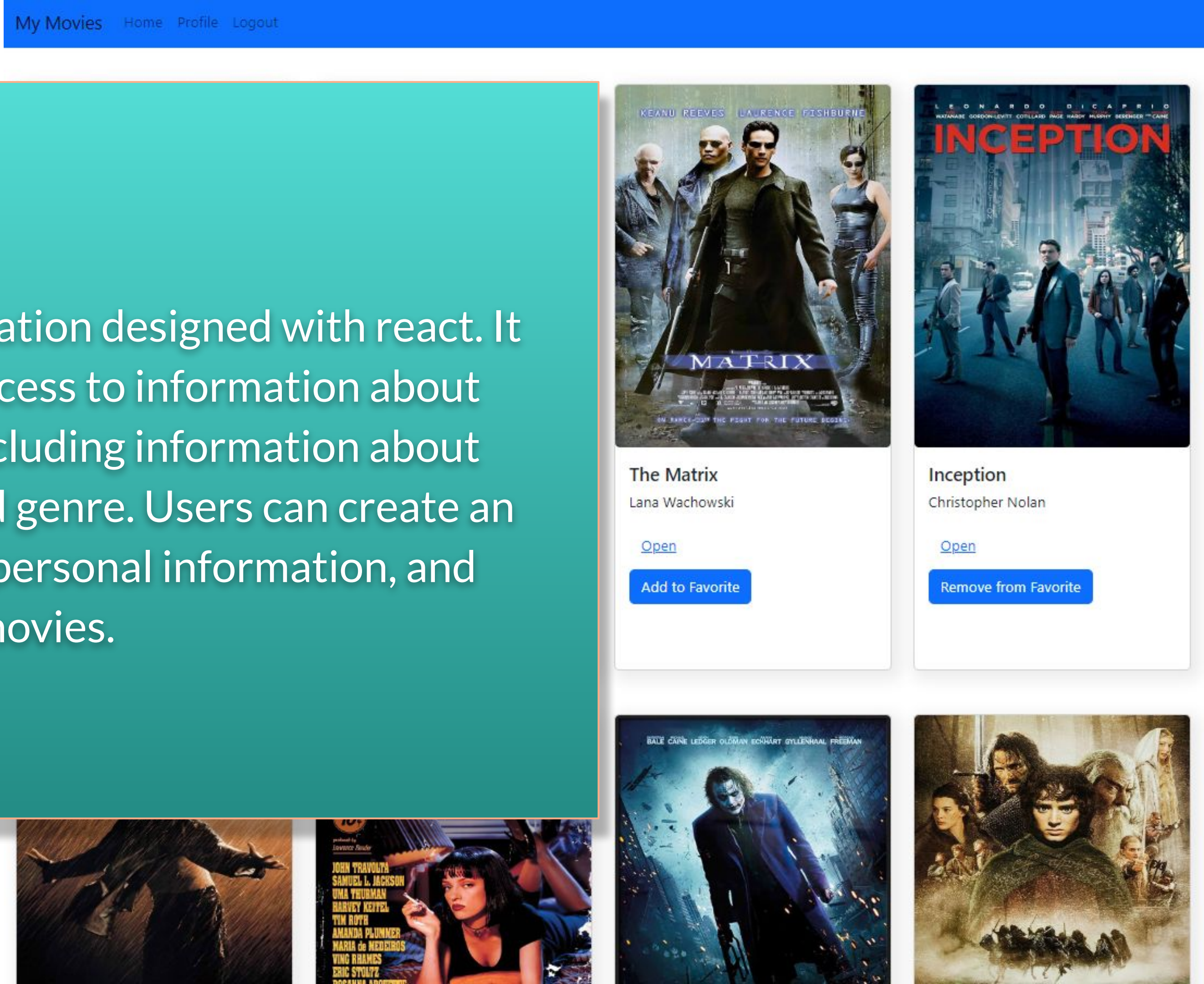
A FULL STACK CASE STUDY

My Flix App

Rebekah Hagen Willis

Overview

My Flix is a web application designed with react. It provides users with access to information about my favorite movies, including information about the film's directors and genre. Users can create an account, update their personal information, and save a list of favorite movies.



The Matrix
Lana Wachowski

Inception
Christopher Nolan

Purpose/Objective

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

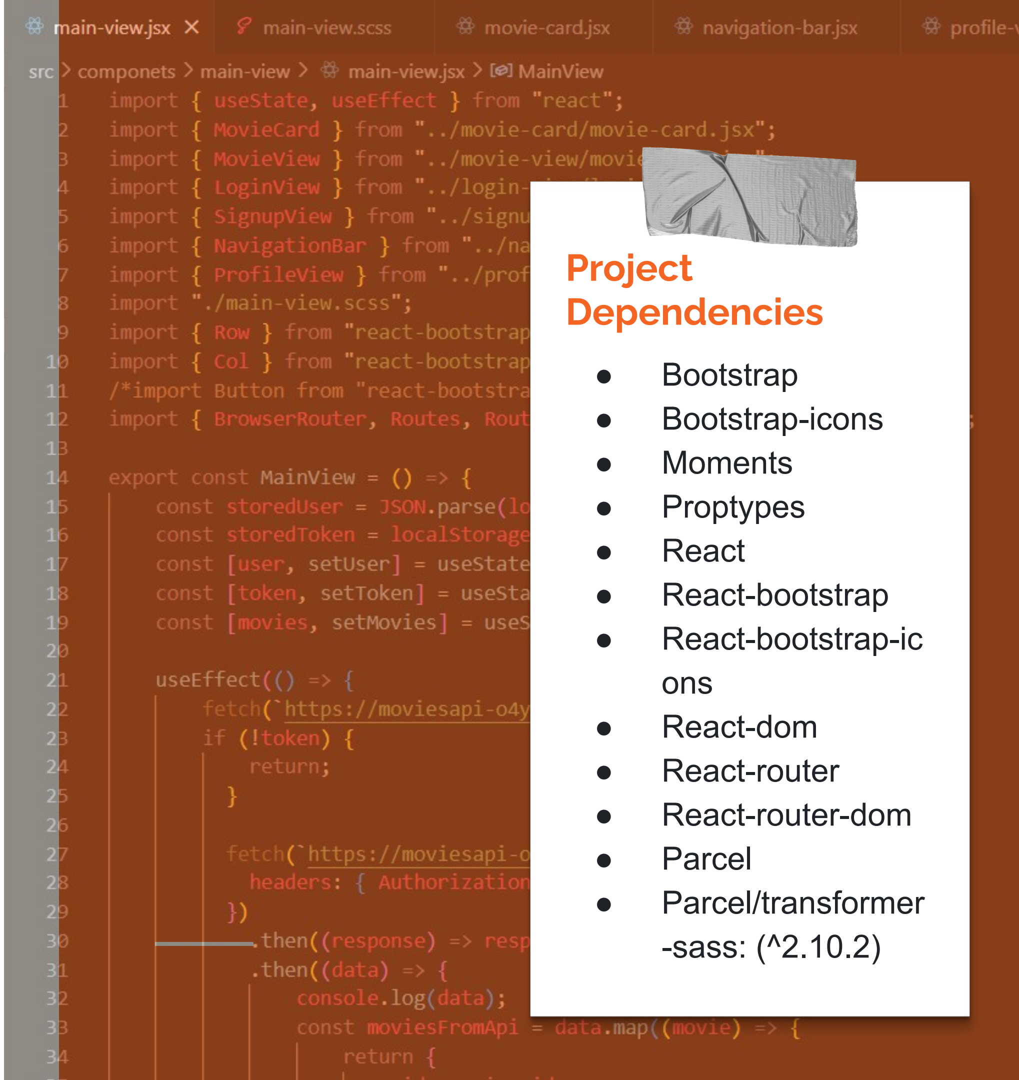
➔ **The aim of the project was to build a complete server side and client side web application from scratch to include in a professional portfolio.**

Duration

Creating the server-side took about twice as long as creating the client-side. I wanted to make sure I had a good grasp on back-end development. As far as the client-side, working with React wasn’t too different from JavaScript and went a lot faster.

Credits

Lead Developer: Rebekah Hagen Willis
Tutor: Christopher Abramson
Mentor: Alfredo Salazar Vélez



```
import { useState, useEffect } from "react";
import { MovieCard } from "../movie-card/movie-card.jsx";
import { MovieView } from "../movie-view/movie
import { LoginView } from "../login-
import { SignupView } from "../signu
import { NavigationBar } from "../na
import { ProfileView } from "../prof
import "./main-view.scss";
import { Row } from "react-bootstrap
import { Col } from "react-bootstrap
/*import Button from "react-bootstra
import { BrowserRouter, Routes, Rout

export const MainView = () => {
    const storedUser = JSON.parse(lo
    const storedToken = localStorage
    const [user, setUser] = useState
    const [token, setToken] = useSta
    const [movies, setMovies] = useS

    useEffect(() => {
        fetch(`https://moviesapi-o4y
        if (!token) {
            return;
          }

          fetch(`https://moviesapi-o
            headers: { Authorization
          })
            .then((response) => resp
            .then((data) => {
                console.log(data);
                const moviesFromApi = data.map((movie) => {
                    return {
```

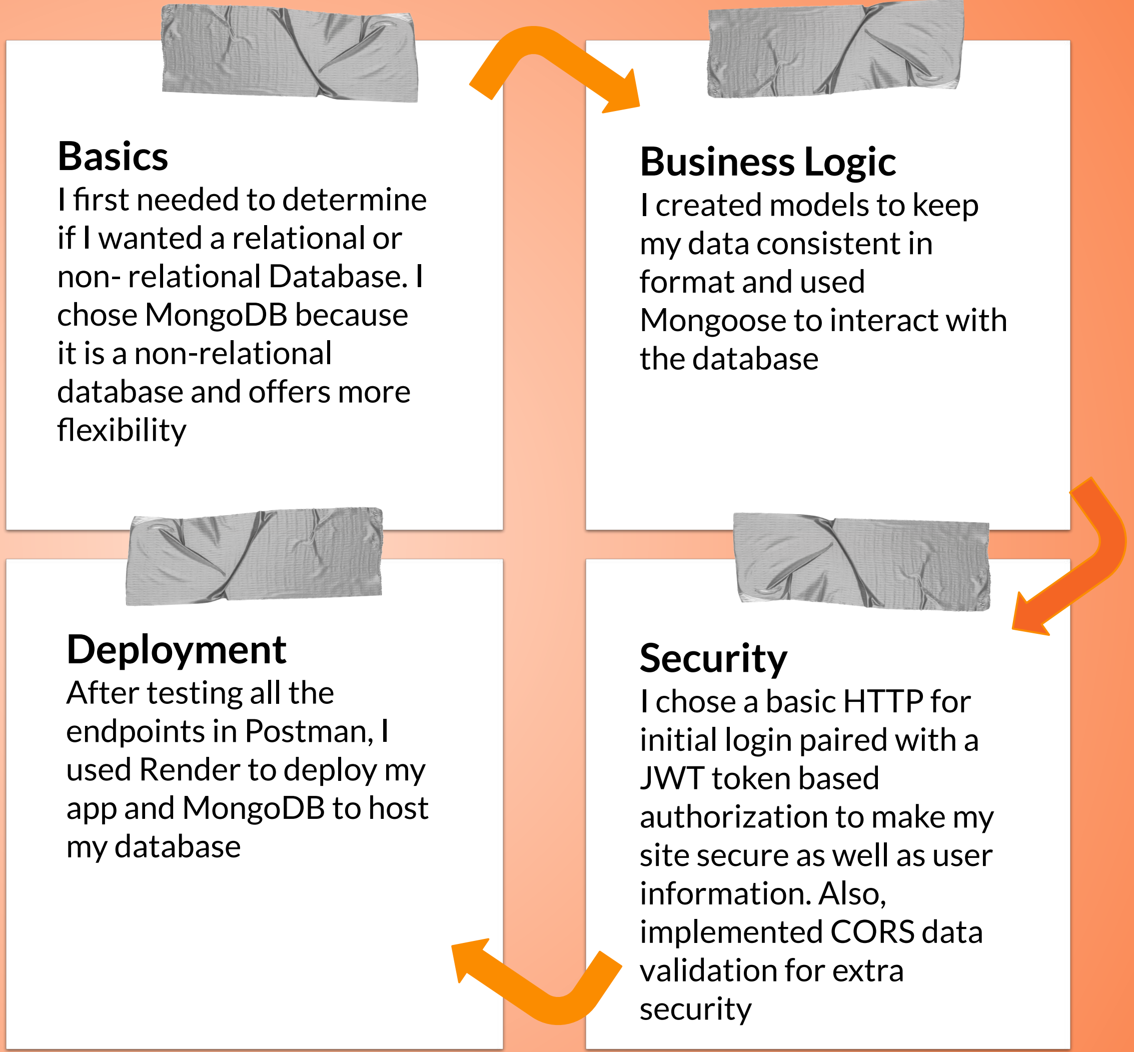
Project Dependencies

- Bootstrap
- Bootstrap-icons
- Moments
- Proptypes
- React
- React-bootstrap
- React-bootstrap-icons
- React-dom
- React-router
- React-router-dom
- Parcel
- Parcel/transformer-sass: (^2.10.2)

Server-Side

Github

I created a RESTful API using Node.js and Express, that interacts with a non-relational database (MongoDB). The API can be accessed via HTTP methods like GET, POST, PUT, and DELETE. 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. I chose MongoDB because it is a non-relational database and offers more flexibility

Business Logic
I created models to keep my data consistent in format and used Mongoose to interact with the database

Security
I chose a basic HTTP for initial login paired with a JWT token based authorization to make my site secure as well as user information. Also, implemented CORS data validation for extra security

Deployment
After testing all the endpoints in Postman, I used Render to deploy my app and MongoDB to host my database

Client-Side

Github

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).

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.

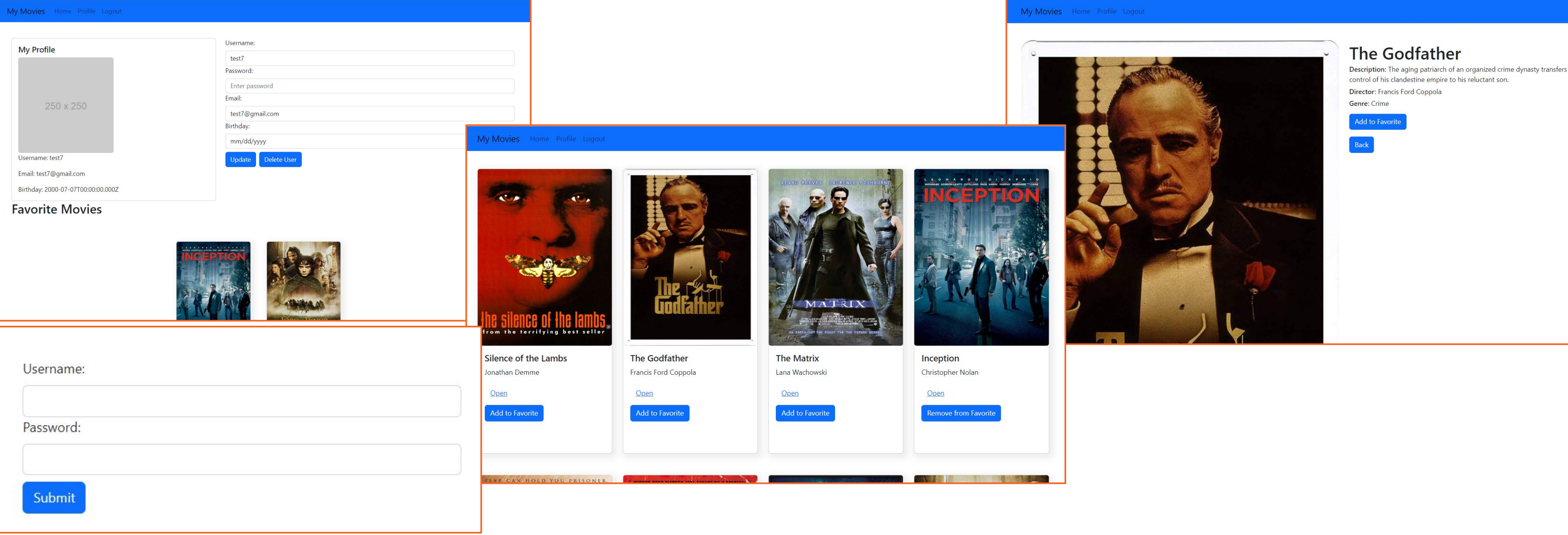
Final Product

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

Design

I used React Bootstrap to design the layout of the pages and cards and to ensure consistent styling.

**Will add more design elements later on in course



My Profile

Username: test7

Email: test7@gmail.com

Birthday: 2000-07-07T00:00:00.000Z

Favorite Movies

Username: test7

Password: Enter password

Email: test7@gmail.com

Birthday: mm/dd/yyyy

Silence of the Lambs
Jonathan Demme

The Godfather
Francis Ford Coppola

The Matrix
Lana Wachowski

Inception
Christopher Nolan

The Godfather

Description: The aging patriarch of an organized crime dynasty transfers control of his clandestine empire to his reluctant son.

Director: Francis Ford Coppola

Genre: Crime

Username:

Password:

Page Views

Users can login securely. They can access and update their profiles. On the Main View they can see all movies available. Clicking on **Open** will give a more detailed description of the film. From there they can also get information on the genre as well as information about the director.

Reflection

What went well?

I very much enjoyed building the structure and setting up the frontend. I am a visual learner and I love to design. I was able to fly through that build.

What didn't go well?

The biggest challenge was time. I was juggling teaching full time with a one year old while learning how to build this and struggled.Once I had a good schedule in place, I was able to get a fuller understanding of the Node.js and React.

Future Steps

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