Full Stack Case Study

My Flix

User Problem

Many movie enthusiasts struggle to keep track of their favorite films and access detailed information about them in one convenient location. My Flix addresses this issue by providing a seamless interface where users can discover, organize, and save their preferred movies while accessing in-depth details about each film. By integrating user accounts, favorites lists, and an intuitive browsing system, My Flix enhances the movie discovery experience, making it more personalized and engaging.

Giuseppe Mancini

Overview

My Flix is a web application built on the powerful MERN (MongoDB, Express, React, Node.js) stack, which leverages JavaScript across both backend and frontend development. The application enables users to explore curated details about top-rated movies—including information on directors, genres, and casts. In addition to browsing content, users can create an account, update their personal details, and compile a personalized list of their favorite films.

My Flix site

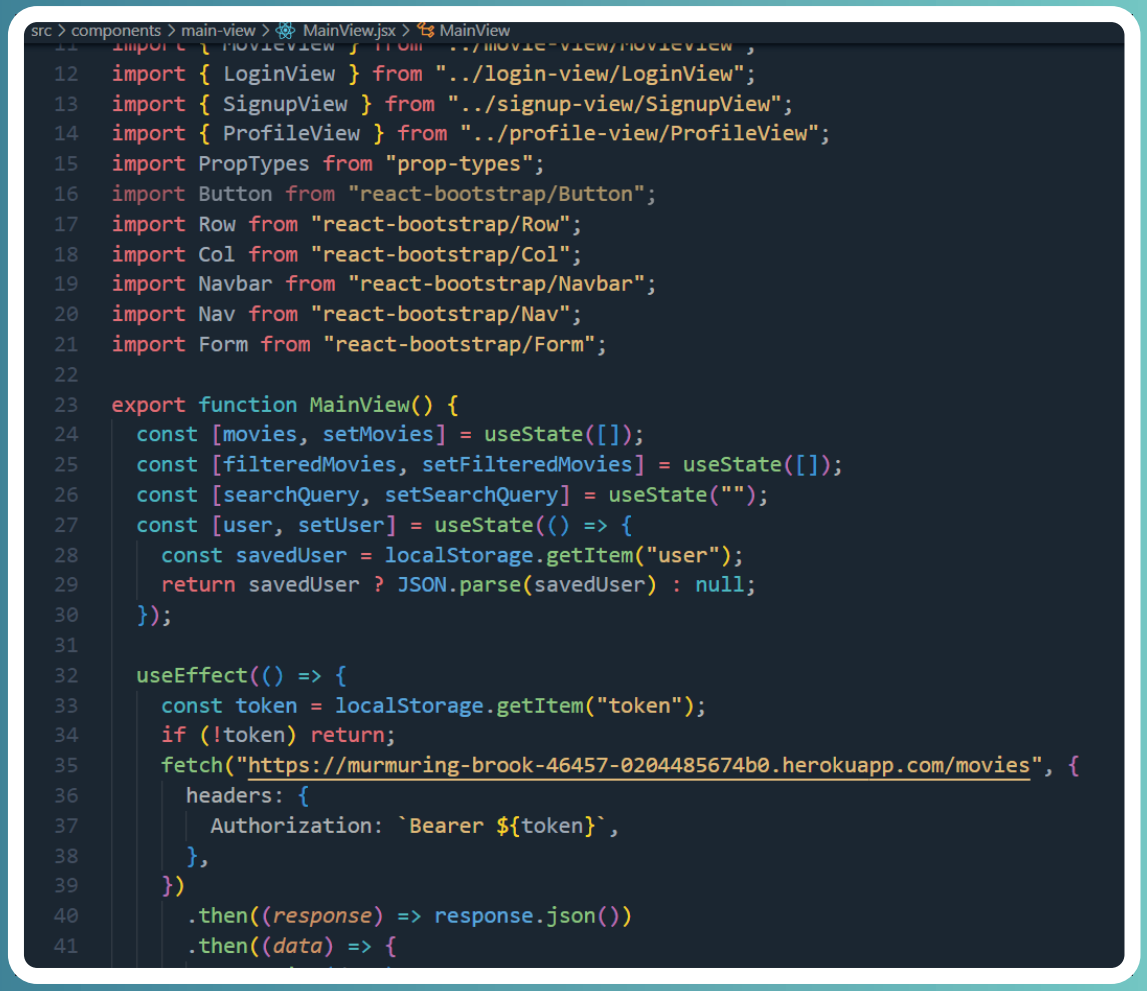
The Matrix

A computer hacker learns about the true nature of reality and his role in the war against its controllers.

Open | Add to Favorites | Delete

Purpose

This application was developed as part of the Full Stack Web Development course at CareerFoundry. Under the expert guidance of a Senior Developer Mentor for overall strategy and a dedicated Tutor for in-depth technical advice, the project was designed to encapsulate best practices and advanced methodologies in modern web development.



My Flix Movie Application

Shutter Island

A U.S. Marshal investigates the disappearance of a patient from a mental institution, uncovering dark secrets.

Genre: Stories that build suspense and intrigue.

Director: Martin Scorsese

Actors:

Back | Remove from Favorites

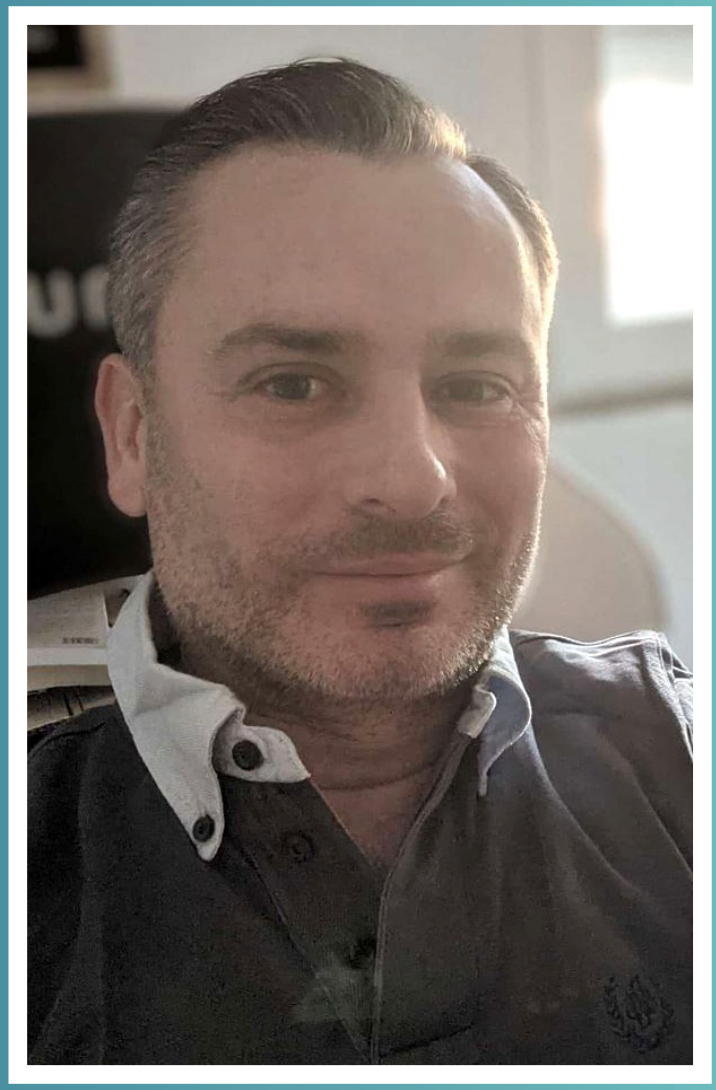
Objective

The primary goal was to deepen mastery over web technologies by crafting a complete application—from building a robust backend infrastructure with efficient database interactions to developing an engaging and responsive front-end interface and successfully deploying the final product online.

Project Duration

The overall project spanned approximately 6 weeks, with time allocated as follows:

- **Planning & Research:** 1 weeks (database selection, defining API structure, outlining UI/UX)
- **Backend Development:** 2 weeks (API development, authentication implementation, database integration)
- **Frontend Development:** 2 weeks (React UI design, state management, API integration, styling
- **Deployment & Testing:** 1 week (debugging, Postman testing, final adjustments, Heroku deployment)



Methodologies and Tools

- Node.js
- React
- MongoDB
- React Bootstrap
-Postman
- Heroku

Credits

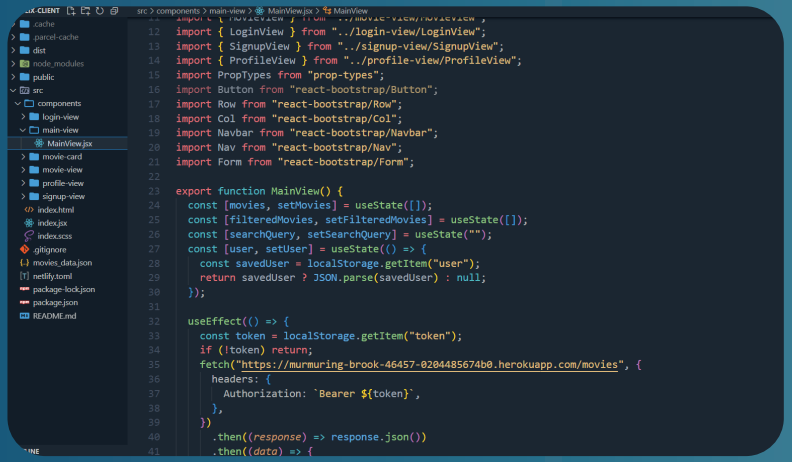
Lead Developer: Giuseppe Mancini
Tutor: Stanley Okwii
Mentor: Lucien Chemaly

Sever-side Development

For the backend, I developed a RESTful API using the Express framework on Node.js—leveraging MongoDB as the database for effective data management. The API supports standard HTTP methods (GET, POST, UPDATE, DELETE), enabling comprehensive CRUD operations while providing responses in JSON format.

The first key decision was selecting the appropriate database. Extensive research reaffirmed that MongoDB was an ideal choice, particularly favored by the JavaScript and Node.js communities. To ensure data consistency and streamline processing, I integrated the Mongoose framework, which allowed me to structure and manage data models efficiently.

GitHub Repo



Security was paramount; thus, I implemented JWT (JSON Web Tokens) for authentication and authorization, and enabled CORS to safeguard data privacy. Following development, each endpoint was rigorously tested with Postman, and once stability was confirmed, the API was successfully deployed on Heroku.

Client-side Development

To provide users with a smooth and engaging experience, the next step was to build an intuitive graphical user interface that seamlessly interacts with the API. This was achieved using React, enabling the creation of a Single-Page Application (SPA) that is fluid, modular, and high-performing.

Styling with Bootstrap

For a polished and responsive design, Bootstrap was utilized to style the components, ensuring consistency and flexibility across various devices. Its pre-built components and grid system helped streamline the development process while maintaining a professional aesthetic.

Dynamic Interaction with the Backend

The frontend is composed of multiple interfaces that dynamically interact with the backend. Users can easily register and log in, manage their profiles, view detailed movie information, compile a favorites list, and search for specific films—all within a seamless and engaging experience.

Building with Parcel

To efficiently bundle and optimize the application, Parcel was used as the build tool. Its zero-configuration approach simplified asset management and improved performance, making the development process smoother and faster.

Deployment on Netlify

For hassle-free hosting and continuous deployment, the application was deployed using Netlify. This provided fast and reliable delivery of the frontend, ensuring that users could access the platform anytime without issues.

Accomplishments

I gained a comprehensive understanding of the multifaceted elements that form a complete application—from the foundational planning and design phases to the technical nuances of assembling each component. This project has significantly bolstered my confidence and competence as a Full Stack Developer.

Challenges

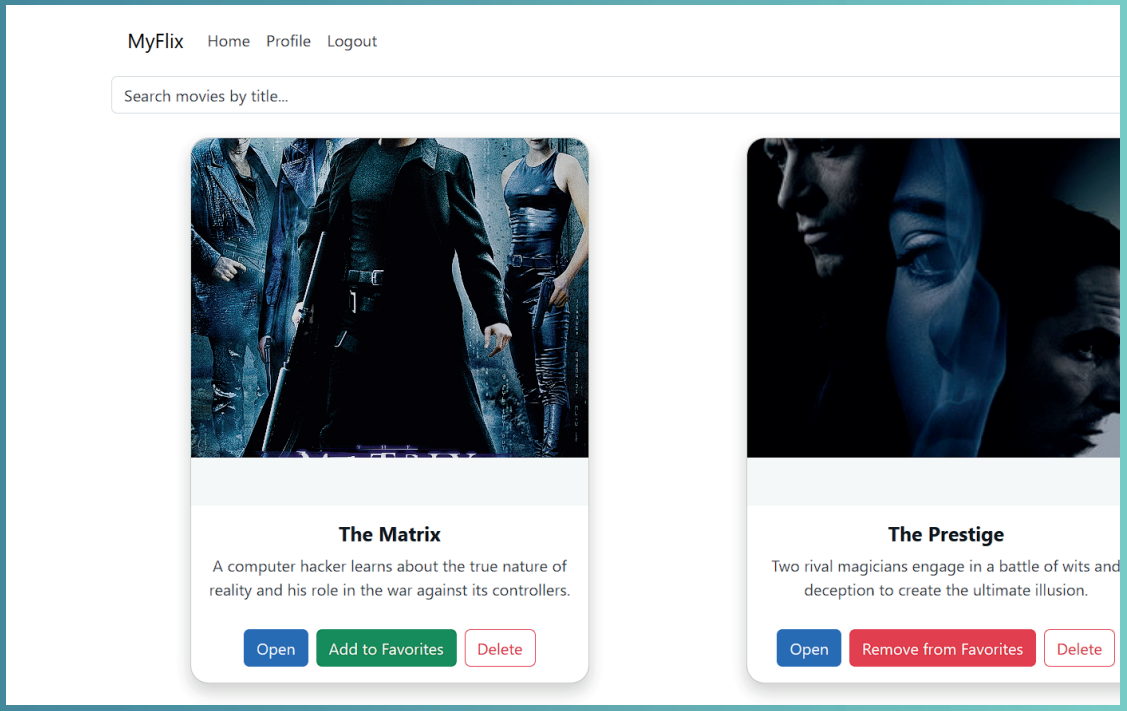
Implementing user authentication and authorization was particularly challenging, taking more time than initially anticipated. Frequent consultation of documentation was essential to correctly integrate these critical features. However, I am optimistic that future projects will benefit from this experience, leading to more efficient development cycles.

```
162    return (
163      <Router>
164        <Navbar expand="lg" className="p-3">
165          <Navbar.Brand as={Link} to="/">
166            MyFlix
167          </Navbar.Brand>
168          <Navbar.Toggle aria-controls="basic-navb
169          <Navbar.Collapse id="basic-navbar-nav">
170            <Nav className="ml-auto">
171              {user ? (
172                <>
173                  <Nav.Link as={Link} to="/">
174                    Home
175                  </Nav.Link>
176                  <Nav.Link as={Link} to="/profile
177                    Profile
178                  </Nav.Link>
179                  <Nav.Link onClick={handleLogout}
180                </>
181              ) : (
182                <>
183                  <Nav.Link as={Link} to="/login">
184                    Login
185                  </Nav.Link>
186                  <Nav.Link as={Link} to="/signup"
187                    Signup
188                  </Nav.Link>
189                </>
190              )}
191            </Nav>
192          </Navbar.Collapse>
```

Future Steps

Looking ahead, I plan to:

- Refine the overall layout and user interface of the application.
- Enhance the quality and availability of the data stored in the database.
- Implement graphical visualizations to represent movie genres and categories more dynamically.



Final Considerations

Designing and developing the My Flix App was a complex journey that demanded perseverance, effective stress management, and a deep commitment to learning. Despite the challenges, the project provided invaluable insights into every phase of application development—from initial planning to final deployment—while equipping me with the technical skills necessary to build modern, seamless, and high-performance applications.