TrafficTelligence: Advanced Traffic Volume Estimation with Machine Learning

Siri R Kulakarni 21BCE2351
Aayushi Mittal 21BCE0969
Sameeksha Nanda 21BIT0481
Ananya Narayan 21BKT0084

CONTENTS

1. INTRODUCTION

1.1 Project Overview:

The project centers on revolutionizing traffic estimation and management through the creation of a machine learning-powered system. This innovative system aims to resolve persistent issues faced by commuters and city planners due to traffic congestion, unreliable commute times, and inefficient traffic management systems.

Key Components:

Machine Learning-Based Traffic Volume Estimation: Development of a system that integrates diverse data sources, including weather conditions, historical traffic patterns, holidays, and road conditions, to provide accurate traffic volume predictions.

I**ncorporation of Advanced Algorithms:** Utilization of sophisticated machine learning models to ensure continual enhancement and adaptability, enabling real-time adjustments to evolving traffic patterns.

User-Friendly Interface: Creation of an intuitive interface accessible to both commuters and city planners, ensuring ease of use and accessibility.

1.2 Purpose:

The primary objectives and purposes of this project are:

Enhanced Prediction Accuracy: To offer commuters highly accurate traffic volume predictions, mitigating stress, and enabling more predictable commute times.
Actionable Insights for Planners: Providing city planners with valuable data insights for better traffic management, leading to reduced congestion and safer road conditions.
Improved Quality of Life: Enhancing commuter experiences by reducing stress, ensuring predictability in commute times, and ultimately contributing to an improved quality of life.
Efficient Revenue Model: Implementing a subscription-based revenue model that caters to both individual commuters and city planning authorities, offering tiered plans with free basic access and premium features for a fee. Exploring potential partnerships with navigation and transportation companies for further integration and revenue streams.
Scalability and Adaptability: Designing the system to leverage machine learning algorithms and cloud-based infrastructure for scalability, allowing seamless handling of

increased data volumes as the user base expands. Ensuring adaptability across various cities and regions, enabling potential expansion to different markets.

2. LITERATURE SURVEY

2.1 Existing Problem:

The current state of traffic estimation and management presents multifaceted challenges that significantly impact urban life. The escalation in vehicular numbers, coupled with a reluctance towards public transport, has intensified traffic-related issues, leading to chronic congestion and inefficiencies in transportation systems. Existing traffic estimation systems exhibit limitations in accuracy and real-time data availability, resulting in unpredictable commutes, safety concerns, and ineffective traffic management strategies. These deficiencies undermine the quality of life for commuters, heightening stress levels and compromising safety during travel. Moreover, city planners face hurdles in making informed decisions regarding infrastructure development, road expansions, and traffic management due to the inadequacy of precise and real-time traffic data. Consequently, these challenges underscore the urgent need for innovative solutions that harness technology to accurately predict traffic volume, enhance data accuracy, and provide actionable insights for effective traffic management.

2.2 References:

[1] Khan, Sakib Mahmud, Kakan C. Dey, and Mashrur Chowdhury. "Real-time traffic state estimation with connected vehicles." *IEEE Transactions on Intelligent Transportation Systems* 18.7 (2017): 1687-1699.

[2] Çetiner, B. Gültekin, Murat Sari, and Oğuz Borat. "A neural network based traffic-flow prediction model." Mathematical and Computational Applications 15.2 (2010): 269-278.

[3] Akhtar, Mahmuda, and Sara Moridpour. "A review of traffic congestion prediction using artificial intelligence." Journal of Advanced Transportation 2021 (2021): 1-18.

[4] Lee, Kyungeun, et al. "Short-term traffic prediction with deep neural networks: A survey." IEEE Access 9 (2021): 54739-54756.

[5] Ma, Dongfang, Xiang Song, and Pu Li. "Daily traffic flow forecasting through a

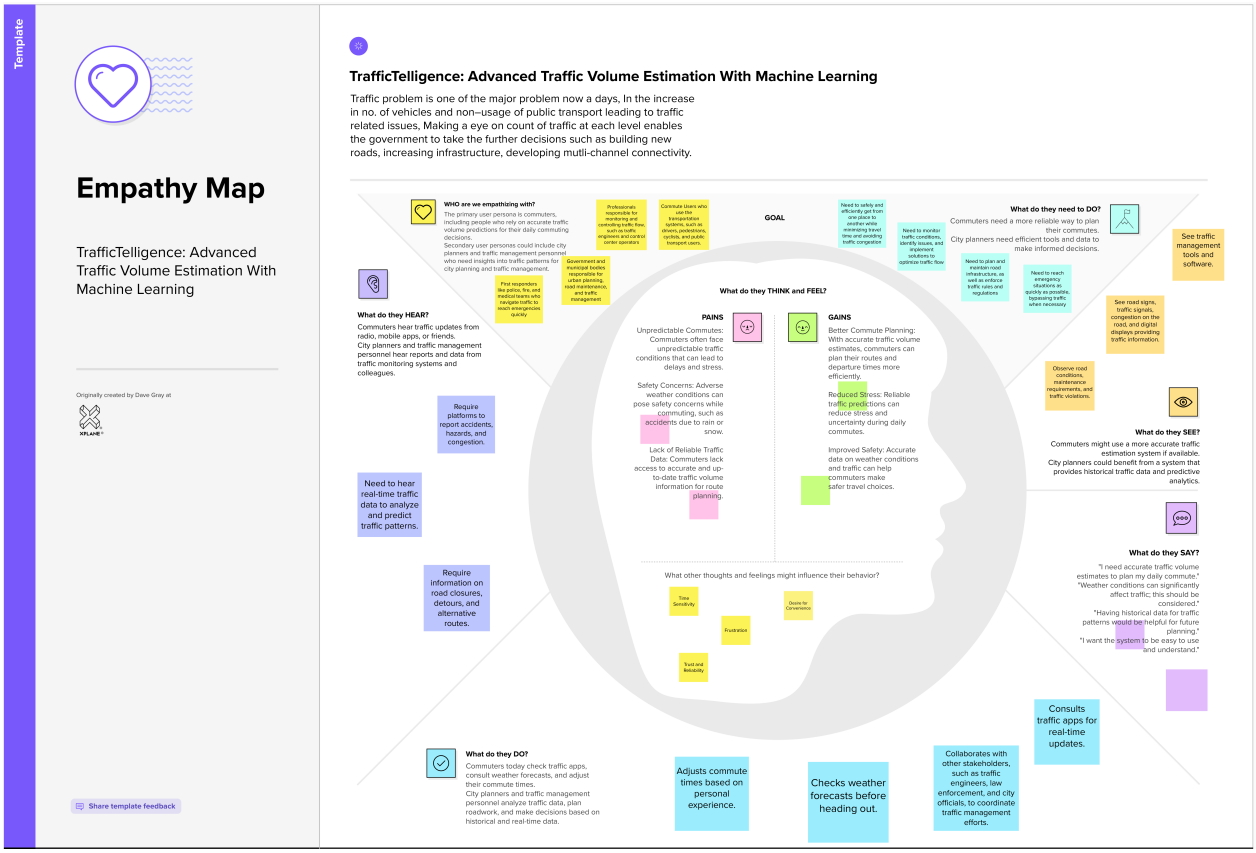
contextual convolutional recurrent neural network modeling inter-and intra-day traffic patterns." IEEE Transactions on Intelligent Transportation Systems 22.5 (2020): 2627-2636.

2.3 Problem Statement Definition:

The existing traffic estimation systems lack accuracy and real-time data availability, resulting in chronic congestion, unpredictable commutes, safety concerns for commuters, and challenges for city planners in making informed decisions. There is a pressing need for the development of a robust machine learning-based traffic volume estimation system that integrates multiple data sources, weather information, historical traffic patterns, and relevant variables to accurately predict traffic volumes. This system should not only cater to commuter needs by providing accurate predictions but also offer actionable insights to city planners for effective traffic management, thus alleviating congestion and enhancing road safety.

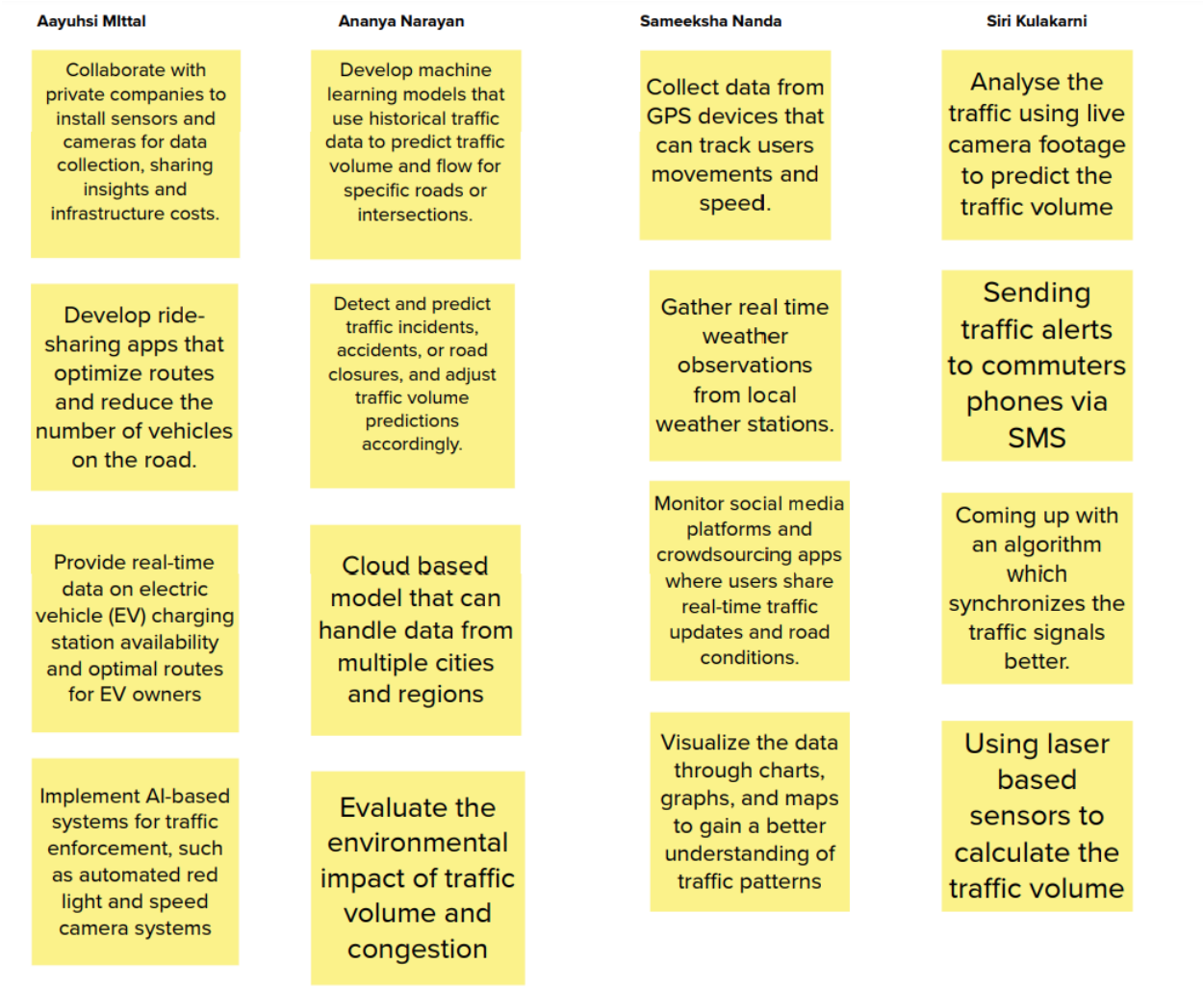
3. IDEATION AND PROPOSED SOLUTION:

3.1 Empathy Map Canvas



3.2 Ideation And Brainstorming:

Brainstorming:



4. REQUIREMENT ANALYSIS:

4.1 Functional Requirements:

Data Collection and Preprocessing:

The system should be able to collect real-time traffic data from multiple sources, including cameras, sensors, and other IoT devices.
Implement preprocessing techniques to clean and standardize the incoming data.

Vehicle Detection and Classification:

Develop a deep learning model capable of accurately detecting and classifying vehicles based on types (e.g., car, truck, motorcycle).
Ensure the model's performance is robust under varying weather and lighting conditions.

Traffic Flow Analysis:

Implement algorithms to analyze the flow of traffic, including average speeds, congestion points, and traffic patterns.
Provide real-time updates on traffic conditions.

Congestion Detection:

Develop a mechanism to identify and notify users about congested areas in real-time.
Implement adaptive routing suggestions to alleviate congestion.

Data Visualization:

Create a user interface that presents traffic insights in an easily understandable format.
Utilize charts, graphs, and maps to visually represent traffic data.

4.2 Non-functional Requirements:

-Performance:

The system should be capable of handling a high volume of real-time data with minimal latency.
Ensure the model's inference time is optimized for real-time processing.

-Reliability:

The system should have a failover mechanism to handle unexpected outages or data source failures.

Implement data integrity checks to ensure the accuracy of processed information.

-Security:

Implement encryption mechanisms for data transmission and storage.
Ensure secure access controls to prevent unauthorized use of the system.

- Scalability:

Design the system to scale efficiently with an increasing number of data sources and users.
Consider future expansions and integrations with additional traffic monitoring devices.

-Usability:

Ensure the user interface is intuitive and user-friendly for both technical and non-technical users.
Provide documentation and training materials for users and administrators.

5. PROJECT DESIGN

5.1 Data Flow Diagrams & User Stories

Data flow diagrams (DFDs) reveal relationships among and between the various components in a program or system. DFDs are an important technique for modeling a system's high-level detail by showing how input data is transformed to output results through a sequence of functional transformations.

Project Setup & Infrastructure:

The project initiation involves setting up the development environment with essential tools and frameworks. This process, led by the development team, establishes the foundation for subsequent stages.

Dataset Collection:

City Traffic Departments spearhead the comprehensive gathering of data, collaborating with various external entities. The dataset encompasses real-time traffic data, weather data, historical traffic data, road condition data, and holiday data. This diverse dataset

serves as the cornerstone for training an accurate machine learning model.

Data Preprocessing:

Public Safety Agencies take charge of preprocessing the collected dataset. This involves meticulous cleaning, organization, and structuring of the data. The cleaned dataset, now enriched with normalized variables from weather data, historical patterns, road conditions, and holiday information, is segregated into training and validation sets, setting the stage for effective machine learning.

Model Evaluation:

Traffic Engineers lead the evaluation of various machine learning models to identify the most suitable one for traffic volume estimation. Through a comprehensive exploration of different ML architectures, a model is selected that aligns seamlessly with the project's objectives.

Model Training:

Transportation Technology Companies take the helm in training the selected machine learning model. This phase involves feeding the pre-processed dataset into the chosen model while closely monitoring its performance on the validation set. The goal is to ensure the model's effectiveness and readiness for real-world traffic predictions.

Model Enhancement and Optimization:

Navigation Providers play a pivotal role in this phase, implementing optimization techniques to enhance the model's accuracy and robustness. Continuous improvement strategies, including augmentation, hyperparameter tuning, and optimization based on real-time user feedback, are employed to fine-tune the model for real-world scenarios.

Model Deployment & Integration:

Traffic Control System Integrators and App Developers collaborate to deploy the trained machine learning model. The model is encapsulated as an API and seamlessly integrated into an intuitive user interface, allowing end-users to receive real-time traffic

predictions.

Testing & Quality Assurance:

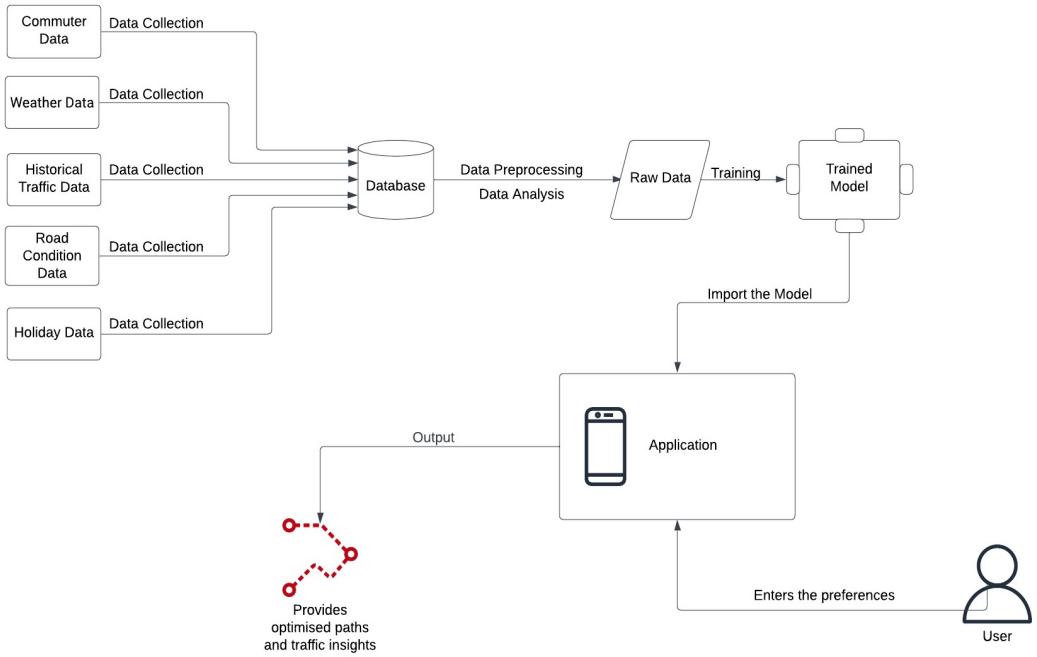
Quality Assurance Teams conduct thorough testing of both the model and the user interface. Any identified issues or bugs are reported and addressed, leading to the optimization of model performance based on feedback and testing results. This iterative process ensures the reliability and effectiveness of the deployed system.

Data Stores:

The dataset is a comprehensive repository that includes real-time traffic data, weather data, historical traffic data, road condition data, and holiday data. The trained model serves as a critical data store, encapsulating the knowledge gained from the training phase.

External Entities:

External entities such as weather data providers, historical traffic data sources, road condition data sources, and holiday data sources contribute valuable information during the dataset collection and preprocessing stages, enriching the dataset with contextual variables for improved model accuracy.



USER STORIES:

As a Traffic Analyst, I want to access real-time traffic patterns.
Acceptance Criteria: Log in to the system, select a specific city area, and visualize current traffic conditions. Receive updates in real-time for effective traffic monitoring.

As a Data Scientist, I want a diverse dataset for model training:
Acceptance Criteria: Access a well-organized dataset containing real-time traffic data, weather conditions, and historical patterns for developing and training machine learning models.

As a Traffic Engineer, I want to evaluate different machine learning models:
Acceptance Criteria: Explore various ML architectures and assess their performance to choose the most suitable model for accurate traffic volume prediction.

As a Machine Learning Developer, I want a preprocessed dataset for efficient model training:
Acceptance Criteria: Receive a preprocessed dataset that includes cleaned data, normalized variables, and segregation into training and validation sets for efficient machine learning model training.

As a Navigation Provider, I want a robust and accurate traffic prediction model:
Acceptance Criteria: Collaborate with Traffic Engineers to implement optimization techniques, ensuring the model's accuracy and robustness for real-time traffic predictions.

As a Traffic Control System Integrator, I want a scalable model deployment solution:
Acceptance Criteria: Deploy the trained machine learning model as an API or service that can scale with increasing traffic data. Ensure seamless integration into existing traffic control systems.

As an App Developer, I want an intuitive user interface for traffic analysis:
Acceptance Criteria: Integrate the machine learning model's API into a user-friendly interface that allows users to easily access and interpret real-time traffic predictions.

As a Quality Assurance Analyst, I want to ensure the system's reliability and performance:
Acceptance Criteria: Conduct thorough testing of the model and interface, identify any issues or bugs, and work with development teams to optimize the model based on feedback and testing results.

As a Public Safety Officer, I want to receive alerts about potential traffic incidents:
Acceptance Criteria: Utilize the Traffic Intelligence System to receive real-time alerts about anomalies or incidents detected by the machine learning model, enhancing public safety measures.

User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Traffic Management Authorities	Project setup & Infrastructure	USN-1	Set up the development environment with the necessary tools and frameworks for the traffic volume estimation project.	Successfully configured development environment with required tools and frameworks.	High	Sprint 1
City Traffic Departments	Dataset Collection	USN-2	Gather a comprehensive dataset containing real-time traffic data and relevant contextual variables.	Collected diverse and comprehensive dataset including traffic flow data, weather conditions, and historical patterns.	High	Sprint 1
Public Safety Agencies	Data Preprocessing	USN-3	Preprocess the gathered dataset by cleaning, organizing, and structuring it for machine learning.	Successfully preprocessed dataset including data cleaning, normalization, and segregation into training and validation sets.	High	Sprint 2
Traffic Engineers	Model Evaluation	USN-4	Explore and evaluate different machine learning models to determine the most effective for traffic volume estimation.	Thoroughly explored different ML architectures and selected the most suitable model for traffic volume prediction.	High	Sprint 2
Transportation Technology Companies	Model Training	USN-5	Train the selected machine learning model using the preprocessed traffic dataset and monitor its performance on the validation set.	Successfully trained the model and assessed its performance on the validation set.	High	Sprint 3
Navigation Providers	Model Enhancement and Optimization	USN-6	Implement optimization techniques to enhance model accuracy and robustness for real-time traffic predictions.	Improved model performance through augmentation, hyperparameter tuning, and optimization based on real-time user feedback.	Medium	Sprint 3
Traffic Control System Integrators and App Developers	Model Deployment & Integration	USN-7	Deploy the trained machine learning model as an API or service for traffic detection. Integrate the model's API into a user-friendly interface for traffic analysis.	Deployed scalable API and integrated it into an intuitive user interface allowing users to receive real-time traffic predictions.	Medium	Sprint 4
Quality Assurance Teams	Testing & Quality Assurance	USN-8	Conduct thorough testing of the model and interface. Identify and report any issues or bugs. Optimize model performance based on feedback and testing results.	Completed rigorous testing, reported issues, fine-tuned model based on feedback.	Medium	Sprint 5

5.2 SOLUTION ARCHITECTURE

Implementing a Traffic Intelligence System on the AWS cloud platform enables scalability, reliability, and flexibility. Leveraging AWS services, we can efficiently handle data processing, machine learning model training, and real-time traffic analysis. The following solution architecture outlines the key components and services used to build a robust Traffic Intelligence System.

User Interface:

Host the web application on Amazon Web Services using Amazon Elastic Beanstalk. This service provides scalable and reliable infrastructure for hosting web applications. Leverage Amazon S3 to store and serve static assets such as HTML, CSS, and JavaScript files, ensuring efficient content delivery and reduced latency.

Data Collection and Preparation:

Store and manage trained models in Amazon S3 for easy access and scalability. Utilize Amazon Kinesis Data Streams for ingesting and processing real-time traffic data from various sources like sensors, cameras, or external APIs. Configure data streams to efficiently handle large volumes of streaming data. Employ Amazon Kinesis Data Firehose to load streaming data from Kinesis Data Streams into data stores like Amazon S3, facilitating further analysis. This streamlining process enhances the efficiency of data collection and preparation.

Machine Learning Model:

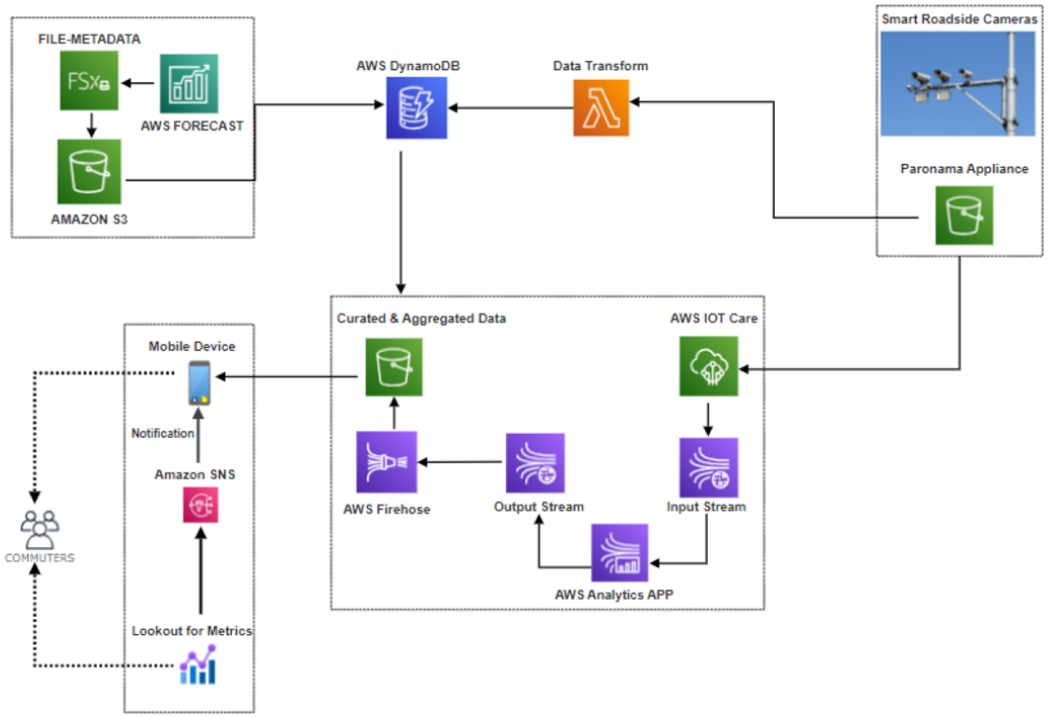
If the machine learning model requires real-time data inputs, enable it to consume data directly from Amazon Kinesis Data Streams. This allows the model to make predictions on the fly as new data arrives. Leverage AWS Lambda to create serverless endpoints for deploying machine learning models and making predictions. This approach ensures scalability and responsiveness in deploying and updating machine learning models.

Data Storage:

Use Amazon S3 to store both raw and processed data. Configure Kinesis Data Firehose to deliver data streams to specific S3 buckets. For structured data, such as user preferences or application settings, leverage Amazon RDS (Relational Database Service) or Amazon DynamoDB. This combination of services ensures effective storage and management of diverse data types within the application.

Data Delivery:

Employ Amazon Kinesis Data Firehose to deliver processed data to various destinations, including databases or analytical tools. Configure Firehose to transform, compress, and encrypt the data before delivery. By integrating Amazon Kinesis and Kinesis Firehose into the architecture, the application can effectively handle real-time traffic data streaming, preprocessing, and delivery. This integration ensures up-to-the-minute insights and predictions in the traffic analysis application, streamlining data ingestion, analysis, and flow within the application.



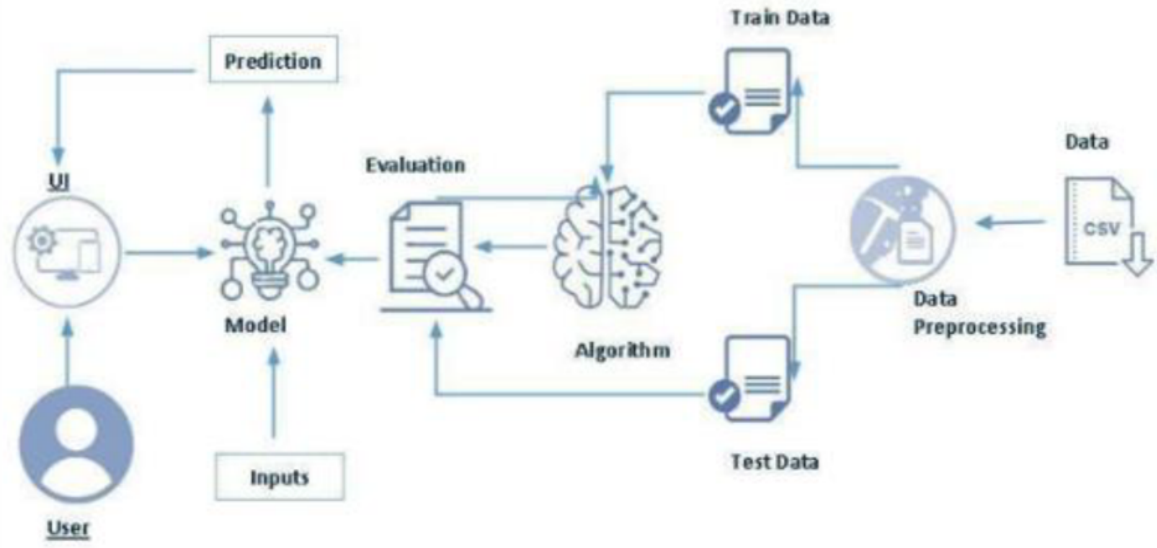
6. PROJECT PLANNING & SCHEDULING

6.1 Technical Architecture

Building a robust technical architecture for artificial intelligence involves designing a system that effectively integrates AI technologies, frameworks, and infrastructure to deliver intelligent solutions. The architecture must support data processing, model training, deployment, and continuous improvement.

User Interface (UI):

- The front-end of the application is built using HTML, CSS, JavaScript, and ReactJS.
- The UI offers an intuitive experience with input fields, dropdown menus, and date/time pickers for easy data entry.
- Users submit parameters such as area, date, and time to access real-time traffic analysis.



Data Collection and Preparation:

- Data collection begins with a link provided to gather relevant data.
- Python, Google Colab, and Microsoft Excel are used for data handling.
- Essential libraries like Pandas, NumPy, Seaborn, and Scikit Learn are employed.
- The workflow includes meticulous handling of missing values, date-time formatting, and systematic correlation assessment to ensure the reliability of the prepared dataset.

Data Analysis:

- Data analysis is performed in Python using the Seaborn package.
- Univariate analysis utilizes distplot and countplot graphs to understand individual feature characteristics.
- Bivariate analysis explores relationships between two features.
- Multivariate analysis provides inferences on linearity and feature correlations, facilitating data-driven insights for informed decision-making.

Machine Learning Model:

- A robust ensemble of regression techniques is employed for predictive modeling.
- Models include linear regression, decision tree regressor, random forest regressor, SVM, and XGBoost.
- The models are trained using Python libraries such as Scikit Learn and XGBoost, ensuring versatile and precise predictive capabilities across various data scenarios.

Database:

- The application uses local storage to house all datasets.
- This local storage system ensures secure and efficient data management, maintaining data integrity and accessibility within a closed network.

6.2 Sprint Planning & Estimation

Project Setup & Infrastructure (USN-1): In the initial sprint, the development environment was set up with the necessary tools and frameworks required for the Traffic Volume Estimation project. The successful configuration ensures a smooth transition to subsequent phases.

Dataset Collection (USN-2): City Traffic Departments took charge of gathering a comprehensive dataset that includes real-time traffic data along with relevant contextual variables such as weather conditions and historical patterns. This diverse dataset forms the foundation for accurate machine learning model training.

Data Preprocessing (USN-3): Public Safety Agencies focused on preprocessing the gathered dataset. This involved cleaning, organizing, and structuring the data to prepare it for machine learning. The successfully pre-processed dataset includes cleaned data, normalized variables, and segregation into training and validation sets.

Model Evaluation (USN-4): Traffic Engineers led the evaluation of different machine learning models to determine the most effective one for traffic volume estimation. Thorough exploration of various ML architectures resulted in the selection of a model that aligns with the project's goals.

Model Training (USN-5): Transportation Technology Companies took charge of training the selected machine learning model using the pre-processed traffic dataset. The model's performance was closely monitored on the validation set to ensure its effectiveness.

Model Enhancement and Optimization (USN-6): Navigation Providers played a crucial role in implementing optimization techniques to enhance the model's accuracy and robustness for real-time traffic predictions. This phase involved continuous improvement through augmentation, hyperparameter tuning, and optimization based on real-time user feedback.

Model Deployment & Integration (USN-7): Traffic Control System Integrators and App

Developers were responsible for deploying the trained machine learning model as an API or service for traffic detection. The model's API was seamlessly integrated into an intuitive user interface, allowing users to receive real-time traffic predictions.

Testing & Quality Assurance (USN-8): Quality Assurance Teams conducted thorough testing of the model and interface in Sprint 5. Any identified issues or bugs were reported and addressed, leading to the optimization of model performance based on feedback and testing results.

6.3 Sprint Delivery Schedule:

Sprint 1 - Project Setup & Infrastructure (USN-1):

The development environment was established, incorporating the necessary tools and frameworks. Successful configuration ensures a smooth transition to subsequent phases.

Sprint 2 - Dataset Collection (USN-2):

City Traffic Departments took charge of gathering a comprehensive dataset, incorporating real-time traffic data and relevant contextual variables. The diverse dataset formed the foundation for accurate machine learning model training.

Sprint 2 - Data Preprocessing (USN-3):

Public Safety Agencies focused on preprocessing the gathered dataset, involving cleaning, organizing, and structuring the data. The pre-processed dataset includes cleaned data, normalized variables, and segregation into training and validation sets.

Sprint 3 - Model Evaluation (USN-4):

Traffic Engineers led the evaluation of different machine learning models to determine the most effective one for traffic volume estimation. Thorough exploration resulted in the selection of a model aligned with project goals.

Sprint 3 - Model Training (USN-5):

Transportation Technology Companies took charge of training the selected machine learning model using the pre-processed traffic dataset. The model's performance was closely monitored on the validation set to ensure effectiveness.

Sprint 4 - Model Enhancement and Optimization (USN-6):

Navigation Providers played a crucial role in implementing optimization techniques to enhance the model's accuracy and robustness for real-time traffic predictions. Continuous improvement involved augmentation, hyperparameter tuning, and optimization based on user feedback.

Sprint 5 - Model Deployment & Integration (USN-7):

Traffic Control System Integrators and App Developers deployed the trained machine learning model as an API for traffic detection. The model's API was seamlessly integrated into an intuitive user interface for real-time traffic predictions.

Sprint 6 - Testing & Quality Assurance (USN-8):

Quality Assurance Teams conducted thorough testing of the model and interface. Identified issues and bugs were addressed, leading to optimization based on feedback and testing results.

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint 1	Project setup & Infrastructure	USN-1	Set up the development environment with the necessary tools and frameworks for the traffic volume estimation project.	1	High	Aayushi Mittal
Sprint 1	Dataset Collection	USN-2	Gather a comprehensive dataset containing real-time traffic data and relevant contextual variables.	1	High	Ananya Narayan
Sprint 2	Data Preprocessing	USN-3	Preprocess the gathered dataset by cleaning, organizing, and structuring it for machine learning.	2	High	Sameeksha Nanda
Sprint 2	Model Evaluation	USN-4	Explore and evaluate different machine learning models to determine the most effective for traffic volume estimation.	2	High	Siri R Kulakarni
Sprint 3	Model Training	USN-5	Train the selected machine learning model using the preprocessed traffic dataset and monitor its performance on the validation set.	5	High	Siri R Kulakarni
Sprint 3	Model Enhancement and Optimization	USN-6	Implement optimization techniques to enhance model accuracy and robustness for real-time traffic predictions.	3	Medium	Ananya Narayan
Sprint 4	Model Deployment & Integration	USN-7	Deploy the trained machine learning model as an API or service for traffic detection. Integrate the model's API into a user-friendly interface for traffic analysis.	4	Medium	Aayushi Mittal
Sprint 5	Testing & Quality Assurance	USN-8	Conduct thorough testing of the model and interface. Identify and report any issues or bugs. Optimize model performance based on feedback and testing results.	2	Medium	Sameeksha Nanda

7. CODING & SOLUTIONING

7.1 The Dataset

It is the hourly Interstate 94 Westbound traffic volume for MN DoT ATR station 301, roughly midway between Minneapolis and St Paul, MN. There are 48204 rows x 8 columns. The 8 attributes are:

1. Holiday – A categorical data of US National Holidays
2. Temperature – Average Temperature in kelvin
3. Rain – Amount of rain occurring in an hour measured in mm
4. Snow – Amount of snow occurring in an hour measured in mm
5. Weather – Categorical data about the weather condition on the particular day
6. Date – Date
7. Time – Time stamp of the data collected

8. Traffic Volume – Numerical value of the number of vehicles as per reported.

```
Data columns (total 8 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   holiday         48204 non-null  object
 1   temp            48151 non-null  float64
 2   rain            48202 non-null  float64
 3   snow            48192 non-null  float64
 4   weather         48155 non-null  object
 5   date            48204 non-null  object
 6   Time            48204 non-null  object
 7   traffic_volume  48204 non-null  int64
```

7.2 Data Preprocessing

There were null values in the columns *temp, rain, snow and weather.* The null values in *temp, rain* and *snow* were replaced with the mean of each attribute and the null values in *weather* were replaced by the mode of the attribute.

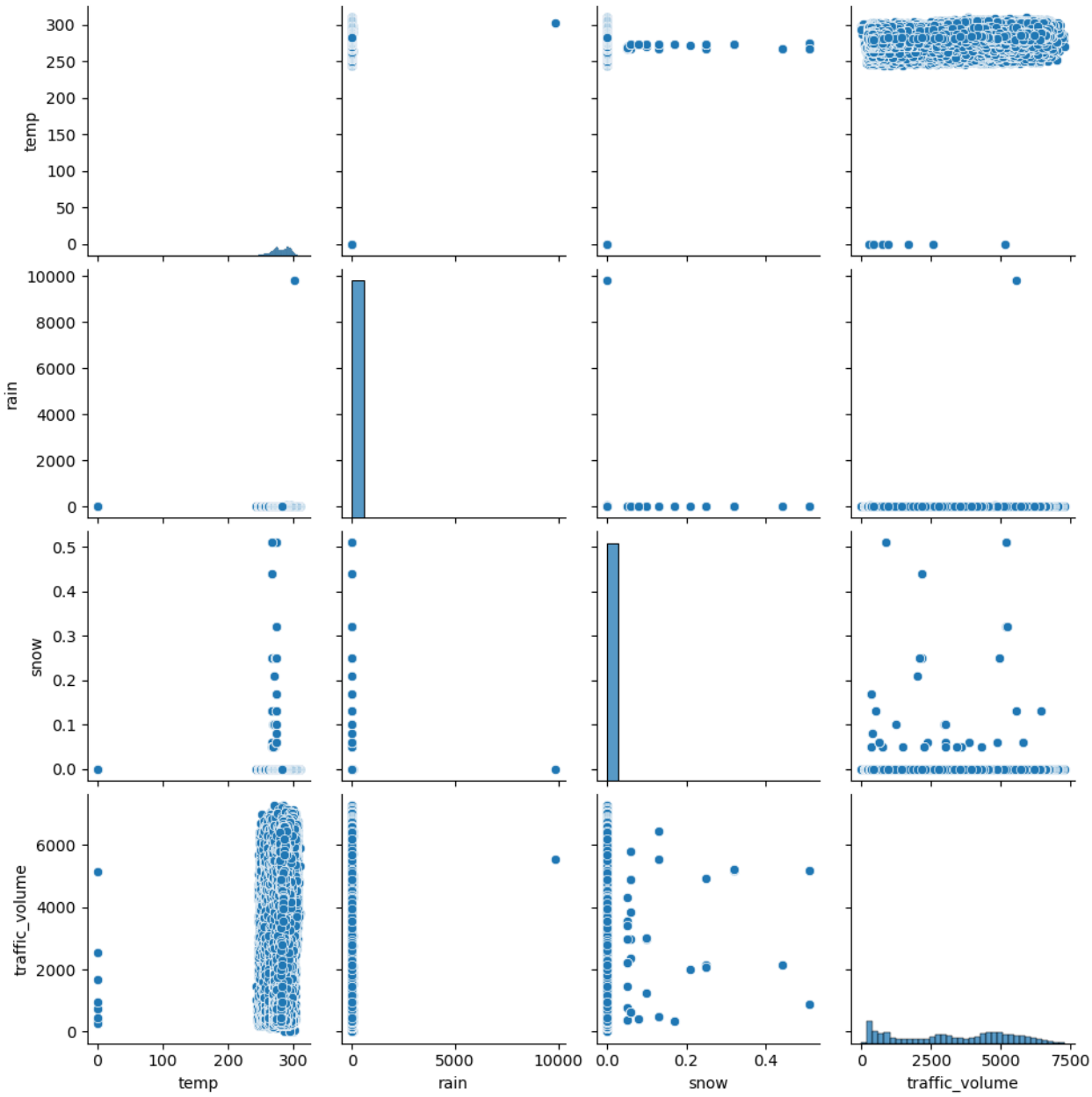
```
data["temp"].fillna(data["temp"].mean(), inplace = True)
data["rain"].fillna(data["rain"].mean(), inplace = True)
data["snow"].fillna(data["snow"].mean(), inplace = True)
data["weather"].fillna(data["weather"].mode()[0], inplace = True)
```

The *date* and *time* columns were then split into *Day*, *Month*, *Year* and *Hours*, *Minutes* and *Seconds* respectively.

```
data[["Day", "Month", "Year"]] = data["date"].str.split("-", expand = True)
data[["Hours", "Minutes", "Seconds"]] = data["Time"].str.split(":", expand =True)
data.drop(columns = ["date", "Time"], axis = 1, inplace = True)
```

7.3 Exploratory Data Analysis

The following image describes the dependency of all attribute with each other:



7.4 Splitting dependent and independent variables

The dataset was later divided in to X and Y where Y had the attribute *traffic_volume* and all the other attributes were put into X.

```
y = data["traffic_volume"]

x = data.drop(columns = ["traffic_volume"], axis = 1)
```

7.5 Encoding and Feature Scaling

The data was encoded using LabelEncoder and scaled using MinMaxScaler from sklearn. The data was later divided into train and test set in the ration 80:20.

```
from sklearn.preprocessing import LabelEncoder

LE = LabelEncoder()

x["weather"] = LE.fit_transform(x["weather"])

x["holiday"] = LE.fit_transform(x["holiday"])


from sklearn.preprocessing import MinMaxScaler

MS = MinMaxScaler()

x_Scaled = pd.DataFrame(MS.fit_transform(x), columns = x.columns)


from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(x_Scaled, y, test_size = 0.2, random_state = 0)

print("Shape of x_train:", x_train.shape)

print("Shape of x_test:", x_test.shape)

print("Shape of y_train:", y_train.shape)

print("Shape of y_test:", y_test.shape
```

7.6 Model Building

We totally built 5 different models for comparing performance and selecting the best one.

```
import sklearn as sk
from sklearn import linear_model
from sklearn import tree
from sklearn import ensemble
from sklearn import svm
import xgboost
lin_reg = linear_model.LinearRegression()
Dtree = tree.DecisionTreeRegressor()
Rand = ensemble.RandomForestRegressor()
svr = svm.SVR()
XGB = xgboost.XGBRegressor()
lin_reg.fit(x_train, y_train)
Dtree.fit(x_train, y_train)
Rand.fit(x_train, y_train)
svr.fit(x_train, y_train)
XGB.fit(x_train, y_train)
```

8. PERFORMANCE TESTING

8.1 Performance Metrics

We fitted and evaluated 5 different models and analysed their performance using R2–score.

Model	R2 - Score	
	On Train results	On Test results
Linear Regression	-5.517285423636809	-5.3993963983221285
Decision Tree Regressor	1.0	0.6938933700945613
Random Forest Regressor	0.9744599411825714	0.8014312682305788
Support Vector Regressor	-19.43147500585336	-19.197492508380627
XGB Regressor	0.8472604817696772	0.8065613781045625

We can observe that the R2 – scores of Random Forest Regressor are the best in both train and test compared to all other models. Hence, we select Random Forest Regressor for further evaluation.

We calculated the R2 – Score, Root Mean Squared Error, and Mean Absolute Error for the model.

Metrics	Values
R2 – scores	On train set - 0.9770921235220543
	On test set - 0.8356571592510573
Root Mean Squared Error	801.6894273058929
Mean Absolute Error	507.9025692355565

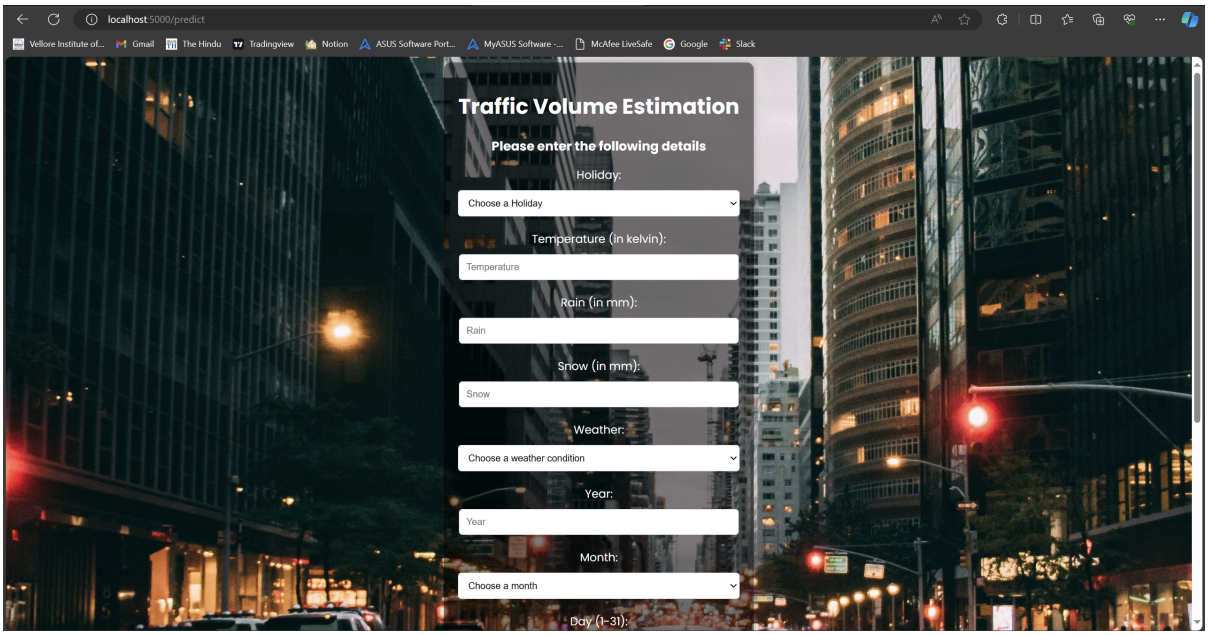
To improve the performance of the model, we used hyperparameter tuning using GridSearchCV and PCA dimensionality reduction. The maximum depth was set to 23 and n_estimators was set to 24 after hyperparameter tuning. The following table gives the comparison between all the models and how they performed.

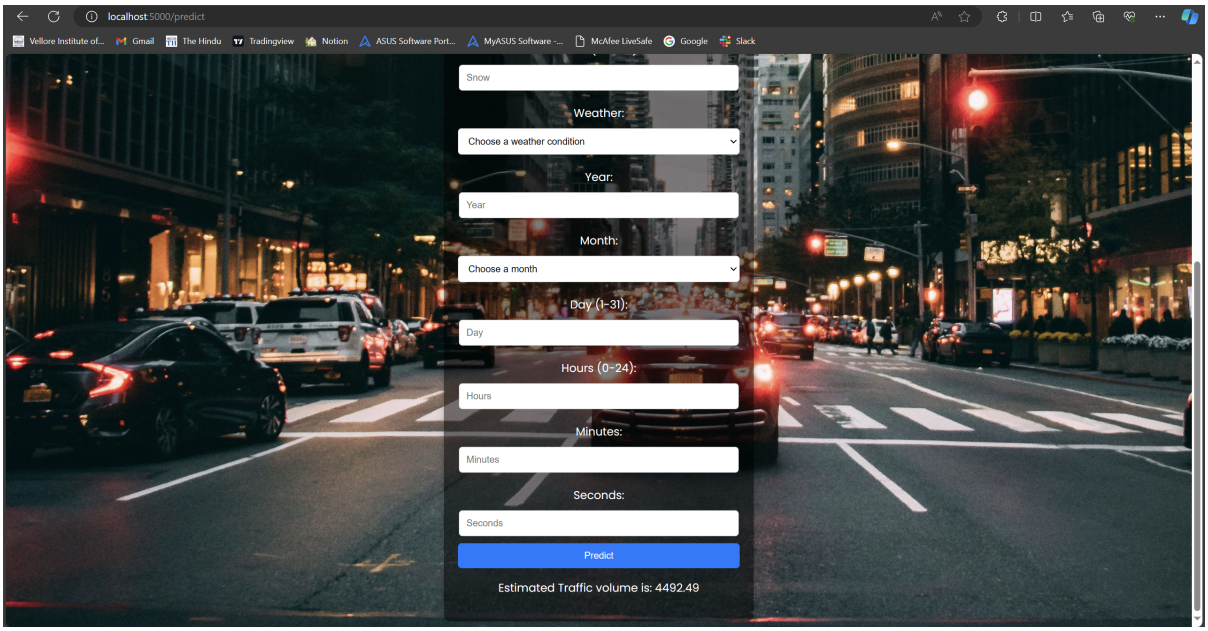
Models	R2-Score		Root Mean Square Error	Mean Absolute Error
	On Train Set	On Test Set		
Regular Random Forest Regressor	0.97709	0.83565	801.68942	507.90256
Hyperparameter Tuned Random Forest Regressor	0.96425	0.82846	819.03382	516.66752
PCA dimensionality reduced Random Forest Regressor	0.97186	0.80324	877.19443	590.86314

After analysing all the metrics, it can be observed that the Regular Random Forest Regressor performs the best. Hence it was chosen as the final model for app building. The model and the scaler were saved using pickle library. A web app was built using HTML and was launched using Flask.

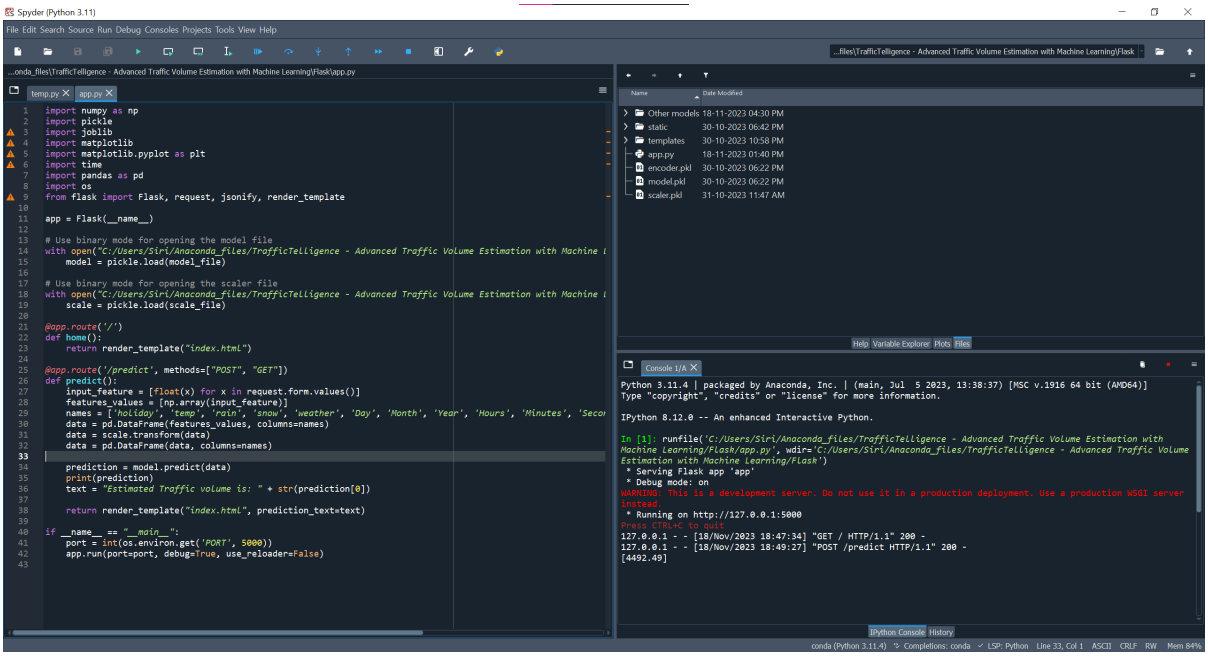
RESULTS

The following screenshots show the user interface of the web app we built.



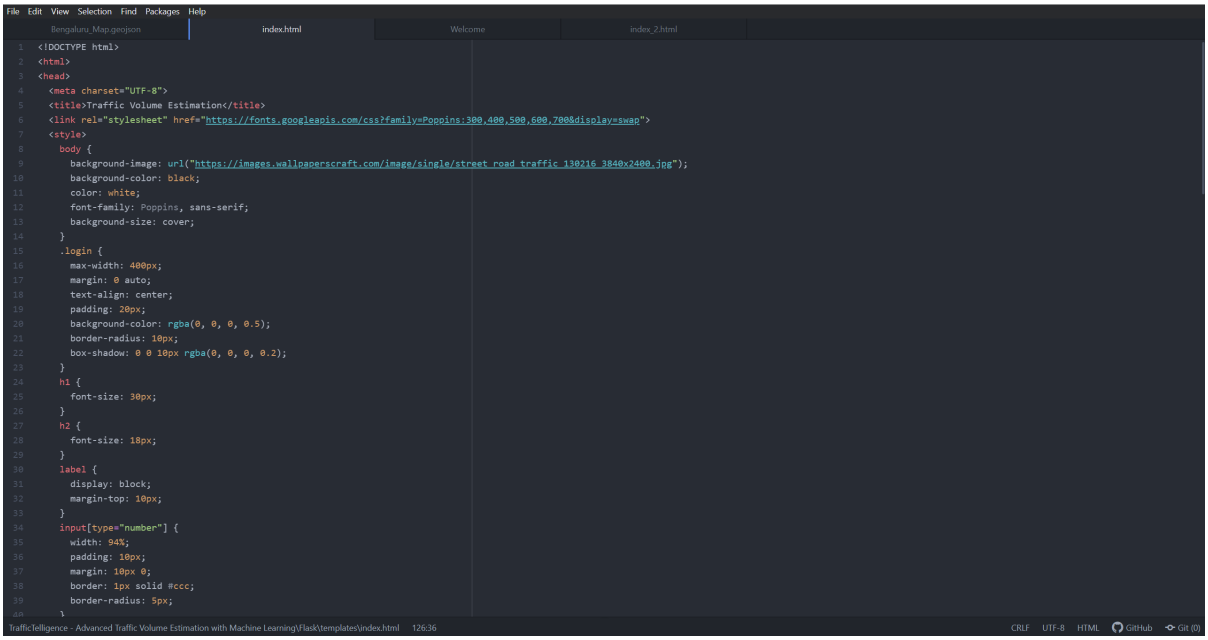


The following screenshot show the codes used in building the above page using Flask and HTML.



```
import numpy as np
import pickle
import joblib
import matplotlib
import matplotlib.pyplot as plt
import time
import pandas as pd
import os
from flask import Flask, request, jsonify, render_template

app = Flask(__name__)

# Use binary mode for opening the model file
with open("C:/Users/Siri/Anaconda_files/TrafficTeLligence - Advanced Traffic Volume Estimation with Machine L
    model = pickle.load(model_file)

# Use binary mode for opening the scaler file
with open("C:/Users/Siri/Anaconda_files/TrafficTeLligence - Advanced Traffic Volume Estimation with Machine L
    scale = pickle.load(scale_file)

@app.route('/')
def home():
    return render_template("index.html")

@app.route('/predict', methods=["POST", "GET"])
def predict():
    input_feature = [float(x) for x in request.form.values()]
    features_values = [np.array(input_feature)]
    names = ['holiday', 'temp', 'rain', 'snow', 'weather', 'Day', 'Month', 'Year', 'Hours', 'Minutes', 'Secon
    data = pd.DataFrame(features_values, columns=names)
    data = scale.transform(data)
    data = pd.DataFrame(data, columns=names)

    prediction = model.predict(data)
    print(prediction)
    text = "Estimated Traffic volume is: " + str(prediction[0])

    return render_template("index.html", prediction_text=text)

if __name__ == "__main__":
    port = int(os.environ.get('PORT', 5000))
    app.run(port=port, debug=True, use_reloader=False)
```

```
Python 3.11.4 | packaged by Anaconda, Inc. | (main, Jul  5 2023, 13:38:37) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 8.12.0 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/Siri/Anaconda_files/TrafficTeLligence - Advanced Traffic Volume Estimation with
Machine Learning/Flask/app.py', wdir='C:/Users/Siri/Anaconda_files/TrafficTeLligence - Advanced Traffic Volume
Estimation with Machine Learning/Flask')
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server
instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
127.0.0.1 - - [18/Nov/2023 18:47:34] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [18/Nov/2023 18:49:27] "POST /predict HTTP/1.1" 200 -
[4492.49]
```



```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>Traffic Volume Estimation</title>
  <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Poppins:300,400,500,600,700&display=swap">
  <style>
    body {
      background-image: url("https://images.wallpaperscraft.com/image/single/street_road_traffic_130216_3840x2400.jpg");
      background-color: black;
      color: white;
      font-family: Poppins, sans-serif;
      background-size: cover;
    }
    .login {
      max-width: 400px;
      margin: 0 auto;
      text-align: center;
      padding: 20px;
      background-color: rgba(0, 0, 0, 0.5);
      border-radius: 10px;
      box-shadow: 0 0 10px rgba(0, 0, 0, 0.2);
    }
    h1 {
      font-size: 30px;
    }
    h2 {
      font-size: 18px;
    }
    label {
      display: block;
      margin-top: 10px;
    }
    input[type="number"] {
      width: 94%;
      padding: 10px;
      margin: 10px 0;
      border: 1px solid #ccc;
      border-radius: 5px;
    }
```

10. ADVANTAGES AND DISADVANTAGES:

-ADVANTAGES:

1. Better Commute Planning: With accurate traffic volume estimates, commuters can plan their routes and departure times more efficiently.
2. Reduces Stress: Reliable traffic predictions can reduce stress and uncertainty during daily commutes.
3. Improved Safety: Accurate data on weather conditions and traffic can help commuters make safer travel choices.
4. Environmental Impact: Optimal route planning can lead to reduced fuel consumption and lower emissions, contributiong to environmental sustainabiltiy.
5. Emergency Response Optimization: Emergency services can utilize real-time traffic data to optimize response times during accidents or other incidents.

-DISADVANTAGES:

1. Unpredictable Commutes: Commuters often face unpredictable traffic conditions that can lead to delays and stress.
2. Safety Concerns: Adverse weather conditons can pose safety concerns while commutuing, such as accidents due to rain or snow.
3. Lack of Reliable Traffic Data: Commuters lack access to accurate and up-to-date traffic volume information for route planning.
4. Privacy Concerns: The collection and analysis of traffic data may raise privacy concerns among individuals, as the system relies on the monitoring of vehciles and potentially drivers. Adequate measures should be in place to address and communicate privacy considerations.
5. Maintainence and Upkeep: Ongoing maintainence and updates to keep the system accurate and efficient may require continous investment and resources.

11. CONCLUSION:

The investigation and assessment of different machine learning algorithms for traffic volume estimation have yielded significant findings regarding how to tackle the enduring problems of traffic jams and erratic commutes. By means of extensive testing with the Linear Regression, Decision Tree, Support Vector, XGB, and Random Forest regression models, notable progress has been achieved in determining the best model for this crucial application.

The study demonstrated that the models performed differently, with differences in predictability, resilience, and ability to adjust to the intricacies of traffic estimation. Although every model shown proficiency in managing distinct facets of the data, the Random Forest Regressor surfaced as the most ideal option for the backend model, taking into account the particular features of our dataset.

The effectiveness of the Random Forest Regressor in managing a variety of input variables, such as the weather, holidays, and past traffic patterns, was demonstrated. Our traffic volume estimation system chose this model because of its adaptability to non-linear interactions and resistance to overfitting, which allowed it to work perfectly with our project's objectives.

This decision was made after giving considerable thought to a number of variables, such as scalability, flexibility to the complexity of the dataset, and model performance indicators. Although our method lacks real-time data, the Random Forest Regressor is still a solid option for producing precise and trustworthy traffic volume estimates.

The foundation for creating a sophisticated traffic estimate system is laid by the wise choice of the Random Forest Regressor as the backend model. By utilizing the knowledge gathered by empathy mapping and comprehending the requirements of both commuters and city planners, this approach has the potential to greatly improve commuter satisfaction, strengthen traffic control tactics, and support more effective urban development.

To sum up, this endeavour is a big step in the right direction toward solving traffic-related issues. We are about to implement a game-changing technology that will improve commuters' quality of life and simplify traffic control procedures by utilizing machine learning, specifically the Random Forest Regressor.

12. FUTURE SCOPE

The envisioned machine learning-based traffic volume estimation system lays the foundation for a dynamic and continually evolving approach to traffic management. Its future scope encompasses:

Integration of IoT and Sensor Networks: Expanding the system to integrate with Internet of Things (IoT) devices and sensor networks placed strategically across the city. This integration will provide more granular and real-time data, enabling precise traffic management decisions.

Predictive Maintenance: Evolving the system to not only predict traffic volume but also anticipate infrastructure issues. By analyzing traffic patterns, wear and tear on roads, and usage, the system could provide insights for predictive maintenance, optimizing road conditions, and preventing potential hazards.

Smart City Integration: Integration with broader smart city initiatives to create an ecosystem where transportation systems communicate with other city infrastructure, such as public transit, emergency services, and environmental sensors, fostering a holistic approach to urban management.

Global Expansion and Customization: Adapting the system to suit various cities worldwide by customizing algorithms and features based on local traffic behaviors, infrastructure, and cultural nuances.

Incorporation of AI Ethics and Governance: Addressing ethical considerations such as data privacy, transparency, and bias mitigation. Implementing governance frameworks to ensure responsible AI usage and maintain public trust.

Partnerships and Collaborations: Forming strategic alliances with academia, transportation companies, and government bodies to foster research, innovation, and policy implementation in the field of traffic management and urban planning.

The future evolution of this solution lies in its adaptability, scalability, and commitment to leveraging cutting-edge technologies to revolutionize urban mobility and transportation management.

13. APPENDIX

The full source code of this project is available in the github repository:
https://github.com/smartinternz02/SI-GuidedProject-612418-1698838488.git

The link to all the phase documents are available in the github repository:
https://github.com/smartinternz02/SI-GuidedProject-612418-1698838488.git

A live demo link of the project is available in:

https://www.youtube.com/embed/uaOWWKIkDHM