Select and Install Passenger Counters:

Sensor Selection: Choose passenger counters that suit the type of public transportation (e.g., buses, trams, trains). These sensors typically use infrared, ultrasonic, or other technologies.

Installation Location: Mount passenger counters near the entrance/exit points of the vehicle, where they can effectively track passengers boarding and alighting.

Power Supply: Ensure the passenger counters have a reliable power source. They often use low-voltage power and can be connected to the vehicle's electrical system.

Wiring and Connections: Connect the passenger counters to the IoT device (e.g., Raspberry Pi) to transmit passenger data.

Data Transmission to Transit Information Platform:

Develop Data Collection Scripts: Write Python scripts to collect data from the installed GPS sensors and passenger counters. These scripts should format the data and send it to your transit information platform.

Communication Protocol: Choose a suitable communication protocol (e.g., MQTT, HTTP) for transmitting data from the IoT devices to your transit information platform.

Real-time Data Transmission: Ensure that the IoT devices send data in real-time, and implement error-handling mechanisms for data transmission issues.

Test and Calibrate Sensors:

Test Sensors: Conduct thorough testing to ensure that the sensors are accurately capturing and transmitting data. Verify that GPS data is precise and passenger counters are counting accurately.

Calibration: If necessary, calibrate the sensors to improve data accuracy.

Regular Maintenance:

Establish a maintenance schedule to inspect and maintain the sensors and IoT devices regularly. Address any issues promptly.

Keep firmware and software up to date to ensure the sensors function correctly.

Python scripts for sensors:

PASSENGER COUNTING SENSORS

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

# Load the passenger counting data

passenger\_counting\_data = pd.read\_csv("passenger\_counting\_data.csv")

# Calculate the average ridership for each route

average\_ridership = passenger\_counting\_data.groupby("route")["ridership"].mean()

# Identify the routes with the highest ridership

most\_popular\_routes = average\_ridership.nlargest(5)

# Identify the routes with the lowest ridership

least\_popular\_routes = average\_ridership.nsmallest(5)

# Plot the average ridership for each route

plt.bar(most\_popular\_routes.index, most\_popular\_routes.values)

plt.xlabel("Route")

plt.ylabel("Average Ridership")

plt.title("Most Popular Public Transportation Routes")

plt.show()

# Plot the average ridership for each route

plt.bar(least\_popular\_routes.index, least\_popular\_routes.values)

plt.xlabel("Route")

plt.ylabel("Average Ridership")

plt.title("Least Popular Public Transportation Routes")

plt.show()

# Recommend changes to the public transportation system

# For example, we could increase the frequency of buses on the most popular routes

# and decrease the frequency of buses on the least popular routes.

print("Recommended changes to the public transportation system:")

print("Increase the frequency of buses on the following routes:")

for route in most\_popular\_routes.index:

print(route)

print("Decrease the frequency of buses on the following routes:")

for route in least\_popular\_routes.index:

print(route)

OCCUPANCY SENSORS

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

# Load the occupancy sensor data

occupancy\_sensor\_data = pd.read\_csv("occupancy\_sensor\_data.csv")

# Calculate the average occupancy for each route

average\_occupancy = occupancy\_sensor\_data.groupby("route")["occupancy"].mean()

# Identify the routes with the highest occupancy

most\_occupied\_routes = average\_occupancy.nlargest(5)

# Identify the routes with the lowest occupancy

least\_occupied\_routes = average\_occupancy.nsmallest(5)

# Plot the average occupancy for each route

plt.bar(most\_occupied\_routes.index, most\_occupied\_routes.values)

plt.xlabel("Route")

plt.ylabel("Average Occupancy")

plt.title("Most Occupied Public Transportation Routes")

plt.show()

# Plot the average occupancy for each route

plt.bar(least\_occupied\_routes.index, least\_occupied\_routes.values)

plt.xlabel("Route")

plt.ylabel("Average Occupancy")

plt.title("Least Occupied Public Transportation Routes")

plt.show()

# Recommend changes to the public transportation system

# For example, we could increase the frequency of buses on the most occupied routes

# and decrease the frequency of buses on the least occupied routes.

print("Recommended changes to the public transportation system:")

print("Increase the frequency of buses on the following routes:")

for route in most\_occupied\_routes.index:

print(route)

print("Decrease the frequency of buses on the following routes:")

for route in least\_occupied\_routes.index:

print(route)

ENVIRONMENTAL SENSORS

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

# Load the environmental sensor data

environmental\_sensor\_data = pd.read\_csv("environmental\_sensor\_data.csv")

# Calculate the average environmental impact for each route

average\_environmental\_impact = environmental\_sensor\_data.groupby("route")["environmental\_impact"].mean()

# Identify the routes with the highest environmental impact

most\_polluted\_routes = average\_environmental\_impact.nlargest(5)

# Identify the routes with the lowest environmental impact

least\_polluted\_routes = average\_environmental\_impact.nsmallest(5)

# Plot the average environmental impact for each route

plt.bar(most\_polluted\_routes.index, most\_polluted\_routes.values)

plt.xlabel("Route")

plt.ylabel("Average Environmental Impact")

plt.title("Most Polluted Public Transportation Routes")

plt.show()

# Plot the average environmental impact for each route

plt.bar(least\_polluted\_routes.index, least\_polluted\_routes.values)

plt.xlabel("Route")

plt.ylabel("Average Environmental Impact")

plt.title("Least Polluted Public Transportation Routes")

plt.show()

# Recommend changes to the public transportation system

# For example, we could increase the frequency of electric buses on the most polluted routes

# and decrease the frequency of diesel buses on the least polluted routes.

print("Recommended changes to the public transportation system:")

print("Increase the frequency of electric buses on the following routes:")

for route in most\_polluted\_routes.index:

print(route)

print("Decrease the frequency of diesel buses on the following routes:")

for route in least\_polluted\_routes.index:

print(route)

SECURITY SENSORS

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

# Load the security sensor data

security\_sensor\_data = pd.read\_csv("security\_sensor\_data.csv")

# Calculate the average number of security incidents for each route

average\_security\_incidents = security\_sensor\_data.groupby("route")["security\_incidents"].mean()

# Identify the routes with the highest number of security incidents

most\_dangerous\_routes = average\_security\_incidents.nlargest(5)

# Identify the routes with the lowest number of security incidents

least\_dangerous\_routes = average\_security\_incidents.nsmallest(5)

# Plot the average number of security incidents for each route

plt.bar(most\_dangerous\_routes.index, most\_dangerous\_routes.values)

plt.xlabel("Route")

plt.ylabel("Average Number of Security Incidents")

plt.title("Most Dangerous Public Transportation Routes")

plt.show()

# Plot the average number of security incidents for each route

plt.bar(least\_dangerous\_routes.index, least\_dangerous\_routes.values)

plt.xlabel("Route")

plt.ylabel("Average Number of Security Incidents")

plt.title("Least Dangerous Public Transportation Routes")

plt.show()

# Recommend changes to the public transportation system

# For example, we could increase the number of security guards on the most dangerous routes

# and install more security cameras on the least dangerous routes.

print("Recommended changes to the public transportation system:")

print("Increase the number of security guards on the following routes:")

for route in most\_dangerous\_routes.index:

print(route)

print("Install more security cameras on the following routes:")

for route in least\_dangerous\_routes.index:

print(route)

LIGHT AND MOTION SENSORS

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

# Load the light and motion sensor data

light\_and\_motion\_sensor\_data = pd.read\_csv("light\_and\_motion\_sensor\_data.csv")

# Calculate the average light and motion levels for each route

average\_light\_levels = light\_and\_motion\_sensor\_data.groupby("route")["light\_levels"].mean()

average\_motion\_levels = light\_and\_motion\_sensor\_data.groupby("route")["motion\_levels"].mean()

# Identify the routes with the lowest light levels

darkest\_routes = average\_light\_levels.nsmallest(5)

# Identify the routes with the highest motion levels

most\_crowded\_routes = average\_motion\_levels.nlargest(5)

# Plot the average light levels for each route

plt.bar(darkest\_routes.index, darkest\_routes.values)

plt.xlabel("Route")

plt.ylabel("Average Light Levels")

plt.title("Darkest Public Transportation Routes")

plt.show()

# Plot the average motion levels for each route

plt.bar(most\_crowded\_routes.index, most\_crowded\_routes.values)

plt.xlabel("Route")

plt.ylabel("Average Motion Levels")

plt.title("Most Crowded Public Transportation Routes")

plt.show()

# Recommend changes to the public transportation system

# For example, we could increase the lighting on the darkest routes

# and add more vehicles to the most crowded routes.

print("Recommended changes to the public transportation system:")

print("Increase the lighting on the following routes:")

for route in darkest\_routes.index:

print(route)

print("Add more vehicles to the following routes:")

for route in most\_crowded\_routes.index:

print(route)

AIR QUALITY SENSORS

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

# Load the air quality sensor data

air\_quality\_sensor\_data = pd.read\_csv("air\_quality\_sensor\_data.csv")

# Calculate the average air quality for each route

average\_air\_quality = air\_quality\_sensor\_data.groupby("route")["air\_quality"].mean()

# Identify the routes with the poorest air quality

most\_polluted\_routes = average\_air\_quality.nsmallest(5)

# Identify the routes with the best air quality

least\_polluted\_routes = average\_air\_quality.nlargest(5)

# Plot the average air quality for each route

plt.bar(most\_polluted\_routes.index, most\_polluted\_routes.values)

plt.xlabel("Route")

plt.ylabel("Average Air Quality")

plt.title("Most Polluted Public Transportation Routes")

plt.show()

# Plot the average air quality for each route

plt.bar(least\_polluted\_routes.index, least\_polluted\_routes.values)

plt.xlabel("Route")

plt.ylabel("Average Air Quality")

plt.title("Least Polluted Public Transportation Routes")

plt.show()

# Recommend changes to the public transportation system

# For example, we could increase the frequency of electric buses on the most polluted routes

# and decrease the frequency of diesel buses on the least polluted routes.

print("Recommended changes to the public transportation system:")

print("Increase the frequency of electric buses on the following routes:")

for route in most\_polluted\_routes.index:

print(route)

print("Decrease the frequency of diesel buses on the following routes:")

for route in least\_polluted\_routes.index:

print(route)