COMP 2406B PROJECT REPORT

README Requirements:

PROJECT

- Movie Database Project

GROUP MEMBERS

- Hilaire Djani
- Ebubechukwu Okelekwe

1. OPEN STACK INSTRUCTIONS

- Instance information:
  - Public Key: "134.117.129.59"
  - Password: "password"
  - Username: "student"
- Setup instructions
  - ssh into open stack instance using the command "ssh student@134.117.129.59"
  - Provide the password "password"
  - Once in the instance, change the environment to production by running "export NODE_ENV=production" (This is because we are using a deployed mongodb database for our openstack instance)
  - cd into the project folder "movie_project"
  - Start the server by running the command "npm start"
  - Login with user email: "user1@test.com", password: "password" or register a new user
  - Enjoy!!!

- INSTALLATION AND TEST INSTRUCTIONS

You will need to have mongodb installed locally to run the project from zip

- To install and setup the project:
  - Unzip the project folder
  - Run the command 'npm install' to install node modules
  - Start the mongo daemon with the command "mongod --dbpath=db-mongo
  - Run the command 'npm run seed' to seed test data into the database
  - Run the project with the command 'npm start'

-To test the project:
  - You will need postman to test the project's api routes
  - Enter the route URL for the api route you want to test
  - Enter the corresponding query parameters
  - Make the request on postman and observe the output

2. FUNCTIONALITIES IMPLEMENTED

- User Accounts:

- Users can create new accounts by specifying a unique username and password. Users can log in and out of the system using their username and password. Users once logged in can change between regular and contributor account types, view and manage people they follow, view, and manage other users they follow, view recommended movies, and search for movies using either title/name/genre etc.

- Viewing Movies:

- When viewing movies users can see basic movie information such as title, release year, rating, runtime, plot etc. They can see genre keywords as well as see the directors, writers and actors and navigate their pages directly. Users are also able to see a list of similar movies and see reviews that have been added for a movie. Users can add a full review by specifying a score out of 10, a summary and a full review text.

- Viewing People (directors, writers, actors):

- When viewing people users can see the history of a persons work and a list of their frequent collaborators. They also can follow people. Users receive notifications 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:

- When viewing a page for another user, current users can see a list of all the reviews this user has made and be able to read each full review. They can see a list of all the people this user has followed and be able to navigate to each person's page. Users also receive notifications any time a user they follow posts a review.

- Contributing Users:

- Contributing users can do everything regular users can do as well as add a new person to the database by specifying their name, add a new movie by specifying all the minimum information required and their able to edit movies.

- REST API

- The database provides a public JSON REST API that supports all the listed routes and parameters.

3. - Web Scraper:

- A web scraper was used to retrieve additional movie data from IMDB, and it is then incorporated into our system. The scraper populates a form used to add movies and it also gives user time to double check the information before it is submitted.

- React:

- React was used as a framework in developing the front end of the project

- Redux:

- Redux was used in state managing the front end as it centralizes its state and logic.

- Bootstrap:

- Bootstrap was used to boost its responsiveness across devices.

- Socket.io:

- This was used to handle notifications.

- MongoDB:

- This was used as the database.

4. - The use of bootstrap to boost the systems responsiveness as well as its adaptability across devices.

- The use of a scraper to ease the process of filling out a movie form as data is parsed from the IMDB database.

- The ability for users to easily access and navigate through each part of the system.

- The fact that the system makes use of separation of concern i.e., as opposed to server side rendering the client and back end are separated. This improves efficiency as the backend does not need rendering.

5. - The extensive handling of errors on the front end.

- The algorithm that fetches frequent collaborators is not as efficient as searches are only performed by genre.

6. - React:

- React was used as a framework in developing the front end of the project

- Redux:

- Redux was used in stage-managing the front end as it centralizes its state and logic.

- Bootstrap:

- Bootstrap was used to boost its responsiveness across devices.

- Socket.io:

- This was used to handle notifications.

7. The fact that react was used in the front end. It allows for easy dom manipulation with the use of state