Smart Reconciliation and Anomaly Detection System

Problem Statement

In the financial sector, fraudulent or anomalous transactions can lead to financial losses and harm an organization's reputation. Traditional rule-based approaches often fail to detect emerging patterns of fraud. This application aims to:

- Automatically detect anomalies in transactional data.
- Provide meaningful explanations for detected anomalies using Generative AI.
- Allow users to visualize anomalies, sort, and filter results dynamically.
- Generate and send anomaly reports via email for further analysis.

Approach to Solve the Problem

1 Data Ingestion

- Users can upload CSV files via a web interface or stream real-time transaction data.

2 Data Preprocessing

- Handle missing values, normalize data, and encode categorical variables.
- Drop unnecessary fields such as `TransactionID`, `DeviceID`, and `IP Address` to preserve privacy.

3 Anomaly Detection using Isolation Forest

- Isolation Forest detects anomalies by identifying points that are easiest to isolate.
- The algorithm assigns each transaction an `Anomaly` or `Normal` label.

4 Anomaly Explanation using Generative AI (LLM)

- Use a lightweight language model (e.g., `distilgpt2` or `flan-t5-small`) to generate a human-readable explanation.
- Prompt-tuning ensures the model returns concise, one-line explanations.

5 Visualization and Reporting

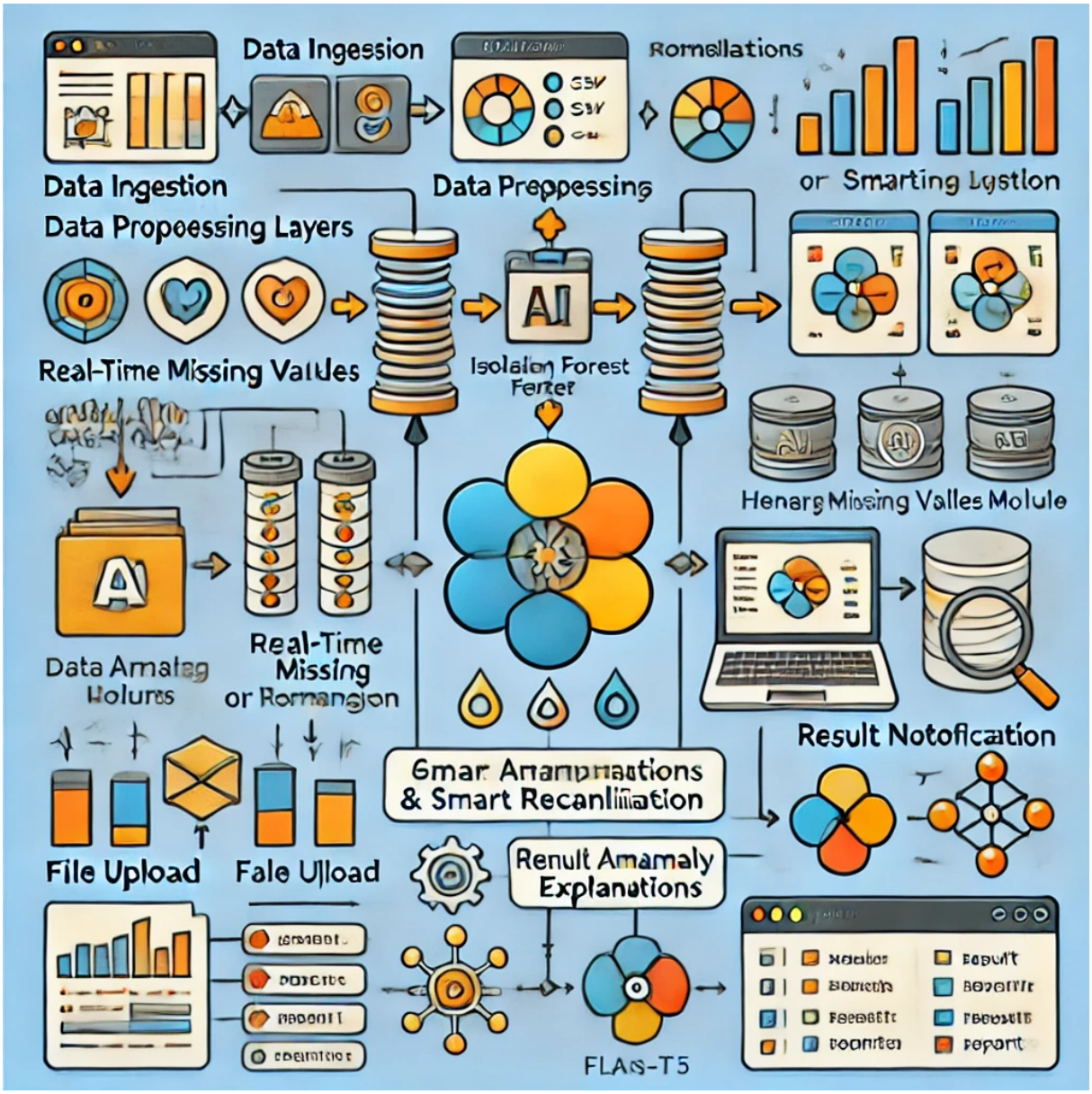
- Flask Web Application displays results with sortable, paginated, and filterable tables.
- Anomaly rows are highlighted for better visibility.
- Anomaly reports are generated and emailed as attachments for detailed review.

Features of the UI

File Upload for Transaction Data (CSV)
Data Preprocessing and Anomaly Detection
Sortable, Paginated, and Searchable Results Table
Anomaly Color Coding (Green for Normal, Red for Anomalies)
Dynamic Updates with Loading Indicator
Anomaly Report Generation and Email Delivery

Architecture Diagram

Below is the architecture diagram of the system:



Steps to Run the Application

1 Clone the Repository

bash git clone https://github.com/your-repo/smart-reconciliation-anomaly-detection.git cd smart-reconciliation-anomaly-detection

2 Set Up a Virtual Environment

```bash
```

Create virtual environment

python3 -m venv env

Activate environment

source env/bin/activate # On Linux/Mac

OR

env\Scripts\activate # On Windows ```

3 Install Required Dependencies

```
bash pip install -r requirements.txt
```

4 Run Flask Application

```
bash python3 run.py
```

5 Access Application in Browser

```
http://127.0.0.1:5000/
```

Run BDD Test Cases

1 Run All Test Cases

```
bash behave
```

2 Run Specific Test Scenario

```
bash behave features/anomaly_detection.feature -n "Detect
anomalies in transactions"
```

3 Generate HTML Test Report

```
bash behave -f html -o report.html
```

Email Configuration

Set up email credentials for sending anomaly reports. Update the following variables in `app/config.py`: `python SMTP_SERVER = 'smtp.example.com' SMTP_PORT = 587 EMAIL_ADDRESS = 'your-email@example.com' EMAIL_PASSWORD = 'your-email-password'`