ONLINE FRAUD PAYMENT DETECTION SYSTEM

---

1. Abstract

Online payment systems have grown rapidly due to digital transformation. However, this growth has led to increased fraudulent activities. Fraud detection is essential to prevent financial losses and ensure transaction security.

This project presents a machine learning–based fraud detection system that predicts whether a transaction is fraudulent or legitimate. The system uses historical transaction data to train classification algorithms and deploys the trained model as a web application for real-time fraud prediction.

---

6. Ideation Phase

6.1 Introduction

Online payment fraud is a growing concern in digital financial systems. Traditional systems are inefficient; machine learning provides adaptive solutions to detect fraud effectively.

6.2 Problem Statement

Design a web-based machine learning system to predict fraudulent transactions accurately.

6.3 Proposed Solution

- Preprocess dataset
- Train a classification model
- Deploy as a web application for real-time prediction

6.4 Objectives & Scope

- Fraud detection with ML
- Web-based real-time predictions
- Scalable for financial institutions

6.5 Feasibility

- **Technical:** Python, ML libraries, Flask
- **Economic:** Open-source tools, low cost
- **Operational:** Easy-to-use interface

---

7. Requirement Analysis

7.1 Functional Requirements

- Input transaction details
- Predict Fraudulent / Legitimate

- Display results on web interface

7.2 Non-Functional Requirements

- Performance (<2 sec prediction)
- Security (HTTPS, input validation)
- Scalability, reliability, maintainability

7.3 Hardware & Software Requirements

- **Hardware:** 8GB RAM, Intel i3+, 500GB storage
- **Software:** Python 3.8+, Flask, Pandas, NumPy, Scikit-learn

---

8. Project Design Phase

8.1 System Architecture

- Data Module → Model Module → Web App → Deployment

8.2 Module Design

- Data preprocessing
- Model training & evaluation
- Web interface
- Deployment

8.3 Database Design (Optional)

Table	Columns	Description
transactions	id, amount, type, user_id, label	Store transaction data

8.4 UML & Diagrams

- Use Case: User inputs transaction, admin views reports
- Sequence Diagram: User → Web → Model → Result
- Class Diagram: Transaction, Model, WebApp

---

9. Project Planning Phase

9.1 Timeline & Milestones

Phase	Duration	Task
Ideation	1–2 weeks	Problem identification

Phase	Duration	Task
Design	3–4 weeks	Architecture, UML diagrams
Development	5–12 weeks	Data preprocessing, ML model, Web app
Deployment	11–12 weeks	Cloud hosting
Documentation	13–14 weeks	Prepare final report

9.2 Resources

- Hardware: PC/Laptop
- Software: Python, Flask, Jupyter Notebook
- Data: Transaction datasets
- Cloud: Render / Heroku / AWS

9.3 Risk Management

- Dataset quality → Data cleaning
- Model accuracy → Hyperparameter tuning
- Deployment → Cloud best practices

---

10. Project Development Phase

10.1 Data Preprocessing

- Clean data, handle missing values
- Scale features, encode categorical variables
- Balance dataset using SMOTE

10.2 Model Development

- Train-test split 80:20
- Algorithm: Random Forest / Logistic Regression
- Evaluation: Accuracy, Precision, Recall, F1-score

10.3 Web Application

- Flask interface with HTML/CSS forms
- /predict endpoint integration
- Display real-time prediction

10.4 Deployment

- Cloud hosting on Render/Heroku
- Secure HTTPS communication
- Testing for reliability and performance

10.5 Testing

- Unit testing, integration testing, system testing
- User Acceptance Testing (UAT)

---

11. Results & Screenshots

- Web form for transaction input
- Prediction: Fraudulent / Legitimate
- Confusion Matrix & Accuracy

---

12. Advantages & Limitations

Advantages:

- Real-time fraud detection
- User-friendly interface
- Scalable system

Limitations:

- Dependent on dataset quality
- False positives/negatives possible
- Requires retraining for new data

---

13. Future Enhancements

- Deep learning model for higher accuracy
- Integration with banking APIs
- Mobile application
- Multi-layer security for transactions

---

14. Conclusion

The project demonstrates the effective application of machine learning for online fraud detection. The web-based system predicts fraudulent transactions in real-time, providing enhanced security for online payments.

---

15. References

1. Scikit-learn Documentation – Machine Learning Library
2. Flask Documentation – Web Framework
3. IEEE Papers on Fraud Detection
4. Research papers on Random Forest & Logistic Regression