PHP Website Project

[Presented by - Team LeftBracket] • [Shuwen Ju - Gordon Bert - Mohamed Rabi Andaloussi]

Intro:

Our project aims to create a web application that functions as a personal stock wallet, allowing users to keep track of their stock purchases and view real-time data on the market. We will be using the polygon.io API to pull real-time stock data and display it to the user. Our website will have a user registration and login system, allowing users to access their own personalized portfolio pages where they can add, view, delete, and modify their bought stock information.

Landing page

Login page

Registration page

Portfolio page

Website flow

Homepage:

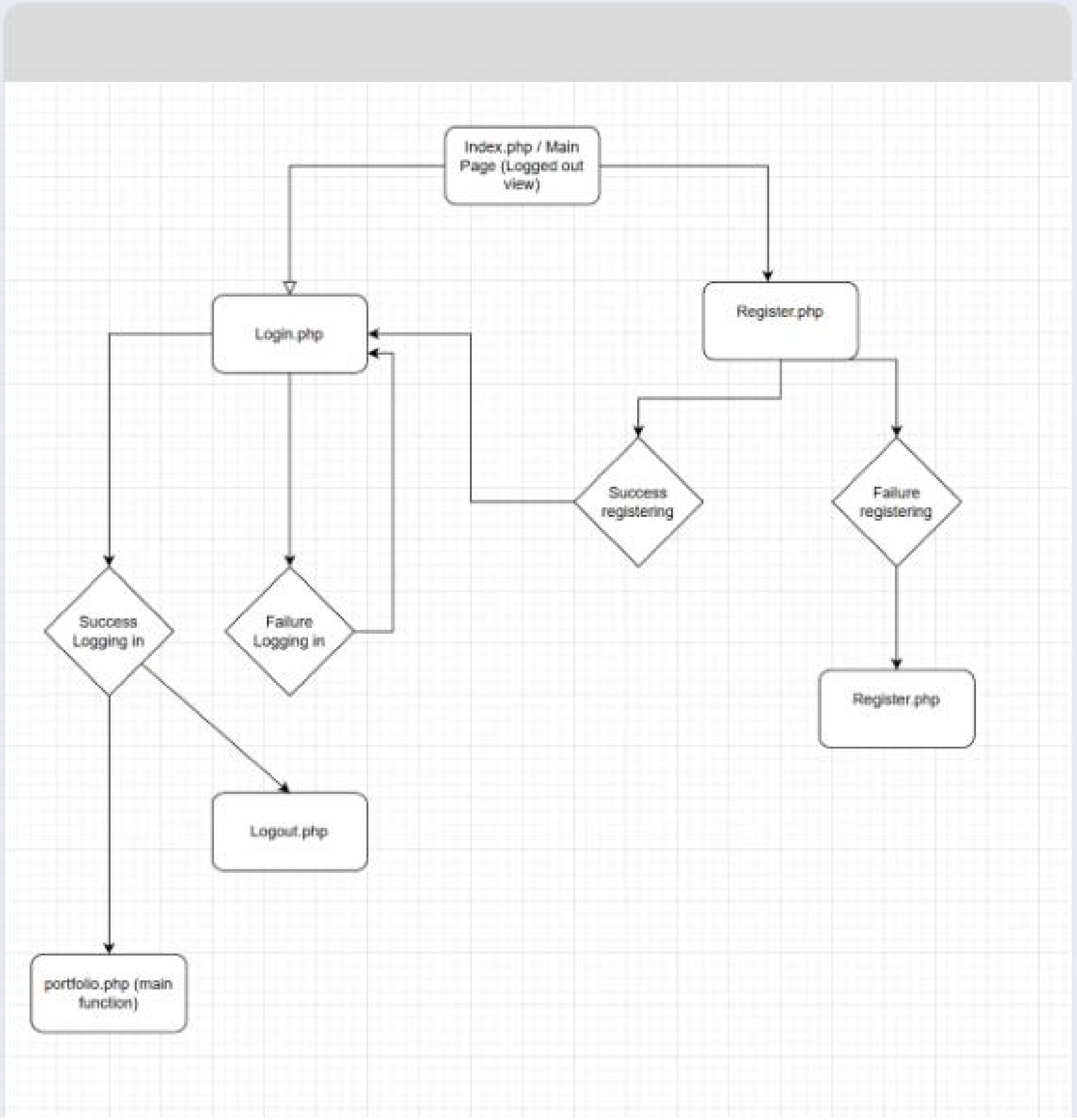
- Introduction of the website and its services
- Display of a random selection of stocks from our database
- Navbar with links to register or sign in when user is not logged in;
- When user is logged in, the nav bar only shows user name and link to log out.

Registration page:

- Form for user registration
- Upon successful registration, redirect to the login page

Login page:

- Form for user login using email and password
- Upon successful login, redirect to the user's portfolio page



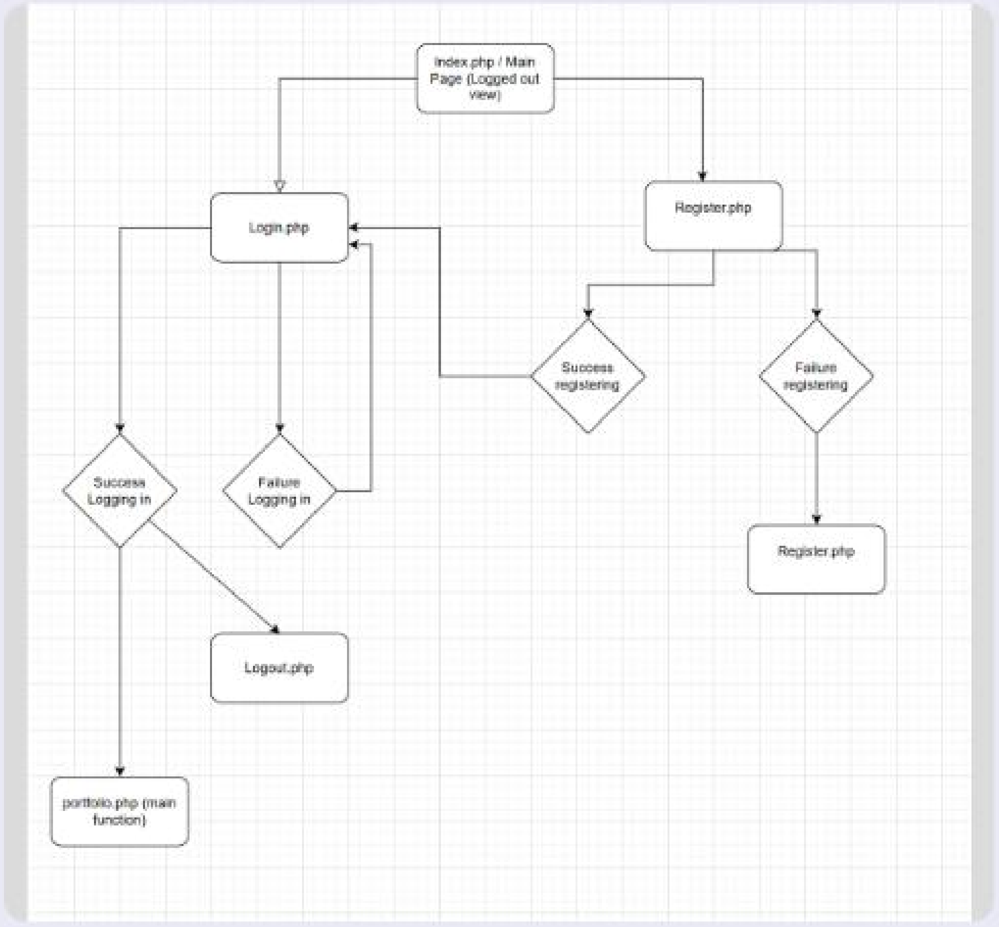
Website flow

Portfolio page:

- Display of user's current stocks with details such as stock name, ticker, initial price, and buying date, compare to today's value showing assets gaining/losing in percentage.
- Option to add, delete, and modify the user's bought stock information
- Real-time data on market prices of stocks and the comparison with the user's bought price
- Use of charts and graphs to support the visual representation of data

Logout:

- Ability to logout from the website



Registration system features:

- Validation - Password Hashing - Session Management - Error Handling

User validation details:

The system will validate user input using the following rules:

Username:

- Must be between 5 and 20 characters long
- Must only contain letters, numbers, and underscores
- Must start with a letter

Password:

- Must be at least 8 characters long
- Must contain at least one uppercase letter, one lowercase letter, and one number
- Must not contain any spaces or special characters

Email:

- Must be a valid email address format
- Must not already exist in the system

Address:

- Must not be empty
- Must contain at least one space

First Name and Last Name:

- Must not be empty
- Must only contain letters
- If any of the user input fails to meet these validation rules, the system will display an error message to the user indicating which fields need to be corrected.

User portfolio - CRUD stock functions - Personalized stock management

This feature allows users to add information about the stocks they own to their portfolio.

When a user adds a stock, they will need to provide information such as the stock ticker, stock name, the date they bought the stock, and the number of shares they purchased.

Each time a user adds stock information, it is saved in the database as a unique transaction id.

User is able to edit and delete stock later as they wish.

Database tables

Overall

Three tables

User_info table:

- Saves user information.
- Through validation, securely store user info.
- Let user stay logged in through session management.

user_id	**username**	**password**	email	address	first_name	lst_name
1	shushu	$2y$10$GKKruoMpe	shu@mail.com	123 Rue Ave	Shu	Jus
2	icarus	$28Dkm4$wlb12aeZ	ica@mail.com	64 Alley Rd	Ica	Rus

Database tables

Stock_info table:

- Saves stock information that the website offers to users.
- Landing page connects to this table to randomly populate the stock info.
- When adding/editing stock, we refer to this database to make sure the stock ticker and stock name match.

stock_info_id	**stock_name**	**stock_ticker**	stock_owner	stock_description	country_of_company
1	Apple Inc.	AAPL	Apple	Multinational technology compar	USA
2	AT&T Inc.	T	AT&T	American multinational telecomm	USA

Database tables

User_financial_info table:

- Saves user stock transaction information.
- User is able to delete, edit and add stock/transactions.
- With the buying date, initial price and quantity values, we can display to user his original assets, comparing to our api drawing real-time today's value.

user_stock_id	**user_id**	**stock_ticker**	stock_name	stock_initial_pri	buying_date	quantity
1	1	AAPL	Apple.Inc	121.13	2022-12-13	1.3
2	2	AMZN	Amazon.com Inc.	133.31	2021-03-11	2.2

Technologies:

Front-end:	HTML, CSS, Javascript, Bootstrap
Back-end	PHP, MySQL
APIs	Chart.js, polygon.io

Difficulties and contributions

Difficulties:

- The polygon.io api is free, but only limits 5 calls per minute, which means while our team works together, when testing exceeds the 5 per min calls, we get error messages.
- The polygon.io api is a real-time stock api, which means it does not work on the current day, it also does not work on weekends, so in our validations, we have to be extra careful about the dates and date formatting as well.
- Incorporating javascript with PHP was a challenge. It was really difficult to read and syntax errors often comes up.
- Using bootstrap was convenient but also difficult to change the styling.
- It was really difficult when using modal to save changes/add stock on the portfolio page with php validations since bootstrap by default closes the modal on submit, we couldn't prevent the modal to close, so user has to open the modal again to see the error messages.

Shuwen Ju:

Landing page, templating, portfolio page table for user, portfolio page Api for real-time data display on the charts, edit stock modal and stock validations.

Gordon Bert:

Debugging styling, user login page, user registration page, user validations, portfolio page table pulling real-time data showing % difference, modal debugging.

Mohamed Rabi Andaloussi:

Debugging styling, user login page, portfolio page add stock modal, general debugging, stock validations.