**Online Wallet**



|  |
| --- |
| **Team Members** |
| **Prakash Devar(Scrum Master)** |
| **Rajeshwari Bhirud(Product Owner)** |
| **Hutesh Mahajan** |
| **Shankar Linganwad** |
| **Shubham Sethi** |
|  |

**Table of Contents**

|  |  |
| --- | --- |
| 1. | Introduction |
| 2. | Overview |
| 3. | Epic and User Stories |
| 4. | UML Diagrams |
| 5. | System Requirements |

1. Introduction

This document outlines a case study for Sprint 1 project. The project is to develop an Online Wallet as integration of all independent microservices. This document contains the work flow of the system and gives guidelines on how the functionality are built gradually in each of the course modules.

2. Overview

Online Wallet system has following functional requirements:

1. Creating wallet account
2. Adding amount in your wallet
3. Display the account information(Balance)
4. Show Transaction
5. Transfer Money from one account to another

The model followed was an agile model. Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product. Agile Methods break the product into small incremental builds. These builds are provided in iterations.

The project had to go under 2 sprints. First sprint had design implemented using spring rest web service and microservice architecture. The assumed data is stored in h2 in-memory database in dev mode and mysql in prod mode. In second sprint we design frontend for backend designed using microservice architecture. In this project we are following CQRS pattern. Implementation of sprint to is with spring with rest services along with Logger implementation and h2 database the front end of this project is designed in AngularJS. Some of DevOps tools are also used such as Jenkins and SonarQube. Herein the entire application is ready to serve a shopping site with the major functionalities.

4. UML Diagrams

1. **Pre-requisites**

User must be logged in wallet to perform operation.

1. **Non-functional requirement**

Proper form validations are maintained. To make it more user friendly proper alert messages are shown wherever required. Any changes in database will only be performed if user gives a confirmation to those pop up warnings.

1. **Register Wallet and Login**

Class Diagrams:

Following is the class diagram of User Registraion and Login which shows how the classes are designed for UserDataProvider, UserQueryService, UserCommandService microservice.

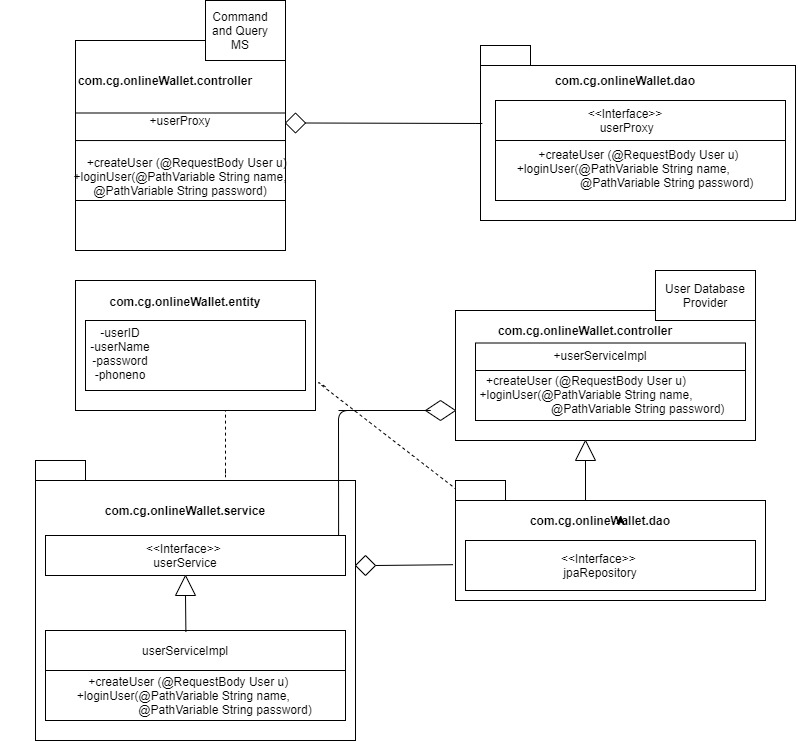


Figure 1: User Login Class Diagram

1. **Wallet management**

Class Diagrams:

Following is the class diagram of Wallet management shows how the classes are designed for WalletDataProvider, WalletQueryService, WalletCommandService microservice.

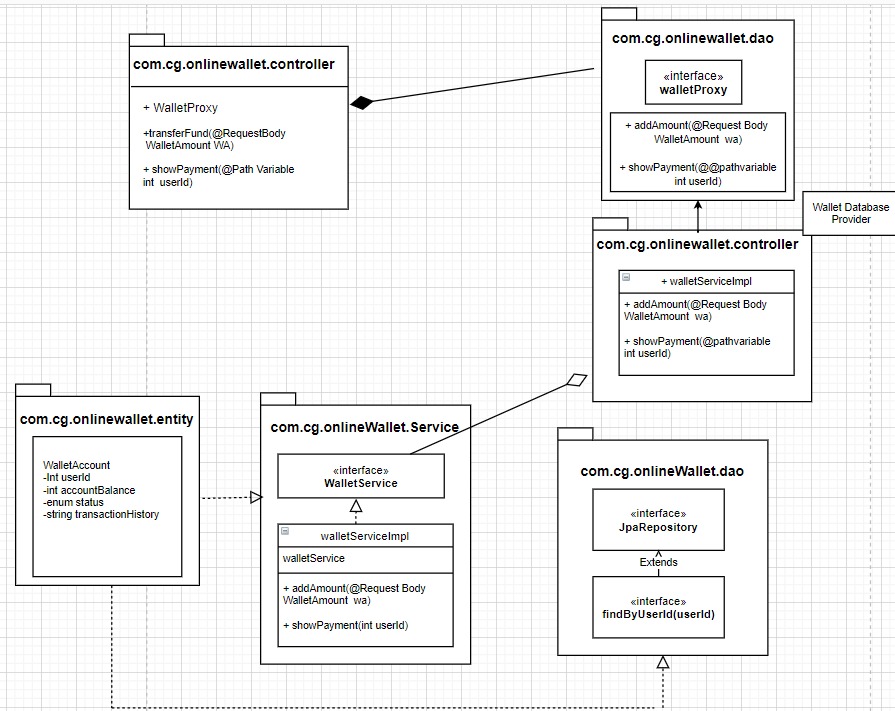


Figure 2: Class Diagram for Wallet Account

1. **Wallet Transaction**

Class Diagrams:

Following is the class diagram of Wallet Transaction shows how the classes are designed for WalletTransactionDataProvider, WalletTransactioQueryService, WalletTransactionCommandService microservice.

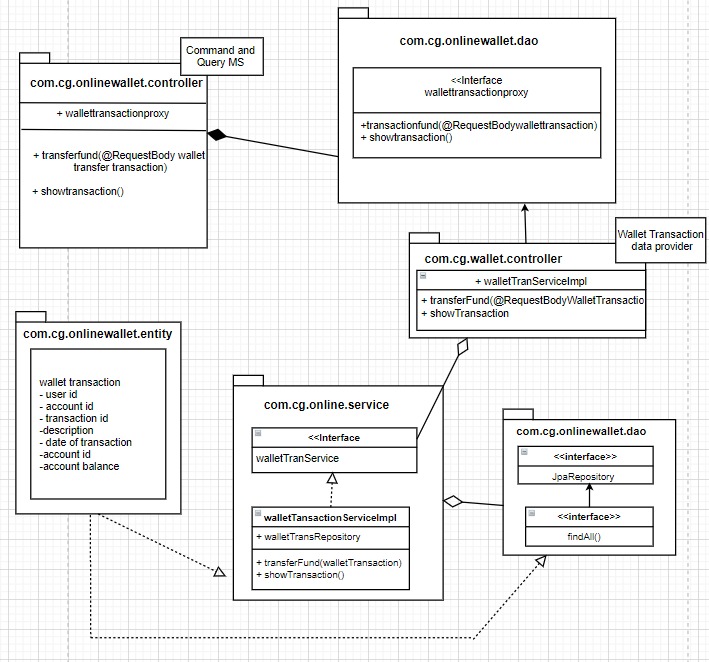


Figure 6: Class Diagram for Wallet Transaction

5. System Requirements

1. **Minimum System Requirement**

* Intel Pentium 90 or higher (P166 recommended)
* Microsoft Windows 7 and above
* Memory: 8GB of RAM (8GB or more recommended)
* Internet Explorer 11.0 or higher or Chrome 45 or above

1. **Software/Tool Requirement**

* JDK 8
* IDE-STS(Spring Tool Suite)
* MAVEN
* PostMan Master
* GitHub- Version Control System
* MySql or H2 database
* IDE-Visual Studio Code