CAB432 Cloud Computing – Semester 2 2019

Assignment 1 Mashup Project Report

John Santias N9983244

Contents

Introduction 2
News API 2
IBM Watson Natural Language Understanding 2
Google Charts 2
Use Cases 3
Use Case A 3
Use Case B 4
Use Case C 5
Use Case D 6
Technical Breakdown 7
Client side 7
Server side 8
Response Filtering 8
Docker 9
PUG 9
Difficulties 10
Storing the response 10
Analysing article content 10
Extensions 11
Related YouTube Videos 11
Advanced Search 11
Testing 12
Snippet A 12
Snippet B 13
Test Cases 14
Appendix A – User Guide 15
Appendix B – Test Case Screenshots 16

Introduction

Reporial provides users the latest news information with an emotional analysis of the content. Utilising the NewsAPI, IBM Watson Natural Language Understanding and Google Charts, users can search for articles based on any topic and be presented with graphed information based on the emotions expressed in the article. Articles presented to the user can come from thousands of other news sources.

When a user searches for a topic, the website will automatically fill up with articles in a tile card format. Each card contains a headline, image, description and a graph displaying the emotional analysis.

News API

https://newsapi.org/

The News API is a simple HTTP REST API for searching and retrieving articles live from all over the web. News API obtains articles from over 30,000 news sources and blogs for reporial to retrieve data.

Reporial utilises one API endpoint provided; *newsapi.v2.everything or https://newsapi.org/v2/everything,* which is used to search for articles based on a search query. This endpoint returns a maximum of 25 articles sorted by published date.

IBM Watson Natural Language Understanding

https://www.ibm.com/watson/services/natural-language-understanding/

The IBM Watson Natural Language Understanding API is one of IBM's Watsons machine learning API. It is used to process advanced text analysis and extract metadata from content such as concepts, entities, keywords, categories, sentiment, emotions, relations, and semantic roles. Some of which can provide a score for the tone or a list of words that have great meaning for the user.

Reporial uses this API to analyse the emotion the article content and return a score between zero and one. Given the highest score is the emotion analysed in the content.

Google Charts

https://developers.google.com/chart/

Google Charts is a JavaScript API used to display visualized data on websites, From simple line charts to complex hierarchical treemaps, google's chart gallery provides a large number of ready-to-use charts. The charts are interactive and rendered using HTML5/SVG technology to provide cross-browser compatibility (including VML and older browsers) and cross-platform compatibility to smartphones (iOS/Android).

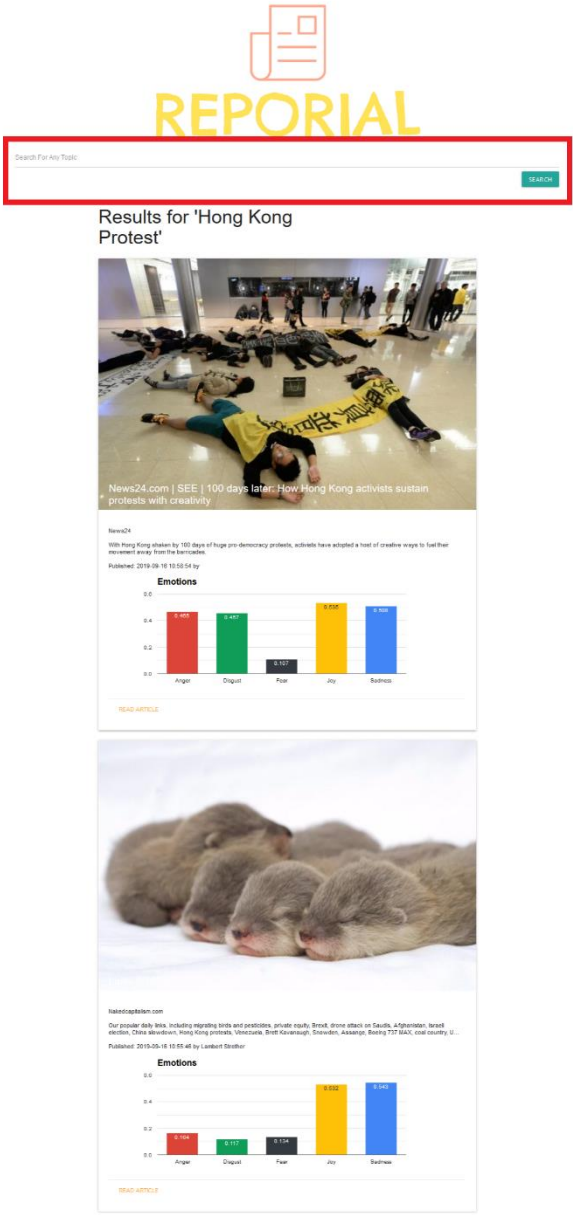
Reporial includes the library within the client-side and is obtained via CDN. The charts values are displayed when articles have been analysed and the web page has completed loading.

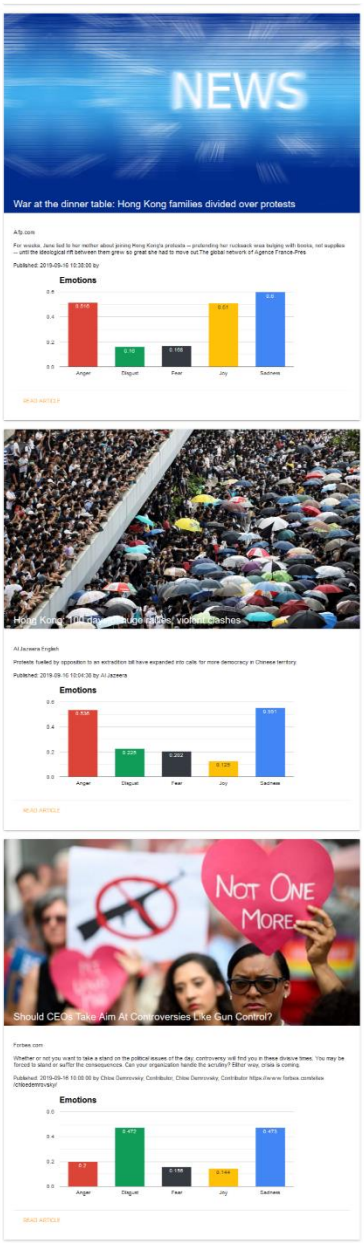
Use Cases

Use Case A

As a newsreader, I want to be able to search for articles based on a topic so I can scroll through a list of headlines and read the ones based that I'm interested in.

User can run a search at the top of the page. When initiated, the server will return a page full of results.





Use Case B

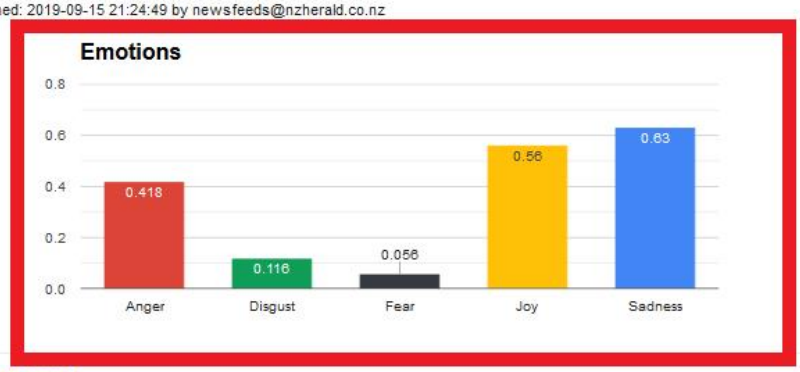
As a newsreader, I want to search for articles based on a topic and see an analysis of each article so that I can see a visual representation of emotional analysis of the article's content.

After running a search, the results are displayed on the page. Underneath each article will reveal a column chart handled by Google Charts API. Displaying the score for each emotion predicted by the IBM Natural language understanding API. The highest score is the best emotion.

Nzherald.co.nz

With the Rugby World Cup now just days away, Spark Sport came close to keeping a clean sheet during the weekend's English Premier League streaming - only to concede a snag with its Manchester United-Leicester...

Published: 2019-09-15 21:24:49 by newsfeeds@nzherald.co.nz

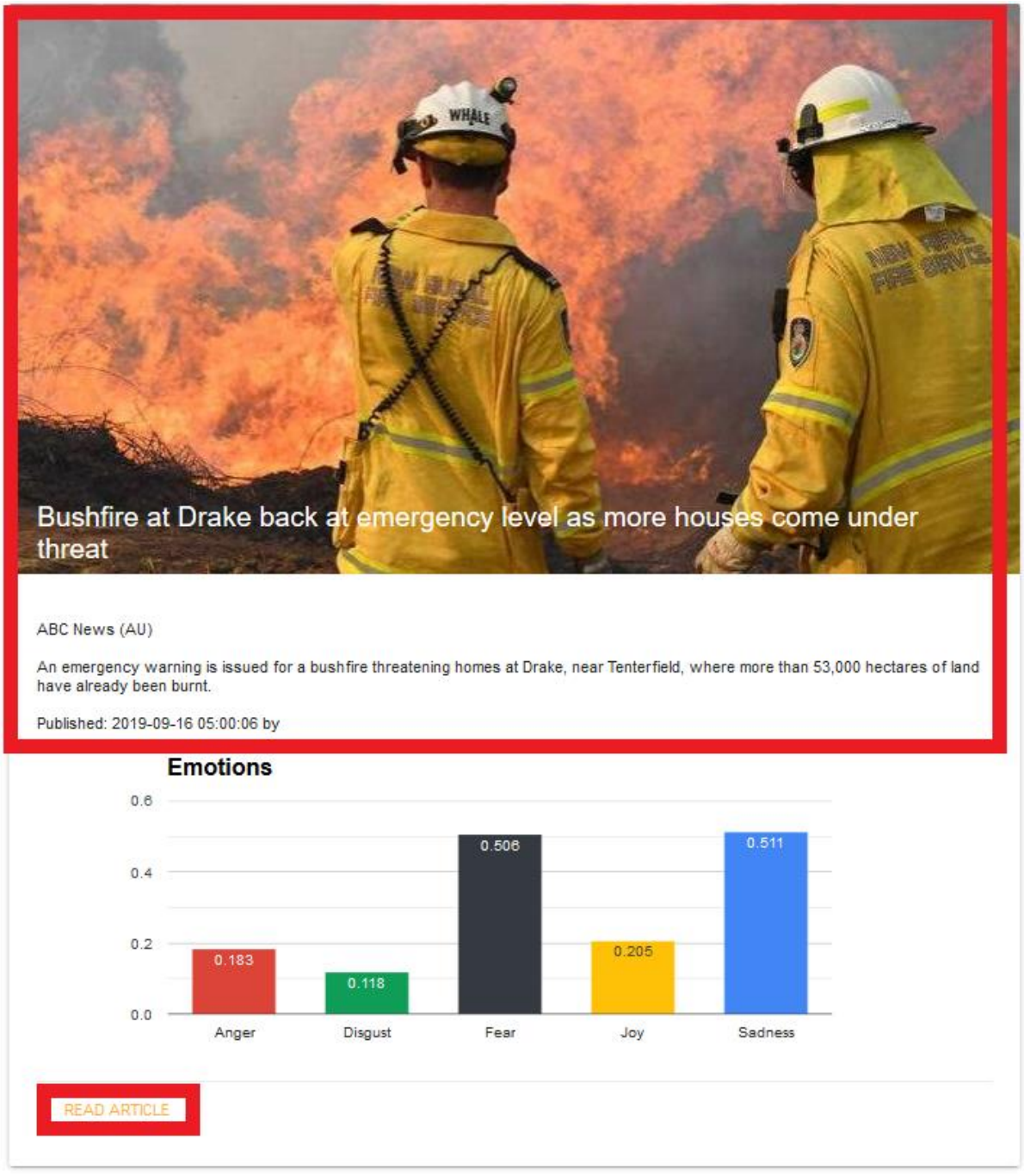


READ ARTICLE

Use Case C

As a newsreader, I want to search for articles and see a listed view of the articles returned including the image, headline, source, and description so I can get an insight on the story.

When the user searches for a topic, the page returns a list of articles. Each article displayed as a card. Each card contains an image, headline, description, source, author and publish date. The user also has the option to reading the article.



Use Case D

As a newsreader, I want to identify the emotion of each article by its colour so I can easily identify the emotion and discover the reasoning.

A graph is displayed in each of the article displayed to the user. Each emotion had a colour associated with it.

CBS News

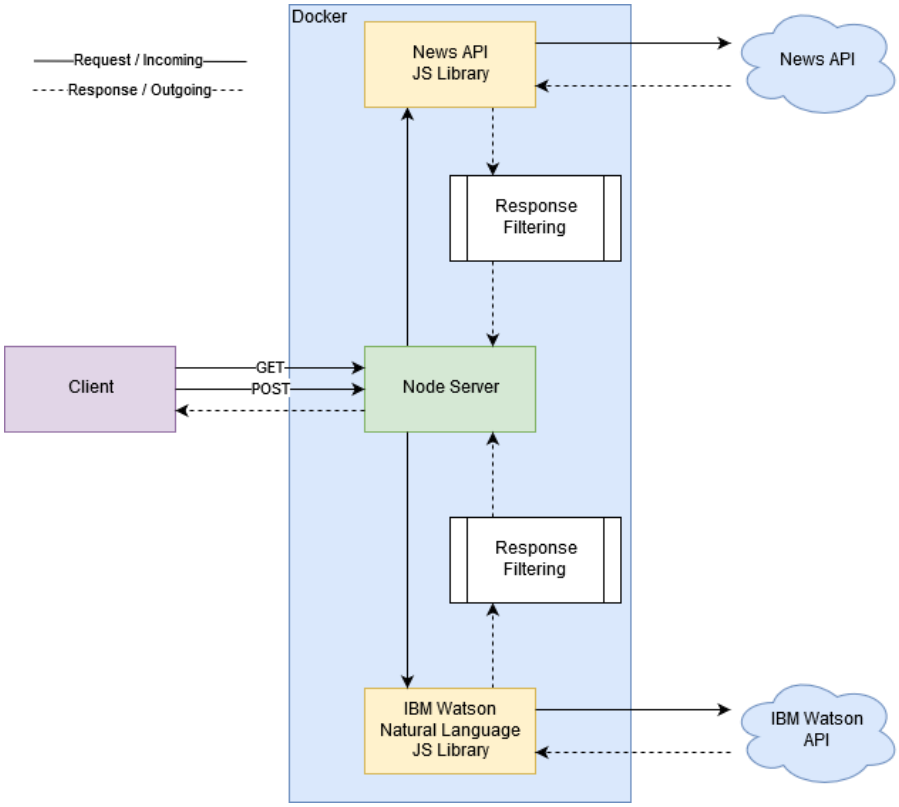
The Trump administration blames Iran for the strike that damaged one of Saudi Arabia's most important oil export sites. Also, more than 48,000 auto workers walk out after contract talks break down with the largest U.S. automaker. All that and all that matters...

Published: 2019-09-16 11:18:43 by CBS News

READ ARTICLE

Technical Breakdown

Reporial was designed to keep client-side processing to a minimum. This was done by ensuring all most of the processing was conducted on the server-side, where responses were filtered and optimised before sending the final page rendering to the client.



This web application was developed in stages to ensure minimal processing on the client-side and data was presented to the client in the expected format. The first stage involved retrieving the data from the News API endpoint and rendering the results on the client-side. The second stage consisted of analysing the content of each retrieved article. The final stage involved displaying the results on the client-side with Google's graph visualisation.

Through developing the web application in stages, I was able to minimise development time and priorities key components. I also noticed I was able to abandon any unnecessary functionalities. The application was developed using the Node Express Generator which created the skeleton for the application. With this method, I was also able to accelerate development.

Client side

The client can initiate two types of requests., a GET and a POST. The first request made from the user is the GET request when landing on the home page. When this is initiated, a GET request is sent to the server and returns the index page with search bar and logo. When the user initiates a search, the form body sends a POST request to the server for processing.

The server handles the query sent from the form, processes the API calls with the query, then displays the results on the page. Each of the articles is displayed in individual cards containing information about the article and the graphs representing the emotional analysis

Server side

The Node Server Express manages the incoming client request and outgoing responses. When it receives a GET request from the client, the server renders and returns the home page with a search bar and a logo. There is no heavy processing in this request because the client hasn't searched for anything and is only requesting the home page.

When the server receives a POST request, it calls the News API endpoint with the body query obtained from the search form. This API endpoint returns a JSON response with an array of objects. Each object representing an article. The server filters the response and stores them in a new array of objects with only the relevant information for display. This information is then passed to IBM's Natural Processing Language API where the emotional analytic scores for each article. This information is then rendered and sent to the client for display.

Response Filtering

The News API and IBM Natural Language processing JS libraries handle the calls to the respective API's and sends the response back to the server. Both return the data in JSON format. When received, this information is further refined to minimise the payload being sent back to the client. This also helps the server to handle the information more simply. Each of the articles returned from News API is created into an object, where only the necessary information to be displayed on the client-side is stored.

These objects are then stored in an array and passed to another function that calls the IBM Natural Language Processing API. The URL for each article is passed to the function. The response from the API is stored in an array.

At any point these News API causes an error, the server will render the error page and send to the client with information about the error. If an article doesn't have an image, it gets replaced with a regular news image. If the IBM API causes an error in analysing the content, the data is substituted with a value full of zeros for each emotion.

When all response has been refined, the data is rendered on the index page and displayed to the client.

Docker

Docker was used to containerise the Reporial application and include the node server, JavaScript libraries, and front-end PUG templates. The Dockerfile below was used to build an image of the web application. The official Docker Node image, Node 10, was derived in the build of the docker image.

```
FROM node:10
MAINTAINER John Santias
Add . /App
EXPOSE 3000
WORKDIR /App/reporial
ENV NEWS_KEY bf12e[redacted]
ENV IBM_KEY c_rd[redacted]Y
RUN npm install
CMD npm start
```

The first line of this Dockerfile sets the node image, a dependency required to run the Reporial application. The maintainer is listed as me. The next line adds a new folder /App, then exposes port 3000 for the Node server. The working directory is then set to /App/reporial which allows us to change the environment to production. Environment variables are set to load the API keys for the API libraries and endpoint. The second last line installs all the node packages dependent on running the application. The last line starts the application.

PUG

PUG is a frontend templating engine used in Reporial. It is used on the server-side to render pug files. PUG is a clean, whitespace sensitive syntax for writing HTML. This allowed me to shorten my code and spend less time on developing front-end. The pug files were utilised in this project. The first is *error.pug* which rendered and displayed to the client in any case an error occurred in processing a request. The second file *layout.pug* holds the search bar, header logo and the container that is displayed on all pages displayed to the client. The third file *index.pug* is an extension of the *layout.pug* file which is used to display all the results searched by the user.

Difficulties

Storing the response

I noticed that API calls should be done in a Promise followed by a Response and Reject. When running the application, the variable does not immediately store the response. The program executes each line in the program and can make the variable 'undefined' when the server hasn't received the API response.

To combat this, I made my function as an asynchronous which makes the app wait for the passed Promise's resolution, then resumes the async function's execution and evaluates the resolved value. By using async functions and Promises, I was able to pass data to IBM API without an undefined error.

Similarly, I had the trouble of running IBM analysis on all the articles. Since the API only accepts one URL at a time, I had to use a recursive function to run the analysis for each article. Placing an *await Promise.all()* made the application wait for all Promise resolution and allowing me to display the correct results on the page.

Analysing article content

The News API returns an array of articles including the publish date, source, etc. However, the content is shortened because to the 100 character limit in a free API plan. Since this can affect the emotion analysis result, we can pass the URL of the article instead.

Some of the news sources deny programs from accessing, manipulating and extracting data. This causes an error when trying to analyse an article with the IBM API. This was proven by using the online IBM demo analysing each article obtained. There were responses such as 'Access Denied, source denied request'.

I tried using multiple NPM packages that can extract the article with a URL. However, with my findings, it was impossible to extract data from some of the news sites. I decided to replace the data with the same JSON object with zero values.

Extensions

Related YouTube Videos

It was initially proposed that users would be able to watch YouTube videos related to the news article selected. This functionality was abandoned due to time constraints and the quota limit for using YouTube's Data API. Including this feature would be useful for people who would prefer to watch a video. This could provide users with a better vision of the story.

Advanced Search

It was a consideration to further develop a more complex search engine, allowing the user to choose news sources to obtain articles from, or how many articles to display on the page, different types of graphs, sorting, etc. However, this was not implemented due to time constraints. Including this functionality to the web application would provide users with greater control of the information they search for and how it's displayed.

Testing

Error handling was managed by Promises inside asynchronous functions. Specifically, the functions for retrieving articles and emotional analysis were created as asynchronous and use a Promise to resolve or reject the outcome (Snippet A). Reject when API call causes an error or resolve when no errors found during the call. The response of the function is then handled by the Node Express router to display the outcome (error or not) with the data rendered on the PUG template engine (Snippet B). Errors occurred in the application is displayed on the page to give a user an explanation on the occurrence.

Snippet A

```
async function getAllArticles(query) {
    return new Promise((resolve, reject) => {
        newsapi.v2.everything({
            q: query,
            language: 'en',
            sortBy: 'publishedAt',
            pageSize: 25,
        }).then(response => {

            // articles found for the search query
            if (response.totalResults !== 0) {
                resolve(response);
            }

            // No articles found
            else if (response.articles.length === 0) {
                reject("We could not find any articles for: " + query);
            }

            // Promise rejected
            else {
                reject(response);
            }
        });
    });
}
```

Snippet B

```
router.post('/', function(req, res, next) {

  // Store search query
  let query = req.body.query;

  // Set the title
  var title = "Results for '" + query + "'";

  // Obtain all the articles from NewsAPI
  processNews.getAllArticles(query).then((response) => {

    // Obtain certain values from the NewsAPI response
    let articles = manageArticles.buildArticles(response);

    // Run emotional analysis on each article
    processAnalysis.buildAnalytics(articles).then((analyticResults) => {

      //reder all results on index.pug
      res.render('index', {title: title, reports: articles, analytics: analyticResults});
    })

    // Render error page when analysis is processed incorrectly
    .catch((err) => {
      res.render('error', { message: "An Error Occured Analysing Content", error: err });
    });
  })

  // Render error page when NewsAPI is processed incorrectly
  .catch((err) => {
    res.render('error', { message: "An Error Occured Getting News", error: err });
  });
});
```

Test Cases

TASK	EXPECTED OUTCOME	RESULT	SCREENSHOT/S (APPENDIX B)
Retrieve Home	Display the logo	PASS	1.A
	Display the search box	PASS	1.B
Search all articles	Queried text at top of the results	PASS	2.A
	Article Image displays	PASS	2.B
	Article Headline displays	PASS	2.C
	Empty image replaced with regular news image	PASS	5
	Link to news article is correct	PASS	2.F
	Article description is displayed	PASS	2.D
	News source, published date and author is displayed	PASS	2.E
	Link to news article opens in a new tab	PASS	4
	Denied article access for analysis displays zero valued graph	PASS	3
Handle News API response error	Application continues running, display error	PASS	6
Handle IBM Natural Language Understanding Error	Application continues running, display error	PASS	7

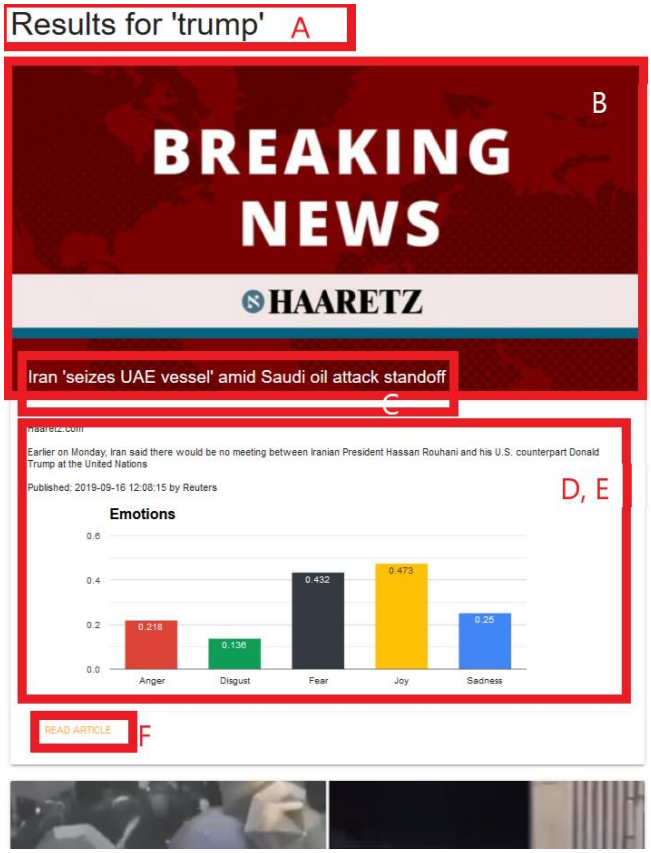
Appendix A – User Guide

1. Users enters a topic
2. User clicks the search button or presses enter to start the search
3. Results are displayed in individual cards. Individual results are displayed below and the user can scroll down to view
4. Emotional scores are displayed under each article
5. Clicking on the 'Read Article' opens the article in a new tab

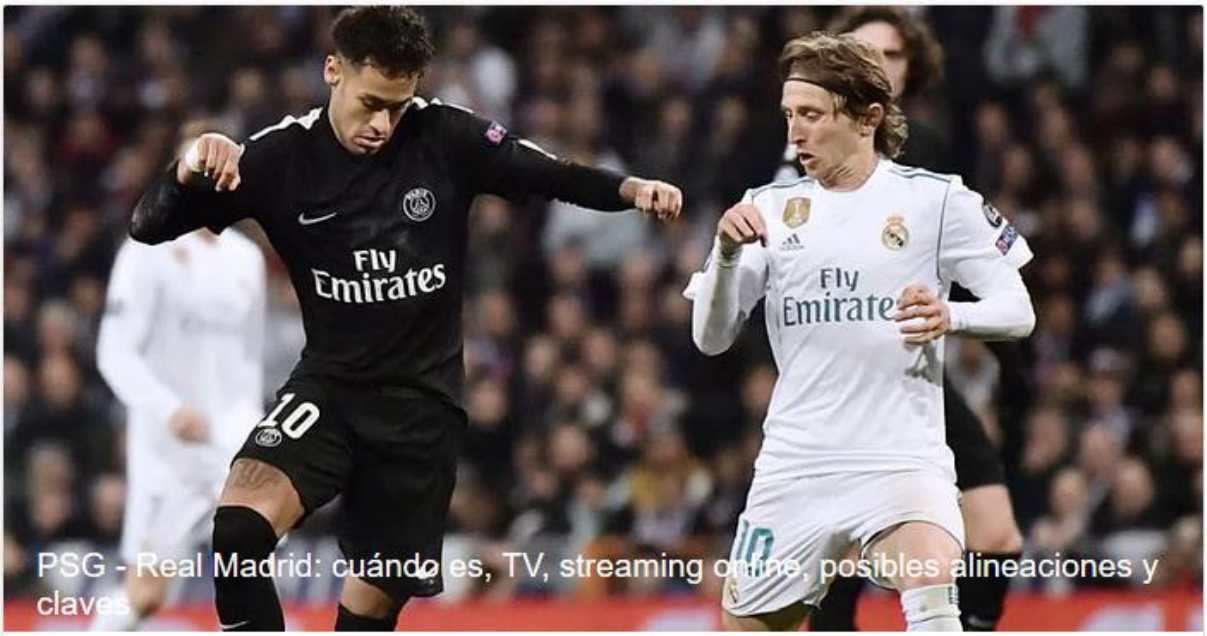
Appendix B – Test Case Screenshots

1.A – Display the logo
1.B – Display the search box

2.A – Queried text at top of the results
2.B – Article Image displays
2.C – Article Headline displays
2.D – Article Description is displayed
2.E – News source, published date and author is displayed
2.F – Link to news article is correct



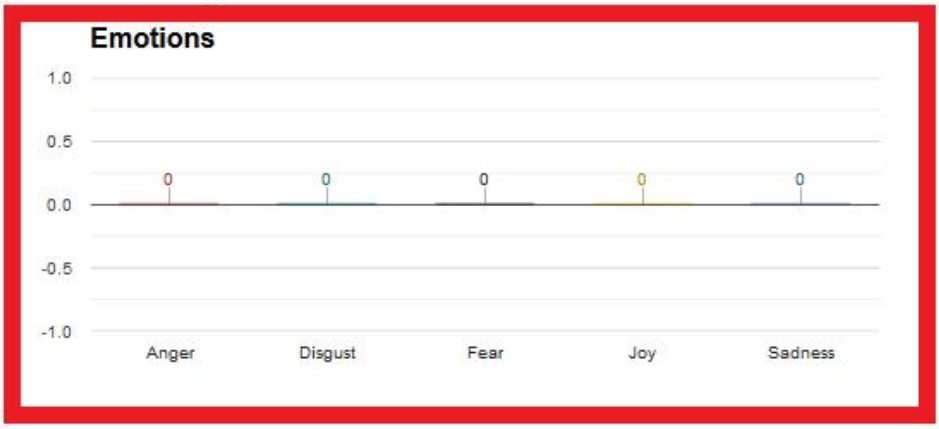
3. Denied article access for analysis displays zero valued graph



Si.com

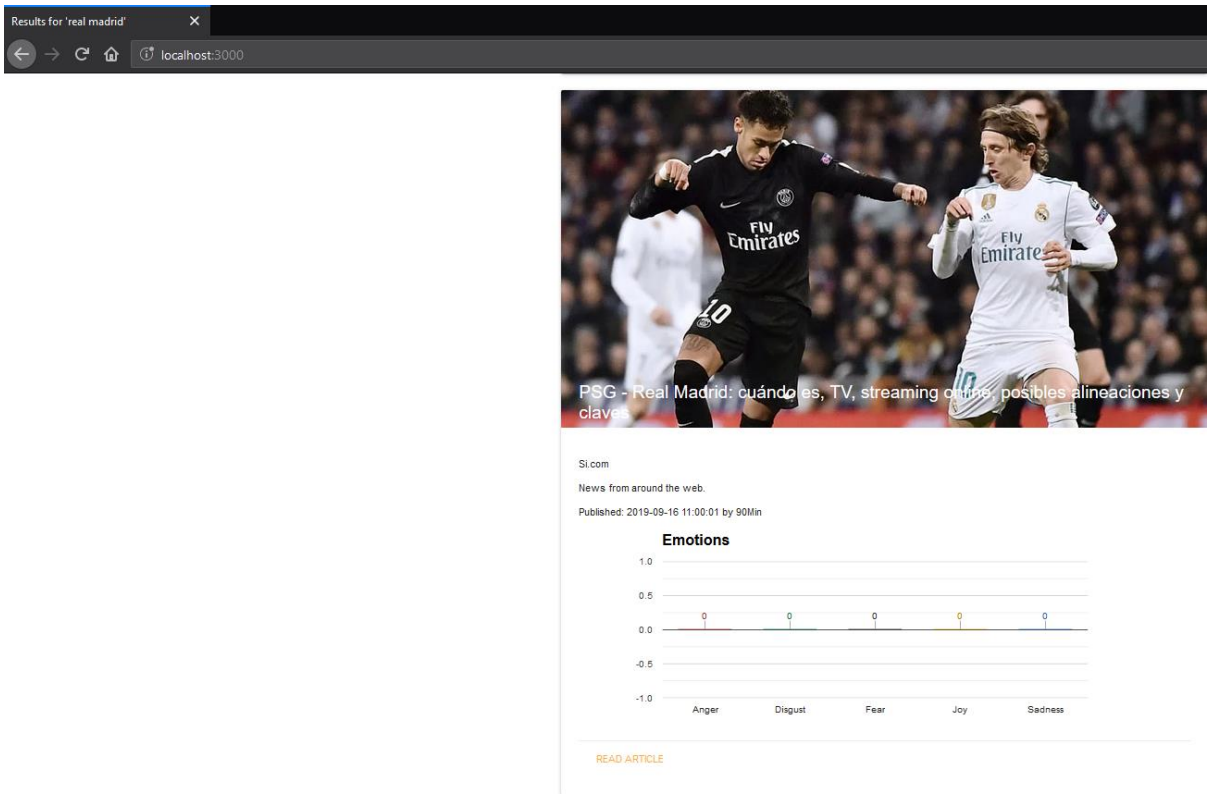
News from around the web.

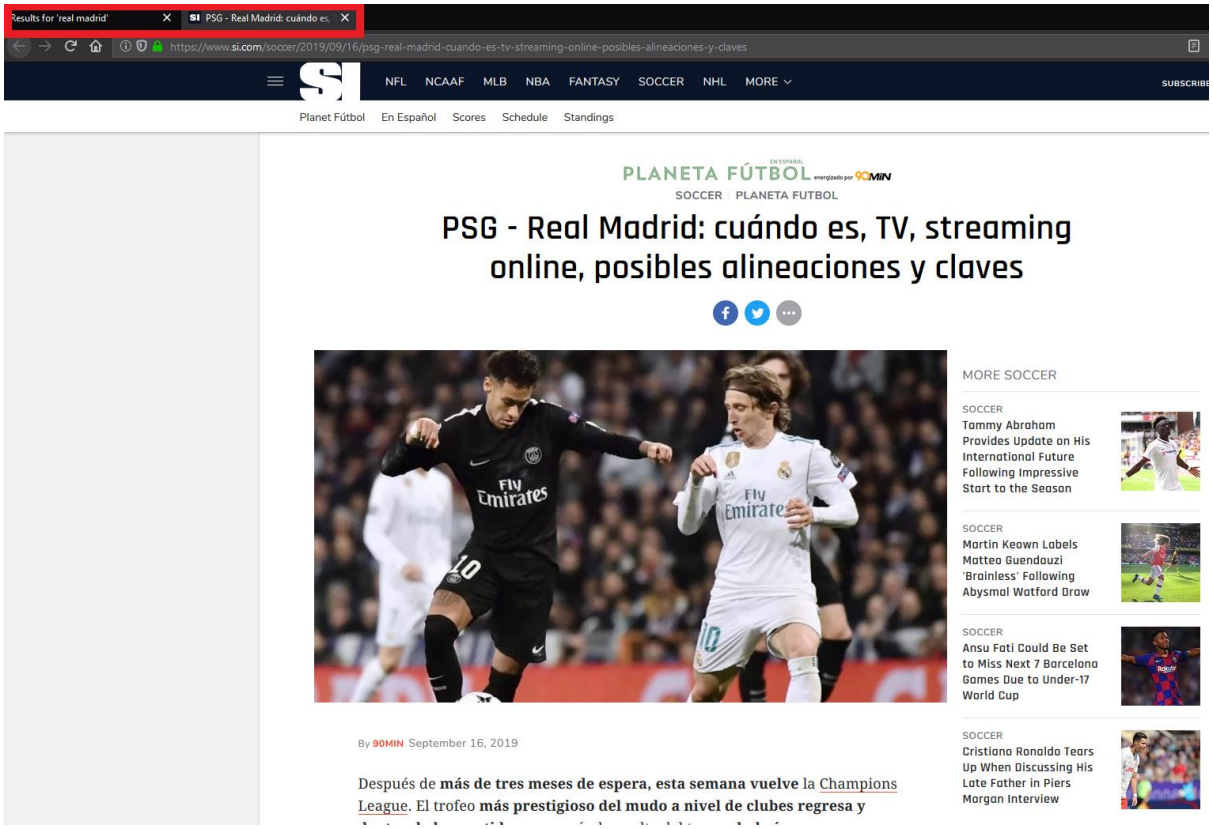
Published: 2019-09-16 11:00:01 by 90Min



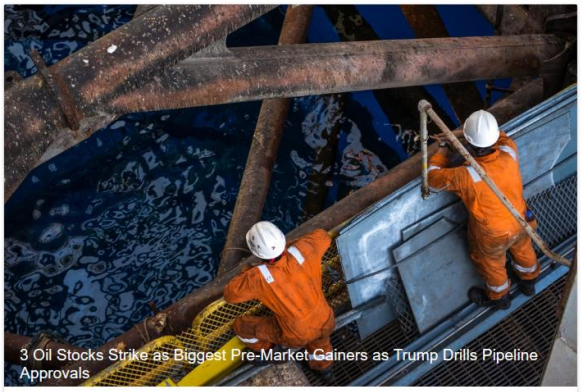
READ ARTICLE

4. Link to news article opens in a new tab





5. Empty image replaced with regular news image



Crypto Coins News

The warning by President Donald Trump that the U.S. is "locked and loaded" ready to attack Iran over suspicion that it was behind the drone attacks on refineries in Saudi Arabia not only sent crude oil prices surging but also saw upstream stocks climbing. In ...

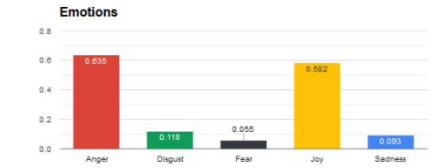
Published: 2019-09-16 12:13:22 by Mark Emem

READ ARTICLE

Pjmedia.com

IF YOU'RE NOT FAMILIAR WITH LAWFARE: Sundance at the Last Refuge explains why you should be if you want to understand some key characteristics of the anti-Trump jihad, including why they just can't give up on Kavanaugh's destruction.

Published: 2019-09-16 12:15:44 by Mark Tapscott

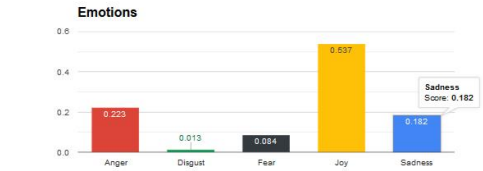


READ ARTICLE

Slashdot.org

Strong U.S. Dollar Prompts Speculation of Trump InterventionThe Wall Street JournalThe Trump administration has considered whether to use currency intervention as a weapon in its trade war, which has raised a host of questions among .....

Published: 2019-09-16 12:13:00 by feedfeeder

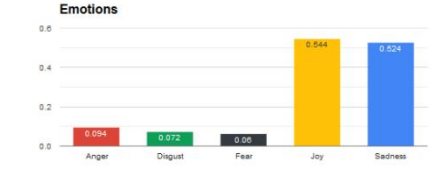


READ ARTICLE

Slashdot.org

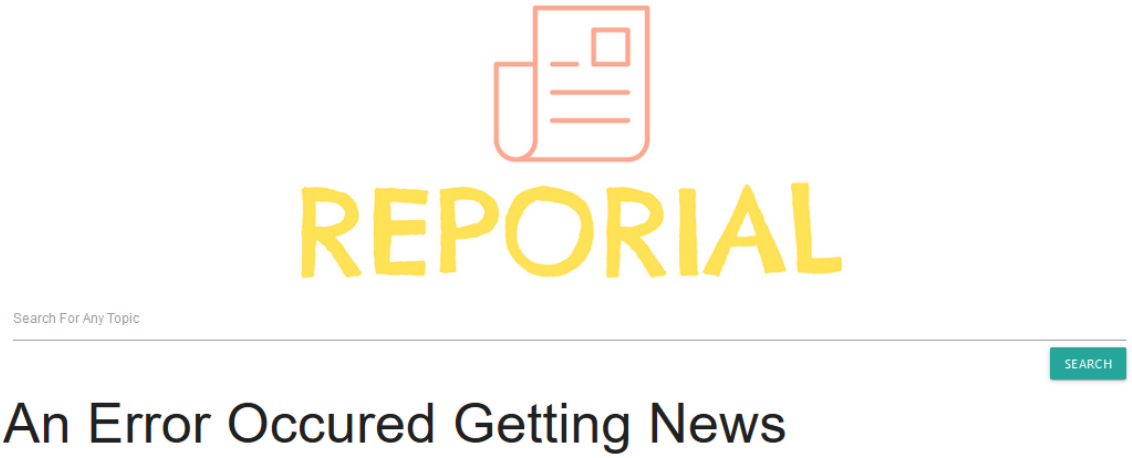
SB1: Lawmaker defies Newsom to guard California against Trump policyLos Angeles Times California state lawmakers pass bill requiring public universities to offer abortion pill on campus | TheHillThe Hill View full coverage on Google News ...

Published: 2019-09-16 12:13:42 by feedfeeder



READ ARTICLE

6. Application continues running, display error



7. Application continues running, display error