

Project Design Phase-II

Technology Stack (Architecture & Stack)

| | |
|----------------------|--|
| Date | 7 FEBRUARY 2026 |
| Team ID | LTVIP2026TMIDS79486 |
| Project Name | Project - Online Payments Fraud Detection Using ML |
| Maximum Marks | 4 Marks |

Technical Architecture:

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

Guidelines:

1. Include all the processes (As an application logic / Technology Block)
2. Provide infrastructural demarcation (Local / Cloud)
3. Indicate external interfaces (third party API's etc.)
4. Indicate Data Storage components / services
5. Indicate interface to machine learning models (if applicable)

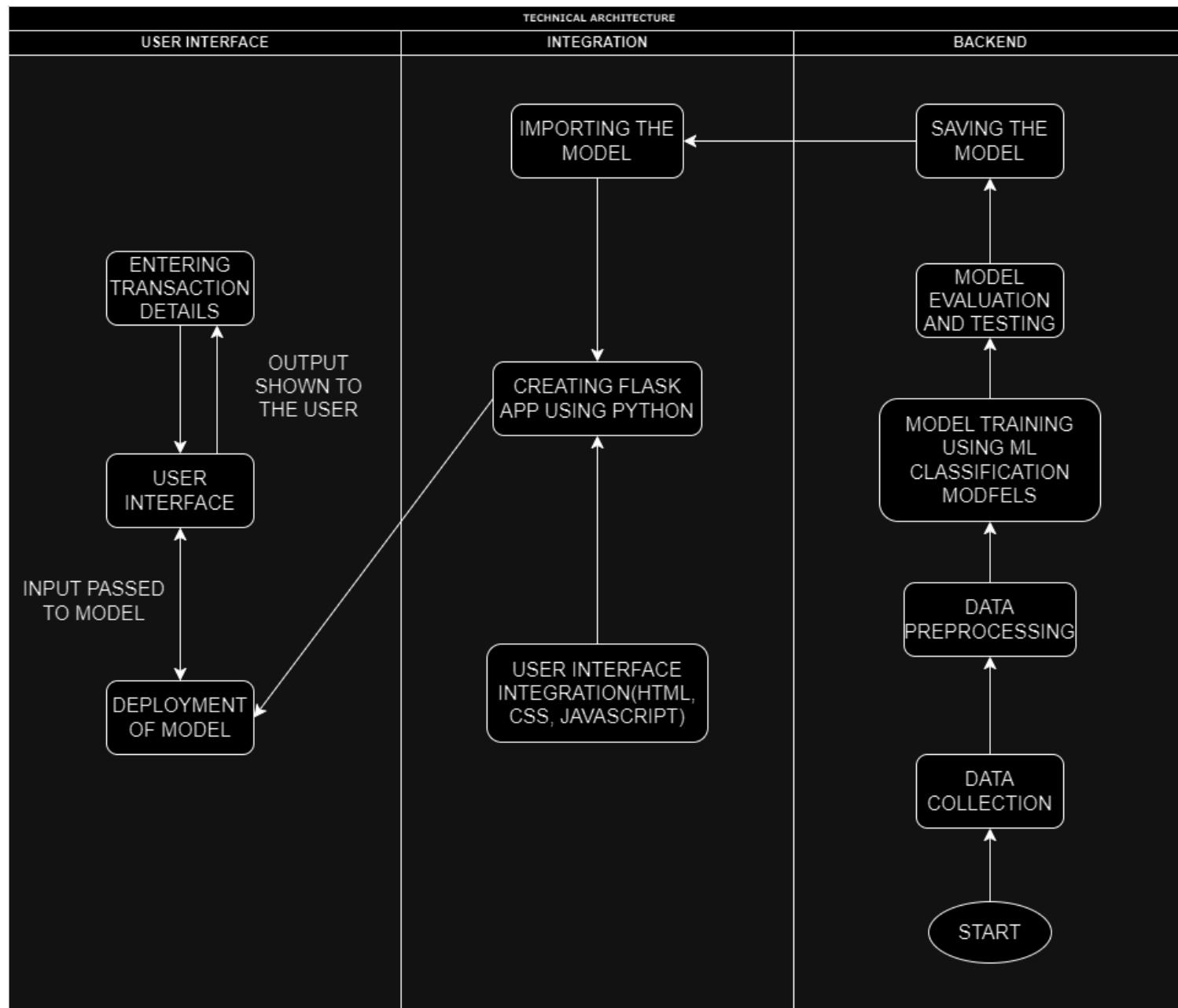


Table-1 : Components & Technologies:

| S.No | Component | Description | Technology |
|------|------------------------|---|--|
| 1. | User Interface | How user interacts with application e.g. Web UI, Mobile App, Chatbot etc. | HTML, CSS, JavaScript |
| 2. | Data Collection | Gathering the Transaction details | Kaggle |
| 3. | Data Preprocessing | Feature Scaling, Normalization, data splitting, Data Visualization, etc. | Seaborn, Matplotlib, Scikit-learn, Numpy, Pandas |
| 4. | Application Logic | Core Application Logic | Python (Flask) |
| 5. | Database | Storing Transaction Data | Relational: PostgreSQL or NoSQL: MongoDB |
| 6. | File Storage | Storing and Analyzing Data | Amazon S3, Google Cloud Storage |
| 7. | Deployment | Creating API or application for predicting the decisions | Flask API |
| 7. | Machine Learning Model | Fraud Detection Models | Decision Tree, Random Forest Classifier, SVM, XGBoost, Extra Tree Classifier |
| 8. | Evaluation | Model Evaluation on validation and testing model using dataset | Accuracy Score, Confusion Matrix, Classification Report |
| 9. | Infrastructure | Application Deployment on Cloud | Local: On-Premises Servers or Cloud: AWS, Azure, Google Cloud and Scaling Kubernetes, Docker |

Table-2: Application Characteristics:

| S.No | Characteristics | Description | Technology |
|------|------------------------|--|---|
| 1. | Open-Source Frameworks | Utilizing open-source frameworks and tools | Python Flask, Bootstrap |
| 2. | Scalable Architecture | Embracing a scalable microservices | Microservices architecture using Docker and Kubernetes, Load balancing using Nginx |
| 3. | Availability | Ensuring high availability through redundancy | Load balancers for even traffic distribution, redundant servers for failover, global server redundancy for uninterrupted service |
| 4. | Performance | Optimizing performance through caching and CDN integration | Caching with Redis for rapid data retrieval, Content Delivery Networks (CDNs) for faster content delivery, Performance monitoring and optimization for handling a specified number of requests per second |