Creating an IoT program to interface with a Raspberry Pi for a public transportation optimization project involves several steps. Below is a simplified example using Python and MQTT (Message Queuing Telemetry Transport) as the communication protocol. This example focuses on real-time bus tracking:

1. Install Required Libraries :

First, make sure you have the necessary libraries installed. For this example, you’ll need `paho-mqtt` for MQTT communication.

```bash

Pip install paho-mqtt

```

1. Set Up MQTT Broker:

You’ll need an MQTT broker (e.g., Mosquitto) for communication. Ensure the Raspberry Pi can connect to it.

1. Raspberry Pi Code:

Here’s an example of a Python script on the Raspberry Pi that collects GPS data and publishes it to an MQTT topic:

```python

Import paho.mqtt.client as mqtt

Import time

From gps import GPS # Use the library for your GPS module

# Initialize GPS module

Gps = GPS()

# MQTT configuration

Mqtt\_broker = “mqtt\_broker\_ip”

Mqtt\_port = 1883

Topic = “bus\_location”

# Create an MQTT client

Client = mqtt.Client(“BusPi”)

# Connect to the MQTT broker

Client.connect(mqtt\_broker, mqtt\_port)

Try:

While True:

# Read GPS data (latitude and longitude)

Latitude, longitude = gps.read\_location()

# Prepare data

Bus\_data = {“latitude”: latitude, “longitude”: longitude}

# Publish data to the MQTT topic

Client.publish(topic, str(bus\_data))

Time.sleep(10) # Publish every 10 seconds

Except KeyboardInterrupt:

Print(“Exiting...”)

Client.disconnect()

```

1. User Application:

Passengers and operators can subscribe to the “bus\_location” MQTT topic to receive real-time bus location updates. You can create a web or mobile app for this purpose.

Here’s an example of how to subscribe to the MQTT topic using Python:

```python

Import paho.mqtt.client as mqtt

Def on\_message(client, userdata, message):

# Process the received bus location data

Print(f”Received message ‘{message.payload}’ on topic ‘{message.topic}’”)

Mqtt\_broker = “mqtt\_broker\_ip”

Mqtt\_port = 1883

Topic = “bus\_location”

Client = mqtt.Client(“BusApp”)

Client.on\_message = on\_message

Client.connect(mqtt\_broker, mqtt\_port)

Client.subscribe(topic)

Client.loop\_forever()

```

This is a basic example. In a real-world scenario, you would need to implement more features like data validation, security, and integration with your optimization algorithms. Additionally, consider using a database to store and analyze historical data for route optimization.