**Project Design Phase-II**

**Technology Stack (Architecture & Stack)**

|  |  |
| --- | --- |
| Date | 22 June 2025 |
| Team ID | LTVIP2025TMID41800 |
| Project Name | TrafficTelligence: Advanced Traffic Volume Estimation with Machine Learning |
| Maximum Marks | 4 Marks |

**Technical Architecture:**

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2

**Example: Order processing during pandemics for offline mode**

**Reference:** [**https://developer.ibm.com/patterns/ai-powered-backend-system-for-order-processing-during-pandemics/**](https://developer.ibm.com/patterns/ai-powered-backend-system-for-order-processing-during-pandemics/)

Guidelines:

. All processes (logic + tech) included

. infra demarcation (local hosting now, cloud possible later)

. external interfaces (mentioned for future integration)

. data storage (CSV files, model files)

. ML model integration shown



**Table-1 : Components & Technologies:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
|  | User Interface | Interface where users upload data and view results/graphs | HTML, CSS, JavaScript |
|  | Backend server | Handles user requests and connects frontend with ML model | Flask (Python) |
|  | Data Preprocessing | Cleans, transforms, and prepares data for model prediction | Pandas, NumPy |
|  | Machine Learning Model | Predicts traffic volume based on trained data | Scikit-learn/Random Forest/XGBoost |
|  | Visualization module | Displays graphs like histogram, bar chart for analysis | Matplotlib, Seaborn |
|  | Data storage | Stores the dataset and model files locally or in cloud | Local File System/ CSV / Pickle |
|  | External interface | Interface to future APIs (real-time traffic or sensor data sources) | Open APIs (for future scalability) |
|  | Deployment platform | Platform where app is hosted and accessed | Localhost (can extend to Heroku) |

**Table-2: Application Characteristics:**

| **S.No** | **Characteristics** | **Description** | **Technology** |
| --- | --- | --- | --- |
|  | Open-Source Frameworks | List the open-source frameworks used | Flask, scikit-learn, pandas, NumPy, matplotlib, seaborn |
|  | Security Implementations | Basic access control at backend, input validation, secure from handling. For future: encryption and OWASP best practices. | SHA-256(future use), Flask security, OWASP |
|  | Scalable Architecture | System supports modular design with separate layers for frontend, backend, and model logic. Can be extended to microservices or cloud deployment. | 3-tier architecture (web UI – App – ML model) |
|  | Availability | Localhost deployment currently; can be enhanced with cloud hosting and load balancers for high availability in production environments. | Cloud platforms (e.g., Heroku, AWS) |
|  | Performance | Designed for efficient model response. Use of optimized libraries ensures fast execution. Future enhancements may include caching and CDN integration. | Scikit-learn, pickle (for loading models quickly) |

**References:**

[**https://c4model.com/**](https://c4model.com/)

[**https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/**](https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/)

[**https://www.ibm.com/cloud/architecture**](https://www.ibm.com/cloud/architecture)

[**https://aws.amazon.com/architecture**](https://aws.amazon.com/architecture)

[**https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d**](https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d)