**Project Design Phase-II**

**Technology Stack (Architecture & Stack)**

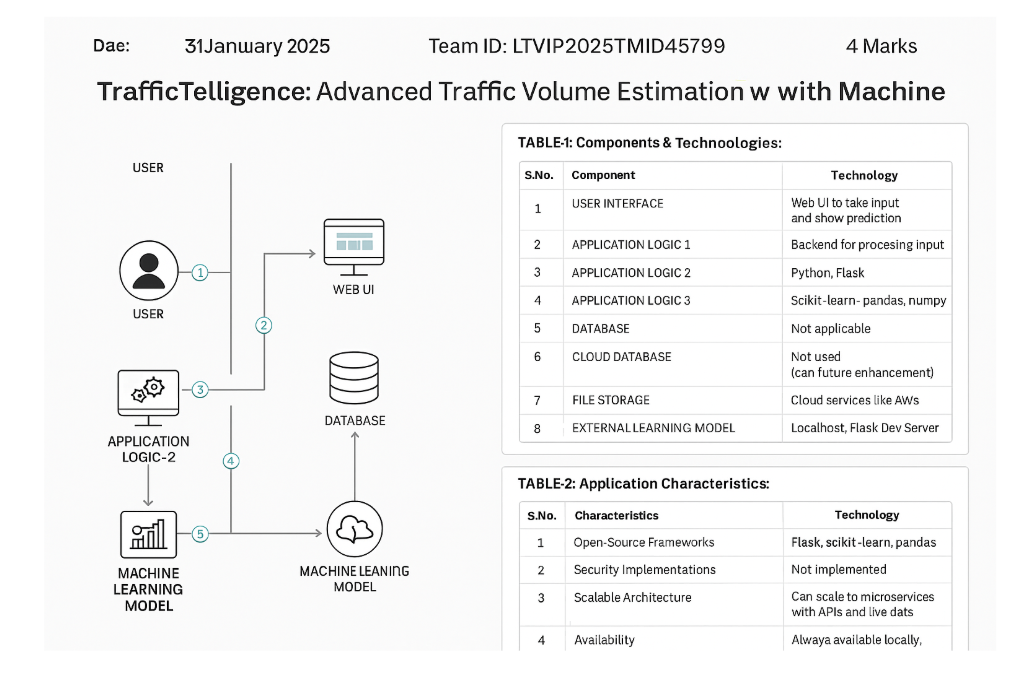
|  |  |
| --- | --- |
| Date | 31 January 3035 |
| Team ID | LTVIP2025TMID45799 |
| Project Name | **TrafficTelligence: Advanced Traffic Volume Estimation with Machine** |
| Maximum Marks | 4 Marks |

**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/)



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

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
|  | User Interface | Web UI to take input and show prediction | HTML, CSS, JavaScript |
|  | Application Logic-1 | Backend for processing input | Python, Flask |
|  | Application Logic-2 | ML-based prediction logic | scikit-learn, pandas, numpy |
|  | Application Logic-3 | Not applicable | — |
|  | Database | Local CSV or SQLite for storing logs | CSV, SQLite |
|  | Cloud Database | Not used (can be future enhancement) | — |
|  | File Storage | For saving prediction logs | Local filesystem |
|  | External API-1 | Weather Data API (optional future enhancement) | OpenWeatherMap API (optional) |
|  | External API-2 | Not applicable | — |
|  | Machine Learning Model | Predict traffic volume using trained model | RandomForestRegressor (scikit-learn) |
|  | Infrastructure (Server / Cloud) | Deployment and hosting | Localhost, Flask Dev Server (can move to AWS/GCP) |

**Table-2: Application Characteristics:**

| **S.No** | **Characteristics** | **Description** | **Technology** |
| --- | --- | --- | --- |
|  | Open-Source Frameworks | List the open-source frameworks used | Flask, scikit-learn, pandas |
|  | Security Implementations | Basic data validation (no login implemented). | Not implemented |
|  | Scalable Architecture | Can scale to microservices with APIs and live database later | Flask (monolithic now, can modularize) |
|  | Availability | Always available locally; can deploy on cloud for 24x7 access | Cloud services like AWS, GCP (future) |
|  | Performance | Fast prediction with lightweight ML model | Efficient backend, low latency Flask |

**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)