PRODUCTION GUIDE DOCUMENT

Movies Project

Swati Khandare
(swati.khandare@epita.fr)

Table of Contents

Roles

User Role

1. Create User Login
2. After login, below options are displayed:

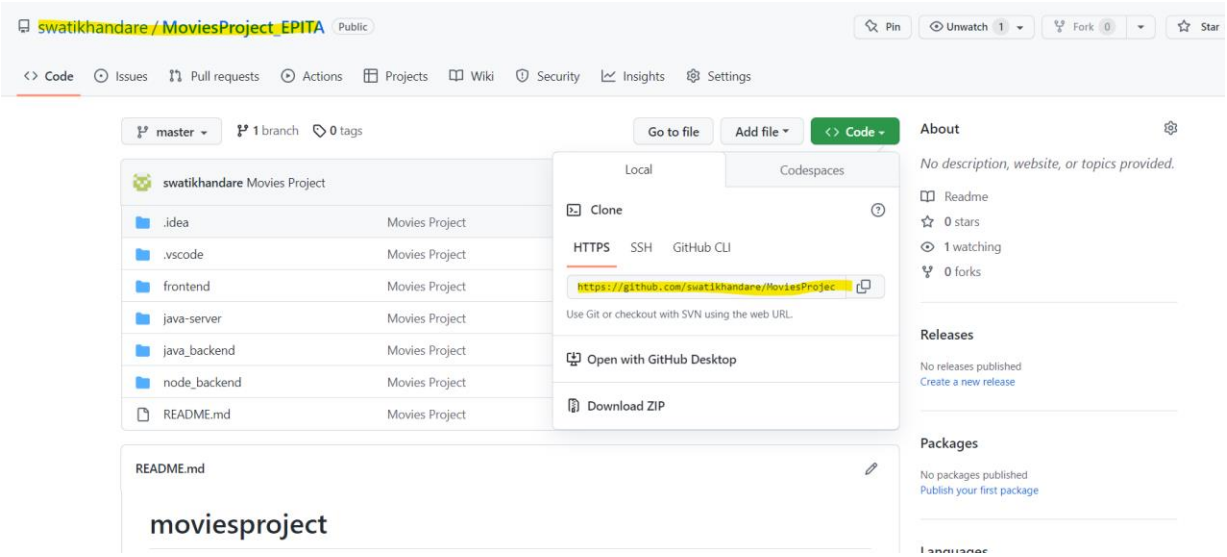
    - HomePage Trailer
    - Continue Watching
    - Latest Movies
    - Recommended Movies
    - Top 10 Most Popular Movies
    - Top 10 Most Rated Movies

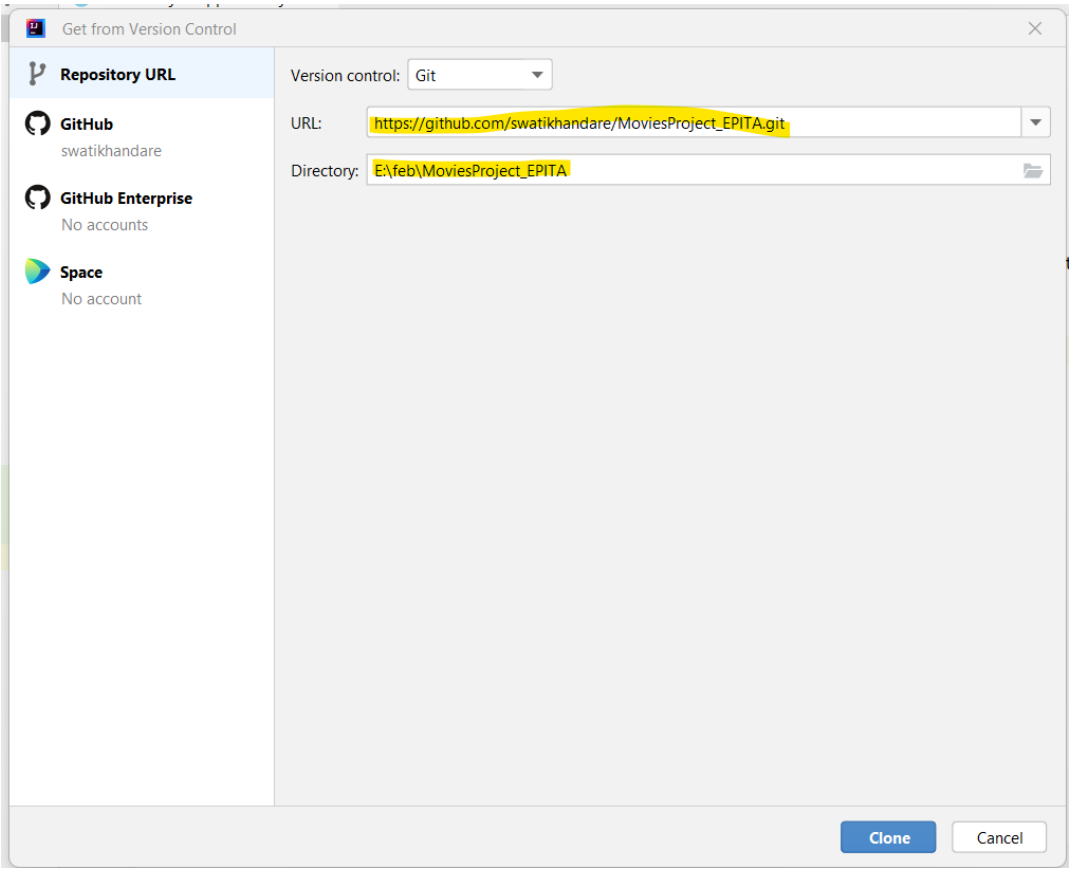
    Click on play button to watch the trailer!

Compilation and Installation Steps

1. Copy the URL link and clone the project in IntelliJ.

URL: https://github.com/swatikhandare/MoviesProject_EPITA.git



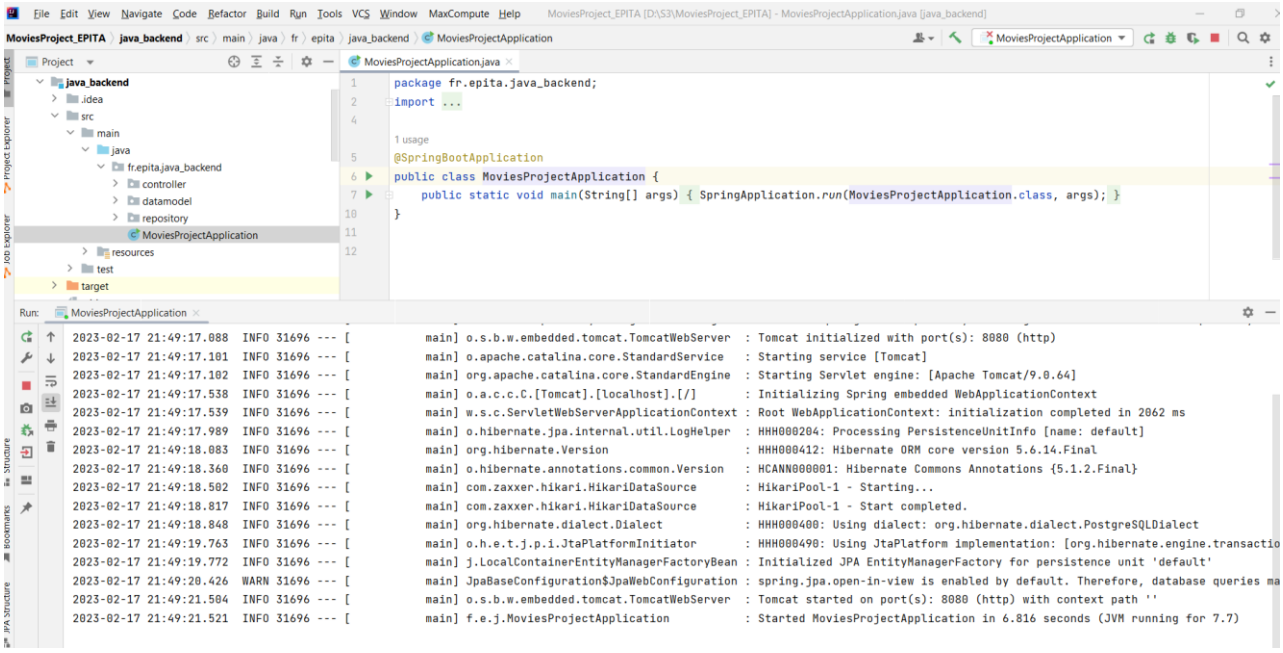


2. Start the PostgreSQL and create the database using the details from "java_backend/src/main/resources/application.properties" file.

3. Create MongoDB cluster and collection with user and password on MongoDB Atlas

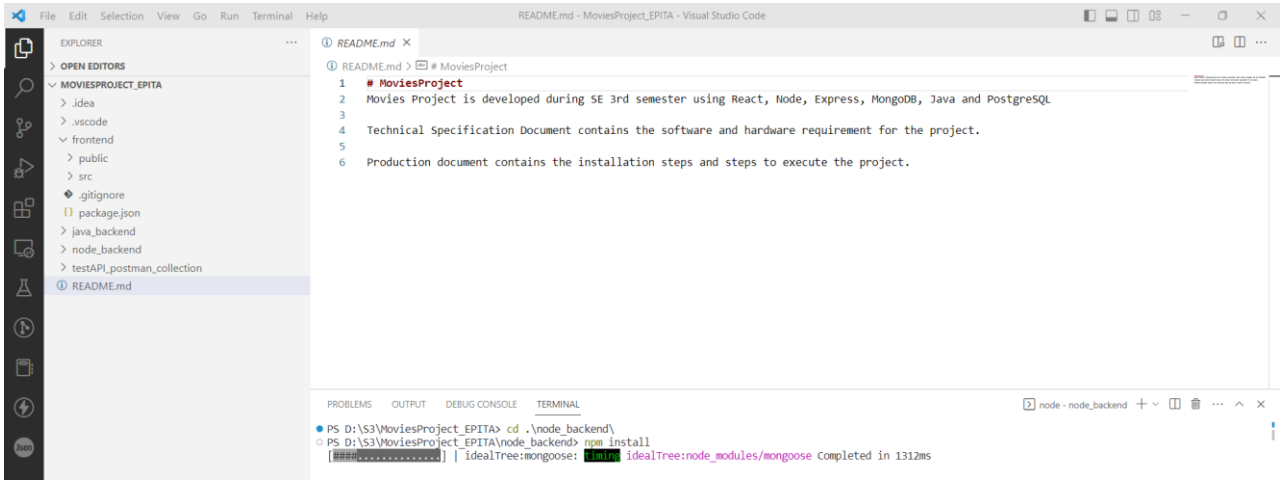
4. Create a file ".env" at path "node_backend" and add the MonogoDB details (for reference you check file "node_backend/.env.example")

5. Run the SpringBootApplication at path "java_backend/src/main/java/fr/epita/java_backend/MoviesProjectApplication.java" to start the application.



6. Open the project in VSCode.

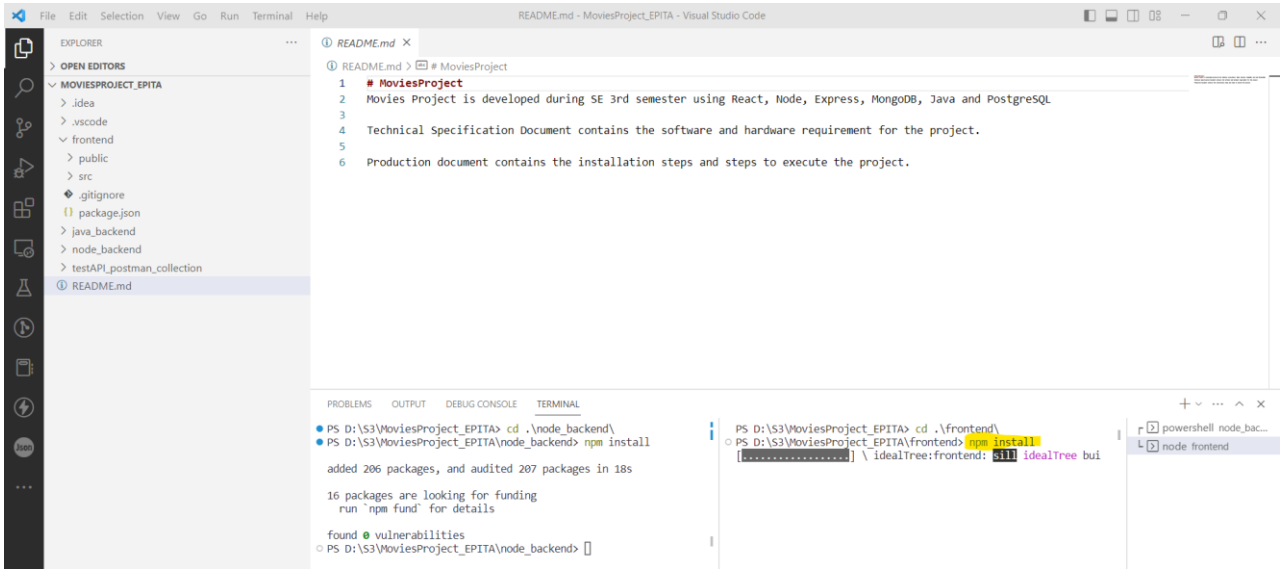
7. Navigate to folder “node_backend” and execute command “npm install”.



8. Execute command “npm run dev” to start the backend server.

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

added 206 packages, and audited 207 packages in 18s

16 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS D:\S3\MoviesProject_EPITA\node_backend> npm run dev

> server@1.0.0 dev
> nodemon index.js

[nodemon] 2.0.20
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
MongoDB Connected...
(node:2644) [MONGOOSE] DeprecationWarning: Mongoose: the `strictQuery` option will be switched back to `false` by default
 in Mongoose 7. Use `mongoose.set('strictQuery', false);` if you want to prepare for this change. Or use `mongoose.set('s
trictQuery', true);` to suppress this warning.
(Use `node --trace-deprecation ...` to show where the warning was created)
Server running on port 3001
```

9. Navigate to folder “frontend” and execute command “npm install”.

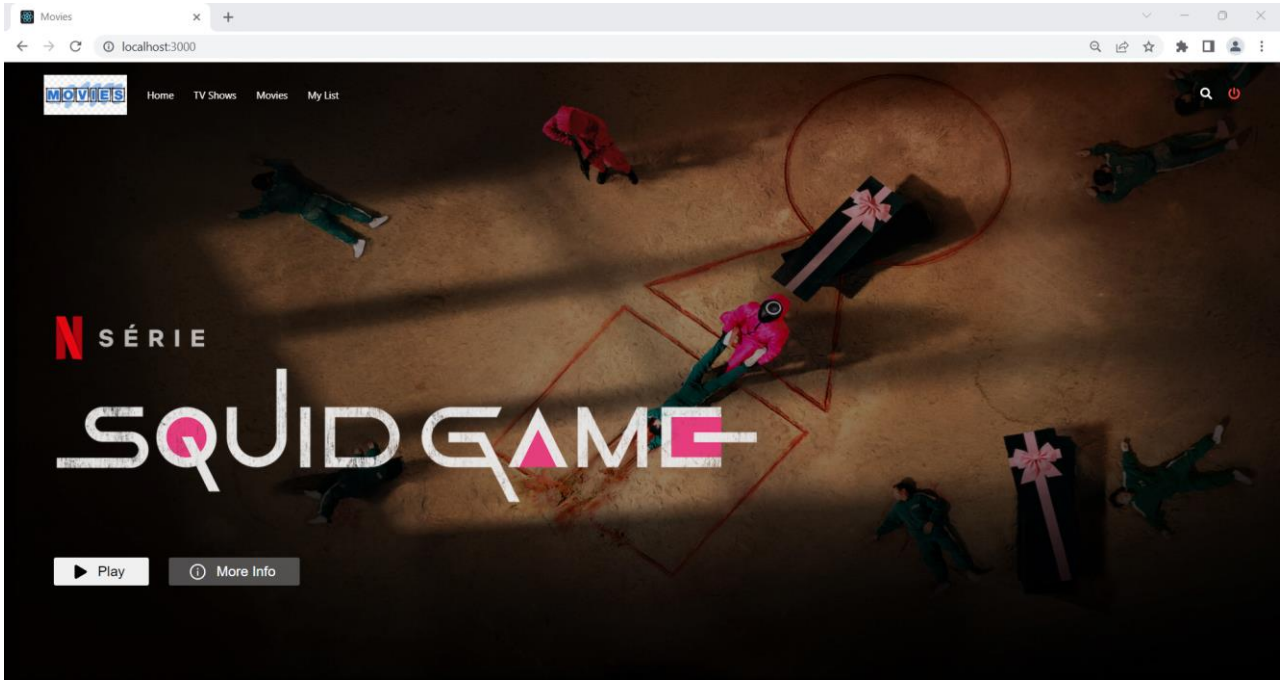


10. Execute command “npm run start” to start the front end.

```
TERMINAL

PS D:\S3\MoviesProject_EPITA\frontend> npm run start
```

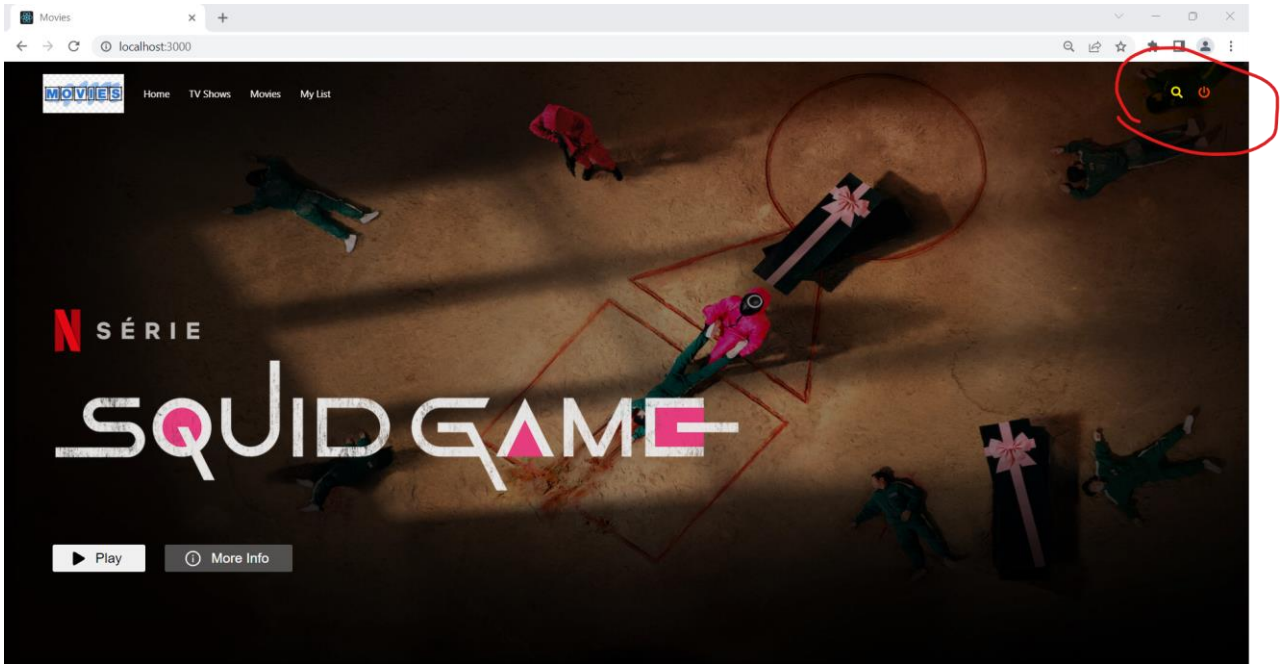
```
SOLE    TERMINAL
Compiled successfully!

You can now view client in the browser.

  Local:            http://localhost:3000
  On Your Network:  http://10.188.83.198:3000

Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
```

11. The application will be launched in your localhost at port 3000!

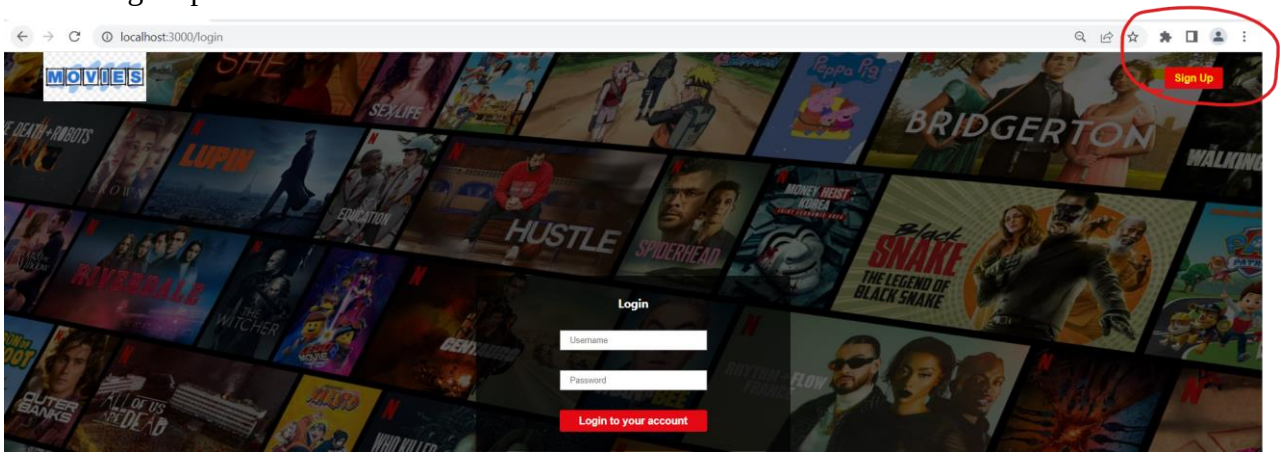


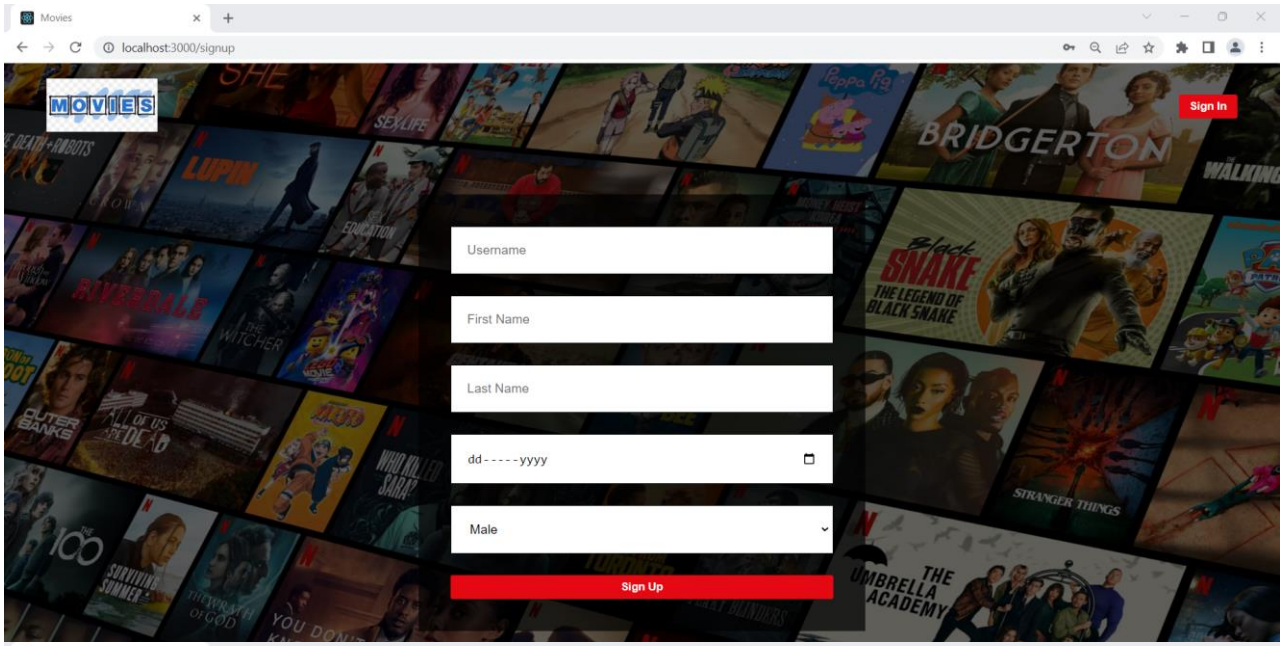
Execution

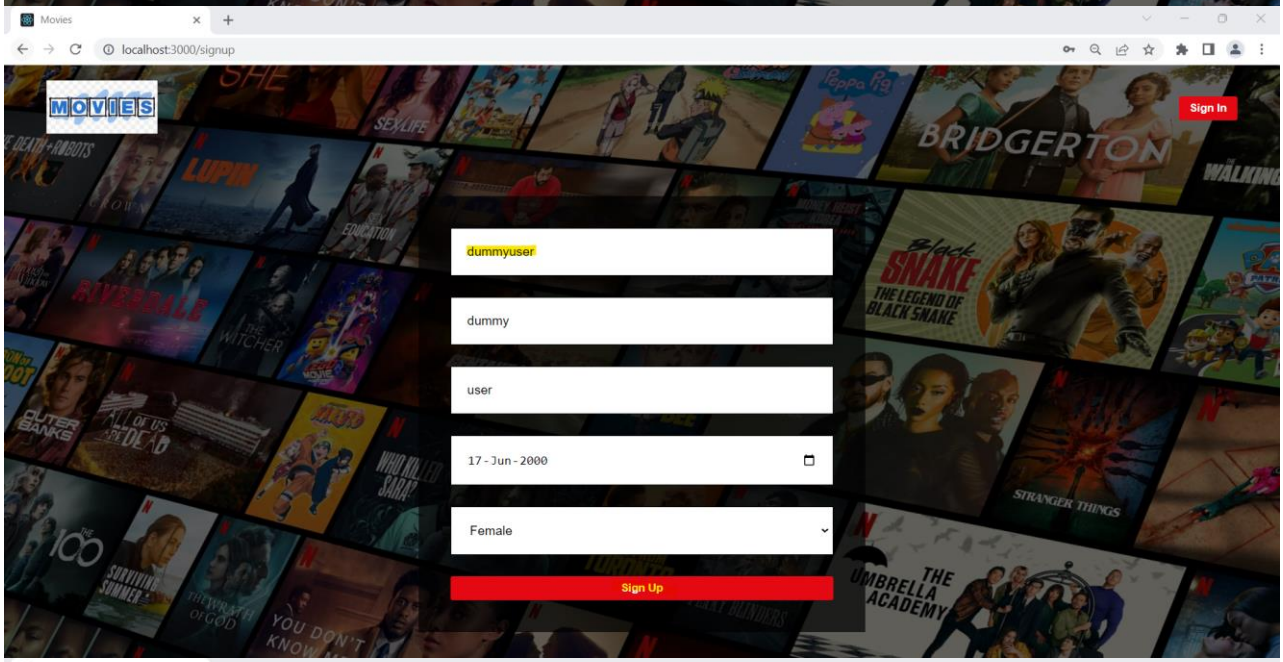
1. Landing page

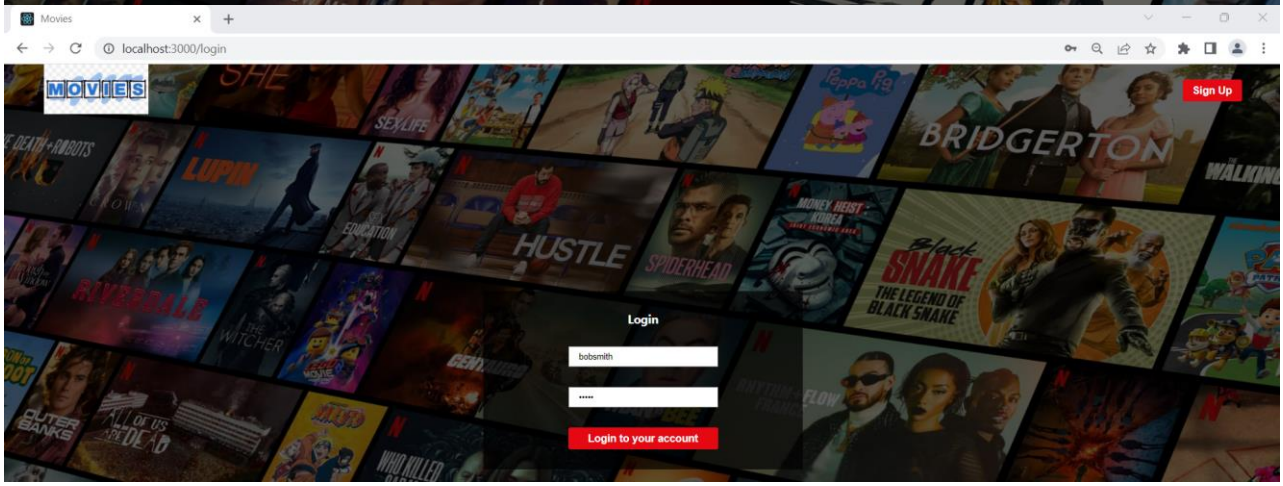

Movies
localhost:3000
MOVIES
Home
TV Shows
Movies
My List
SÉRIE
SQUID GAME
Play
More Info

2. Sign Up

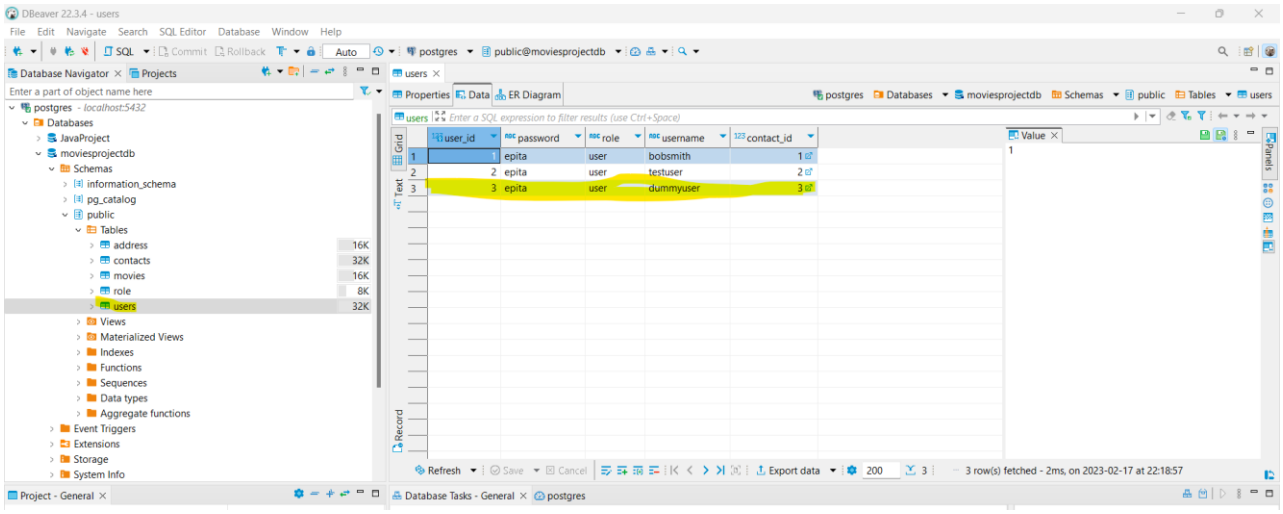

localhost:3000/login
MOVIES
Sign Up
Login
Username
Password
Login to your account


Movies
localhost:3000/signup
MOVIES
Sign In
Username
First Name
Last Name
dd-----yyyy
Male
Sign Up

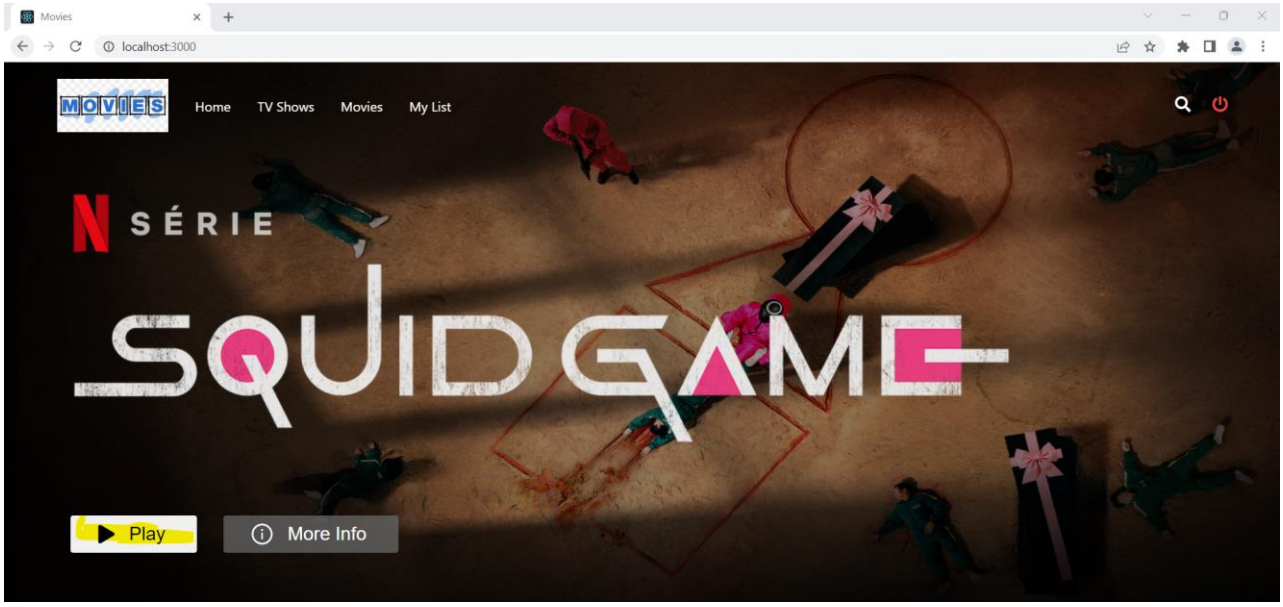

Movies
localhost:3000/signup
MOVIES
Sign In
dummyuser
dummy
user
17-Jun-2000
Female
Sign Up


Movies
localhost:3000/login
MOVIES
Sign Up
Login
bobsmith
Login to your account

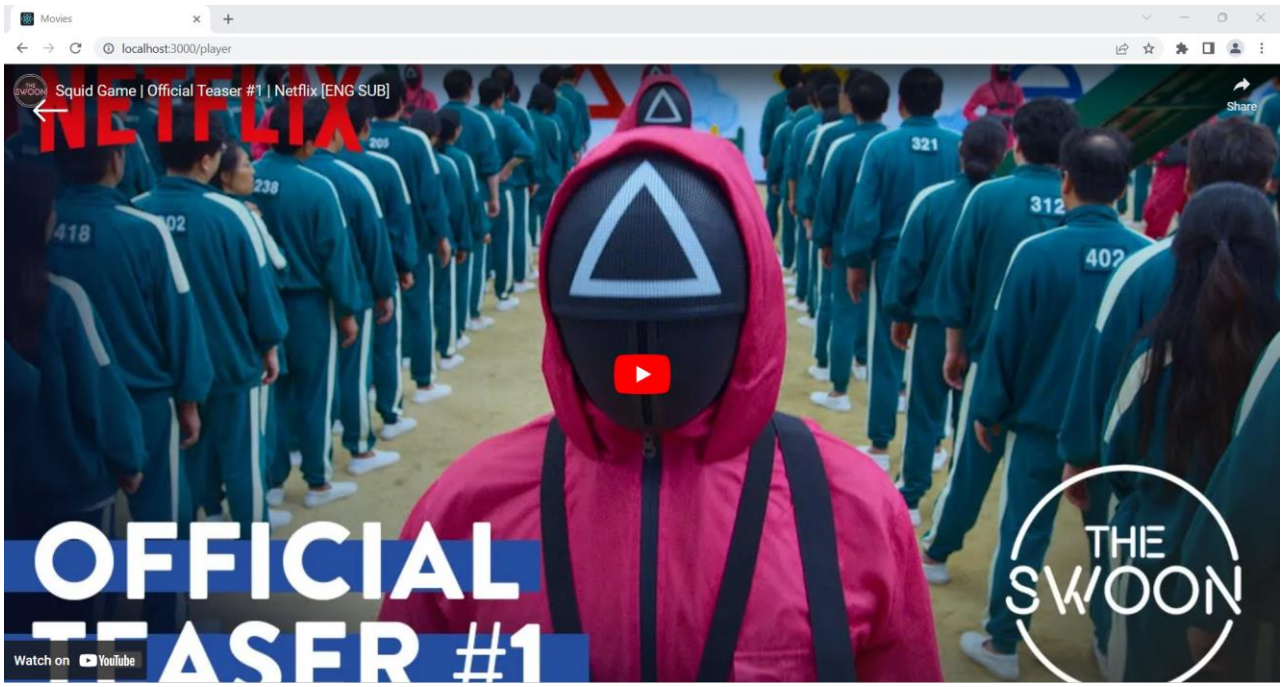
3. User added in users postgresql db



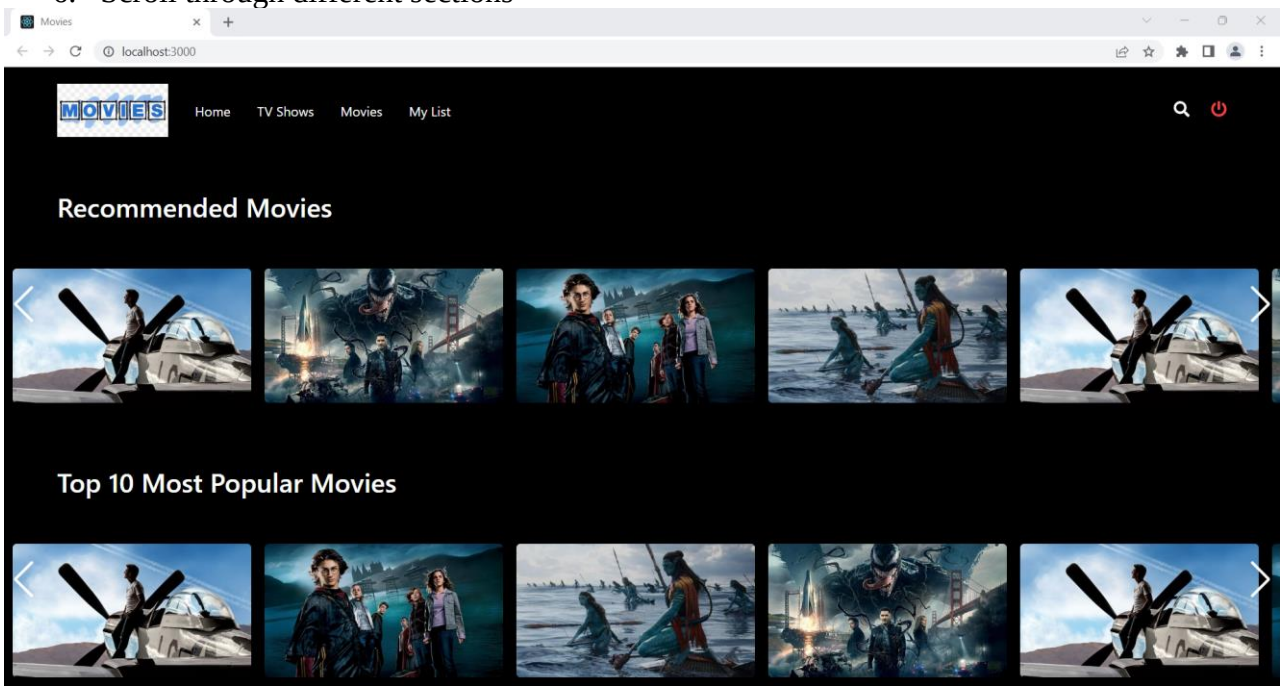
4. Sign in with the user created



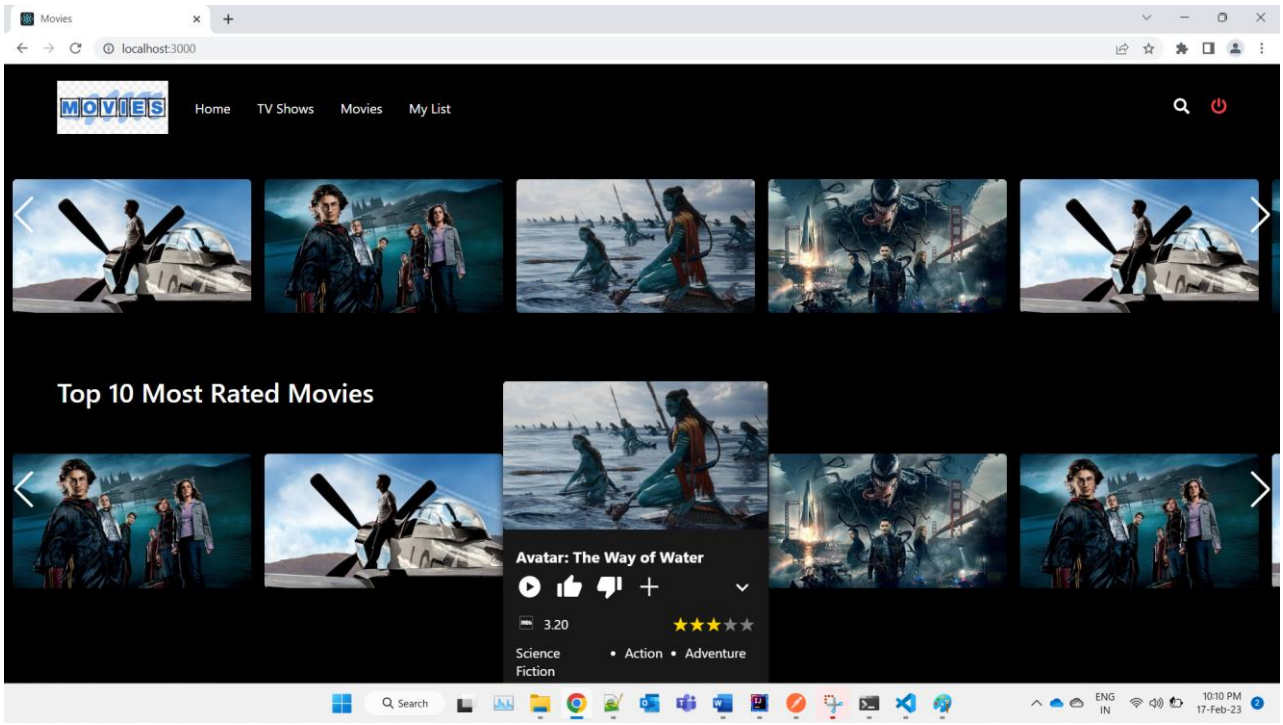
5. Click on play to play the Squid Game trailer.



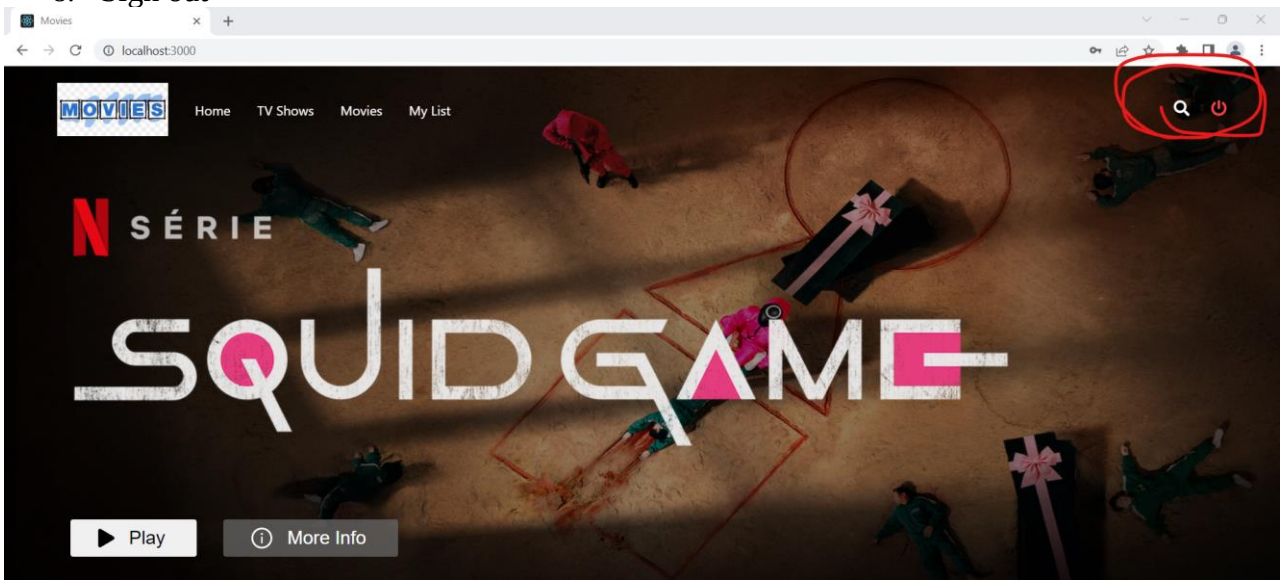
6. Scroll through different sections



7. Check for the ratings by hovering over a particular movie



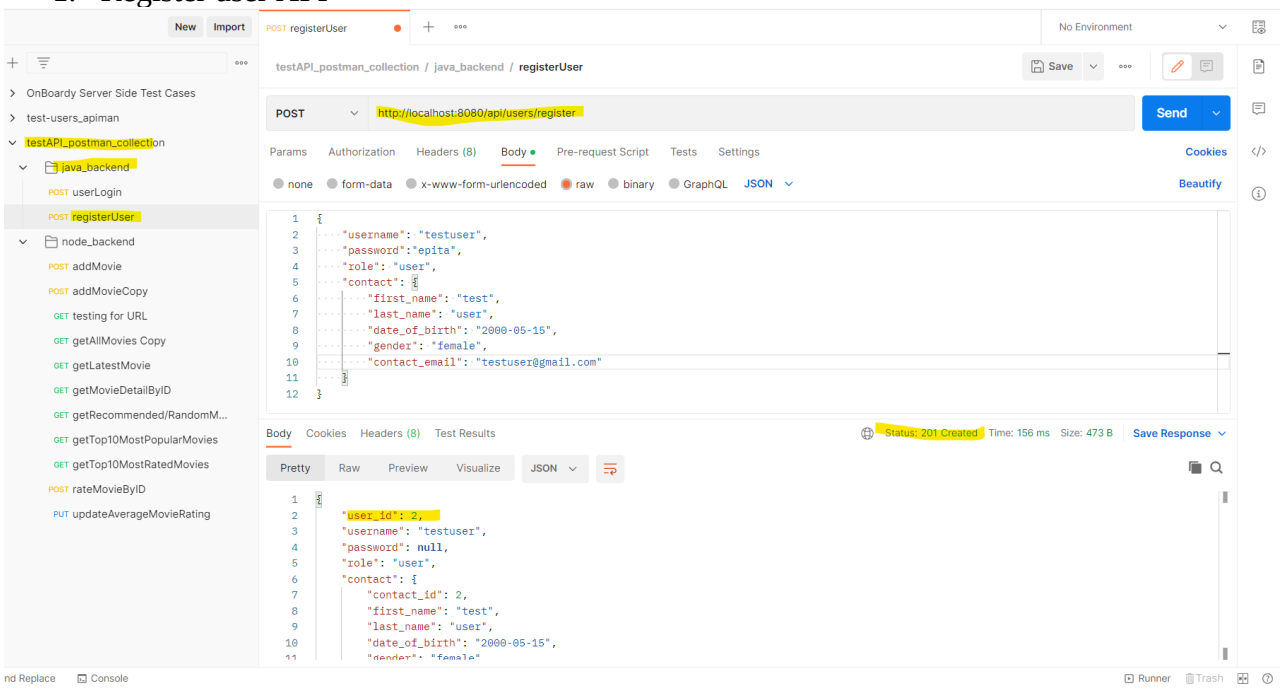
8. Sign out



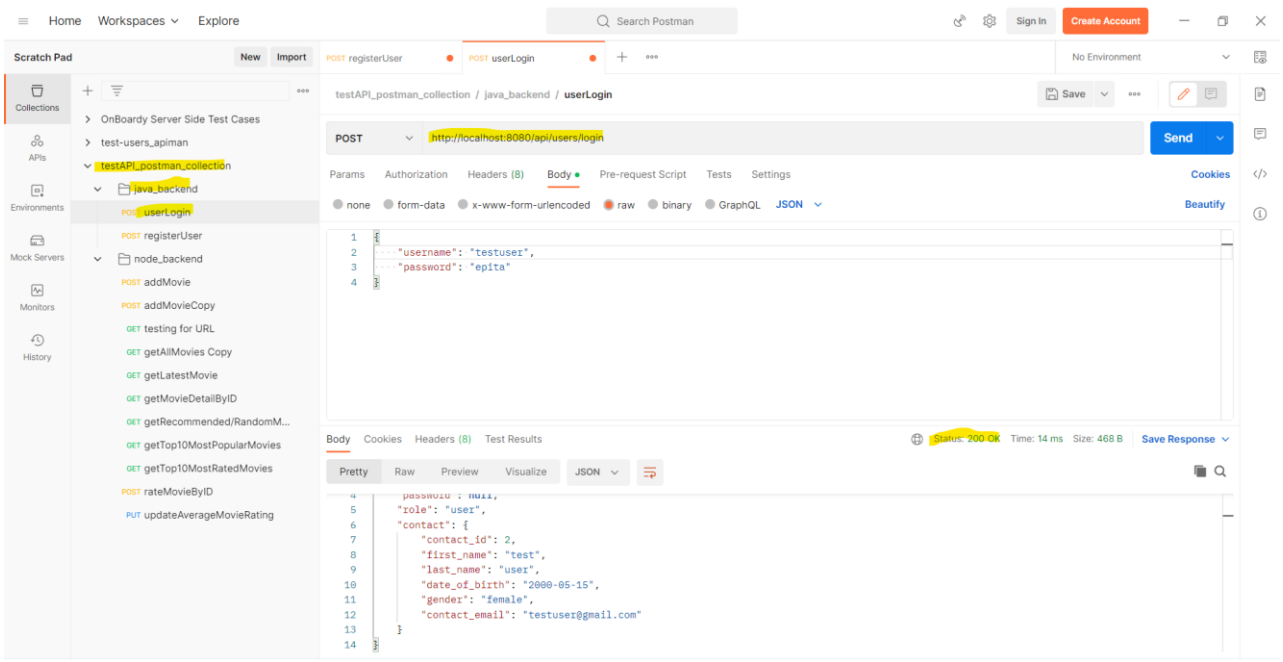
API testing

Import the postman collection from .json "**testAPI_postman_collection.postman_collection.json**" to your local postman to test the **java backend** and **node backend** for APIs "api/users", "/api/movies/" and "/api/movies/rating/" .

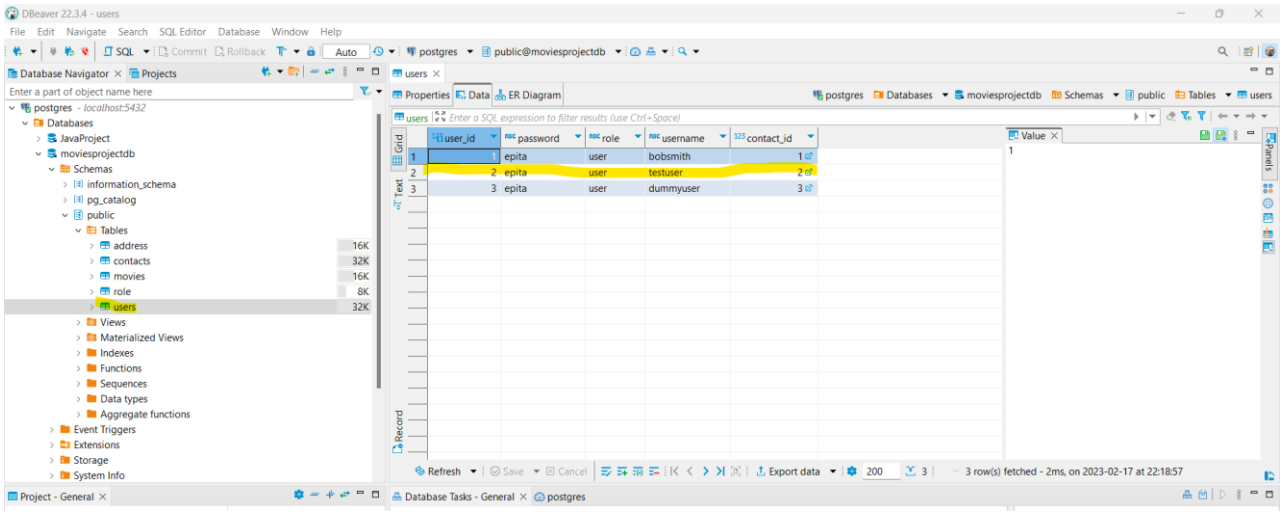
1. Register user API



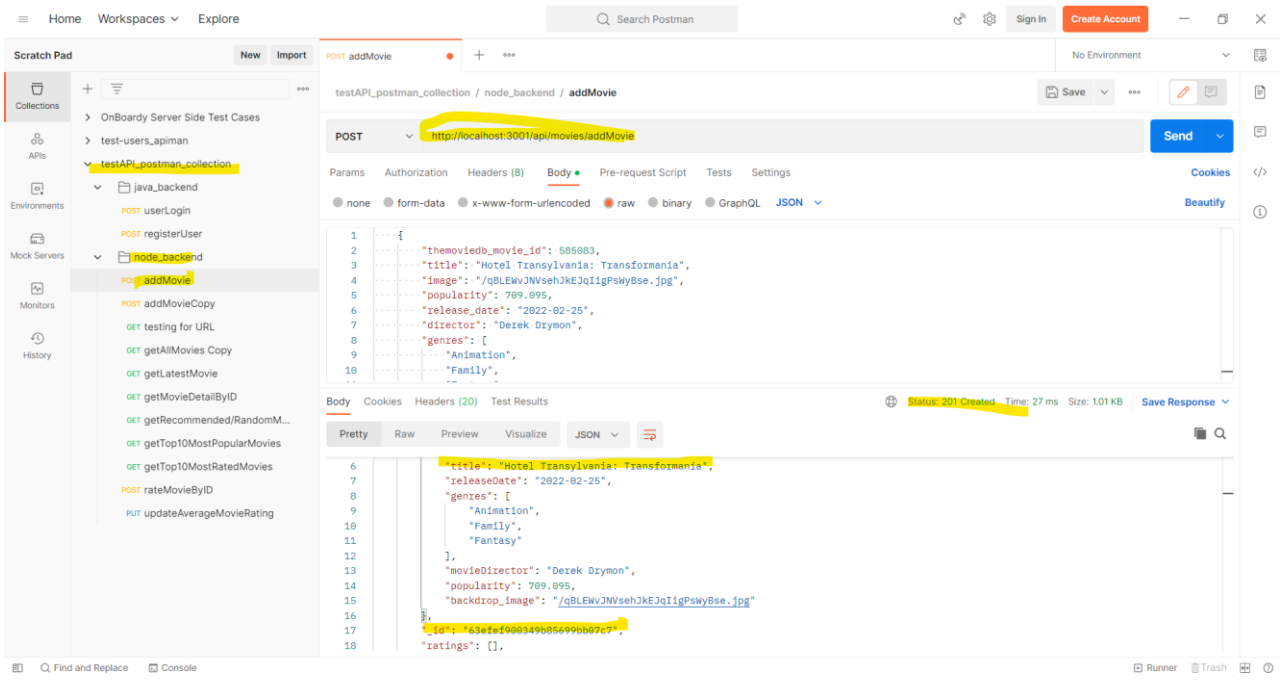
2. User login API



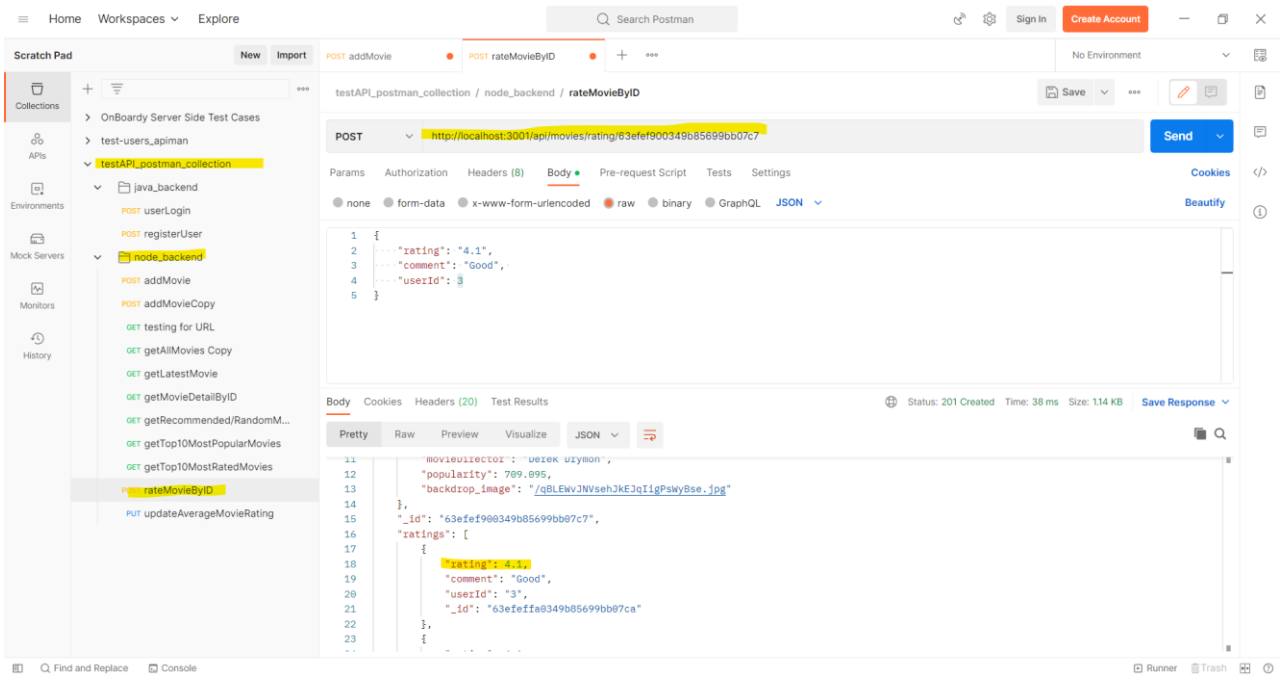
New user is added in the postgresql db table



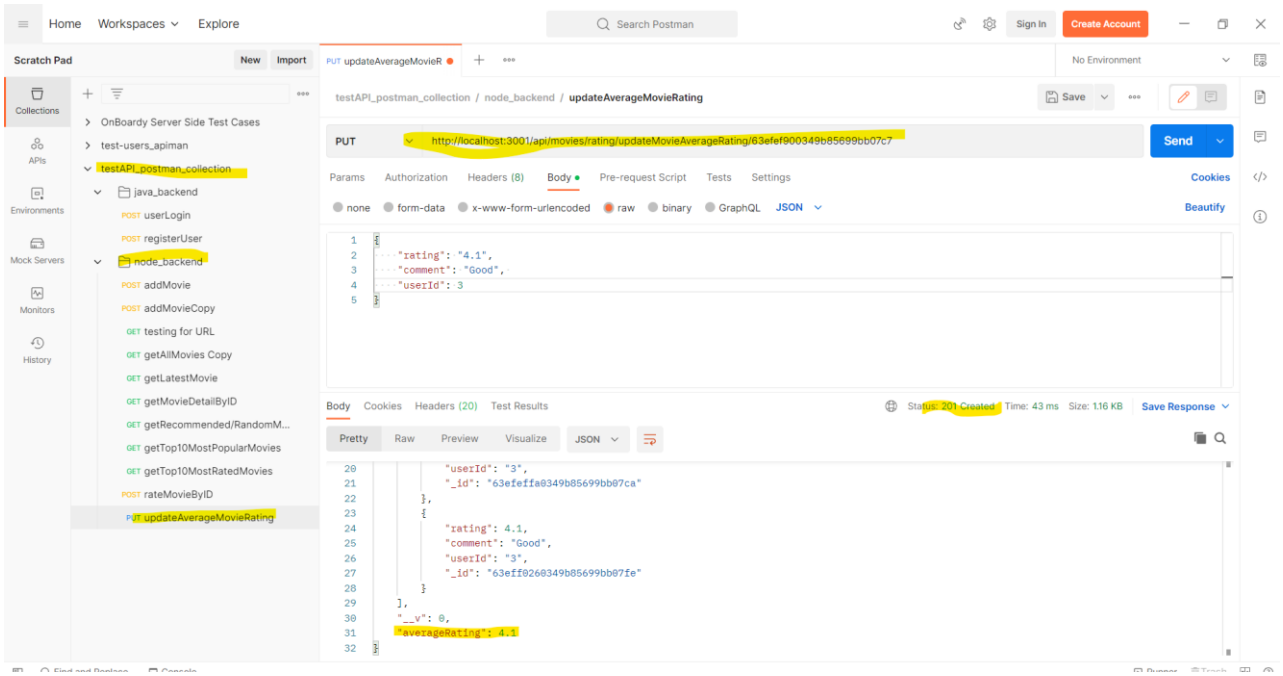
3. Add movie API



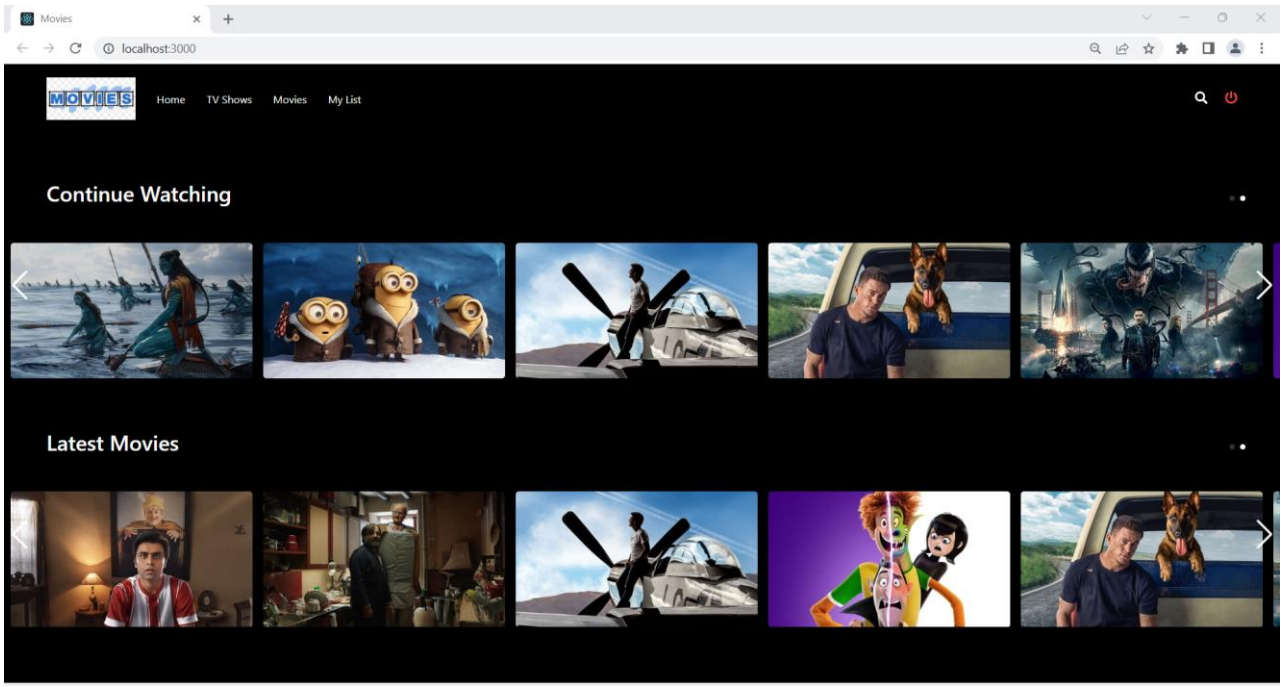
4. Add movie rating by movie id API



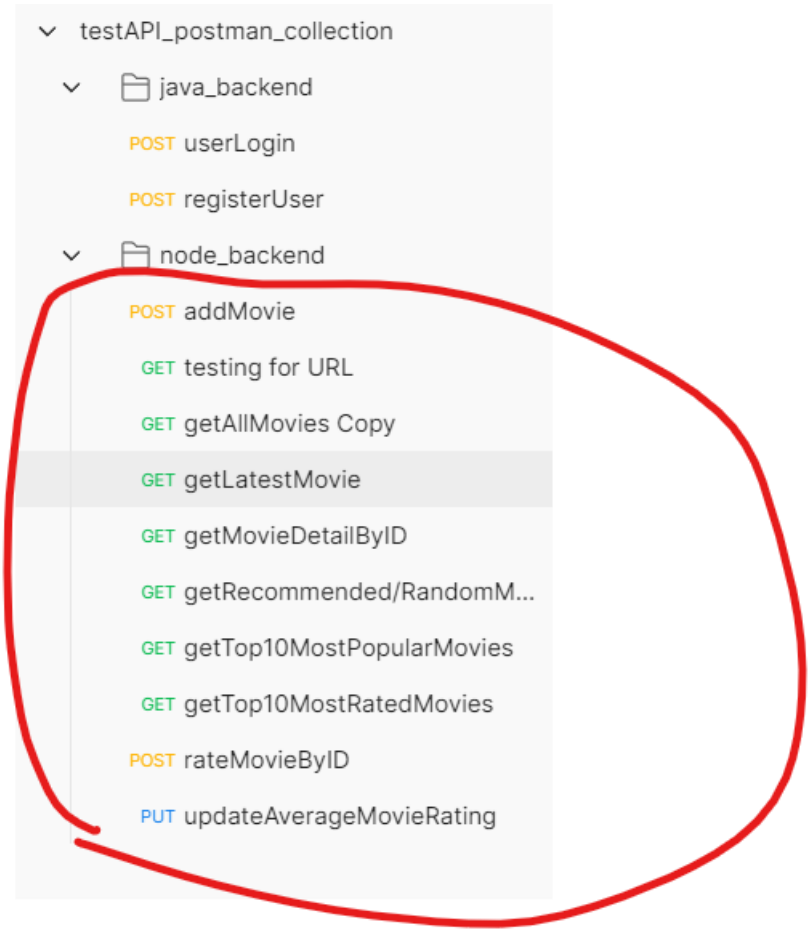
5. Add average rating by movie id API



Movie is added



6. Other different test APIs for get all movies, get latest movies, get movies details , get top 10 movies etc.


testAPI_postman_collection
java_backend
POST userLogin
POST registerUser
node_backend
POST addMovie
GET testing for URL
GET getAllMovies Copy
GET getLatestMovie
GET getMovieDetailByID
GET getRecommended/RandomM...
GET getTop10MostPopularMovies
GET getTop10MostRatedMovies
POST rateMovieByID
PUT updateAverageMovieRating

S