PRODUCTION GUIDE DOCUMENT

Movies Project

Swati Khandare
(swati.khandare@epita.fr)

Table of Contents

Roles 3
User Role 3
Compilation and Installation Steps 3
Execution 7
API testing 12

Roles

User Role

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

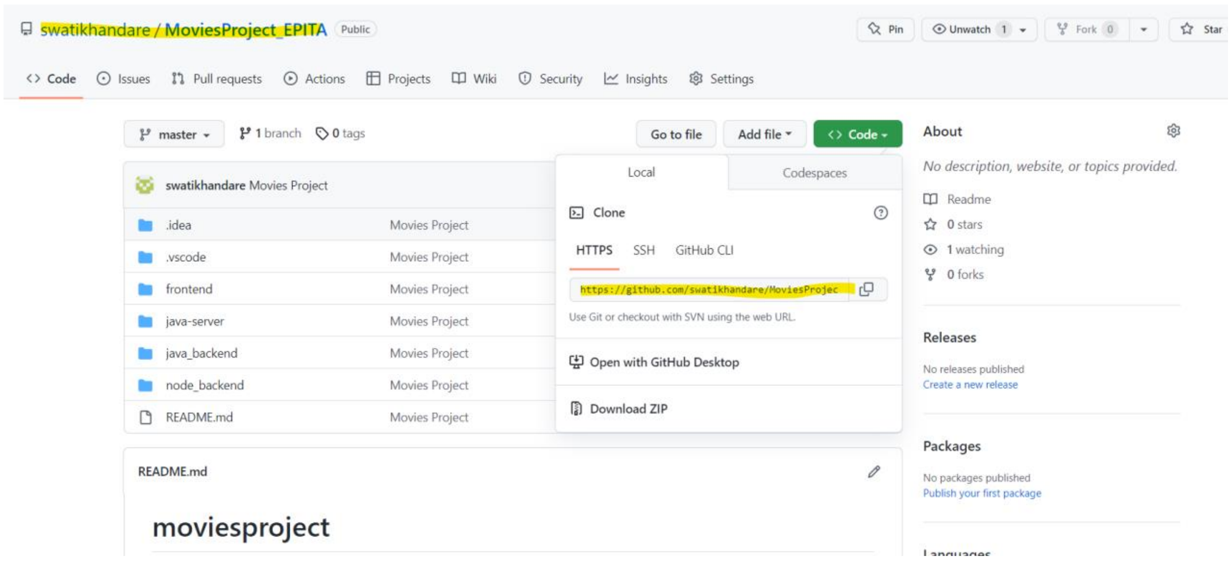
   - HomePage Trailer
   - Continue Watching
   - Latest Movies
   - Recommended Movies
   - Top 10 Most Popular Movies
   - Top 10 Most Rated Movies

   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



swatikhandare Movies Project

.idea	Movies Project
.vscode	Movies Project
frontend	Movies Project
java-server	Movies Project
java_backend	Movies Project
node_backend	Movies Project
README.md	Movies Project

https://github.com/swatikhandare/MoviesProjec

Use Git or checkout with SVN using the web URL.

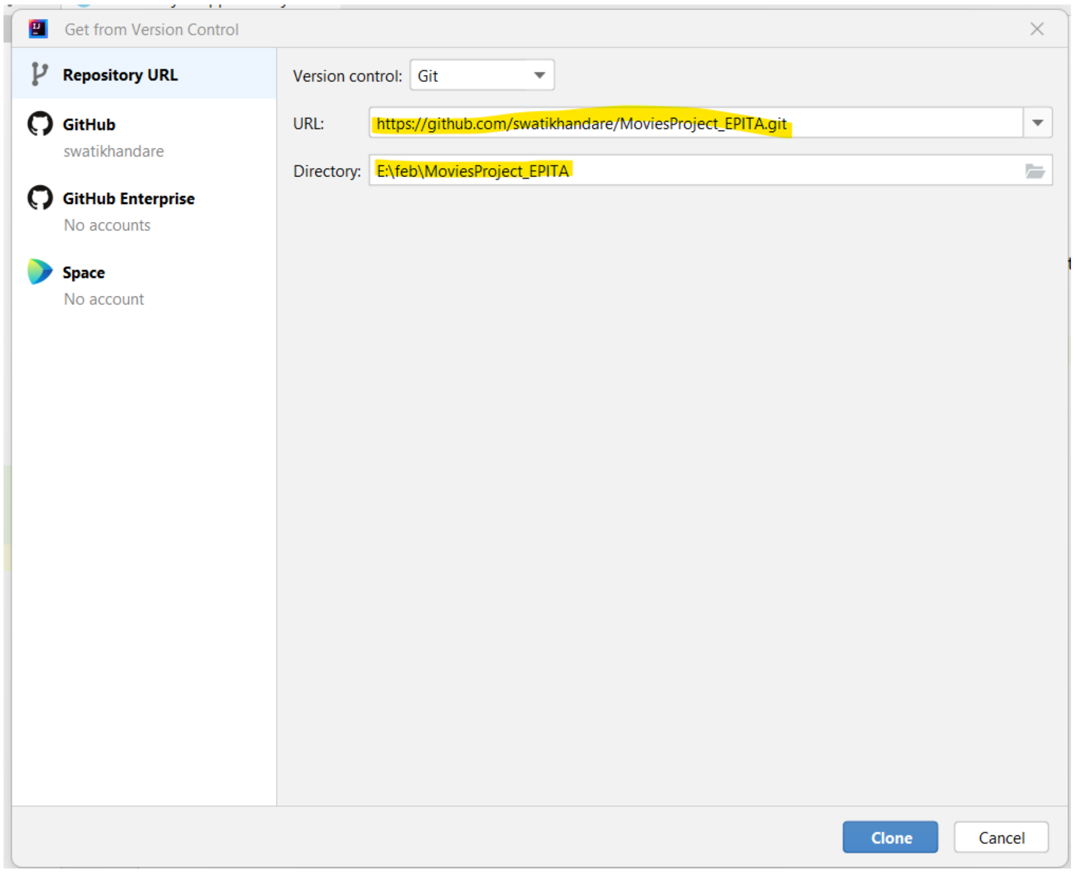
No description, website, or topics provided.

No releases published

No packages published

README.md

moviesproject



Version control: Git

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

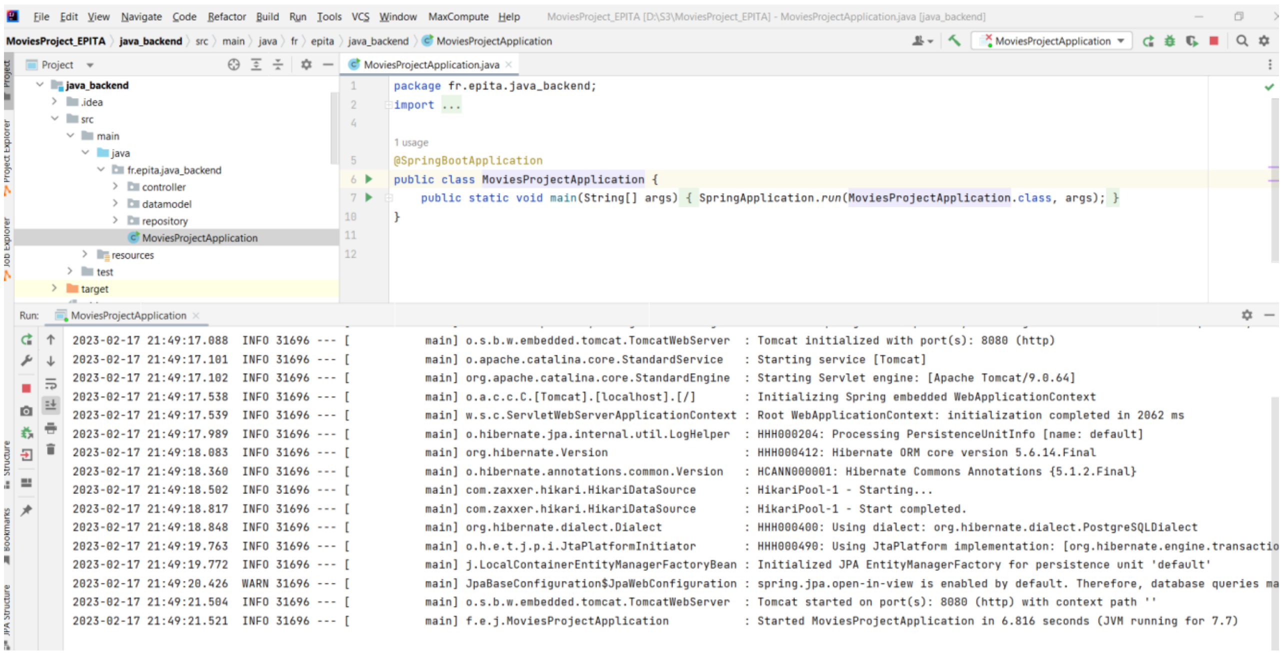
Directory: E:\feb\MoviesProject_EPITA

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.

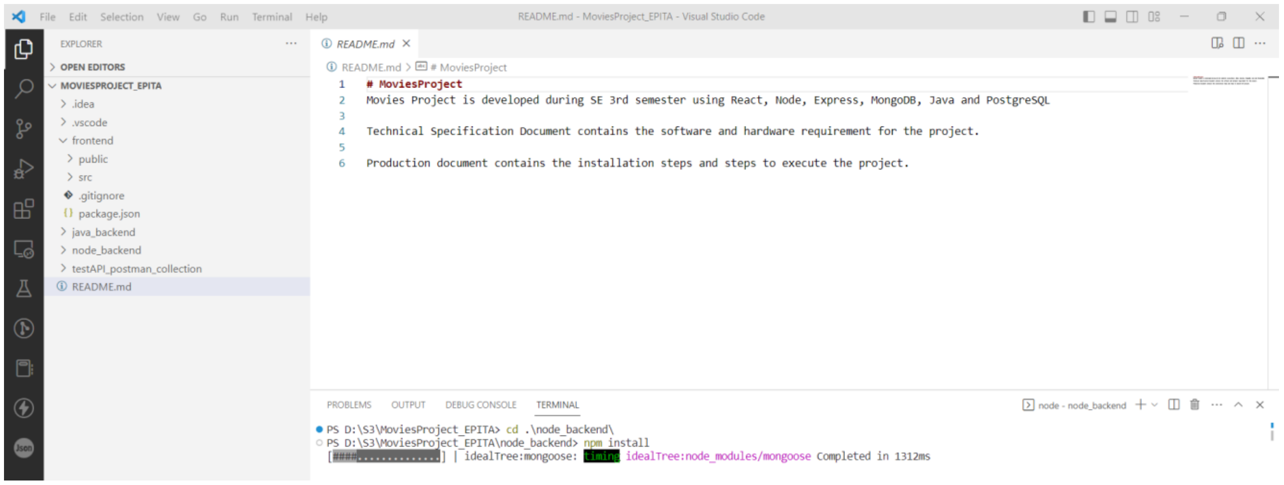


```java
package fr.epita.java_backend;
import ...

@SpringBootApplication
public class MoviesProjectApplication {
    public static void main(String[] args) { SpringApplication.run(MoviesProjectApplication.class, args); }
}
```

```
2023-02-17 21:49:17.088  INFO 31696 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 8080 (http)
2023-02-17 21:49:17.101  INFO 31696 --- [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2023-02-17 21:49:17.102  INFO 31696 --- [           main] org.apache.catalina.core.StandardEngine  : Starting Servlet engine: [Apache Tomcat/9.0.64]
2023-02-17 21:49:17.538  INFO 31696 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationContext
2023-02-17 21:49:17.539  INFO 31696 --- [           main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 2062 ms
2023-02-17 21:49:17.989  INFO 31696 --- [           main] o.hibernate.jpa.internal.util.LogHelper  : HHH000204: Processing PersistenceUnitInfo [name: default]
2023-02-17 21:49:18.083  INFO 31696 --- [           main] org.hibernate.Version                    : HHH000412: Hibernate ORM core version 5.6.14.Final
2023-02-17 21:49:18.360  INFO 31696 --- [           main] o.hibernate.annotations.common.Version   : HCANN000001: Hibernate Commons Annotations {5.1.2.Final}
2023-02-17 21:49:18.502  INFO 31696 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Starting...
2023-02-17 21:49:18.817  INFO 31696 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Start completed.
2023-02-17 21:49:18.848  INFO 31696 --- [           main] org.hibernate.dialect.Dialect            : HHH000400: Using dialect: org.hibernate.dialect.PostgreSQLDialect
2023-02-17 21:49:19.763  INFO 31696 --- [           main] o.h.e.t.j.p.i.JtaPlatformInitiator       : HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transactio
2023-02-17 21:49:19.772  INFO 31696 --- [           main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
2023-02-17 21:49:20.426  WARN 31696 --- [           main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries ma
2023-02-17 21:49:21.504  INFO 31696 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8080 (http) with context path ''
2023-02-17 21:49:21.521  INFO 31696 --- [           main] f.e.j.MoviesProjectApplication           : Started MoviesProjectApplication in 6.816 seconds (JVM running for 7.7)
```

6. Open the project in VSCode.

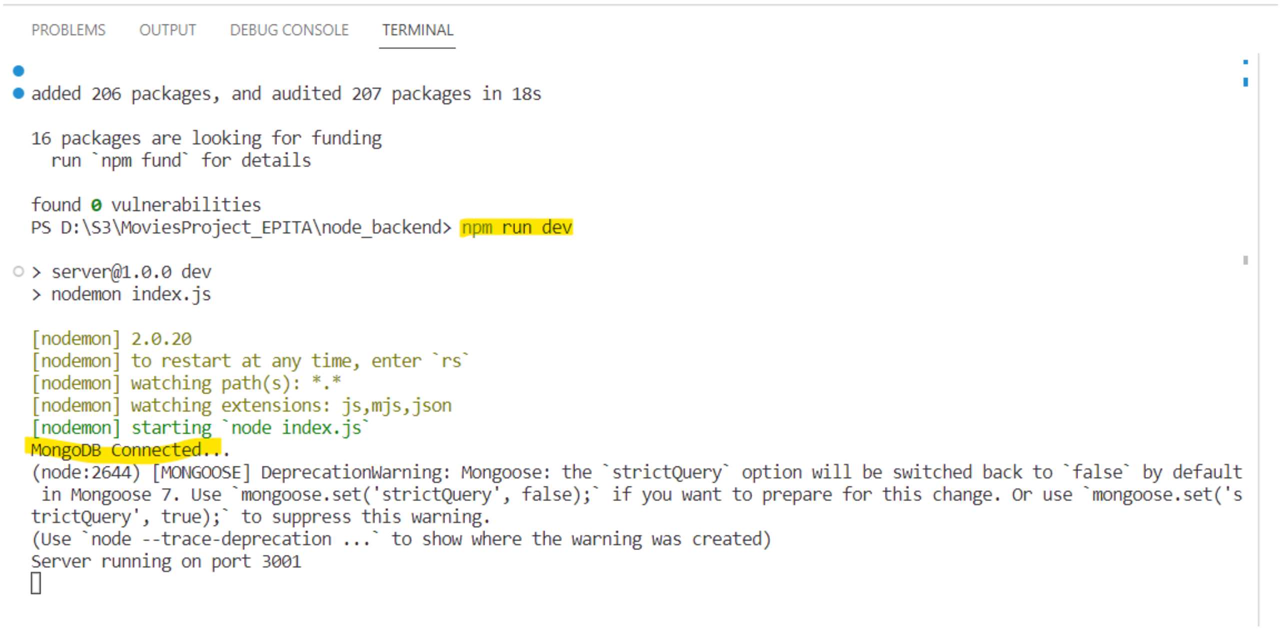
7. Navigate to folder “node_backend” and execute command “npm install”.



```
# MoviesProject
Movies Project is developed during SE 3rd semester using React, Node, Express, MongoDB, Java and PostgreSQL

Technical Specification Document contains the software and hardware requirement for the project.

Production document contains the installation steps and steps to execute the project.
```

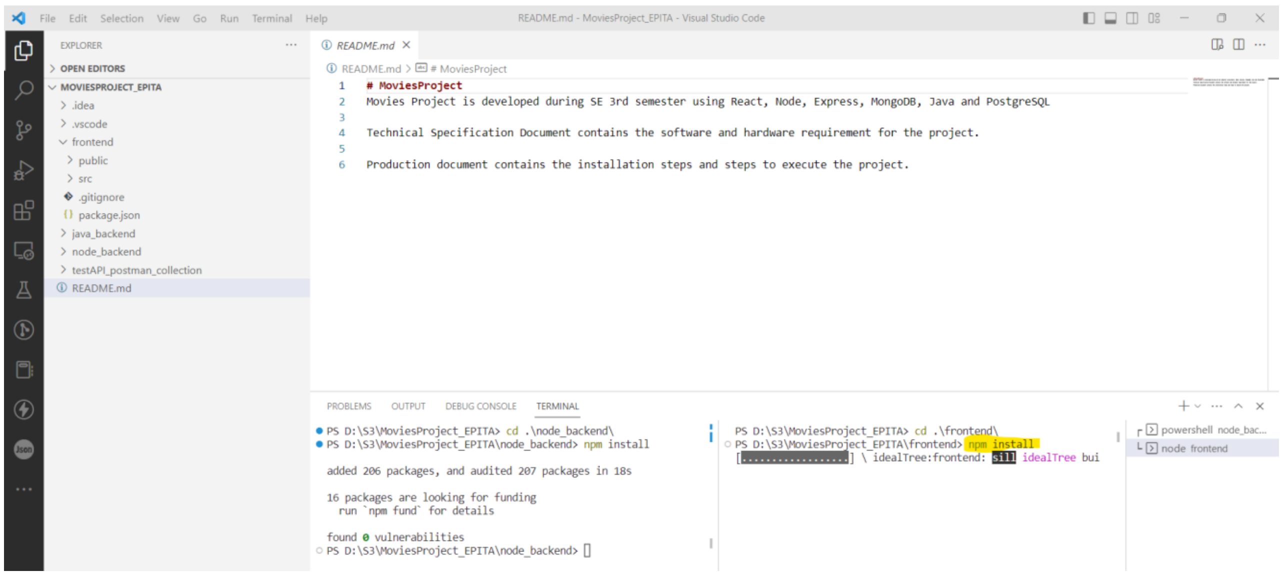
```
PS D:\S3\MoviesProject_EPITA> cd .\node_backend\
PS D:\S3\MoviesProject_EPITA\node_backend> npm install
[####..............] | idealTree:mongoose: timing idealTree:node_modules/mongoose Completed in 1312ms
```

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



```
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”.

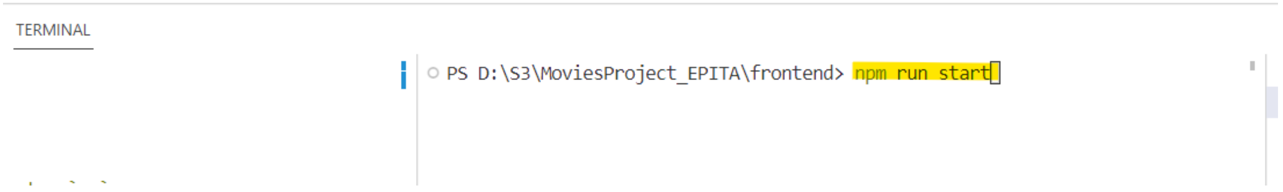


```
# MoviesProject
Movies Project is developed during SE 3rd semester using React, Node, Express, MongoDB, Java and PostgreSQL

Technical Specification Document contains the software and hardware requirement for the project.

Production document contains the installation steps and steps to execute the project.
```

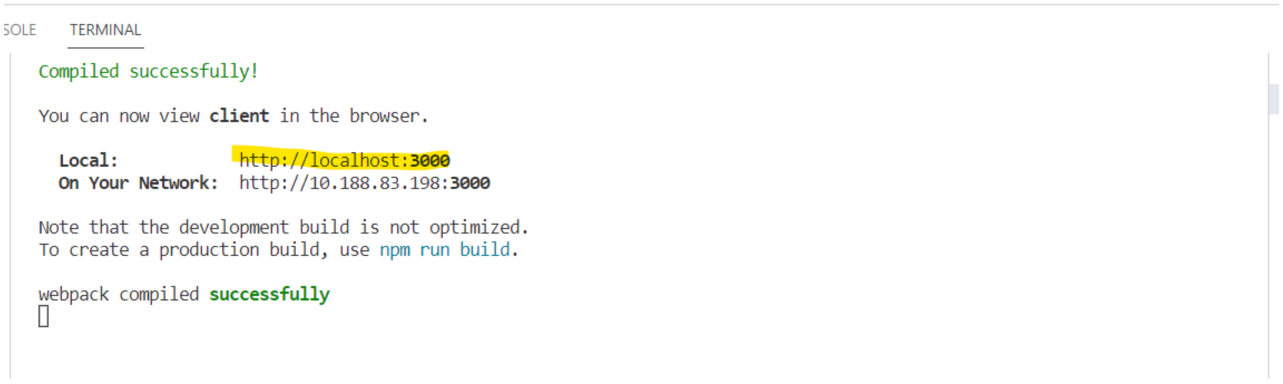
```
PS D:\S3\MoviesProject_EPITA> cd .\node_backend\
PS D:\S3\MoviesProject_EPITA\node_backend> npm install

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>
```

```
PS D:\S3\MoviesProject_EPITA> cd .\frontend\
PS D:\S3\MoviesProject_EPITA\frontend> npm install
[.................] \ idealTree:frontend: sill idealTree bui
```

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

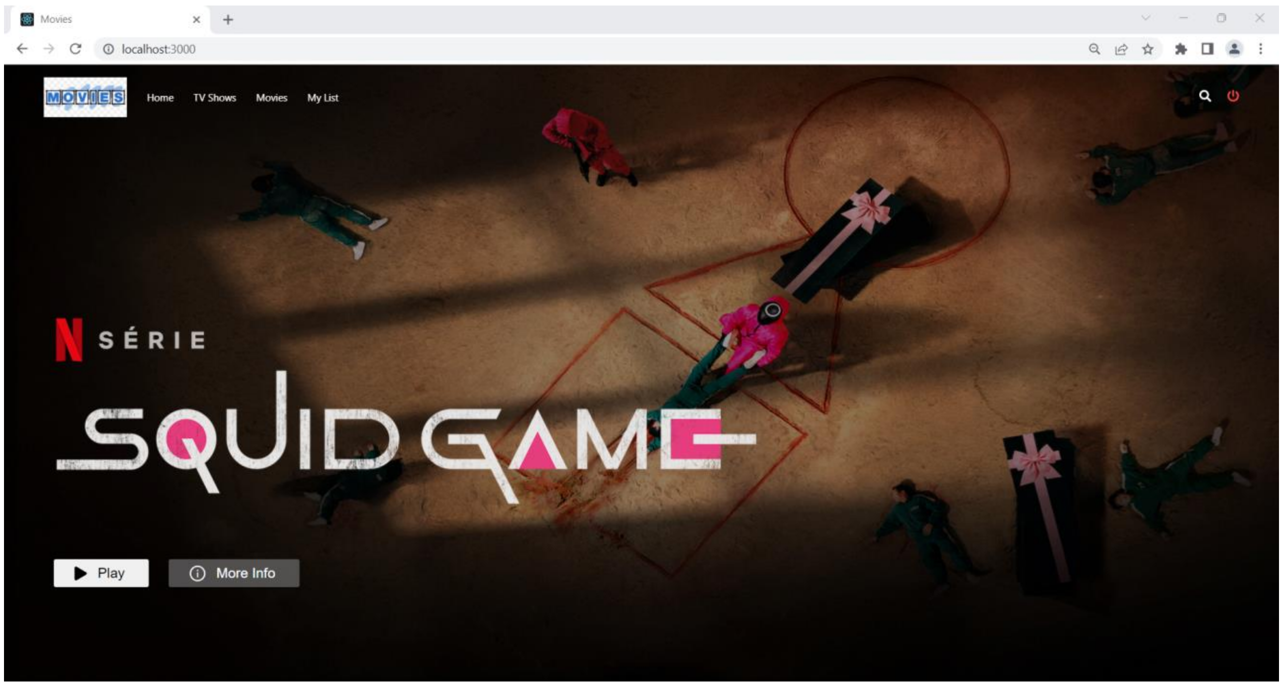


```
PS D:\S3\MoviesProject_EPITA\frontend> npm run start
```



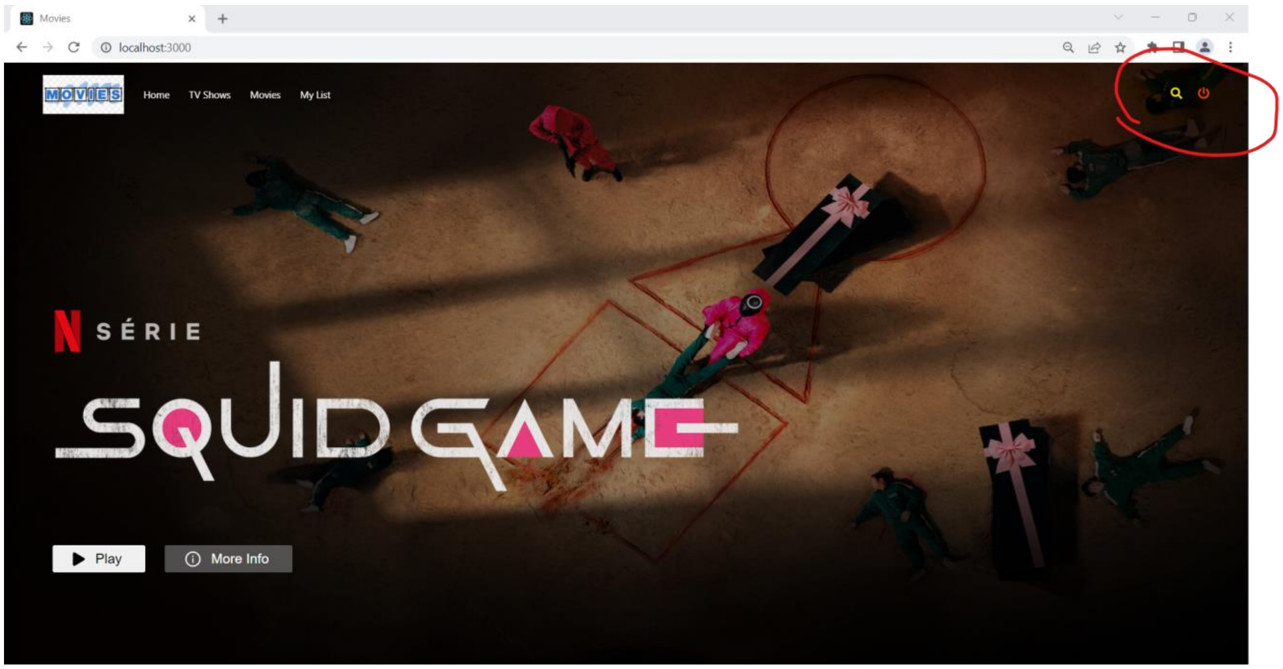
```
Compiled successfully!

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!

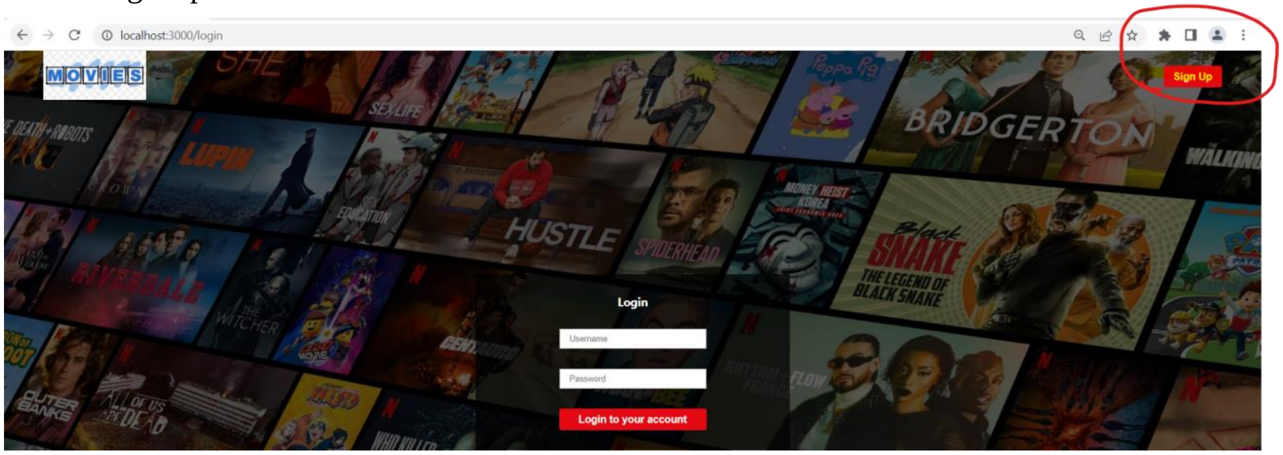


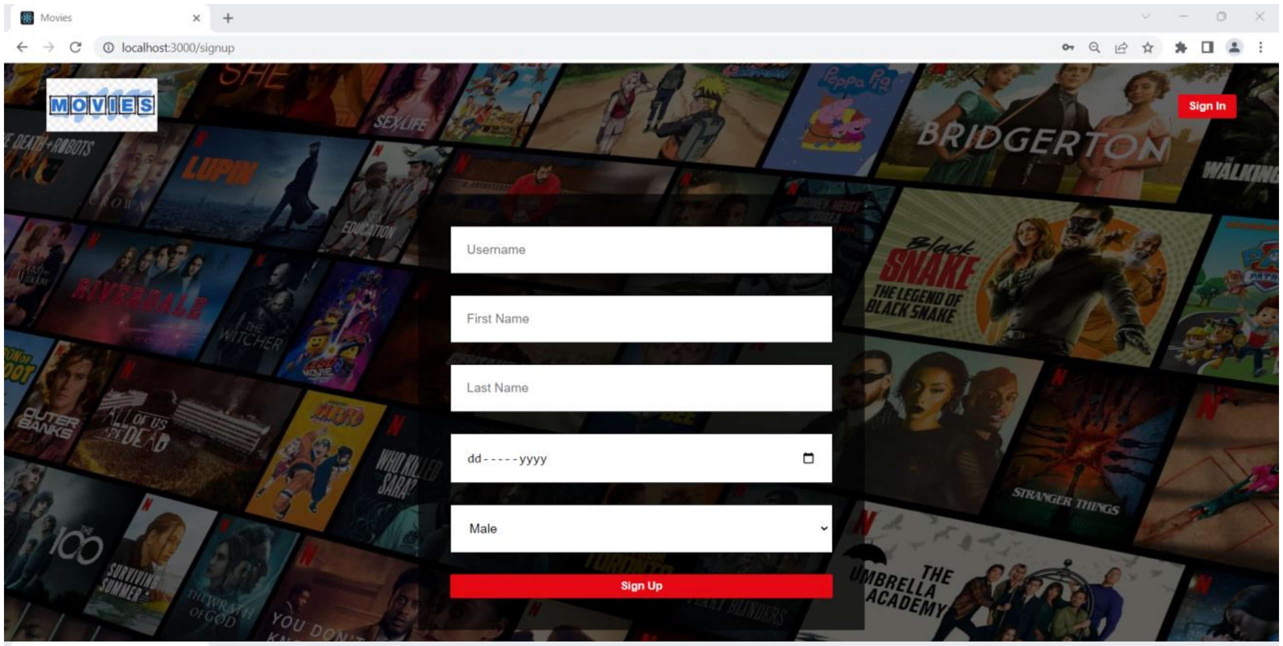
Execution

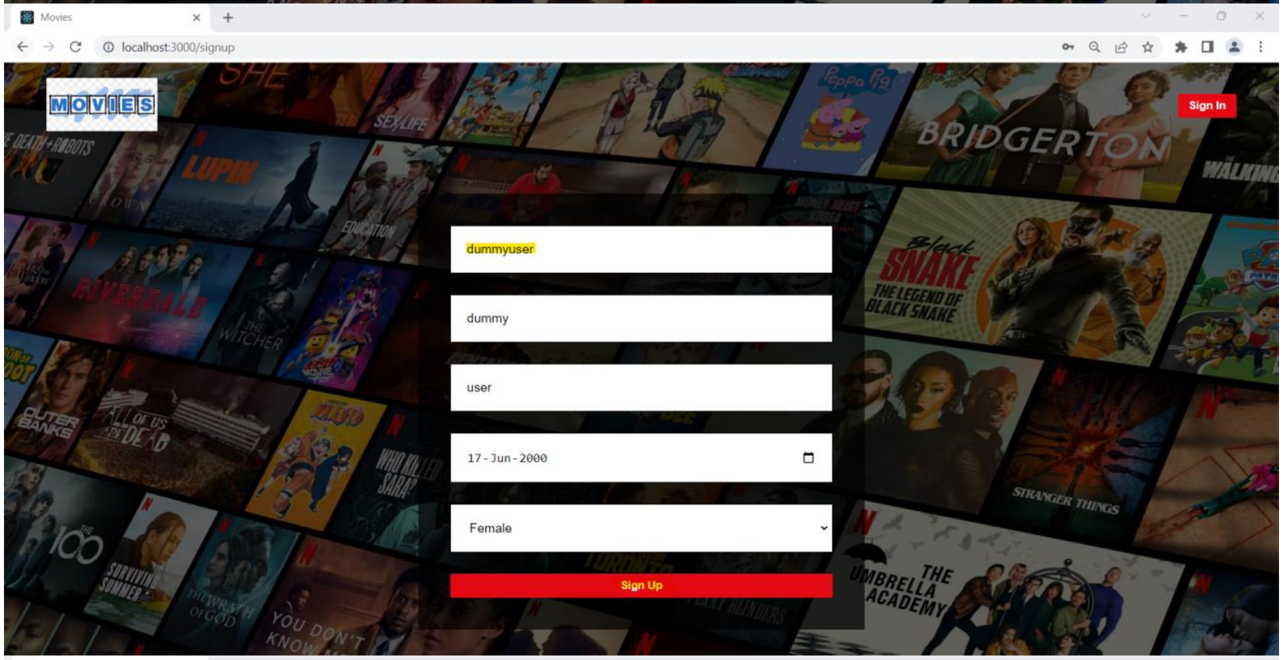
1. Landing page


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

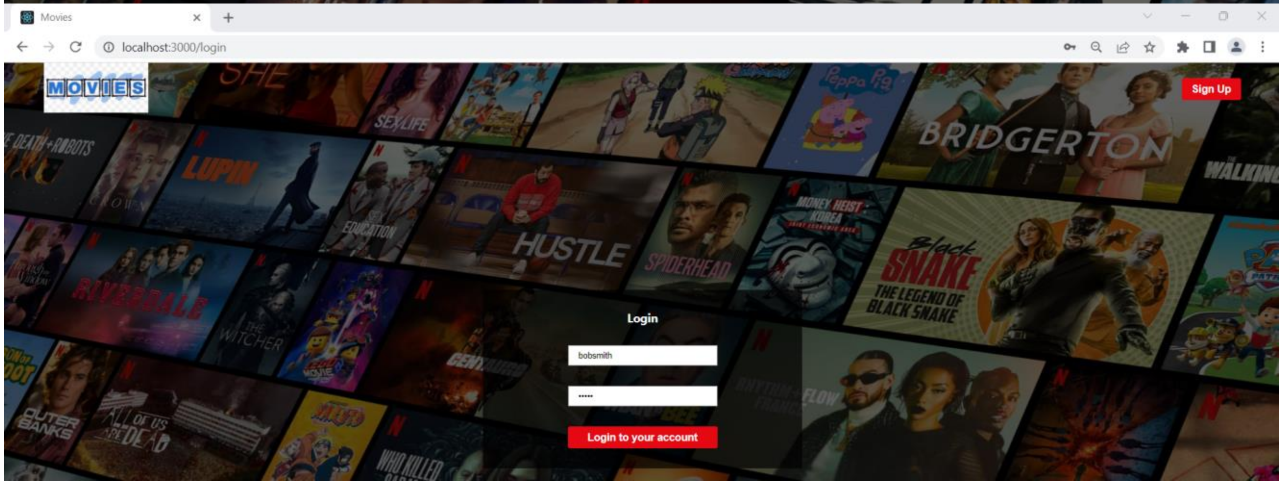
2. Sign Up


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



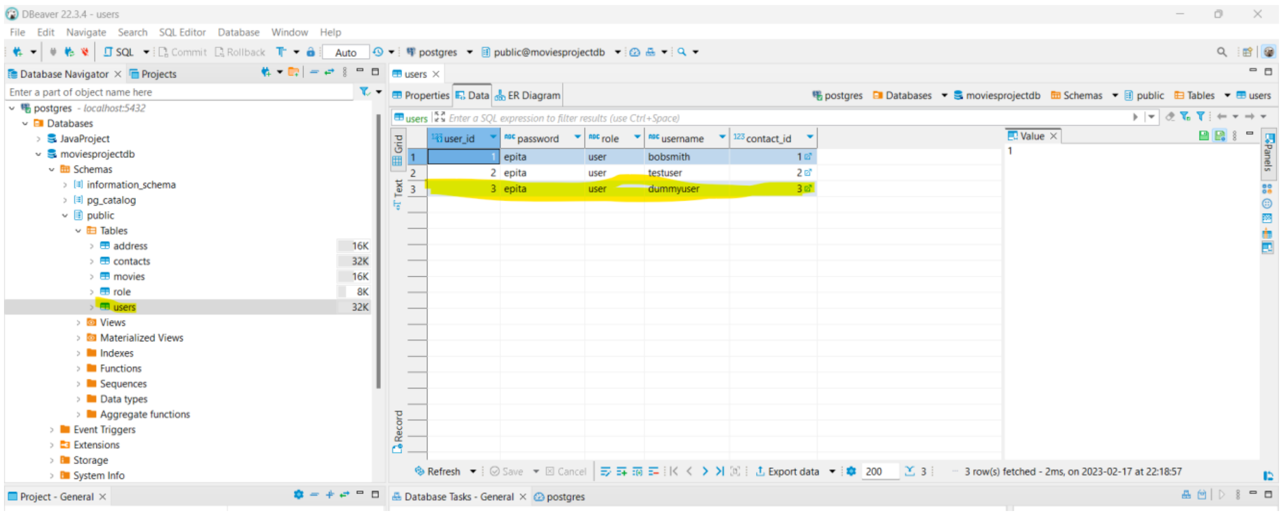


dummyuser
dummy
user
17-Jun-2000
Female



bobsmith
•••••

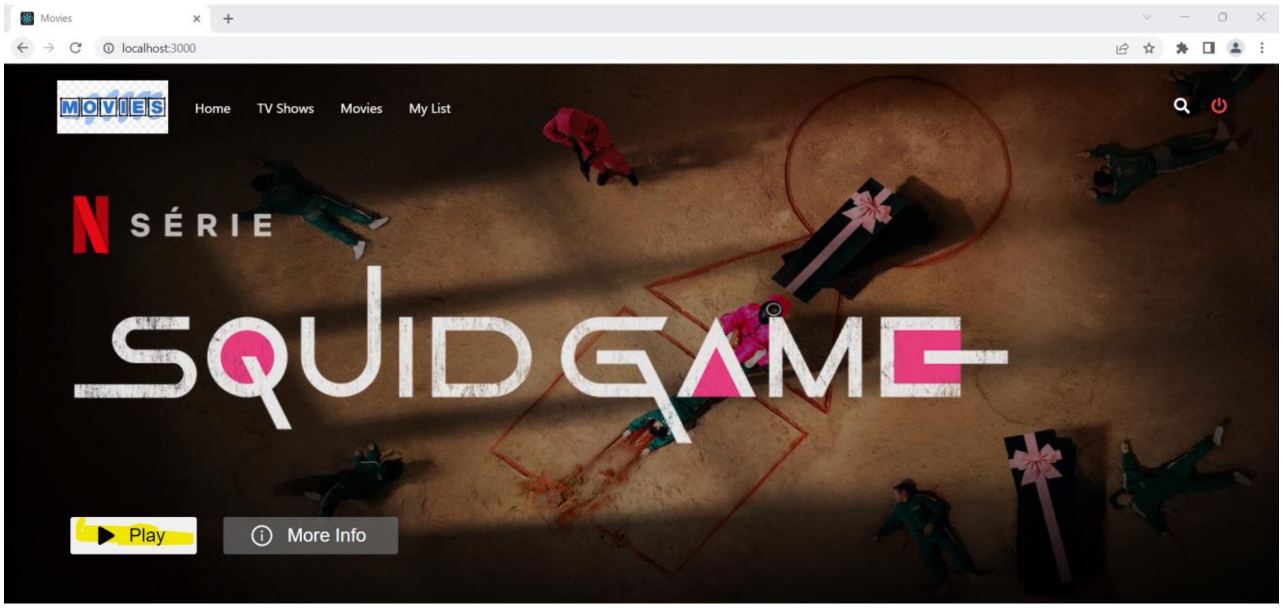
3. User added in users postgresql db



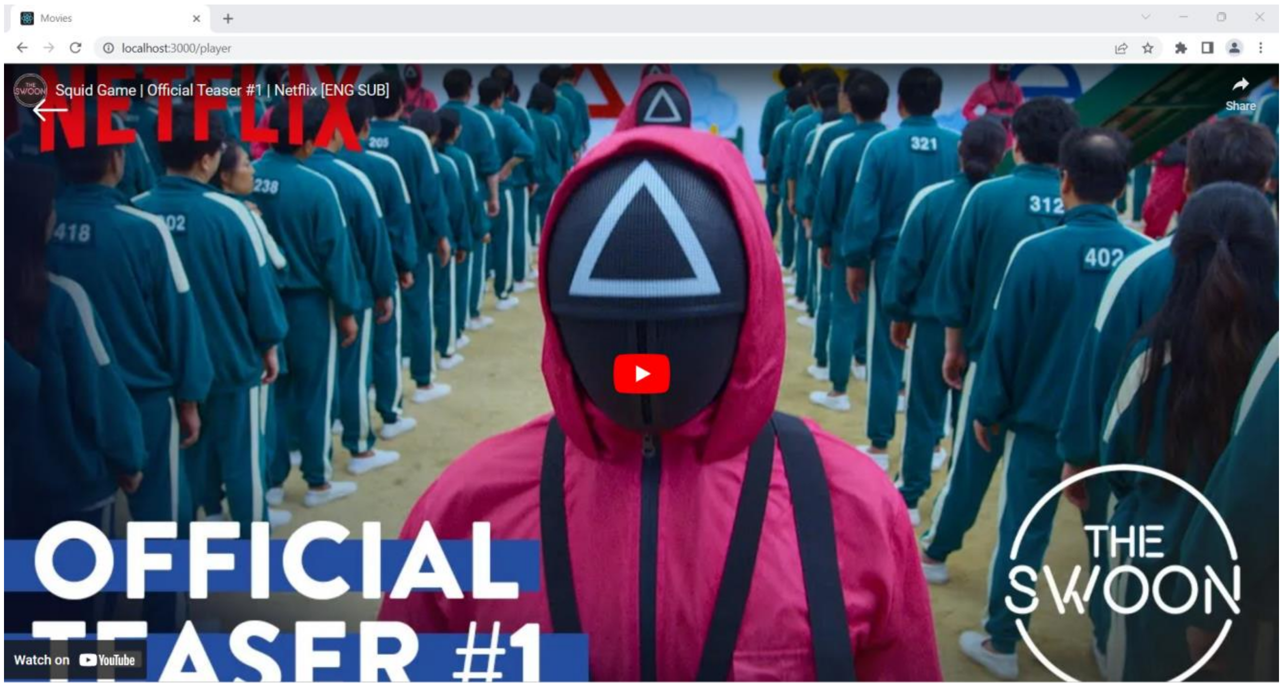
	user_id	password	role	username	contact_id
1	1	epita	user	bobsmith	1
2	2	epita	user	testuser	2
3	3	epita	user	dummyuser	3

3 row(s) fetched - 2ms, on 2023-02-17 at 22:18:57

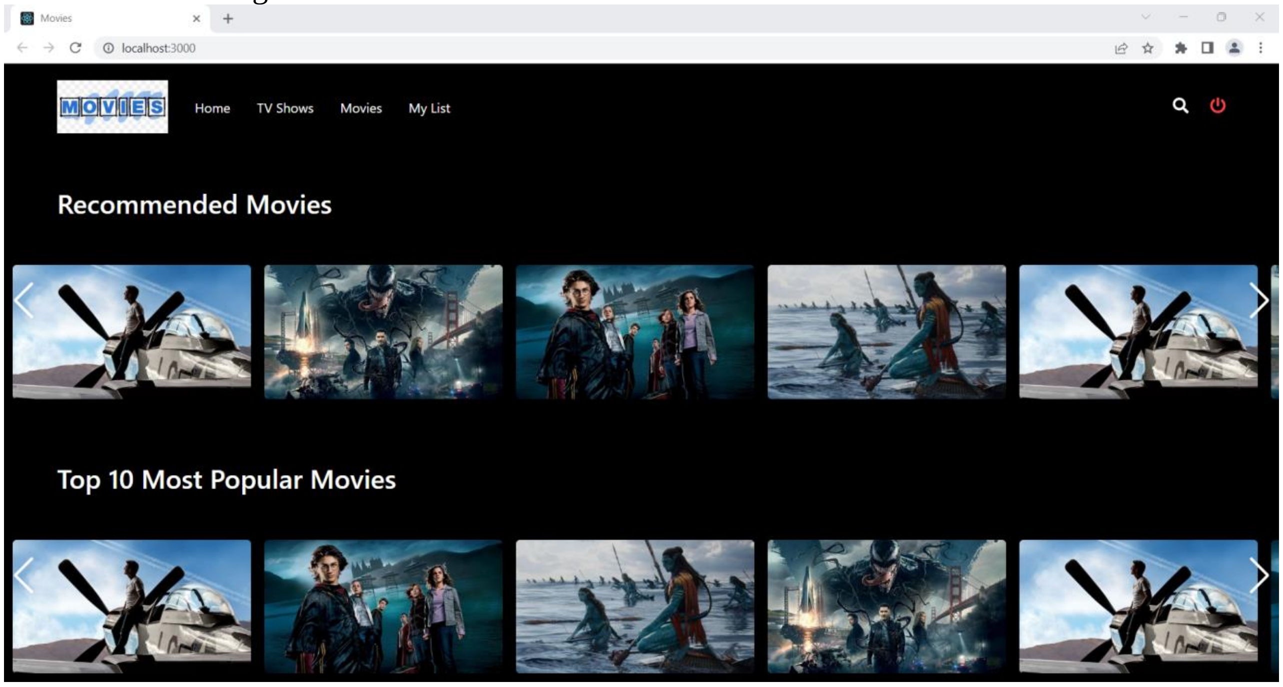
4. Sign in with the user created



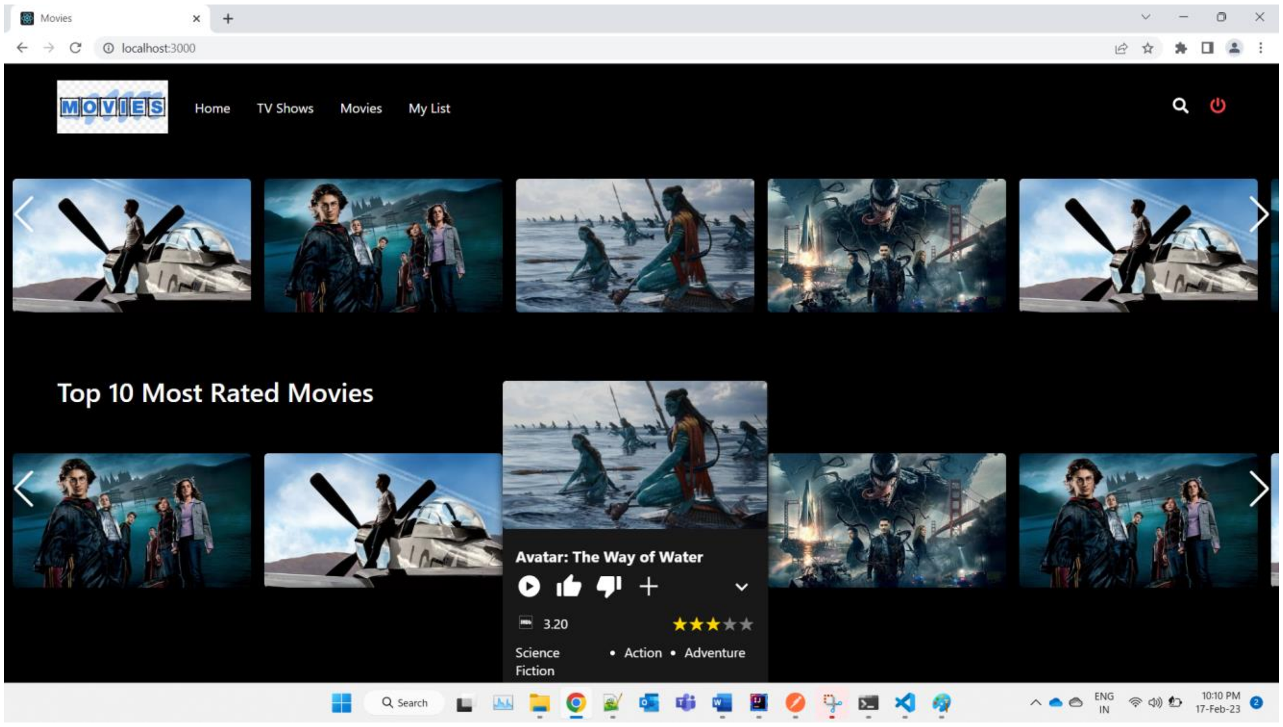
5. Click on play to play the Squid Game trailer.



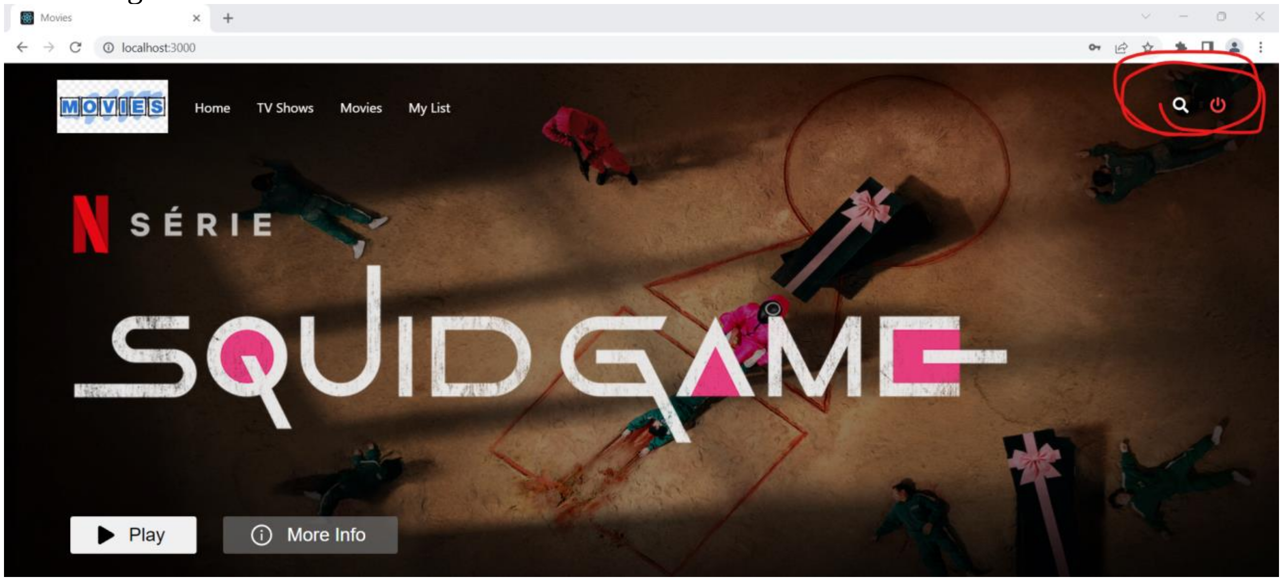
6. Scroll through different sections



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



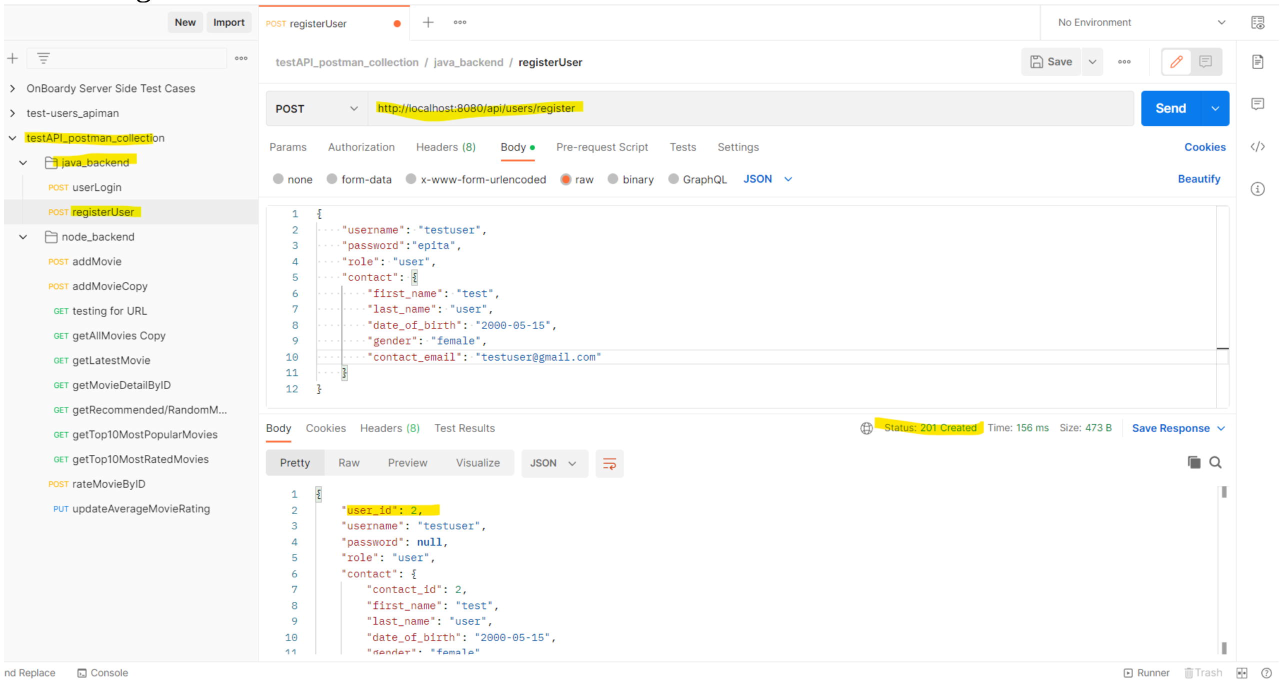
8. Sign out



API testing

Import the postman collection from folder “**testAPI_postman_collection**” 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



testAPI_postman_collection / java_backend / registerUser

POST http://localhost:8080/api/users/register

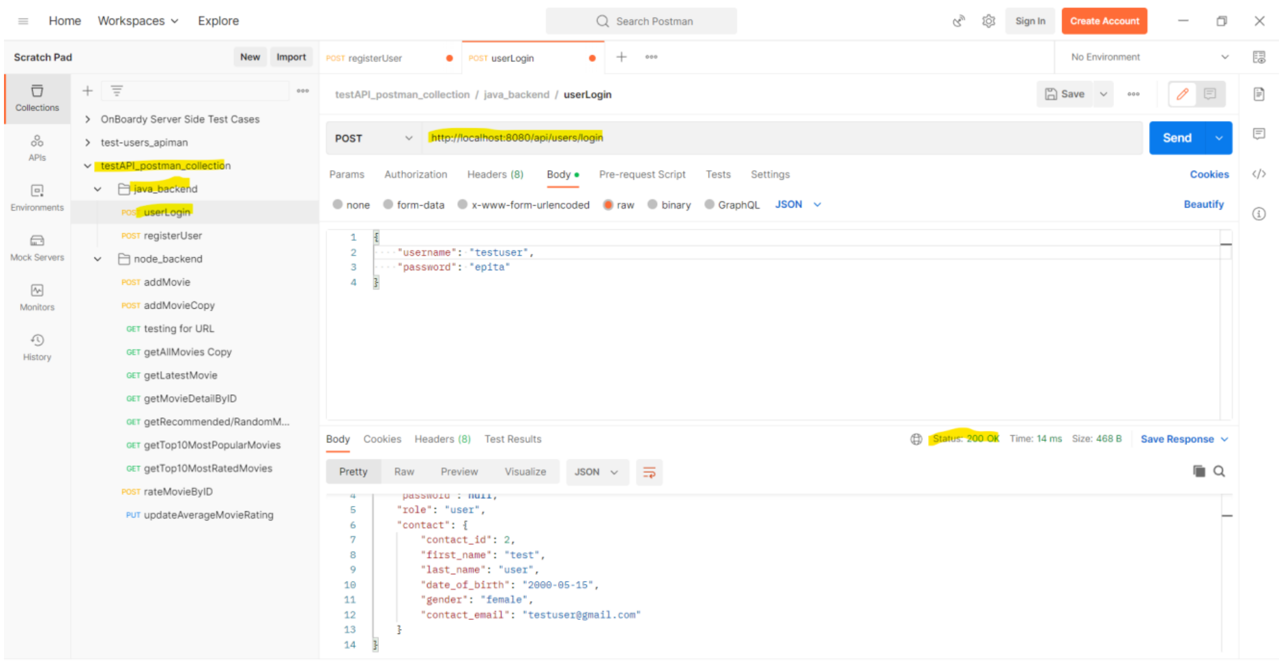
none form-data x-www-form-urlencoded [✓ raw] binary GraphQL JSON

```
{
    "username": "testuser",
    "password":"epita",
    "role": "user",
    "contact": {
        "first_name": "test",
        "last_name": "user",
        "date_of_birth": "2000-05-15",
        "gender": "female",
        "contact_email": "testuser@gmail.com"
    }
}
```

Body Cookies Headers (8) Test Results — Status: 201 Created Time: 156 ms Size: 473 B

```
{
    "user_id": 2,
    "username": "testuser",
    "password": null,
    "role": "user",
    "contact": {
        "contact_id": 2,
        "first_name": "test",
        "last_name": "user",
        "date_of_birth": "2000-05-15",
        "gender": "female"
```

2. User login API



testAPI_postman_collection / java_backend / userLogin

POST http://localhost:8080/api/users/login

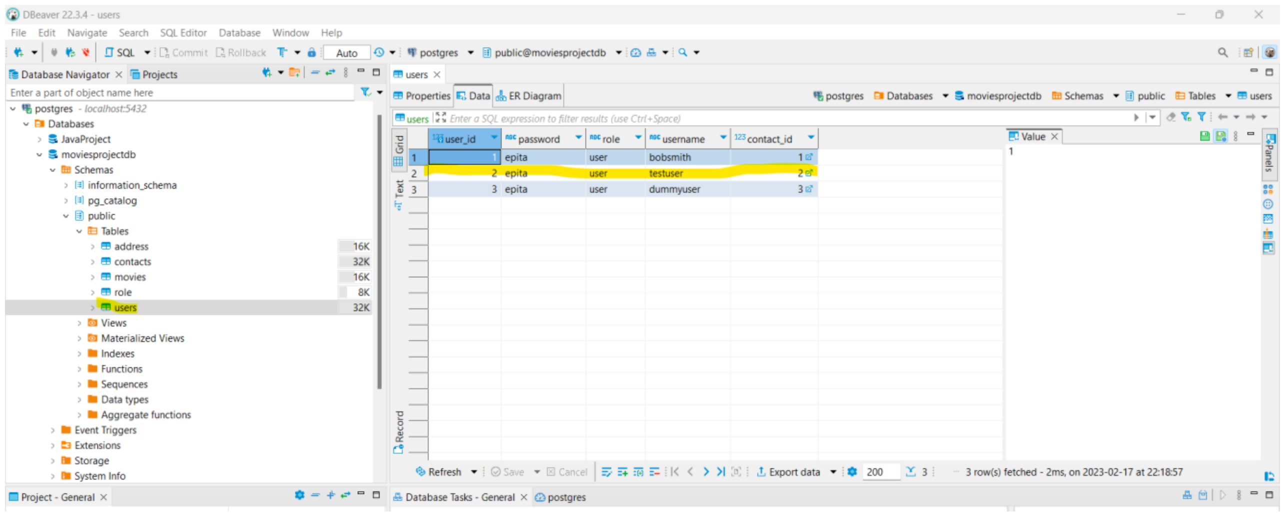
none form-data x-www-form-urlencoded [✓ raw] binary GraphQL JSON

```
{
    "username": "testuser",
    "password": "epita"
}
```

Body Cookies Headers (8) Test Results — Status: 200 OK Time: 14 ms Size: 468 B

```
    "password": null,
    "role": "user",
    "contact": {
        "contact_id": 2,
        "first_name": "test",
        "last_name": "user",
        "date_of_birth": "2000-05-15",
        "gender": "female",
        "contact_email": "testuser@gmail.com"
    }
}
```

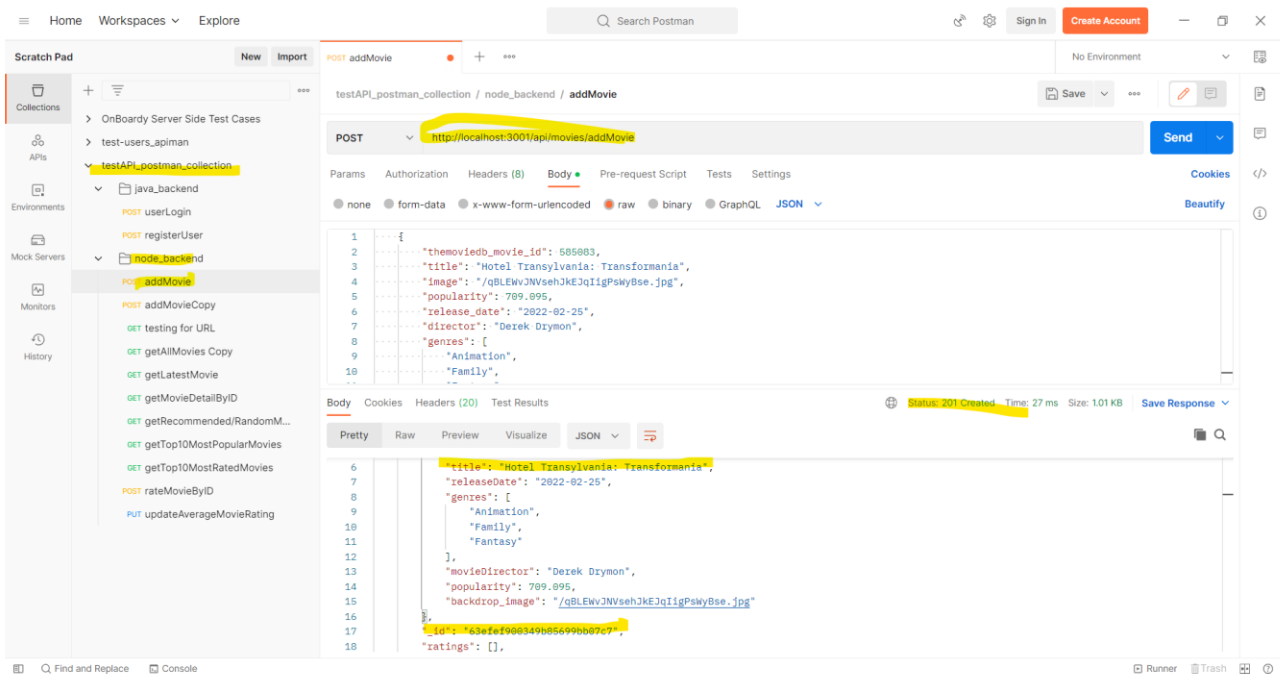
New user is added in the postgresql db table



user_id	password	role	username	contact_id
1	epita	user	bobsmith	1
2	epita	user	testuser	2
3	epita	user	dummyuser	3

3 row(s) fetched - 2ms, on 2023-02-17 at 22:18:57

3. Add movie API



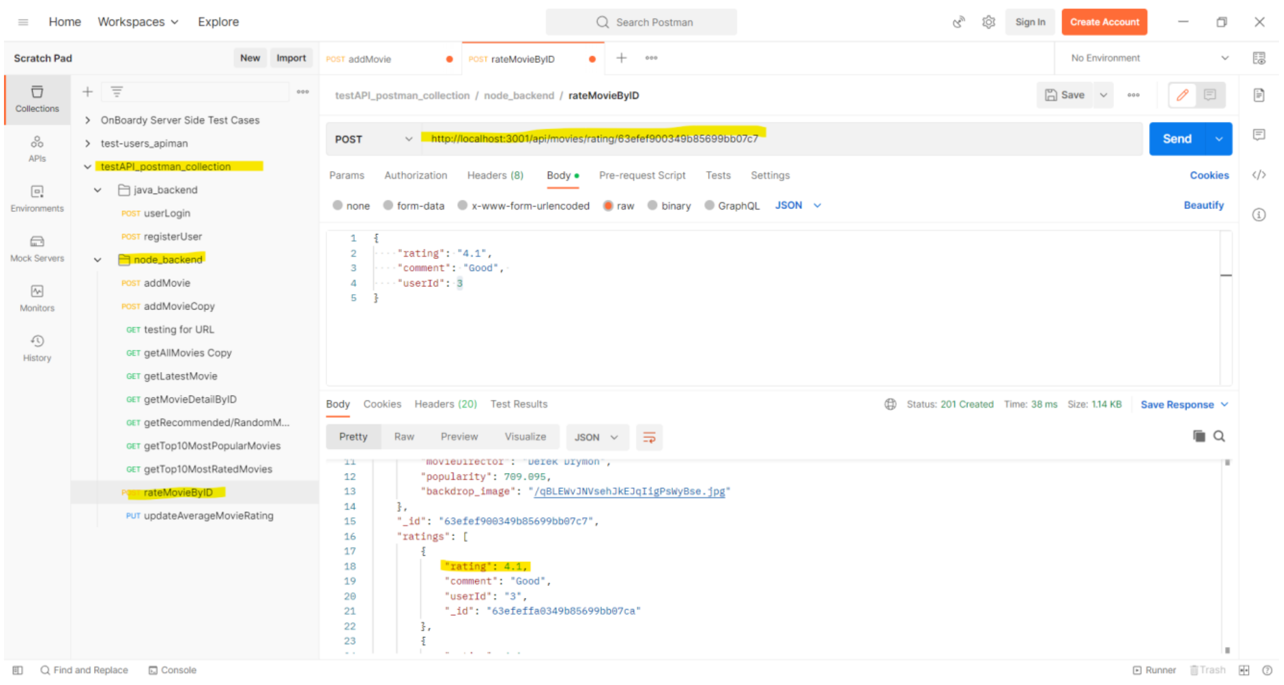
POST http://localhost:3001/api/movies/addMovie

```
{
    "themoviedb_movie_id": 585083,
    "title": "Hotel Transylvania: Transformania",
    "image": "/qBLEWvJNVsehJkEJqIigPsWyBse.jpg",
    "popularity": 709.095,
    "release_date": "2022-02-25",
    "director": "Derek Drymon",
    "genres": [
        "Animation",
        "Family",
```

Status: 201 Created Time: 27 ms Size: 1.01 KB

```
        "title": "Hotel Transylvania: Transformania",
        "releaseDate": "2022-02-25",
        "genres": [
            "Animation",
            "Family",
            "Fantasy"
        ],
        "movieDirector": "Derek Drymon",
        "popularity": 709.095,
        "backdrop_image": "/qBLEWvJNVsehJkEJqIigPsWyBse.jpg"
    },
    "_id": "63efef900349b85699bb07c7",
    "ratings": [],
```

4. Add movie rating by movie id API



testAPI_postman_collection / node_backend / rateMovieByID

POST http://localhost:3001/api/movies/rating/63efef900349b85699bb07c7

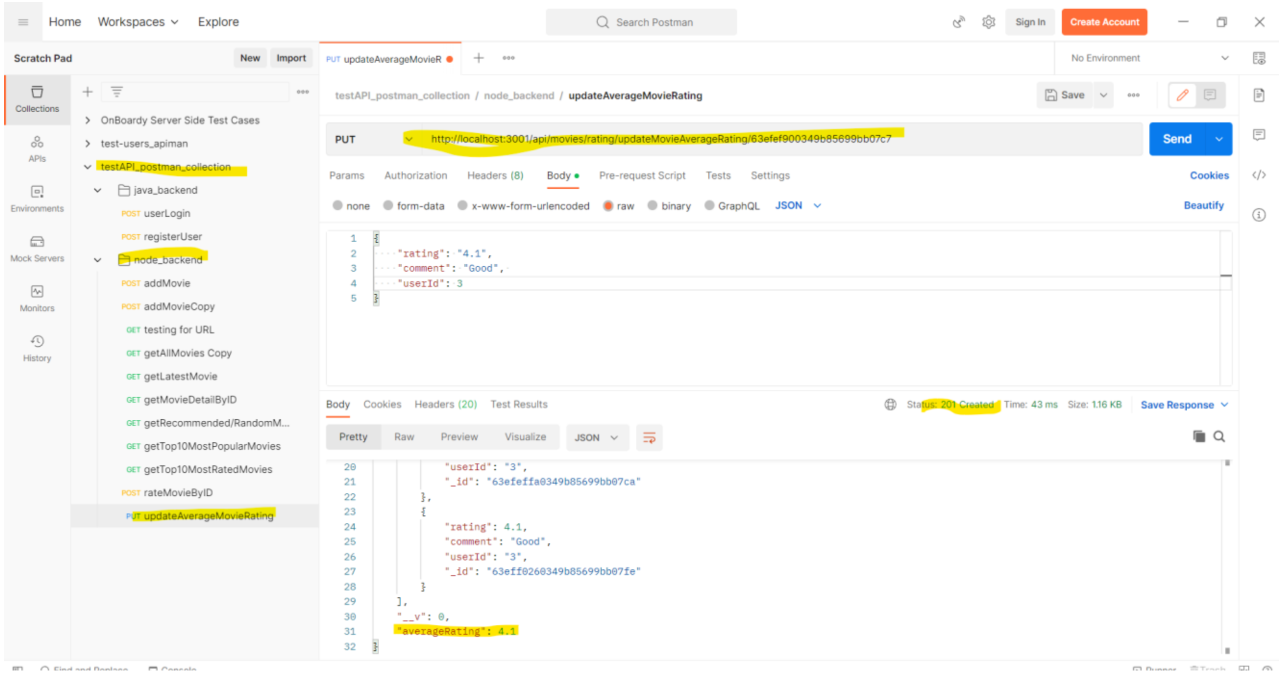
Body: raw, JSON

```
{
    "rating": "4.1",
    "comment": "Good",
    "userId": 3
}
```

Status: 201 Created Time: 38 ms Size: 1.14 KB

```
            "movieDirector": "Derek Drymon",
            "popularity": 709.095,
            "backdrop_image": "/qBLEWvJNVsehJkEJqIigPsWyBse.jpg"
        },
        "_id": "63efef900349b85699bb07c7",
        "ratings": [
            {
                "rating": 4.1,
                "comment": "Good",
                "userId": "3",
                "_id": "63efeffa0349b85699bb07ca"
            },
            {
```

5. Add average rating by movie id API



testAPI_postman_collection / node_backend / updateAverageMovieRating

PUT http://localhost:3001/api/movies/rating/updateMovieAverageRating/63efef900349b85699bb07c7

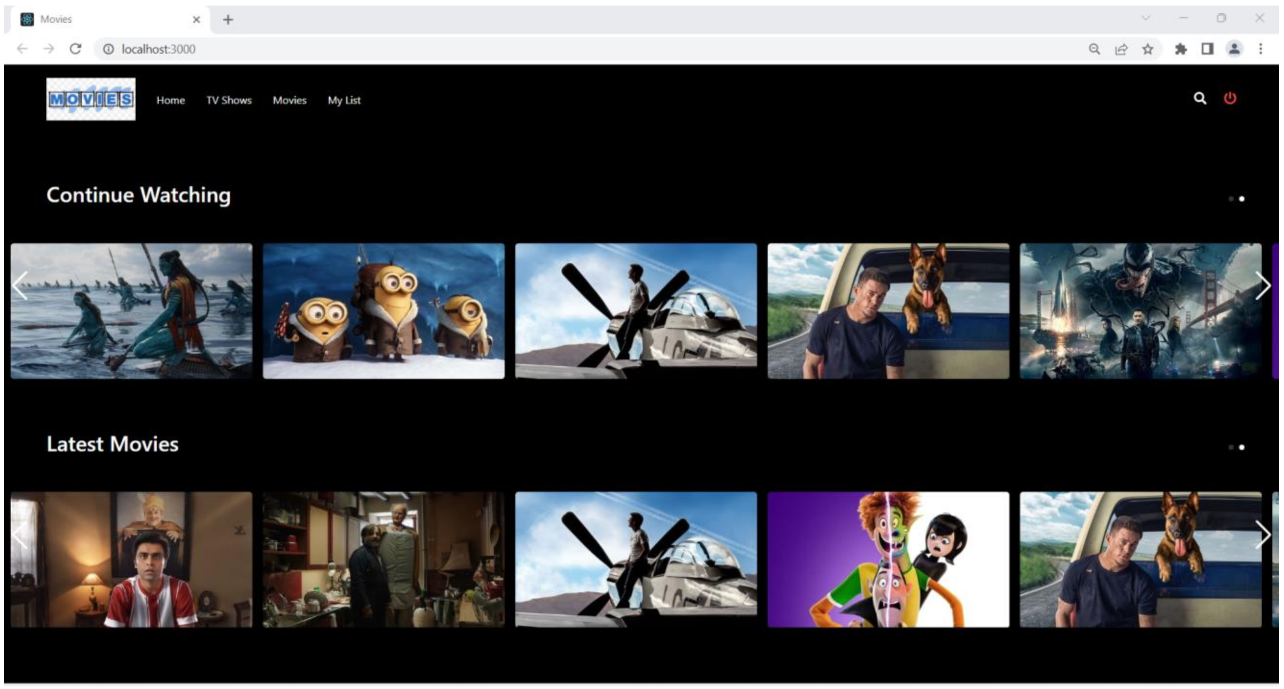
Body: raw, JSON

```
{
    "rating": "4.1",
    "comment": "Good",
    "userId": 3
}
```

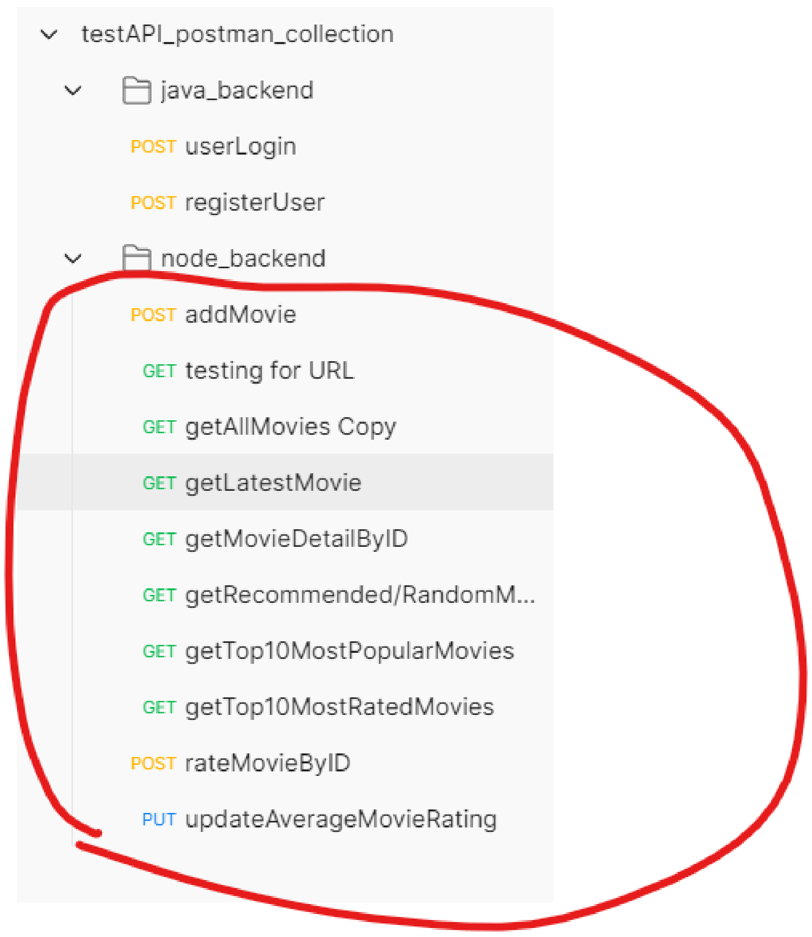
Status: 201 Created Time: 43 ms Size: 1.16 KB

```
                "userId": "3",
                "_id": "63efeffa0349b85699bb07ca"
            },
            {
                "rating": 4.1,
                "comment": "Good",
                "userId": "3",
                "_id": "63eff0260349b85699bb07fe"
            }
        ],
        "__v": 0,
        "averageRating": 4.1
}
```

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