OOAD with Java (UE20CS352)
Project Report

Banking System

Submitted By:
K Y Pradyumna Hathwar (PES1UG20CS554)
Rajat Rayaraddi (PES1UG20CS578)
Sada Kakarla (PES1UG20CS586)
Tanmai N (PES1UG20CS601)

GitHub Repository: https://github.com/RajatRayaraddi24/Banking-System-Project-UE20CS352

Problem Statement

Abstract:
The banking system app is a Java-based web application that allows customers to perform financial transactions, manage their accounts, and access banking services online. The app provides features such as user authentication, account management, transaction history.

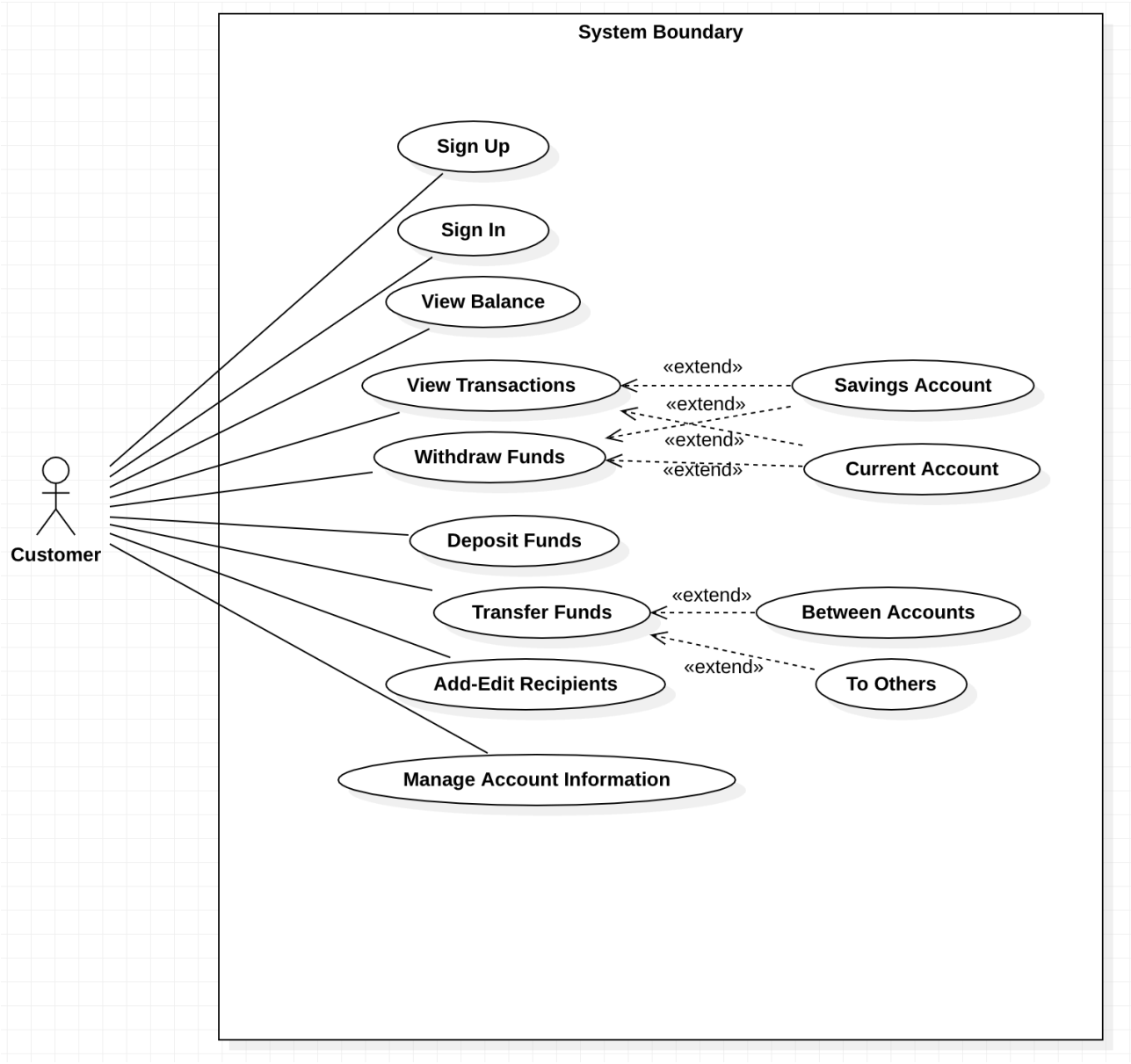
Introduction:
Customers can create new accounts and login to the app using their credentials. They can view their account balance, deposit or withdraw money, transfer funds, and manage their account information. The app also provides a transaction history that allows customers to view their past transactions.

Overall, the banking system app provides a convenient and secure way for customers to manage their finances online while allowing the system to ensure security and reliability.

Tools and Platforms Used:
- Java SDK (17)
- Spring Boot Framework
- Spring Data
- Spring Security
- Hibernate
- MySQL
- Microsoft Visual Studio Code
- Eclipse IDE

Models

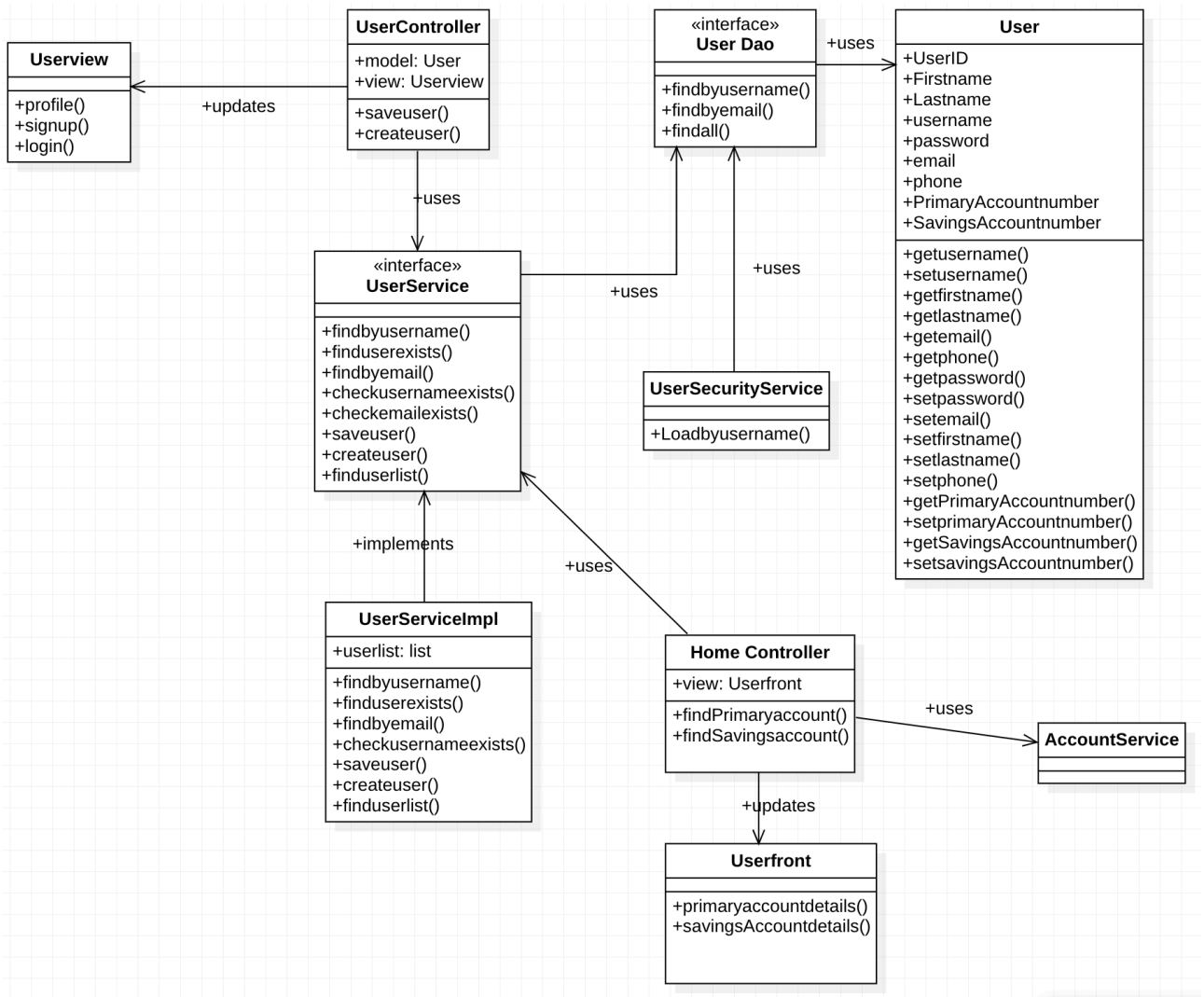
Use Case Diagram

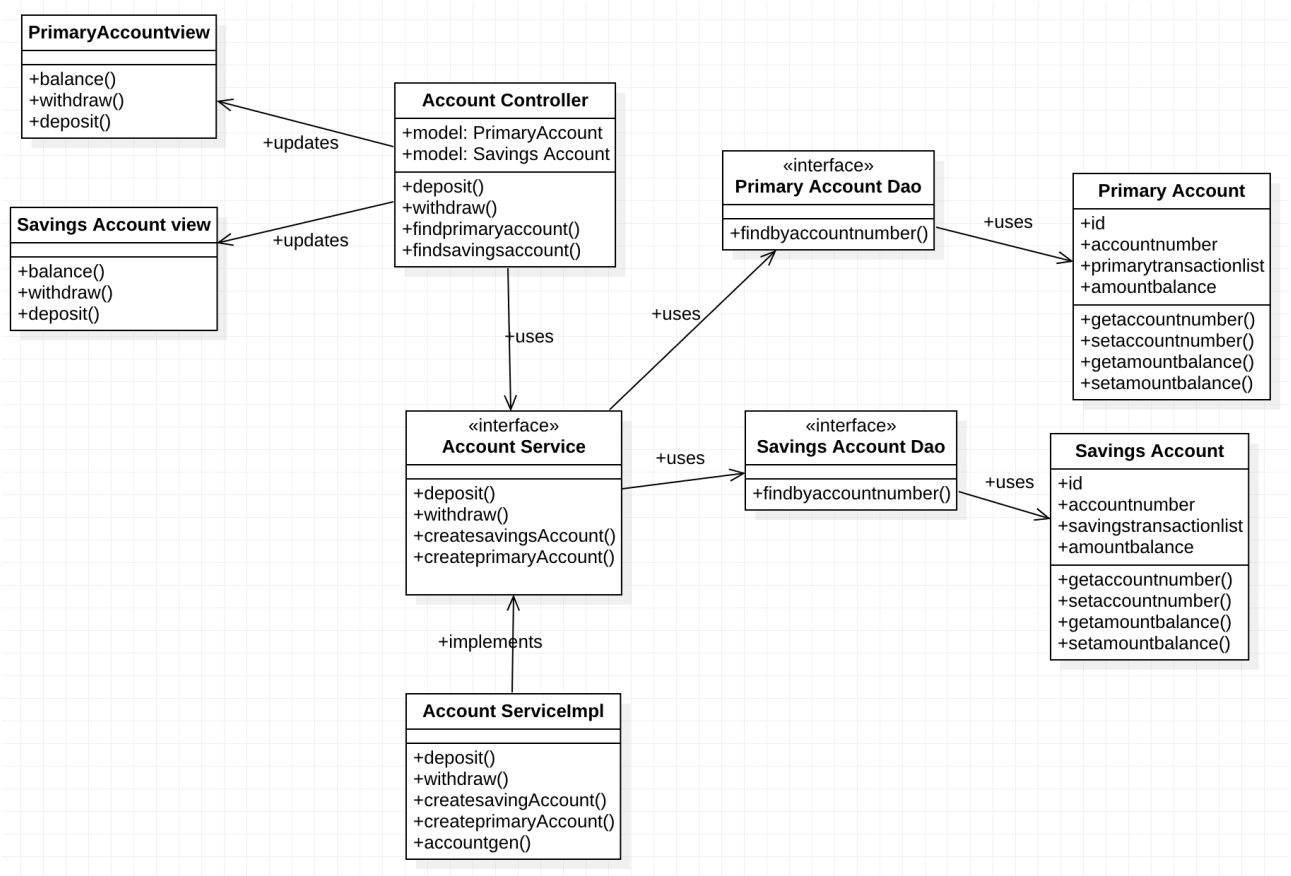


Use Case Descriptions

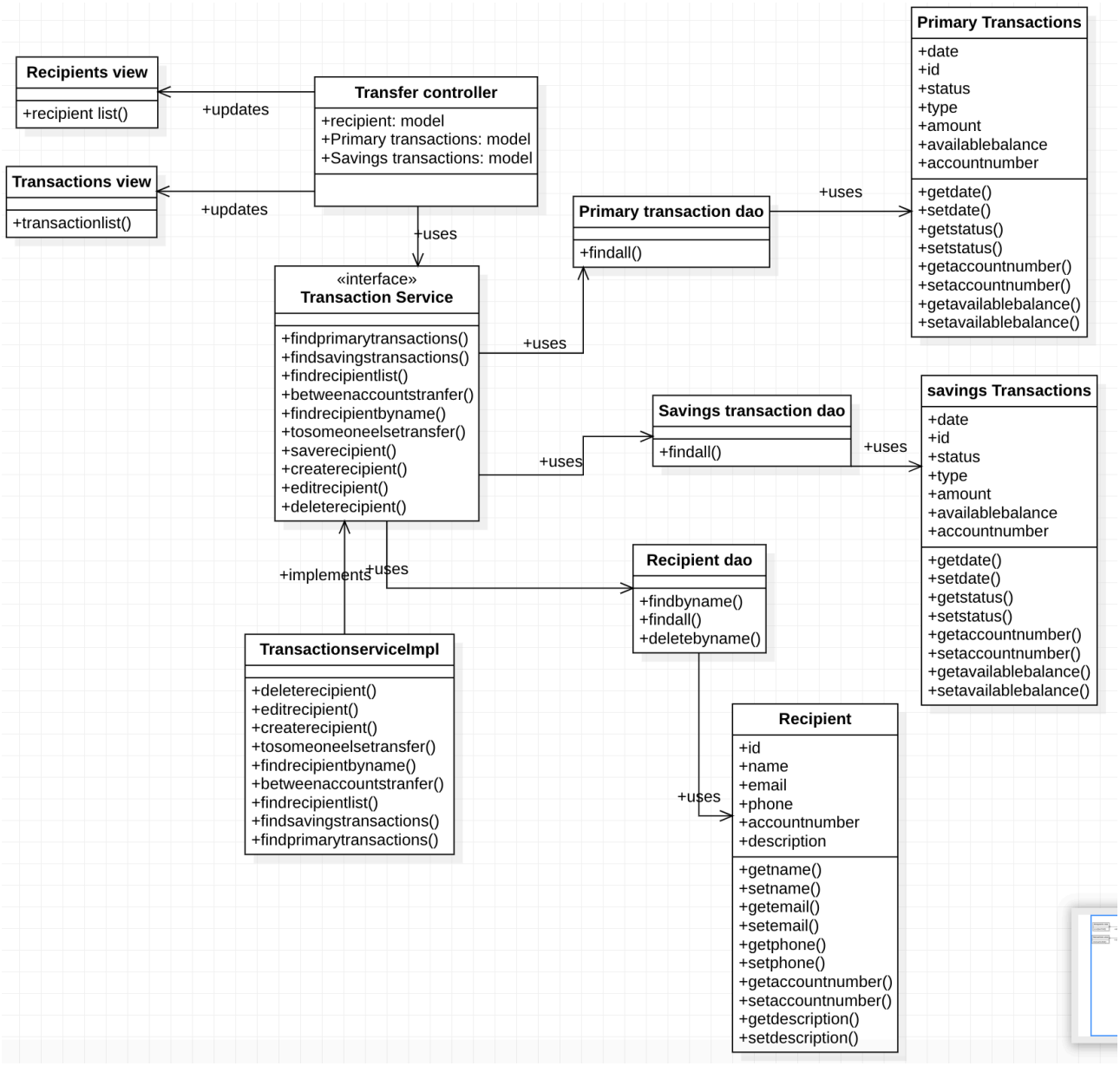
- User registration: A user can register for an account with the application by providing their personal information and creating login credentials.
- Account creation: A registered user can create a primary or savings account with the application.
- Register Recipients: A user can add recipients or edit information about existing recipients, to whom the user can then transfer funds to.
- Account deposit: A user can deposit money into their primary or savings account.
- Account withdrawal: A user can withdraw money from their primary or savings account.
- Account transfer: A user can transfer money between their primary and savings accounts. And also to other registered recipients.
- View account balance: A user can view the current balance of their primary or savings account.
- View transaction history: A user can view the transaction history of their primary or savings account.
- User profile management: A user can edit their personal information, such as their name, address, and email.

Class Model Diagram





Class Model Diagram (Continued)



Design Patterns

- Model-View-Controller (MVC) pattern: This pattern is used in the Spring MVC framework to separate the application into three main components - Model, View, and Controller.

- Dependency Injection (DI) pattern: Spring framework uses this pattern to manage the dependencies of objects and provide them with the required dependencies.

- Factory Method pattern: This pattern is used in the AccountService interface to create instances of PrimaryAccount and SavingsAccount.

- Template Method pattern: This pattern is used in the AccountService interface to define a template for the deposit and withdrawal methods.

- Command pattern: This pattern is used in the TransactionService interface to define commands that can be executed to perform actions such as saving transactions and transferring funds.

- Data Access Object (DAO) pattern: This pattern is used in the UserDao and RoleDao interfaces to provide an abstract interface to interact with the database.

- Builder pattern: This pattern is used in the User class to create a builder for creating user objects with a fluent API.

Design Principles

- Single Responsibility Principle: Each class has a single responsibility and is focused on doing one thing well. For example, the AccountService interface is responsible for handling banking operations, while the TransactionService interface is responsible for managing transactions.

- Open/Closed Principle: The code is open for extension but closed for modification. For example, in the TransactionService interface, we can add new types of transactions without modifying the existing methods.

- Dependency Inversion Principle: High-level modules (interfaces) should not depend on low-level modules (implementations). Both should depend on abstractions. For example, the TransactionService interface depends on the PrimaryTransaction and SavingsTransaction domain classes, but it does not depend on any implementation details.

- Interface Segregation Principle: Clients should not be forced to depend on methods they do not use. For example, the TransactionService interface has separate methods for depositing and withdrawing money from primary and savings accounts, rather than one generic method that handles all transactions.

- Liskov Substitution Principle: Subtypes should be substitutable for their base types. For example, the PrimaryTransaction and SavingsTransaction classes can be substituted for their base class, Transaction, without affecting the behavior of the code.

Architecture Pattern

Model-View-Controller (MVC) - This pattern is commonly used in web development and separates the application into three components: the Model (data and business logic), the View (presentation layer), and the Controller (handles user input and manages communication between the Model and the View).

Individual Contributions

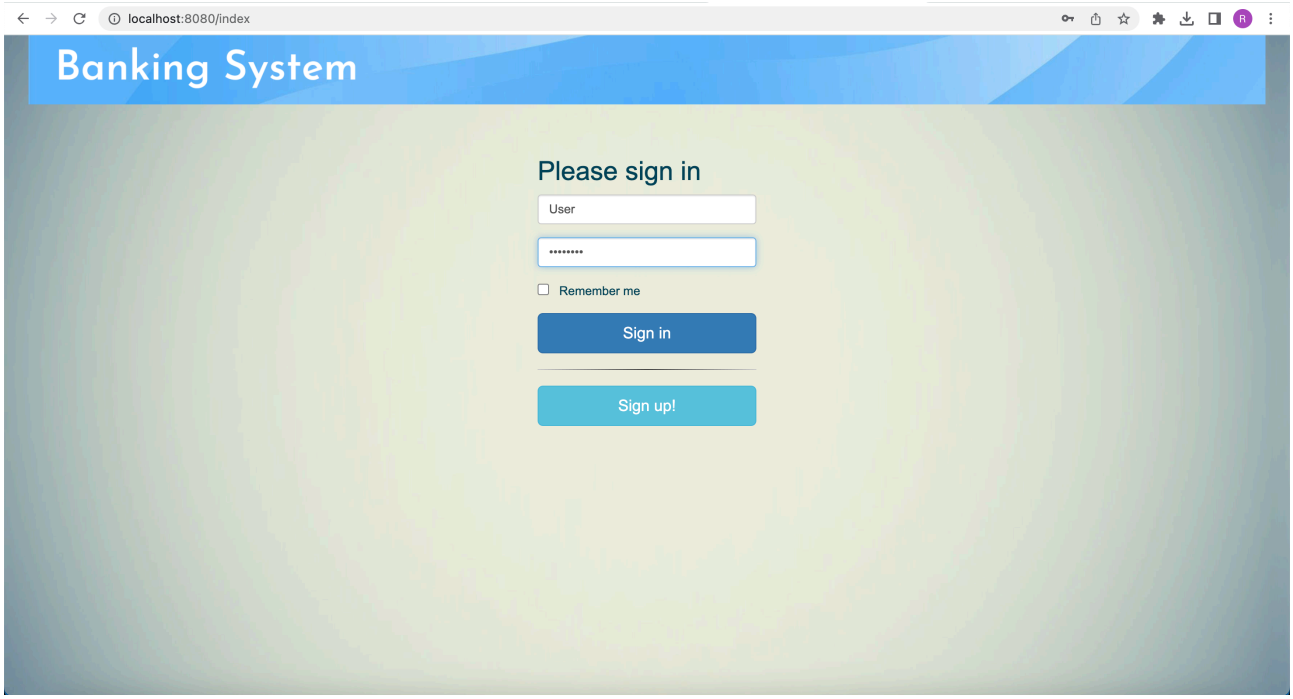
Pradyumna (PES1UG20CS554): Implemented Sign-Up and Sign-In functionalities for authentication as well as facilitating updating details of a user's account.

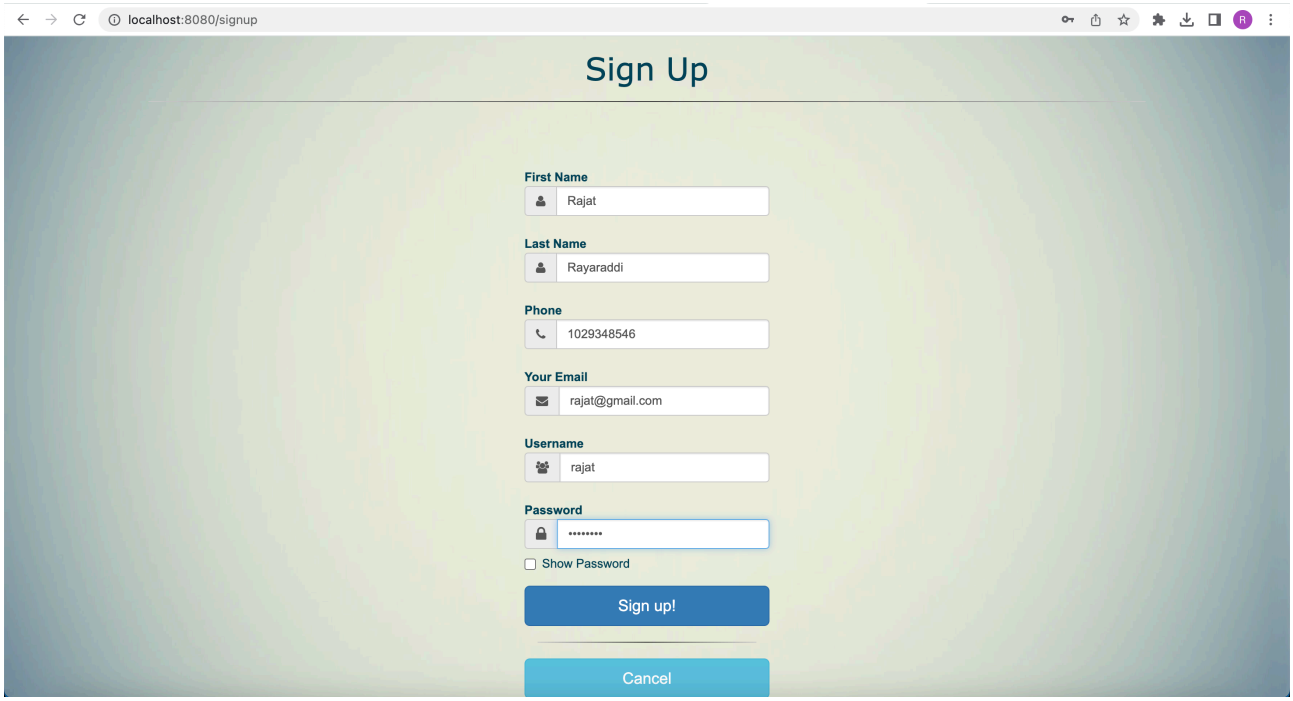
Rajat (PES1UG20CS578): Implemented functions for depositing and withdrawing funds from Current and Savings account.

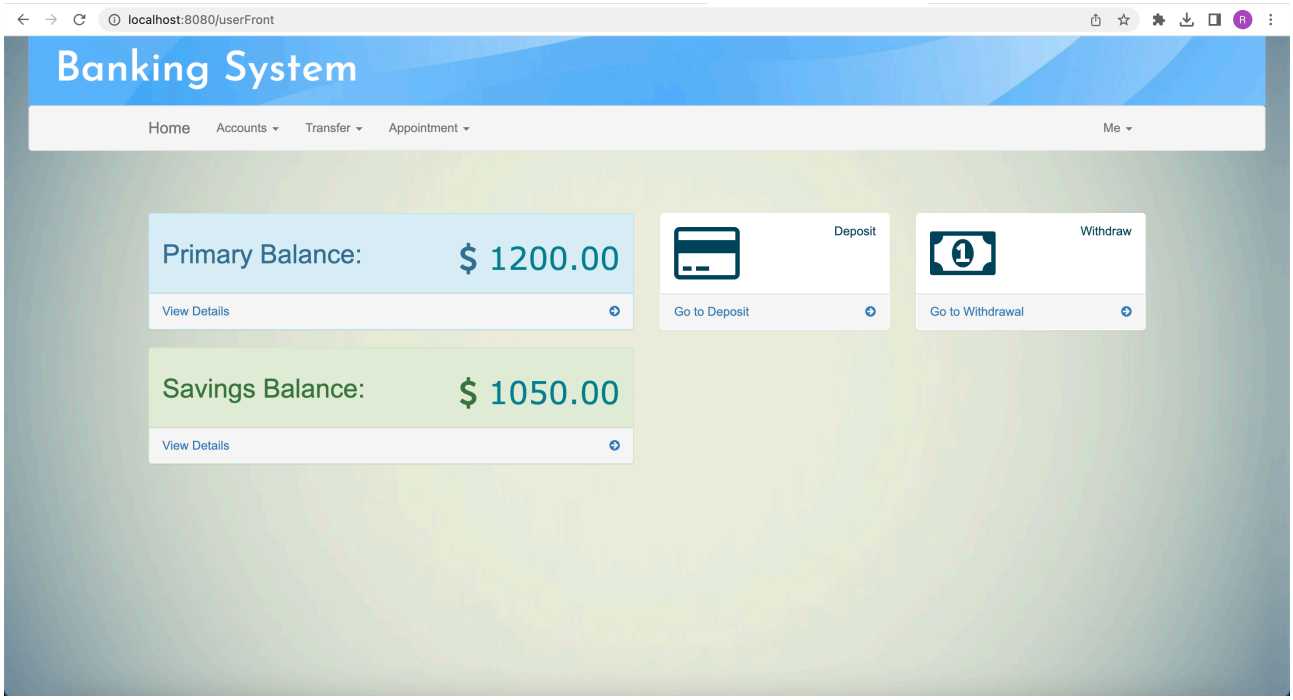
Sada (PES1UG20CS586): Implemented code for checking balance and viewing transaction history. Also, for adding and editing recipients.

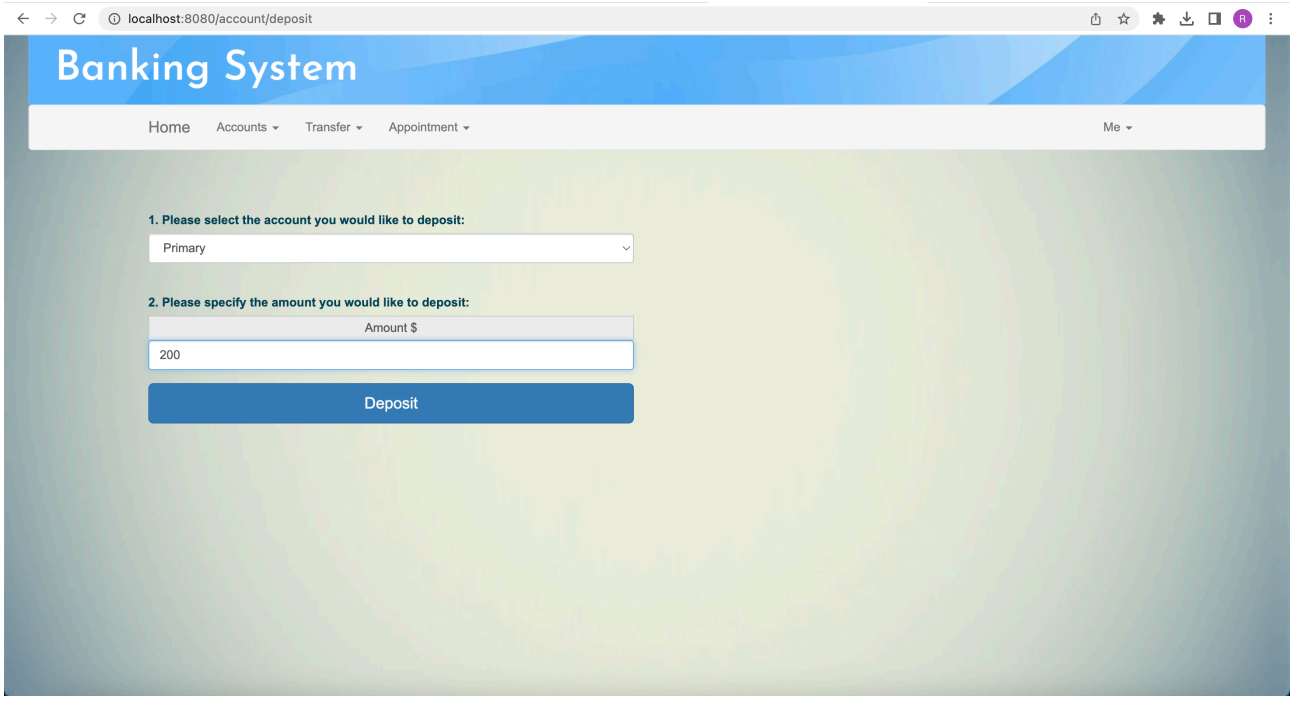
Tanmai (PES1UG20CS586): Implemented functionalities for transferring funds between accounts and to other recipients.

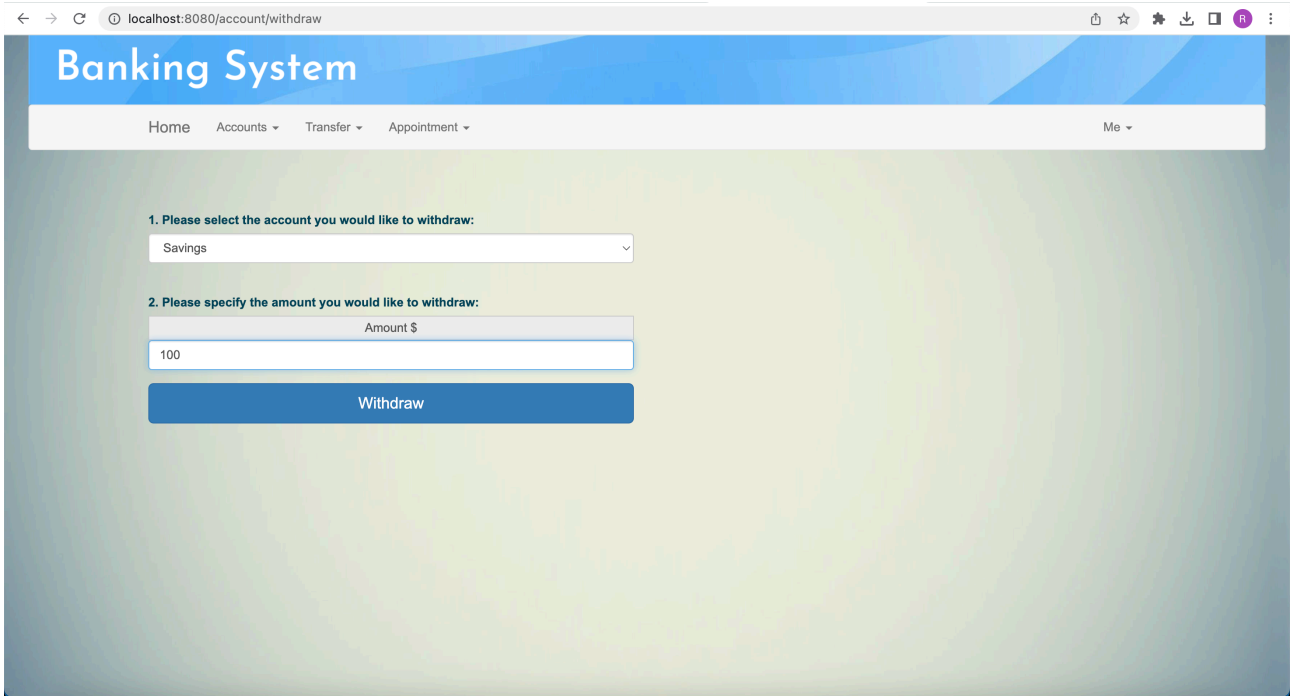
Screenshots

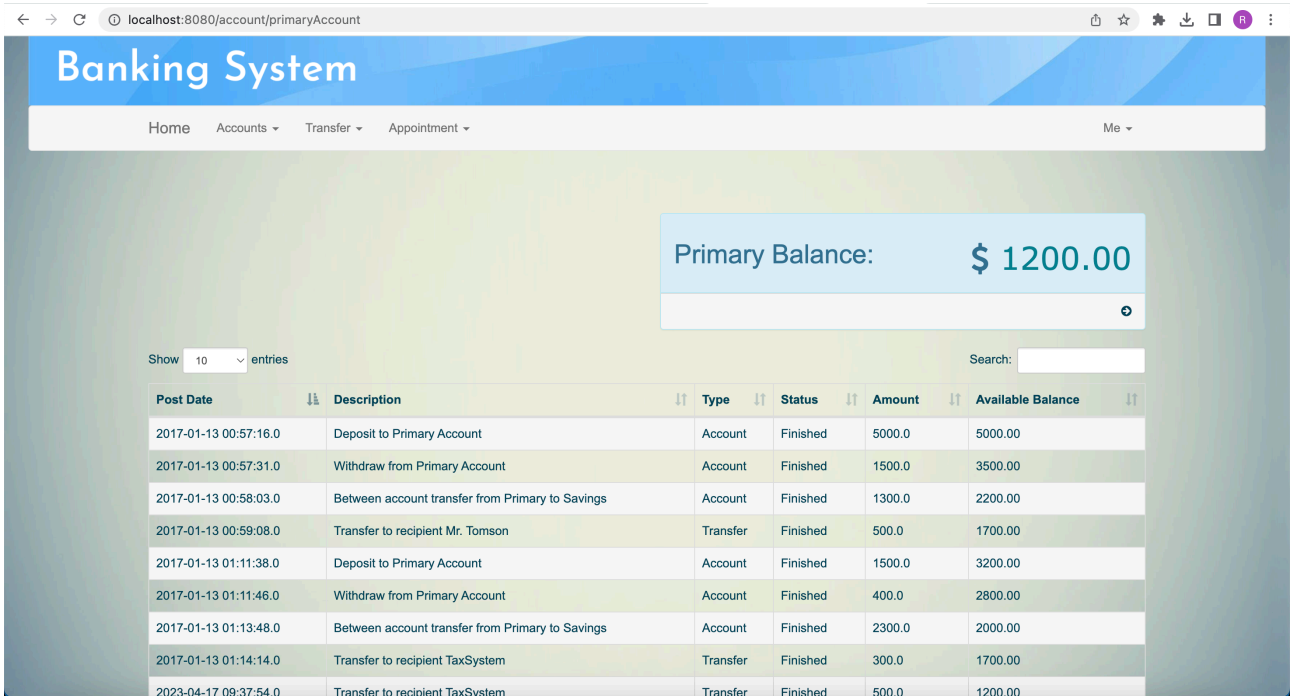




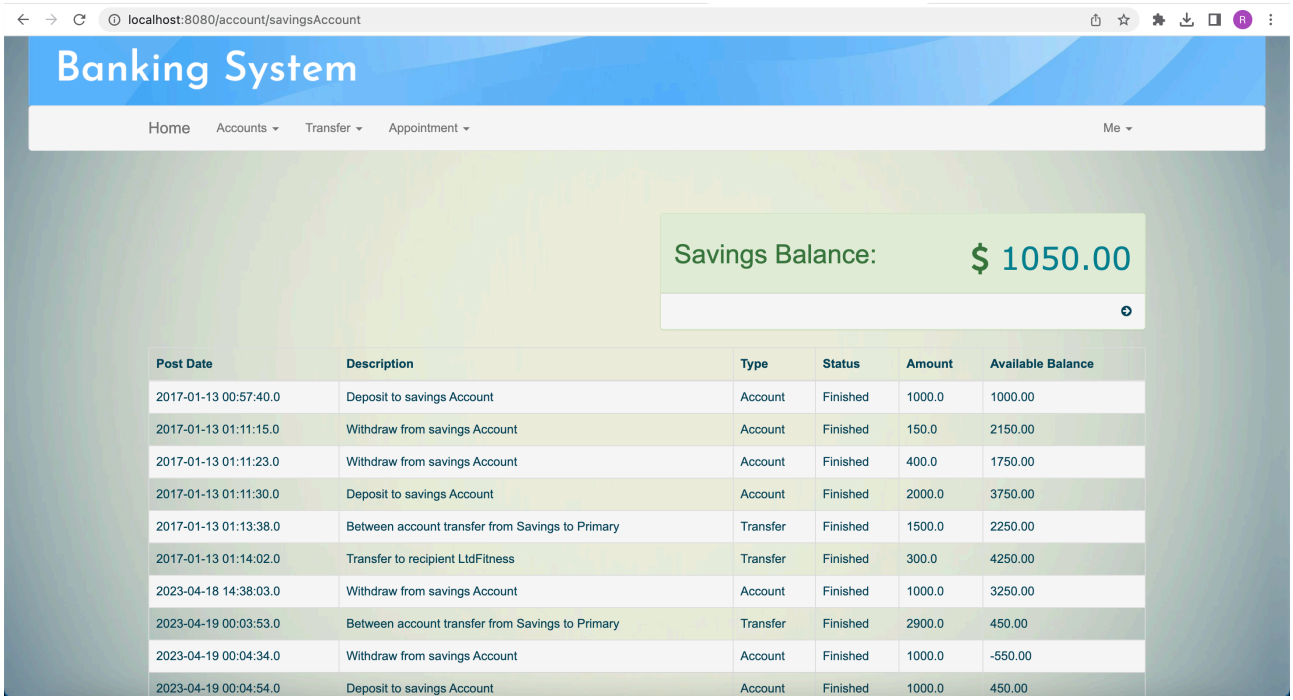

localhost:8080/userFront
Banking System
Home
Accounts
Transfer
Appointment
Me
Primary Balance:
$ 1200.00
View Details
Deposit
Go to Deposit
Withdraw
Go to Withdrawal
Savings Balance:
$ 1050.00
View Details


localhost:8080/account/deposit
Banking System
Home
Accounts
Transfer
Appointment
Me
1. Please select the account you would like to deposit:
Primary
2. Please specify the amount you would like to deposit:
Amount $
200
Deposit

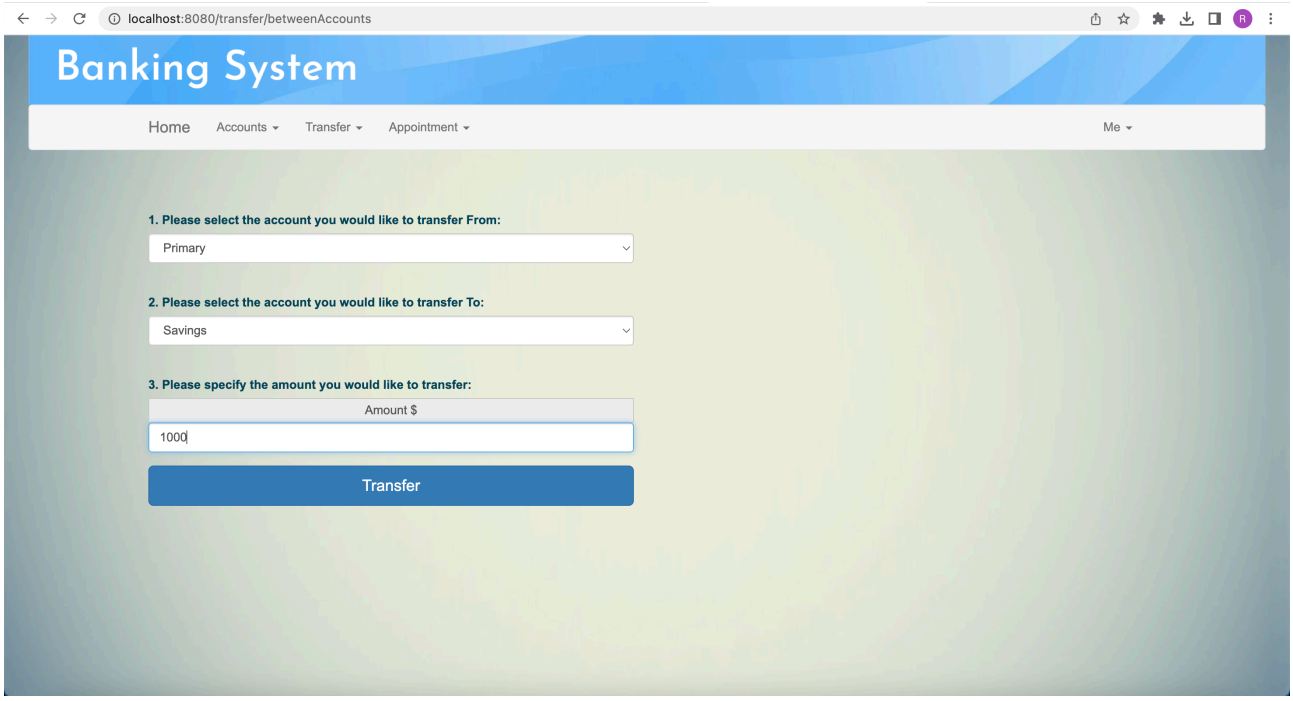

localhost:8080/account/withdraw
Banking System
Home
Accounts
Transfer
Appointment
Me
1. Please select the account you would like to withdraw:
Savings
2. Please specify the amount you would like to withdraw:
Amount $
100
Withdraw

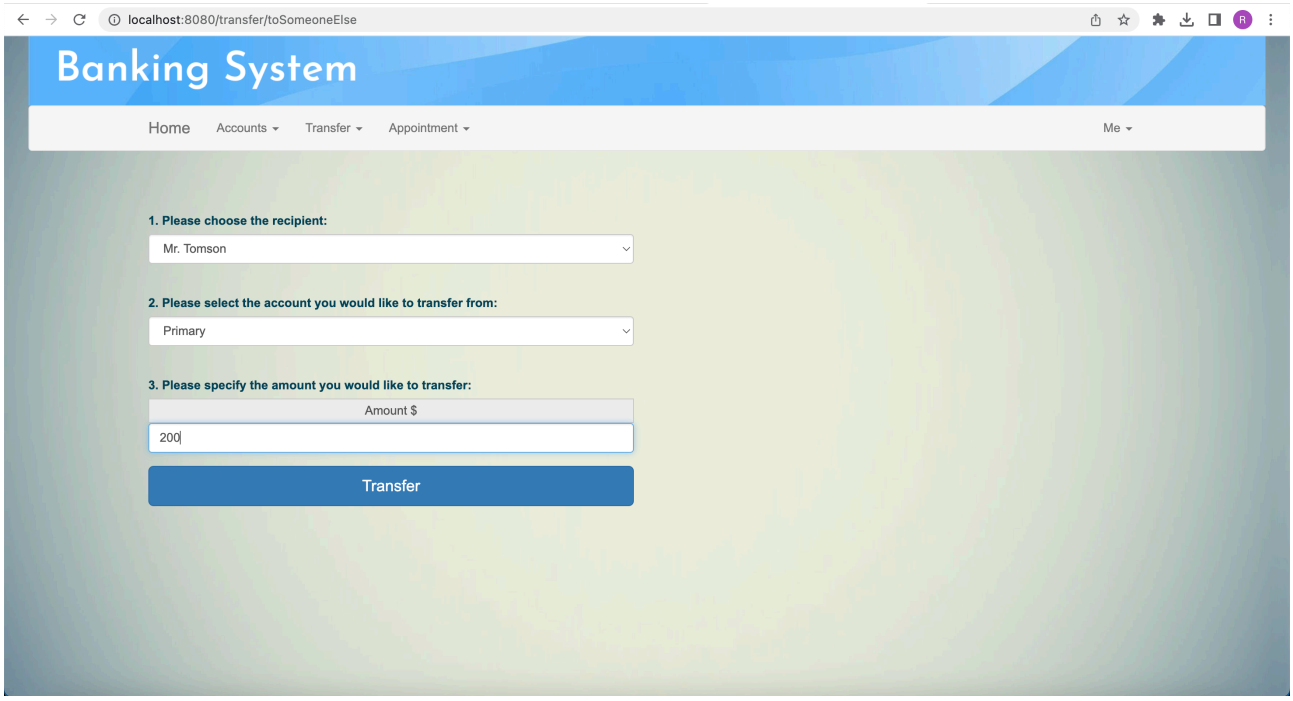


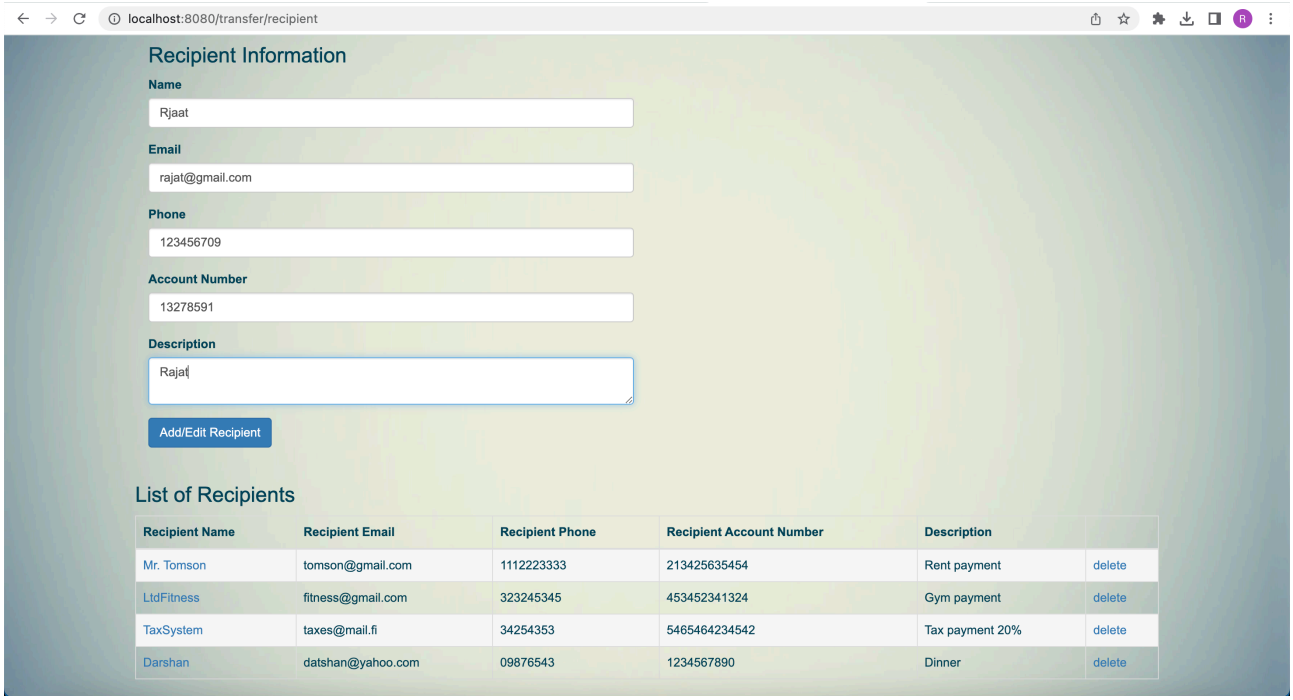
Post Date	Description	Type	Status	Amount	Available Balance
2017-01-13 00:57:16.0	Deposit to Primary Account	Account	Finished	5000.0	5000.00
2017-01-13 00:57:31.0	Withdraw from Primary Account	Account	Finished	1500.0	3500.00
2017-01-13 00:58:03.0	Between account transfer from Primary to Savings	Account	Finished	1300.0	2200.00
2017-01-13 00:59:08.0	Transfer to recipient Mr. Tomson	Transfer	Finished	500.0	1700.00
2017-01-13 01:11:38.0	Deposit to Primary Account	Account	Finished	1500.0	3200.00
2017-01-13 01:11:46.0	Withdraw from Primary Account	Account	Finished	400.0	2800.00
2017-01-13 01:13:48.0	Between account transfer from Primary to Savings	Account	Finished	2300.0	2000.00
2017-01-13 01:14:14.0	Transfer to recipient TaxSystem	Transfer	Finished	300.0	1700.00
2023-04-17 09:37:54.0	Transfer to recipient TaxSystem	Transfer	Finished	500.0	1200.00

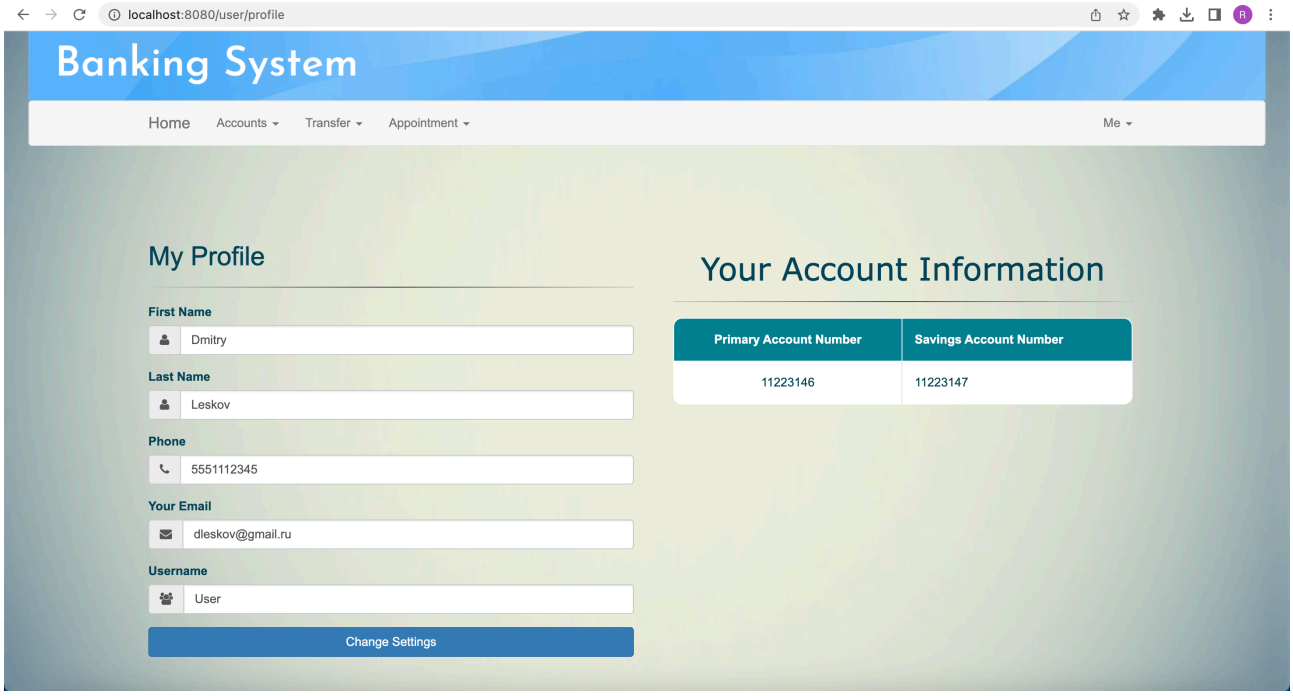


Post Date	Description	Type	Status	Amount	Available Balance
2017-01-13 00:57:40.0	Deposit to savings Account	Account	Finished	1000.0	1000.00
2017-01-13 01:11:15.0	Withdraw from savings Account	Account	Finished	150.0	2150.00
2017-01-13 01:11:23.0	Withdraw from savings Account	Account	Finished	400.0	1750.00
2017-01-13 01:11:30.0	Deposit to savings Account	Account	Finished	2000.0	3750.00
2017-01-13 01:13:38.0	Between account transfer from Savings to Primary	Transfer	Finished	1500.0	2250.00
2017-01-13 01:14:02.0	Transfer to recipient LtdFitness	Transfer	Finished	300.0	4250.00
2023-04-18 14:38:03.0	Withdraw from savings Account	Account	Finished	1000.0	3250.00
2023-04-19 00:03:53.0	Between account transfer from Savings to Primary	Transfer	Finished	2900.0	450.00
2023-04-19 00:04:34.0	Withdraw from savings Account	Account	Finished	1000.0	-550.00
2023-04-19 00:04:54.0	Deposit to savings Account	Account	Finished	1000.0	450.00




localhost:8080/transfer/toSomeoneElse
Banking System
Home
Accounts
Transfer
Appointment
Me
1. Please choose the recipient:
Mr. Tomson
2. Please select the account you would like to transfer from:
Primary
3. Please specify the amount you would like to transfer:
Amount $
200
Transfer


localhost:8080/transfer/recipient
Recipient Information
Name
Rjaat
Email
rajat@gmail.com
Phone
123456709
Account Number
13278591
Description
Rajat
Add/Edit Recipient
List of Recipients
Recipient Name
Recipient Email
Recipient Phone
Recipient Account Number
Description
Mr. Tomson
tomson@gmail.com
1112223333
213425635454
Rent payment
delete
LtdFitness
fitness@gmail.com
323245345
453452341324
Gym payment
delete
TaxSystem
taxes@mail.fi
34254353
5465464234542
Tax payment 20%
delete
Darshan
datshan@yahoo.com
09876543
1234567890
Dinner
delete


localhost:8080/user/profile
Banking System
Home
Accounts
Transfer
Appointment
Me
My Profile
First Name
Dmitry
Last Name
Leskov
Phone
5551112345
Your Email
dleskov@gmail.ru
Username
User
Change Settings
Your Account Information
Primary Account Number
Savings Account Number
11223146
11223147