Project Development

1. Technology Stack Used:

***Programming Language:**

*Python– Core language for building machine learning models and data processing.

***Frameworks & Libraries:**

Scikit-learn – For implementing ML algorithms like regression, classification, etc.

Pandas & NumPy– For data manipulation and numerical computation.

Matplotlib & Seaborn – For data visualization and graphical analysis.

OpenCV – For processing CCTV and video footage

***Development Tools:**

Jupyter Notebook – For interactive coding, testing, and visualization.

VS Code – For writing modular Python scripts and maintaining the project structure.

Anaconda – For managing packages and creating isolated development environments.

***APIs & Integration :**

Flask – To develop REST APIs for serving the ML model and receiving traffic data.

CSV / JSON – As data input/output formats.

---

2. Development Process:

***Data Collection:**

Collected traffic datasets from public sources or simulated data.

Formats: Video footage, GPS logs, or sensor data.

*** Data Preprocessing:**

Removed null values, duplicates, and outliers.

Normalized and structured data for ML input.

If using OpenCV: Extracted frames, identified vehicles using object detection.

*** Exploratory Data Analysis (EDA):**

Visualized traffic trends, peak hours, and traffic volume distribution using Matplotlib and Seaborn.

*** Model Building:**

Selected suitable ML model (e.g., Random Forest, Linear Regression).

Trained and validated model using Scikit-learn.

Evaluated with metrics like MAE, RMSE, or R^2 score.

*** Model Deployment :**

Exposed trained model via Flask/FastAPI for real-time predictions.

*** UI or Dashboard :**

* Built simple HTML page to view predictions and graphs.

---

3. Challenges and Fixes

Challenge 1: Data Quality and Availability

Problem:

Lack of real-time traffic data and inconsistency in formats.

Fix:

Used open datasets (like from Kaggle or government portals) and simulated dummy data for initial testing.

Challenge 2: Model Accuracy Fluctuations

Problem:

The model underperformed on certain traffic scenarios (e.g., weekends or off-peak hours).

Fix:

Enhanced feature engineering by adding time-based features and weather/holiday indicators.

Challenge 3: Large Dataset Handling

Problem:

Jupyter slowed down with large datasets.

Fix:

Switched to VS Code with optimized scripts and used batch processing.