Women in STEM Progress Report

Kathleen Barta	kbarta2@illinois.edu
Christine Zhou	xizhou4@illinois.edu
Sharon Lin	xinyiyl2@illinois.edu
Kris Kamaei	mkamaei2@illinois.edu
Payel Chakraborty	payelc2@illinois.edu

Progress:

- Build a browser extension
    - Created initial chrome browser extension with search bar
- Build a scraper
    - Code to pull pages based on a given string search
- Research a bookmark feature
    - Researched technical implementation, using port messaging with chrome extension and an external Firebase database to save pages for a user
- Build an index and ranking
    - Code to index and search a dataset is complete using Faiss FlatIndexL2
- Documentation
    - Created initial README.md for instructions on using the tool.

Tasks To Do:

- Build a modal that will display results
- Build bookmark feature
    - Code the frontend and backend of the bookmark feature.
    - Integrate with other parts of the chrome extension.
    - Set up external Firebase databases and integrate with the bookmark feature with chrome extension
- implement bookmark feature
- Update documentation
- Test end to end and integration
    - Hosting the backend code for the frontend to access.
    - Integration with scraper and extension
- Create a presentation + tutorial

Challenges:

- Hosting on cloud needs more work than expected
- Web scraper: dynamic coursera webpages only return partial html return
    - Library requests work better on static web pages than dynamic
    - Selenium only would return login page
    - 2fa authentication caused issues with scraping UIUC MCS specific pages
    - TA suggested using websites that do not involve authentication

Team Member	Progress
Overall	Completed Tasks: Our team has delegated specific roles and responsibilities for the development of the extension Tasks left: Individual completion, integration and documentation
Kathleen Barta	Created initial web extension files for a popup extension with search bar. Wrote readme file with instructions on how to use the extension. Tasks left: Incorporate back end search results into extension Challenges: How to integrate python code into web extension - most likely will use PyScript
Christine Zhou	
Sharon Lin	Completed tasks: Researched and looked into technical implementation for the bookmark feature for the chrome extension. This includes the communication within the chrome extension using port messaging and with an external database (Firebase) to add and remove course related pages for a specific user. Tasks left: Code the frontend and backend of the bookmark feature Integrate with other parts of the chrome extension. Challenges: No major blockers.

Kris Kamaei	Completed tasks: Code to index and search a dataset is complete using Faiss FlatIndexL2 Tasks left: Hosting the backend code for the frontend to access. Integration with scraper and extension Challenges: Hosting on cloud needs more work than expected
Payel Chakraborty	With a given search string, the code is able to pull the pages that get returned on a search on TED. Have to integrate this with the other pieces. There were multiple hurdles that I encountered during the web scraper implementation. To start with I inspected the nature of the coursera web page. I found out that it was a Dynamic Webpage. For context, a dynamic webpage is dynamically rendered by the client through embedded javascript instead of pulling the entire html from the server. So, when I tried scraping the dynamic webpage, only a partial html got returned through the 'requests' library. The most common library 'requests' works excellently with static web page but falters with dynamic webpage. On further investigation I found out that the libraries requests-html and selenium are able to render the entire html by executing the javascript in the backend. I tried scraping the coursera page using selenium, but it would only return the payload of the login page instead of the actual page. That was hurdle 2 , doing the authentication. However, I tried using the username/password combination by using it within the url and also created a config file with the credentials to use it while calling the selenium webdriver. None of them worked, then I realized

	that the website has 2fa authentication and selenium fails with such 2fa authentication systems. Finally, I contacted the TA and he suggested using websites that wouldn't require such authentication, such as Ted Talk.