Canvas Report for Database Project Part IV

My project is akin to that of MyAnimeList and IMDb where users have access to a personal “watch list”. The user can create an account or log into an existing account. When creating an account, there are multiple fields to be inputted and the front-end side of the program ensures that after registration, the only information being queried are that in which belongs to the user who has logged in. For example, if I wanted to look at my watch list then I should not be able to see my friend’s watch list. When logging into an existing account, the same logic applies. After creation of an account or log in, the user can then create or edit their watch list. No matter which they choose, they will then be prompted with three options in which they can edit their plan to watch list, currently watching list, or completed watch list. If they originally chose to create/edit their watchlist then they will be prompted to enter in the movies they want to add to their watch list in which the front-end application uses SQL to update the database, making sure that it matches with their account. If they chose to view their watch list then the front-end program ensures that the sql code being used only pulls up their information that they have entered.

A screenshot of a computer program

Description automatically generated A screen shot of a computer program

Description automatically generated

The first image shows a hierarchy of the files that exist in my project. Everything that are in the frontend folders are GUIs created for the application while the backend is non-gui. I actually did not end up using DBconnection in the final project but it was very useful when I was debugging code and using as a template for connecting my program to the database. **The Main.java** located in src\moua\_justin\_app.backend NEEDS to be the one that is ran first. It will instantiate a variable apart of a class I created located in moua\_justin\_app.frontend\_unauthenticated called MainpageJF.java to bring up a GUI.

A screen shot of a computer program

Description automatically generated

In the README section, I noted that changing the url, user, and password needs to be done to match the database that the grader (you/the person reading this) is using. This screenshot shows specifically what I am talking about and it exists in multiple files. You can find what they are again by scrolling up and reading the IMPORTANT NOTE in the README section under “how to run program”. This image shows what it will look like.

A screen shot of a computer program

Description automatically generated

This screenshot shows how RegisterJF.java located in moua\_justin\_app.frontend.unauthenticated register’s the user into the database, then grabs their ID to be passed onto the next pages to ensure that we only grab information that belongs to the user. You see there are also print statements on lines 214-218 which are very helpful for debugging when viewing the console.

A screenshot of a computer screen

Description automatically generated

This screenshot showcases how LoggedInJF.java on the left works with SelectWatchListJF on the right. You can see that I highlighted specific keywords. On the left hand shows on line 55 and 68 that the string “edit” and “view” are being passed into the instantiation of the SelectWatchListJF class to open up that GUI so the user can select whether they want to edit or view their plan to watch, currently watching, or completed watchlist. Once it is passed, you can see that on the right side on lines 92, 98, 117 and 123, show how that passed variable is being used so the code understands whether the user wants to view or edit their watchlist(s). (Note: screenshot doesn’t show the code commented but I’ve gone ahead and commented as I realize it may be more useful for grading purposes).

A screenshot of a computer code

Description automatically generated A screenshot of a computer

Description automatically generated

Here we can see SQL code that ensures a table and movie have their own specific IDs and whatnot. The front-end application works with SQL to update and grab information from tblUser and grab movies or tv shows from tblMovie\_or\_TVShow.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer screen

Description automatically generated

These show the insert into statements in SQL relates two tables to one another. The “tblPlanToWatch” for example shows which user is tied to which movie. The front-end code ensures that the user can not add a duplicate if it already exists in java!