Eric Chaing (echai009@ucr.edu)
Ashley McDaniel (amcda005@ucr.edu)
Jacques Fracchia (jfrac001@ucr.edu)
Allison Nguyen (anguy129@ucr.edu)

R'Budget

CS180 : Intro to Software Engineering

Team: Money Management

rbudget.xyz

r'Budget

Description:

This software project is a web-based implementation of a money budget tracker. The user should be able to keep track of all expenses such as utilities, rent, groceries and any other daily expense. Their budget should be visualized in an easy to understand fashioned manner. This web-app will be a necessity for users of all ages and backgrounds who are interested in improving their financial well being.

Manage Budget is a tool for expense control and manage home budget. Thanks to carefully selected functions not only will you be able to save more money, but also analyze your expenses more precisely

User Stories:

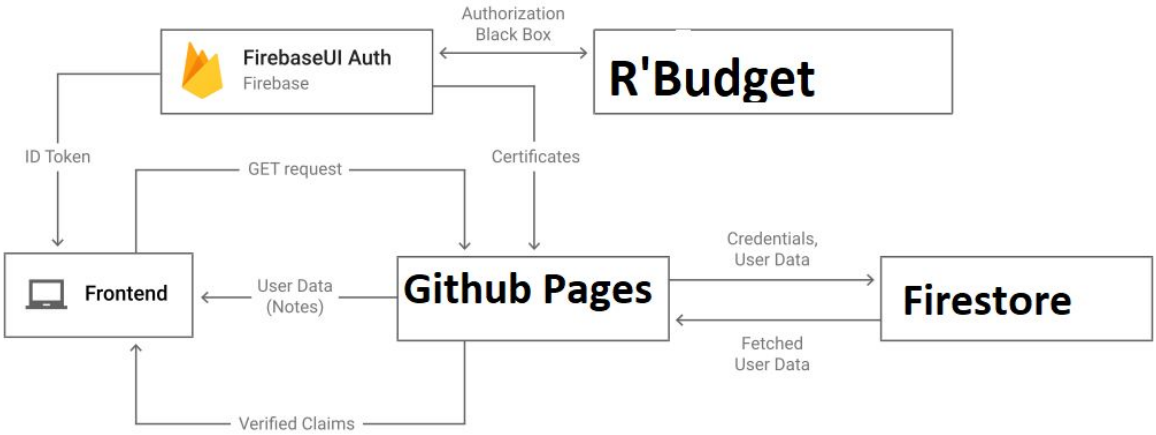
1. Build an interface to keep track of the budget for different types of expenses. Save the information in a structured database collection. [8]
2. Have recommendations based off of total budget for expense categories. Categories: Utilities, Rent, Entertainment, Food, Transportation. Such as 10% for transportation. Pop up alert when you passed it. Handled in back end, they will just see the categories in the visual display and log [8]
3. Visualize categorical spending on a selection of charts. Have each expense category shown as a percentage of the chart. Each slice will be labeled with its category name. [7]
4. Create a log of statements to display from the database date, budget, expense category, and an optional description. [7]
5. As a user I would like to store my Name, Email, Housing Address and phone number on my account, view it in profile and the homepage and have it stored to the database. I would like to see an intro, about, and a 'About Us' tab on the homepage that will show a picture of the four developers and a small statement about each of the four of us. [4]
6. As a user, I would like to determine my recommended spending amount for each of the nine categories. If I do not specify any amount I would like to use the default percentage break down that r'Budget has recommended. These inputs for the categories should be shown in the account page. After clicking a button? [7]
7. As a user I would like to plug in items I am saving up for, such as a car or laptop. I would plug in a value and a description for what it is. This will fall under the savings category and once the user has saved up enough for that one particular item It would create an HTML element telling you you have saved up enough money for that item. This would be in a goals tab on the homepage.
8. As a data curious user, I would like to see the comparison of my spending against the recommended amounts as a bar graph on the Analytics tab. This comparison should also work if I decided to put in my own recommended spending amounts. [5]
9. As a user I would like to have my own login and password to view my finances. [7]
10. As a registered user, I would like to view my general information on homepage. [4]
11. a)As an impulsive consumer, I want to be able to set a budget for myself so I can see when a purchase will put me over my budget.
    b)As a person who does not work with money and finances in-depth, I want to be able to see my budget and information about my money in a simplified and visual way.
    c) set different kinds of budget plans that is unique for the user

Meeting Schedule:
Weekly
Tuesday & Thursday
2:00 PM - 5:00 PM
Orbach Library

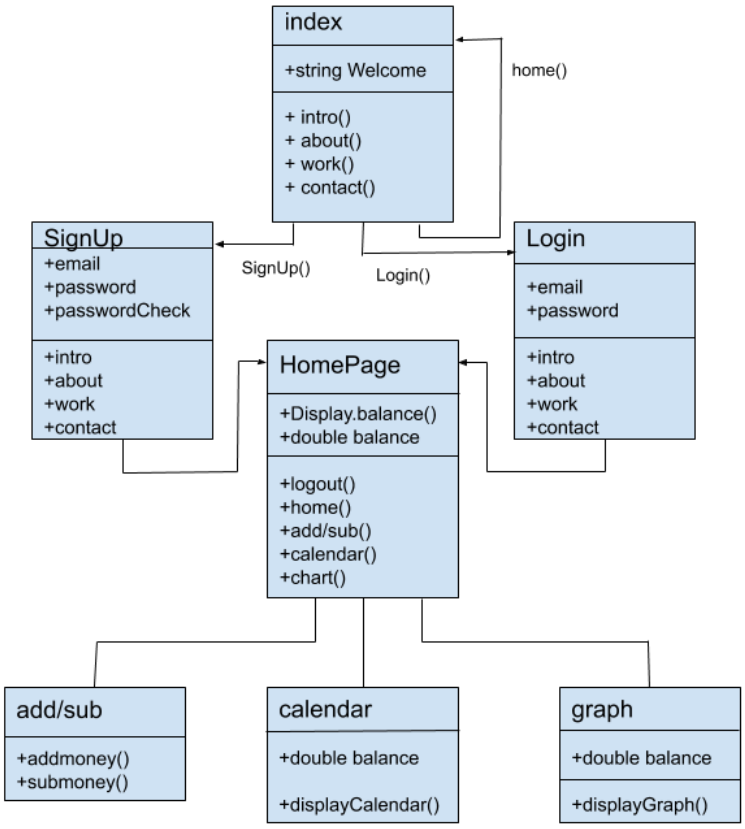
Communication

- Text and Facebook Messenger

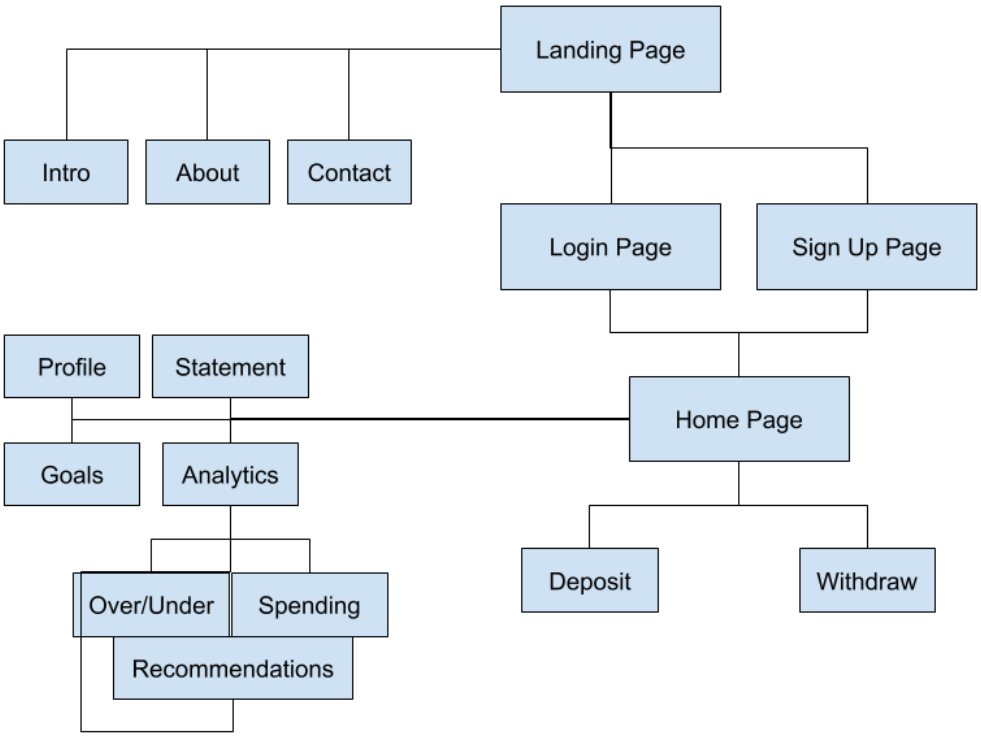
Dependencies UML:



Code UML:



Interaction UML:



Front End:

Index.html:

- Presents to the user what the web app is about. Displays rbudget and created by money management.
- Has buttons “Intro” “about” “contact”
- Each button are clickable and gives a photo and a brief paragraph for each.
- Taskbar: taskbar at the top has “Home” “login” “signup”

Signup.html:

- After clicking “signup” in the taskbar of the index.html you will be redirected to a sign in page
- Sign in asks a user to plug in a valid email, a password and confirm password.
- This information gets processed in the signup.js file where it gets sent to the backend.

Signin.html:

- After clicking “signin” in the taskbar of the index.html you will be redirected to a loginpage
- Sign in asks a user to plug in a registered email and password
- This information gets processed in the signin.js file where it gets sent to the backend.

Homepage.html:

- After the user clicks submit in either the signup page or login page, and the email and password is a valid user they get sent to the homepage.
- The home page presents the users budget under “My Budget: “
- The user then can type in their budget in the field to add or subtract values.
- Data gets sent to the database to store the value.
- Keeps a log of each entry to graph out their finances in database. This is shown as statements page similar to bank statement. It shows each date, along with balance amount category type and description.
- Has clickable buttons below that will display “Statement” “Analytics” “Goals” “Profile”.
- When the user refreshes page it pulls budget data from database and puts it in the budget field.
- Taskbar is now displaying home and logout. Home takes you back to homepage.html
- Button for deposit and withdraw. User can enter fields for amount, category and description. This data then gets added as a log in each individual users database.
- Profile: the user can save their email, name, house address, phone number, and can plug in a photo of themselves.
- Statement: Shows the log of the users spending, similar to a bank statement.

- Analytics: Shows two pie charts and one line graph. The pie charts shows the users recommended spending, their actual spending, and the line graph shows them how close over and under their graph is from recommended amounts.

Backend

Signup.js:

- Uses firebase UI authentication to create a new user. Double checks that all fields are filled and uses a valid email address. If it is sends user to the homepage.

Signin.js:

- Uses firebase UI authentication double check that the email and password provided is a valid email. If it is send user to the homepage.

Budget.js:

- GetBudget() retrieves the latest variable stored in the database under budget. Then displays the latest budget in the user's field when the first log into the homepage.
- SetBudget() checks if the field has been updated with a new budget. If a new value was entered, this function then sends the data from the field into a new log entry for budget inside of the database.
- Deposit()
    - Grabbes values from the deposit fields: "Amount" "Total Budget" "Description" and "category". Plugs into the firestore database in the following hierarchy: email->Budget->Month->day HH::MM->log.
- Withdraw()
    - Grabbes values from the withdraw fields: "Amount" "Total Budget" "Description" and "category". Plugs into the firestore database in the following hierarchy: email->Budget->Month->day HH::MM->log.

Visual.js (Analytics):

- Analytics displays three graphs:
    - One over/under display that shows each spending category and whether you went over the recommended amount or not
    - Display Categories: displays the amount of money you are spending for each category in a pie chart. If you hover over it, it shows the amount for each category.
    - Recommendations: Displaying the percentage in a pie chart of the recommended spending for each category. You can compare it with the display categories pie chart to see if it looks similar or not as well.
- We displayed the charts using charts library. Where we can plug in n number of variables to display and can plug in our values.

Statements

- Helps the user to see their deposits and withdraws
- STATEMENT:
    - Date: Sun Dec 01 2019 17:09:39 GMT-0800(PST)
    - Balance: 1600
    - Amount:12
    - Category:food
    - Description
- Works by retrieving the timestamp variable that is stored in the database for each transaction
- When there is a transaction there is a value saved called Timestamp from firebases server timestamp this was necessary to implement to order the data by time accurately
- Then each statement is sorted by Timestamp from the newest date to the oldest date

<u>Goals</u>

- The user can use the Goals page to organize and keep track of their goals
- The user can create up to 3 goals: Goal #1, Goal #2, Goal #3
- For Example:
    - Goal #1
    - What are you saving for?
        - User Data Field
    - Set how much you want to save for your goal
        - User Data Field
    - Set what percentage of your savings you want to go towards your goal
        - User Data Field
    - SUBMIT BUTTON
- Clicking the submit button will save the users entries to the database and reload back to the homepage
- Submitting a new goal will overwrite the previous goal
- The page will save the users data field entries to a variable so that when they go back to this page their data is still there
- To meet their savings requirement we multiply the value that they have withdrawn to savings * Percentage of Savings
    - Compare this value to how much they want to save and see if the user meets their goal savings requirement
- When the user meets their savings requirements:
    - When the user meets their savings goal requirement the page will show a message for that particular goal saying: “*You have the savings required for goal #(1/2/3)!*”
    - Also when the user meets their savings a PURCHASED button will appear
        - Clicking the purchased button will alert the user to “Enjoy their Purchase!’ and withdraw the requirement amount from the users savings category and reload back to the homepage

- After clicking the purchased button the data field will be reset telling the user to enter a new goal, have 999999 in the amount, and 0 for the percentage for the user to enter a new goal.

Notes

Categories for spending.

STATEMENT: (break up by months)

Date: 10/29/2019 Category: Food, Spent: 20, Overall Balance: 70
Description:
Date: 10/28/2019, Category: Transportation, Spent: 10, Overall Balance: 90
Description:

CATEGORIES:

- Entertainment
    field: description
- Housing/Rent - Recurring payment checkbox
    field: description
- Utilities - Recurring payment checkbox
    field: description
- Food
    field: description
- Transportation
    field: description
- Education
    field: description
- Loan Repayment
    field: description
- Child Care
    field: description
- Savings
    field: description

+ Income

field: description
+ Financial Aid & Grants
field: description