Group Members:
Felicia Wang (felicia5) - Captain
Phyllis Wang (pjwang3)
Eshia Rustagi (eshiafr2)

# Progress Report

For progress we've made so far, we created the initial project structure for the extension. This includes the manifest.json file, required for all chrome extensions which contains basic metadata (name, permissions, content scripts, etc.) about the extension. popup.html, css, and js files, will contain all of the frontend design and functionality which the user will interact with. The final file, content.js, aims to have the BM25 algorithm as well as various other functions that will help parse Wikipedia articles and PDFs. We added functionality for the extension to look at current tabs and only work on Wikipedia pages based on the URL, and output a message on the extension if it's not a Wikipedia page. If it is a Wikipedia article, we access the webpage's HTML to retrieve all the text in paragraphs when a user clicks on the extension. We also implemented the function that takes all Wikipedia paragraph text, filters out the stopwords, and stems the remaining list using a Porter Stemmer algorithm written by different authors based on a paper about the Porter Stemmer. We also added a form to search query in extension popup, and created functions for getting vocabulary and frequency of words per paragraph, which will be used for finding inverse document frequency as part of BM25. Finally, we worked on combining filtering and stemming logic with the vocabulary and document frequency logic.

In terms of remaining tasks, we plan to integrate the created functions with other remaining functions required to calculate BM25. We also need to figure out how best to return query results back to the user, such as in a table in the popup or some other feature. We want to have some way of linking results to places on the page so that users can click on a link to find the place on the page the result is, and make the popup look better. We had initially planned for our extension to work on any PDF document, as well, and so we will need to look at how to parse PDF text, possibly using the PDF.js library, which helps with parsing and rendering PDFs. If we do use a library, we aim to understand how the library works and what functions we can use to successfully extract the text and perform BM25 search.

Finally, for current challenges, we found it difficult understanding the scope of files, and were unsure how to make global variables within files (such as initially wanting a global stopwords array) that can be accessed outside the file, and running into errors when trying to import packages in files listed as modules. We were also wondering if the list of stopwords obtained is "good enough". Although obtained from NLTK, the list comes from several years ago and there are much more exhaustive but varying lists on different websites, which makes us unsure of which to use. Using more exhaustive lists may yield more accurate results, but will be

significantly slower to run. We want our extension to work on PDF files, but it's harder to figure out how to detect a PDF file and read the text, and so we tabled that for now as we may need to look into external libraries, like PDF.js or file-reading. Sometimes we found it difficult to know what is feasible and what isn't in terms of the extension because of our unfamiliarity with it. Generally, understanding how to build a chrome extension and use the chrome APIs has been challenging, but we have learned a lot in this process.