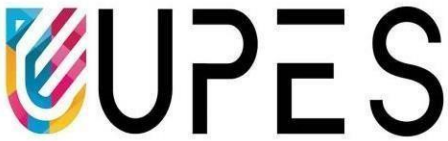


PROJECT TOPIC:

E-Wallet

MASTER OF COMPUTER APPLICATION
SCHOOL OF COMPUTER SCIENCE

Table of Contents

Abstract:

This project aims to develop an e-wallet system that allows users to make online transactions through a computer or smartphone. The system will consist of two micro-services: one for managing users and their wallets, and another for processing transactions and storing logs. The project will use Spring Boot as the framework, Java as the programming language, MySQL as the database and Postman as the testing tool. It will implement features such as user registration, wallet association, transaction history and location awareness. The project will demonstrate the benefits of e-wallets for both consumers and merchants in terms of ease of use, efficiency and reliability.

Analysis phase:

Project statement:

The e-wallet project aims to develop a secure, convenient and user-friendly mobile application that allows users to store, manage and transfer money using their smartphones. The e-wallet app will enable users to link their bank accounts, credit cards, debit cards and other payment methods to their e-wallet account and use them to pay for goods and services online and offline. The e-wallet app will also allow users to send and receive money from other e-wallet users. The e-wallet app will also comply with the relevant regulations and standards in the markets where it operates.

The main objectives of e-wallet:

1. To provide a convenient and cost-effective alternative to cash and traditional payment methods for users.

2. To increase the financial inclusion and empowerment of users, especially those who are unbanked or underbanked.

3. To create a platform that connects users with various financial service providers and offers them tailored solutions based on their needs and preferences.

4. To enhance the user experience and satisfaction by offering features such as rewards, discounts, loyalty programs, personalization and gamification.

5. To generate revenue from fees, commissions, advertisements and partnerships with merchants and financial service providers.

The main deliverables of the e-wallet project are:

1. A functional and user-friendly e-wallet mobile application for Android and iOS devices.

2. A robust and scalable backend system that supports the e-wallet app functionality and integrates with various payment methods and financial service providers.

3. A secure and transparent network that records and verifies the transactions and data on the e-wallet app.

4. A comprehensive marketing strategy that promotes the e-wallet app to potential users, merchants and partners.

Scope of project:

The purpose of this documents is to present a detailed description of the e-wallet system that will help the user and companies to provide a cashless and transparent payment methods. It will explain the purpose and features of the e-wallet system, the interface of the system, what the system will do, and how the different user will use the application and how the transaction process will be carried out. This document is intended for both the stakeholder and the developers.

Module description:

User module: user module is the module that holds the information of the user which will have the features of adding money, sending money, paying bills and recharge with a user-friendly interface. User will be able to access

- Dashboard
- Currency supported
- Payment gateway
- Deposit/transfer

Admin module: admin module will have access to all the features so they can track and manage all the activities that will be performed by the user and will give the admin a over view of the application using the report module**.**

- Dashboard
- Member info
- Currency supported
- Payment gateway
- Deposit/transfer
- Transection setting
- Reports
- Data base backup

Transaction module: Transaction module will have all the transaction history/updates that the user has done with using a e-wallet such as bill payment money transfer etc.

Account module: Account module will have all the related information related to account such as account details, balance in this module user will be able to change or update their bank information such as account no. IFSC code bank name.

Wallet module: Wallet module will have the feature of adding/ sending the money to the users, will also have the feature for bill payment as well as mobile or DTH recharge.

Report module: Report module will generate the report of all the transaction and how the system is functioning this module will only be accessible by the admin.

Hardware Requirements:

Processor: 1.0 GHZ processor or higher.

Memory: Minimum 4GB RAM.

Display size :1280*720 pixels or above.

Software Requirements:

Operating system: Android 8.0 or above, IOS 10 or above, windows 10 or above.

Technology: Java version 18 or above.

Testing tool: Post man.

Technical Feasibility:

The technical feasibility factors of an e-wallet :

- Secure platform for data storage and transaction processing
- Robust encryption protocols to protect user data
- Reliable payment gateway integration with various payment methods and networks
- Adherence to industry standards and best practices, such as PCI DSS
- Implementation of multi-factor authentication and fraud detection mechanisms
- Support for different operating systems and devices, such as smartphones, tablets, and desktops
- Integration with third-party services, such as loyalty programs and merchant partnerships

Operation Feasibility:

Here are some operational feasibility factors to consider when implementing an e-wallet:

- Availability of reliable internet connectivity
- Access to smartphones or other mobile devices by target users
- Integration with existing payment and banking systems
- Compliance with relevant laws and regulations
- Adequate resources for ongoing maintenance and updates
- User adoption and behavior change

Economic Feasibility:

Here are some economic feasibility factors to consider when evaluating an e-wallet:

- Cost of development and implementation, including software, hardware, and infrastructure costs
- Potential cost savings for users, such as reduced transaction fees or elimination of physical payment methods
- Revenue generation opportunities for providers, such as transaction fees or data analytics services
- Potential for increased financial inclusion and access to financial services for underbanked populations
- Increased efficiency in payment processing, leading to cost savings for merchants and other stakeholders

Social Feasibility:

Here are some social feasibility factors to consider when implementing an e-wallet:

- User acceptance and trust in the security and reliability of e-wallets
- Accessibility and inclusivity for users with different socioeconomic backgrounds and technical skills
- Potential impact on employment and labor markets, such as job displacement in industries related to physical payment methods
- Environmental impact of e-wallets compared to traditional payment methods, such as reduced paper waste
- Impact on financial literacy and education, such as increased awareness of digital financial services and budgeting tools

SOFTWARE REQUIREMENT SPECIFICATIONS:

The purpose of a Software Requirements Specification (SRS) is to clearly document the requirements for a software system that is being developed. An SRS is a comprehensive description of the intended functionality, performance, and design of the software system, and serves as a contract between the software development team and the stakeholders or clients who will use the software.

Functional requirements :

Functional requirements are an essential component of a Software Requirements Specification (SRS) and describe the specific functions and features that the software system must provide to meet the needs of its users. Here are some examples of functional requirements that might be included in an SRS:

1. User authentication: The software must provide a secure method for users to authenticate themselves, such as through usernames and passwords, biometric identification, or other means.
2. Data input and management: The software must allow users to input and manage data, such as customer information, transaction records, or inventory data.
3. Search and retrieval: The software must allow users to search for and retrieve specific information, such as customer records or sales reports.
4. Reporting and analytics: The software must provide reporting and analytics capabilities, such as generating sales reports, tracking inventory levels, or analyzing customer behavior.
5. Payment processing: The software must provide secure and reliable payment processing capabilities, such as accepting credit card payments or integrating with third-party payment processors.

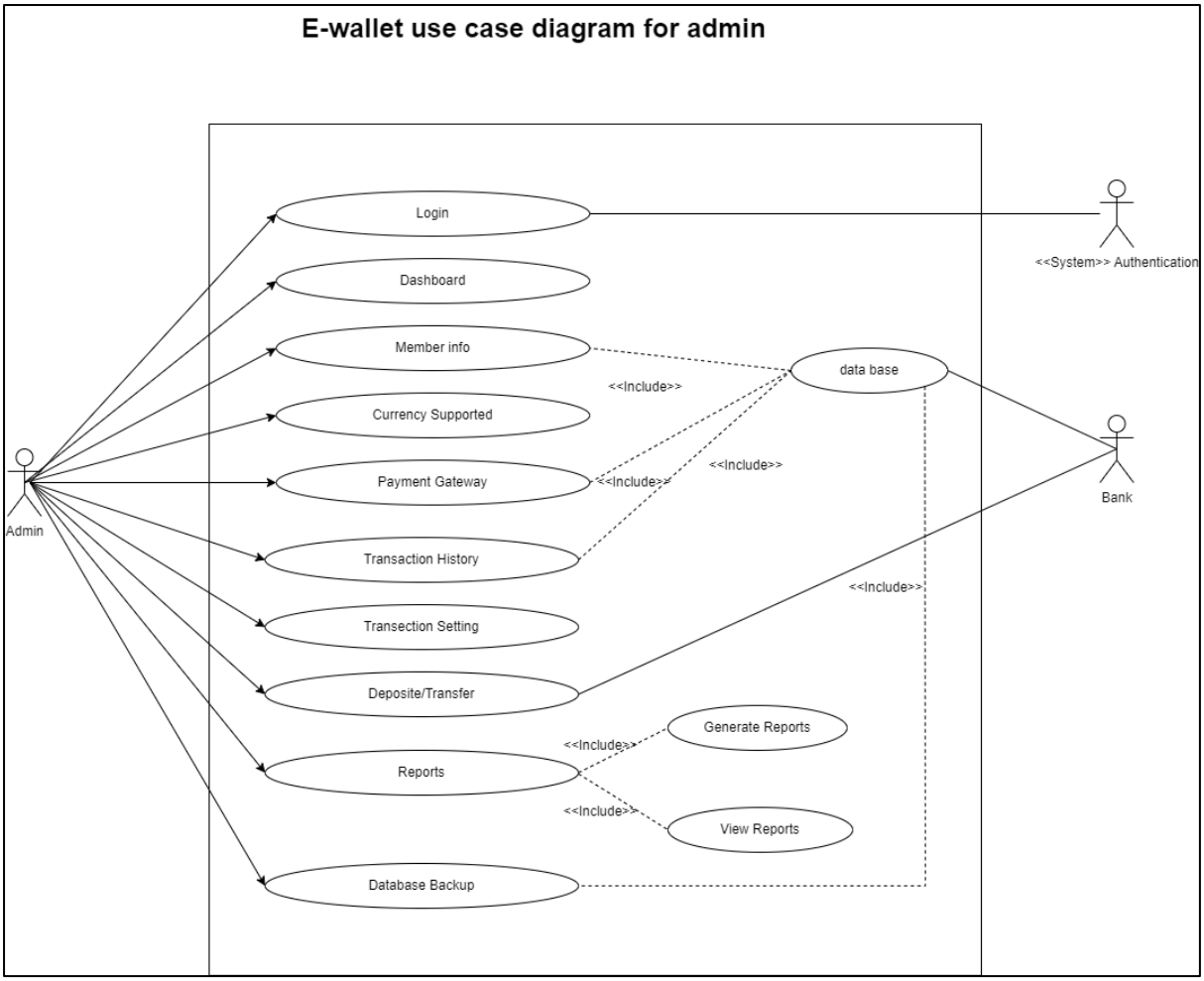
Non-functional requirements :

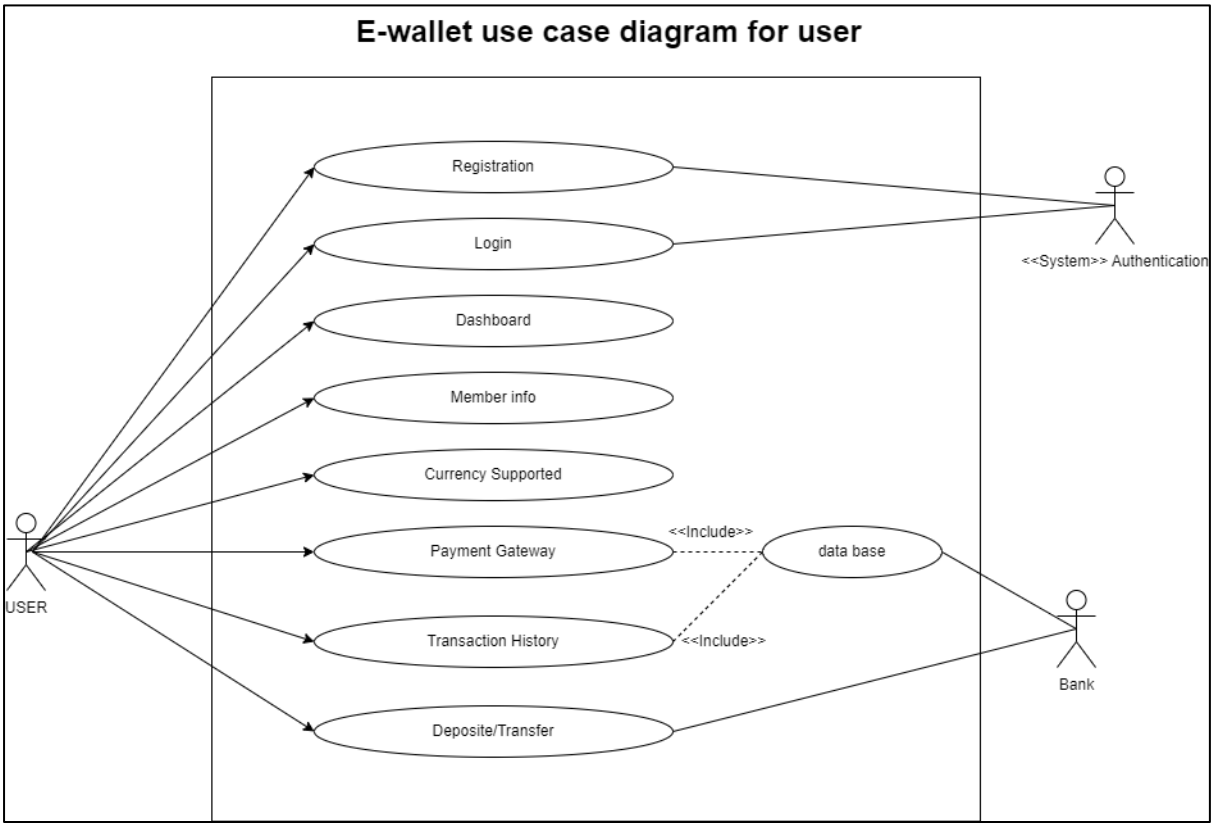
Non-functional requirements describe the qualities or characteristics of the software system that are not directly related to its specific functions or features. They are important considerations for the overall performance, usability, and maintainability of the software system. Here are some examples of non-functional requirements that might be included in an SRS:

- Performance: The software must meet specific performance criteria, such as response time, throughput, or scalability.
- Reliability: The software must be reliable and operate without errors or downtime, and must be able to recover quickly from any failures.
- Usability: The software must be easy to use and navigate, with an intuitive user interface and clear instructions or documentation.
- Security: The software must be secure and protect against unauthorized access, data breaches, or other security threats.
- Compatibility: The software must be compatible with specific hardware, operating systems, or other software systems, as defined by the requirements.

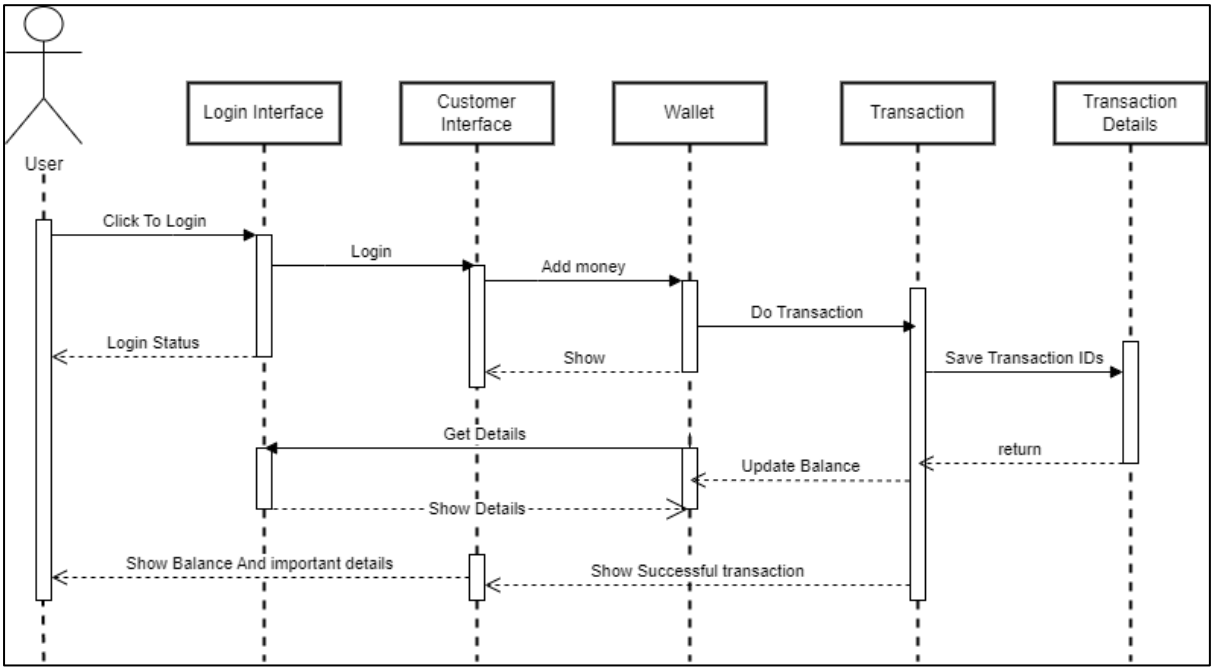
Diagram:

Use-case diagram:

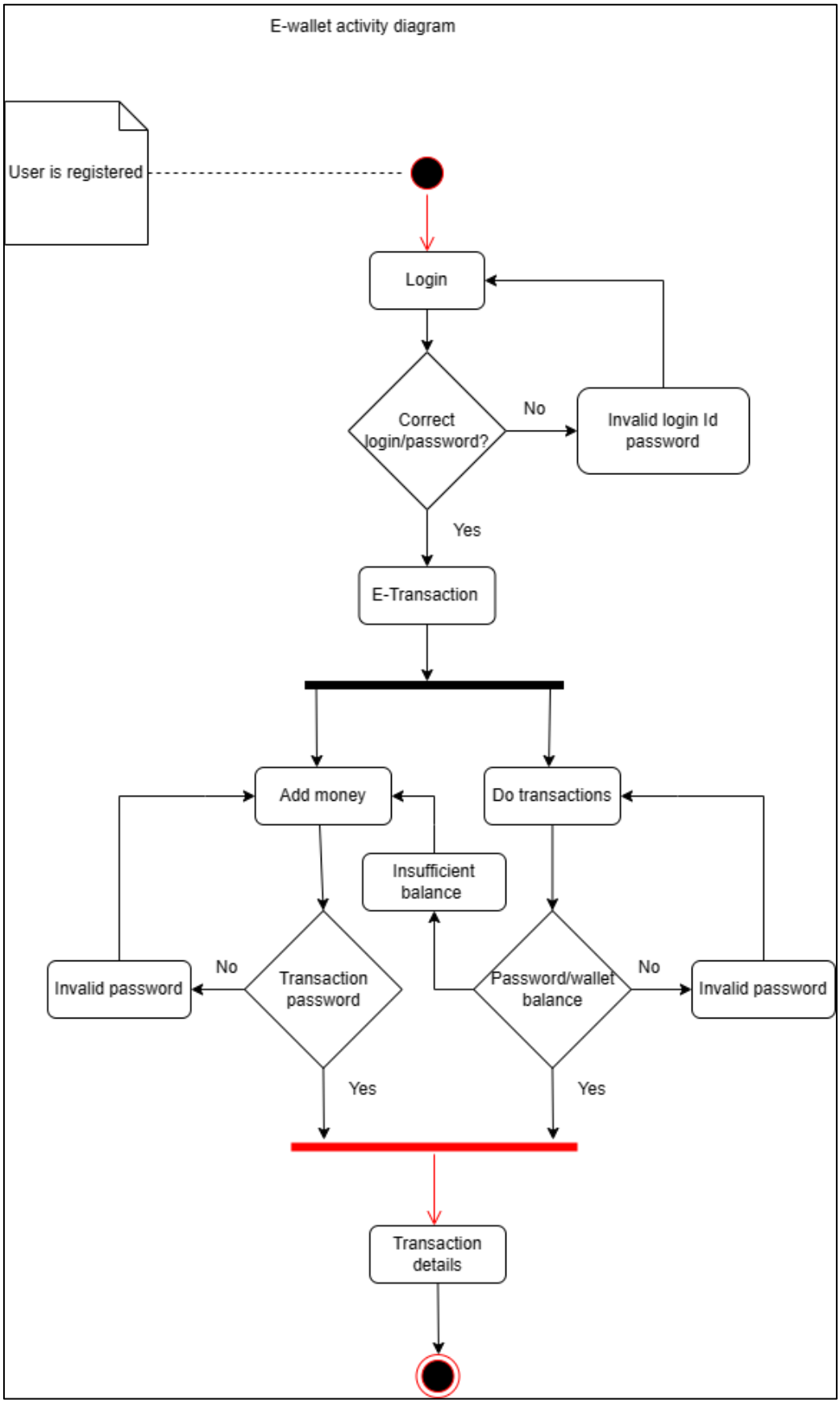



E-wallet use case diagram for user
Registration
Login
Dashboard
Member info
Currency Supported
Payment Gateway
Transaction History
Deposite/Transfer
<<Include>>
<<Include>>
data base
USER
<<System>> Authentication
Bank

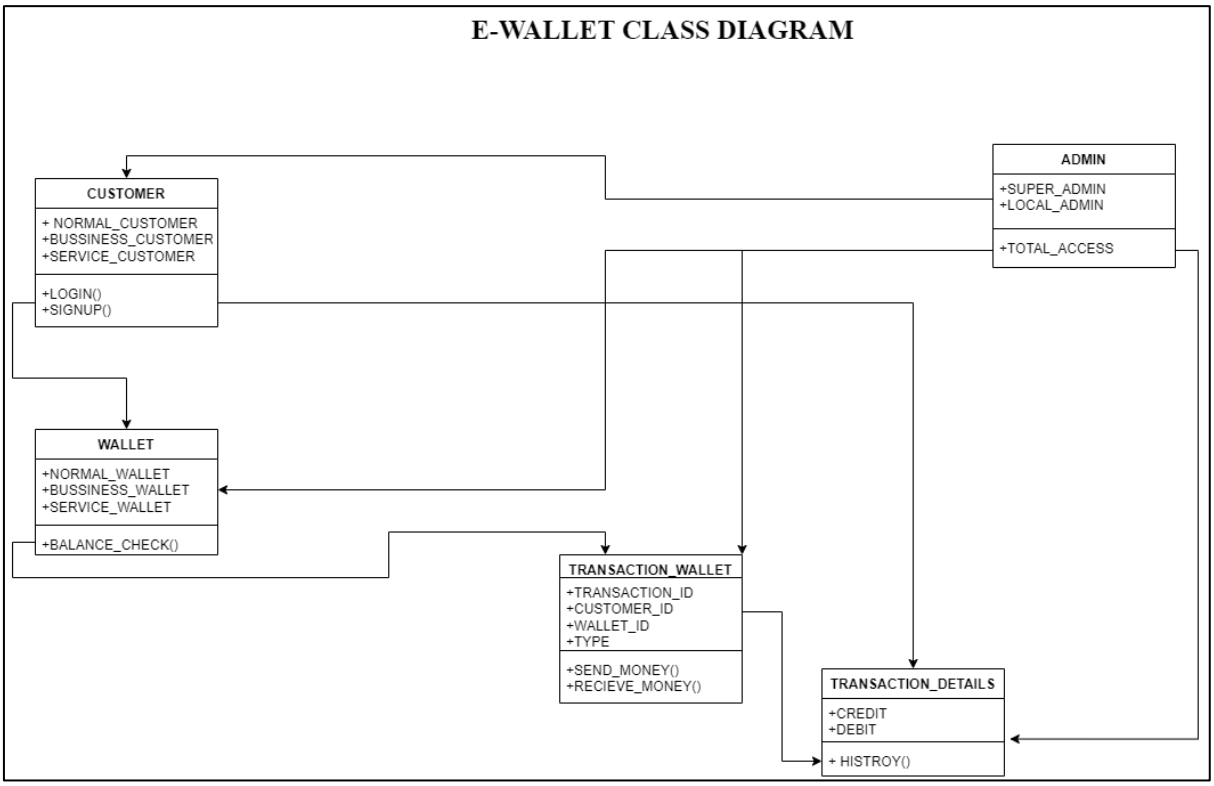
Sequence diagram:



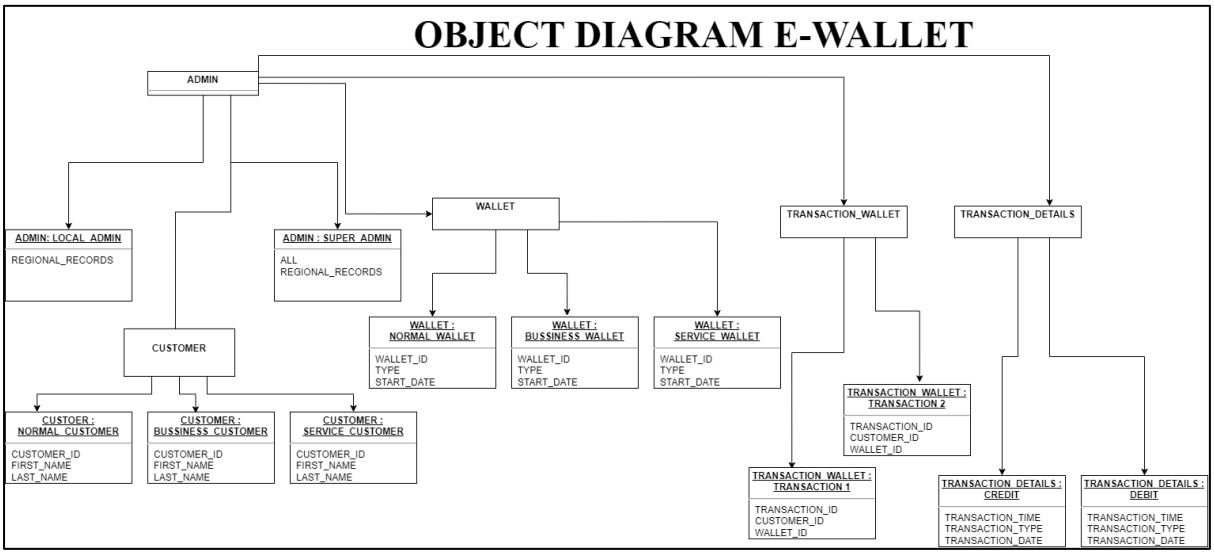
Activity diagram:



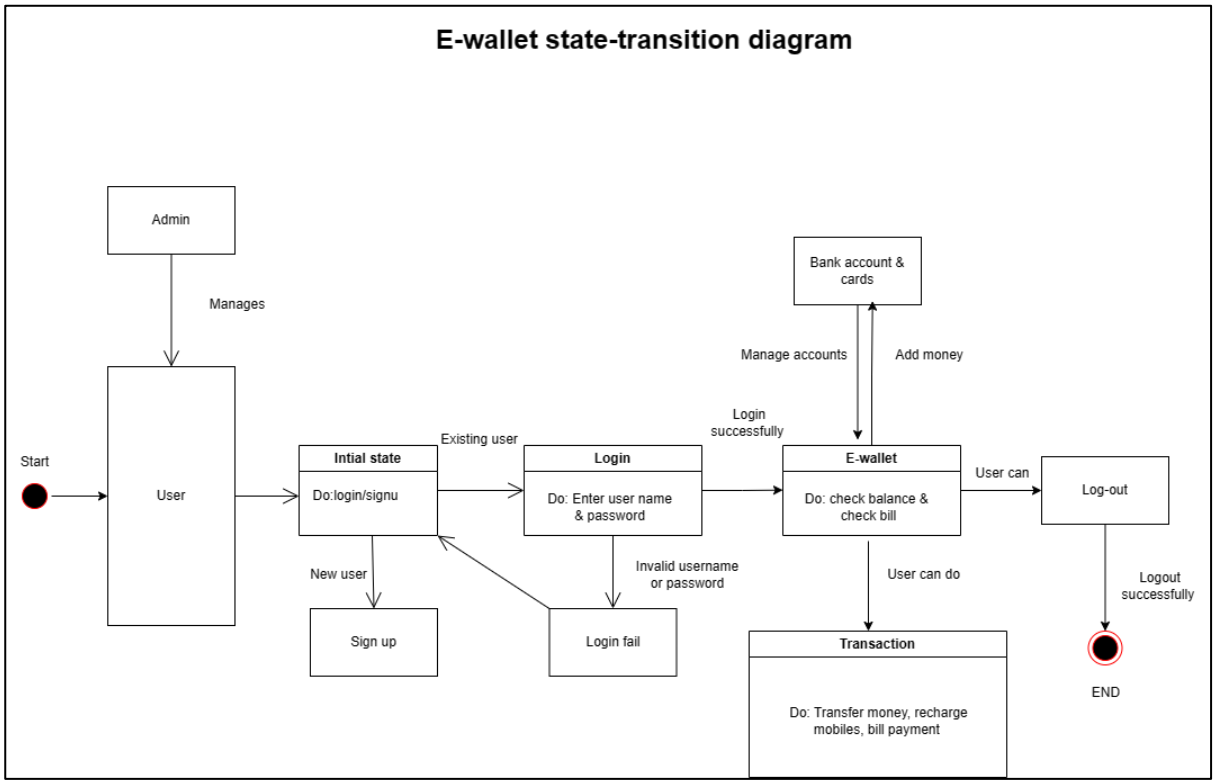
Class diagram:



Object diagram:



State transition diagram:



Deployment diagram: