Full Stack Development with MERN

Project Documentation format

1. Introduction

Project Title: Online Payment Fraud Detection Using Machine Learning

Team Members:

* [CHARAN KUMAR] – Full Stack Developer
* [PRADEEP NAIDU] – Backend & ML Integration
* [VINAY] – Frontend Developer

2. Project Overview
Purpose
The purpose of this project is to detect fraudulent online payment transactions using Machine Learning techniques. The system analyzes transaction data in real-time and predicts whether a transaction is legitimate or fraudulent, helping financial institutions reduce fraud losses.

Features
* User registration and login
* Secure transaction input system
* Real-time fraud prediction using ML model
* Transaction history dashboard
* Admin panel to monitor fraud statistics
* Graphical fraud analysis reports
* Secure REST API integration

3. Architecture

Frontend
* Built using *React.js*
* Responsive UI for users and admin
* Axios used for API communication
* Dashboard with charts for fraud analytics

Backend
* Developed using *Node.js* and *Express.js*
* RESTful API architecture
* Handles authentication, transaction processing, and ML model integration

* Middleware for validation and security

Database
* *MongoDB*
* Stores:
  * User data
  * Transaction records
  * Fraud prediction results
* Mongoose used for schema modeling

Machine Learning Model
* Developed using Python (Scikit-learn / Random Forest / Logistic Regression)
* Trained on historical transaction dataset
* Integrated with backend using REST API or child process execution
* Outputs fraud probability score

4. Setup Instructions

Prerequisites
* Node.js
* MongoDB
* Python 3.x
* Required npm packages
* Required Python libraries (scikit-learn, pandas, numpy, joblib)

Installation
1. Clone the repository:
   git clone <repository-url>
2. Install backend dependencies:
   cd server
   npm install
3. Install frontend dependencies:
   cd client
   npm install
4. Set environment variables:
   * MONGO_URI
   * JWT_SECRET
   * PORT
5. Run the application.
5. Folder Structure
Client (React Frontend)

```
client/
 ├── src/
 │   ├── components/
 │   ├── pages/
 │   ├── services/
 │   ├── App.js
 │   └── index.js
```

Server (Node Backend)

```
server/
 ├── models/
 ├── routes/
 ├── controllers/
 ├── middleware/
 ├── ml-model/
 ├── server.js
```

6. Running the Application

Backend
cd server
npm start

Frontend
cd client
npm start
Backend runs on: http://localhost:5000
Frontend runs on: http://localhost:3000

7. API Documentation

User Routes

* POST /api/users/register – Register user
* POST /api/users/login – Login user

Transaction Routes

* POST /api/transactions – Add transaction & predict fraud

* GET /api/transactions – Get user transactions

Admin Routes

* GET /api/admin/stats – Get fraud statistics

8. Authentication

* JWT (JSON Web Token) based authentication
* Password hashing using bcrypt
* Protected routes using middleware
* Token stored securely on client side

9. User Interface

* Login & Registration page
* Transaction input form
* Fraud prediction result page
* Dashboard with fraud percentage graph
* Admin analytics panel

10. Testing

* Unit testing for backend APIs
* Model accuracy testing (Accuracy, Precision, Recall, F1-Score)
* Manual UI testing
* Postman used for API testing

11. Screenshots or Demo

* Screenshots of:

  * Login Page
  * Transaction Form
  * Fraud Detection Result
  * Admin Dashboard

(Demo link can be added here)

12. Known Issues

* Model accuracy depends on dataset quality
* Performance may slow with very large transaction data

* Requires retraining for new fraud patterns

13. Future Enhancements

* Real-time streaming fraud detection
* Deep learning implementation
* Integration with payment gateways
* SMS/Email alerts for fraud detection
* Deployment on cloud (AWS/Azure)
* Role-based access control