A Visual Dictionary: Mapping Creole Variation

Alana Reyes
University of Maryland, Baltimore County Baltimore, US
alanar1@umbc.edu

Aleem Prince
University of Maryland, Baltimore County Baltimore, US
do71730@umbc.edu

Jeremy Singh
University of Maryland, Baltimore County Baltimore, US
jeremys1@umbc.edu

***Abstract* — Due to the lack of research done on Creole languages, there is also a lack of visualization tools. Because of this absence, our group has decided to develop an interactive map of the various Creoles worldwide that allows us to compare and contrast the appearance of words or phrases in various languages. After developing this tool, we presented it to our client, Dr. David Beard, an assistant professor of Linguistics at the University of Maryland Baltimore County, who gave us feedback.**

I. INTRODUCTION

Creole is an amalgamation of one or more languages created out of necessity. Most of the Creoles in the dataset were created during the Transatlantic Slave Trade. Creoles started as Pidgins, however, when passed down to a younger generation as the native language, Pidgins became Creoles.

Creoles exist in an environment where another language exists. Because of this, one language has a higher prestige than the other. The language with the higher prestige is called the Acrolect, while the lower prestige language is called the Basilect. The Acrolect is the standardized language; it is often used in formal settings. The Basilect is the Creole language itself. Oftentimes, individuals will mix the Basilect and Acrolect. This is called the Mesolect. Due to this, Creoles have variations which can be documented on a Post-Creole Continuum.

Creoles are an interesting linguistic phenomenon. Unfortunately, there is not much research done on Creoles. This also means that there are not a lot of visualizations. To fill this niche, our team developed various visualization tools that would allow us to compare and contrast various Creoles. These tools included an interactive map, pie charts, and bar charts.

These tools were developed with a client, Dr. David Beard, an assistant professor in the Linguistics Department at the University of Maryland Baltimore County. During different stages of the development process, the project was shown to Dr. Beard, who gave us feedback on improving our project, visualization and location of additional datasets.

The purpose of this paper is to describe our tools and their creation processes.

II. RELATED WORK

The main inspiration for our project comes from the *Atlas of Pidgin and Creole Language Structures Online* (APiCS) [1] visualization tool, which shows Creoles around the world. This visualization is an interactive map that has a colored dot that shows where a Creole is in the world. The colors of the dots represent a specific Creole's Lexifier (for example, English-based, Spanish-based, or Dutch-based). When you hover over these dots, it will show you the name of that specific Creole. You can also filter the Creoles based on their lexifier, show and hide the language names, and change the size of the dots to make them bigger or smaller. Underneath the map is a table that shows all the Creoles currently present on the map, the language's Lexifier, and their location.

Our project differs from *APICS's* website [1] even though it includes an interactive map. In our project, the user can search for an English translation/Acrolect or Creole word/phrase, and the map will show indicators of the locations where the search can be found. By hovering over the indicators, a pop-up will appear that will show additional information, such as the name of the Creole spoken in that location, the Creole's Acrolect, the Creole's words or phase, etc.

The user can also view additional language information on the static visualization page. This includes the percentages of each language in our dataset, letter n-grams (the sequences of letters in a word), etc.

III. IMPLEMENTATION

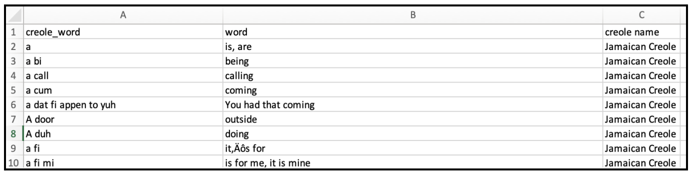
In this section, we will go through the steps we took to develop this project. This project was made using the following libraries: *Leaflet* version 1.7.1 [2], *Python* version 3.9.10, and *Flask* version 2.0.2 [3].

B. The Dataset

It has proven to be fairly difficult to find Creole dictionaries. Some Creoles have and still face many issues related to their orthography or spelling systems. Even though Creoles are developed grammatically, phonetically and verbally, they are not developed morphologically—everyone has their spelling. During the beta release, we managed to find a few online dictionaries for some of the more researched languages: Haitian Creole [4], Jamaican Creole [5], and Louisiana Creole [6]. Later, we found additional dictionaries for Antillean Creole [7], Martinique Creole [8], and Nigerian English Pidgin [9] — even though Pidgin is included in the official name, this is a Creole —, and Saramaccan [10].

To retrieve the data from these dictionaries, we scraped the data from their respective websites, cleaned them, and organized them so they could be fed to the front end. This was mainly done using *Python*, and some of its libraries including:

1) *Requests*[11] - To retrieve the raw HTML.
2) *Re (regex)* [12] - To parse and clean the data from the HTML.
4) *BeautifulSoup* - To assist in parsing the raw HTML data [13].
3) *Pandas* [14] - To create DataFrames to store all the language data and export it to CSV and JSON files with the following columns:
   - **Creole_word** - This column contains the words or phrases from the Creole languages.
   - **Word** - This column contains the English words or phrases that correspond to the data entry in the *Creole_word* column.
   - **Creole_name** - The name of the language for the *Creole_word*.



	A	B	C
1	creole_word	word	creole name
2	a	is, are	Jamaican Creole
3	a bi	being	Jamaican Creole
4	a call	calling	Jamaican Creole
5	a cum	coming	Jamaican Creole
6	a dat fi appen to yuh	You had that coming	Jamaican Creole
7	A door	outside	Jamaican Creole
8	A duh	doing	Jamaican Creole
9	a fi	it‚Äôs for	Jamaican Creole
10	a fi mi	is for me, it is mine	Jamaican Creole

Fig. 1. Show a section of the Jamaican Creole CSV.

C. The Back End

The backend is built using a combination of *Leaflet [2]*, HTML, and the *Flask [3]* framework in *Python*. Its two main purposes are to take in the user input from the front end, query the dataset, and then process the data into a workable display format for the different pages on our website.

The website uses three pages: The homepage, the static images page, and the map page. The homepage is solely HTML based. The images page is also made up of HTML with some *CSS* [15] (formats the pie chart and word cloud next to each other). For *Flask* [3] to recognize and accept the images, they were stored in a folder called "static" in the website's root directory. The map page is the only page that uses *Javascript* and a combination of in-line *Javascript* and *Javascript* files—"universal.js", located in the "static/js" as instructed by *Flask [3]*. Universal.js creates the map, defines a function for adding information to the markers appearing on the map, and adds the necessary legalese related to using *OpenStreetMap* [16]. The in-line Javascript, along with HTML, defines a layer for the markers to appear on and then populates the map with markers with the correct features defined in universal.js.

The *Flask [3]* framework that forms the backbone of the web application is defined in "website.py". Website.py defines the homepage, the images page, and the map page. The homepage and image page definitions return the corresponding HTML to be rendered to the screen. The map page tracks the input sent from the homepage and determines whether it came from the English/Acrolect or Creole input. Then, the query is sent to the appropriate function defined in "read_data.py".

Read_data.py contains the entirety of the database querying and is where the query gets formatted into the appropriate format for the front end. The data structures for each Creole language are associated with the following attributes: the corresponding CSV file location, the corresponding Acrolect, and the latitude and longitude where that Creole language is spoken. If the Creole language appears in more than one country, the coordinates for the additional countries are also included. These points were generated using an existing online site that helps visualize the GeoJSON file format [17].

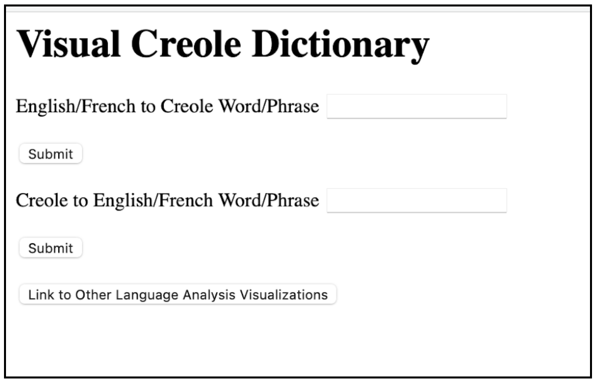
For all queries, all the CSV files are parsed to look for matches in each Creole's CSV. Even after cleaning the datasets after scraping them from the web, the string inputs needed to be parsed as lists [18]. The results are then formatted into appropriate GeoJSON [19] data, a special JSON format specifically for map visualizations and is then read by the frontend using code defined in "universal.js" [20].

D. The Front End

The main crux of this project lies in the front end, as it contains the entirety of what the user sees when interacting with the project. Ease and clarity of use were of the uppermost priority, and to that end, this project aims to use very clear user interfaces that should be familiar to anyone that has used modern websites. The front end was created using *Leaflet [2]*, an open-source Javascript library, and *OpenStreetMap*s [21].

This project was built and ran on a local machine which required the user to install Flask and *Python*. In the past, we recommended using a machine with a Bash terminal to run the application with easy access. However, since then, we have found a web hosting site, pythonanywhere.com [22]. This site successfully ran the code for the project with very little adjustment. It is recommended that users access the website using a desktop or laptop computer as this website is non-responsive.

Once the website is loaded, the user can enter a word into one of two sections. One section is for English translation/Acrolect words, which is used to find the corresponding word in each of the available Creole languages. The other section is for Creole word search, which allows the user to find the corresponding Creole languages for the Creole word.

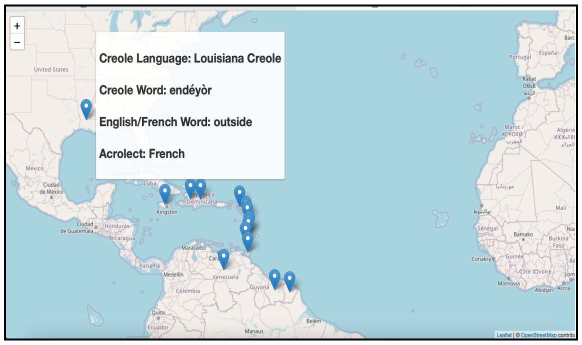

Visual Creole Dictionary

English/French to Creole Word/Phrase

Creole to English/French Word/Phrase

Fig. 2. The User input Page. The user can search for either an English Translation/Acrolect or Creole word/ phrase.

After hitting the submit button under the search bar, the program traverses the datasets where the data is stored. If a match is found, the user is taken to a page with a map of the world with one or more points marked on the map. The user can view more information about the word by hovering over one of the points. They will see the name of the language, the word, the Acrolect, and its definition. If the query yielded no results, the user will not be able to see any points on the map and will have to go back to the homepage and enter another string. While on the map, the user can zoom in and out, if they need to adjust the size of their view. To pan around the map, the user must click and hold and then drag across the screen.



Creole Language: Louisiana Creole

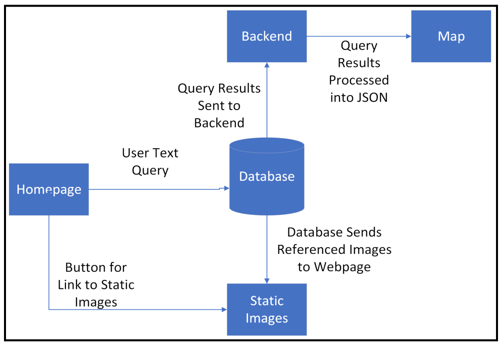
Creole Word: endéyòr

English/French Word: outside

Acrolect: French

Fig. 3. The interactive map after a search for the English word "Outside". The results show the location of where the word can be found. The pop-up from hovering over one of the dots shows more information, including that the Creole identified is Louisiana Creole and the Creole word for the searched word "endèyòr".

The last button is a link to the static visualizations that were created to compare the data we had gathered together. An overview of the project can be found in Fig. 4.



Backend
Map
Query Results Processed into JSON
Query Results Sent to Backend
User Text Query
Homepage
Database
Database Sends Referenced Images to Webpage
Button for Link to Static Images
Static Images

Fig. 4. Shows a summary of the flowchart for our website [23].

E. Static Visualizations

The static visualizations give more insight into the Creoles and how they compare. These visualizations, as the name implies, are not interactive.

The first visualization that we generated was a Word Cloud. The word cloud, *Fig. 5,* shows the most common words across all the Creoles in our dataset. The bigger the word appears in the image, the more frequent it is in the dataset.

The second visualization we generated was a pie chart. This pie chart, *Fig. 8,* depicts the distribution of the different Creoles in our dataset. With it, we could show the percentages of each Creole that made up our dataset.

The next visualizations we created were bar charts showing the frequency of various character N-grams. N-grams are a structure commonly used in NLP [24]. Essentially, N-grams in our use case consist of groups of N consecutive characters in our words. We decided to look at bigrams (2-grams) and trigrams (3-grams). We plotted two bar charts for bigrams. The two charts showed the ten most frequent and least frequent bigrams for a specific Creole and showed the count for each of those bigrams. The same information was encoded for trigrams. One of these plots is seen in *Fig. 6.*

The last set of visualizations that we generated builds off of the bar charts mentioned above. We generated two stacked bar charts, one for bigrams and one for trigrams. These two charts show our dataset's most frequent bigrams and trigrams. The bars were then segmented to show how much of that data came from each Creole, with each color corresponding to the colors used for each Creole's bigram and trigram charts. One of these stacked bar charts is depicted in *Fig. 7.*

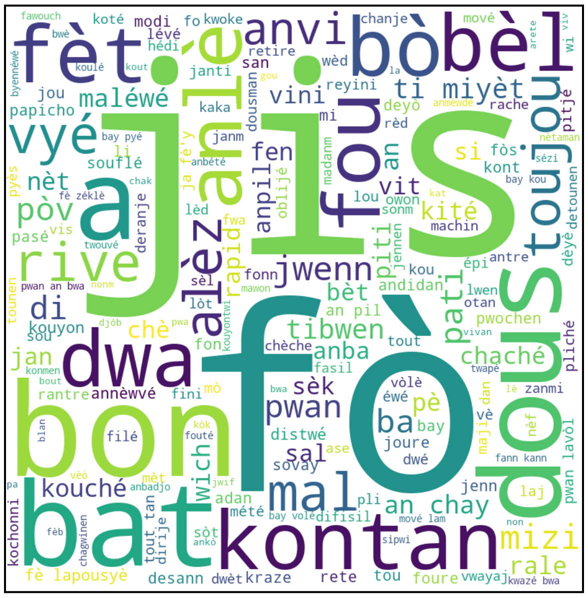


Fig. 5. The word cloud shows our dataset's most common words.

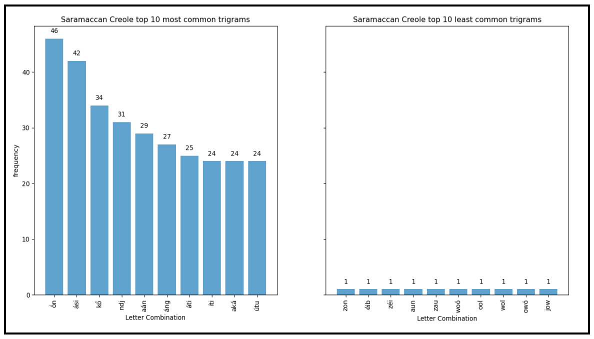


Fig. 6. The bar chart shows the ten most common and least common trigrams for Saramaccan.

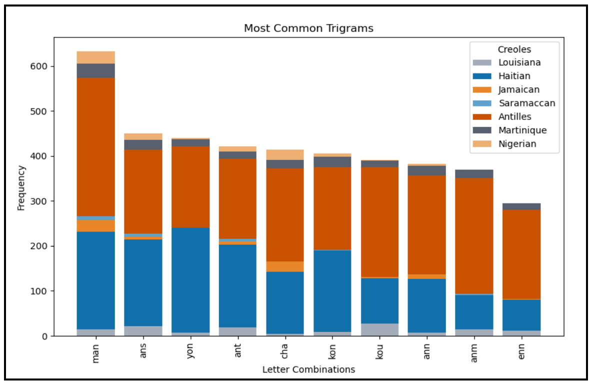


Fig. 7. The stacked chart shows the most common trigrams.

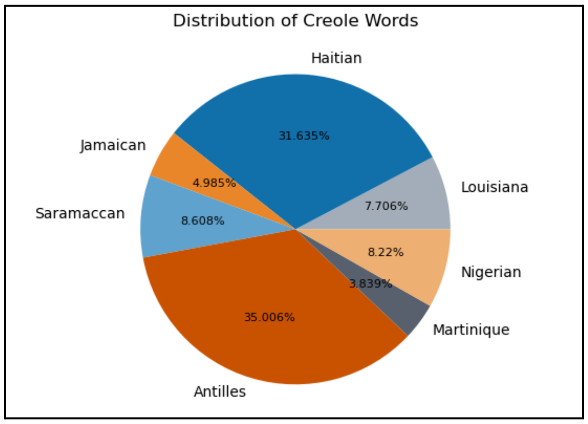


Fig. 8. Pie Chart shows the Distribution of Creoles in our dataset

IV. RESULTS

A. The Map

The final result of our project came in two parts. The first part was the interactive map. The data shown on the map is a filtered form of our dataset. It is based on the user's search query on the home page. We used a spatial region channel encoding by placing markers on the locations on the map where the user's search appears [25]. We did this because it aligned with our original plan to compare where certain words or phrases appear in the world. Using spatial regions made it easy to compare this information on a world map and in a single frame. The colors on the map do not encode any information. All the marks on the map are colored blue, the default color, so no color map was selected.

B. The Static Visualizations

The word cloud encodes a word's appearance frequency using area. This is the standard way word clouds tend to encode data. We are not interested in seeing how often a certain word appears compared to others. We just want to see what some of the more frequent words are relative to the rest in the dataset. There is no color encoding in this visualization, so no specific colormap was chosen. The colors chosen were picked because they are visually appealing together.

We chose a categorical segmented colormap with seven bins for all the other static visualizations generated [25]. The goal for these visualizations is to compare and contrast the different Creoles. The information encoded for the colors was consistent across these visualizations. The color encoding primarily uses hue thus, it does not imply order [25]. However, to keep the colors complementary and ensure no one category stands out, we added luminance to our colormap. Since we still use various hues, our visualizations do not inadvertently imply the luminances as ordered data [25].

The bar charts we used are the standard bar charts where the marks are the bars and encode the count of each bigram and trigram by length [25]. The taller the bar, the more frequently the n-gram occurred. We chose to encode the frequency of the bigram and trigrams using length because they are one of the most effective ways to encode magnitude [25]. For the regular bar charts (not the stacked bar charts), there is also a common scale (all the bars start at 0) to make it easier to compare the heights of the bars [25]. The bars were also ordered on the x-axis from highest to lowest frequency to make the differences easier to see.

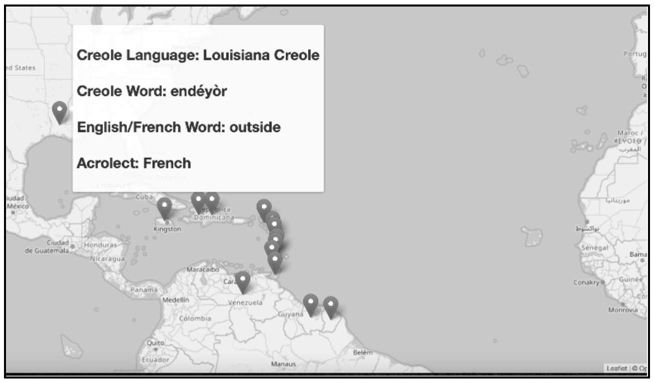
For the stacked bar charts, the frequency of each bigram and trigram is encoded by length and the bars are also ordered on the x-axis in descending order [41]. However, unlike the regular bar charts, there is one more level of encoding. Color is used to encode the different Creoles. Though the frequency of the n-grams for each Creole is encoded using length as well, the scales are unaligned, making it very difficult to compare the Creoles to each other within the bars themselves and across the bars [41]. Despite this, we chose the stacked bar chart because the goal was not to directly compare the frequency for each Creole. Instead, these visualizations aimed to see what bigrams and trigrams were most frequent among the Creoles, and then see the distribution of each Creole in those counts. This is similar to what we want to show in our pie chart.

With the pie chart, we want to depict the percentage distribution of our entire dataset. This was done by taking the count of the number of words in each Creole in our dataset. These numbers are encoded using area. This encoding is satisfactory for our needs as we just want to see a broad overview of the distribution of the Creoles. For more specifics, we have each slice of the pie labeled with a percentage.

C. Validation

To test our visualization tool's effectiveness, we met with our client to get his input and feedback. Overall, he was very excited about our tools and what it shows. He appreciated the idea we had about using the map to show the location of the Creoles around the world. He also liked the use of bigram and trigram frequencies across the languages. One thing that he suggested that would be interesting to see would be Phonemic charts and graphs. Phonemic charts are tabular visualization showing how a word is spelt and contrasting it with how it is pronounced. Dr. Beard suggested we look at these charts and make plots similar to our bigram and trigram charts. However, due to the nature of our data, it would be fairly difficult to accomplish this. Most of the dictionaries we parsed did not have pronunciation for words.

We also wanted to test our project for accessibility, specifically, color blindness. We tested each of our different pages and static visualizations against a colorblindness visualization website[26] for each type of colorblindness. We wanted to show off the results for a monochromatic and a green-weak version of our visualization. Monochrome was chosen, because if it was still understandable in monochrome, that would bode well for its viewability for other types of colorblindness. Green-weak was chosen as it is the most common form of colorblindness.

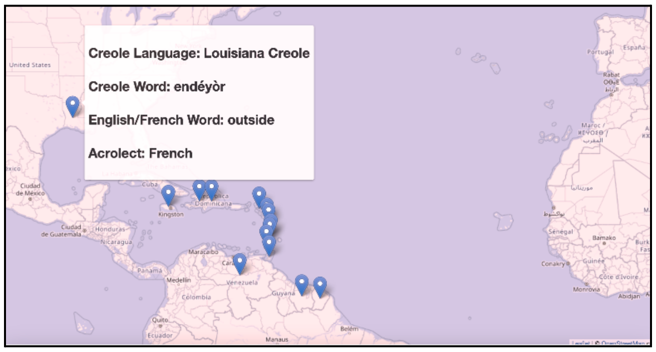

Creole Language: Louisiana Creole

Creole Word: endéyòr

English/French Word: outside

Acrolect: French

Fig. 9. Monochrome version of Fig. 3.


Creole Language: Louisiana Creole

Creole Word: endéyòr

English/French Word: outside

Acrolect: French

Fig. 10. Green-Weak/Deuteranomaly colorblind version of Fig. 3.

D. Scalability

This visualization tool is fairly scalable, as long as the CSV files are in the right format, we can add many additional languages. However, the performance may decrease as it takes longer to search each CSV for the user's query.

V. FURTHER DIRECTIONS

We would like to do many things to expand on this project. Besides trying to find more data (as stated above, it has proven very difficult to find Creole dictionaries), some of the expansions we would like to implement are:

- A tool that allows the user to compare the similarities between certain phrases across the various Creole languages.
- Bar charts that show many Creole languages fall under each Lexifier (i.e. how many languages are French-based, how many are English based, etc.)
- A timeline that shows the estimated time the languages first appeared.

In future, we would like to color code the markers to correspond with the different Creoles in our dataset. This will allow the user to see the different languages without hovering over the marker.

VI. CONCLUSION

Our project encapsulates our original goal -- a visualization tool for Creole languages. With the help of our client, Dr. Beard and Data Visualization professor, Dr. Engel, we accomplished this goal by creating an interactive map and different static visualizations.

VII. REFERENCES

[1] E. O. Aboh et al., “Atlas of Pidgin and Creole Language Structures Online,” *APiCS Online* -. [Online]. Available: https://apics-online.info/ . [Accessed: 05-Oct-2022].

[2] “An open-source JavaScript library for interactive maps,” *Leaflet*. [Online]. Available: https://leafletjs.com/. [Accessed: 07-Nov-2022].

[3] “Welcome to Flask,” *Welcome to Flask - Flask Documentation (2.2.x)*. [Online]. Available: https://flask.palletsprojects.com/en/2.2.x/#user-s-guide. [Accessed: 07-Nov-2022].

[4] “Haiti 1804 Kreyol,” *Kreyol*. [Online]. Available: https://kreyol.com/.

[5] Jamaican Patwah, “Jamaican patois and slang dictionary,” *Jamaican Patwah*. [Online]. Available: https://jamaicanpatwah.com/dictionary.

[6] “Browse Louisiana Creole – English,” *Louisiana Creole KouriVini Dictionary*. [Online]. Available: https://www.webonary.org/louisiana-Creole/browse/browse-vernacular/. [Accessed: 12-Dec-2022].

[7] “Parcourir Le Kwéyòl,” *Dictionnaire Kwyl*. [Online]. Available: https://www.webonary.org/kweyol/browse/kweyol-English/?lang=fr. [Accessed: 12-Dec-2022].

[8] F. Palli, *Page d'accueil*. [Online]. Available: https://www.potomitan.info/dictionnaire/francais.php. [Accessed: 12-Dec-2022].

[9] “The Nigerian pidgin English dictionary.,” *Naijalingo*. [Online]. Available: http://naijalingo.com/. [Accessed: 12-Dec-2022].

[10] *Saramaccan - English Interactive Dictionary*. [Online]. Available: http://suriname-languages.sil.org/Saramaccan/English/SaramEngDictIndex.html. [Accessed: 12-Dec-2022].

[11] “Quickstart,” *Quickstart - Requests 2.28.1 documentation*. [Online]. Available: https://requests.readthedocs.io/en/latest/user/quickstart/.

[12] “Re - regular expression operations,” *re - Regular expression operations - Python 3.11.0 documentation*. [Online]. Available: https://docs.python.org/3/library/re.html.

[13] “Beautiful Soup documentation¶,” *Beautiful Soup Documentation - Beautiful Soup 4.4.0 documentation*. [Online]. Available: https://beautiful-soup-4.readthedocs.io/en/latest/. [Accessed: 12-Dec-2022].

[14] “User guide,” *User Guide - pandas 1.5.1 documentation*. [Online]. Available: https://pandas.pydata.org/docs/user_guide/index.html#user-guide.

[15] “W3Schools online HTML editor,” *W3Schools Online Web Tutorials*. [Online]. Available: https://www.w3schools.com/howto/tryit.asp?filename=tryhow_css_images_side_by_side. [Accessed: 12-Dec-2022].

[16] *OpenStreetMap*. [Online]. Available: https://www.openstreetmap.org/copyright. [Accessed: 07-Nov-2022].

[17] Mapbox, “Powered by Mapbox,” *geojson.io*. [Online]. Available: https://geojson.io/#map=2/0/20. [Accessed: 12-Dec-2022].

[18] *Stack Overflow*, 12-Dec-2009. [Online]. Available: https://stackoverflow.com/questions/1894269/how-to-convert-string-representation-of-list-to-a-list. [Accessed: 12-Dec-2022].

[19] “Geojson,” *GeoJSON*. [Online]. Available: https://geojson.org/. [Accessed: 12-Dec-2022].

[20] “Passing variables from flask to JavaScript,” *Stack Overflow*, 16-May-2016. [Online]. Available: https://stackoverflow.com/questions/37259740/passing-variables-from-flask-to-javascript. [Accessed: 12-Dec-2022].

[21] S. McNeal, “How to make a web map with Python's flask and leaflet,” *Medium*, 20-Jan-2022. [Online]. Available: https://medium.com/geekculture/how-to-make-a-web-map-with-pythons-flask-and-leaflet-9318c73c67c3. [Accessed: 12-Dec-2022].

[22] P. A. LLP, “Host, run, and code python in the cloud!,” *PythonAnywhere*. [Online]. Available: https://www.pythonanywhere.com/. [Accessed: 12-Dec-2022].

[23] “Flowchart Maker and diagramming software: Microsoft visio,” *Microsoft*. [Online]. Available: https://www.microsoft.com/en-us/microsoft-365/visio/flowchart-software. [Accessed: 12-Dec-2022].

[24] “Generating n-grams from sentences in python: Albert Au Yeung,” *Generating N-grams from Sentences in Python | Albert Au Yeung*. [Online]. Available: https://albertauyeung.github.io/2018/06/03/generating-ngrams.html/. [Accessed: 12-Dec-2022].

[25] Munzner, T. (2014). Visualization Analysis & Design. CRC Press, Taylor & Francis Group.

[26] “Coblis - Color Blindness Simulator,” Colblindor. [Online]. Available: https://www.color-blindness.com/coblis-color-blindness-simulator/. [Accessed: 05-Oct-2022].