Traffictelligence – Advanced Traffic Volume Estimation With Machine

**1. Introduction**

Project Title: Traffictelligence – Advanced Traffic Volume Estimation With Machine

Team Members:  
- [kesanasetty Rohith lakshman] – Frontend Developer  
- [kancharlapalli tej deep] – Backend Developer  
- [kancharla ajay] – ML Engineer  
- [kanna kavya] – Documentation & Testing

**2. Project Overview**

Purpose:  
Traffictelligence is an AI-powered web application designed to estimate traffic volume in real time using machine learning and computer vision. It assists city planners, traffic authorities, and researchers in analyzing congestion trends for better infrastructure planning and traffic management.

Features:  
- Video/image upload for traffic analysis  
- YOLO-based vehicle detection  
- Real-time traffic volume estimation  
- Interactive data visualization dashboard  
- User-friendly interface  
- RESTful API endpoints for data access

**3. Architecture**

Frontend:  
Built using React.js with Redux, React Router, Chart.js, and Bootstrap.

Backend:  
Node.js with Express.js; communicates with a Python-based ML model.

Database:  
MongoDB using Mongoose ODM for schema and interaction.

**4. Setup Instructions**

Prerequisites:  
- Node.js >= 16  
- MongoDB  
- Python 3.x  
- pip

Installation:  
```bash  
git clone https://github.com/your-repo/traffictelligence.git  
cd traffictelligence  
npm install  
cd client && npm install  
cd ../ml-model && pip install -r requirements.txt  
```

Environment Variables:  
```  
MONGO\_URI=your\_mongo\_uri  
PORT=5000  
```

**5. Folder Structure**

Client:  
- /src/components: UI components  
- /src/pages: Main pages  
- /src/services: API calls

Server:  
- /routes: API routes  
- /controllers: Logic for each route  
- /models: MongoDB schema models  
- /ml-service: Communicates with Python model

**6. Running the Application**

Frontend:  
```bash  
cd client  
npm start  
```

Backend:  
```bash  
npm start  
```

ML Model Server (Python):  
```bash  
cd ml-model  
python app.py  
```

**7. API Documentation**

| Method | Endpoint | Description |  
|--------|------------------|-----------------------------|  
| POST | /api/upload | Uploads image/video |  
| GET | /api/results | Retrieves processed results |  
| POST | /api/auth/login | User login |

**8. Authentication**

JWT-based Authentication  
- Users log in and receive a token  
- Token is stored in localStorage  
- Protected routes validate the token

**9. User Interface**

Features:  
- Upload interface for traffic media  
- Real-time detection result view  
- Dashboard with statistics (bar/line/pie charts)  
- Login and registration forms

**10. Testing**

Frontend: Jest + React Testing Library

Backend: Postman + Jest

ML Model: Unit testing on detection accuracy using OpenCV

**11. Screenshots or Demo**

Insert screenshots or video demo link here.

Live Demo: https://traffictelligence.vercel.app/

**12. Known Issues**

- Detection accuracy may decrease under poor lighting

- Video size >50MB may slow processing

- ML model takes 2–3 seconds per frame on CPU

**13. Future Enhancements**

- Real-time traffic heatmap on city map

- Live video stream support

- Admin dashboard for analytics over time

- Integration with IoT devices for traffic light control