

Manual Test Plan

Manual test plan for the GoodReads web scraper

OS support

The following OS can be used for testing:

- Windows 10
- Mac OS X
- Linux

Terminal

Terminals are recommended for testing the web scraper

Java version

Java 1.2 and above

Dependencies

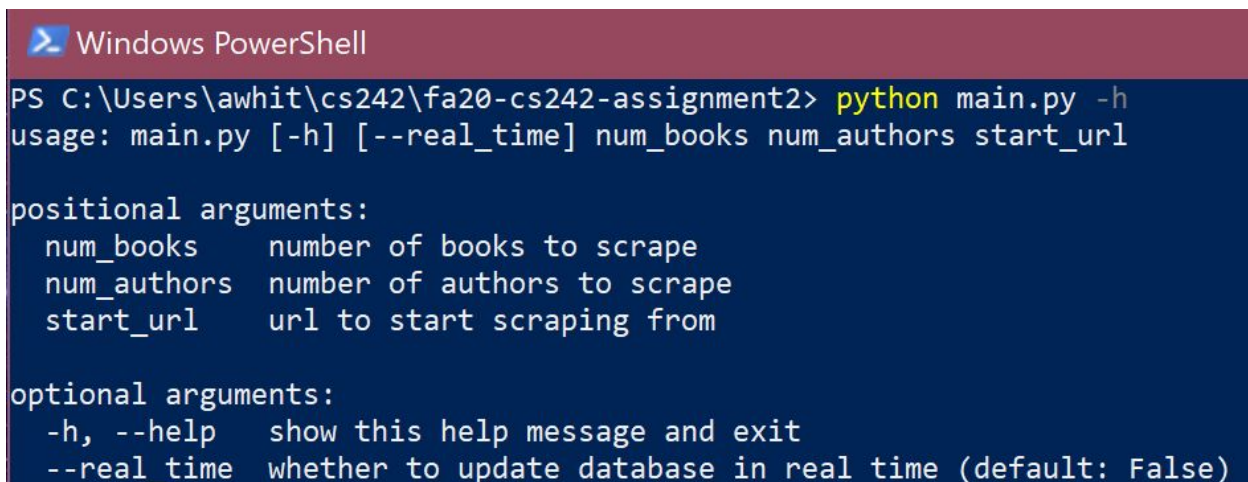
- The program requires that the following libraries are installed:
dotenv
pymongo

Database

- The project data is stored using MongoDB
- The data is stored on a MongoDB Atlas cluster, and can be accessed through the MongoDB Atlas GUI or PyMongo

Run program

- The program can be run through the terminal by executing main.py in the project root directory with arguments as shown in the picture below



```
Windows PowerShell
PS C:\Users\awhit\cs242\fa20-cs242-assignment2> python main.py -h
usage: main.py [-h] [--real_time] num_books num_authors start_url

positional arguments:
  num_books      number of books to scrape
  num_authors    number of authors to scrape
  start_url      url to start scraping from

optional arguments:
  -h, --help      show this help message and exit
  --real_time      whether to update database in real time (default: False)
```

- Note that the program takes four arguments, “num_books”, “num_authors”, “start_url” and “--real_time”. The former three are mandatory; “--real_time” is optional and is set to False by default. “start_url” must be an url to a book page.
- Enter different positive integers for “num_books” and “num_authors” to scrape the information of different number of books and authors
- Enter a different “start_url” to start scraping from a different book

Test data storage

- When “--real_time” is set to True, the data scraped from websites are stored directly into the database; otherwise, data is stored into “books.json” and “authors.json” and then stored into the database
- Connect to the database through the following ways:
Use the connect_to_db() function in db.py
Use the MongoDB Atlas GUI
*Note that the client string is stored in the .env file for security purposes.
The .env file should always be loaded with load_dotenv() before every access to the database
- The database seen from MongoDB Atlas is shown below:

The screenshot displays the MongoDB Atlas web interface. On the left sidebar, the 'library' namespace is selected, showing sub-namespaces 'authors' and 'books'. The main content area is titled 'library.authors' and shows collection statistics: COLLECTION SIZE: 221.06KB, TOTAL DOCUMENTS: 821, INDEXES TOTAL SIZE: 4KB. Below this, there are tabs for 'Find', 'Indexes', 'Schema Anti-Patterns', 'Aggregation', and 'Search Indexes'. A filter bar is present with a placeholder text 'FILTER { "filter": "example" }'. The query results section shows 'QUERY RESULTS 1-20 OF MANY' and displays two document entries. Each entry is a JSON object with the following fields: _id (ObjectId), name, author_url, author_id, rating, review_count, image_url, related_authors (Array), and author_books (Array). The first document is for 'J.K. Rowling' and the second is for 'John Tiffany'. A green chat icon is visible in the bottom right corner of the interface.

- The above is the result of scraping 300 books and 70 authors. Note that the actual number of documents in both collections is much larger, this is because of the incomplete book and author entries added to the lists of books and authors while scraping similar books and similar authors. There should be approximately 300 complete entries for books and 70 complete entries for authors

Test error handling

1. Bad url or unreachable tags
 - In the case when a bad url is encountered, or the program is unable to find the tag containing the information it's looking for, a customized error message will be logged into logs/scrape_books_log.log or logs/scrape_authors_log.log, depending on whether it's triggered during scraping books or scraping authors
2. Bad connection to the database
 - This technically should never happen since the client string is stored within .env
 - In the case when a "connection refused" error is encountered, make sure load_dotenv() is called before connecting to the database