ARGYLE DOCUMENTATION

Methodology

For the login process and site navigation, I used the Selenium library, which is an easy-to-use and maintainable tool, which is also capable of rendering pages containing JavaScript and we need it to browse Upwork platform.

For data scraping, I used the Beautiful Soup library. When navigation is on pages with objects to scrape, the page is translated into a Beautiful Soup object. From there, it is possible to extract text, values and other elements from the HTML code of the page.

Although Selenium is capable of extracting the data, it is slower as each command goes through the JSON wire HTTP protocol and there is a substantial overhead.

I added some settings in the Selenium driver to improve the performance, first I put the headless parameter that makes Selenium run without user interface, saving memory and cpu. To avoid errors when running in a docker container, I disabled the gpu and dev-shm-usage, since chrome 65+ no longer need shm.

I stored the login's information in a SQLite database just to avoid user credentials in plain text on the code.

I did some actions to avoid robot detection . I created a small list of user agents, the user agents are changed in each iteration, where each interaction the script extracts data from a user. I added a small random sleep time to simulate common user interactions, such as time for the user to enter login and password and to change pages.

Improvements

It is possible to use Selenium with parallel computing libraries, such as Joblib, to access multiple users in parallel, reducing the time to scrape data from thousands or millions of users. To run at this scale, we might need to use some proxy rotation service to evade robot detectors.