

Banking System with Advanced Fraud Detection

GitHub: [peedaluk/Banking-system-with-Advanced-Fraud-detection](https://github.com/peedaluk/Banking-system-with-Advanced-Fraud-detection)

1. Project Overview

The Banking System with Advanced Fraud Detection is a secure, web-based application designed to deliver essential banking services while integrating advanced machine learning techniques for fraud detection. The system provides users with a seamless interface for managing accounts and transactions, while backend analytics monitor and flag suspicious activities in real time.

2. Features

- User Registration and Login
- Account Management
- Transaction Processing
- Real-time Fraud Detection
- Secure Password Handling
- User-friendly Web Interface

3. Technologies Used

Component	Technology
Backend	Python (Flask)
Frontend	HTML, CSS, JS
ML Models	Python (scikit-learn, etc.)

Templates	Jinja2
Passwords	Custom module

4. Project Structure

Folder/File	Purpose
<code>app.py</code>	Main Flask application
<code>models/</code>	Machine learning models for fraud detection
<code>routes/</code>	Flask route handlers
<code>templates/</code>	HTML templates for web pages
<code>static/</code>	Static assets (CSS, JS, images)
<code>screenshots/</code>	Project screenshots
<code>password.py</code>	Password management utilities

<code>requirements.txt</code>	Python dependencies
<code>README.md</code>	Project documentation

5. Setup Instructions

1. Clone the repository:
2. `bash`

```
git clone  
https://github.com/peedaluk/Banking-system-with-Advanced-Fraud-d  
etection.git
```

```
cd Banking-system-with-Advanced-Fraud-detection
```

- 3.
4. Install dependencies:
5. `bash`

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

- 6.
7. Run the application:
8. `bash`

```
python app.py
```

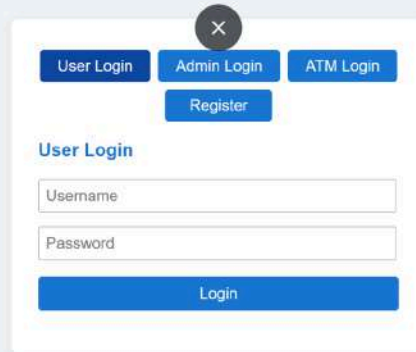
- 9.
10. Access the app:
Open your browser and go to `http://localhost:5000`

6. Screenshots

Login Pages

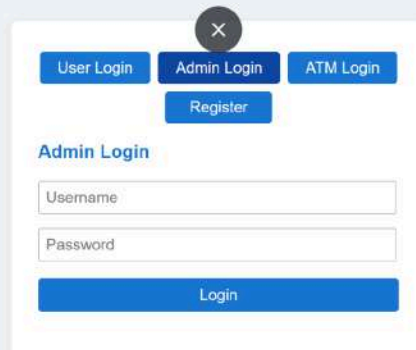
User Login

![[login.png]]



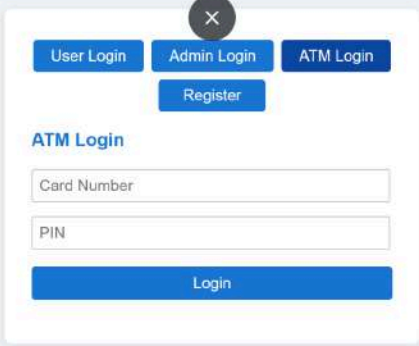
A modal form for user login. At the top is a dark grey circle with a white 'X' for closing. Below it are three blue buttons: 'User Login', 'Admin Login', and 'ATM Login'. Under these is a single blue button labeled 'Register'. The section is titled 'User Login' in blue text. It contains two white input fields: 'Username' and 'Password'. At the bottom is a wide blue button labeled 'Login'.

![[admin_login.png]]



A modal form for admin login. It has the same layout as the user login form, with a close button at the top, followed by 'User Login', 'Admin Login', and 'ATM Login' buttons, and a 'Register' button. The section is titled 'Admin Login' in blue text. It contains two white input fields: 'Username' and 'Password'. At the bottom is a wide blue button labeled 'Login'.

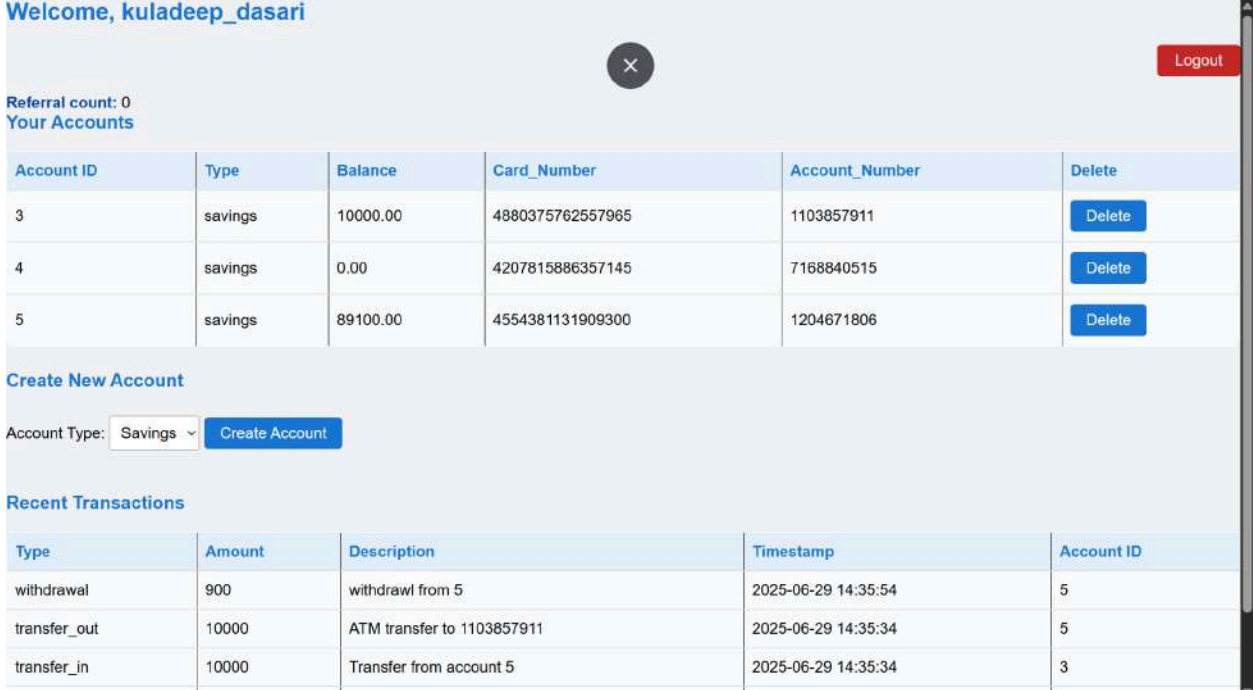
![atm_login.png]



A modal form for ATM login. At the top, there are three buttons: 'User Login', 'Admin Login', and 'ATM Login', with a 'Register' button below them. The 'ATM Login' section contains two input fields: 'Card Number' and 'PIN', followed by a 'Login' button. A close button (X) is at the top center of the modal.

User Dashboard

![user_dashboard.png]



User Dashboard interface for user 'kuladeep_dasari'. It includes a 'Logout' button, a 'Referral count: 0' indicator, and a 'Your Accounts' table with 3 rows. Below is a 'Create New Account' section with a dropdown for 'Savings' and a 'Create Account' button. At the bottom is a 'Recent Transactions' table with 3 rows.

Welcome, kuladeep_dasari

Referral count: 0

Your Accounts

Account ID	Type	Balance	Card_Number	Account_Number	Delete
3	savings	10000.00	4880375762557965	1103857911	Delete
4	savings	0.00	4207815886357145	7168840515	Delete
5	savings	89100.00	4554381131909300	1204671806	Delete

Create New Account

Account Type: Savings Create Account

Recent Transactions

Type	Amount	Description	Timestamp	Account ID
withdrawal	900	withdrawal from 5	2025-06-29 14:35:54	5
transfer_out	10000	ATM transfer to 1103857911	2025-06-29 14:35:34	5
transfer_in	10000	Transfer from account 5	2025-06-29 14:35:34	3

!admin_dashboard.png]

Admin Panel

Dashboard

User Management

Transactions

Analytics

Fraud Detection

Settings

Logout

Admin Dashboard

TOTAL USERS

2

TOTAL BALANCE

\$99100.00

TRANSACTIONS

4

FRAUD ALERTS

0

Recent Transactions

Refresh

ID	Type	Amount	Account	User	Time	Actions
5	WITHDRAWAL	\$900.00	5	4	2025-06-29 14:35:54	<div></div> <div></div>
3	TRANSFER OUT	\$10000.00	5	4	2025-06-29 14:35:34	<div></div> <div></div>
4	TRANSFER IN	\$10000.00	3	4	2025-06-29 14:35:34	<div></div> <div></div>
2	DEPOSIT	\$100000.00	5	4	2025-06-29 14:32:37	<div></div> <div></div>

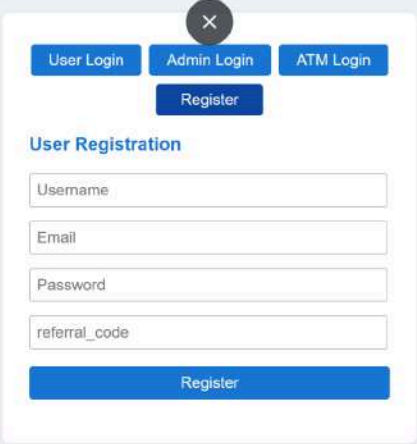
User Management

Search users...

User Management

ID	Username	Email	Joined	Status	Actions
1	admin1	admin1@example.com	2025-06-29	Admin	<div></div>
4	kuladeep_dasari	kuladeepdasari2005@gmail.com	2025-06-29	Active	<div>Block</div> <div></div>

Registration Page
![register.png]



A modal form for user registration is displayed on a light blue background. The modal has a white background and a dark gray close button (X) at the top center. At the top of the modal, there are four blue buttons: "User Login", "Admin Login", "ATM Login", and "Register". Below these buttons, the title "User Registration" is displayed in blue. The form contains four input fields: "Username", "Email", "Password", and "referral_code". At the bottom of the form, there is a large blue "Register" button.

×

User LoginAdmin LoginATM LoginRegister

User Registration

Username

Email

Password

referral_code

Register

![database.png]

```
Windows PowerShell
Tables_in_bank_db
+-----+
| accounts |
| transactions |
| users |
+-----+
9 rows in set (0.095 sec)

mysql> DESCRIBE users;
+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+
| user_id | int | NO | PRI | NULL | auto_increment |
| username | varchar(50) | NO | UNI | NULL | |
| email | varchar(100) | NO | UNI | NULL | |
| password_hash | varchar(200) | NO | | NULL | |
| is_admin | tinyint(1) | YES | | 0 | |
| is_locked | tinyint(1) | YES | | 0 | |
| created_at | timestamp | YES | CURRENT_TIMESTAMP | DEFAULT_GENERATED | |
| referral_code | varchar(20) | YES | UNI | NULL | |
| parent_id | int | YES | MUL | NULL | |
+-----+
9 rows in set (0.057 sec)

mysql> DESCRIBE accounts;
+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+
| account_id | int | NO | PRI | NULL | auto_increment |
| user_id | int | NO | MUL | NULL | |
| account_type | enum('savings','checking') | NO | | NULL | |
| account_number | varchar(20) | NO | UNI | NULL | |
| balance | decimal(15,2) | NO | | 0.00 | |
| card_number | varchar(16) | YES | UNI | NULL | |
| pin_hash | varchar(128) | YES | | NULL | |
| created_at | timestamp | YES | CURRENT_TIMESTAMP | DEFAULT_GENERATED | |
+-----+
8 rows in set (0.036 sec)

mysql> DESCRIBE transactions;
+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+
| transaction_id | int | NO | PRI | NULL | auto_increment |
| account_id | int | NO | MUL | NULL | |
| type | enum('deposit','withdrawal','transfer_in','transfer_out') | NO | | NULL | |
| amount | decimal(12,2) | NO | | NULL | |
| description | varchar(200) | YES | | NULL | |
| related_account_id | int | YES | MUL | NULL | |
| timestamp | timestamp | YES | CURRENT_TIMESTAMP | DEFAULT_GENERATED | |
+-----+
7 rows in set (0.029 sec)

mysql>
```

Account created successfully!
Account Number: 2555045898
Card Number: 41049140813600
PIN: 9701 (Save this securely!)

Recent Transactions

![[screenshot.png]]

7. Fraud Detection Logic

- **Model Training:** Machine learning models are trained on transaction data to distinguish between legitimate and fraudulent activities.
- **Real-time Prediction:** Every transaction is evaluated by the model before processing. If flagged as suspicious, the user is alerted and the transaction may be blocked or require further verification.

Sample Code:

python

app.py (snippet)


```

from models.fraud_model import predict_fraud

@app.route('/transaction', methods=['POST'])
def transaction():
    data = request.form

    is_fraud = predict_fraud(data)

    if is_fraud:
        flash('Fraudulent transaction detected!', 'danger')
        return redirect(url_for('dashboard'))

    # Process transaction if not fraud

```

8. Conclusion & Future Work

This project demonstrates a secure, user-friendly banking system with integrated fraud detection.

Future enhancements could include:

- Transaction visualization dashboards
- Multi-factor authentication
- Integration with real banking APIs

9. References

- [Project GitHub Repository](#)
- Flask Documentation
- Hajdu, L., & Krész, M. (2020). Temporal Network Analytics for Fraud Detection in the Banking Sector. In L. Bellatreche et al. (Eds.), ADBIS/TPDL/EDA 2020 Workshops and Doctoral Consortium, CCIS 1260, pp. 145–157. Springer Nature Switzerland AG. [ResearchGate Link](#)

This paper introduces a temporal network-based methodology for fraud detection

in banking, focusing on detecting cycles in transaction networks, which can reveal hidden fraudulent patterns not easily found with traditional client-centric approaches.

Instructions: