ONLINE WALLET MANAGEMENT SYSTEM

Team Members

Umang Agarwal

Utkarsh Singh

**INTRODUCTION**

This project is aimed at developing an online Wallet Management System. This is a web-based application that can be accessed over the web. This user can Show payment wallet account balance, Transfer funds from one account to another, adding amount to the payment wallet account, Create payment wallet account. This is an integrated system that contains both the user component, and the Admin component. There are features like report generators etc. in this system.

## Functional components of the project

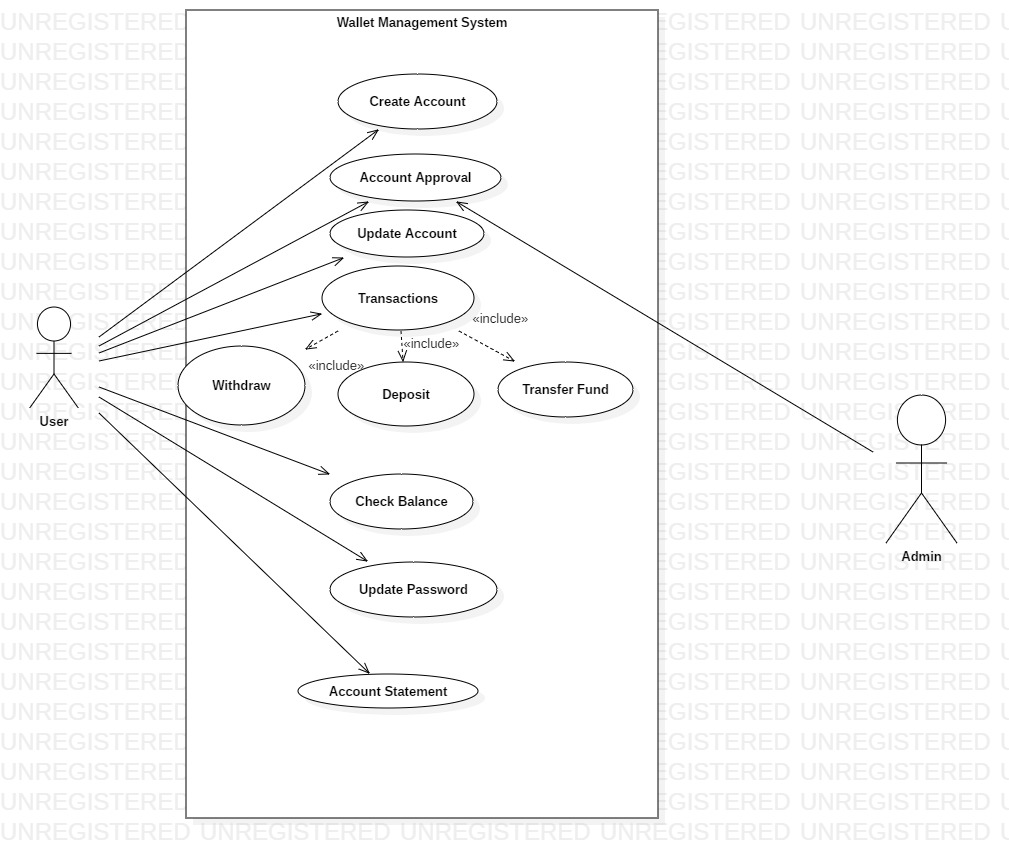
**Admin Operation**

Admin can only approve account.

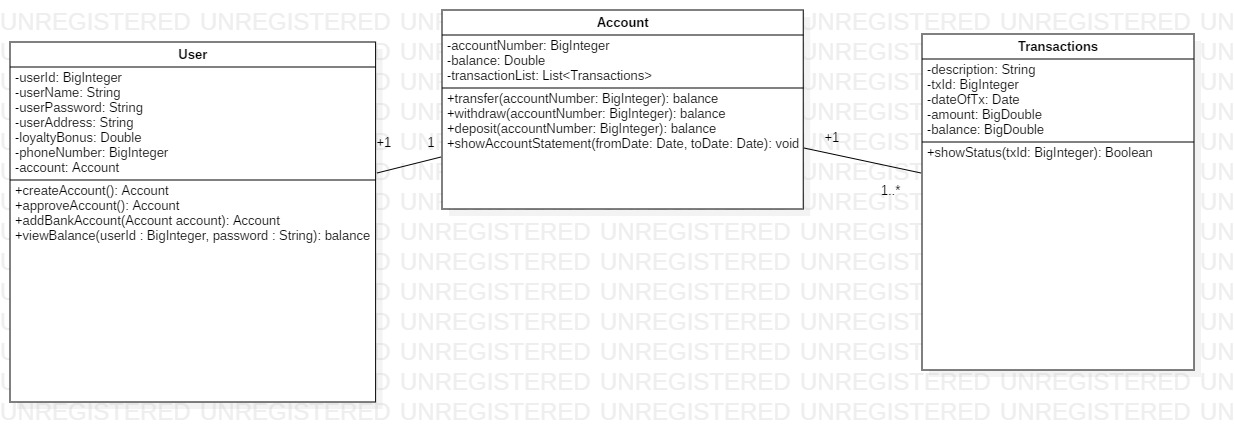
**User Operation**

User can Show payment wallet account balance, Transfer funds from one account to another, adding amount to the payment wallet account, Create payment wallet account. Status should be approved by Admin.

USE CASE VIEW



**CLASS DIAGRAM**



CLASSES AND METHODS

The mobile wallet consists of the following classes:-

1. User – This class stores the details of user who can be customer or administrator.

Attributes –

userId : BigInteger

userName : String

userPassword : String

userAddress : String

loyaltyBonus : Double

phoneNumber : BigInteger

bankAccount : Account

Methods:

viewBalance(id : BigInteger, password : String) : balance

Users can view their wallet balance by using their credentials.

addBankAccoount(account : Account) : Account

Users can add new bank accounts or update existing ones using

this method.

createAccount() : Account

This method is used for the creation of new user account.

approveAccount(): Account

This method is used by the admin to approve a newly created account.

1. Transactions – This class contains the details of transactions.

Attributes:

description : String

transactionId : BigInteger

dateOfTx : Date

amount : BigDouble

balance : BigDouble

Methods:

showStatus(txId : BigInteger): Boolean

It shows whether the transaction is a success or not.

1. Account- This class is used to use new bank accounts by user.

Attributes :

accountNumber : BigInteger

balance : BigDouble

transactionList : List<Transactions>

Methods :

withdraw(accountNumber : BigInteger) : balance

This method is used when the user wants to withdraw money.

deposit(accountNumber : BigInteger) : balance

This method is used when the user wants to deposit money.

transfer(accountNumber : BigInteger): balance

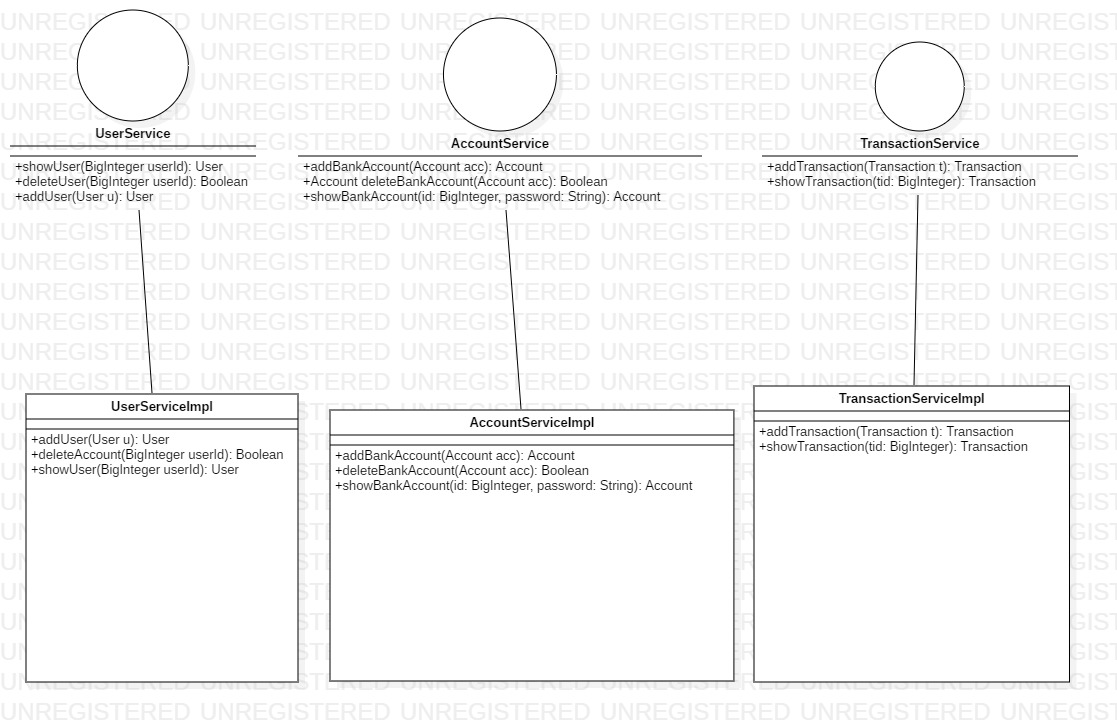
This method is used when the user wants to transfer money from one account to other.

showAccountStatement(fromDate: Date, toDate: Date): void

This method give the information about the transactions done from one date to another.

**Service Layer**

**Class Diagram**

****

Interfaces:

1. UserService

Methods

1. showUser(User u): User

2. deleteUser(User u): User

3. addUser(User u): User

2.AccountService

Methods

1. showBankAccount(Account acc): Account

2. deleteBankAccount(Account acc): Account

3. showBankAccount(id: BigInteger, password: String): Account

3.TransactionService

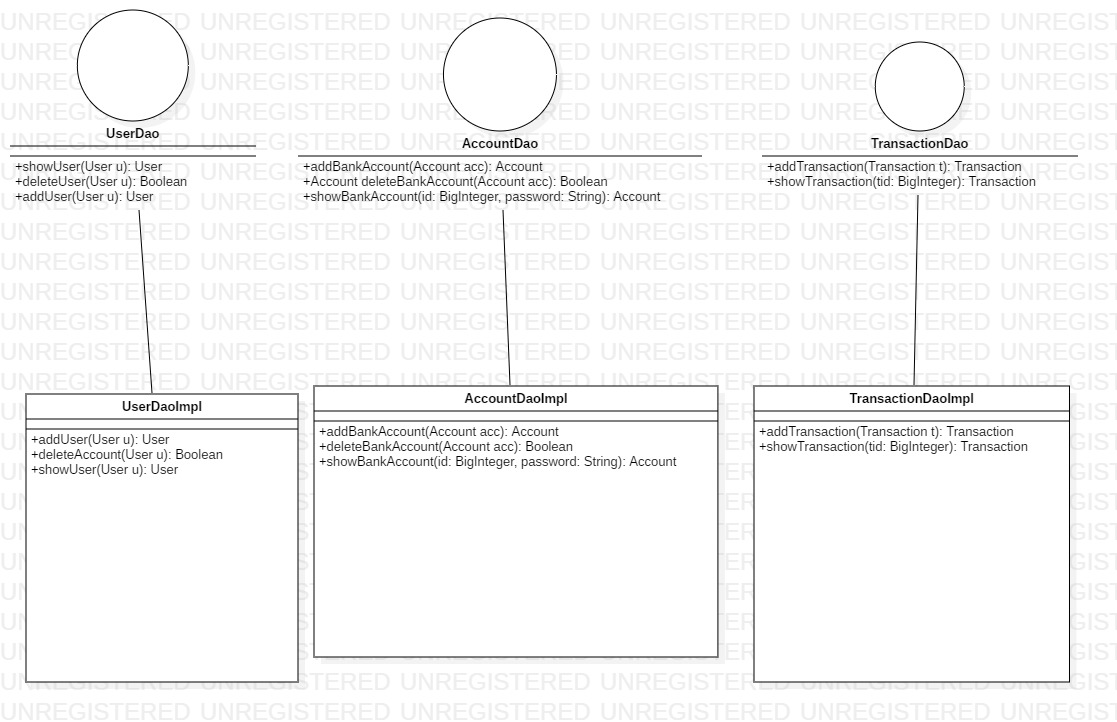
Methods

1. addTransaction(Transaction t): Transaction

2. showTransaction(tid : BigInteger): Transaction

**DAO Layer**

**Class Diagram**



Interfaces:

1. UserDao

Methods

1. showUser(User u): User

2. deleteUser(User u): User

3. addUser(User u): User

2.AccountDao

Methods

1. showBankAccount(Account acc): Account

2. deleteBankAccount(Account acc): Account

3. showBankAccount(id: BigInteger, password: String): Account

3.TransactionDao

Methods

1. addTransaction(Transaction t): Transaction

2. showTransaction(tid : BigInteger): Transaction

**Validation Rules**

1. UserId should be of 8 digits.
2. Phone no should be of 10 digits.
3. First letter of name should be capital letter.
4. Password should be of atleast 8 characters.
5. Bank Account Number should be of 12 digits.
6. FromDate should be less than the ToDate.
7. FromDate shouldn’t be present date.
8. ToDate couldn’t be greater than PresentDate.