

Final Project Report

MyMovieList

San Jose State University
CS 157A
Dr. Ching-seh Wu

Jiawei Zhang
Udaypal Singh
Adan Hernandez

Project Requirement

Project Description

Our proposed project is MyMovieList, a social media web application for both avid and casual moviegoers!

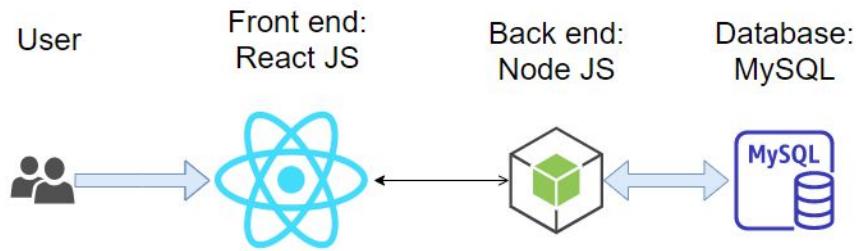
The purpose of MyMovieList is to give movie watchers a platform to show off all the movies and tv shows they have watched, as well as leave reviews for these movies and tv shows. Ultimately we want this application to be one that people will use for their movie inquiries and connecting people with similar interests.

At the moment, there is no technology out there for people to connect with other moviegoers or for people to keep track of what they have watched. Many tend to forget which movies they have watched, so using this web application will allow them to track the list of movies they have watched with the rating given by the user. It is almost positive that there has been a time where someone or yourself is trying to remember the name of a movie that they watched in the past. It's on the tip of their tongue, but they just can't remember. This then leads to an awkward pause or a disconnect between the people in the conversation. Our tool aims to solve this problem and help people connect in a much easier and comfortable way with each other. Often, people have the same interest as their friends, so people can look at their friends' movies list and look at the reliable rating and suggestions made by their friend. This application will be designed specifically for ease of use and appealing to the users.

The reason MyMovieList is the to-go application for your movie needs is because we put our customer first. We want it to be more than just a list of movies. We want it to be fun, interactive, and useful. We want people to connect with others around the world over the shared interests we know they all have.

The stakeholders for this web application will be mostly teens and young adults, along with movie enthusiastic people. Stakeholders can be anyone from a general population to a professional critic. The targeted audience for this web application is passionate movie and film viewers.

System Environment



Hardware and Software Used:

- HTTP Apache Tomcat
- Windows

RDBMS Used:

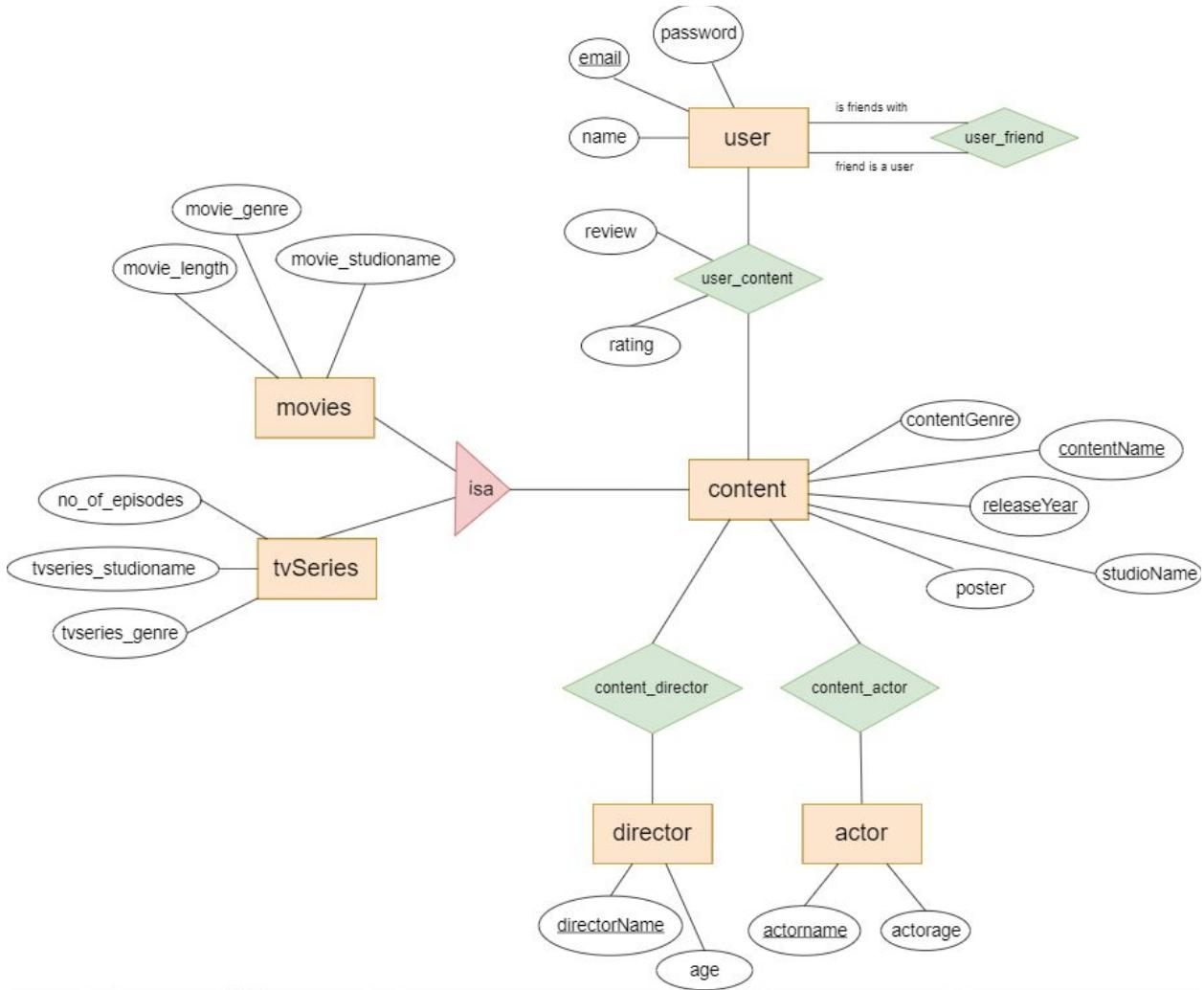
- MySQL

Application Languages Use:

- Javascript
- React JS
- CSS
- HTML
- Node JS

Project Design

MyMovieList ERD



Entity and attributes

- Entity: User
 - User(email, `password`, `name`)
 - Constraint: Email is primary key
 - User has access to website.
- Entity: Content
 - Content(releaseYear, contentName, `contentGenre`, `studioName`, `poster`)
 - Constraint: releaseYear and contentName are primary key.
 - Content has all the movie and TV Series information.
- Entity: Director
 - Director(directorName, `age`)
 - Constraint: directorName is primary key.
 - Director stores the information of directors.
- Entity: Actor
 - Actor(actorName, `actorage`)
 - Constraint: actorName is primary key.
 - Actor stores the information of Actor.
- Entity: Movies

- Movies(movieName, movie_genre, movie_length, movie_studioname)
 - Constraint: movieName is primary key.
 - Movie stores the information of Movies.
- Entity: TvSeries
 - TvSeries(TvName, no_of_episodes, tvseries_studioname, tvseries_genre)
 - Constraint: TvName is primary key.
 - TvSeries store all the information TvSeries.
- Relationship: user_content
 - user_content(review, rating, email, contentName, releaseYear)
- Relationship: content_director
 - content_director(contentName, releaseYear, directorName)
- Relationship: content_actor
 - content_director(contentName, releaseYear, actorName)
- Relationship: user_friend
- Non-Trivial Functional Dependencies
 - For User:
 - Email ----> name, password
 - For Content:
 - contentName, releaseYear ----> contentGenre, poster, studioName
 - For Director
 - directorName ----> age
 - For Actor
 - actorName ----> actorAge
 - For Movies
 - movieName ----> movie_genre, movie_studioname, movie_length
 - For TvSeries
 - tvName ----> no_of_episodes, tvseries_studioname, tvseries_genre

Relations:

- User
 - email → name, password. Since email is a superkey, User is in BCNF.
- Content
 - contentName, releaseYear → contentGenre, poster, studioName.
 - contentName and releaseYear are super key, hence content is in BCNF.
- Director
 - directorName → age
 - DirectorName is key and superkey, Director is in BCNF.
- Actor
 - actorName → actorAge
 - actorName is key and super key, Actor is in BCNF.
- Movies
 - movieName → movie_genre, movie_studioname, movie_length
 - movieName is key and super key. Movie is in BCNF.
- TvSeries
 - tvName → no_of_episodes, tvseries_studioname, tvseries_genre
 - tvName is key and super key. TvSeries is in BCNF.

We can sell all the table has super key at the left hand side and non-prime attribute on right hand side. All functional dependencies fulfill the requirement of BCNF.

Functional Requirement

System Interaction with users

The system provides various functionalities for users. Consumers shall be able to register with users' email and password and use these information to login the system. Consumers shall be able to view different categories of movie list, add the movie to their list and remove a movie from the list. Consumers shall not modify the system movie fix lists, such as the top trending, most popular movie list. The system shall be able to recognize registered or non recognized users. Registered users have the privilege to create their own favorite movie list whereas non registered users shall only be able to view movie list. Consumers can access the system through the web browser on the internet.

Functional Requirement:

Sign Up

- A user shall be able to use their email and password to create a user in our system.
- Once a user enters the information, the system shall be able to create a end-user and enable the user with all the privileges and functions associated with the website.

Login

- A user shall be able to log into the website using correct email and password.
- Once a user logs in, the system shall give access to the user, along with all the privileges and functions associated with the website.

Movies and TV series Listing

- A user shall be able to browse the movies and T.V series from the database provided by the system. The user shall be able to click on the movie in order to enter the subset of the movie.
- The system shall provide a list of movies for user to browse. Each of the movie shall include a brief introduction of the movie.

Display content information

- For a particular content, if the user clicks on the "more info" button, the user shall be able to see a brief description about the content, as well as the actors, directors, genre of content, studio and production company, movie length in minutes (for movies), and number of episodes (for T.V. series).

Rate content

- If the user clicks on the review button for a particular content, the system shall open up a new page, where the system shall provide review writing section and rating dropdown to select a rating (from scale 1 to 10).

Review content

- The user shall be able to write a review comment(400 words max) to a movie.
- When a user clicks on submit, the review and rating should be added into the database.
- The movie, along with the review and the rating, will be stored into user's movie list so that users can have quick access to their movies.

Add a Friend

- The user shall be able to create a connection with other users by entering their email information.
- Once user adds another user, the system shall be able to insert the friend's email information to the user's friend list table.

See Friend's Review List

- A user shall be able to see his or her friend's review list by clicking on the email of a friend from a dropdown menu.

Add to movie list

- A registered user shall be able to add a movie to their movie list by clicking on the submit review button after completing reviewing and rating for a movie.
- The system shall store be able to store a movie to a movie list associated to a user account in order to let the user browse through later.

Delete movie from list

- A user shall be able to delete a desire movie from the movie list. When a user clicks on the delete button the movie is deleted from the database and movie list if automatically reloaded again.

Edit a review and rating

- A user shall be able to edit review and rating
- When a user clicks on edit button, a popup will be shown where the review and the rating will be pre-loaded, and a user shall be able to edit it.
- After clicking on the submit button, the review and rating shall be updated in the database and the page will automatically refreshed to show the updated data.

Non-functional Issues

Graphical User Interface:

The user interface of this application will be user-friendly and easy to use. The search bar for searching movies would be placed on top of the web application for easy accessibility. The features of the application such as movie list would be visible to the user by sorting based on latest, oldest, and rating. There will be dedicated buttons “Invite” which will take the user to his or her friend’s list and allow user to invite other users to watch a movie together. To minimize complexity, the rating will be shown next to the movie title and can also be accessed to edit or change reviews of a movie. The rating will be shown in sets of star shape ranging from 1 to 5. This will increase the accessibility and interaction of user with our web application.

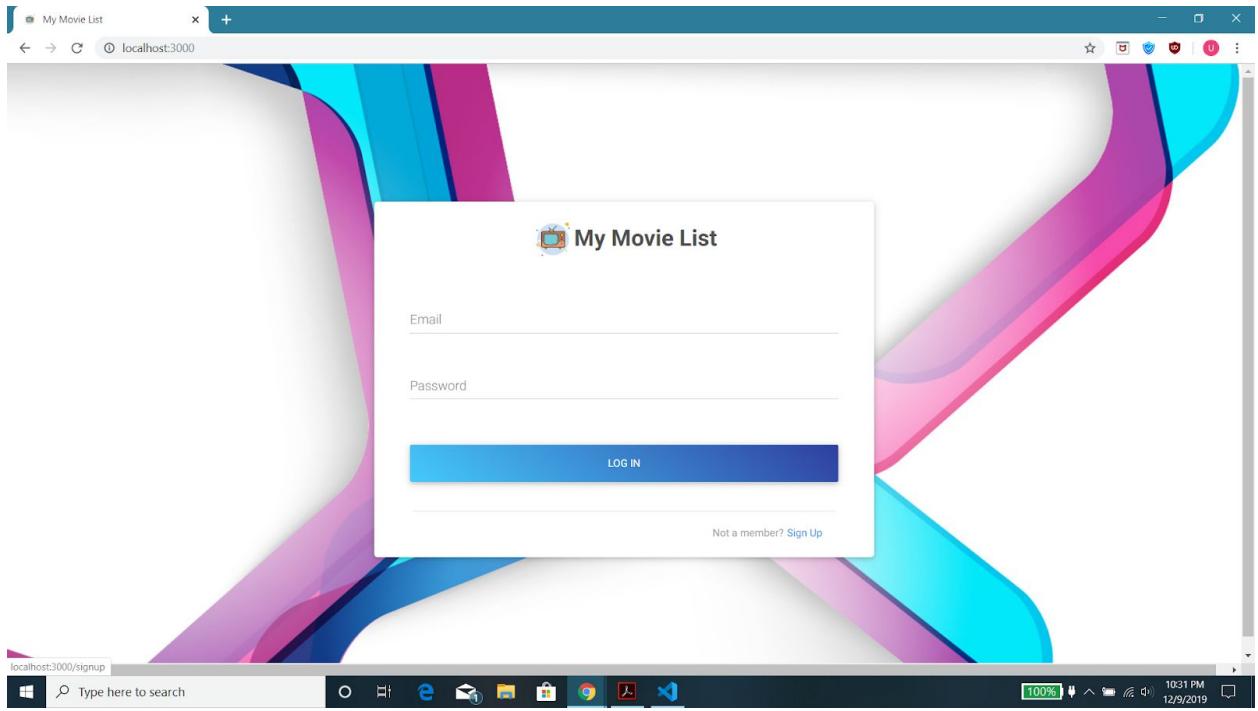
Security:

The user must login with username and password to access his or her movie list. The user must provide a valid email account to create a login credentials for this web application. This prevents others to gather information about the user and maintains confidentiality. The user will have an option to view his or her profile with the other users in the friend’s list. The HTTP server uses TCP/SSL connection which ensures the data will be encrypted, and it will not be easily decrypted by an attacker. TCP also ensures the connection from both ends, server and the client, so the data will be delivered to the client securely without losing the packet throughout the process.

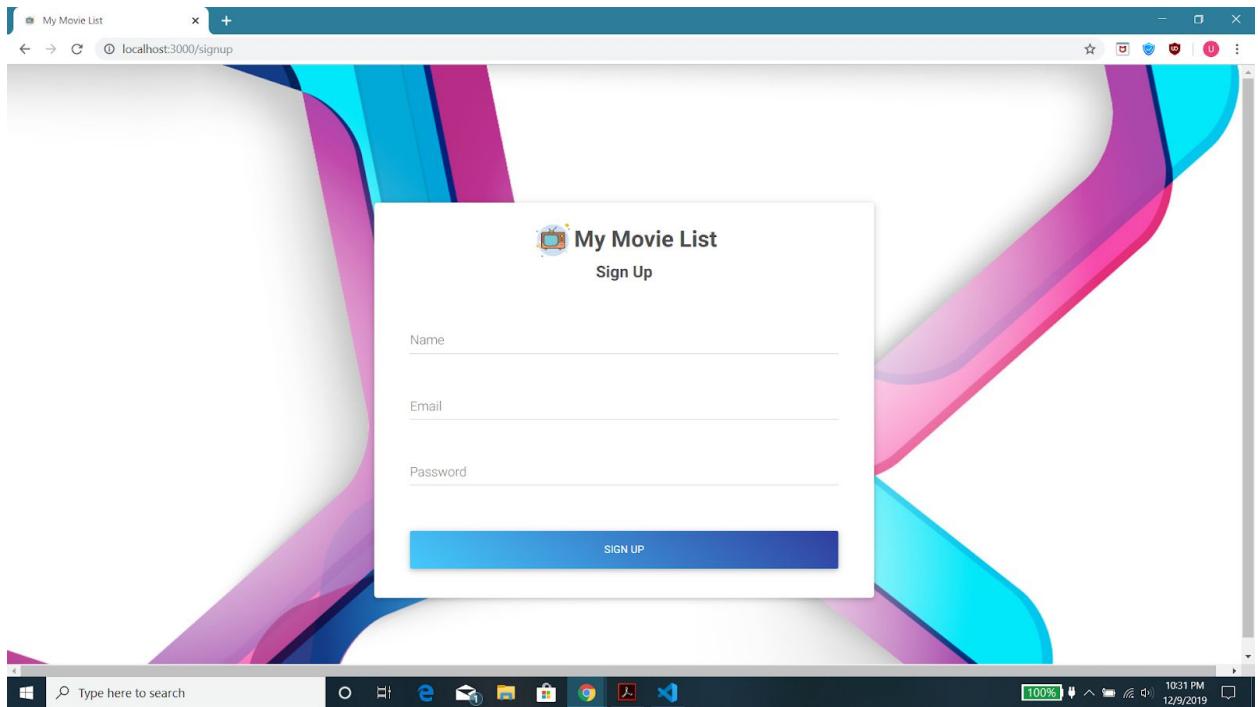
Access Control:

A user will be able to access his or her own movie list, and movies from the database. A user can also share movie list with friends and other users. However, a user can not change or edit other users' movie list. Every user will only have access to their own list, with read and write permission.

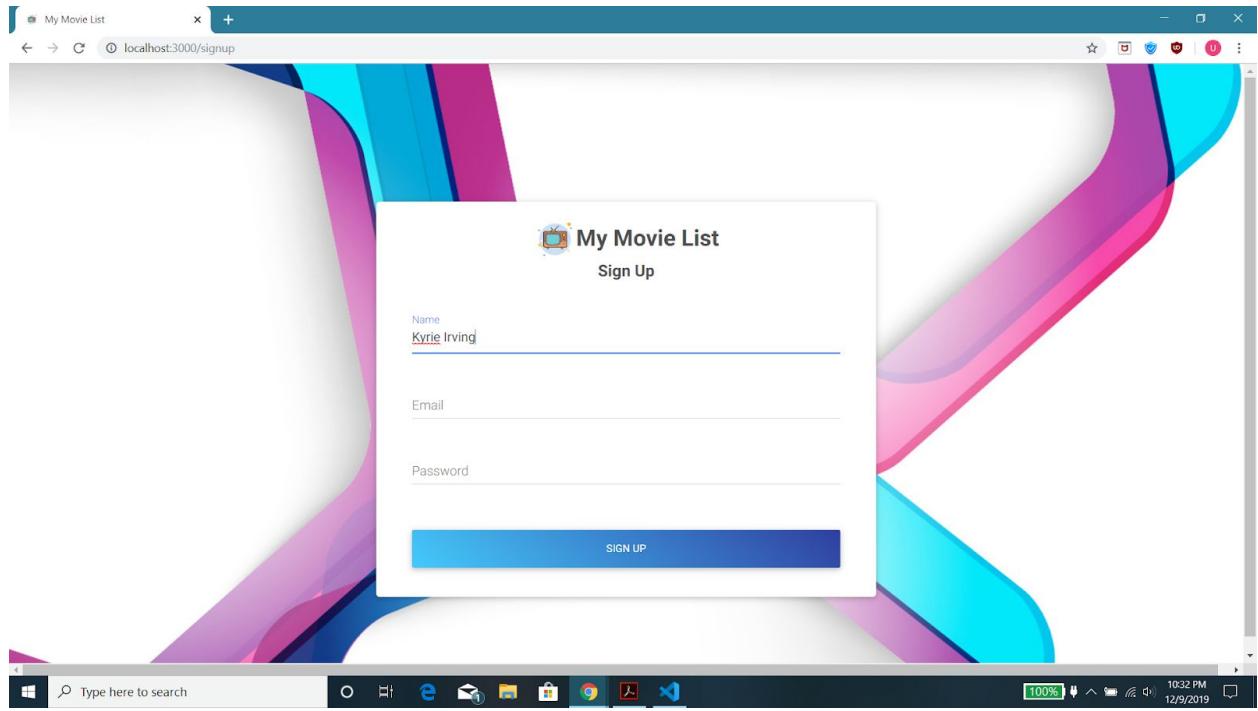
Screenshot of Website:



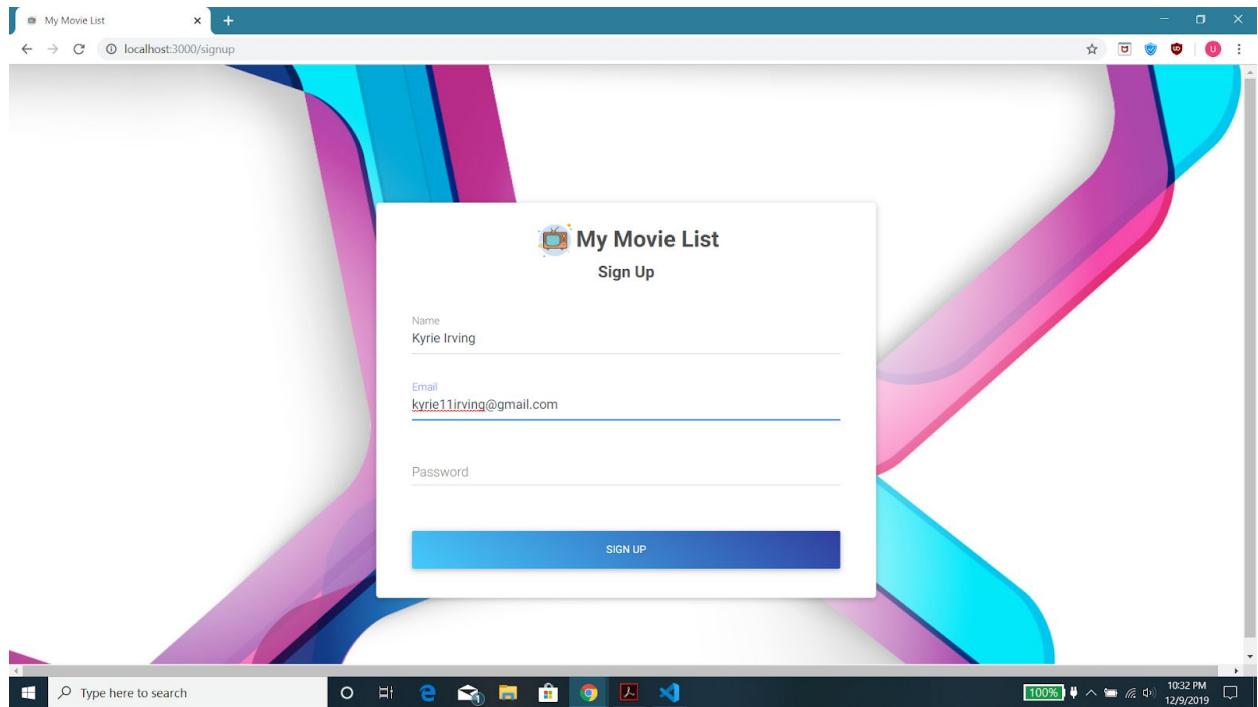
This is the homepage for mymovielist. To register a user, a user must click on the Sign Up link in the bottom right.



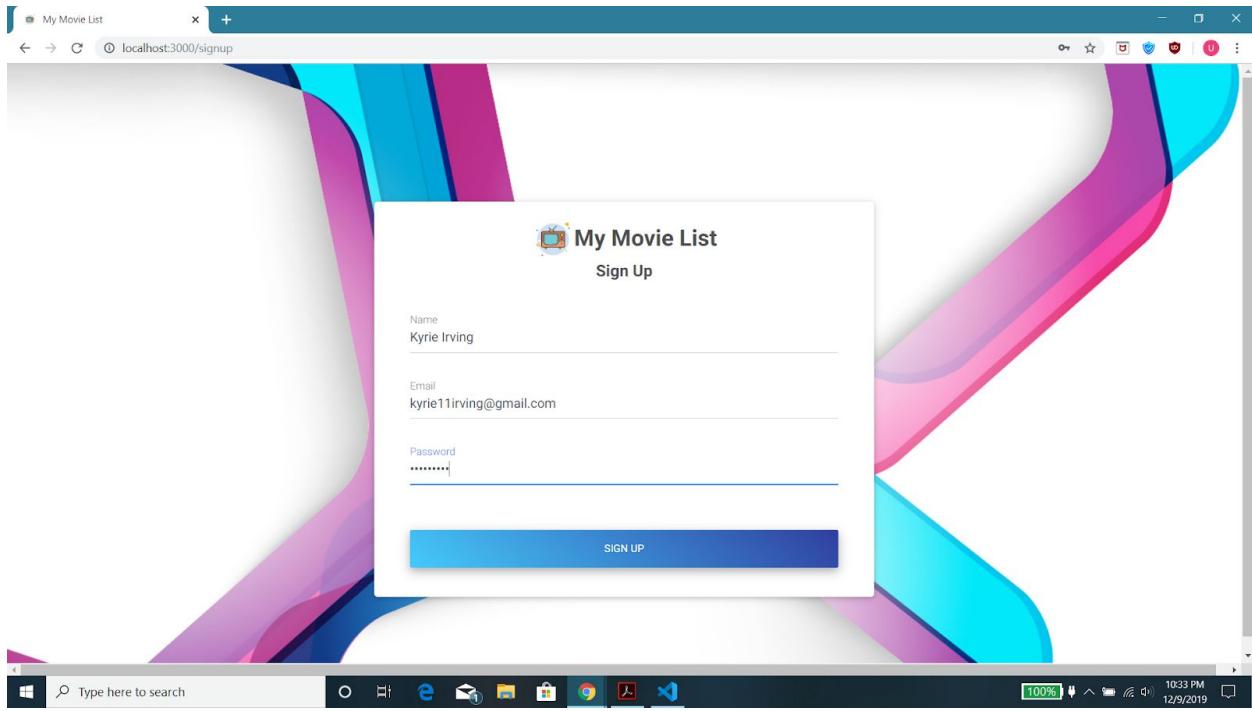
This is the page for Sign Up.



The first input is the name of the user.



Second input of the email of the user.



Third input is the password of the user.

A screenshot of MySQL Workbench showing a query results grid for the "user" table. The grid displays columns: email, password, and username. Data rows include:

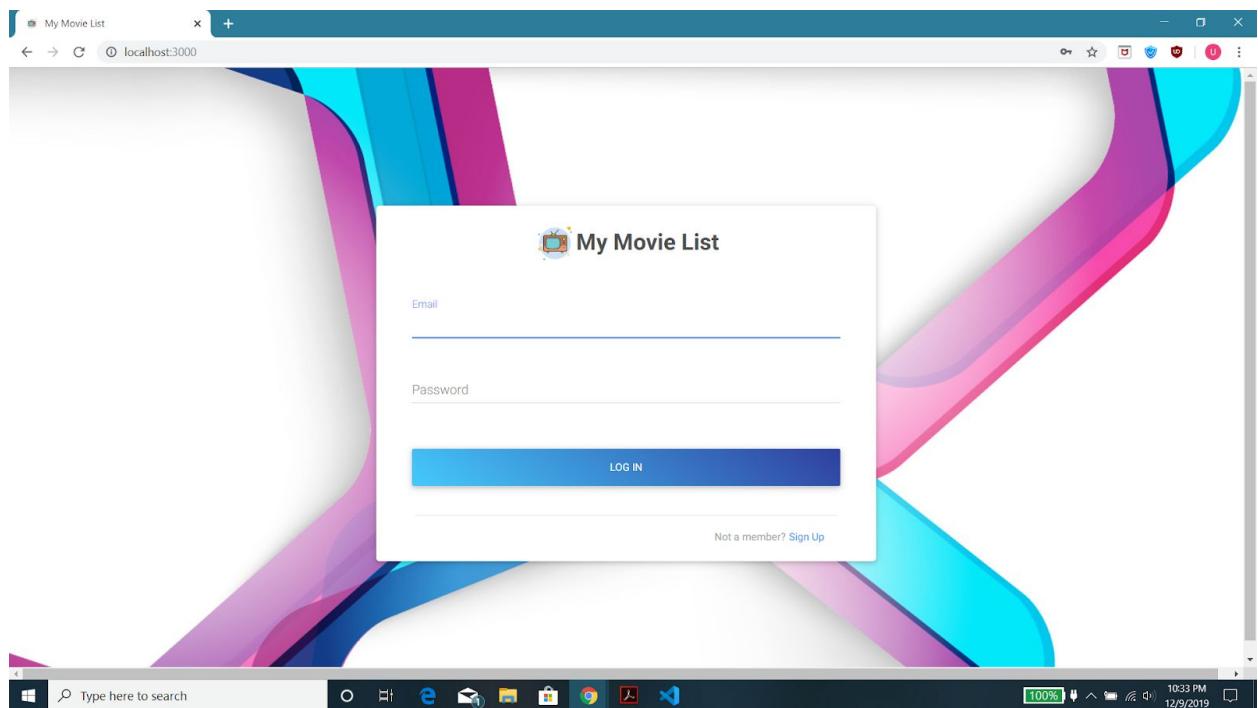
email	password	username
george@gmail.com	pg13	george13
james@gmail.com	leadscore13	harden13
john@gmail.com	12345	john
kamara@gmail.com	drem941v41	alvinkamara
kyle@gmail.com	lowriya	kyle07
kyrie11irving@gmail.com	handlegod	Kyrie Irving
levine@gmail.com	timberwolves	zaiklavine
leonard@gmail.com	kaw	kawhiale...
pooyi@gmail.com	147	pooyitr
williams@gmail.com	sweet6ou	23owill
wu@gmail.com	123456	mike wu
NULL	NULL	NULL

The "Action Output" pane shows three SELECT queries executed successfully.

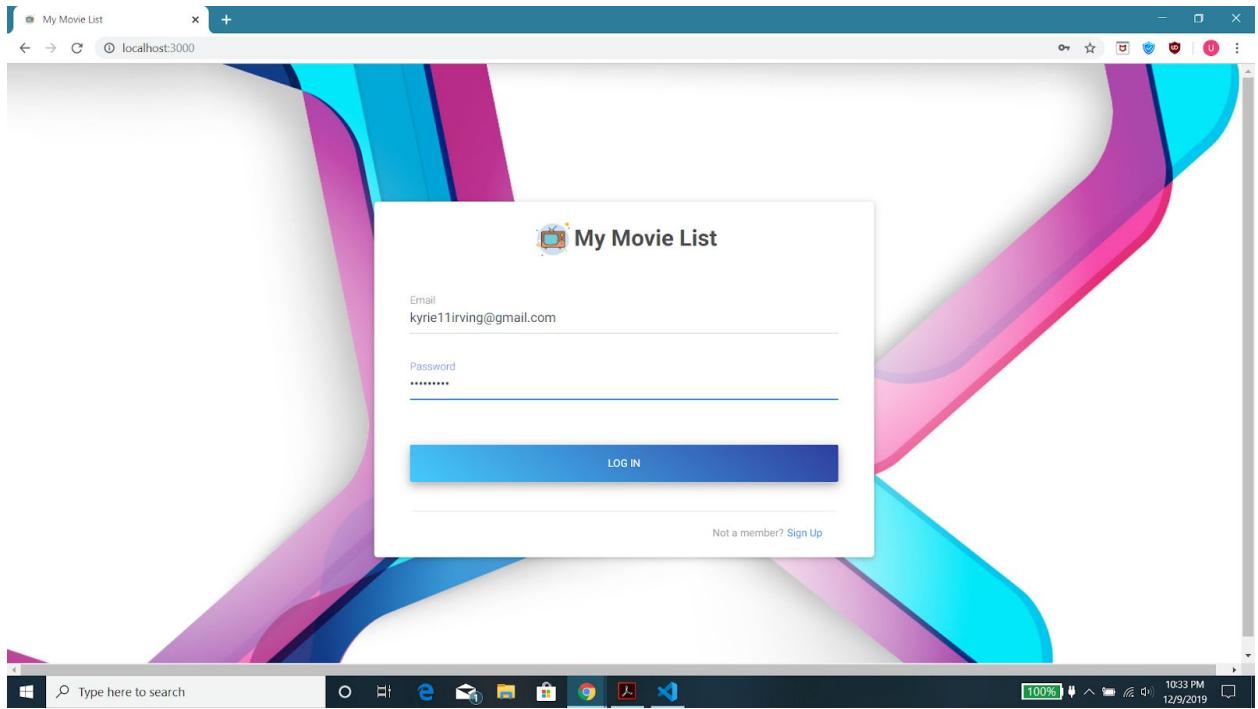
A user is successfully added to the database

```
221 app.get("/signup", (req, res) => {
222   const { email, password, username } = req.query;
223   const INSERT_USER = `INSERT INTO User VALUES('${email}', '${password}', '${username}')`;
224   db.query(INSERT_USER, (err, results) => {
225     if (err) {
226       return res.send(err);
227     } else {
228       return res.send("user sucessfully added");
229     }
230   });
231 });
232
```

This query is sent from the backend to add a user to the user table



After clicking on the Sign Up button the user will be redirected to Log In page.



After putting in credentials for a user, the user will be logged into the website.

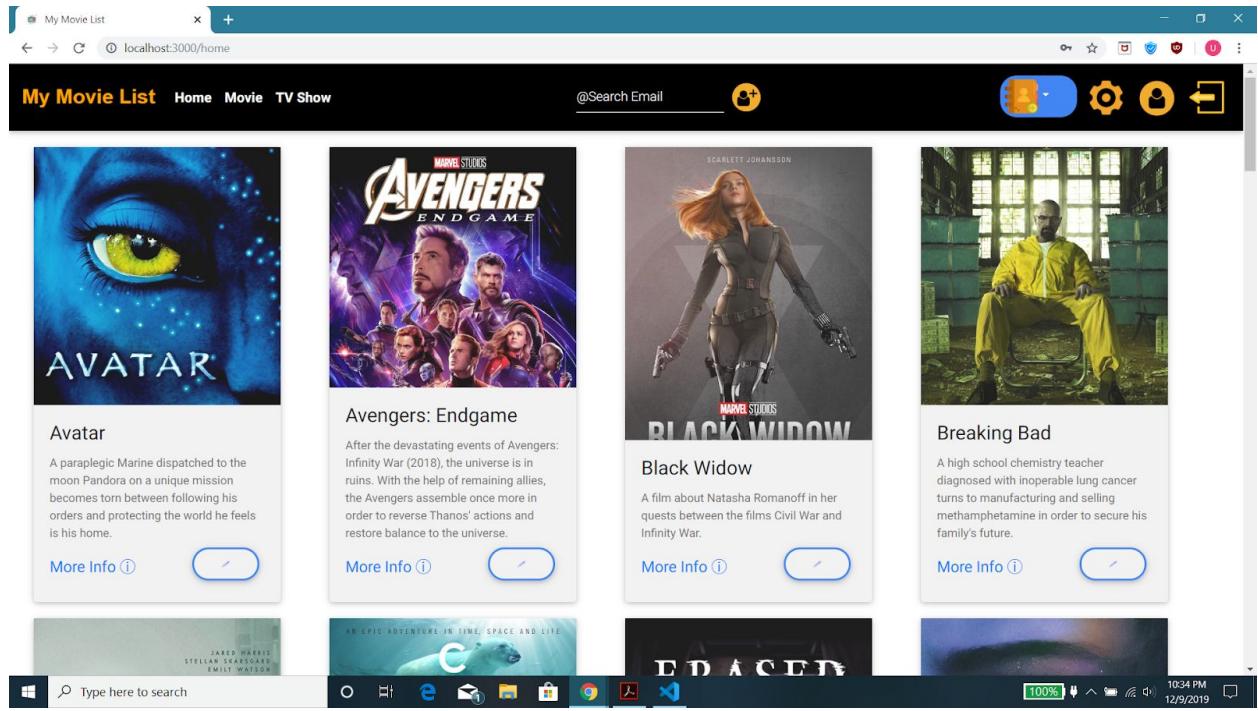
```
79 app.get("/login", (req, res) => {
80   const { email } = req.query;
81   db.query(`SELECT * FROM User WHERE email = '${email}'`, (err, result) => {
82     if (err) {
83       return res.send(err);
84     } else {
85       return res.json({
86         data: result
87       });
88     }
89   });

```

This query gets the user with the email.

```
116  login = async _ => {
117    const { user } = this.state;
118    await fetch(`http://localhost:4040/login?email=${user.email}`).then(res =>
119      res.json().then(res =>
120        this.setState(
121          {
122            qResult: res.data
123          },
124          () =>
125            this.state.qResult.map(db =>
126              this.setState({
127                email: db.email,
128                password: db.password,
129                name: db.username
130              })
131            )
132          )
133        );
134        if (this.state.password === this.state.user.password) {
135          this.setState({ auth: true });
136        }
137      );
138    );
139  }
140 }
```

The frontend check the credentials of the user and authorizes the user to log into the website.



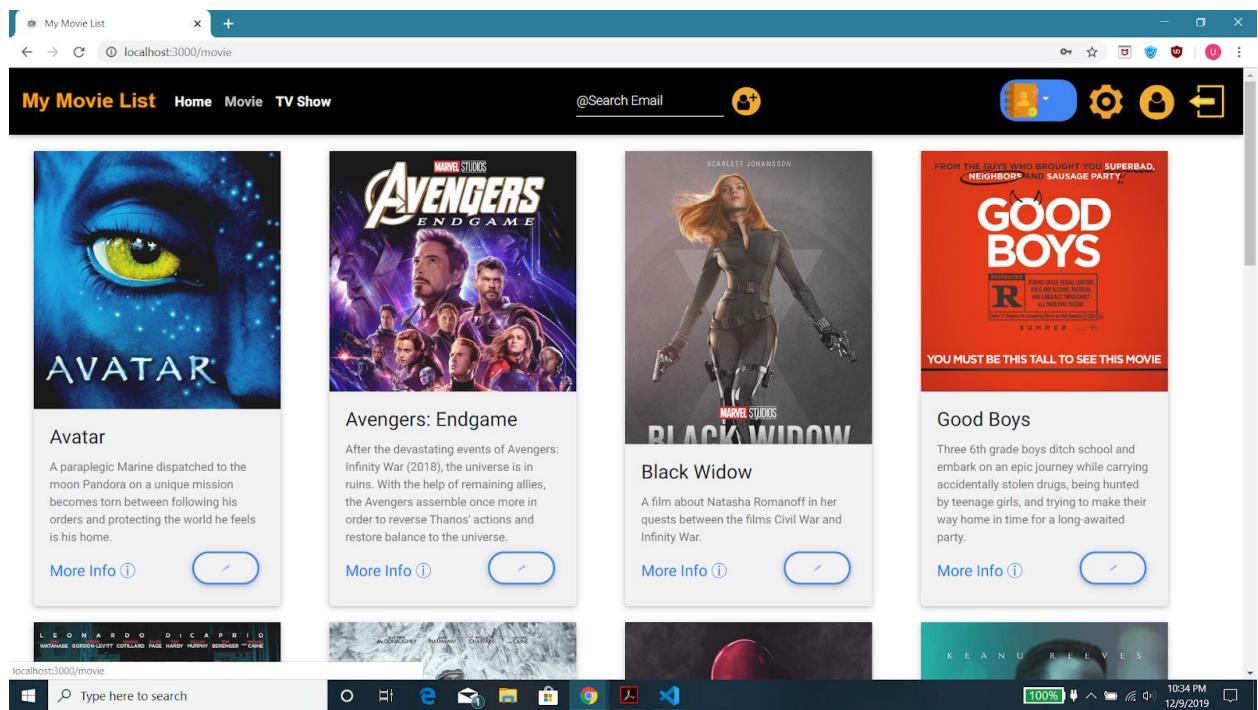
This is the homepage of the website with all the movies and tv series.

```

66
67 app.get("/content", (req, res) => {
68   db.query("SELECT * FROM CONTENT", (err, results) => {
69     if (err) {
70       return res.send(err);
71     } else {
72       return res.json({
73         data: results
74       });
75     }
76   });
77 });
78

```

This query gets the content from content entity set.

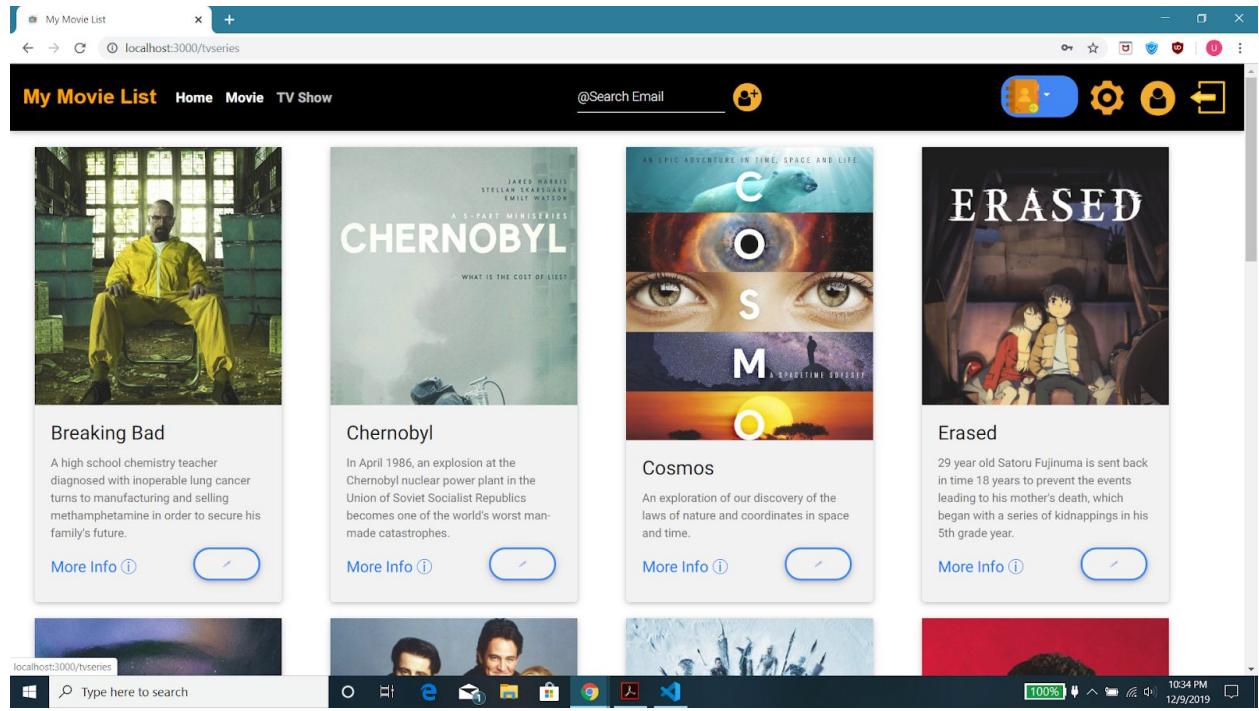


A user can click on “Movie” on the Navbar to list all the movies.

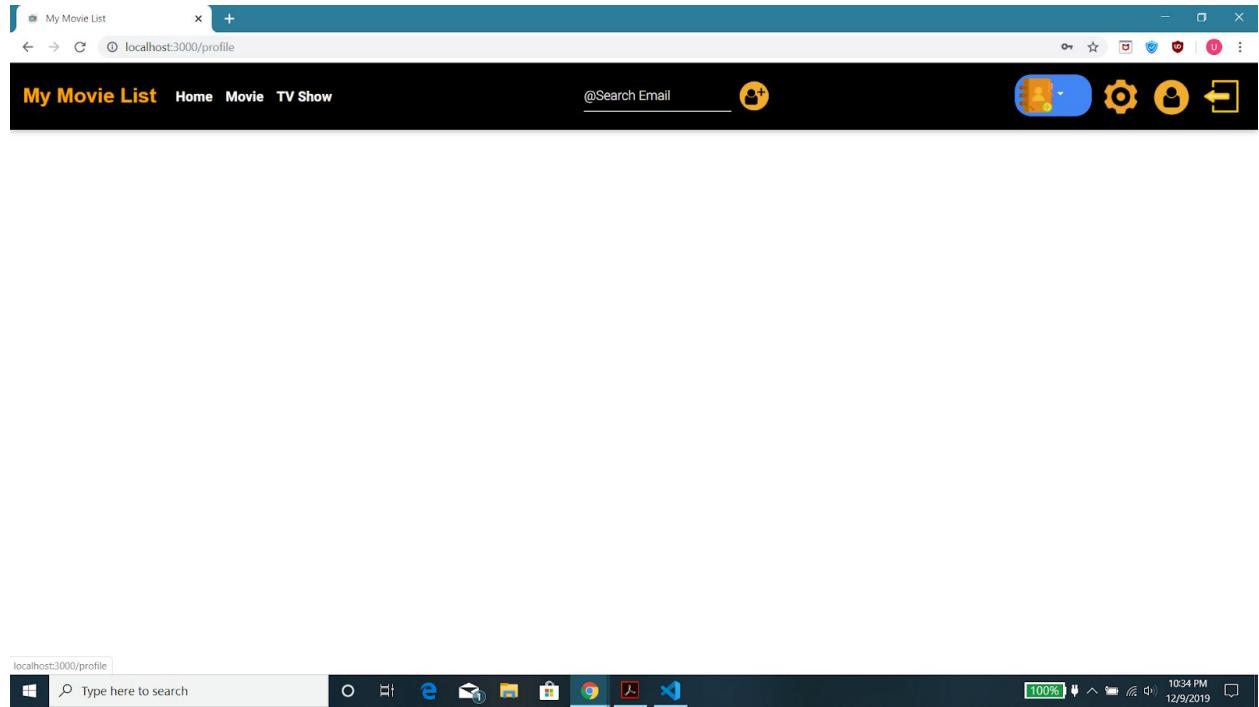
```

206 app.get("/getMovie", (req, res) => {
207   const { movie_contentname, movie_releaseyear } = req.query;
208   const GET_CONTENT = `SELECT * FROM movie WHERE movie_contentname = '${movie_contentname}' AND movie_releaseyear = '${movie_releaseyear}'`;
209
210   db.query(GET_CONTENT, (err, results) => {
211     if (err) {
212       return res.send(err);
213     } else {
214       return res.json({
215         data: results
216       });
217     }
218   });
219 });
220 
```

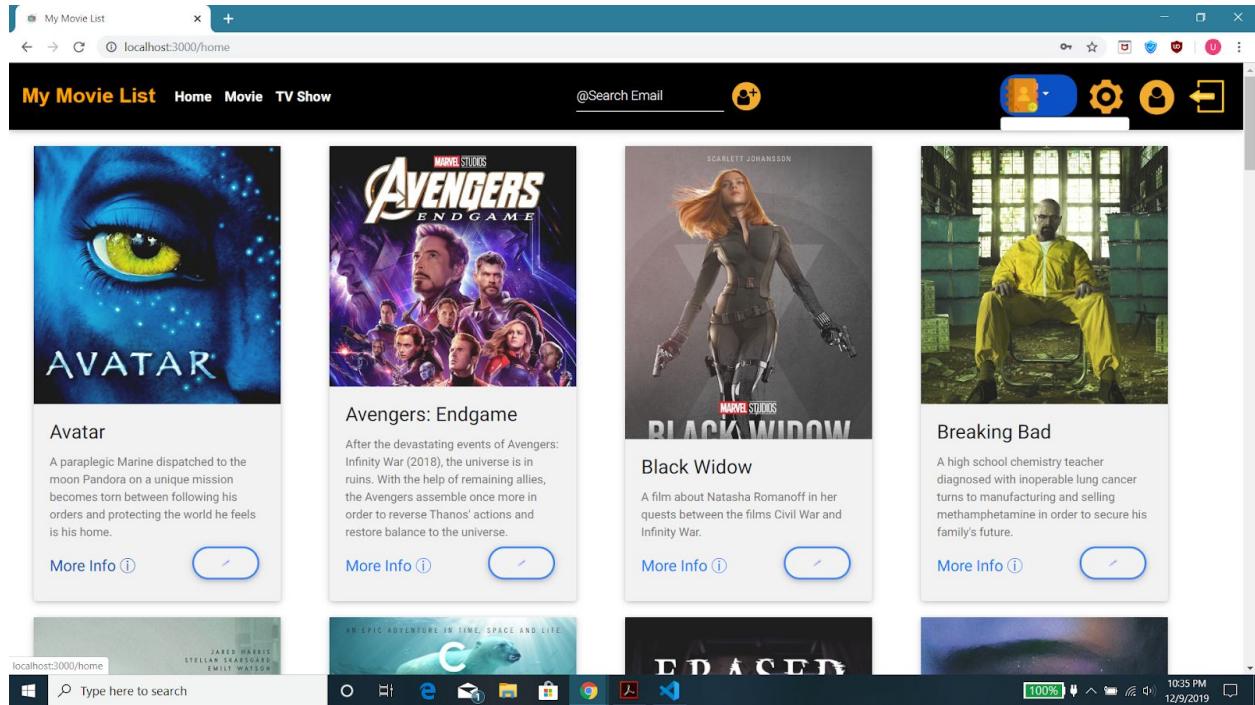
This query gets all the movies from the database.



A user can click on “TV Show” in the Navbar to list all the tv series.



When a user clicks on the profile icon on the right side of the navbar, a user can see all the reviews given by a user. Right now, since we just created a user, we don't have any reviews yet.



When you click on the friends list dropdown menu you can see your friends list, you can see your friends

The screenshot shows the 'My Movie List' application running on a Windows operating system. The window title is 'My Movie List'. The address bar shows the URL 'localhost:3000/home'. The interface includes a top navigation bar with links for 'Home', 'Movie', and 'TV Show', and a search bar labeled '@Search Email'. On the right side of the header are icons for profile, settings, and navigation. Below the header, there are four movie cards displayed in a grid:

- Avatar**: A paraplegic Marine dispatched to the moon Pandora on a unique mission becomes torn between following his orders and protecting the world he feels is his home.
- Avengers: Endgame**: After the devastating events of Avengers: Infinity War (2018), the universe is in ruins. With the help of remaining allies, the Avengers assemble once more in order to reverse Thanos' actions and restore balance to the universe.
- Black Widow**: A film about Natasha Romanoff in her quests between the films Civil War and Infinity War.
- Breaking Bad**: A high school chemistry teacher diagnosed with inoperable lung cancer turns to manufacturing and selling methamphetamine in order to secure his family's future.

Each movie card features a 'More Info' button with a tooltip and a small circular icon.

From homepage, a user can click on “More Info” on the movie card, and that redirects the user to more info page.

The screenshot shows the 'movieInfo' page for the movie 'Avatar' (2009) within the 'My Movie List' application. The window title is 'My Movie List'. The address bar shows the URL 'localhost:3000/movieInfo'. The interface includes a top navigation bar with links for 'Home', 'Movie', and 'TV Show', and a search bar labeled '@Search Email'. On the right side of the header are icons for profile, settings, and navigation. The main content area displays the movie poster for 'Avatar' (2009) on the left and detailed information on the right:

Avatar (2009)

Detail
A paraplegic Marine dispatched to the moon Pandora on a unique mission becomes torn between following his orders and protecting the world he feels is his home.

Genre
Adventure

Studio
Twentieth Century Fox

Length
162 Minutes

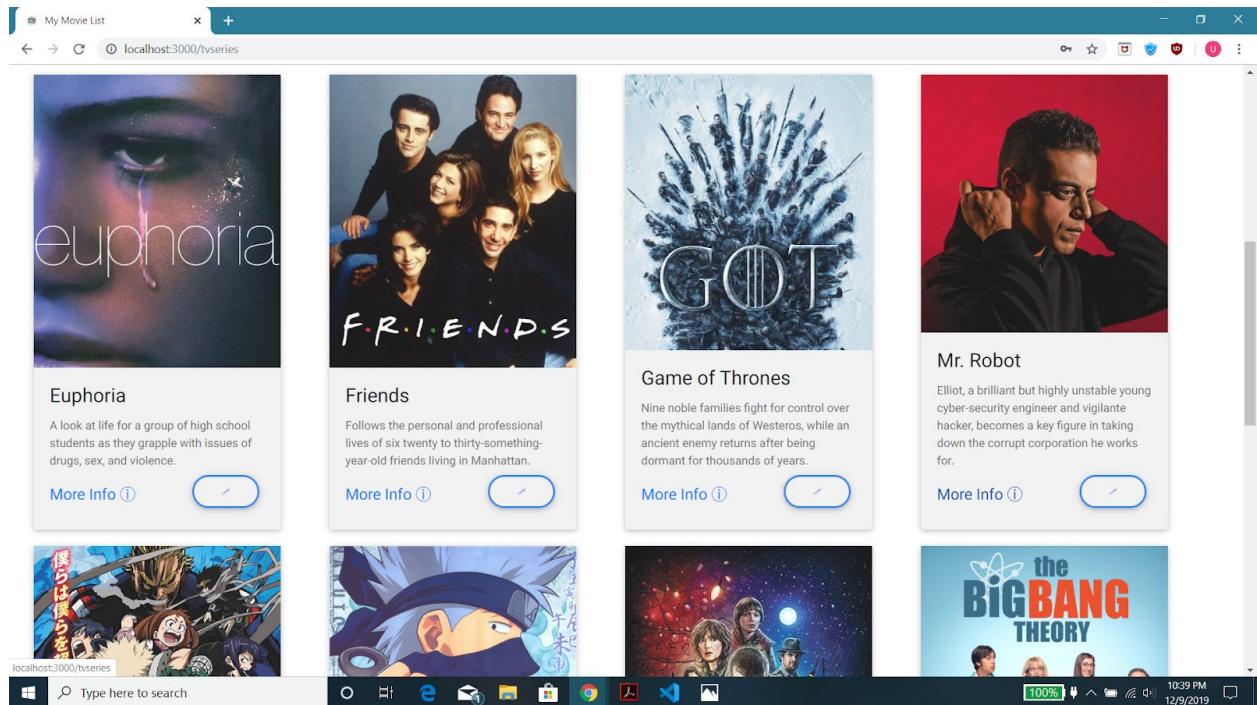
Director
James Cameron

Actors
Sam Worthington, Zoe Saldana, Sigourney Weaver, Stephen Lang, Michelle Rodriguez, Giovanni Ribisi

More info page gives you a detailed summary of the movie, along with genre, studio, length, directors, and actors.

```
159 app.get("/getContentUnionMovie", (req, res) => {
160   const { contentname, releaseyear } = req.query;
161   console.log(contentname);
162   const GET_CONTENT = `SELECT content.contentname, content.releaseyear, contentgenre, studioname, poster, movie_length, directorname, description
163     FROM content, content_director JOIN movie ON contentname = movie.contentname AND releaseyear = movie.releaseyear
164     WHERE content.contentname = content_director.contentname AND content.contentname = '${contentname}' AND content.releaseyear = '${releaseyear}'`;
165   db.query(GET_CONTENT, (err, results) => {
166     if (err) {
167       return res.send(err);
168     } else {
169       return res.json({
170         data: results
171       });
172     }
173   });
174 });
175
```

This query returns all the information for movie content.



From tv series, you can also click on more info, and it gives more detail about a tv series.



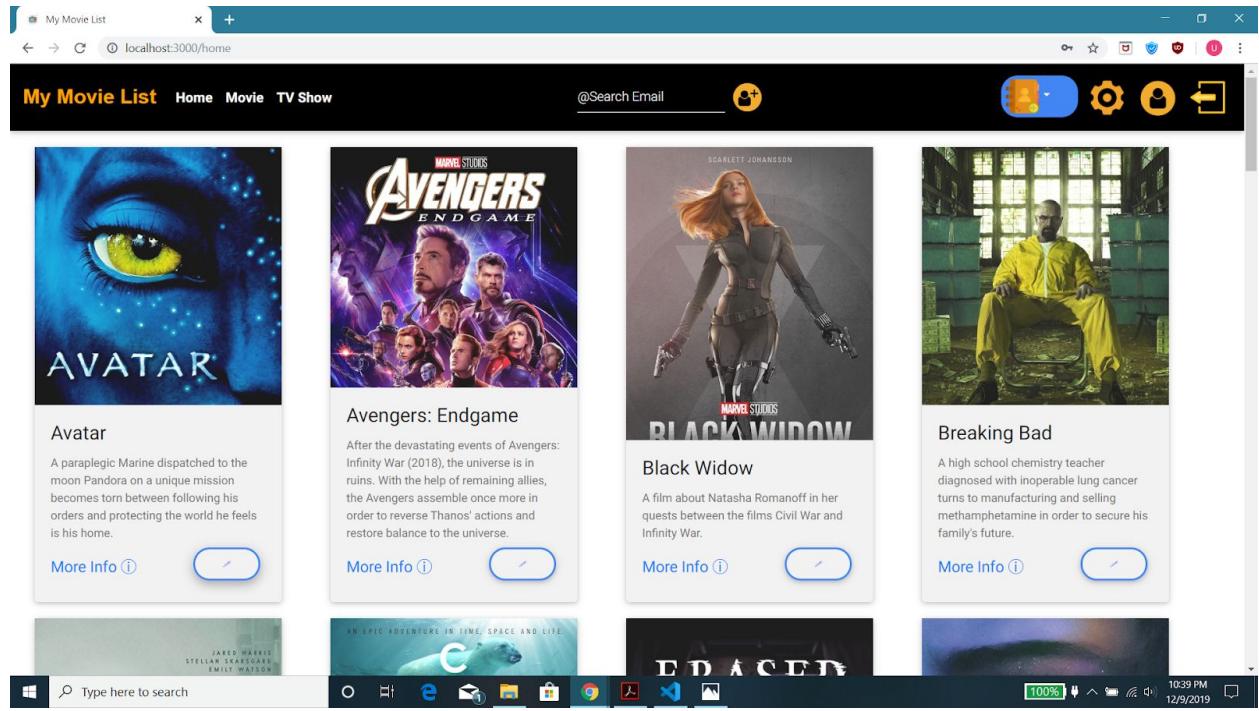
“More info” of a tv series gives you Episodes, instead of movie length.

```

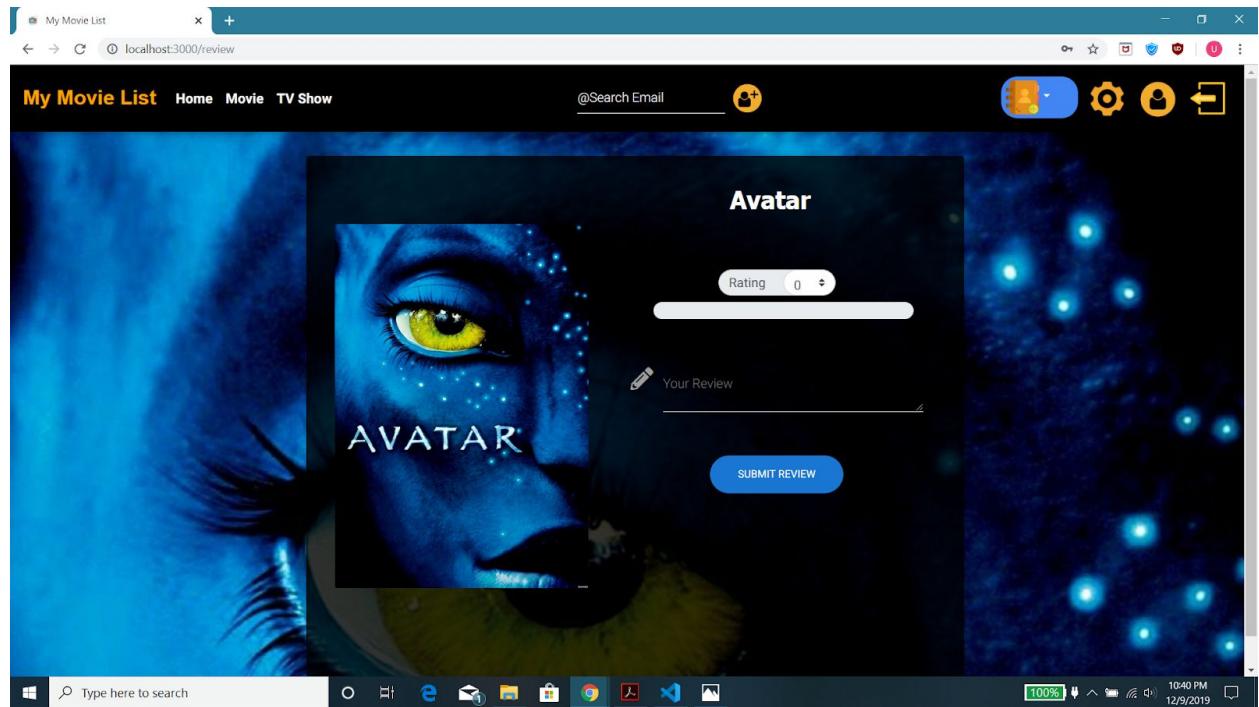
176 app.get("/getContentUnionTvSeries", (req, res) => {
177   const { contentname, releaseyear } = req.query;
178   const GET_CONTENT = `SELECT content.contentname, content.releaseyear, contentgenre, studioname, poster, no_of_episodes, directortname, description
179   FROM content, content_director JOIN tseries ON contentname = tseries_contentname AND releaseyear = tseries_releaseyear
180   WHERE content.contentname = content_director.contentname AND content.contentname = '${contentname}' AND content.releaseyear = '${releaseyear}'`;
181   db.query(GET_CONTENT, (err, results) => {
182     if (err) {
183       return res.send(err);
184     } else {
185       return res.json({
186         data: results
187       });
188     }
189   });
190 });
191

```

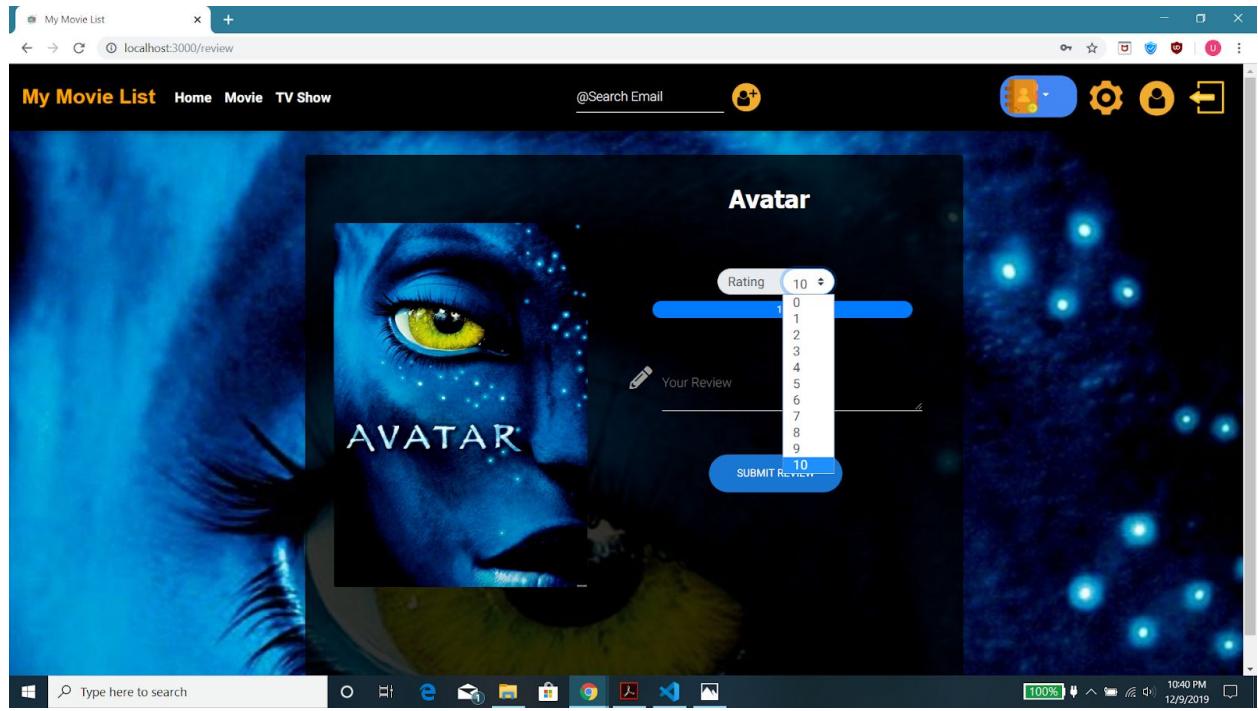
This returns the data for tv show content



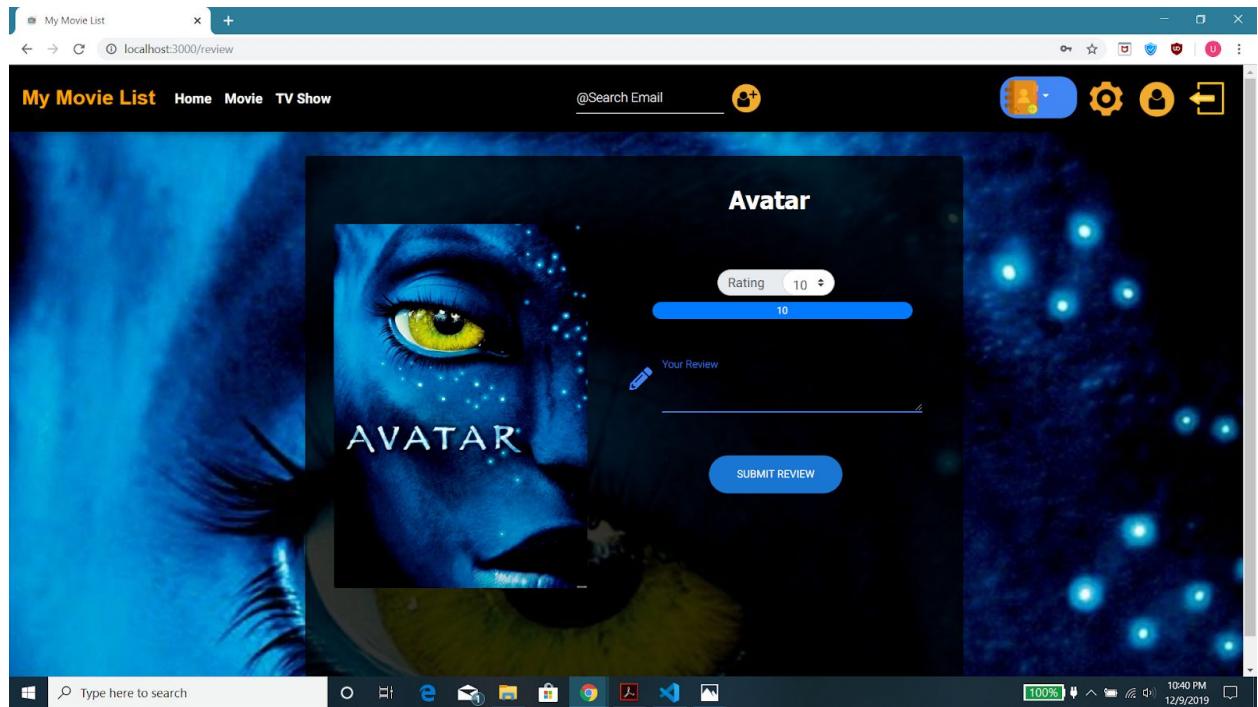
From homepage, a user can also click on the review button, and it will take a user to review page



A review page gives you two inputs: rating and review.



From the drop down a user can rate a movie from a scale (1 to 10).



The review bar progresses according to the rating given by a user.



A user can write a review for a movie, and once a user click on submit the review is added to the backend MySql, and the user is redirected to the homepage.

```
147 app.get("/addReview", (req, res) => {
148   const { email, contentname, releaseyear, review, rating } = req.query;
149   const INSERT_USER_CONTENT = `INSERT INTO user_content VALUES('${email}', '${contentname}', '${releaseyear}', '${review}', '${rating}')`;
150   db.query(INSERT_USER_CONTENT, (err, results) => {
151     if (err) [
152       return res.send(err);
153     ] else {
154       return res.send("review sucessfullt added");
155     }
156   });
157 });
158 }
```

This query adds review to the backend for a particular user.

1 • SELECT * FROM cs157a.user_content;

	email	contentname	releaseyear	review	rating
ent	ben@gmail.com	Cosmos	2014	Tyson makes it the best. But OG Cosmos by Carl Sagan is legendary	10
ent_actor	ben@gmail.com	It	2017	Great jump scare. Stephen King is a king of horror movies.	7
ent_director	ben@gmail.com	John Wick	2014	Keanu Reeves is a legend, period.	5
tor	ben@gmail.com	John Wick: Chapter 3 – Parabellum	2019	A legend never dies.	9
ie	ben@gmail.com	Ready Player One	2018	Liked the concept of VR games coming to real life.	8
ries	bob@gmail.com	Naruto	2002	one of the best anime of all time	10
_content	carmelo.anthony@gmail.com	The Big Bang Theory	2007	Hilarious show. Gets funnier every time I watch it. New seasons are kinda slow tho.	7
_friend	carmelo.anthony@gmail.com	The Boys	2019	It was a good show. Finished in one day.	10
rocedures	george@gmail.com	Erasers	2016	This anime is a master piece. Waiting for the next season.	9
s	george@gmail.com	Inception	2010	I watched it many times. This movie is so well written. The story line is incredible.	8
gh	george@gmail.com	Mr. Robot	2015	Inspires me every day to become a hacker. Solid 10/10 show	10
chemas	john@gmail.com	Avengers: Endgame	2019	Very good movie.	8
ected	john@gmail.com	Stranger Things	2016	Great thriller. I like demogorgons.	10
	kyrie11irving@gmail.com	Avatar	2009	Really good movie. Show the beauty of our universe.	10
	leonard@gmail.com	Spider-Man: Far from Home	2019	The scenery is incredible. Like the direction where marvel phase 4 is headed.	10
	poiy@gmail.com	Avengers: Endgame	2019	Deserves 10 but my favorite avenger dies so I am giving it a 9. Overall, a great movie.	9
	NULL	NULL	NULL	NULL	NULL

As can be seen a review is added to the backend under the email of "kyrie11irving@gmail.com", contentname as "Avatar" and releaseyear "2009".

My Movie List x + localhost:3000/home

My Movie List Home Movie TV Show @Search Email 🔒

Avatar

A paraplegic Marine dispatched to the moon Pandora on a unique mission becomes torn between following his orders and protecting the world he feels is his home.

More Info ⓘ

Avengers: Endgame

After the devastating events of Avengers: Infinity War (2018), the universe is in ruins. With the help of remaining allies, the Avengers assemble once more in order to reverse Thanos' actions and restore balance to the universe.

More Info ⓘ

Black Widow

A film about Natasha Romanoff in her quests between the films Civil War and Infinity War.

More Info ⓘ

Breaking Bad

A high school chemistry teacher diagnosed with inoperable lung cancer turns to manufacturing and selling methamphetamine in order to secure his family's future.

More Info ⓘ

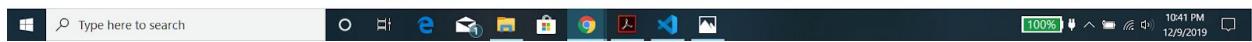
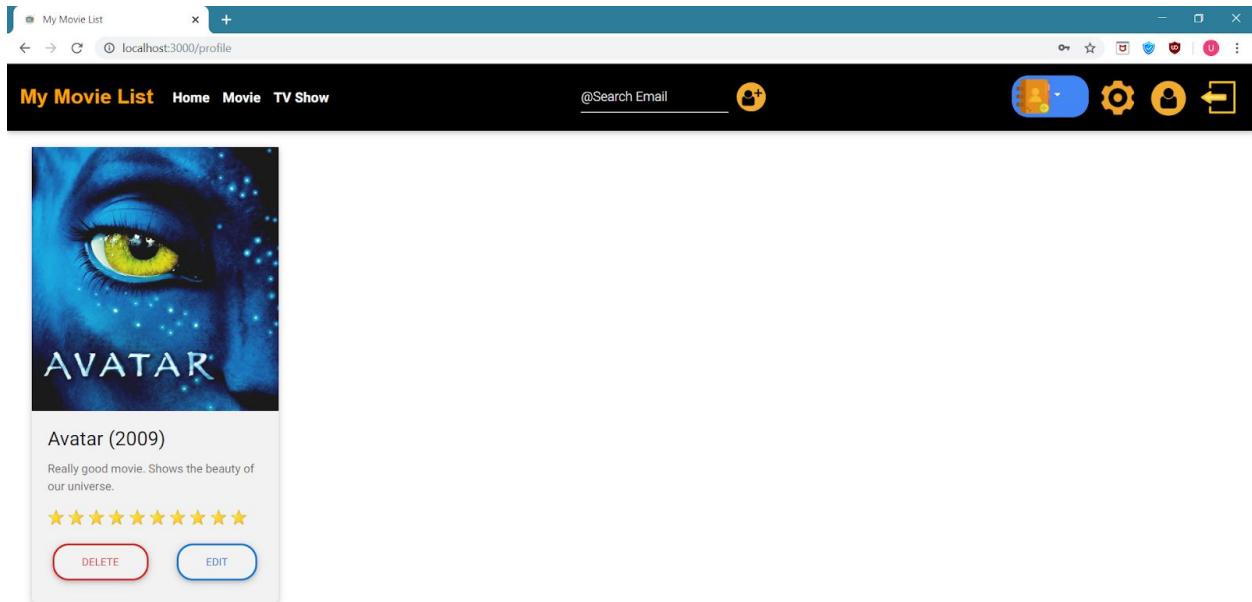
DEAINED

JARED HARRIS STELLAN SKARSGÅRD EMILY VANCAMP

AN EPIC ADVENTURE IN TIME, SPACE AND LIFE

Type here to search 10:41 PM 12/9/2019

After redirecting to the homepage, user can visit his or her profile page to see reviewed content.

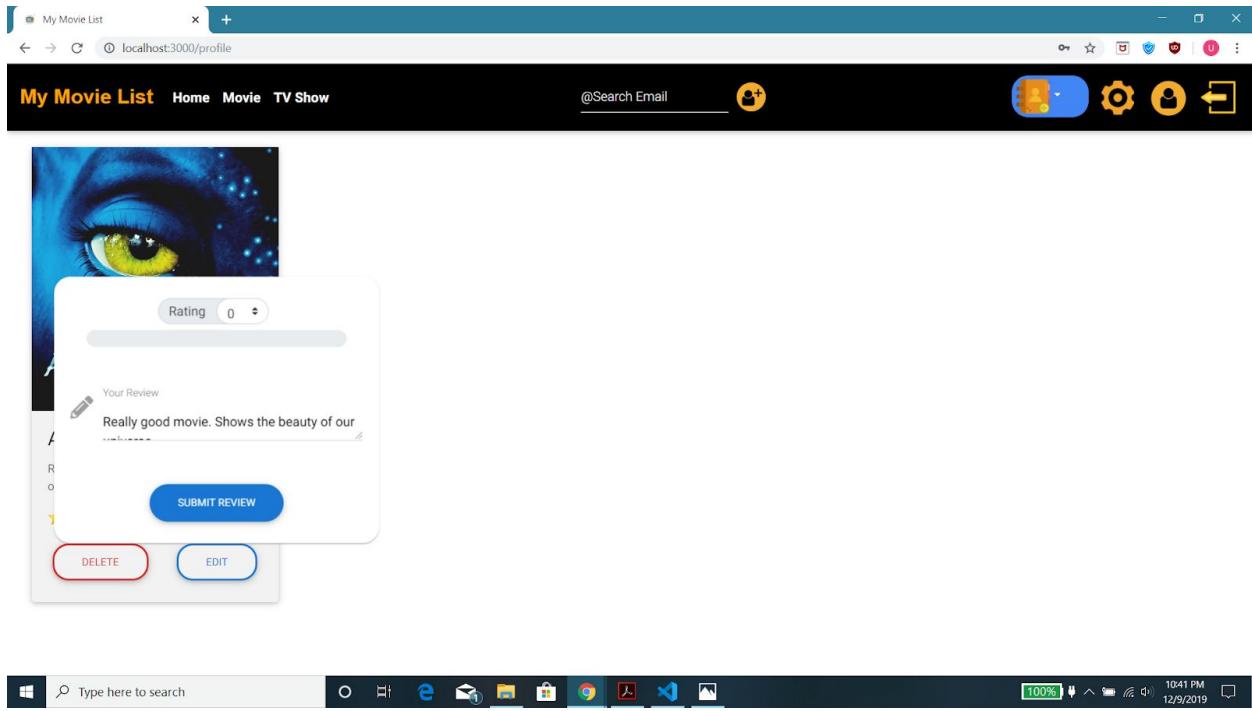


After click on the profile page a user can see the review and rating given by him or her under profile.

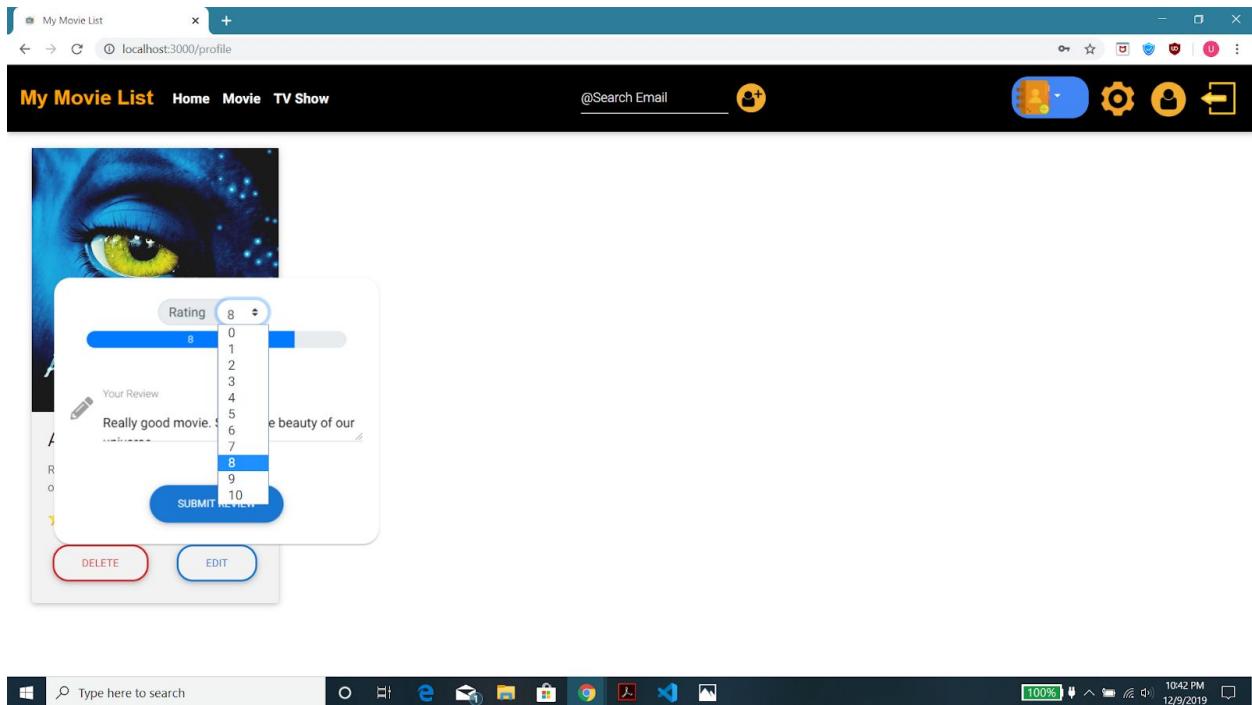
```

251 app.get('/getReview', (req, res)=>{
252   const{email} = req.query;
253   const GET_CONTENT ='SELECT content.contentname, content.releaseyear, content.poster, review, rating
254   FROM content JOIN user_content ON content.contentname = user_content.contentname AND content.releaseyear = user_content.releaseyear
255   WHERE user_content.email = '${email}'';
256   db.query(GET_CONTENT, (err, results)=>{
257     if(err){
258       return res.send(err);
259     }
260     else{
261       return res.json({
262         data: results
263       })
264     }
265   });
266 }
267 }
268 
```

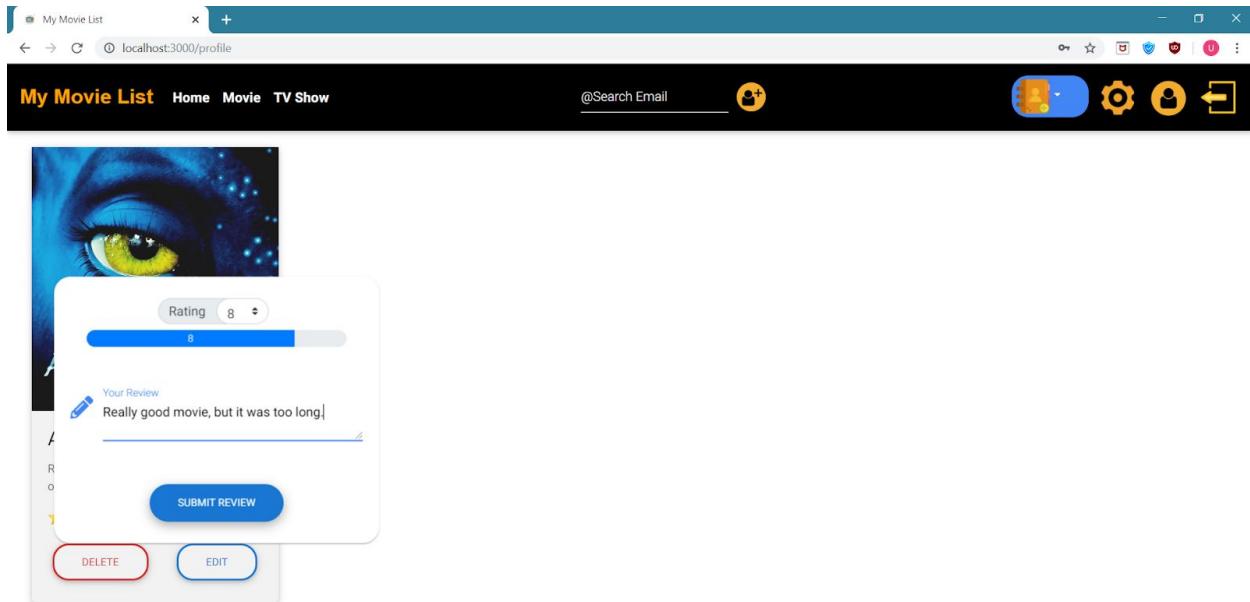
This query returns the list of reviews for a user.



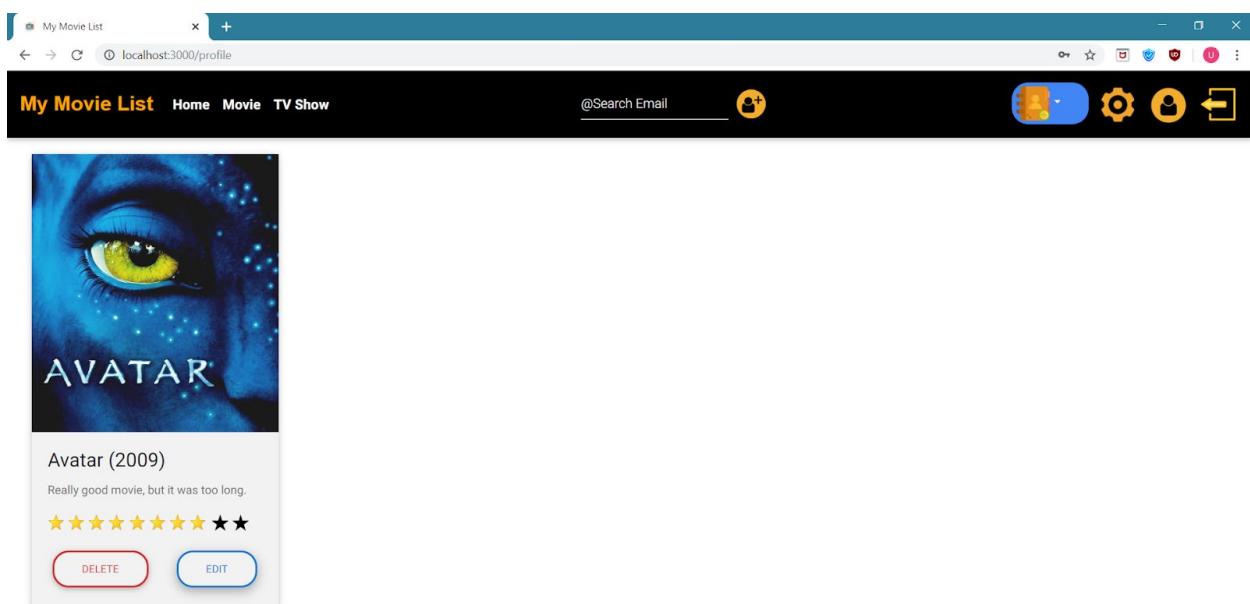
A user is allowed to edit a review by clicking on the review button.



A user can modify the rating by the dropdown scaled (1 to 10).



A user can also edit the review and click on submit.



After clicking on submit, the page is automatically refreshed and change the review and rating on the page and the backend.

```

281 app.get("/deleteReview", (req, res) => {
282   const { email, contentname, releaseyear } = req.query;
283   const DELETE_REVIEW = `DELETE FROM user_content WHERE email='${email}' AND contentname ='${contentname}' AND releaseyear='${releaseyear}'`;
284   db.query(DELETE_REVIEW, (err, results) => {
285     if (err) {
286       return res.send(err);
287     } else {
288       return res.send("Review Deleted");
289     }
290   });
291 });
292
293 app.get("/updateReview", (req, res) => {
294   const { email, contentname, releaseyear, review, rating } = req.query;
295   const UPDATE REVIEW =
296     `UPDATE user_content SET review='${review}', rating = '${rating}' WHERE email = '${email}' AND contentname = '${contentname}' AND releaseyear = '${releaseyear}'`;
297   db.query(UPDATE REVIEW, (err, results) => {
298     if (err) {
299       return res.send(err);
300     } else {
301       return res.send("Successfully Updated");
302     }
303   });
304 });

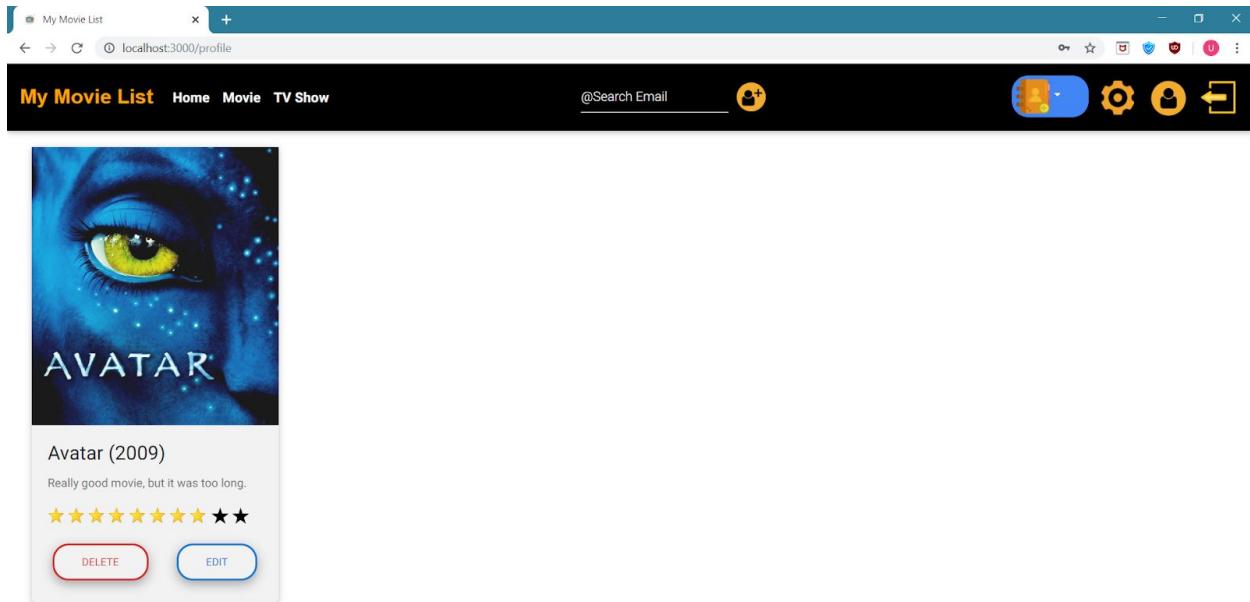
```

These two queries are used to update or delete a review for a user

email	contentname	releaseyear	review	rating
bob@gmail.com	Naruto	2002	one of the best anime of all time	10
carmelo.anthony@gmail.com	The Big Bang Theory	2007	Hilarious show. Gets funnier every time I watch it. New seasons are kinda slow tho.	7
carmelo.anthony@gmail.com	The Boys	2019	It was a good show. Finished in one day.	10
george@gmail.com	Erasied	2016	This anime is a master piece. Waiting for the next season.	9
george@gmail.com	Inception	2010	I watched it many times. This movie is so well written. The story line is incredible.	8
george@gmail.com	Mr. Robot	2015	Inspires me every day to become a hacker. Solid 10/10 show	10
john@gmail.com	Avengers: Endgame	2019	Very good movie.	8
john@gmail.com	Stranger Things	2016	Great thriller. I like demogorgons.	10
kyrie11irving@gmail.com	Avatar	2009	Really good movie, but it was too long.	8
leonard@gmail.com	Spider-Man: Far from Home	2019	The scenery is incredible. Like the direction where marvel phase 4 is headed.	10
poliy@mail.com	Avengers: Endgame	2019	Deserves 10 but my favorite avenger dies so I am giving it a 9. Overall, a great movie.	9
NULL	NULL	NULL	NULL	NULL

As can be seen in the picture the value for review and rating is updated for user

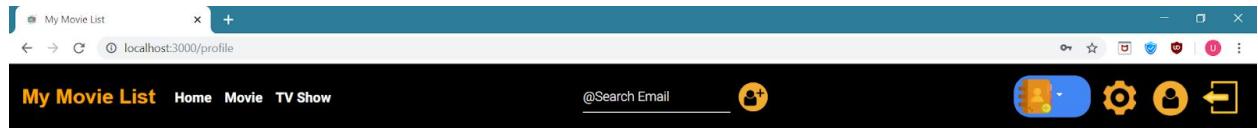
kyrie11irving@gmail.com.



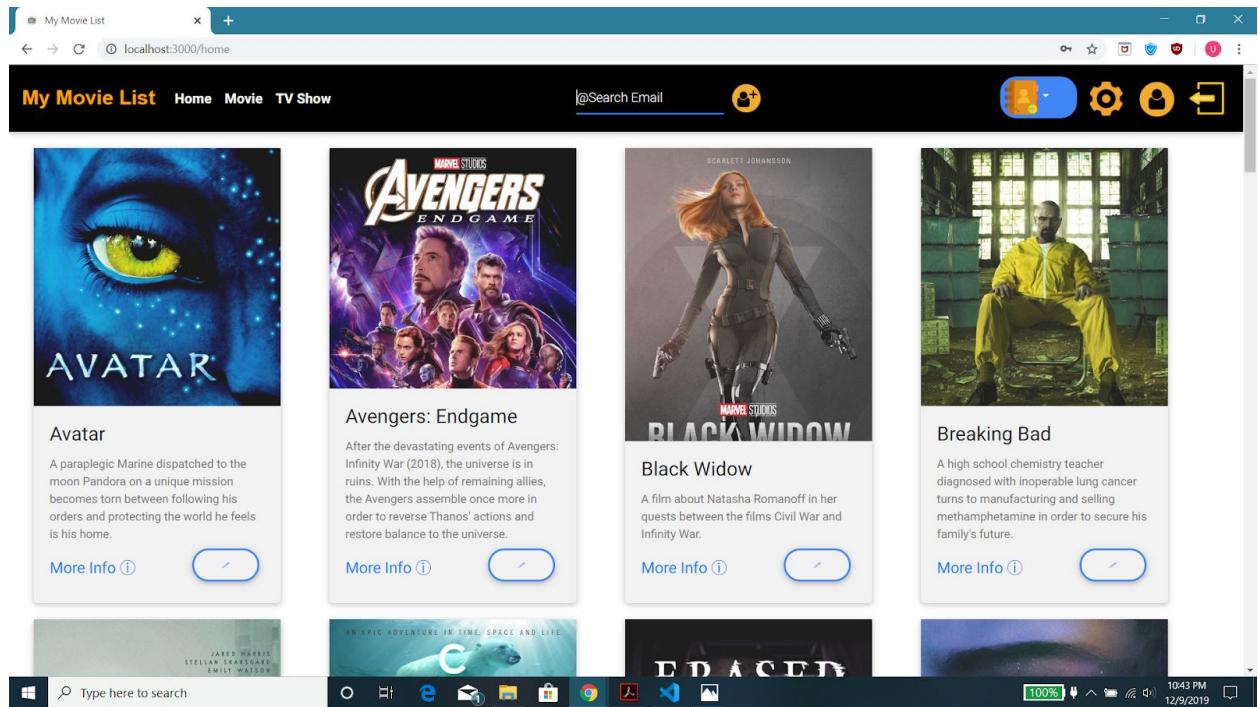
When you click on the delete button, it deletes the movie from a user's list instantly.

email	contentname	releaseyear	review	rating
ben@gmail.com	Cosmos	2014	Tyson makes it the best. But OG Cosmos by Carl Sagan is legendary	10
ben@gmail.com	It	2017	Great jump scare, Stephen King is a king of horror movies.	7
ben@gmail.com	John Wick	2014	Keanu Reeves is a legend, period.	5
ben@gmail.com	John Wick: Chapter 3 – Parabellum	2019	A legend never dies.	9
ben@gmail.com	Ready Player One	2018	Liked the concept of VR games coming to real life.	8
bob@gmail.com	Naruto	2002	one of the best anime of all time	10
carmelo.anthony@gmail.com	The Big Bang Theory	2007	Hilarious show. Gets funnier every time I watch it. New seasons are kinda slow tho.	7
carmelo.anthony@gmail.com	The Boys	2019	It was a good show. Finished in one day.	10
george@gmail.com	Erasied	2016	This anime is a master piece. Waiting for the next season.	9
george@gmail.com	Inception	2010	I watched it many times. This movie is so well written. The story line is incredible.	8
george@gmail.com	Mr. Robot	2015	Inspires me every day to become a hacker. Solid 10/10 show	10
john@gmail.com	Avengers: Endgame	2019	Very good movie.	8
john@gmail.com	Stranger Things	2016	Great thriller. I like demogorgons.	10
leonard@gmail.com	Spider-Man: Far from Home	2019	The scenery is incredible. Like the direction where marvel phase 4 is headed.	10
poiyy@gmail.com	Avengers: Endgame	2019	Deserves 10 but my favorite avenger dies so I am giving it a 9. Overall, a great movie.	9
*	NULL	NULL	NULL	NULL

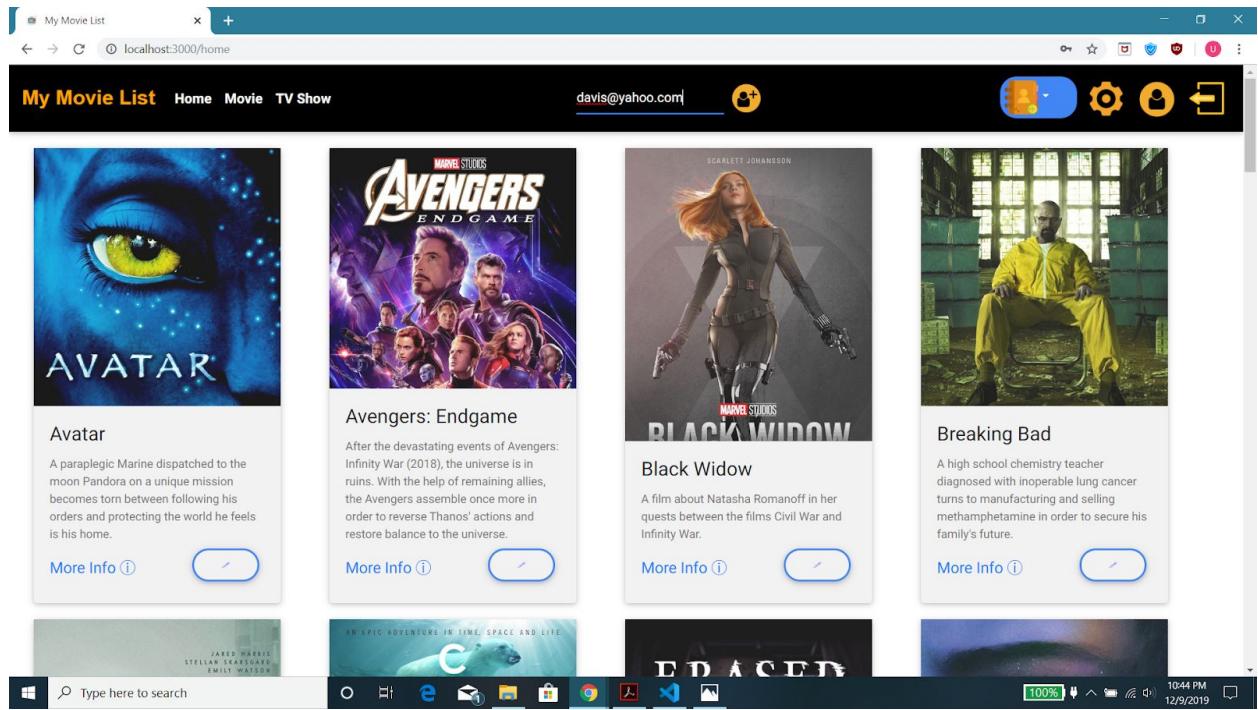
After deletion, the movie review deleted from the mysql table.



The profile page after the user "kyrie11irving@gmail.com" deletes the movie refreshes and shows that the movie review is deleted.

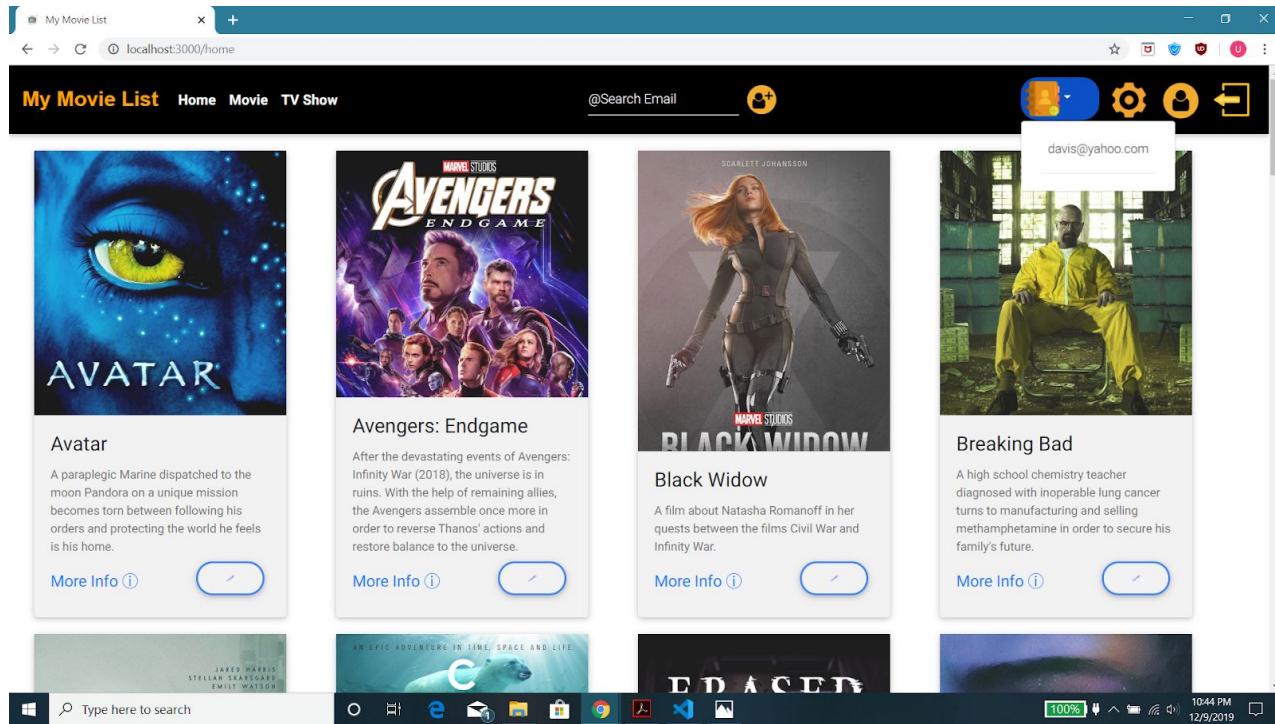


From the search bar a user can add other users as friends.



[davis@yahoo.com](#) is another user in the database, we can add the user by typing in

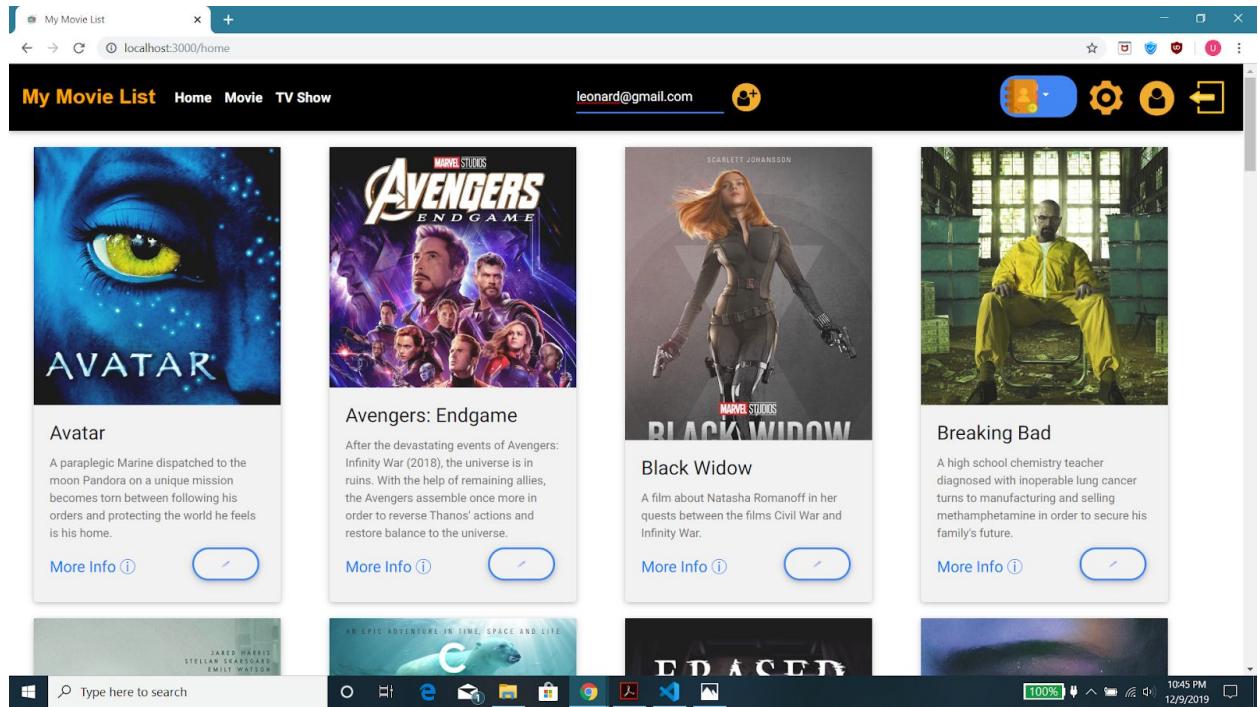
[davis@yahoo.com](#) in the input of search bar and click on the add user button next to it to add a friend.



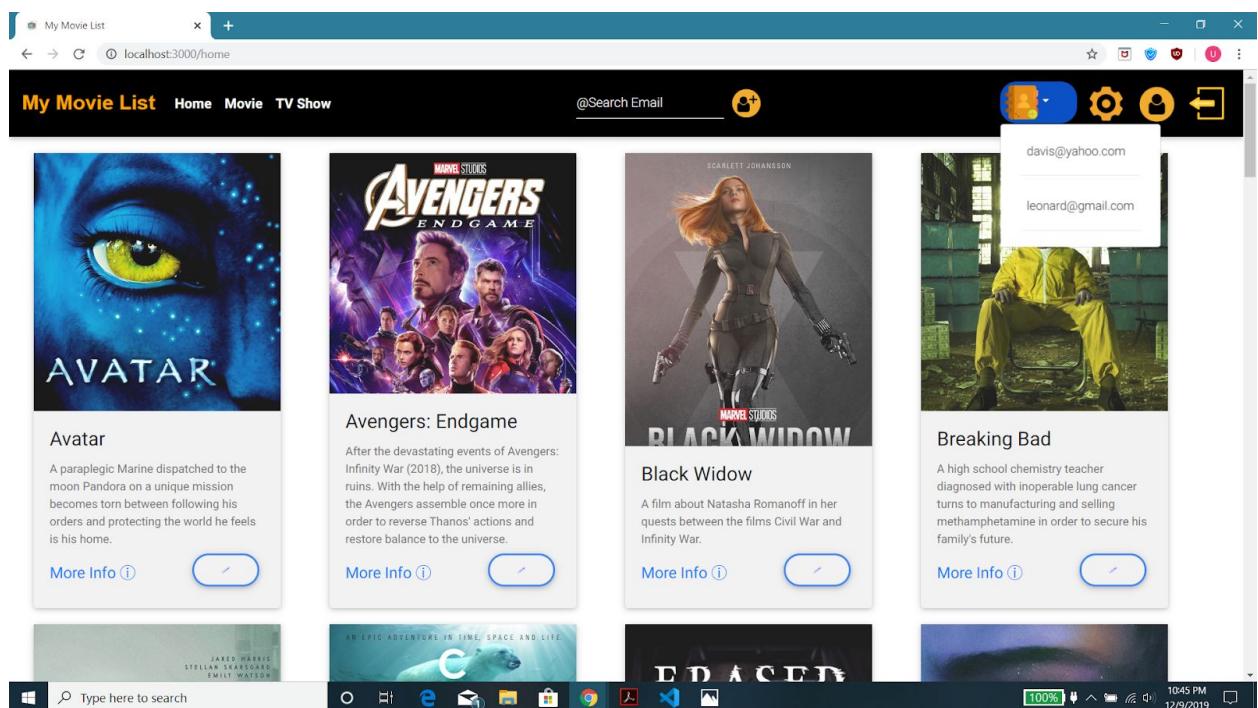
After adding a user, the dropdown menu of friend's list shows davis@yahoo.com.

Result Grid		Filter Rows:	Export:	Wrap Cell Content:
email	friend_email			
ben@gmail.com	john@gmail.com			
wu@gmail.com	ben@gmail.com			
wu@gmail.com	john@gmail.com			
carmelo.anthony@gmail.com	george@gmail.com			
leonard@gmail.com	george@gmail.com			
leonard@gmail.com	leonard@gmail.com			
leonard@gmail.com	carmelo.anthony...			
leonard@gmail.com	davis@yahoo.com			
carmelo.anthony@gmail.com	james@gmail.com			
carmelo.anthony@gmail.com	williams@gmail.com			
carmelo.anthony@gmail.com	kamara@gmail.com			
kyrie11irving@gmail.com	davis@yahoo.com			

As can be seen, now kyrie11irving@gmail.com has a friend davis@yahoo.com in the database.



A user can add many friends, user can add another user as a friend. As in this, a user adds leonard@gmail.com to his friend's list.



From the dropdown menu we can see the leonard@gmail.com is also added to the friend's list.

Result Grid | Filter Rows: Export: Wrap Cell Content:

email	friend_email
wu@gmail.com	ben@gmail.com
wu@gmail.com	john@gmail.com
carmelo.anthony@gmail.com	george@gmail.com
leonard@gmail.com	george@gmail.com
leonard@gmail.com	leonard@gmail.com
leonard@gmail.com	carmelo.anthony...
leonard@gmail.com	davis@yahoo.com
carmelo.anthony@gmail.com	james@gmail.com
carmelo.anthony@gmail.com	williams@gmail.com
carmelo.anthony@gmail.com	kamara@gmail.com
kyrie11irving@gmail.com	davis@yahoo.com
kyrie11irving@gmail.com	leonard@gmail.com

Leonard@gmail.com is added to [kyrie11irving@gmail.com](#) in MySql.

My Movie List Home Movie TV Show @Search Email

Avatar

A paraplegic Marine dispatched to the moon Pandora on a unique mission becomes torn between following his orders and protecting the world he feels is his home.

More Info ⓘ

Avengers: Endgame

After the devastating events of Avengers: Infinity War (2018), the universe is in ruins. With the help of remaining allies, the Avengers assemble once more in order to reverse Thanos' actions and restore balance to the universe.

More Info ⓘ

Black Widow

A film about Natasha Romanoff in her quests between the films Civil War and Infinity War.

More Info ⓘ

Breaking Bad

A high school chemistry teacher diagnosed with inoperable lung cancer turns to manufacturing and selling methamphetamine in order to secure his family's future.

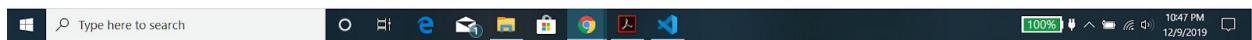
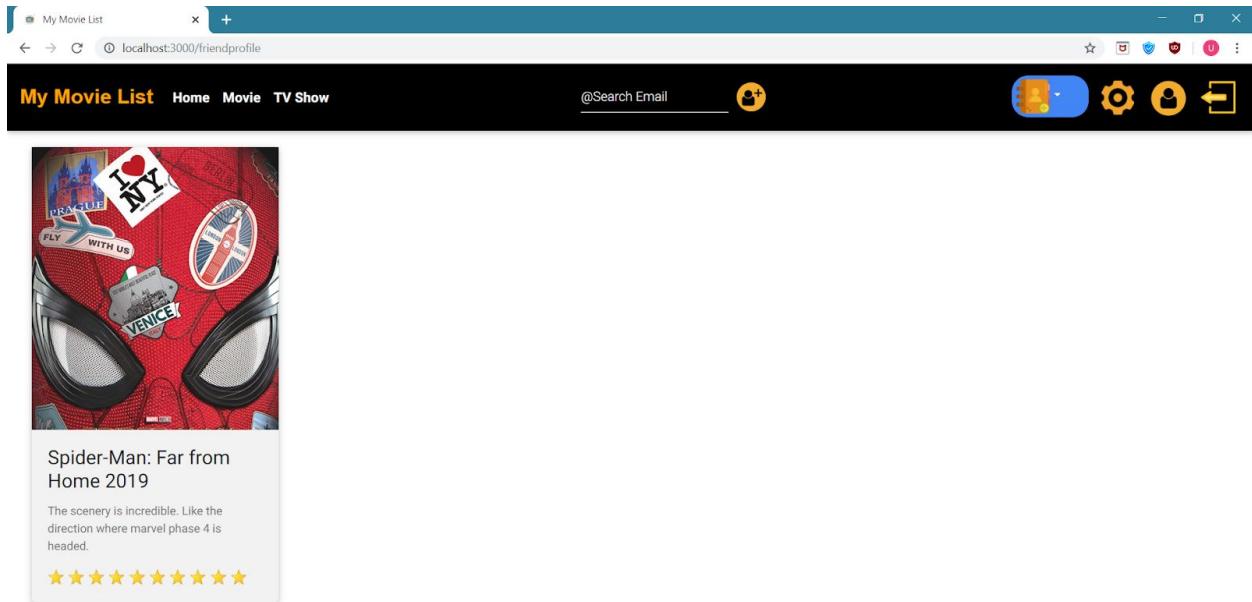
More Info ⓘ

localhost:3000/friendprofile

localhost:3000/home

100% 10:47 PM 12/9/2019

When you click on [leonard@gmail.com](#) from the dropdown menu, you can see the reviews given by leonard@gmail.com



[Leonard@gmail.com](#) has only rated Spider-Man: Far from Home which shows when a user clicks on [leonard@gmail.com](#) from the dropdown menu.

Result Grid						Filter Rows:	Edit:	Export/Import:	Wrap Cell Content:
email	contentname	releaseyear	review		rating				
ben@gmail.com	Cosmos	2014	Tyson makes it the best. But OG Cosmos by Carl Sagan is legendary		10				
ben@gmail.com	It	2017	Great jump scare. Stephen King is a king of horror movies.		7				
ben@gmail.com	John Wick	2014	Keanu Reeves is a legend, period.		5				
ben@gmail.com	John Wick: Chapter 3 – Parabellum	2019	A legend never dies.		9				
ben@gmail.com	Ready Player One	2018	Liked the concept of VR games coming to real life.		8				
bob@gmail.com	Naruto	2002	one of the best anime of all time		10				
carmelo.anthony@gmail.com	The Big Bang Theory	2007	Hilarious show. Gets funnier every time I watch it. New seasons are kinda slow tho.		7				
carmelo.anthony@gmail.com	The Boys	2019	It was a good show. Finished in one day.		10				
george@gmail.com	Erasied	2016	This anime is a master piece. Waiting for the next season.		9				
george@gmail.com	Inception	2010	I watched it many times. This movie is so well written. The story line is incredible.		8				
george@gmail.com	Mr. Robot	2015	Inspires me every day to become a hacker. Solid 10/10 show		10				
john@gmail.com	Avengers: Endgame	2019	Very good movie.		8				
john@gmail.com	Stranger Things	2016	Great thriller. I like demogorgons.		10				
leonard@gmail.com	Spider-Man: Far from Home	2019	The scenery is incredible. Like the direction where marvel phase 4 is headed.		10				
poiy@gmail.com	Avengers: Endgame	2019	Deserves 10 but my favorite avenger dies so I am giving it a 9. Overall, a great movie.		9				
*	NULL	NULL	NULL		NULL				

As can be seen in the screenshot of MySql, leonard@gmail.com has a review for “Spider-Man: Far from Home” and it is shown to [kyrie11irving@gmail.com](#) from friend’s list.

Screenshot of MySQL statement:

Actor

The screenshot shows the MySQL Workbench interface with the following details:

- Navigator:** Shows the schema `cs157a` with tables: actor, content, content_actor, content_director, director, movie, tvseries, user, user_content, user_friend.
- Query Editor:** The current query is `SELECT * FROM cs157a.actor;`. The results grid displays the following data:

actorname	actorage
Aaron Paul	40
Adam Driver	36
Adrian Rawlins	61
Alfie Allen	33
Andy Serkis	55
Angela Kinsey	48
Angela Sarafyan	36
Angus Wright	25
Anne G尦tta	51
Anne Hathaway	27
Anthony Hopkins	81
Antony Starr	44
Ayaka Nanase	25
Ayone Sakura	25
Barbie Ferreira	22
Ben Mendelsohn	50
Bonnie Wright	42

- Output:** The log shows the following actions:

Time	Action	Message	Duration / Fetch
19 21:22:25	SELECT * FROM cs157a.actor LIMIT 0, 1000	185 row(s) returned	0.000 sec / 0.000 sec
20 21:22:48	INSERT INTO content_actor VALUES("Good Boys", 2019, "Jacob Tremblay"), ("Good Boys", 2019, "Keith W...")	5 row(s) affected Records: 5 Duplicates: 0 Warnings: 0	0.000 sec
21 21:26:22	SELECT * FROM cs157a.actor LIMIT 0, 1000	185 row(s) returned	0.000 sec / 0.000 sec

content

The screenshot shows the MySQL Workbench interface with the following details:

- Navigator:** Shows the schema `cs157a` with tables: actor, content, content_actor, content_director, director, movie, tvseries, user, user_content, user_friend.
- Query Editor:** The current query is `SELECT * FROM cs157a.content;`. The results grid displays the following data:

contentname	releaseyear	contentgenre	studioname	poster	description
Avatar	2009	Adventure	Twentieth Century Fox	http://www.gstatic.com/v/thumb/22vodart/3...	A paraplegic Marine dispatched to the moon Pandora on a unique mission becomes torn ...
Avengers: Endgame	2019	Action	Marvel	http://www.gstatic.com/v/thumb/22vodart/3...	After the devastating events of Avengers: Infinity War (2018), the universe is in ruins...
Black Widow	2020	Action	Marvel Studio	https://i.pnning.com/73ka/3/a/fe/3c4efb...	A film about Natasha Romanoff in her quests between the films Civil War and Infinity War.
Breaking Bad	2008	Crime	HBO	http://www.gstatic.com/images/M/MS9B...	A high school chemistry teacher diagnosed with inoperable lung cancer turns to manufacture...
Chernobyl	2019	History	HBO	https://m.media-amazon.com/images/M/MS9B...	In April 1986, an explosion at the Chernobyl nuclear power plant in the Union of Soviet ...
Cosmos	2014	Documentary	National Geographic Channel	https://m.media-amazon.com/images/M/MS9B...	An exploration of our discovery of the laws of nature and coordinates in space and time...
Erasred	2016	Animation	A-1 Pictures	https://m.media-amazon.com/images/M/MS9B...	29 year old Satoru Fujinuma is sent back in time 18 years to prevent the events leading ...
Euphoria	2019	Drama	HBO	http://www.gstatic.com/v/thumb/vbanners/1...	A look at life for a group of high school students as they grapple with issues of drugs, sex...
Friends	1994	Comedy	Crane Production	https://m.media-amazon.com/images/M/MS9B...	Follow the personal and professional lives of six twenty to thirty-something-year-old fri...
Game of Thrones	2011	Action	HBO	https://imagesvc.msn.com/v3/fanimage...	Nine noble families fight for control over the mythical lands of Westeros, while an ancien...
Good Boys	2019	Comedy	Point Grey Pictures	https://m.media-amazon.com/images/M/MS9B...	Three 6th grade boys ditch school and embark on an epic journey while carrying acciden...
Inception	2010	Adventure	Warner Bros.	http://www.gstatic.com/v/thumb/22vodart/7...	A thief who steals corporate secrets through the use of dream-sharing technology is give...
Interstellar	2014	Fantasy	Paramount Pictures	https://m.media-amazon.com/images/M/MS9B...	A team of explorers travel through a wormhole in space in an attempt to ensure human...
It	2017	Horror	New Line Cinema	https://m.media-amazon.com/images/M/MS9B...	In the summer of 1989, a group of bullied kids band together to destroy a shape-shiftn...
John Wick	2014	Crime	Summit Entertainment	https://m.media-amazon.com/images/M/MS9B...	An ex-Hitman comes out of retirement to track down the gangsters that killed his dog a...
John Wick: Chapter...	2019	Crime	Summit Entertainment	https://m.media-amazon.com/images/M/MS9B...	John Wick is on the run after killing a member of the international assassin's guild, and w...
Mr. Robot	2015	Thriller	Universal Cable Productions	http://www.gstatic.com/v/thumb/vbanners/1...	Elliot, a brilliant but highly unstable young cyber-security engineer and vigilante hacker, ...
Westworld	2016	Sci-Fi	Kilter Factor	https://m.media-amazon.com/images/M/MS9B...	A science-fiction series based on the 1973 film of the same name, it follows a group of hosts

- Output:** The log shows the following actions:

Time	Action	Message	Duration / Fetch
21 21:26:22	SELECT * FROM cs157a.actor LIMIT 0, 1000	185 row(s) returned	0.000 sec / 0.000 sec
22 21:27:30	SELECT * FROM cs157a.content LIMIT 0, 1000	196 row(s) returned	0.000 sec / 0.000 sec
23 21:27:35	SELECT * FROM cs157a.content LIMIT 0, 1000	30 row(s) returned	0.000 sec / 0.000 sec

content_actor

The screenshot shows the MySQL Workbench interface with the query results for the content_actor table. The table has three columns: contentname, releaseyear, and actortname. The results list various movies and their actors from 2017 to 2019.

contentname	releaseyear	actortname
War for the Planet of the Apes	2017	Andy Serkis
War for the Planet of the Apes	2017	Woody Harrelson
War for the Planet of the Apes	2017	Karin Konoval
Avengers: Endgame	2019	Robert Downey Jr.
Avengers: Endgame	2019	Mark Ruffalo
Avengers: Endgame	2019	Chris Evans
Avengers: Endgame	2019	Chris Hemsworth
Avengers: Endgame	2019	Scarlett Johansson
Avengers: Endgame	2019	Jeremy Renner
Avengers: Endgame	2019	Chadwick Boseman
Avengers: Endgame	2019	Brie Larson
Avengers: Endgame	2019	Zoe Saldana
Avengers: Endgame	2019	Josh Brolin
Naruto	2002	Junko Takeuchi
Naruto	2002	Chie Nakamura
Naruto	2002	Noriko Sugiyama
Stranger Things	2016	Finn Wolfhard
Stranger Things	2016	Millie Bobby Brown

content_director

The screenshot shows the MySQL Workbench interface with the query results for the content_director table. The table has three columns: contentname, releaseyear, and directortname. The results list various movies and their directors from 2017 to 2019.

contentname	releaseyear	directortname
It	2017	Andy Muschetti
John Wick	2014	Chad Stahelski
John Wick: Chapter 3 – Parabellum	2019	Chad Stahelski
Avengers: Endgame	2019	Anthony Russo
Star Wars: The Rise Of Skywalker	2019	Jeffrey Jacob Abrams
War for the Planet of the Apes	2017	Matt Reeves
Naruto	2002	Masashi Kishimoto
Stranger Things	2016	Matt Duffer
Game of Thrones	2011	David Benioff
Spider-Man: Into the Spider-Verse	2018	Bob Persichetti
Ready Player One	2018	Steven Spielberg
The Fast and the Furious	2001	Rob Cohen
Black Widow	2020	Cate Shortland
Interstellar	2014	Christopher Nolan
Euphoria	2019	Sam Levinson
Breaking Bad	2008	Vince Gilligan
Chernobyl	2019	Craig Mazin
Call Me By Your Name	2017	Call Me By Your Name

Director

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator Schemas

Tables

Table: actor

Columns:

- actorname varchar(500) PK
- actorage int(11)

Result Grid

actorname	actorage
Andy Muschetti	46
Anthony Russo	49
Bob Peracetti	46
Cate Shortland	51
Chad Stahelski	51
Christopher Nolan	49
Chuck Lorre	67
Craig Mazin	48
David Benioff	49
David Crane	62
Eric Kripke	45
Gene Stupinski	42
Greg Daniels	56
Jane Cameron	55
Jeffrey Jacob Ab...	53
Jon Watts	38
Kei Sanbe	69
Kubrat Khlevnitski	??

director 1

Action Output

#	Time	Action	Message	Duration / Fetch
24	21:28:33	SELECT * FROM cs157a.content_actor LIMIT 0, 1000	196 row(s) returned	0.016 sec / 0.000 sec
25	21:29:25	SELECT * FROM cs157a.content_director LIMIT 0, 1000	30 row(s) returned	0.000 sec / 0.000 sec
26	21:30:00	SELECT * FROM cs157a.director LIMIT 0, 1000	28 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

Query Completed

movie

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator Schemas

Tables

Table: actor

Columns:

- actorname varchar(500) PK
- actorage int(11)

Result Grid

actorname	actorage
Andy Muschetti	46
Anthony Russo	49
Bob Peracetti	46
Cate Shortland	51
Chad Stahelski	51
Christopher Nolan	49
Chuck Lorre	67
Craig Mazin	48
David Benioff	49
David Crane	62
Eric Kripke	45
Gene Stupinski	42
Greg Daniels	56
Jane Cameron	55
Jeffrey Jacob Ab...	53
Jon Watts	38
Kei Sanbe	69
Kubrat Khlevnitski	??

movie 1

Result Grid

movie_contentname	movie_releaseyear	movie_length	movie_genre	movie_studioname
Avatar	2009	162	Adventure	Twentieth Century Fox
Avengers: Endgame	2019	181	Action	Marvel
Black Widow	2020	150	Action	Marvel Studio
Good Boys	2019	90	Comedy	Point Grey Pictures
Inception	2010	148	Adventure	Warner Bros.
Interstellar	2014	169	Fantasy	Paramount Pictures
It	2017	135	Horror	New Line Cinema
John Wick	2014	101	Crime	Summit Entertainment
John Wick: Chapter 3 – Parabellum	2019	131	Crime	Summit Entertainment
Ready Player One	2018	140	Action	Warner Bros.
Spider-Man: Far From Home	2019	129	Action	Marvel Studio
Spider-Man: Into the Spider-Verse	2018	117	Animation	Sony
Star Wars: The Rise Of Skywalker	2019	155	Adventure	Lucasfilm
The Fast and the Furious	2001	106	Action	Universal Pictures
The War for the Planet of the Apes	2017	140	Action	Twentieth Century Fox

Output

Action Output

#	Time	Action	Message	Duration / Fetch
26	21:30:00	SELECT * FROM cs157a.director LIMIT 0, 1000	28 row(s) returned	0.000 sec / 0.000 sec
27	21:30:29	SELECT * FROM cs157a.movie LIMIT 0, 1000	15 row(s) returned	0.000 sec / 0.000 sec
28	21:30:37	SELECT * FROM cs157a.movie LIMIT 0, 1000	15 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

Query Completed

Tvseries

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator Schemas

Tables

Table: actor

Columns:

- actorname varchar(500) PK
- actorage int(11)

Table: tseries

Columns:

- no_of_episodes int(11)
- tseries_studioname varchar(500)
- tseries_genre varchar(500)
- tseries_contentname varchar(500)
- tseries_releaseyear int(11)

Result Grid

no_of_episodes	tseries_studioname	tseries_genre	tseries_contentname	tseries_releaseyear
62	HBO	Crime	Breaking Bad	2008
5	HBO	History	Chernobyl	2019
26	National Geographic Channel	Documentary	Cosmo	2014
13	A-1 Pictures	Animation	Erasied	2016
9	HBO	Drama	Euphoria	2019
236	Crane Production	Comedy	Friends	1994
73	HBO	Action	Game of Thrones	2011
46	Universal Cable Productions	Thriller	Mr. Robot	2013
72	HBO	Bones	My Hero Academia	2016
220	Pilotnot	Animation	Naruto	2002
8	EUE Screen Gems Studios	Horror	Stranger Things	2016
281	Chuck Lorre Productions	Comedy	The Big Bang Theory	2007
10	Amazon Studios	Action	The Boys	2019
188	NBC	Comedy	The Office	2005
28	HBO	Mystery	Westworld	2016

Output

Action Output

#	Time	Action	Message	Duration / Fetch
27	21:30:29	SELECT * FROM cs157a.movie LIMIT 0, 1000	15 row(s) returned	0.000 sec / 0.000 sec
28	21:30:37	SELECT * FROM cs157a.movie LIMIT 0, 1000	15 row(s) returned	0.000 sec / 0.000 sec
29	21:31:13	SELECT * FROM cs157a.tseries LIMIT 0, 1000	15 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

Query Completed

User

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator Schemas

Tables

Table: user

Columns:

- email varchar(500)
- password varchar(500)
- username varchar(500)

Result Grid

email	password	username
alshon@gmail.com	po254d4jls	alshon01
antonio@gmail.com	an9tbo0	antonio84
ben@gmail.com	thissatempass	ben
bob@gmail.com	karnalop09	boban
carmelo.anthon@gmail.com	hoddiemelo	melo00
davis@yahoo.com	pelicanstrollers	davis13
fox5@yahoo.com	fastakal	fox05
george@gmail.com	pg13	george13
james@gmail.com	leadscorer13	harden13
john@gmail.com	12345	john
kamara@gmail.com	drer9salvin41	alvinkamara
kyle@gmail.com	lowriya	kyle07
lavine@gmail.com	timberwolves	zacklavine
leonard@gmail.com	klaw	kawahLe...
perry@gmail.com	147	pouytr
williams@gmail.com	sweetbou	23lourwill
wu@gmail.com	123456	mike wu

Output

Action Output

#	Time	Action	Message	Duration / Fetch
28	21:30:37	SELECT * FROM cs157a.movie LIMIT 0, 1000	15 row(s) returned	0.000 sec / 0.000 sec
29	21:31:13	SELECT * FROM cs157a.tseries LIMIT 0, 1000	15 row(s) returned	0.000 sec / 0.000 sec
30	21:31:48	SELECT * FROM cs157a.user LIMIT 0, 1000	7 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

Query Completed

User_content

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator Schemas

Insert in Tables* user_content user user_content

1 • SELECT * FROM cs157a.user_content;

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Contents: |

email	contentname	releasyear	review	rating
ben@gmail.com	Cosmos	2014	Tyson makes it the best. But OG Cosmos by Carl Sagan is legendary	10
ben@gmail.com	It	2017	Great jump score. Stephen King is a king of horror movies.	7
ben@gmail.com	John Wick	2014	Keanu Reeves is a legend, period.	5
ben@gmail.com	John Wick: Chapter 3 –Parabellum	2019	A legend never dies.	9
ben@gmail.com	Ready Player One	2018	Liked the concept of VR games coming to real life.	8
ben@gmail.com	Naruto	2002	one of the best anime of all time	10
carmelo.anthony@gmail.com	The Big Bang Theory	2007	Hilarious show. Gets funnier every time I watch it. New seasons are kinda slow tho.	7
carlito.anthony@gmail.com	The Boys	2019	It was a good show. Finished in one day.	10
george@gmail.com	Erasd	2016	The anime is a master piece. Waiting for the next season.	9
george@gmail.com	Inception	2010	I watched it many times. This movie is so well written. The story line is incredible.	8
george@gmail.com	Mr. Robot	2015	Inspires me every day to become a hacker. Solid 10/10 show	10
john@gmail.com	Avengers: Endgame	2019	Very good movie.	8
john@gmail.com	Stranger Things	2016	The scenery is incredible. Like the direction where marvel phase 4 is headed.	10
leoren@gmail.com	Spider-Man: Far from Home	2019	Deserves 10 but my favorite avenger dies so I am giving it a 9. Overall, a great movie.	9
poory@gmail.com	Avengers: Endgame	2019	MAX	MAX

Result Grid Form Editor Field Types Query Stats Execution Plan

Action Output

#	Time	Action	Message	Duration / Fetch
32	22:01:07	SELECT * FROM cs157a.user_content LIMIT 0, 1000	12 row(s) returned	0.000 sec / 0.000 sec
33	22:06:50	SELECT * FROM cs157a.user_content LIMIT 0, 1000	14 row(s) returned	0.000 sec / 0.000 sec
34	22:09:29	SELECT * FROM cs157a.user_content LIMIT 0, 1000	15 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

Query Completed

User_friend

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator Schemas

Insert in Tables* user_content user user_content user_friend

1 • SELECT * FROM cs157a.user_friend;

Result Grid | Filter Rows: | Export: | Wrap Cell Contents: |

email	friend_email
ben@gmail.com	poory@gmail.com
ben@gmail.com	ben@gmail.com
john@gmail.com	ben@gmail.com
john@gmail.com	bob@gmail.com
ben@gmail.com	bob@gmail.com
ben@gmail.com	john@gmail.com
wu@gmail.com	ben@gmail.com
wu@gmail.com	john@gmail.com
carmelo.anthony@gmail.com	george@gmail.com
leonard@gmail.com	george@gmail.com
leonard@gmail.com	leonard@gmail.com
leonard@gmail.com	carmelo.anthony...
leonard@gmail.com	dave@yahoo.com
carmelo.anthony@gmail.com	james@gmail.com
carmelo.anthony@gmail.com	williams@gmail.com
carmelo.anthony@gmail.com	kamara@gmail.com

Result Grid Form Editor Field Types Query Stats Execution Plan

Action Output

#	Time	Action	Message	Duration / Fetch
39	22:13:09	Apply changes to user_friend	Error 1091: Can't DROP 'user_friend_ibfk_1'; check that column/key exists SQL Statement: ALTER TABLE `cs...	0.000 sec / 0.000 sec
40	22:15:21	SELECT * FROM cs157a.user_friend LIMIT 0, 1000	13 row(s) returned	0.000 sec / 0.000 sec
41	22:16:48	SELECT * FROM cs157a.user_friend LIMIT 0, 1000	16 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

Query Completed

Normalization: BCNF

Actor

Actor(actorname, actorage)

Let A = actorname, B = actorage

Functional Dependency: A->B

A->B is a nontrivial functional dependency and A is a superkey; therefore the Actor table is in BCNF.

content

content(contentname, releaseyear, contentgenre, studioname, poster, description)

Let A = contentname, B = releaseyear, C = contentgenre, D = studioname, E = poster,

F = description

Functional Dependency: A->BCDEF

A->BCDEF is a nontrivial functional dependency and A is a superkey; therefore the content table is in BCNF.

content_actor

content_actor(contentname, releaseyear, actorname)

Let A = contentname, B = releaseyear, C = actorname

Functional Dependency: AC->B

AC->B is a nontrivial functional dependency and AC is a superkey; therefore the content_actor table is in BCNF.

content_director

content_director(contentname, releaseyear, directorname)

Let A = contentname, B = releaseyear, C = directorname

Functional Dependency: A->BC

A->BC is a nontrivial functional dependency and A is a superkey; therefore the Content_director table is in BCNF.

Director

Director(directorname, age)

Let A = directorname, B = age

Functional Dependency: A->B

A->B is a nontrivial functional dependency and A is a superkey; therefore the Director table is in BCNF.

movie

movie(movie_contentname, movie_releaseyear, movie_length, movie_genre, movie_studioname)

Let A = movie_contentname, B = movie_releaseyear, C = movie_length,

D = movie_genre, E = movie_studioname

Functional Dependency: A->BCDEF

A->BCDEF is a nontrivial functional dependency and A is a superkey; therefore the movie table is in BCNF.

Tvseries

Tvseries(no_of_episodes, tvseries_studioname, tvseries_genre, tvseries_contentname
tvseries_releaseyear)

Let A = no_of_episodes, B = tvseries_studioname, C = tvseries_genre,
D = tvseries_contentname, E = tvseries_releaseyear

Functional Dependency: D->ABCEF

D->ABCEF is a nontrivial functional dependency and D is a superkey; therefore the
Tvseries table is in BCNF.

User

User(email, password, username)

Let A = email, B = password, C = username

Functional Dependency: A -> BC, C ->AB

Both functional dependencies are nontrivial and A and C are superkeys; therefore the
User table is in BCNF.

User_content

User_content(email, contentname, releaseyear, review, rating)

Let A = email, B = contentname, C = releaseyear, D = review, E = rating

Functional Dependency = A->BCDE

A->BCDE is a nontrivial functional dependency and D is a superkey; therefore the
User_content table is in BCNF.

User_friend

User_friend(email, friend_email)

Let A = email, B = friend_email

Functional Dependencies = A ->B and B->A

A->B and B->A are nontrivial functional dependencies and both are superkeys;
therefore the User_friend table is in BCNF.

Conclusion:

Jiawei Zhang(Team leader):

Throughout this database project, it was my first time completing a full stack website application. It was quite a challenging project because I have zero experience in terms of integrating front end and backend with MySQL database. I was responsible for front end development and backend connection. The hardest part for me was to fetch the data from MySQL database, parse to JSON format and then display the result data with style in CSS. However as I finished the task I gained a better understanding of Three-Tier architecture and Restful API design. Besides the technical aspect, it was also my first time being a team leader. I was able to bring my team to deliver the project in time and improve my management skills as well.

Udaypal Singh:

The development process of this website, with MySql database and React, has taught me many valuable skills which I can carry on with me. Constructing an ER diagram is a base for the

project, which I believe did a really good job. Starting the project from scratch and connecting backend from Sql to frontend using node.js and express was not easily achieved as I had no previous experience with this. I worked on retrieving the data from the backend including list of content, movies, tvseries, and using union of the table content UNION movie, and content UNION tvseries and then using JOIN to querying actors and directors from these complex queries were a hassle. I was also responsible for inserting, editing, and deleting the review and rating for a user reviewed movie from the frontend to backend. I also faced difficulties when working with the UI; for example displaying the content cards in-line, and designing moreinfo, rating on top of the background image involved many UI design complexities. Despite the challenges, I believe I learned a lot, and now I feel confident that I can develop a website with a solid database structure.

Adan Hernandez:

Throughout the course of the project I gained very valuable skills like SQL programming and React. It was very challenging, and being the first time I worked on a three tier architecture project from scratch I had a hard time learning the skills needed to complete the project. The making of the ER diagram and the documentation of everything were other key skills that I learned that I know will benefit me in the long run. In other classes, there usually isn't documentation that goes into it, however, it is an important skill because it is expected to document everything in internships. I worked on some of the front-end, and after this project I know that I really enjoy front-end. I especially like being able to connect the front-end with back-end and see how it all comes together.

Future Improvement of the Application:

There are some improvements that can be made within the application. For example, the database of our application is based on local storage MySQL database. If the user who runs the application does not have the data set up in MySQL, then the application will have blank information when the user enters on the homepage of our application. In the future we could set up a cloud database and share the resource via cloud storage, hence every user who enter our application will have the same content showed on homepage. In the future we could also have movie database API setup in our application. This would allow our application to scale and manipulate a lot more data overtime.