

## Phase-2 Submission

**Student Name:** Dinesh A

**Register Number:** 712523104019

**Institution:** PPG INSTITUTE OF TECHNOLOGY

**Department:** CSE

**Date of Submission:** 10-05-2025

**GitHub Repository Link:**

[https://github.com/Dineshkavitha2005/NM\\_Dinesh\\_DS.git](https://github.com/Dineshkavitha2005/NM_Dinesh_DS.git)

---

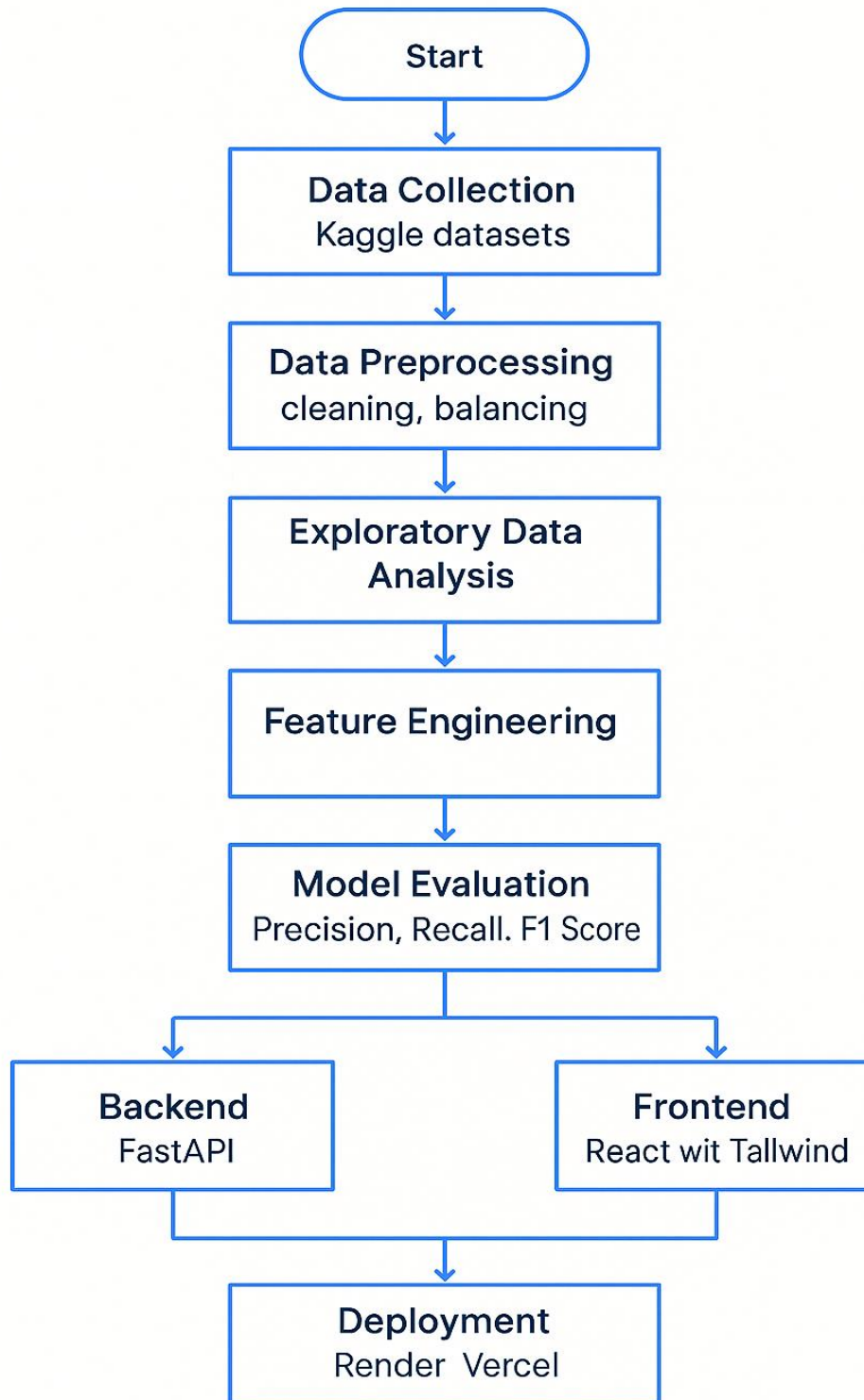
### 1. Problem Statement

*Credit card fraud continues to cause massive financial losses globally, and traditional detection methods are not adaptive enough for modern fraud patterns. This project tackles a binary classification problem using machine learning to detect fraudulent transactions. Solving this helps improve financial security, prevent monetary loss, and protect customers in digital banking environments.*

### 2. Project Objectives

- *Detect fraudulent transactions using classification models.*
- *Achieve high accuracy, recall, and F1-score.*
- *Build a real-time fraud detection API and frontend interface.*
- *Provide visual insights to explain prediction behavior.*
- *Objectives evolved after EDA: focus shifted to handling class imbalance and improving recall.*

### 3. Flowchart of the Project Workflow



## 4. Data Description

- **Dataset:** Credit Card Fraud Detection (Kaggle)
- **Type:** Structured, tabular
- **Records:** 284,807 rows, 31 features
- **Static dataset**
- **Target Variable:** Class (0 = Not Fraud, 1 = Fraud)
- **Data Source Link:** <https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud?select=creditcard.csv>

Transaction ID	Transaction Date	Amount	Merchant ID	Transaction Type	Location	Is Fraud
1	15:35.5	4189.27	688	refund	San Antonio	0
2	20:35.5	2659.71	109	refund	Dallas	0
3	08:35.5	784	394	purchase	New York	0
4	50:35.5	3514.4	944	purchase	Philadelphia	0

## 5. Data Preprocessing

- *No missing values found*
- *Duplicates checked and removed*
- *Outliers detected via boxplots and z-score*
- *Data is already scaled (PCA-transformed), so minimal normalization*
- *No categorical variables*
- *Final data checked for consistency and balanced using under sampling*

## 6. Exploratory Data Analysis (EDA)

- **Univariate:** Countplot showed only 492 frauds (~0.17%).
- **Bivariate:** High correlation between some PCA components and Class.
- **Insights:** Feature V14 and V17 strongly impact prediction.
- **Target imbalance highlighted:** required use of stratified sampling.

## 7. Feature Engineering

- Created new binary feature: *is\_high\_amount*
- Added transaction time binning
- Removed low-variance features
- Feature selection with correlation and importance analysis
- PCA already applied in dataset, so dimensionality reduction was not repeated

## 8. Model Building

- *Models used: Logistic Regression, Random Forest, XGBoost*
- *Data split: 80% train, 20% test (stratified)*
- *Random Forest and XGBoost showed best recall*
- *Evaluation metrics:*
  - *Accuracy: ~99.9% (but not sufficient alone)*
  - *Precision & Recall: Focused on high recall due to fraud sensitivity*
  - *F1-score: Balanced evaluation used to compare models*

## 9. Visualization of Results & Model Insights

- *Confusion matrix: Showed low false negatives in XGBoost*
- *ROC curve: AUC > 0.98 for best models*
- *Feature importance: V14, V10, V17 were top predictors*
- *Charts used: bar plots, heatmaps, confusion matrix, ROC curve*

## 10. Tools and Technologies Used

- **Programming Language:** Python
- **IDE:** Jupyter Notebook, VS Code
- **Libraries:** pandas, numpy, seaborn, matplotlib, scikit-learn, xgboost
- **Visualization Tools:** matplotlib, seaborn, plotly
- **Backend API:** FastAPI
- **Frontend:** React + Tailwind CSS
- **Deployment:** Render, Vercel

## 11. Team Members and Contributions

NAME	ROLE	WORK
Jayaprakash K	Frontend Developer	UI using React, styling with Tailwind
Prajith R	Backend Developer	FastAPI model integration
Lohith R	ML Engineer	Preprocessing, EDA, ML modeling
Dinesh A	Documentation and Presentation	Reports, PPT, flowcharts
Prakadeeshwaran A	Testing and Deployment	QA, Vercel + Render setup