Tu Nguyen

San Yeung

CS-2300

12/11/2022

Project Report

1. **Problem Statement:**

My project idea is a web application for a movie database that allows users easily to look up an Englishmovie or TV show and find out where it is streaming. The web application will be similar to the IMDb,but it will have smaller scale and specific purpose, which is searching English movies based on their services. A Three-Tier Client/Server Architecture for Web Application for storing and searching movies/TV shows and movies/TV shows’ services. The web application is defined by the movie database, so a database is the major component. If the movie database and DBMS have more data and information about movies, the web application is going to become more essential, and users will use it more frequently. For this project, the movie database should have the following information: movies/TV shows’ names as well as actors, directors, writers, genres, lengths of movies, language, country, status, certification, Youtuber trailer videos, posters, release date, and the streaming services. There will be a lot of actors, directors, writers, streaming services, and genes in a movie. While I am self- learning JavaScript and working on this project at the same time, the framework for this web application  
will be either node.js or Reactjs because these frameworks are easy to learn. If node.js is the  
framework, I will need to pick a web server such as Hostinger or GoDaddy. If Reactjs is the framework, I will need the DBMS. It is depending on the phase 2 and phase 3 and other classes if I have more free time to learn node.js. For the DBMS, MySQL will work good for both node.js and Reactjs

1. **Conceptual Database Design:**

Diagram

Description automatically generated

**Every movie/TV show (title) should have Title\_ID, Poster as image data type, Title type (movie or tv/show), Status (finished date or ongoing), Country, Movie\_name, Length\_of\_movie, Release\_date, Certification, Youtube\_trailer\_links, streaming services, Genre, actors, directors, producers, writers, and story line.**

**Movie should have at least 1 ACTOR, DIRECTOR, WRITER, STORYLINE, and PRODUCER, and ACTOR, DIRECTOR, WRITER, and PRODUCER are sub classes of the super class PERSON connecting with overlap constraints. Each ACTOR, DIRECTOR, WRITER, and PRODUCER should be in at least 1 movie/tv show.**

**Movie should have at least 1 genre to multiple genres. A genre in the database does not have to be in 1 movie.**

**Movie can have no streaming service because of movie theater exclusive deals, or a move can have multiple streaming services. A streaming service should have more than 1 movie.**

1. **Functional Requirements:**

This web application will have a search bar for search options by using the TITLE, ACTOR, PRODUCER, DIRECTOR, GENRE, and STREAMING\_SERVICE entities. You can search by a movie name, country, date, certification, language, genre, and streaming service.

It will have a filter option that can optimize the searching experience.

It should have movie information when user clicks on that movie by using the TITLE entity. It needs to provide link of the service if available.

There will be YouTube links for the movie trailers that can actually run the video. There will be streaming service links that lead to the service website or service website that has the movie searched.

It should show new or hot movies in the home page for advertisement.

There should be an admin page that let admins manipulate movies or tv shows, or we can do that on the DBMS.

1. **Logical Database Design**

Diagram, schematic

Description automatically generated

1. **Application Program Design**

Search\_Name(name)

for every Title\_name in Title table

if(Title\_name = name)

add to the result list

return the result list

Search\_Actor(name)

for every A\_name in Actor table

if (A\_name = name)

get Title\_ID and Actor\_ID

add to the result list

return the result list

Search\_Director(name)

for every D\_name in Director table

if (D\_name = name)

get Title\_ID and Director\_ID

add to the result list

return the result list

Search\_Streaming\_Service(service)

for every service\_name in Streaming\_service table

if( service\_name = service)

get service\_ID and Title\_ID

add to the result list

return the result list

Search\_Genre(genre)

for every genre\_name in Genre table

if(genre\_name = genre)

get genre\_name and title\_ID

add to the result list

return the result list

Filter(option)

result list to store the result

if(option = “title name”)

name = get name from search bar

result = Search\_Name(name)

if(option = “title type”)

type = get the type’s option from interface

result = Search\_Type(type)

if(option = “status”)

status = get the status’s option from interface

result = Search\_Status(status)

if(option = “release date”)

date = get date from search bar

result = Search\_release\_date(date)

if(option = “certification”)

certification = get certification from search bar

result = Search\_certification(certification)

if(option = “actor”)

actor = get actor name from search bar

result = Search\_actor(actor)

if(option = “director”)

director = get director name from search bar

result = Search\_Director(director)

if(option = “producer”)

producer = get producer name from search bar

result = Search\_Producer(producer)

if(option = “service”)

service = get service name form search bar

result = Search\_Streaming\_ Service(service)

if(option = “genre”)

genre = get genre name from search bar

result = Search\_Genre(genre)

return result list

1. **Installation** **Instructions**

**1.Download the zip file**

**2.Extract the file**

**3.Install npm and node.js :** [**https://nodejs.org/en/download/**](https://nodejs.org/en/download/)

**4.Create the Project Database:**

1. download and open MySQL
2. import/insert SQL files (cs\_2300\_db.sql and insert.sql) from the SQL files folder(\CS-2300\movie-app\sql files)
3. run cs\_2300\_db.sql first to create a database and then run insert.sql to insert data into database

**5.Connect Database to reactjs Server:**

1. open index.js in server folder (.\CS-2300\movie-app\server\index.js)
2. configure the host, user and password in the function (from line 7 to 12)Text

   Description automatically generated
3. IN TERMINAL, type : npm run devStart
   1. if (**running on port 3001 / Connected to the MySQL Server**) is displayed in the console, you are connected to the MySQL
   2. if it throws error message, go to MySQL and run this query

ALTER USER 'root'@'localhost' IDENTIFIED BY 'MY\_NEW\_PASSWORD';

FLUSH PRIVILEGES;

MY\_NEW\_PASSWORD is your current or new password. This will make react project connect to MySQL, and I don’t know why it works

* 1. fix the password in the index.js, then run it again

**6.Run the Website:**

1. Go to index.js in server folder (.\CS-2300\movie-app\server\index.js) and open terminal at that current server folder and run npm run devStart to start the API. If you already have done it in the previous step, you don’t have to do it again. You can check the API by going to localhost:3001/api/get website to check if the API is working
2. Install react-script in the project folder by going to cs-2300-main\movie-app\client => npm install react-script
3. Go to App.js in client folder (.\CS-2300\movie-app\client\App.js) and open terminal at that current client server folder and run npm start
4. **User Manual**
   1. You can search a title name, actor/director/writer name, genre, streaming service in the search box with the filter option next to search boxGraphical user interface, application

      Description automatically generated
   2. A screenshot of a video game

      Description automatically generated with medium confidenceGraphical user interface, application

      Description automatically generatedIf you hover the mouse the movie picture, you can see the length of the title, and if you click on the picture, you will redirect to the title information page where you can see the title detail info. You can click on the service names above the youtube video to go directly to that service website.
   3. When you are in the title information page, you can click Home on the top to go back to the main page

<https://github.com/Tyan206/cs-2300>

credit:

<https://www.youtube.com/@ChrisBlakely> for showing how to format the website

<https://www.youtube.com/@PedroTechnologies> for showing how to use react