A FULL STACK CASE STUDY

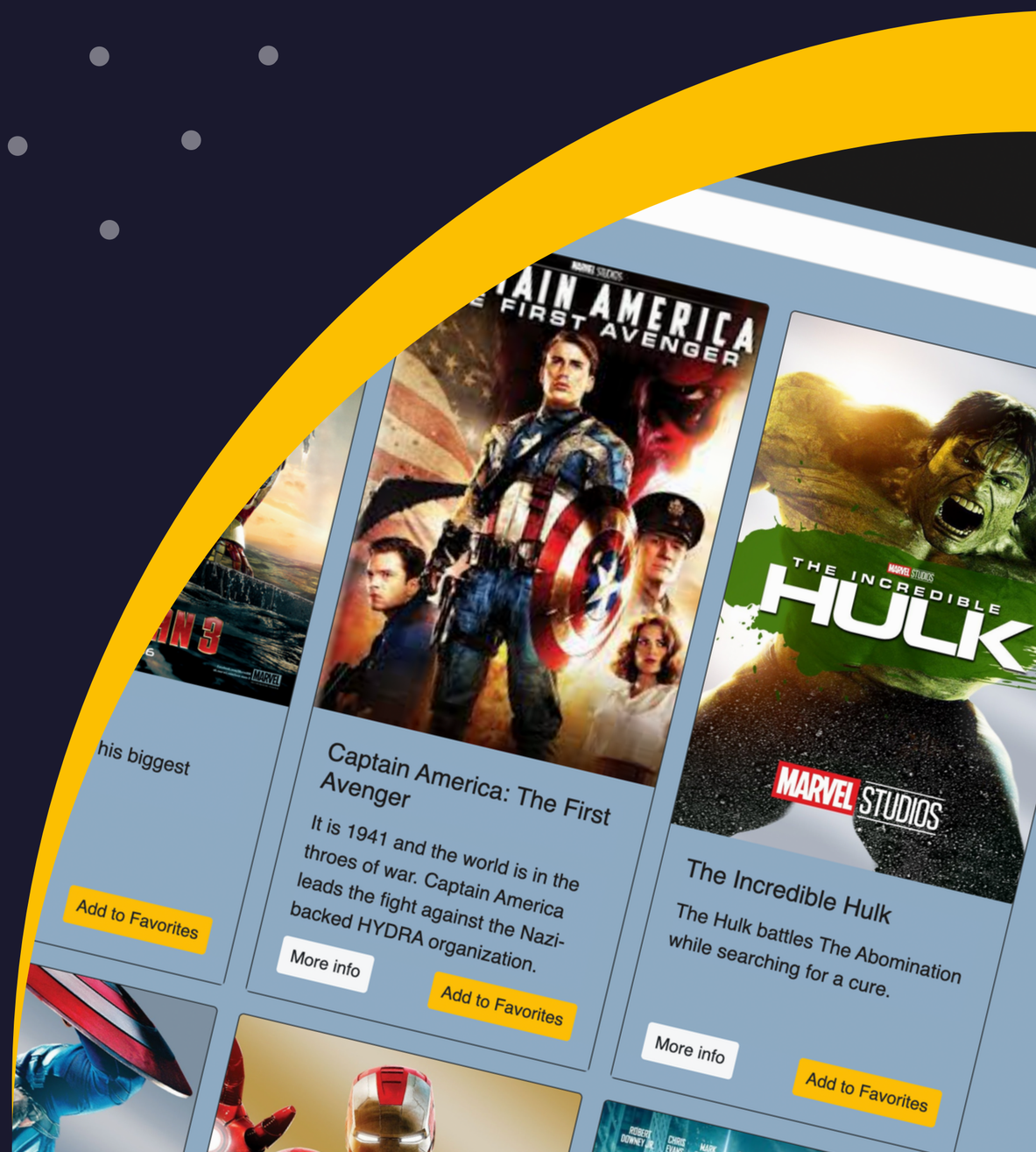
Katy Molony

Oveview

SUPER FLIX

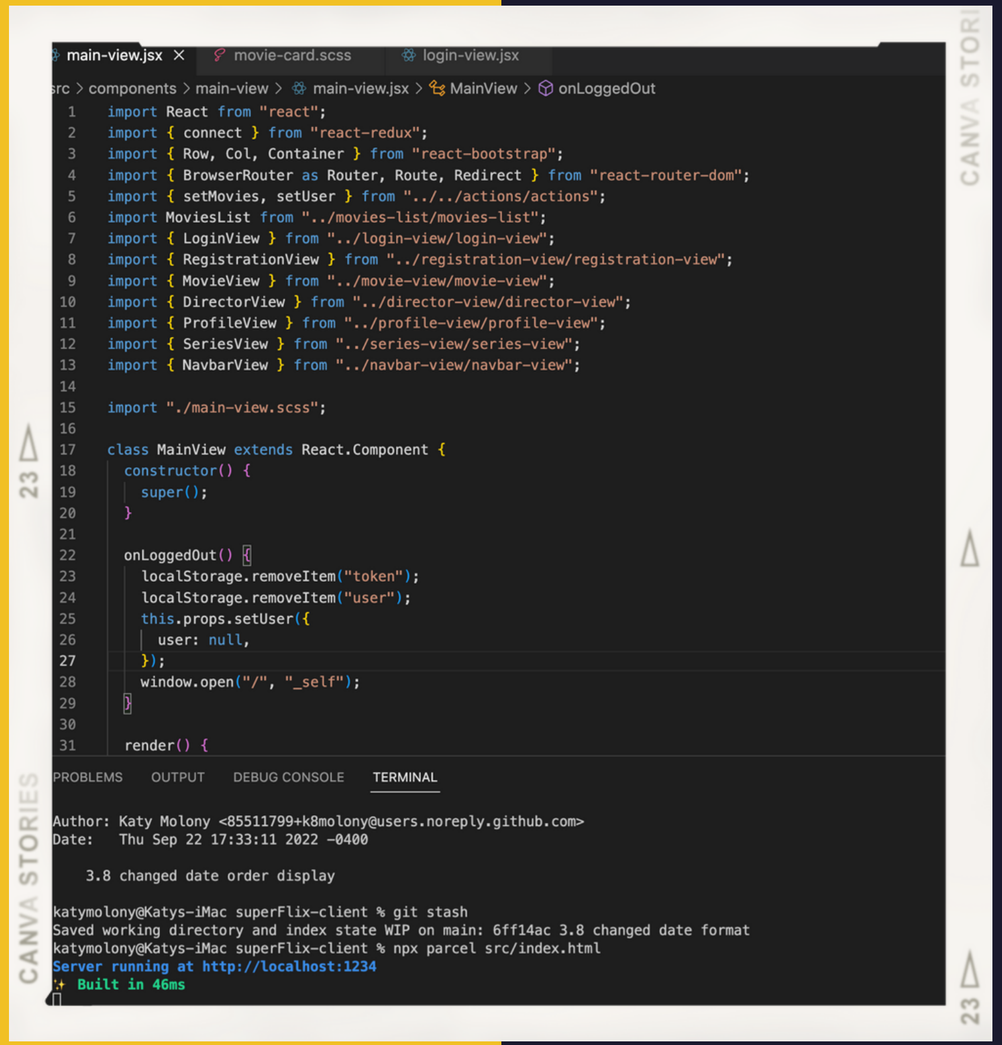
superFlix is a web application designed with the MERN stack. It provides users with access to information about superhero movies, including information about the film's directors as well as which comic universe they belong to. Users can create an account, update their personal information, and save a list of favorite movies.

SEE THE SITE



PURPOSE

This project was created as part of my web development course at CareerFoundry to demonstrate mastery of full stack JavaScript development.



OBJECTIVE

The aim of the project was to build a complete server side and client side web application from scratch to include in a professional portfolio.

Duration

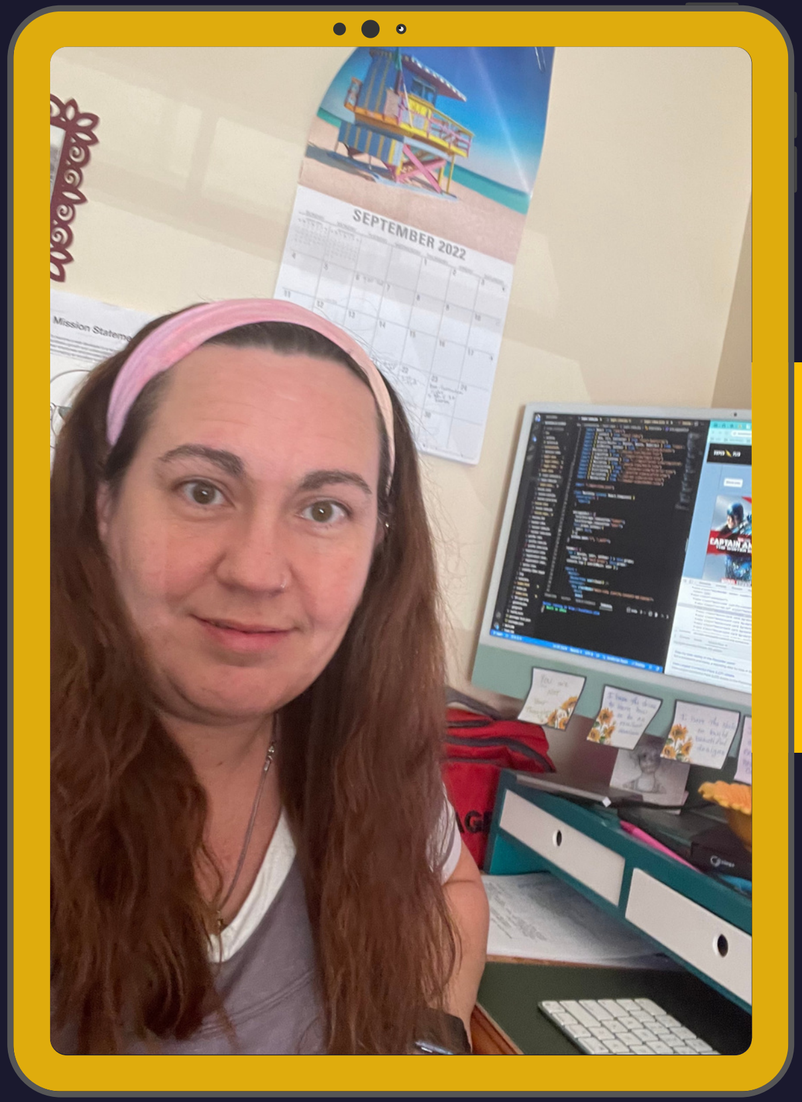
Creating the client-side took about twice as long as creating the server-side. I wanted to make sure I had a good grasp on how React and Redux work and achieve the desired result.

Credits

Lead Developer: Katy Molony
Tutor: Jay Quach
Mentors: Drew Mercer, Ted Walther

Methodoligies

- MERN stack
- Postman
- Heroku
- React Bootstrap
- Axios
- Redux



Github

SERVER-SIDE

I created a RESTful API using Node.js and Express, that interacts with a non-relational database (MongoDB). The API can be accessed via commonly used HTTP methods like GET or POST. To retrieve and store data in the database, CRUD methods are used. The API provides movie information in JSON format.

Basics

I first needed to determine if I wanted a relational or non-relational Database. After testing with PostGreSQL and MongoDB, I chose MongoDB because it is a non-relational database and offers more flexibility.

Business Logic

Next, I created models to keep my data consistently formatted and used Mongoose to interact with the database

Security

I chose basic HTTP for initial login paired with a JWT token based authorization to make my site secure. Then, I implemented CORS password hashing and data validation for extra security.

Deployment

After testing all endpoints using Postman, I finally used Heroku to deploy my app and MongoDB Atlas to host my database.

The Build

One goal of this project was to understand how to use MVC architecture as a design pattern. To facilitate this, I chose to use React because it is fast, easy to maintain, and well documented. Then, I used Parcel to complete the build operations.

Create Components

Once the initial build was done, I created components for the different views and hooks to control the state. I used Axios to pull in the API I had previously created.

Design

I used React Bootstrap to design the layout of the pages and cards and to ensure consistent styling. I also used client side app-routing to add authentication to access views.

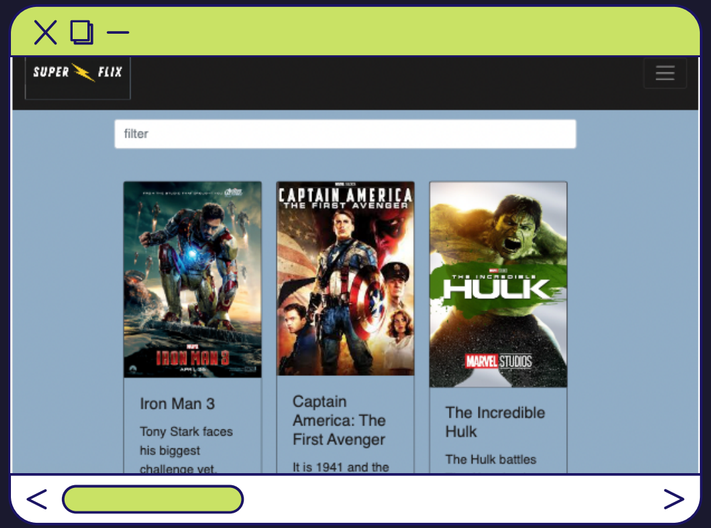
Final Product

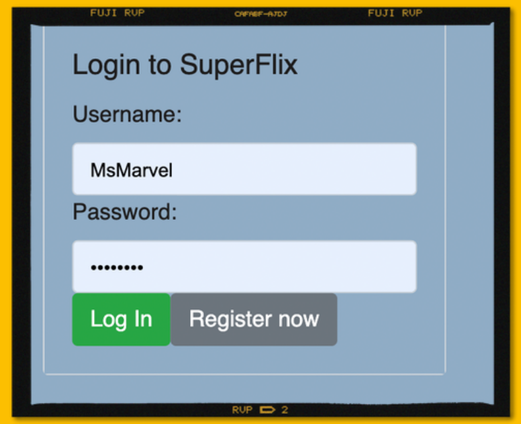
Finally, I added Redux to better manage the application's state and ensure that my app would be scalable. Once integrated, I hosted my completed project on Netlify.

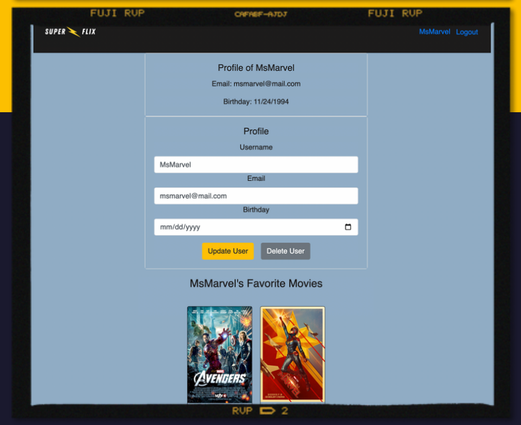
Github

CLIENT SIDE

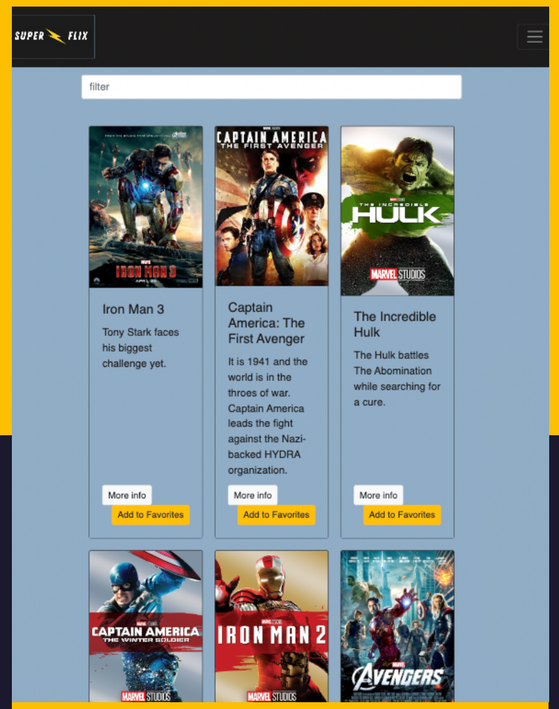
After completing the API, I built the interface users would need to interact with the server-side. It is a single-page, responsive application, developed with React and React-Redux. It provides several interface views including a movie view, a login view, a registration view and a profile view (where users can update their user data and list of favorites).



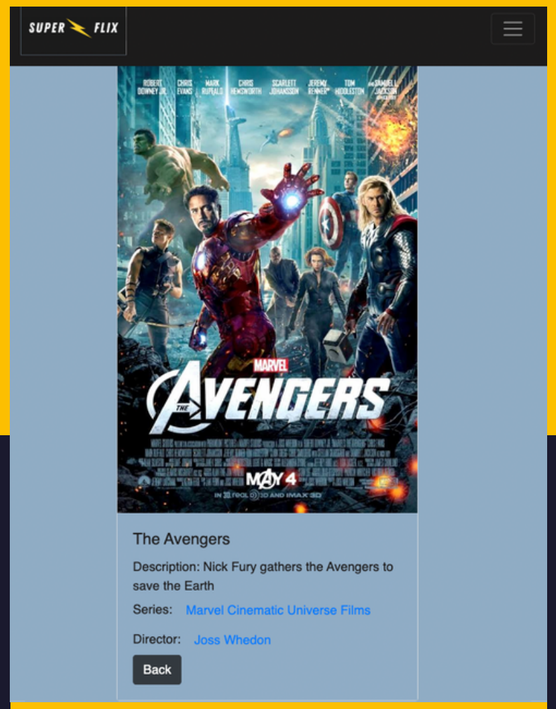




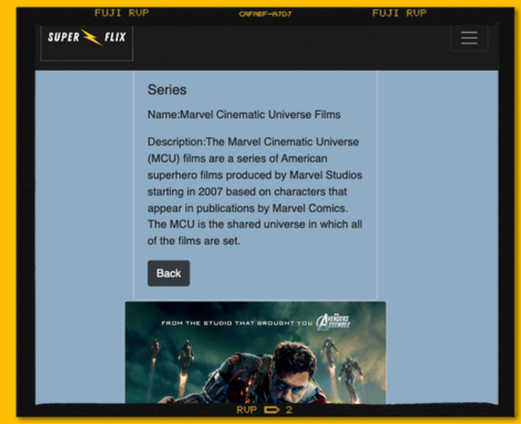
Login View
Profile View



Main View



Movie View



Series View
Director View

PAGE VIEWS

Users can login securely. They can access and update their profiles. On the Main View they can see all movies as well as search for particular movies. Clicking on **More Info** will give a more detailed description of the film. From there they can also get information about which superhero series the film belongs to as well as information about the director.

Retrospective

What went well?

I very much enjoyed building the database and setting up the backend. My analytical mind liked to see the order and structure of it, and I was able to fly through that build.

What didn't go well

The biggest challenge was time. I was juggling teaching full time while learning how to build this and struggled. Once school got out and I could devote myself to study, the pieces all fell into place and things went much smoother. Plus, by taking my time I was able to get a fuller understanding of React.

Future Steps

I would like to continue to build on the database to include more than just Marvel movies. I would also like to incorporate some sorting features to order the movies chronologically by setting and release dates.

Final Thoughts

Overall, I'm quite proud of the product I produced. Adding Redux was a bit of a challenge at first, but once I got the hang of it I really enjoyed working with it and feel like I have a deep understanding of the full process of building a website.