Full Stack Development Documentation

Project Title: Traffictelligence:Advanced Traffic Volume Estimation With Machine

ID: LTVIP2025TMID35728

Team Leader: Vuyyuru Sri Harsha

Team Member:Vemuri Sri Meghana

Team Member:Venkatesh M

Team Member:Venkata Aasrit

1. Introduction

Traffictelligence is a smart traffic management system that uses cameras, sensors, and machine learning to understand and control traffic on roads. By using artificial intelligence, it can also predict how traffic will be in the next few minutes or hours

2. Project Overview

Purpose:

This helps reduce traffic jams, saves travel time and improve road safety. Overall, the project is designed to make traffic systems faster, smarter and more efficient for cities and people.

Features:

-Real-Time Traffic Monitoring

-Smart Video Analysis

-Cloud Based & Scalable

-Location Based Alerts

3. Architecture

Frontend: Built using React.js to display real-time waste classification data, graphs, and bin status.

Backend: Node.js with Express.js serves APIs for handling classification results, user auth, and reporting.

Database: MongoDB stores classification results, timestamps, and user data

3. Setup Instructions

Prerequisites:

- Node.js

- MongoDB

- Git

- Python (for ML model server)

Installation:

1. Clone the repository

2. Run `npm install` in client and server folders

3. Set up `.env` files for keys and DB URIs

4. Folder Structure

Client:

/client/src/components – UI Components

/client/src/pages – React Pages

/client/src/api – Axios calls

Server:

/server/routes – Express routes

/server/models – Mongoose schemas

/server/controllers – Business logic

6. Running the Application

Frontend: `npm start` in `/client`

Backend: `npm start` in `/server`

Model Server: Run Python script serving the model (Flask or FastAPI)

7. API Documentation

POST `/api/classify` – Receives image data, returns waste category

GET `/api/stats` – Returns classification summary

POST `/api/login` – Authenticates user

8. Authentication

JWT-based authentication. Login API returns a token stored in localStorage for protected routes.

9. User Interface

Includes dashboard view, live feed viewer, and result logs. Responsive design with basic theming.

10. Testing

Used Jest and Postman for unit and API testing. Accuracy validated using benchmark image datasets.

11. Screenshots or Demo

To be included: dashboard screenshot, live classification, output logs.

12. Known Issues

- Misclassification under poor Camera Quality

- Data Privacy Concerns

-Needed For Skilled Team

13. Future Enhancements

- Voice Assistant Integration

- User Feedback System

- Smart Traffic Light Control

-Drone Surveillance