**Adv. Front-End Programming Final Project Proposal**

**Ron Dyakov and Ahmed Waseem**

**Scope:**

The aim of this project is to develop a React application that allows users to search for articles using keywords and filters and display some information about the article including the word count, view count, and its sentiment.

**Functionalities:**

1. Archive Search Function

* Allows users to search for any article they’d like by entering the year and month
* Website will display articles posted during the entered month and year

1. Filter Search Function

* Allows users to apply filters such as genre, date, etc
* All articles meeting the filters will be displayed

1. Article Length

* Displays the total word count

1. Top Stories

* Display articles currently on the front page of different sections (i.e arts, business, science, etc.)

To navigate each page on our website we will use routing.

Most components will be a child component of the "ArticleList" component. The ArticleList component contains functions which fetch the necessary API calls and stores the result using Reacts "useState" hook. The json returned by an API call is passed down using props, along with the function used to fetch the given data. This is done since each API call has some input that changes depending on what the user wants to filter. There will be multiple components which get user input and will then call the function passed down by ArticleList. That's the component composition, ArticleList contains the functions that fetch and the variables that store the result. ArticleList then passes down only what other components need, whether that be data or functions.

The ArticleList component will first display some default message which will notify the user that our website is waiting for a result of a fetch to an API. Once the API has returned a result ArticleList will update and all components will display the returned articles. This is done using the "useEffect" hook.

Other components will either take user input (i.e. a button or text box) or display article information (probably using elements like <p>). These components will once again rely on ArticleList to receive or fetch this information.

**APIs:**

We will be using four API's provided by the New York Times, their Archive, Article Search, Most Popular, and Top Stories API, which return articles based on different parameters. We will also be using a language analysis API developed by Google.

**Links:**

**Archive API:** https://api.nytimes.com/svc/archive/v1/2019/1.json?

**Article Search API:** https://api.nytimes.com/svc/search/v2//articlesearch.json?q={query}&fq={filter}

**Most Popular API:** https://api.nytimes.com/svc/mostpopular/v2/viewed/1.json?

**Top Stories API:** https://api.nytimes.com/svc/topstories/v2/

/{section}.json

**Cloud Natural Language API:** https://language.googleapis.com/v2/documents:analyzeSentiment

**Composition:**

We will be using composition by passing props.children from our ArticleList component, into our ArchiveSearch component. This will display the website’s header for the archive search page.

**Timeframe:**

Week 11 – Complete the project proposal

Week 12 – Complete the skeleton of the project

Week 13 – Complete the entire project

Week 14 – Test the project, fix any bugs