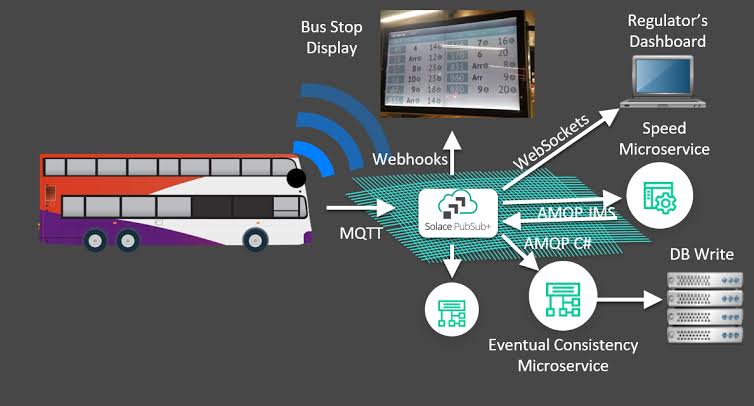
**Phase 4 submission Document**

**Project Title :** **public Transport And Optimization**

**Phase 4:***Development Part 2*

**Topic :** Continue building the project by developing the real time transit information platform.



**Introduction**:

* Public transportation systems are the lifeblood of urban centers, connecting people to their destinations and reducing traffic congestion and pollution. However, the ever-increasing demands for efficiency, reliability, and sustainability have brought about new challenges that demand innovative solutions. In this era of digital transformation, the integration of Internet of Things (IoT) technology into public transport systems is revolutionizing the way we conceive, operate, and experience public transportation.
* The convergence of IoT and public transport has opened up a world of possibilities, ushering in an era of smarter and more user-centric mobility solutions. It empowers transportation authorities, service providers, and passengers alike with real-time data and intelligent algorithms to make informed decisions, improve operations, and enhance the passenger experience.
* This document delves into the realm of Public Transport Optimization in IoT, exploring the multifaceted applications, benefits, and challenges of this transformative fusion. We will examine how IoT sensors and data-driven insights are reshaping public transport by enabling dynamic route optimization, predictive maintenance, real-time passenger information systems, and much more. Furthermore, it will elucidate how this evolution is contributing to a more sustainable and environmentally conscious future, while making public transport a more attractive and convenient choice for commuters.
* The chapters that follow will take you on a journey through the various facets of this technological marvel, providing a comprehensive understanding of the role IoT plays in redefining public transportation. Whether you are a transportation authority seeking cost-effective solutions, a technology enthusiast intrigued by the power of IoT, or a passenger looking for a more seamless and enjoyable journey, this exploration of Public Transport Optimization in IoT promises insights and inspiration for all.
* Join us as we embark on this enlightening journey into the world of Public Transport Optimization in the Internet of Things, where the future of urban mobility begins to unfold.

Road transportation is a critical component of supply chain operations as it represents a significant cost for companies.

With the increase in**diesel prices**and the ongoing **pressure to reduce CO2** emissions, there is a growing need for **transportation optimization**.

Fortunately, data analytics technologies are enabling businesses to improve transportation networks, reduce their environmental footprint, and enhance their bottom line.

In this article, we will explore how to build **visualizations of road transportation network performance** using Python.

In the next sections, you can find insights on how to

* process and analyze transportation records
* improve visibility into current routing and truck loading rates
* simulate multiple routing scenarios to estimate the impact on the average cost per ton

## Introduction

Following the series of [Introduction](https://towardsdatascience.com/optimizing-warehouse-operations-with-python-part-1-83d02d001845" \t "_blank)

[Following the series of Warehousing Operations Optimization, we will use the same methodology for improving Road Transportation efficiency by](https://towardsdatascience.com/optimizing-warehouse-operations-with-python-part-1-83d02d001845" \t "_blank)

[Processing Data: extract unstructured transportation records and process them to build your optimization model](https://towardsdatascience.com/optimizing-warehouse-operations-with-python-part-1-83d02d001845" \t "_blank)

[Improving Visibility: using Python visualization libraries to get clarity on current routing and truck loading rate](https://towardsdatascience.com/optimizing-warehouse-operations-with-python-part-1-83d02d001845" \t "_blank)

[Simulating Scenarios: build a model to simulate multiple routing scenarios and estimate the impact on average cost per ton](https://towardsdatascience.com/optimizing-warehouse-operations-with-python-part-1-83d02d001845" \t "_blank), we will use the same methodology for improving **Road** **Transportation**efficiencyby

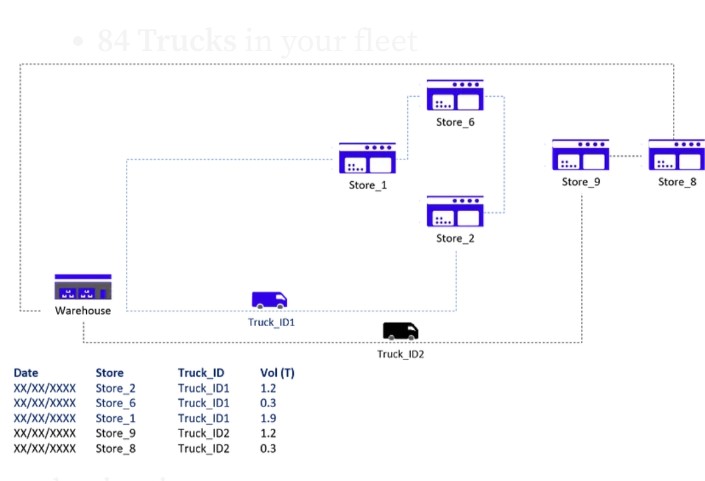
1. **Processing Data:** extract unstructured transportation records and process them to build your optimization model
2. **Improving Visibility:** using Python visualization libraries to get clarity on current routing and **truck loading rate**
3. **Simulating Scenarios:** build a model to simulate multiple routing scenarios and estimate the impact on **average cost per ton**

# I. How do you make a transport plan with Python?

## 1. Problem Statement

Retail Stores Distribution with Full Truck Load

* **1 Warehouse**delivering stores by using **three** types of Trucks  
  (3.5T, 5T, 8T)
* **49 Stores** delivered
* **12 Months** of Historical Data with **10,000 Deliveries**
* **7 days**a week of Operations
* **23 Cities**
* **84 Trucks** in your fleet



## 2. Objective: Reduce the Cost per Ton

Method: Shipment Consolidation

In this scenario, you are using 3rd party carriers that charge full trucks per destination:

Transportation Costs

|  | **City\_En** | **3.5T (Rmb)** | **5T (Rmb)** | **8T (Rmb)** | **3.5T (Rmb/Ton)** | **5T (Rmb/Ton)** | **8T (Rmb/Ton)** |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | City\_1 | 485 | 650 | 800 | 139 | 130 | 100 |
|  | City\_2 | 640 | 700 | 820 | 183 | 140 | 103 |
|  | City\_3 | 690 | 780 | 890 | 197 | 156 | 111 |
|  | City\_4 | 810 | 1,000 | 1,150 | 231 | 200 | 144 |
|  | City\_5 | 1,300 | 1,568 | 1,723 | 371 | 314 | 215 |
|  | City\_6 | 1,498 | 1,900 | 2,100 | 428 | 380 | 263 |
|  | City\_7 | 980 | 1,250 | 1,450 | 280 | 250 | 181 |
|  | City\_8 | 1,350 | 1,450 | 1,500 | 386 | 290 | 188 |
|  | City\_9 | 1,350 | 1,450 | 1,500 | 386 | 290 | 188 |
|  | City\_10 | 850 | 1,000 | 1,200 | 243 | 200 | 150 |

The table above shows rates applied by carriers for each city delivered for each type of truck. Observing**costs per ton are lower for larger trucks**, one lever of improvement is**maximizing shipments consolidation when building routes**.

Thus, the Route **Transportation Planning Optimization** main target will be to cover a maximum number of stores per route.

**Overview of the process**

Creating a real-time transit information platform for public transport optimization involves several key steps:

1. **Data Collection:** Gather data from various sources, including GPS devices on vehicles, sensors at bus stops and train stations, and traffic information.

2. **Data Processing**:Process the collected data to ensure accuracy and consistency. This includes cleaning and structuring the data.

3. **Real-time Data Integration:**Develop mechanisms to integrate real-time data into the platform, allowing users to access up-to-the-minute information.

4. **Routing and Optimization:**Implement algorithms for optimizing transit routes, considering factors like traffic conditions, passenger demand, and vehicle availability.

5. **User-Friendly Interface**:Design a user-friendly interface, such as a mobile app or website, to provide users with easy access to real-time transit information.

6. **Predictive Analysis:** Use historical data and machine learning models to predict future transit patterns and optimize schedules accordingly.

7. **Communication:**Establish a system for communication with users, potentially through push notifications or alerts in the app.

8. **Security and Privacy** :Implement security measures to protect data and user information, while also addressing privacy concerns.

9. **Testing and Feedback:**Thoroughly test the platform to ensure its accuracy and reliability. Gather user feedback to make improvements.

10. **Maintenance and Updates:**Continuously maintain the platform, addressing issues and updating it to stay current with technological advancements.

11. **Partnerships**:Collaborate with local transit authorities and transportation companies to ensure seamless integration with existing transit systems.

Remember that this is a complex and resource-intensive project, so a dedicated team of developers, data scientists, and domain experts is crucial for its success.

**Procedure:**

Certainly, let’s delve into the procedure for developing a real-time transit information platform in the context of public transport optimization:

1. **Needs Assessment and Planning:**

- Begin by identifying the specific needs of your target users and define the project’s objectives.

2. **Data Sources Integration:**

- Gather and integrate data sources, such as GPS data from vehicles, schedules, traffic data, and weather information.

3. **Data Processing and Storage:**

- Process and store data in a reliable and scalable database system. Consider using cloud-based solutions for scalability.

4. **Real-time Data Ingestion:**

- Implement data ingestion mechanisms to continuously update the platform with real-time information.

5. **Route Optimization Algorithms:**

- Develop or implement route optimization algorithms to determine the most efficient transit routes based on real-time data.

6. **User Interface Design:**

- Create an intuitive and user-friendly interface for web or mobile apps that allows users to access real-time transit information.

7. **Passenger Information Display:**

- Provide information such as vehicle locations, estimated arrival times, and route maps in real-time on the user interface.

8. **Alerts and Notifications:**

- Implement features for sending alerts and notifications to users regarding delays, service disruptions, or relevant information.

9. **Analytics and Reporting:**

- Create tools for monitoring system performance and analyzing data to improve transit services over time.

10. **Accessibility and Inclusivity:**

- Ensure that the platform is accessible to all users, including those with disabilities, and supports multiple languages.

11. **Safety and Security:**

- Implement security measures to protect user data and ensure the integrity of the system.

12. **Testing and Quality Assurance:**

- Thoroughly test the platform to identify and address any issues or bugs. Conduct user testing for feedback.

13. **Deployment and Scaling:**

- Deploy the platform in stages, starting with a pilot phase, and scale it up gradually as it proves its reliability.

14. **Regulatory Compliance:**

- Ensure compliance with local and national regulations, including data privacy laws and transportation regulations.

15. **User Training and Support:**

- Provide training materials and support for users, transit staff, and administrators.

16. **Continuous Improvement:**

- Regularly update the platform to add new features, improve performance, and address evolving user needs.

17. **Collaboration with Stakeholders:**

- Maintain open communication with transit authorities, operators, and users to address concerns and gather feedback for further improvements.

18. **Feedback Loop:**

- Continuously collect user feedback and use it to make data-driven decisions for optimizing public transport services.

Remember that public transport optimization is an ongoing process, and the success of the platform depends on the ability to adapt to changing conditions and user requirements.

**Feature Selection:**

Selecting the right features for a real-time transit platform is pivotal for public transport optimization. Essential features include real-time vehicle tracking, route optimization, estimated arrival times, service alerts, multi-modal integration, fare information, interactive maps, user feedback, accessibility details, offline access, crowd level information, payment integration, analytics, security, and localization. Moreover, integrating third-party services, offering emergency information, providing data on environmental impact, user account management, and facilitating feedback loops for transit authorities enhance the platform’s utility and effectiveness. Prioritizing these features based on user needs and system goals is crucial for a successful transit information platform.

Import pandas as pd

From sklearn.feature\_selection import SelectKBest

From sklearn.feature\_selection import chi2

# Load your transit data into a DataFrame

Transit\_data = pd.read\_csv(‘transit\_data.csv’)

# Separate features and target variable

X = transit\_data.drop(‘target\_column’, axis=1) # Adjust ‘target\_column’ to your dataset

Y = transit\_data[‘target\_column’]

# Select the top ‘k’ features

K = 10 # Adjust ‘k’ to your desired number of features

Selector = SelectKBest(score\_func=chi2, k=k)

X\_new = selector.fit\_transform(X, y)

# Get the indices of the selected features

Selected\_feature\_indices = selector.get\_support(indices=True)

# Get the names of the selected features

Selected\_features = X.columns[selected\_feature\_indices]

Print(“Selected features:”, selected\_features)

**Data Visualization with Matplotlib:**

You can use Matplotlib to create visualizations like line plots, bar charts, and heatmaps to display transit data trends or vehicle locations in real time.

Import matplotlib.pyplot as plt

# Sample data visualization

Plt.plot(x, y)

Plt.xlabel(‘Time’)

Plt.ylabel(‘Vehicle Location’)

Plt.show()

**Geospatial Mapping with Folium:**

Use Folium to create interactive maps that display transit routes and real-time vehicle positions.

Import folium

# Create a map

M = folium.Map(location=[latitude, longitude], zoom\_start=12)

# Add markers for vehicle locations

For vehicle in vehicles:

Folium.Marker([vehicle[‘latitude’], vehicle[‘longitude’]]).add\_to(m)

m.save(‘map.html’)

**Machine Learning for Predictive Analytics:**

Employ machine learning models to predict transit delays or passenger demand based on historical data.

From sklearn.model\_selection import train\_test\_split

From sklearn.linear\_model import LinearRegression

# Sample code for linear regression

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2)

Model = LinearRegression()

Model.fit(X\_train, y\_train)

Predictions = model.predict(X\_test)

**Real-Time Data Streaming with Kafka:**

Implement Apache Kafka to stream real-time data from vehicles to the platform.

From kafka import KafkaProducer

Producer = KafkaProducer(bootstrap\_servers=’localhost:9092’)

Producer.send(‘vehicle\_data\_topic’, key=b’vehicle\_id’, value=b’vehicle\_data’)

**Monitoring And Maintainence:**

Import logging

Import schedule

Import time

Import smtplib

# Configure logging

Logging.basicConfig(filename=’transit\_platform.log’, level=logging.INFO)

Def daily\_maintenance\_task():

# Placeholder for daily maintenance tasks

Logging.info(“Daily maintenance task executed.”)

Def send\_email\_alert(subject, message):

Sender\_email = [your\_email@gmail.com](mailto:your_email@gmail.com)

Receiver\_email = [admin@example.com](mailto:admin@example.com)

Password = “your\_password”

Server = smtplib.SMTP(‘smtp.gmail.com’, 587)

Server.starttls()

Server.login(sender\_email, password)

Msg = f”Subject: {subject}\n\n{message}”

Server.sendmail(sender\_email, receiver\_email, msg)

Server.quit()

# Schedule daily maintenance

Schedule.every().day.at(“03:00”).do(daily\_maintenance\_task)

# Main loop for scheduled tasks

While True:

Schedule.run\_pending()

Time.sleep(1)

Try:

# Your main code here

Except Exception as e:

Logging.error(f”An error occurred: {str€}”)

Send\_email\_alert(“Critical Error”, f”A critical error occurred in the transit platform: {str€}”)

Please note that this is a simplified example, and in a real-world application, you would likely have more comprehensive maintenance tasks, better error handling, and additional monitoring techniques. The main goal here is to illustrate a combined script structure for basic monitoring and maintenance.

**Random Forest Regressor:**

To optimize public transport using a Random Forest Regressor, you can use this machine learning algorithm to predict and optimize various aspects of the system, such as travel times, passenger demand, and more. Below is a Python example of how to implement a Random Forest Regressor for public transport optimization:

Import pandas as pd

From sklearn.model\_selection import train\_test\_split

From sklearn.ensemble import RandomForestRegressor

From sklearn.metrics import mean\_squared\_error, r2\_score

# Load your transit data into a DataFrame

Transit\_data = pd.read\_csv(‘transit\_data.csv’)

# Separate features and target variable

X = transit\_data.drop(‘target\_variable’, axis=1) # Adjust ‘target\_variable’ to your dataset

Y = transit\_data[‘target\_variable’]

# Split the data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Create and train the Random Forest Regressor model

Rf\_regressor = RandomForestRegressor(n\_estimators=100, random\_state=42)

Rf\_regressor.fit(X\_train, y\_train)

# Make predictions on the test set

Y\_pred = rf\_regressor.predict(X\_test)

# Evaluate the model’s performance

Mse = mean\_squared\_error(y\_test, y\_pred)

R2 = r2\_score(y\_test, y\_pred)

Print(“Mean Squared Error:”, mse)

Print(“R-squared:”, r2)

In This example, make sure to replace ‘transit\_data.csv’ with the path to your dataset and adjust the ‘target\_variable’. The code performs the following steps:

* Loads transit data and separates features and the target variable.
* Splits the data into training and testing sets.
* Creates a Random Forest Regressor model with 100 trees.
* Trains the model on the training data.
* Makes predictions on the test set.
* Evaluates the model’s performance using Mean Squared Error (MSE) and R-squared.

**XG Boost Regression**

To implement public transport optimization using XGBoost regression, you can follow a similar approach to the Random Forest Regressor example. XGBoost is a powerful machine learning algorithm that can handle regression tasks effectively. Here’s how to use XGBoost for public transport optimization with a Python example:

Import pandas as pd

From sklearn.model\_selection import train\_test\_split

Import xgboost as xgb

From sklearn.metrics import mean\_squared\_error, r2\_score

# Load your transit data into a DataFrame

Transit\_data = pd.read\_csv(‘transit\_data.csv’)

# Separate features and target variable

X = transit\_data.drop(‘target\_variable’, axis=1) # Adjust ‘target\_variable’ to your dataset

Y = transit\_data[‘target\_variable’]

# Split the data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Create and train the XGBoost Regressor model

Xgb\_regressor = xgb.XGBRegressor(n\_estimators=100, random\_state=42)

Xgb\_regressor.fit(X\_train, y\_train)

# Make predictions on the test set

Y\_pred = xgb\_regressor.predict(X\_test)

# Evaluate the model’s performance

Mse = mean\_squared\_error(y\_test, y\_pred)

R2 = r2\_score(y\_test, y\_pred)

Print(“Mean Squared Error:”, mse)

Print(“R-squared:”, r2)

In this code:

* Load your transit data and separate the features and the target variable.
* Split the data into training and testing sets.
* Create an XGBoost Regressor model with 100 estimators (trees).
* Train the model on the training data.
* Make predictions on the test set.
* Evaluate the model's performance using Mean Squared Error (MSE) and R-squared.

XGBoost is known for its performance and robustness in regression tasks. You can use this model to predict and optimize various aspects of public transport, such as travel times, passenger demand, and more, making data-driven decisions for improving public transit services. Adjust the parameters and features according to your specific dataset and optimization goals.

**POLYNOMIAL Training**

Polynomial regression is a useful technique for modeling non-linear relationships in public transport optimization, where simple linear regression may not suffice. You can use Python with libraries like NumPy and scikit-learn to implement polynomial regression. Here’s a simplified example:

Import numpy as np

From sklearn.linear\_model import LinearRegression

From sklearn.preprocessing import PolynomialFeatures

Import matplotlib.pyplot as plt

# Sample data for demonstration

X = np.array([1, 2, 3, 4, 5, 6, 7]).reshape(-1, 1) # Independent variable (e.g., time)

Y = np.array([3, 8, 12, 8, 4, 2, 1]) # Dependent variable (e.g., passenger demand)

# Transform features into polynomial features

Degree = 2 # Adjust the degree for desired polynomial order

Poly = PolynomialFeatures(degree=degree)

X\_poly = poly.fit\_transform(X)

# Fit a linear regression model to the polynomial features

Model = LinearRegression()

Model.fit(X\_poly, y)

# Predict values using the trained model

X\_test = np.array([8, 9, 10]).reshape(-1, 1) # Test data

X\_test\_poly = poly.transform(X\_test)

Y\_pred = model.predict(X\_test\_poly)

# Visualize the results

Plt.scatter(X, y, color=’blue’, label=’Data points’)

Plt.plot(X\_test, y\_pred, color=’red’, label=’Polynomial Regression’)

Plt.xlabel(‘Time’)

Plt.ylabel(‘Passenger Demand’)

Plt.legend()

Plt.show()

This example:

* We use NumPy to create sample data. Replace this data with your actual transit-related data.
* We transform the independent variable (e.g., time) into polynomial features using PolynomialFeatures.
* We fit a linear regression model to the polynomial features, allowing it to capture non-linear patterns.
* We predict values for new data points and visualize the results.

**Model Training**

Leveraging IoT data for public transport optimization is a transformative approach. IoT devices, such as sensors on vehicles, infrastructure monitoring, and passenger counting mechanisms, provide real-time insights that can enhance transit services. The process starts with data collection and integration, ensuring that data from these devices is harmoniously integrated into the transit platform.

Data preprocessing follows, focusing on data cleansing, handling missing values, and aligning timestamps. Feature engineering is crucial to derive relevant features from the IoT data, such as traffic congestion, weather conditions, and passenger behavior.

For model training, a choice of machine learning models like Random Forest Regressors or even deep learning architectures can be considered. Training on historical IoT data allows these models to capture patterns, making them capable of predicting passenger demand, optimizing routes, and dynamically managing transit operations in real time.

Once deployed within the platform, these models continuously process incoming IoT data, offering real-time predictions and optimization decisions. The platform’s effectiveness is refined through a feedback loop that uses real-time predictions and user feedback to adapt and fine-tune the models. IoT-driven public transport optimization offers the potential to enhance service reliability, reduce costs, and improve the overall travel experience for passengers, aligning transit services with the demands of modern urban environments.

**Conclusion**

In conclusion, integrating IoT technology into public transport optimization presents a transformative opportunity to create a more efficient and responsive transit system. IoT devices, including sensors, cameras, and passenger counters, provide a wealth of real-time data that can be harnessed to improve various aspects of public transportation.

By collecting, preprocessing, and integrating IoT data, we can derive valuable insights about factors like traffic conditions, vehicle performance, and passenger demand. These insights enable the development of predictive models that optimize routes, reduce congestion, and enhance the overall travel experience.

Public transport optimization through IoT is a dynamic process that continues to evolve. It offers the potential to reduce operational costs, increase service reliability, and minimize environmental impact. Moreover, real-time monitoring and data-driven decision-making can adapt to changing circumstances and unexpected events, resulting in a more responsive and customer-centric transit system.

As cities continue to grow and the demand for efficient public transportation rises, the role of IoT in optimization becomes increasingly vital. Embracing these technologies and leveraging the data they provide is not just a forward-looking approach but a necessary step in building sustainable and intelligent transit systems for the future. With ongoing innovation and adaptation, IoT-driven public transport optimization will contribute to smarter, more accessible, and more environmentally friendly urban mobility solutions.