291 MINI PROJECT 2

Joshua Ji (jnji)
Kailash Seshadri (kailash)
Vedant Vyas (vvyas1)

SYSTEM OVERVIEW/USER GUIDE:

User Guide:

This is a movie and cast member data processing app that allows users to manage a database of movies and cast members that acted in the movies. When using the app, the user is placed into the main menu, where they can choose between 6 options to navigate the program. The options are navigated using the arrow keys, then selected by pressing the Enter key.

THE AVAILABLE OPTIONS INCLUDE:

- **Search for a title**
  Prints the titles that match a search input and allows the user to view more information about the movie.
  Returns the first 50 results on keyword input. The user can choose between the displayed results, or choose to view more results (This prints 50 results at a time). They can also choose to start another search.
  Upon selecting a movie, the program displays the average rating, the number of votes, and all the cast/ crew that worked on the movie. The user can then choose to search again or return to the main menu.

- **Search for a genre**
  Prints a list of movies of a specific genre, with more than an input number of votes.
  The user enters a genre, and a minimum number of votes (suppose X). The program then returns the first 100 movies that match the genre while having over X votes (sorted by average rating desc). The user can choose to view more results (100 at a time) or return to the main menu.

- **Search for a cast/crew member**
  Prints a list of cast/crew members whose name matches the input, and all the movies they worked on (including their role and their job).
  The user enters a member name and the program prints one cast member from the results and all the movies they have played a part in (including the character they played and their job). After each member, the user can either choose to view the next cast/crew member (until all the cast members have been displayed), choose to start another search or choose to return to the main menu.

  Understanding results:
  The program prints all the information received from the database. [NULL] values in the database are ignored. For example:
  > Played <char> (<job>) on <movie title> (<title ID>)
  Character, job and movie title are not [NULL]
  > Worked as <job> on <movie title> (<title ID>)
  Character is [NULL] but, job and movie title are not
  > Played <char> (<job>) on the movie with ID {titleID} (Title unknown)
  Character and job are not [NULL] but movie title is

> Worked on the movie with ID {titleID} (Title unknown)
Character, job and movie title are all [NULL]

- **Add a movie**
  The user can add a movie to the database by inputting the following:
  unique id, a title, a start year, a running time, list of genres
  Both the primary title and the original title will be set to the provided title, the title type is set to movie and isAdult and endYear are set to Null (denoted as \N).

- **Add a cast/crew member**
  Add a cast/crew member role to an existing movie by inputting the following:
  cast/crew member id, a title id, and a category
  The ordering will be set to the highest ordering plus 1 (or 1 by default), and job and characters will be set to Null.

- **Exit**
  Quits the program.

Note: While inside any option, the user can choose to input 'exit' at any point to return back to the main menu.

SOFTWARE DESCRIPTION/SPECIFICATION:

Main program flow is noted in the flowchart below. Please access the flowchart using this link (ualberta email):

https://drive.google.com/file/d/1FTL2vjjFT-1pma_R-PX6OWz-7Spq3noz/view?usp=sharing

To run the file ensure the following:
- You have the dist/ folder in your current directory. This folder has all the distributed files (load-json, tsv-to-json, main)
- You also have the .tsv files in your current directory.

To start, run the commands below:
- ./dist/tsv-to-json
- ./dist/load-json
- ./dist/main

If the executable does not run successfully, ensure you have the dependencies in `requirements.txt` in `src` installed and are working.
The user can then run `main.py` in `src` by calling `python3 ./src/main.py`

TESTING:

1. Manual Testing (with other groups)
   We manually tested our search movie and update_recommendations outputs with other students in the class. Using this, we can test our code in different (usually larger) databases.
2. Manual Testing (with our own dbs)
   We have our own dbs we created which we use in our group, where we know exactly what the outputs of search movie and update_recommendations are. Using this, we can quickly test our code with a small sample size.

GROUP WORK:

We devised the program flow on Monday (14/3/2022). As our group members had staggered midterms, Vedant and Josh started first. After that, we divided up the remaining functions.

UI styling was split amongst us (with each member tasked with styling the function they work on), since we found a couple of libraries to use for nice terminal interfaces.

tsv-2-json.py	Vedant Vyas
load-json.py	Vedant Vyas
utils.py	Joshua Ji
main.py	Kailash Seshadri
search_title.py	Joshua Ji
search_genre.py	Vedant Vyas
search_cast_crew.py	Kailash Seshadri
add_movie.py	Vedant Vyas
add_cast_crew.py	Joshua Ji
Colours and Styling	Joshua Ji
Group Work Report	Kailash Seshadri
Program Flowchart	Kailash Seshadri
Bug Fixes	Group
Testing	Group

We used VSCode live share to collaborate on the initial design and complete the first few functions. Once this basic structure was complete (up to a scaffolded system() function that had the logic of the main system page), we were able to upload the files needed to GitHub, and work on our own functions on separate personal branches.

Each member was expected to complete their functions by Tuesday (22/03/2022). During this time we would check up on each others' progress every other day via Discord. This allowed us to ensure progress was being made and we could help out anyone that was stuck. We then met on Monday (28/03/2022) to merge them to the main branch. Merge commits happened and they were quite annoying, but we were able to fix them together over the meeting.