***Software Architecture Document - SplitSmart***

# Table of Contents

[**I. Table of Contents 1**](#_gjdgxs)

[**1. Introduction**](#_3r3bctdfc9am) **2**

[1.1 Purpose 2](#_1fob9te)

[1.2 Scope 2](#_3znysh7)

[1.3 References 2](#_1t3h5sf)

[**2. Architectural Representation 3**](#_4d34og8)

[**3. Architectural Goals and Constraints 3**](#_2s8eyo1)

[**4. Use-Case View 3**](#_17dp8vu)

[**5. Logical View 8**](#_1xa59jc434kx)

[Overview 8](#_oui7r1adqk7h)

[Logical view 10](#_nmcqcwrf47o3)

[Presentation Package 11](#_3ukqfi982945)

[Application Package 12](#_98tgv89mc8z8)

[Domain Package 13](#_c0w2fngdppdq)

[Profile Package 14](#_pwqhg05sk8sv)

[Support Package 15](#_n3wm7x8m1pf3)

[Persistence Package 15](#_bj4qot9nolmt)

[**6. Process View 16**](#_cbymorpd6qvb)

[**7. Deployment View 17**](#_bnvlovpvb9cx)

[**8. Implementation View 18**](#_3j2qqm3)

[**9. Size and Performance 18**](#_2bn6wsx)

[**10. Quality 18**](#_qsh70q)

## 

# 1. Introduction

## 1.1 Purpose

This document provides a comprehensive architectural overview of the system, using a number of different architectural views to depict different aspects of the system. It is intended to capture and convey the significant architectural decisions which have been made on the system

## 1.2 Scope

The scope of this document encompasses the architectural and design aspects of the SplitSmart software. This includes its fundamental capabilities to create and manage user accounts, manage groups, track shared expenses, notify users, monitor balances, track payments, and generate reports. This document focuses on elucidating the technical components of these features, the relations and interactions between them, and their alignment with user requirements. It does not cover non-technical elements such as marketing, financial, and business development strategies associated with the software. This document is relevant to all stakeholders including but not limited to project managers, software developers, testers, and end-users who seek an understanding of the system's structure and functionality.

## 1.3 References

1. Software Architecture Document Example (UofM Canvas)
2. Enterprise Architecture (UML diagrams)
3. LucidChart (UML diagrams)
4. Requirement Document SplitSmart (CIS 376 group 2) (our group)
5. Planning Document SplitSmart (CIS 376 group 2) (our groups)
6. Our groups Github repo ([Link](https://github.com/ParP924/split-smart))
7. Meeting notes (on github repo listed above)
8. Lecture videos (UofM Canvas)

# 2. Architectural Representation

This Document illustrates the architecture of SplitSmart in a series of views - use case view, process view, deployment view, and implementation view. These views are designed using Unified Modeling Language (UML).

# 3. Architectural Goals and Constraints

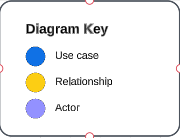
The following requirements and system constraints heavily influence the architectural design:

1. The application must be compatible with all commercially available web browsers and must support high traffic volumes.
2. Payment tracking information should be sent to an external financial system for real-world transactions (post release requirement). Therefore, it is important to design an interface capable of integrating with such systems.
3. All credit card or other financial transactions must be transmitted securely.

# 4. Use-Case View

The use-case view of the software architecture describes the set of scenarios and/or use cases that represent critical functionality. It also outlines the scenarios that may exercise many architectural elements or stress a particular aspect of the architecture. The use cases in SplitSmart are as follows (bolded ones have a significant bearing on the architecture).

* **Create User Account**
* **Create Group**
* **Add Expense**
* Approve Expense
* **Receive Notification**
* **Track Balance**
* **Record Payment**
* Generate Report
* Edit User Profile
* Edit Group Details
* Leave Group
* **Close Account**

The following diagrams depict the use cases in the system.  


*Figure 1: Diagram Key for use-case diagrams*

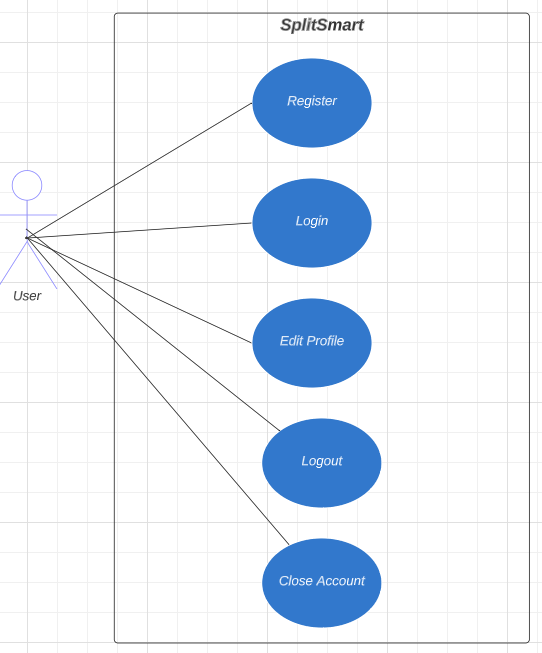
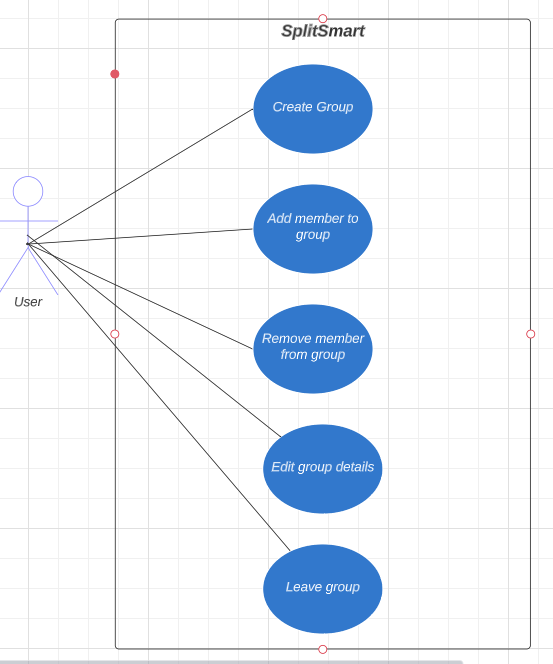
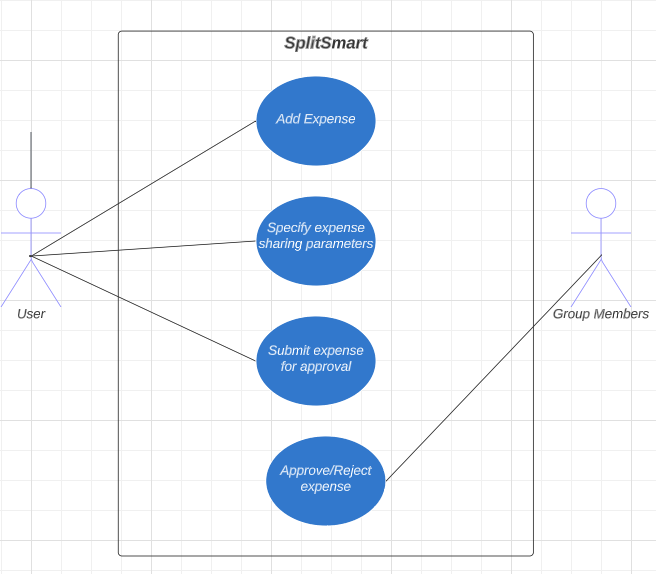


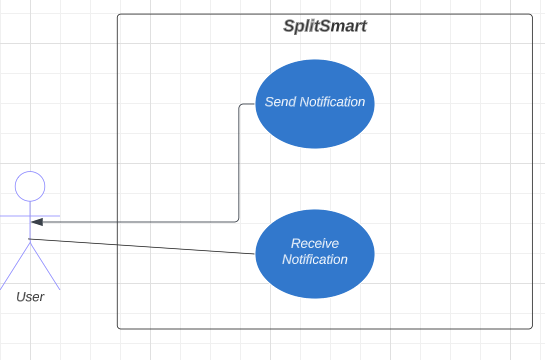
Figure 2: ***User Account Management*** use-case



*Figure 3:* ***Group Management*** *use-case*

**

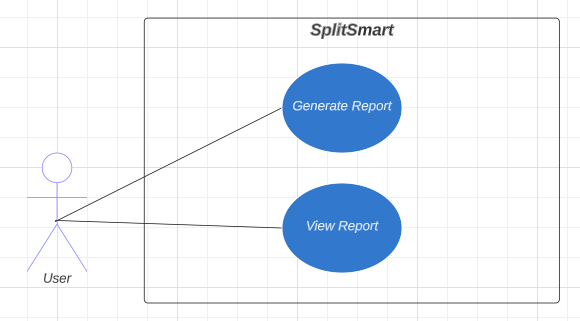
*Figure 4:* ***Expense Management*** *use-case*

**

*Figure 5:* ***Notification Management*** *use-case*

## 

*Figure 6:* ***Balance and payment Management*** *use-case*

**

*Figure 7:* ***Reporting*** *use-case*

**Significant Use Case Descriptions**

1. **User Account Management**

Register: A new user enters their personal information and creates a new account.

Login: An existing user enters their username and password to log into their account.

Edit Profile: A logged-in user changes their profile information, such as their name, email, or password.

Close Account: A logged-in user chooses to permanently close their account. All personal information is deleted from the system.

1. **Group Management**

Create Group: A logged-in user creates a new group and invites other users to join.

Add/Remove Member: The group creator adds or removes members from the group.

Edit Group Details: The group creator or authorized member modifies the group's details, such as its name or description.

Leave Group: A group member chooses to leave the group.

1. **Expense Management**

Add Expense: A group member creates a new expense, providing details such as the amount, date, and how the cost should be split.

Specify Splitting Parameters: The user who added the expense specifies how it should be split among the group members.

Submit Expense for Approval: The added expense is submitted to other group members for approval.

Approve/Reject Expense: The other group members review the expense and either approve or reject it.

1. **Notification Management**

Send Notification: The system sends a notification to group members whenever a new expense is added or an existing expense is updated.

Receive Notification: The group members receive the notifications sent by the system.

1. **Balance and Payment Management**

View Individual/Group Balance: A user views their individual balance or the total balance for the entire group.

Record a Payment: A user records a payment that they've made to another group member.

Update Balance: The system updates the balances of the group members based on the recorded payments.

1. **Reporting**

Generate Report: A user requests a report of their expenses, balances, and payments for a specified period.

View Report: The user views the generated report.

# 5. Logical View

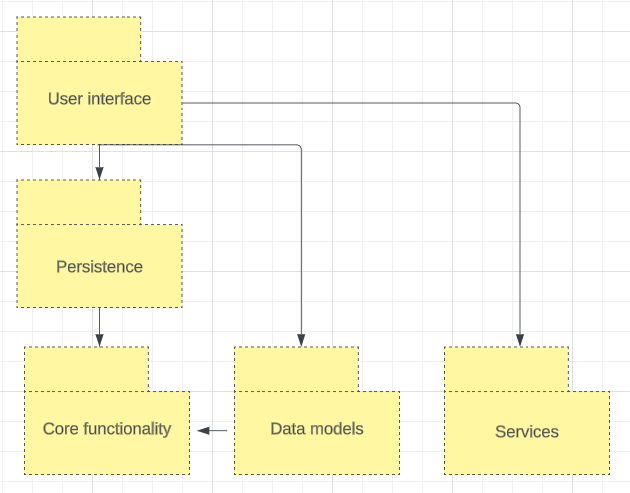
## Overview

The logical view of the SplitSmart app provides an overview of its internal structure and organization. It includes the arrangement of classes, their categorization into service packages and subsystems, and the layering of these subsystems.

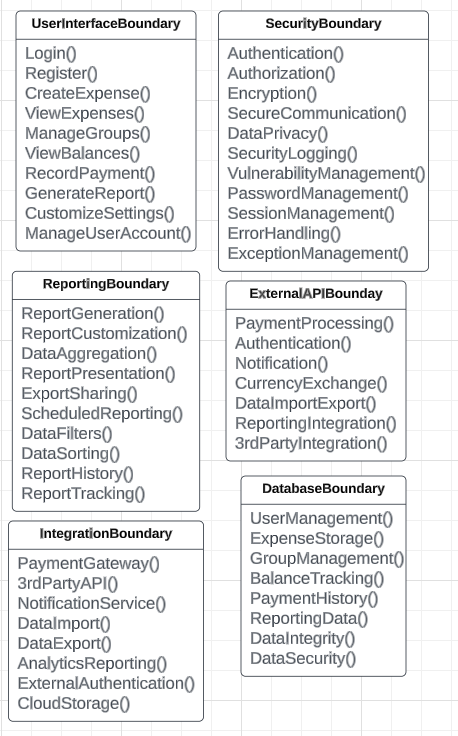
The logical view of SplitSmart comprises the following main packages:

* **User Interface**
  + This package contains classes responsible for handling user interactions and displaying the app's screens and forms. It includes boundary classes that support user account management, expense creation, group management, payment processing, and reporting functionalities.
* **Core Functionality** 
  + The package focuses on major processing functionality within the app. It includes control classes that support expense management, group coordination, user authentication, profile management, payment processing, and reporting functionalities.
* **Data and Models** 
  + This package contains classes that represent the essential entities and data models used in SplitSmart. It includes classes related to expenses, user profiles, groups, payments, and other relevant data structures.
* **Persistence**
  + This package consists of classes responsible for persisting specific objects within the app. Currently, user profiles are persisted, but there may be future considerations for persisting other data entities, such as expenses or payments, depending on the chosen architecture and technology.
* **Services**
  + This package encompasses classes that provide system-level functionality for maintenance purposes. Currently, all maintenance activities are performed manually, but in the future, automated services may be introduced to handle specific maintenance tasks.

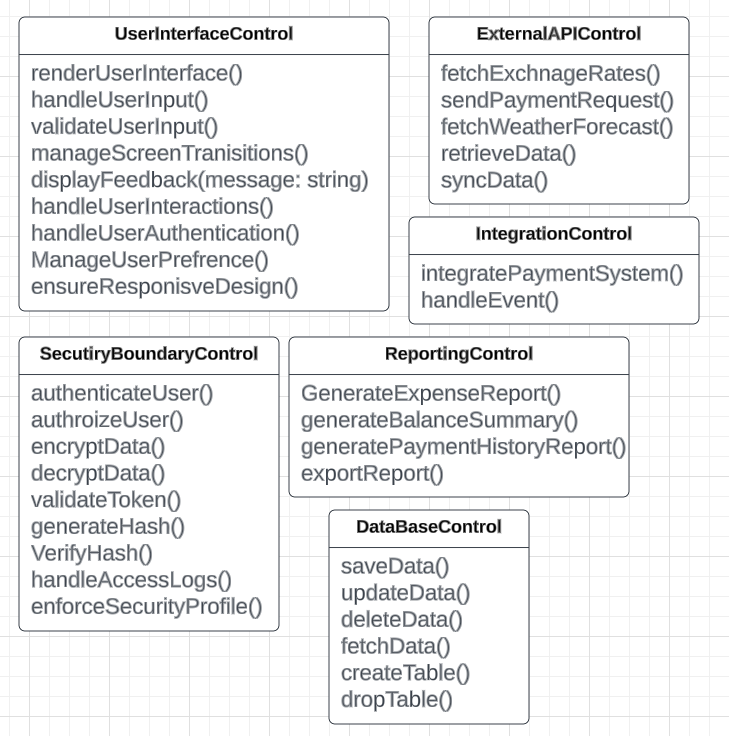
The logical view of SplitSmart emphasizes the organization of classes, packages, and subsystems that collectively contribute to the app's functionality and behavior. Class diagrams and other visual representations can be included to illustrate the relationships between architecturally significant classes, subsystems, packages, and layers within the app's logical architecture.



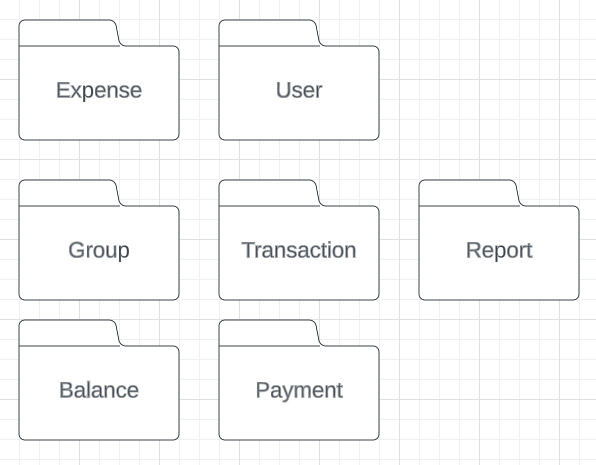
## Logical view



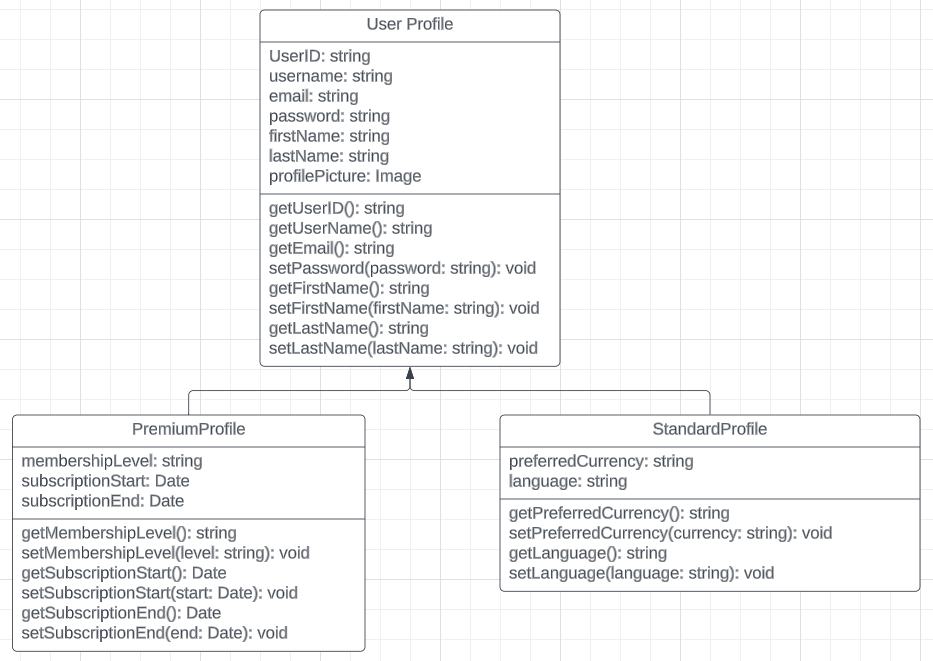
## Presentation Package



## Application Package

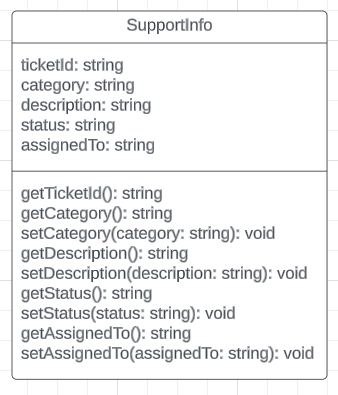


## Domain Package



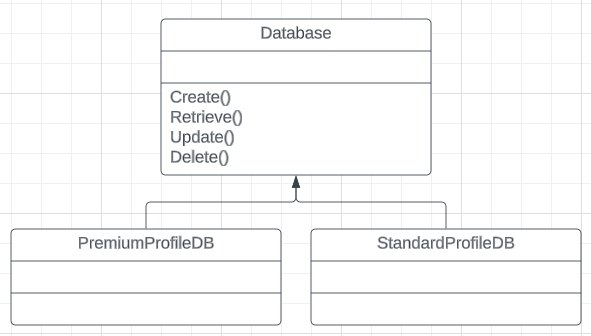
## Profile Package

* For the *Subscribe Package* see the above profile package for **PremiumProfile**

****

## Support Package

* Our support will be on the application of SplitSmart for customers and users to submit tickets for our team to follow up on.



## Persistence Package

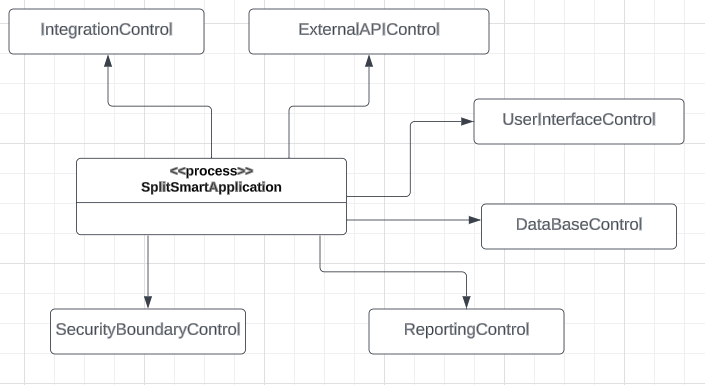
# 6. Process View

* The server process view represents the main server-level functions of SplitSmart.
* It is responsible for handling incoming client requests, coordinating the processing of various application functions, and managing the overall system operation.
* The server process interacts with other components of the system to fulfill client requests and provide the required services.

Within the server process, multiple threads or lightweight processes can be employed to handle different application functions. These threads are responsible for executing specific tasks and interacting with relevant components. The specific application functions and their associated threads can be identified based on the system requirements and functionalities defined in earlier sections of the document.

The communication between the server process and the threads can be achieved through various modes, such as message passing or method invocations. Synchronization mechanisms, such as locks or semaphores, may be employed to ensure proper coordination and thread safety.

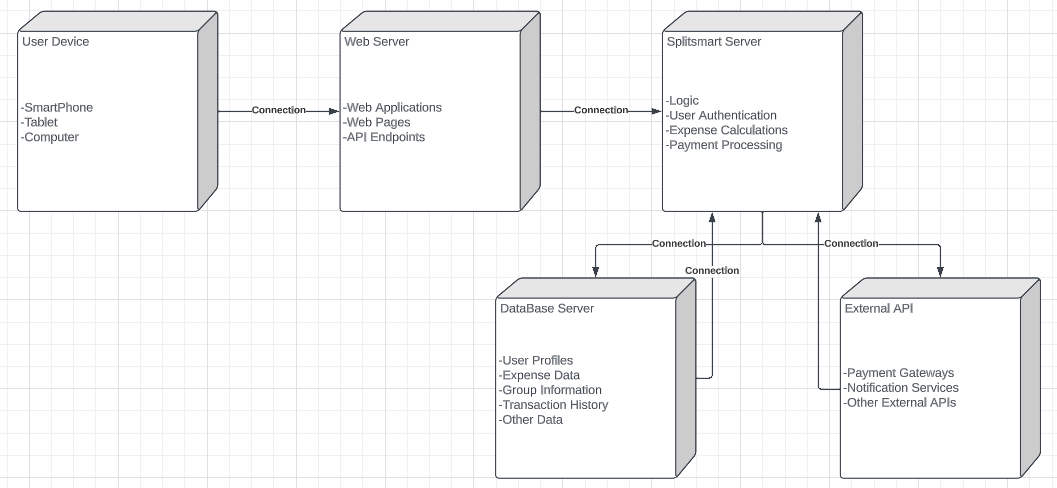
Overall, the process view illustrates the high-level organization of processes within SplitSmart and highlights the server process as the central component responsible for managing the system's functions and coordinating interactions with other components and threads.



# 7. Deployment View

The deployment view of SplitSmart involves a client-server architecture, where users access the system through various devices such as smartphones, tablets, laptops, or desktop computers equipped with modern web browsers or mobile applications. The web server acts as an intermediary, hosting the SplitSmart application and facilitating communication between the user devices and the SplitSmart server. The SplitSmart server, residing in the backend, handles the core functionalities including business logic, user authentication, and expense management, while the database server stores and manages the persistent data required by the system. External APIs play a crucial role by integrating external services like payment gateways and notification services. The successful deployment of SplitSmart relies on the proper configuration and interconnection of these components, ensuring a reliable and efficient system that caters to the needs of users.

In the SplitSmart deployment, the web server serves as the bridge between user devices and the SplitSmart server, while the database server stores and manages the system's data. The client devices, ranging from smartphones to desktop computers, allow users to access SplitSmart's features and functionalities. External APIs play a vital role in integrating services like payment gateways and notification systems. By orchestrating these components in a well-configured and interconnected manner, SplitSmart ensures seamless user experience and efficient system operations.



# 8. Implementation View

The server software is organized into a single layer, ensuring a cohesive and streamlined architecture. The browser client serves as the primary access layer, allowing users to interact with SplitSmart through a web browser interface.

# 9. Size and Performance

SplitSmart aims to provide a responsive and seamless user experience, ensuring that users can interact with the application without significant delays or disruptions. This includes quick loading times for pages, smooth navigation, and efficient processing of user requests. Performance optimization techniques, such as caching, database indexing, and efficient query management, may be implemented to enhance system performance.

The software architecture of SplitSmart is designed to handle a significant user load, supporting a large number of concurrent users. It is built to efficiently accommodate up to 25,000 concurrent users accessing the system simultaneously

It's important to note that the actual size and performance of SplitSmart can be influenced by factors such as server hardware, network infrastructure, database management, and code efficiency. To accommodate a larger user base or increasing usage demands, additional resources and scaling strategies can be implemented, such as load balancing, horizontal scaling, or cloud-based infrastructure.

# 10. Quality

SplitSmart upholds high standards of design, usability, and reliability. While it may have similarities in terms of user interface principles and graphical standards with other applications, SplitSmart has its own distinct design and user experience tailored specifically for managing expenses and balancing accounts. The software provides an intuitive and user-friendly interface that allows users to easily navigate through features and functionalities. SplitSmart prioritizes data accuracy, consistency, and security to ensure reliable financial management and transparent expense tracking for its users.