Subreddit Trend

Subreddit Trend **1**
Table of Contents **1**
Abstract **2**
Introduction **3**
Problem Statement 3
Objective 3
End-User Needs 3
Literature Survey and References **3**
Requirement Specifications **4**
Market Analysis 4
Constraints 4
Assumptions 5
Risk 5
System Development **5**
System Design 5
Operational Flowchart 7
Design Flowchart 8
Functional Graph Decomposition 9
Testing **10**
Test 1 10
Test 2 11
Test 3 11
Conclusion **12**
Future Work 14
References **15**

Abstract

The aim of this project is to provide users with an overview of the top trends on a couple of the most popular social media platforms. Using well-known tools, React and Node.js, I will develop a website that fully incorporates the front-end and back-end from scratch. Additionally, if times permit, an inclusion of a chatbot will be implemented to provide users with additional information in case a query arises.

Introduction

Problem Statement

The internet has grown substantially since its inception in 1983, with large social media platforms integrated into our daily lives. Popular social media platforms such as TikTok, Instagram, Reddit, Meta, etc hold and create a vast amount of information every second. The fast-moving pace of information can be overwhelming, but if tools are used, information can be extracted to understand the popular trends that are happening. The goal of this project is to extract information from several of these websites and present it in a readable manner. With this, I plan to use React and Node.js. Some simple styles that would be present would be graphs and tables.

Objective

The specific objectives of this project are to visualize data in a table, include tabs to be able to sort the data, and include tabs to be able to filter the table. The completed project will include a full-stack web application that is fully functioning.

End-User Needs

The end-user needs for this project will include those who are interested in exploring Reddit and the different top subreddits that are available. According to Reddit, the number of monthly unique users crosses 1.1 billion. With the many users, a large amount of data and insights will be produced and gained. This project aims to provide the users with a table and visualizations that aggregate specific subreddit data for the users to easily infer from and gain insight from.

Literature Survey and References

There are solutions and references available for this sort of topic of scraping data off the internet. Projects from large companies such as Google and Steam have available solutions that can aggregate their specific data into simple and well-designed visualizations. Smaller-scale projects have also been developed to dive into Reddit data. For this project, these solutions helped the team understand the scope of the project and the presentation style for the data.

Google Trends provides the most searched queries from the Google search engine. There are a number of pages on this website that display the trends, but this project will focus on the tabular section. This page provides a table representing all the most popular searches throughout the entire search engine. Above the table, there are a number of dropdown menus that allow the

user to filter specific variables: Country, Time, Categories, Trends, and Sort By. The table is simple, providing the title of the trend and the search volume. Additionally, there is a search feature, where users can search for the trend of interest. Although the website does not scrape data off of Reddit, the visual elements and design features are significant.

Reddit Metric is a project created by benkulcsar. This website shows the metrics of a popular social media platform called "Reddit". Reddit contains subreddits, which are small communities that contain variables such as upvotes/downvotes, comments, posts, and more. The website Reddit Metric extracts the variables (upvotes, comments, and posts) and provides them in a table and on graphs.

Steam Charts is a website that aggregates useful information, such as Most Played, Top Selling Games, etc, into visualizations that are intuitive for the end users. The Most Played section provides an intuitive table that ranks the games by number of current players. Additionally, certain design aspects are integrated with the inclusion of small visualizations that indicate rank. Although the website does not scrape data off of Reddit, the visual elements and design features are significant.

Requirement Specifications

Market Analysis

As of Q3 of 2024, Reddit has an average of 365 million users. Compared to last quarter, this was a 23 million increase in users. Based on the reporting of Backlinko, it has shown that Reddit has been on a continual increase of monthly and weekly active users. This data shows that the number of users and untapped information residing inside each subreddit will continue to grow.

Constraints

For this project, there are a couple of constraints that will significantly impact the development of this project. These constraints include API request limit, development team knowledge, and deadline.

The development of this application requires the use of the Reddit API, which introduces the request limits. With the request limit, the development team will need to understand the scope of the project and the request limits.

The development of the application requires the use of libraries, programming languages, and frameworks that the team is unfamiliar with. With very little experience in programming

languages such as TypeScript and JavaScript, the team will need to research while implementing each feature.,

The last constraint is the deadline, which is April 20th, the date that the topic and results will need to be presented to the course audience. Additionally, the code will need to be finalized during that time. This constraint will impact the features that will be implemented in this project,t as the team will need to focus on deciding which features are important within the given timeframe.

Assumptions

For this project, the assumptions will be that the APIs and tools needed for this project will remain fully accessible to the team. Request limits for the Reddit api will be assumed to stay the same. Additionally, the services that are needed for deployment of the web application will remain popular deployment services, like Heroku have been discontinued before. Lastly, the team assumes that the cost of the development and services that are needed will remain free.

Risk

The risk for this project will be the inability to meet the April deadline and provide a functioning product for the user. As for the clients, there should not be any risk, considering the project will remain free.

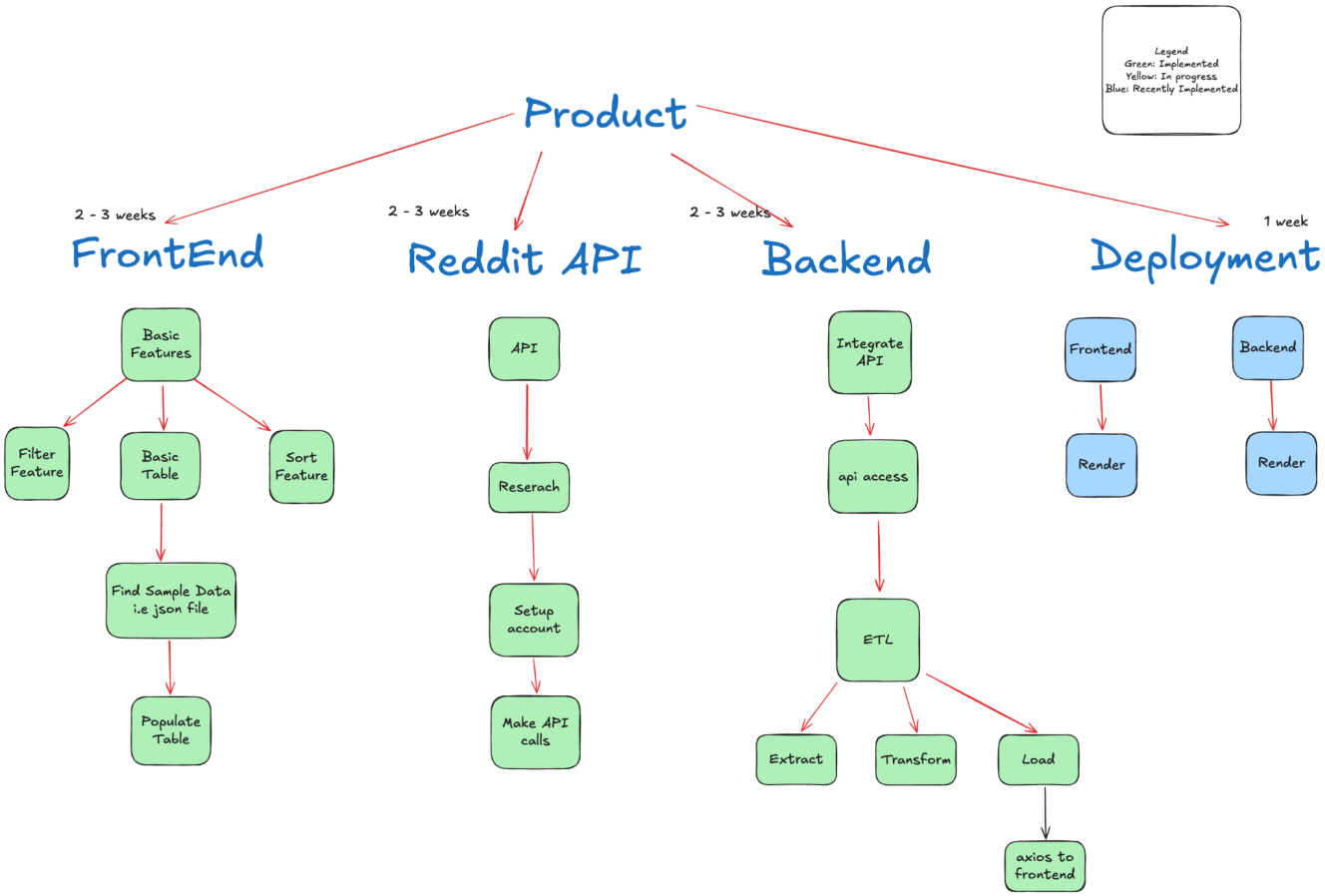
System Development

System Design

For the system design process, the team decided to use React for the frontend development, MongoDB for the database, and Express for the backend development. The libraries and frameworks for these projects were decided based on popularity and usefulness. React is a popular JavaScript library that is perfect for the frontend. Additionally, specific libraries could be used in React, such as Tanstack tables and Rechart. Both Tanstack and Rechart will be important to the development of the dashboard, as Tanstack includes the resources necessary to build a table and Rechart includes the resources necessary to build charts, graphs, and other visualizations. MongoDB was chosen for the database as the non-relational database works well with storing JSON-like documents. Express was chosen as it was a popular framework that could provide the tools needed for web scraping and api get requests.

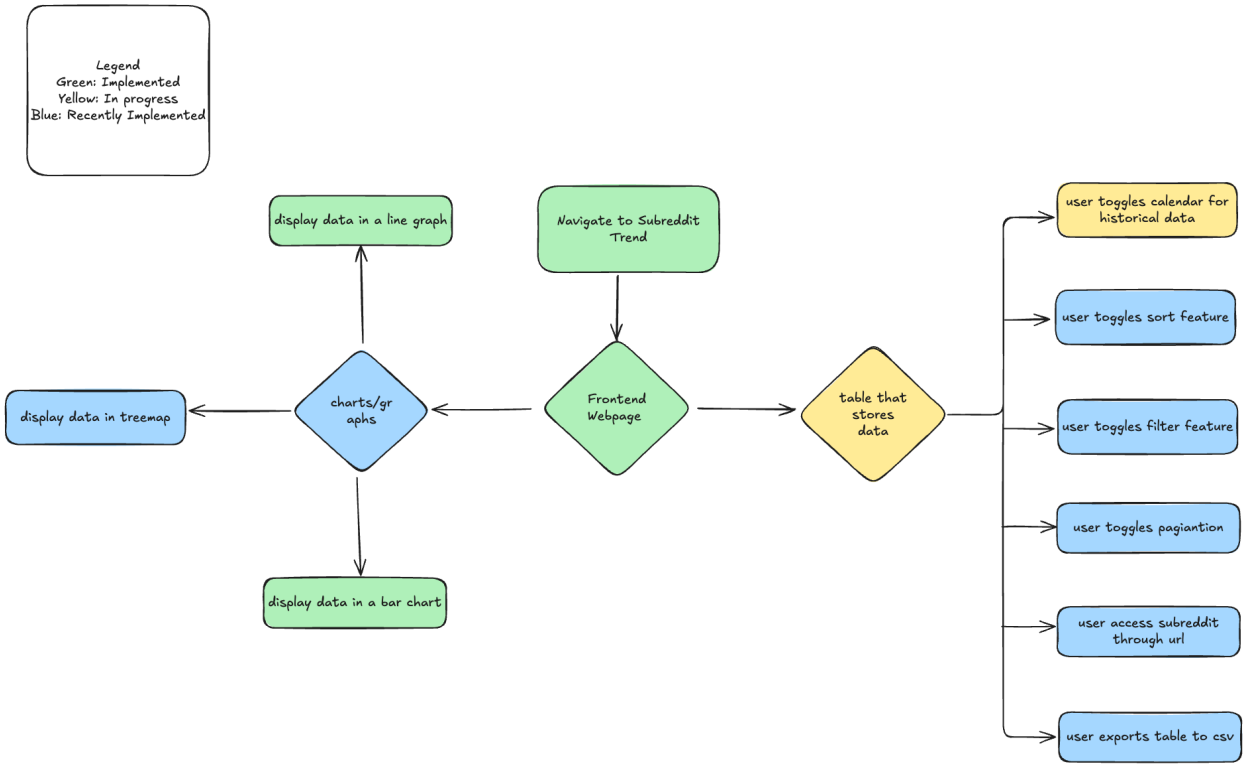
1	**Front End (Draft)**				
1.1	Setup React and Node Environment	100%	2/10/25	2/11/25	2/11/25
	Research fundamentals of Chartjs	10%	2/12/25	2/13/25	
1.2	Implement Basic Features	100%	2/14/25	2/18/25	2/13/25
1.2.1	Import sample data and populate the table	100%	2/14/25	2/18/25	2/13/25
1.2.2	Add Buttons	100%	2/14/25	2/18/25	2/13/25
1.3.1	Implement sort feature (pullout tab)	5%	2/14/25	2/18/25	
1.3.2	Implement filter feature (pullout tab)	5%	2/14/25	2/18/25	
2	**API**				
2.1	Research Reddit API	0%	2/19/25	2/20/25	
2.2	Pull data from Reddit	0%	2/21/25	3/26/25	
2.3	Integrate data onto table	0%	2/21/25	3/26/25	
2.4		0%			
3	**Back-End (Subject to Change)**				
3.1	Connect Reddit API	0%	2/27/25	3/3/25	
3.1.		0%			
4	**Front-End (Finalize)**				
4.1	Implement start menu (optional)	0%	3/4/25	4/13/25?	
4.2	Improve Visuals and Styles	0%	3/4/25	4/13/25?	
4.3	Testing	0%	3/4/25	4/13/25?	
4.4		0%			

The implementation plan is broken down into specific parts. The images above and below aim to break the project down into 4 components. The image above shows the specific task broken down into a Gantt chart on 2/18/2025. The Pert chart below visualizes the different components in a graph. The team will focus on the frontend first, implement the table and small



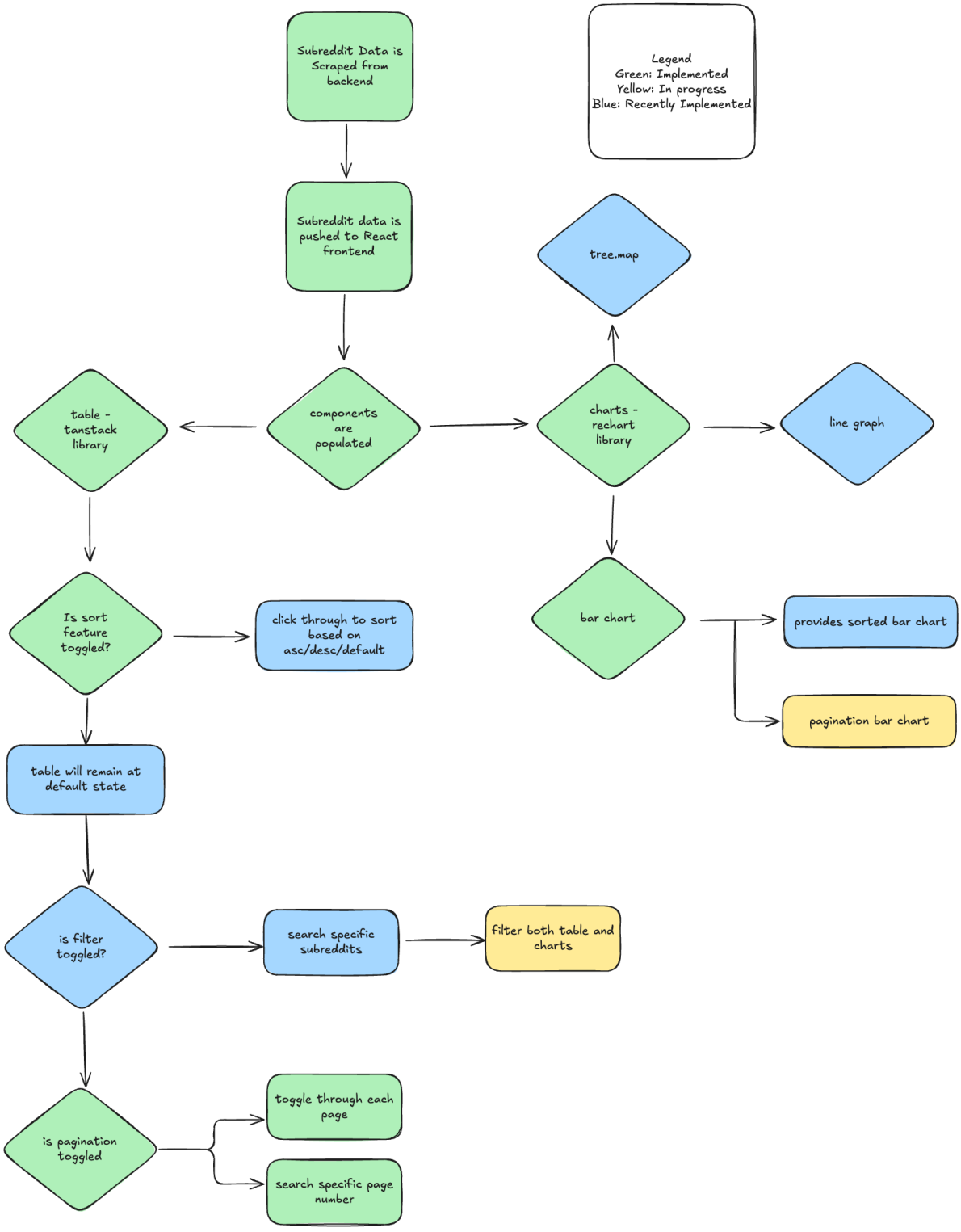
features such as filter and sort. Next, the team will research the Reddit API and learn how to make requests. Third, the team will set up the backend. The backend will implement the GET requests to grab data from the Reddit API and the FETCH requests to populate the table in the frontend. Additionally, the backend will include setting up the MongoDB database to connect the system. Last, the team will deploy both the backend and frontend using Render.

Operational Flowchart



The operational flowchart is designed to visualize how the end-user would interact with the program. In the image above, the frontend will display two major components: a table and charts. For the table, the features that the user can interact with are calendar, sort, filter, pagination, URL link, and export button. These are designed to be either URL links, dropdown menus, buttons, or search bars. The charts will mostly display information to the users with visualizations such as treemap, line graph, and bar chart.

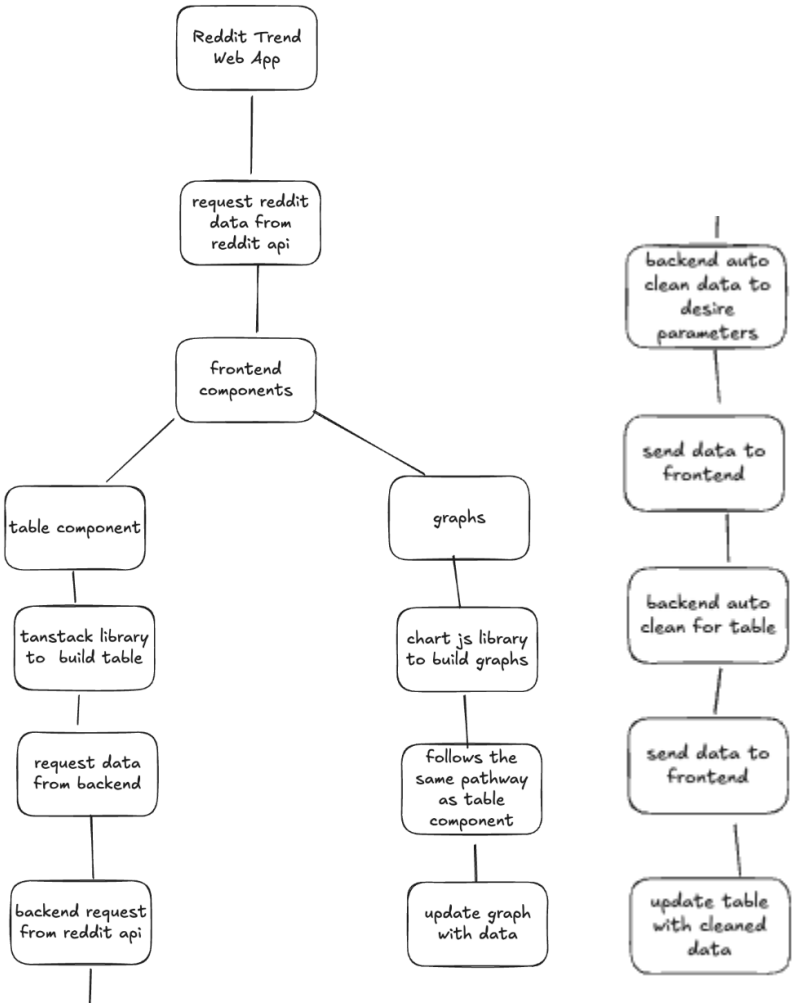
Design Flowchart



The design flowchart was developed to understand how each component of the frontend worked and how it would handle user inputs. From the backend, data is scraped to populate the

components. This includes the components from the Tanstack library and the Rechart library. From the Tanstack library, the table will be populated, and several features need to be understood. The sort feature will indicate the state of the sorted column. This includes either ascending, descending, or default. The filter feature will indicate whether the table is filtered for the specific values that are imputed. The pagination will indicate the several different pages that hold the rows. For the rechart library, the components included will be treemap, bar chart, and line graph. Since these are visualizations and require no user input, the components will indicate where they are populated or not.

Functional Graph Decomposition



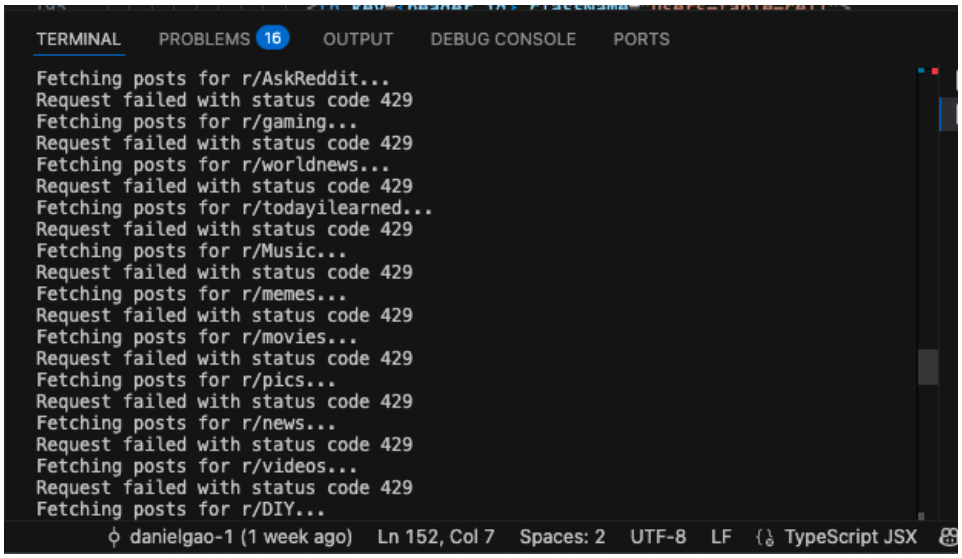
The functional decomposition is a graph that breaks down the application. Since the image was too long, the image was broken down into two parts, lines of boxes on the right match up with the boxes on the far left. The functional decomposition is displayed into two parts: the table component and the rechart component. For the table component, the steps are to fetch the data

taken from the backend, clean it, and populate the table with the data. The rechart component follows a similar format.

Testing

Test 1

The first test was to check if the user can crash the site by making too many refreshes, as the current implementation links the refresh to the API requests. While testing, the refresh was linked to 4-5 refreshes before the API limit was reached.



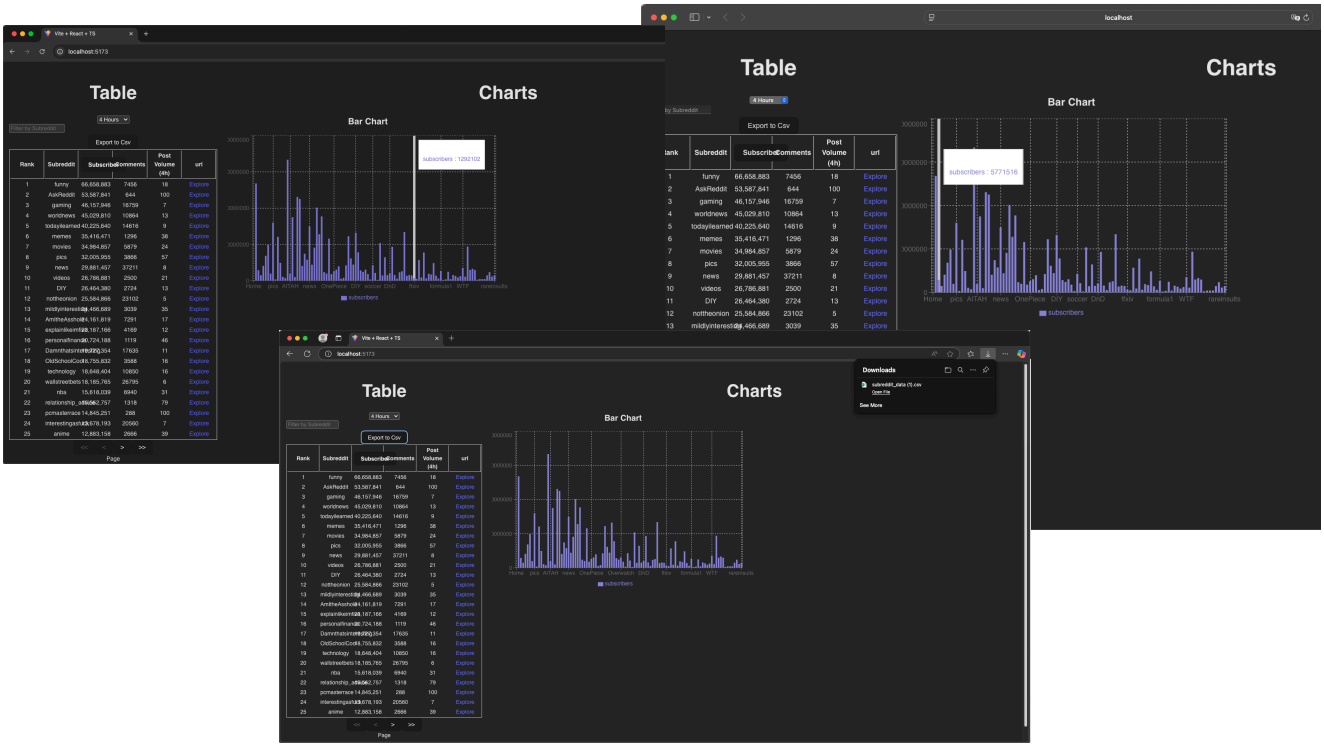
My solution to prevent the user from making the API request via refreshing was to make a fetch to a MongoDB server. This prevented the user from making API requests via refresh and prevented the user from reaching the API request limit. I did this by setting up a custom endpoint that calls the data from the MongoDB database. This way, I just replace the URL in the hook function.

```
app.get("/api/tofrontend", async (req, res) => {
  try{
    const subreddits = await SubredditDB.find().limit(100);
    res.json(subreddits);
  }
  catch (error) {
    console.error(error.message)
  }
});
```

Issue: One issue with this implementation is that MongoDB will not be updated. The solution was to implement an automation using the cron library. The cron library allowed me to run the two API calls at specific times. I chose to run the call at every hour.

Test 2

For the second test, I wanted to see if the application worked across different browsers. I tested the application on Chrome, Safari, and Windows Edge. From the testing, all the functions that were implemented worked without an issue. From this testing, the Export to CSV feature also worked without any issues.

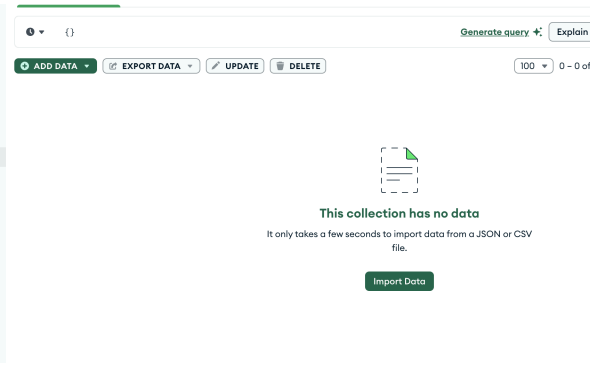


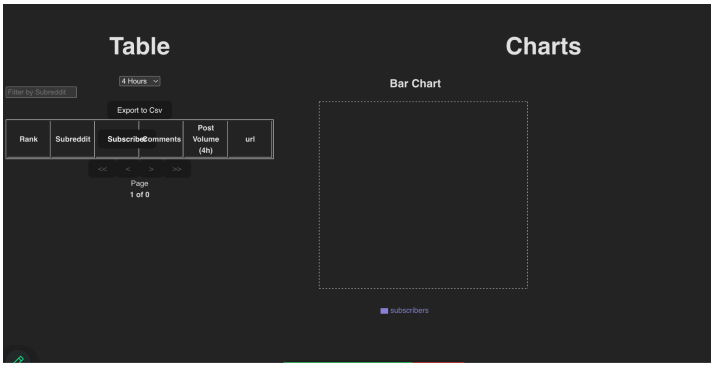
Test 3

For this test, I did a couple of edits in the MongoDB database to check if editing the values would introduce errors. From my initial creation of the schema, changing the types of the values did not introduce any issues. This was due to the way MongoDB handles types, but altering the database in different ways did create issues.

Issue 1: The issue was changing the URL to something else. From the image below, changing the URL will impact the URL link in the application, therefore, the link in the application will duplicate the page rather than go to the intended website.

Issue 2: The second issue was deleting the entire MongoDB database and checking if that impacted the application. Deleting the database resulted in the application displaying the skeleton of the table and chart.





Solution to both issues: As testing was done while the database was already populated by previous implementations, the testing has led me to know that certain endpoints were not automatically called by the application. To fix this issue, I had to recall the endpoint again. Since I have not set up an automatic system to call each endpoint, the endpoint had to be called manually. Automation will be set up in the next patch.

Conclusion

The goal of this project was to provide tools that allowed users to develop insights on Reddit data. As the course progressed, the project's goal remained similar, but certain aspects of the project were changed. Many features that I thought would be important to the project were not finished or had just entered the development stage. I attributed these setbacks to the single team and the lack of experience using programming languages such as CSS, JavaScript, and TypeScript. The lack of experience significantly reduced my progress toward the goals, as I needed to research and learn about every feature I implemented. For example, using the tanstack library to build a table was difficult, and I spent 1-2 months learning how to integrate different components into the table. Additionally, using the Reddit API without an API wrapper such as PRAW further increased the difficulty. Overall, this project has helped me gain experience in developing a full-stack web application. I was able to implement a filter search bar, a dropdown feature for posts (by hour), an export feature, a custom URL to subreddit, a sort feature, and different visualizations. Additionally, I was able to connect the backend, MongoDB, and frontend to provide a robust application. All the tasks that were finished can be seen in the images below.

1	**Front End (Draft)**				
1.1	Setup React and Node Environment	100%	2/10/25	2/11/25	2/11/25
	Research fundamentals of Chartjs	100%	2/12/25	2/13/25	Moved
1.2	Implement Basic Features	100%	2/14/25	2/18/25	2/13/25
1.2.1	Import sample data and populate the table	100%	2/14/25	2/18/25	2/13/25
1.2.2	Pagination	100%	2/14/25	2/18/25	2/13/25
1.3.1	Implement sort feature (pullout tab)	100%	2/14/25	2/18/25	Moved
1.3.2	Implement filter feature (pullout tab)	100%	2/14/25	2/18/25	Moved
2	**Reddit API**				
2.1	Research Reddit API	100%	3/3/25	3/10/25	3/7/25
2.1	Research Postman	100%	3/3/25	3/10/25	3/7/25
2.1	Setup Postman	100%	3/3/25	3/10/25	3/7/25
2.1	Set up axois	100%	3/3/25	3/10/25	3/7/25
2.2	ETL Process (Extract, Transform, Load)	100%	3/3/25	3/10/25	4/20/25
2.2.1	Extract	100%	3/3/25	3/10/25	4/20/25
	extract comments	100%	3/13/25	3/13/25	4/4/25
	extract upvotes/downvotes	100%	3/13/25	3/13/25	4/4/25
3	**Back-End (Subject to Change**				
3.1	Research Express	100%	3/3/25	3/10/25	3/7/25
	Set up Express	100%	3/3/25	3/10/25	3/7/25
	Set up Fetch Request	100%	3/3/25	3/10/25	3/7/25
3.2	Connect Reddit API	100%	3/3/25	3/10/25	3/7/25
3.3	Set up Mongodb	100%			3/21/25

4	**Front-End/Deployment**				
4.0.0	Implement start menu (optional)	0%	3/4/25	4/13/25	
4.0.0	Improve Visuals and Styles	100%	3/4/25	4/13/25	4/20/25
4.0.0	Testing	100%	3/4/25	4/13/25	4/20/25
4.0.0	Implement sort feature (button)	100%	2/14/25	2/18/25	3/12/25
4.0.0	Implement filter feature (pullout tab)	100%	2/14/25	2/18/25	3/30/25
4.0.1	pagination fix (show 25 rows)	100%	3/13/25	3/13/25	3/12/25
4.0.1	add row indicators (ranking)	100%	3/13/25	3/13/25	3/30/25
4.0.1	add default sort (asc)	100%			3/25/2025
	export feature	100%			3/26/25
	post by hour	100%			4/6/25
	add filter calendar- filter by hour	0%	3/31/25	4/6/25	Moved
	search by recency	100%			4/19/25
	extract more subreddits	100%			3/30/25
	make charts with pagination				Moved
	figure relevant columns	100%			3/30/25
	sort by alphabetic	100%			4/20/25
	make row a button to provide more information				Moved
	fix sort button to desc/asc	100%			4/20/25
	overwrite rather than duplicate database	100%			4/2/25
	publish website	100%			4/20/23
	frontend formatting	100%			4/20/23
	add more charts	100%			4/20/25
	make url as links	100%			3/25/25
	sentiment analysis model	0%			Moved
	i.e. provide a serach bar, user suggest a subreddit, provide a simple table with # of subscribers, post, comments,	100%	3/31/25	4/4/25	Moved
	fix sort button to desc/asc	100%			
	overwrite rather than duplicate database	100%			4/2/25
	search feature to search subreddits not in the 200 list	0%			

If I were to restart this entire course, I would have chosen Python as the programming language rather than JavaScript/TypeScript. Since I was more familiar with this language, I believed I would have been able to utilize machine learning models in this project. Additionally, rather than building a table/charts from scratch, I would have used data visualization tools such as Power BI or Tableau. I believe using existing software would have provided an easier introduction to implementing the features I wanted.

Future Work

For future work, there are tons of features that could be implemented to make the application a better experience. Towards the end of the project, I realized that a sentiment analysis would be perfect for this type of project, as I lacked the time I needed to polish other features. Additional features that could be implemented are listed below.

1. Calendar filter to access historical data
2. API request to gather more information
3. Adding different visualizations
4. Adding search features that provide information on subreddits based on user input was missing.

References

Backlinko Team. "Reddit User and Growth Stats (Updated January 2025)." *Backlinko*, Jan. 2025, https://backlinko.com/reddit-users. Accessed 20 Apr. 2025.

Google. *Google Trends*. https://trends.google.com/trending?geo=US. Accessed 20 Apr. 2025.

Google. *Reddit Metrics*. Looker Studio, https://lookerstudio.google.com/u/0/reporting/865759fa-0b1a-4bee-8b67-89cb2ed0d2f0/page/1XG8C. Accessed 20 Apr. 2025.