Artificial Intelligence and Machine Learning

Project Documentation format

1. Introduction

- **Project Title: Online Payments Fraud Detection using Machine Learning**

- **Team Members:** E Indravathi
  E Mahitha
  Ezhil R
  Gayathri L

Project Overview

- **Purpose:** Online payment fraud is increasing rapidly due to the growth of digital transactions such as UPI, credit card payments, and online banking. Fraudulent activities cause financial losses to customers and banks. Manual fraud monitoring systems are slow and rule-based systems fail to detect complex fraud patterns.
  The purpose of this project is to automatically detect fraudulent online transactions using Machine Learning techniques by analyzing transaction data such as amount, balances, and transaction type.
  This system helps in faster, accurate, and cost-effective fraud detection, ensuring secure digital payment systems.

Features:

- Upload transaction details for analysis
- Automatic classification of transactions (Fraud / Not Fraud)
- Machine learning–based prediction with high accuracy
- User-friendly web interface
- Secure handling of transaction data
- Fast and reliable prediction results
- Real-time fraud detection capability

2. Architecture

◇ Frontend

The frontend is developed using **React.js**, providing a responsive and interactive user interface.

Key Features:

- Transaction input form
- Display of fraud prediction results
- Clean and intuitive dashboard
- Real-time status alerts

◇ Backend

The backend is implemented using Node.js and Express.js, which:

- Handles transaction input requests
- Sends transaction data to the ML model for prediction
- Returns fraud classification results to the frontend
- Manages API routing and validation

◇ Machine Learning Model

- Developed using **Python (Scikit-learn)**
- Random Forest classifier used for fraud detection
- Model saved as .pkl file
- Integrated with backend using API calls

◇ Database

MongoDB is used to store:

- User details
- Transaction metadata
- Prediction results
- Fraud detection logs

3.Setup Instructions

- Prerequisites

Install the following software:

1.Node.js

2.npm

3.MongoDB

4.Python

5.Scikit-learn

6.Pandas

7.NumPy

8.OpenCV (if needed for image-based processing)

4.Folder Structure

```
client/
|— src/
|   ├— components/       # Reusable UI components
|   ├— pages/            # Home, Transaction Input, Results
|   ├— services/         # API service calls
|   ├— App.js            # Main React component
|   └— index.js          # Entry point
|— public/
|— package.json
```

server/

|— routes/ # API routes

|— controllers/ # Business logic

|— models/ # MongoDB schemas

|— app.js # Main backend file

|— model/ # ML model (.pkl file)

|— package.json|— package.json

3. Running the Application

◇ Frontend

Open terminal inside `client` directory:

```
npm install
npm start
```

◇ Backend

Open terminal inside `server` directory:

```
npm install
npm start
```

4. API Documentation

1. Predict Fraud

Endpoint:

```
POST /api/predict
```

Request Body Example:

```
{
  "amount": 5000,
  "oldbalanceOrg": 10000,
  "newbalanceOrig": 5000,
  "oldbalanceDest": 2000,
  "newbalanceDest": 7000,
  "type": "TRANSFER"
}
```

Response Example:

```
{
  "prediction": "Fraud",
  "confidence": 0.92
}
```

◇ 2. Get Transaction History

```
GET /api/transactions
```

Returns stored transaction prediction logs.

.

8. Authentication

Authentication is implemented using **JWT (JSON Web Tokens)**.

◇ **How It Works:**

1. User registers/logs in
2. Server generates JWT token
3. Token stored in frontend
4. Token sent with every request
5. Backend verifies token before processing

◇ **Security Measures:**

- Password hashing using bcrypt
- Protected API routes
- Token expiration handling
- Role-based access (Admin/User)