Online Payments Fraud Detection using Machine Learning

1. INTRODUCTION

1.1 Project Overview

Online payment systems have become an essential part of modern digital transactions. With the rapid growth of e-commerce, mobile banking, and digital wallets, fraudulent activities in online payments have also increased significantly. Fraudulent transactions lead to huge financial losses for banks, businesses, and customers.

This project, Online Payments Fraud Detection, is a machine learning-based web application that predicts whether a given transaction is fraudulent or genuine. The system analyzes transaction details such as transaction type, amount, old balance, and new balance to determine fraud probability.

The model is trained using historical transaction data and deployed as a web application for real-time fraud prediction.

1.2 Purpose

The main purpose of this project is:

- To build a predictive system that identifies fraudulent online transactions.
- To reduce financial losses caused by fraud.
- To deploy a trained ML model as a web application.
- To demonstrate practical implementation of ML in real-world financial systems.

2. IDEATION PHASE

2.1 Problem Statement

Online payment fraud has become a major issue in digital financial systems. Traditional rule-based fraud detection systems are not efficient enough to handle modern fraud patterns.

Problem:
Design and develop a machine learning-based system that can accurately detect fraudulent online payment transactions using historical transaction data.

2.2 Empathy Map Canvas

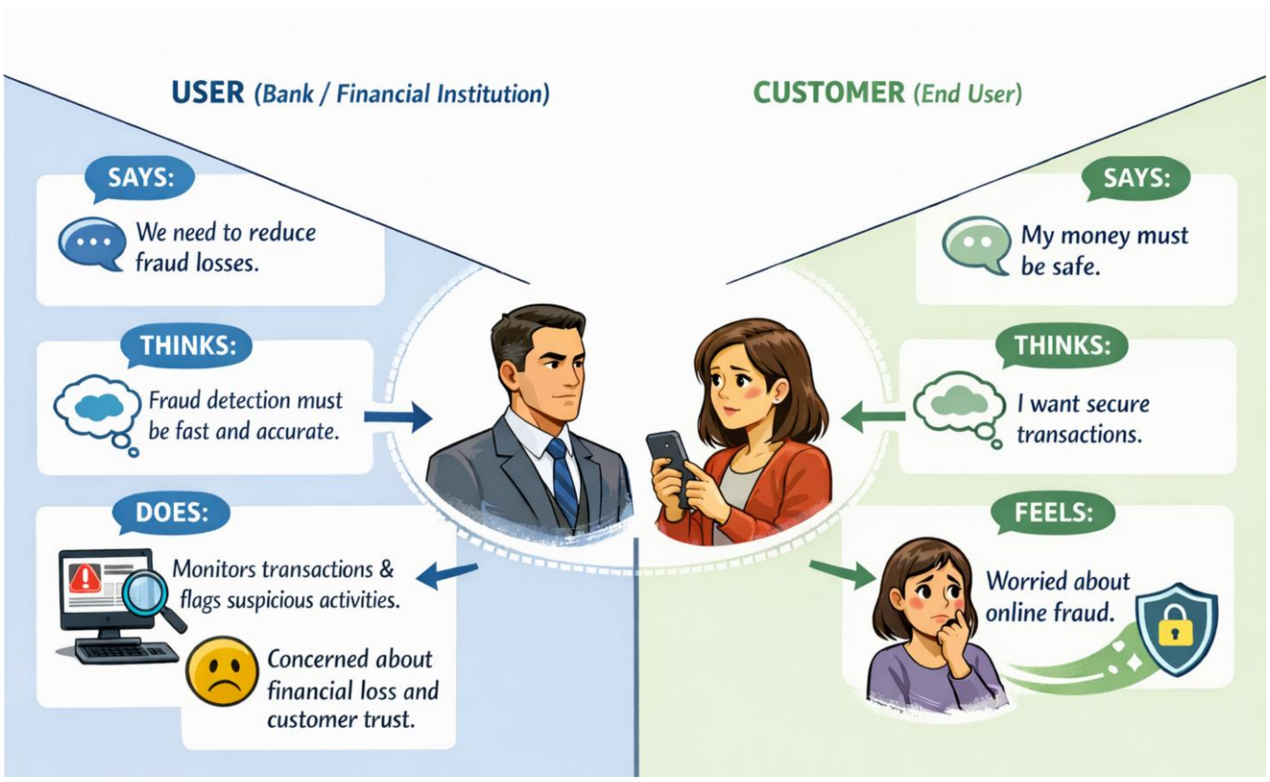
User (Bank / Financial Institution):

- Says: We need to reduce fraud losses.
- Thinks: Fraud detection must be fast and accurate.

- Does: Monitors transactions and flags suspicious activities.
- Feels: Concerned about financial loss and customer trust.

Customer (End User):

- Says: My money must be safe.
- Thinks: I want secure transactions.
- Feels: Worried about online fraud.


USER (Bank / Financial Institution)
CUSTOMER (End User)
SAYS:
We need to reduce fraud losses.
THINKS:
Fraud detection must be fast and accurate.
DOES:
Monitors transactions & flags suspicious activities.
Concerned about financial loss and customer trust.
SAYS:
My money must be safe.
THINKS:
I want secure transactions.
FEELS:
Worried about online fraud.

2.3 Brainstorming

Different approaches were discussed:

- Rule-based fraud detection
- Statistical anomaly detection
- Machine learning models (Logistic Regression, Random Forest, XGBoost)
- Real-time fraud scoring system

Finally, machine learning-based classification was selected as the best solution due to higher accuracy and adaptability.

3. REQUIREMENT ANALYSIS

3.1 Customer Journey Map

1. User enters transaction details.
2. Data is sent to backend.

3. ML model processes input.
4. System predicts fraud or genuine.
5. Result is displayed to user.

3.2 Solution Requirement

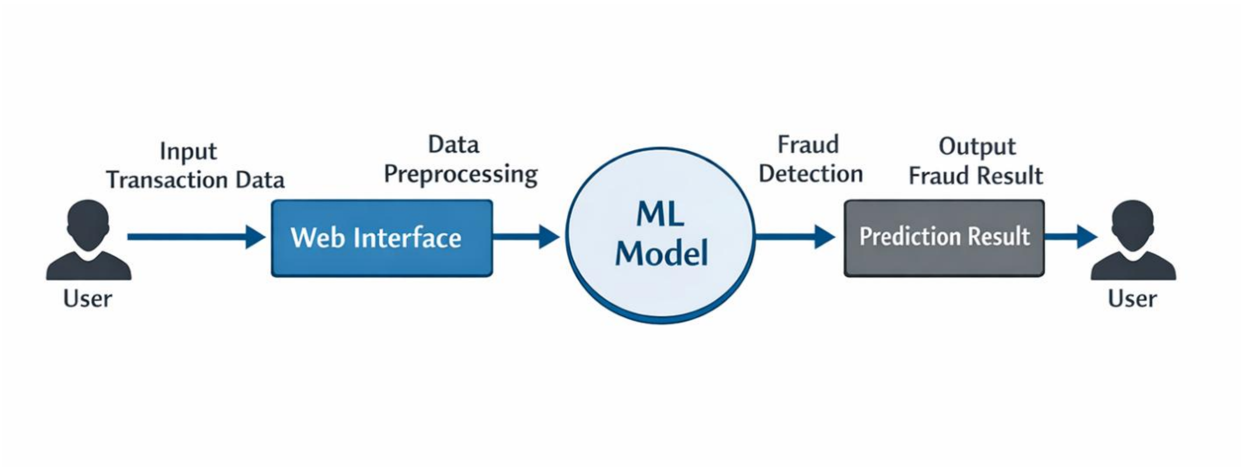
Functional Requirements:

- User should input transaction details.
- System should process input data.
- System should predict fraud status.
- Display result clearly.

Non-Functional Requirements:

- Fast prediction time.
- High accuracy.
- Secure data handling.
- Easy-to-use interface.

3.3 Data Flow Diagram (DFD)



3.4 Technology Stack

- Programming Language: Python
- Machine Learning Library: XGBoost / Scikit-learn
- Frontend: HTML, CSS
- Backend Framework: Flask
- Deployment Platform: Render
- Dataset Source: Kaggle Online Payment Fraud Dataset

4. PROJECT DESIGN

4.1 Problem Solution Fit

The problem requires real-time fraud detection with high accuracy. Machine learning models are suitable because:

- They learn complex patterns.
- They improve over time with more data.
- They reduce manual monitoring.

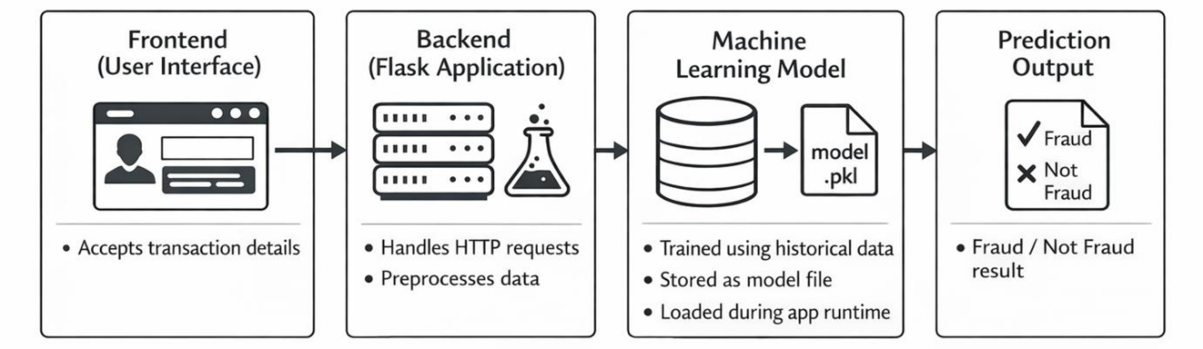
Thus, ML-based classification fits the problem effectively.

4.2 Proposed Solution

The proposed system:

- Takes transaction input.
- Converts categorical values to numerical form.
- Applies trained ML model.
- Predicts whether transaction is fraudulent or genuine.
- Displays result with prediction probability.

4.3 Solution Architecture


Frontend (User Interface)
• Accepts transaction details
Backend (Flask Application)
• Handles HTTP requests
• Preprocesses data
Machine Learning Model
model .pkl
• Trained using historical data
• Stored as model file
• Loaded during app runtime
Prediction Output
✓ Fraud
✗ Not Fraud
• Fraud / Not Fraud result

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

Phase 1 – Research

- Study fraud detection techniques.
- Explore dataset.

Phase 2 – Data Preprocessing

- Clean data.
- Handle missing values.
- Encode categorical variables.

Phase 3 – Model Training

- Train multiple models.
- Compare accuracy.
- Select best model.

Phase 4 – Web Application Development

- Build frontend.
- Integrate ML model.

Phase 5 – Testing & Deployment

- Test predictions.
- Deploy on Render.

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Performance Testing

Model Evaluation Metrics:

- Accuracy
- Precision
- Recall
- F1-Score
- Confusion Matrix

Performance Result:

- Model achieved high accuracy.
- Good fraud detection rate.
- Low false positive rate.

Prediction response time is fast (less than a few seconds).

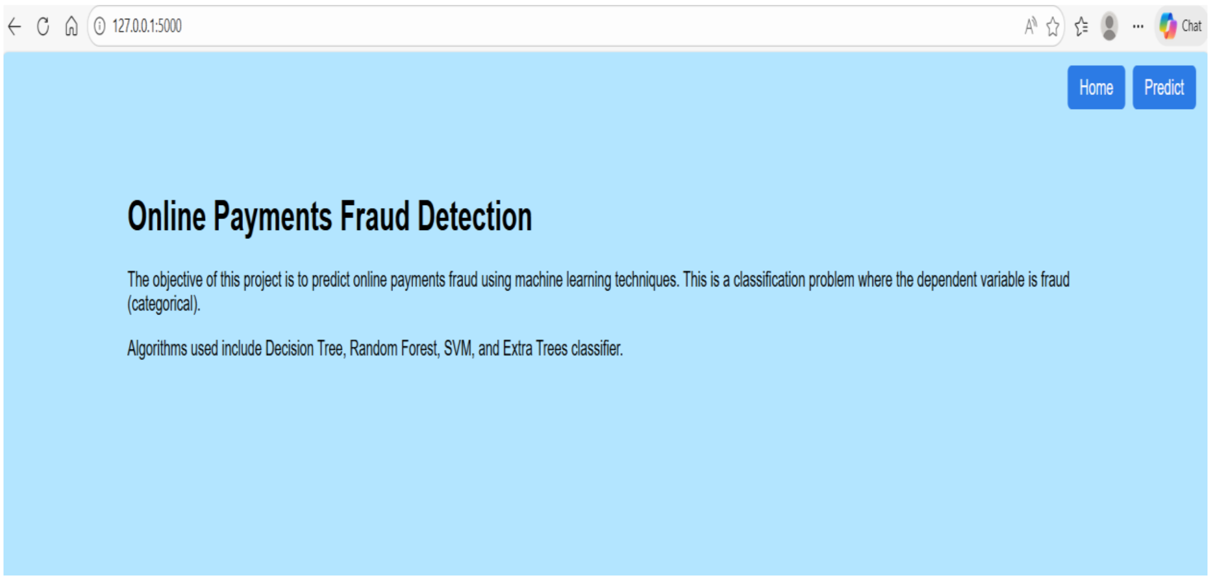
7. RESULTS

The system successfully:

- Detects fraudulent transactions.
- Provides real-time prediction.
- Works through web interface.
- Gives reliable fraud probability.

7.1 Output Screenshots

a. Home Page

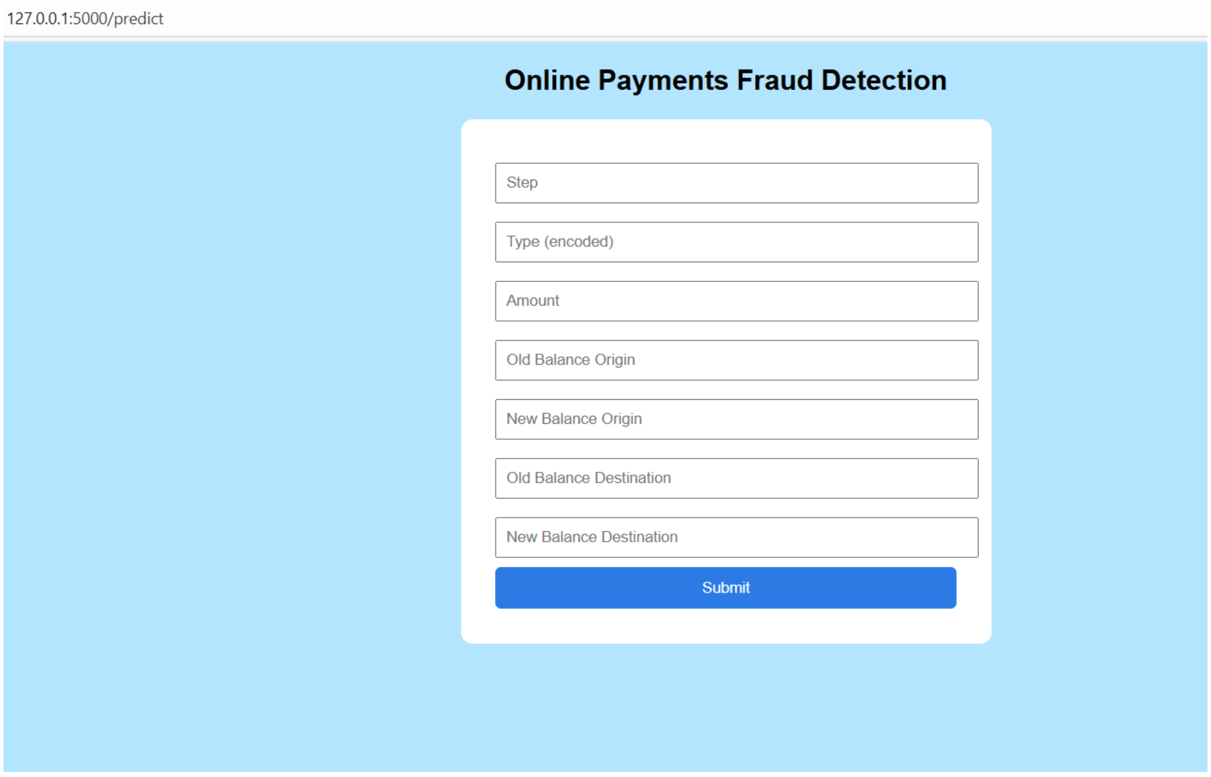

127.0.0.1:5000

Online Payments Fraud Detection

The objective of this project is to predict online payments fraud using machine learning techniques. This is a classification problem where the dependent variable is fraud (categorical).

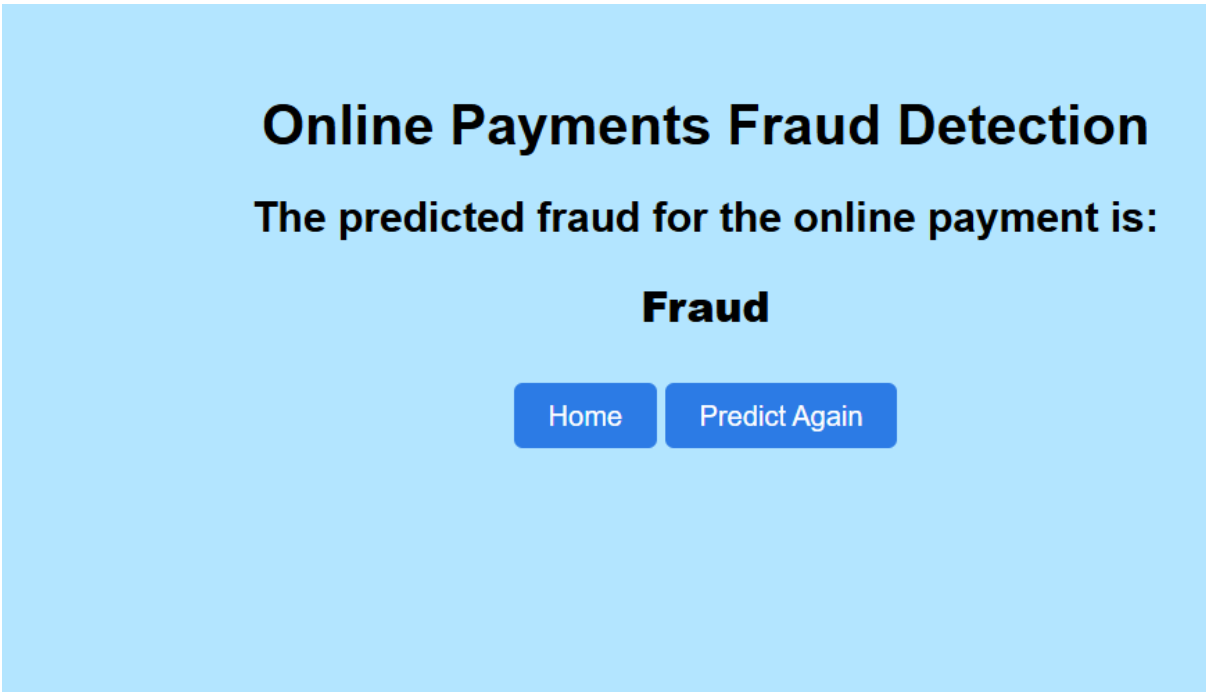
Algorithms used include Decision Tree, Random Forest, SVM, and Extra Trees classifier.

b. Input Form


127.0.0.1:5000/predict

Online Payments Fraud Detection

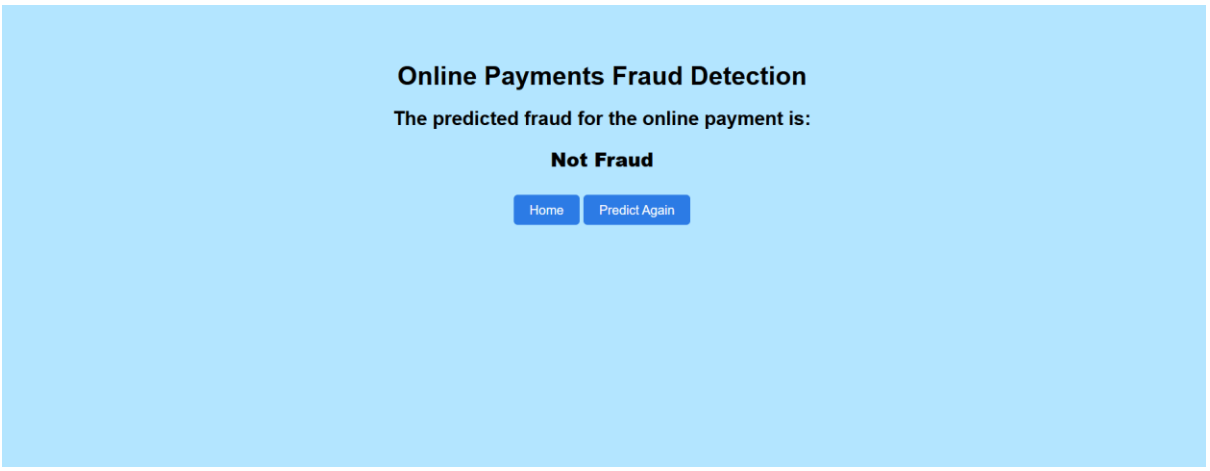
c. Fraud Prediction Result


Online Payments Fraud Detection

The predicted fraud for the online payment is:

Fraud

d. Genuine Transaction Result


Online Payments Fraud Detection

The predicted fraud for the online payment is:

Not Fraud

8. ADVANTAGES & DISADVANTAGES

Advantages:

- Real-time fraud detection
- High prediction accuracy
- Automated system
- Reduces financial losses

Disadvantages:

- Requires quality training data
- May produce false positives
- Needs regular model updates

9. CONCLUSION

The Online Payments Fraud Detection system successfully demonstrates how machine learning can be used to detect fraudulent transactions. The model accurately classifies transactions based on input features and provides real-time results through a web application. This project highlights the importance of AI in financial security and fraud prevention.

10. FUTURE SCOPE

- Integration with live banking systems
- Deep learning-based fraud detection
- Real-time streaming data analysis
- Mobile application integration
- Continuous model retraining

11. APPENDIX

Source Code:

Flask Code:

```
from flask import Flask, render_template, request

import pickle

import numpy as np

app = Flask(__name__)

# Load trained model

model = pickle.load(open('payments.pkl', 'rb'))

@app.route('/')

def home():

   return render_template('home.html')

@app.route('/predict')

def predict():
   return render_template('predict.html')
@app.route('/submit', methods=['POST'])
def submit():
   step = float(request.form['step'])
   type_ = float(request.form['type'])
   amount = float(request.form['amount'])
   oldbalanceOrg = float(request.form['oldbalanceOrg'])
   newbalanceOrig = float(request.form['newbalanceOrig'])
   oldbalanceDest = float(request.form['oldbalanceDest'])
   newbalanceDest = float(request.form['newbalanceDest'])
   # SAME ORDER as training
   features = np.array([[step, type_, amount,
                         oldbalanceOrg, newbalanceOrig,
                         oldbalanceDest, newbalanceDest]], dtype=float)
   prediction = model.predict(features)
   result = "Fraud" if prediction[0] == 1 else "Not Fraud"
   return render_template('submit.html', prediction=result)
if __name__ == "__main__":
   app.run(debug=True)
```

ML Code:

```
import pandas as pd
import pickle
import numpy as np
from xgboost import XGBClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score


# Load dataset

df = pd.read_csv("payments.csv")


# Encode type column

type_map = {

    "PAYMENT": 0,

    "TRANSFER": 1,

    "CASH_OUT": 2,

    "DEBIT": 3,

    "CASH_IN": 4

}


df['type'] = df['type'].map(type_map)


# Select features in SAME ORDER as Flask

X = df[['step', 'type', 'amount',

        'oldbalanceOrg', 'newbalanceOrig',

        'oldbalanceDest', 'newbalanceDest']]

# Target column

y = df['isFraud']

# Split data

X_train, X_test, y_train, y_test = train_test_split(

    X, y, test_size=0.2, random_state=42
)

# Train XGBoost model

model = XGBClassifier(

    n_estimators=100,

    max_depth=5,

    learning_rate=0.1,

    use_label_encoder=False,

    eval_metric='logloss'

)

model.fit(X_train, y_train)

# Test accuracy

y_pred = model.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)

print("Model Accuracy:", accuracy)

# Save model

pickle.dump(model, open("payments.pkl", "wb"))

print("Model saved as payments.pkl")
```

Dataset Link:

Kaggle – Online Payment Fraud Detection Dataset
https://www.kaggle.com/datasets/rupakroy/online-payments-fraud-detection-dataset

GitHub & Project Demo Link

GitHub Repository:
https://github.com/Nandhuarumalla/fraud-detection-flask

Project Demo (Render):
https://fraud-detection-flask-9njt.onrender.com/