Banking System

July 2025

Project Documentation

Submitted by:

Monica Hany Makram

Under the Supervision of:

Eng. Mohamed Mostafa
Eng. Seif Shaaban

National Bank of Egypt (NBE)

Contents

1 Introduction

The rapid digitization of financial services has emphasized the need for secure, efficient, and user-friendly banking systems. This project, titled **Banking System**, was developed as a comprehensive platform to simulate and manage various banking operations within a web-based environment. The goal was to provide a reliable solution for handling transactions, user management, and financial records, all in a structured and scalable manner.
This system was built in collaboration with the **National Bank of Egypt (NBE)** as part of a practical internship experience, combining both technical development and real-world banking workflows.

2 Project Overview

2.1 Purpose of the System

The main purpose of the Banking System is to offer an interactive platform for managing bank accounts, processing transactions, and uploading financial data via structured Excel files. It also supports deferred (scheduled) transactions that are not executed immediately but instead handled through background jobs on their value date.
This system is intended to serve both bank administrators and customers in managing their financial interactions seamlessly and securely.

2.2 Motivation Behind Building It

This project was inspired by the increasing demand for automation and accuracy in banking operations. Traditional manual processing of transactions can be error-prone and time-consuming. The system aims to reduce this burden by:

- Allowing batch upload of transactions via Excel sheets.
- Providing clear validation and error reporting on transactions.
- Supporting scheduled execution of future-dated transactions.
- Supporting user-friendly interfaces for admins and customers.

2.3 Technologies Used

The following tools and technologies were used in the development of the project:

- **ASP.NET Web Forms (C#)** – for backend server-side logic and session management.
- **Microsoft SQL Server** – for relational database design and secure data storage.
- **EPPlus Library** – for reading and validating uploaded Excel (.xlsx) files.
- **Windows Scheduled Jobs (via SQL flag)** – to automate the execution of future-dated transactions.
- **HTML, CSS, JavaScript** – for styling and interactive UI components.
- **IIS/Visual Studio** – for local testing and deployment.

2.4 Target Users

The system is designed for two primary categories of users:

- **Bank Administrators** – who can manage user accounts, validate transactions, and monitor financial activity.
- **Customers** – who can register, view their accounts, perform transactions, and review their transaction history.

3 Problem Statement

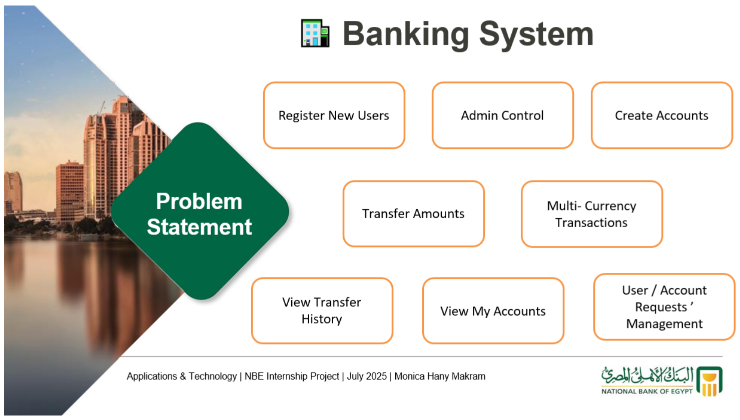


Figure 1: Presentation Slide Highlighting the Problem Statement

In the modern financial environment, the management of banking operations demands precision, speed, and security. Manual handling of banking tasks such as money transfers, account updates, and transaction tracking often leads to inefficiencies, data inconsistencies, and security vulnerabilities.

This project addresses the following key challenges:

- **Lack of automation:** Manual data entry for large batches of transactions increases the risk of human error.
- **Delayed processing:** No support for scheduled or future-dated transactions in traditional systems.
- **Data validation:** No standardized process to validate transaction files before processing them.
- **Administrative burden:** Admins often lack tools to quickly filter, correct, and execute or reject transactions efficiently.
- **No audit trail:** Inability to trace transaction status and reasons for rejection.

Why a Digital Platform is Needed

A secure and centralized digital banking system helps overcome these limitations by:

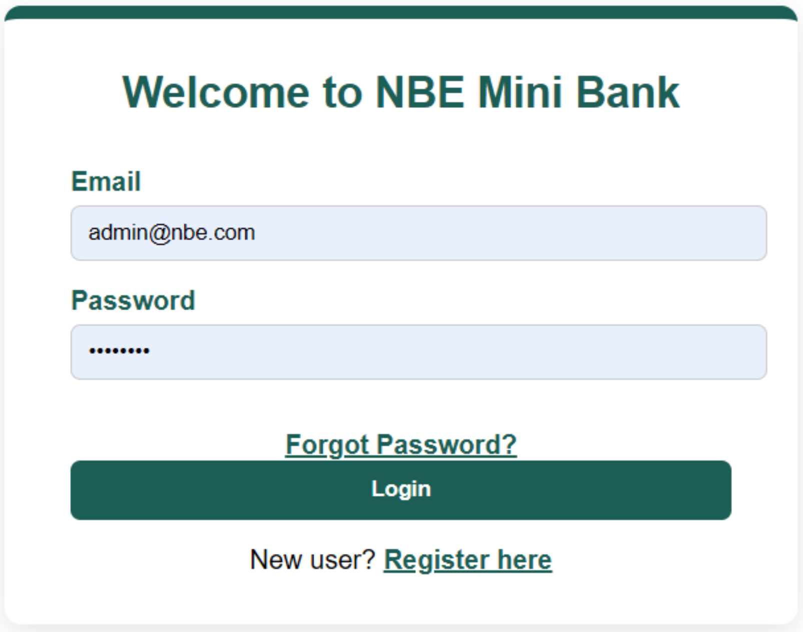
- Enabling safe, fast, and validated upload of transaction files.
- Automatically executing valid transactions and queuing future ones using scheduled jobs.
- Providing clear UI and reporting for admins to monitor and manage all activities.
- Ensuring compliance with modern security and validation standards.

4 System Features

The Banking System includes a comprehensive set of features designed to facilitate secure, efficient digital banking. The system supports both customer-facing operations and admin-level controls.

4.1 Register & Authentication

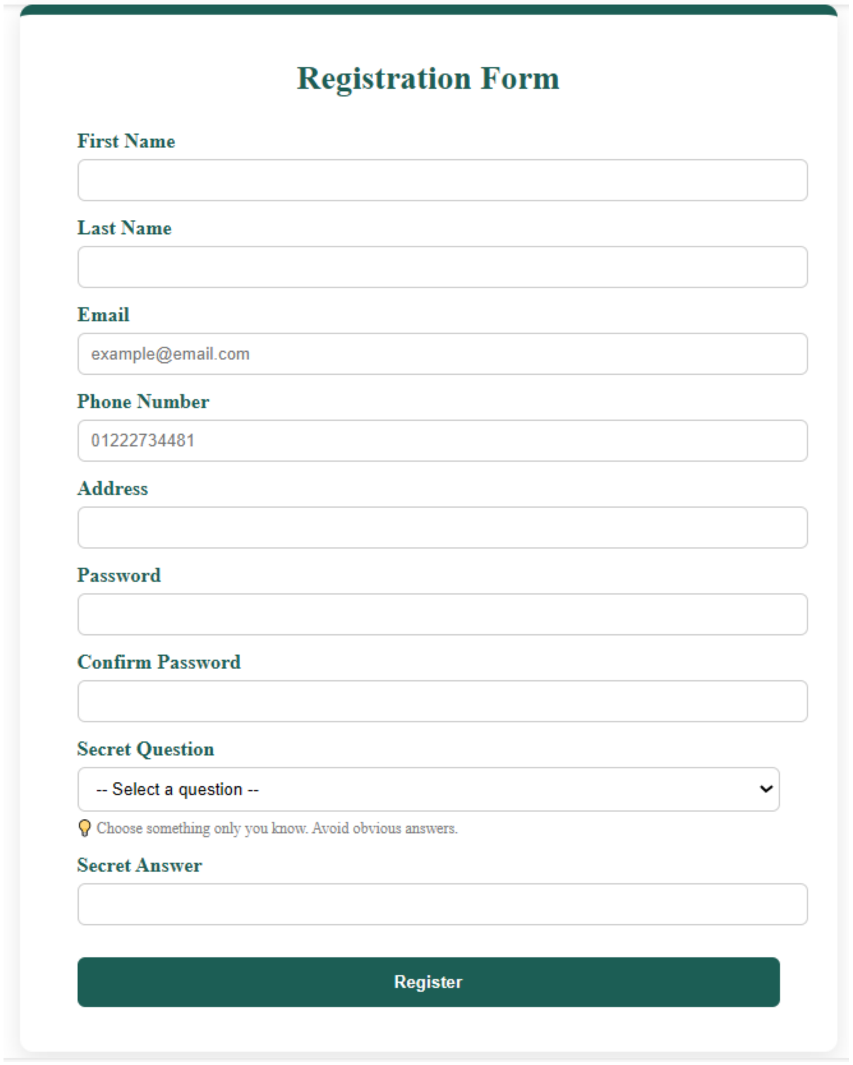
Users can create new accounts, log in using secure credentials, and recover forgotten passwords. Validation and error handling are in place to ensure account integrity.


Welcome to NBE Mini Bank

Email: admin@nbe.com

Password: ••••••••

Figure 2: Login Interfaces


Registration Form

First Name

Last Name

Email

example@email.com

Phone Number

01222734481

Address

Password

Confirm Password

Secret Question

-- Select a question --

💡 Choose something only you know. Avoid obvious answers.

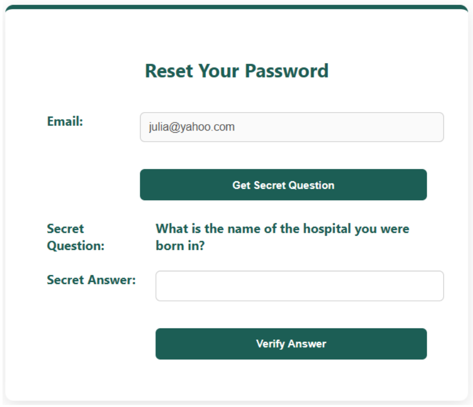
Secret Answer

Register

Figure 3: Registration Interfaces

4.2 Password Recovery

The system provides a secure password recovery mechanism based on secret questions and answers. When registering, users are required to select a predefined question and provide a corresponding answer.
If a user forgets their password, they must correctly answer the chosen secret question to verify their identity before being allowed to reset the password. This method ensures a lightweight and secure recovery process without relying on external email or OTP services.

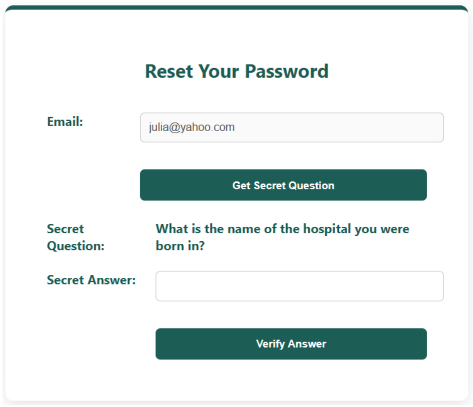

Reset Your Password

Email: julia@yahoo.com

Secret Question: What is the name of the hospital you were born in?

Secret Answer:

Figure 4: Forget Password Page


Reset Your Password

Email: julia@yahoo.com

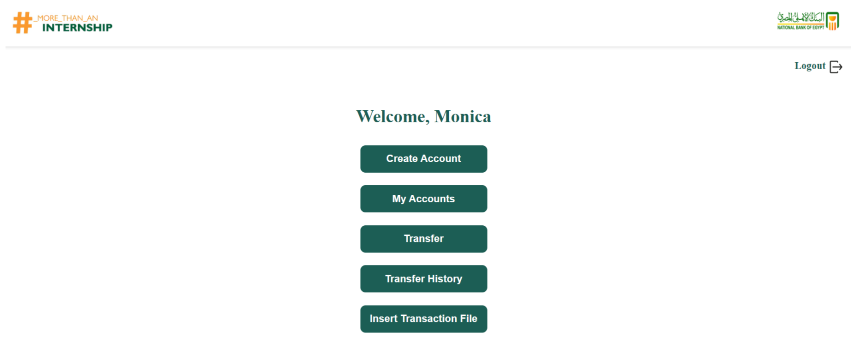
Secret Question: What is the name of the hospital you were born in?

Secret Answer:

Figure 5: Password Reset Successful

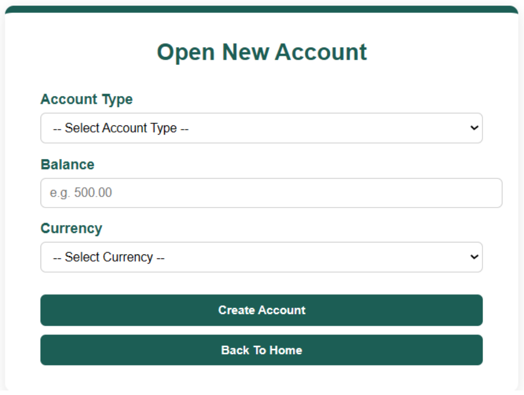
4.3 User Account Management

Once logged in, users can access their account details, including balances and transaction history. This allows users to monitor financial activity in real time.



Welcome, Monica

Figure 6: User Dashboard



Open New Account

Account Type

-- Select Account Type --

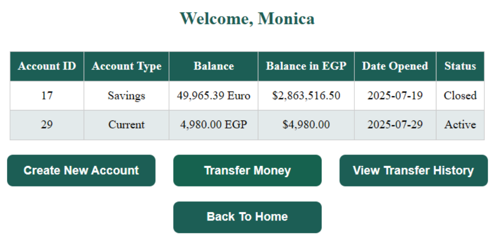
Balance

e.g. 500.00

Currency

-- Select Currency --

Figure 7: Create Account Page



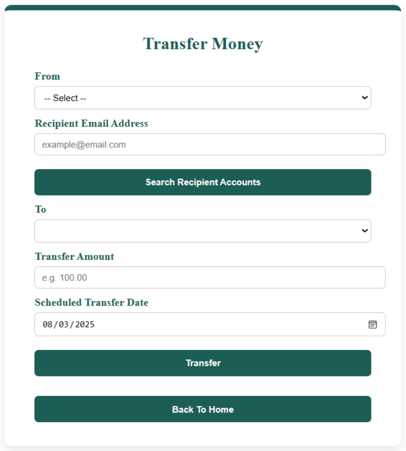
Welcome, Monica

Account ID	Account Type	Balance	Balance in EGP	Date Opened	Status
17	Savings	49,965.39 Euro	$2,863,516.50	2025-07-19	Closed
29	Current	4,980.00 EGP	$4,980.00	2025-07-29	Active

Figure 8: View My Accounts

4.4 Transaction Processing

Users can initiate secure transfers between accounts. The system supports multi-currency transactions and includes validation to prevent errors (e.g., negative balances, inactive accounts).



Transfer Money

From: -- Select --

Recipient Email Address: example@email.com

Search Recipient Accounts

To:

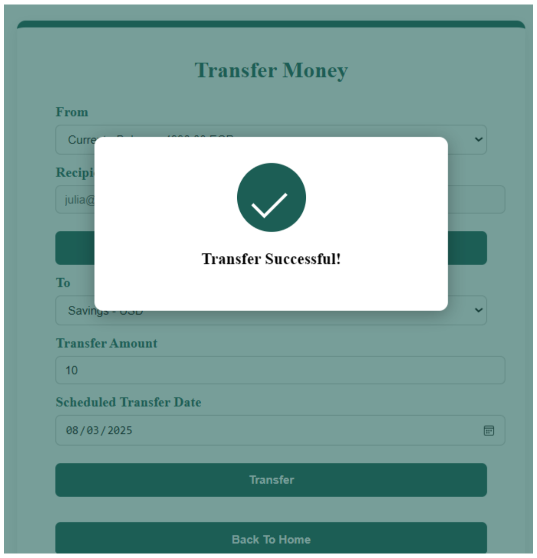
Transfer Amount: e.g. 100.00

Scheduled Transfer Date: 08/03/2025

Transfer

Back To Home

Figure 9: Transaction Page with Multi Currency Support



Transfer Money

From: Curre

Recipi: julia@

To: Savings - USD

Transfer Amount: 10

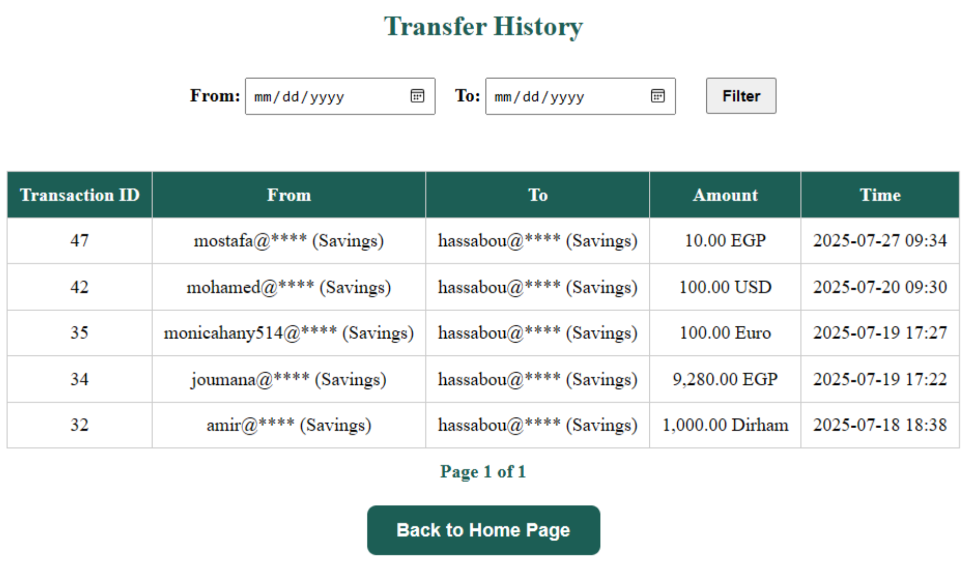
Scheduled Transfer Date: 08/03/2025

Transfer

Back To Home

Transfer Successful!

Figure 10: Successful Transaction



Transaction ID	From	To	Amount	Time
47	mostafa@**** (Savings)	hassabou@**** (Savings)	10.00 EGP	2025-07-27 09:34
42	mohamed@**** (Savings)	hassabou@**** (Savings)	100.00 USD	2025-07-20 09:30
35	monicahany514@**** (Savings)	hassabou@**** (Savings)	100.00 Euro	2025-07-19 17:27
34	joumana@**** (Savings)	hassabou@**** (Savings)	9,280.00 EGP	2025-07-19 17:22
32	amir@**** (Savings)	hassabou@**** (Savings)	1,000.00 Dirham	2025-07-18 18:38

Figure 11: View Transfer History

4.5 File Upload & Validation

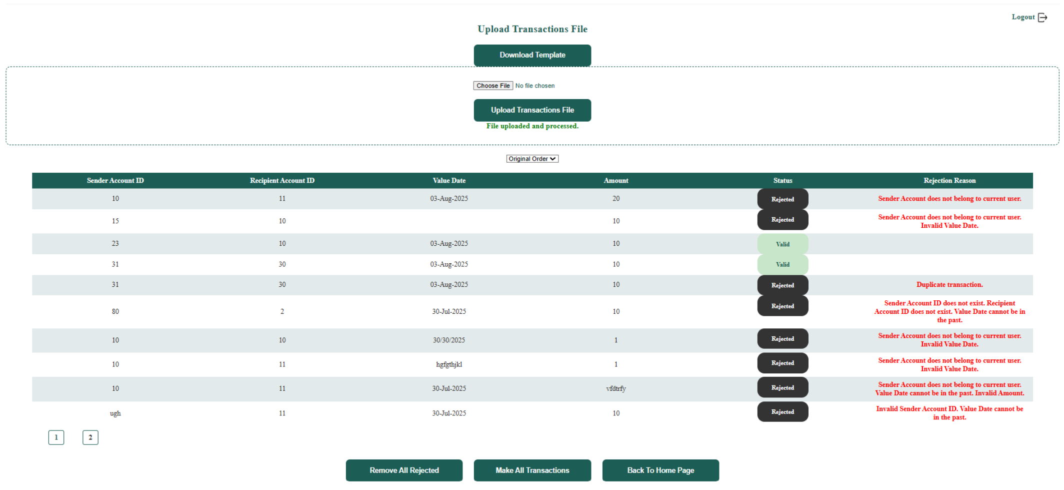
The system allows bulk transaction uploads via an Excel file (.xlsx). To begin, users can download a preformatted template, fill in the transaction details, and upload the completed sheet back into the system.

Each transaction row is validated based on a set of business rules, such as:

- Proper date formatting and ensuring future-dated transactions.
- Valid and existing sender/recipient account IDs.
- Positive transaction amount.

Invalid rows are flagged with a clear rejection reason and are displayed in a dynamic GridView for user review. Users can click **"Remove All Rejected"** to filter out invalid transactions, or **"Make All Transactions"** to commit all valid entries.

Additionally, users can sort or filter transactions by their status (e.g., showing rejected entries first). Upon clicking **"Make All Transactions"**, a bulk insert into the database is triggered, with a revalidation step to ensure data consistency at the point of commit.

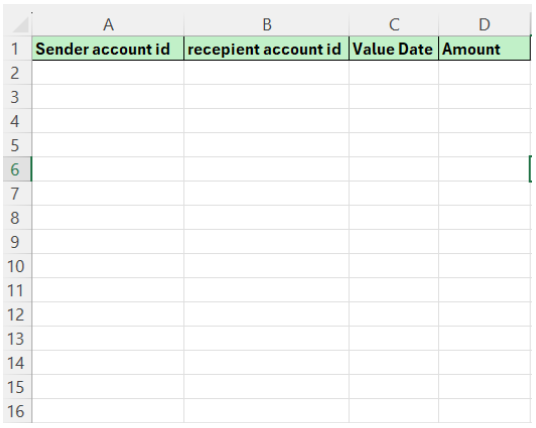


File uploaded and processed.

Original Order

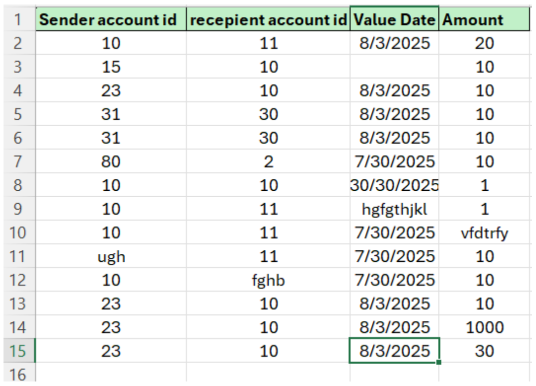
Sender Account ID	Recipient Account ID	Value Date	Amount	Status	Rejection Reason
10	11	03-Aug-2025	20	Rejected	Sender Account does not belong to current user.
15	10		10	Rejected	Sender Account does not belong to current user. Invalid Value Date.
23	10	03-Aug-2025	10	Valid	
31	30	03-Aug-2025	10	Valid	
31	30	03-Aug-2025	10	Rejected	Duplicate transaction.
80	2	30-Jul-2025	10	Rejected	Sender Account ID does not exist. Recipient Account ID does not exist. Value Date cannot be in the past.
10	10	30/30/2025	1	Rejected	Sender Account does not belong to current user. Invalid Value Date.
10	11	hgfgthjkl	1	Rejected	Sender Account does not belong to current user. Invalid Value Date.
10	11	30-Jul-2025	vfdtrfy	Rejected	Sender Account does not belong to current user. Value Date cannot be in the past. Invalid Amount.
ugh	11	30-Jul-2025	10	Rejected	Invalid Sender Account ID. Value Date cannot be in the past.

Figure 12: Excel File Upload and Validation Grid



	A	B	C	D
1	Sender account id	recepient account id	Value Date	Amount
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				

Figure 13: Sample Template downloaded



1	Sender account id	recepient account id	Value Date	Amount
2	10	11	8/3/2025	20
3	15	10		10
4	23	10	8/3/2025	10
5	31	30	8/3/2025	10
6	31	30	8/3/2025	10
7	80	2	7/30/2025	10
8	10	10	30/30/2025	1
9	10	11	hgfgthjkl	1
10	10	11	7/30/2025	vfdtrfy
11	ugh	11	7/30/2025	10
12	10	fghb	7/30/2025	10
13	23	10	8/3/2025	10
14	23	10	8/3/2025	1000
15	23	10	8/3/2025	30
16				

Figure 14: Sample Template Uploaded

4.6 Admin Control

Admins have the ability to create, modify, and deactivate user accounts. Additionally, admins can view all transactions, approve pending requests, and audit system activity.

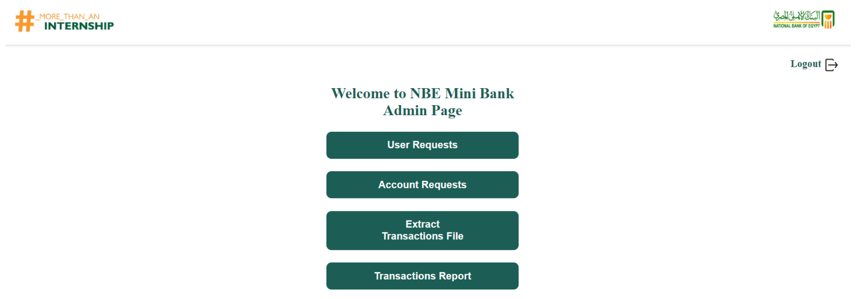
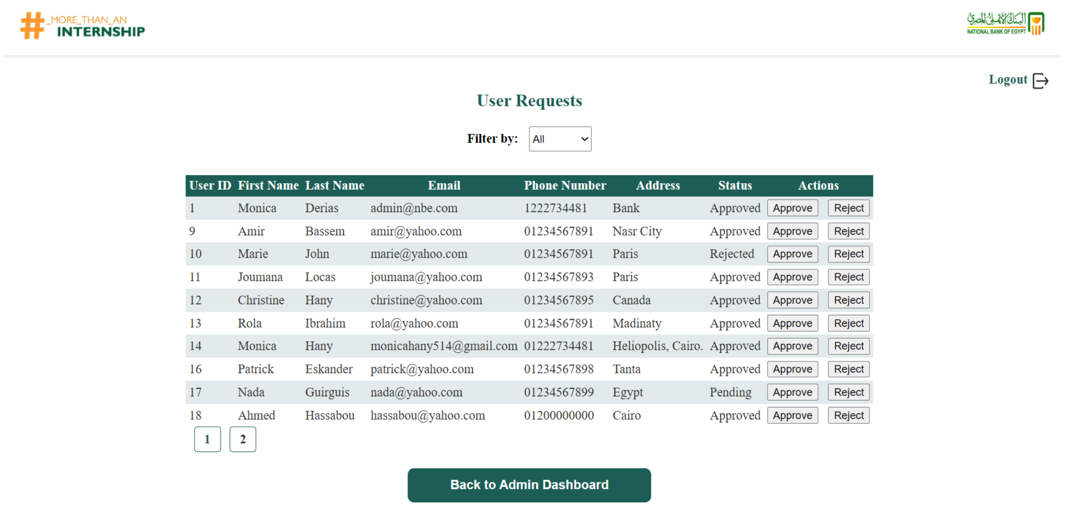
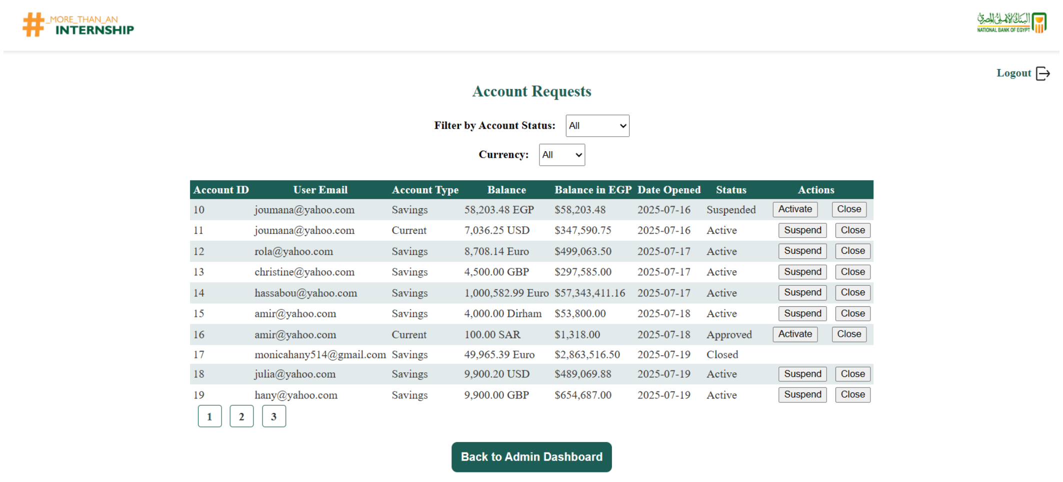


Figure 15: Admin Control Panel



User ID	First Name	Last Name	Email	Phone Number	Address	Status	Actions
1	Monica	Derias	admin@nbe.com	1222734481	Bank	Approved	Approve Reject
9	Amir	Bassem	amir@yahoo.com	01234567891	Nasr City	Approved	Approve Reject
10	Marie	John	marie@yahoo.com	01234567891	Paris	Rejected	Approve Reject
11	Joumana	Locas	joumana@yahoo.com	01234567893	Paris	Approved	Approve Reject
12	Christine	Hany	christine@yahoo.com	01234567895	Canada	Approved	Approve Reject
13	Rola	Ibrahim	rola@yahoo.com	01234567891	Madinaty	Approved	Approve Reject
14	Monica	Hany	monicahany514@gmail.com	01222734481	Heliopolis, Cairo.	Approved	Approve Reject
16	Patrick	Eskander	patrick@yahoo.com	01234567898	Tanta	Approved	Approve Reject
17	Nada	Guirguis	nada@yahoo.com	01234567899	Egypt	Pending	Approve Reject
18	Ahmed	Hassabou	hassabou@yahoo.com	01200000000	Cairo	Approved	Approve Reject

Figure 16: User Requests Page



Account Requests

Filter by Account Status: All

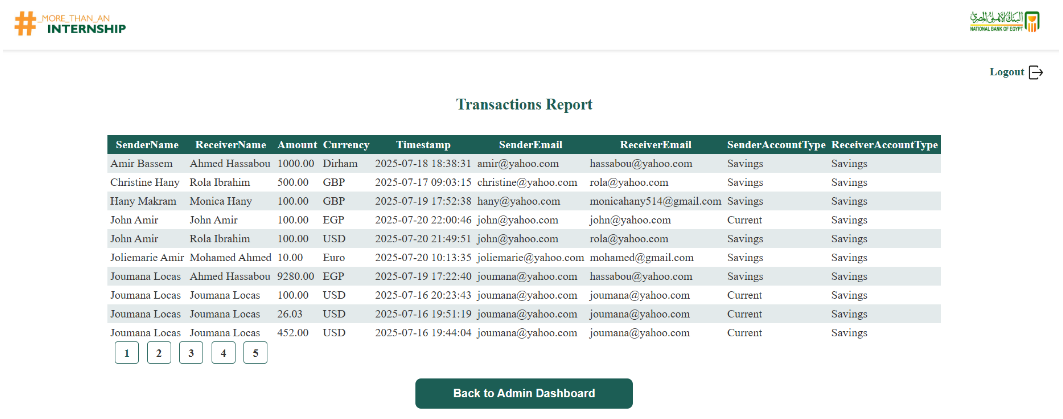
Currency: All

Account ID	User Email	Account Type	Balance	Balance in EGP	Date Opened	Status	Actions
10	joumana@yahoo.com	Savings	58,203.48 EGP	$58,203.48	2025-07-16	Suspended	Activate Close
11	joumana@yahoo.com	Current	7,036.25 USD	$347,590.75	2025-07-16	Active	Suspend Close
12	rola@yahoo.com	Savings	8,708.14 Euro	$499,063.50	2025-07-17	Active	Suspend Close
13	christine@yahoo.com	Savings	4,500.00 GBP	$297,585.00	2025-07-17	Active	Suspend Close
14	hassabou@yahoo.com	Savings	1,000,582.99 Euro	$57,343,411.16	2025-07-17	Active	Suspend Close
15	amir@yahoo.com	Savings	4,000.00 Dirham	$53,800.00	2025-07-18	Active	Suspend Close
16	amir@yahoo.com	Current	100.00 SAR	$1,318.00	2025-07-18	Approved	Activate Close
17	monicahany514@gmail.com	Savings	49,965.39 Euro	$2,863,516.50	2025-07-19	Closed	
18	julia@yahoo.com	Savings	9,900.20 USD	$489,069.88	2025-07-19	Active	Suspend Close
19	hany@yahoo.com	Savings	9,900.00 GBP	$654,687.00	2025-07-19	Active	Suspend Close

Figure 17: Account Requests Page

4.7 Transaction Reporting

This functionality is designed to extract transaction data from the database into an Excel file. The system then reads and parses the Excel data to generate insightful summaries and statistics. The parsed results are displayed in the form of a structured transaction report.



Transactions Report

SenderName	ReceiverName	Amount	Currency	Timestamp	SenderEmail	ReceiverEmail	SenderAccountType	ReceiverAccountType
Amir Bassem	Ahmed Hassabou	1000.00	Dirham	2025-07-18 18:38:31	amir@yahoo.com	hassabou@yahoo.com	Savings	Savings
Christine Hany	Rola Ibrahim	500.00	GBP	2025-07-17 09:03:15	christine@yahoo.com	rola@yahoo.com	Savings	Savings
Hany Makram	Monica Hany	100.00	GBP	2025-07-19 17:52:38	hany@yahoo.com	monicahany514@gmail.com	Savings	Savings
John Amir	John Amir	100.00	EGP	2025-07-20 22:00:46	john@yahoo.com	john@yahoo.com	Current	Savings
John Amir	Rola Ibrahim	100.00	USD	2025-07-20 21:49:51	john@yahoo.com	rola@yahoo.com	Savings	Savings
Joliemarie Amir	Mohamed Ahmed	10.00	Euro	2025-07-20 10:13:35	joliemarie@yahoo.com	mohamed@gmail.com	Savings	Savings
Joumana Locas	Ahmed Hassabou	9280.00	EGP	2025-07-19 17:22:40	joumana@yahoo.com	hassabou@yahoo.com	Savings	Savings
Joumana Locas	Joumana Locas	100.00	USD	2025-07-16 20:23:43	joumana@yahoo.com	joumana@yahoo.com	Current	Savings
Joumana Locas	Joumana Locas	26.03	USD	2025-07-16 19:51:19	joumana@yahoo.com	joumana@yahoo.com	Current	Savings
Joumana Locas	Joumana Locas	452.00	USD	2025-07-16 19:44:04	joumana@yahoo.com	joumana@yahoo.com	Current	Savings

Figure 18: Transactions Report Page

5 Scheduled Transactions & Background Job Handling

One of the key features of our system is the ability to schedule future transactions. If a transaction's value date is later than today's date, it is not processed immediately. Instead, it is stored with a flag indicating that it needs a job to run it later.

To handle these scheduled transactions, we implemented a background job mechanism that runs periodically (e.g., every minute) to check the database for any due transactions. If the value date matches today's date and the transaction is still pending, the job automatically processes it, updates the account balances, and marks the transaction as completed.

This design ensures that:

- Future transactions are not lost or forgotten.
- The system remains efficient by decoupling heavy processing from the user-triggered workflow.
- Time-based transaction logic is centralized and maintainable.

6 Database Design

The system is built on a structured relational database in SQL Server. It ensures consistency, integrity, and flexibility to manage all banking operations including user accounts, transactions, branches, and lookup tables.

Key Components:

- **Users:** Stores user information including login credentials, contact details, and security questions for password recovery. Each user has a role (admin or customer) and a status (active/inactive).
- **Accounts:** Each user can hold multiple accounts. Every account stores balance, opening/closing dates, and is linked to a branch, status, type, and currency.
- **Transactions:** Logs all transfers between accounts. Each transaction includes sender/receiver IDs, timestamp, and amount.
- **Branches:** Represents the different branches where accounts can be held.

- **Lookup Tables:** Normalize repeated information such as:
  - `Look_Currency` – Holds currency names and exchange values.
  - `Look_UserType` – Distinguishes between admin and customer.
  - `Look_AccountType`, `Look_AccountStatus`, `Look_UserStatus` – Define account states and types.
  - `Look_Secret_Questions` – Enables secure password recovery.

The relationships follow proper normalization:

- One-to-many between `Users` and `Accounts`.
- One-to-many between `Accounts` and `Transactions` (as sender and receiver).
- All foreign keys (e.g., currency, account type) point to lookup tables for scalability and standardization.

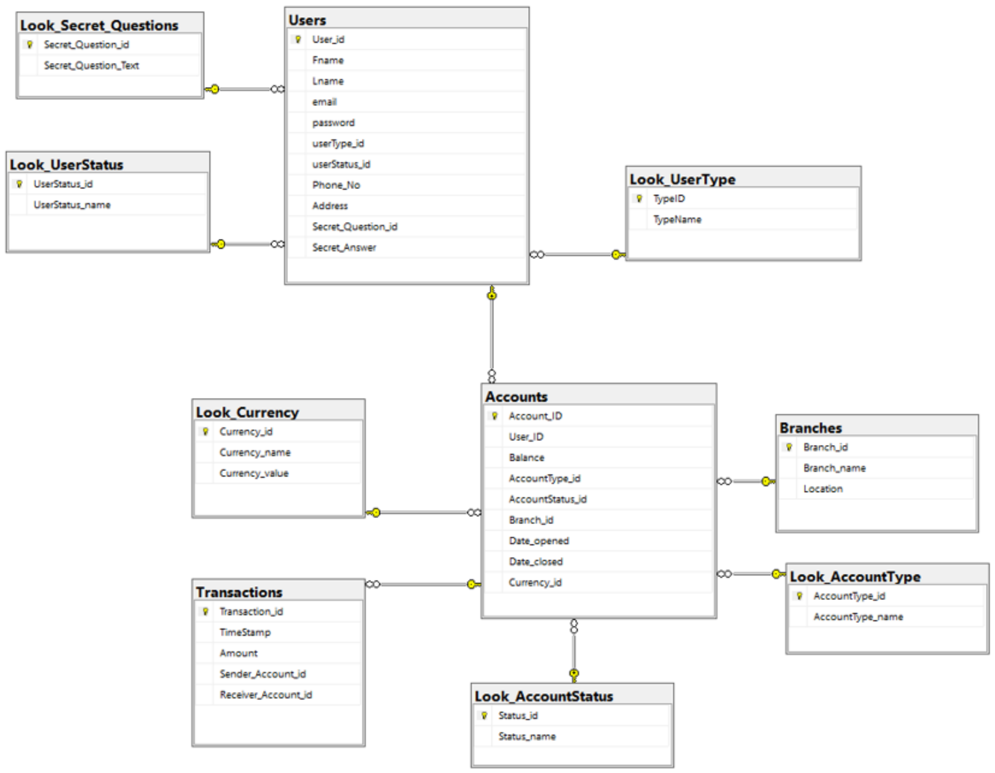


Figure 19: Database Entities of the Banking System

7 Technology Stack

The Banking System was developed using a set of robust and proven technologies, ensuring reliability, maintainability, and performance.

- **Backend:** Developed using **C#** and **ASP.NET WebForms**, providing a structured server-side architecture to handle business logic and data flow.
- **Frontend:** Designed with **HTML** and **CSS** for creating user-friendly and responsive web pages.
- **Database: SQL Server** was used as the relational database management system to store and manage all application data securely.
- **Excel Integration: EPPlus** library enabled reading, validating, and writing Excel files (.xlsx) for bulk transaction upload and reporting.
- **Hosting Environment:** The system was hosted locally using **IIS (Internet Information Services)**, enabling realistic deployment and testing scenarios.

Demo Video and GitHub Repository

You can view a complete walkthrough of the system in the demo video below:

- **Demo Video:** Click here to view the demo
- **GitHub Repository:** https://github.com/Monica-hany/Banking$_{system}$

Conclusion

The Banking System project provided a comprehensive and practical implementation of a core banking platform. From secure user authentication to advanced features like multi-currency transactions, batch file uploads, and administrative controls, the system demonstrates key aspects of modern financial software.

By leveraging technologies such as ASP.NET WebForms, SQL Server, and EPPlus for Excel integration, we were able to build a robust, scalable, and user-friendly application tailored for both bank administrators and customers.

Beyond the core functionalities, the project also involved practical implementation of backend processing concepts such as scheduled jobs, transactional database operations, file parsing, and bulk data insertions. These additions significantly enhanced the system's performance and reliability, particularly when handling large-scale financial data with precision and consistency.

The system also utilized **session management** to securely maintain user-specific data during runtime, such as uploaded transaction files, authenticated states, and role-based access control — enabling a seamless and personalized user experience.

This project not only reinforced our technical skills in backend and frontend development, but also deepened our understanding of critical topics like data integrity, job scheduling, exception handling, and efficient database interaction — all of which are essential in real-world financial systems.

Future improvements could include integrating real-time notifications, RESTful APIs, and full deployment on a cloud platform for scalability and accessibility.