Team Avocado
Ka Ming Chan: kmchan2 (Captain)
Iris Fu: xingfu2

Project Proposal

For the final project topic of course CS 410, Team Avocado decided to develop a web application which can extract physical addresses information from an input website link. The application will crawl an input website and parse the addresses. If capacity is available, the parsed addresses can be visualized on a map. A use case of the application would be, for example, for a user to extract the addresses from a web page containing a list of great restaurants in Seattle, and take notes of the addresses for their convenience. Oftentimes in websites like Timeout or Yelp, users need to flip through long articles in order to get the address they need, and this application simplifies the action. To evaluate the web app, we could compare the extracted information and the website's content manually.

The team plans to use Python for the backend of the project, and JavaScript for the frontend. At this stage, the team is planning to use Flask for the backend framework and React as the frontend framework.

Since neither member has extensive web development experience, we need 14 hours to get a basic web application running including the frontend and the backend. This includes the time cost for the team members to pick up the techniques and tools for web app development, like React and HTML, set up the environment, and write and run the code. Then we need 10 hours to develop a crawler for the website provided which includes searching and testing effective crawling libraries, and integrating them into the app. Intuitively, address parsing may be easy to do using regular expression matching, and there are some existing Python libraries like PyAp, but it does not provide support for all countries. In real life situations, physical addresses may be written in different formats, thus making writing an exhaustive set of rules difficult. We will use 6 hours to investigate and integrate an address parsing strategy using an existing address parsing library with limited country support. If the team has more time, we could invest 10 hours to visualize the returned addresses on the frontend. We plan to use MapBox for this and the hour includes setting up the MapBox account, integrating the library into the web app, and testing the visualization results. If time is available, the team can use 20+ hours to try to figure out how to use deep learning to extract the address intelligently. The work hours of the tasks listed are assumed to be divided equally among each team member.