* 1. **COMP 2406 Final Project Report**
  2. **Movie Database**

Thomas Soubliere 101144900

Wenyu Zhang 100941511

Date: dec. 6th, 2020

Fall, 2020

**1. Running instructions:**

a. The openStack address and password are as below:

IP: 134.117.134.114

VM\_USERNAME: student

VM\_PASSWORD: student

* 1. b. To start the database.

node database-generator

* 1. c. No extra initialization required
  2. d. To run the server.
     + 1. node server
  3. **2. Provide a summary of what functionality you have implemented successfully and what functionality you have not implemented.**

User Accounts

* Change between a ‘regular’ user account and a ‘contributing’ user account.
* View and manage the people they follow.
* View and manage the other users they follow.
* View recommended movies.
* Search for movies by title, name, and/or genre keyword, at minimum.

Viewing Movies

* See the basic movie information
* See the genre keywords and allow the user to navigate to search results that contain movies with that genre keyword.
* See the director, writer, and actors the movie has, which should also allow the user to navigate directly to each person’s page.
* See a list of similar movies to this one
* See movie reviews that have been added for the movie.
* Add a review

Viewing People

* See a history of all of this person’s work. Each movie entry should allow the user to navigate to that movie’s page.
* See a list of frequent collaborators of this person. That is, a list of people this person has worked with the most, according to your database information.
* Choose to follow this person.
* If a user follows a person, the user should receive a notification any time a new movie is added to the database that involves this person, or any time this person is added to an existing movie.

Viewing Other Users

* See a list of all of the reviews this user has made and be able to read each full review.
* See a list of all of the people this user has followed and be able to navigate to each person’s page.
* Choose to follow this user. If a user X follows a user Y, user X should receive a notification any time user Y creates a new review.

Contributing Users

* Add a new person to the database by specifying their name.
* Add a new movie by specifying all of the minimum information required by your system, including at least one writer, director, and actor.
* When viewing a movie, be able to edit the movie by adding actors, writers, and/or directors.

REST API

* GET /movies – Allows searching for movies in the database.
* GET /movies/:movie – Allows retrieving information about a specific movie with the unique ID movie, assuming it is a valid ID.
* POST /movies – Allows a new movie to be added into the database. It will accept a JSON representation of a movie and is responsible for checking the data is valid before adding it to the database.
* GET /people – Allows searching for people within the movie database.
* GET /people/:person – Retrieves the person with the given unique ID, if they exist.
* GET /users – Allows searching the users of the application.
* GET /users/:user – Get information about the user with the given unique ID, if they exist.
* GET /reviews/:review – Get information about the review with the given unique ID, if they exist.
  1. **3. Describe any extensions you included beyond the required specification.**

1. A mongoose database is included in the project to store movies, people, reviews, and user’s data.
2. A webscraper is used to add movies to the database using imdb urls

Most movies on IMDB are already in the database. Here are some new release urls to test the web scraper:

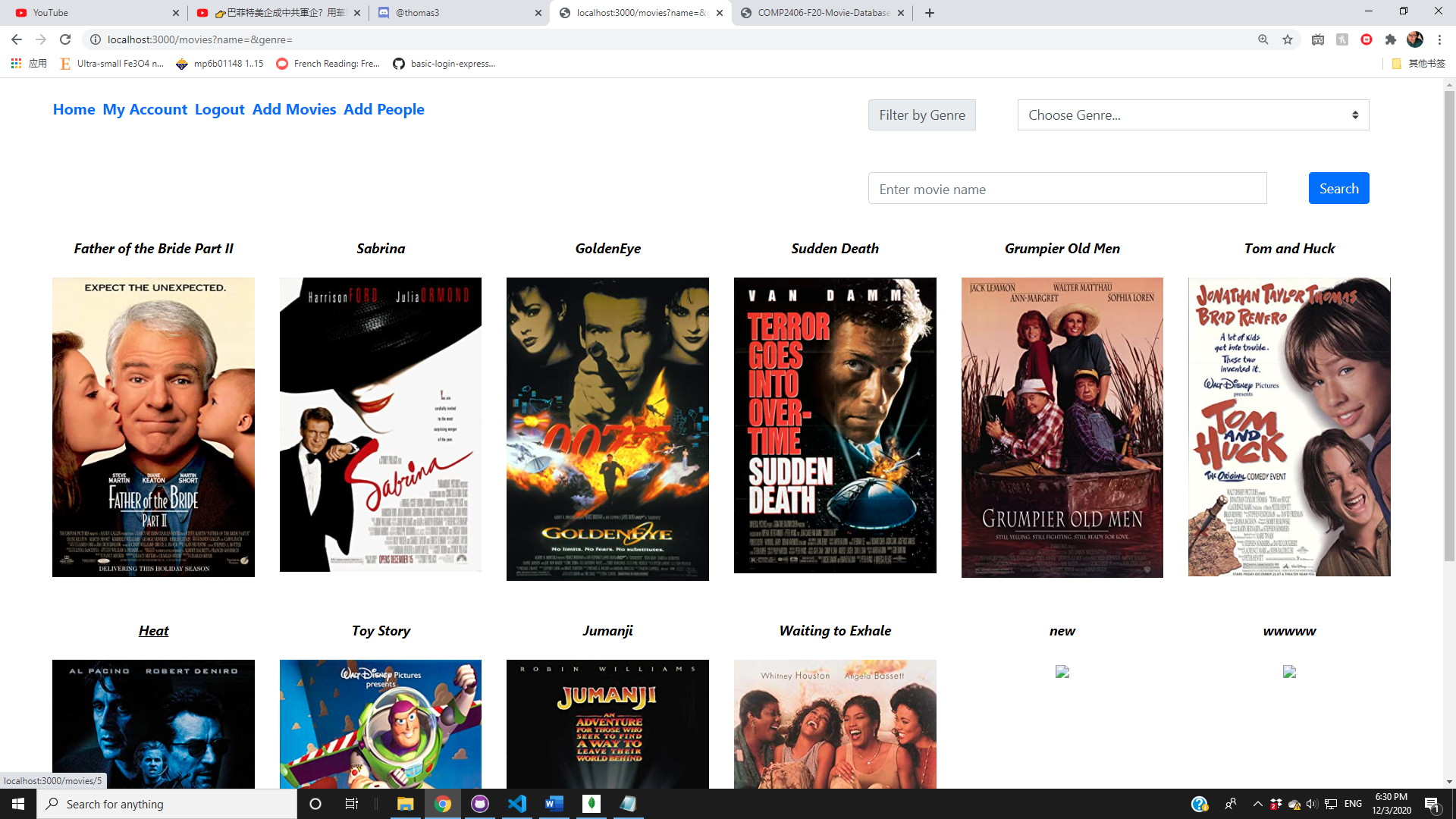
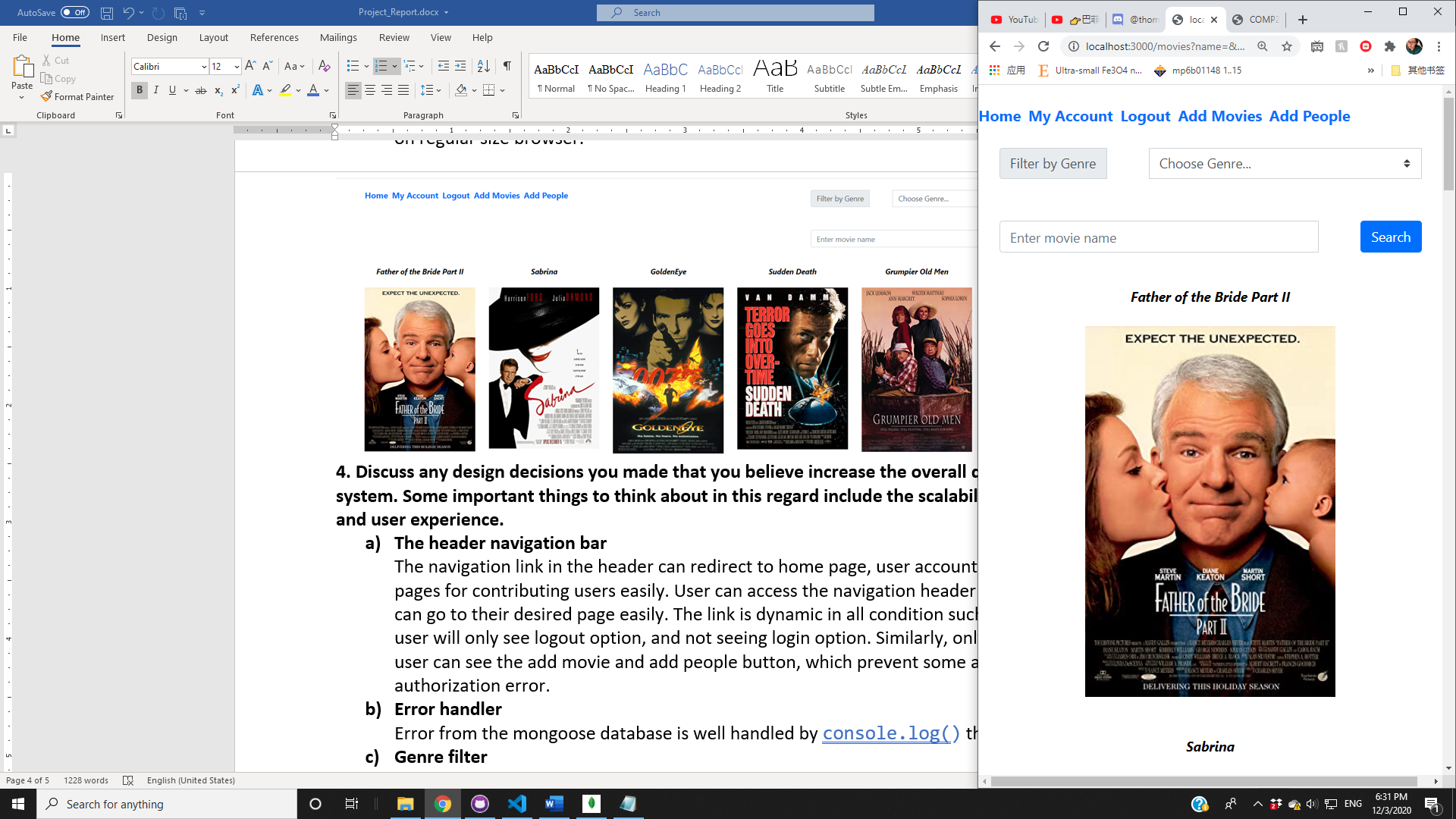
<https://www.imdb.com/title/tt10833270/?pf_rd_m=A2FGELUUNOQJNL&pf_rd_p=&pf_rd_r=15FRR1SVS705V6XMRVPV&pf_rd_s=center-3&pf_rd_t=60601&pf_rd_i=&ref_=il_tl_li_tt>

<https://www.imdb.com/title/tt10054316/?pf_rd_m=A2FGELUUNOQJNL&pf_rd_p=&pf_rd_r=15FRR1SVS705V6XMRVPV&pf_rd_s=center-3&pf_rd_t=60601&pf_rd_i=&ref_=il_tl_li_tt>

<https://www.imdb.com/title/tt4353270/?pf_rd_m=A2FGELUUNOQJNL&pf_rd_p=&pf_rd_r=15FRR1SVS705V6XMRVPV&pf_rd_s=center-3&pf_rd_t=60601&pf_rd_i=&ref_=il_tl_li_tt>

1. The web pages are compatible to all browser size and in all devices.

The profile image of each movie is scalable with the browser’s size. When the width of the browser is too small, the movie list page will show one movie poster a line (Figure 1b), instead of 6 posters on regular size browser (Figure 1a). Meanwhile, the filter and search bar will be under the navigation bar when there is not enough space.

* 1. (a) (b)
  2. **Figure 1**. Displaying the movie list in a regular size browser (a), and a narrower browser size (b).
  3. **4. Discuss any design decisions you made that you believe increase the overall quality of your system. Some important things to think about in this regard include the scalability, robustness, and user experience.**

1. **The header navigation bar**

The navigation link in the header can redirect to home page, user account page and editing pages for contributing users easily. User can access the navigation header in all pages and can go to their desired page easily. The link is dynamic in all condition such that a logged in user will only see logout option, and not seeing login option. Similarly, only a contributing user can see the add movie and add people button, which prevent some access and authorization error.

1. **Error handler**

Error from the mongoose database is well handled by console.log() the error.

1. **Genre filter**

The genre filter is added next to the search bar to help user select their interest genre in the database. Users can check all genres in the database through the list

* 1. **5. Discuss any improvements to your system that you think could be made to increase its overall quality. This is an opportunity to demonstrate your understanding of course concepts that you feel were not adequately demonstrated in your project implementation.**

1. **Improve the UI design**
   1. Background

A more colorful background may help to improve the webpage and attract users.

* 1. Interactive features

Some interactive animation can be added to buttons to improve the user interface design

* 1. …

1. **Allow users to upload their profile photo**

This will make the webpage more attractive. And users can identify each other more easily.

1. **Send real-time notification**

While notifications are handled through email currently, it would be better to send a real-time in web notification to users when there are any new activities. This way can send the new message to user and notify them directly, without bypassing through email

1. **User chat room**

A user chatroom may also improve the overall webpage quality and provide more activities for users. Users can chat to each other to discuss their favorite movies or people.

1. …
   1. **6. Identify any modules, frameworks, or other tools that you used and justify their use.**
2. ***body-parser***

body-parser are commonly used in every submission form, including register and login page, add and edit movie/people. This tool assist parsing out the user input from the webpage and pass that to each POST function.

1. ***mongoose:***

mongoose database is used to store movie, people, users, and reviews in the database. Whenever there is a change required, the specific object will be found through matching their ID or name (findOne() or findByID()), then updated back to the database (updateOne()). Each object will be assigned with a unique object ID, which makes the rendering in REST API a lot easier.

1. ***passport:***

passport is used for user login. When the username and password are matched, it will render the success login function

1. ***nodemailer:***

nodemailer dealt with the user’s notifications by sending them an email. When the person is added to a new movie or a followed user writes a new review, an email with the update information will be pre-edited and sent to the user’s email address. For now, a temporary outlook email account is created to send out all the notifications.

* 1. **7. What do you like most about your project? What would you say is the best feature(s)?**

Filter by Genre?