Documentation for CS410 Final Project

Haoyue Bai, Yuqun Wu

1. Overview

We implemented a book crawler that can scrape the books given a starting book. The crawler would scrape books based on the similar books and books under the same author. The scraped data would be stored in the local Mongo DB database. We also implement some small back-end functions, and a front-end server to let user interreact with the database, like searching a book, showing the top 5 rated books in the databased etc.

2. Implementation Details

Basically, we have a back-end system, and a front-end server. The back-end's main function is the crawler, which is implemented using the python package scrapy. It provides a good structure of all functions, and provide a good api for the local mongo db, so we just need to specify the xpath that we want to scrape, and they would be stored into the database. To help the interaction, we also have some function like searching item, deleting item, and showing the top reviewed books, and connect them to the front-server via flask.

In the front-end, we use react. We design the server in a pretty easy way, and jut connect every reaction of website to the specific back-end function, and run it in the back-end. We explicitly show the GUI in the video.

3. Usage of the code

It would be easier to watch the video for the usage!

1). Install the required packages and software
You would like to install the Mangodb database on your own. You might also need to install packages related to Flask, React, etc. You would also need to install the required package using the npm in the front-end directory.

2). Start the MongoDB database
You would need to connect the local Mongodb database in the first place, so that you can store data there. We use the default root: mongodb://localhost:27017

3). Run the back-end server
You would like to go to the dir inside the goodreads/utils, and run 'python api.py' to start the flask back-end server.

4). Start the front-end server

You would like to go to the front-end directory, and run npm start to start the front-end website. You can now enter the front-end web to see the GUI.

5). Play with the website

You can now begin to scrape any book in goodreads by giving the start url, and maximum book number. The crawler would begin to scrape books, and store them. You can then see what's the top rated books, and search them by names to see the details information about them. This would be helpful if you want to find some similar high-rated books given a start book, and begin to read it.

4. Contribution

Our team includes two person: Haoyue Bai and Yuqun Wu. Haoyuq Bai is the team leader. We discuss the idea, implement the functions, and finish all steps together.

5. Limitation

In the first place, we want to add more functionality for the project, like recommending some good books based on quries. For example, after I scrape 1000 books, I want to find a top-rated book about the earth, then I can just search earth, and it would recommend a list of books for me.

However, there's a problem about this. The abstract of the books are not uniform, so we are not able to scrape them, which means that we need to rank the book based on the title. We tried to find whether there's some existing pipelines that can find all synonyms of a word, so that we can fetch it into our system, and use the provided synonyms to search books that might be related, but we failed. We believe this could be done with some more effort, and it would also be better to add it. Hopefully somedays we could collaborate together to finish it. Thanks for reading!