# Customer's Requirements Specification Document

---

**Evan Chu, Matthew Hobbs**

**ICS4U**

**Gordon Roller**

# Table of Contents

## 1.0 - Project Description/Context

### 1.1 - Context

News-Bias-Detector (temporary name) is a service created by two students in a 3-month classroom environment. News-Bias-Detector provides users with the combined functionality that most services miss. There are many news aggregators, and there are bias detecting websites on the internet, but there are no services that provide both. As we are providing a service to the public, there is not a specific client. However, we are targeting our service for advertisers, so they could be considered a client.

### 1.2 - Curriculum Expectations

In Strand A, we will be meeting the expectations by employing many aspects of fundamental computer science. In substrand A1, we will be implementing type conversion by converting the XML response from the RSS feed into JavaScript objects. This will be important information for us to extract to allow us to display the title and description of each article the user would like to read. Substrand A2 will be simply met by using Angular.JS, which makes use of highly modularized components to make up the front end of the website. Substrand A3 will be met by using a mLab as a database, using a binary search algorithm for querying a specific topic or event in this mLab database, and implementing quicksort to sort the articles before they will be added to the database.

We will meet the expectations of Strand B primarily by using git, github, and other project management tools such as GanttProject. This will also be met by documenting our research and progress in the weekly journals, and continuing to create documentation.

## 2.0 - Overview of Product and its Desired Functionality

News-Bias-Detector is a news aggregator that will have the capability to sort articles by source and story. This aggregator will generate a synopsis for each story to assist the user in determining the most factually correct version of the story, free from bias. The goal for this web application is that users will be able to quickly and easily access an unbiased synopsis of a story where they can trust that all of the information is accurate. As we have no specific client, our product development will not be affected by outside sources. However, our requirements will be heavily dependant on research we conduct, as this service must provide the most functionality for our users.

## 3.0 - Software Category/Genre

This service will be used as an organizational and analytical tool for consuming news. This service will satisfy the analytical aspect by reading articles, and generating a synopsis that delivers the most important and factually correct information. The service will also act as an news aggregator, that organizes news articles by story, and source.

## 4.0 - Software Specific Requirements (Client Approved)

We are creating a web application as our platform for our news bias detector. The web application needs to be easy to use and aesthetically appealing. The features of our web application are listed below.

### 4.1 - Sort By Source

The web application shall implement a sort by source feature. The sort by source feature will not incorporate any synopsis generation, instead this section will function as a simple news aggregator by gathering all articles and grouping them based on source. The website will then link out to each article so the user can read into the story on the sources website.

### 4.2 - Sort By Story

The web application shall feature a sort by story functionality which will be much more complicated as it will group articles based on similar stories. For example, all articles that are talking about Hurricane Florence will be grouped together. This allows the user to see all articles based on a specific story in one place.

#### 4.2.1 - Synopsis Generation

Our sort by story feature shall implement synopsis generation. This will allow users an easy and quick form of consuming unbiased and factually correct information. The user will still be able to access all of the articles on the specific story, however, the synopsis should have enough information to explain the entire story in an unbiased manner. This synopsis generation will be completed by word embedding, which is a natural language processing method that represents words as vectors.

### 4.3 - Login and Preferences

To save specific preferences to users, we will have 2 main ways of signing in.

#### 4.3.1 - Google Sign-In

Our web application shall implement Google Sign-In. Google Sign-In is a service that Google provides to allow the user to sign in to websites without making extra accounts. It also provides a very secure authentication system, and allows for SMS verification for our service. Users can also create accounts on our service, have the convenience of being signed in on all devices, and skip entering credentials on all their devices using Smart Lock.

#### 4.3.2 - Proprietary Sign-In (optional)

Our web application shall incorporate a proprietary sign-in system. Our service will allow users to create an account to store preferences, it will also require password storage. For this reason, we will be encrypting passwords with salted password hashing. This requirement is optional, and will be used as a learning experience, as we will have already implemented a sign-in option using Google sign-in.

#### 4.3.3 - User Feed

The web application shall implement a user feed that gives access to a more personalized news feed. This news feed is 100% configurable by the individual user. The user will be able to add different specific tags to help sort through the massive amount of news. For example, a user could type in ‘Trump’ as one of their tags. Their personal feed will now only display stories about

Trump. The user can type in more than one tag if they want to receive more news in their personal feed. This same user might also have 'Hurricane Florence' as a tag, they will now receive stories about Trump and Hurricane Florence. This personal news feed will not overwrite the other two feeds (sort by story and sort by source), it will be in addition to the other feeds.

#### 4.3.4 - Email Feed (optional)

We shall implement a feature in which the user can opt into receiving weekly emails containing the top stories from their personal feed. This feature will allow users to receive their news without even having to log onto the web application. This would be very convenient for the user.

### 4.4 - User Interface

The web application shall be very simple to use, we are targeting a broad user base so we need to make it very accessible. We will implement a simple to use navigation bar to help the user find exactly what they are looking for, this navigation bar will include sections such as; home, articles by story, articles by source, about, and contact.

#### 4.4.1 - Home (Landing Page)

When a user first arrives on our web application they shall be greeted by a landing page. This landing page will explain what the application does and will include a call to action that links to the articles by story page. The whole idea for this page is to leave a good first impression so the user continues to browse through our service.

#### 4.4.2 - Articles By Story

This page shall be where users will spend most of their time when using our web application. This section of our web application is broken down into two separate sections, a navigation area on the left, and the story information on the right.

##### 4.4.2.1 - Articles By Story Navigation

This area of the articles by story page shall be used to select what story the user would like to read about. This area has to be easily navigable to ensure the user is able to find anything that they want. The top of this area will show the top story (title and description). The top story is determined by how many articles and different sources are talking about it. If there are more articles in a recent time frame it is very likely that the user would like to read about it. Below this top story will be the story search area. This area will consist of a search bar and a search results area. By default the search bar is empty and the search results area will include every story in the database. However, when the user types something into the search bar the search results will be updated with stories relating to the query.

##### 4.4.2.2 - Articles By Story Content

This area shall contain all of the information relating to the story that the user has selected. The information will be organized as follows. First, the story title will be displayed. The automatically generated synopsis will then be displayed below the title. This synopsis is very important in our web application so we will try to push it to the user as soon as possible. The content area will then be filled out by all of the articles relating to the specific story.

#### 4.4.3 - Articles By Source

The articles by source page shall be similar in layout to the articles by story page. The only differences between these two pages is that the navigation area will contain sources instead of stories (no top story), and the content area will not contain a story title and synopsis (just articles by the specified source).

#### 4.4.4 - About

The about page of our website shall explain in detail the functionality of our web application. It will also include a quick and simple tutorial on how to use the web application. This page is designed to make our website more accessible to a larger audience as users without experience on the web will still be able to use our application. News has been delivered in person through newspapers for a long time and some people still lack the experience with websites. Our web application should make the transition from a hard copy to a digital copy easy.

#### 4.4.5 - Contact

This final page on our web application shall once again enhance the usability of our web application. This page will include an easy method of contacting us through an email address associated with our web application. The users will be able to reach out with any problems experienced while using our service. This will also help us fix any bugs that we may not catch during the testing phase of development.

## 5.0 - Target Audience

The target audience general public who are looking to gain more knowledge in a rapidly changing news climate. This service will specifically be targeting people that are cautious in taking information from a news source, and would like to be able to verify information using one streamlined service.

## 6.0 - Platform

This project makes use of many different frameworks, libraries, and languages. Node.js makes installation of packages very easy through NPM (Node Package Manager) and VSCode allows for easy transition between different languages.

### 6.1 - MEAN Stack

We will be using the MEAN stack to develop our website. The MEAN stack is a JavaScript software stack for developing web applications. The acronym MEAN stands for MongoDB, Express.js, Angular.js, and Node.js.

#### 6.1.1 - MongoDB

MongoDB is a noSQL database that uses JSON like documents to store data. Some cloud MongoDB services such as mLab offer free deployments that allow the user to store up to 500MB of information. MongoDB can also be set up locally on a machine, however, since we are working in a group we will be using mLab for our database.

#### 6.1.2 - Express.js

Express.js a web application framework for Node.js. Express is commonly known as the standard server framework for Node.js. We will be using Express.js along with CORS to complete our backend Angular.js will then make HTTP requests to this Express.js server.

#### 6.1.3 - AngularJS

AngularJS is frontend web application framework developed and maintained by Google. AngularJS is used to develop responsive single-page web applications. Angular.js makes use of modularity through the implementation of components and modules.

#### 6.1.4 - Node.js

Node.js is a JavaScript runtime environment that executes JavaScript code outside of the browser. Node.js allows the developer to write and run server-side scripts to produce dynamic content before the page is sent to the web browser. Node.js allows a fully JavaScript based development environment from the frontend to the backend.

### 6.2 - Version Control

Version control is extremely important in a large project, it allows us to return to previous iterations of our code if we require to do so. We will use a version control software to keep all of our code backed up and organized.

#### 6.2.1 - Git

Git is a version control software that allows collaboration, and tracking within large and small projects. Git will be used extensively to update the specific changes that each of us have made to files, and will act as a secondary way to track progress.

#### 6.2.2 - GitHub

GitHub is an online cloud service that provides hosting for repositories which will contain all of our code. We will push our code to GitHub with Git.

### 6.3 - Languages

Node.js allows us to use JavaScript throughout our entire project which simplifies the programming process quite a bit as we only need to learn one language for both the frontend and backend. We will also be using HTML and CSS.

#### 6.3.1 - JavaScript/TypeScript

JavaScript is a weakly typed programming language that is usually used for frontend web development, however, we will also be using it for our backend through the use of Node.js. JavaScript is an extremely versatile and popular language, this language will be great for development of our website. TypeScript is a superset of JavaScript that allows for optional static typing. Angular.js uses TypeScript by default, although JavaScript code can be typed into a TypeScript file. TypeScript compiles to JavaScript.

#### 6.3.2 - HTML/CSS

We will be using HTML and CSS to create the visual aspects of the frontend. HTML is the standard markup language for creating web pages and web applications. HTML is used alongside CSS and JavaScript to create functional and aesthetically pleasing websites. CSS stands for Cascading Style Sheets and it is used to describe the look and feel of HTML elements.

#### 6.3.3 - Python

Python is a multi-paradigm programming language typically used in scripting, and object oriented programming. Python is strongly typed since version 3.5, and is dynamic opposed to Java, which is statically typed. We will be using Python to write our machine learning models, as the API that we will be using requires it to be written in Python.

### 6.4 - Libraries

Our project will make use of machine learning, there are a few machine learning libraries that we will implement to utilise this powerful method of analysing data.

#### 6.4.1 - TensorFlow.js

Our project will implement TensorFlow.js, which is a deep learning library developed by Google to create, train, and deploy machine learning models on the browser. This model will be identifying bias, and creating the synopsis for each news article.

#### 6.4.2 - Keras

Keras is a high level deep learning API created to improve user friendliness, and increase the speed that a machine learning model can be prototyped and created. Keras is written in Python, and is capable to run on TensorFlow, CNTK, and Theano, which are all deep learning libraries. We will be using Keras to create and train the model, however, we will be implementing this model into TensorFlow.js to allow the model to be used in the browser (Keras models are supported in TensorFlow.js).

### 6.5 - Packages

We will make use of some packages to help pull and sort information from the internet. NPM (Node Package Manager) makes the installation process of these packages very simple. It is important to note that we may decide to use different packages when we begin programming, the following packages are packages that we believe will help us throughout our coding process as they are very popular in MEAN web application development.

#### 6.5.1 - CORS

CORS stands for cross-origin resource sharing. CORS manages cross-origin requests and specifies who can access information on the server. CORS can also decide what HTTP request methods are allowed from different external sources. This helps prevent malicious attacks from outside sources.

#### 6.5.2 - Request

Request is used to make HTTP requests to websites. We need to gather as much information about a story to develop an accurate synopsis, it is for this reason that we use request to pull all of the HTML from each article page. The HTML code is then handled with Cheerio (explained below).

#### 6.5.3 - Cheerio

Cheerio is used for parsing information from HTML code. We have already gathered the raw HTML through Request and will now use Cheerio to pull the specific information we require. Cheerio can be provided an HTML tag, class, or id, and it will pull all of the information from the HTML code that fits the specified tag, class, or id. It is important to note that we will be using Request and Cheerio to pull articles from the internet as there is not feed to pull these pieces of information.

#### 6.5.4 - RSS-Parser

RSS-Parser is a simple library that can pull information from an RSS-feed. This library turns RSS XML feeds into JavaScript objects. It is very simple to access the information after it has been turned into a JavaScript object (for example, item.title and item.pubDate).

#### 6.5.5 - Mongoose

Mongoose is an object data modeling (ODM) library. Mongoose is an easier way to create, access, and delete documents from a mongo database. This added functionality comes from a robust strongly-typed schema system that is mapped to a MongoDB document.

#### 6.5.6 - Node-Cron

Node-cron is a a task scheduler in JavaScript. We will be using node-cron to systematically check for new articles and save them to our database. Without node-cron it would be much more difficult to have a reliable schedule for searching for new articles.

#### 6.5.7 - Nodemon

Nodemon is a package that helps with the development of our web application. Typically, every time we would want to make a change we would have to manually restart our node server in the command line. Instead, we can run our server with nodemon. Nodemon checks for changes and restarts itself so we do not have to waste our time restarting our program. This allows for a live view of changes when programming.

### 6.6 - IDE (Integrated Development Environment)

#### 6.6.1 - Visual Studio Code

Visual Studio Code is a free source code editor developed by Microsoft. VSCode has support for Git which will be useful when we want to push our code to the GitHub repo. VSCode also allows for access to any type of file, this is useful in our project as we are not using a single language for everything. An IDE such as Eclipse would only allow use for Java.

#### 6.6.2 - PyCharm

PyCharm is a free IDE developed by JetBrains for use with Python. PyCharm is an incredibly useful code editor when working with Python code as it provides the developer with integration with version control systems (in our case, Git). We will be using PyCharm to write and debug our Keras model.

### 6.7 - Target Platform

We are creating a web application which should be accessible from any device that can access the internet. We will make sure to implement scaling to allow for use on mobile devices and other devices with different aspect ratios.

## 7.0 - Competition

Our idea for the service was to provide users benefits that no other services currently have. This means we tailored the requirements to specifically target areas that no other services are currently providing. There are two main services that could be considered our competition, Politifact, and Google News.

### 7.1 - Politifact

Politifact provides fact-checking for most of the popular news articles or quotes from political figureheads. However, Politifact is fact-checked by a large group of contributors, and lacks the scalability of our service. Politifact also does not act as an aggregator, which is one of the main functionalities of our service.

### 7.2 - Google News

Google News provides what Politifact lacks, which is a news aggregator. Google News misses our main functionality which is bias detection. There are many news aggregators on the web, however none combine the aggregation and bias detection into one fluid service.