Team SAW: Alvin Wu, William Yin, Jonathan Lee, Saqif Abedin
SoftDev
P5: This is the End
2021-05-24

Personal Expense Tracker:
Our website allows users to track their spending and savings and display it in helpful and useful ways. It can compare how much money you spend on different categories for each week and give you a summary on overall spending and savings. It can also make suggestions to spend less on certain areas and more in others.

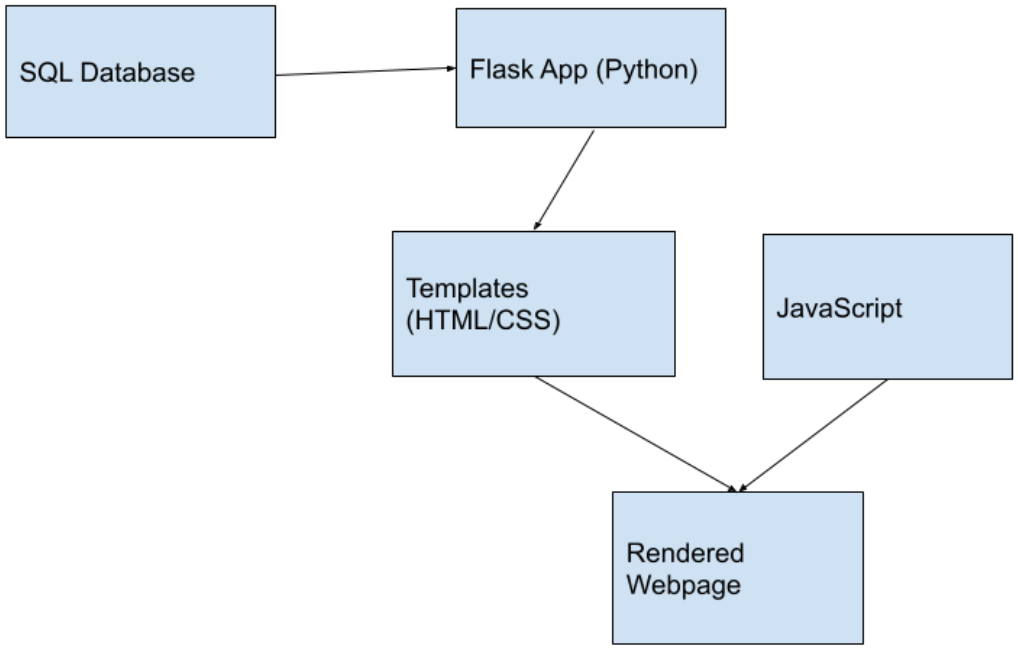
Components

- README.md:
  - Clearly visible at top: **<Project Name> by <Team Name>**
  - Roster with Roles
  - Description
  - Launch Codes:
    - How to clone/install
    - How to run
- SQLite Database:
  - Table 1: Users
    - ID (non-negative auto-incrementing integer)
    - Username (unique field)
    - Password
    - Weekly Budget (positive integer, defaults to 0)
  - Table 2: Expenses
    - ID (non-negative auto-incrementing integer)
    - User ID
    - Expense Name
    - Expense Description
    - Amount
    - Timestamp (integer)
- Flask App:
  - connects backend files (i.e. database) to frontend files (i.e. html templates). This Python script will redirect users to another webpage based on the buttons they click and their input in HTML forms.
- Templates:
  - signup.html: contains an HTML form that allows users to create a username and password for their account. The backend will check if the usernames chosen by

new users already exist in the Login table and verify if they fulfill other requirements (i.e. password length). Users will be redirected to login.html or error.html depending on whether their account was created successfully.

- login.html: contains an HTML form that allows the user to enter username and password. When the user clicks the "Submit" button, app.py will check if the credentials match an entry of the Login table in the SQLite database. Users will be redirected to layout.html or error.html depending on whether they logged in successfully.
- error.html: If users fail to login or create an account successfully, they will be directed to this page and be told what went wrong. Below the error message, there is a "Sign Up" button that redirects the user to signup.html and a "Login" button that redirects the user to login.html
- main.html: When the user signs in, they are brought to the main site page. The page will have form fields for the user to enter specific information. For example: their budget, and their expenses. When the user fills in the expense fields, they can click an "add" button which will use JS to update the database, display the new entry, and also add another row to the table on the page. After doing so, we will use JS to calculate and display the remaining budget, updated accordingly. If the user exceeds the budget, they will be given a warning when they try to add the entry. We should also allow users to delete previous entries if they wish to do so.

Component Map



Database Organization

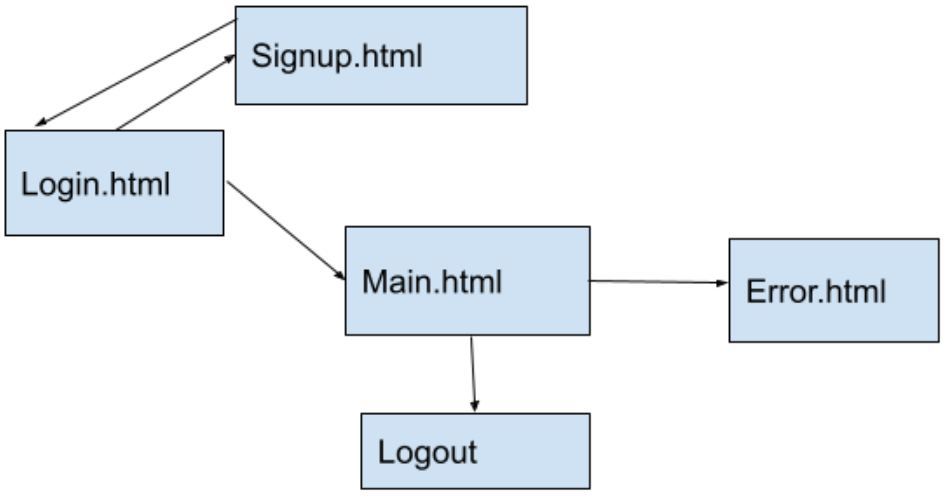
- Users table:
    - There will be a row for each user.
    - There will be 4 columns: ID, Username, Password, Weekly Budget

ID	Username	Password	Weekly Budget ($)
0	blogger1	t0g2ka	500.00
<non-negative integer>	<string>	<password>	<non-negative double>

- Expense table:
    - There will be a row for each expense.
    - There will be 5 columns: ID, User ID, Expense Name, Expense Description, Amount (in dollars.cents), timestamp (hour/min in military format)

ID	User ID	Expense Name	Expense Description	Amount	Timestamp
0	0	Food	Bought lunch the other day	25.00	2021-05-21-1500
<non-negative integer>	<non-negative integer>	<string>	<string>	<non-negative double>	<yyyy-mm-dd-tttt>

Site Map for Frontend



Tasks

- Create Flask app with different routes and functions
- Create HTML Templates
    - Home page
    - Login/Register
    - Error.html
    - Main.html with most of the User information
- Authentication with SQL tables / python
    - Store money data for each user and manage data
    - Budget/ Money Management functions

Timeline

- Will update once we know due date