Title: Metro Operations Optimization

Name: Sirisha P G

USN: 1BO23MC040

Department: MCA(VTU)

Sem : Third

❖ Objective:-

The objective of Metro Operations Optimization is to enhance efficiency, reliability, and sustainability while minimizing costs and improving the passenger experience. This involves optimizing train schedules and frequency to match demand, reducing dwell time at stations, and implementing energy-efficient driving strategies. Ensuring reliability and safety is crucial, achieved through predictive maintenance, automated train control, and rapid incident management. Passenger experience is enhanced by minimizing wait times, preventing overcrowding, and providing real-time information with seamless ticketing systems. Cost reduction is another key focus, involving efficient workforce planning, energy-saving measures like regenerative braking, and optimizing rolling stock utilization. Additionally, sustainability is prioritized by reducing the carbon footprint through green energy initiatives, smart infrastructure planning, and promoting public transport to alleviate urban congestion.

- Maximizing Efficiency
- Enhancing Reliability & Safety
- Improving Passenger Experience

❖ Problem Statement:-

The increasing demand for efficient, reliable, and sustainable urban transportation presents significant challenges for metro rail systems. Inefficiencies in train scheduling, high operational costs, energy wastage, frequent service disruptions, and overcrowding negatively impact both operators and passengers. Traditional methods of metro operations struggle to adapt to fluctuating passenger demand, leading to delays, increased maintenance costs, and suboptimal resource utilization. Additionally, safety concerns, environmental impact, and the need for seamless passenger experiences further complicate operations. The lack of real-time data integration and advanced predictive analytics results in inefficient decision-making, affecting service quality and financial sustainability.

❖ Solution:-

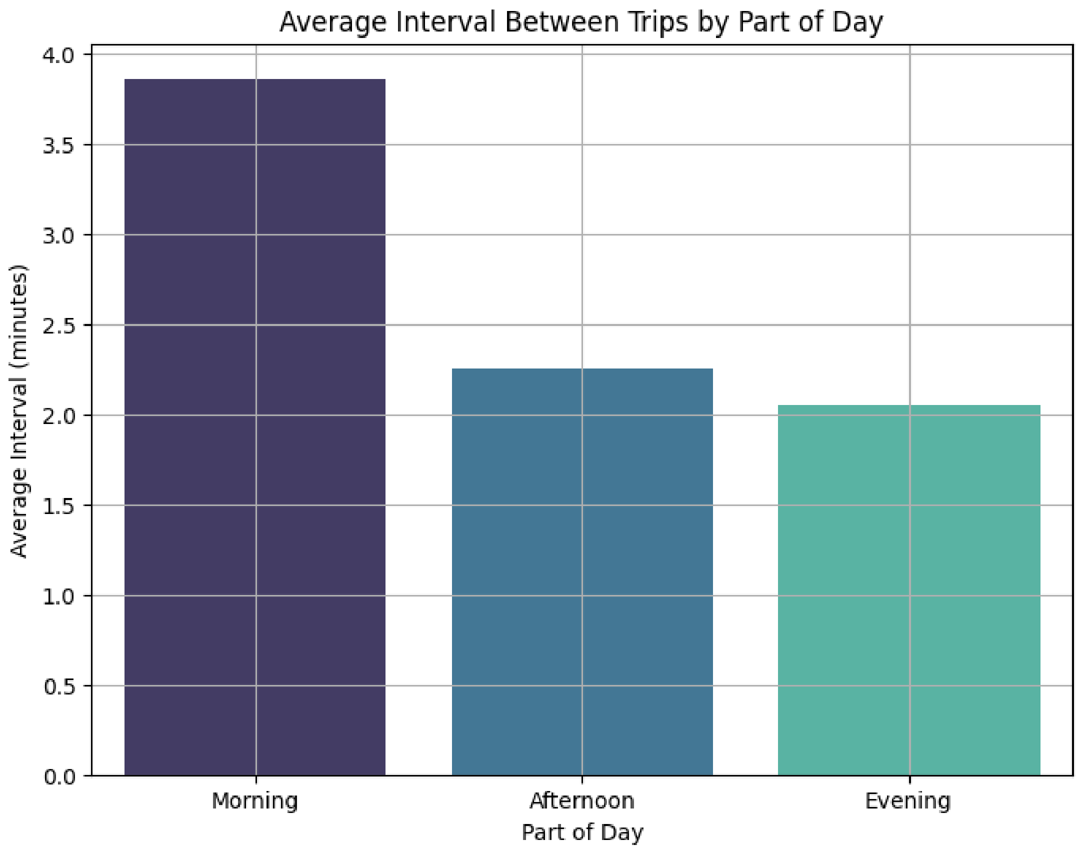
To address the challenges in metro operations, an **optimized metro operations framework** should leverage advanced technologies and data-driven strategies to enhance efficiency, reduce costs, and improve passenger satisfaction. Key solutions include:

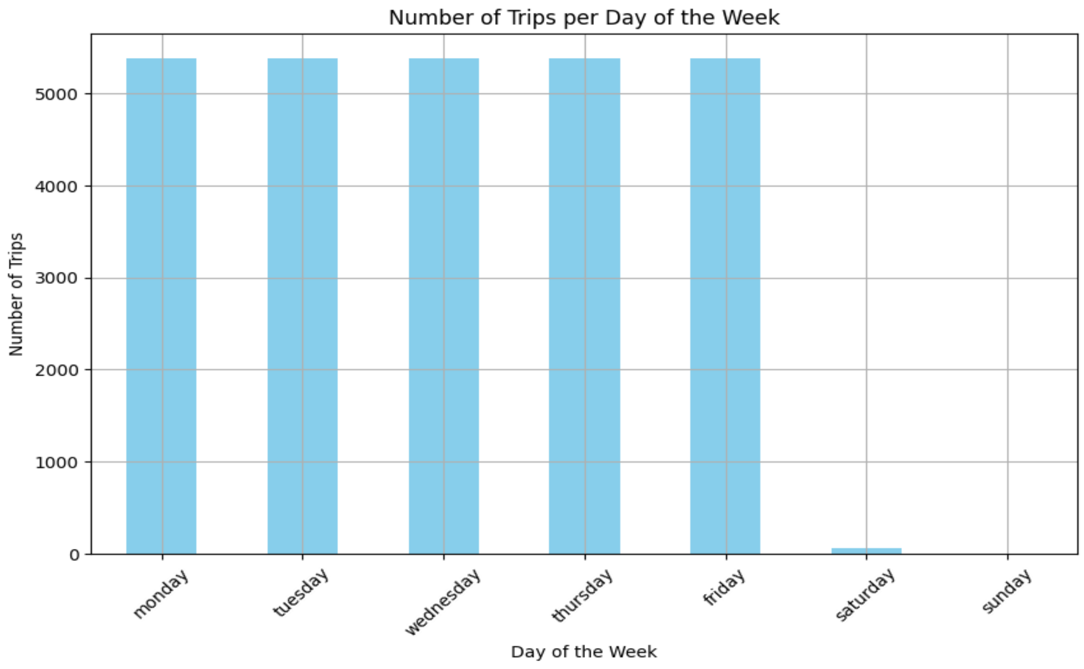
1. Smart Scheduling and Real-Time Optimization
2. Predictive Maintenance and Asset Management
3. Energy Efficiency and Sustainability
4. Automated and AI-Driven Operations
5. Data-Driven Decision Making

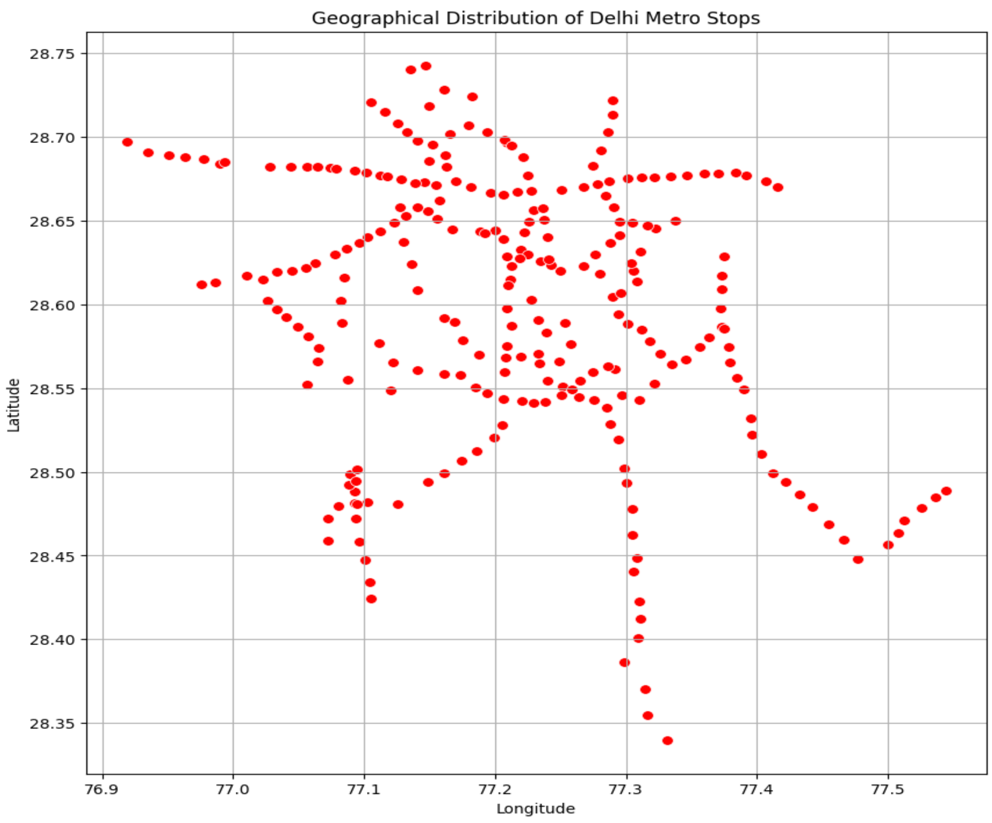
❖ Implementation:-

```
import pandas as pd

# load all the data files

agency = pd.read_csv('agency.txt')

calendar = pd.read_csv('calendar.txt')

routes = pd.read_csv('routes.txt')

shapes = pd.read_csv('shapes.txt')

stop_times = pd.read_csv('stop_times.txt')

stops = pd.read_csv('stops.txt')

trips = pd.read_csv('trips.txt')

# show the first few rows and the structure of each dataframe

data_overviews = {

    "agency": agency.head(),

    "calendar": calendar.head(),

    "routes": routes.head(),

    "shapes": shapes.head(),

    "stop_times": stop_times.head(),

    "stops": stops.head(),

    "trips": trips.head()

}

data_overviews

import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(10, 8))

sns.scatterplot(x='shape_pt_lon', y='shape_pt_lat', hue='shape_id', data=shapes, palette='viridis', s=10, legend=None)

plt.title('Geographical Paths of Delhi Metro Routes')

plt.xlabel('Longitude')

plt.ylabel('Latitude')

plt.grid(True)

plt.show()

plt.figure(figsize=(10, 10))

sns.scatterplot(x='stop_lon', y='stop_lat', data=stops, color='red', s=50, marker='o')

plt.title('Geographical Distribution of Delhi Metro Stops')

plt.xlabel('Longitude')

plt.ylabel('Latitude')

plt.grid(True)

plt.show()

# converting stop_times 'arrival_time' from string to datetime.time for easier manipulation

import datetime as dt

# function to convert time string to datetime.time

def convert_to_time(time_str):

    try:

        return dt.datetime.strptime(time_str, '%H:%M:%S').time()

    except ValueError:
        # Handle cases where the hour might be greater than 23 (e.g., 24:00:00 or 25:00:00)
        hour, minute, second = map(int, time_str.split(':'))
        return dt.time(hour % 24, minute, second)
stop_times['arrival_time_dt'] = stop_times['arrival_time'].apply(convert_to_time)
# calculate the difference in arrival times for subsequent trips at each stop
stop_times_sorted = stop_times.sort_values(by=['stop_id', 'arrival_time_dt'])
stop_times_sorted['next_arrival_time'] =
stop_times_sorted.groupby('stop_id')['arrival_time_dt'].shift(-1)
# function to calculate the difference in minutes between two times
def time_difference(time1, time2):
    if pd.isna(time1) or pd.isna(time2):
        return None
    full_date_time1 = dt.datetime.combine(dt.date.today(), time1)
    full_date_time2 = dt.datetime.combine(dt.date.today(), time2)
    return (full_date_time2 - full_date_time1).seconds / 60
stop_times_sorted['interval_minutes'] = stop_times_sorted.apply(lambda row:
time_difference(row['arrival_time_dt'], row['next_arrival_time']), axis=1)
# drop NaN values from intervals (last trip of the day)
stop_times_intervals = stop_times_sorted.dropna(subset=['interval_minutes'])
# average intervals by time of day (morning, afternoon, evening)
def part_of_day(time):
  if time < dt.time(12, 0):
        return 'Morning'
    elif time < dt.time(17, 0):
        return 'Afternoon'
    else:
        return 'Evening'
stop_times_intervals['part_of_day'] = 
stop_times_intervals['arrival_time_dt'].apply(part_of_day)
average_intervals = 
stop_times_intervals.groupby('part_of_day')['interval_minutes'].mean().reset_index()
plt.figure(figsize=(8, 6))
sns.barplot(x='part_of_day', y='interval_minutes', data=average_intervals, order=['Morning', 
'Afternoon', 'Evening'], palette='mako')
plt.title('Average Interval Between Trips by Part of Day')
plt.xlabel('Part of Day')
plt.ylabel('Average Interval (minutes)')
plt.grid(True)
plt.show()
```

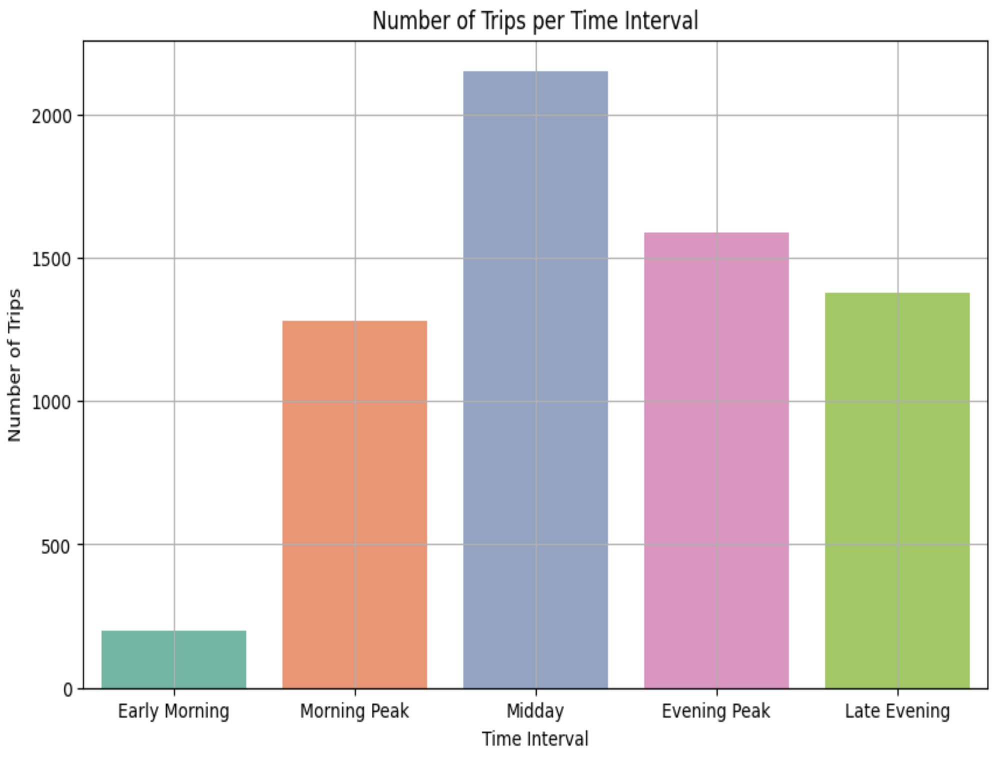
❖ Output:-


Average Interval Between Trips by Part of Day
Average Interval (minutes)
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Morning
Afternoon
Evening
Part of Day


Number of Trips per Day of the Week
Number of Trips
5000
4000
3000
2000
1000
0
monday
tuesday
wednesday
thursday
friday
saturday
sunday
Day of the Week



Geographical Distribution of Delhi Metro Stops
Latitude
Longitude
28.75
28.70
28.65
28.60
28.55
28.50
28.45
28.40
28.35
76.9
77.0
77.1
77.2
77.3
77.4
77.5



Number of Trips per Time Interval
Number of Trips
Time Interval
0
500
1000
1500
2000
Early Morning
Morning Peak
Midday
Evening Peak
Late Evening