Team Name: GUO

Members (and uni): Ahmed Alzubairi (afa2135), Michael Stone (ms5512), Mauricio Antonio Quintanilla (maq2119), Christian Kowalczyk (cjk2159)

Date of meeting w/ mentor: 11/2/20

What we talked about:

- The concept of connecting people to watch a movie must be a part of the MVP
- Can't use password based authentication
- A possible follow up after we finish the MVP of our website is to also send a follow up email w/ the zoom link at the exact time of the session the user noted
- Remember that we changed our minds about the search. It is no longer just searching a movie, it is the search movie/friend.

Part 1:

Our project will allow users to create a list of movies that they plan to watch in the future. These movies can be added to the list via search or from a list of upcoming movies. Next to the list, we will show essential information (release date, rating, poster, possibly where it can be watched). There will also be a way to search & add friends, so that you can invite them to watch a movie with you.

We would also like users to be able to add a "watch date" to each movie, showing the day they plan to watch the movie. From there, users would be connected with others to virtually watch the movie through a service like Netflix Party/Zoom. We plan to have a registration for an account with the service along with a sign in and explicit logout.

Feedback: Can no longer use password based authentication. We plan to use Flask for authentication. Being able to connect friends and generate a link is now part of the MVP, the "watch date" feature is for after we complete the MVP.

The users of this application will be people of all ages who enjoy watching movies, and specifically more avid movie watchers who actively plan to watch films and enjoy watching with others.

Demo: In our demo, we plan to show a user being able to create an account and sign in. From there demo the search movie and the search people feature. Then we can showcase the user profile and settings, along with the home page. Finally showcase how to invite a friend to a movie and generate a link.

Feedback: Added inviting friends to watch a movie into the demo, in addition to the concept of searching/adding friends to be able to invite.

We potentially want to demo adding a date next to each movie.

The data that we plan to store would be the list of movies the user wants to watch. This way if the user logs out, we still have access to their list in the database. We will also be storing a users friend list, movie wishlist, and appointment links, and any profile information (email, name, etc...). As for possible API's, we will use the IMdB API to search for movies. After we completed the first stage of our project and have time to work on the next part, we would like to use the Zoom api to schedule meeting rooms so that a group of people are able to watch the movie. Both APIs are public. For login/ logout and authorization, we plan to use heroku.

Part 2:

< label >: As a < type of user >, I want < some goal > so that < some reason >.

Movie:

<Organizing List> As a movie hobbyist, I want a place to store movies on my watch list so that I can have an organized list of my favorite movies. My condition of satisfaction is that I am able to search for any movie and be able to have it into my list.
<Anticipating new movies> As a film critic, I want to know what movies are coming out so that I can figure out what movies I will need to review. My conditions of satisfaction is that if I click some button or some functionality, I should be able to see the upcoming movie or current movies in theaters.
<Movie ratings> As a busy person I want to know if a movie has a good rating and trailer before watching it so that I don't feel like I wasted my time watching a bad movie. My condition of satisfaction is that I can search for a movie on the website and be given some metric to determine its rating.
<Meeting for Movie> As a solo movie watcher, I want to be linked with others who plan to watch a movie, so that I can watch it virtually with them. My condition of satisfaction is that I am able to get a message or some sort of notification to zoom link where I can watch the movie with others.

My conditions of satisfaction are < list of common cases and special cases that must work >.

The type of user (role) does not need to be human. You may optionally include a wishlist of additional user stories to add if time permits. Keep in mind that the type of user, the goal, the reason (if applicable within the system), all the common cases and all the special cases must be testable and demoable.

Part 3:

Acceptance Testing:

1. (Organizing List): user signs up for an account (will need username, email) and they check if they see their correct information they signed up with in their profile (username, email) and their wishlist.
    a. If the user already has an account they should login with no failure and still be able to see their profile information
    b. If a user inputs invalid login or hasn't signed up yet they should be warned (email doesn't exist, invalid password)

2. (Organizing List): Ask logged in user to add a movie to their wish list and allow them to check if it was actually added to their wish list
3. (Movie ratings): user searches up a real movie name and sees if the correct corresponding movie is returned to the user with a name, trailer, poster, rating, description.
    a. If the movie is fake or doesn't exist we should return a message to the user telling them we haven't found it.
4. (Anticipated new movies): user looks up a movie that isn't currently out yet. If movie exists, return the name , poster, description, and trailer (if available)
    a. If the movie doesn't exist then return a message to the user notifying them.
5. (Meeting for Movie): Ask users to invite someone else to watch a movie with them. The second user should receive a zoom link to connect and watch the movie together.

Part 4:

We plan to use Heroku to deploy our site, we will also be using postgres as our backend database as well. We'll use Visual Studio Code as our preferred IDE. For code coverage we will use CodeCov. We will be using PyTest for testing purposes. We plan to use flake8 for checking code style.

We'll use pycheck to statically check for bugs the same way spotBugs would. We are thinking about also using sonarqube, but we haven't confirmed that yet since the previous technologies already do what Sonarqube does.