**Fraud Detection System**

# Overview

This is a Flask-based machine learning application that detects fraudulent transactions using supervised learning algorithms. The app allows users to input transaction data via a simple web interface and predicts whether the transaction is legitimate or fraudulent.

# Project Structure

dsa-project/  
├── app.py # Flask web app  
├── train\_and\_save\_model.py # Model training script  
├── scaler.pkl # Saved scaler  
├── model.pkl # Trained ML model  
├── feature\_columns.pkl # Feature list used for inference  
├── smote\_balanced\_data.csv # Cleaned and balanced dataset  
├── prediction\_log.csv # Logs user inputs and predictions  
├── requirements.txt # Python package dependencies  
├── Procfile # For Heroku deployment  
├── runtime.txt # Python version (optional)  
├── templates/  
│ ├── index.html # Input form  
│ └── result.html # Output page

# Setup Instructions

1. **Clone the repository:**  
 git clone <https://github.com/btolawoyin/dsa.git>   
 cd dsa-fraud-detector

The following files were saved here due to git size restrictions: <https://bit.ly/dsa_ai_ml>

* smote\_balanced\_data.csv
* train\_transaction.csv

2. **Set up a virtual environment:**  
 python -m venv venv  
 venv\Scripts\activate # On Windows  
 source venv/bin/activate # On Mac/Linux  
  
3. **Install dependencies:**  
 pip install -r requirements.txt  
  
4. **Run the Flask app locally:**  
 python app.py  
 Open <http://127.0.0.1:5000> in your browser

# Model Details

The model was trained on a public dataset (train\_transaction.csv) with:  
- Class balancing via SMOTE  
- Feature scaling using StandardScaler  
- Categorical encoding for transaction types  
  
**Algorithms Used:**  
- Logistic Regression  
- Random Forest  
- XGBoost  
  
**Evaluation Metrics:**  
- Accuracy  
- Precision  
- Recall  
- F1 Score  
- ROC-AUC  
Deployment

**To deploy to Heroku:**  
heroku login  
heroku create your-app-name  
git push heroku main  
heroku open  
  
**Ensure your repo contains:**  
- Procfile  
- requirements.txt  
- runtime.txt (optional)

# Testing

Sample inputs (5 sets) for form testing are provided inside the app, or you can test with your own transaction values. Probabilities and predictions are logged in prediction\_log.csv.

# License

This project is for educational and non-commercial use only. Fraud detection models should be properly validated before use in production systems.

# Author

Built by Bukky as part of a final AI/ML course project.  
For enquiries or collaboration, feel free to connect via LinkedIn or email.