Grace Zhu (gjzhu)
Brett Merkosky (bamerkos)
Fatih Artar (fartar)

CMPUT 291 Mini Project 2

General Overview

This project is a Python program that interacts with a mongoDB database using the Python module pymongo. This program allows users to access a movie database to both search for documents and add new documents.

User Guide and Design

Before running `project2.py`, users should first run `tsv-2-json.py` to create .json files in place of the .tsv files, and then `load-json.py` to populate the database `291db` with these files. When `load-json.py` is first run, the user is asked which port they would like to connect to for mongodb. This should be the port on which mongo is already running. After successful connection, the program will run and populate `291db` with all of the necessary information and create any necessary indices. After running `load-json.py` the user can then proceed to run `project2.py`.
When `project2.py` is first run, the user is once again asked for the port to use. After successful connection, they are brought to the main menu, where they have several options to choose from:

```
Which port would you like to use (empty response is the default port: 27017)? 27017
Connecting...
Connection successful
=========================
CMPUT 291 - Mini Project 2
=========================
1 - Search for titles
2 - Search for genres
3 - Search for cast/crew members
4 - Add a movie
5 - Add a cast/crew member
6 - Exit
=========================
Choose an option:
```

Primary Functions

The program is contained in three files. The first file, `tsv-2-json.py` contains the single function:

- **convertJson:** Used to convert a .tsv file to a .json file, with the proper variable types

The second file, `load-json.py` does not contain any functions other than the main function.

The third file, `project2.py` contains the following functions:

- **connect:** Used to connect to a database. Prompts the user for the port number, attempts to check availability of server, and returns the pymongo.db object of the server connection
- **start:** Prints the menu of the program. Users are also directed back to this page after each action.
- **titleSearch:** Prompts the user for one or more keywords, and retrieves all titles - with complete information - matching every keyword. A keyword matches against both titles and release years. From there, users should be able to select a title to see the rating, number of votes, and names of cast/crew members and their characters.
- **searchGenres:** Prompts the user for a genre and minimum vote count, and retrieves all titles under the provided genre that have the given number of votes or more. Results are sorted based on average rating.
- **searchPeople:** Prompts the user for a cast/crew member names, and retrieves the professions, and the jobs and characters (if applicable) for each title they have been in.
- **addMovie:** Prompts the user for a unique movie id, title, start year, running time, and list of genres, then adds the movie to the database (in title_basics).
- **addMoviePeople:** Prompts the user for a cast/crew member id, a title id and a category (role of the movie person) and adds the role to the database (in title_principals).
- **exitProgram:** Closes the program.

Testing strategy

Testing was done by testing individual queries on the dataset provided, with some entries added to test edge cases. The output of the queries were compared with the expected output to ensure accuracy.

Each of the major functions was tested in as many cases as we could think of, both when writing them and after they had been completed. This included (but is not limited to) cases in which the user provides invalid input, cases when the queries return empty, and cases where the user might not have any options to pick from.

Notable bugs found (and fixed) include:

- Slow-running queries (fixed with proper indices)
- Improper storage of null values and other data types
- Searching for genres and people would return incorrect names (fixed with proper regex)

Group Work Strategy

Communication was done primarily online through Discord and a private Github repository was created to collaborate on the code itself. As well, several in-person meetings were scheduled to work on the project together.

Below is a break-down of the assigned work between the three members. All three members ended up making full progress on their parts.

Grace Zhu *approx. 13 hours total*

- searchPeople function - Completed (5 hours)
- searchGenres function - Completed (5 hours)
- Assorted testing and bug fixes - Assisted (3 hours)

Brett Merkosky *approx. 18 hours total*

- Phase 1 - Completed (6 hours)
- Adding cast/crew members - Completed (3 hours)
- Python framework and structure - Completed (1 hour)
- Project report and README file - Completed (1 hour)
- Optimization of searchGenres - Assisted (3 hours)
- Searching for titles - Assisted (2 hours)
- Assorted testing and bug fixes - Assisted (2 hours)

Fatih Artar *approx. 15 hours total*

- titleSearch function - Completed (7 hours)
- addMovie function - Completed (4 hours)
- Assorted testing and bug fixes - Assisted (4 hours)