***Software Requirements Specification - SplitSmart***

# Table of Contents

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

[**1.1 Introduction 3**](#_3r3bctdfc9am)

[1.2 Goals and Objectives 3](#_1fob9te)

[1.3 System Statement of Scope 3](#_3znysh7)

[1.4 System Context 6](#_3dy6vkm)

[1.5 Major Constraints 7](#_1t3h5sf)

[**2.0 Usage Scenario 8**](#_4d34og8)

[2.1 User Profiles 8](#_2s8eyo1)

[2.2 User Stories 8](#_17dp8vu)

[2.3 Special usage scenarios 9](#_1xa59jc434kx)

[**3.0 Functional Requirements 10**](#_cbymorpd6qvb)

[3.1 High priority requirements 10](#_6jbx72xscvur)

[3.2 Medium Priority Requirements 11](#_soincs8st4or)

[3.3 Low Priority Requirements 11](#_kql0t7t1mb1f)

[**4.0 Non Functional Requirements 13**](#_rfi6znnv9skf)

[4.1 Reliability 13](#_u4ywkd7s9elp)

[4.2 Usability 13](#_nn4aqzmupga4)

[4.3 Performance 13](#_mssnavcckohh)

[4.4 Security 13](#_lfxlx6xdivtr)

[4.5 Accessibility 13](#_nsi2l8fiudui)

[4.6 Compatibility 13](#_9oormkiiynsz)

[4.7 Maintainability 13](#_z2cpk1lbj6tp)

[4.8 Data Backup and Recovery 13](#_wbtkghdo7cu7)

[4.9 Monitoring and logging 14](#_l8o2truun4p6)

[**5.1 Functional Model and Description 15**](#_3j2qqm3)

[5.1.1 Description for Function 1 15](#_l9a1icrjifzo)

[5.1.2 Description for Function 2 16](#_ghodpw21i4oz)

[5.1.3 Description for Function 3 17](#_4iw8atrm3ehf)

[5.1.4 Description for Function 4 18](#_ssnwqdcwqz73)

[5.1.5 Description for Function 5 19](#_omnmv2eednxt)

[5.2 Software Interface Description 20](#_654fgn10p05n)

[5.3 Use case Diagrams 20](#_wexlvxlno9zb)

[**6.0 Restrictions, Limitations and Constraints 21**](#_2bn6wsx)

[**7.0 Validation Criteria 22**](#_qsh70q)

[7.1 Classes of tests 22](#_nt7bwo2p2hjk)

[7.2 Expected Software Response 22](#_nk06j2f8eo6u)

[7.3 Performance Bounds 22](#_fgioc4aed6cq)

[**8.0 Appendices 24**](#_ceg7xif97jus)

[8.1 Product Strategies 24](#_qmjdeg7wex2b)

[8.2 Analysis Metrics to be used 24](#_wombzonjr2tp)

## 

# Introduction

## Goals and Objectives

The purpose of this document is to outline the software requirements for the SplitSmart application. SplitSmart is a tool that allows users to create and track expenses, as well as make payments to settle balances owed. It can be accessed through a web browser or a mobile app and is designed to facilitate expense sharing among friends, roommates, family members, or coworkers. The goals of the SplitSmart application are:

* Develop a user-friendly and intuitive expense-sharing application that allows users to easily create, track, and settle expenses within groups.
* Enable seamless collaboration among individuals or groups by providing features for creating and managing expense groups, inviting members, and assigning expenses.
* Provide accurate and real-time calculations of individual and group balances, ensuring transparency and accountability in expense sharing.
* Facilitate efficient splitting of expenses among group members, allowing users to specify the amount owed by each member and automatically updating balances accordingly.
* Implement secure user registration and authentication mechanisms to protect user data and ensure privacy.
* Enable users to make payments within the application to settle balances owed, providing convenient and reliable payment options.
* Provide comprehensive expense history and reporting features, allowing users to view and analyze spending patterns within a group.
* Ensure cross-platform compatibility by offering a web browser interface and mobile application for both iOS and Android devices.
* Prioritize data security by implementing encryption and adhering to best practices for storing and handling user data, including personal information and payment details.
* Conduct thorough testing and user acceptance testing to ensure that the application functions correctly, is user-friendly, and meets the needs and expectations of the target users.

These goals and objectives serve as a guide to ensure that the development and implementation of the SplitSmart application align with the intended purpose and desired outcomes.

## System Statement of Scope

* + 1. **General Requirements**

General requirements for our SplitSmart application are listed on the next page of the document:

1. **Expense management:**
   1. Allow users to create and categorize expenses, including description, amount, and date.
   2. Provide options for adding tags or labels to expenses for easy organization and filtering.
   3. Enable users to edit or delete expenses as needed.
   4. Implement search functionality to quickly locate specific expenses.
2. **User authentication and security:**
   1. Implement a secure user registration and login system.
   2. Use encryption techniques to protect user data and sensitive information.
   3. Apply secure authentication protocols to prevent unauthorized access.
   4. Regularly update and patch security vulnerabilities to ensure data safety.
3. **Real-time balance calculations:**
   1. Keep track of individual and group expenses in real-time.
   2. Automatically calculate and update balances owed between users.
   3. Handle complex scenarios such as splitting bills unequally or settling debts partially.
4. **Payment integration:**
   1. Integrate with payment gateways or third-party payment platforms.
   2. Allow users to link their bank accounts or credit cards for seamless payment transactions.
   3. Provide options for manual entry of payments made outside the app.
   4. Facilitate payment transfers between users to settle balances.
5. **Group expense tracking:**
   1. Enable users to create groups and invite members to join.
   2. Allow group members to add expenses that are shared among the group.
   3. Display a consolidated view of group expenses and balances.
   4. Provide features for managing group settings, such as adding or removing members.
6. **Notifications and alerts:**
   1. Send notifications to users for important events, such as new expenses, payments received, or group invitations.
   2. Provide reminders for pending payments or upcoming shared expenses.
   3. Allow users to customize their notification preferences.
7. **Reporting and analytics:**
   1. Generate detailed expense reports for users to track their spending habits.
   2. Provide visualizations and charts to help users analyze their expenses.
   3. Offer filters and sorting options for customized report generation.
   4. Present insights and trends based on historical data

* **Interface enhancements**

Developers are adamant on creating an easy to use GUI for every user to be able to understand and use. It will allow for all sizing of every element that is in the Augmented room.

* **Database Administrative Interface**

No interface will be available differently for administrative purposes.

* **Online Help**

A help menu will be developed for the UI when pausing the tool with a menu for Quitting, Saving, Help, Etc.

* **Training**

Will be available on the tool with an option to turn on guides and tips that are shown all in augmented reality.

Any help necessary with the tool or bugs found can be answered or reported on the tool when being used.

* + 1. **Extended Enhancement**

While there has not been any extra enhancements that have been added to the development of the tool yet. Our developers have come up with some other enhancements that may be interesting for the SplitSmart tool overall. Below are the listed enhancements:

**Multi-platform accessibility:**

* Develop a standalone mobile application for SplitSmart compatible with popular operating systems such as iOS and Android.
* Create a web-based version of SplitSmart accessible through standard web browsers.
* Ensure seamless synchronization of data across all platforms to provide users with a consistent experience.

**Smart notifications and reminders:**

* Implement intelligent notifications based on user preferences and spending patterns.
* Send reminders for upcoming shared expenses, payment due dates, or potential savings opportunities.
* Utilize machine learning algorithms to offer personalized insights and financial recommendations.

**Integration with banking and financial services:**

* Enable direct integration with users' bank accounts or financial institutions to fetch transaction data.
* Implement automatic expense tracking by importing and categorizing transactions.
* Provide real-time bank balance updates to help users monitor their financial status

## System Context

SplitSmart is a comprehensive expense sharing and management application that operates within a broader system context. It interacts with various external entities and systems to deliver its functionalities effectively. The system context of SplitSmart includes the following components:

* Users:
  + Individual users: These are the primary users of SplitSmart, including friends, roommates, family members, or coworkers who share expenses.
  + Group users: SplitSmart allows users to form groups and share expenses within those groups, enabling collaborative expense tracking and settlement.
* Third-Party Financial Institutions:
  + Banks and financial institutions: SplitSmart integrates with external banking and financial services to fetch transaction data, provide real-time balance updates, and facilitate payment transfers.
  + Payment gateways: The application interfaces with secure payment gateways to enable seamless payment transactions and settle balances owed between users.
* Data Storage and Cloud Services:
  + Database servers: SplitSmart utilizes databases to store user profiles, expense data, shared group information, and transaction records securely.
  + Cloud storage: The application utilizes cloud-based storage services to store receipt images, ensuring accessibility and data backup for users across platforms.
  + Synchronization services: SplitSmart employs synchronization services to ensure seamless data synchronization across different devices and platforms, enabling users to access their expenses from anywhere
* User Devices:
  + Mobile devices: SplitSmart operates on various mobile devices such as smartphones and tablets, allowing users to access the application through a dedicated mobile app.
  + Web browsers: The application is accessible through standard web browsers on desktops and laptops, providing flexibility in accessing SplitSmart's features.
* SplitSmart Administration:
  + Administration interface: SplitSmart includes an administration interface for system administrators to manage user accounts, perform maintenance tasks, and ensure the overall system's smooth operation.

## Major Constraints

**Hardware limitations**

One of the primary constraints is the hardware limitations of the devices used to run the SplitSmart application. The tool must be optimized to run smoothly on a variety of devices, including smartphones, tablets, and PCs, and it must be designed to work within the constraints of these devices' processing power, memory, and battery life.

**Data security**

Data security is a critical constraint for any tool that stores user data, including the virtual rooms created by users. The SplitSmart application must be designed with robust security features to protect user data from unauthorized access or theft.

**Funding**

The cost of developing and deploying the SplitSmart application can be a significant constraint, especially for smaller organizations or startups. The tool must be designed to minimize development and deployment costs, while still delivering a high-quality user experience.

**Scalability and performance**

As the user base of SplitSmart grows, the system needs to handle increased data volumes, user interactions, and concurrent requests without compromising performance. Scalability measures should be in place to ensure smooth operation under varying loads.

# Usage Scenario

## User Profiles

There will be Four levels of users for the tool:

* + - Individual Users (Read/Write)
    - Group Members (Read/Write/Modify)
    - Group Administrators (Full Access)
    - Administrative Personnel

## User Stories

**Individual Users: (Read/Write)**

Individual users are those who primarily use the app for personal expense management. They create their own user accounts and track their own expenses, balances, and payments within the app. They may use the app to manage their personal finances, track shared expenses with friends or family, or record their own transactions.

**Group Members: (Read/Write/Modify)**

Group members are users who participate in shared expenses within a group. They create user accounts and join specific groups within the app. Group members can create and manage expenses within the group, view balances with other group members, and contribute to settling shared expenses. This profile is suitable for roommates, colleagues sharing expenses, or friends planning a trip together.

**Group Administrators (Full Access)**

Group administrators are users who have additional privileges within a group. They create and manage groups, invite or remove group members, and oversee the expense management process within the group. Group administrators ensure smooth group coordination and manage group-specific settings. This profile is suitable for individuals responsible for managing shared expenses within a group.

**Administrative Personnel: (All Access)**

This level is only for administrators (creators) of the tool itself. This allows Administrators to do everything the other role can, while allowing for changes in user profiles as well as access to the backend/frontend of the software itself.

Administrative Personnel may be individuals responsible for managing the SplitSmart Application and its associated database. They may require access to advanced analytics and reporting tools to track usage patterns, identify popular features, and monitor performance metrics. They may also require tools to manage user accounts, permissions, and data security

## 2.3 Special usage scenarios

**Roommates expense sharing:** SplitSmart allows roommates to effortlessly track and settle shared expenses such as rent, utilities, and groceries, ensuring a fair and transparent distribution of financial responsibilities within the household.

**Group Trips/Vacations:** SplitSmart simplifies expense management by enabling users to create a dedicated group, record and split expenses like accommodation, transportation, and dining, making it easy to track balances and settle costs among the trip participants.

**Collaborative expenses:** SplitSmart's group functionality allows colleagues or team members to manage shared expenses like equipment purchases, travel costs, and shared subscriptions, promoting financial transparency, accurate tracking, and convenient payment settlements.

**Event planning/Celebrations**: SplitSmart streamlines expense tracking and coordination by providing a dedicated group for organizing expenses related to parties, weddings, or group outings, ensuring efficient financial management and effortless cost sharing among participants.

**Family Expense Sharing:** SplitSmart facilitates financial coordination among family members by creating a family group, enabling seamless tracking of shared costs like groceries, bills, and family outings, empowering families to stay organized and efficiently manage their collective expenses.

# 3.0 Functional Requirements

## 3.1 High priority requirements

* + - **User Account Creation and Authentication:** Users should be able to create individual accounts and authenticate themselves securely to access their expense and balance information.
    - **Group Management:** The software should allow users to create and manage groups, enabling seamless tracking of shared expenses and balances within those groups.
    - **Expense Creation and Management:** Users should be able to create new expenses, including essential information such as amount, date, description, shared manner (equally split or split by %), and optional receipt image. They should also be able to specify which users were involved in the expense for appropriate balance adjustments.
    - **Approval Workflow:** The software should include a system for reviewing and approving expenses before they are added to the system and used to adjust balances. This could involve multiple levels of approval or a single review process, depending on the needs of the group.
    - **Notification System:** The software should have a notification system to alert users when other group members create new expenses or when there are important updates or reminders related to their groups.
    - **Balance Tracking:** The software should accurately track the balance owed by each user to every other user, considering all the expenses that have been entered. Users should be able to view their balances with other group members in real-time.
    - **Payment Tracking:** Users should be able to record when payments are made to settle balances owed, allowing the balances to be accurately reflected in the system. This feature helps maintain up-to-date and accurate balance information.
    - **Reporting and Summaries:** The software should provide users with comprehensive reports and summaries of their expenses, balances, and payments. This includes the ability to generate custom reports with specified date ranges and other parameters to facilitate expense analysis and financial planning.

## 3.2 Medium Priority Requirements

* + - **User Profile Customization:** Users should have the ability to customize their profiles by adding personal information, profile pictures, and other relevant details to enhance their user experience.
    - **Expense Categorization:** Users should be able to categorize expenses into different predefined or custom categories (e.g., food, transportation, entertainment) for better expense management and tracking.
    - **Split Method Flexibility:** The software should provide flexibility in choosing the split method for expenses, allowing users to select options beyond equal splits or percentage splits. This could include options like splitting expenses based on specific amounts or specific individuals.
    - **Commenting and Communication:** Users should be able to add comments or notes to individual expenses, facilitating communication and clarification within the group regarding specific expenses or transactions.
    - **Multi-Currency Support:** The software should support multiple currencies, allowing users to record and track expenses in different currencies, particularly beneficial for users involved in international transactions or traveling.
    - **Reminders and Notifications**: Users should receive reminders and notifications for pending approvals, upcoming payments, or approaching expense due dates, ensuring timely actions and reducing the likelihood of missed or delayed expenses.
    - **Search and Filter Functionality:** Users should be able to search for specific expenses or apply filters to view expenses within a specific date range, category, or other criteria, facilitating efficient expense retrieval and analysis.
    - **Data Export and Import:** Users should have the ability to export their expense data or generate reports in various formats (e.g., PDF, CSV) for external use or personal record-keeping. Additionally, the software should allow users to import expense data from external sources for easy migration or consolidation of financial information.

## 3.3 Low Priority Requirements

* + - **Multiple Language Support:** The software could offer support for multiple languages, allowing users to switch between different languages for the user interface and communication within the app.
    - **Data Backup and Restore:** Implementing a feature to enable users to back up their expense data and restore it in case of device loss, data corruption, or other unforeseen circumstances.
    - **Expense Splitting Equally Among Selected Users:** Providing an option to split an expense equally among a selected group of users, instead of splitting it among all group members, for more granular expense allocation.
    - **Integration with Expense Tracking Tools**: Allowing integration with popular expense tracking tools or services to import expenses directly into the SplitSmart app for streamlined data management.
    - **Budgeting and Spending Limits:** Incorporating budgeting features to help users set spending limits for specific categories or time periods, providing visual indicators or alerts when nearing or exceeding those limits.
    - **Recurring Expenses:** Supporting the creation of recurring expenses, such as monthly rent or subscription payments, with automated reminders and pre-filled expense details for convenience.
    - **Expense Attachments:** Enabling users to attach documents or additional files related to expenses, such as invoices or receipts, for better record-keeping and reference purposes.
    - **Integration with Payment Gateways:** Integrating with popular payment gateways to facilitate direct payment settlements within the app, allowing users to easily settle balances by initiating payments from within SplitSmart.

# 4.0 Non Functional Requirements

### 4.1 Reliability

* The software should have a high uptime percentage, with minimal service disruptions or downtime.
* It should handle concurrent user requests without system crashes or data inconsistencies.

## 4.2 Usability

* The user interface should be intuitive, with clear navigation and easily understandable labels and controls.
* The software should provide helpful tooltips, contextual guidance, and error messages to assist users in performing tasks accurately.

## 4.3 Performance

* The application should load and respond quickly to user interactions, providing a seamless and smooth user experience.
* It should efficiently process complex calculations and handle large volumes of data without noticeable delays.

## 4.4 Security

* The software should use encryption algorithms to protect user data during transmission and storage.
* It should implement strong user authentication mechanisms, such as two-factor authentication or biometric authentication, to prevent unauthorized access.

## 4.5 Accessibility

* The software should adhere to accessibility standards, such as WCAG 2.0, to ensure compatibility with assistive technologies and support for users with disabilities.
* It should provide alternative text for images, keyboard navigation options, and color contrast enhancements for improved accessibility.

## 4.6 Compatibility

* The software should be compatible with popular web browsers, such as Chrome, Firefox, and Safari, across different versions.
* It should support multiple platforms, including desktop, mobile, and tablet devices, with consistent functionality and user experience.

## 4.7 Maintainability

* The code should follow best practices, be well-documented, and use modular design patterns for ease of maintenance and future enhancements.
* The software should have a clear separation of concerns, allowing for efficient bug fixes and updates without impacting the entire system.

## 4.8 Data Backup and Recovery

* Regular automated backups of expense data should be performed to prevent data loss and allow for recovery in case of system failures or errors.
* The software should provide options for users to manually back up their data and restore it when needed.

## 4.9 Monitoring and logging

* The software should log important system events, user activities, and error messages for effective troubleshooting and system optimization.
* It should include performance monitoring tools to track response times, resource utilization, and identify performance bottlenecks for proactive maintenance and improvements.

# 5.1 Functional Model and Description

## 5.1.1 Description for Function 1

**Use case name**

**Use Case Name:** Save Changes

**Description:** Users will have separate databases where their individual and shared expenses, account balances, connected friends, social media accounts and settings will be stored.

**Actors**

**User:** The person who wants their information to be stored on a database when they sign out.

**Preconditions**

The user has a stable internet connection and there are no conditions that will close the browser window (such as power outage, malware or scheduled update)

**Triggers**

As the user clicks on “{username} > Sign Out”, changes will be saved into a database.

S**cenario Description**

1. Users perform the changes when they log into their account.
2. The user clicks on their name (in other words {username}).
3. From the pop-up menu, the user clicks on “Sign Out”.
4. A window pops up asking “Are you sure you want to save these changes?”
5. The user clicks “Yes”.
6. Web application saves the changes into a database
7. Web application redirects to the login page.

**Post conditions**

* The user can log back into the web application and continue where they left.
* The user can also sign in on other devices compatible with the web application.

**Exceptions**

* If the web browser unexpectedly closes down, the user has to start from previous changes saved.
* If the server is down, developers have to recover as much data as possible.
* If the server data was corrupted because of a hack, developers have to set up a new server and release a patch for users.

## **5.1.2 Description for Function 2**

**Use case name**

**Use Case Name:** Add Expense

**Description:** The user will create an expense and add who they will share that expense with. Based on the number of people who share the expense, the expense by default will automatically be divided into the number of people for that expense. Those who will be part of that expense will be immediately notified when the expense was created.

**Actors**

**Principal User:** The user is the one who creates the expense.

**Shared Users:** Those who will be sharing the expense with the principal user.

**Preconditions**

The user was logged into their browser and the connection is open.

**Triggers**

User clicks on the “Create Expense” icon.

**Scenario Description**

* + - * “Create Expense” window pops up .
      * The user enters the name of the event, the date of the event, who they will share it with and how much each will share.
      * For sharing expenses there will be a sliding bar where users can set up the event.
      * The user will then click on the “Create” button and an expense will be created.

**Post conditions**

* The event is created and the user is able to see the event on their home screen.
* Shared users are also able to see the event.
* Shared users are also notified about the event.

**Exceptions**

* If the user fails to enter all required information (name of the event, the date of the event, who they will share it with and how much each will share), the window will show an error message and redirect to the “Add Event” window again.
* If the connection interrupts, the user shall start all over again.

## 5.1.3 Description for Function 3

**Use case name**

**Use Case Name:** Add Group

**Description:** The user will create a group for upcoming events and each group participant shall be notified about the release of new events and changes to upcoming events (such as canceling or rescheduling).

**Actors**

**Principal User:** The user is the one who creates the group.

**Shared Users:** Those who will be part of the group created by the principal user.

**Preconditions**

The user logged in to their account and there is a stable internet connection.

**Triggers**

The user clicks on the “Create Group” icon.

**Scenario Description**

* + - * A window form pops up and asks for the name of the group, the participants and what type of group it will be.
      * The user clicks “OK” to create the group.

**Post conditions**

* After the group has been created, participants shall be notified about the group`s creation.

**Exceptions**

* The group will not be created if the group name already exists.
* If the group was created with the intention of a business purpose to avoid taxes.

## 5.1.4 Description for Function 4

**Use case name**

**Use Case Name:** Add Balance

**Description:** The user will be able to manage the allocated budget for all events.

**Actors**

**User:** The one who will create a budget for all events.

**Preconditions**

The user already created an account and is logged in.

**Triggers**

The user clicks on their name and a menu pops up. Then the user clicks on their budget in numbers.

**Scenario Description**

* + - * The user clicks on the “+” sign to increase their budget.
      * “Add Balance” window pops up.
      * The user enters the new budget into a box. (Ex. $3500)
      * The user clicks on “Set Budget”.

**Post conditions**

* A new balance has been set up. The user can allocate that budget for various events.

**Exceptions**

* If the budget was set up by an administrator, the user cannot change the previously set up budget. (The administrator can be their family member or their supervisor).

## 5.1.5 Description for Function 5

**Use case name**

**Use Case Name:** Add Friend

**Description:** The user will enter the name of their friend and if they find the person they will become friends.

**Actors**

**User:** The one who searches for the person.

**Preconditions**

The user already logged in to their account.

**Triggers**

The user clicks on the search bar and starts to enter the name of the user.

**Scenario Description**

* + - * If the user's name pops up the user will click on their name.
      * The user then clicks on “Add Friend”.

**Post conditions**

* If the other user who received the invitation responds as “Accept Invitation”, they become friends in the web application.

**Exceptions**

* If the other user denies the invitation, then the user cannot add the other user to the friends list.
* The other user also has a block option. If blocking was enabled the user can neither search for the user nor add them as a friend.

## 5.2 Software interface description

**External machine interfaces**

* Touch screen mobile device or tablet with access to the internet.
* Desktop computer or laptop with access to the internet.
* Smart Television with access to the internet.

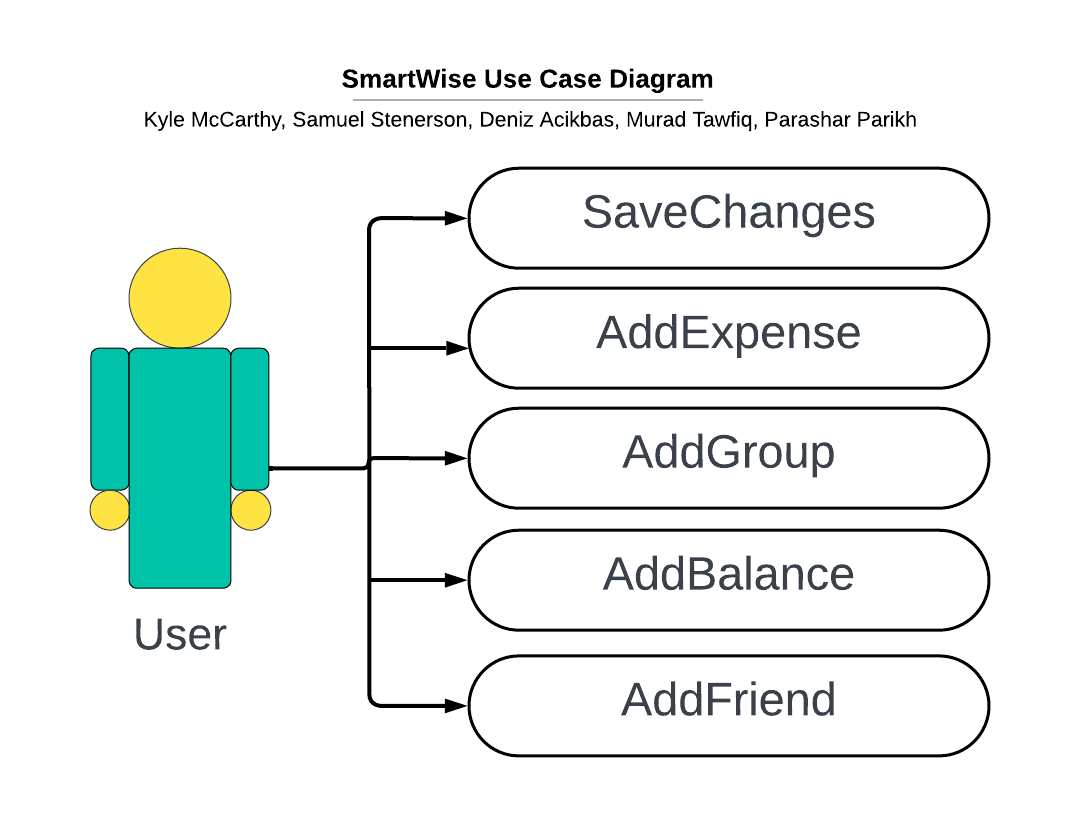
**External system interfaces**

* Cloud-based storage for saving data
* Online inventory database for storing events, groups, user data and their settings.

**Human interface**

* Graphical user interface for the budget management web application
* Touchscreen-based interaction that does the same purpose with GUI.

## 5.3 Use case Diagrams



## 

# 6.0 Restrictions, Limitations and Constraints

**Restrictions:**.

**Data Privacy:** The app must adhere to relevant data protection and privacy regulations to ensure the privacy and security of user data. This includes implementing appropriate data encryption, access controls, and user consent mechanisms to protect sensitive information.

**Third-Party Integrations Dependencies:** The functionality of the app may rely on integrating with third-party services, including payment gateways and receipt processing APIs. It is important to note that the availability, reliability, and compatibility of these services are external factors that could potentially impact the overall performance and functionality of the app.

**Limitations:**

**Platform Compatibility:** The app needs to be developed and optimized for multiple platforms, such as iOS and Android, which may have different technical requirements and constraints. Ensuring consistent functionality, user experience, and performance across these platforms can be a challenge.

**Network Connectivity:** SplitSmart relies on a stable internet connection to synchronize data, provide real-time updates, and enable collaborative expense management. Users may experience limitations or disruptions in functionality when operating in areas with poor network coverage or in situations with limited or no internet access.

**Constraints:**

**Budget and Time Constraints:** The development and maintenance of SplitSmart must be completed within specified budget and time constraints. Balancing feature development, quality assurance, and timely delivery can be a challenge, requiring effective project management and resource allocation.

**Device Compatibility and performance:** The app should be designed to work efficiently on a range of devices with different specifications and screen sizes. However, limitations in device capabilities, such as processing power, memory, or screen resolutions, may affect the app's performance and user experience on certain devices.

**Localization and Internationalization:** To cater to a global user base, the app may require localization and internationalization efforts to support multiple languages, regional date and currency formats, and cultural variations in expense management practices. These factors may introduce additional complexity and constraints during development.

# 7.0 Validation Criteria

## 7.1 Classes of tests

1. **Functional Testing**: Verify that the application performs as expected, including account creation and login, group creation and management, expense entry and approval, balance updates, payment recordings, and report generations.
2. **Usability Testing:** Evaluate the user interface to ensure ease of use and effective/clear navigation..
3. **Compatibility Testing:** Test the application on various types of mobile touch screen devices and operating systems to ensure multi-platform compatibility.
4. **Performance Testing:** Measure the application’s response time, memory utilization, and load times.
5. **Security Testing:** Verify the user data and projects are protected from unauthorized access.
6. **Integration Testing:** Test integration with external systems, such as notification systems and potential future payment gateways.

## 7.2 Expected software response

* User account created and able to log in.
* Group created with specified members and able to manage (add/remove members).
* Proper creation and approval process of expenses.
* Accurate tracking and updating of balances following an expense approval or payment recording.
* Successful generation of reports regarding expenses, balances, and payments.
* Software is compatible with various types of devices, web browsers, and operating systems.
* Errors and exceptions are handled properly.

## 7.3 Performance bounds

1. **Response Time:** The application should respond to user input promptly to ensure a smooth user experience.
2. **Memory Usage:** The application should not consume excessive memory resources on devices.
3. **Loading Time:** The application should load promptly (< 5 s), and individual expense or report loading should not be excessive (<10 s).
4. **Network Latency:** Interaction with external systems such as notification systems and potential future payment gateways should have minimal latency.
5. **Scalability:** The application should be able to handle an increasing number of users, groups, and expenses without significant and noticeable performance impact.

# 8.0 Appendices

## 8.1 Product Strategies

1. **Target Market**: Individuals and groups seeking an intuitive solution for managing shared expenses. The target audience ranges from roommates, friends, and families to coworkers, travel groups, and event organizers.
2. **Competitive Advantage:** SplitSmart stands out with its user-friendly interface, efficient expense tracking, and dynamic group management features. The application's notification and approval system provides an added layer of transparency, enabling users to handle their shared expenses effectively.
3. **Potential pricing deals:** Offer a freemium model where basic features are available for free, such as expense tracking, group management, and basic reporting. For premium subscribers, additional features can be included such as advanced reporting, receipt image attachment, priority customer support, and ad-free experience.
4. **Distribution**: SplitSmart will be released on major app stores such as Google Play Store and Apple App Store for easy access and download. It can also be made accessible through a web application for those who prefer to use it on a laptop or desktop. Marketing strategies will encompass social media promotion, online advertisements, and partnerships with relevant communities or platforms.

## 8.2 Analysis metrics to be used

1. **Addressed Requirements Coverage**: The percent of requirements addressed by the software.
2. **Addressed Use Case Coverage**: The percentage of use cases covered by the test cases.
3. **Defect Occurrence:** The number of defects identified during testing efforts.
4. **Test Case Effectiveness:** The percentage of test cases that identify defects or exceptions in the software.
5. **Code complexity:** Measure the complexity of the software code; assess scalability and maintainability.
6. **User satisfaction:** Feedback provided by users on various app stores, as well as through in-app feedback mechanisms. This can be used to gauge satisfaction regarding the application's performance, usability, and functionality.
7. **Active User Count:** Track the number of daily and monthly active users to measure the application's user engagement and growth rate.
8. **Conversion Rate:** The percentage of free users who upgrade to the premium version, helping to assess the effectiveness of the freemium model.
9. **Expense Tracking Usage:** The frequency and volume of expense entries made by users, indicating the core feature's usage.
10. **Report Generation Usage:** Monitor how often users generate reports, which provides insight into the usefulness of this feature.
11. **Report Generation Usage:** Monitor how often users generate reports, which provides insight into the usefulness of this feature.