

COP290: Assignment 1

Aviral Singh (2022CS11290)

Daksh Dhaker (2022CS51264)

Subtask 2:

Tech Stack

Backend

- Flask
- Jinja
- jugaad-data
- SQL-Alchemy

Frontend

- Graph.js

Design Choices

Logo

The logo is AI generated. OpenAI's DALL-E2 was used to generate it. The following prompt was used. "an icon of a bar graph in light blue metallic iridescent material, 3D render isometric perspective on dark background"

Authentication and Security

In the management of the website's session authentication, the implementation involves a conditional handling of session permanence based on the user's selection of the "remember me" option. When the user opts for "remember me", the session is set to permanent(True); otherwise, it is set to non-permanent (False).

If the session is marked as permanent, and the user closes the browser without logging out, the session stays active. The next time they visit the login page, they're automatically redirected to the dashboard without needing to log in again. If the user logs out, the session ends and they have to log in again later.

Additionally, to access any page on the website, you must log in first or have an active session from the last login. Trying to access a page directly through its URL is restricted and if they do so, they are redirected to the login page. The session is set to permanent until the next login occurs within 30 days. This approach enhances the account security and allows users to return to the dashboard smoothly without repeated logins.

Filter

We have used 4 filters for stocks:

- Close Price - It is the close price of the most recent stock. The close price being the last traded value of the day, reflects a more stable and reliable snapshot of market sentiment. The closing price's role as the settlement price for daily transactions ensures accuracy and incorporates the latest available information.
- Relative Strength Index(RSI) - It has the ability to assess the stock's momentum and potential overbought or oversold conditions. RSI values range from 0 to 100, with readings above 70 indicating potential overbought conditions and readings below 30 suggesting potential oversold conditions. For calculation of average gain and average loss in RSI, we have averaged them over a window of 3 days.
- Average Price : It offers the advantage by providing a balanced representation of the day's trading range. This dynamic measure is flexible, supporting trend identification and offering a midpoint perspective between bullish and bearish extremes.
- Value/ Volume ratio: A low value/volume ratio might suggest that the stock is undervalued, especially if accompanied by increasing volume. Conversely, a high ratio could indicate over evaluation, potentially signalling a reversal. Traders often look for such anomalies as they may signal significant prices.

The implementation of the search functionality on the client side (`filterSearchResults()`) reduces the need for server requests, improving the responsiveness of the application.

We used html forms for filters as it would reduce client side overhead.

The initial set of search results is embedded in the HTML as a JSON object (`'searchResults'`). This approach can reduce the need for immediate server requests upon page loading, contributing to a faster initial loading time.

The code creates a clear and user-friendly display of search results, recent percentage gains , and arrow indicators.

It also incorporates conditional styling based on the direction of percentage change, with red arrows for losses and green arrows for gains. The page is also fully responsive.

Favourites

In the dashboard page, users have the option to personalise their experience by adding their favourite stocks. Once added, these selected stocks are prominently displayed as individual cards on the dashboard. Each card provides a snapshot of the stocks' recent percentage profit or loss, accompanied by a visual indicator in the form of a red or green arrow pointing downwards or upwards respectively.

This design facilitates a quick and convenient glance at the trends of the user's favourite stocks directly from the dashboard. Without the need to navigate to the filters page for analysis, users can efficiently monitor the performance of their selected stocks and stay informed about market trends. This approach aims to enhance the overall usability and accessibility of our dashboard, providing a seamless and personalised experience for investors and traders.

The code iterates through the lists of valid stocks and processes them in a loop. This batch processing approach minimises the number of calls to external data sources as the search occurs in only recent stocks which are directly fetched from `ans_df`, thus optimising resource utilisation.

The addition of new favourites and their display are handled on the client side without the need for server round trips, contributing to a more responsive user experience. This reduces the server load by offloading some rendering responsibilities to the user's browser.

The code also includes error handling of the cases where the new favourite stock is not present in the valid stocks or when the user attempts to add a stock that already exists in their favourites.

The addition and removal of favourite stocks are achieved through form submissions, which is a standard and efficient way to interact with the user. The page is fully responsive.

Graph

The graph itself was made using Chart.js, a popular Javascript plotting library.

As asked in the problem statement, we can plot various metrics for any NIFTY50 symbol for as many years, months, weeks as we would like.

We can compare metrics with as many other symbols as we want at the same time.

Using Chart.js we're plotting multiple stocks on the same HTML canvas for ease of viewing.

We have also incorporated a stock ticker that calculates a percentage delta of the selected metric over the selected time frame.

The chart itself has been made fully responsive.

You can hover over the line graph to see data about a particular day.

We have implemented panning and zooming using extensions for Chart.js. You can also zoom into a particular region by pressing ctrl, then selecting the desired region. To reset the plot, you can simply press the plot button again.

For better visibility we've used different colours for different stock plots on the graph. We decide to go with really vibrant auto generated colours using a Chart.js extension called auto-colours.

For fast load times, we're using a CDN to deliver Chart.js.

We have used the JS fetch api for all requests,, with error handling, ensuring error handling. Any invalid stock symbol input will be highlighted in red at the end of a plot request. All other symbols will be drawn.

All CSS has been made from scratch using CSS's flexbox. The colour scheme is kept consistent with our icon's accents.

API

For our front end data fetching needs, we've implemented an API wrapper around the jugaad-data library.

We decided to use JSON based responses, as we can jsonify pandas' to_dict output which we directly got from jugaad-data.

We're also statically storing all the NIFTY50 stock names (as they don't change often). This helps us in performing stock symbol verification in various places in the frontend.

Deployment

For deployment we decided to use gunicorn, a popular unix-only wsgi server.

Insights

jugaad-data is a serious bottleneck for performance.

We've decided to display data of only NIFTY50 stocks, as we value robustness over quantity. To add some context, a lot of stock data is missing for hundreds of stocks through the jugaad-data API, hence we decided not to include stocks our software can't deliver on.

Features

1. Dashboard:

- **Favourites list** : User can add or remove stocks from his favourite list. This is done so that it is easy for the user to have a glance at the current trend in stocks of his interest.
- **Use of stock ticker**: We have used an upward/downward arrow for representing profits/losses for recent trades. This helps the user to have a better clarity in differentiating the stocks on the basis of profit or losses just by viewing.

2. Filters:

- **Search bar**: We have used a search bar to search from the filtered stocks. It makes it easy for the user to search for a particular stock. It also adds to the dynamic user-interaction.

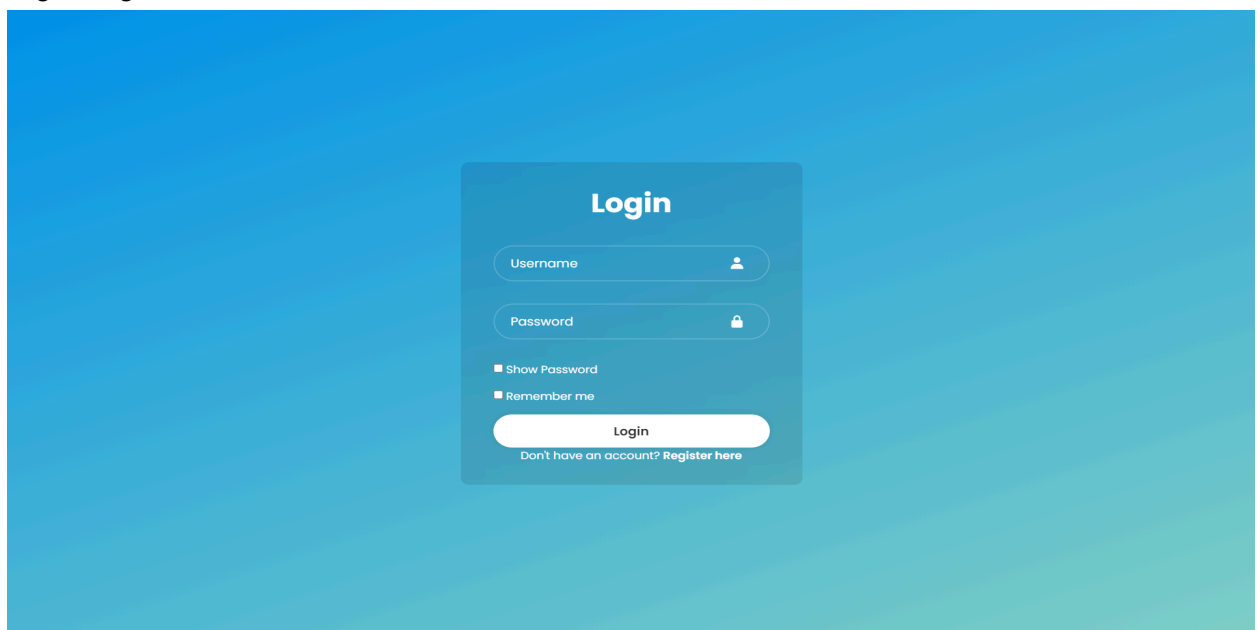
3. Graph:

- **Multiple Stocks Comparison**: Any number of stocks can be compared simultaneously. The comparison fields are added dynamically by adding html code using javascript at the time when the user presses the add button to add fields.

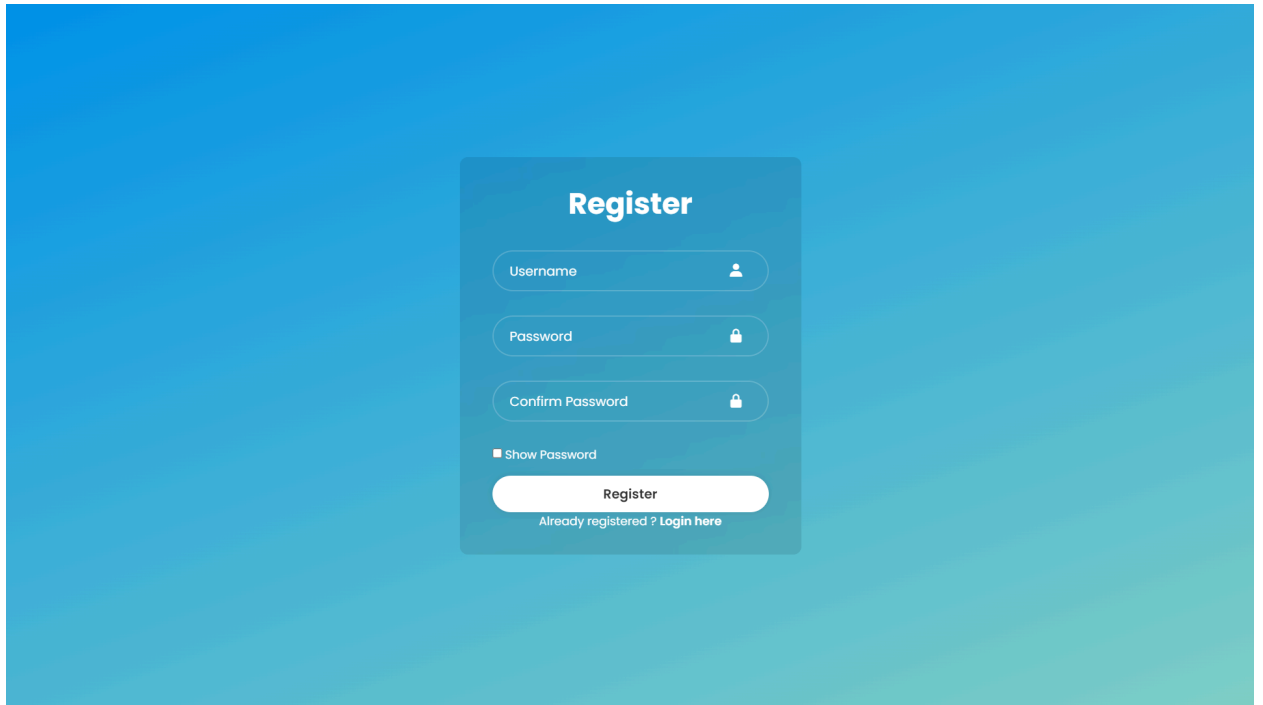
4. All the pages are made fully responsive and are adaptable to any size of the screen.

Screenshots of website pages

1. Login Page





2. Register Page




The Register page features a central white form on a blue background. The form is titled "Register" and contains fields for Username, Password, and Confirm Password, each with a corresponding icon (person, lock, and lock respectively). Below these fields is a checkbox labeled "Show Password" and a large "Register" button. At the bottom of the form, there is a link that says "Already registered ? Login here".

Register

Username 

Password 

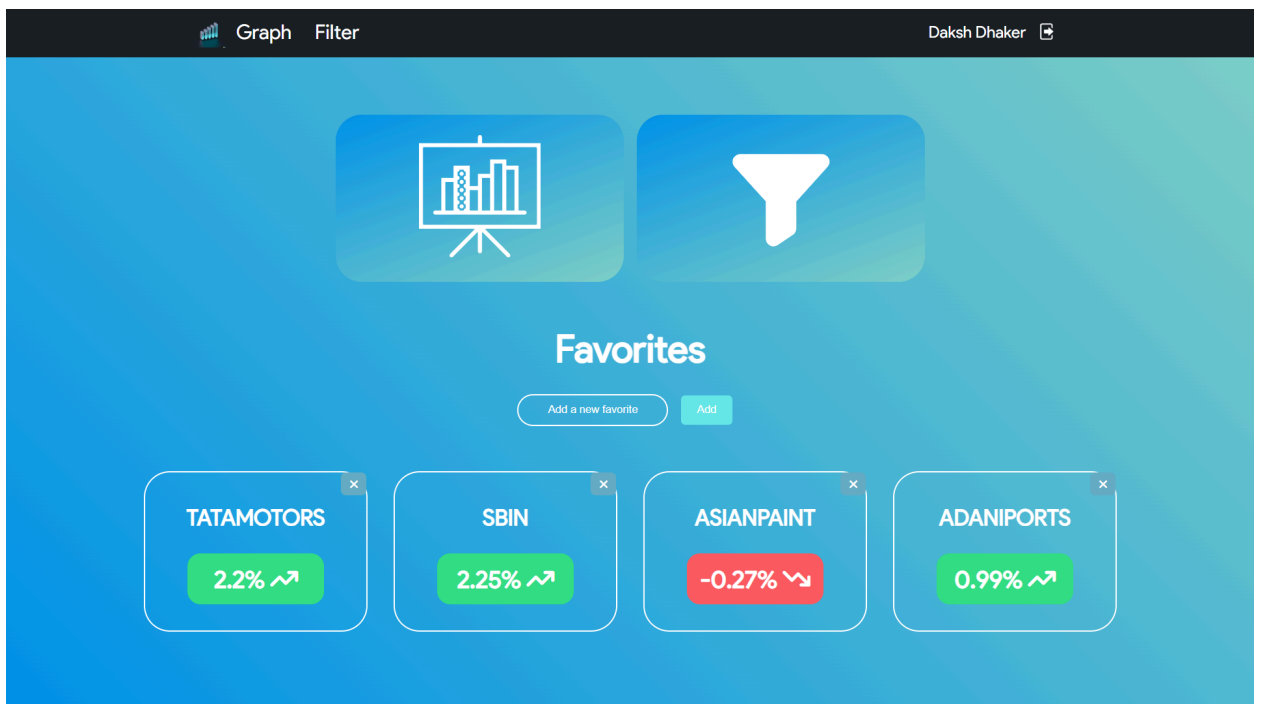
Confirm Password 

☐ Show Password

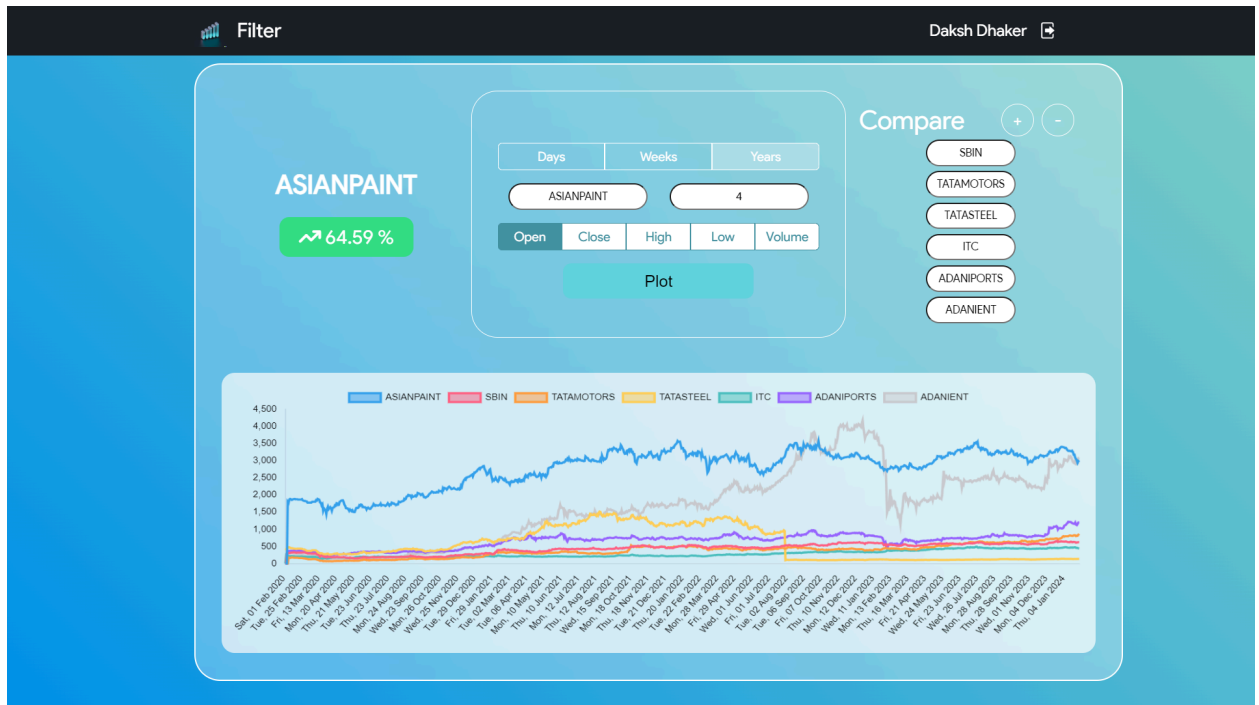
Register

Already registered ? [Login here](#)

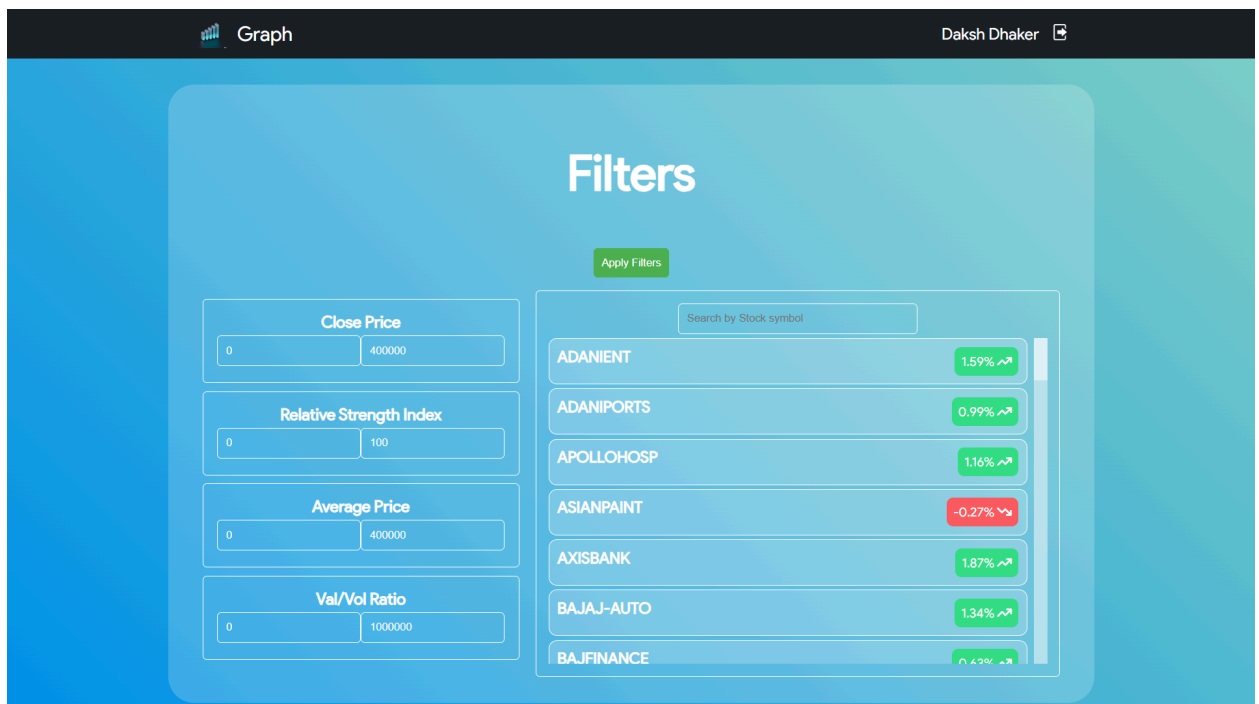
3. Dashboard



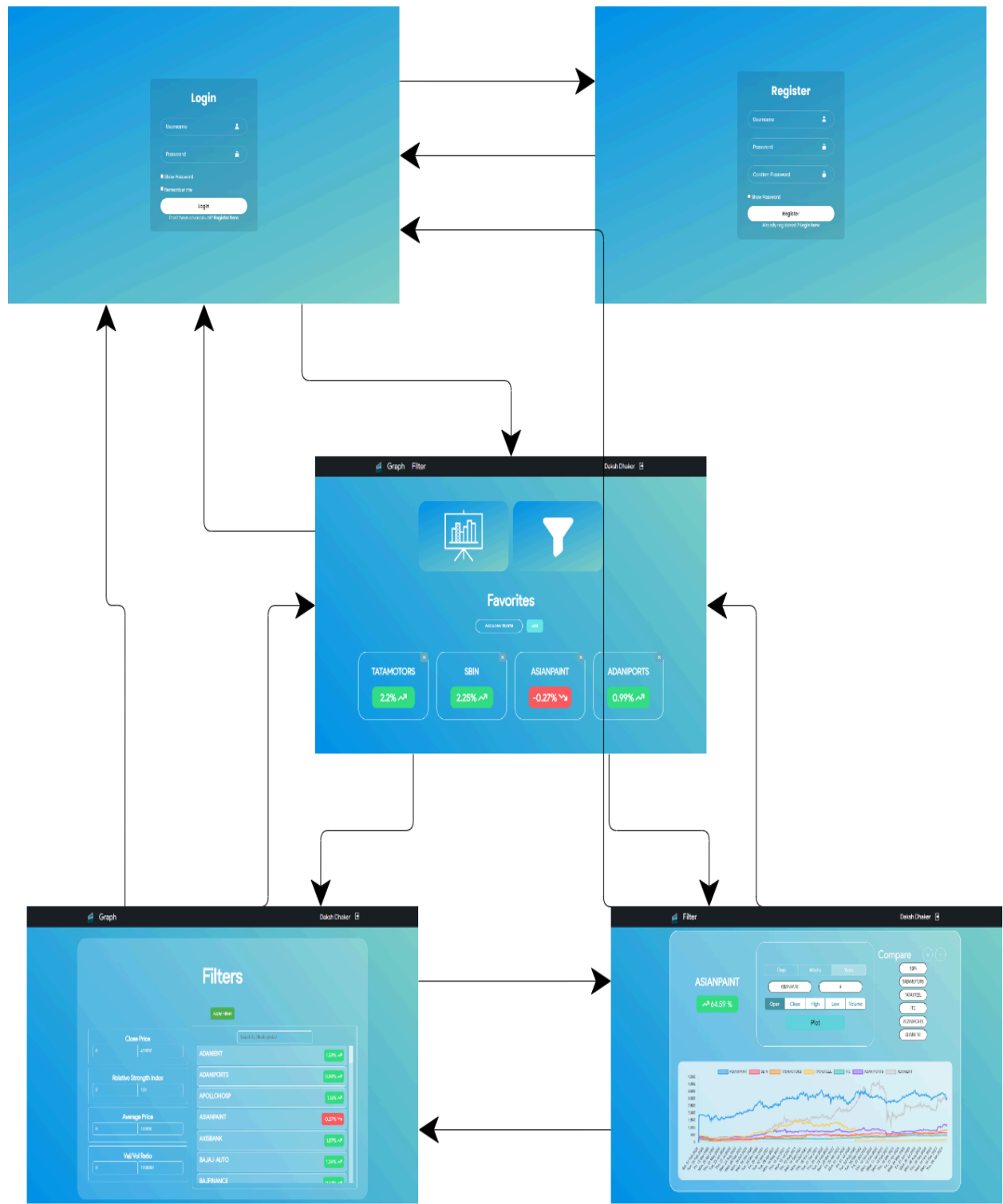
4. Graph



5. Filters



Website Map



Commit History



Commits		
main	All users	All time
Commits on Feb 1, 2024		
Remove extra files	CosmicPegasis committed 2 minutes ago	3ae4392
Fix colors	CosmicPegasis committed 48 minutes ago	3bd355a
Relayout	CosmicPegasis committed 1 hour ago	b0e992f
Improve welcome page	CosmicPegasis committed 4 hours ago	21a0af0
Make graph responsive	CosmicPegasis committed 5 hours ago	667faf4
Fix welcome page	CosmicPegasis committed 6 hours ago	cd95daf
Add favicon on login and register page		

Commits on Feb 1, 2024		
Minor changes	Daksh Dhaker committed 8 hours ago	279768c
add filters.py	Daksh Dhaker committed 8 hours ago	d486555
Commits on Jan 31, 2024		
Add username on navbar	Daksh Dhaker committed yesterday	77eeef9
Commits on Jan 30, 2024		
Minor changes	Daksh Dhaker committed 2 days ago	486b369
Commits on Jan 29, 2024		
Add requirements.txt	CosmicPegasis committed 3 days ago	9924631
Commits on Jan 28, 2024		
Modifications	Daksh Dhaker committed 4 days ago	dc11650
Modifications	Daksh Dhaker committed 4 days ago	2c0bc29
some modification		