Software Design Specification

for

Umawi Bank

Version 1.0

Prepared by
HAMZAH KOURDIEH

May 28, 2024

Table of Contents

1. Introduction

1.1 Purpose

This document is to present a detailed description of the Umawi Bank. It explains the purpose, features, interfaces of the system, constraints under which it must operate and how the system will react. This document is intended for developers and will be proposed to Dr. Baydogan for his approval.

1.2 Scope of the Development Project

The Umawi Bank software system is designed for educational settings, enabling students to create accounts for submitting their project forms. teachers can create accounts to grade the projects.

1.3 Definitions, acronyms, and abbreviations

Term	Definition
SDS	Software Design Specification.
customer	People use the program to send and receive money.
Admin	Person managing the bank

1.4 References

Dasgupta and D. W. Callahan. (2002). Development of improved tornado tracking device. IEEE 34th Southeastern Symposium on System Theory, pp. 363-365

1.5 Overview of Document

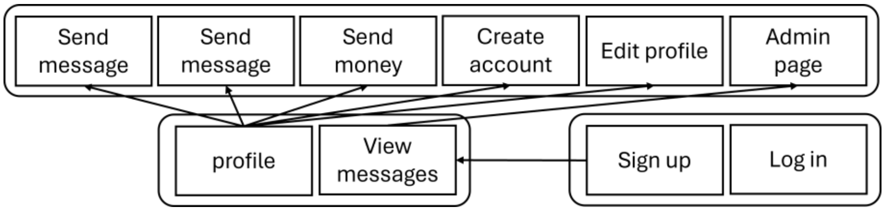
outlines the architecture, functionality, and technical specifications of a software system.

2. System Architecture Description

2.1 System Architecture

Development of the Umawi Bank will involve creating a website using C#. Each change will be logged and synchronized for team collaboration. The development environment will be set up on Windows.

2.1.1 High Level Architecture Diagram

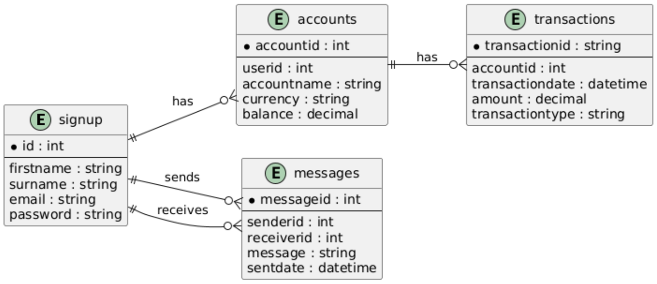


2.1.2 Architecture Narrative:

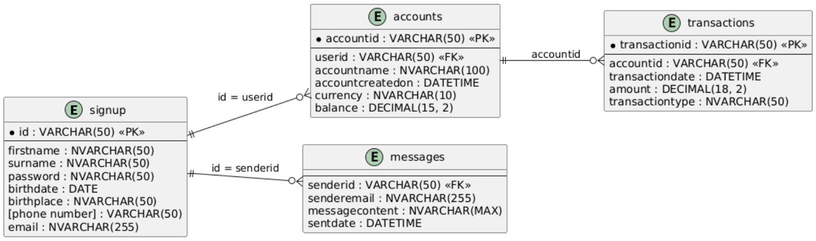
The architecture includes Azure SQL for database needs, and C# for dynamic report generation. DevExpress serves as middleware for design support, Using VS on Windows.

2.2 Database Components

2.2.1 Entity Relationship Diagram (ERD):



2.2.2 Database Table Definitions:


E accounts
• accountid : VARCHAR(50) «PK»
userid : VARCHAR(50) «FK»
accountname : NVARCHAR(100)
accountcreatedon : DATETIME
currency : NVARCHAR(10)
balance : DECIMAL(15, 2)
accountid
E transactions
• transactionid : VARCHAR(50) «PK»
accountid : VARCHAR(50) «FK»
transactiondate : DATETIME
amount : DECIMAL(18, 2)
transactiontype : NVARCHAR(50)
id = userid
E signup
• id : VARCHAR(50) «PK»
firstname : NVARCHAR(50)
surname : NVARCHAR(50)
password : NVARCHAR(50)
birthdate : DATE
birthplace : NVARCHAR(50)
[phone number] : VARCHAR(50)
email : NVARCHAR(255)
id = senderid
E messages
senderid : VARCHAR(50) «FK»
senderemail : NVARCHAR(255)
messagecontent : NVARCHAR(MAX)
sentdate : DATETIME

2.3 GUI Components with Layouts and Navigation

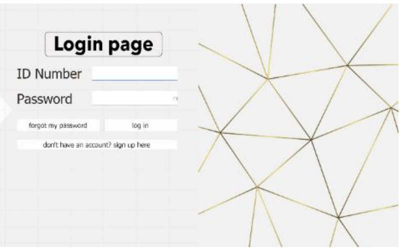
2.3.1 User Interface Issues

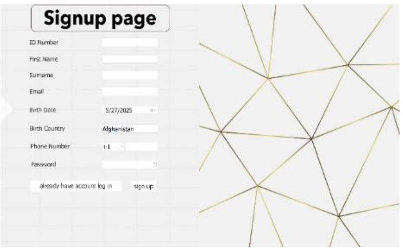
The PGS is a website designed with simplicity in mind with simple GUI interfaces.

2.3.2 List of User Interface Screens and Reports

Contact, Login, sign up, transfer money, create bank statement / transaction history, create account.

2.3.3 User Interface Screens or Reports (Layouts)

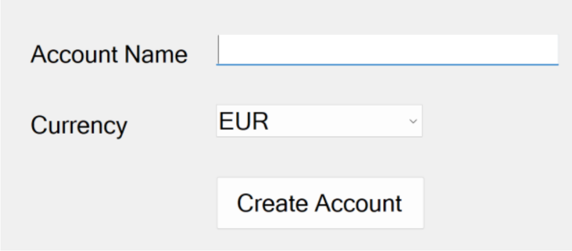




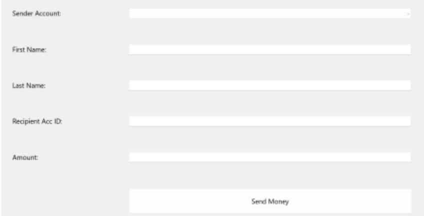
Birth Date 5/27/2025

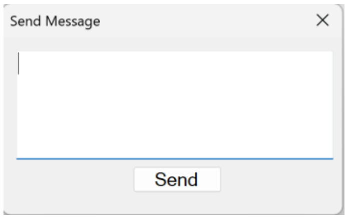
Birth Country Afghanistan

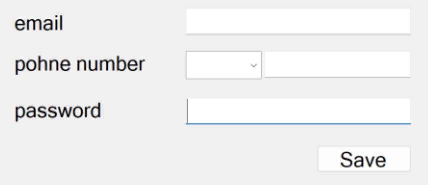
Phone Number +1

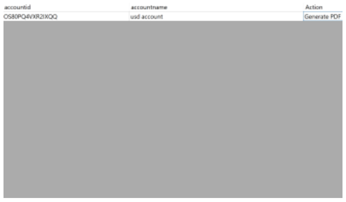


Currency EUR





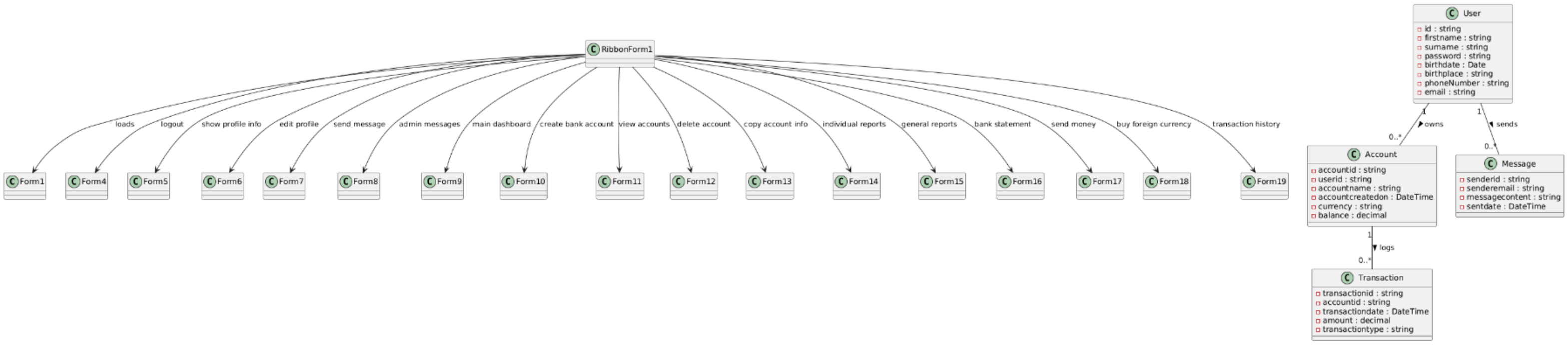




accountid	accountname	Action
OS80PQ4VXR2IXQQ	usd account	Generate PDF

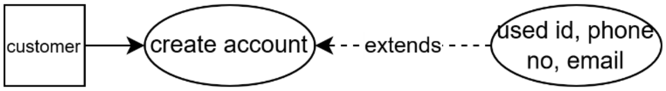
3. Detailed Description of System Components

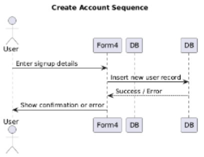
3.1 Class Diagram



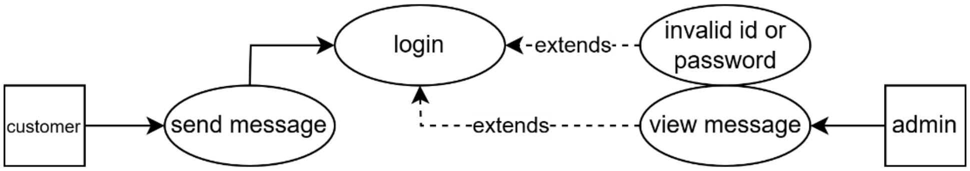
3.2 Use Cases with Sequence Diagrams

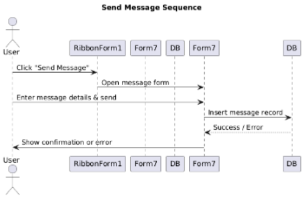
3.1.1 Sequence Diagram for Use Case 1



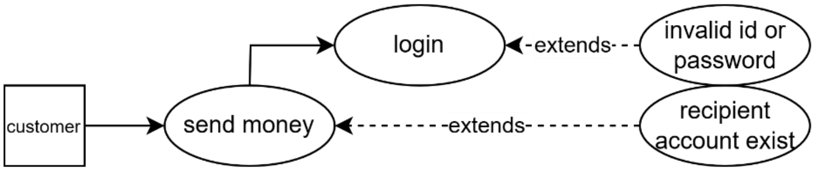


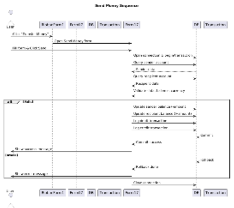
3.1.2 Sequence Diagram for Use Case 2



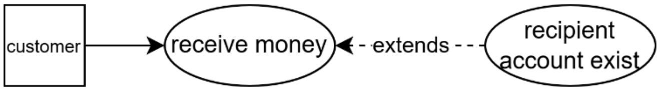


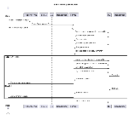
3.1.3 Sequence Diagram for Use Case 3



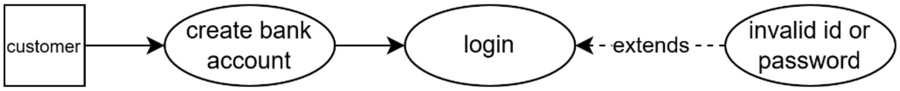


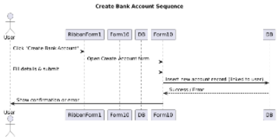
3.1.4 Sequence Diagram for Use Case 4



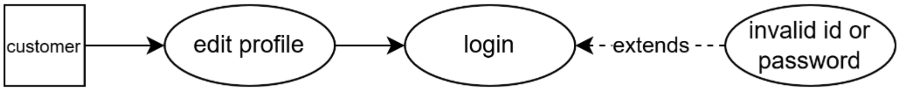


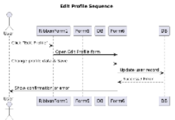
3.1.5 Sequence Diagram for Use Case 5



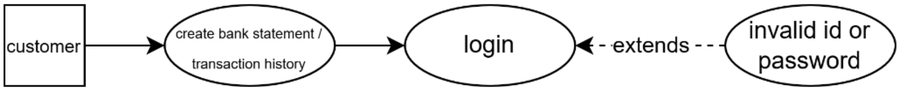

Create Bank Account Sequence
User
RibbonForm1
Form10
DB
Form10
DB
Click "Create Bank Account"
Open Create Account form
Fill details & submit
Insert new account record (linked to user)
Success / Error
Show confirmation or error
User
RibbonForm1
Form10
DB
Form10
DB

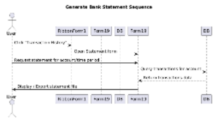
3.1.6 Sequence Diagram for Use Case 6



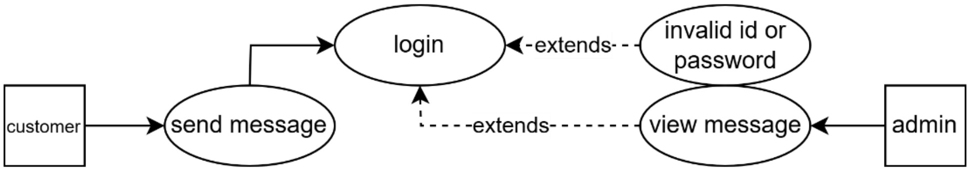


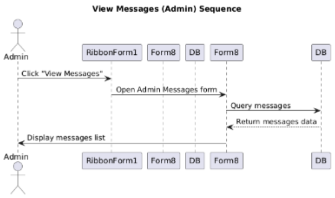
3.1.7 Sequence Diagram for Use Case 7




Generate Bank Statement Sequence
User
RibbonForm1
Form19
DB
Form19
DB
Click "Transaction History"
Open Statement form
Request statement for account/time period
Query transactions for account
Return transactions data
Display / Export statement file
User
RibbonForm1
Form19
DB
Form19
DB

3.1.8 Sequence Diagram for Use Case 7





4. Design Decisions and Tradeoffs

4.1 Desktop App

We chose to develop the Umawi Bank as a website using C#.

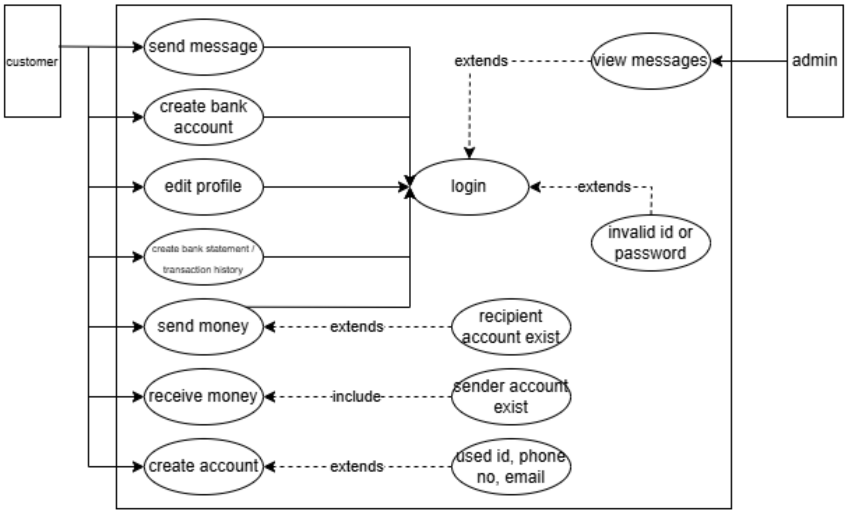
4.2 Feature Selection

Initially, we considered various features for Umawi Bank. We decided to focus on core functionalities: allowing customers to send and receive money.

4.3 Modular Design

We adopted a modular design for Umawi Bank making each feature a separate module.

Appendix A: Use-Case Diagram



Appendix B: Data Dictionary

B1: Use-Case Description

1. Customer Use Cases: The customer has the following sets of use cases:

1. Create Account:

Brief Description / Goal: The "Create Account" functionality enables customers to register for a new account on the Umawi Bank website.

Pre-conditions:

1. The customer must have a computer connected to the internet with email and phone number.

Initial Step-By-Step Description / Scenario:

Accessing Registration Page: The customer navigates to the "sign up" page.

Providing User Information: The customer fills out the registration form, providing necessary details such as Full Name, Email Address, Password, and Contact Information**.**

Submitting Registration Form: The customer clicks on the "sign up" button.

Confirmation: Upon successful registration, the system confirms the creation of the account.

Post-conditions:

1. The customer's account information is securely stored in the Umawi Bank database, and they receive confirmation of successful registration.
2. The customers can log in using their provided credentials.

Exceptions:

1. Used id number, phone number or email: If the id no, phone number or email address is already used, the customer is prompted to provide a different username or email address.

2. **Send Message:**

Brief Description / Goal: Allows customers to send messages or inquiries to the bank's support team.

Pre-conditions:

1. Customers must be logged in.

Initial Step-By-Step Description / Scenario:

1. Navigating to the "Contact" section and click on "message" button.
2. Composing a message and click on "send" button.

Post-conditions:

1. Message is delivered to bank support.

3. **Send Money:**

Brief Description / Goal: Allows customers to transfer funds to other accounts.

Pre-conditions:

1. Customers must be logged in and have sufficient balance.
2. Recipient account information is available.
3. Or the customer should have two accounts to exchange currency between them.

Initial Step-By-Step Description / Scenario:

1. Accessing "transactions" section, "send money button".
2. Entering recipient details and amount.
3. Confirming and submitting transaction.

Post-conditions:

1. Funds are transferred successfully.
2. Transaction details are recorded.

4. **Receive Money:**

Brief Description / Goal: Allows customers to receive funds transferred from other accounts.

Pre-conditions:

1. Customer must have an active account.

Initial Step-By-Step Description / Scenario:

1. Viewing transaction history to confirm receipt.

Post-conditions:

1. Funds are reflected in the customer's account balance.

4. **Create Bank Account:**

Brief Description / Goal: Allows customers to create new bank accounts (e.g., my account, USD).

Pre-conditions:

1. Customer must be logged in.

Initial Step-By-Step Description / Scenario:

1. Navigating to "account management" section.
2. Selecting account currency and providing necessary information.
3. Clicking on “create account” button.

Post-conditions:

1. A new bank account is being created and linked to the customer’s main profile.

5. **Edit Profile:**

Brief Description / Goal: Enables customers to update their personal information.

Pre-conditions:

1. Customer must be logged in.

Initial Step-By-Step Description / Scenario:

1. Accessing "Profile" page then “edit profile” button.
2. Updating details such as password, phone number, email.
3. Saving changes by clicking on “save” button.

Post-conditions:

1. Updated profile information is saved securely.

7. View Bank Statement / Transaction History:

Brief Description / Goal: Allows customers to view their transaction history and bank statements.

Pre-conditions:

1. Customer must be logged in.

Initial Step-By-Step Description / Scenario:

1. Navigating to the “Reports" section.
2. Click on “transaction history” or “bank statement”.
3. Select the account you would the document for it and click on “generate” button.

Post-conditions:

1. Report will be saved as a pdf file.

2. Admin Use-Cases: The Admin has the following use cases:

1. View Messages:

Brief Description / Goal: Allows the admin to view messages sent by customers.

Pre-conditions:

1. Admin must be logged in with appropriate privileges.

Initial Step-By-Step Description / Scenario:

1. Navigating to the “contact” section, "Messages" button.

2. Reviewing customer messages.

Post-conditions:

1. Admin manages customer communications efficiently.

B2: Actor Descriptions

Customer: People use the program to send and receive money.

Admin: Person managing the bank

B3: Class Descriptions

Form1: Main login form handling user authentication and storing logged-in user ID.

Form2: main sign up form, user registers his information into database.

Form4: User registration/signup form for creating new user accounts.

Form5: Displays the logged-in user's profile information.

Form6: Allows the logged-in user to edit and update their profile details.

Form7: User interface for sending messages to other users.

Form8: Admin message management form for viewing and handling all messages.

Form9: Default MDI child form loaded on app start, showing dashboard or welcome info.

Form10: Form for creating new bank accounts (e.g., USD account) linked to a user.

Form11: Displays a list of user's bank accounts.

Form12: Interface for deleting existing user bank accounts.

Form13: Form to copy or display detailed account information.

Form14: Form handling individual (personal) account operations or info.

Form15: Form handling general account operations or info.

Form16: Generates and displays bank statements.

Form17: Manages sending money between accounts with validation and transaction logging.

Form18: Handles currency exchange and transferring money between accounts with different currencies.

Form19: Displays transaction history or allows exporting bank statement files.

RibbonForm1: Main application container with DevExpress Ribbon UI, launching child forms.

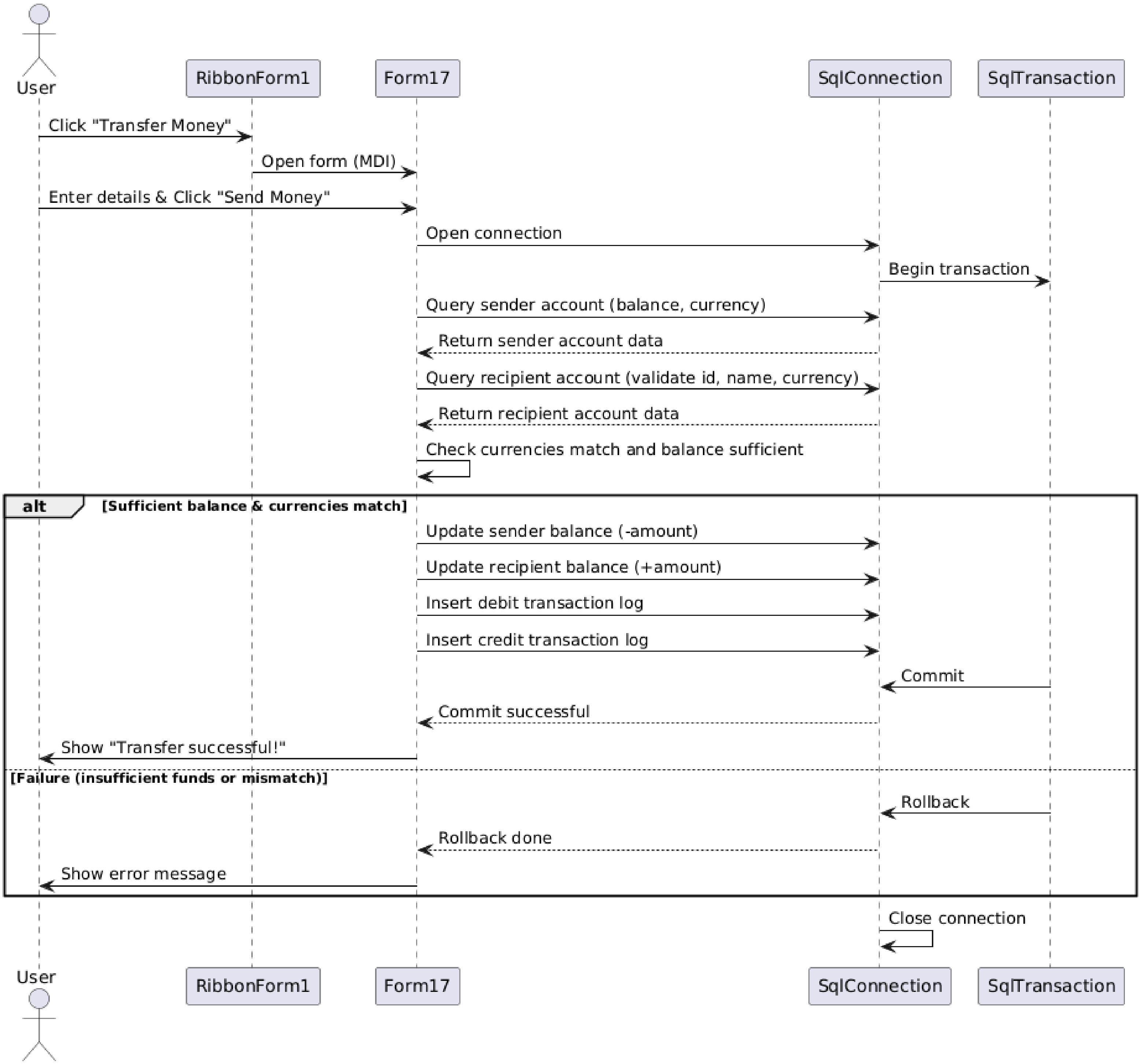
Appendix C: List of Inputs and Outputs

Inputs: Customers enter personal details (name, email, phone, password) during registration and login credentials for authentication. For transactions, they provide recipient info and amounts. Customers select account types and currencies when creating new accounts and update personal info in their profiles. Messages to support are also input by customers. Admins log in and access customer messages.

Outputs: The system confirms successful registrations, logins, transactions, account creations, and profile updates. It displays error messages for invalid actions. Transaction confirmations and updated balances are shown to customers. Admins receive notifications of new messages. Bank statements and transaction reports are provided as downloadable PDFs.

Appendix D: Sequence Diagram

Money Transfer Sequence


User
RibbonForm1
Form17
SqlConnection
SqlTransaction
Click "Transfer Money"
Open form (MDI)
Enter details & Click "Send Money"
Open connection
Begin transaction
Query sender account (balance, currency)
Return sender account data
Query recipient account (validate id, name, currency)
Return recipient account data
Check currencies match and balance sufficient
alt
[Sufficient balance & currencies match]
Update sender balance (-amount)
Update recipient balance (+amount)
Insert debit transaction log
Insert credit transaction log
Commit
Commit successful
Show "Transfer successful!"
[Failure (insufficient funds or mismatch)]
Rollback
Rollback done
Show error message
Close connection
User
RibbonForm1
Form17
SqlConnection
SqlTransaction