Software Requirements Specification

for

Bill Splitting Application

Version 1.0

Prepared by, Akash C and Anuraag A Srivatsa

PES1UG22AM016 & PES1UG22AM026

PES University - 5A

2 Sep 2024

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

This SRS describes the software functional and nonfunctional requirements for release 1.0 of the Bill Splitting Application. This document is intended to be used by the members of the project team that will implement and verify the correct functioning of the system. Unless otherwise noted, all requirements specified here are high priority and committed for release 1.0.

1.2 Document Conventions

The below describes conventions used in structuring the document

Bold 18 Times : Section Heading **[Sample Content]**

Bold 14 Times : Subheading **[Sample Content]**

Font 11 Arial: Content [Sample Content]

Asterix (*) can be used to add more information about certain groups under a Subheading

1.3 Intended Audience and Reading Suggestions

SRS is intended for developers, project managers, marketing staff, users, testers, and documentation writers.

Readers new to the project should begin with Section 1 for an overview.

Developers should read Sections 3 and 4 for technical requirements.

Project managers should read Section 2 for scope, schedule, and budget information.

Marketing staff should read Sections 1 and 4 to understand product features and benefits.

Users should read Sections 1 and 4 to learn how the software works and how to use it.

Testers should read Sections 3 and 4 to develop test cases.

Documentation writers should read the entire SRS for all requirements.

1.4 Product Scope

The Bill Splitting Application is a user-friendly and convenient tool designed to simplify the process of splitting bills fairly and efficiently among a group of people. It aims to reduce the hassle of manual calculations, ensuring a seamless and transparent experience for all participants.

1.5 References

1. PostgreSQL Database: PostgreSQL: The world's most advanced open source database

2. Frontend Framework: React

3. Full stack framework: Next.js by Vercel - The React Framework (nextjs.org)

3. Reference Application: Split expenses with friends. :: Splitwise

2. Overall Description

2.1 Product Perspective

The Bill Splitting Application (version 1.0) is a standalone software product designed to simplify and fairly split bills among groups. It is accessible via a user-friendly web interface and operates on various devices and platforms.
The product is Designed to be Simple and User friendly while also having a robust backend framework according to industry standards and specifications

2.2 Product Functions

- Add Bills
- Add Friends in application
- Split bills among friends evenly or by custom proportions
- Track individual credit with each friend
- Detailed History of all Splits created with every friend
- Send payment notifications
- Clear Pending Payments by attaching Payment reference

2.3 User Classes and Characteristics

Primary Users:

- **Bill Payers:** Individuals who regularly initiate bill splits and track expenses among their friends or group members.
- **Bill Sharers:** Individuals who participate in group bill splits and need to manage their individual balances and payments.

Secondary Users:

- **Accountants or Financial Advisors:** Professionals who may use the application to assist their clients with expense tracking and budgeting.
- **Event Planners:** Individuals who organize events or gatherings and need to manage group expenses efficiently.
- **Small Business Owners:** Entrepreneurs who use the application to track and split business-related expenses among team members or partners.

The most important class of users for the product will be the Primary Users which are the Bill Payers and Bill Sharers as they are frequent users of the application

2.4 Operating Environment

The Bill Splitting Application will operate in the following environment:

- Hardware Platform: Any device with a modern web browser
- Operating System: Any modern operating system (Windows, macOS, Linux, etc.)
- Web Browser: Any modern web browser (Chrome, Firefox, Safari, etc.)
- Standard Network Connection which can send and receive information via HTTP/S protocol
- Database: PostgreSQL 14 or later *
- Backend Framework: Node.js 18 or later *
- Frontend Framework: React 18 or later *

The application will be designed to be responsive and work well on all devices, from smartphones to desktops.

* These Are host requirements and not User Requirements

2.5 Design and Implementation Constraints

- All Code written will be labeled and version control will be implemented through git
- All Programs will be stored in GitHub
- The Database Engine will be powered by the latest PostgreSQL DB management System
- Frontend Will be powered with React (Next.js)
- Backend will be powered by Next.js' inbuilt node runtime

2.6 User Documentation

- User Documentation will be mentioned in the readme file maintained in the repository on GitHub
- Additionally user can refer to this manual to gain deeper insight on the product

2.7 Assumptions and Dependencies

Assumptions:

- The PostgreSQL database will be available and reliable.
- The Node.js and React frameworks will be stable and well-supported.

- The application will be hosted on a server with sufficient resources.
- The application will be compatible with all major web browsers.
- Users will have a basic understanding of how to use web applications.

Dependencies:

- The application depends on the PostgreSQL database for data storage.
- The application depends on the Node.js and React frameworks for its functionality.
- The application depends on a server with sufficient resources to host it.

3. External Interface Requirements

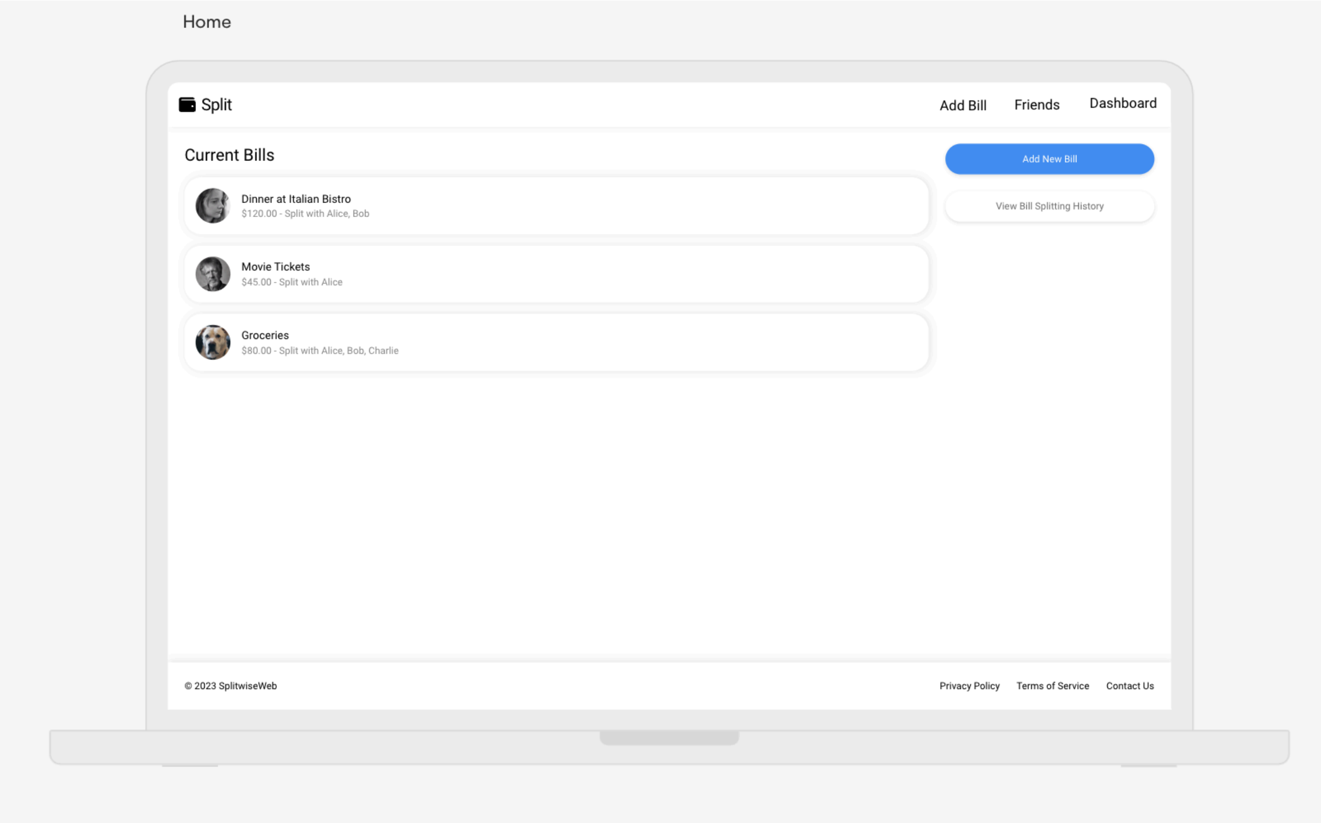
3.1 User Interfaces

3. External Interface Requirements

3.1 User Interfaces

The Bill Splitting Application will feature a user-friendly and intuitive interface that is consistent across all platforms and devices. The interface will be designed to be easy to use and understand, even for users who are not familiar with financial management applications.

Sample Screen Images

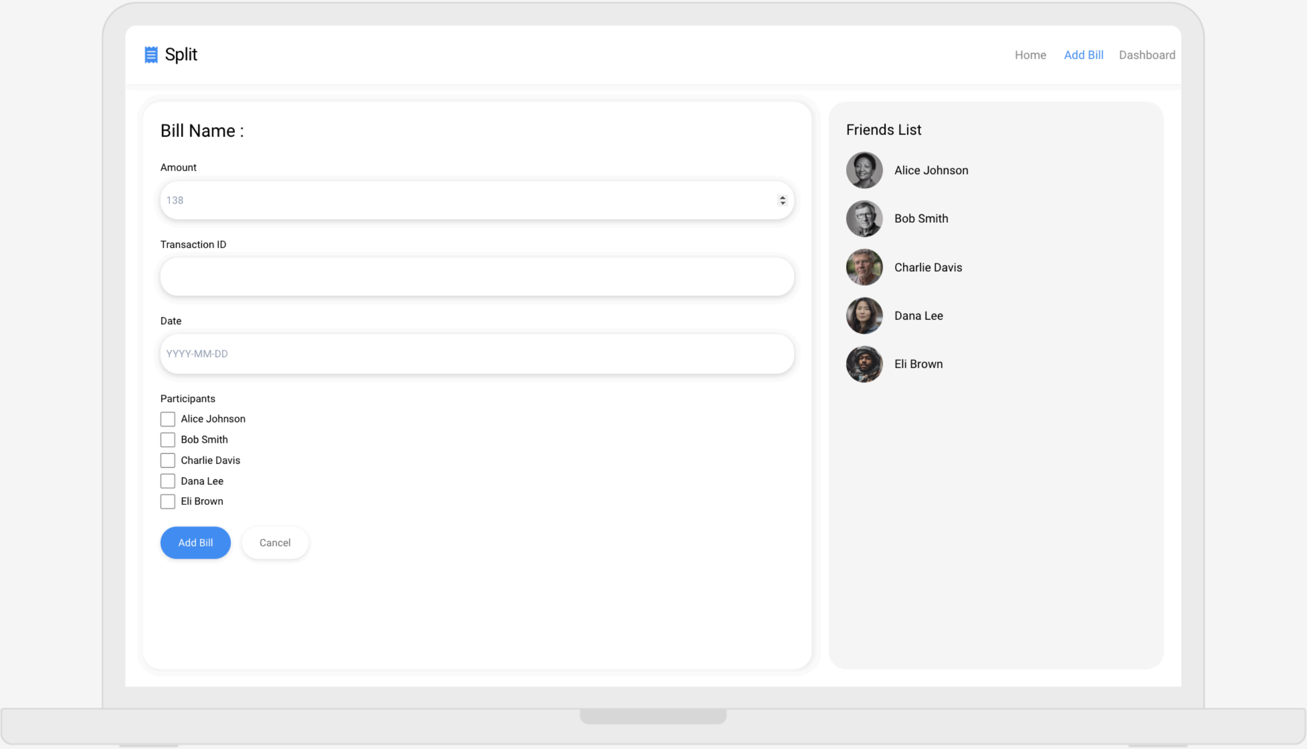


Dinner at Italian Bistro
$120.00 - Split with Alice, Bob

Movie Tickets
$45.00 - Split with Alice

Groceries
$80.00 - Split with Alice, Bob, Charlie

Add Bill



Bill Name :

Amount

138

Transaction ID

Date

YYYY-MM-DD

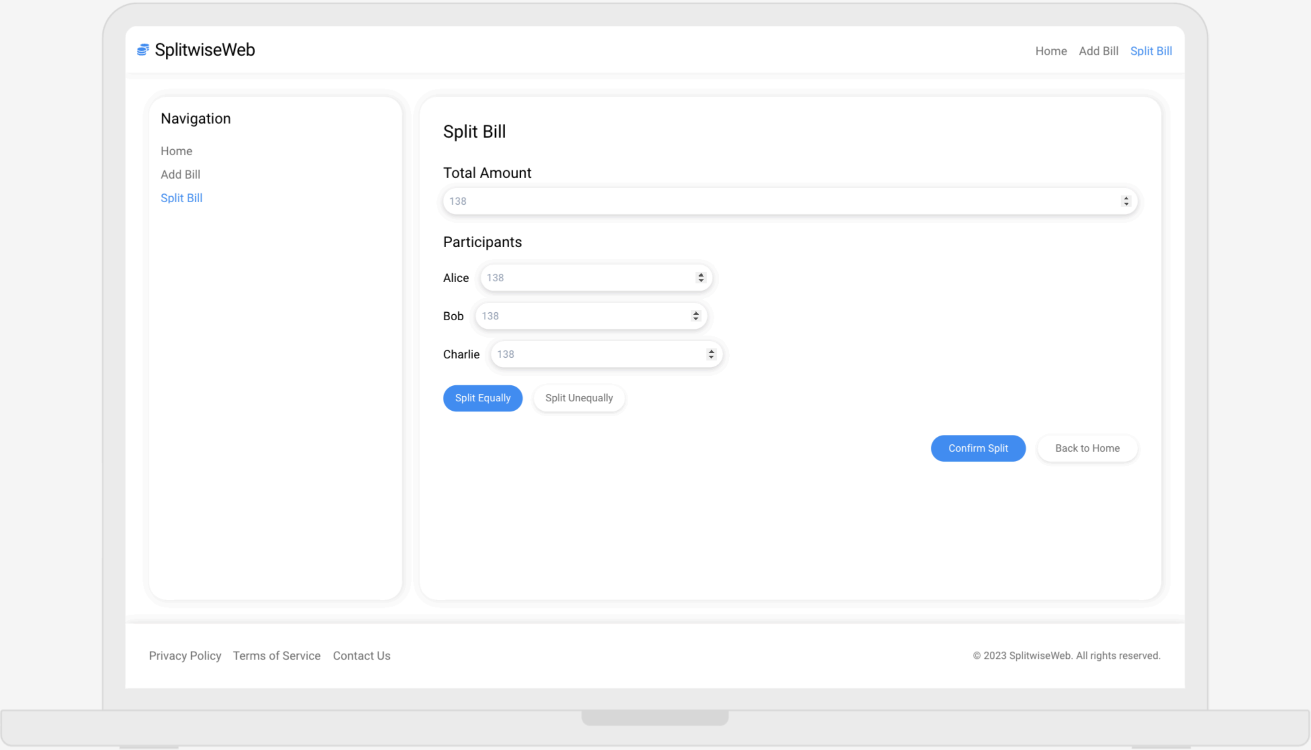
Participants

□ Alice Johnson
□ Bob Smith
□ Charlie Davis
□ Dana Lee
□ Eli Brown

Friends List

Alice Johnson
Bob Smith
Charlie Davis
Dana Lee
Eli Brown

Split Bill



Split Bill

Total Amount

138

Participants

Alice 138
Bob 138
Charlie 138

© 2023 SplitwiseWeb. All rights reserved.

GUI Standards and Product Family Style Guides

The application will use Tailwind, Aceternity and DaisyUI frameworks. This will ensure that the application has a consistent look and feel across all platforms and devices.

The following software components will require a user interface:

- **Bill Entry:** This component will allow users to enter bills into the application.
- **Friend Management:** This component will allow users to add and manage friends in the application.
- **Bill Splitting:** This component will allow users to split bills among their friends.
- **Payment Tracking:** This component will allow users to track payments on bills.

3.2 Hardware Interfaces

Host:

The product is a web Application which will hosted on Vercel

Vercel has 100 edge locations where data is stored in cloud based architecture

User:

User would require

1. Stable Internet Connection
2. Minimum of 1.4 GB RAM (Google Chrome recommended for 64 bit machines)
3. Web Browser

3.3 Software Interfaces

Connections to Other Software Components

- **Database:** PostgreSQL 14 or later
- **Backend Framework:** Node.js 18 or later
- **Frontend Framework:** React 18 or later

Data Items and Messages

- **Incoming Data:**
  - Bills from users
  - Payments from users
  - Friend requests from users
- **Outgoing Data:**
  - Bill summaries to users
  - Payment confirmations to users
  - Friend list updates to users

Services Needed

- **Database Storage:** PostgreSQL

- **Backend Processing:** Node.js
- **Frontend Rendering:** React

Communications

- **Database:** SQL queries and commands
- **Backend:** RESTful API calls
- **Frontend:** HTTP requests

Implementation Constraints

- **Data Sharing:** The data will be shared across software components using a RESTful API.

3.4 Communications Interfaces

Communications Requirements for the Bill Splitting Application

Web Browser:

- The application will be a web application that is accessed through a web browser.
- The application will be compatible with all major web browsers.
- The application will use HTTPS to encrypt all data transmitted between the client and the server.
- The application will use JWT token for user authentication

Electronic Forms:

- The application will use electronic forms to allow users to enter bills, payments, and friend requests.
- Electronic forms will be validated on the client side before being submitted to the server.

Data Transfer Rates:

- The application will be designed to handle a high volume of data traffic.
- The application will use a content delivery network (CDN) provided by Vercel to improve data transfer rates.

Synchronization Mechanisms:

- The application will use inbuilt features of PostgreSql for synchronization to ensure that data is consistent across all devices.

4. System Features

4.1 Expense Splitting

4.1.1 Description and Priority

The Expense Splitting feature allows users to split expenses among a group of participants. This is the **highest priority** feature of the application, as it forms the foundation for managing shared costs and calculating individual contributions. The system must accurately divide expenses based on user preferences, ensuring fair and transparent tracking of who owes what.

4.1.2 Stimulus/Response Sequences

- **Stimulus:** The user clicks "Add Expense" and inputs the total amount.
    - **Response:** The system prompts the user to add the amount, transaction number, and a nickname for the bill.
- **Stimulus:** The user selects a bill to split.
    - **Response:** The system prompts the user to add friends to the bill. The default splitting is done by equally dividing the bill among the added friends.
- **Stimulus:** The user selects "Split Unequally" and inputs custom amounts for each participant.
    - **Response:** The system checks the total amount to ensure it matches the sum of the custom amounts and, if valid, updates the balances accordingly.
- **Stimulus:** The user visits a friend's profile to view all splits shared with that friend.
    - **Response:** The system displays a list of all bills involving the selected friend, showing the amounts owed or to be received, along with a summary of the total credit or debt between the user and the friend.
- **Stimulus:** The user visits the "Pay" tab to view all pending payments from others.
    - **Response:** The system displays a summary of all outstanding payments, listing each person who owes the user, along with the respective amounts and due dates.
- **Stimulus:** The user selects a bill and opts to delete it after creation.
    - **Response:** The system confirms the deletion action with the user, and if confirmed, the bill is removed from the system. Any associated debts or credits related to the deleted bill are updated accordingly to reflect the removal.

4.1.3 Functional Requirements

- **REQ-1**: The system shall allow users to split an expense equally among all selected participants.
- **REQ-2**: The system shall allow users to split an expense unequally, where each participant can have a custom share.
- **REQ-3**: The system shall validate that the sum of custom amounts matches the total expense amount when using unequal splitting.
- **REQ-4**: The system shall automatically update each participant's balance based on their share of the expense.
- **REQ-5**: The system shall provide a detailed view of each expense, showing the amount each participant is responsible for and their current balance.
- **REQ-6**: The system shall handle input errors gracefully, ensuring that users cannot proceed if the total amount and the sum of individual shares do not match.

4.2 User Authentication and Profile Management

4.2.1 Description and Priority

User Authentication and Profile Management handle the creation and management of user accounts. This is a **high priority** feature necessary for securing access to the application and personalizing the user experience.

4.2.2 Stimulus/Response Sequences

- **Stimulus**: The user signs up or logs into the application.
  - **Response**: The system verifies the user's credentials and grants access to their account.
- **Stimulus**: The user updates their profile information.
  - **Response**: The system saves the changes and updates the profile.

4.2.3 Functional Requirements

- **REQ-1**: The system shall provide a secure sign-up process, allowing users to create accounts with unique email addresses and passwords..
- **REQ-2**: The system shall enforce strong password policies.
- **REQ-3**: The system shall ensure that user sessions are secure and that sensitive information is encrypted.

4.3 Transaction History

4.3.1 Description and Priority

Transaction History tracks all past expenses and settlements within the app. This is a **medium priority** feature, providing users with a log of all financial interactions for transparency and record-keeping.

4.3.2 Stimulus/Response Sequences

- **Stimulus**: The user navigates to the "Transaction History" section.
  - **Response**: The system displays a chronological list of all past transactions, including expenses added, settled debts, and edits.

4.3.3 Functional Requirements

- **REQ-1:** The system shall maintain a history of all transactions, including expenses, settlements, and edits.
- **REQ-2:** The system shall allow users to filter transactions by group, date, or participant.
- **REQ-3:** The system shall ensure that all transactions are time stamped and linked to the relevant group and participants.

4.4 Debt Settlement

4.4.1 Description and Priority

Debt Settlement allows users to settle their debts either by marking them as paid or by adjusting them against future expenses. This is a **medium priority** feature, crucial for ensuring that users can clear balances effectively.

4.4.2 Stimulus/Response Sequences

- **Stimulus**: The user selects an unsettled debt and clicks "Settle."
    - **Response**: The system provides options to mark it as paid or adjust it against another expense.

4.4.3 Functional Requirements

- **REQ-1**: The system shall allow users to mark debts as settled manually.
- **REQ-2**: The system shall allow users to add a reference to their debt settlement.
- **REQ-3**: The system shall update all related balances upon debt settlement.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- **Response Time**: The application shall respond to user actions (e.g., adding an expense, splitting an expense, viewing reports) within 2 seconds under normal conditions.
- **Scalability**: The application must support up to 1,000 concurrent users without performance degradation.
- **Data Processing**: The system shall process and update all user balances and group data within 1 second of an expense being added, edited, or deleted.
- **Load Time**: The initial load time for the application shall not exceed 3 seconds on standard broadband connections.
- **Transaction Throughput**: The system shall handle up to 100 transactions per second during peak usage times.

5.2 Safety Requirements

- **Data Integrity**: The application shall prevent loss of financial data in case of unexpected shutdowns or errors. All transactions must be atomic to ensure that incomplete operations do not corrupt the database.
- **Backup**: The system shall perform automated daily backups of all user data to prevent data loss in case of system failures.
- **User Safety**: The system shall not allow actions that could result in financial harm to users, such as the deletion of a friends/bill without settling all associated debts first.

5.3 Security Requirements

- **Authentication**: The application shall require all users to authenticate using a secure login process (JWT Tokens), including the use of strong passwords.
- **Data Encryption**: All sensitive user data, including financial details and personal information, must be encrypted in transit and at rest.
- **Access Control**: The system shall enforce role-based access control to ensure that users can only access data and perform actions that are authorized for their role.
- **Privacy**: The system shall comply with data privacy regulations, such as GDPR, ensuring

that user data is only collected, stored, and used with explicit con

5.4 Software Quality Attributes

- **Usability**: The application shall be user-friendly, with an intuitive interface that requires minimal learning. Users should be able to perform core functions (e.g., splitting expenses, creating groups) within 3 minutes of first use.
- **Reliability**: The system shall be available 99.9% of the time, with minimal downtime for maintenance. The system should gracefully handle unexpected errors and ensure data consistency.
- **Maintainability**: The codebase shall be modular and well-documented to facilitate future maintenance and updates. The application should support continuous integration and deployment (CI/CD) practices.
- **Flexibility**: The system architecture shall allow for easy addition of new features without requiring major changes to the existing codebase.

5.5 Business Rules

- **Data Retention**: User data will be retained as per GDDR regulation after account deletion.
- **Expense Approval**: For shared expenses, all involved participants must approve the expense before it is finalized (optional feature).
- **Subscription Model**: The transaction ID given by the user when adding bills will be encrypted for privacy
- **Data Visibility**: All data uploaded can be recorded and decoded if necessary by the Organisation as Financial Products are subject to Transparency

Appendix A: Glossary

Glossary of Terms

Acronyms and Abbreviations

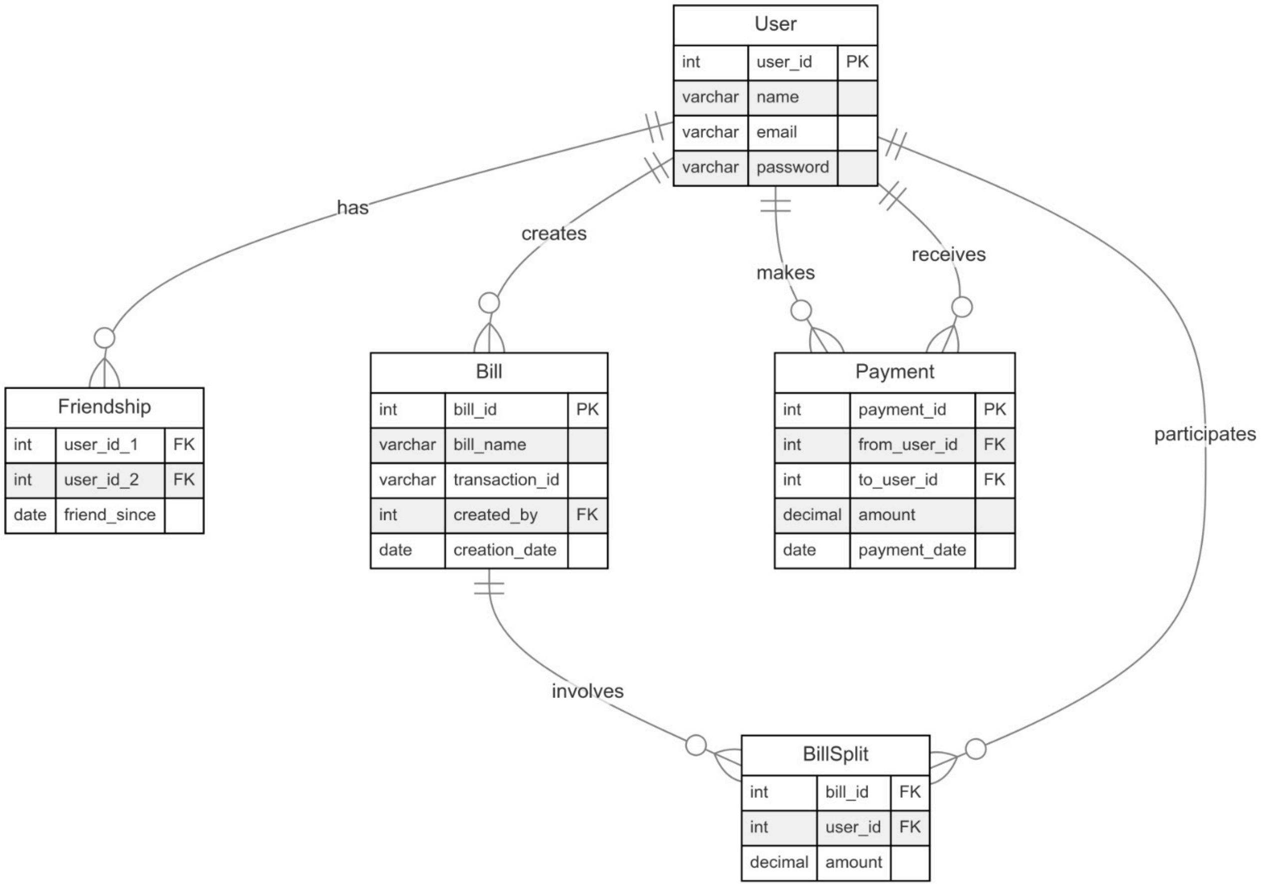
- REQ: Requirement
- UI: User Interface
- UX: User Experience
- API: Application Programming Interface

Terms

- **Authentication:** The process of verifying the identity of a user.
- **Data Synchronization:** The accuracy and consistency of data.
- **Encryption:** The process of converting data into a form that cannot be easily understood by unauthorized people.
- **Functionality:** The ability of a system to perform its intended functions.
- **Performance:** The speed and efficiency with which a system operates.
- **Scalability:** The ability of a system to handle an increasing amount of work.
- **Security:** The protection of data and resources from unauthorized access.

Appendix B: Analysis Models

Entity Relation Diagram for the Project



Appendix C: To Be Determined List

- REQ-4: The system shall generate personal Financial Reports to reflect a person’s financial spending [To be Determined] : Mentioned in 4.3.3 point 4
- REQ-4: The system shall generate personal Financial Reports to reflect a person’s financial spending [To be Determined] : Mentioned in 4.4.3 point 5

Appendix D: Synopsis

Overview:

Bill Splitting Application is a revolutionary web-based tool designed to simplify and streamline the process of dividing expenses fairly among a group. It addresses the common challenges of managing shared costs, ensuring equitable distribution, and maintaining transparent tracking of financial transactions.

Objective:

The primary objective of the Bill Splitting Application is to simplify the process of bill splitting, payment management, and financial tracking within a group. It aims to reduce manual effort in expense calculation and settlement, providing a user-friendly and intuitive interface for both bill payers and sharers.

Key Features:

- **Expense Splitting:**
  - Allows equal or unequal expense division, accurately calculating each participant's share based on their contribution or usage.
- **User Authentication and Profile Management:**
  - Secure sign-up and login process with multi-factor authentication, including third-party authentication options for added convenience.
- **Transaction History:**
  - Tracks all financial transactions, providing a comprehensive record of past expenses, payments, and settlements.
- **Debt Settlement:**
  - Enables users to settle debts conveniently by marking them as paid or adjusting them against future expenses.

Target Audience:

The Bill Splitting Application caters to a diverse range of users:

- **Primary Users:**
  - Individuals who regularly share expenses with friends, family members, roommates, or group travelers.
- **Secondary Users:**
  - Accountants, financial advisors, event planners, and business owners who need to manage group expenses efficiently and transparently.

Technical Specifications:

- **Frontend:**
  - Built using cutting-edge React framework, presenting a modern and responsive user interface.
- **Backend:**
  - Powered by Node.js and integrated with a robust PostgreSQL database, ensuring efficient data storage and retrieval.
- **Hosting:**

    - Hosted on Vercel for reliable performance, fast data access, and scalability.

Non-Functional Requirements:

- **Performance:**
    - Optimized to handle up to 1,000 concurrent users, enabling quick and seamless processing of transactions.
- **Security:**
    - Utilizes encryption protocols to protect sensitive data and employs JSON Web Tokens (JWT) for secure authentication.
- **Reliability:**
    - Guarantees 99.9% uptime through robust infrastructure and continuous monitoring, ensuring minimal disruption to users.

Conclusion:

The Bill Splitting Application is a comprehensive solution for managing shared expenses, catering to a wide range of users. Its focus on simplicity, security, and reliability makes it an invaluable tool for individuals and groups looking to streamline their financial management and promote financial transparency.