**Understanding Our Project: Objectives and Functionality**

Our project is designed to solve the frustration users face with Depop being overrun by resellers, by offering a script that leverages web scraping tools such as Selenium and Beautiful Soup to find and filter actual deals. Users would be given the option to enter two categories with an optional third one. Our goal while creating the application was to help enhance user experience and find actual bargains.

**Getting Started: Running the Program**

Getting started is quite simple, we designed the program while prioritizing simplicity and the web scraping aspect. The application ensures that running the application is straightforward and hassle-free. For a user to use our script, they would begin by specifying the Python version, then write the file path, followed by the category, men or women, and a max price point. Additionally, a user is also given the option to specify the size needed as an optional argument. This was done intentionally since not all Depop listings offer a size, which could be shoe size, pants size, or traditional sizing. An example of the script being run would look like this, “python depop.py mens 100 –size L”.

**Program Output: Analysis and Interpretation**

Using the program is easy; it is as simple as running the command line and then the user would be prompted with all the outputs in the terminal. Additionally, the user would also be given the outputs in a text file which is created in the same path as the script. When viewing the outputs it would be difficult for one to be confused. The output begins with the date and time the listings were scraped so when viewing back everything it would be clear when each listing was pulled. Followed by the date is the listings, formatted with a clear showcase of the item link, item price, and item size if present, then a divider, and this same format would repeat for all the other items as well.

**Annotated Bibliography**

Depop. (n.d.). *How Depop ranks search results and recommends listings.* Retrieved from <https://depophelp.zendesk.com/hc/en-gb/articles/9422984899985-How-Depop-ranks-search-results-and-recommends-listings#how-depop-ranks-search-results-and-recommends-listings-0-0>

Using the article published by Depop on how Depop ranks search results and recommends listings, we were able to understand Depop’s search algorithm. With this knowledge, we implemented an incognito mode option when automating web browsing. This feature allows the script to view listings which wouldn’t be relevant to one’s past search history, ensuring the users can find actual deals instead of items they have previously viewed.

Bhattacharjee D. (2021). *How to open browser window in incognito/private mode using python selenium webdriver?*  Tutorialspoint. Retrieved from <https://www.tutorialspoint.com/how-to-open-browser-window-in-incognito-private-mode-using-python-selenium-webdriver>

Using the article by Tutorialspoint, we learned how to use the ChromeOptions class in python. After creating the ChromeOptions object using the class we would then have to add a simple incognito parameter to open our web browser in incognito/private mode.

Robot Framework. (n.d.). *SeleniumLibrary.* Retrieved from <https://robotframework.org/SeleniumLibrary/SeleniumLibrary.html>

Using the public SeleniumLibrary we were able to learn more about web automation, and how to scrape dynamic web pages. The JavaScript aspect made it impossible to scrape the listings by using simple tools such as Requests and Beautiful Soup only which one would use for static web pages.

Beautiful Soup. (n.d.). *Beautiful Soup Documentation*. Retrieved from <https://beautiful-soup-4.readthedocs.io/en/latest/>

Using the Beautiful Soup library, we found information about how to parse an HTML document. We used the library to learn how to pull specific items with certain elements, making the web scraping easy to navigate.