



CS 307

Design Document

Team 7:

Haris Sohail, Mayur Patil, Jack Crawford, Jacob Riggs



Table of Contents

Table of Contents	2
Purpose	3
Functional Requirements:	3
Non Functional Requirements:	6
Design Outline	8
High-level System Overview	8
System Component Interaction	8
Structure Diagram	9
Design Issues	10
Functional Issues	10
Non-Functional Issues:	12
Design Details	14
Class Descriptions and Interactions	14
Class Diagram	16
Sequence Diagram	17
UI Mockups	21



Purpose

Day-to-day expense tracking can be difficult, but learning to budget is a valuable skill. One of the easiest ways for one to adjust their spending habits is by gathering information on what products and services they spend the most money on and determining how to reduce the amount spent among different areas of expense. Our product will provide a way for users to do just that. With it, they may digitally track their expenses and decide which areas of expense they are most willing to reduce cost in. Our product will then provide suggested levels of spending based on user preferences and available spending money.

Functional Requirements:

1. User Account
 - a. As a user,
 - i. I would like to be able to register an account
 - ii. I would like to be able to login to my account with a selected username and password
 - iii. I would like to be able to reset my password
 - iv. I would like to be able to change my email
 - v. I would like to be able to change the theme of the application to my liking
 - vi. I would like to be able to change my currency of choice
 - vii. I would like to be able to change the color of different spending categories to my choosing



2. User Event Posting

a. As a user,

- i. I would like to be able to manually enter purchases
- ii. I would like to be able to create new spending categories
- iii. I would like to be able to categorize my purchases
- iv. I would like to be able to enter recurring spending amounts
- v. I would like to be able to track what day of the week I buy certain items
- vi. I would like to be able to create a budget for next month
- vii. I would like to be able to plan savings for retirement
- viii. I would like to be able to compare spending on different types of a certain item (i.e. different brands of coffee, separate items of clothing)
- ix. I would like to be able to set reviews for bought items for future reference
- x. I would like to be able to plan budgets for months in the future
- xi. I would like to be able to make a budget for a trip/vacation
- xii. I would like to be able to make a budget for Christmas/Birthdays

b. As a parent,

- i. I would like to be able to monitor my kids' spending
- ii. I would like to be able to have an interface for the budget of my entire family
- iii. I would like to be able to create a budget for my kids and share it with them



- iv. I would like to be able to modify my kids' budget to reward/punish them
- v. I would like to be able to send messages through the application to my kids (if time allows)

3. User Event Viewing

- a. As a user,
 - i. I would like to be able to view graphs that show my daily, biweekly, and monthly spending.
 - ii. I would like to be able to view graphs that show my spending for specific categories.
 - iii. I would like to be able to receive spending suggestions

4. User Notifications

- a. As a user,
 - i. I would like to enable notifications for my daily, biweekly, and monthly spending trends
 - ii. I would like to enable notifications for spending on specific items
 - iii. I would like to enable notifications for spending in specific categories
 - iv. I would like to be able to enable notifications on Google Chrome
 - v. I would like to be able to set reminders for buying certain items at specific times
 - vi. I would like to be able to receive notifications on what categories to cut back on



5. User Sharing

- a. As a user,
 - i. I would like to be able to share my monthly savings on Facebook
 - ii. I would like to be able to share my monthly savings on Twitter
 - iii. I would like to be able to share my budget with other users

Non Functional Requirements:

1. Architecture

- a. As a developer,
 - i. I would like to have a functional back-end built on Node.js and MySQL.
 - ii. I would like a functional front-end written in Angular.
 - iii. I would like requests between the front-end and back-end to be efficient and easily manageable.

2. Security

- a. As a developer,
 - i. I would like user data to be anonymous and login information to be kept secure.
 - ii. I would like prevent our database from becoming vulnerable or corrupted.



3. Usability

- a. As a developer,
 - i. I would like the interface to be intuitive and the functionality to be easily grasped.
 - ii. I would like the application to be able to run on most standard web browsers.

4. Performance

- a. As a developer,
 - i. I would like the database to be able to sustain 10,000 users worth of data.
 - ii. I would like query responses to take no more than 400 ms.

5. Appearance

- a. As a developer,
 - i. I would like the interface to be clean and free of clutter.



Design Outline

High-level System Overview

This application will be a web-based budgeting tool designed to allow users to create, manage, and share their budgets. Feedback will be given to users based on their expense history, spending preferences, and allotted funds. Budget sharing will be designed especially with families in mind, allowing parents to manage budgets for their children and teach fiscal responsibility.

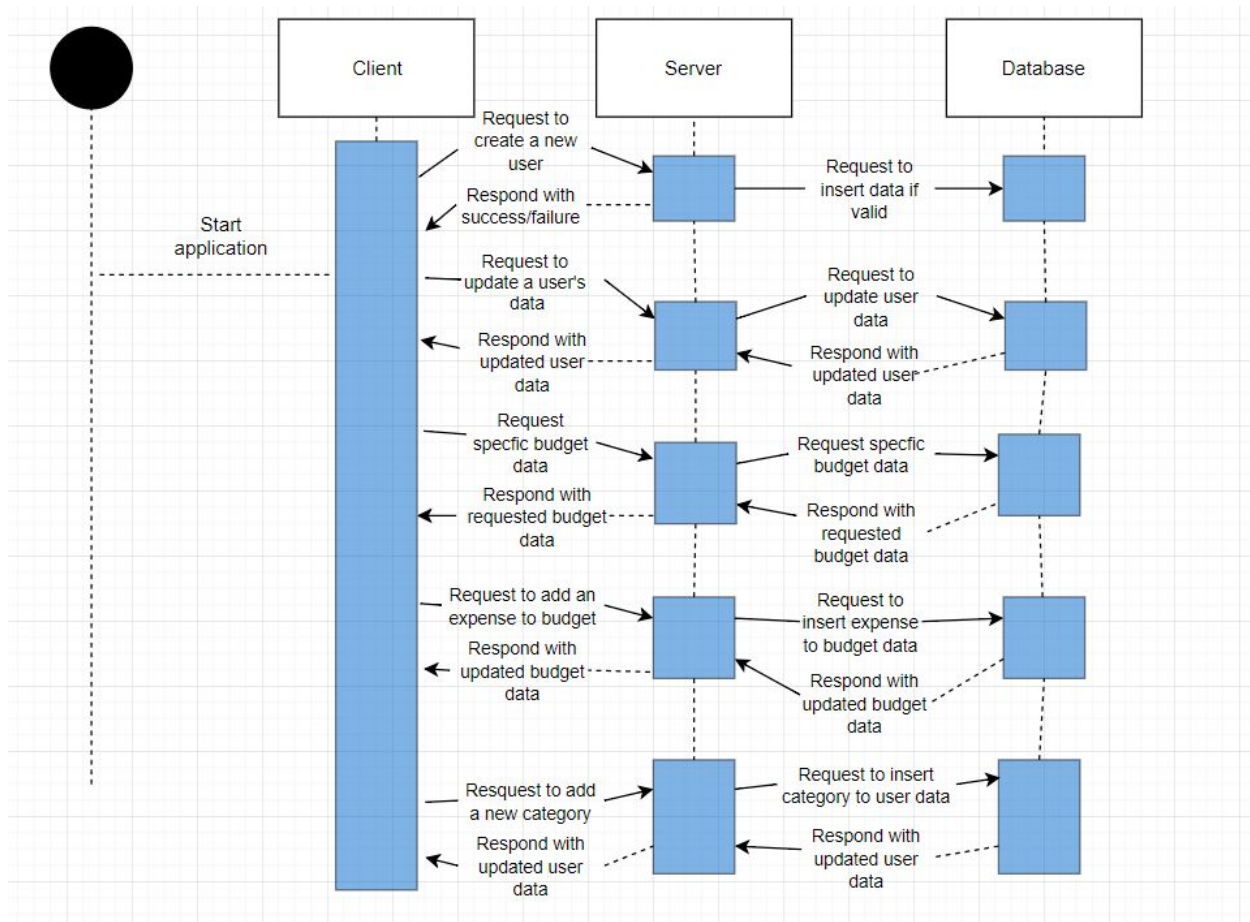
Each user will have a client interface to interact with which will communicate with a server to send and retrieve information. This server will then send queries to a database to collect and provide data for clients on user's expenditures, budgets, and other information pertaining to recommendations for reducing spending.

System Component Interaction

1. Client
 - a. Client will provide the interface for our application.
 - b. Client will send requests to the server.
2. Server
 - a. Server will handle the requests from the clients.
 - b. Server will query the mySQL database to add or retrieve data when requested.
 - c. Server will respond to the clients with appropriate responses, depending on feedback from the database.
3. Database
 - a. The database will store the user information, along with their budgets and expenses.
 - b. The database will respond to server requests and send back the pertinent information.



Structure Diagram





Design Issues

Functional Issues

1. What information is needed for creating an account?

- **Option 1: Username, email, and password**
- **Option 2: Username, password, email, phone number**
- **Option 3: Email and password only**

Choice: Option 3

Option 1 allows for simplicity. The username will not be very difficult to implement, and will allow for a uniqueness factor per user. Notifications for meeting the budget goals will be sent through email, so phone number is not needed.

2. How should we send notifications to the user?

- **Option 1: Through email**
- **Option 2: Through SMS message**
- **Option 3: Through app notification**

Choice: Option 1

SMS is more expensive as it requires an API and app notifications are not as direct to our users. Email is perfect because it is direct to their inbox and can contain more details about the users' progress to meeting their budgets.



3. Should the user be able to create their own categories?

- **Option 1: Predefined categories**
- **Option 2: Customizable categories**
- **Option 3: Predefined categories with limited customization**

Choice: Option 3

Option 3 allows us to develop generic categories that will cater to most of our user base and it also allows our application to be used for other types of budget tracking which the user can add their own categories. This makes the application more flexible and useful.

4. How should statistics be displayed on the webpage?

- **Option 1: Through bar graphs**
- **Option 2: Through numbers**
- **Option 3: Through bar graphs, pi graphs, and line graphs**

Choice: Option 3

Option 3 is best because we can display useful information through bar graphs, pi graphs, and line graphs as far as displaying how the user's budget goals are going. More graphs provide the user different types of approaches to analyzing where their spending can be improved to reach the budget target faster.



Non-Functional Issues:

1. Where will we host this application?

- **Option 1: Heroku**
- **Option 2: Azure**
- **Option 3: Digital Ocean**

Choice: Digital Ocean - we already have a VPS here, so we might as well use that. It's online all the time and allows us to have the convenience of having our product online 24x7 during development.

2. What language/framework will be used for the backend?

- **Option 1: PHP**
- **Option 2: Node.JS**
- **Option 3: Django**

Choice: Node.JS

Node.JS has many libraries and is easy to pickup as it's essentially javascript. Multiple members on our team also know Node.JS so that's helpful. Node.JS is also really easy to work with for web apps.

3. What language/framework will be used for the frontend?

- **Option 1: Angular**
- **Option 2: Raw HTML + JS**
- **Option 3: React**

Choice: Angular

We chose angular because we have the most experience with it and it's really convenient for building web apps.



4. Which database will be used?

- **Option 1: mySQL**
- **Option 2: MongoDB**
- **Option 3: Firebase**

Choice: mySQL

Node.JS has helpful NPM libraries that make mySQL very easy to use. Our team has members that are experienced with using it too.

5. How will notifications be sent out and through what API?

- **Option 1: SMTP**
- **Option 2: Mobile SMS API**
- **Option 3: Leave notifications only on the actual platform**

Choice: SMTP

SMTP notifications are really easy to implement and they're more likely to grab the attention of the user. They're also way cheaper than having to use a mobile SMS API.



Design Details

Class Descriptions and Interactions

- **User**

- User object is created when a user registers onto BudgetWise
- User will have its own universally unique identifier (UUID)
- User will have a username, email address and, password
- User will begin with monthly budget and be able to add additional budgets
- User will have their own budget categories which will include predefined categories and categories that the user adds
- User will store expenses
- User can only edit its own categories
- User will have a settings class

- **Budget**

- Linked to the user object
- Each budget will be assigned a unique budget ID
- Each budget will have a name, cash allowance, and time interval at creation
- Each budget will store a notification object defining notification preferences

- **Expense**

- Each expense has a name and a price
- Expense belongs to a category
- Each item will be assigned a unique expense ID



- **Settings**

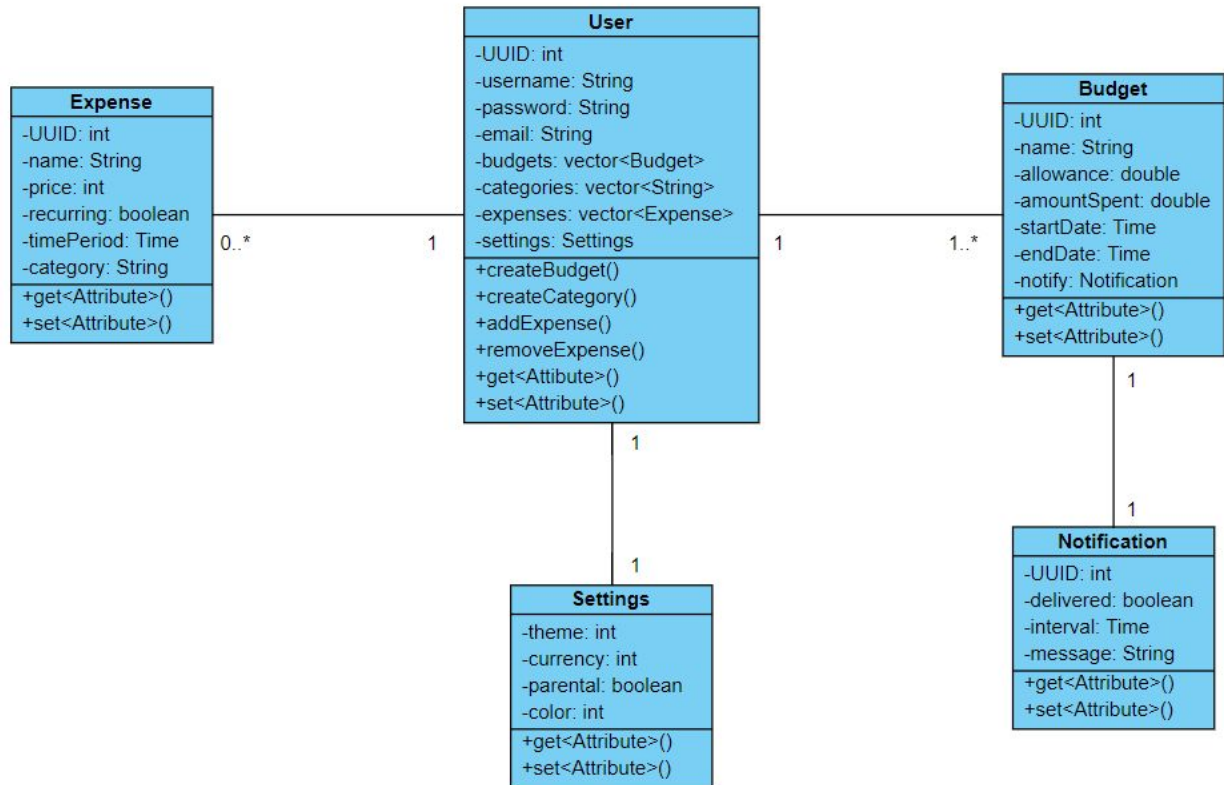
- Settings will be stored as a field underneath the user object
- Setting option for dark/light mode
- Setting option for color scheme
- Setting option for currency
- Setting option for parental account

- **Notification**

- Notification object will be sent based on time frame and how far the user object is from reaching the set budget, which will be reflected in the message string
- Notification object will have a universally unique identifier (UUID)
- Notification object will have a delivered tag (boolean based on if emailed)
- Notification will have an interval for how often reports are sent to user



Class Diagram

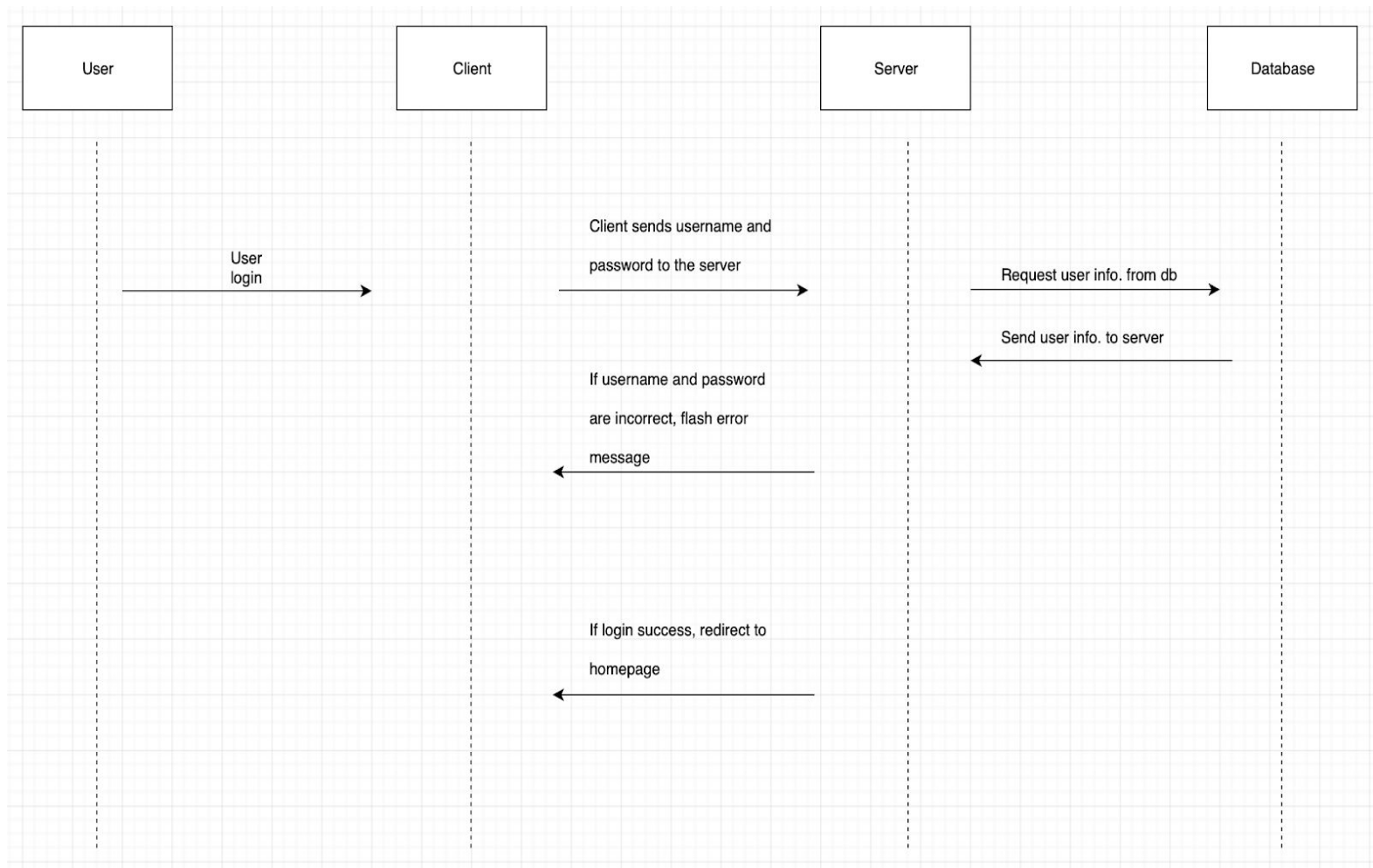




Sequence Diagram

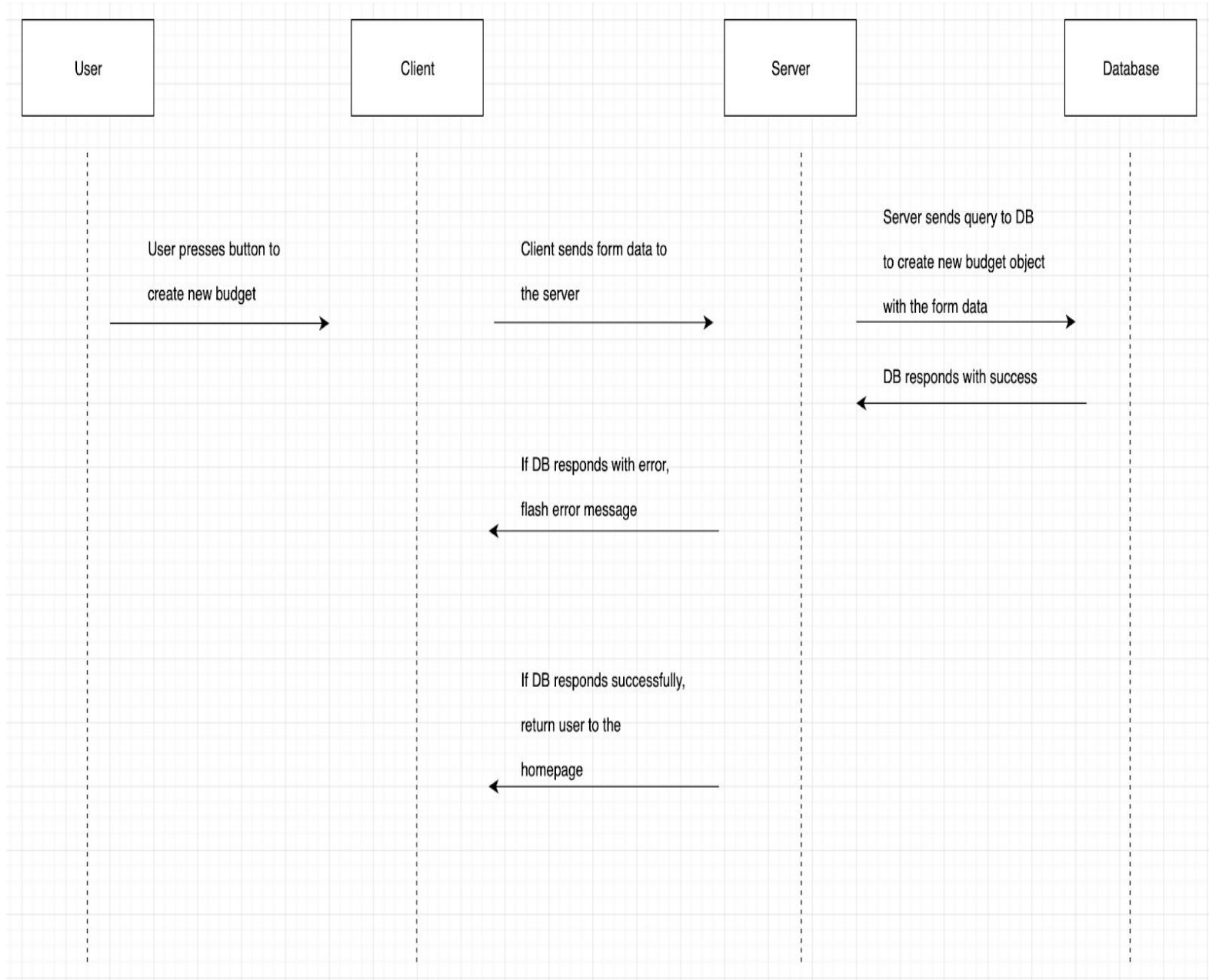
The following diagrams represent the connection between the user, client, server, and database for common interactions such as login, budget creation, and expense creation.

1. Sequence of events when user logs in



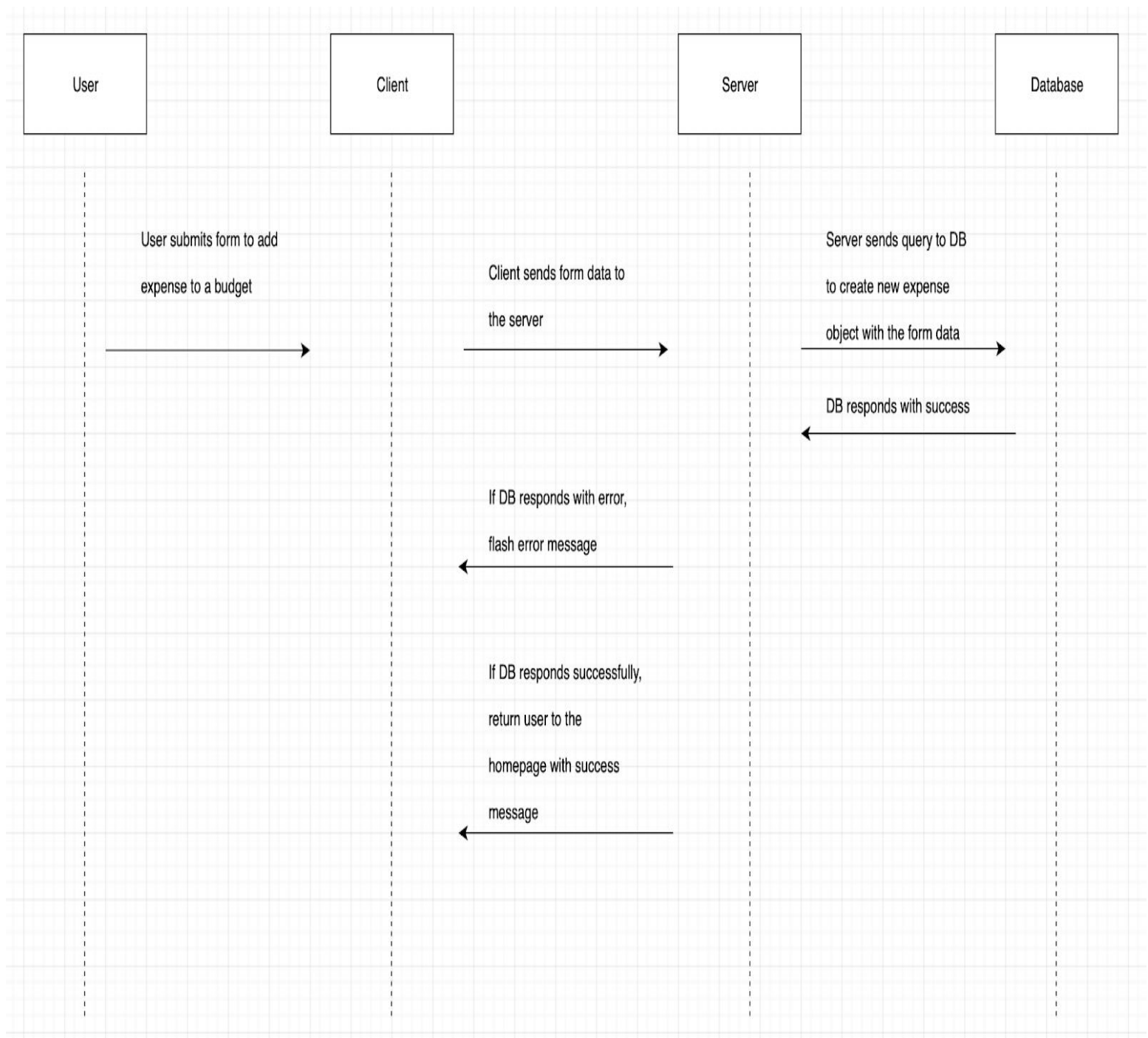


2. Sequence of events when a budget is created



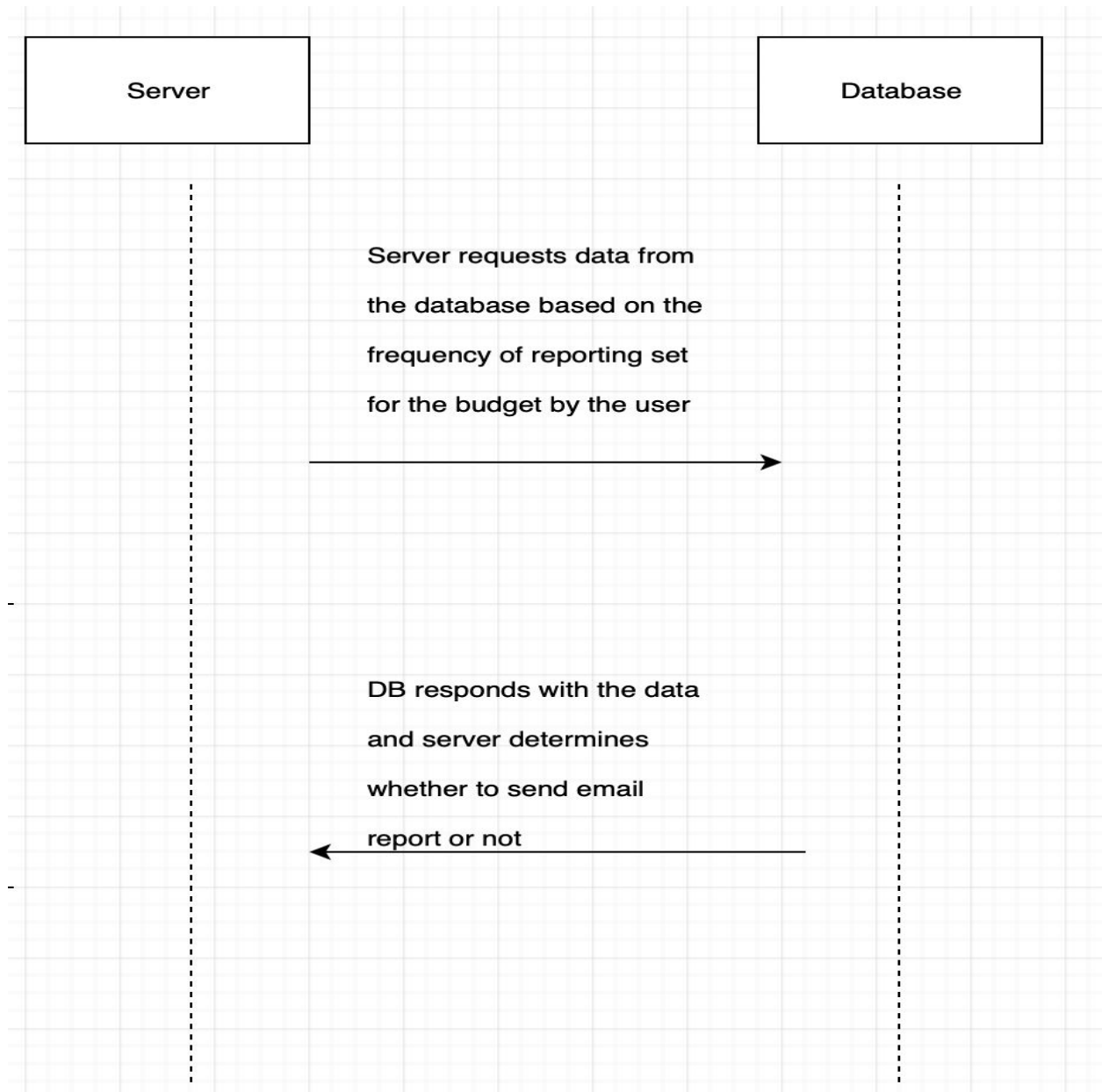


3. Sequence of events when an expense is added to a budget





4. Sequence of events when report is sent to user





UI Mockups

- Login page

A mockup of a web browser window titled "BudgetWise". The browser's address bar is empty. The main content area features the BudgetWise logo, which consists of three hexagons (orange, blue, and green) arranged in a triangle, with the text "BudgetWise" below them. Below the logo is a white rectangular box with a thin gray border. Inside this box, the word "LOGIN" is centered at the top. Below "LOGIN" are two input fields: "Username" and "Password". Below the "Password" field is a checkbox labeled "Remember Me". To the right of the "Remember Me" checkbox is a dark gray button labeled "Sign In". Below the "Sign In" button is a green button labeled "Forgot Password". Below the "Forgot Password" button is a blue button labeled "Create Account".

This is the login page. The user will be able to sign in with either their username or email. This user can also hit forgot password to reset their password, or click create account if they are a new user.



- **Create an Account Page**

A screenshot of a web browser window titled "BudgetWise". The page features the BudgetWise logo, which consists of three hexagons (orange, blue, and green) arranged in a triangle above the text "BudgetWise". Below the logo is a "Create An Account" form. The form contains six input fields: "First Name", "Last Name", "Email", "Username", "Password", and "Re-Enter Password". At the bottom of the form is a blue button labeled "Create Account".

BudgetWise

Create An Account

First Name

Last Name

Email

Username

Password

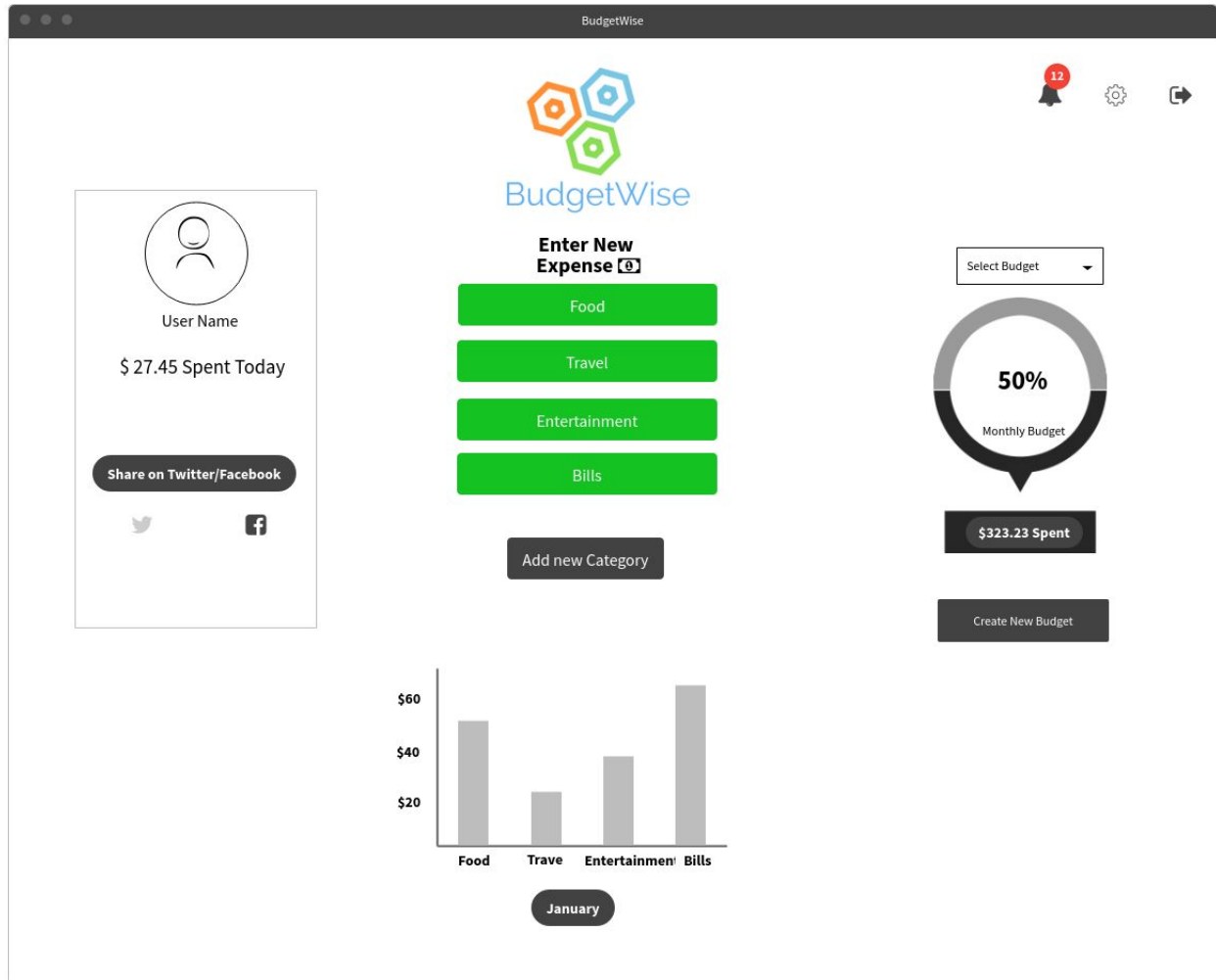
Re-Enter Password

Create Account

This is the page for creating a new account. This page will check if the username or email are already associated with an account, and will also have minimum password requirements.



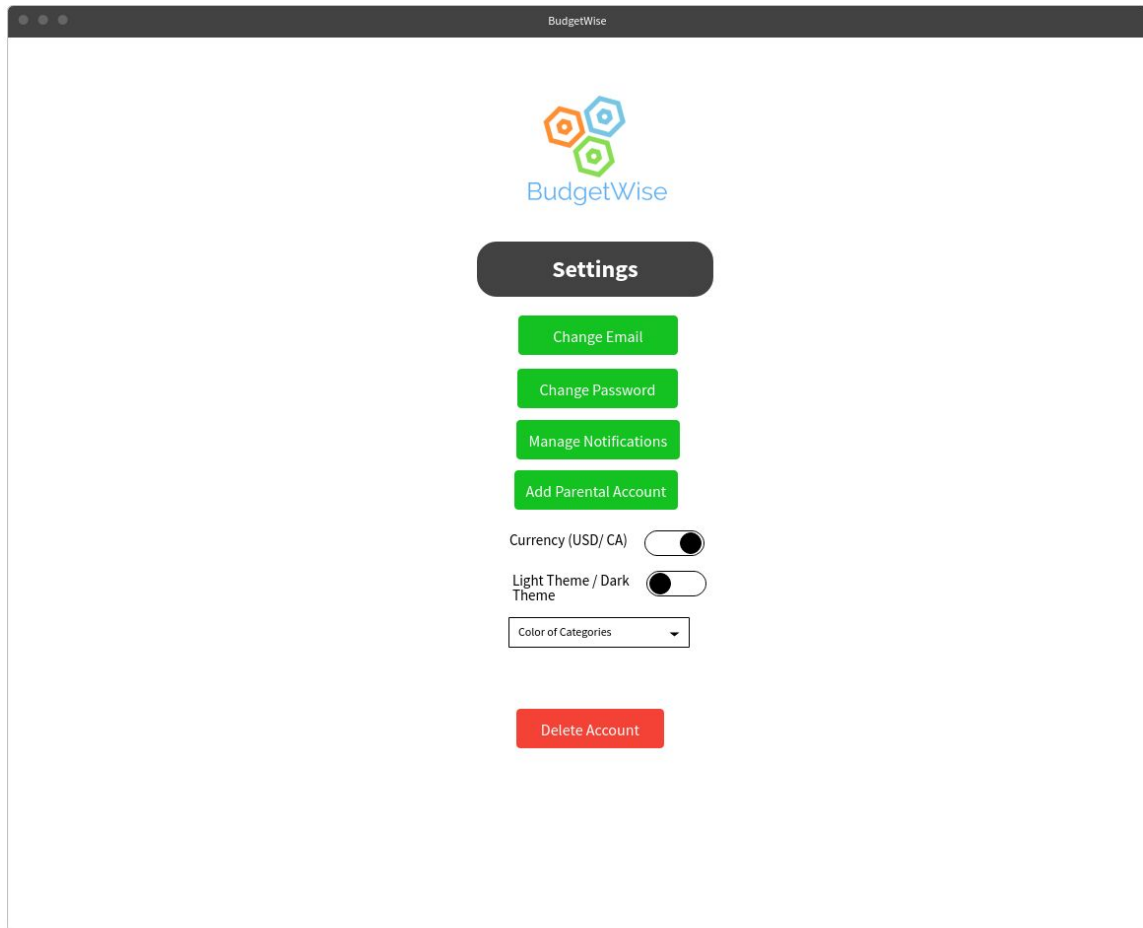
- Example Main Page



This is the main page for the application. When a user has an expense, they will click a category to enter it into. They will be able to enter the specific expense as well as the price of the expense. They will be presented with the option to set the expense as recurring. This page also allows the user to change which budget graph is showed to them.



- **Example Settings Page**




The settings page will have options to change email, change password, change currency, change theme, add parental controls, and change color of category buttons on main page. They can also click the button for setting notifications which will navigate them to the notifications page.



- **Create a Budget Page**

BudgetWise


BudgetWise

New Budget

Set Notifications


Create Budget

This is the new budget page accessed through the main page. The set notifications button will navigate the user to the notifications page with the new budget selected. The user can click Create Budget and the application will navigate the user back to the main page.



- **Notifications Page**

BudgetWise


BudgetWise

Notifications

Select Budget ▼

☒

 All Notifications

☒

 General Notification Interval

Daily ▼

☒

 Warning Notification

\$300

Save

The notifications page will allow the user to opt out of all notifications at once, as well as set the notifications for each budget. The notifications will be presented by email.