# SpookyVibes Case Study

A Full-Stack Web Development Project

# Overview

| | |
|---|---|
| **Objective:** | Create a complete web app, including both the server-side and client-side, from scratch that provides users with details about different movies, including the title, genre, description and director.<br><br>Users should be able to sign up for the site, log in, view their account details and add/remove movies from a list of favorites. |
| **Background:** | This project was part of my Full-Stack Immersion course with CareerFoundry. It introduced and familiarized me with the MERN stack (MongoDB, Express, React and Node.js). |
| **My Role:** | Lead Developer |
| **Technologies Used:** | • **HTML** – a markup language for creating web pages<br>• **CSS** – specifies how documents are laid out, styled, etc.<br>• **JavaScript** – a programming language used to create interactive effects within web browsers<br>• **Bootstrap** – a CSS framework with design templates for typography, forms, buttons, navigation and other interface components<br>• **React** – a front-end JavaScript library for building user interfaces<br>• **Node.js** – JavaScript runtime environment for server-side scripting<br>• **Express** – a back-end web application framework for building RESTful APIs with Node.js<br>• **MongoDB with Mongoose** – NoSQL database and Object Data Modeling library for Node.js<br>• **Postman** – a platform used to design, develop, test and monitor APIs |

# The Requirements

The project brief for the movie app included several required features:

- Return a list of ALL movies to the user

- Return data (description, genre, director, image URL) about a single movie by title to the user

- Return data about a genre (description) by name/title (e.g., "Thriller")

- Return data about a director (bio, birth year, death year) by name

- Allow new users to register

- Allow users to update their user info

- Allow users to add a movie to their list of favorites

- Allow users to remove a movie from their list of favorites

- Allow existing users to deregister

# The Process

## Sever-Side

Building SpookyVibes began with building the server-side of the application, which is the "backend" code that deals with data management, business logic and server configuration.

Movie and user data is stored in a non-relational (NoSQL) database on MongoDB with business logic modeled with Mongoose. Access to the data is done via a REST API, which uses HTTP requests to receive, send, create, and delete data or files. The API is built with a combination of Node.js and Express.
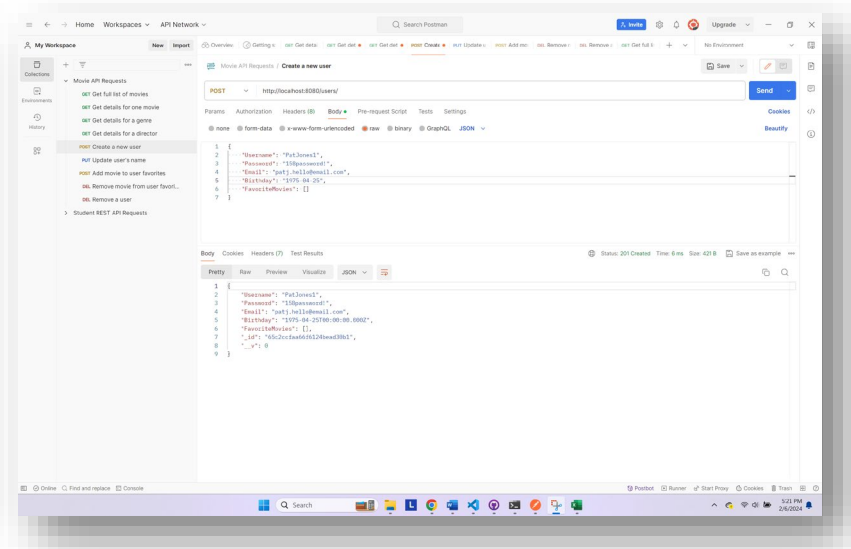
Endpoints were created to route the HTTP requests and fulfill the app requirements.

Review the endpoints here.

Postman was then used to test the endpoints and ensure that data was returned in JSON format.
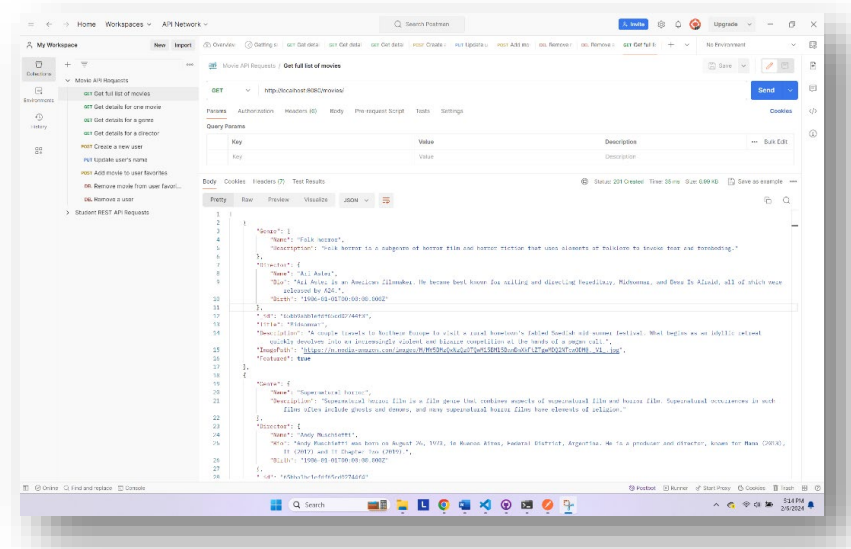
Click to view larger image

Click to view larger image

# The Process

## Client-Side

Once the backend of SpookyVibes was built and tested, it was time to move onto the client-side of the app, which uses the React library to create the interface used by visitors when making requests to – and receiving responses from – the server-side.

The single-page application (SPA) uses the React library to render several components for the user, each created based on the app requirements.

The components include a movie card (with the image, title and description), movie details (with genre and director included), forms for logging in and registering, and more.

The UI library Bootstrap is utilized to standardize the look and feel of the app and ensure its responsiveness.
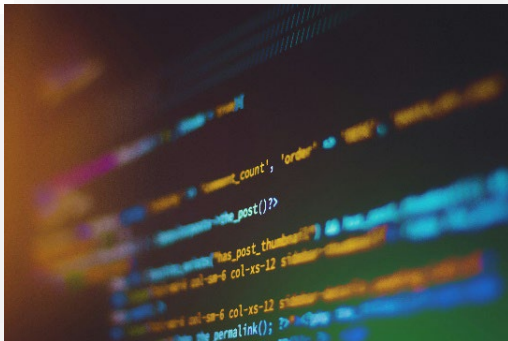
```jsx
import { useParams } from "react-router";
import { Link } from "react-router-dom";
import "./movie-view.scss";

export const MovieView = ({movies}) => {
    const { movieId } = useParams();

    const movie = movies.find((m) => m.id === movieId);

    return (
        <div>
            <div>
                <img height={500} alt={`${movie.title} poster`} src={movie.image} />
            </div>
            <div>
                <span>Title: </span>
                <span>{movie.title}</span>
            </div>
            <div>
                <span>Description: </span>
                <span>{movie.description}</span>
            </div>
            <div>
                <span>Director: </span>
                <span>{movie.director}</span>
            </div>
            <div>
                <span>Genre: </span>
                <span>{movie.genre}</span>
            </div>
            <Link to={`/`}>
                <button className="back-button" style={{cursor:"pointer"}}>
                    Back
                </button>
            </Link>
        </div>
    );
};
```

Example of component code for MovieView

# SpookyVibes Lives!

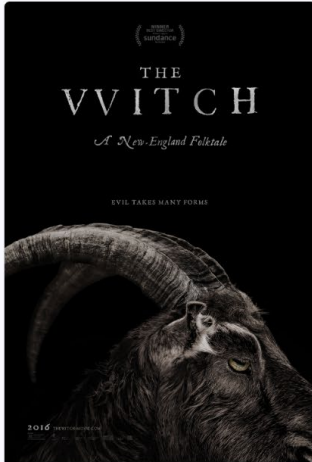After all that hard work, SpookyVibes was finally live, hosted on Netlify, and ready to use!

# Highs & Lows

## Looking Back

The server-side part of this project went smoothly and I really enjoyed learning how to build a database and getting familiar with the terminal, which had always intimidated me. I found that the concepts of building a database came pretty naturally to me and it's something I'd love to explore more in the future.

The client-side, on the other hand, was my biggest challenge to date in the Full-Stack Immersion course. I applied a coding-from-scratch approach and utilized my resources, including my CareerFoundry peers, my tutor and tools like ChatGPT to assist me along the way.

This project taught me to be more resourceful in looking for answers and fixes. It also taught me how to better read code in order to spot the errors (finding the broken code is the first step to remedying it!).

It was the biggest learning curve but the end result was the most satisfying!

# Future Steps

Moving forward, I'd love to add a search/filter feature to help users find the movies they want as well as a way for users to add new movies to the database if the one they're searching for isn't already available.

# Related Links

**Server-Side Code**  **Client-Side Code**  **Live App**