Development of a system for collecting and storing data on the movement of public transport using machine learning models to predict the arrival time of transport at stops

Nurzhanova Zarina, Kuandyk Sultaniyar, Saken Aldiyar

Astana IT University

Table of context

Abstract

In this study, we present the development of a system for collecting and storing data on public transport routes and locations. The system employs an API to receive real-time data from the Avtobus system and stores it in a structured database. Additionally, machine learning models are applied to predict the arrival time of public transport at stops, enhancing service efficiency and passenger satisfaction. This work involved collaboration with data engineers to design and deploy a PostgreSQL database and implement security measures to protect the API from unauthorized access. Our findings demonstrate that the integration of data from various sources and its automatic storage can significantly contribute to public transport management and forecasting systems.

Keywords: Public transport, API, database, PostgreSQL, machine learning, data engineering, public transit prediction

Introduction

The efficient management of public transport systems is crucial for urban mobility and passenger satisfaction. The ability to predict the arrival time of buses at stops has become increasingly important as cities grow, and transit systems become more complex. Many cities now use real-time tracking systems to monitor bus locations, providing passengers with more accurate information about arrivals and departures. However, this data often exists in silos, making it difficult to collect, store, and analyze in a way that facilitates further improvements in transit operations. This article describes the development of a system that collects real-time data on public transport routes and locations, stores this information in a database, and utilizes machine learning models to predict the arrival time of transport at stops. Our approach integrates data engineering principles and machine learning techniques to improve both the management of public transport and the passenger experience.

Materials and Methods[1]

The proposed system for public transport data collection and storage was developed using the following steps:

Data Collection via API. Data on public transport routes, stops, and vehicle locations was collected from the Avtobus system via an API. Basic authorization using a login and password provided by Avtobus system owners was configured for secure access to the API.

Database Design. A database schema was designed to store the incoming data, including tables for transport routes, stops, and real-time coordinates. The database was implemented using PostgreSQL, a robust and flexible DBMS well-suited for handling large datasets.

Security Measures. To protect the system against unauthorized access, a whitelist of authorized IP addresses was created, and Basic authorization was implemented for all API requests.

Machine Learning Model for Time Prediction. The core of the system involved predicting bus arrival times at stops using machine learning models. The model was trained using historical data from the API, including timestamps, route IDs, and stop coordinates.

Automation and Data Storage. A script was developed to automate the process of querying the API and storing the data in the PostgreSQL database at regular intervals.

The combination of these components resulted in a comprehensive system capable of not only storing but also analyzing transport data for improved predictions.

Table 1: Database Schema Overview

	Description	Primary Key	Foreign Keys
Routes	Stores information about bus routes	RouteID	
Stops	Contains data on bus stops and locations	StopID	RouteID (references Routes)
Coordinates	Stores real-time coordinates of buses	CoordinateID	RouteID (references Routes), StopID
PredictionLogs	Records predictions of bus arrival times	PredictionID	RouteID (references Routes), StopID

General Part[2]

API Integration and Data Flow

The system starts by querying the Avtobus system's API for live data on public transport routes and locations. The API is accessed using Basic authorization, ensuring that only authorized users can retrieve the data. Upon receiving data, the system processes it and saves it to the corresponding tables in the PostgreSQL database. The entire process is automated, with queries sent at predefined intervals (e.g., every minute), ensuring that the database is continuously updated with fresh information.

The Avtobus API provides the following data:

- Route ID
- Vehicle coordinates (latitude and longitude)
- Stop information (stop ID, name, location)
- Timestamp of data retrieval

A typical API response contains all the necessary data for real-time tracking of buses. This data is parsed and stored in the database for future use in prediction models.

Database Design and Deployment

The database design was one of the critical aspects of the project. The schema (as shown in Table 1) was structured to accommodate different data types while ensuring referential integrity using primary and foreign keys. PostgreSQL was chosen due to its scalability, ease of use, and advanced features for handling spatial data, which is important for tracking transport locations.

Table 2: Example API Response Parsed into Database

RouteID	StopID	Latitude	Longitude	Timestamp
25	101	52.1234	76.5432	2024-09-22 10:34:56
25	102	52.1245	76.5435	2024-09-22 10:35:56

Results and Discussion[3]

The successful deployment of the system resulted in a real-time, automated collection and storage of public transport data. The PostgreSQL database was populated with data from the Avtobus system, which included route information, stop details, and real-time vehicle locations. The data was continuously updated, providing a rich dataset for further analysis and machine learning predictions.

Machine Learning Model Performance

To predict the arrival time of buses at stops, we implemented a supervised learning model. Historical data from the API, combined with timestamps and geographic coordinates, formed the input for training the model. The model achieved a reasonable level of accuracy, with a mean absolute error (MAE) of 2.5 minutes. This result indicates that the system can reliably predict bus arrival times within a few minutes of actual arrival.

Table 3: Machine Learning Model Results

Model	MAE (minutes)	Training Data Size	Test Data Size
Linear Regression	2.5	10,000 data points	2,000 data points
Random Forest	2.2	10,000 data points	2,000 data points

The results show that our system can provide accurate predictions using real-time data from the Avtobus system, contributing to a better passenger experience and more efficient public transport management.

Conclusion

This article outlined the development of a system for the collection and storage of public transport data, and the use of machine learning models to predict bus arrival times at stops. By automating the data retrieval process from the Avtobus system and securely storing it in a PostgreSQL database, we created a reliable system capable of handling real-time data for analysis and prediction purposes. The integration of machine learning models further enhances the system's utility by providing accurate arrival time predictions, potentially improving urban mobility and passenger satisfaction.

References

1. Smith, J., & Wang, L. (2020). *Real-Time Data Collection Systems for Public Transport*. Journal of Urban Transport, 12(3), 45-67
2. Johnson, M., & Lee, K. (2021). *Predictive Modeling of Public Transport Arrival Times Using Machine Learning*. Data Engineering Journal, 5(2), 87-102.
3. PostgreSQL Documentation (2024). Retrieved from https://www.postgresql.org/docs/
4. Avtobus API Documentation (2023). Available from the Avtobus System Owners

Tables

Table 1: Database Schema Overview

	Description	Primary Key	Foreign Keys
Routes	Stores information about bus routes	RouteID	
Stops	Contains data on bus stops and locations	StopID	RouteID (references Routes)
Coordinates	Stores real-time coordinates of buses	CoordinateID	RouteID (references Routes), StopID
PredictionLogs	Records predictions of bus arrival times	PredictionID	RouteID (references Routes), StopID

Table 2: Example API Response Parsed into Database

RouteID	StopID	Latitude	Longitude	Timestamp
25	101	52.1234	76.5432	2024-09-22 10:34:56
25	102	52.1245	76.5435	2024-09-22 10:35:56

Table 3: Machine Learning Model Results

Model	MAE (minutes)	Training Data Size	Test Data Size
Linear Regression	2.5	10,000 data points	2,000 data points
Random Forest	2.2	10,000 data points	2,000 data points