PROJECT EXECUTION STEPS

The project titled **"UPI Fraud Detection Using Machine Learning"** follows a structured execution process involving data collection, preprocessing, machine learning model training, web development using Flask, integration with MySQL via XAMPP, and deployment through a user-friendly interface. Below are the detailed steps followed during the execution phase of the project.

Step 1: Dataset Collection and Preparation

- Collect a dataset containing transaction records with features
- Clean the dataset:
  - Handle missing/null values.
  - Encode categorical variables (Label Encoding / One-Hot Encoding).
- Split the data into training and testing sets using:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

Step 2: Model Training

- Train the following ML models using the prepared data:
  - Decision Tree
  - Random Forest
  - XGBoost Classifier
  - Gradient Boosting
- Evaluate each model’s performance using metrics:
  - Accuracy, Precision, Recall, F1-score, Confusion Matrix
- Example (Random Forest):

```
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier()
```

```
model.fit(X_train, y_train)
```

Step 3: Web Application Development (Flask)

- Build a web interface using **Flask** with the following pages:
    - Home Page
    - User Registration & Login
    - Dataset Upload
    - Model Selection
    - Prediction Output

Step 4: Backend Web Application (Flask Framework)

- A lightweight web app is developed using **Flask**.
- Functionalities include:
    - **User Registration & Login** with MySQL integration
    - **Dataset Upload** for model input
    - **Model Selection** page for choosing ML algorithms
    - **Prediction Page** to display fraud detection results
- Flask routes and HTML templates manage navigation between:
    - /register, /login, /upload, /select_model, /predict

Step 5: Database Integration Using XAMPP & MySQL

- **XAMPP** is used to run **Apache** and **MySQL** locally.
- **phpMyAdmin** is used to:
    - Create a database named upi_fraud_detection
    - Define tables for user data and logs

Step 6: Real-Time Prediction Flow

- After model training and selection:
  - Users can input transaction details manually or upload new data.
  - The selected model predicts whether the transaction is **Fraudulent (1)** or **Legitimate (0)**.
- Results are displayed directly on the prediction page:
  - ✅ **Transaction Successful: Details Verified and Processed**
  - ❌ **Transaction Failed: Fraudulent Pattern Detected**

Step 7: Testing and Evaluation

- Various types of software testing were performed:
  - **Unit Testing** for individual components
  - **Integration Testing** for combined workflows
  - **Functional Testing** for UI and backend flow
  - **White Box & Black Box Testing** to ensure reliability
  - **Acceptance Testing** to verify against project requirements

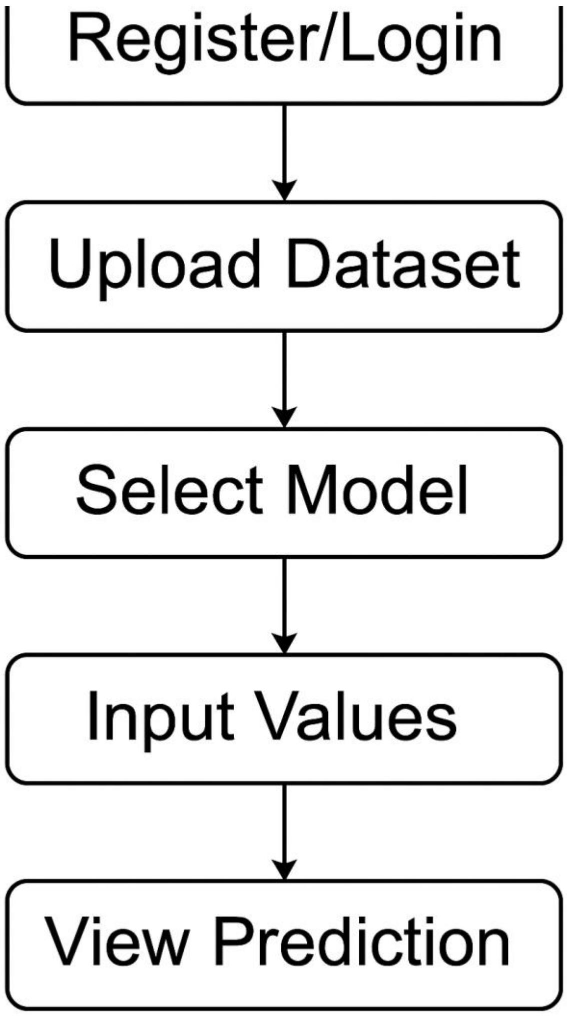
Step 8: Final Deployment and Execution

- Steps to run the complete system:
  1. Launch **XAMPP**, start **Apache** and **MySQL**.
  2. Open **phpMyAdmin** and ensure the database is active.
  3. In terminal, activate the Anaconda environment:

     ```
     conda activate upi_fraud_env
     ```
  4. Run the Flask application:

     ```
     python app.py
     ```
  5. Open a browser and go to:

     http://localhost:5000


Register/Login
Upload Dataset
Select Model
Input Values
View Prediction