Name — Andrew Juang, Eliza Knapp, Patrick Ging, Yuqing Wu
Softdev
P01: ArRESTed Development
2021-12-08

Program Components

- Login system
- Leaderboard (closest to 0 at any given time)
- Buy & Sell stock (search for stock symbol)
    - Error message if can't find stock
- See info/news about stocks
- See other people's stocks
- Update information/leaderboard
- You win when the net worth is 0 (*when you create an account, everyone starts with the same amount of money)
- Each time you win, you add one to the score and the money resets so you can try again
- Update info every time the page that displays relevant information about a stock is refreshed.

Component Relationships

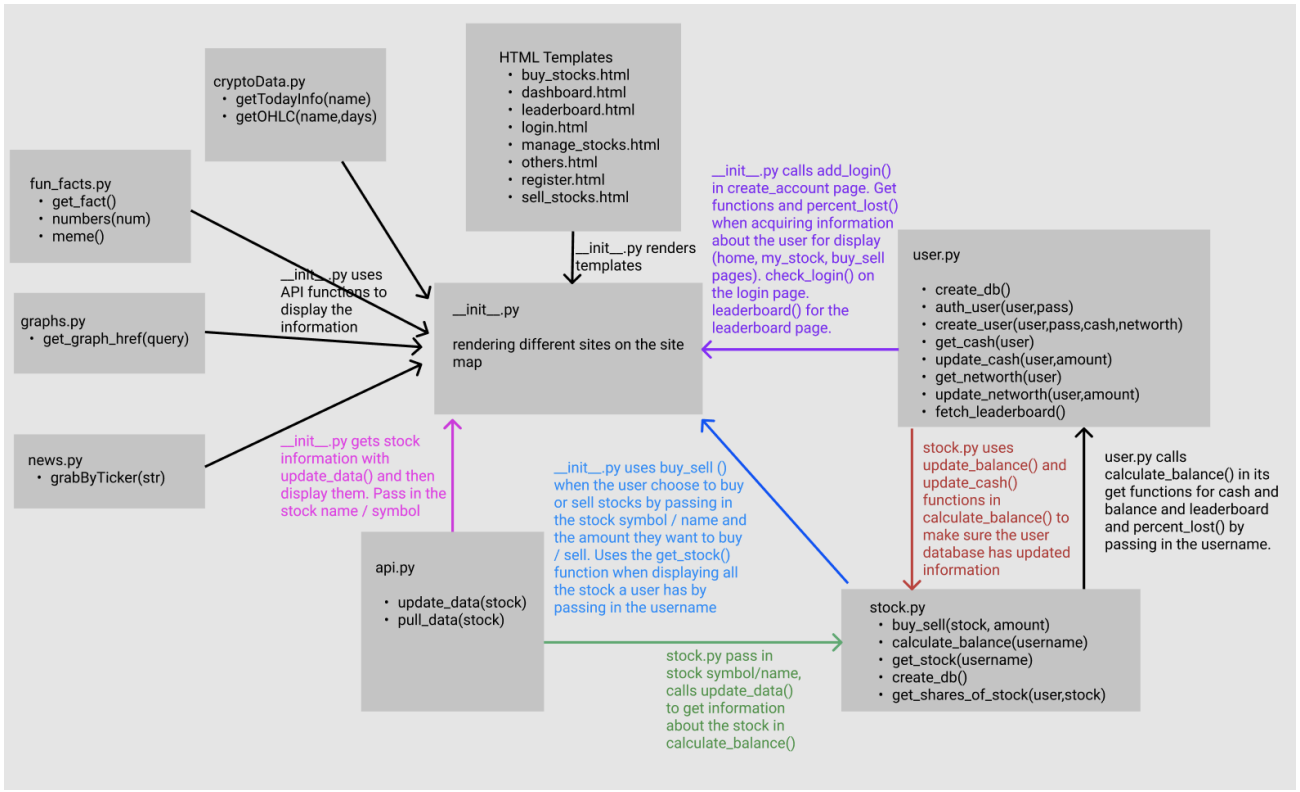
- __init__.py uses the HTML Templates, api.py, stock.py, and user.py
    - Renders the different routes using the HTML Jinja templates
    - Uses functions from user.py to interact with the user SQLite database and create basic register/login/logout functionality
    - Uses functions from stocks.py to interact with stock SQLite database
    - Uses functions from api.py, graph.py, news.py, fun_facts,py, cryptoData.py to pull data from apis.
    - Details:
        - login page:
            - Checks username and password.
            - create account page
                - Add login information to database
        - home page
            - Displays balance, cash, updated stock information.
            - Lottery to win & lose certain money
        - Manage stock page:
            - Display stock with updated information
            - Sell stocks
        - buy and sell page:
            - Search information for a stock
            - Buy some shares of a stock
        - leaderboard page:
            - Leaderboard ranked with lowest networth winning
            - view other people's profile page when click on links
        -

- user.py does database operations on the user table in database
    - create_db() set up the user database
    - create_user(), add login entries (creating account, initialize money)
    - functions to get all the information in the user database
    - auth_user() checks if username and password matches
    - update_networth () changes the net worth of the user's stocks + cash, update_cash () changes the amount of cash the user has
    - get_cash(), get_networth() gets cash and networth of a user.
    - fetch_leaderboard () function to return sorted leaderboard, will be used in __init__.py leaderboard page.

- stock.py uses api.py and user.py to update a certain user's stocks
    - create_db() set up stock database
    - calculate_networth() calculates the networth given stocks owned by user
    - get_shares_of_stock() gets all the updated information about the stocks a user owns, pass in the username of the user, and will be used in __init__.py for displaying the stocks a user owns and updating data.
    - buy_sell() function to modify stock database (buy & sell certain shares, buy + amount, sell - amount, as parameters)
        - if sells all shares remove stock from database,
        - if stock didn't exist, add to database

- api.py gets stock information from apis.
    - update_data() function that pulls new data from api regarding stock prices.
- cryptoData.py gets information regarding crypto from api
    - getTodayInfo() gets cryptocurrency name, symbol, social media data etc.
    - getOHLC() gets graph of a crypto for a time frame
- fun_facts.py provides fun number fact based on lottery value
    - numbers() gets data from numbers api
- graph.py graphs stock prices
    - get_graph_href() creates graph, save graph, returns the file path
- News.py gets news data of a stock
    - grabByTicker() gets news from the news api, give the token and symbol of stock to get.

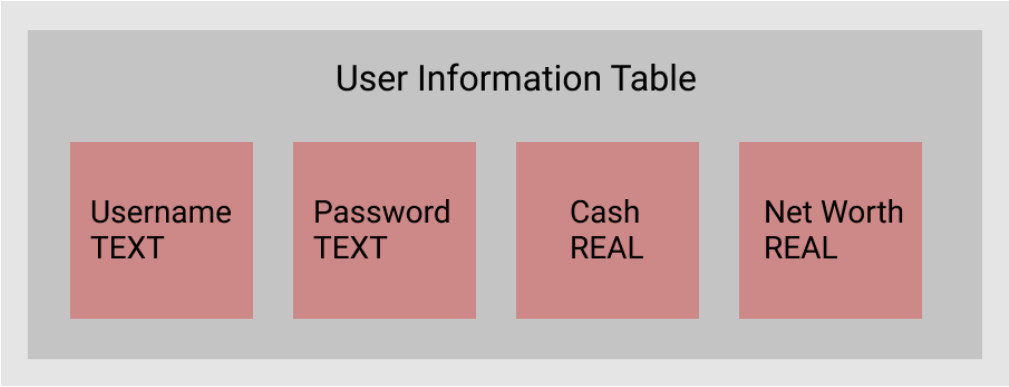
https://www.figma.com/file/QgfTb1xFWV62GsHPXuDjTA/Component-Map?node-id=0%3A1



Database

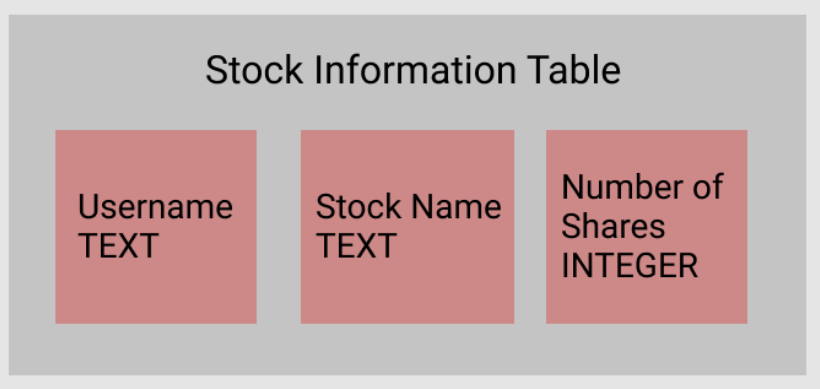
User database- *each new entry is a new user*

- Username | password | total money left not in stocks (everyone starts with the same amount) | net worth (money + money in stocks)



Stock database- *each new entry is for a new stock purchased by a certain user*

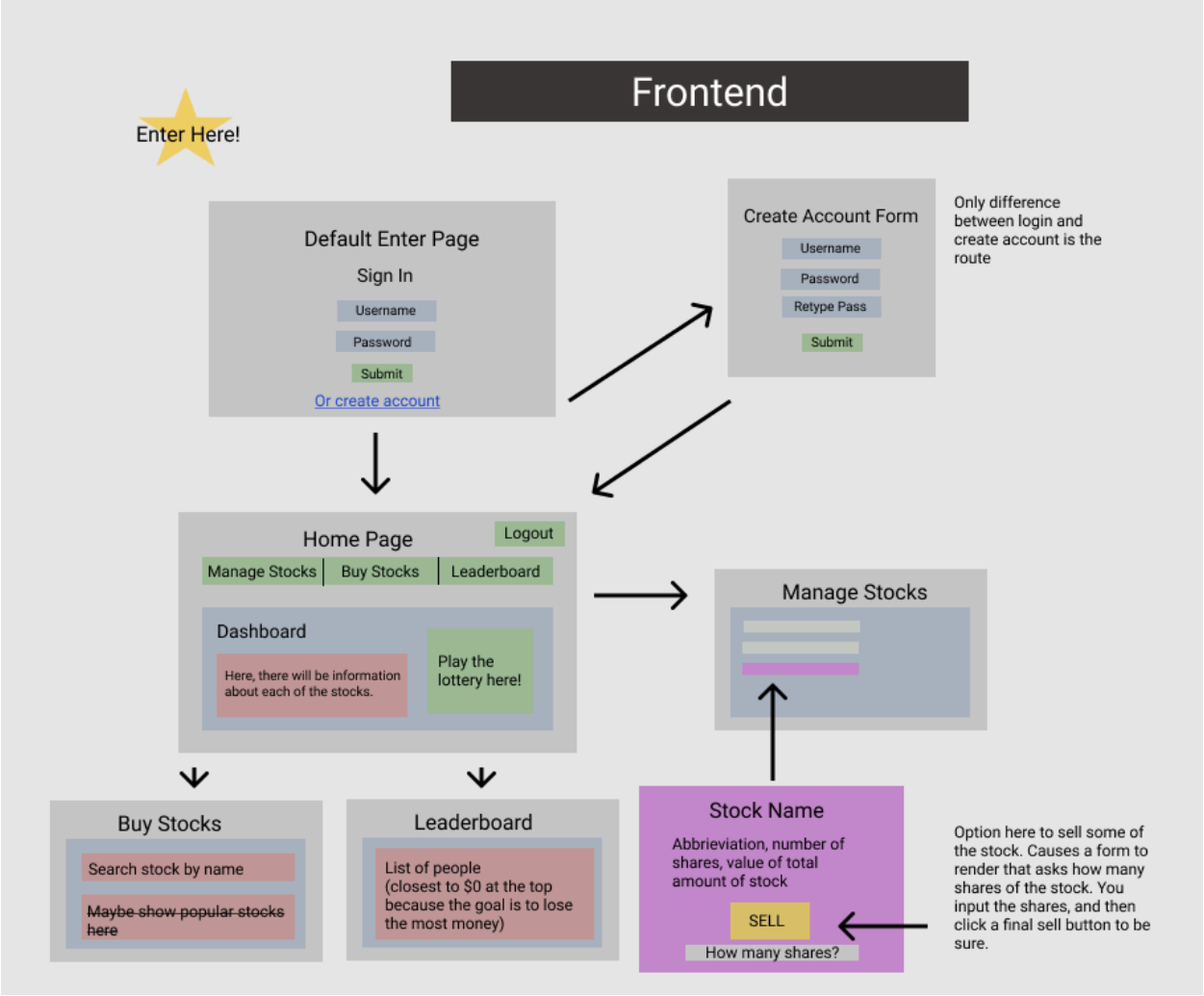
- username | stock name | how many shares



Relationships

- When a user creates an account, they create an entry in the user database with a username, password, and net worth + total money set to a given amount
- Each time a user buys a stock, it creates an entry in the stock database with their username and keeps track of the stock and how many shares
- When we want to render the stocks of a user, you search the stock database where the username corresponds.

- Site map for front end



- A breakdown of the different tasks required to complete this project, with target ship date
    - Pat: API for stock and crypto information, news, graphs, fixed leaderboard link
    - Andrew: Login, graphs with Pat, functions in user.py to get information about user, leaderboard
    - Eliza: Bootstrap, flask for stocks, search function, sell stocks flask
    - Yuqing: stock database functions, lottery & numbers api.

APIS

- NewsAPI
    - A rest API enabling us to grab news from a given period of time surrounding a particular topic, works well with stock tickers. Provides a rich amount of data including links to articles, their descriptions, etc.
    - Extremely simple to use and legible responses
    - Does require an API key, but it purportedly supports 8,000 requests a day. It would be able to suffice even during a hypothetical competition with 80 students.
- Numbers API
    - Get random facts about a number when you pass in a number.

    - Does not require API keys.
    - We are using it for the lottery feature where the random number fact is about the number of the lottery
- Coingecko API
    - High quality, simple API for grabbing statistical and real time market data
    - Quota, but it's VERY generous.
    - Simple and reliable

Frontend Framework

Why Bootstrap?

We will use bootstrap for this project. First of all, three out of four of us have already become familiar with bootstrap through the frontend frameworks assignment. Second of all, because bootstrap has a wide variety of easily combinable components and we aren't masters of css/designing, it is probably better for us to be less unique while creating something that looks decent. The features of bootstrap that we are currently thinking of using the tables and flexboxes to make our page easily resizable. We will also use the nice designs for button creation and navbars. Also, on our login page, instead of having a create account and login button, we are thinking of having a get started dropdown menu and putting create account and login there. Bootstrap also has an interesting chart functionality which we haven't quite yet looked into in conjunction with the graph data we will receive from the API but we think that it will be possible to incorporate the information together to make nicer graphs.

Because it was difficult to figure out how to manipulate different color choices in bootstrap, we added css in addition to the bootstrap to supplement some parts, in particular, the color, the background color, and the width of different components.