

Name — Andrew Juang, Eliza Knapp, Patrick Ging, Yuqing Wu  
Softdev

**P01: ArRESTed Development**

**2021-12-08**

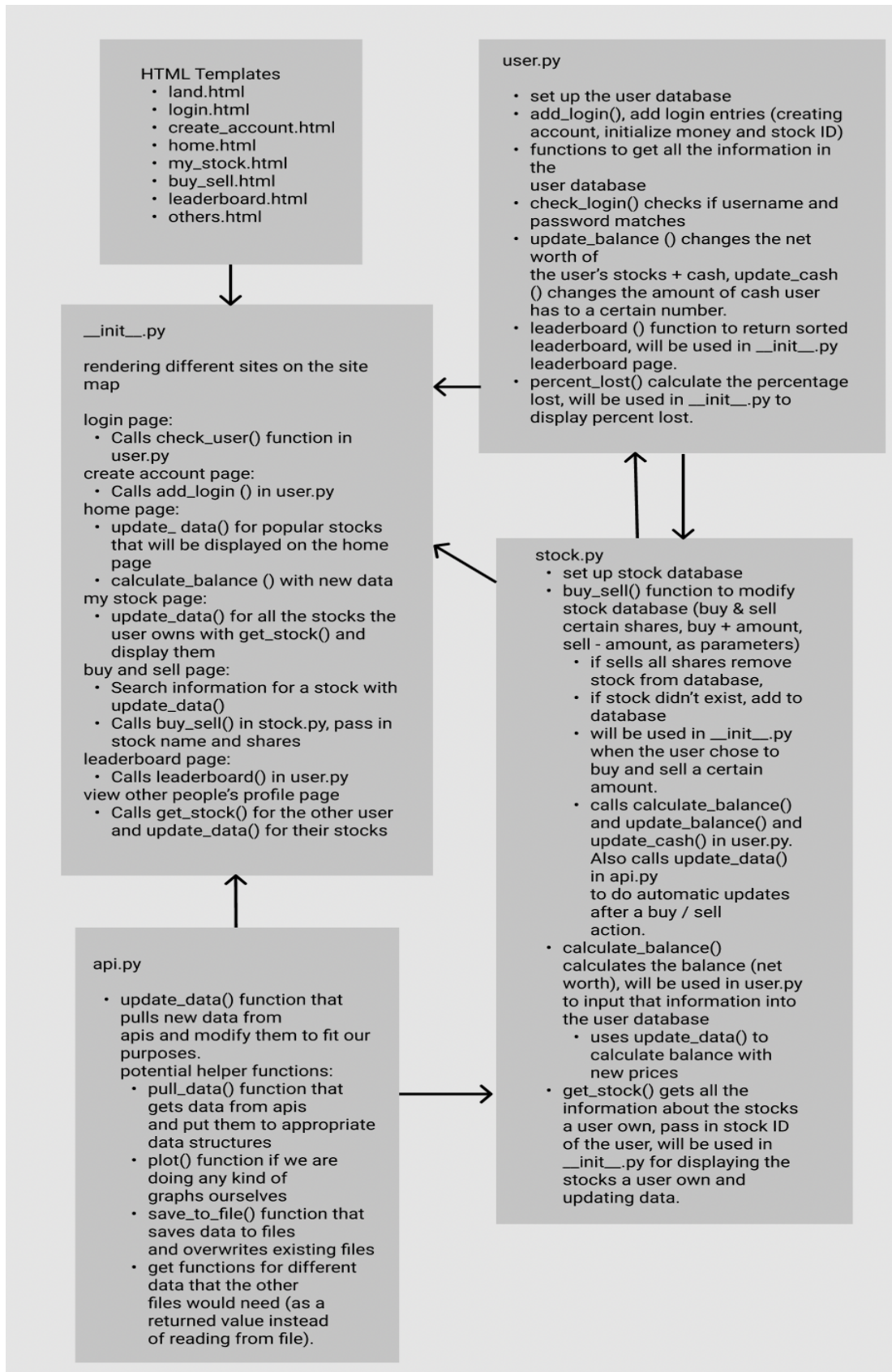
### **Program Components**

- Login system
- Leaderboard (closest to 0 at any given time)
- Buy & Sell stock (search for stock symbol)
  - Error message if can't find stock
  - *Popular stocks - optional*
- 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

### **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` to pull data from the REST api
- `user.py` does database operations on the user table in database
- `stock.py` uses `api.py` and `user.py` to update a certain user's stocks
- `api.py` gets information from apis.

<https://www.figma.com/file/QgfTb1xFWV62GsHPXuDjTA/Untitled?node-id=0%3A1>



## Database

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

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

User Information Table				
Username TEXT	Password TEXT	Stock ID INTEGER	Cash REAL	Balance REAL

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

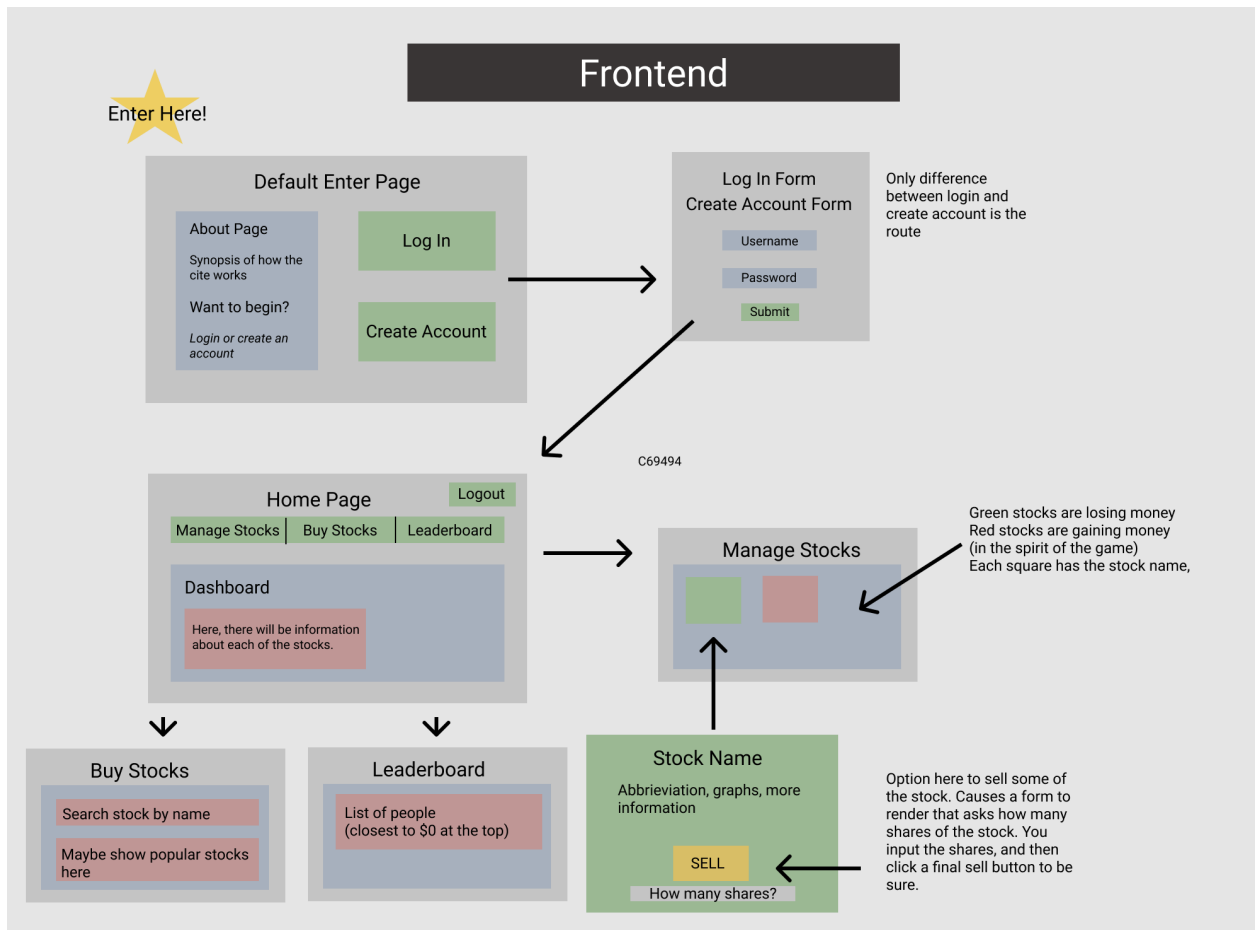
- username | stock name/symbol | how many shares

Stock Information Table		
Stock ID INTEGER	Stock Name / Symbol TEXT	Shares Number INTEGER

## 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 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
  - 12/13
    - Login system from last project + initializing money & stock ID - Andrew
    - Establish all the html linking & render templates - Eliza
    - Setting up apis (get keys and check is able to retrieve info we need) - Pat
    - Setting up all the python files with docstrings and function headers - Yuqing
  - 12/15
    - Buying and selling stocks (do this on like 5 stocks first), adding this information to the database, nothing with prices yet because the api is not set up. - Eliza & Yuqing
    - Getting info from api to our site, at least get the prices for certain stocks. - Andrew & Pat
  - 12/17
    - Putting buy and sell and api prices together so that it actually displays the balance correctly. - Eliza & Yuqing
    - Updating api & balance data with refreshing - Andrew & Pat
    - The entire base of the game should work and you should be able to play it
  - 12/20

- Searching stocks for buy and sell, actually able to buy / sell whatever stock we want. - Eliza
  - Leaderboard - Andrew
  - Displaying additional info with API (like company news and stuff) - Yuqing & Pat
- 12/22
  - Making sure everything works, error handling, debugging
  - Displaying additional info with API (like company news and stuff) if more time is needed. - Yuqing & Pat
  - Viewing other people's profiles - Andrew & Eliza
- During break
  - Finish everything
  - Bootstrap, css
    - Collaborate on a base template for all of the pages
    - Dividing up the pages for fine tuning each page if needed.
  - Debug
- Target ship date: Jan 4th

## 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.
- Finnhub.io API
  - An API providing copious amounts of data, however we intend to use this to aid us in creating candlestick graphs.
  - Very simple just like the NewsAPI
  - This requires an API key, their free plan supports 60 requests a minute, so it might crash while under heavy load. We might need to stack API keys in the case of 429 errors.
  - Might also be useful for other charts and statistical data. To be decided what else it is used for considering it has a 60 call/min cap.
- Yahoo Finance API via third party wrapper
  - The Yahoo finance API provides a lot of individual data regarding stock prices, news, volume, etc.
  - It does require a key, but with the use of this exceptional third party wrapper, it seems volume isn't an issue.
  - We shall see the limitations of this wrapper soon, but right now it's working brilliantly.
  - One thing is that it does have a lot of dependencies....so we're going to see if this is a problem.

## 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.