ARTIFICIAL INTELLIGENCE & MACHINE LEARNING

Project Documentation format

# Introduction

* + **Project Title: “**Traffic Intelligence : Advanced Traffic Volume Estimation with Machine Learning”
  + **Team Members:**

TEAM ID : LTVIP2025TMID44293

TEAM MEMBERS : C Eswar

A Naga Harini

# Project Overview

* + **Purpose:** The purpose of the Traffic Intelligence : Advanced Traffic Volume Estimation using ML project is to leverage ML techniques to analyze and predict traffic volume patterns. It aims to help city planners, transportation departments and commuters make informed decisions by reducing congestion , improving road safety and optimizing traffic flow through smart predictions based on historical and real-time data.
  + **Features:** i. Real-time Traffic Volume Estimation

ii. ML-based Forecasting

iii. Historical Data Analysis

iv. Scalability & Integration

# Architecture

* + **Frontend:** To provide an intuitive UI for users to interact with the system

Tech Stack : HTML , CSS

* + **Backend:** To process inputs , run ML models and serve predictions.

Tech Stack : Flask , Scikit-learn , TensorFlow

* + **Database:** To store Traffic data , historical records , user inputs and predictions.

Tech Stack : SQLite , MYSQL

# Setup Instructions

**Prerequisites:**

**Python packages:**

Open anaconda prompt as administrator.

* Type “pip install numpy” and click enter.
* Type “pip install pandas” and click enter.
* Type “pip install matplotlib” and click enter.
* Type “pip install scikit-learn” and click enter.
* Type “pip install Flask” and click enter.
* Type “pip install xgboost” and click enter.

**Installation:**

Step-1 : Clone the Repositoty

git clone <https://github.com/Eswar02104/Traffic_Intelligence.git>

cd Traffic\_Intelligence

Step-2 : Setup the Python Virtual Environment

python -m venv venv

Venv\Scripts\activate

Step-3 : Install Python dependencies

pip install -r requirements.txt

Step-4 : Setup the Environment Variables

touch .env

Step-5 : Run the Flask Backend Server

python app.py

# Folder Structure

# TrafficTelligence/

# │

# ├── Flask/

# │ ├── templates/

# │ │ ├── index.html

# │ │ └── result.html

# │ ├── static/

# │ ├── app.py

# │ ├── model.pkl # Trained ML model

# │ └── encoder.pkl # Label encoder or any preprocessing encoders

# │

# ├── IBM/

# │ └── (Notebook files or IBM-related work)

# │

# ├── requirements.txt

# ├── traffic\_volume.csv

# └── README.md

# Running the Application

# Install the dependencies and run using app.py

# API Documentation

# The Traffic Intelligence backend is built using Flask and exposes a set of RESTful APIs for client interaction. The APIs allow users to upload traffic data, receive predictions and fetch model.

# /predict - Traffic Volume Prediction Endpoint

# /history - Fetch Previous Predictions

# /upload-data - Upload CSV Dataset

# /model-info - View Model Metadata

# Authentication

# Authentication : Verifying User Identity

# API Key-Based Authentication(for APIs)

# Login System(for Web Interface)

# Authorization : Granting Permissions

# Role-based Authorization

# Protected Routes in Backend

# User Interface

# The UI is designed to be interactive, informative and user-friendly , offering clear visualizations and easy to access to traffic prediction features.

# Home/Dashboard Page

# Upload Page

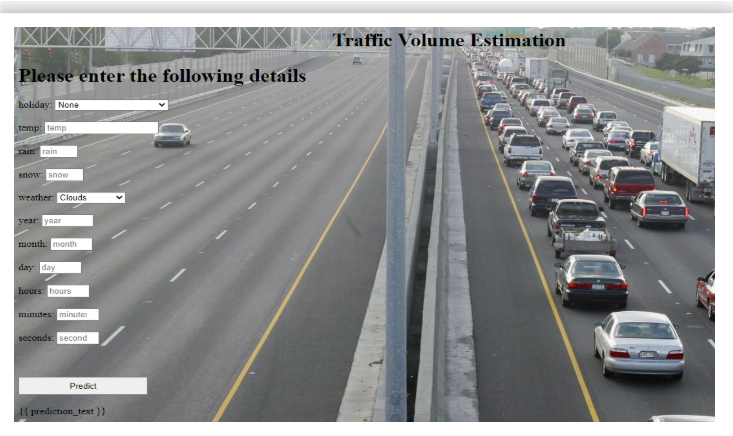
# Results/Analytics Phase

# Testing

The Testing strategy covers multiple layers of the application to ensure it is reliable, accurate and robust across both the frontend, backend and ML components.

1. Unit Testing
2. Integration Testing
3. Model Evaluation Tests
4. Security & Error Handling Tests

# Screenshots





# Known Issues

1. Limited Model Accuracy in Complex Scenarios
2. No Real-time Streaming Support
3. Lack of Geo-location Context
4. Large File Upload Limitations

# Future Enhancements

# Real-time Video Stream Integration

# GPS & Map-based Visualization

# Multi-Class Vehicle Detection

# Automated Alerts & Notifications

# 