Ridership Analysis

```
# Example: Calculate daily ridership trends
daily_ridership = df.groupby('date')
['riders'].sum()

# Data Visualization
# Example: Plot daily ridership trends
plt.figure(figsize=(12, 6))
plt.plot(daily_ridership.index,
daily_ridership.values)
plt.title("Daily Ridership Trends")
plt.xlabel("Date")
plt.ylabel("Ridership")
plt.show()

# Reporting: Generate summary reports or
dashboards with the analysis results.

# Save or export analysis results and
reports as needed.

# Automation: Consider creating a script that
can be run periodically to update the
analysis with new data.

# Deployment: Deploy your program within
your organization, and share reports with
relevant stakeholders.
```

This is a simplified framework. In practice, you'll need to customize it based on the data you have and the specific analysis you want to perform. Additionally, you may need to use more advanced tools and techniques, especially for predictive anal

```
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

# Data Collection: Load public transport data (e.g., schedules, ridership, performance metrics) into DataFrames using Pandas.

# Data Preprocessing: Clean and prepare the data for analysis (e.g., handling missing values, data transformation).

# Data Analysis: Calculate key performance metrics such as on-time performance, ridership, and service frequency.

# Data Visualization: Create plots and charts to visualize the data and analysis results using Matplotlib or Seaborn.

# Performance Metrics Calculation
# Example: On-Time Performance
on_time_data = df['arrival_time'] - df['scheduled_arrival_time']
on_time_percentage = (on_time_data <= threshold).sum() / len(on_time_data)
```

2023-10-30