**Installation/Usage Instructions**

**Overview:** Our application aims to generate smart course recommendations for students. As input, it takes the job role the student wants to aim for, and any skills the student wants to gain. The application will perform a semantic search through available course descriptions that we have scraped, and courses that match are analysed based on the metrics found on Smart Evals. A score is generated from both the analyses and used to calculate the final ranking of each course. Top 10 courses are displayed to the user. Additionally, the courses that are found to be relevant are used to fetch the top job roles that they are known to lead to, thus helping the student explore alternative career paths. A PDF available on Heinz Career Resources page on Handshake is used for this lookup. The job roles found are also returned to the user.

1. The following files must be available in the current working directory:
   1. coursicle.py
   2. employment\_stats.py
   3. etl.py
   4. heinz\_courses.py
   5. course\_similarity.py
   6. docsim.py
   7. scourse.py
   8. handshake.pdf (*downloaded directly from Handshake website, Career Services Resources section*)
   9. handshake.csv (*cleaned up file after scraping from PDF*) – DATA SOURCE 1
   10. Smartevals-2019-ALL.csv (*downloaded directly from SmartEvals website, export to csv option*) – DATA SOURCE 2

The below files are generated by the python code on the fly, and should be placed in the project directory if they don’t need to be scraped fresh.

1. coursicle\_data\_dump.txt – DATA SOURCE 3
2. heinz\_courses.txt – DATA SOURCE 4
3. There are two dependencies outside of Anaconda that need to be installed. These are:
   1. Tabula (used for scraping PDFs)

Run the following command on the terminal: **pip install tabula**

* 1. Gensim (used for semantic search)

Run the following command on the terminal: **pip install gensim**

1. From the terminal, the application can be launched in two ways. One where it scrapes the needed web data afresh, and the other where it uses older scraped data (described in Step 1 and 2).

For running the script, use:

**python scourse.py**

1. The user will be prompted if they want to perform web scraping, or use already scraped data. Responding in yes performs scraping and refreshes data files, otherwise already scraped data will be used for running the application.
2. Once the script runs, it will take 4-5 minutes for generating a corpus to be used by the document similarity checking algorithm for semantic search. A progress bar will be shown for some of this period.
3. Once the corpus is generated, a welcome text will be printed on the console. The user can now provide an input of the job roles they want to target. Press the enter key when you are done entering the job role. For example: **Data Scientist**
4. Next, the user can type in a comma separated list of skills the user wants to target. For example: **Python, Statistics, Machine Learning**
5. Once you press enter, the algorithm looks through the data set (course descriptions) and performs a semantic search for the query terms entered. The top results (courses that match your search criteria the most) are returned. Data from SmartEvals is also taken into consideration before recommending a course, and highly rated courses are more likely to show up in the query results.
6. The program also displays a list of historically recorded career outcomes for the recommended courses. These have been scraped from the Handshake Resources section, and will give you an idea of where people who have taken these courses have ended up.