CLOUD COMPUTING MASHUP

News-Stock trend analysis platform

Michael Leontieff-Smith
N9455396

Table of Contents

Introduction

The purpose of my News-Stock trend analysis platform mashup is to provide a web solution which will allow users to search for (by providing a name and symbol) and monitor the performance of publicly traded companies that they are interested in with respect to said company's portrayal in the media. Given the reactive nature of stocks and their fluctuations in value, the site will aim to expose a key variable in company performance – the positivity or negativity of trending news articles that directly criticize or praise a company's motives, representatives or business dealings.

This will be achieved by contrasting company stock prices over a historical time period against the positive or negative sentiments of headlining news articles published online over the same time period. In theory, if a company is barraged by negative press, the negativity will likely translate into a discernible drop in stock prices that simple graphing tools can convey. Therefore, the solution will provide the ability for users to enter a company and have all trending articles pertaining to the company analysed and the results graphed. Alongside this graph will be two others, one which will present the stock prices over the same time period contrasted against the aforementioned sentiment data and another which outlines the average sentiment for each news outlet who has had their news articles processed.

The site will expand upon this by caching news article sentiments allowing for a historical record to grow, which will add to the extensive stock and news article information that will be provided.

Service API's Utilised

A multitude of services and user-facing frameworks culminate to deliver this experience, of which are listed below.

Service 1: News API

https://newsapi.org/

The News API service allows for news articles to be queried for from a wide array of news outlets. The callable endpoints allow for consumers to target trending news articles as opposed to all articles as well as being able to define a string query further refine the search.

This service will be the source of news articles for the application – the primary data that is processing in the application. All other services are employed to either process or store this data in some way.

Service 2: Google Natural Language API

https://cloud.google.com/natural-language/

A natural language processing service whose *processSentiment* endpoint affords consumers the ability to process input-text and assign it a sentiment value between -1 and 1.

The App utilises this service to analyse the sentiments for each article the News API response provides pertaining to a company. This result is then associated with the original article and used as the first set of data for comparison.

Service 3: Alpha Advantage

https://www.alphavantage.co/

A service which provides stock information based upon a provided trading symbol e.g. TSLA. Lots of control is given in the increments (days, weeks) and summarising of data.

This service will provide the second set of data, pairing with the sentiment output to provide the experience. This data will undergo various transforms to allow it to be compared to sentiments within the range -1<=y<=1, as stock prices cannot be negative, nor will they likely be between 0 and 1.

Service 4: Microsoft Translation API

https://www.microsoft.com/en-us/translator/business/translator-api/

The translation API by Microsoft provides the ability to auto-detect and translate input text to a defined language. Microsoft's service was chosen as a Google service is already employed and as such would infringe upon the already limited API quota.

This service is being used as Google's natural language API only supports a limited number of languages, and the news API returns articles in varied languages, so this service will be used to normalise the data before being analysed. This allows for a more broader view of a company's perception to be analysed.

Service 5: MongoLab

https://mlab.com/

Given the costs to all of these services, and the desire to accumulate analysis results, MongoLab will be utilised to cache translated articles and sentiment calculations to build up a historical record of perceptions. Whenever a user makes a request, both articles from the accumulating MongoLab cache and the News API are returned to bolster the richness of the application experience.

Client-side Frameworks

Chart.js

Chart.js is a client-side charting library which will present the data to the user. The final stage in the processing of data will be to align it with what this library expects.

Other Technologies Utilised

Pug Templating Engine

HTML pre-processor used to split up the pre-processed HTML into partials to drive reuse and ease of understanding.

JQuery

Used to query the Application API's and perform basic DOM operations

Typescript

A JS superset which forms the basis for all server and client side code. The application uses the type system extensibly to create more maintainable, compile-time-error-checking code. The compilation settings for this Application are configured to target ES2015.

Nodemon

Node monitor that's paired with the Typescript Compiler to drive efficient development and debugging through auto-compilation and serving of code.

SASS

CSS pre-processor to allow cleaner style rules.

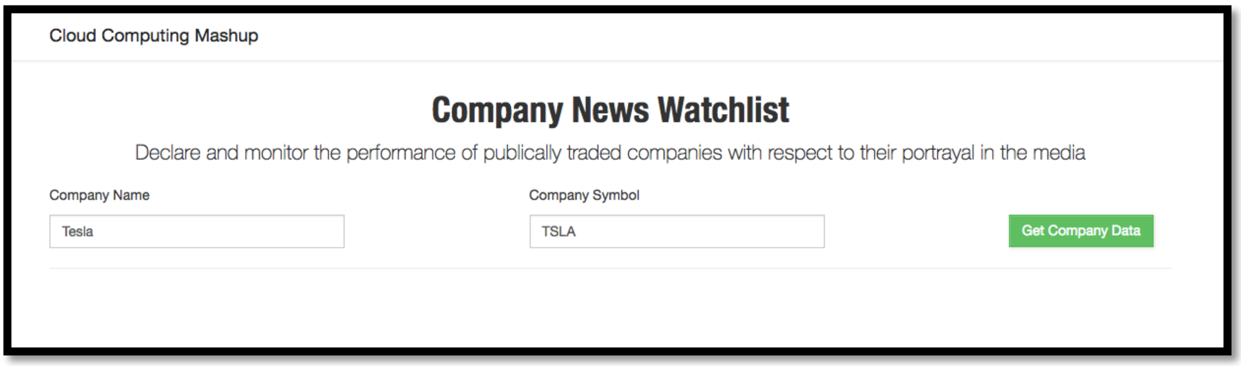
Use Cases

Use Case A [Uses Services #1, #2, #4, #5]

As a hobbyist Investor looking to invest in the automotive sector, I would like to be able **to specify publicly traded companies** that interest me and have the **general reflection of the Company** (positive or negative) by news outlets **conveyed to me** so that I can make a quick judgement on viability before committing.

Step 1: Specify publicly traded company

To achieve this, the user shall navigate to the home page of the app where they will be presented with a form that they may enter their query as shown.



Cloud Computing Mashup

Company News Watchlist

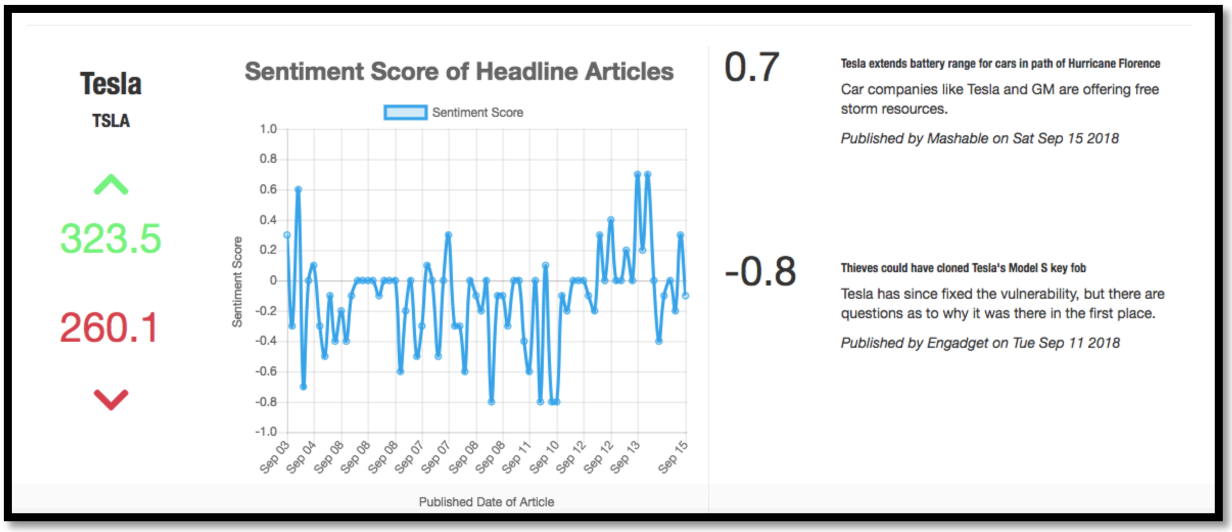
Declare and monitor the performance of publically traded companies with respect to their portrayal in the media

Company Name	Company Symbol	
Tesla	TSLA	Get Company Data

Given the "popularity" regarding Tesla and its CEO, the user decides to target their query toward this company as a potential investment option.

Step 2: General reflection of company conveyed to me

The user shall then click "Get Company Data" to initiate the search, to which they will then be presented with the following output after the loading spinner regresses.



Tesla

TSLA

323.5

260.1

0.7

Tesla extends battery range for cars in path of Hurricane Florence

Car companies like Tesla and GM are offering free storm resources.

Published by Mashable on Sat Sep 15 2018

-0.8

Thieves could have cloned Tesla's Model S key fob

Tesla has since fixed the vulnerability, but there are questions as to why it was there in the first place.

Published by Engadget on Tue Sep 11 2018

The first row displayed to the user provides the general reflection data, where the user can view the following information and **make a quick judgement on viability.**

- Left: Highest and Lowest stock price between the range of the last twenty trading days. The User observes that the fluctuation space is quite large.

  NOTE: the time frame is highlighted in a following user story.

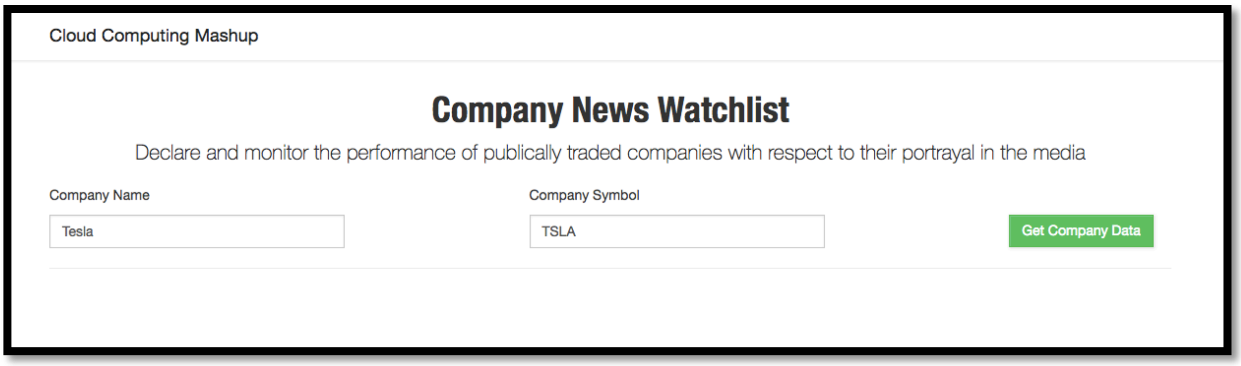
- Middle: Each article found matching the defined company, both new and historical and their associated sentiment/positivity rating. These results are graphed where the X axis represents each article ordered by published date and the Y axis being the associated sentiment value. With this response, the user observes that things aren't looking too good for Tesla, as the vast majority of sentiment ratings are below 0.

- Right: The most positive and most negative articles as determined by the application. These match up with the data in the sentiment table. The user observes this output, whereby they deduce that the most negative article alludes to thieves being able to circumvent the security of Tesla's and gain access, which is not very good from a stock holder perspective, which is a role this user is considering.

Step 3: Conclusion
Using this information, the User concludes that things are looking a little grim for Tesla at the moment, and instead decides to hold off until things stabilise.

Use Case B [Uses Services #1, #2, #3, #4, #5]
As a hobbyist Investor and market cynic, I would like this Application to be able to contrast a Company's performance in the media with its performance on the stock exchange, **displaying trends in both domains against each other in a normalised way** to allow connections and conclusions to be drawn as to the future performance of a company stock.

Step 1: Specify publicly traded company
To achieve this, the user shall navigate to the home page of the app where they will be presented with a form that they may enter their query as shown.


Cloud Computing Mashup

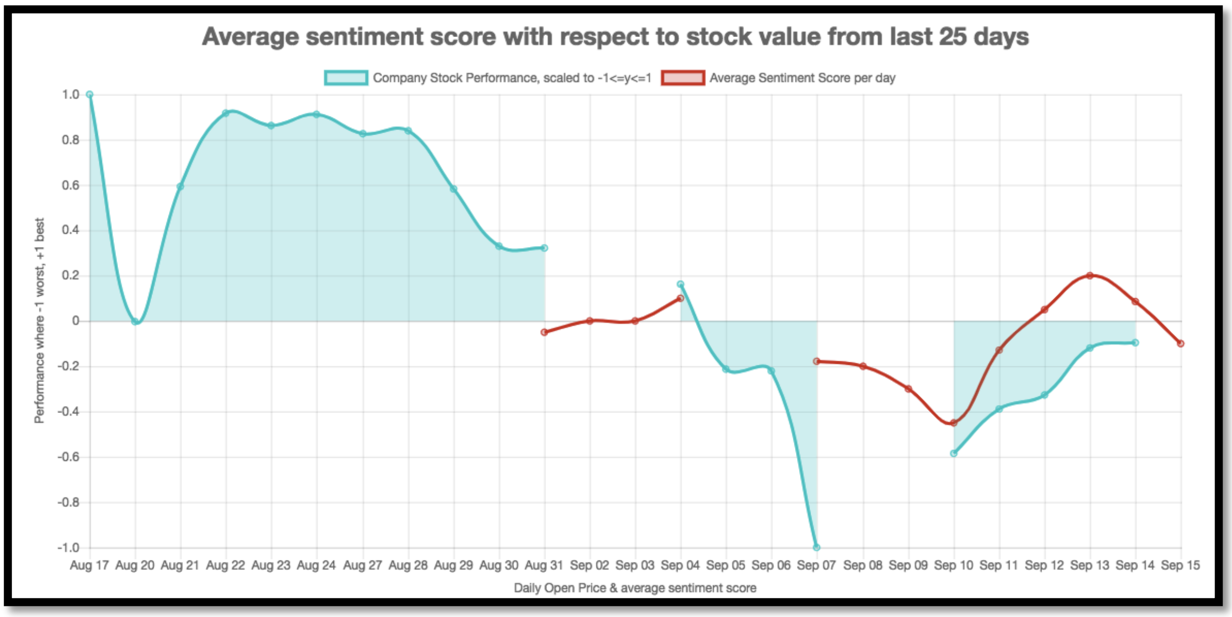
Company News Watchlist

Declare and monitor the performance of publically traded companies with respect to their portrayal in the media

Company Name	Company Symbol	
Tesla	TSLA	Get Company Data

Once again, Tesla's tribulations provide the perfect subject for analysis.

Step 2: Displaying trends in both domains
When the search is executed and returns, the second row of the analysis page contains the contrasting stock and sentiment values composited into a single graph.



Here the user is presented with both sets of data. The green represents the open price of the stock for that day, scaled to the range -1<=y<=1 translating to the highest stock price for the period being +1 and the lowest being -1. The red represents the average daily sentiment score for news articles featuring that company. The transformation performed on the stock data allows our User to directly compare the stock fluctuation with the sentiment values. With this, the user notes that the increase in average sentiment values of headlining articles between September 10 and September 14 directly correlates to the increase in share price over that exact period.

Step 3: Conclusion

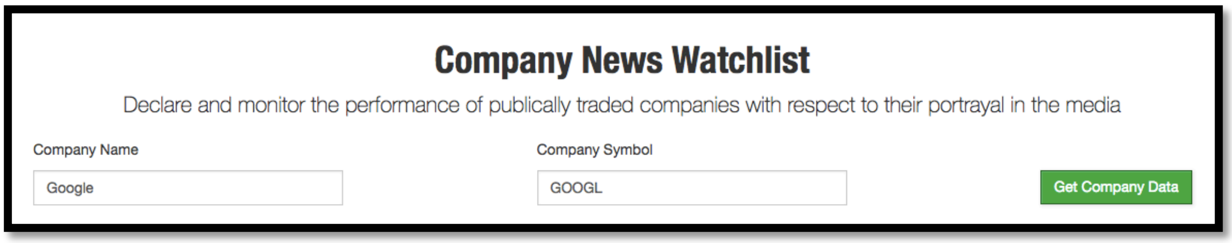
This information provides the User with insight as to the direction that the stock will continue in.

Use Case C [Uses Services #1, #2, #3, #4, #5]

As a Hobbyist Investor and Conspiracy Theorist, I would like to be able to view the totally **bias nature of a particular news outlet with respect to a particular company**, so that I may know the favouritism or lack thereof expressed by available establishments.

Step 1: Specify publicly traded company

To achieve this, the user shall navigate to the home page of the app where they will be presented with a form that they may enter their query as shown.



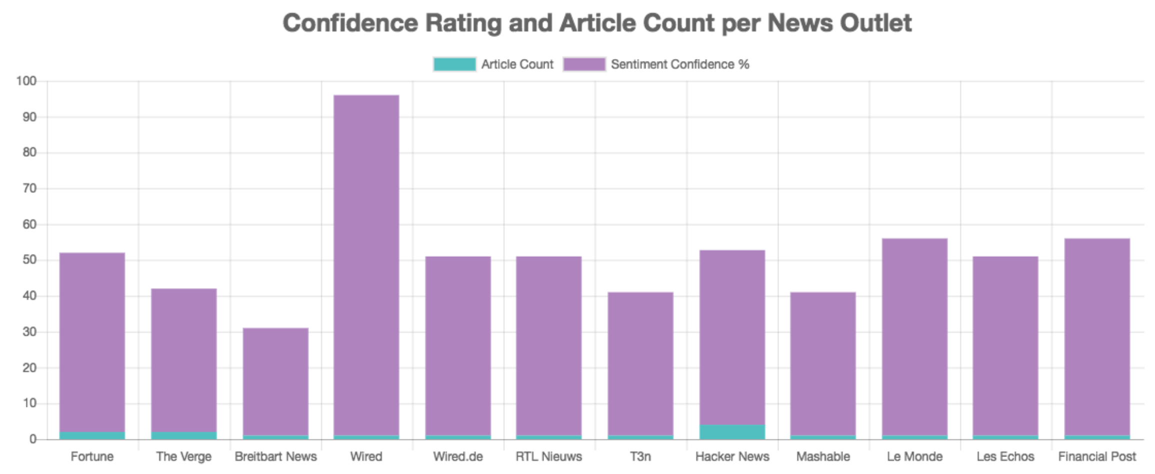
Company News Watchlist

Declare and monitor the performance of publically traded companies with respect to their portrayal in the media

Company Name	Company Symbol
Google	GOOGL

For this use case, the User searches for Google.

Step 2: Displaying outlet bias



The final row in the response presents the user with a breakdown of the number of articles processed per institution (aqua) and the average sentiment expressed as a confidence rating between 0% and 100% (purple).

Step 3: Conclusion

Using this information, our user concludes that Wired is an outlier due to their more-positive-than-others portrayal for tech companies like Google and decides that they shouldn't be trusted as a source for reliable information regarding Google.

Technical Summary

During the ideation phase, the primary objectives of the application architecture were outlined to provide a solid grounding for a robust, extendable application. These objectives are encapsulated in the following components:

Mock Data Source

External services are the cornerstone of this application and as such are required in all workflows of the application. These external services have an associated cost, and attempts were made to reduce this cost during development whilst also allowing these services to be replaced with mock data, allowing development to occur fluidly without a dependency on these services.

This is facilitated through the definition of an *IDataSource* interface, included as Appendix Item 1, whose concrete implementation is provided by both the *ServiceDataSource* and the *MockDataSource* classes, the latter of which provides mock-data-returning implementations of the costlier API calls. Once the Cache had been implemented in the app, implementation work shifted from the mock to the real one as it was no longer needed.

Method Implementations for the *ServiceDataSource* are what you would expect, with calls to the request library with relevant parameters. In the case of the MockDataSource, data is either pulled from saved JSON responses from real executions or generated on the fly, such as that included as Appendix Item 2.

Persistent Cache and Request Optimisation

Persistent Cache

It is obvious that an article needs only to be translated and sentiment-analysed once, as redundant processing would be detrimental to the API quotas and performance/scalability. Therefore, each article that is returned by the News API Service has an MD5 key generated from the pre-translated title and published date. Then, a lookup is performed against the cache to check if this article has been processed in the past. If so, it is fetched from the cache and not processed, otherwise the end-to-end processing is performed, and the output is saved to the cache upon completion so next time it may be requested more cheaply.

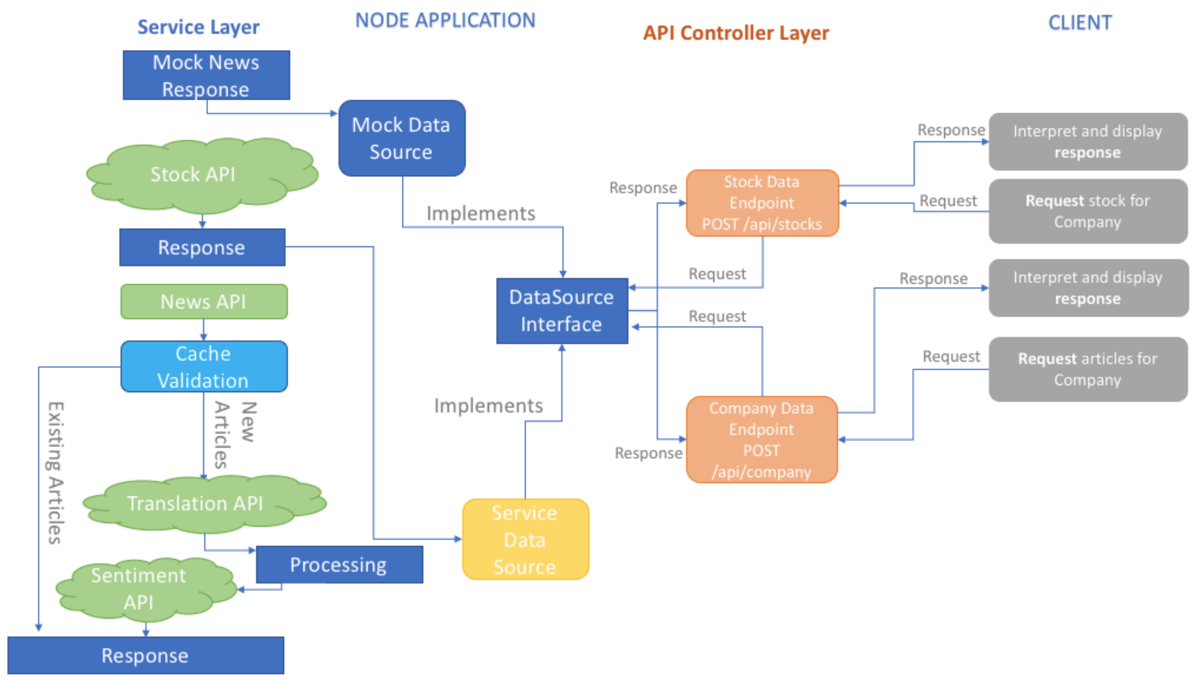
Request Optimisation

Whilst an API may not charge per request, instead charging by the request size, it is still beneficial to reduce the burden on bandwidth by optimising each request to be as effective as possible. This is particularity the case with the Translation API, which has the potential to process up to 25 records within a single request as opposed to just one. Therefore, as exemplified in Appendix Item 3, effort is taken to ensure that the most effective means of calling the third-party API's is taken by packaging up the payload in the most effective manner.

Overall Representation

These components described above, in addition to the surrounding architecture result in the following architecture diagram, which outlines the characteristics of the service.

It foregrounds the interplay between the cache and fresh requests whilst also providing a high level view of the mock data source and client interaction with the system.



Client/Server Role Breakdown

Summary of the above architecture diagram

Client

The data pertaining to a user's query is requested and responded to through the App's API endpoints which are called client-side by JQuery's AJAX methods. Both exposed endpoints are POST requests, which are performed during the following points in the use cases:

Validation

To ensure the User's input data is correct or supported, when the "Get Company Data" action is performed a request is made to the Application for the stock data "/api/stocks" pertaining to the given stock symbol. If stock data is not found, then an error is presented to the user and no further requests are made.

Post-Validation Fetching

If the first API call is a success, the "/api/company" endpoint is then called with the name of the company provided by the user. Again, if this query fails to find and articles, either new or from within the cache, then an error is returned.

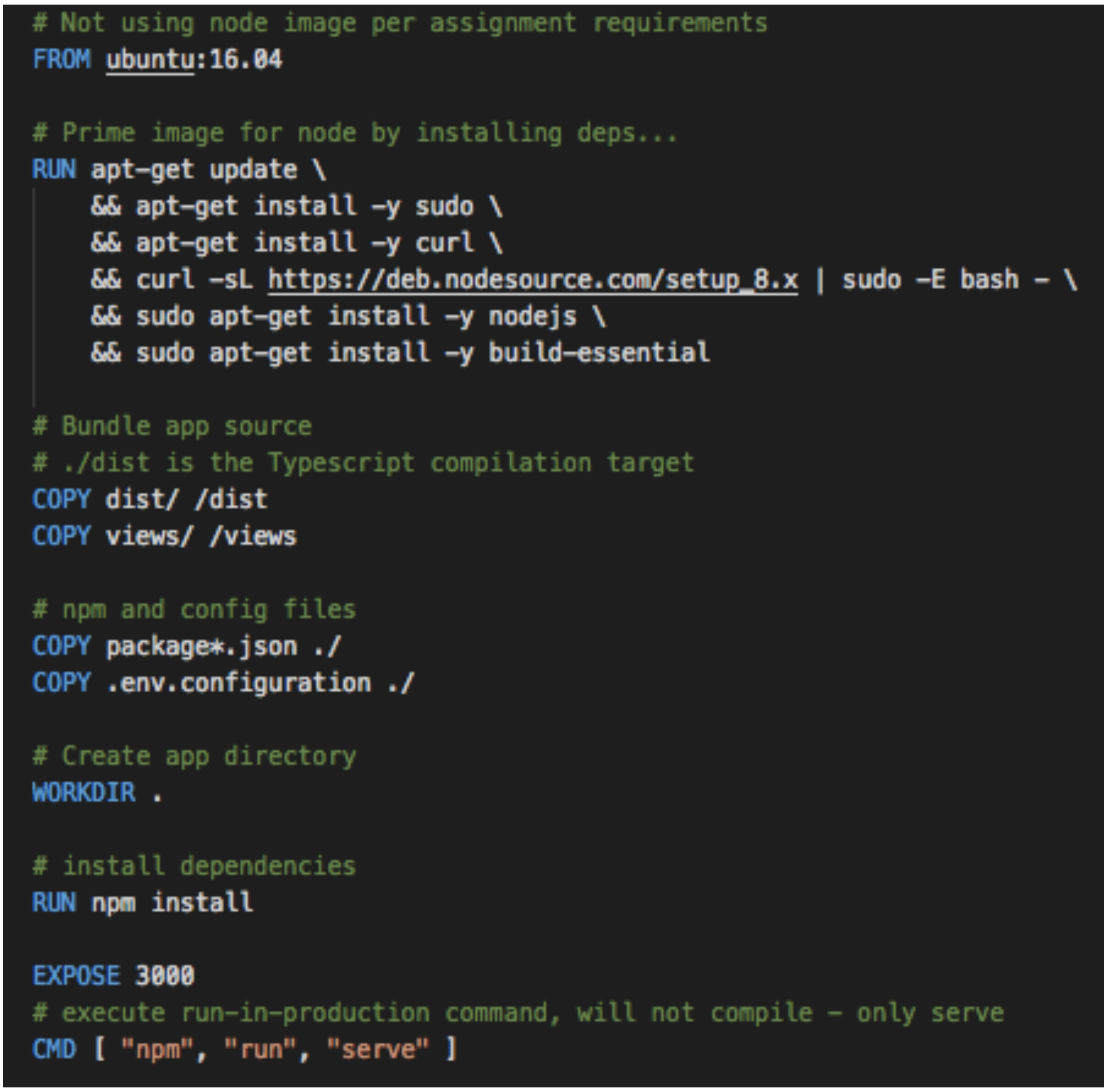
If both of these responses are successful, then the data from both is merged client-side for display by the charting library.

Server

When the server receives a POST to one of its exposed endpoints, is it routed to the relevant controller for the endpoint which in turn delegates the call to the relevant method(s) on the data source. The data source to be used is determined by the Data Manager class, which instantiates one or the other based upon a given parameter. The data source is an abstraction which performs all third-party API calls, and upon resolve returns data to the controller for cleansing and formatting before being returned to the client.

Docker

This application has been wholly containerised within docker to allow for seamless deployments with little to no environment configuration. Below is the associated Docker file for this project with comments that outline the intent for each step.



```
# Not using node image per assignment requirements
FROM ubuntu:16.04

# Prime image for node by installing deps...
RUN apt-get update \
    && apt-get install -y sudo \
    && apt-get install -y curl \
    && curl -sL https://deb.nodesource.com/setup_8.x | sudo -E bash - \
    && sudo apt-get install -y nodejs \
    && sudo apt-get install -y build-essential

# Bundle app source
# ./dist is the Typescript compilation target
COPY dist/ /dist
COPY views/ /views

# npm and config files
COPY package*.json ./
COPY .env.configuration ./

# Create app directory
WORKDIR .

# install dependencies
RUN npm install

EXPOSE 3000
# execute run-in-production command, will not compile - only serve
CMD [ "npm", "run", "serve" ]
```

Difficulties

Normalising data for Services

One of the issues that was quickly realised revolves around the interplay between endpoints. The News API is the primary source of information that is processed, and as such that data needs to be in a state that is consumable by the other services that perform the processing. The News API returns trending articles in multiple languages, with the content not being tagged with its associated language. This is an issue as the Google Natural Language API only supports a small subset of languages and feeding un-validated content to it could cause un-foreseen errors.

To rectify this, either the articles whose languages are not supported could be discarded from further processing (which would still require a service call to detect a language) or they could simply be translated into a language that is supported. The latter option is the action that was taken, adding another layer of overhead to the application.

API Quotas

The richness of the application is in part dictated by the amount of data that it can draw from, and in the case of this application, each record (news article) undergoes a translation and a sentiment analysis. Both of these services are rate-limited under free plans (former measured by characters translated and the latter by request). To alleviate this issue, a cache layer was added to the application whereby each article that is returned by the News API is validated by a unique key (hash of the title and publish date) against a MongoLab store. If the record exists, it is returned. If the record doesn't exist, then it is processed in its entirety then saved.

With this action, regardless of the number of requests by multiple users, an article is only processed once, allowing the cost to be one with respect to request counts.

Fault Tolerance and Testing

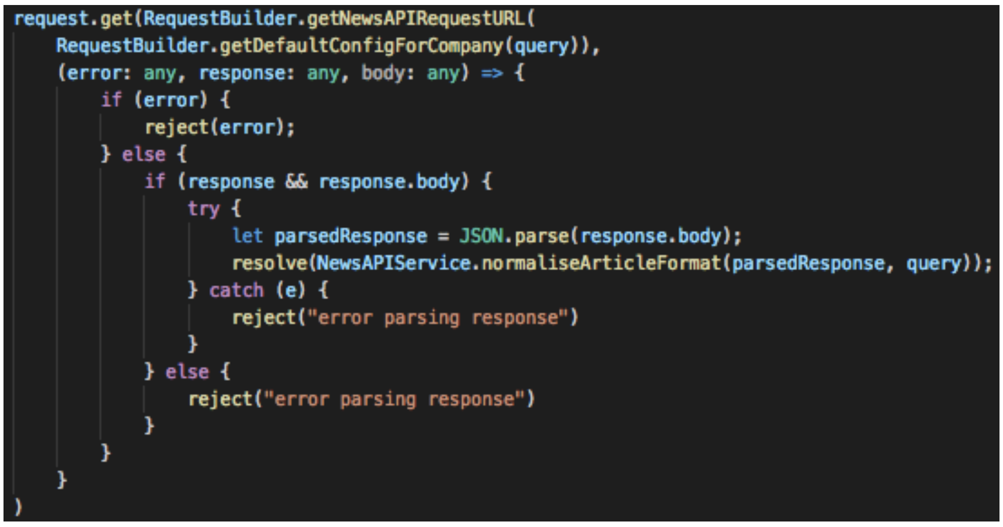
Targeted Platform

The application has been tested against Chrome, Firefox and Safari which all report identical functionality without variation.

Robustness

Given the extensive use of asynchronous operations and the dependence on external services, this Application employs rigorous checking of the responses and subsequent error handling through promise rejections, error call-backs and try-catch blocks. The checking of errors comes in the form of truthy checks on responses and their properties to ensure the data is in the expected form before proceeding.

All errors that are encountered by calling the service API's are caught and have a corresponding error string associated with them. These errors are expressed through promise rejects, and if these are not supported by the libraries being used (such as request) then the callback pattern is wrapped in a promise as shown below:



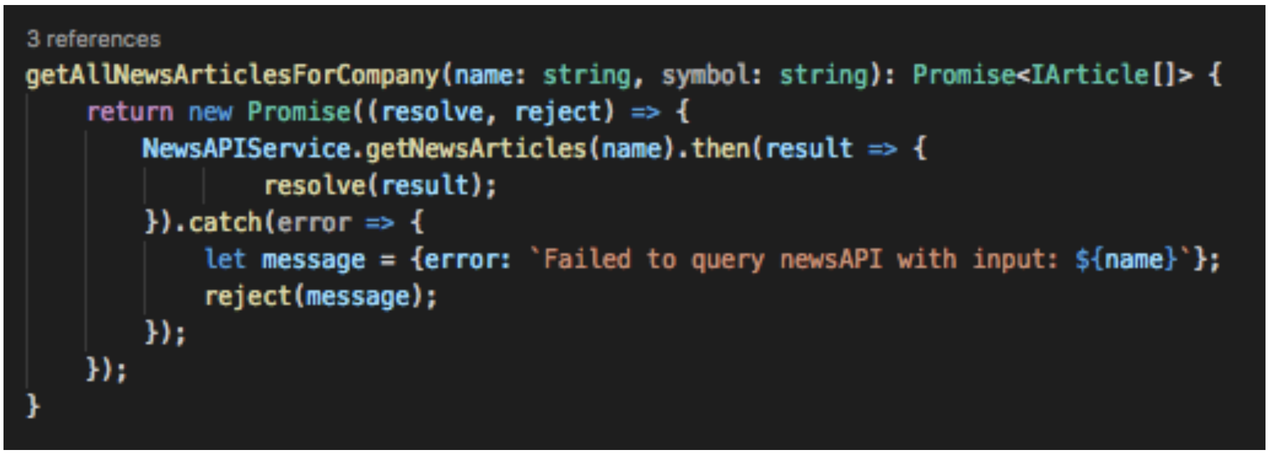
```
request.get(RequestBuilder.getNewsAPIRequestURL(
    RequestBuilder.getDefaultConfigForCompany(query)),
    (error: any, response: any, body: any) => {
        if (error) {
            reject(error);
        } else {
            if (response && response.body) {
                try {
                    let parsedResponse = JSON.parse(response.body);
                    resolve(NewsAPIService.normaliseArticleFormat(parsedResponse, query));
                } catch (e) {
                    reject("error parsing response")
                }
            } else {
                reject("error parsing response")
            }
        }
    }
)
```

These errors are then thrown upwards through the use of another rejection whereby they are caught by the API controller which packages up the error in the response and returns it to the client.

If an error is encountered, the client will display it to the user if it is relevant. All technical errors that have little user-facing meaning are generalised. Shown below as an example is the ordered propagation of an error due to an invalid input hitting the News API endpoint.

ServiceDataSource

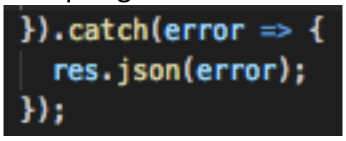
Performing the request and catching potential errors before propagating with a generalised error.



```
3 references
getAllNewsArticlesForCompany(name: string, symbol: string): Promise<IArticle[]> {
    return new Promise((resolve, reject) => {
        NewsAPIService.getNewsArticles(name).then(result => {
                resolve(result);
        }).catch(error => {
            let message = {error: `Failed to query newsAPI with input: ${name}`};
            reject(message);
        });
    });
}
```

API Controller

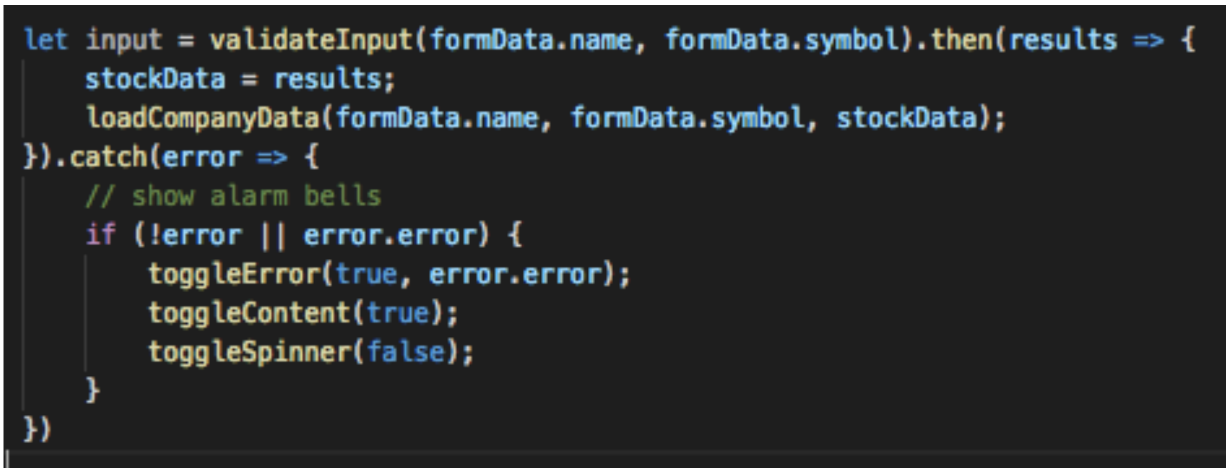
Accepting the error with the catch and returning as the response



```
}).catch(error => {
    res.json(error);
});
```

Client-Side JS

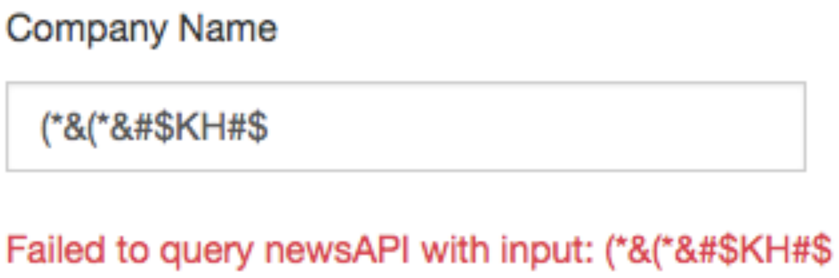
Detecting the error in the response and rejecting so that the error may be picked up by the validation code on the client.



```
let input = validateInput(formData.name, formData.symbol).then(results => {
    stockData = results;
    loadCompanyData(formData.name, formData.symbol, stockData);
}).catch(error => {
    // show alarm bells
    if (!error || error.error) {
        toggleError(true, error.error);
        toggleContent(true);
        toggleSpinner(false);
    }
})
```

Resulting User Feedback

NOTE: Normally this error would be caught by the client-side validation, however, this was disabled temporarily to illustrate the point.


Company Name

(*&(*&#$KH#$

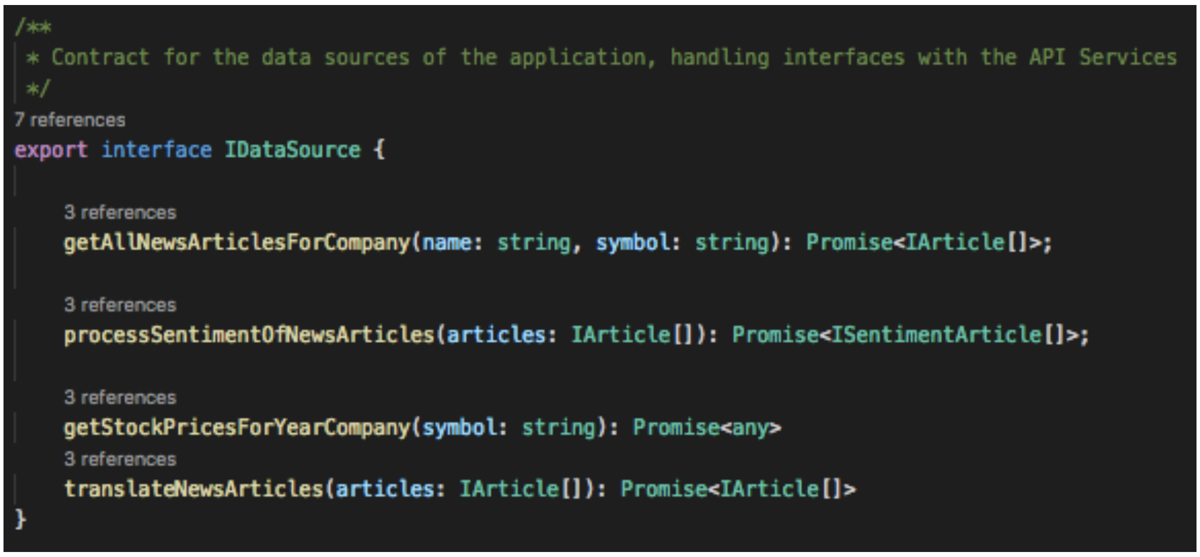
Failed to query newsAPI with input: (*&(*&#$KH#$

Test Cases

Test Case	Status
Validation and Retrieval From Cache [Appendix Item 4.A]	**PASS**
Successful Fetch of Articles [Appendix Item 4.B]	**PASS**
Translation of Article Contents [Appendix Item 4.C]	**PASS**
Article Sentiment Analysis [Appendix Item 4.D]	**PASS**
Sentiment Chart [Appendix Item 4.E]	**PASS**
Composition Chart [Appendix Item 4.F]	**PASS**
Company Bias Chart [Appendix Item 4.G]	**PASS**
Client-Side Search [Appendix Item 4.H]	**PASS**
Error Handling: Invalid Company Name [Appendix Item 4.I]	**PASS**
Error Handling: Invalid/Unknown Company Symbol [Appendix Item 4.J]	**PASS**

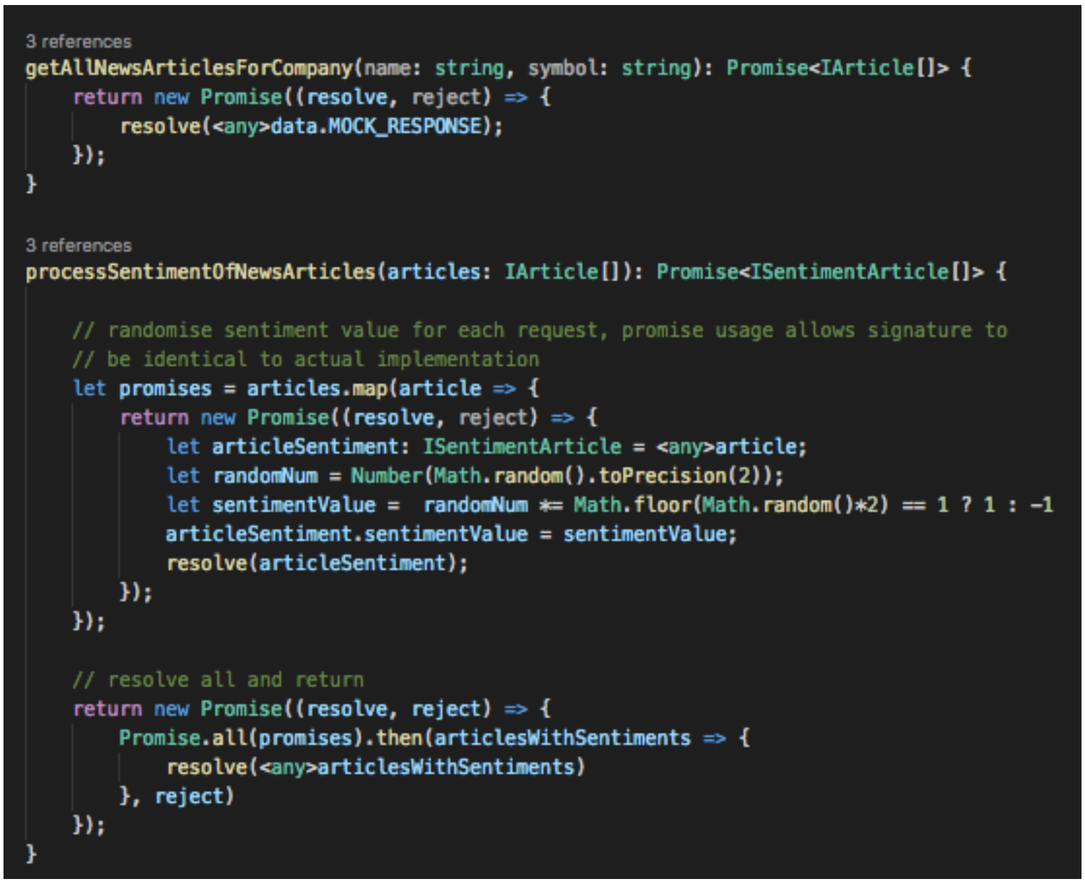
Appendix

Item 1: IDataSource Interface



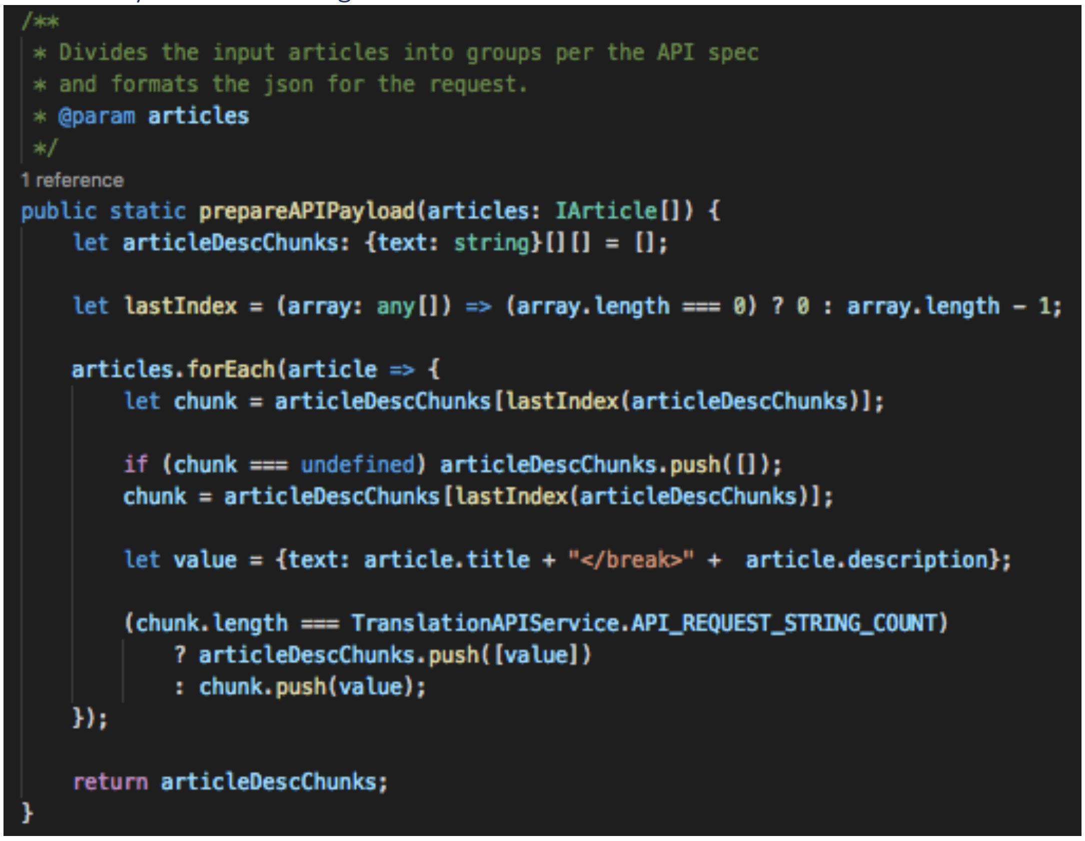
```
/**
 * Contract for the data sources of the application, handling interfaces with the API Services
 */
7 references
export interface IDataSource {

    3 references
    getAllNewsArticlesForCompany(name: string, symbol: string): Promise<IArticle[]>;

    3 references
    processSentimentOfNewsArticles(articles: IArticle[]): Promise<ISentimentArticle[]>;

    3 references
    getStockPricesForYearCompany(symbol: string): Promise<any>
    3 references
    translateNewsArticles(articles: IArticle[]): Promise<IArticle[]>
}
```

Item 2: Mock Data Source



```
3 references
getAllNewsArticlesForCompany(name: string, symbol: string): Promise<IArticle[]> {
    return new Promise((resolve, reject) => {
        resolve(<any>data.MOCK_RESPONSE);
    });
}

3 references
processSentimentOfNewsArticles(articles: IArticle[]): Promise<ISentimentArticle[]> {

    // randomise sentiment value for each request, promise usage allows signature to
    // be identical to actual implementation
    let promises = articles.map(article => {
        return new Promise((resolve, reject) => {
            let articleSentiment: ISentimentArticle = <any>article;
            let randomNum = Number(Math.random().toPrecision(2));
            let sentimentValue =  randomNum *= Math.floor(Math.random()*2) == 1 ? 1 : -1
            articleSentiment.sentimentValue = sentimentValue;
            resolve(articleSentiment);
        });
    });

    // resolve all and return
    return new Promise((resolve, reject) => {
        Promise.all(promises).then(articlesWithSentiments => {
            resolve(<any>articlesWithSentiments)
        }, reject)
    });
}
```

Item 3: Payload Formatting

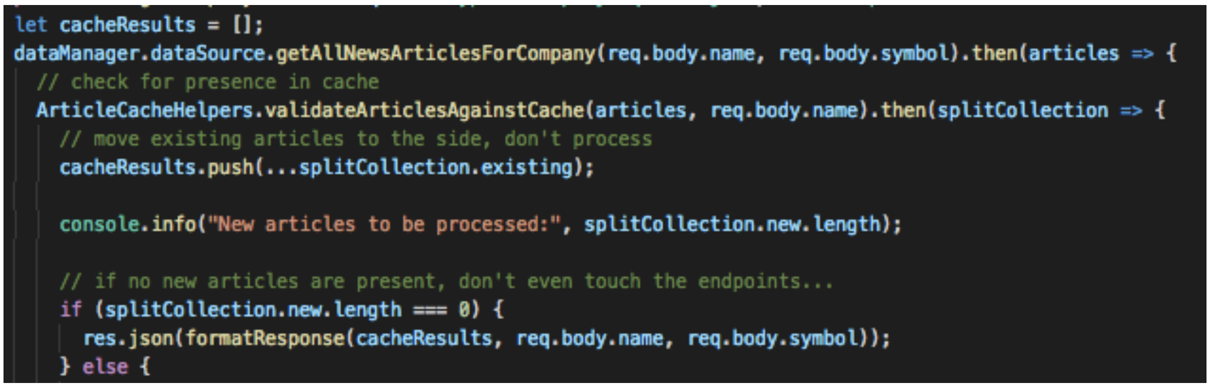


```
/**
 * Divides the input articles into groups per the API spec
 * and formats the json for the request.
 * @param articles
 */
1 reference
public static prepareAPIPayload(articles: IArticle[]) {
    let articleDescChunks: {text: string}[][] = [];

    let lastIndex = (array: any[]) => (array.length === 0) ? 0 : array.length - 1;

    articles.forEach(article => {
        let chunk = articleDescChunks[lastIndex(articleDescChunks)];

        if (chunk === undefined) articleDescChunks.push([]);
        chunk = articleDescChunks[lastIndex(articleDescChunks)];

        let value = {text: article.title + "</break>" +  article.description};

        (chunk.length === TranslationAPIService.API_REQUEST_STRING_COUNT)
            ? articleDescChunks.push([value])
            : chunk.push(value);
    });

    return articleDescChunks;
}
```

Item 4: Test Verification

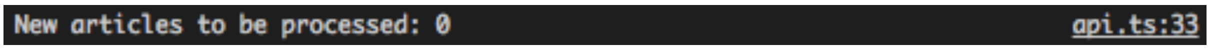
Item A.1

To verify sole-retrieval from the cache, queries with the same company and symbol were performed multiple times so that the News API would only return re-processed articles. To confirm this, logging was added within the request and as is evidenced in *Item A.3* the request succeeds with data, if the cache had not been functioning, then the request would safely return an error.



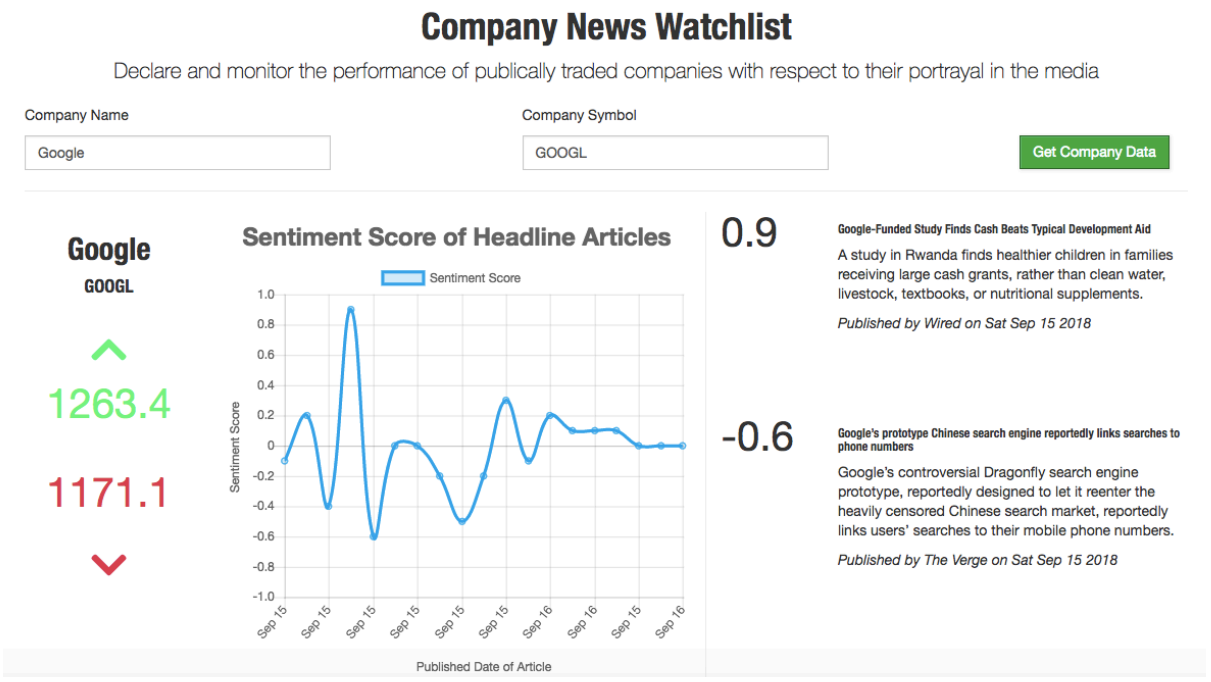
```
let cacheResults = [];
dataManager.dataSource.getAllNewsArticlesForCompany(req.body.name, req.body.symbol).then(articles => {
  // check for presence in cache
  ArticleCacheHelpers.validateArticlesAgainstCache(articles, req.body.name).then(splitCollection => {
    // move existing articles to the side, don't process
    cacheResults.push(...splitCollection.existing);

    console.info("New articles to be processed:", splitCollection.new.length);

    // if no new articles are present, don't even touch the endpoints...
    if (splitCollection.new.length === 0) {
      res.json(formatResponse(cacheResults, req.body.name, req.body.symbol));
    } else {
```

Item A.2



```
New articles to be processed: 0                                    api.ts:33
```

Item A.3



Company News Watchlist

Declare and monitor the performance of publically traded companies with respect to their portrayal in the media

Company Name: Google

Company Symbol: GOOGL

Get Company Data

Google

GOOGL

1263.4

1171.1

Sentiment Score of Headline Articles

0.9

Google-Funded Study Finds Cash Beats Typical Development Aid

A study in Rwanda finds healthier children in families receiving large cash grants, rather than clean water, livestock, textbooks, or nutritional supplements.

Published by Wired on Sat Sep 15 2018

-0.6

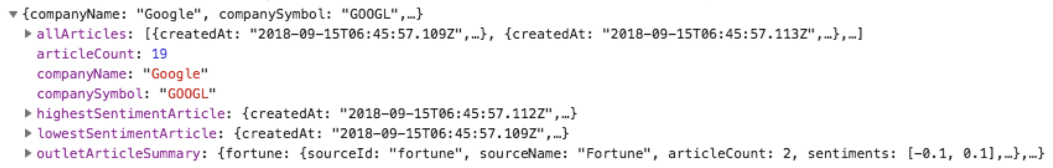
Google's prototype Chinese search engine reportedly links searches to phone numbers

Google's controversial Dragonfly search engine prototype, reportedly designed to let it reenter the heavily censored Chinese search market, reportedly links users' searches to their mobile phone numbers.

Published by The Verge on Sat Sep 15 2018

Item B.1

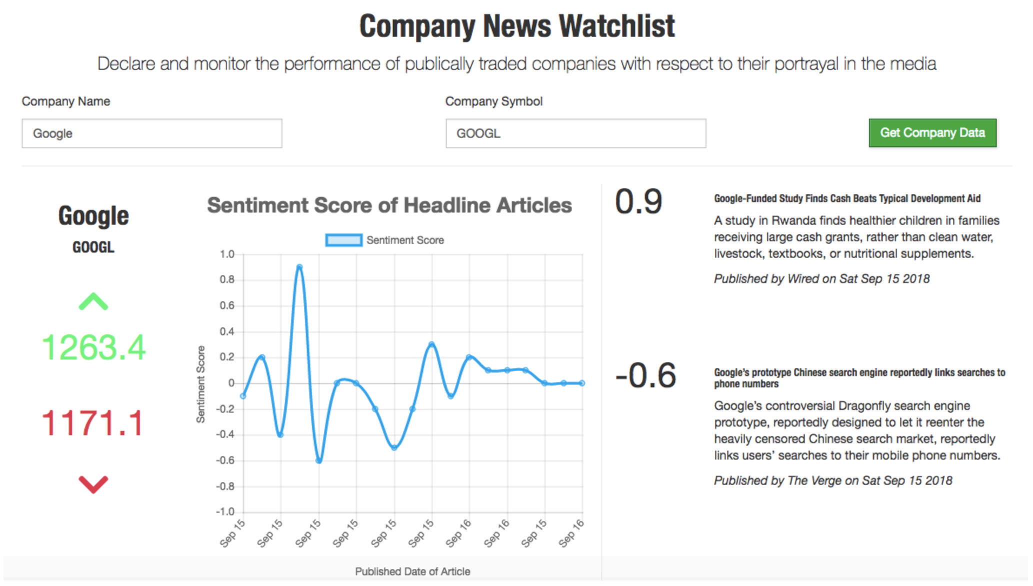
API response proving that articles are fetched correctly



```
▼{companyName: "Google", companySymbol: "GOOGL",…}
  ▶allArticles: [{createdAt: "2018-09-15T06:45:57.109Z",…}, {createdAt: "2018-09-15T06:45:57.113Z",…},…]
   articleCount: 19
   companyName: "Google"
   companySymbol: "GOOGL"
  ▶highestSentimentArticle: {createdAt: "2018-09-15T06:45:57.112Z",…}
  ▶lowestSentimentArticle: {createdAt: "2018-09-15T06:45:57.109Z",…}
  ▶outletArticleSummary: {fortune: {sourceId: "fortune", sourceName: "Fortune", articleCount: 2, sentiments: [-0.1, 0.1],…},…}
```

Item B.2

Output shown to the user



Company News Watchlist

Declare and monitor the performance of publically traded companies with respect to their portrayal in the media

Company Name: Google

Company Symbol: GOOGL

Get Company Data

Google

GOOGL

1263.4

1171.1

Sentiment Score of Headline Articles

0.9

Google-Funded Study Finds Cash Beats Typical Development Aid

A study in Rwanda finds healthier children in families receiving large cash grants, rather than clean water, livestock, textbooks, or nutritional supplements.

Published by Wired on Sat Sep 15 2018

-0.6

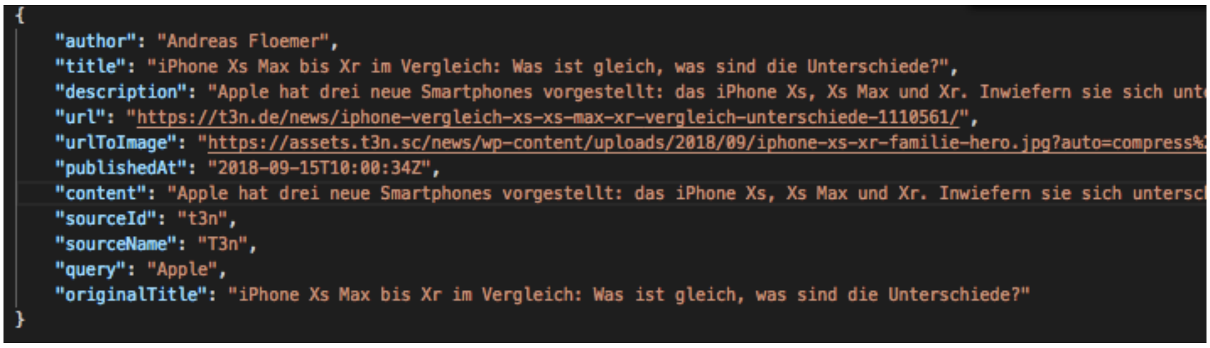
Google's prototype Chinese search engine reportedly links searches to phone numbers

Google's controversial Dragonfly search engine prototype, reportedly designed to let it reenter the heavily censored Chinese search market, reportedly links users' searches to their mobile phone numbers.

Published by The Verge on Sat Sep 15 2018

Item C.1

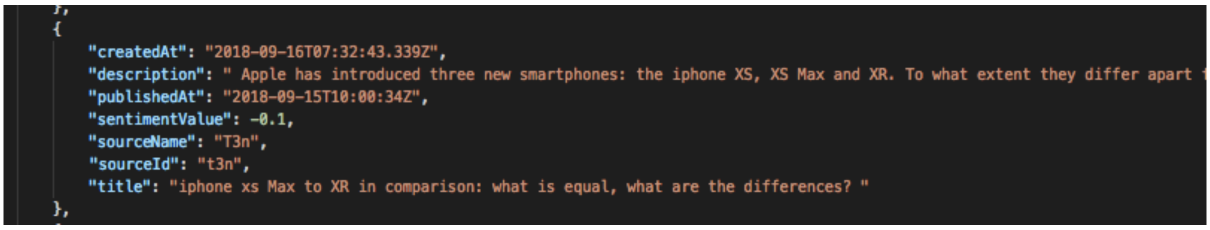
A newly detected article that has yet to be translated, this was acquired by break pointing the application as no original-language versions of articles are recorded.



```
{
    "author": "Andreas Floemer",
    "title": "iPhone Xs Max bis Xr im Vergleich: Was ist gleich, was sind die Unterschiede?",
    "description": "Apple hat drei neue Smartphones vorgestellt: das iPhone Xs, Xs Max und Xr. Inwiefern sie sich unte
    "url": "https://t3n.de/news/iphone-vergleich-xs-xs-max-xr-vergleich-unterschiede-1110561/",
    "urlToImage": "https://assets.t3n.sc/news/wp-content/uploads/2018/09/iphone-xs-xr-familie-hero.jpg?auto=compress%
    "publishedAt": "2018-09-15T10:00:34Z",
    "content": "Apple hat drei neue Smartphones vorgestellt: das iPhone Xs, Xs Max und Xr. Inwiefern sie sich untersc
    "sourceId": "t3n",
    "sourceName": "T3n",
    "query": "Apple",
    "originalTitle": "iPhone Xs Max bis Xr im Vergleich: Was ist gleich, was sind die Unterschiede?"
}
```

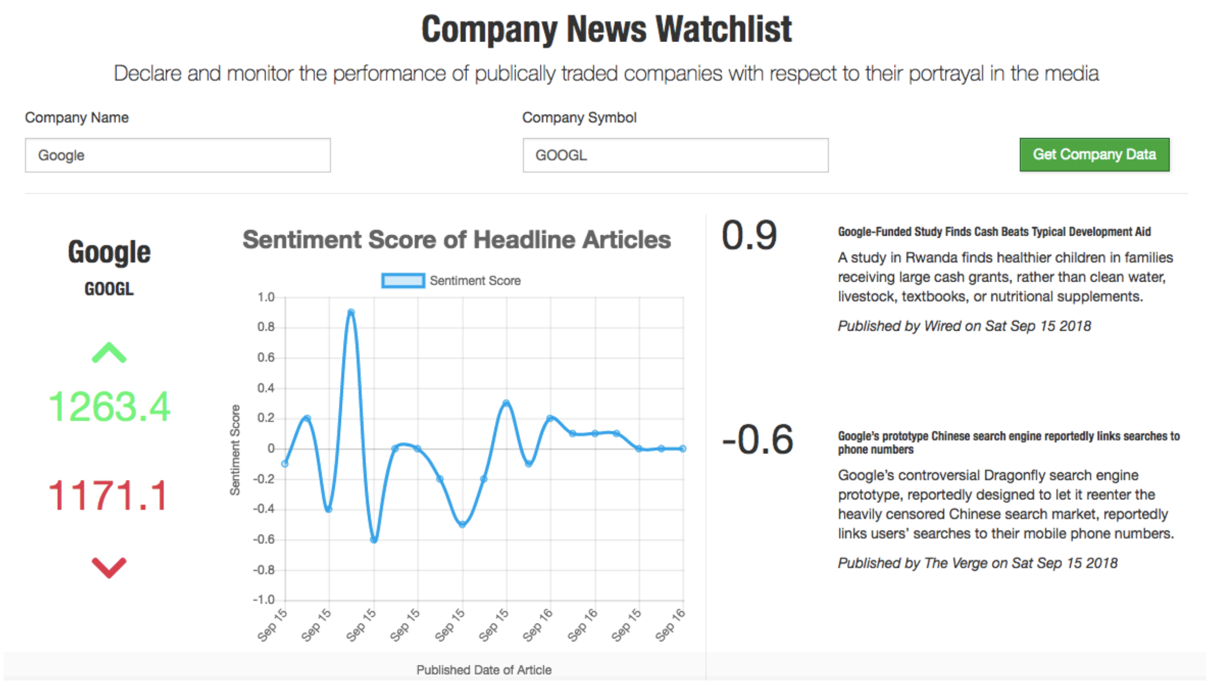
Item C.2

After execution, the query was run again and the corresponding article to the one above was found in its translated form, confirming that the translation service is working as intended.



```
    },
    {
        "createdAt": "2018-09-16T07:32:43.339Z",
        "description": " Apple has introduced three new smartphones: the iphone XS, XS Max and XR. To what extent they differ apart
        "publishedAt": "2018-09-15T10:00:34Z",
        "sentimentValue": -0.1,
        "sourceName": "T3n",
        "sourceId": "t3n",
        "title": "iphone xs Max to XR in comparison: what is equal, what are the differences? "
    },
```

Item D.1



Company News Watchlist

Declare and monitor the performance of publically traded companies with respect to their portrayal in the media

Company Name: Google

Company Symbol: GOOGL

Get Company Data

Google
GOOGL

1263.4

1171.1

0.9

Google-Funded Study Finds Cash Beats Typical Development Aid

A study in Rwanda finds healthier children in families receiving large cash grants, rather than clean water, livestock, textbooks, or nutritional supplements.

Published by Wired on Sat Sep 15 2018

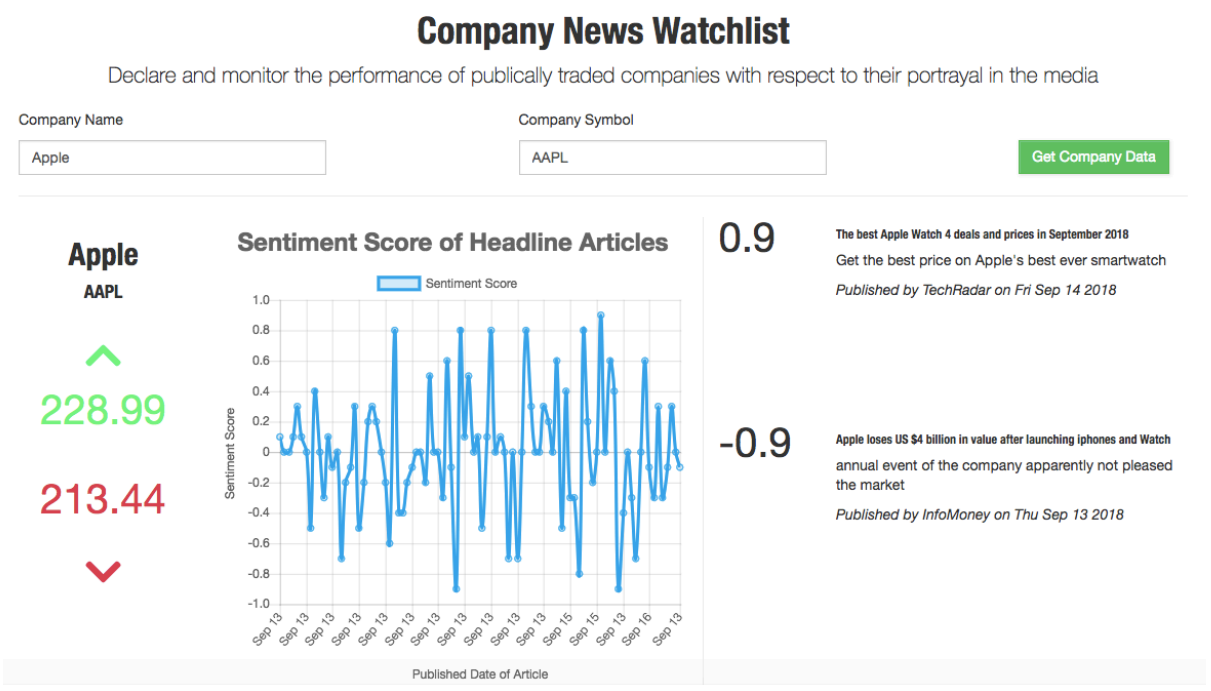
-0.6

Google's prototype Chinese search engine reportedly links searches to phone numbers

Google's controversial Dragonfly search engine prototype, reportedly designed to let it reenter the heavily censored Chinese search market, reportedly links users' searches to their mobile phone numbers.

Published by The Verge on Sat Sep 15 2018

Item E.1



Company News Watchlist

Declare and monitor the performance of publically traded companies with respect to their portrayal in the media

Company Name: Apple

Company Symbol: AAPL

Apple

AAPL

228.99

213.44

0.9

The best Apple Watch 4 deals and prices in September 2018

Get the best price on Apple's best ever smartwatch

Published by TechRadar on Fri Sep 14 2018

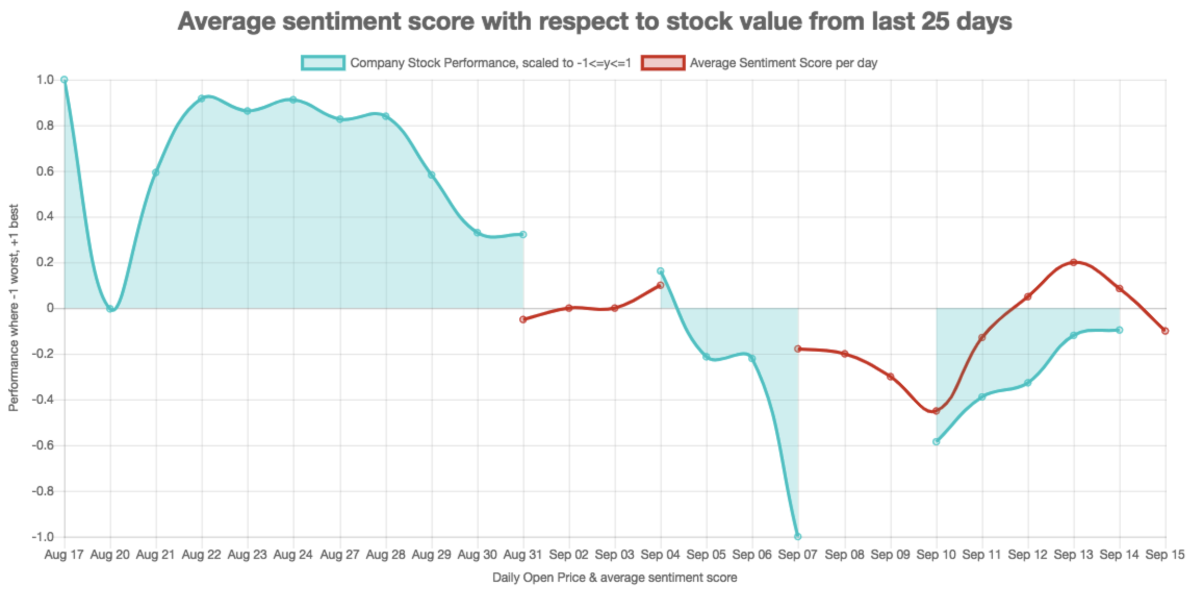
-0.9

Apple loses US $4 billion in value after launching iphones and Watch

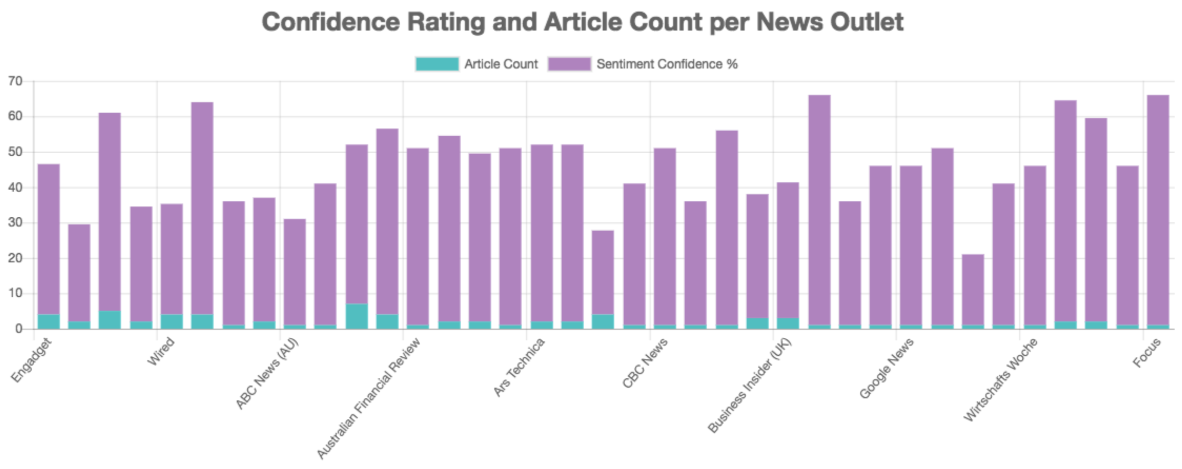
annual event of the company apparently not pleased the market

Published by InfoMoney on Thu Sep 13 2018

Item F.1

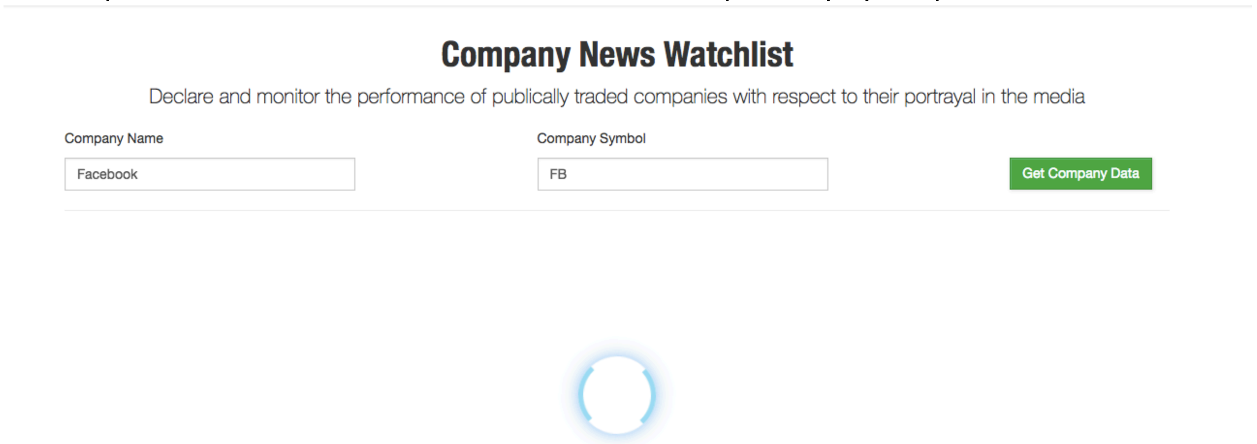


Item G.1



Item H.1

Upon search, a loading spinner presents itself for the entirety of the asynchronous action. On completion the results are returned as evidenced repeatedly by the previous test cases.

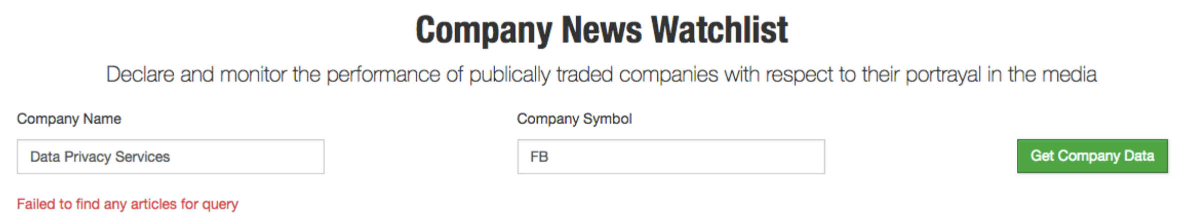


Company News Watchlist

Declare and monitor the performance of publically traded companies with respect to their portrayal in the media

Company Name	Company Symbol	
Facebook	FB	Get Company Data

Item I.1



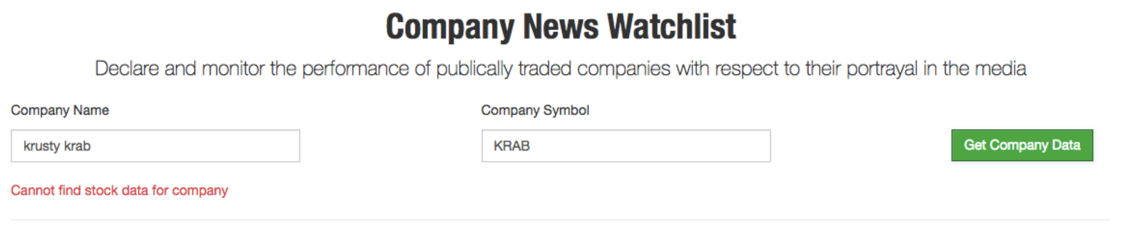
Company News Watchlist

Declare and monitor the performance of publically traded companies with respect to their portrayal in the media

Company Name	Company Symbol	
Data Privacy Services	FB	Get Company Data

Failed to find any articles for query

Item J.1



Company News Watchlist

Declare and monitor the performance of publically traded companies with respect to their portrayal in the media

Company Name	Company Symbol	
krusty krab	KRAB	Get Company Data

Cannot find stock data for company