Project Plan
News-Bias-Detector

Evan Chu, Matthew Hobbs

ICS4U

Gordon Roller

Table of Contents

1.0 - Introduction

News-Bias-Detector is a news aggregator that will make use of machine learning to deliver an unbiased synopsis of news articles from various news sources. Any additional information about this project can be found in the concept plan.

2.0 - Requirements

The requirements for our program have been described in extensive detail in the concept plan. These requirements include the following pages on our website; home, articles by story, articles by source, about, and contact. The website will also include a user login system which is once again explained thoroughly in the concept plan, and in section 4.1.6 in this document.

3.0 - Deliverable List

- Source Code

Marketing:

- Logo
- Promotional Advertisement

Documentation:

- Concept Plan
- Project Plan
- Design Proofs (paper prototype, and digital prototype design)
- Alpha Test Analysis
- Beta Test Analysis
- Design Document
- Final Presentation
- Weekly Journals

4.0 - Tasks

4.1 - Work Breakdown

4.1.1 - Product Management

Most of the product management in this project has already been outlined in the concept plan, and in section 4.3 - 4.5 of this document. The rest of the product management for this project will be maintaining the schedule that has been created, and maintaining the continual documentation, such as the weekly journals.

4.1.2 - Frontend

The frontend is responsible for displaying the article information that our backend code gathers. This section will make use of *ngFor and *ngIf (if statements and for loops in HTML code) to help display the information required. We will attempt to complete the majority of code in the backend to help ensure a quick and enjoyable to use frontend.

The components in the angular frontend will use the HTTPClient module to make HTTP requests to our express server. We can now make HTTP requests to the backend using a simple line of code; var *information* = http.get(*insert link here*).

4.1.3 - User Interface

We will be using Adobe XD to prototype our user interface design. Adobe XD is an extremely powerful user experience design software as it allow you to route sections of your design to mimic a working webpage. The rest of this design tool acts similarly to Adobe Illustrator and Adobe Photoshop. A more detailed description of our planned user interface can be seen in our concept plan.

4.1.4 - Backend

Our backend will include RSS and article parsing. This section will be responsible for gathering article information from the internet. The packages required for this section of the backend are Node-Cron, Request, Cheerio, and RSS-Parser. Cron will call the function to perform this parsing on a regular schedule. RSS-Parser will then gather each article’s title, publish date, link, and any other important information. This is followed by the use of the request package to gather the HTML code from the article page. Cheerio will parse this HTML code to return the article text and not the other HTML elements. Once all of the information has been gathered it will be saved to the mLab database.

The article information will be made accessible to the frontend through the express router. The express router helps create routes where the frontend can access the specific information required, for example, one route may return articles sorted by source and another route will return articles sorted by story. Express routing is extremely useful as it places the processing required to sort the articles within the backend, thus speeding up the load times for the individual user when accessing the frontend of the website. This routing also greatly simplifies passing information to the frontend.

The backend will include sorting articles by source and generating a synopsis of each story. These sections are explained in detail in sections 4.1.4 and 4.1.5.

4.1.5 - Sort by story

The web application will implement a sort by story functionality which will complicated as it will group articles based on similar stories. This allows the user to see all articles based on a specific story in one place. This will be completed by comparing the vectors of the words in different articles, to allow them to be sorted by the closest distance between the sum of these vectors.

4.1.6 - 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.

4.1.7 - Login and Preferences

As stated in the concept plan, we will be implementing google sign-in to store user preferences. These preferences include, having a personal user feed that can be customized, and an optional email feed that could send updates on topics the user is interested in. A proprietary sign-in option will be implemented if we have time at the end, as it will act as a learning experience.

4.2 - Schedule

Below is the schedule for all of the features that will be implemented in our program, the schedule contains the dates that each feature must be implemented by.

Major Milestones <u>underlined</u> and *Italicized* in Schedule*

Sunday, September 30

- *<u>Concept Plan</u>*

Sunday, October 14

- *<u>Project Plan</u>*

Monday, October 15

- Begin Learning Tensorflow

Wednesday, October 17

- Finished Preliminary UI Design

Friday, October 19

- *<u>Design Proofs</u>*
- Created Angular Components
- Created Express Server
- Created Angular Routing

Friday, October 26

- *<u>Implemented RSS Parsing</u>*
- *<u>Implemented Article Parsing (Request and Cheerio)</u>*
- Finish Learning Tensorflow
- Begin Sort by Story with Machine Learning

Friday, November 2

- Implemented Landing Page UI
- Implemented About Page UI
- Implemented Contact Page UI

Friday, November 9

- Implemented Sort by Source UI
- *Finish Sort by Story with Machine Learning*
- Begin Synopsis Generation with Machine Learning

Friday, November 16

- Implemented Sort by Story UI

Monday, November 19

- *Alpha Test Analysis*

Friday, November 23

- Implemented Google Login

Friday, November 30

- Implemented User Feed Functionality
- Implemented User Feed UI
- *Finish Synopsis Generation with Machine Learning*
- Begin Training Machine Learning Models
- Evan Begin QA / Verification Until End of Project

Friday, December 7

- Implemented Email Feed (*optional*)

Friday, December 14

- Finished UI Cleanup
- Matthew Begin QA / Verification Until End Of Project

Monday, December 17

- *Beta Test Analysis*

Friday, December 21

- Finish Training Machine Learning Models

January 14-18

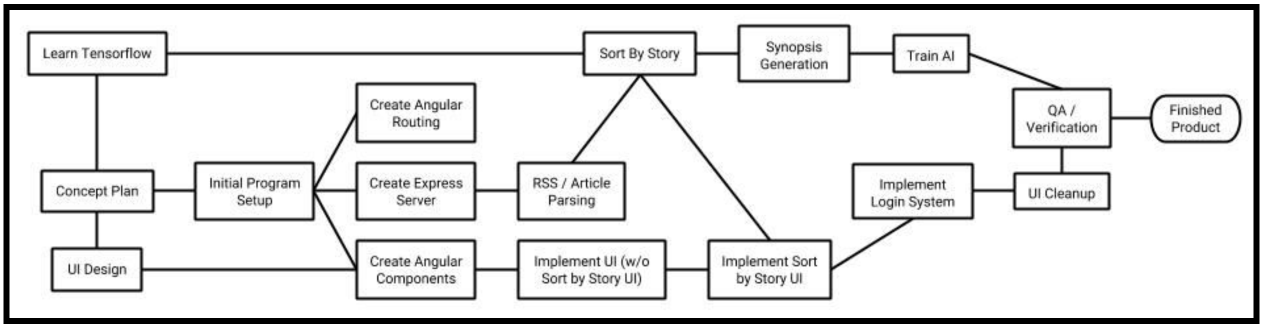
- *Final Presentation*

January 24

- *Design Document*

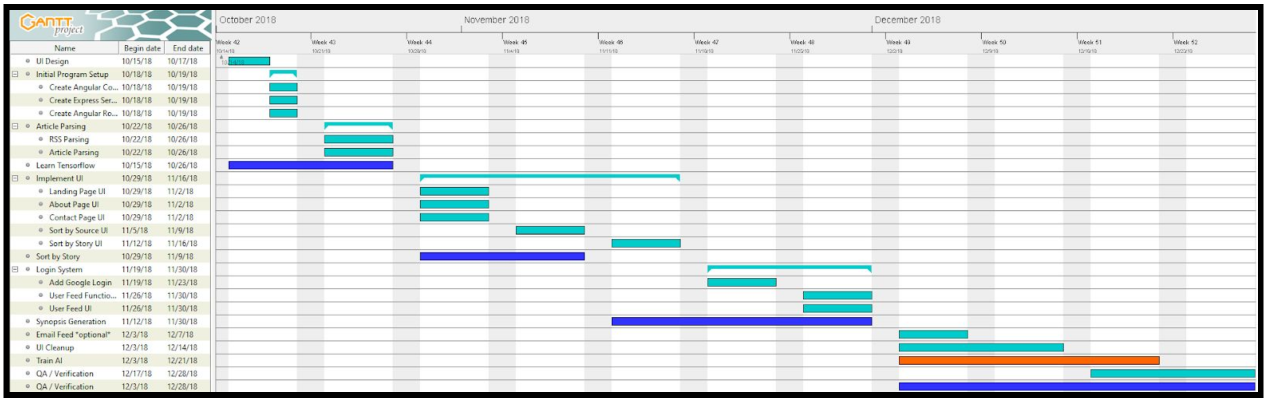
4.3 - Critical Path

It is very important that we stay on schedule throughout our project, the is due to the fact that in some cases one group member can not start until the other member has completed their section of the program. The critical path below is a good visual representation of the intertwined nature of our project.



4.4 - Gantt Chart

The Gantt Chart is colour coded based on the developer required to complete the task. Matthew is required to complete the light blue sections while Evan is required to complete the dark blue sections. The orange section represents the machine learning models training time which will run while we are working on other aspects of the program. In any group project, it is imperative that the delegation of tasks is completed fairly. This is why Evan's section involves long stretches of time with a single task. Due to the complexity of implementing the specified features through machine learning, the time to complete this task is lengthened. Matthew has many more sections with less time for each feature, this is due to the fact that these features are quicker to implement than the machine learning models.



Name	Begin date	End date
UI Design	10/15/18	10/17/18
⊟ Initial Program Setup	10/18/18	10/19/18
Create Angular Co...	10/18/18	10/19/18
Create Express Ser...	10/18/18	10/19/18
Create Angular Ro...	10/18/18	10/19/18
⊟ Article Parsing	10/22/18	10/26/18
RSS Parsing	10/22/18	10/26/18
Article Parsing	10/22/18	10/26/18
Learn Tensorflow	10/15/18	10/26/18
⊟ Implement UI	10/29/18	11/16/18
Landing Page UI	10/29/18	11/2/18
About Page UI	10/29/18	11/2/18
Contact Page UI	10/29/18	11/2/18
Sort by Source UI	11/5/18	11/9/18
Sort by Story UI	11/12/18	11/16/18
Sort by Story	10/29/18	11/9/18
⊟ Login System	11/19/18	11/30/18
Add Google Login	11/19/18	11/23/18
User Feed Functio...	11/26/18	11/30/18
User Feed UI	11/26/18	11/30/18
Synopsis Generation	11/12/18	11/30/18
Email Feed *optional*	12/3/18	12/7/18
UI Cleanup	12/3/18	12/14/18
Train AI	12/3/18	12/21/18
QA / Verification	12/17/18	12/28/18
QA / Verification	12/3/18	12/28/18

Larger Version of Gantt Chart: https://gyazo.com/d50de745df7eb6adec8accdf53bf14b0

5.0 - Resources

5.1 - Team Structure

We have decided to split the workload based on our strengths in programming. Matthew has been learning web development throughout the summer and will be responsible for the majority of UI design and web implementation. Evan has researched machine learning and will be responsible for the majority of the machine learning aspects in our project. It is important to note that we will be working together on many features in the program as we would both like to understand everything that is being implemented.

5.2 - Software

The software that will being used to complete project was extensively explained in the concept plan. However, we have decided to use another piece of software called Adobe XD. Adobe XD is a user experience design software application designed by Adobe Systems. We will be using this software to create functional prototypes for our UI. This will be extremely useful when attempting to create the aesthetics of our website using HTML and CSS as we will have a detailed reference to look at.

Extra software dependencies such as Nvidia's CUDA Toolkit, and the cuDNN SDK have been added. These dependencies will be required to allow GPU support with TensorFlow.

5.3 - Hardware

As we are using a third party server to store our data from the news articles, we do not have many hardware requirements.

However, to train our machine learning models, we will be using TensorFlow with GPU support. GPU's typically decrease the amount of time to train a model by a significant margin, especially if the GPU is on the higher end. As previously mentioned, we have two GTX 1070's which we will be using. These two GPU's are in different computers, so we will have to create a cluster of TensorFlow servers, and distribute a computation graph across this cluster.

General hardware requirements include two laptops for development during class time (Lenovo Thinkpad's).

6.0 - Risk Analysis

Our program contains many complex components and risk analysis is vital to ensure we do not suffer any catastrophic failures later in the development of our website. We have determined that all of the possibles risks have solutions and as long as we stay on schedule these risks will likely never become a reality.

For the following section, the formula Risk Exposure = probability (decimal between 0 and 1) * impact (value between 1 and 10) will be used to assess the magnitude of the risk.

6.1 - Group Member Falls Behind Schedule

Following the predetermined schedule will be very important in creating a completely functional and polished service. If a group member falls behind schedule, compromises might have to be made to the quality, or the project's scope might have to be limited.

To ensure this risk does not affect our service, clear communication will be required as the other group member could possibly delay what they are working on, and help complete the task that is behind schedule. Also, the verification phase has some extra time allotted specifically for this scenario, acting as overflow if certain tasks were not completed in the time specified in the schedule.

Risk Exposure: 0.8 x 8 = 6.4

6.2 - Creating and/or Training the Network is Completed Too Late

In order to have accurate grouping by story, and an accurate synopsis generation, the machine learning models must be implemented properly and given as much time to train as possible. If the model does not have necessary amount of time to train, the synopsis generation or grouping by story will become much less accurate, and therefore decrease the functionality of our service.

To prevent this risk from affecting our service, ensuring that the machine learning model is implemented within the allotted time (Section 4.5) will be necessary. One other way of decreasing the chance this risk affects our service is to train the models in parallel, as there are two strong GPU's (GTX 1070's) at our disposal.

Risk Exposure: 0.7 x 8 = 5.6

6.3 - Disagreement Between Team Members

Throughout the development cycle, there are chances of disagreements between team members. These disagreements could possibly delay the completion of the task, and therefore delay the overall schedule.

Collaboration between components is very important, however, letting the group member who is responsible for the task use their discretion will be important to ensure that components work well together.

Risk Exposure: 0.5 x 5 = 2.5

6.4 - mLab Database Offline

mLab will be used to store all of the data from the news articles. If mLab goes down for maintenance, changes its terms of use, or the data that will be stored exceeds 500 megabytes, our service will be directly affected.

Our program will likely not exceed the 500 megabyte limit set by mLab. After a small amount of testing we determined that 1000 documents (news articles) will use approximately 5 megabytes of storage. This approximation would mean that we could store 100,000 documents before running out of space. The program will contain a built in feature that deletes old articles, our initial limit will be 7 days. If we determine that we can not store 7 days of articles we could change this limit to 5 days, however, we would prefer to store a larger backlog of articles to give the user the ability to search for older stories.

If all else fails and we are unable to use mLab as our cloud hosted database we could either move to a different cloud service, or set up our own database and create our own method of connecting to it through the internet.

Risk Exposure: 0.3 x 8 = 2.4

6.5 - RSS Feeds Offline

To pull simple data (title, description, link to article) from a news source you must use the RSS feeds that are provided. However, these RSS feeds often go down, which will not allow the data that is required for our project to be pulled.

This risk is out of our control, and is a very common occurrence. However, this risk can be corrected by ignoring the RSS feed that is down, and only pulling from other news sources. This will not affect the overall usability of our website as important stories will likely be talked about by the majority of news sources.

Risk Exposure: 0.9 x 2 = 1.8

6.6 - Data Loss

When the project is being worked on, there is a chance that the current progress could be lossed. Git and GitHub will help mitigate this loss, as all of the previous versions will have a backup, however, it is possible that the progress that has not been committed to the repository to be lost. This amount of progress would be relatively low, so the overall impact of this data loss is very low.

Risk Exposure: 0.2 x 1 = 0.2

7.0 - Team Policies

7.1 - Regular Repository Commits

We will be using Git as our version control software for our project. It is vital that we regularly commit our code to the GitHub repository, this ensures that we both understand the progress that the other team member is at. While it is important to commit regularly to the repository, it is also important that we do not partake in useless commits. It is important to only commit when a specific change or addition was made. This will help keep the repository clean and easy to find previous commits.

7.2 - Communication Policy

In tandem with regular repository commits, active and constructive communication will be necessary to complete this project within the designated time, and also to decrease the chances for misunderstandings, which is the largest contributor to intrateam friction. During class time, communication will be simply done through talking to the other team member, however, outside of class time, Discord will be the main mode of communication. Discord was chosen as it is one of the forms of messaging that both of the team members are active on, it is very simple to use, and it has extra functionality such as voice calling and video calling, which could be used to further the discussion. As we will be using scrum as our software development life cycle, we will have daily scrum meetings (shortened due the amount of time we have in class), and a sprint retrospective, which will go over what was done well during the sprint, and what can be improved for the next one. The usual time to complete a sprint is 2-4 weeks, however based on the amount of time we have to complete this project, each sprint will approximately take 1 week.

7.3 - Quality Assurance

This project is designed and meant to act as service that will be provided to the public. This means it is extremely important to maintain the utmost quality that we can achieve. To ensure this quality policy is met, it will be very important to verify every component is working without flaws before integrating it with the rest of the codebase. This will most likely be implemented during the sprint review, as we can discuss the quality of the code that was written during the sprint, and if any of the code is not up to the standards set out, the individual responsible for that task will have to make the necessary changes. One other timeline change that we have implemented to help this policy is extending the verification phase. This means that if a certain task is not completed to our quality standard, it might overflow into this verification phase. We value quality over implementing the optional aspects of our requirements outlined in the concept plan. This policy is very important to this project as we hope to continue developing the service past the scope of this course.