PROJECT REPORT

1. INTRODUCTION

1.1 Project Overview

The **Revolutionizing Liver Care** project aims to transform the early detection and management of liver cirrhosis through the implementation of advanced machine learning techniques for accurate and timely risk prediction. With the growing prevalence of liver-related diseases and the limitations of traditional diagnostic methods, leveraging data-driven approaches has become essential for improving clinical outcomes and enabling more proactive healthcare interventions.

1. Project Objectives:

- Develop a robust machine learning model for early and accurate prediction of liver cirrhosis.
- Implement real-time health data acquisition methods to continuously improve model performance.
- Enhance predictive accuracy and reliability through the integration of diverse clinical and biochemical data sources.
- Provide a user-friendly interface for healthcare professionals to access, interpret, and act on cirrhosis risk predictions.

2. Methodology:

- **Data Collection:** Gather clinical data from various sources, including electronic health records (EHRs), lab results, and patient history related to liver function.
- **Feature Engineering:** Identify relevant features such as liver enzyme levels, bilirubin, albumin, patient age, and lifestyle factors that may influence cirrhosis risk.
- **Machine Learning Model:** Train advanced machine learning models (e.g., neural networks, decision trees, regression models) using labeled patient data to accurately predict the risk of liver cirrhosis.
- **Real-Time Integration:** Implement mechanisms for continuous data feed to update and refine the model in real-time.

3. Expected Outcomes:

- Accurate and early prediction of liver cirrhosis using patient clinical and biochemical data.
- Improved liver disease management and timely interventions for healthcare providers
- Enhanced clinical decision-making through insights derived from the analysis of liver health patterns.

4. **Significance of the Project:**

- **Urban Planning:** Support healthcare professionals in making informed, data-driven decisions for early diagnosis and treatment of liver cirrhosis.
- **Resource Optimization:** Enable better allocation of medical resources by identifying high-risk patients and prioritizing care.
- **Environmental Impact:** Contribute to improved patient outcomes, reduced healthcare costs, and enhanced quality of life through early intervention and preventive care.

5. **Challenges:**

- Ensuring the accuracy, completeness, and seamless integration of heterogeneous clinical data from various healthcare systems.
- Model Adaptability: Designing a machine learning model that can generalize well across diverse patient populations and evolving medical practices.
- Privacy Concerns: Addressing privacy issues related to the collection and use of traffic data.

1.2 Purpose

The **"Revolutionizing Liver Care: Predicting Liver Cirrhosis Using Advanced Machine Learning Techniques"** project serves a critical purpose in advancing modern healthcare by addressing the growing challenge of liver disease diagnosis and management. At its core, the project aims to enhance early detection and risk assessment of liver cirrhosis through the deployment of advanced machine learning techniques. By providing accurate and timely predictions based on clinical, biochemical, and lifestyle data, the project seeks to empower healthcare professionals, medical researchers, and policy-makers with actionable, data-driven insights. This, in turn, supports informed clinical decisions, timely interventions, and better allocation of healthcare resources. The ultimate goal is to improve patient outcomes, reduce long-term treatment costs, and contribute to more proactive and personalized liver disease management. A user-friendly interface ensures that medical personnel can easily interpret and apply predictive insights, making this tool accessible and impactful across various levels of the healthcare system.

2. IDEATION PHASE

2.1 Problem Statement

Liver cirrhosis remains a significant global health concern, with rising prevalence and high mortality rates due to delayed diagnosis and limited access to early intervention tools. Traditional diagnostic methods often rely on invasive procedures or late-stage clinical manifestations, making early detection challenging and inefficient. These conventional approaches frequently lack the ability to process large volumes of heterogeneous clinical data and fail to deliver accurate, real-time predictions that can aid in timely decision-making. Additionally, inconsistencies in medical data collection and integration across healthcare systems further hinder the development of a unified and predictive diagnostic model. As a result, many patients experience delayed treatment, preventable complications, and poor health outcomes. Addressing these challenges is crucial for improving liver disease management and patient care. The "Revolutionizing Liver Care: Predicting Liver Cirrhosis Using Advanced Machine Learning Techniques" **project is initiated to overcome these limitations by utilizing state-of-the-art machine learning algorithms to deliver accurate, early, and scalable cirrhosis risk predictions—ultimately enabling proactive, data-driven healthcare interventions.**

2.2 Empathy Map Canvas

An empathy map is a clear, easy-to-digest visual that captures essential insights into the behaviors, needs, and attitudes of individuals affected by liver cirrhosis—particularly patients, caregivers, and healthcare professionals. It serves as a valuable tool for teams to better understand the lived experiences of those impacted by the disease.

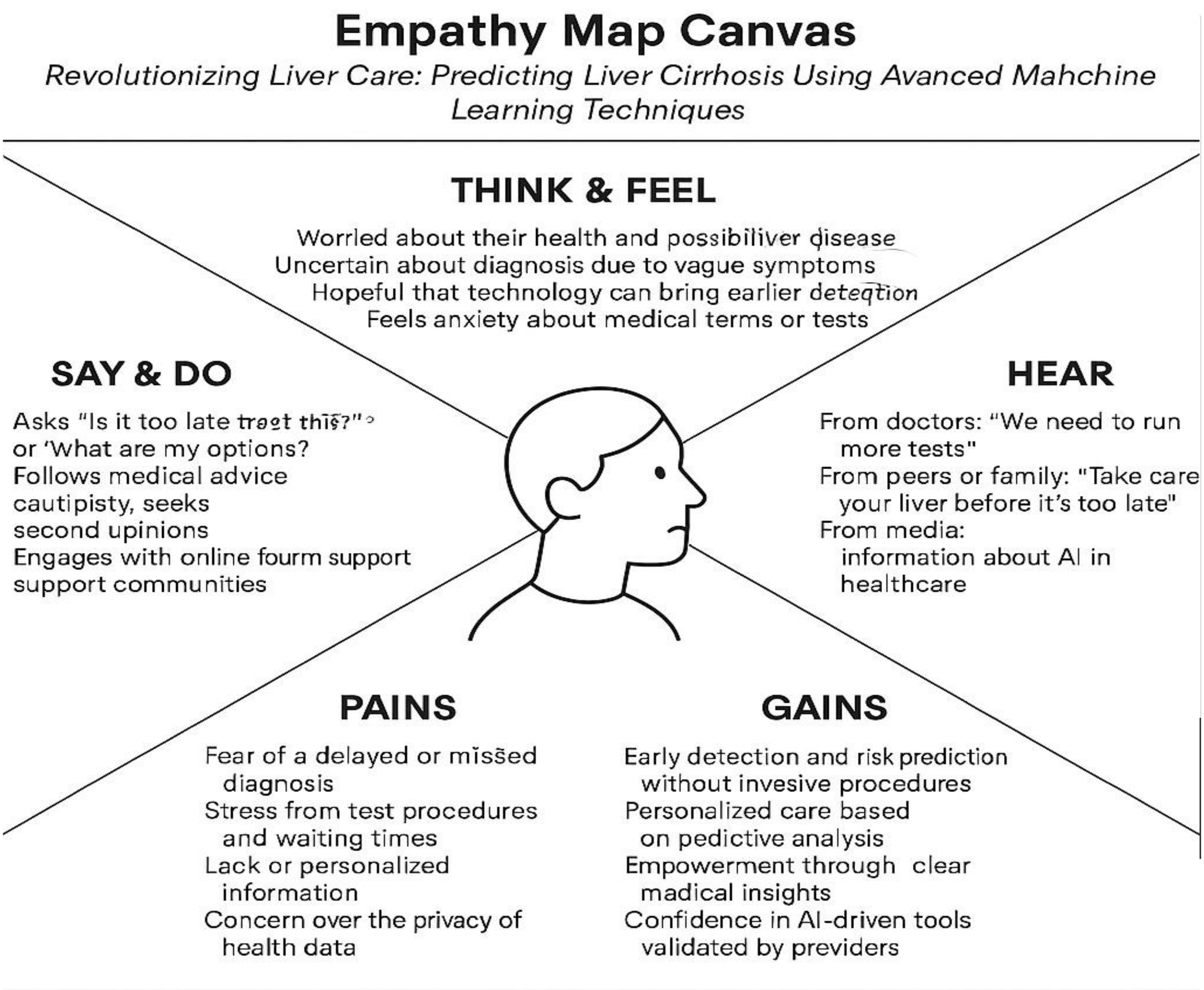
In the context of **"Revolutionizing Liver Care: Predicting Liver Cirrhosis Using Advanced Machine Learning Techniques"**, creating an effective solution requires a deep understanding of the real-world challenges, emotional states, goals, and frustrations of end users. The exercise of building this map helps the team view the situation through the lens of the user, ensuring that the predictive model and clinical interface are designed with empathy, usability, and real-world impact in mind.

Revolutionizing Liver Care: Predicting Liver Cirrhosis Using Advanced Machine Learning Techniques

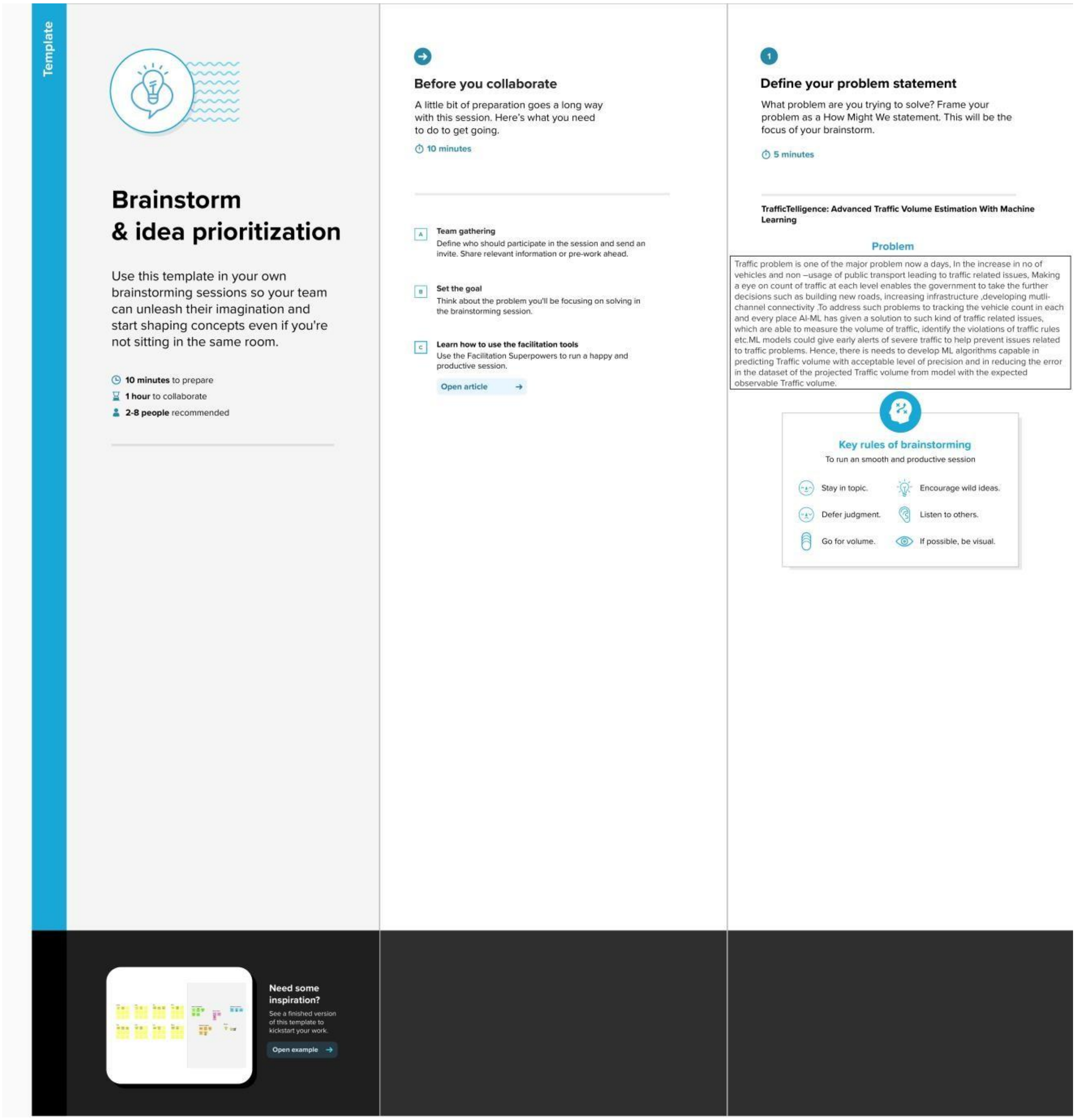
Liver cirrhosis is one of the leading chronic health issues globally, often going undetected until it reaches advanced stages. Contributing factors such as poor lifestyle habits, underlying liver diseases, and late diagnosis significantly increase the burden on healthcare systems and reduce patients' quality of life. Early and accurate prediction of cirrhosis can enable timely medical intervention, personalized treatment, and improved patient outcomes.

To address these challenges, artificial intelligence and machine learning offer powerful solutions. By analyzing vast amounts of clinical and biochemical data, ML models can assess the risk of cirrhosis with high accuracy, helping healthcare professionals make informed decisions. These predictive tools can serve as early warning systems, enabling proactive care, reducing complications, and ultimately transforming the way liver diseases are managed.

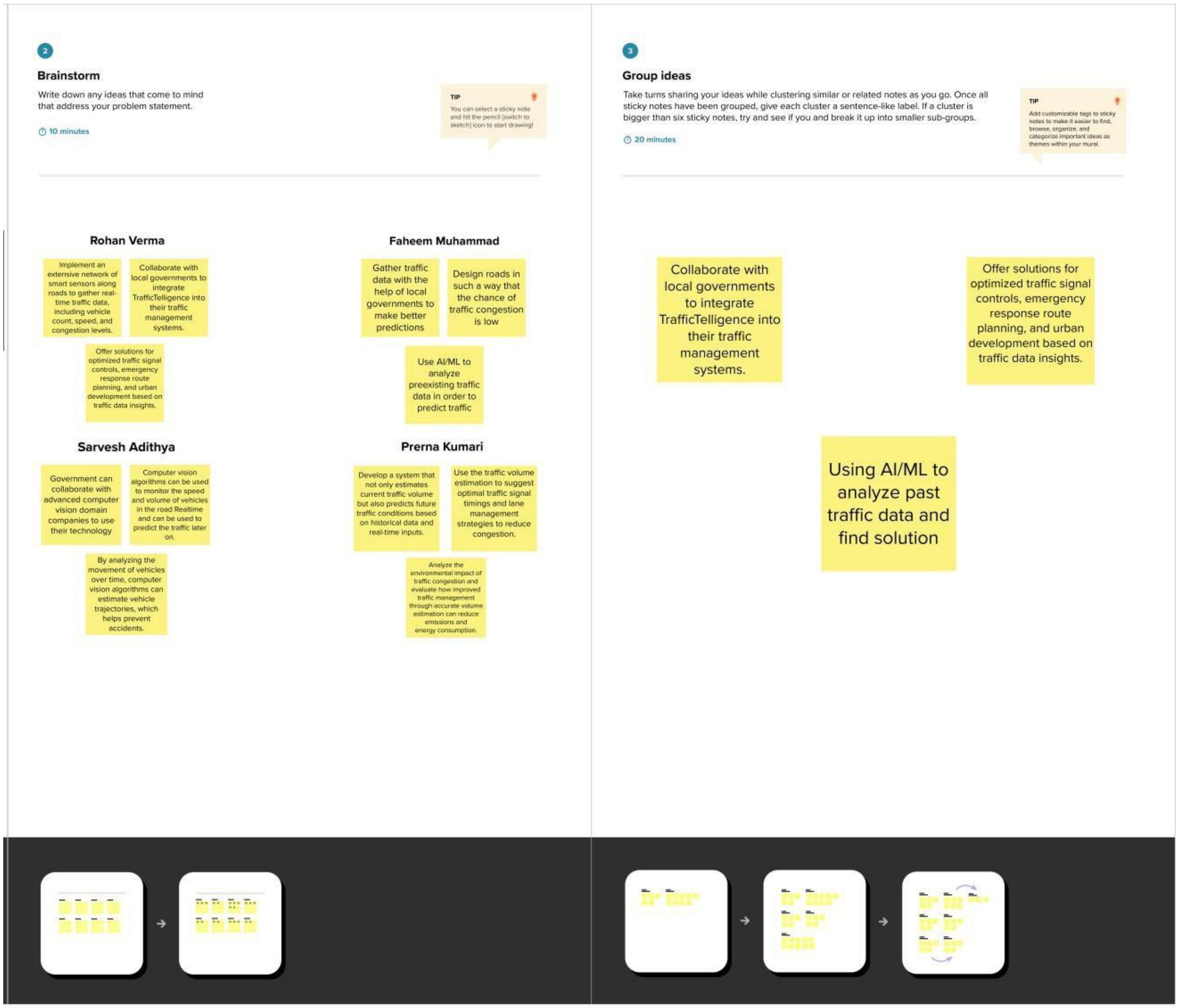
Empathy Canvas Map for the Project:



Step-1 : Team Gathering, Collaboration and Select the Problem Statement

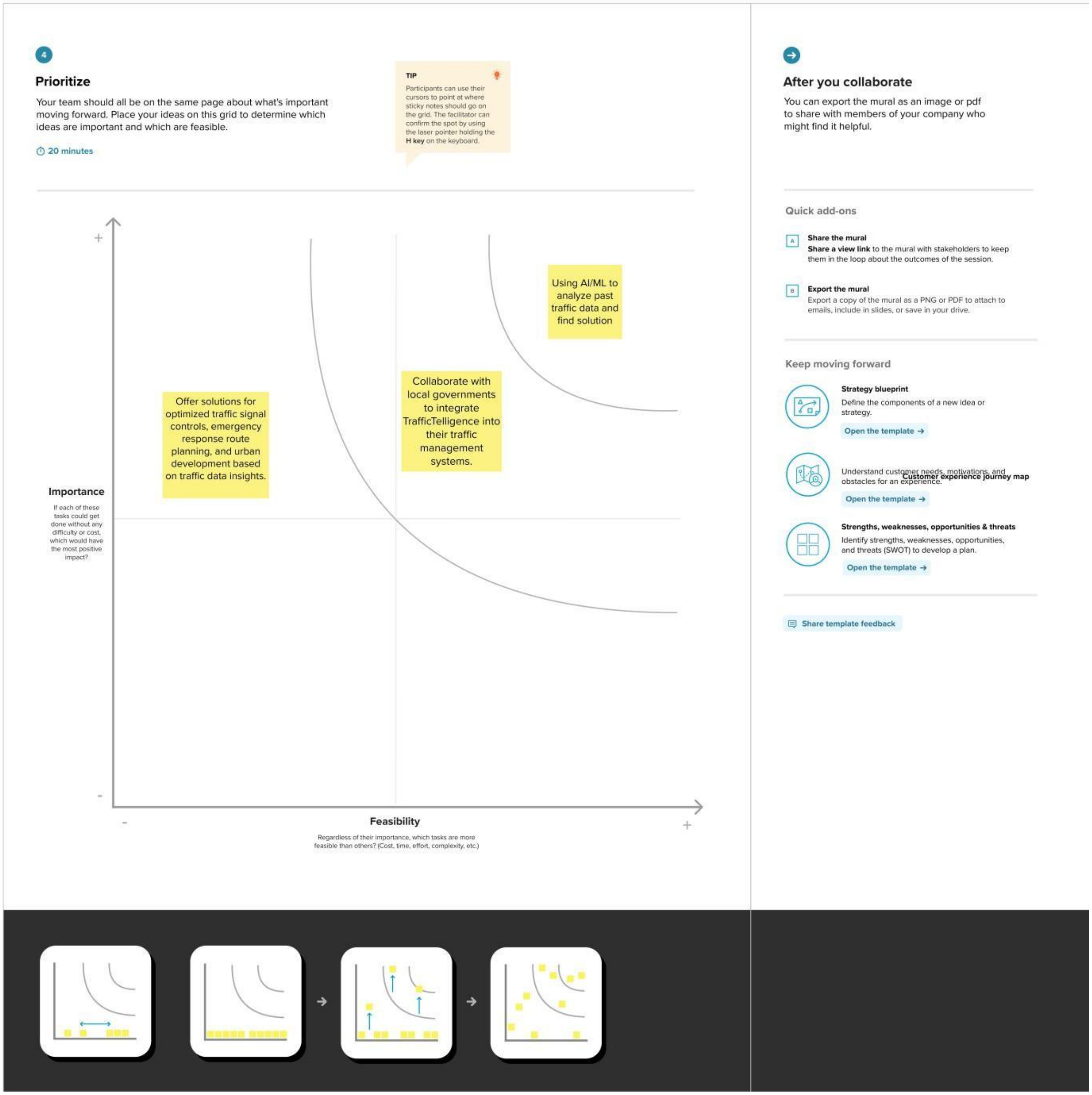


Step-2: Revolutionizing Liver Care, Idea Listing and grouping



Step-3: Idea Prioritization

Idea prioritization is the process of ranking or assessing ideas based on specific criteria such as feasibility, impact, cost, or strategic importance to determine which ideas should be implemented or pursued first.



Using AI/ML to analyze past medical data and find a solution” as our top priority in **Revolutionizing Liver Care: Predicting Liver Cirrhosis Using Advanced Machine Learning Techniques**. Among all the other ideas, this was the most important to us because if the model is not accurate enough, the prediction of liver cirrhosis may not be reliable. Hence, ensuring high model accuracy became our most prioritized goal.

“Collaboration with local healthcare authorities and medical institutions to integrate the liver cirrhosis prediction model into routine clinical workflows.” This was prioritized next because embracing social responsibility is essential—not just for advancing our project, but also for improving patient care across communities. Partnering with hospitals, government health bodies, and research organizations can make this technology widely accessible and beneficial for early diagnosis and intervention.

Our next idea focuses on **“Offering extended features such as predictive tools for treatment planning, monitoring disease progression, and supporting public health decision-making based on liver health data insights.”** Once we achieve our core objective—accurately predicting liver cirrhosis—we aim to scale our model to support broader functionalities. This will enhance the overall value of the project by contributing to both clinical precision and public health advancement.

3. REQUIREMENT ANALYSIS

3.1 Functional requirement

Functional requirements specify the fundamental actions that a system must perform. For the "Traffic Intelligence - Advanced Traffic Volume Estimation using Machine Learning" project, functional requirements might include:

1. **Data Collection:**
   - The system should collect real-time traffic data from various sources, including cameras, sensors, and historical records.
   - It should ensure the continuous and reliable acquisition of data for training and updating the machine learning model.
2. **Feature Engineering:**
   - The system must identify and incorporate relevant features for traffic volume estimation, such as time of day, weather conditions, and special events.

    - It should have the capability to adapt and update features as traffic patterns evolve.
3. **Machine Learning Model:**
    - Develop and implement a machine learning model (e.g., neural network, regression models) for accurate traffic volume prediction.
    - The model should be capable of continuous learning and adaptation to dynamic traffic conditions.
4. **Real-Time Integration:**
    - Implement mechanisms for real-time data integration to ensure the model is continually updated with the latest traffic information.
    - The system should be capable of handling and processing large volumes of real- time data efficiently.
5. **User Interface:**
    - Develop a user-friendly interface for stakeholders to visualize traffic data, predictions, and insights.
    - The interface should provide interactive features for exploring different parameters and scenarios.
6. **Prediction Accuracy:**
    - Define performance metrics for the machine learning model, specifying the required level of accuracy for traffic volume predictions.
    - Regularly assess and improve the model's accuracy through ongoing monitoring and updates.

7. **Alerts and Notifications:**
    - Implement a system for generating alerts and notifications in real-time for abnormal traffic conditions or incidents.
8. **Documentation:**
    - Provide comprehensive documentation for the system, including data sources, model architecture, and interface functionalities.
    - Include user manuals and technical documentation for future maintenance and updates.

3.2 Solution Requirement

1. Data Sources & Acquisition
    - Integrate data collection from:
        - Traffic cameras
        - IoT-based traffic sensors
        - Historical traffic datasets (CSV, public datasets, etc.)
    - Support scheduled and real-time data ingestion.

2. Technology Stack
    - Programming Language: Python (for ML modeling and backend logic)
    - Framework: Flask (for backend web framework and API)

- Frontend: HTML, CSS, JavaScript (for UI)
- ML Libraries: scikit-learn, XGBoost, pandas, NumPy
- Visualization Tools: Matplotlib, Seaborn, Plotly (for UI insights)

3. Machine Learning Pipeline
   - Use supervised learning models (e.g., Random Forest, XGBoost, Linear Regression).
   - Implement hyperparameter tuning for model optimization.
   - Enable model retraining with updated datasets for continuous improvement.
4. Data Storage & Management
   - Store and manage datasets using local files or cloud storage (e.g., Google Drive).
   - Maintain a structured dataset directory for raw, processed, and result files.
   - Store trained models using joblib or pickle.

5. Real-Time System Integration
   - Build APIs to receive new traffic input and return predicted volumes.
   - Ensure seamless data flow from sensors → model → UI.
   - Auto-refresh predictions at regular intervals (e.g., every 5 minutes).

6. User Interface Requirements
   - Develop a responsive web-based dashboard for stakeholders.
   - Provide input fields for time, date, and location.
   - Display predictions, charts, and traffic patterns in an intuitive layout.

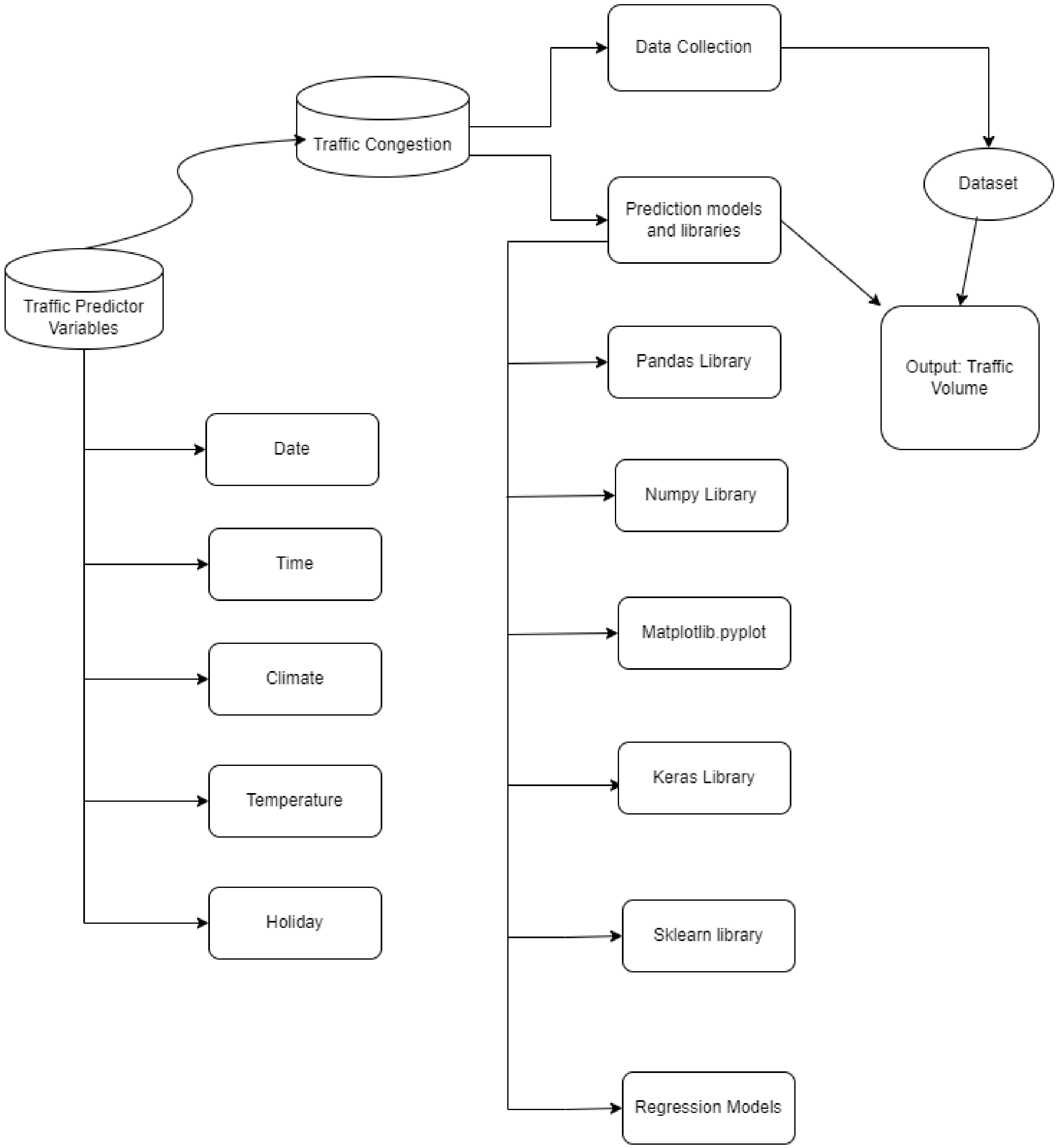
7. Deployment and Infrastructure
   - Deploy the application on a local server (initially) with optional scalability to cloud platforms (AWS/GCP) if needed.
   - Ensure system runs in a high-availability environment.

8. Integration & Compatibility
   - Ensure compatibility with standard browsers (Chrome, Firefox).
   - Design system to allow future integration with:
     - Government traffic systems
     - Navigation apps (e.g., Google Maps)

9. Security & Privacy Measures
   - Protect data using secure access control to APIs and interfaces.
   - Anonymize any sensitive or personally identifiable data from traffic sources.
   - Enable secure storage and encrypted communication channels.

10. Documentation and Versioning
    - Maintain:
      - Technical documentation for developers (code structure, APIs)
      - User documentation for system use
    - Use version control tools like Git and host code on GitHub.

3.3 Data Flow Diagram



User Stories

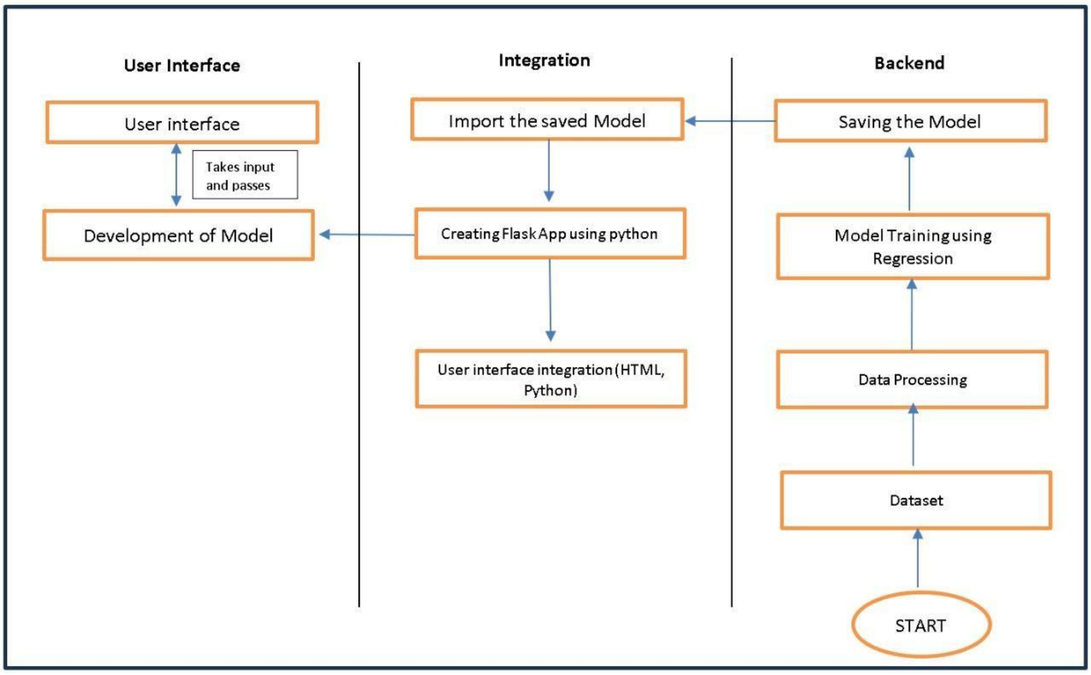
User Type	Function al Requirement (Epic)	User Story Numb er	User Story / Task	Acceptance criteria	Priori ty	Release

Traffic Manager	Real-time Traffic Estimation	USN-1	As a Traffic Manager, I want to access real-time traffic volume estimations to make informed decisions for traffic control.	System provides accurate real-time traffic volume predictions. Data updates occur at least every 5 minutes. Data accuracy is within a 95% confidence interval.	High	Sprint 1
Driver	Real-time Traffic Estimation	USN-2	Application suggests a approximate congestion in the route.	Application suggests an approximate congestion in the route.	High	Sprint 1
Traffic Analyst	Data Insights on congestion volume	USN-3	As a Traffic Analyst, I want a Volume number displaying in-depth traffic insights for informed analysis and decision-making.	Volume number showcases traffic trends over various timeframes.	Mediu m	Sprint 2
Website Develop er	Model building	USN-4	As an Web Developer, I want access to models that integrate TrafficTelligence data for incorporation into existing navigation applications.	Models provide accurate traffic data. Well-documented Models for easy integration. Allows access to real-time and predictive traffic estimations.	High	Sprint 2
City Planner	Customizable Traffic Solutions	USN-5	As a City Planner, I want customizable traffic solutions to accommodate specific city development needs.	System allows adjustments to traffic control strategies. Customization based on specific traffic conditions.	High	Sprint 3
Educational Institutions	Training	USN-6	implement data augmentation techniques (e.g., rotation, flipping) to improve the model's robustness and accuracy.	we could do testing	medium	Sprint 4
	Testing & quality assurance	USN-7	conduct thorough testing of the model and web interface to identify and report any issues or bugs. fine-tune the model hyperparameters and optimize its performance based on user feedback and testing results.	we could create web application	medium	Sprint 5

3.4 Technology Stack

Technical Architecture:

The Deliverable shall include the architectural diagram as below and the information as per the table 1 & table 2



:

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	Critical element designed for both Traffic Managers and everyday users, ensuring an intuitive and informative experience.	HTML, CSS, JavaScript
2.	Application Logic-1	Involves a robust backend system responsible for processing, analyzing, and managing traffic data.	Python
3.	Database	Involves the storage and management of diverse traffic data for analysis.	File Manager, csv
4.	File Storage/ Data	Involves managing diverse types of data, including raw traffic data, machine learning models, and configuration files.	Local System, Google Drive
5.	Frame Work	It is a crucial part of our program as it is responsible for connecting the frontend with the backend.	Python Flask
6.	Machine Learning Model	The machine learning model is responsible for predicting future outcomes based on available data	Machine learning model created using regression algorithms
7.	Infrastructure (Server / Cloud)	Involves a combination of servers and cloud services to support the computational and storage needs of the application.	Local

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Open-source frameworks can accelerate development and ensure the reliability of TrafficTelligence, contributing to a more efficient and maintainable solution.	Python's Flask
2.	Scalability	Using cameras to collect data and to make models for specific locations.	Computer vision, dynamic databases.
3.	Performance	Regular performance testing, monitoring, and optimization are integral components of the development and maintenance processes, ensuring that TrafficTelligence consistently delivers timely and efficient traffic volume estimations.	R squared, Root mean squared error, Root Mean Square deviation
4.	Availability	Website can be made available all time in a webserver. This makes the website running without any issues	High speed Linux based webservers.

4. PROJECT DESIGN

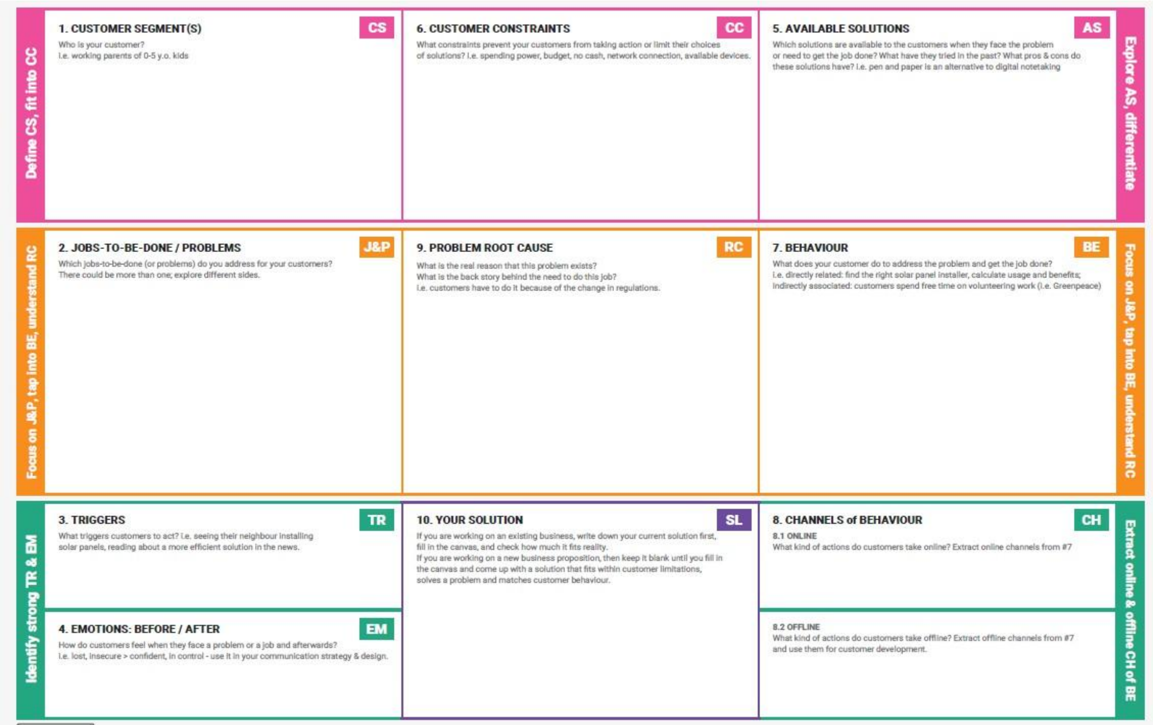
4.1 Problem Solution Fit

Problem – Solution Fit Template:
The Problem-Solution Fit simply means that you have found a problem with your customer and that the solution you have realized for it actually solves the customer's problem. It helps entrepreneurs, marketers and corporate innovators identify behavioral patterns and recognize what would work and why

Purpose:

- Solve complex problems in a way that fits the state of your customers.
- Succeed faster and increase your solution adoption by tapping into existing mediums and channels of behavior.
- Sharpen your communication and marketing strategy with the right triggers and messaging.
- Increase touch-points with your company by finding the right problem-behavior fit and building trust by solving frequent annoyances, or urgent or costly problems.
- **Understand the existing situation in order to improve it for your target group.**

Template:



References:

1. https://www.ideahackers.network/problem-solution-fit-canvas/
2. https://medium.com/@epicantus/problem-solution-fit-canvas-aa3dd59cb4fe

4.2 Proposed Solution

Proposed Solution Template:

Project team shall fill the following information in proposed solution template.

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Traffic problem is one of the major problem now a days, In the increase in no of vehicles and non –usage of public transport leading to traffic related issues, Making a eye on count of traffic at each level enables the government to take the further decisions such as building new roads, increasing infrastructure ,developing mutli-channel connectivity .To address such problems to tracking the vehicle count in each and every place AI-ML has given a solution to such kind of traffic related issues, which are able to measure the volume of traffic, identify the violations of traffic rules etc.ML models could give early alerts of severe traffic to help prevent issues related to traffic problems. Hence, there is needs to develop ML algorithms capable in predicting Traffic volume with acceptable level of precision and in reducing the error in the dataset of the projected Traffic volume from model with the expected observable Traffic volume.
2.	Idea / Solution description	Traffic Intelligence: Advanced Volume Estimation Using Machine Learning" aims to enhance traffic volume estimation for urban planning and management. By collecting diverse traffic data and applying machine learning, the project seeks to provide real-time, accurate traffic volume predictions, historical analysis, and anomaly detection, ultimately contributing to more efficient and informed traffic management.

3.	Novelty / Uniqueness	The uniqueness of this project lies in applying advanced machine learning for real-time traffic volume predictions, integrating diverse data sources, and offering anomaly detection, all with a user-friendly interface. This approach stands out in its potential to transform traffic management and urban planning.
4.	Social Impact / Customer Satisfaction	TrafficTelligence: Advanced Traffic Volume Estimation With Machine Learning enhances traffic management by accurately predicting real-time traffic volume. This innovation not only aids authorities in proactive decision-making but also empowers drivers with alternate routes, reducing congestion and travel time. Its commitment to continual improvement ensures heightened user satisfaction, making it a transformative solution for smoother traffic flow and increased efficiency in urban mobility.
5.	Business Model (Revenue Model)	The business revolves around licensing this technology. There can be strategic collaborations with authorities/government in order to help regulate traffic better in return for more data to make the model better
6.	Scalability of the Solution	Its flexible architecture seamlessly integrates with existing infrastructures, ensuring quick deployment without disruption. With the ability to handle varying data loads and continual improvement, TrafficTelligence remains at the forefront of efficiency, adapting to changing traffic patterns and specific regional needs. This scalability ensures its relevance and effectiveness in diverse traffic management scenarios, catering to various urban, suburban, and rural settings.

4.3 Solution Architecture

Solution Architecture:

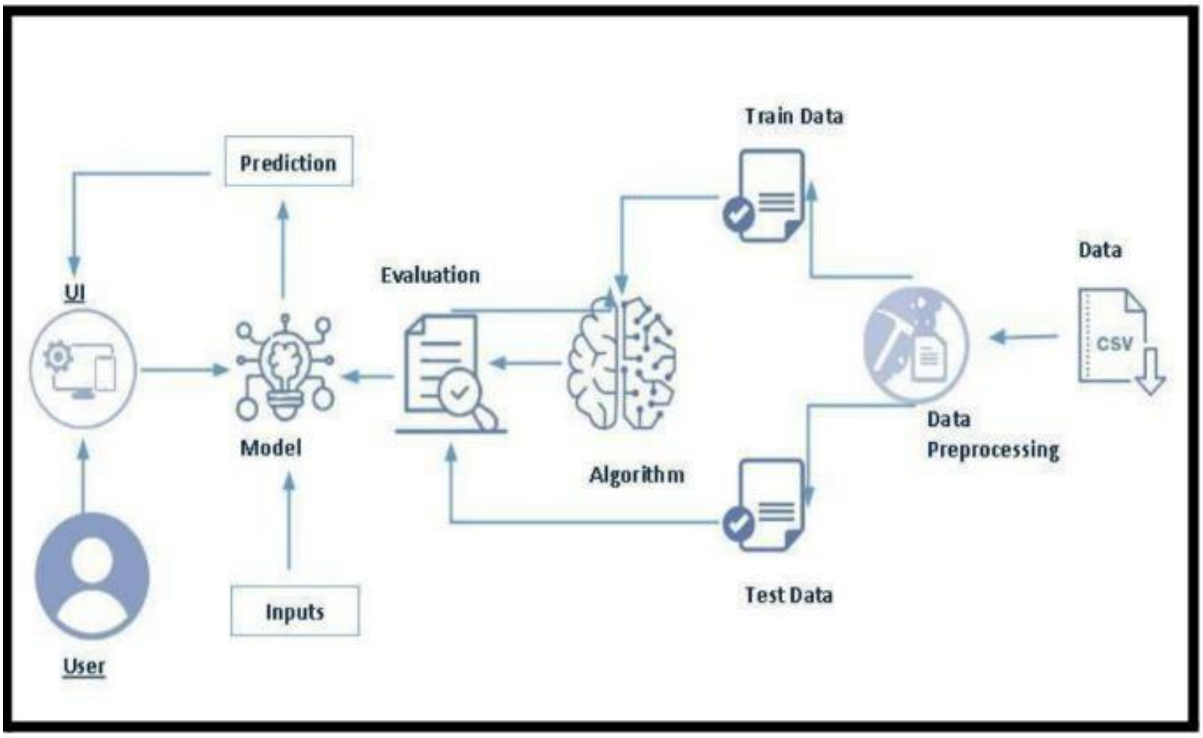
Traffic Intelligence: Advanced Volume Estimation Using Machine Learning" aims to enhance traffic volume estimation for urban planning and management. By collecting diverse traffic data and applying machine learning, the project seeks to provide real-time, accurate traffic volume predictions, historical analysis, and anomaly detection, ultimately contributing to more efficient and informed traffic management.

Our solution uses many advanced Machine learning Algorithms to address the Traffic Volume Estimation problem effectively.

Steps to be followed: -

1. Data Collection: Sensors, cameras, and IoT devices capture real-time traffic data.
2. Data Pre-processing: Clean and preprocess data to make an effective model.
3. Train Model: Using preprocessed data to make predictive models for forecasting traffic volume patterns for real-time estimations.
4. Test Model: To make sure that the model is accurate and efficient.
5. Integrating Model: To make a user-facing application so that the user can interact with the model.

Solution Architecture Diagram:



6.

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

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 required tools and frameworks to start the project	1	High	Naga Sashank
Sprint-2	Data collection	USN-2	Gather a diverse dataset of Date, time, holidays and climatic conditions.	2	High	Naga Sashank
Sprint-2	data preprocessing	USN-3	Preprocess the collected dataset by removing outliers and null values etc. Explore and evaluate different deep learning architectures (e.g., Regressions) to select the most suitable model for the project.	3	High	Kiran Achari
Sprint-3	model development	USN-4	train the selected machine learning model using the preprocessed dataset and monitor its performance on the validation set.	4	High	Kiran Achari
Sprint-3	Training	USN-5	The data set will be trained with suitable algorithms to improve robustness and accuracy.	6	medium	Sunil Kumar
Sprint-4	model deployment & Integration	USN-6	deploy the trained machine learning model as a web service to make it accessible for users. Integrate the model's API into a user-friendly web interface for users to input variables such as date, time, holidays etc and receive predicted volume results.	1	medium	Sunil Kumar

Sprint-5	Testing & quality assurace	USN-7	conduct thorough testing of the model and web interface to identify and report any issues or bugs. fine-tune the model hyperparameters and optimize its performance based on user feedback and testing results.	1	medium	Gopi

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

Sprint	**Total Story Points**	**Duration**	**Sprint Start Date**	**Sprint End Date (Planned)**	**Story Points Completed (as on Planned End Date)**	**Sprint Release Date (Actual)**
Sprint-1	1	3 Days	3 Nov 2023	6 Nov 2023	1	6 Nov 2023
Sprint-2	5	2 Days	6 Nov 2023	8 Nov 2023	5	8 Nov 2023
Sprint-3	10	5 Days	8 Nov 2023	13 Nov 2023	10	13 Nov 2023
Sprint-4	1	5 Days	13 Nov 2023	18 Nov 2023	1	20 Nov 2023
Sprint-5	1	4 Days	18 Nov 2023	22 Nov 2023	1	21 Nov 2023

Velocity:

Imagine we have a 29-days sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

AV= 19/3.8 = 5

Burndown Chart:

A burndown chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.

https://www.visual-paradigm.com/scrum/scrum-burndown-chart/
https://www.atlassian.com/agile/tutorials/burndown-charts

Reference:

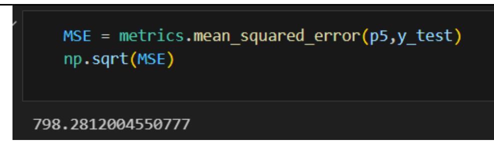
https://www.atlassian.com/agile/project-management
https://www.atlassian.com/agile/tutorials/how-to-do-scrum-with-jira-software
https://www.atlassian.com/agile/tutorials/epics **https://www.atlassian.com/agile/tutorials/sprints**
https://www.atlassian.com/agile/project-management/estimation
https://www.atlassian.com/agile/tutorials/burndown-charts

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Performance Testing

Model Performance Testing:

Project team shall fill the following information in model performance testing template.

S.No.	Parameter	Values	Screenshot
1.	Metrics	**Regression Model:** RMSE - 798.2812004550777	

7. RESULTS

7.1 Output Screenshots

```python
#Model Building
from sklearn import linear_model
from sklearn import tree
from sklearn import ensemble
from sklearn import svm
import xgboost
```

Python

```python
lin_reg = linear_model.LinearRegression()
Dtree = tree.DecisionTreeRegressor()
Rand = ensemble.RandomForestRegressor()
svr = svm.SVR()
XGB = xgboost.XGBRegressor()
```

Python

```python
#Testing the model
#1.using R-squared_score
from sklearn.metrics import r2_score
p1 = lin_reg.predict(x_test)
print(r2_score(p1,y_test))
```

Python

```
-5.365817964773322
```

```python
p2 = Dtree.predict(x_test)
print(r2_score(p2,y_test))
```

Python

```
0.6886039409255853
```

```python
p3 = Rand.predict(x_test)
print(r2_score(p3,y_test))
```

Python

```
0.8008634651129952
```

```python
p4 = svr.predict(x_test)
print(r2_score(p4,y_test))
```

Python

```
-11.990577978126487
```

```python
p5 = XGB.predict(x_test)
print(r2_score(p5,y_test))
```

Python

```
0.8047597408294678
```

```python
#2.Using Root mean squared error(RMSE)
from sklearn import metrics
```
Python

```python
MSE = metrics.mean_squared_error(p1,y_test)
np.sqrt(MSE)
```
Python

```
1838.0090792531755
```

```python
MSE = metrics.mean_squared_error(p2,y_test)
np.sqrt(MSE)
```
Python

```
1105.4484997704067
```

```python
MSE = metrics.mean_squared_error(p3,y_test)
np.sqrt(MSE) #Less compared to others
```
Python

```
803.1617839722985
```

```python
MSE = metrics.mean_squared_error(p4,y_test)
np.sqrt(MSE)
```
Python

```
1715.5541279662643
```

```python
MSE = metrics.mean_squared_error(p5,y_test)
np.sqrt(MSE)
```
Python

```
798.2812004550777
```

```python
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_state=0)
```
Python

```python
y = data['traffic_volume']
x = data.drop(columns=['traffic_volume','holiday','weather'],axis=1)
```
Python

```python
names = x.columns
```
Python

```python
from sklearn.preprocessing import scale
```
Python

```python
x = scale(x)
```
Python

```python
x = pd.DataFrame(x,columns=names)
```
Python

```python
x.head()
```
Python

	temp	rain	snow	day	month	year	hours	minutes	seconds	holiday_v2	weather_v2
0	0.530485	-0.007463	-0.027235	-1.574903	1.02758	-1.855294	-0.345548	0.0	0.0	0.031687	-0.566452
1	0.611467	-0.007463	-0.027235	-1.574903	1.02758	-1.855294	-0.201459	0.0	0.0	0.031687	-0.566452
2	0.627964	-0.007463	-0.027235	-1.574903	1.02758	-1.855294	-0.057371	0.0	0.0	0.031687	-0.566452
3	0.669205	-0.007463	-0.027235	-1.574903	1.02758	-1.855294	0.086718	0.0	0.0	0.031687	-0.566452
4	0.744939	-0.007463	-0.027235	-1.574903	1.02758	-1.855294	0.230807	0.0	0.0	0.031687	-0.566452

```python
#Model Deployment
#saving the model
import pickle
from sklearn.preprocessing import LabelEncoder
le = le = LabelEncoder()
pickle.dump(Rand, open("model.pkl",'wb'))
pickle.dump(le, open("encoder.pkl", "wb"))
```

Python

```python
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)
```

Python

XGBRegressor

▶ Parameters

```python
#importing necessary libraries

import pandas as pd
import numpy as np
import seaborn as sns
import sklearn as sk
import matplotlib.pyplot as plt
from sklearn import linear_model
from sklearn import tree
from sklearn import ensemble
from sklearn import svm
from collections import Counter
import xgboost
```

Python

```python
#importing the data

data = pd.read_csv('traffic volume.csv')
```

Python

```python
#displaying first 5 rows of the data
data.head()
```

Python

```python
#used to display the basic information of the data

data.info()
```

Python

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 48204 entries, 0 to 48203
Data columns (total 8 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   holiday         61 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
dtypes: float64(3), int64(1), object(4)
memory usage: 2.9+ MB
```

```python
# used to display the null values of the data

data.isnull().sum()
```

```
holiday          48143
temp                53
rain                 2
snow                12
weather             49
date                 0
Time                 0
traffic_volume       0
dtype: int64
```

```python
data['temp'].fillna(data['temp'].mean(),inplace=True)
data['rain'].fillna(data['rain'].mean(),inplace=True)
data['snow'].fillna(data['snow'].mean(),inplace=True)

print(Counter(data['weather']))
```

```
Counter({'Clouds': 15144, 'Clear': 13383, 'Mist': 5942, 'Rain': 5665, 'Snow': 2875, 'Drizzle': 1818, 'Haze': 1359, 'Thunderstorm': 1033, 'Fog': 912, nan: 49, 'Smol
```

```python
data['weather'].fillna('Clouds',inplace=True)
```

```python
#splitting the date column into year,month,day
data[["day", "month", "year"]] = data["date"].str.split("-", expand = True)
```

```python
#splitting the Time column into hour,minute,second
data[["hours", "minutes", "seconds"]] = data["Time"].str.split(":", expand = True)
```

```python
data.drop(columns=['date','Time'],axis=1,inplace=True)
```

```python
data.head()
```

	holiday	temp	rain	snow	weather	traffic_volume	day	month	year	hours	minutes	seconds
0	NaN	288.28	0.0	0.0	Clouds	5545	02	10	2012	09	00	00
1	NaN	289.36	0.0	0.0	Clouds	4516	02	10	2012	10	00	00
2	NaN	289.58	0.0	0.0	Clouds	4767	02	10	2012	11	00	00
3	NaN	290.13	0.0	0.0	Clouds	5026	02	10	2012	12	00	00
4	NaN	291.14	0.0	0.0	Clouds	4918	02	10	2012	13	00	00

```python
#used to understand the descriptive analysis of the data
data.describe()
```

Python

	temp	rain	snow	traffic_volume
count	48204.000000	48204.000000	48204.000000	48204.000000
mean	281.205351	0.334278	0.000222	3259.818355
std	13.336338	44.789133	0.008168	1986.860670
min	0.000000	0.000000	0.000000	0.000000
25%	272.180000	0.000000	0.000000	1193.000000
50%	282.429000	0.000000	0.000000	3380.000000
75%	291.800000	0.000000	0.000000	4933.000000
max	310.070000	9831.300000	0.510000	7280.000000

```python
# Import label encoder
from sklearn import preprocessing

# label_encoder object knows
# how to understand word labels.
label_encoder = preprocessing.LabelEncoder()
```

Python

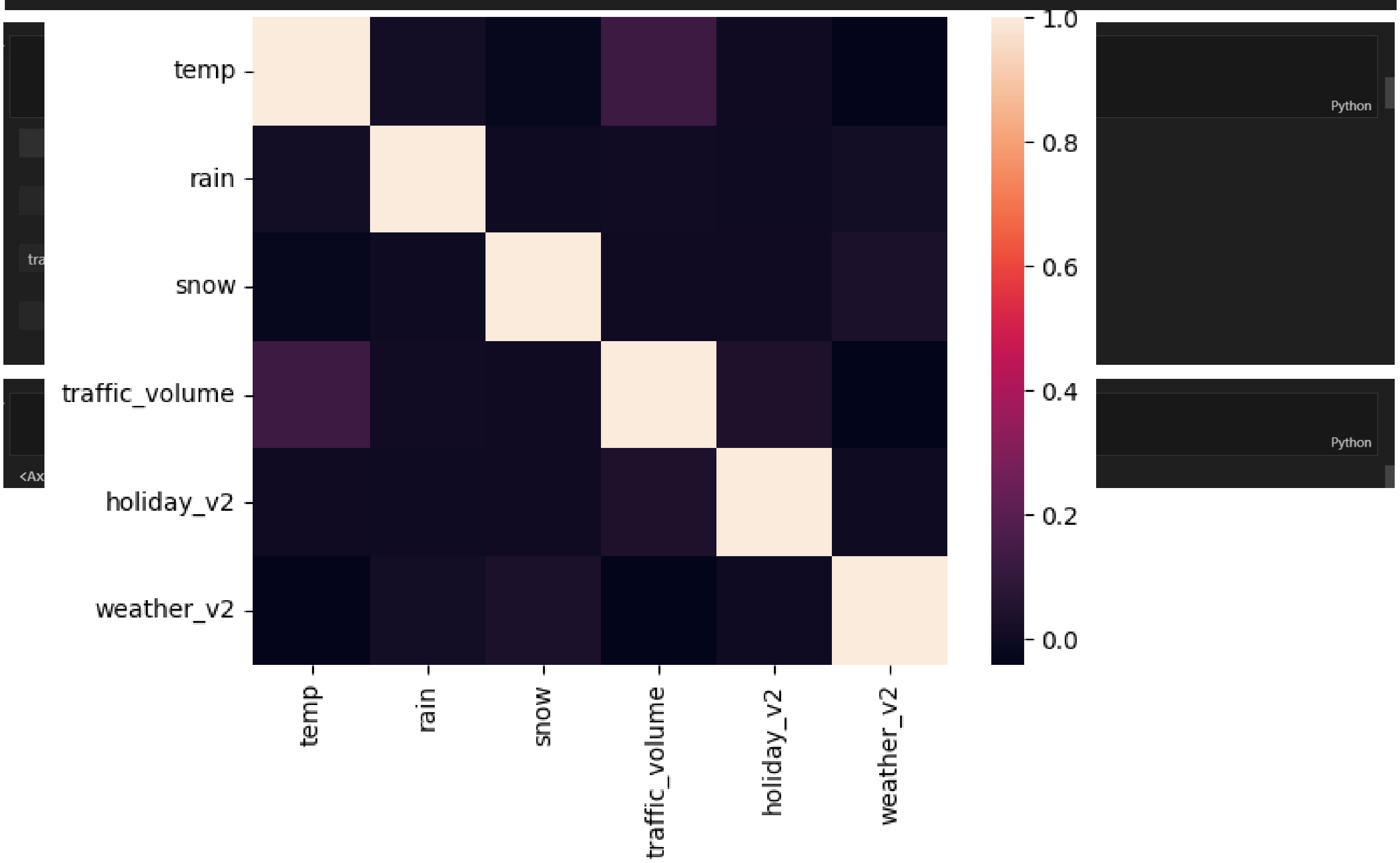
```python
data['holiday_v2'] = label_encoder.fit_transform(data['holiday'])
```

Python

```python
data['weather_v2']= label_encoder.fit_transform(data['weather'])
print(label_encoder.classes_)
data['weather_v2'].unique()
```

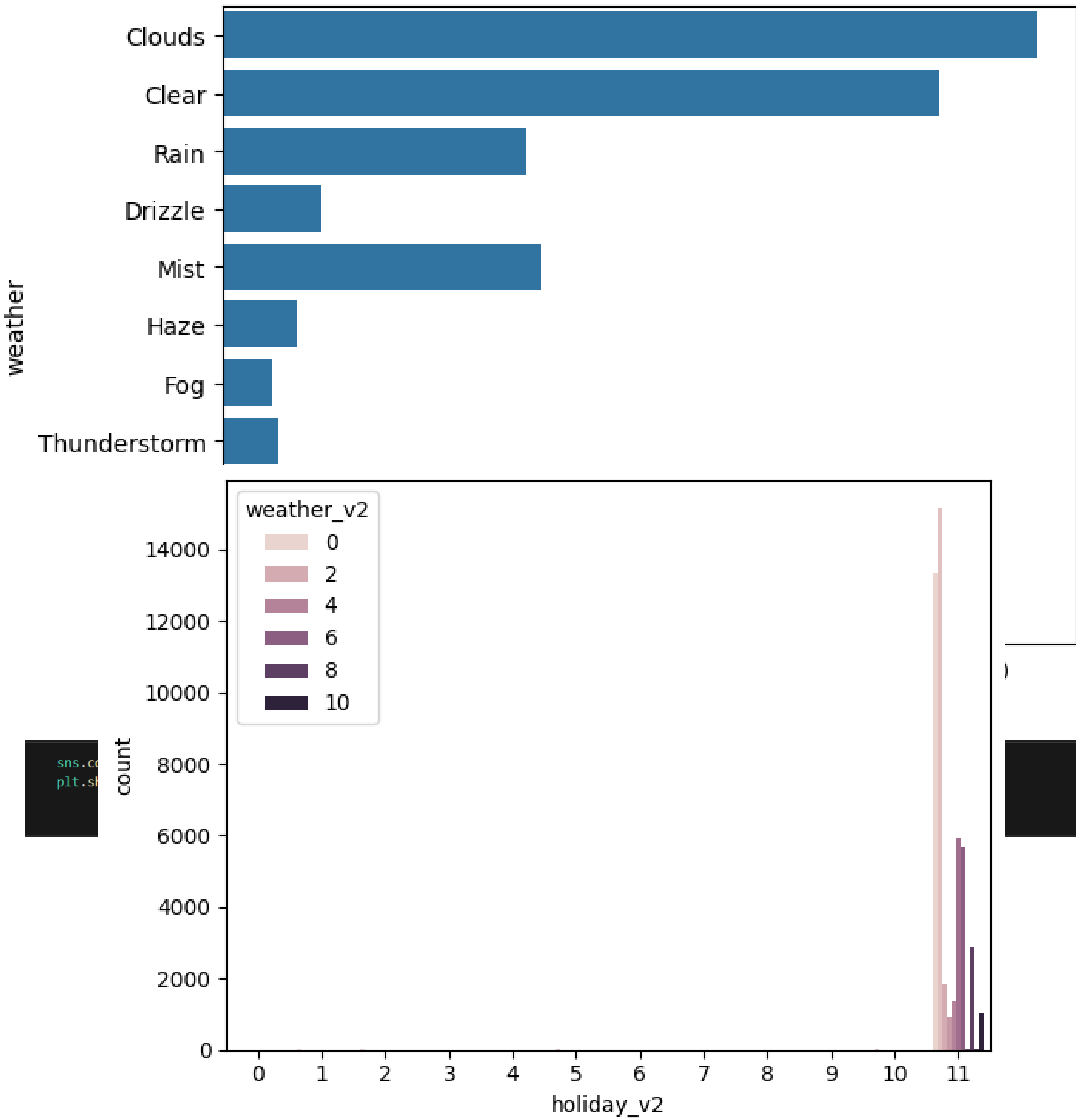
Python

```
['Clear' 'Clouds' 'Drizzle' 'Fog' 'Haze' 'Mist' 'Rain' 'Smoke' 'Snow'
 'Squall' 'Thunderstorm']

array([ 1,  0,  6,  2,  5,  4,  3, 10,  8,  9,  7])
```



```python
sns.countplot(data['weather'])
```

```
<Axes: xlabel='count', ylabel='weather'>
```




```python
sns.c
plt.s
```

```python
sns.pairplot(data)
```

Python

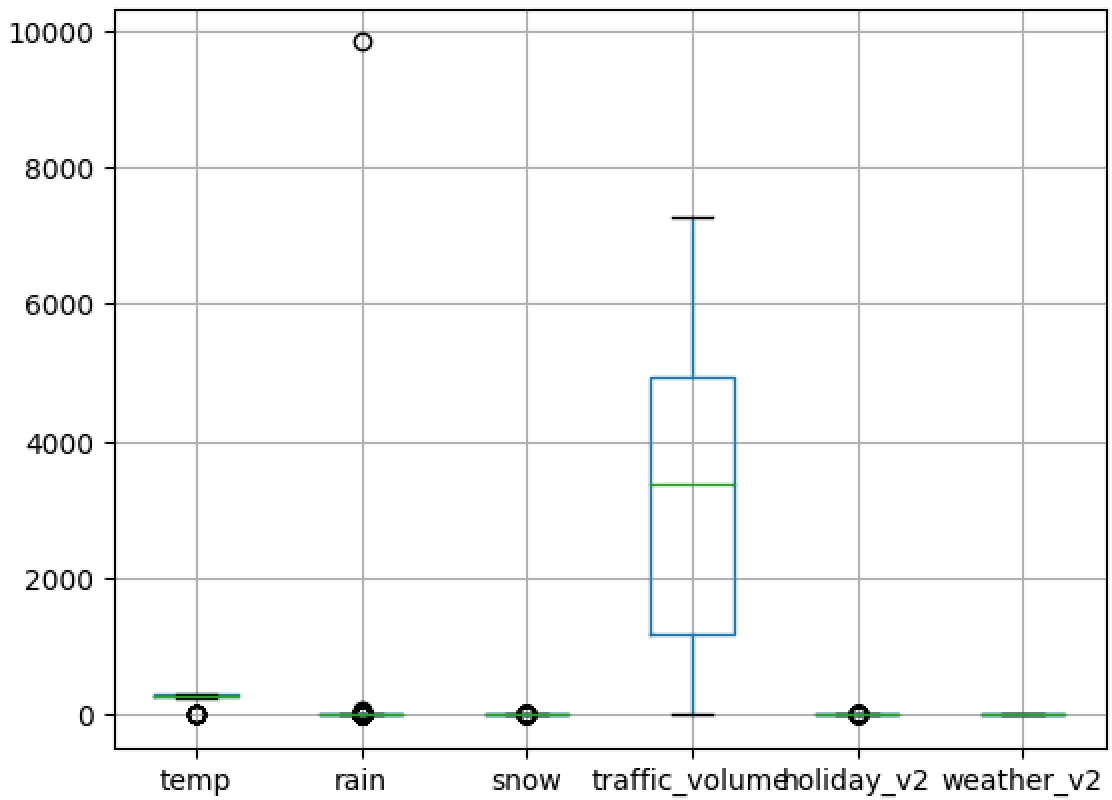
```
c:\Users\sunil\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\axisgrid.py:123: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)

<seaborn.axisgrid.PairGrid at 0x2540478d010>
```

```
data.boxplot()
```
Python

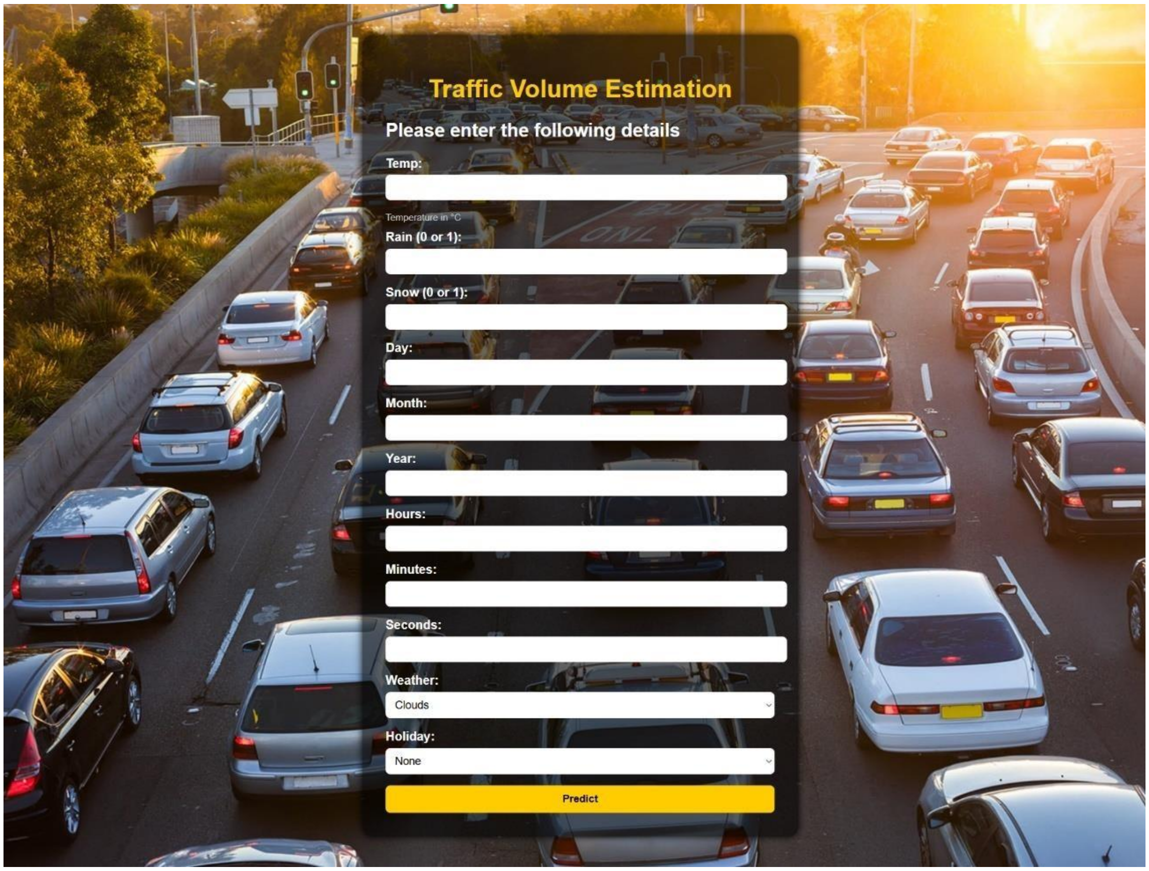
<Axes: >


```
data.boxplot()
```
Python

<Axes: >



Our Website will be looking like this:



After filling all the fields, the final website looks like this:

8. ADVANTAGES & DISADVANTAGES

Advantages:

1. Improved Accuracy**:**

- Machine learning models can analyze large datasets and identify complex patterns that may be challenging for traditional methods. This leads to more accurate traffic volume predictions.

2. **Integration with Sensor Data:**
    - Machine learning models can effectively integrate data from various sources, such as traffic cameras, sensors, and GPS devices, providing a comprehensive view of the traffic situation.
3. **Scalability:**
    - Machine learning algorithms can scale to handle large and complex datasets, making them suitable for cities with extensive traffic networks.
4. **Predictive Capabilities:**
    - Machine learning models can be used to predict future traffic conditions based on historical data, helping authorities proactively manage traffic flow and prevent congestion.

Disadvantages:

1. **Data Dependency:**
    - Machine learning models heavily rely on high-quality and representative data. If the training data is biased or incomplete, the model's predictions may be inaccurate or skewed.
2. **Complexity:**
    - Building and maintaining machine learning models can be complex and require specialized knowledge. This complexity can hinder the adoption of these systems, especially for smaller municipalities with limited resources.
3. **Dynamic Nature of Traffic:**
    - Traffic patterns are influenced by a wide range of factors, and they can change rapidly. Machine learning models may struggle to keep up with these dynamic changes, especially if not continuously updated and retrained.

9. CONCLUSION

In conclusion, the application of machine learning for advanced traffic volume estimation in the realm of traffic intelligence brings forth a set of notable advantages and challenges. The accuracy and adaptability offered by machine learning models present a promising avenue for enhancing traffic management. Real-time analysis capabilities, integration with diverse data sources, scalability, and predictive capabilities contribute to more efficient and proactive traffic control.

However, the successful implementation of machine learning in this context requires addressing several challenges. The dependency on high-quality and unbiased data, the inherent complexity of building and maintaining these models, and the interpretability issues associated with certain algorithms pose significant hurdles. Additionally, the dynamic nature of traffic patterns and the computational resources required for training and running sophisticated models underscore the need for careful consideration and resource allocation.

10. FUTURE SCOPE

In the future, the application of advanced traffic volume estimation using machine learning holds tremendous promise in reshaping urban mobility and transportation systems. Ongoing research efforts are likely to focus on enhancing prediction accuracy through the exploration of sophisticated algorithms, feature engineering techniques, and ensemble methods. A significant avenue for development lies in the integration of traffic intelligence with broader smart city initiatives, facilitating interconnected urban transportation systems that optimize traffic flow and minimize environmental impact. The adoption of edge computing is poised to enable real-time analysis at the source, reducing latency and enhancing responsiveness. Overcoming the interpretability challenge by incorporating explainable AI techniques will be crucial for building trust among city planners and the public. Future systems may extend beyond road traffic to encompass multi-modal transportation, incorporating pedestrians, cyclists, and public transit. The dynamic adaptation of machine learning models to unforeseen events and continuous improvement mechanisms through online learning and feedback loops are vital considerations. Collaborative efforts between municipalities, transportation agencies, and technology providers can lead to more comprehensive and effective traffic management solutions, fostering a connected and efficient transportation network. Ultimately, the future of machine learning in traffic intelligence lies in its ability to create sustainable, adaptive, and energy-efficient urban mobility solutions.

11. APPENDIX

Our Complete Source Code

1. Model Python

2. [Flask app integration]
3. [Web UI] [(HTML Code)]
4. [Data Set]
5. [Project Demo]