Design Document

11.23.2022

Scott Gebert
Logan Thimer

Short Overview

This is a simple python application that connects to a mongodb database and allows the user to import a json file to initialize the database.

User Guide

Building Document Store

To begin the program that initializes the database with a json file, run the command “python3 load-json.py <filename> <port>”. This will import the file to the database using the given port that the mongodb server is running on.

Operating on the Document Store

To run the program that operates on the document store, run the command “python3 main.py <port>” where port is the port number that the mongodb server is running on.

Functions

Once logged in you will have options to

1. Search for Articles
    a. Provide a list of keywords to search for
        i. Select a search result to see more information about the article. Information displayed contains article id, title, year, and venue.
        ii. Go back
2. Search for Authors
    a. Provide a keyword for an author to search for
        i. Select a search result from the list to see the title, year, and venue of all articles by that author

3. List of Venues
    a. Enter a number to see a listing of the top venues
        i. View the list of results. Each entry in the results contains the venue, the number of articles in that venue, and the number of articles that reference a paper in that venue.
4. Add an Article
    a. Provide a unique id, title, year, and a list of the authors. Once the fields are filled out, the article will be added unless the id is not unique. If the id is not unique, you will be prompted to enter a new id

5. Exit

Detailed Design

Implemented the functionality using a CLI python application. The architecture of the app has been broken down by file.

load-json.py

This file is used to load in a json file to the database. Once the file is loaded, the text index is created for text searching. This file also converts the year fields to a string.

main.py

This is the main file that is run when the application is called. It is responsible for reading in the command line argument, calling the function to initialize the database and switching between the article search, author search, listing of venues, and adding article functions.

articleSearch.py

This file contains all functions related to the searching of articles. The functions are:

- Search for articles: performs a database search for articles based on multiple keywords that are passed into the function. The search is based on AND semantics of the keywords and searches the title, authors, abstract, venue, and year fields.
- Get articles that reference: this returns all articles that reference a certain id.
- Print article: this prints out all information about the article as well as the articles that reference that article.

authorSearch.py

This file contains all functions related to the searching of authors.

- findAuthors - Executes a query that finds authors matching the keyword - word search not a partial search. Takes in the keyword to search and the collection containing the documents.

venueSearch.py

This file contains all functions related to the listing of venues.

- findVenues - takes in the number of venues, collection, db and client. Executes a query to find the top venues in the db based on the number of papers that reference the venue with the top most cited venues shown first.

addArticle.py

This file handles the functionality of adding a new article to the collection. This file will also perform a check on the unique id that is specified by the user to make sure that the id is actually unique.

Testing Strategy

Our testing strategy was based around correctness of output and error checking/failure testing. We generated test data for each document in the database so that all of the test cases could be verified. For correctness of output, we simply came up with a number of valid actions from the user that tested the querying of different combinations of the test data and we verified that the output was correct. We also did error/failure testing where we would purposely perform an invalid action as the user or enter input that resulted in no results.

Group-Work Breakdown

We used Github for collaborating and version control of our project.

I.Scott Gebert

- Search for Authors
- List the Venues

Time spent: 25

Progress Made: Completed all work assigned

II.Logan Thimer

- Searching for Articles

- Adding Articles
- Loading json file to database & creating a text index

Time spent: 25 hours

Progress made: Completed all work assigned