Solution Architecture

Online Payments Fraud Detection using Machine Learning

Architecture Overview

The application follows a simple 3-tier architecture: Presentation → Application → Model Layer.

Tier 1: Presentation Layer

- home.html — Landing page (notebook aesthetic, project overview)
- predict.html — Transaction input form (type selector + 6 numeric fields)
- submit.html — Result page (animated probability bar, verdict, annotation)
- Styling: inline CSS with Google Fonts (Caveat + Nunito)

Tier 2: Application Layer (Flask)

- app.py — Main Flask application with 3 routes
- Route / : renders home.html
- Route /predict : renders predict.html
- Route /submit (POST): validates input, runs inference, renders submit.html
- Error handling: KeyError and ValueError caught, user-friendly error returned

Tier 3: Model Layer

- payments.pkl — Serialized RandomForestClassifier (scikit-learn)
- Loaded once at application startup using pickle.load()
- Inference: model.predict_proba(features)[0][1] returns P(fraud)

File Structure

```
flask/
├── app.py              # Flask backend
├── payments.pkl        # Trained Random Forest model
├── requirements.txt    # Python dependencies
└── templates/
    ├── home.html       # Landing page
    ├── predict.html    # Input form
    └── submit.html     # Result page
```

```
training/
└── fraud_model.ipynb    # Jupyter notebook: EDA + model training
```

Deployment

- Local: python app.py → http://127.0.0.1:5000
- Requirements: Python 3.10+, Flask 3.0+, scikit-learn 1.3+, numpy, pandas
- Install: pip install -r requirements.txt
- Cloud deployment: Compatible with Heroku, Render, Railway (add Procfile)