Trip Based Modeling of Fuel Consumption in Modern Fleet Vehicles

IBM–DOCUMENTATION UNDER THE GUIDANCE OF

Industry Mentor Name	:	Prof Swetha
Faculty Mentor Name	:	D.NARASHIMAN

TEAM ID : PNT2022TMID35586

SUBMITTED BY:

1) Ashok G 2019115023
2) Dhileepan S 2019115029
3) Dhinu Praveen 2019115030
4) Prateek Kumar Srirangan 2019115068

DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY

COLLEGE OF ENGINEERING, GUINDY

ANNA UNIVERSITY: 2019-2023

1. INTRODUCTION

1.1 Project Overview

Fleet vehicles are very useful and irreplaceable when it comes to transportation of goods. But the effect they have on the environment is very high and is very alarming in nature. They consume a lot of fuel and pollute the environment. Fleet vehicles account for almost 25% of the fuel consumption In countries like the US. Amount of fuel consumption varies with parameters like driver behavior, weather conditions and other vehicle parameters. These trucks leave a very high level of carbon footprint. Hence it is very useful to predict the fuel consumption of fleet vehicles and take necessary actions so as to reduce the fuel consumption. In this paper, a flask app with a machine learning model was developed and used to estimate the fuel consumption based on certain parameters.

1.2 Purpose

The cautious usage of fuel by the heavy-duty trucks can improve the country's economy and reduce the impact on the environment. It would also reduce a significant amount of the money that is being spent towards fuel.

2. LITERATURE SURVEY

2.1 Existing Problem

In a large number of countries the cost of fuel consumption is very high. They have adopted various methods to reduce consumption of fuel but it is not being reduced significantly. We must be able to predict the fuel consumption for a trip and then be able to point out major reasons for the consumption. This can help us to improve in particular areas and reduce fuel consumption.

Another way of looking at this problem is to implement a high quality and highly connected public transport system. Even they have some disadvantages like high initial cost and takes a long time to be profitable.

2.2 References

[1] Katreddi, S.; Thiruvengadam, A. Trip Based Modeling of Fuel Consumption in Modern Heavy-Duty Vehicles Using Artificial Intelligence. Energies 2021, 14, 8592.

[2] Gajendran, Prakash, "Development of a heavy duty diesel vehicle emissions inventory prediction methodology" (2005). *Graduate Theses, Dissertations, and Problem Reports*. 2658.

[3] H. Bandi, S. Joshi, S. Bhagat, and A. Deshpande, “Assessing Car Damage with Convolutional Neural Networks,” 2021 International

Conference on Communication information and Computing Technology (ICCICT). IEEE, Jun. 25, 2021. doi: 10.1109/iccict50803.2021.9510069.

[4] Ali Hakimelahi, K.V. Krishna Rao, S.L. Dhingra, Sina Borzooei,

Fuel Consumption Monitoring for Travel Demand Modeling,

Transportation Research Procedia, Volume 17, 2016, Pages 703-712,

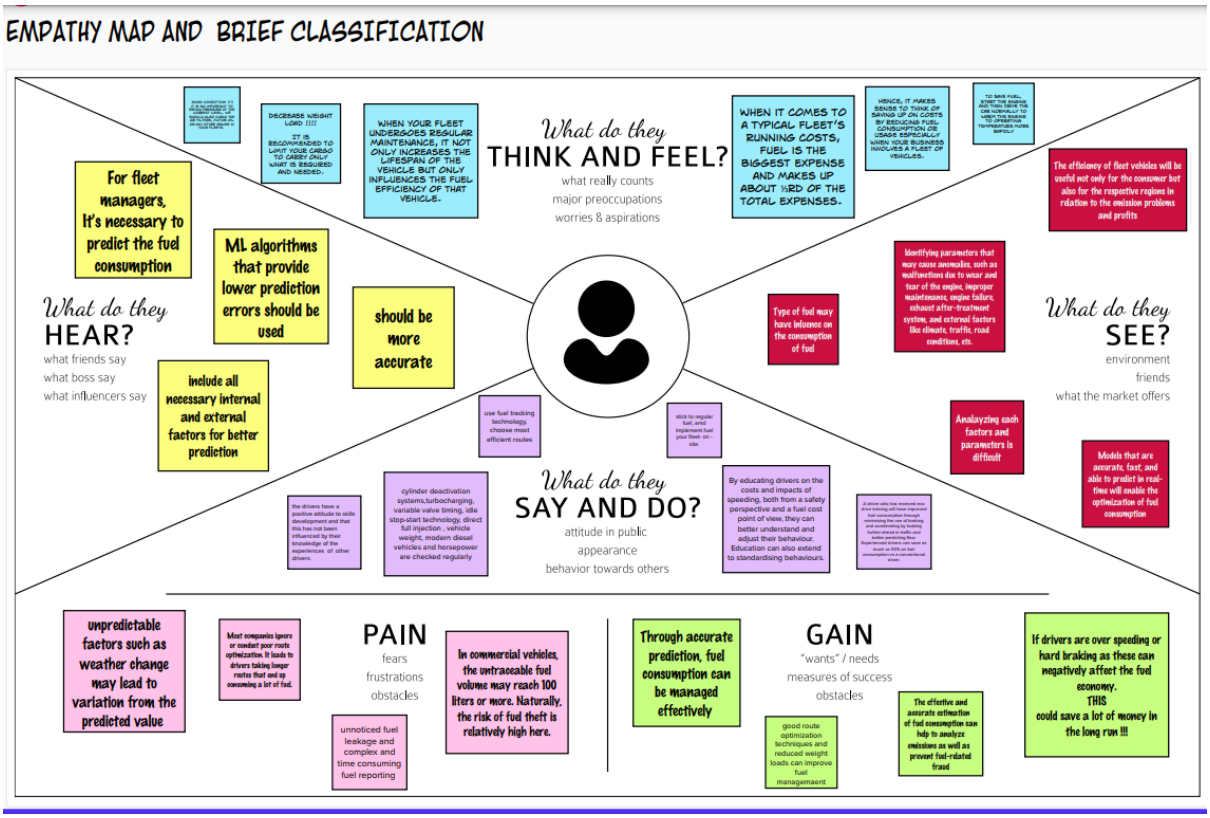
ISSN 2352-1465,

2.3 Problem Statement Definition

To predict the fuel consumption of fleet vehicles based on information from the on-board vehicle diagnostics sensors, vehicle information and weather to help fleet owners get an estimate of mileage and tackle illegal activities.

3. IDEATION AND PROPOSED SOLUTION

3.1 Empathy Map Canvas



3.2 Ideation And Brainstorming

2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

Dhileepan

- Track trip information in real-time
- Tire Pressure Sensor
- Dynamic trip suggestion
- Mobile App

Ashok

- Road Traffic monitor
- Easy to navigate UI
- Trip Metrics
- Vehicle Health analysis

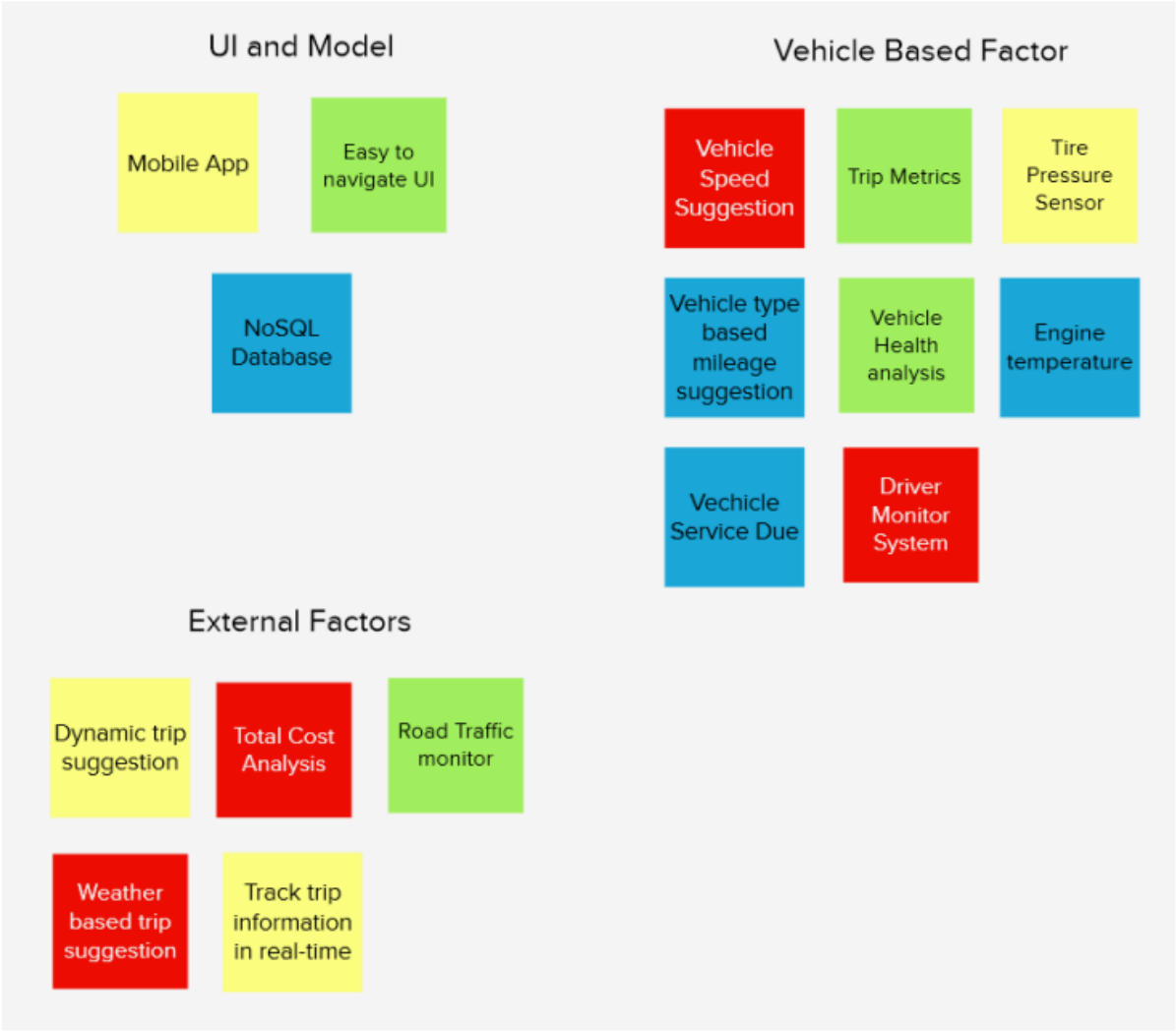
Prateek Kumar

- Vehicle Speed Suggestion
- Total Cost Analysis
- Weather based trip suggestion
- Driver Monitor System

Dhinu Praveen

- NoSQL Database
- Engine temperature
- Vechicle Service Due
- Vehicle type based mileage suggestion

Step-2: Brainstorm, Idea Listing and Grouping



Step-3: Idea Prioritization

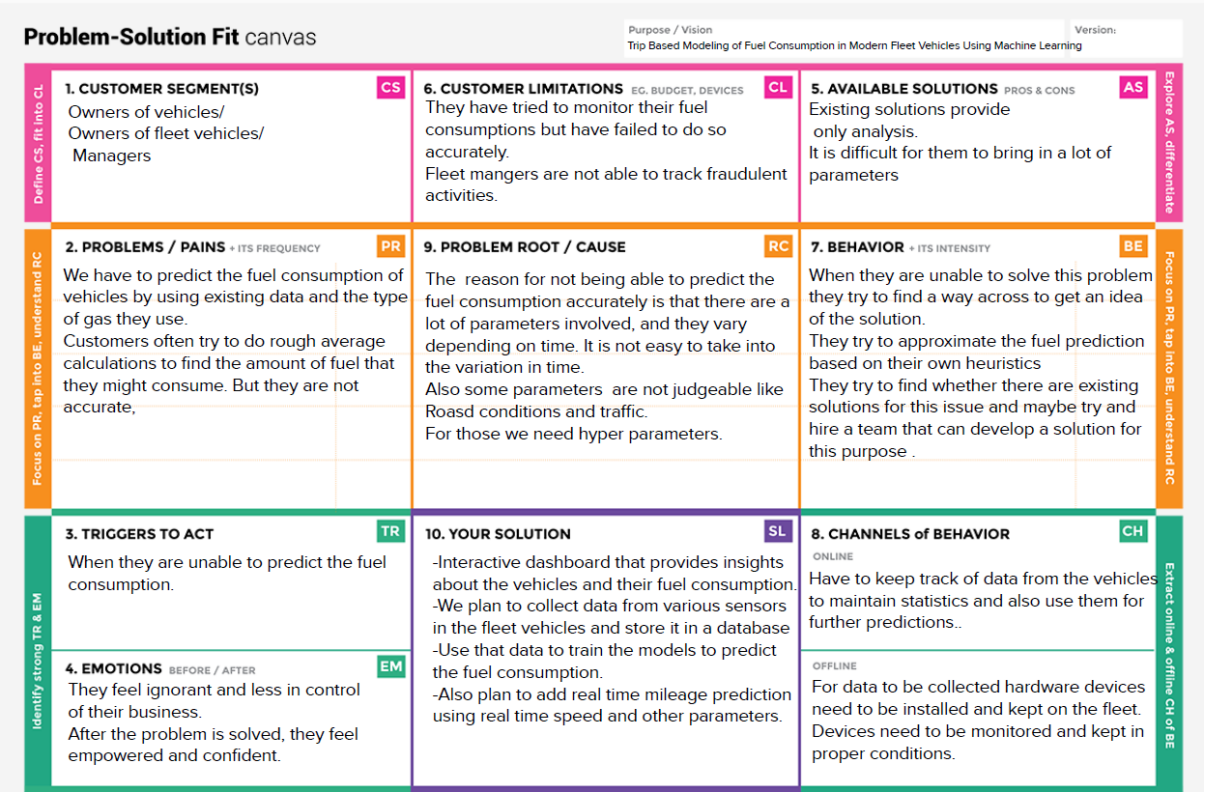
3.3 Proposed solution

S. No.	Parameter	Description
1.	Problem Statement (Problem To Be Solved)	Problem statement is to track fuel consumption in fleet vehicles and provide detailed statistics and suggestions for efficiency in a website.
2.	Idea/Solution description	A website to display the fleet information and provide detailed statistics. Also provide suggestions for fuel efficiency improvement using ML models.

3.	Novelty / Uniqueness	Predict fuel consumption based on vehicle speed, engine temperature, weather and fuel type. ● User Feedback and Reviews ● Prediction History ● Responsive UI
4.	Social Impact / Customer Satisfaction	Reduce fuel consumption and thereby reducing carbon emissions. Fleet owners enjoy increased profits and will be able to track fraudulent activities in their fleet.
5.	Business Model	Subscription based business model where all necessary sensors will be installed with technical support and training. Premium subscribes get real-time data of their fleet vehicles.

6.	Scalability of the solution	This project can be scaled further to common people who want to track the fuel consumption for their cars. Logistic and shipping companies can heavily benefit from the prediction result to increase profits.

3.4 Proposed Solution Fit



Problem-Solution Fit canvas

Purpose / Vision: Trip Based Modeling of Fuel Consumption in Modern Fleet Vehicles Using Machine Learning

Version:

Define CS, fit into CL

1. CUSTOMER SEGMENT(S) CS
Owners of vehicles/
Owners of fleet vehicles/
Managers

6. CUSTOMER LIMITATIONS EG. BUDGET, DEVICES CL
They have tried to monitor their fuel consumptions but have failed to do so accurately.
Fleet mangers are not able to track fraudulent activities.

5. AVAILABLE SOLUTIONS PROS & CONS AS
Existing solutions provide only analysis.
It is difficult for them to bring in a lot of parameters

Explore AS, differentiate

Focus on PR, tap into BE, understand RC

2. PROBLEMS / PAINS + ITS FREQUENCY PR
We have to predict the fuel consumption of vehicles by using existing data and the type of gas they use.
Customers often try to do rough average calculations to find the amount of fuel that they might consume. But they are not accurate,

9. PROBLEM ROOT / CAUSE RC
The reason for not being able to predict the fuel consumption accurately is that there are a lot of parameters involved, and they vary depending on time. It is not easy to take into the variation in time.
Also some parameters are not judgeable like Roasd conditions and traffic.
For those we need hyper parameters.

7. BEHAVIOR + ITS INTENSITY BE
When they are unable to solve this problem they try to find a way across to get an idea of the solution.
They try to approximate the fuel prediction based on their own heuristics
They try to find whether there are existing solutions for this issue and maybe try and hire a team that can develop a solution for this purpose .

Focus on PR, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS TO ACT TR
When they are unable to predict the fuel consumption.

4. EMOTIONS BEFORE / AFTER EM
They feel ignorant and less in control of their business.
After the problem is solved, they feel empowered and confident.

10. YOUR SOLUTION SL
-Interactive dashboard that provides insights about the vehicles and their fuel consumption.
-We plan to collect data from various sensors in the fleet vehicles and store it in a database
-Use that data to train the models to predict the fuel consumption.
-Also plan to add real time mileage prediction using real time speed and other parameters.

8. CHANNELS of BEHAVIOR CH
ONLINE
Have to keep track of data from the vehicles to maintain statistics and also use them for further predictions..

OFFLINE
For data to be collected hardware devices need to be installed and kept on the fleet.
Devices need to be monitored and kept in proper conditions.

Extract online & offline CH of BE

4. REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement	Sub Requirement(Story/Sub-Task)
FR-1	User Registration	Registration through Form Registration through Email
FR-3	Vehicle Fuel Consumption Prediction Page	Vehicle detail form Edit/Update vehicle details Prediction result and visualization
FR-4	User Dashboard	New prediction option Tabs with history of past predictions

4.2 NON-FUNCTIONAL REQUIREMENTS

Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	User friendly interface that is easy to understand and navigate.
NFR-2	**Security**	User authentication using email verification.

NF R-3	**Reliability**	Fast and accurate predictions..
NF R-4	**Performance**	Light weight ML model deployment using flask for quick and accurate predictions. Fast loading time for the web-pages.
NF R-5	**Availability**	Cloud based web application deployment for 24x7 website availability
NF R-6	**Scalability**	Highly scalable since web app is deployed on IBM cloud. System hardware can be improved and purchased with increase in website traffic.

5. PROJECT DESIGN

5.1 DATA FLOW DIAGRAM

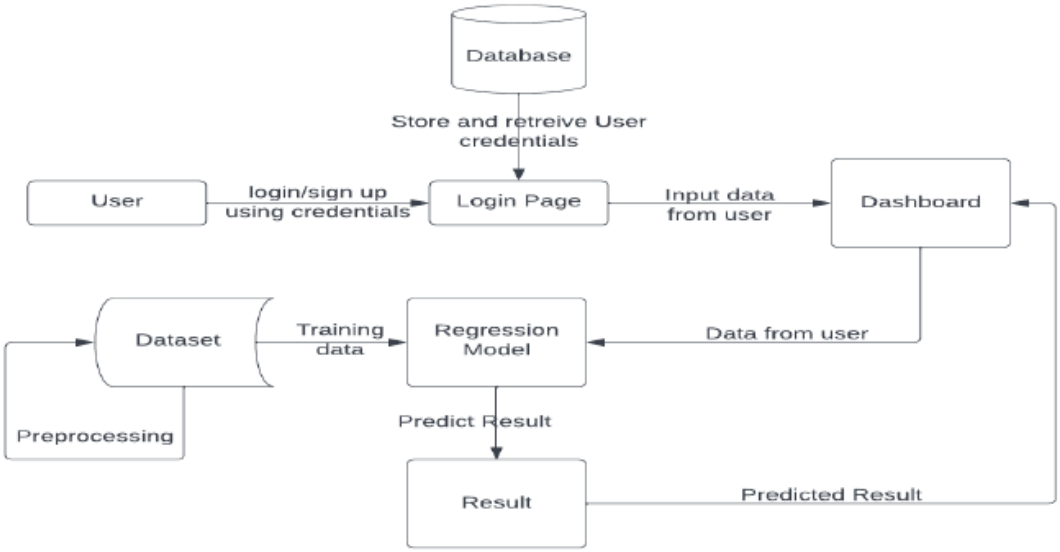


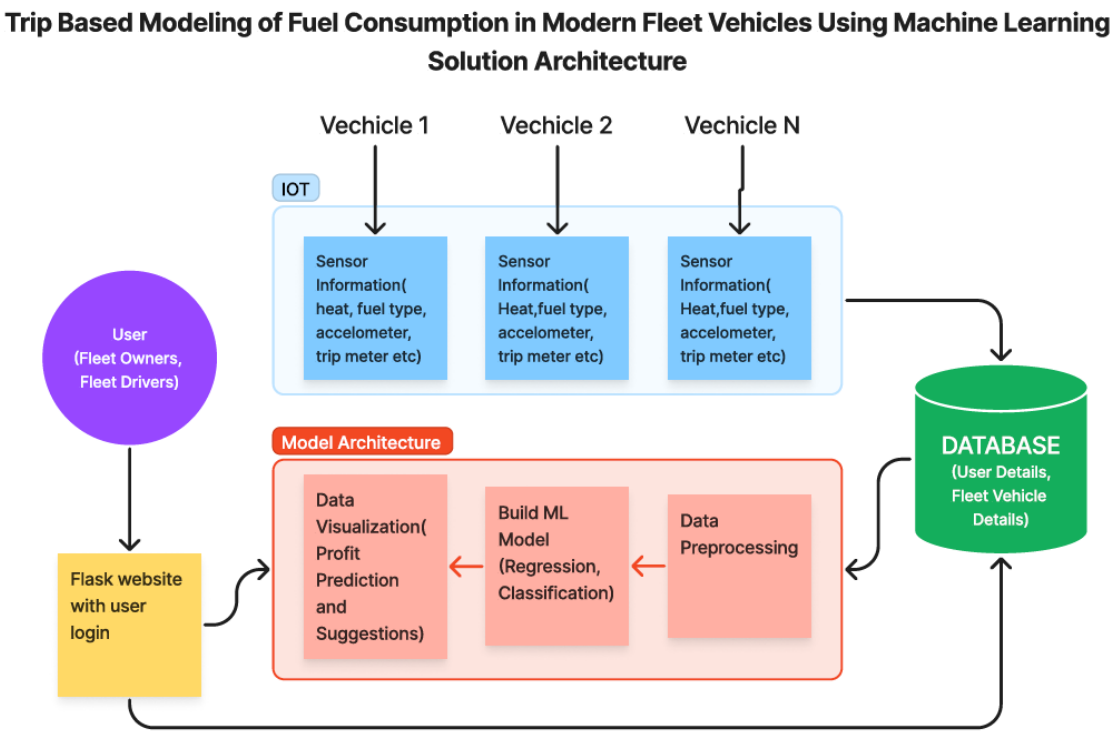
Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	Build a website to interact with user.	HTML, CSS, JavaScript, Bootstrap 4
2.	Database	Store data for training data and user updates.	Flask-SQLAlchemy (sqllite)
3.	Data Cleaning	Pre-process the data to reduce robustness.	Numpy, Pandas
4.	API	Extract data from the dashboard.	Python, Flask, Sklearn
5.	Machine Learning Model	Regression model is used to predict output.	Linear Regression, Random forest, Decision Tree
6.	Infrastructure	Application Deployment on Cloud.	IBM Cloud Foundry

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	CSS styling framework, RDMS, Backend Framework.	IBM Cloud Foundry, Python Flask
2.	Security Implementations	Authentication	Encryption Techniques
3.	Scalable Architecture	Can be scalable	IBM Cloud Service
4.	Availability	Increase by load balancer	IBM Cloud Hosting
5.	Performance	Handle large number of users at the same time	Load Balancer, Distributed Server.

5.2 SOLUTION AND TECHNICAL ARCHITECTURE

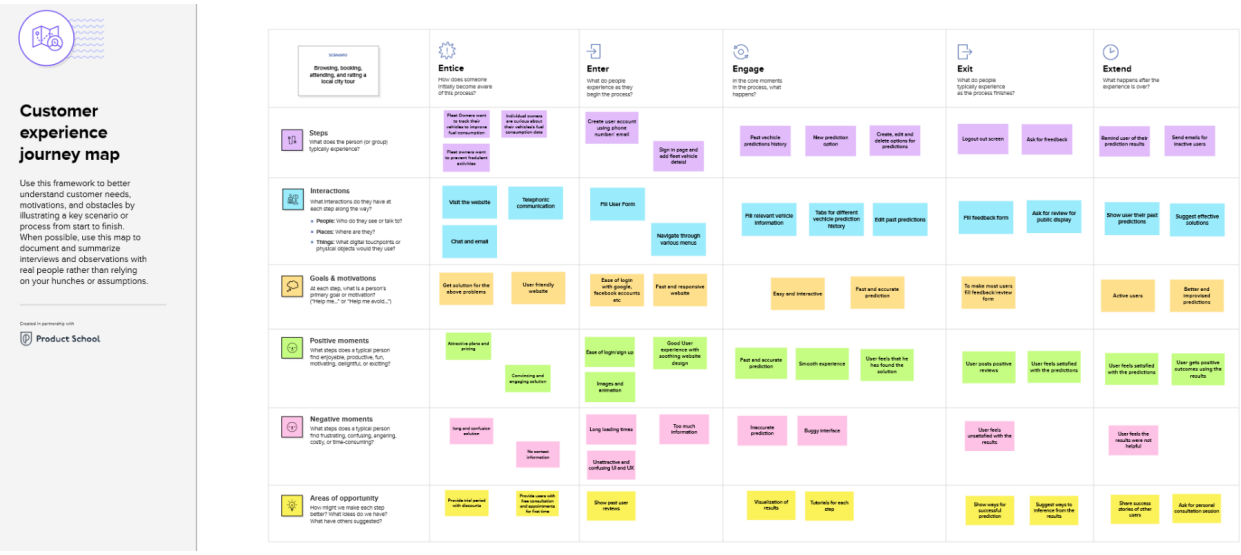


5.3 USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	Create Account in that application	High	Sprint-1
Customer	login	USN-2	As a user, I will receive confirmation email once I have registered for the application	Login using credentials	High	Sprint-1
Customer	Dashboard	USN-3	Once I enter the dashboard, I can input values.	Give input values	High	Sprint-3
Customer		USN-4	As a User, I can get the predicted value	Get output values	High	Sprint-4
Developer	Register	USN-5	As a developer, I will store the login credentials in the database	Store User credentials in database	Medium	Sprint-2

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Developer	Login	USN-6	As a developer, I'll verify the login credentials using database	Validate the user details for login	Medium	Sprint-2
Developer	Dashboard	USN-7	As a developer, I'll build a webpage useful for customers to enter input data and get predicted value	Give a web frame to input data and get output value	Medium	Sprint-3,4
Developer	Model	USN-8	preprocess the dataset and train the model with training data in dataset	Clean the dataset	High	Sprint-3,4

5.4 Customer Journey Map



6. PROJECT PLANNING AND SCHEDULING

6.1 SPRINT PLANNING AND ESTIMATION

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	4	High	Dhileepan S, Prateek Kumar
Sprint-1		USN-2	As a user, I will receive verification email once I have registered for the application	4	High	Ashok G
Sprint-1		USN-3	As a user, I can register for the application through Gmail	2	Medium	Dhinu Praveen
Sprint-2	Login	USN-4	As a user, I can log into the application by entering email & password	2	High	Ashok G, Dhinu Praveen
Sprint-3	Dashboard	USN-5	As a user, I can make a new prediction by filling the required vehicle details	4	High	Dhileepan S
Sprint-3		USN-6	As a user, I can view previous predictions tabs	2	Medium	Prateek Kumar
Sprint-2	Build Model	USN-7	Pre-process and clean the dataset	2	High	Dhinu Praveen
Sprint-2		USN-8	Build a ML model using the cleaned dataset	6	High	Dhileepan S, Ashok G
Sprint-3	Deploy Model	USN-9	Deploy model using flask	4	High	Prateek Kumar

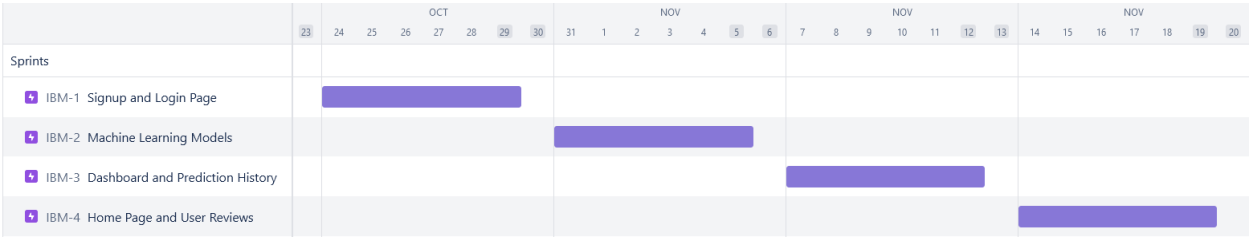
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-3	Result Visualization	USN-10	Visualize the result using various graphs	2	Medium	Dhinu Praveen
Sprint-4	Suggestion for Improvement	USN-11	Suggest best practises and improvements based on results	2	Low	Ashok G
Sprint-4	User Interface	USN-12	UI/UX improvements with images and animations	4	Medium	Prateek Kumar
Sprint-4	Documentation	USN-13	User manual guide for new users and documentation for developers	4	High	Dhilleepan S, Dhinu Praveen

6.2 SPRINT DELIVERY SCHEDULE

Project Tracker, Velocity & Burndown Chart: (4 Marks)

Sprint	Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	10	6 Days	24 Oct 2022	29 Oct 2022	8	31 Oct 2022
Sprint-2	10	6 Days	31 Oct 2022	05 Nov 2022	12	5 Nov 2022
Sprint-3	12	6 Days	07 Nov 2022	12 Nov 2022	10	16 Nov 2022
Sprint-4	10	6 Days	14 Nov 2022	19 Nov 2022	12	24 Nov 2022

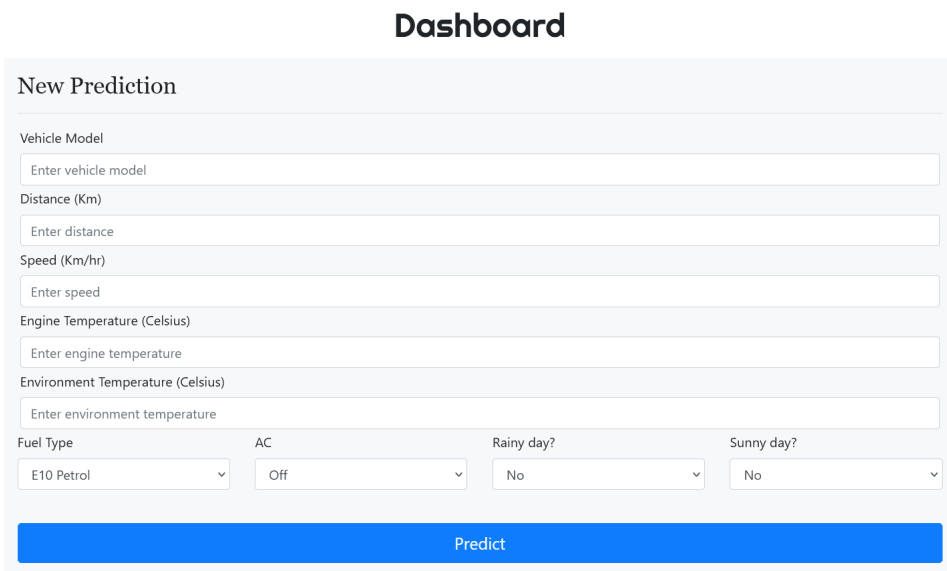
6.3 REPORTS FROM JIRA

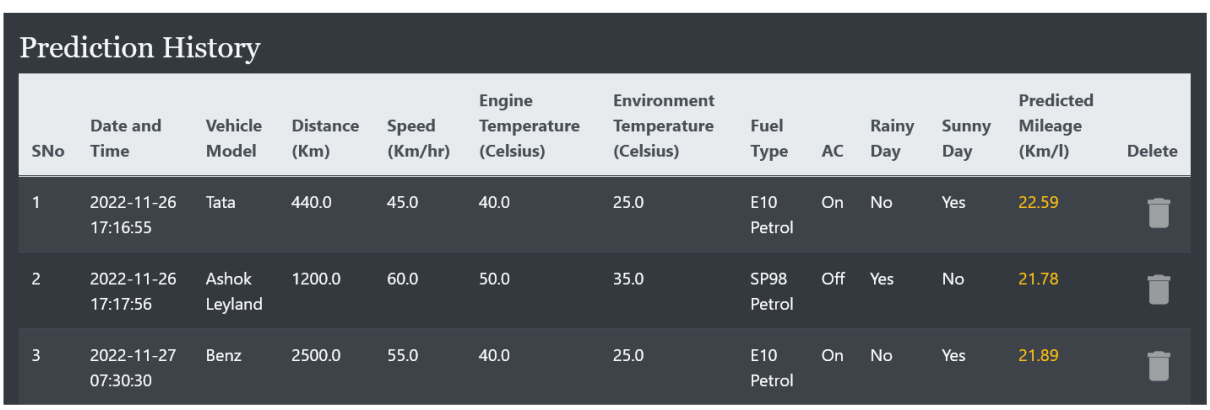


7. CODING AND SOLUTIONING

7.1 FEATURE 1

We use random forest algorithm to predict the fuel consumption of the fleet vehicle based on distance traveled, speed, engine temperature, environment temperature, fuel type, AC status and weather. Previous predictions are also show to the respective users with the option delete it.



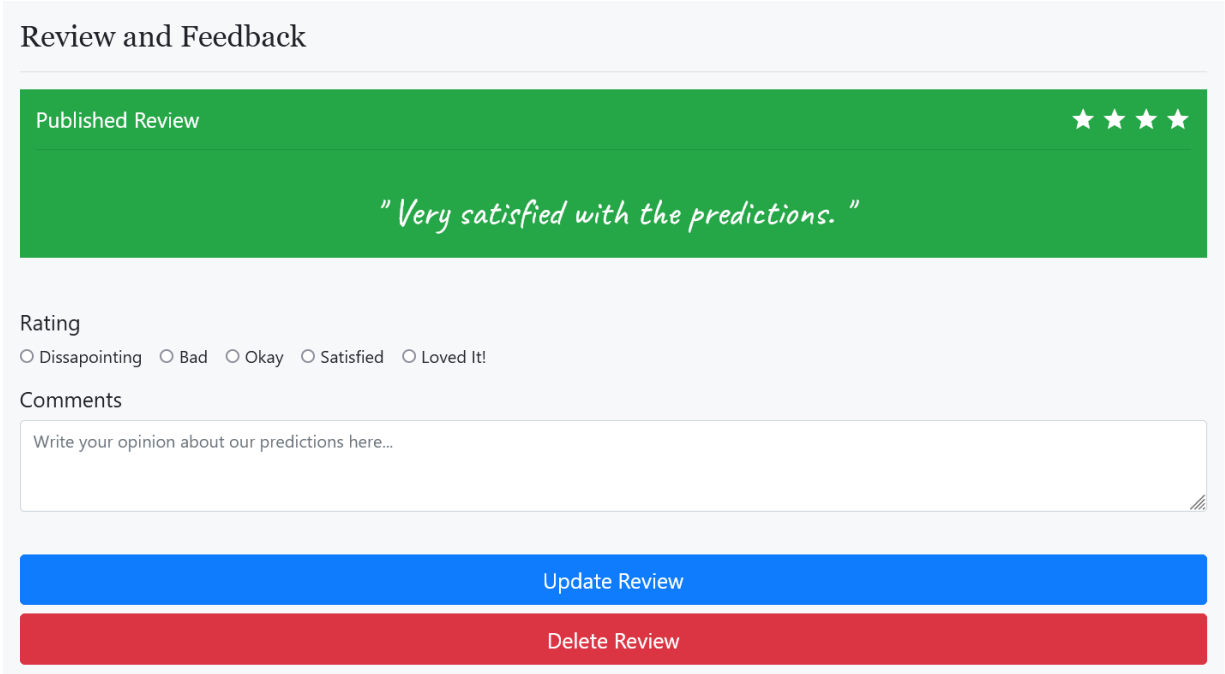


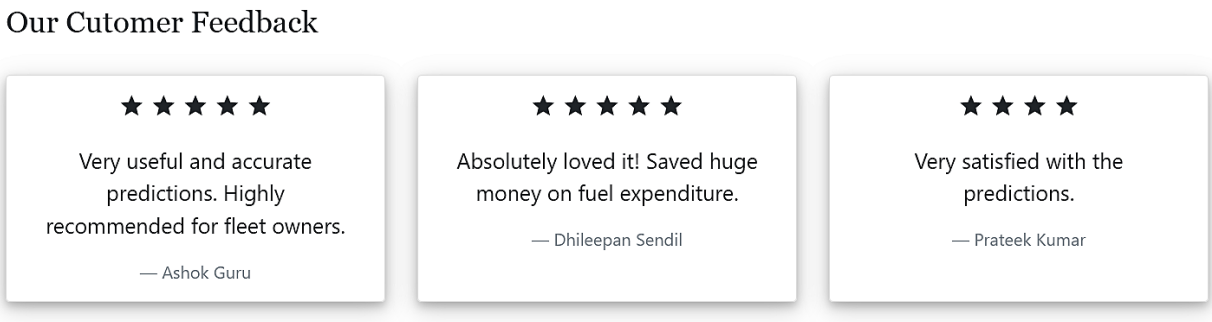
Prediction History

SNo	Date and Time	Vehicle Model	Distance (Km)	Speed (Km/hr)	Engine Temperature (Celsius)	Environment Temperature (Celsius)	Fuel Type	AC	Rainy Day	Sunny Day	Predicted Mileage (Km/l)	Delete
1	2022-11-26 17:16:55	Tata	440.0	45.0	40.0	25.0	E10 Petrol	On	No	Yes	22.59	
2	2022-11-26 17:17:56	Ashok Leyland	1200.0	60.0	50.0	35.0	SP98 Petrol	Off	Yes	No	21.78	
3	2022-11-27 07:30:30	Benz	2500.0	55.0	40.0	25.0	E10 Petrol	On	No	Yes	21.89	

7.1 FEATURE 2

Users can give feedback and comments about the website. Top 3 positive reviews are displayed on the homepage. The user can also edit or delete his review.





7.2 DATABASE SCHEMA (FLask-SQLAlchemy)

```python
class Prediction(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    vmodel=db.Column(db.String(100))
    distance = db.Column(db.Float)
    speed = db.Column(db.Float)
    temp_inside = db.Column(db.Float)
    temp_outside = db.Column(db.Float)
    gas_type = db.Column(db.Integer)
    ac = db.Column(db.Integer)
    rain = db.Column(db.Integer)
    sun = db.Column(db.Integer)
    consume = db.Column(db.Float)
    date = db.Column(db.DateTime(timezone=True), default=func.now())
    user_id = db.Column(db.Integer, db.ForeignKey('user.id'))

class User(db.Model, UserMixin):
    id = db.Column(db.Integer, primary_key=True)
    email = db.Column(db.String(150), unique=True)
    password = db.Column(db.String(30))
    first_name = db.Column(db.String(30))
    last_name = db.Column(db.String(30))
    rating = db.Column(db.Integer)
    text = db.Column(db.String(300))
    predictions = db.relationship('Prediction')
```

7.3 DATASET

1	distance	consume	speed	temp_insic	temp_outs	specials	gas_type	AC	rain	sun	refill liters	refill gas
2	28	5	26	21,5	12		E10	0	0	0	45	E10
3	12	4,2	30	21,5	13		E10	0	0	0		
4	11,2	5,5	38	21,5	15		E10	0	0	0		
5	12,9	3,9	36	21,5	14		E10	0	0	0		
6	18,5	4,5	46	21,5	15		E10	0	0	0		
7	8,3	6,4	50	21,5	10		E10	0	0	0		
8	7,8	4,4	43	21,5	11		E10	0	0	0		
9	12,3	5	40	21,5	6		E10	0	0	0		
10	4,9	6,4	26	21,5	4		E10	0	0	0		
11	11,9	5,3	30	21,5	9		E10	0	0	0		
12	12,4	5,6	42	21,5	4		E10	0	0	0		
13	11,8	4,6	38	21,5	0		E10	0	0	0		
14	12,3	5,9	59	21,5	10		E10	0	0	0		
15	24,7	5,1	58	21,5	12		E10	0	0	0		
16	12,4	4,7	46	21,5	11		E10	0	0	0		
17	17,3	5,1	24	21,5	5		E10	0	0	0		
18	33,4	5,6	36	21,5	3		E10	0	0	0		
19	11,8	5,1	32	21,5	3		E10	0	0	0		
20	25,9	4,9	39	21,5	8		E10	0	0	0		
21	11,8	4,7	40	21,5	4		E10	0	0	0		
22	25,3	5,5	32	21,5	3		E10	0	0	0		
23	14,2	5,9	38	21,5	1		E10	0	0	0		
24	17,9	5,7	37	21,5	1		E10	0	0	0		
25	11,8	4,7	36	21,5	1		E10	0	0	0		
26	12,3	5,9	62	21,5	6		E10	0	0	0		
27	12,4	4,1	57	21,5	9		E10	0	0	0		
28	18,4	5,7	21	22,5	2		E10	0	0	0		
29	18,4	5,8	28	21,5	3		E10	0	0	0		

8. TESTING

8.1 TEST CASE

Date	03-Nov-22
Team ID	PNT2022TMID35586
Project Name	Project - Trip based modelling of fuel consuption in modern fleet vehicles
Maximum Marks	4 marks

Test case ID	Feature Type	Component	Test Scenario	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Executed By
1	UI	Home Page	Verify if all UI elments are working properly and customer reviews are displayed.	1) Open the page 2) Check if all the UI elements are visible	127.0.0.1	The Home page must be displayed properly with top 3 customer reviews.	Working as expected	PASS	Prateek Kumar
2	Functional / UI	Login Page	Check if only verified user is able to log in.	1) Open the page 2) Enter email and password 3) Click on login and wait for dashboard page to load	Email and password	Verify email and password and redirect to dashboard	Working as expected	PASS	Dhileepan S
3	Functional / UI	Sign Up Page	Check if all field are correctly validated and user is able to sign up.	1) Open the page 2) Enter necessary details 3) Click sign up and wait for login page to load	Email, first name, last name, pasword, confirm password	User is denied new account if email already exists else create a new account.	Working as expected	PASS	Ashok G
4	Functional / UI	Dashboard Page	Check if all fields of prediction form are accepted and predicted result is displayed.	1) Open the page 2) Give the necessary details 3) Check if prediction result is displayed	Distance, speed, fuel type, ac, sunny day, rainy day, engine and environment temperatire	Predicted value must be displayed and should reflect in prediction history table	Working as expected	PASS	Dhinu Praveen
5	Functional / UI	Dashboard Page	Check if user is able to give, edit or delete his feedback.	1) Open the page 2) Enter/Check review details 3) Click Publish/Update/Delete button 4) Check if respective action is performed	Rating, comment	Review must be displayed/deleted or edited properly.	Working as expected	PASS	Prateek Kumar

8.2 USER ACCEPTANCE TESTING

2. Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	1	0	1	0	2
Duplicate	0	0	0	0	0
External	0	0	2	0	2
Fixed	4	1	0	1	6
Not Reproduced	0	0	0	1	1
Skipped	0	0	0	1	1
Won't Fix	1	0	1	0	2
Totals	6	1	4	3	14

3. Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
UI	5	0	0	5
Functional	4	0	0	5

9. RESULTS

9.1 PERFORMANCE METRICS

1) Linear Regression

In [16]:
```
from sklearn.linear_model import LinearRegression
model=LinearRegression()
model.fit(x_train,y_train)
y_pred=model.predict(x_test)
```

In [18]:
```
find_accuracy(y_test,y_pred)
```

```
Results of sklearn.metrics:
MAE: 0.5746966611396568
MSE: 0.5479636225308521
RMSE: 0.7402456501262619
R-Squared: 0.12354862570779879
```

2) Decision Tree

In [27]:
```
from sklearn.tree import DecisionTreeRegressor as DTR
dtr=DTR(random_state=0)
dtr.fit(x_train,y_train)
```

Out[27]:
```
DecisionTreeRegressor(random_state=0)
```

In a Jupyter environment, please rerun this cell to show the HT
On GitHub, the HTML representation is unable to render, please

In [29]:
```
y_pred=dtr.predict(x_test)
find_accuracy(y_test,y_pred)
```

```
Results of sklearn.metrics:
MAE: 0.5487179487179489
MSE: 0.5564102564102564
RMSE: 0.7459291229133345
R-Squared: 0.11003848823269613
```

3) Random Forest

In [24]:
```
from sklearn.ensemble import RandomForestRegressor as rf
rf_model=rf(n_estimators=100,random_state=0)
rf_model.fit(x_train,y_train)
```

Out[24]:
```
RandomForestRegressor(random_state=0)
```

In a Jupyter environment, please rerun this cell to show the HTML
On GitHub, the HTML representation is unable to render, please tr

In [25]:
```
y_pred=rf_model.predict(x_test)
find_accuracy(y_test,y_pred)
```

```
Results of sklearn.metrics:
MAE: 0.405172466422466
MSE: 0.2784897491750975
RMSE: 0.5277212798202262
R-Squared: 0.55456400141405
```

10.1 ADVANTAGES

- Save fuel costs by not over-paying the drivers for refueling.
- Track fraudulent activities such as fuel hoarding, using vehicle for personal use etc.
- Helps fleet owners to track and visualize trip details of each vehicle.
- Will help to decide if a particular job from the client is profitable or not.

10.2 DISADVANTAGES

- Prediction may not be accurate as it depends on multiple factors
- Dataset is limited and currently insufficient to train a robust model
- Road conditions, driving style, braking, acceleration, vehicle condition, total weight (freight + vehicle) etc. are to be also considered for prediction fuel consumption which is not the case for the current model.

11. CONCLUSION

This paper presented a machine learning model that can be conveniently developed for each heavy vehicle in a fleet. The model relies on eight predictors: distance, speed, engine temperature, environment temperature, fuel type, AC on or off, sunny day or not and rainy day or not. All of the predictors of the model are derived from vehicle speed and weather. These variables are readily available from telematics devices that are becoming an integral part of connected vehicles.

Moreover, the predictors can be easily computed on-board from these two variables. The model predictors are aggregated over a fixed distance traveled instead of a fixed time interval. This mapping of the input space to the distance domain aligns with the domain of the target output, and produced a machine learning model for fuel consumption with an RMSE. Different models such as linear regression, decision tree and random forest was trained on the pre processed dataset. Random forest showed

least error and hence was used in the flask app for prediction.

12. FUTURE SCOPE

The prediction can be improved in future by considering various other factors such as road conditions, driving style, braking, acceleration, vehicle condition, total weight (freight + vehicle) etc. Moreover, the current vehicles that are powered by gasoline pollute, but as technologies improve and the human way of life changes alternatively powered vehicles enter the automotive industry. These vehicles developed to achieve better gas mileage and to help slow the production of the gasses that cause Global Warming. The hybrid vehicle is one of the newest and most popular alternatively powered vehicles. Air pollution is the term used to describe any harmful gases in the air we breathe. Pollution can be emitted from natural sources such as volcanoes, but humans are responsible for much of the pollution in our atmosphere. In future electric cars combined with power of data science will help us achieve most efficient vehicles with least pollutants.

13. APPENDIX

13.1. Source Code

File: main.py

```
from website import create_app
import os

app = create_app()
port=os.getenv('VCAP_APP_PORT', '8080')

if __name__ == '__main__':
```

```
    app.run(debug=True, host='0.0.0.0', port=port)
```

File: views.py

```
from flask import Blueprint, render_template, request, flash, jsonify
from flask_login import login_required, current_user
from .models import Prediction,User
from . import db
import json
import pickle
import os
import pandas as pd

#importing model from pickle file
my_dir = os.path.dirname(__file__)
pickle_file_path = os.path.join(my_dir, 'static/IBM_RF_model.pickle')
with open(pickle_file_path, 'rb') as f:
    LRmodel = pickle.load(f)

views = Blueprint('views', __name__)

db description=db.engine.execute("describe")
@views.route('/')
def home():
    reviews=db.engine.execute("Select first_name,last_name,text,rating from
User Where not text='None' Order By rating DESC limit 3;")
    return render_template("home.html", user=current_user,
reviews=reviews)

@views.route('/dashboard', methods=['GET', 'POST'])
@login_required
def dashboard():
    result=1
    display="none"
    reviewdisplay="block"
    reviewbtn="Update Review"
```

```
    userreview=""
    stars=0

    if current_user.text is None:
        reviewdisplay="none"
        reviewbtn="Publish"
    else:
        user review=current_user.text
        stars=int(current_user.rating)

    if request.method == 'POST':
        if 'newprediction' in request.form:
            vmodel=request.form.get('vmodel')
            distance = request.form.get('distance')
            speed = request.form.get('speed')
            temp_inside = request.form.get('temp_inside')
            temp_outside = request.form.get('temp_outside')
            ac = request.form.get('ac')
            gas_type = request.form.get('gas_type')
            rain = request.form.get('rain')
            sun = request.form.get('sun')

            if len(vmodel) < 1:
                flash('Vehicle model field can\'t be empty.',
category='error')
            elif len(distance) < 1:
                flash('Distance field can\'t be empty.', category='error')
            elif len(speed) < 1:
                flash('Speed field can\'t be empty.', category='error')
            elif len(temp_inside) < 1:
                flash('Engine temperature field can\'t be empty.',
category='error')
            elif len(temp_outside) < 1:
                flash('Environment temperature field can\'t be empty.',
category='error')
            elif gas_type is None:
```

```
                flash('Fuel Type is not selected.', category='error')
            elif ac is None:
                flash('AC field is not selected.', category='error')
            elif rain is None:
                flash('Rainy day field is not selected.', category='error')
            elif sun is None:
                flash('Sunny day field is not selected.', category='error')
            else:

list=[[float(distance),float(speed),float(temp_inside),float(temp_outside),int
(gas_type),int(ac),int(rain),int(sun)]]
                pred=pd.DataFrame(list)
                result=LRmodel.predict(pred)[0]

                new_pred = Prediction(
                    vmodel=vmodel,
                    distance=float(distance),
                    speed=float(speed),
                    temp_inside=float(temp_inside),
                    temp_outside=float(temp_outside),
                    gas_type=int(gas_type),
                    ac=int(ac),
                    rain=int(rain),
                    sun=int(sun),
                    consume=result,
                    user_id=current_user.id
                    )
                db.session.add(new_pred)
                db.session.commit()
                display="block"
                flash('Prediction successfull !', category='success')

        elif 'submitreview' in request.form:
            rating=request.form.get('rating')
            reviewtext=request.form.get('comment')
```

```
            if rating is None:
                flash('Rating is not selected.', category='error')
            elif len(reviewtext)<3:
                flash('Review is too short.', category='error')
            else:
                current_user.rating=int(rating)
                current_user.text=str(reviewtext)
                db.session.commit()
                reviewdisplay="block"
                reviewbtn="Update Review"
                userreview=reviewtext
                stars=int(rating)
                flash('Review posted!', category='success')

    return render_template("dashboard.html", user=current_user,
prediction_result=result, display=display,
reviewdisplay=reviewdisplay,userreview=userreview,reviewbtn=reviewbtn,star
s=stars)


@views.route('/delete-pred', methods=['POST'])
def delete_pred():
    pred = json.loads(request.data)
    predId = pred['predID']
    pred = Prediction.query.get(predId)
    if pred:
        if pred.user_id == current_user.id:
            db.session.delete(pred)
            db.session.commit()
            flash('Prediction deleted!', category='error')
    return jsonify({})

@views.route('/delete-review', methods=['POST'])
def delete_review():
    cuser = json.loads(request.data)
    userId = cuser['userID']
```

```
    cuser = User.query.get(userId)
    if cuser:
        cuser.rating=None
        cuser.text=None
        db.session.commit()
        flash('Review deleted!', category='error')
    return jsonify({})
```

File: auth.py

```
from flask import Blueprint, render_template, request, flash, redirect,
url_for
from .models import User
from werkzeug.security import generate_password_hash, check_password_hash
from . import db
from flask_login import login_user, login_required, logout_user,
current_user
# from flask_mail import Message
# from . import mail
# from random import randint
import re

emailregex = re.compile(r'([A-Za-z0-9]+[.-_])*[A-Za-z0-9]+@[A-Za-z0-9-
]+(\.[A-Z|a-z]{2,})+')

auth = Blueprint('auth', __name__)


@auth.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        email = request.form.get('email')
        password = request.form.get('password')
        user = User.query.filter_by(email=email).first()
        if user:
```

```
            if check_password_hash(user.password, password):
                flash('Logged in successfully!', category='success')
                login_user(user, remember=True)
                return redirect(url_for('views.dashboard'))
            else:
                flash('Incorrect password, try again.', category='error')
        else:
            flash('Email does not exist. Please sign up to continue.',
category='error')

    return render_template("login.html", user=current_user)


@auth.route('/logout')
@login_required
def logout():
    logout_user()
    return redirect(url_for('auth.login'))


@auth.route('/sign-up', methods=['GET', 'POST'])
def sign_up():
    if request.method == 'POST':
        email = request.form.get('email')
        first_name = request.form.get('firstName')
        last_name = request.form.get('lastName')
        password1 = request.form.get('password1')
        password2 = request.form.get('password2')

        user = User.query.filter_by(email=email).first()
        if user:
            flash('Email already exists.', category='error')
        elif not re.fullmatch(emailregex, email):
            flash('Please enter valid email.', category='error')
        elif len(first_name) < 1:
            flash('First name cannot be blank.', category='error')
```

```
        elif len(last_name) < 1:
            flash('Last name cannot be blank.', category='error')
        elif password1 != password2:
            flash('Passwords don\'t match.', category='error')
        elif len(password1) < 8:
            flash('Password must be at least 8 characters.',
category='error')
        else:
            # otp_generated=randint(1000,9999)
            new_user = User(email=email, first_name=first_name,
last_name=last_name, password=generate_password_hash(password1,
method='sha256'))
            # otp=otp_generated,verified=False)
            db.session.add(new_user)
            db.session.commit()
            flash('Account created successfully. Login to
continue.',category='success')
            return redirect(url_for('auth.login'))

    return render_template("sign_up.html", user=current_user)
```

File: models.py

```
from . import db
from flask_login import UserMixin
from sqlalchemy.sql import func


class Prediction(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    vmodel=db.Column(db.String(100))
    distance = db.Column(db.Float)
    speed = db.Column(db.Float)
    temp_inside = db.Column(db.Float)
    temp_outside = db.Column(db.Float)
```

```
    gas_type = db.Column(db.Integer)
    ac = db.Column(db.Integer)
    rain = db.Column(db.Integer)
    sun = db.Column(db.Integer)
    consume = db.Column(db.Float)
    date = db.Column(db.DateTime(timezone=True), default=func.now())
    user_id = db.Column(db.Integer, db.ForeignKey('user.id'))

class User(db.Model, UserMixin):
    id = db.Column(db.Integer, primary_key=True)
    email = db.Column(db.String(150), unique=True)
    password = db.Column(db.String(30))
    first_name = db.Column(db.String(30))
    last_name = db.Column(db.String(30))
    rating = db.Column(db.Integer)
    text = db.Column(db.String(300))
    predictions = db.relationship('Prediction')
```

File: __init__.py

```
from flask import Flask
from flask_sqlalchemy import SQLAlchemy
import os
from flask_login import LoginManager
# from flask_mail import Mail

db = SQLAlchemy()
DB_NAME = "database.db"
# mail= Mail()

def create_app():
    app = Flask(__name__)
    app.config['MAIL_SERVER']='smtp.gmail.com'
    app.config['MAIL_PORT']=465
    app.config['MAIL_USERNAME']='fleetify.official@gmail.com'
    app.config['MAIL_PASSWORD']='velrsrkermbvehyh'
```

```
    app.config['MAIL_USE_TLS']=False
    app.config['MAIL_USE_SSL']=True

    app.config['SECRET_KEY'] = os.urandom(12)
    app.config['SQLALCHEMY_DATABASE_URI'] = f'sqlite:///{DB_NAME}'

    db.init_app(app)
    # mail.init_app(app)

    from .views import views
    from .auth import auth

    app.register_blueprint(views, url_prefix='/')
    app.register_blueprint(auth, url_prefix='/')

    from .models import User

    with app.app_context():
        db.create_all()

    login_manager = LoginManager()
    login_manager.login_view = 'auth.login'
    login_manager.init_app(app)

    @login_manager.user_loader
    def load_user(id):
        return User.query.get(int(id))

    return app
```

File: base.html

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
```

```
    <meta name="viewport" content="width=device-width, initial-scale=1" />

    <link
      rel="stylesheet"

href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min
.css"
      integrity="sha384-
Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh"
      crossorigin="anonymous"
    />
    <link
      rel="stylesheet"
      href="https://stackpath.bootstrapcdn.com/font-
awesome/4.7.0/css/font-awesome.min.css"
      crossorigin="anonymous"
    />



    <link href="https://gitcdn.github.io/bootstrap-
toggle/2.2.2/css/bootstrap-toggle.min.css" rel="stylesheet">
    <script src="https://gitcdn.github.io/bootstrap-
toggle/2.2.2/js/bootstrap-toggle.min.js"></script>

    <link rel="stylesheet" href="../static/style.css"/>
    <link href="https://fonts.googleapis.com/icon?family=Material+Icons"
rel="stylesheet"/>

    <title>{% block title %}Home{% endblock %}</title>
  </head>
  <body>
    <nav class="navbar navbar-expand-lg bg-light">
      <button
        class="navbar-toggler"
        type="button"
        data-toggle="collapse"
```

```
        data-target="#navbar"
      >
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbar">
        <a href="/"><span class="material-icons md-36 pull-left">
          local_shipping
          </span>
        </a>
        <div class="navbar-nav ml-auto">
          <a class="nav-item nav-link" id="home" href="/">Home</a>
          {% if user.is_authenticated %}
          <a class="nav-item nav-link" id="dashboard"
href="/dashboard">Dashboard</a>
          <a class="nav-item nav-link text-warning" id="logout"
href="/logout">Hello {{user.first_name}}, Logout</a>
          {% else %}
          <a class="nav-item nav-link" id="login" href="/login">Login</a>
          <a class="nav-item nav-link" id="signUp" href="/sign-up">Sign
Up</a>
          {% endif %}
        </div>
      </div>
    </nav>
    <div id="wrap">
      <div id="main">
        {% with messages = get_flashed_messages(with_categories=true) %} {%
if
          messages %} {% for category, message in messages %} {% if
category ==
          'error' %}
          <div class="alert alert-danger alter-dismissable fade show"
role="alert">
            {{ message }}
            <button type="button" class="close" data-dismiss="alert">
              <span aria-hidden="true">&times;</span>
```

```
            </button>
          </div>
          {% else %}
          <div class="alert alert-success alter-dismissable fade show"
role="alert">
            {{ message }}
            <button type="button" class="close" data-dismiss="alert">
              <span aria-hidden="true">&times;</span>
            </button>
          </div>
          {% endif %} {% endfor %} {% endif %} {% endwith %}

          {% block content %} {% endblock %}
      </div>
    </div>

<footer class="page-footer bg-primary text-white">
  <div class="footer-copyright text-center py-3">Fleetify © 2022
Copyright:
  </div>
</footer>

    <script
      src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
      integrity="sha384-
KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
      crossorigin="anonymous"
    ></script>
    <script

src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min
.js"
      integrity="sha384-
ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
      crossorigin="anonymous"
    ></script>
```

```
    <script

src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
      integrity="sha384-
JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
      crossorigin="anonymous"
    ></script>

    <script
      type="text/javascript"
      src="{{ url_for('static', filename='index.js') }}"
    ></script>
  </body>
</html>
```

File: home.html

```
{% extends "base.html" %} {% block title %}Home{% endblock %} {% block
content
%}
<div class="d-flex" id="bgpic">
</div>
<div align="center" style="font-family:'Anton'; font-size:
3rem;margin:2rem;">Want to improve the profits and performance of your fleet
vehicles?<br>Use our best in class technology for accurate predictions.
<br><span class="text-primary" style="font-size: 2.5rem">Sign up
now!</span></div>
<hr>
<div class="container">
  <br>
  <h3>Our Cutomer Feedback</h3>
  <br>
  <div class="card-deck">
    {% for ureview in reviews%}
      <blockquote class="blockquote card">
```

```
        <div>
          {% for stars in range(ureview[3]) %}
        <span class="material-icons">
          star
        </span>
        {% endfor %}
        </div>
        <p class="mb-0 card-text" style="padding:1rem">{{ureview[2]}}</p>
        <footer class="blockquote-footer">{{ureview[0]+'
'+ureview[1]}}</footer>
      </blockquote>
    {%endfor%}
  </div>
</div>
{% endblock %}
```

File: dashboard.html

```
{% extends "base.html" %} {% block title %}Dashboard{% endblock %} {%
block content
%}
<h1 align="center">Dashboard</h1>
<div class="container pt-3 bg-light">
  <form method="POST">
  <h3>New Prediction</h3>
  <hr>
  <div class="form-group row">
    <label for="vmodel">Vehicle Model</label>
    <input
      type="text"
      class="form-control"
      id="vmodel"
      name="vmodel"
      placeholder="Enter vehicle model"
    />
  </div>
```

```
<div class="form-group row">
  <label for="email">Distance (Km)</label>
  <input
    type="text"
    class="form-control"
    id="distance"
    name="distance"
    placeholder="Enter distance"
  />
</div>
<div class="form-group row">
  <label for="speed">Speed (Km/hr)</label>
  <input
    type="text"
    class="form-control"
    id="speed"
    name="speed"
    placeholder="Enter speed"
  />
</div>
<div class="form-group row">
  <label for="temp_inside">Engine Temperature (Celsius)</label>
  <input
    type="text"
    class="form-control"
    id="temp_inside"
    name="temp_inside"
    placeholder="Enter engine temperature"
  />
</div>
<div class="form-group row">
  <label for="temp_outside">Environment Temperature (Celsius)</label>
  <input
    type="text"
    class="form-control"
    id="temp_outside"
```

```
    name="temp_outside"
    placeholder="Enter environment temperature"
  />
</div>
<div class="row">
  <div class="form-group col">
    <label for="gas_type">Fuel Type</label>
    <select class="form-control" id="gas_type" name="gas_type">
      <option value="0">E10 Petrol</option>
      <option value="1">SP98 Petrol</option>
    </select>
  </div>

  <div class="form-group col">
    <label for="ac">AC </label>
    <select class="form-control" id="ac" name="ac">
      <option value="0">Off</option>
      <option value="1">On</option>
    </select>
  </div>

  <div class="form-group col">
    <label for="rain">Rainy day?</label>
    <select class="form-control" id="rain" name="rain">
      <option value="0">No</option>
      <option value="1">Yes</option>
    </select>
  </div>

  <div class="form-group col">
    <label for="sun">Sunny day? </label>
    <select class="form-control" id="sun" name="sun">
      <option value="0">No</option>
      <option value="1">Yes</option>
    </select>
  </div>
```

```
    </div>
    <br/>
    <button type="submit" class="btn btn-primary btn-lg btn-block"
name="newprediction">Predict</button>
    <br/>
  </form>
  <br/>
  <div class="container pt-3" style="display:{{display}}">
    <h4>Predicted fuel consumption is <span class="text-
danger">{{prediction_result|round(2)}}</span> (litres/100Km).
      <br>
        Predicted mileage is <span class="text-
danger">{{(100/prediction_result)|round(2)}}</span> Km per litre.
    </h4>
  </div>
</div>
<hr>
<div class="container bg-dark pt-3 text-light" >
  <h3>Prediction History</h3>
  <table class="table table-hover table-striped table-dark"
id="historytable">
    <tr>
    <thead class="thead-light">
      <th>SNo</th>
      <th>Date and Time</th>
      <th>Vehicle Model</th>
      <th>Distance (Km)</th>
      <th>Speed (Km/hr)</th>
      <th>Engine Temperature (Celsius)</th>
      <th>Environment Temperature (Celsius)</th>
      <th>Fuel Type</th>
      <th>AC</th>
      <th>Rainy Day</th>
      <th>Sunny Day</th>
      <th>Predicted Mileage (Km/l)</th>
      <th>Delete</th>
```

```
  </thead>
</tr>
{% for pred in user.predictions %}
<tr>
  <td class="counterCell"></td>
  <td>{{pred.date}}</td>
  <td>{{pred.vmodel}}</td>
  <td>{{pred.distance}}</td>
  <td>{{pred.speed}}</td>
  <td>{{pred.temp_inside}}</td>
  <td>{{pred.temp_outside}}</td>
  <td>{% if pred.gas_type==0 %}
      <span>E10 Petrol</span>
      {% else %}
      <span>SP98 Petrol</span>
      {% endif %}
  </td>
  <td>{% if pred.ac==0 %}
    <span>Off</span>
    {% else %}
    <span>On</span>
    {% endif %}
</td>
<td>{% if pred.rain==1 %}
  <span>Yes</span>
  {% else %}
  <span>No</span>
  {% endif %}
</td>
<td>{% if pred.sun==1 %}
  <span>Yes</span>
  {% else %}
  <span>No</span>
  {% endif %}
</td>
<td class="text-warning">{{(100/pred.consume) | round(2)}}</td>
```

```
  <td class="align-middle">
    <button type="button" class="close" onClick="deleteNote({{ pred.id
}})">
      <span class="material-icons md-light md-36">
        delete
        </span>
    </button>
  </td>
  </tr>
  {%endfor%}
  </table>
</div>
<br>
<hr>
<div class="container pt-3 bg-light">
  <form method="POST">
  <h3>Review and Feedback</h3>
  <hr>
  <div class="container pt-3 bg-success text-white"
style="display:{{reviewdisplay}}">
    <h5>Published Review
      <span class="pull-right">
        {% for stars in range(stars) %}
        <span class="material-icons">
          star
        </span>
        {% endfor %}
      </span>
    </h5>
    <hr>
    <div class="d-flex justify-content-center" style="font-
family:'Caveat';font-size:2.2rem;padding:1rem;">" {{userreview}} "</div>
  </div>
  <br><br>
  <div class="form-group">
    <label class="h5" for="rating">Rating</label>
```

```
  <br>
  <div class="form-check form-check-inline">
    <input class="form-check-input" type="radio" name="rating"
id="inlineRadio1" value="1">
    <label class="form-check-label"
for="inlineRadio1">Dissapointing</label>
  </div>
  <div class="form-check form-check-inline">
    <input class="form-check-input" type="radio" name="rating"
id="inlineRadio2" value="2">
    <label class="form-check-label" for="inlineRadio2">Bad</label>
  </div>
  <div class="form-check form-check-inline">
    <input class="form-check-input" type="radio" name="rating"
id="inlineRadio3" value="3">
    <label class="form-check-label" for="inlineRadio3">Okay</label>
  </div>
  <div class="form-check form-check-inline">
    <input class="form-check-input" type="radio" name="rating"
id="inlineRadio2" value="4">
    <label class="form-check-label" for="inlineRadio2">Satisfied</label>
  </div>
  <div class="form-check form-check-inline">
    <input class="form-check-input" type="radio" name="rating"
id="inlineRadio3" value="5">
    <label class="form-check-label" for="inlineRadio3">Loved It!</label>
  </div>
  </div>

    <div class="form-group">
      <label class="h5" for="comment">Comments</label>
      <textarea
        class="form-control"
        id="comment"
        name="comment"
        maxlength="300"
```

```
        rows="3"
        placeholder="Write your opinion about our predictions here..."
      ></textarea>
    </div>
    <br/>
    <button type="submit" class="btn btn-primary btn-lg btn-block"
name="submitreview">{{reviewbtn}}</button>
    <button type="button" style="display:{{reviewdisplay}}" class="btn
btn-danger btn-lg btn-block" name="deletereview"
onClick="deleteReview({{user.id}})">Delete Review</button>
  </form>
</div>
{% endblock %}
```

File: login.html

```
{% extends "base.html" %} {% block title %}Login{% endblock %} {% block
content
%}
<div class="container">
<form method="POST">
  <h1 align="center">Login</h1>
  <div class="form-group">
    <label for="email">Email Address</label>
    <input
      type="email"
      class="form-control"
      id="email"
      name="email"
      placeholder="Enter email"
    />
  </div>
  <div class="form-group">
    <label for="password">Password</label>
    <input
```

```
      type="password"
      class="form-control"
      id="password"
      name="password"
      placeholder="Enter password"
    />
  </div>
  <br />
  <button type="submit" class="btn btn-primary"
name="loginform">Login</button>
</form>
</div>
{% endblock %}
```

File: signup.html

```
{% extends "base.html" %} {% block title %}Sign Up{% endblock %} {% block
content %}
<div class="container">
<form method="POST">
  <h1 align="center">Sign Up</h1>
  <div class="form-group">
    <label for="email">Email Address</label>
    <input
      type="email"
      class="form-control"
      id="email"
      name="email"
      placeholder="Enter email"
    />
  </div>
  <div class="form-group">
    <label for="firstName">First Name</label>
    <input
```

```
    type="text"
    class="form-control"
    id="firstName"
    name="firstName"
    placeholder="Enter first name"
  />
</div>
<div class="form-group">
  <label for="lastName">Last Name</label>
  <input
    type="text"
    class="form-control"
    id="lastName"
    name="lastName"
    placeholder="Enter last name"
  />
</div>
<div class="form-group">
  <label for="password1">Password</label>
  <input
    type="password"
    class="form-control"
    id="password1"
    name="password1"
    placeholder="Enter password"
  />
</div>
<div class="form-group">
  <label for="password2">Password (Confirm)</label>
  <input
    type="password"
    class="form-control"
    id="password2"
    name="password2"
    placeholder="Confirm password"
  />
```

```
  </div>
  <br />
  <button type="submit" class="btn btn-primary">Submit</button>
</form>
</div>
{% endblock %}
```

File: style.css

```
@import
url('https://fonts.googleapis.com/css2?family=Anton&family=Caveat&family=C
omic+Neue&family=Indie+Flower&family=Righteous&family=Rubik+Dirt&family=Ru
bik+Distressed&display=swap');@import
url('https://fonts.googleapis.com/css2?family=Anton&family=Comic+Neue&fami
ly=Indie+Flower&family=Righteous&family=Rubik+Dirt&family=Rubik+Distressed
&display=swap');

body,html{
    margin: 0px;
    height: 100%;
}

#wrap{
    min-height: 85%;
}

#main{
    overflow: auto;
    padding-bottom: 5rem;
}

#bgpic{
    background: url('../static/fleet.jpg');
    background-position: center;
    background-size: cover;
```

```
    background-repeat: no-repeat;
    height: 60vh;
    width: 100vw;
}

.container.pt-3.bg-light {
    padding:1rem;
    margin-bottom:1rem;
}

.form-group.row{
    margin:0.2rem;
}

#historytable {
    counter-reset: tableCount;
}
.counterCell:before {
    content: counter(tableCount);
    counter-increment: tableCount;
}

h1{
    margin: 1rem;
    font-family: 'Righteous';
}

h3{
    font-family:Georgia, 'Times New Roman', Times, serif;
}
.material-icons.md-light { color: rgba(255, 255, 255, 1); }
.material-icons.md-36 { font-size: 36px; }

.card{
    padding:1rem;
    box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0,
```

```
0.19);
    text-align: center;
}
```

File: index.js

```
function deleteNote(predId) {
  var result = confirm("Are you sure you want to delete this prediction?");
  if (result) {
    fetch("/delete-pred", {
      method: "POST",
      body: JSON.stringify({ predID: predId }),
    }).then((_res) => {
      window.location.href = "/dashboard";
    });
  }
}

function deleteReview(userId) {
  var result = confirm("Are you sure you want to delete your review?");
  if (result) {
    fetch("/delete-review", {
      method: "POST",
      body: JSON.stringify({ userID: userId }),
    }).then((_res) => {
      window.location.href = "/dashboard";
    });
  }
}
```

GITHUB AND PROJECT DEMO LINK

Project URL:

https://python-quiet-civet-vj.eu-gb.mybluemix.net/

GitHub link:

https://github.com/IBM-EPBL/IBM-Project-7801-1658899563

Demo video link:

https://drive.google.com/file/d/16hdYovkV3YAxSPFtX3ZWyTqdOqwQ5dmT/view?usp=share_link