

CART-351-2252-A

André Neder de Almeida | 40208953

Hugo Waterfall |

Owen Hill |

Project #3 – An Explorable Networked-Space

“Journal: Mood Analyzer”

As a group, we brainstormed a lot on what to do for this project at first, and despite all the ideas that seem crazy right now, we decided to keep it as simple as possible, while making it visually appealing and interactive. Determined to work with data obtained from strings. A system in which users would type an entry and receive data stored in MongoDB but also manipulated by an API.

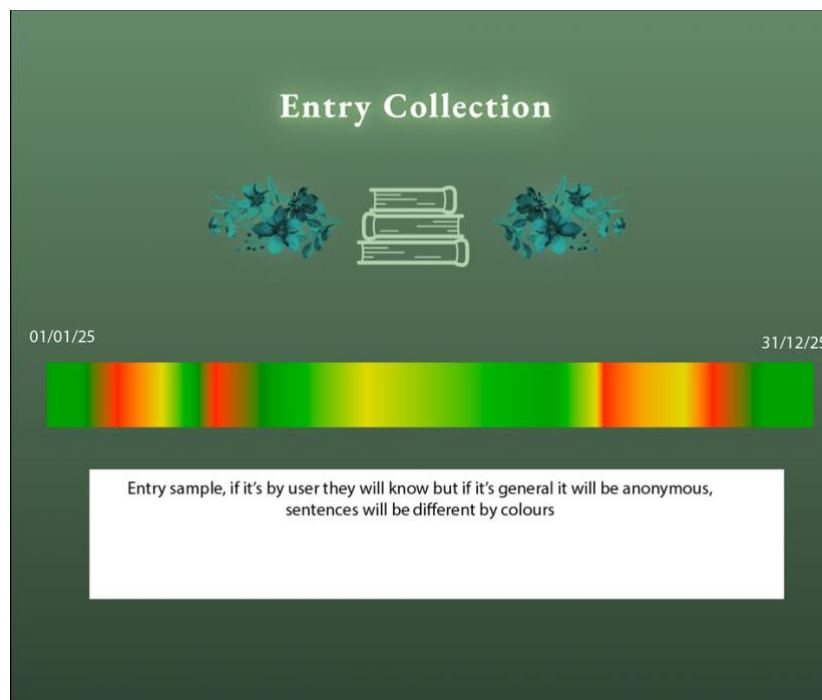
The journal idea was simple, but after revisiting ML5, an AI library for P5 JavaScript, we decided to make a system in which the user would input a journal entry and have the mood of that entry analyzed to determine how positive it is on a scale from 0% to 100%.

From there we expanded quite a bit, but the main focus was to submit and read entries. In the beginning we had our previous project as base, but soon enough decided to build up from another template, where we build the core fore of three pages: Home, Append and Collection.

Home would be simply a landing page where the user can have an idea of the website but still feel intrigued by it.

Append would be a form to make the entries and submit, and we might have overcomplicated our development by adding steps to it, but it definitely paid off since it enhanced the user experience. First the user would have an inspiring message and click on the “Begin” button to access the form, then they would type their name, entry and click submit, then the page displays the positive percentage with the option of doing another entry using the button. All these elements are already present in the HTML and are triggered to display with JavaScript events. When clicking submit, the JS and the ML5 model process the entry and assign values to the whole message as well for each sentence delimited by a comma or period. Then all this info is routed to the Python, which stores the username, an array of sentiment values and the date and time of submission. Upon submitting the Python also stores the username to a cookie in case the user wants to entry another message, that information will be already filled up. While we first planned on having more variables to display different things once a message was submitted, we decided to divert our efforts to the display of entries in the Collection page

From the very beginning, we knew that Collection was going to be the page where users could read entries, but we had no idea of how to display it. While it might have been the most confusing and unorganized part of this project, it has also worked in our favour since it resulted in a very pleasant experience to visualize entries. The result is a page that has two routes in which users can see all the entries or just the ones from a particular username. When first landing in Collection the user sees a message card and a button for all entries or their own. If clicked on the single user option, the page will display a form just to entry the username for filtering, but through either button the user will be displayed a table of a single row that works as a timeline, the cells are equally distributed and each represents an entry and has a different colour depending on the positivity level of that entry. On top of the table there are the dates of the first and last entry to left and right, at the center there is also the date of the entry when hovered over. When an entry is selected the text with the positivity level and the text box with the entry is displayed, the text box is white with an inner shadow of the same colour attributed to the entry and the text has different fonts depending on the positivity of each sentence. The data is retrieved with Python and routed with IDs to JavaScript, where it is processed and parsed. The colour is mapped from red to yellow and green, where red is 0%, yellow is 50% and green 100%. The font is also assigned depending on the level of each phrase. Our confusion with development of this page was mainly due to the fact that we knew what had to be displayed but due to a busy end of the year and schedule conflicts different group member worked on the same task and files, resulting in the some ideas being put aside to present a solid project but later implemented giving Collection a nice charm.



Early sketch of Collection using Adobe XD

The scope of this project seemed a lot larger and almost unattainable due to all the complications that a student can have at the end of a semester, but we were able to coordinate and each work in their own time to accomplish tasks and after each commit on GitHub, the project took a better shape. It was hard sometimes to work on files that another person was working so we wouldn't override each other's changes, but we got around that by using different files which sometimes were duplicated from the main one that would be integrated.

ML5 might have been easy to implement and a great way to measure the sentiment of texts, but the model showed its age and lack of updates by not giving reliable results at times, producing some numbers with a lot of decimals or even very positive or negative reviews when the text was clearly showing the opposite.

Overall, this project represents a balance between experimentation and restraint, combining AI-driven analysis with a clear focus on user interaction and data visualization. Despite technical limitations, such as ML5's occasional unreliable sentiment readings, and the logistical challenges of coordinating work at the end of the semester, we were able to adapt and refine our approach. Through iteration, collaboration, and careful integration of our individual contributions, the project evolved into a cohesive system that successfully connects user input, machine learning, and visual feedback, while also reflecting the realities and compromises of working on a complex group project.

Most of our planning was mostly done with google docs for live sharing:

https://docs.google.com/document/d/12XXb9qSrXZeEwKFECrPOhi8s_V4sHmYJYxbxEUe1_Q/edit?usp=sharing