**Design and Deployment of an AI-Powered Predictive System**

**Project Overview**

**Project Title:**  
Design and Deployment of an AI-Powered Predictive System

**Project Question:**  
How can AI models be developed and deployed in real-world applications to support decision-making and improve task automation through a simple web interface?

**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.

**Use Case:**  
**Fraud Detection** – Identifying fraudulent transactions using payment data patterns.

**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  
├── runtime.txt # Python version  
├── templates/  
│ ├── index.html # Input form  
│ └── result.html # Output page

**Model Summary**

The following supervised ML models were trained and evaluated:

* Logistic Regression
* Random Forest
* XGBoost
* Neural Network (MLP Classifier)

**Performance Metrics:**

* Precision
* Recall
* F1 Score
* ROC AUC

Each model was assessed to determine predictive accuracy on imbalanced transaction datasets.

**Data & Resources**

Download datasets and resources here: [**Google Drive - DSA AI/ML Resources**](https://bit.ly/dsa_ai_ml)

**Files Available:**

* train\_transaction.csv (Raw dataset)
* smote\_balanced\_data.csv (Balanced dataset using SMOTE)
* model.pkl (Trained model)
* scaler.pkl (Feature scaler used in training)
* feature\_columns.pkl (Selected input features)
* Flask app files (app.py, templates/, etc.)

**Note:** Large datasets are not committed to GitHub due to file size limits.

**Setup Instructions (Windows 11 – VS Code)**

1. **Clone the repository:**git clone https://github.com/btolawoyin/dsa\_ai\_ml.git

cd dsa\_ai\_ml

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

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

1. **Create a virtual environment:**

python -m venv venv

venv\Scripts\activate

1. **Install all dependencies:**

pip install -r requirements.txt

**Running the Flask Web App**

To start the application locally: python app.py

Then open your browser and go to: <http://127.0.0.1:5000/>

You’ll see a form to enter transaction data and get a fraud prediction.

**Reflection**

**What Worked**

* SMOTE helped address extreme class imbalance.
* XGBoost produced superior model performance.
* Flask UI allowed intuitive interaction with the ML model.

**What Didn’t**

* Neural Networks took excessive training time.
* GitHub push was blocked due to large files.

**Future Improvements**

* Integrate streaming API for real-time fraud prediction.
* Dockerise and orchestrate with Kubernetes.
* Integrate alert systems for high-risk transactions.

**Tools Used**

* **Programming Language:** Python 3.10
* **Libraries:**
  + Pandas, NumPy, Scikit-learn
  + XGBoost, imblearn
  + Flask, Seaborn, Matplotlib
* **Version Control:** Git, GitHub
* **IDE:** Visual Studio Code on Windows 11

# 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**

**Name:** Bukky Olawoyin  
**GitHub:** [github.com/btolawoyin/dsa\_ai\_ml](https://github.com/btolawoyin/dsa_ai_ml)   
**Project:** DSA Projects - Artificial Intelligence