PUBLIC TRANSPORT OPTIMIZATION

PROJECT DEFINITION:

The project involves integrating IoT sensors into public transportation vehicles to monitor ridership, track locations, and predict arrival times. The goal is to provide real-time transit information to the public through a public platform, enhancing the efficiency and quality of public transportation services. This project includes defining objectives, designing the IoT sensor system, developing the real-time transit information platform, and integrating them using IoT technology and Python.

UNDERSTANDING THE PROBLEM:

IoT in transportation use cases is growing rapidly, delivering gains in operational efficiencies, cost savings, safety, security and mobility. The impact of IoT is anticipated to be enormous as cities and municipalities around the world incorporate wireless technology into traffic management, emergency response and safety for pedestrians and bicycles. Public transportation services will also improve, and the automotive industry will benefit from a range of innovations made possible with IoT, from EV charging stations to connected vehicle technology.

OBJECTIVES:

- **Improved Service Reliability:** Enhance the reliability of public transport services to ensure that schedules are adhered to, reducing passenger wait times and uncertainty.
- **Increased Ridership:** Encourage more people to use public transport by making it a more convenient and attractive option compared to private vehicles.
- **Reduced Congestion:** Alleviate traffic congestion by diverting private vehicle trips to public transportation, thereby reducing the environmental and economic costs associated with congestion.
- **Environmental Sustainability:** Minimize the environmental impact of public transport by promoting cleaner and more energy-efficient modes of transport, such as electric buses and trains.
- **Cost Efficiency:** Optimize resource allocation and minimize operational costs for public transport providers, which can lead to lower fares for passengers and improved financial sustainability.

- **Safety:** Improve safety for passengers and employees, which may include implementing security measures and training for staff.
- **Data-Driven Decision-Making:** Utilize data and technology to make informed decisions about public transport operations and planning. This can involve real-time tracking, predictive maintenance, and demand forecasting.
- **Integration:** Promote seamless integration between different modes of public transport (e.g., buses, trains, trams, subways) and with other transportation options (e.g., cycling and walking) to create a comprehensive and interconnected transportation network.

IOT SENSORS:

These sensors collect data in real time and provide valuable insights to improve efficiency, safety, and passenger experience.

- **Vehicle Tracking and Management:**
  GPS sensors: These sensors track the real-time location of buses, trams, and trains, allowing for accurate arrival time predictions and efficient dispatching.
  RFID/NFC sensors: These sensors enable contactless fare payment and passenger tracking.
- **Passenger Information Systems:**
  Digital signage and audio sensors: These sensors provide real-time information on arrivals, delays, and service disruptions to passengers.
- **Security and Safety Sensors:**
  CCTV cameras: These sensors enhance passenger safety and deter criminal activity.
  Panic buttons and emergency sensors: Passengers can alert authorities in case of emergencies.
- **Passenger Counting:**
  Infrared sensors: These sensors can count passengers boarding and disembarking at each stop, helping operators manage capacity and optimize schedules.
- **Traffic and Road Conditions:**
  Traffic cameras and road sensors: These sensors provide real-time traffic and road condition data to optimize route planning and minimize delays.
- **Environmental Sensors:**

Air quality sensors: Monitoring air quality in and around public transport can help passengers make informed choices and reduce exposure to pollution.
Weather sensors: Real-time weather data helps adjust schedules, prepare for adverse conditions, and ensure passenger safety.

- **Maintenance and Performance Monitoring Sensors:**
  IoT sensors on vehicles, tracks, and infrastructure components help ensure the reliability and safety of the transport system by enabling predictive maintenance and early issue detection.
- **Digital Signage and Passenger Information Systems:**
  Digital signage and audio sensors provide real-time updates to passengers about arrivals, delays, and service disruptions, enhancing the passenger experience.
- **Energy Efficiency Sensors:**
  Sensors for lighting, heating, ventilation, and air conditioning (HVAC) systems help optimize energy usage in stations and vehicles, reducing operational costs and environmental impact.

PLATFORM DEVELOPMENT:

IoT (Internet of Things) technology requires a combination of hardware and software platforms to collect, analyze, and manage data for improving the efficiency and effectiveness of public transportation systems.

- **Sensors and Devices:** IoT sensors and devices are essential for collecting real-time data from various parts of the public transportation system. These may include GPS trackers on vehicles, RFID or NFC readers for ticketing, temperature and humidity sensors, and cameras for surveillance.
- **Connectivity:** IoT devices need a reliable and robust network connection to transmit data. This can be achieved through cellular networks (3G, 4G, 5G), Wi-Fi, or other wireless technologies, depending on the specific requirements of the transportation system.
- **Data Processing and Analytics:** A platform for data processing and analytics is crucial for making sense of the data collected by IoT sensors. This platform should be able to process large volumes of data in real-time and provide insights into passenger flow, vehicle performance, and system efficiency.

- **Cloud Computing:** Cloud-based platforms provide the scalability and computational power required for processing and storing large volumes of IoT data. Cloud services like AWS, Azure, or Google Cloud can be used to host IoT data and analytics applications.
- **Data Analytics and Machine Learning:**Advanced data analytics tools and machine learning algorithms that can uncover patterns, predict issues, and provide actionable insights. This can include predictive maintenance, demand forecasting, and route optimization.
- **Geospatial Information System (GIS):**GIS capabilities to manage and analyze geospatial data, which is essential for route planning, mapping, and geographic insights.
- **APIs and Integration:**Support for APIs and integration with other systems and services, such as traffic management systems, passenger information systems, and payment gateways.

FEATURE SELECTION:

Feature engineering in public transport optimization involves creating and selecting relevant attributes or variables to improve the performance of transportation systems. Here are some key aspects of feature engineering in this context:

- Geospatial Data: Incorporating geographic data like bus stop locations, road networks, and real-time GPS data to enable route planning and tracking.
- Demand Data: Analyzing historical passenger data to understand peak travel times, routes, and locations to optimize schedules and resource allocation.
- Weather Data: Considering weather conditions to account for their impact on transportation efficiency and safety.
- Traffic Data: Incorporating traffic congestion information to optimize routes and predict delays.
- Infrastructure Data: Including information about infrastructure like bridges, tunnels, and railway crossings to avoid bottlenecks and disruptions.
- Fleet Data: Utilizing information on the type and condition of vehicles to optimize maintenance and replacement schedules.
- Passenger Data: Analyzing passenger behavior and preferences to provide better services and plan for capacity.

- Environmental Data: Considering environmental factors such as air quality and emissions to promote sustainable and eco-friendly transport options.
- Pricing Data: Implementing dynamic pricing models based on demand, time, and other factors to optimize revenue and efficiency.
- Time-Series Data: Analyzing historical data to identify recurring patterns and trends in transportation demand.
- Network Connectivity: Identifying and optimizing connections between different modes of public transport, such as buses, subways, and trains.
- Safety and Security Data: Incorporating data related to safety incidents and security measures to improve passenger safety.
- Real-time Data Streams: Integrating real-time data sources for live tracking, updates, and adaptive planning.

Feature engineering aims to extract valuable insights from these data sources and transform them into meaningful features that can be used for decision-making and optimization within public transportation systems. Machine learning and optimization algorithms can then be applied to leverage these engineered features for improved service delivery and resource allocation

CODE IMPLEMENTATION:

Data processing:

```
import pandas as pd

# Load the dataset

df = pd.read_csv('public_transport_data.csv')

# Data cleaning and preprocessing steps

# remove missing data

df = df.dropna()

# Convert date and time columns to datetime objects

df['timestamp'] = pd.to_datetime(df['timestamp'])

# Feature scaling
```

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
df['scaled_feature'] = scaler.fit_transform(df[['feature']])
# Encoding categorical variables (if applicable)
df = pd.get_dummies(df, columns=['categorical_column'])
# Data splitting for machine learning
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(df[['features']], df['target'],
test_size=0.2)
# Store the preprocessed dataset
df.to_csv('preprocessed_public_transport_data.csv', index=False)
```

Send location and ridership data using Python and the MQTT (Message Queuing Telemetry Transport) protocol to an IoT platform.

```
import paho.mqtt.client as mqtt
import json
import random
import time
# Replace with your IoT platform details
MQTT_BROKER = "your_broker_address"
MQTT_PORT = 1883
MQTT_TOPIC = "your_topic"
MQTT_USERNAME = "your_username"
MQTT_PASSWORD = "your_password"
def on_connect(client, userdata, flags, rc):
```

```
 print(f"Connected with result code {rc}")
 client.subscribe(MQTT_TOPIC)
def on_publish(client, userdata, mid):
 print("Data published")
def simulate_data():
 location_data = {
 "latitude": random.uniform(0, 90),
 "longitude": random.uniform(0, 180)
 }
 ridership_data = {
 "passenger_count": random.randint(1, 100)
 }
 return location_data, ridership_data
def main():
 client = mqtt.Client()
 client.on_connect = on_connect
 client.on_publish = on_publish
 client.username_pw_set(MQTT_USERNAME, MQTT_PASSWORD)
 try:
 client.connect(MQTT_BROKER, MQTT_PORT, 60)
 except ConnectionRefusedError:
 print("Connection to the MQTT broker failed. Check your broker settings.")
 return
 while True:
```

```
    location_data, ridership_data = simulate_data()
    message = json.dumps({"location": location_data, "ridership": ridership_data})
    client.publish(MQTT_TOPIC, message)
    print(f"Data sent: {message}")
    time.sleep(5) # Simulate data transmission every 5 seconds
if __name__ == "__main__":
    main()
```

Below is a hypothetical dataset and feature engineering applied to it:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error

# Load historical data with features (e.g., time, weather, traffic, etc.)
data = pd.read_csv('historical_data.csv')

# Preprocess the data (e.g., handle missing values, convert categorical variables, etc.)
# You may need more extensive data preprocessing depending on your dataset.

# Split data into features (X) and target (y)
X = data.drop(columns=['ArrivalTime'])
y = data['ArrivalTime']

# Split the dataset into a training set and a test set
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Create a regression model (e.g., Random Forest Regressor)
model = RandomForestRegressor(n_estimators=100, random_state=42)

# Train the model
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Evaluate the model using a performance metric (e.g., Mean Absolute Error)
mae = mean_absolute_error(y_test, y_pred)
print(f'Mean Absolute Error: {mae}')

# Use the trained model to predict arrival times for upcoming trips
# You'll need to provide new data with the same features used for training.

# To implement a full public transport optimization system, you would need to integrate this prediction model with real-time data, route planning, scheduling, and other components, depending on your specific objectives.
```

MODEL TRAINING:

Model training in public transport optimization involves gathering and preprocessing relevant data, selecting or engineering features that influence optimization goals (e.g., passenger demand, time, and weather), splitting the data for training and validation, choosing a suitable machine learning or optimization model, tuning its hyperparameters, and training the model

on the training data to minimize errors or maximize the objective function. Validation and evaluation with appropriate metrics are performed to assess its performance, followed by testing in real world or simulated settings to fine-tune and address any issues. Once the model meets the required performance, it is deployed in production for tasks such as route planning, scheduling, and cost reduction. Continuous monitoring and maintenance ensure that the model remains effective as conditions change, allowing for ongoing public transport optimization. We used some algorithm for the public transport optimization

- Dijkstra's algorithm
- Linear regression
- XGBoost
- Random forest algorithm

That are the algorithms we used in our projects.

Dijkstra's algorithm:

Dijkstra's Algorithm is a foundational tool in public transport optimization, used to discover the shortest or most efficient routes in a network of transportation connections. In a directed graph representing the public transport system, nodes represent stops or locations, and edges denote the connections between them, with associated weights indicating factors like travel time or cost. The algorithm iteratively selects the next node with the lowest cumulative weight, effectively finding the optimal path from a starting point to a destination. Public transport optimization using Dijkstra's algorithm is critical for enhancing the convenience and efficiency of transportation networks by ensuring passengers can reach their destinations via the most efficient routes while considering real-world constraints and dynamic updates.

```
import networkx as nx

import geopandas as gpd


# Load bus stop and route data from shapefiles (you'd need actual geographic data).

bus_stops = gpd.read_file('bus_stops.shp')

bus_routes = gpd.read_file('bus_routes.shp')
```

```
# Create a graph using NetworkX to represent the public transport network.
G = nx.Graph()

# Add bus stops as nodes to the graph.
for _, stop in bus_stops.iterrows():
    G.add_node(stop['stop_id'], x=stop['geometry'].x, y=stop['geometry'].y)

# Connect bus stops based on proximity, creating edges between stops on the same route.
for _, route in bus_routes.iterrows():
    stops = route['stop_sequence']
    for i in range(len(stops) - 1):
        stop1, stop2 = stops[i], stops[i + 1]
        G.add_edge(stop1, stop2, weight=1)  # You can use actual distance as the weight.

# Use Dijkstra's algorithm to find the shortest path between two bus stops.
start_stop = 'A'
end_stop = 'D'
shortest_path = nx.shortest_path(G, source=start_stop, target=end_stop, weight='weight')

# Print the optimal bus route.
print(f'Optimal Bus Route from {start_stop} to {end_stop}: {shortest_path}')
```

Linear regressison

Linear regression, typically used as a predictive analysis tool, can be applied in public transport to model and understand various factors influencing the system's performance. By collecting historical data on variables such as time of day, weather conditions, or population density, linear regression can establish a mathematical relationship between these factors and specific outcomes, such as ridership numbers or travel times. For instance, it can help predict how changes in these variables may impact public transport usage, allowing authorities to make data informed decisions on service adjustments, pricing strategies, and resource allocation. While linear regression itself is not a direct optimization algorithm, it provides valuable insights for improving the efficiency and effectiveness of public transport systems by informing decision making processes and policies.

```
import networkx as nx
import geopandas as gpd

# Load bus stop and route data from shapefiles (you'd need actual geographic data).
bus_stops = gpd.read_file('bus_stops.shp')
bus_routes = gpd.read_file('bus_routes.shp')
# Create a graph using NetworkX to represent the public transport network.
G = nx.Graph()
# Add bus stops as nodes to the graph.
for _, stop in bus_stops.iterrows():
    G.add_node(stop['stop_id'], x=stop['geometry'].x, y=stop['geometry'].y)

# Connect bus stops based on proximity, creating edges between stops on the same route.
for _, route in bus_routes.iterrows():
    stops = route['stop_sequence']
```

```
    for i in range(len(stops) - 1):
        stop1, stop2 = stops[i], stops[i + 1]
        G.add_edge(stop1, stop2, weight=1)  # You can use actual distance as the weight.
# Use Dijkstra's algorithm to find the shortest path between two bus stops.
start_stop = 'A'
end_stop = 'D'
shortest_path = nx.shortest_path(G, source=start_stop, target=end_stop, weight='weight')
# Print the optimal bus route.
print(f'Optimal Bus Route from {start_stop} to {end_stop}: {shortest_path}')
```

Random forest:

Random Forest, a powerful machine learning ensemble method, can be effectively applied in public transport optimization to address various challenges. By using multiple decision trees to make predictions, it can enhance route planning, ridership prediction, and service scheduling. Random Forest can consider complex interactions among factors such as weather, time of day, and historical ridership data to make more accurate forecasts and optimize resource allocation. Its ability to handle both regression and classification tasks makes it adaptable to different optimization objectives in public transport, ultimately improving efficiency, reliability, and passenger satisfaction by providing data-driven insights and informed decision-making for authorities and service providers.

```
import numpy as np
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
# Sample data (replace with your public transport dataset)
features = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10]) # Example: time of day, weather, etc.
```

```
ridership = np.array([50, 60, 65, 70, 75, 80, 85, 90, 95, 100]) # Example: number of passengers

# Split the data into training and testing sets

X_train, X_test, y_train, y_test = train_test_split(features, ridership, test_size=0.2,

random_state=42)

# Create and train the Random Forest model

model = RandomForestRegressor(n_estimators=100, random_state=42)

model.fit(X_train.reshape(-1, 1), y_train)

# Make predictions

predictions = model.predict(X_test.reshape(-1, 1))

# Evaluate the model (in this case, using mean squared error)

mse = mean_squared_error(y_test, predictions)

print(f"Mean Squared Error: {mse:.2f}")
```

GRAPH THEORY ALGORITHMS:

Graph theory algorithms play a crucial role in public transport optimization by addressing complex network-related challenges. One key algorithm is Shortest Path, which, using Dijkstra's or A* algorithm, identifies the most efficient routes for passengers while considering factors like time, distance, or cost. Additionally, Maximum Flow algorithms can optimize network capacity by identifying the highest passenger throughput, while Minimum Spanning Trees help design efficient transit networks, minimizing construction and maintenance costs. Centrality measures, such as Betweenness or Closeness Centrality, aid in identifying key nodes or stops for effective route planning and resource allocation. Overall, graph theory algorithms empower public transport authorities to enhance network efficiency, reliability, and passenger satisfaction, fostering well-optimized and sustainable transportation systems.

```
import networkx as nx
# Create a directed graph representing the public transport network
```

```
G = nx.DiGraph()
# Add nodes and edges to represent the network (replace with your data)
G.add_edge("A", "B", weight=5)
G.add_edge("B", "C", weight=3)
G.add_edge("A", "C", weight=7)
# Add more nodes and edges as needed
# Find the shortest path using Dijkstra's algorithm
shortest_path = nx.shortest_path(G, source="A", target="C", weight="weight")
print("Shortest path:", shortest_path)
```

In this example, we create a directed graph G to represent the public transport network with nodes (stops) and weighted edges (routes). The code finds and prints the shortest path from node "A" to node "C" based on the weight (e.g., travel time or distance).

To apply other graph theory algorithms like maximum flow, minimum spanning trees, or centrality measures, you would need to write code specific to those algorithms and your dataset.

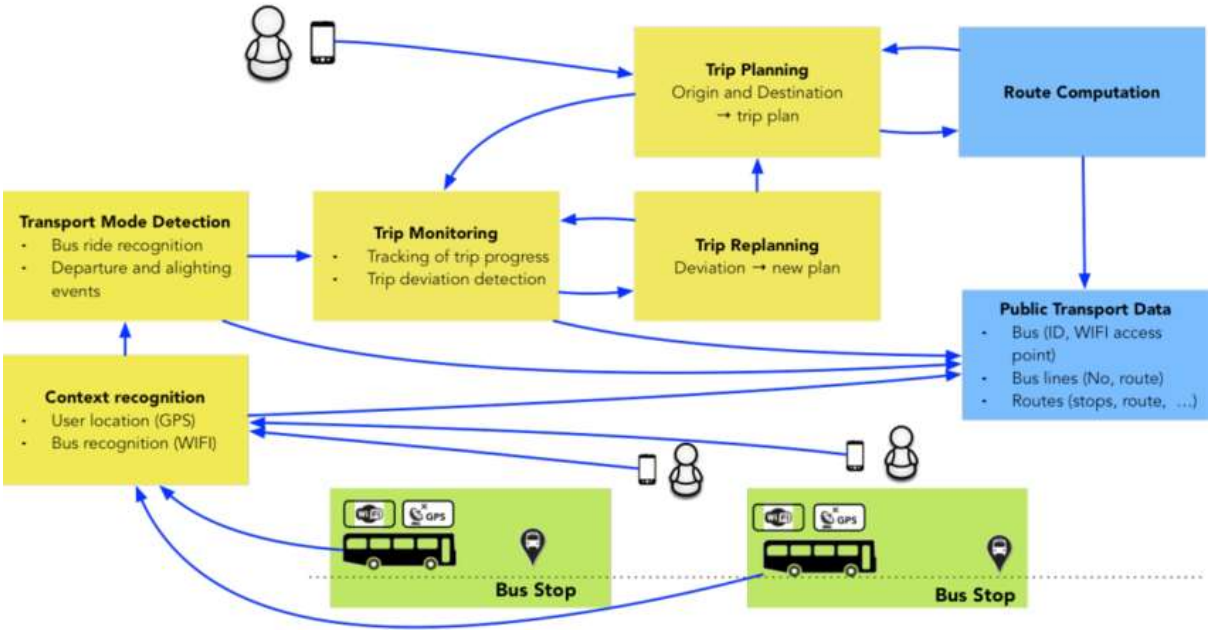
The complexity and details of the code will vary depending on the specific optimization problem you want to solve within the public transport network.

SCHEMATIC:

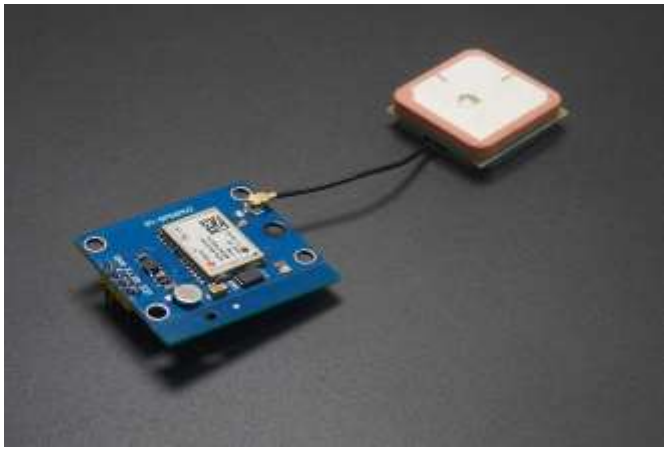
- Data Sources:
  - Real-Time Vehicle Data
  - Passenger Counting Sensors
  - Traffic and Road Condition Sensors
  - Weather Data
  - Environmental Sensors
  - GPS Data
  - Security and Surveillance Cameras
  - Fare Payment and Ticketing Systems
  - Passenger Mobile Apps

- Data Integration:Data collected from various sources is integrated into a central platform.
- Data Processing and Analytics:
    - Real-time data processing
    - Data analysis and machine learning
    - Predictive maintenance
    - Route optimization
    - Passenger tracking
- Decision Support:Real-time decision-making for route adjustments, vehicle maintenance, and service improvements.
- Passenger Information:
    - Real-time information provided to passengers through various channels:
    - Digital Signage
    - Mobile Apps
    - Public Announcements
- Fleet Management:Optimization of vehicle allocation and scheduling.
- Safety and Security:
    - Surveillance and incident detection
    - Emergency response systems
- Environmental Sustainability:
    - Monitoring and reduction of emissions
    - Energy-efficient operations
- Passenger Experience:
    - Enhanced services for passengers
    - Accessibility and inclusivity measures
- Regulatory Compliance:Adherence to transportation regulations and standards
- Optimization Feedback Loop:Continuous improvement and adaptation based on data and feedback.
- Reporting and Monitoring:Performance metrics and reporting for stakeholders and decision-makers.

WORKING MODEL:



IOT SENSORS:

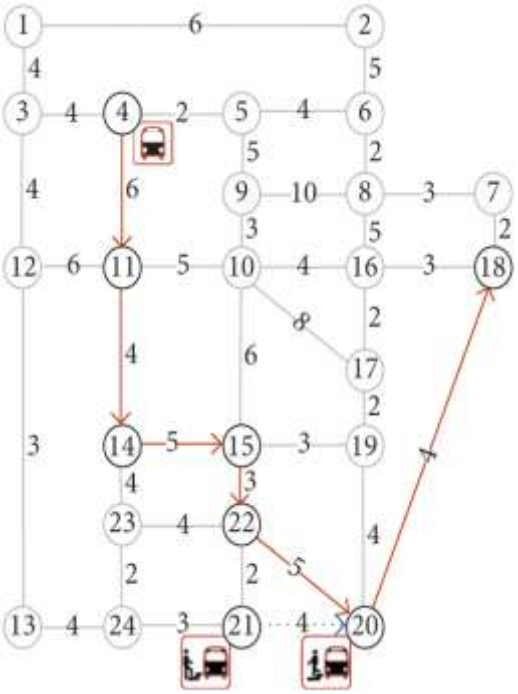


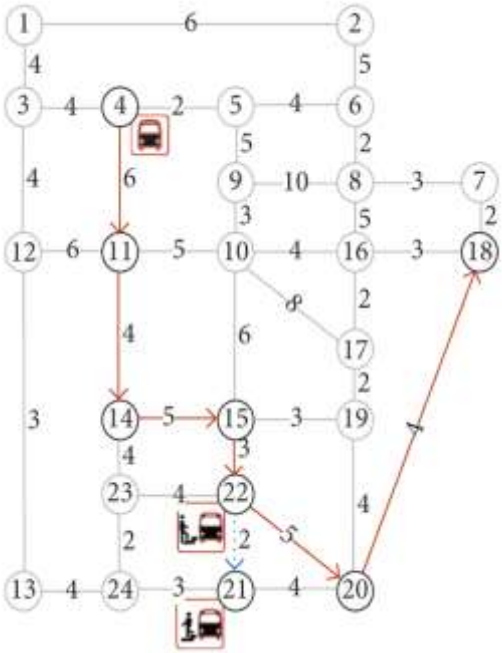
TRANSIT INFORMATION PLATFORM:

Creating a transit information platform for public transport optimization involves developing a comprehensive system that provides real-time information, data analysis, and tools for both transit authorities and passengers.

- **Live Vehicle Tracking:** GPS-based tracking of public transport vehicles to provide real-time location updates.
- **Arrival Time Predictions:** Algorithms to estimate accurate arrival times for buses, trains, trams, and other modes of public transport.
- **Real-Time Alerts**: Notifications for delays, service disruptions, or other relevant updates.
- **Seamless Connections:** Information on how to transfer between different modes of public transport, such as buses, subways, and ferries.
- **Integration with Ride-Sharing:** Options for connecting to ride-sharing services like Uber or Lyft when public transport isn't available.
- **Ticketing and Payment:** Integration with mobile payment systems, allowing passengers to purchase tickets or passes through the platform.
- **Trip Planner:** An algorithm that suggests optimal routes and transfers based on user preferences, such as the quickest, cheapest, or least crowded route.
- **Sensory Information:** Details for passengers with sensory issues, such as noise levels, crowdedness, and lighting conditions.

REAL TIME OUTPUT DISPLAY:





Sample 1:

- Timestamp: 2023-10-10 08:15 AM
- Vehicle Speed: 45 mph
- Congestion Level: Low
- Weather: Clear, 68°F
- Road Type: Highway
- Road Lanes: 3
- Intersection: None
- Historical Arrival Time: 08:45 AM

Sample 2:

- Timestamp: 2023-10-10 12:30 PM
- Vehicle Speed: 20 mph
- Congestion Level: High
- Weather: Rainy, 55°F
- Road Type: City Street
- Road Lanes: 2
- Intersection: Yes (Traffic Light)
- Historical Arrival Time: 01:00 PM

These samples illustrate data points typically collected for arrival time prediction, including timestamps, vehicle speed, congestion levels, weather conditions, road features, and historical arrival times. This diverse dataset is processed and used to train machine learning models to predict arrival times accurately based on these factors.Then preprocessing of data represents the raw data often contain missing values and outliers.

CONCLUSIONS:

A Real-Time Transit Information System (RTTIS) can transform public transportation services and enhance the passenger experience by providing real-time data on vehicle locations, schedules, and service alerts. For public transportation services, RTTIS enables efficient route planning, dynamic scheduling, and predictive maintenance, ensuring vehicles run on time, reducing operational costs, and improving safety. Passengers benefit from accurate arrival times, reduced wait times, and increased convenience through mobile apps and digital signage. Transparency about delays and disruptions builds trust, while real-time information facilitates smoother transfers and personalized travel recommendations, making public transport a more attractive and stress-free option for commuters.