<u>(a) a general overview of your system with a small user guide</u>

The general overview of the system is as follows:
We have split up our program into several smaller files, each containing their own classes with their own functions. By doing this, we are able to more easily bug fix as well as do testing. These files are as follows: customer, editor, main, movie, session, shell and utils.

User guide:
After starting the program, you will be prompted to either log in using an existing ID and password or to create a new account, if you are unsure about what to do simply type in “help” in the command prompt to see your options. After logging in, you’ll have many options to choose from depending on if you are logging in as a customer or an editor. When logged in as an editor, the command line expects you to put in an action but if you are unsure you can type in “help” in the command prompt to see your options, assuming that you have prior experience. When logged in as a customer, all your options will be displayed explicitly and you can type in the actions that you need.

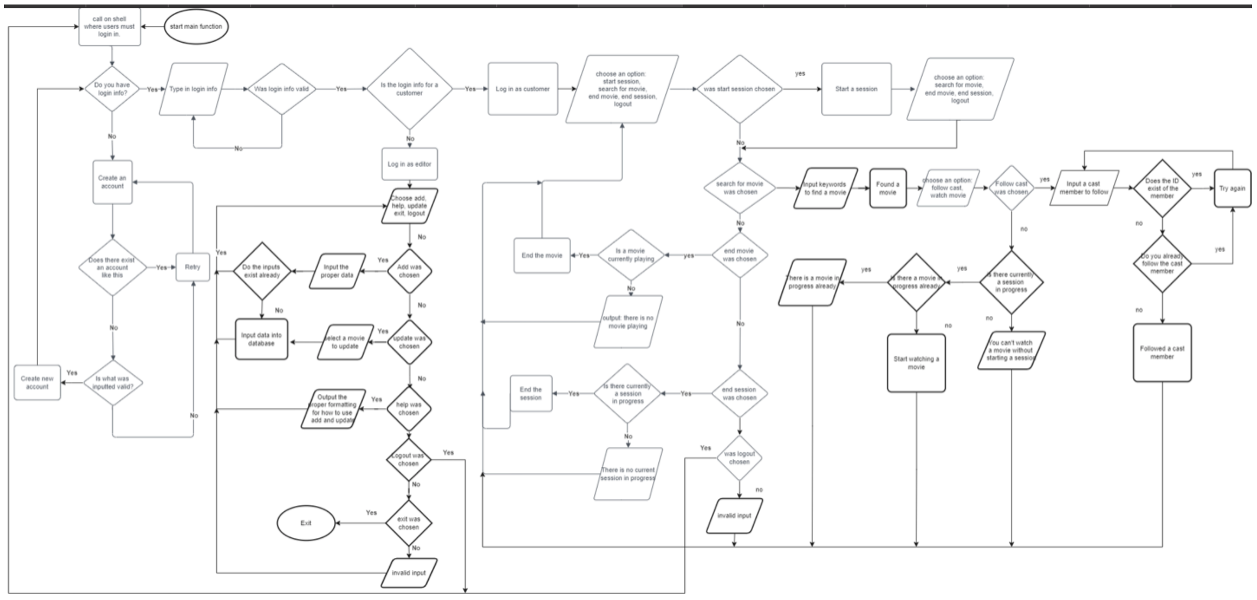
As a customer, you will be given 5 options: Starting a session, searching for a movie, ending a movie, ending a session or logging out.

1. When starting a session, you will be able to search for a movie, end a movie or end the session. Searching for a movie is identical to the other search option with the only difference being that after you search for a movie, you are able to select it and watch it. A movie has to be playing to end it and a session must be started to watch a movie.
2. When searching for a movie, you will be able to search through the database for a movie and look at information about the movie. There is also an option to follow cast members.
3. End the current movie that is playing. There must be a movie playing to end it.
4. End the current session. Must be in a session to end it.
5. Log out: Return the user to the login screen.

As an editor, you are given 5 options: add movies/cast members, help, update, quit/exit, and logout.

1. Adding a movie and cast members is done through this option, where users will input data needed and the data will then be saved into the database.
2. Help shows the proper formatting of how to add values or update the database.
3. Update allows the user to change existing data in the database.
4. Quit/exit exits the program.
5. Logout returns the user to the login screen.

Here is a visual representation of the data flow:
https://drive.google.com/file/d/1RvNYxvbs348Dg2Qk8Uub6v49ih85jznq/view?usp=sharing


start main function
call on shell where users must login in.
Do you have login info?
Type in login info
Was login info valid
Is the login info for a customer
Log in as customer
choose an option: start session, search for movie, end movie, end session, logout
was start session chosen
Start a session
choose an option: search for movie, end movie, end session, logout
Create an account
Does there exist an account like this
Retry
Is what was inputted valid?
Create new account
Log in as editor
Choose add, help, update exit, logout
Add was chosen
Input the proper data
Do the inputs exist already
Input data into database
update was chosen
select a movie to update
help was chosen
Output the proper formatting for how to use add and update
Logout was chosen
exit was chosen
Exit
invalid input
search for movie was chosen
Input keywords to find a movie
Found a movie
choose an option: follow cast, watch movie
Follow cast was chosen
Input a cast member to follow
Does the ID exist of the member
Try again
Do you already follow the cast member
Followed a cast member
Is there currently a session in progress
Is there a movie in progress already
There is a movie in progress already
Start watching a movie
You can't watch a movie without starting a session
end movie was chosen
Is a movie currently playing
End the movie
output there is no movie playing
end session was chosen
Is there currently a session in progress
End the session
There is no current session in progress
was logout chosen
invalid input
Yes
No

(b) a detailed design of your software with a focus on the components required to deliver the major functions of your application

1. Customer

This file, as the name suggests, deals with all customer related functions. The customer file is one of the largest files as it houses the majority of the requirements. Within this file, there are many functions: startSession, searchMovie, followCast, to name a few. We have named each function to give some information about the use of the function. The most important function within the customer file is the handle function. Here, all other customer functions are called and inputs/outputs are processed. Within the handle function, the user is first prompted for an input to start a session, to search for a movie, to end a movie, to end a session or to logout. Choosing to end __ will result in the corresponding end__ function to be called and an output message is shown. Choosing to search for a movie will bring up more prompts such as following casts or using keywords for a more specific search. Following casts will call the followCasts function and using keywords will use the searchMovie function. The same goes for choosing to start a session. The same prompts will be given as choosing to search for a movie, except the difference is that users will be able to choose to watch movies, which calls the startWatching function. Any other input besides the five given will output an error and the user must rechoose.

2. Editor

Like the customer file, the editor file is quite large and hosts a multitude of different functions with its editor class. Also like the customer class, the editor class contains a handle function where all other functions are pooled into. First, users will be prompted with 5 options: add, help, update, quit/exit or logout. Add gives you the option to add movies or cast members using the function addMovieAndCasts, help which shows the proper format to use when adding information into the table, update allows the user to update a recommendation (either adding or removing) using the function update, quit/exit allows the user to return to the initial selection menu and logout is used to exit the program.

3. Main

In this function, all other files and their classes/functions are pooled together. This is where the file is run.

4. Movie

The movie file houses the movie class with two functions: startMovie and endMovie. These two functions are called in other files and as their names suggest, are used to start and end movies. To start a movie, a start time is set and a movie is found from prompted prior information obtained from a previous function. If the movie has already been seen, an exception will be raised. When ending the movie, the duration of the movie is saved unless the movie has been finished.

5. Session

In the session file is the session class and its functions. These functions are called elsewhere, mainly in the customer class. These functions are also quite self explanatory by their names: startSession, startSessionMovie, endSessionMovie, endSession, getmovie. Using the startSessionMovie or the endSessionMovie will call the corresponding function in the movie class. When starting a session, the time is saved and recorded. The time is then updated when the session is ended.

6. Shell

Within the shell file, the shell class and its functions are the frontline of the program, being the first thing called when running the program. The handle function is responsible for account management (logging in and signing up). The handle function has 4 options: login, help, signup and quit. When choosing login, the login function is called and prompts the user to input their credentials (id and password). If the credentials don’t match with that in the database, an error is outputted and the user must reenter their information. If the credentials match with those in the database, then a welcome message is shown along with the type of user logged in (customer or editor). The help function shows users the use of each function. If users don’t have an account, they are able to create one using the signup function. Users then are prompted to create an id and a password, where errors are outputted if the inputted id or password are invalid or if the id exists already. If there are no errors, the newly created id and password are added into the database using the addCustomer function.

7. Utils

The utils class handles all exceptions. This includes proper valid integers, proper strings and proper year date.

(c) your testing strategy

When testing the code, we made sure to isolate each piece of code and slowly work through every requirement listed in the assignment. Since everything was in separate classes and functions, testing was made easier. We also not only tested through requirements in the assignment, but also through each function in the code. We also tested edge cases and tried to handle them accordingly. For example, when signing up, if the id that the customer uses is already in use the program would crash but we were able to handle that among other cases. We found a few bugs including inserting duplicate rows into a table as a consequence of the program or problems with inserting a certain format of ids different from what was shown in the data for assignment2 and this was the nature of most of them but we were able to tackle them and make assumptions when needed which are stated in the readme file.

(d) your group work break-down strategy

As said in the general overview, our thought process was to break the program down into smaller manageable pieces. This way, we were able to separate the work load more easily. For ease of reading, Charles will be group member 1 (gm1), Chengxua will be group member 2 (gm2) and Kevin will be group member 3 (gm3). Gm1 spent about 13 hours working on this project. Gm2 spent about 30 hours on the project. Gm3 spent about 10 hours working on the project.

Within the code itself, gm1 and gm2 did the bulk of coding, while gm3 did some functions here and there. Gm1 coded the movie class, part of the customer class, and the session class. Gm2 did the editor, the shell, utils and main. Gm3 did the other part of the customer class as well as wrote up the whole of the report and the flowchart. All three group members took part in testing the program and made sure the program ran properly.

In order to properly coordinate between each member, a group chat was created so we could communicate with each other. This included progress for code, testing progress or other issues that were brought up during development. We also created a github and created separate branches for each person so that each person could code on their own and build things up. Using github also helped to reduce the possible conflicts between different branches and merging.

Some extra things that we implemented that were not required in the assignment were:

We made inputs from the user that would otherwise lead to a crashing of programs more interactive so the users can have a good experience with it.