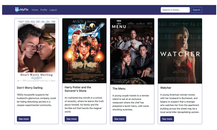
Full-Stack Case Study
myFlix Project

Stela Ceaicovscaia

Purpose

myFlix is a web application designed with the MERN stack. It offers users access to movie-related information, including details about films, directors, and genres. Users are able to create an account, modify their personal information, and create a collection of favorite movies



Purpose

myFlix was developed as a personal project during my web development course at CareerFoundry to showcase my skills in full-stack JavaScript development.

Objective

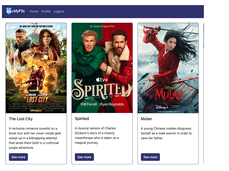
The project's objective was to build a comprehensive server-side and client-side web application entirely from scratch, with the intention of adding it to my professional portfolio.

Duration

It took me approximately 2-3 weeks to build the server side of the application. However, the development of the client side extended tp over a month due to my learning curve with a new tools - React and Redux

Methodologies

- MERN Stack
- Postman
- Heroku
- React Bootstrap
- Redux



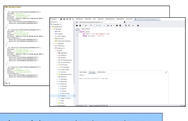
Credits

Lead Developer: Stela Ceaicovscaia
Tutor: Adewunmi Bamishigbin
Mentor: Emmanuel Nsambu

Server-Side

I created a RESTful API using Node.js and Express, which communicates with a non-relational MongoDB database. This API is accessible through standard HTTP methods such as GET, POST or DELETE. The CRUD method is used to fetch and store data in the database, offering movie information in JSON format.

Tests conducted with both PostGreSQL and MongoDB.
I opted for MongoDB due to its non-relational nature and flexibility.



Then, I developed models to maintain consistent data formatting and utilised Mongoose to interact with the database

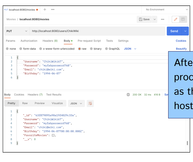
```
const mongoose = require('mongoose');
const Models = require('./models.js');

const Movies = Models.Movie;
const Users = Models.User;

//mongoose.connect('mongodb://localhost:27017/cfDB', {useNewUrlParser: true, useUnifiedTopology: true});
mongoose.connect(process.env.CONNECTION_URI, {useNewUrlParser: true, useUnifiedTopology: true});
```

Server-Side

Basic HTTP for the initial login, coupled with JWT token-based authorization was implemented.
Additionally, I integrated CORS, employed password hashing, and implemented data validation to enhance the overall security of the application.



After testing all endpoints using Postman, I proceeded to deploy my application, using Heroku as the hosting platform and MongoDB Atlas to host my database.

Client-Side

Once API was finalized, I shifted my focus to crafting the user interface that would enable users to interact with the server-side effectively. This interface is a single-page responsive application developed using React and React-Redux. It offers several interface views, including:

Main View

Displays a list of all available movies

Single Movie View

Presents detailed information about a specific movies and allows users to add it to their list of favourite movies

Signup/Login Views

Enables users to create an account/
Provides a platform for user authentication

Profile View

Allows users to update their personal information and manage their list of favorite movies

Retrospective

The original intention was to create a web application with new tools, frameworks and databases. Time management posed a significant challenge since I was working full-time alongside my studies. Nevertheless, I'm content with the outcome and proud of my application. Moving forward, I plan to enhance my skills by practicing and embarking on more projects using React and Redux.

Server-Side GitHub Link

Client-Side GitHub Link

Live Version Link