GPS data...")

        # Read GPS data
        line = ser.readline().decode('utf-8')
        location_data = json.loads(line)["location"]

        ser.close()
        return location_data
    except Exception as e:
        print(f"Error getting location data: {str(e)}")
        return None

# Function to retreive ridership data
def get_ridership_data():
    try:
```

```
        # ridership_data = {"in": 10, "out": 5 } # Sample Data
        # Open serial port in Arduino
        ser = serial.Serial('/dev/cu.usbserial-1440', 9600, timeout=1)
        ser.flush()

        # Send request to get ridership data
        ser.write(b"R")

        # Read ridership data
        line = ser.readline().decode('utf-8')
        ridership_data = json.loads(line)["ridership"]

        ser.close()
        return ridership_data
    except Exception as e:
        print(f"Error getting ridership data: {str(e)}")
        return None

# Initialize MQTT client
mqtt_client = mqtt.Client(protocol=mqtt.MQTTv5)
mqtt_client.connect("localhost", 1883)

while True:
    try:
        location_data = get_location_data()
        ridership_data = get_ridership_data()

        if location_data is not None and ridership_data is not None:
            payload = {
                "location": location_data,
                "ridership": ridership_data
            }

            # Send data to the transit information platform
            topic = "pto_data"
            mqtt_client.publish(topic, json.dumps(payload))

            print("Data sent:", payload)

        time.sleep(10)

    except Exception as e:
        print(f"Error: {str(e)}")
    finally:
        mqtt_client.disconnect()
```