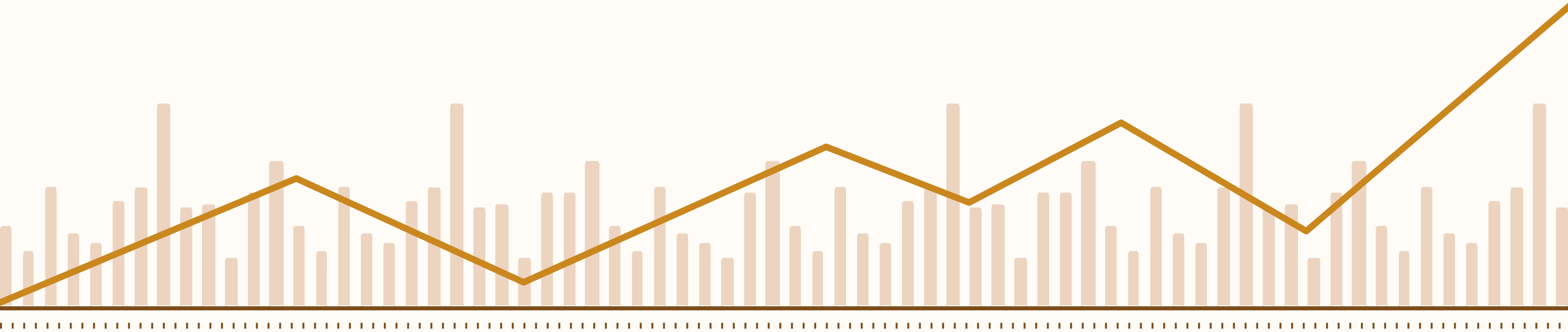


MorningBread

Maximilian Bradshaw and Jake Dorick

Table of contents

1 Background

- Provide a morning financial report to users
- Utilize a web scraping tool for financial website articles
- Find the general topics of the articles
- Create and design a website to give the public a general grasp of financial news for the day
- Provide hourly, highly summarized news headlines and stock tickers
- Implement an account system to allow users to follow certain companies and manage tickers

2

Members and Roles

Max – Back End

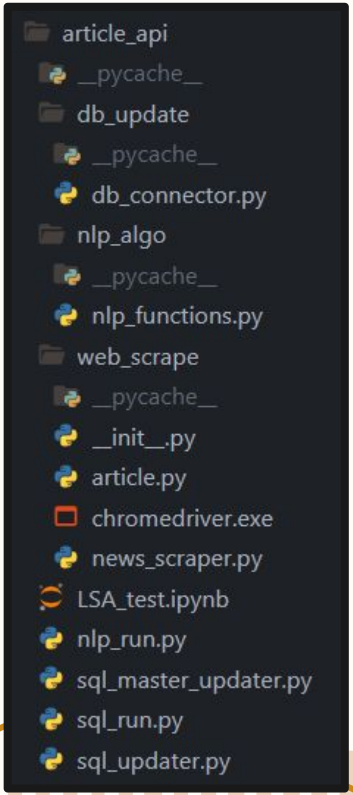
- Provide data to the front end
- Determine news sources
- Parse and process data
- Handle database management
- Handle website hosting

Jake – Front End

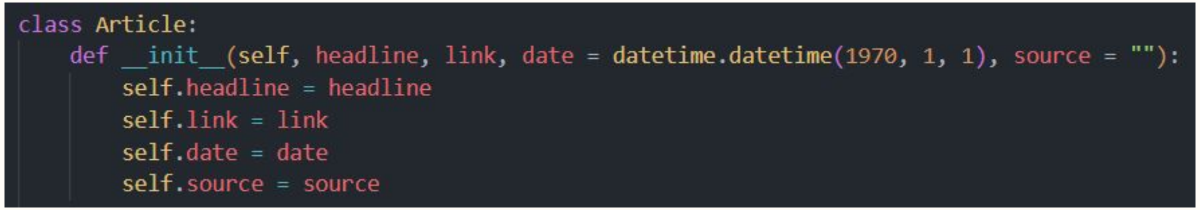
- Develop the aesthetics and logistics of the user interface
- Bring the back end offerings to the front end through APIs
- Dynamically-update content on the website for users

Back End Code Overview

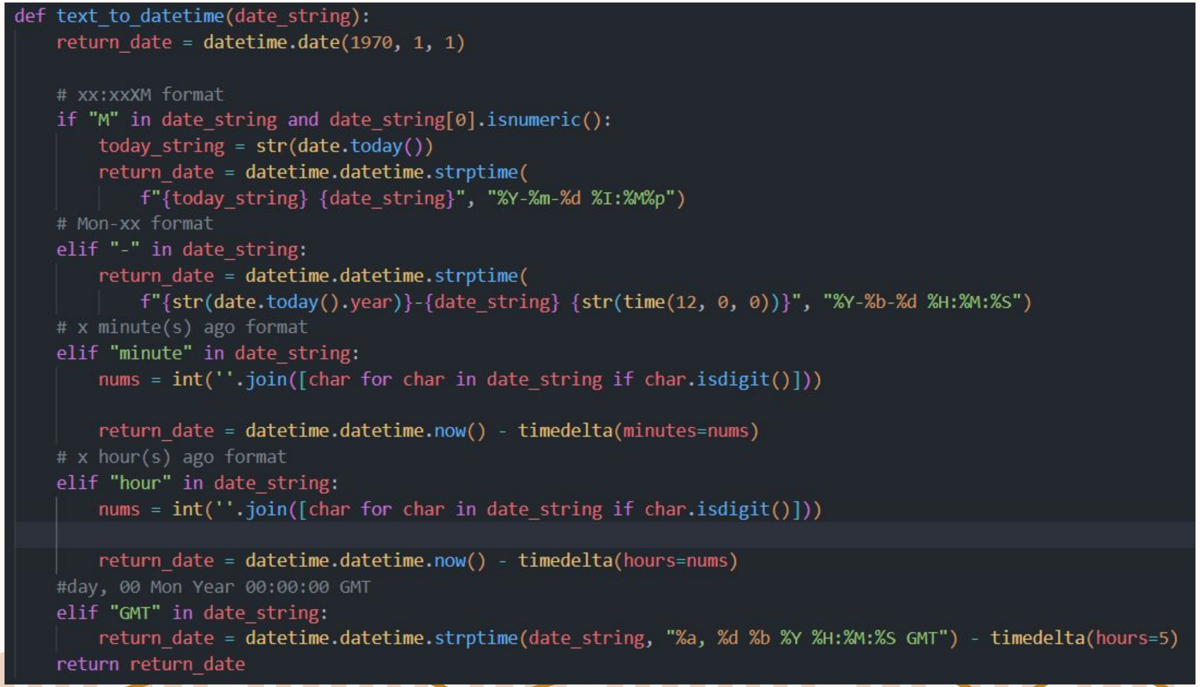
1. **Scraping**
   Scrape predefined websites and reformat all articles, save in a single object
2. **ML Processing**
   Clean text, vectorize (bag of words), group using DBSCAN, optimize with KNN, assign groups in new object, generate summary for each group
3. **Database Upload**
   Upload grouped articles to table, upload summary to new table


```
article_api
  __pycache__
  db_update
    __pycache__
    db_connector.py
  nlp_algo
    __pycache__
    nlp_functions.py
  web_scrape
    __pycache__
    __init__.py
    article.py
    chromedriver.exe
    news_scraper.py
  LSA_test.ipynb
  nlp_run.py
  sql_master_updater.py
  sql_run.py
  sql_updater.py
```

Scraping

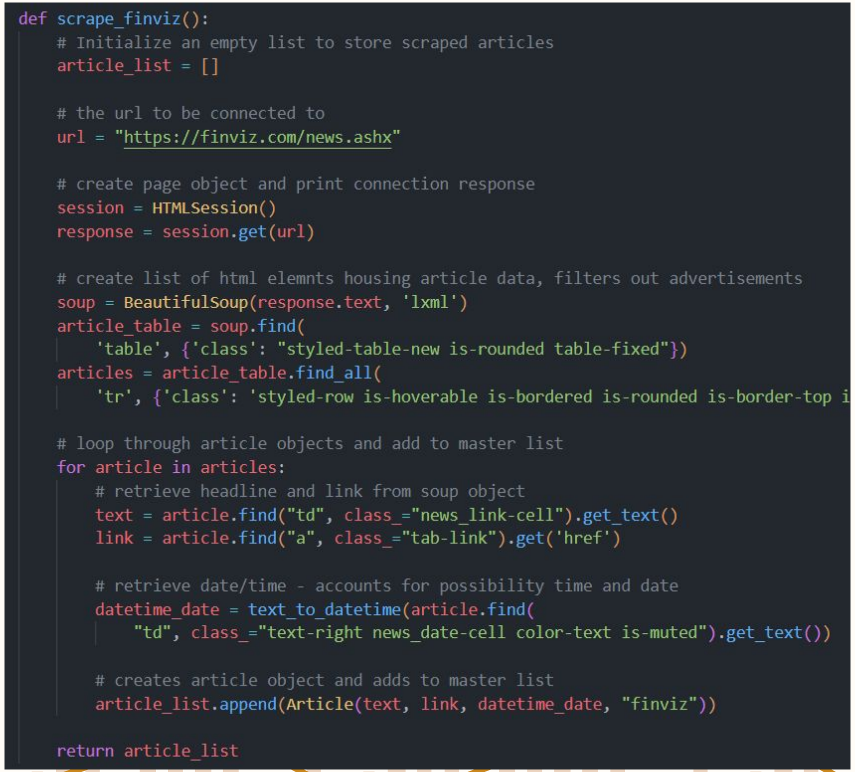


```python
class Article:
    def __init__(self, headline, link, date = datetime.datetime(1970, 1, 1), source = ""):
        self.headline = headline
        self.link = link
        self.date = date
        self.source = source
```

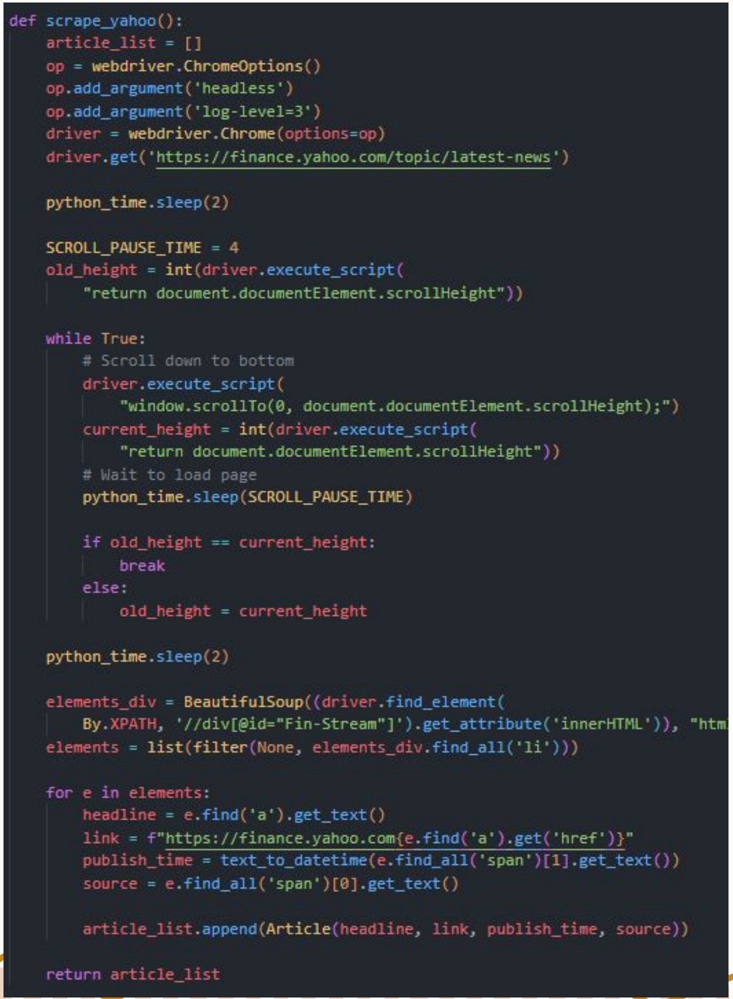


```python
def text_to_datetime(date_string):
    return_date = datetime.date(1970, 1, 1)

    # xx:xxXM format
    if "M" in date_string and date_string[0].isnumeric():
        today_string = str(date.today())
        return_date = datetime.datetime.strptime(
            f"{today_string} {date_string}", "%Y-%m-%d %I:%M%p")
    # Mon-xx format
    elif "-" in date_string:
        return_date = datetime.datetime.strptime(
            f"{str(date.today().year)}-{date_string} {str(time(12, 0, 0))}", "%Y-%b-%d %H:%M:%S")
    # x minute(s) ago format
    elif "minute" in date_string:
        nums = int(''.join([char for char in date_string if char.isdigit()]))

        return_date = datetime.datetime.now() - timedelta(minutes=nums)
    # x hour(s) ago format
    elif "hour" in date_string:
        nums = int(''.join([char for char in date_string if char.isdigit()]))

        return_date = datetime.datetime.now() - timedelta(hours=nums)
    #day, 00 Mon Year 00:00:00 GMT
    elif "GMT" in date_string:
        return_date = datetime.datetime.strptime(date_string, "%a, %d %b %Y %H:%M:%S GMT") - timedelta(hours=5)
    return return_date
```

Scraping

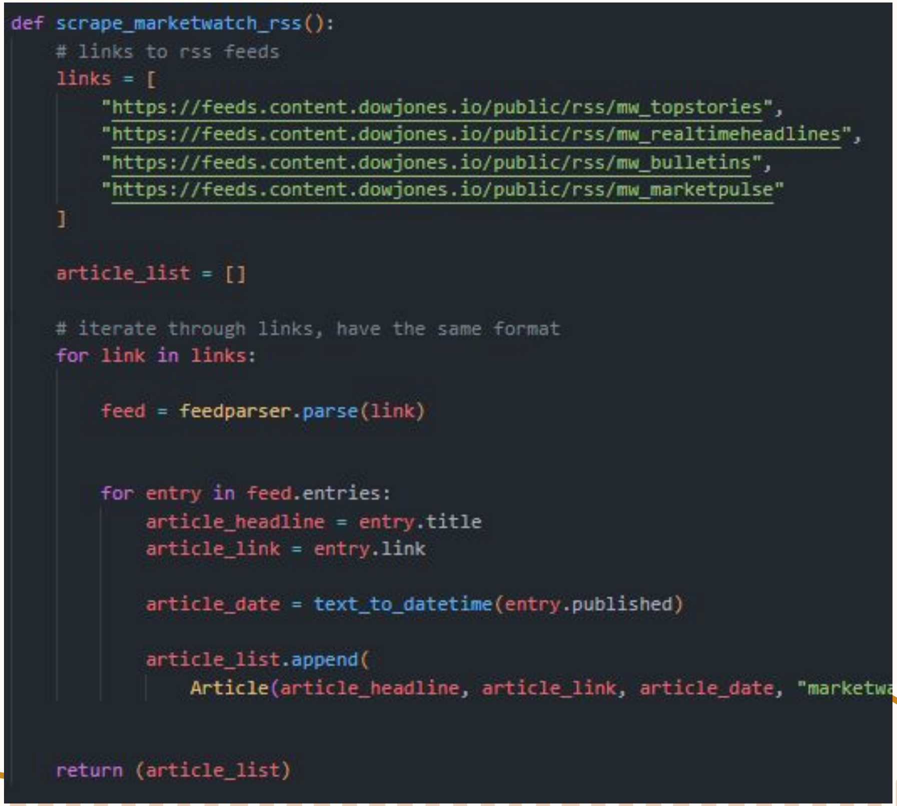


```python
def scrape_finviz():
    # Initialize an empty list to store scraped articles
    article_list = []

    # the url to be connected to
    url = "https://finviz.com/news.ashx"

    # create page object and print connection response
    session = HTMLSession()
    response = session.get(url)

    # create list of html elemnts housing article data, filters out advertisements
    soup = BeautifulSoup(response.text, 'lxml')
    article_table = soup.find(
        'table', {'class': "styled-table-new is-rounded table-fixed"})
    articles = article_table.find_all(
        'tr', {'class': 'styled-row is-hoverable is-bordered is-rounded is-border-top i

    # loop through article objects and add to master list
    for article in articles:
        # retrieve headline and link from soup object
        text = article.find("td", class_="news_link-cell").get_text()
        link = article.find("a", class_="tab-link").get('href')

        # retrieve date/time - accounts for possibility time and date
        datetime_date = text_to_datetime(article.find(
            "td", class_="text-right news_date-cell color-text is-muted").get_text())

        # creates article object and adds to master list
        article_list.append(Article(text, link, datetime_date, "finviz"))

    return article_list
```

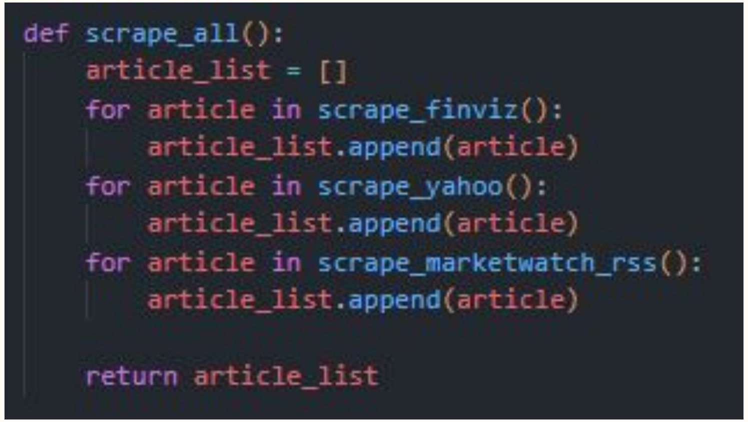


```python
def scrape_yahoo():
    article_list = []
    op = webdriver.ChromeOptions()
    op.add_argument('headless')
    op.add_argument('log-level=3')
    driver = webdriver.Chrome(options=op)
    driver.get('https://finance.yahoo.com/topic/latest-news')

    python_time.sleep(2)

    SCROLL_PAUSE_TIME = 4
    old_height = int(driver.execute_script(
        "return document.documentElement.scrollHeight"))

    while True:
        # Scroll down to bottom
        driver.execute_script(
            "window.scrollTo(0, document.documentElement.scrollHeight);")
        current_height = int(driver.execute_script(
            "return document.documentElement.scrollHeight"))
        # Wait to load page
        python_time.sleep(SCROLL_PAUSE_TIME)

        if old_height == current_height:
            break
        else:
            old_height = current_height

    python_time.sleep(2)

    elements_div = BeautifulSoup((driver.find_element(
        By.XPATH, '//div[@id="Fin-Stream"]').get_attribute('innerHTML')), "htm
    elements = list(filter(None, elements_div.find_all('li')))

    for e in elements:
        headline = e.find('a').get_text()
        link = f"https://finance.yahoo.com{e.find('a').get('href')}"
        publish_time = text_to_datetime(e.find_all('span')[1].get_text())
        source = e.find_all('span')[0].get_text()

        article_list.append(Article(headline, link, publish_time, source))

    return article_list
```

Scraping

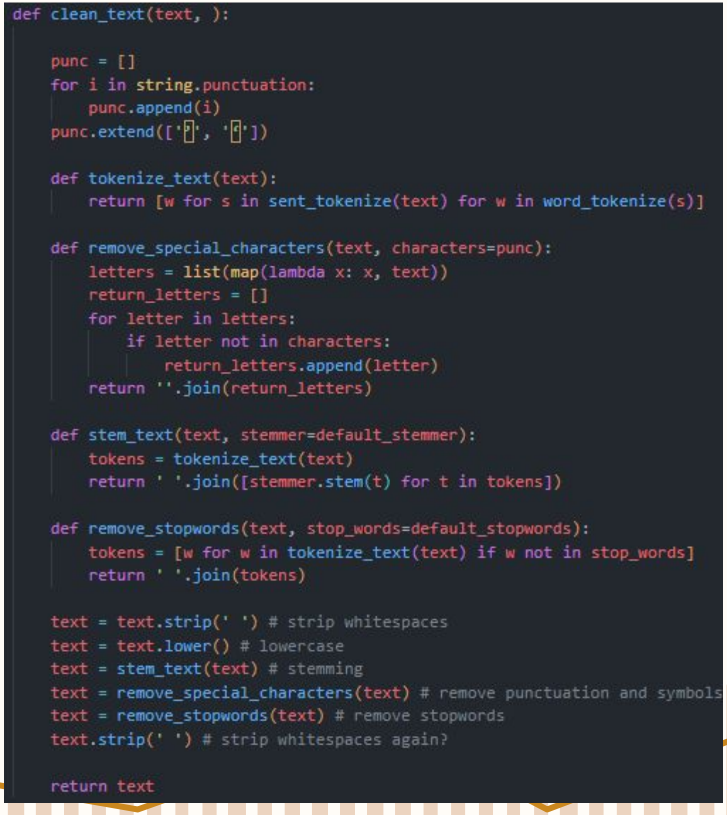


```
def scrape_marketwatch_rss():
    # links to rss feeds
    links = [
        "https://feeds.content.dowjones.io/public/rss/mw_topstories",
        "https://feeds.content.dowjones.io/public/rss/mw_realtimeheadlines",
        "https://feeds.content.dowjones.io/public/rss/mw_bulletins",
        "https://feeds.content.dowjones.io/public/rss/mw_marketpulse"
    ]

    article_list = []

    # iterate through links, have the same format
    for link in links:

        feed = feedparser.parse(link)


        for entry in feed.entries:
            article_headline = entry.title
            article_link = entry.link

            article_date = text_to_datetime(entry.published)

            article_list.append(
                Article(article_headline, article_link, article_date, "marketwa


    return (article_list)
```

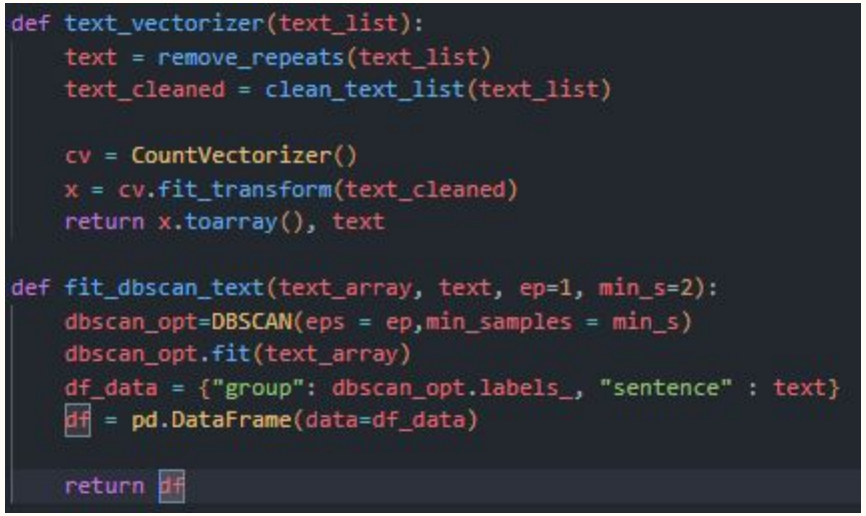


```
def scrape_all():
    article_list = []
    for article in scrape_finviz():
        article_list.append(article)
    for article in scrape_yahoo():
        article_list.append(article)
    for article in scrape_marketwatch_rss():
        article_list.append(article)

    return article_list
```

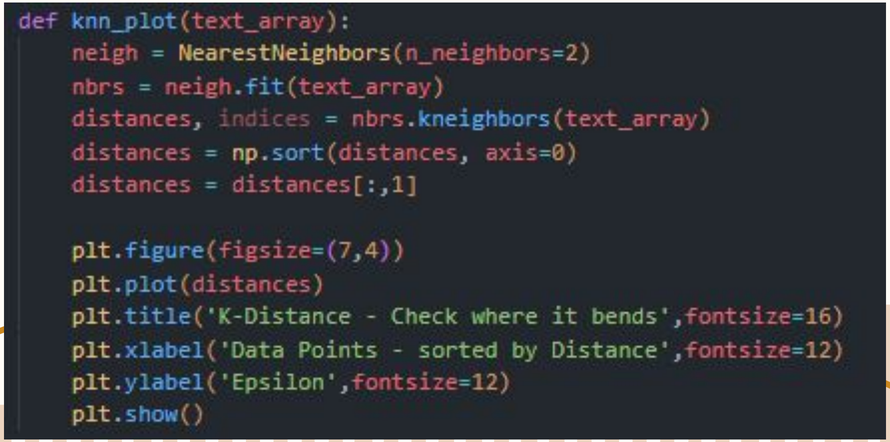
ML Processing



```python
def clean_text(text, ):

    punc = []
    for i in string.punctuation:
        punc.append(i)
    punc.extend(['؟', '،'])

    def tokenize_text(text):
        return [w for s in sent_tokenize(text) for w in word_tokenize(s)]

    def remove_special_characters(text, characters=punc):
        letters = list(map(lambda x: x, text))
        return_letters = []
        for letter in letters:
            if letter not in characters:
                return_letters.append(letter)
        return ''.join(return_letters)

    def stem_text(text, stemmer=default_stemmer):
        tokens = tokenize_text(text)
        return ' '.join([stemmer.stem(t) for t in tokens])

    def remove_stopwords(text, stop_words=default_stopwords):
        tokens = [w for w in tokenize_text(text) if w not in stop_words]
        return ' '.join(tokens)

    text = text.strip(' ') # strip whitespaces
    text = text.lower() # lowercase
    text = stem_text(text) # stemming
    text = remove_special_characters(text) # remove punctuation and symbols
    text = remove_stopwords(text) # remove stopwords
    text.strip(' ') # strip whitespaces again?

    return text
```

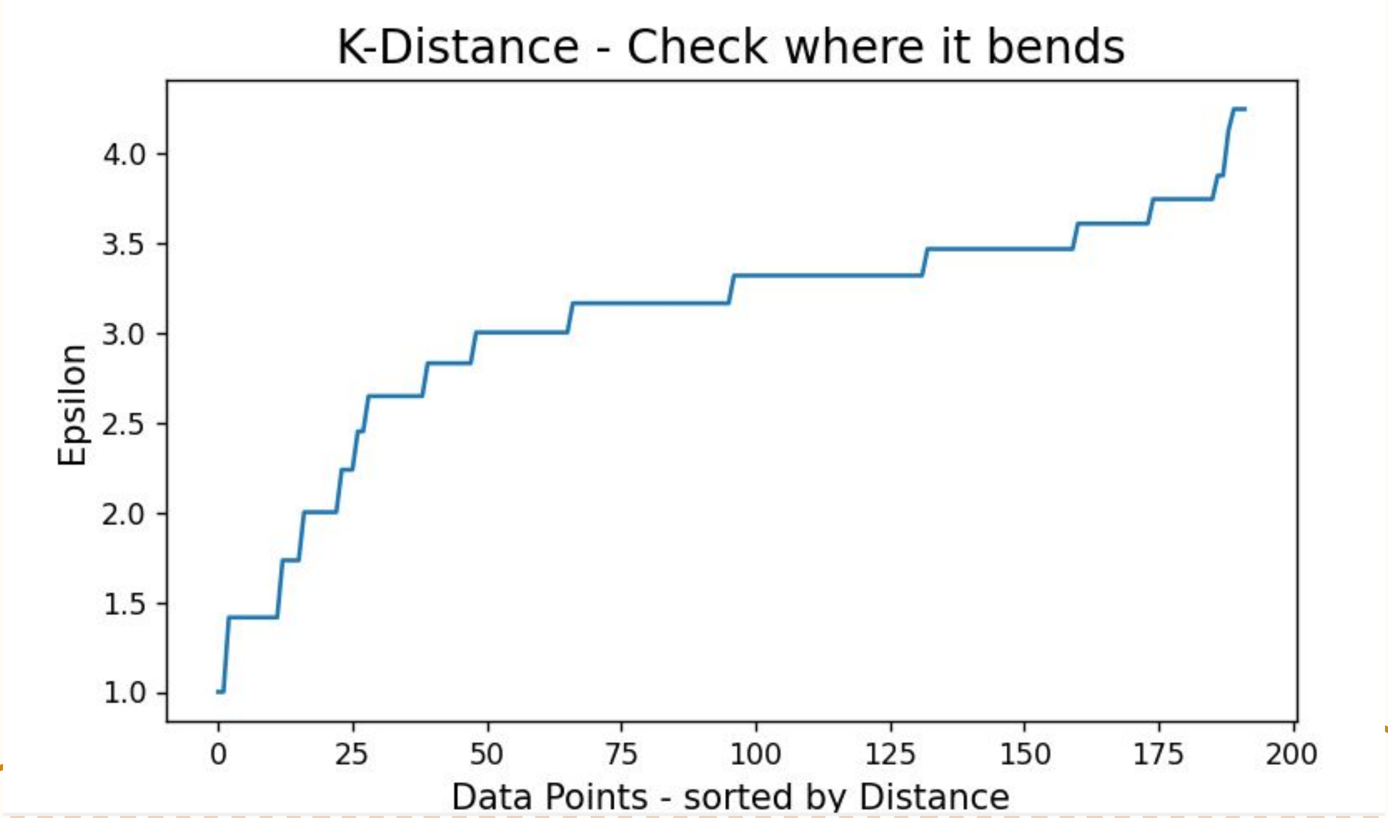


```python
def text_vectorizer(text_list):
    text = remove_repeats(text_list)
    text_cleaned = clean_text_list(text_list)

    cv = CountVectorizer()
    x = cv.fit_transform(text_cleaned)
    return x.toarray(), text

def fit_dbscan_text(text_array, text, ep=1, min_s=2):
    dbscan_opt=DBSCAN(eps = ep,min_samples = min_s)
    dbscan_opt.fit(text_array)
    df_data = {"group": dbscan_opt.labels_, "sentence" : text}
    df = pd.DataFrame(data=df_data)

    return df
```

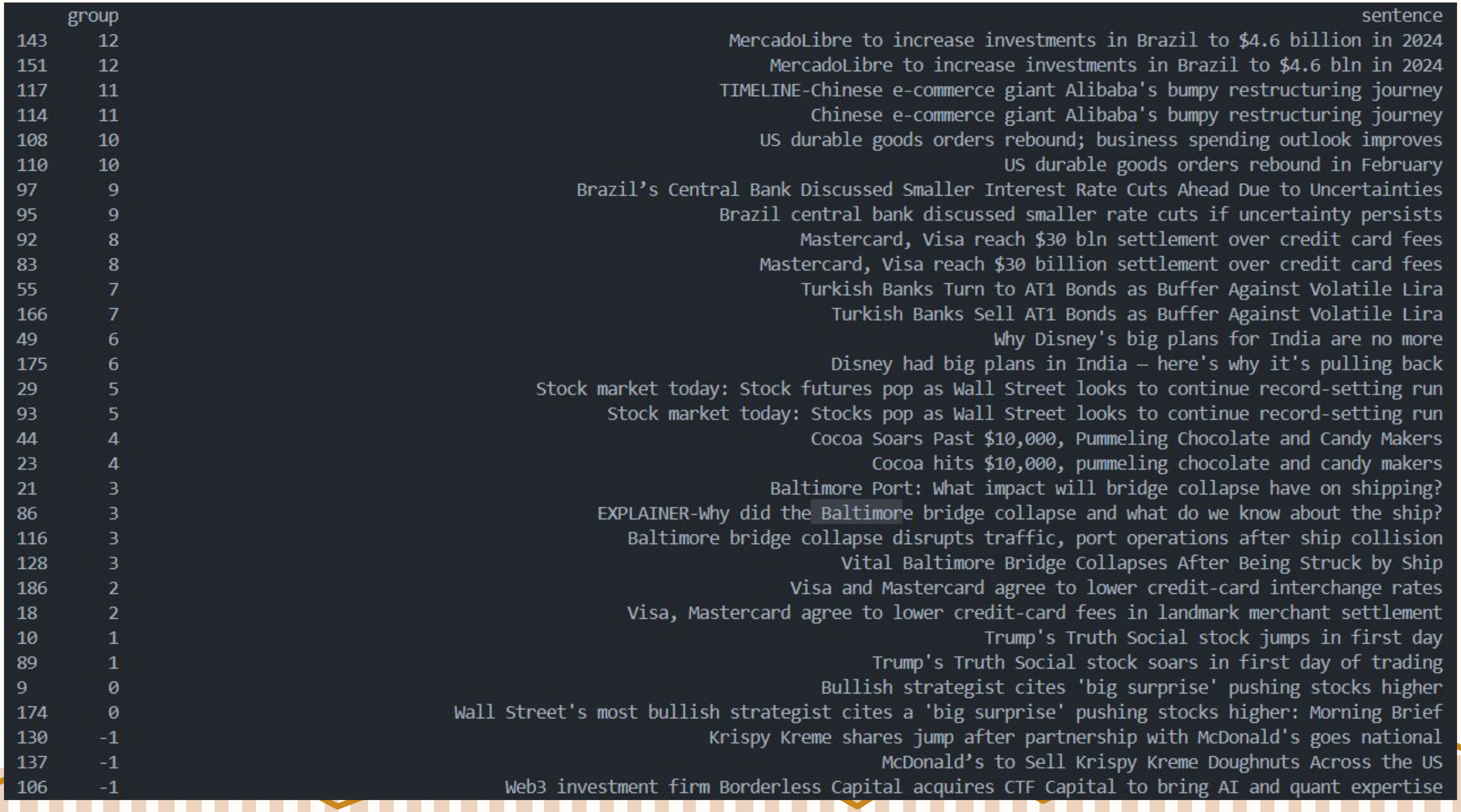


```python
def knn_plot(text_array):
    neigh = NearestNeighbors(n_neighbors=2)
    nbrs = neigh.fit(text_array)
    distances, indices = nbrs.kneighbors(text_array)
    distances = np.sort(distances, axis=0)
    distances = distances[:,1]

    plt.figure(figsize=(7,4))
    plt.plot(distances)
    plt.title('K-Distance - Check where it bends',fontsize=16)
    plt.xlabel('Data Points - sorted by Distance',fontsize=12)
    plt.ylabel('Epsilon',fontsize=12)
    plt.show()
```

ML Processing

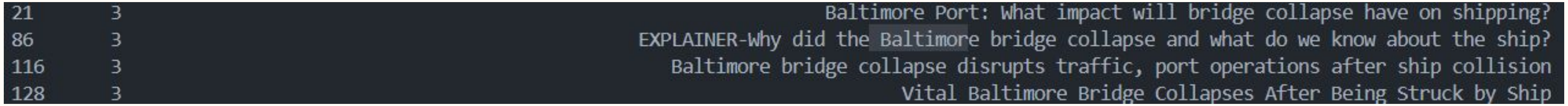


ML Processing



```
     group                                                                                          sentence
143     12                                       MercadoLibre to increase investments in Brazil to $4.6 billion in 2024
151     12                                           MercadoLibre to increase investments in Brazil to $4.6 bln in 2024
117     11                                       TIMELINE-Chinese e-commerce giant Alibaba's bumpy restructuring journey
114     11                                                Chinese e-commerce giant Alibaba's bumpy restructuring journey
108     10                                            US durable goods orders rebound; business spending outlook improves
110     10                                                                    US durable goods orders rebound in February
97       9                               Brazil’s Central Bank Discussed Smaller Interest Rate Cuts Ahead Due to Uncertainties
95       9                                       Brazil central bank discussed smaller rate cuts if uncertainty persists
92       8                                                Mastercard, Visa reach $30 bln settlement over credit card fees
83       8                                            Mastercard, Visa reach $30 billion settlement over credit card fees
55       7                                                Turkish Banks Turn to AT1 Bonds as Buffer Against Volatile Lira
166      7                                                   Turkish Banks Sell AT1 Bonds as Buffer Against Volatile Lira
49       6                                                                Why Disney's big plans for India are no more
175      6                                                   Disney had big plans in India — here's why it's pulling back
29       5                                 Stock market today: Stock futures pop as Wall Street looks to continue record-setting run
93       5                                        Stock market today: Stocks pop as Wall Street looks to continue record-setting run
44       4                                                     Cocoa Soars Past $10,000, Pummeling Chocolate and Candy Makers
23       4                                                           Cocoa hits $10,000, pummeling chocolate and candy makers
21       3                                                 Baltimore Port: What impact will bridge collapse have on shipping?
86       3                                   EXPLAINER-Why did the Baltimore bridge collapse and what do we know about the ship?
116      3                                      Baltimore bridge collapse disrupts traffic, port operations after ship collision
128      3                                                        Vital Baltimore Bridge Collapses After Being Struck by Ship
186      2                                                   Visa and Mastercard agree to lower credit-card interchange rates
18       2                                     Visa, Mastercard agree to lower credit-card fees in landmark merchant settlement
10       1                                                                       Trump's Truth Social stock jumps in first day
89       1                                                            Trump's Truth Social stock soars in first day of trading
9        0                                                         Bullish strategist cites 'big surprise' pushing stocks higher
174      0                         Wall Street's most bullish strategist cites a 'big surprise' pushing stocks higher: Morning Brief
130     -1                                              Krispy Kreme shares jump after partnership with McDonald's goes national
137     -1                                                              McDonald’s to Sell Krispy Kreme Doughnuts Across the US
106     -1                               Web3 investment firm Borderless Capital acquires CTF Capital to bring AI and quant expertise
```

ML Processing

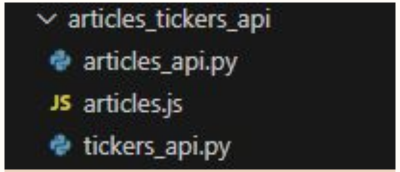


```
21      3                  Baltimore Port: What impact will bridge collapse have on shipping?
86      3    EXPLAINER-Why did the Baltimore bridge collapse and what do we know about the ship?
116     3        Baltimore bridge collapse disrupts traffic, port operations after ship collision
128     3                        Vital Baltimore Bridge Collapses After Being Struck by Ship
```

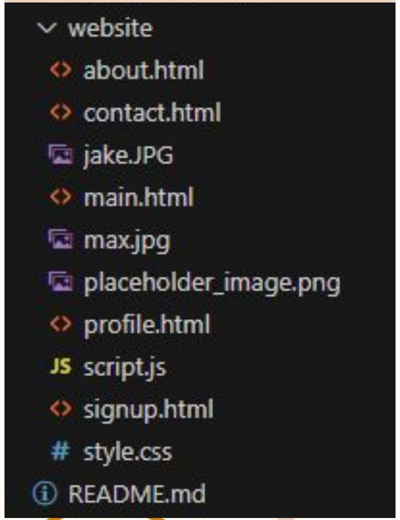
2

Front End Code Overview

1. **Formatting in HTML**
   A majority of the website's pages have the same layout as one another with more or less features depending on its function
2. **Styling in CSS**
   style.css has each page blocked off with comments to separate individual style designs
3. **External and inline JavaScript**
   From displaying the current date and time, to showing pop ups when the user submits a comment form, to tying the API into the HTML



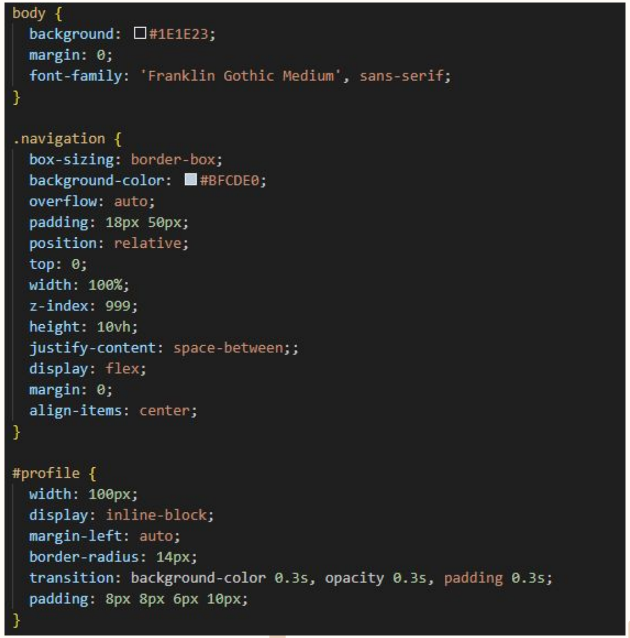
```
articles_tickers_api
  articles_api.py
  articles.js
  tickers_api.py
```



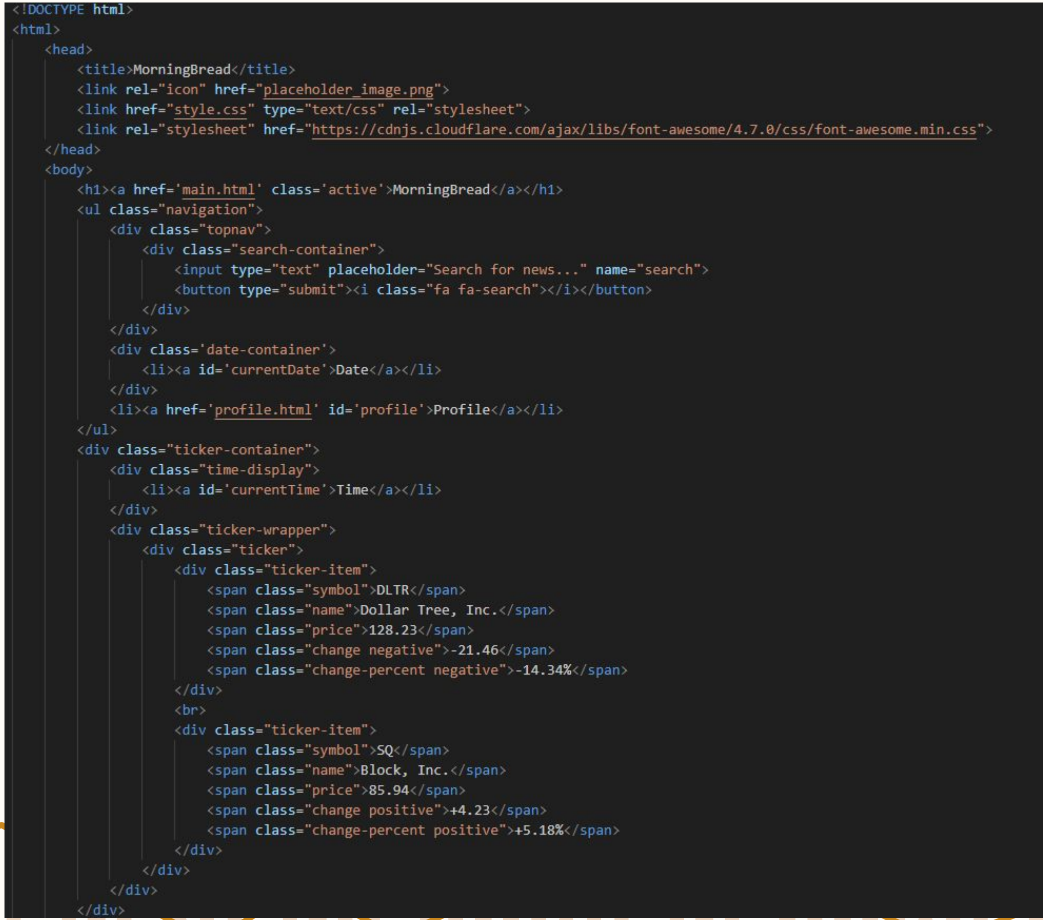
```
website
  about.html
  contact.html
  jake.JPG
  main.html
  max.jpg
  placeholder_image.png
  profile.html
  script.js
  signup.html
  style.css
README.md
```

General Structure

in HTML and CSS

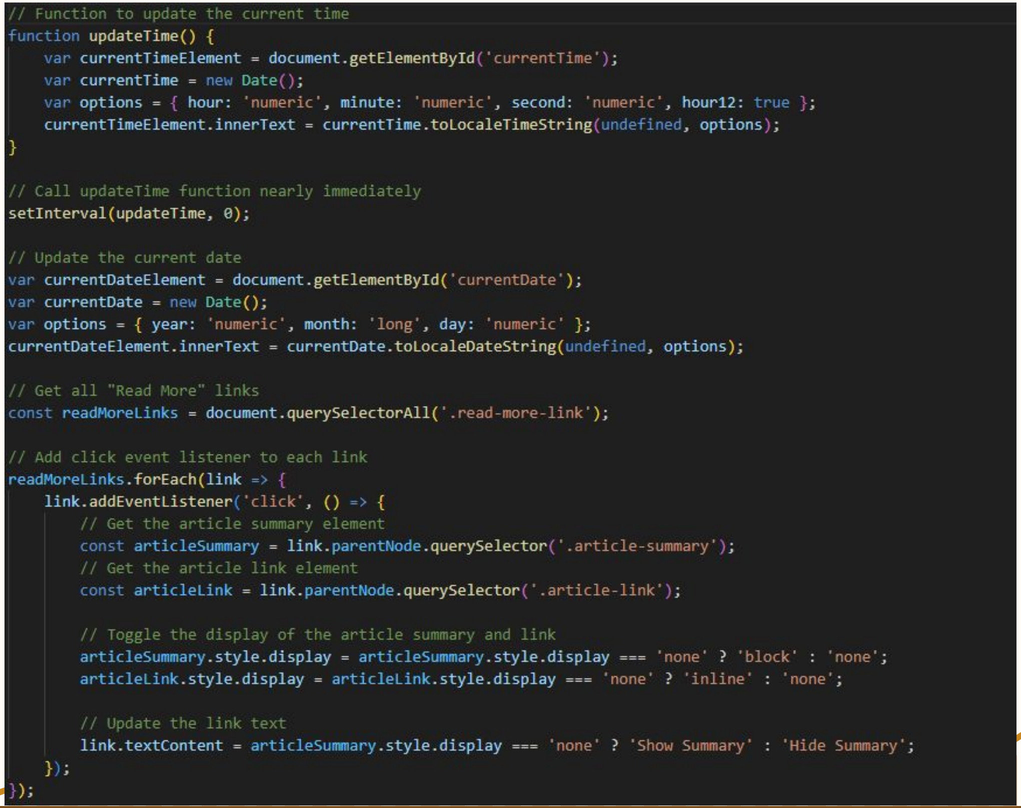


```css
body {
  background: ■#1E1E23;
  margin: 0;
  font-family: 'Franklin Gothic Medium', sans-serif;
}

.navigation {
  box-sizing: border-box;
  background-color: ■#BFCDE0;
  overflow: auto;
  padding: 18px 50px;
  position: relative;
  top: 0;
  width: 100%;
  z-index: 999;
  height: 10vh;
  justify-content: space-between;;
  display: flex;
  margin: 0;
  align-items: center;
}

#profile {
  width: 100px;
  display: inline-block;
  margin-left: auto;
  border-radius: 14px;
  transition: background-color 0.3s, opacity 0.3s, padding 0.3s;
  padding: 8px 8px 6px 10px;
}
```

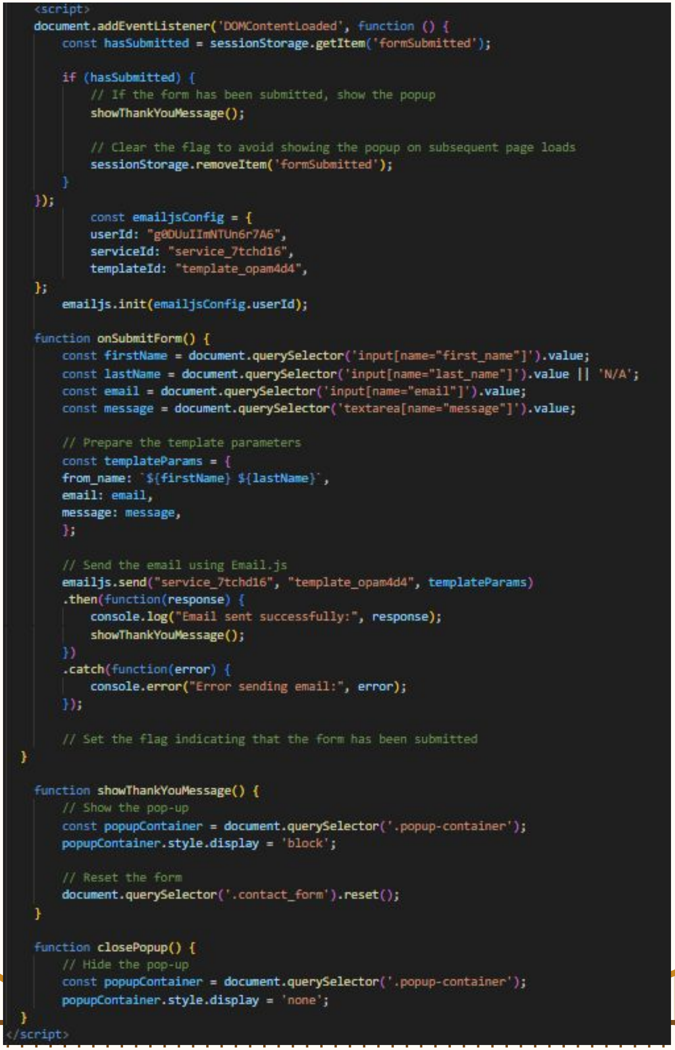


```html
<!DOCTYPE html>
<html>
    <head>
        <title>MorningBread</title>
        <link rel="icon" href="placeholder_image.png">
        <link href="style.css" type="text/css" rel="stylesheet">
        <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
    </head>
    <body>
        <h1><a href='main.html' class='active'>MorningBread</a></h1>
        <ul class="navigation">
            <div class="topnav">
                <div class="search-container">
                    <input type="text" placeholder="Search for news..." name="search">
                    <button type="submit"><i class="fa fa-search"></i></button>
                </div>
            </div>
            <div class='date-container'>
                <li><a id='currentDate'>Date</a></li>
            </div>
            <li><a href='profile.html' id='profile'>Profile</a></li>
        </ul>
        <div class="ticker-container">
            <div class="time-display">
                <li><a id='currentTime'>Time</a></li>
            </div>
            <div class="ticker-wrapper">
                <div class="ticker">
                    <div class="ticker-item">
                        <span class="symbol">DLTR</span>
                        <span class="name">Dollar Tree, Inc.</span>
                        <span class="price">128.23</span>
                        <span class="change negative">-21.46</span>
                        <span class="change-percent negative">-14.34%</span>
                    </div>
                    <br>
                    <div class="ticker-item">
                        <span class="symbol">SQ</span>
                        <span class="name">Block, Inc.</span>
                        <span class="price">85.94</span>
                        <span class="change positive">+4.23</span>
                        <span class="change-percent positive">+5.18%</span>
                    </div>
                </div>
            </div>
        </div>
```

JavaScript Functions

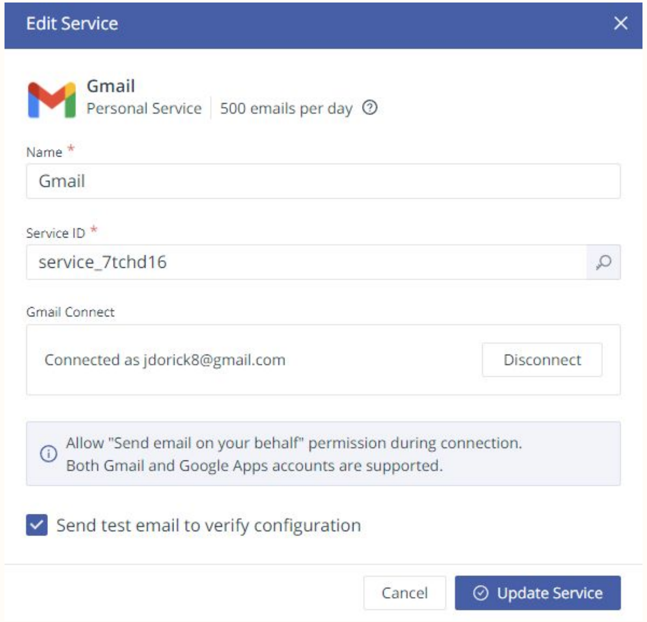


```javascript
// Function to update the current time
function updateTime() {
    var currentTimeElement = document.getElementById('currentTime');
    var currentTime = new Date();
    var options = { hour: 'numeric', minute: 'numeric', second: 'numeric', hour12: true };
    currentTimeElement.innerText = currentTime.toLocaleTimeString(undefined, options);
}

// Call updateTime function nearly immediately
setInterval(updateTime, 0);

// Update the current date
var currentDateElement = document.getElementById('currentDate');
var currentDate = new Date();
var options = { year: 'numeric', month: 'long', day: 'numeric' };
currentDateElement.innerText = currentDate.toLocaleDateString(undefined, options);

// Get all "Read More" links
const readMoreLinks = document.querySelectorAll('.read-more-link');

// Add click event listener to each link
readMoreLinks.forEach(link => {
    link.addEventListener('click', () => {
        // Get the article summary element
        const articleSummary = link.parentNode.querySelector('.article-summary');
        // Get the article link element
        const articleLink = link.parentNode.querySelector('.article-link');

        // Toggle the display of the article summary and link
        articleSummary.style.display = articleSummary.style.display === 'none' ? 'block' : 'none';
        articleLink.style.display = articleLink.style.display === 'none' ? 'inline' : 'none';

        // Update the link text
        link.textContent = articleSummary.style.display === 'none' ? 'Show Summary' : 'Hide Summary';
    });
});
```



```html
<script>
document.addEventListener('DOMContentLoaded', function () {
    const hasSubmitted = sessionStorage.getItem('formSubmitted');

    if (hasSubmitted) {
        // If the form has been submitted, show the popup
        showThankYouMessage();

        // Clear the flag to avoid showing the popup on subsequent page loads
        sessionStorage.removeItem('formSubmitted');
    }
});
        const emailjsConfig = {
        userId: "g0DUuIImNTUn6r7A6",
        serviceId: "service_7tchd16",
        templateId: "template_opam4d4",
};
    emailjs.init(emailjsConfig.userId);

function onSubmitForm() {
    const firstName = document.querySelector('input[name="first_name"]').value;
    const lastName = document.querySelector('input[name="last_name"]').value || 'N/A';
    const email = document.querySelector('input[name="email"]').value;
    const message = document.querySelector('textarea[name="message"]').value;

    // Prepare the template parameters
    const templateParams = {
    from_name: `${firstName} ${lastName}`,
    email: email,
    message: message,
    };

    // Send the email using Email.js
    emailjs.send("service_7tchd16", "template_opam4d4", templateParams)
    .then(function(response) {
        console.log("Email sent successfully:", response);
        showThankYouMessage();
    })
    .catch(function(error) {
        console.error("Error sending email:", error);
    });

    // Set the flag indicating that the form has been submitted
}

function showThankYouMessage() {
    // Show the pop-up
    const popupContainer = document.querySelector('.popup-container');
    popupContainer.style.display = 'block';

    // Reset the form
    document.querySelector('.contact_form').reset();
}

function closePopup() {
    // Hide the pop-up
    const popupContainer = document.querySelector('.popup-container');
    popupContainer.style.display = 'none';
}
</script>
```

EmailJS


Edit Service

Gmail
Personal Service | 500 emails per day

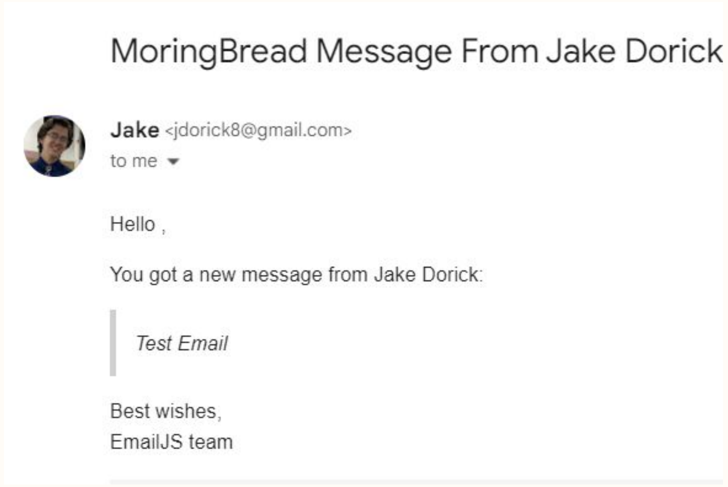
Name *
Gmail

Service ID *
service_7tchd16

Gmail Connect
Connected as jdorick8@gmail.com

Allow "Send email on your behalf" permission during connection.
Both Gmail and Google Apps accounts are supported.

[✓] Send test email to verify configuration


MoringBread Message From Jake Dorick

Jake <jdorick8@gmail.com>
to me

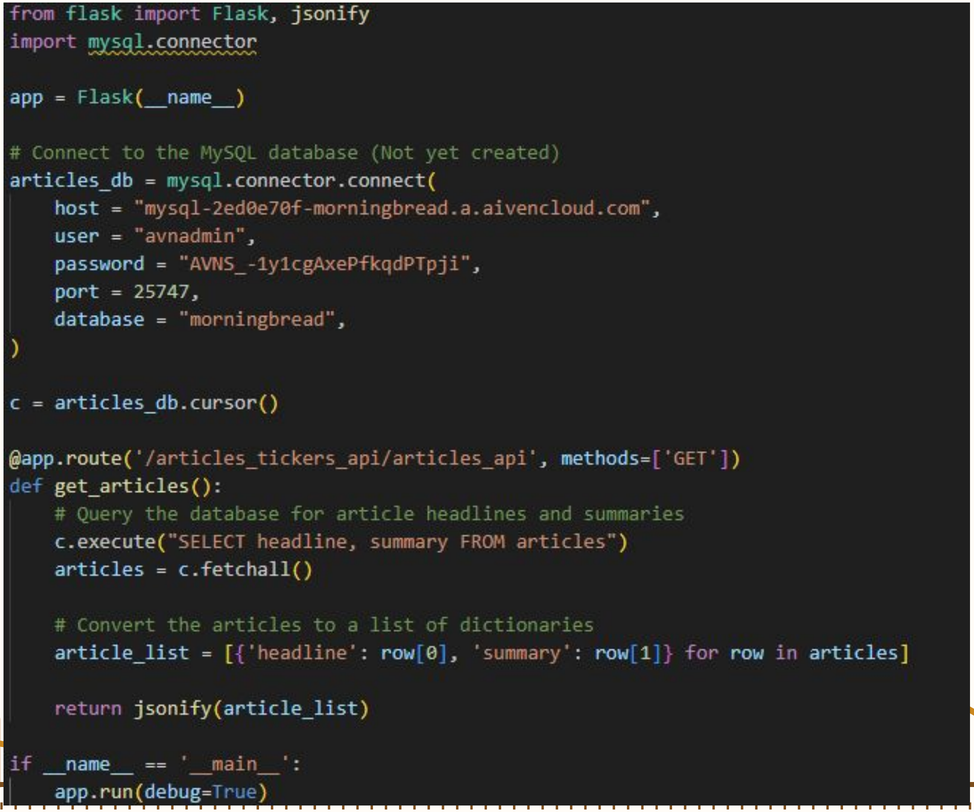
Hello ,

You got a new message from Jake Dorick:

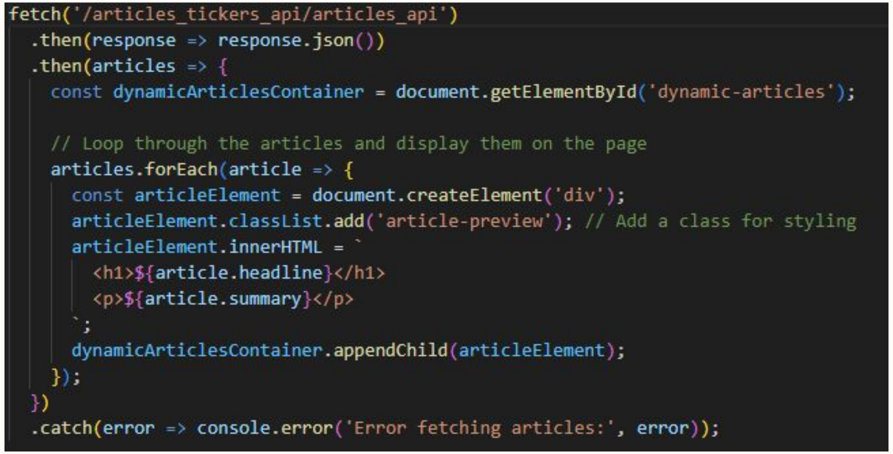
> *Test Email*

Best wishes,
EmailJS team

API (Flask)



```python
from flask import Flask, jsonify
import mysql.connector

app = Flask(__name__)

# Connect to the MySQL database (Not yet created)
articles_db = mysql.connector.connect(
    host = "mysql-2ed0e70f-morningbread.a.aivencloud.com",
    user = "avnadmin",
    password = "AVNS_-1y1cgAxePfkqdPTpji",
    port = 25747,
    database = "morningbread",
)

c = articles_db.cursor()

@app.route('/articles_tickers_api/articles_api', methods=['GET'])
def get_articles():
    # Query the database for article headlines and summaries
    c.execute("SELECT headline, summary FROM articles")
    articles = c.fetchall()

    # Convert the articles to a list of dictionaries
    article_list = [{'headline': row[0], 'summary': row[1]} for row in articles]

    return jsonify(article_list)

if __name__ == '__main__':
    app.run(debug=True)
```



```javascript
fetch('/articles_tickers_api/articles_api')
  .then(response => response.json())
  .then(articles => {
    const dynamicArticlesContainer = document.getElementById('dynamic-articles');

    // Loop through the articles and display them on the page
    articles.forEach(article => {
      const articleElement = document.createElement('div');
      articleElement.classList.add('article-preview'); // Add a class for styling
      articleElement.innerHTML = `
        <h1>${article.headline}</h1>
        <p>${article.summary}</p>
      `;
      dynamicArticlesContainer.appendChild(articleElement);
    });
  })
  .catch(error => console.error('Error fetching articles:', error));
```

Current Top Priority!

3

Technologies – Back End

- Languages
  - Python
  - SQL
- Applications
  - VS Code
  - MySQL Workbench
- Data Host
  - Aiven

- Libraries / Packages
  - mysql.connector
  - sklearn
  - matplotlib
  - nltk
  - bs4
  - requests_html
  - selenium

Technologies – Front End

- Web Development Languages
  - HTML
  - CSS
  - JavaScript
- API Building Languages
  - Python
  - JavaScript
- API Building Services
  - Flask
  - MySQL Workbench
- Applications and Services
  - VSCode
  - EmailJS

4 Progress & Challenges – Back End

- ~80% Complete with base functionality
    - Web scraping complete, ML finished but needs optimization, database upload is in progress
- Web scraping proved difficult and required much trial and error
- ML implementation was fairly easy, but required a lot of research beforehand to ensure I was using the right model

Progress & Challenges – Front End

- All of the individual pages we would need are completed at this point
    - Pop-ups for signing/logging in still need to be made, but for that to happen the account system needs to be made first
- The greatest issues have come recently in trying to connect the MySQL database to the website to display information it contains

5

Priorities

Back End

- Group summarization using LLM
- Finish linking with DB
- Rewrite text cleaning function
- Automatic running using remote server and CRON
- General refactoring

Front End

- Import the APIs for the example articles and tickers (*reach*) onto the website's main page
- Ensure all aesthetics stay the same and user functionality works as intended
- Update the page regularly

5

Reach Goals

- Search function using yahoo search
- Account System
  - Favoriting stocks
  - Personalized home page
- Live tickers on sidebar
- Optimization

6

Demo!