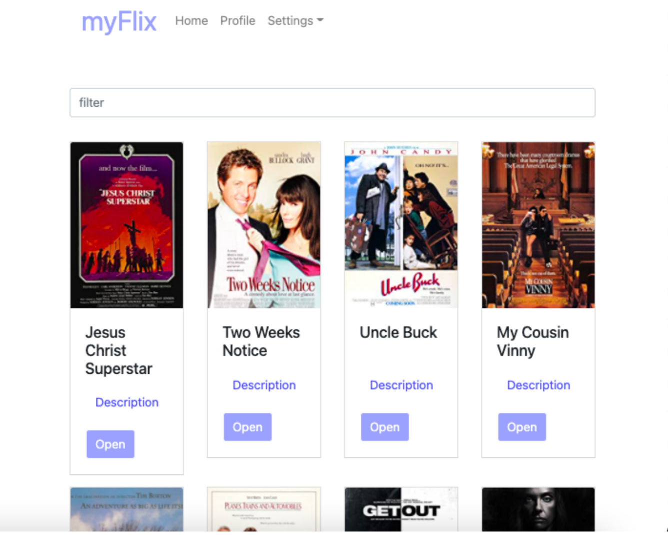
myFlix Case Study

Overview

Developed using the MERN stack, myFlix allows users to browse movies, view details on movie genres and directors, and create a custom list of their favorite movies. Users may also update their personal information or deactivate their account.

Purpose & Context

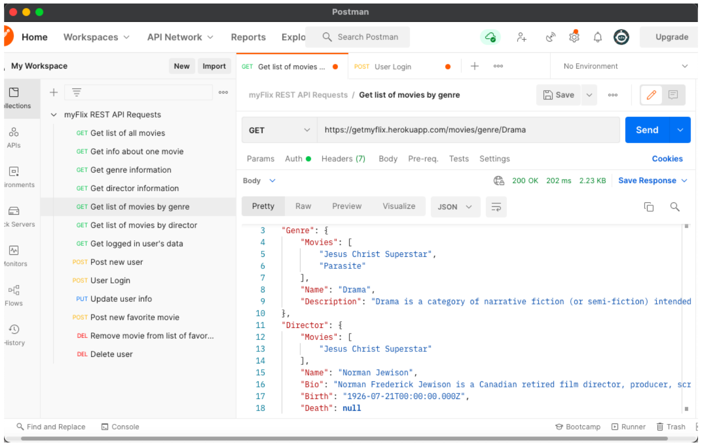
The goal of this project was to complete a truly full-stack project, bringing together both the front-end and back-end skills I had been building throughout the course. Specifically, my aim was to build the complete server-side and client-side for the application from scratch.



Objective

Using the MERN stack, I set out to construct a Netlifx-like application that would allow users to browse, learn about, and save movies to a list of their favorites.

Approach



myFlix REST API Requests

- GET Get list of all movies
- GET Get info about one movie
- GET Get genre information
- GET Get director information
- GET Get list of movies by genre
- GET Get list of movies by director
- GET Get logged in user's data
- POST Post new user
- POST User Login
- PUT Update user info
- POST Post new favorite movie
- DEL Remove movie from list of favor...
- DEL Delete user

myFlix REST API Requests / Get list of movies by genre

GET https://getmyflix.herokuapp.com/movies/genre/Drama

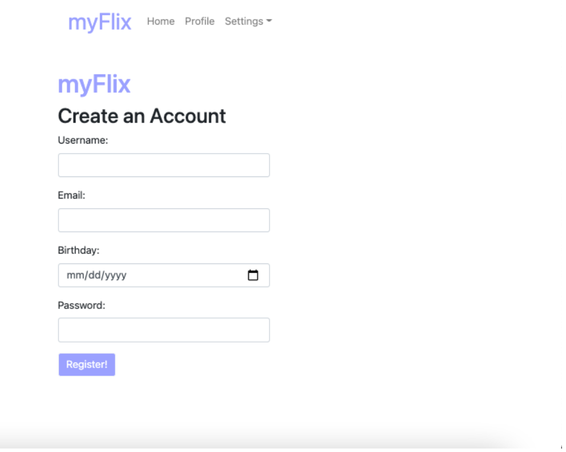
```
"Genre": {
    "Movies": [
        "Jesus Christ Superstar",
        "Parasite"
    ],
    "Name": "Drama",
    "Description": "Drama is a category of narrative fiction (or semi-fiction) intended
},
"Director": {
    "Movies": [
        "Jesus Christ Superstar"
    ],
    "Name": "Norman Jewison",
    "Bio": "Norman Frederick Jewison is a Canadian retired film director, producer, scr
    "Birth": "1926-07-21T00:00:00.000Z",
    "Death": null
```

Server-Side

Using Express and Node.js, I built a REST API containing movie, genre, and director data, which can be accessed by URL endpoints. The movie API is hosted on Heroku and was tested with Postman. The API interacts with a non-relational MongoDB database, and uses CRUD methods to retrieve, store, or manipulate data.

Client-Side

I built the client side using React and React Redux, as React's mobile-friendly, SPA (single page application) format lent itself well to the intended use of this project-- an app where users can easily view movie data and personal data, whether on the go or at home on their desktop. The component-based nature of React allowed for easy creation of the various app views, including a main view where users can browse all movies, a movie view for reading more about a specific movie, genre and director views for additional details respectively, and a user view for managing user data.



myFlix

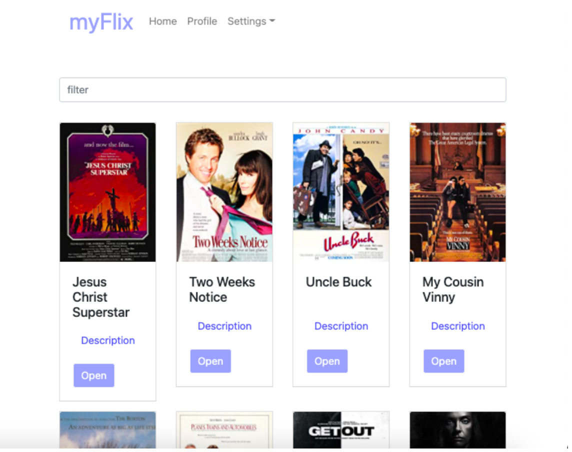
Create an Account

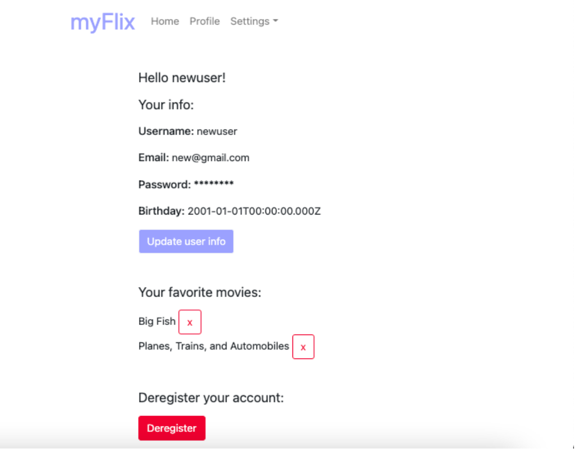
Username:

Email:

Birthday: mm/dd/yyyy

Password:





Hello newuser!

Your info:

Username: newuser

Email: new@gmail.com

Password: ********

Birthday: 2001-01-01T00:00:00.000Z

Your favorite movies:

Big Fish x

Planes, Trains, and Automobiles x

Deregister your account:

Challenges

Being my first comprehensive full-stack endeavor, I faced a number of challenges in developing myFlix. While my data background allowed me to enjoy API-construction and endpoint-testing, and my enthusiasm for creative work made implementing and refining the app's UI a delightful rabbithole, I faced particular difficulty with wrapping my head around the mechanics of React state-changing. Through reviewing course materials, seeking out tutorials on Medium, StackOverflow, and YouTube, and finally, bringing any remaining unanswered questions to my tutor and mentor, I was able to identify and fill in the gaps in my understanding and implement the desired state changes and user flow.

Duration

This project took me roughly 3 months to complete. The learning curve associated with React, and specifically React hooks and state-changing, added significant time to the development process.

Reflections

This being my first robust full stack project, I am proud of having built this web app from scratch-- from constructing the API to designing and implementing a friendly and aesthetic UI. When interacting with the app, I hope users feel curious, eager to click around, and delighted to read more about movies that interest them.

This project taught me a great deal about JavaScript frameworks, and also about my approach to handling development challenges. I learned how frameworks like React greatly streamline the development process by providing flexible and easily scalable structure to your project. I also discovered areas for growth in how I respond to feeling confused and frustrated. This project allowed me to hone the critically important skills of leveraging online resources and community forums, distilling my confusion into specific questions that I could google or ask my mentor, and knowing when to take a break from coding, as well as how to return to it with a fresh perspective. I am eager to have strengthened each of these skills --both technical and ‘soft’-- as I will no doubt draw on them regularly throughout my web development career.

Credits

Role: Lead Developer
Tutor: Andrew Muscara
Mentor: Alexis Gormley