J-HEX

Eric Li, Xing Tao Shi, Jacqueline Woo, Henry Zheng
SoftDev pd8
Project02 -- The Final Frontier
2018-06-12

Description

J-HEX presents Clover, a financial manager where users will be able to input data about their finances, savings, and expenditures. The user will be able to see a history of their past spendings and set budgets for their future expenditures. Users will also be able to clearly see breakdowns of their expenditures and projected savings via graphs and charts generated by d3. Another feature of Clover is that users will be able to track and manage their stocks. They will be able to easily see the profit they make from their stocks via d3 charts and search up new stocks to add to their portfolio.

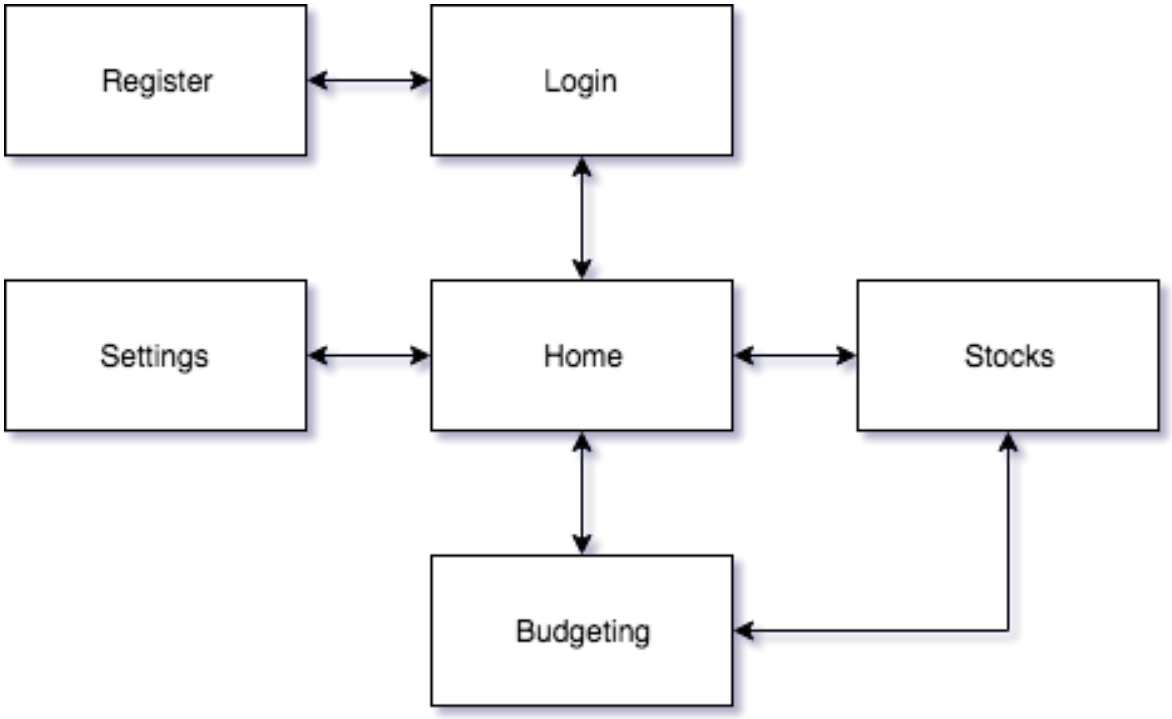
API(s)

Alpha Vantage - JSON of real time and historical stock and crypto data

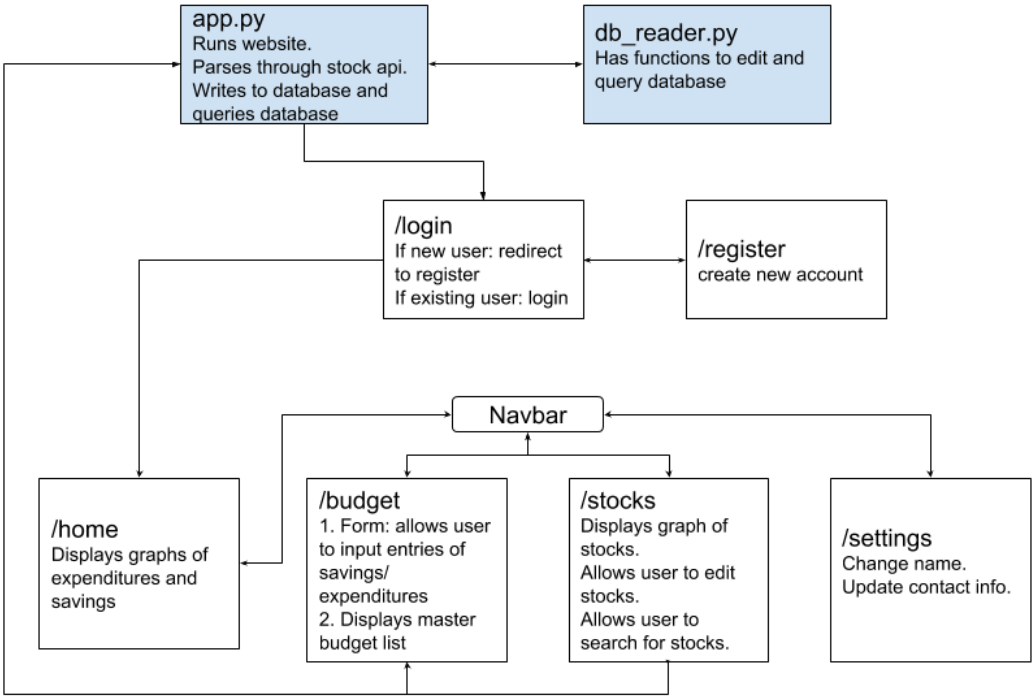
Stages of Development

1.	Setup login and register sites	5/16
2.	Setup database for user portfolios	5/18
3.	Create ability to submit spending and income	5/22
4.	Display spending & savings	5/24
5.	Create ability to buy/sell stocks	5/29
6.	Display stocks	5/31
7.	Display projections of savings and stocks	6/4
8.	Make pretty	6/10
9.	Buffer period	6/11
10.	Profit	6/12

Site Map



Components



HTML

- home.html
    - Displays charts/graphs of :
        - How much user spends each year, month, week (default is monthly). User can choose to compare it to his/her budget.
        - Savings and income for each year, month, week (default is monthly)
        - User's expected income for the next two or three months based on income, expenditures, stocks, and dividends.
        - The percentage of money user spends for food, clothes, housing, education, etc.
    - Displays breakdown of all charts and graphs
- stocks.html
    - Displays graph of loss/gains from stocks
    - Displays chart of stocks you own, including how many stocks and the worth of each stock
    - Can edit portfolio
    - Search bar to search market stocks
- budget.html
  There will be a toggle button at the top of the page to toggle between the form and master budget list. The form allows users to input entries into the database
    - Form fields
        - Input type (savings, spendings)
        - Description of input
        - Amount
        - Duration of saving/spendings
            - Ex. user gets paid every month, user spends money on food every week

  The master budget list will display all the inputs the users has put in. It can be sorted by type (saving/ spendings) and amount. When the master budget list is toggled, the user can also edit past entries.
- settings.html
    - Change name
    - Contact info
- register.html
    - Can register for account or log in

CSS

All pages have navbar

- home.css
    - NOTE: d3 heavy for displaying graphics
- stocks.css
- budget.css
- settings.css
- Bootstrap

Python

- __init__.py
  - budget
    - Handles form data on budget needs
  - config
    - Checks if the user has configured their profile
  - fixcost
    - Takes the form of fixed costs and adds it to the table
  - home
    - Main page for rendering information
  - login
    - Deals with login and verification
  - logout
    - Logs a user out
  - page_not_found
    - Basic 404 error page
  - register
    - Handles registration for new users
  - removefix
    - Takes form for removal of a fixed cost
  - removevar
    - Takes form for removal of a variable cost
  - root
    - Page that contains login and registration
  - settings
    - Page that contains the settings
  - stockdisplay
    - Parses stock data and helps create a chart
  - stockpurchase
    - Takes a form to purchase a stock
  - stocks
    - Page for mantaining stocks
  - stocksell
    - Takes a form to sell a stock
  - varcost
    - Takes the form of variable costs and adds it to the table
- db_builder.py
  - Adds the initial tables for new users
    - addAllocateTable
    - addMoneyTable
    - addUser
    - tableCreation

- Login user-related functions
  - checkUsername
  - check_password
  - dummyUser
    - Adds a dummy user for test purposes
  - getPass
  - getUserID
  - getUserName
  - Hash_password
  - setConfig
- Various mutators to edit database
  - addFixCost
  - addStock
  - addUpdate
  - addVarCost
  - bigUpdater
    - Maintains automatic savings and expenditures and updates savings and current income
  - changeMoney
  - updateAllocateTable
  - updateMoneyTable
- Various accessors to get values from the database
  - getAllFixCost
  - getAllStocks
  - getAllUpdateTable
  - getAllVarCost
  - getAllocateTable
  - getConfig
  - getFixCost
  - getMoneyTable
  - getPercentageByAllocation
  - getRecentUpdateTable
  - getStock
  - getUpdateTable
  - getVarCost
- Removing values from the database
  - removeFixedCost
  - removeStock
  - removeVarCost

Databases

users

INT userID	PRIMARY KEY TEXT user	BLOB pass	TEXT name	INT config

- userID = user id, used for easy reference to an account
- user = username to login
- pass = salted password
- name = name of user
- config = config status

money

INT userID	REAL currentMoney	REAL monthIncome	REAL otherIncome	REAL savings	REAL savingPercent

- currentMoney = current amount of money user has
- monthIncome = monthly income of user, automatically adds to currentmoney
- otherIncome = other income the user may have, resets every month
- savings = total savings
- savingPercent = how much of the monthly income is added to savings

allocate_table

INT userID	REAL entertainment	REAL eatOut	REAL shop	REAL misc	REAL grocery	REAL event

- All of them are just how much a user wants to allocate for each category

fixedCosts

INT userID	INT expID	TEXT fixedName	REAL fixedAmt	TEXT fixedDesc

- fixedName = name of the fixed cost (rent, bills etc), user created
- fixedAmt = monthly cost of fixed expense, automatically deducted
- fixedDesc = description, user's description of the fixed cost

variableCosts

INT userID	INT expID	TEXT expName	INT expType	REAL expAmt	REAL expBud	TEXT expDesc	TEXT dateof

- expName = expenditure name, user creates it
- expType = type of expenditure (food, leisure, etc.), user selects the type; referenced as by an INT id
- expAmt = expenditure amount, user entered
- expBud = expenditure budget, user sets how much they wanted to budget
- expDesc = description, user's description of the expense
- dateof = date of the entry

stocks

INT userID	TEXT ticker	INT current Shares	INT totalShares	INT totalShares Sold	INT totalLoss	INT totalGain

- ticker = stock abbreviation
- currentShares = number of shares at the moment
- totalShares = total number of shares ever bought
- totalSharesSold = number of shares ever sold
- totalLoss = total cost of every share
- totalGain = net profit when selling a stock

Roles

Project Manager and Data Visualization – Henry
Backend – Xing Tao, Jackie
Databases / Frontend – Eric