Project Final Report

Web-based Application Development: Case Study on President Election Sentiment Analysis Using Firebase Realtime Database for Reddit Posts and Comments

Team 67

Armand Patel, Joshua Payapulli, Paul Yoo

Github: https://github.com/PKYOO-116/DSCI551_Sp2024_T67.git

DSCI551 Foundation of Data Management
Prof. Wenseung Wu

Web-based Application Development: Case Study on President Election Sentiment Analysis Using Firebase Realtime Database for Reddit Posts and Comments

1. Introduction

(1) The Project Team

- Man Power and Project Participation

Armand Patel (Leader) -

Skilled in Data Analysis and Probability and Statistics techniques. Domain expertise in healthcare and technology. I have experience with Python, HTML/CSS, R studio, and general project communication. Participated in front-end design, writing API/backend logic for CRUD operations, report writing.

Joshua Payapulli -

Academic background in computer science. Comfortable with Python, Java, C, SQL, JavaScript. Professional experience in regulatory compliance and finance as a Business Analyst and Consultant. Also, I have close to a year's experience as a Full-Stack Software Engineer. Participated in integrating Flask backend with React frontend, writing API/backend logic for CRUD operations, report writing.

Paul Yoo -

Project management experience in manufacturing industry digitization projects with domain knowledge in Business, Finance, and Strategic decision-making. Skills in project management, business insights, python programming, MS Office, and IT project communication. Participated in project management(scheduling and timelining), sentiment analysis, database management, data collection, and report writing for the project.

(2) Project Objective

- Web-based Application for Presidential Election Sentiment Analysis

The project's primary objective is to create a web-based application using databases and conduct a comprehensive sentiment analysis on Reddit posts to predict the outcome of the upcoming U.S. presidential election. Our team has chosen to implement this project using the

Firebase Realtime Database for efficient data storage and management. This database will store data collected from Reddit, specifically from the subreddits, where less biased posts are created and which are relevant to the political discourse surrounding the presidential election.

Initially, the team planned to use Twitter data, but due to recent changes in Twitter's policy that led to the commercialization of its developer tools and limited free access, the team opted for Reddit as an alternative data source. Reddit not only allows for direct data scraping but also provides an API for pulling information, which the team is utilizing to gather data pertinent to their analysis.

The project involves several technical strategies for managing large volumes of data. These include the use of partitioning, replication, and horizontal scaling to enhance data storage and retrieval efficiency. The team has developed a specialized Python script to automate the data collection process from Reddit, ensuring that the data is continuously updated and stored across multiple Firebase databases and facilitating efficient data processing.

As part of their methodology, the team will apply various data preprocessing techniques to clean and prepare the data for analysis. Following this, sophisticated sentiment analysis algorithms are used to analyze the sentiments expressed in the Reddit posts, with the goal of identifying patterns and trends that could indicate public opinion trends related to the presidential candidates.

This sentiment analysis aims not only to predict the election outcome but also to demonstrate the practical application of data science techniques in real-world scenarios, such as political forecasting. Through this project, the team hopes to gain insights into public sentiment and its potential impact on electoral results, providing a unique case study on the intersection of social media, data analytics, and political science.

(3) Database:

- Google Firebase

Data from the most popular posts on Reddit is archived across four distinct Firebase real-time databases, employing a hashing mechanism based on the commenter's ID to distribute the information efficiently.

The use of Google Firebase as the database platform brings several advantages to the project: it offers complimentary data storage, can accommodate vast quantities of data, accelerates application development, provides secure storage with a NoSQL foundation, facilitates easy data sharing among project members, and ensures quick responses to queries. These features collectively enhance the project's operational effectiveness and streamline the data management process, making Firebase an ideal choice for handling the complexities of real-time data analysis in a collaborative environment.

(4) Data Distributed Scaling Approach:
- Partitioning, Replication, and Horizontal Scaling

Databases play a crucial role in the optimal operation of any system that depends on the retrieval and storage of information. When a database faces excessive demand or is constrained by limited storage capacity, the performance of the system can degrade, leading to slow response times. It is therefore critical to enhance database scalability to accommodate the increasing requirements for data storage and system performance. Various strategies are employed to scale databases effectively.

Key methods for scaling and distributing database resources include 1) choosing between horizontal and vertical scaling, 2) opting for replication or sharding techniques, 3) deciding between SQL or NoSQL database systems, and 4) employing various tools and techniques such as load balancers, caching systems, and monitoring solutions.[1][2][3]

In addressing the need for data distribution and scalability, the team implemented several strategies to boost the project's effectiveness. First, archived data storage balance by partitioning data across four separate real-time databases, utilizing a hash function for equitable data allocation. The hash function takes the ascii sum of the content of the reddit post, and calculates the modulo of this by the number of databases we have. This allows us to evenly distribute our data and help with load balancing. Thus, minimizing computation times and increasing availability for future analytic processes. These measures collectively ensure that the system remains robust and responsive, even as data demands grow.

2. Planned Implementation

(1) Plans and Changes from the initial proposal

There have been several adjustments to our initial project proposal. We originally planned to gather tweets using Twitter's API(Currently, X), but changes to their policy that commercialized developer tools and ended free access to data collection from other users prompted us to seek alternatives. The new policy imposed a fee and a cap on data retrieval (limited to 1500 tweets per month), which led us to choose Reddit as our new data source. Reddit permits direct scraping of data from its site and also provides an API for extracting information.

We developed a Python script, named extract_data.py, which facilitates the extraction of Reddit content from specified subreddits, including the number of comments and the specific keywords (separated by commas) relevant to our project's theme. The script utilizes the Reddit

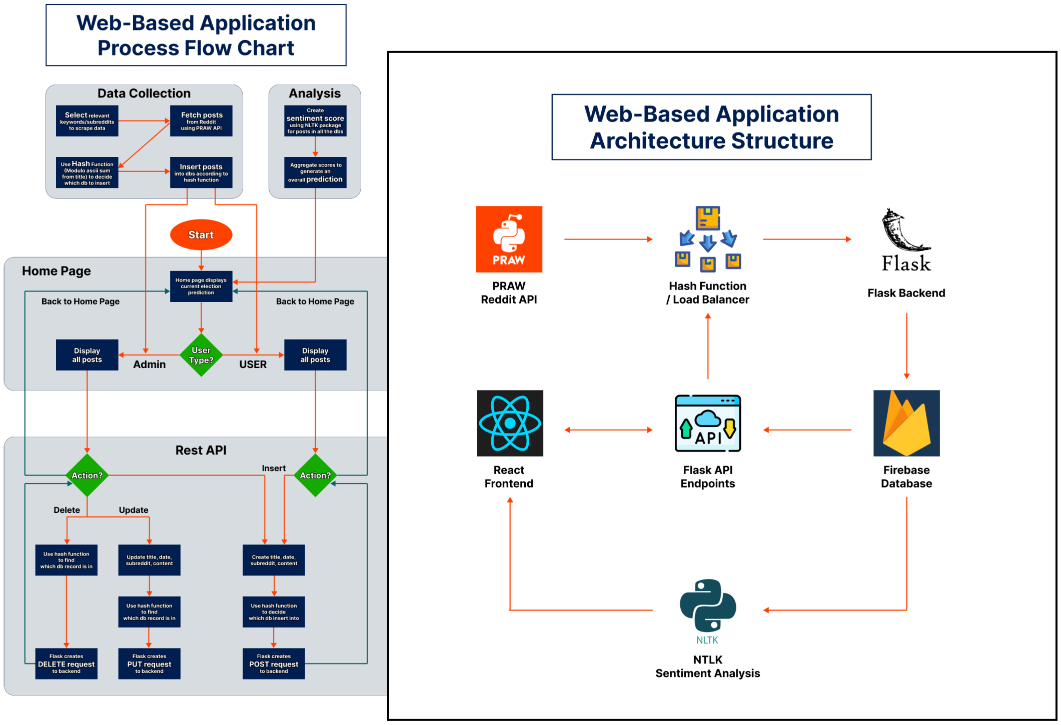
API, known as PRAW, which simplifies the process of fetching posts through keyword searches. Our Firebase Database setup supports the necessary scalability for our project's needs.

As of now, our team has successfully collected over 7000 comments of data from Reddit, encompassing elements such as post titles, comment bodies, URLs, commenter IDs, post votes (upvotes and downvotes), the time of the comment, and sentiment scores. The sentiment scores are derived using the Natural Language Toolkit (NLTK) package in Python, which allows for the scoring of each word based on positive or negative sentiments and computes an overall sentiment score for each comment using the sentiment analyzer module.[4] This tool enables us to discern the emotional tone of comments, which is crucial for understanding how these sentiments could influence future voting behavior in the election.

3. Architecture Design

(1) Flow Diagram and Architecture Diagram

(Go to 7. Reference [5], [6] for higher resolution image)


Web-Based Application Process Flow Chart
Data Collection
Select relevant keywords/subreddits to scrape data
Fetch posts from Reddit using PRAW API
Use Hash Function (Modulo ascii sum from title) to decide which db to insert
Insert posts into dbs according to hash function
Analysis
Create sentiment score using NLTK package for posts in all the dbs
Aggregate scores to generate an overall prediction
Start
Home Page
Home page displays current election prediction
Back to Home Page
Back to Home Page
Display all posts
Admin
User Type?
USER
Display all posts
Rest API
Action?
Insert
Action?
Delete
Update
Use hash function to find which db record is in
Update title, date, subreddit, content
Create title, date, subreddit, content
Use hash function to find which db record is in
Use hash function to decide which db insert into
Flask creates DELETE request to backend
Flask creates PUT request to backend
Flask creates POST request to backend
Web-Based Application Architecture Structure
PRAW
PRAW Reddit API
Hash Function / Load Balancer
Flask
Flask Backend
React Frontend
API
Flask API Endpoints
Firebase Database
NLTK
NLTK Sentiment Analysis

(2) Description

(a) User visits the homepage of the web application, the webpage sends the user's request to the backend and shows the results of user's query, or all data that is currently available in the database

(b) Flask server retrieves data from firebase realtime database
(c) Flask server can update (insert or delete or edit) data into the database in realtime
(d) React displays the data visually
(e) Sentiment analysis

4. Implementation

(1) Functionalities

The application we've created offers a range of advanced functionalities that make sense given the large volume of data we collected initially. The user experience of the application is designed to supplement the data management functionalities and allow for efficiency in the use of the app. One key functionality is the real-time retreiavl and display of information front he Reddit website along with key metadata. By leveraging the Realtime nature of Firebase, our application can continuously fetch posts and comments from Reddit using our extract_data.py script. The app can also continuously update the database with new user posts and edits. The user can also use the search functionality to filter and retrieve specific posts based on keywords, subreddit names, or other criteria specified by the application. The application also provides robust data management capabilities, allowing users to insert new Reddit posts, update existing posts, and delete unwanted posts. These functionalities are complemented by intuitive user interfaces that streamline the user's interaction and enhance usability.

The application also employs sentiment analysis algoirithms to analyze the emotional tone of Reddit comments, and these provide insights into public sentiments surrounding presidential candidates in the upcoming election.

(2) Tech Stack

A Firebase Realtime Database, which is a NoSQL database platform provided by Google, was at the core of our infrastructure. Complementing Firebase, our backend was powered by Flask, a lightweight and extensible web framework for Python. We used Python to write all of our interface and also data extraction scripts. We also utilized existing Python libraries to extract data. Flask facilitated rapid development of RESTful APIs and web services, which enables smooth communication between the frontend and backend components of our application. On the frontend of our application, we used React.js, which was used to build the interface of our application. This allowed the creation of our interactive and dynamic UI components.

(3) Database Integration

Our web application was integrated with Google Firebase Realtime Database to store the data we were pulling. Firebase provides a robust and scalable data storage solution. It served as the backbone of our system, and allowed for the storage of large volumes of data that can easily be edited and updated. We leveraged Firebase's NoSQL database structure to store a bounty of information about each Reddit post. This included metadata from each entry like, User ID, date and time posted, The name of the associated Subreddit, the title of the post, comments associated with the post, ior the comments itself, the amount of upvotes and downvotes received, and more. This metadata or information was all stored in a JSON format in the Firebase. This allowed for real-time retrieval and synchronization. Using Firebase's API, our backend infrastructure with Flask communicates seamlessly with the database to allow for insertion, retrieval, modifications to any data in the database. The firebase database is connected to the web application in real time to allow this user interaction and to allow the user to view what data is stored in the database in a tabular format instead of a JSON format. The databases are instantly propagated to all connected clients, which maintains the consistency of data across the application.

The data was retrieved through a python script that utilizes python's built in library "PRAW," which scrapes real time data from the Reddit website itself. This way we could extract a large volume of data from the Reddit website and store this information in the database. Not only can the user query the data and insert new data into the database, but they can also see the data we already have stored in the database. This part contributes to our overall sentiment analysis portion of the project. A large volume of data was necessary to perform the machine learning sentiment analysis that was shown on the home page.

(4) Data Management Functionality

Our web application incorporates several data management functionalities that are designed to seamlessly integrate our databases into the frontend of the website and allow the users to interact with the data in real time. We used Flask, which handles the HTTP requests sent by a user, which occur when a user clicks on a specific button in the app.

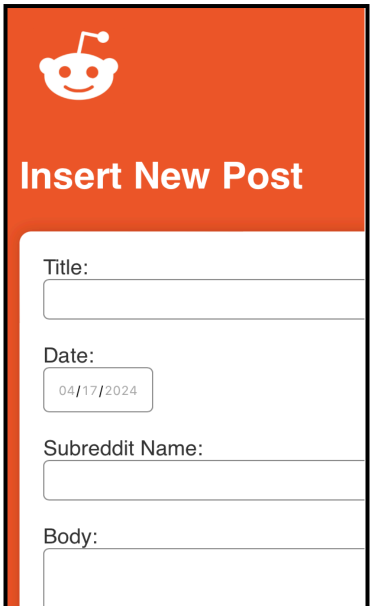
When a user selects the "Insert Post" button, a Post request is sent to the corresponding Flask route, which triggers a function responsible for inserting a new post into the Firebase Realtime Database. This function parses the data from the user's input fields, and constructs a JSON object that represents the post, and utilizes Firebase's API to store the data securely in the database.

When a user wants to update a post, they can select the "Edit" button that lives to the right of each post in the table that is shown in the app itself. The Flask route handles the request and invokes the backend function to retrieve the post that was selected front he database and displays it for editing using the user interface. This interface facilitates the submission of the modified data back to Firebase.

The deletion functionality is implemented in a similar way. The Flask route invokes the backend functionality to remove the post that the user selects upon the user's request. The "Delete" button lices right under each "Edit" button, allowing a seamless interface for the user to directly view the data they want to edit or delete.

Through Flask's integration with our Firebase, our application provides a responsive backend infrastructure that allows users to have many options when interacting with our data in real time.

(5) Implementation Screenshots



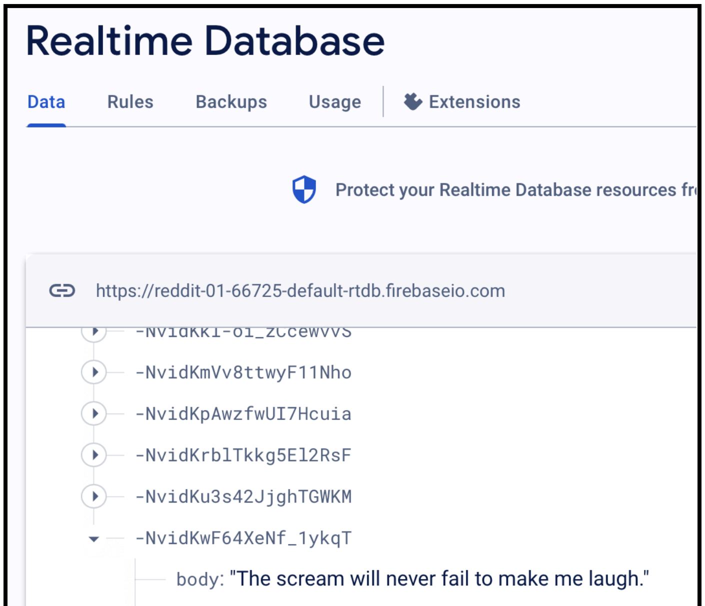
Insert New Post

Title:

Date: 04/17/2024

Subreddit Name:

Body:

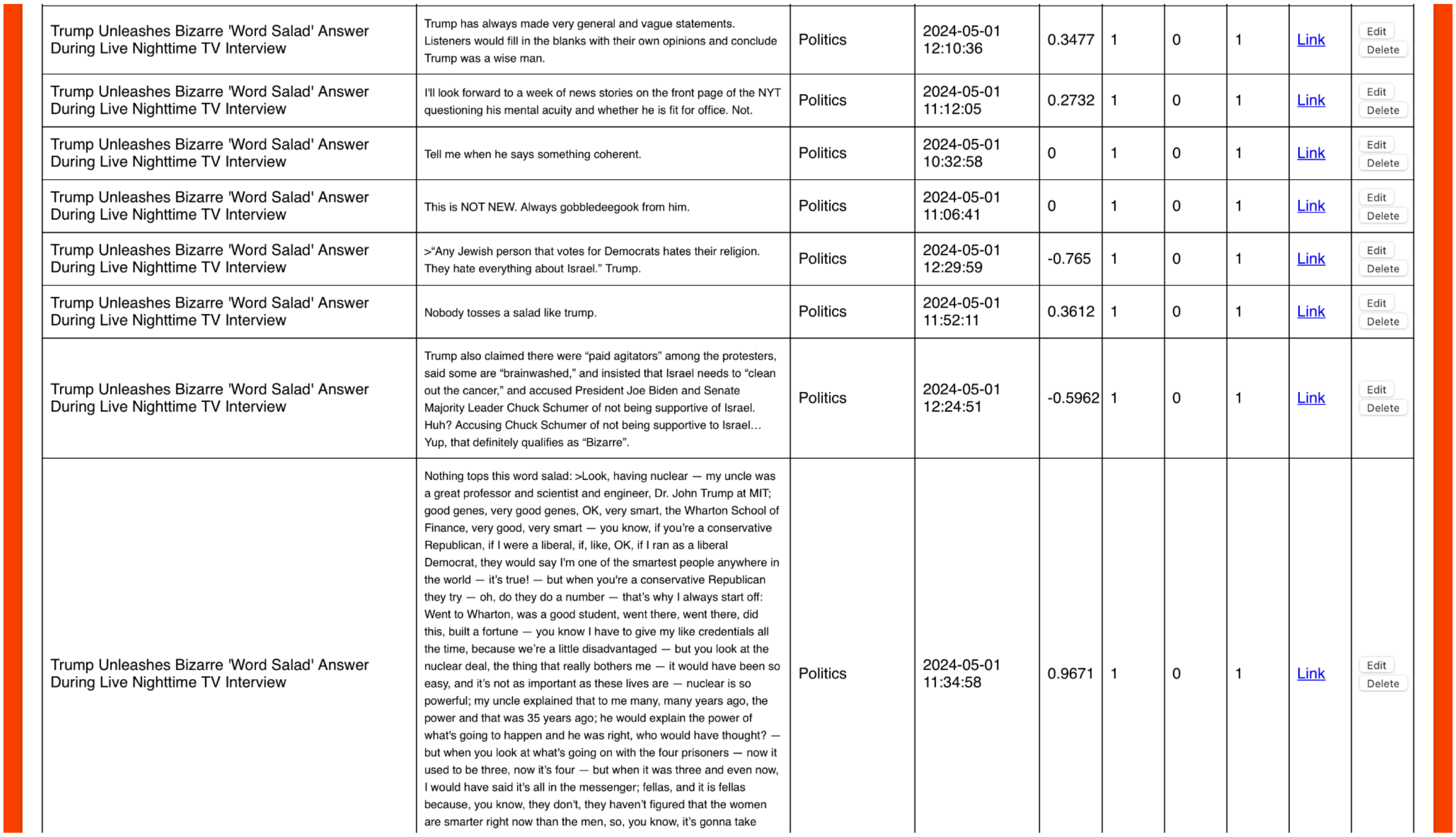


Realtime Database

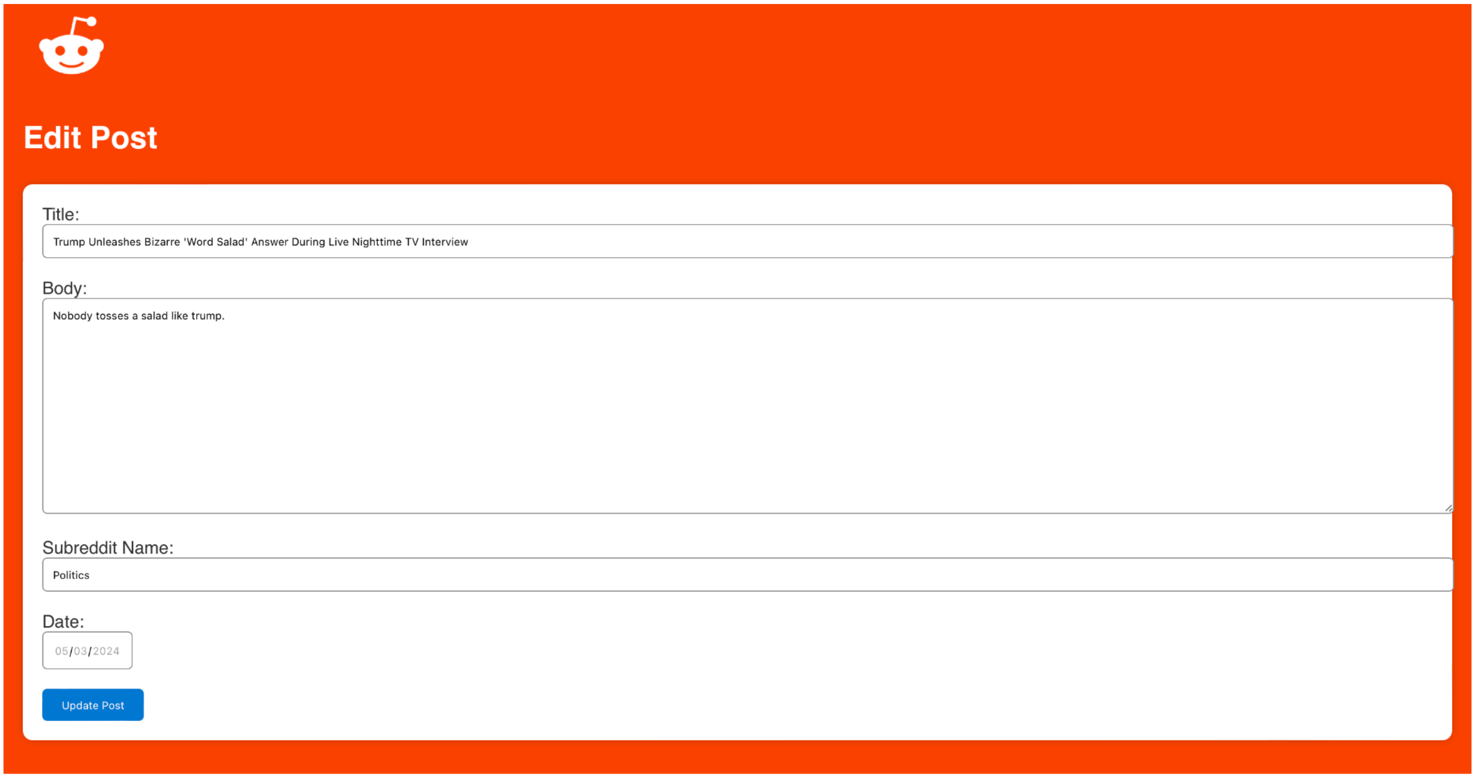
Protect your Realtime Database resources fr

https://reddit-01-66725-default-rtdb.firebaseio.com

-NvidKmVv8ttwyF11Nho
-NvidKpAwzfwUI7Hcuia
-NvidKrblTkkg5El2RsF
-NvidKu3s42JjghTGWKM
-NvidKwF64XeNf_1ykqT
body: "The scream will never fail to make me laugh."



Trump Unleashes Bizarre 'Word Salad' Answer During Live Nighttime TV Interview	Trump has always made very general and vague statements. Listeners would fill in the blanks with their own opinions and conclude Trump was a wise man.	Politics	2024-05-01 12:10:36	0.3477	1	0	1	Link	Edit Delete
Trump Unleashes Bizarre 'Word Salad' Answer During Live Nighttime TV Interview	I'll look forward to a week of news stories on the front page of the NYT questioning his mental acuity and whether he is fit for office. Not.	Politics	2024-05-01 11:12:05	0.2732	1	0	1	Link	Edit Delete
Trump Unleashes Bizarre 'Word Salad' Answer During Live Nighttime TV Interview	Tell me when he says something coherent.	Politics	2024-05-01 10:32:58	0	1	0	1	Link	Edit Delete
Trump Unleashes Bizarre 'Word Salad' Answer During Live Nighttime TV Interview	This is NOT NEW. Always gobbledeegook from him.	Politics	2024-05-01 11:06:41	0	1	0	1	Link	Edit Delete
Trump Unleashes Bizarre 'Word Salad' Answer During Live Nighttime TV Interview	>"Any Jewish person that votes for Democrats hates their religion. They hate everything about Israel." Trump.	Politics	2024-05-01 12:29:59	-0.765	1	0	1	Link	Edit Delete
Trump Unleashes Bizarre 'Word Salad' Answer During Live Nighttime TV Interview	Nobody tosses a salad like trump.	Politics	2024-05-01 11:52:11	0.3612	1	0	1	Link	Edit Delete
Trump Unleashes Bizarre 'Word Salad' Answer During Live Nighttime TV Interview	Trump also claimed there were "paid agitators" among the protesters, said some are "brainwashed," and insisted that Israel needs to "clean out the cancer," and accused President Joe Biden and Senate Majority Leader Chuck Schumer of not being supportive of Israel. Huh? Accusing Chuck Schumer of not being supportive to Israel… Yup, that definitely qualifies as "Bizarre".	Politics	2024-05-01 12:24:51	-0.5962	1	0	1	Link	Edit Delete
Trump Unleashes Bizarre 'Word Salad' Answer During Live Nighttime TV Interview	Nothing tops this word salad: >Look, having nuclear — my uncle was a great professor and scientist and engineer, Dr. John Trump at MIT; good genes, very good genes, OK, very smart, the Wharton School of Finance, very good, very smart — you know, if you're a conservative Republican, if I were a liberal, if, like, OK, if I ran as a liberal Democrat, they would say I'm one of the smartest people anywhere in the world — it's true! — but when you're a conservative Republican they try — oh, do they do a number — that's why I always start off: Went to Wharton, was a good student, went there, went there, did this, built a fortune — you know I have to give my like credentials all the time, because we're a little disadvantaged — but you look at the nuclear deal, the thing that really bothers me — it would have been so easy, and it's not as important as these lives are — nuclear is so powerful; my uncle explained that to me many, many years ago, the power and that was 35 years ago; he would explain the power of what's going to happen and he was right, who would have thought? — but when you look at what's going on with the four prisoners — now it used to be three, now it's four — but when it was three and even now, I would have said it's all in the messenger; fellas, and it is fellas because, you know, they don't, they haven't figured that the women are smarter right now than the men, so, you know, it's gonna take	Politics	2024-05-01 11:34:58	0.9671	1	0	1	Link	Edit Delete



Edit Post

Title: Trump Unleashes Bizarre 'Word Salad' Answer During Live Nighttime TV Interview

Body: Nobody tosses a salad like trump.

Subreddit Name: Politics

Date: 05/03/2024

5. Sentiment Analysis and Prediction

(1) Sentiment Analysis

- Keywords, NLTK, and Prediction based on Sentiment Score

We selected four specific keywords—Trump, Biden, Republican, and Democrat—to gauge the level of support for each candidate and party by assessing the positivity or negativity associated with these terms. For our sentiment analysis, we employed the Natural Language Toolkit (NLTK) in Python, which assigns a sentiment score ranging from -1 to 1 to each word within a comment. A score closer to 1 indicates positivity, while a score nearing -1 suggests negativity. A score close to 0 is indicative of neutrality. The Sentiment Analyzer module allowed us to compute an overall sentiment score for each comment by averaging the scores of its words using a predefined formula.[4]

Once the scores were calculated, they were stored in the database alongside the corresponding comment data. To parse the scores, we retrieved the comment data and identified keywords within the post titles that matched the scores. Because posts often contain multiple keywords, we iterated through each one separately, listing the scores associated with each keyword.

The script, score_parse.py, has the `calculate_average` function that computes the average positive and negative sentiment scores for each keyword. It differentiates between positive and negative scores, calculates the mean for each, and stores these values. If no scores exist for a specific sentiment (positive or negative), the average is set to zero.

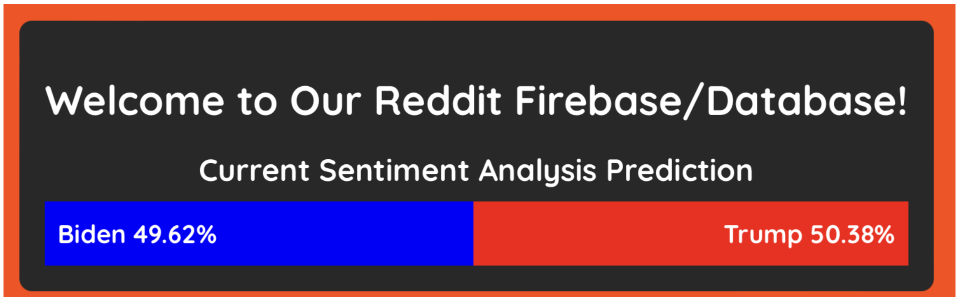
The `calculate_percentage_difference` function processes the average sentiment scores further. It combines the average positive score of one keyword with the absolute value of the average negative score of its counterpart. This step reflects a holistic sentiment measure, assuming that positive support for one side and negative sentiment towards the other can be aggregated to represent overall public support or opposition.

It then calculates the total sentiment scores for both the candidates and parties. The support ratio for each candidate and party is computed by dividing their individual scores by the total scores, effectively normalizing the data to reflect percentages. These percentages indicate the proportion of positive sentiment each candidate and party holds relative to the total positive sentiment observed.

The script calculates an overall support score for each candidate by averaging their support ratio with their party's ratio. This step integrates personal and party-based sentiments into a single metric per candidate. The script then compares these final scores to determine which candidate has higher support.

The winner is predicted based on these support scores, and the margin of victory is calculated by determining the difference between the scores of the two candidates. The script outputs both the predicted winner and the victory margin, offering a quantified prediction of the election outcome based on sentiment analysis of public comments on Reddit.

This methodology showcases how sentiment analysis can be leveraged to gauge public opinion and predict election results, turning qualitative data into quantitative insights. Up until today, the collected data indicates that Trump is leading by a margin of 0.38%.



Welcome to Our Reddit Firebase/Database!

Current Sentiment Analysis Prediction

Biden 49.62% Trump 50.38%

6. Learning Outcomes

(1) Challenges

- Commercialized Tools, Analysis, Lack of experience in Firebase

As students engaged in this project, one of our foremost challenges is sourcing free tools and databases suitable for our research needs. Although all team members are prepared to cover project costs, we have embraced this constraint as an opportunity to enhance our problem-solving skills in a demanding setting.

During the data preprocessing phase, we realized the need for additional attributes or fields that could improve our predictive analysis. Until the progress report, we planned to use the Reddit posts to process sentiment analysis for election prediction, however, we found some of the Reddit posts included no personal opinion but only the URL redirecting to the news articles. Therefore, we changed to use comments even though there are more phrases we need to parse through.

Another significant challenge has been managing the Firebase real-time system. Google offers comprehensive dashboards and monitoring systems for their databases, but these features are accessible only to paid subscribers. Consequently, we are exploring alternative methods to monitor and manage our databases through programming, while also learning to navigate the guidelines provided by Google.

This exploration includes developing custom scripts to automate database health checks and experimenting with open-source tools that might offer monitoring functionalities. Our goal is to maintain an effective workflow and ensure data integrity without incurring additional costs, thereby keeping our project both affordable and technically robust.

(2) Future work

In expanding our sentiment analysis, we plan to integrate data from Twitter alongside our current Reddit dataset. This inclusion will not only enhance the breadth of our data but also improve the accuracy of our predictions by capturing a wider spectrum of public opinions. Given Twitter's significant role in public discourse, especially in political contexts, incorporating its data could provide valuable insights into real-time sentiment trends. We will develop methods to seamlessly combine sentiment data from both platforms, ensuring a robust analysis that leverages the strengths of each social media environment. Our approach will involve adapting our existing scraping tools to comply with Twitter's API requirements, while maintaining efficiency in data processing and analysis.

To keep our sentiment analysis current and dynamic, we will implement scheduled updates for our prediction models. By automating the sentiment analysis script to run every three hours, the system will fetch the latest posts and comments, analyze them, and update the predictions displayed on our front end. This real-time data processing capability will allow our web application to reflect up-to-date public sentiment, enhancing the reliability of our election outcome forecasts. We will utilize cron jobs or similar scheduling tools to manage these updates, ensuring that our application remains responsive and accurate, providing users with the most current insights into electoral sentiments.

(3) Conclusion

This project exemplifies a successful integration of advanced technology and innovative methodologies to address the dynamic challenges of political sentiment analysis. Through the dedicated efforts of the project team, this application not only leverages the expansive data available on Reddit but also demonstrates the practical application of sentiment analysis techniques using the Natural Language Toolkit (NLTK).

The project effectively navigated significant shifts in data sourcing from Twitter to Reddit due to changes in API access policies, showcasing the team’s adaptability and problem-solving capabilities. By employing the Firebase Realtime Database, the application benefits from a scalable, responsive, and highly accessible database structure that ensures real-time data processing and analysis, crucial for maintaining the relevance and accuracy of the sentiment analysis.

The successful collection and analysis of about 1900 comments have provided valuable insights into public sentiment regarding the upcoming U.S. presidential election. This extensive data collection, coupled with robust sentiment analysis algorithms, has allowed the team to predict electoral outcomes with a noteworthy degree of confidence, indicating a potential lead for Trump with a margin of roughly 0.4%.

This project has not only highlighted the capabilities of Flask and React in creating responsive web applications but also underscored the importance of data analysis in understanding and predicting public opinion trends. Moving forward, the integration of additional data sources and the continuous updating of the prediction models promise to enhance the accuracy and reliability of the predictions, offering a more comprehensive view of the electorate's sentiment.

The team’s journey through the complexities of real-time data handling, sentiment analysis, and adapting to technological constraints has provided profound insights and set a benchmark for similar projects we undertake in the future.

7. References

(1) Distributed Scaling of Data

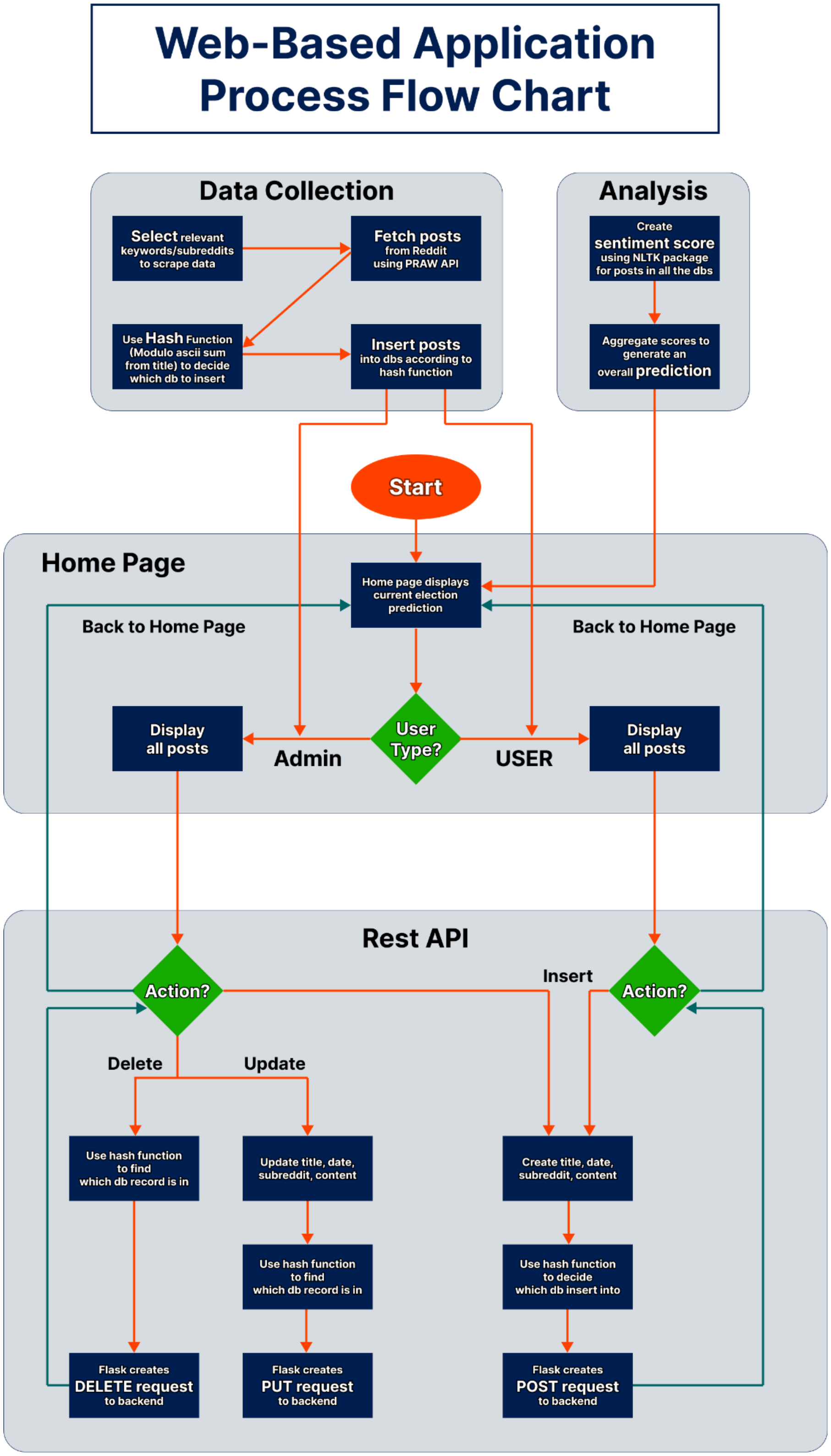
[1] Codecademy, "Definition of Data Distribution and Scaling," Available: https://www.codecademy.com/article/database-scaling-strategies. Accessed: May 2, 2024.

[2] LinkedIn, "Data Distribution and Scaling Approaches," Available: https://www.linkedin.com/advice/0/what-factors-should-you-consider-when-scaling-8f5me#:~:text=Partitioning%20and%20replication%20are%20two,assigning%20them%20to%20different%20nodes. Accessed: May 2, 2024.

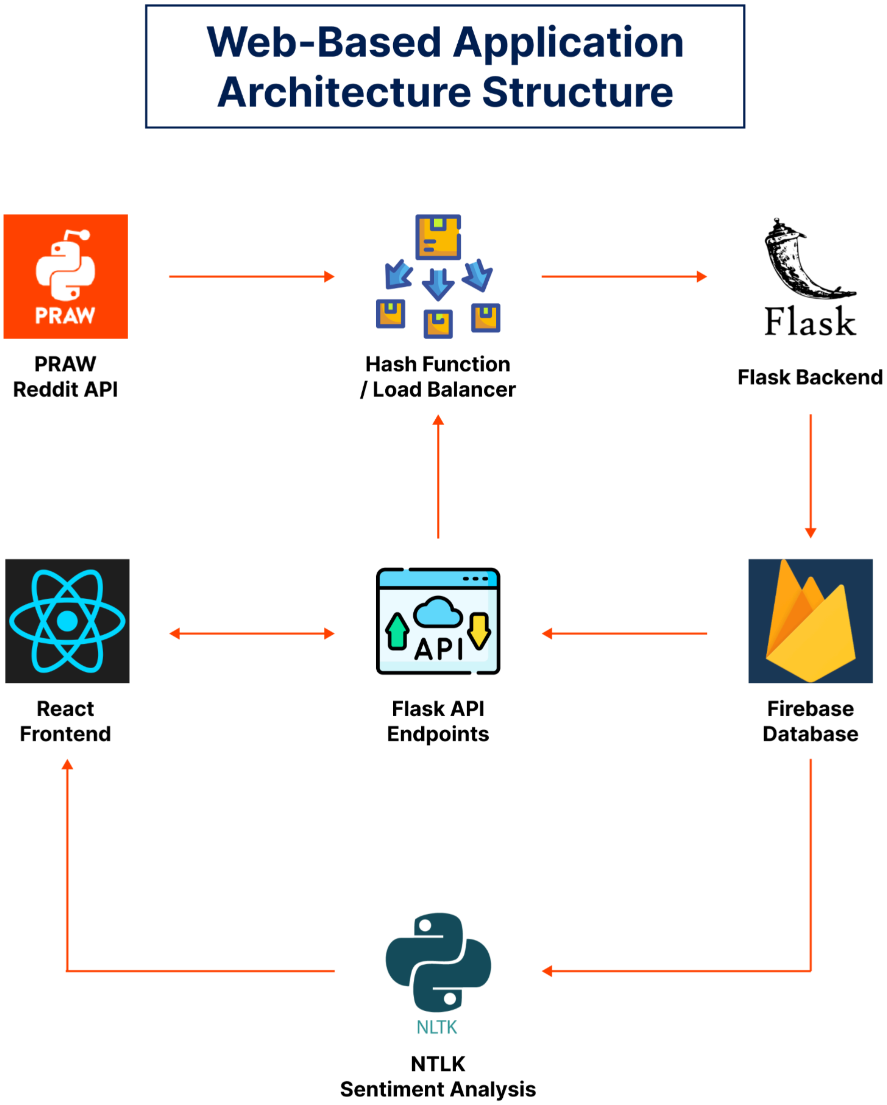
[3] LinkedIn, "Horizontal Scaling," Available: https://www.linkedin.com/advice/0/how-do-you-scale-distribute-your-database#:~:text=Scaling%20and%20distributing%20a%20database,resolution%2C%20and%20monitoring%20are%20key. Accessed: May 2, 2024.

[4] "Natural Language Toolkit," NLTK Project. [Online]. Available: https://www.nltk.org/index.html. Accessed: May 2, 2024.

[5] Process Flow Chart


Web-Based Application
Process Flow Chart
Data Collection
Select relevant keywords/subreddits to scrape data
Fetch posts from Reddit using PRAW API
Use Hash Function (Modulo ascii sum from title) to decide which db to insert
Insert posts into dbs according to hash function
Analysis
Create sentiment score using NLTK package for posts in all the dbs
Aggregate scores to generate an overall prediction
Start
Home Page
Home page displays current election prediction
Back to Home Page
Back to Home Page
Display all posts
Admin
User Type?
USER
Display all posts
Rest API
Insert
Action?
Action?
Delete
Update
Use hash function to find which db record is in
Update title, date, subreddit, content
Create title, date, subreddit, content
Use hash function to find which db record is in
Use hash function to decide which db insert into
Flask creates DELETE request to backend
Flask creates PUT request to backend
Flask creates POST request to backend

[6] Architecture Diagram


Web-Based Application Architecture Structure
PRAW
PRAW Reddit API
Hash Function / Load Balancer
Flask
Flask Backend
React Frontend
API
Flask API Endpoints
Firebase Database
NLTK
NTLK Sentiment Analysis