**Phase-2 Submission Template**

**Student Name:**Subash chandra bose M

**Register Number:**712523205059

**Institution:**PPG Institute of Technology

**Department:**Information Technology

**Date of Submission:**

**Github Repository Link:**

### **1. Problem Statement**

### *In the modern digital economy, the exponential growth of online transactions has significantly increased the risk of credit card fraud, resulting in substantial financial losses and erosion of consumer trust. Traditional rule-based fraud detection systems often fail to identify novel or subtle fraudulent behaviors in real time, leading to either undetected fraud or a high rate of false positives that inconvenience legitimate customers. This project aims to design and develop an AI-powered credit card fraud detection and prevention system that leverages machine learning algorithms to analyze transactional patterns, detect anomalies, and prevent fraudulent activities in real time. The system should continuously learn from new data, adapt to emerging fraud tactics, and maintain a high accuracy rate with minimal false alarms, thereby ensuring secure, efficient, and user-friendly transaction processing.*

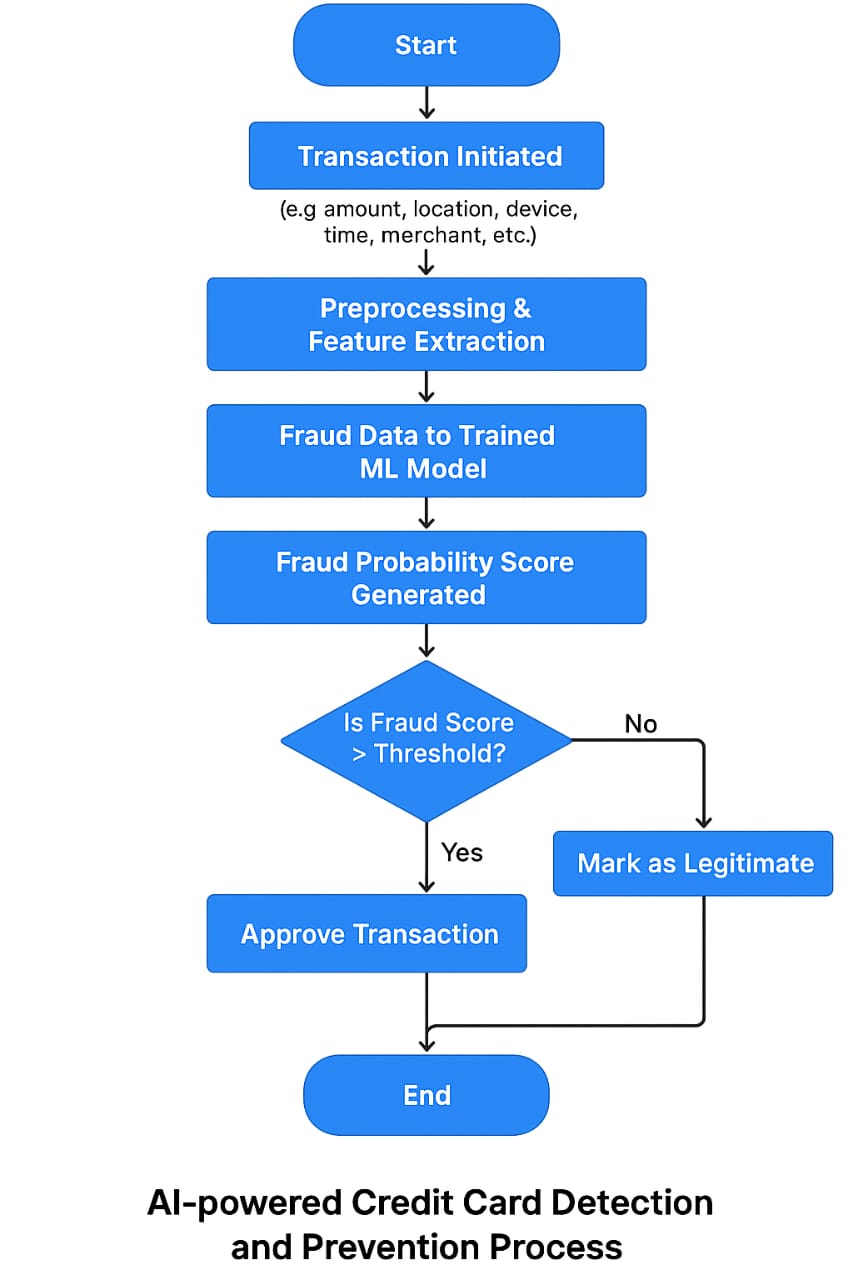
### **Project Objectiv****es**

The objective of this project is to develop an AI-powered system for real-time credit card fraud detection and prevention that accurately identifies and blocks fraudulent transactions while minimizing false positives.

The system will use machine learning models trained on transactional data to detect unusual patterns, adapt to evolving fraud tactics, and provide actionable alerts or automated responses.

This will enhance transaction security, reduce financial losses, and improve customer trust in digital payment systems.

### **3. Flowchart of the Project Workflow**



### 

### **4. Data Description**

*The dataset used for credit card fraud detection consists of anonymized transaction records, each representing an individual credit card transaction.*

1. *Transaction ID*

*2. Timestamp*

*3. Transaction Amount*

*4. Merchant Category*

*5. Location (Geolocation/IP)*

*6. Cardholder ID*

*7. Device Information*

*8. Transaction Mode*

*9. Previous Transaction Patterns*

### **5. Data Preprocessing**

1. *Data Preprocessing*
2. *Data cleaning*
3. *Feature engineering*
4. *Anomaly detection preprocessing*
5. *Label balancing*
6. *Train test split*
7. *Data normalization*

### **6. Exploratory Data Analysis (EDA)**

* *Univariate Analysis:*
  + *Histograms of transaction amount, time, etc.*
  + *Box plots to observe outliers (especially for fraudulent transactions)*
  + *KDE plots to compare feature distributions for fraud vs. legit*
* *Bivariate/Multivariate Analysis*

*Scatter plots or pair plots for selected features*

* + *Grouped bar plots (e.g., transaction mode vs. fraud rate)*
* *Insights Summary:  
  The dataset shows a strong class imbalance, with fraudulent transactions being rare but distinct in behavior, especially in terms of amount and timing. Key features like transaction amount, time, and location reveal useful patterns for accurate fraud detection.*

### 

### **Feature Engineering**

### 

* *1. Time Features: Hour, day of the week, time since last transaction.*
* *2. Behavioral Features: Average and standard deviation of transaction amount.*
* *3. Location Features: Distance from last location, unusual locations.*
* *4. Device Features: New vs. known device, device consistency.*
* *5. Historical Aggregates: User's fraud count, transaction count in past 24h.*
* *6. Merchant Features: Unusual merchant types for user.*
* *7. Risk Scores: Assign fraud risk based on merchant or location.*

### 

### **8. Model Building**

* + *1. Data Splitting: Split into training and test sets with stratified sampling.*
  + *2. Model Selection: Try models like Logistic Regression, Random Forest, XGBoost, or SVM.*
  + *3. Handle Imbalance: Use SMOTE, undersampling, or class weighting.*
  + *4. Model Training: Train and tune models with cross-validation.*
  + *5. Evaluation: Focus on Precision, Recall, F1-Score, and ROC-AUC.*
  + *6. Model Tuning: Optimize with GridSearchCV or RandomizedSearchCV.*
  + *7. Model Comparison: Select the best model based on performance metrics.*

### 

### 

### **9. Visualization of Results & Model Insights**

* *1. Confusion Matrix: Visualize true positives, false positives, true negatives, and false negatives to assess model performance.*
* *2. ROC Curve: Plot the ROC curve to evaluate model discrimination ability.*
* *3. Precision-Recall Curve: Useful for imbalanced datasets to evaluate trade-offs between precision and recall.*
* *4. Feature Importance: Use bar charts to display the most important features influencing fraud prediction (e.g., transaction amount, time).*
* *5. Model Comparison: Compare different models using AUC, F1-Score, or Precision-Recall AUC in a bar chart or table.*

### 

### **10. Tools and Technologies Used***.*

* *Programming Language: Python or R.*
* *IDE/Notebook: Google Colab, Jupyter Notebook, VS Code, etc.*
* *Libraries: pandas, numpy, seaborn, matplotlib, scikit-learn, XGBoost, etc.*
* *Visualization Tools: Plotly, Tableau, Power BI.]*

### 

### **11. Team Members and Contributions**

*Data cleaning - Subash chandra bose . M*

*EDA - Srisankari U*

*Feature engineering - Sowmiya U*

*Model Development - Thangapandi P*

*Documentation and reporting - Suresh Kumar U*