Text-to-speech Sentiment Analysis App

Developers Manual

Skylar Reichle
Jenna Chiarella
Maci Hadley
Madison Champion

Department of Computer Science and Software Engineering Samuel Ginn College of Engineering, Auburn University

COMP 4710 Senior Design Project Digital Humanities Team #1

April 25, 2025

Table of Contents

1 iFrame Integration

1.1 Creating REACT App with Vite

1.1.1 Prerequisite Installation Guide

1. Install Node.js & npm
Go to the official Node.js website and download the latest LTS version
Run the installer and follow the steps
Once done, open your terminal and run:

- node -v
- npm -v

1.1.2 Set Up a React App with Vite

1. Create the App
Open your terminal and run:

- npm create vite@latest my-app --template react
- cd my-app

2. Install Dependencies
Install all required packages:

- npm install

3. Run the App Locally
To start your dev server:

- npm run dev

1.1.3 Add GitHub Pages Support

1. Install the deploy tool

- npm install gh-pages --save-dev

1.1.4 Deployment to GitHub Pages

1. Update package.json by adding this line at the top level (outside “scripts”)

- "homepage": "https://<your-github-username>.github.io/<your-repo-name>"

2. Update or add these “scripts” to package.json

```
"dev": "vite",
"build": "vite build",
"predeploy": "npm run build",
"deploy": "gh-pages -d dist"
```

1.1.5 Push to GitHub

1. Initialize Git
   - git init

2. Create a .gitignore file and add:
   - node_modules
   - dist

3. Create a GitHub repo at: https://github.com/new
Name it the same as your app (e.g., my-react-app)

4. Link your local app to the GitHub repo:
   - git remote add origin https://github.com/<your-username>/<your-repo-name>.git
   - git branch -M main
   - git add .
   - git commit -m "Initial commit"
   - git push -u origin main

1.1.6 Deploy to GitHub Pages

1. In terminal, run:
   - npm run deploy

2. After a few seconds, your app will be live at:
https://<your-github-username>.github.io/<your-repo-name>/

2 Text-to-Speech

2.1 About

For the text-to-speech (TTS), the goal was to have an audio file of a TTS reading of each nonfiction text to be used by the audio player. It was implemented this way to keep the audio player independent from the text-to-speech and to make it easy to switch the TTS software or to use multiple. We decided to use Microsoft Azure speech services to create the TTS readings.

2.2 Creating TTS Output

First, we created a speech resource on Azure which we could use. Using this resource, three scripts were used to automate creating TTS readings:

python 20 char.py

This was used to split each text file into smaller segments within a character limit. The reason that we needed to split the text files was that many of them were far beyond Azure's character limit. Also, results were inconsistent when using the character limit listed by Azure but were consistent when using a smaller limit.

azure_tts.py

This was used to create the TTS audio from the segments using Azure's API. This depends on the pre-existing Azure speech resource and requires its corresponding key.

This python script has "azure-cognitiveservices-speech" as a dependency.

azure_audio.sh

This was used to combine the audio from the segments into one audio file for each text file using ffmpeg after this script had run the previous scripts.

This shell script was intended to be run in the Z shell (zsh).

(These files are in our group's Box folder, which Dr. Beverley has access to.)

Additionally, we needed to compress the output files before uploading them to Omeka because they were too large. This was done with ffmpeg.

3. Audio Player

1. **data.js**
   Added an audio item under sections

2. **ExpandButton.jsx**
   Added the audio object to the ExpandButton function
   Created an audioUrl constant and added it to the console.log
   Added body where the player will only display if the URL exists, created the text to read above player.

3. **App.jsx**
   Added audio to the app function

4. **titles.json**
   Added audio object to each title with URL from metadata on items on the simple page on Omeka

3.1 About the player

The audio player is a function built into HTML and gives the user the option to play, pause, control the audio and adjust the playback speed. Because of this, the programming for this function was really simple and just needed to be implemented in our corresponding items on the page.

4 Sentiment Analysis

4.1 Creating Sentiment Analysis Files and Graphs

4.1.1 Tools used for Sentiment Analysis

We used VADER (Valence Dictionary and sEntiment Reasoner) along with NLTK (Natural Language ToolKit) to conduct our sentiment analysis. Vader is a light weight, open source, and accurate sentiment analysis tool.

Used mostly to analyze sentiments in social media. It is lexicon and rule-based - uses words or phrases and as well as grammatical and syntactical rules to determine the sentiment of the text. It is open sourced from MIT license and provides sentiment scores -1 being most negative, +1 being most positive, along with a compound. The compound is the overall sentiment of the text. It is calculated by summing the valence scores of each word in the text, adjusted according to the rules of the VADER algorithm, and then normalized to fall within the -1 to +1 range.

Vader, in conjunction with NLTK, can handle sentiment analysis on longer texts to decompose paragraphs, articles, publications, or novels. VADER can handle typical negations (“not good”), use contractions as negations (“wasn’t very good”), conventional use of punctuation to signal increased sentiment intensity (“Good!!!”), conventional use of word-shape to signal emphasis (“GOOD”), and using degree modifiers to alter sentiment intensity (“very”, “kind of”, etc.).

It has a large and robust dictionary of words and rules and includes the context of the grammar and punctuation in its sentiment analysis which is important when thinking about the sentiment in context to the words and grammar around it. This is more of what we had in mind from the get go and Voyant also did not offer this kind of sentiment analysis. Using VADER with NLTK allows us to do sentiment analysis on the large non-fiction works. We find that the sentiment analysis stays consistent throughout the entirety of the text no matter the file size.

4.1.2 Sentiment Analysis HTML files and Graphs

After deciding on the sentiment analysis tools to use, we created code that would run through a whole folder of .txt files, perform the sentiment analysis of each .txt file, return the words with positive, negative, and neutral or no sentiment, create a HTML file of the original .txt file with the sentiment analysis within the text, and save all sentiment

analysis HTML files in another folder. All code for the sentiment analysis HTML files can be found in by using the following link:
[Senior-Design-2025: GE's Nonfiction Audio-Visual]
The code can be found in the Sentiment Analysis folder.

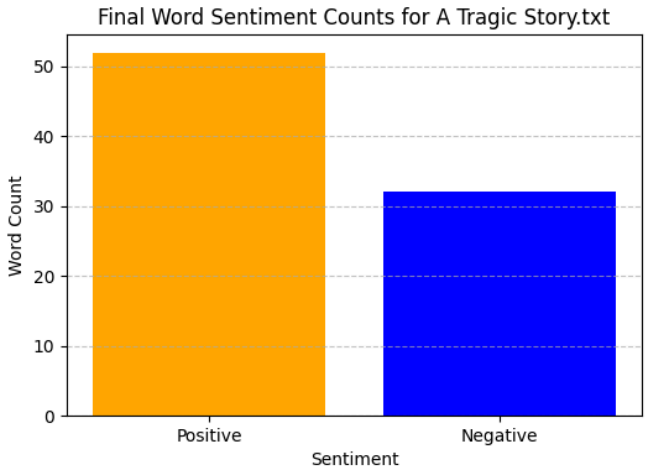
The sentiment analysis HTML files that were produced look to the following example:

A TRAGIC STORY.

Schatzkastlein des Gevaitersmanns ("The Godfather's Jewel-Casket.") by Von Berthold Auerbach. Williams and Norgate. Berthold Auerbach is so deservedly celebrated for his tales of peasant life, that we could not open a new volume of stories by him without some pleasant expectations. The principal defect of his fictions is a too predominant moralizing tendancy which often leads him to sacrifice truthful representation to the desire of enforcing a lesson. This defect is particularly conspicuous in The Godfather's Casket, where he seems divided between the purpose of writing tales about the poor which shall convey a moral to the rich, and that of writing moral tales for the poor themselves. The result is a book that is not well suited to either class. Still he is too sincere and loving a student of popular manners and character to produce a volume of fiction which does not contain some striking and truthful sketches, and there are several such in The Godfather's Casket. We select one as a specimen, not because it is the very best or most agreeable in the book, but because the hint which he intends to convey by it to wealthy German mothers might be usefully adopted by certain English mothers also. Here is the story. Christiane,

The positive sentiment is orange and the negative sentiment is dark blue. The words with neutral or no sentiment are black.

Similarly, for the sentiment analysis graphs, the same tools for sentiment analysis were used and all words with positive and negative sentiment were counted and that count was displayed in a graph for each nonfiction work, as seen below:



4.2 Displaying Sentiment Analysis HTML Files and Graphs

4.2.1 Displaying Sentiment Analysis HTML Files

To see all code for how to display sentiment analysis HTML files use the following link: Senior-Design-2025: GE's Nonfiction Audio-Visual
In the Frontend folder, the files responsible for displaying the the sentiment analysis HTML files are as follows:

Data.js - found in the src folder

Maps the titles to the html filenames.

App.jsk - found in the src folder

Maps the html files to the ExpandButton that will display the html files for each nonfiction work.

ExpandButton.jsk - found in the components folder within src folder

Displays the html sentiment analysis file in its own iframe.

Titles.json - found in src folder

Hold all nonfiction title names and the file names of all nonfiction html sentiment analysis files.

4.2.2 Displaying Sentiment Analysis Graphs

Similarly, to see all code for how to display sentiment analysis HTML files use the following link: Senior-Design-2025: GE's Nonfiction Audio-Visual
In the Frontend folder, the files responsible for displaying the the sentiment analysis HTML files are as follows:

Data.js - found in the src folder

Maps the titles to the graph filenames.

App.jsk - found in the src folder

Maps the graph files to the ExpandButton that will display the graphs for each nonfiction work.

ExpandButton.jsk - found in the components folder within src folder

Displays the sentiment analysis graphs below the sentiment analysis html iframe.

Titles.json - found in src folder

Hold all nonfiction title names and the file names of all nonfiction sentiment analysis graphs.