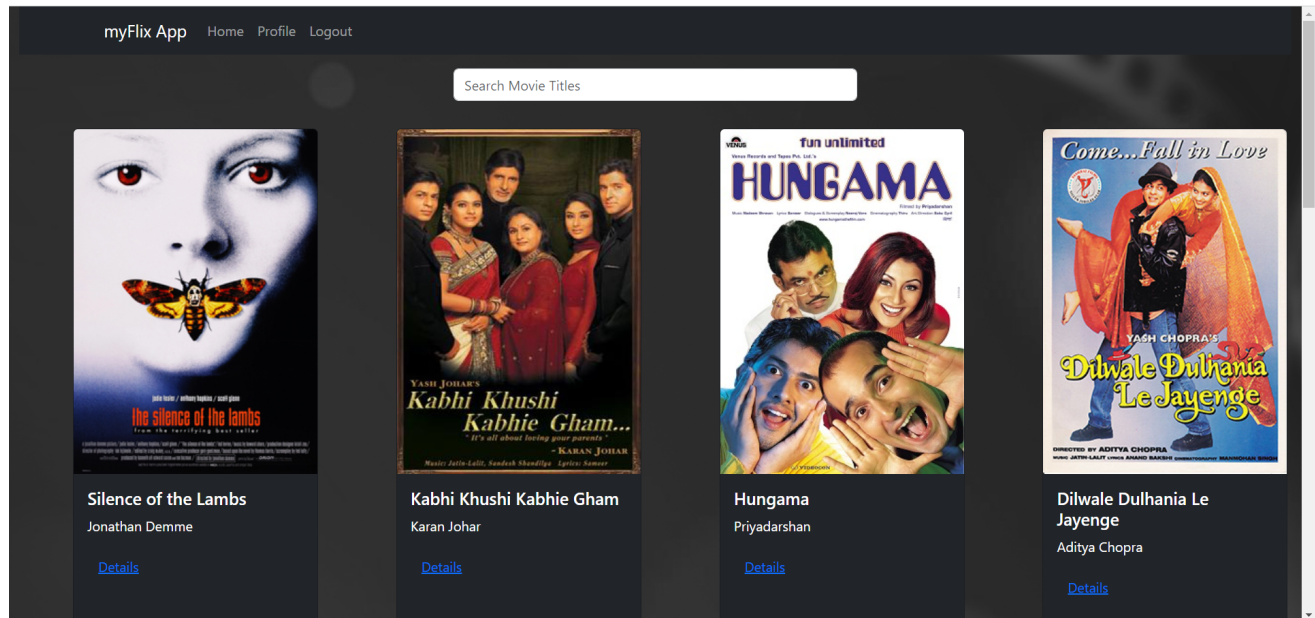


Case Study: myFlix - A Full Stack JavaScript Project



Overview

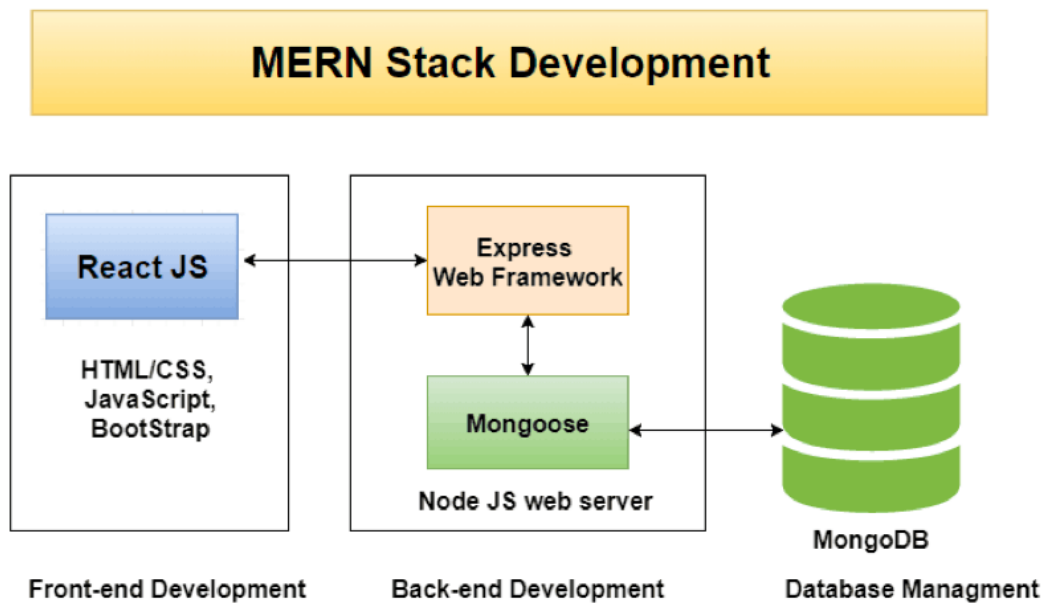
myFlix is a web-application developed using MERN technology stack. The project includes an interactive movie search and other features such as allowing users to access movie list and viewing movie details including genres and directors. Users can create an account and update and view their personal data as well as can add movies to their favorite list.

Purpose

myFlix project is a learning exercise of my CareerFoundry Full-Stack Immersion course that aims to develop proficiency in full-stack JavaScript development using the MERN (MongoDB, Express.js, React.js, Node.js) technology stack.

Objectives

The objective of this project is to gain hands-on experience in full-stack JavaScript development, focusing on creating a user-friendly movie search interface. It aims to implement secure access through HTTP authentication and JWT tokens, allowing users to add movies to their favorites and view movie details. The project was tested using Postman for functionality and reliability, while adhering to Coors principles for secure browsing.



Technologies Used

Front-end: React, Bootstrap

Back-end: Node.js, Express.js

Database: MongoDB

Authentication: HTTP authentication, JWT tokens

Duration

Developing the client side took longer compared to backend API because I wanted to understand the concepts of React such as retrieving data from API, creating interactive components and implementing secure storage and access of data in a web app.

Credits

My Role: Lead Developer

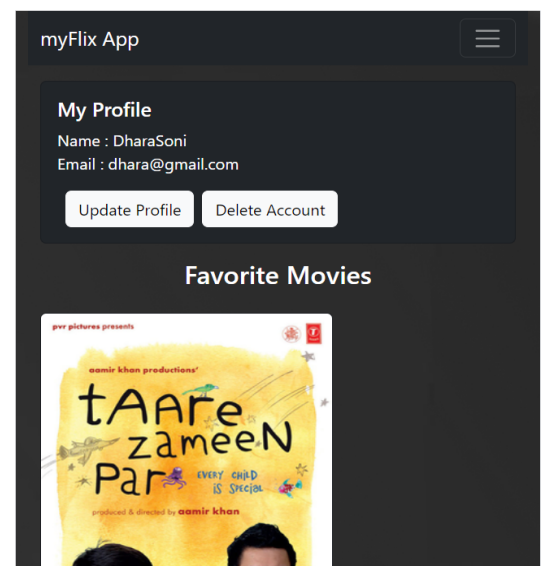
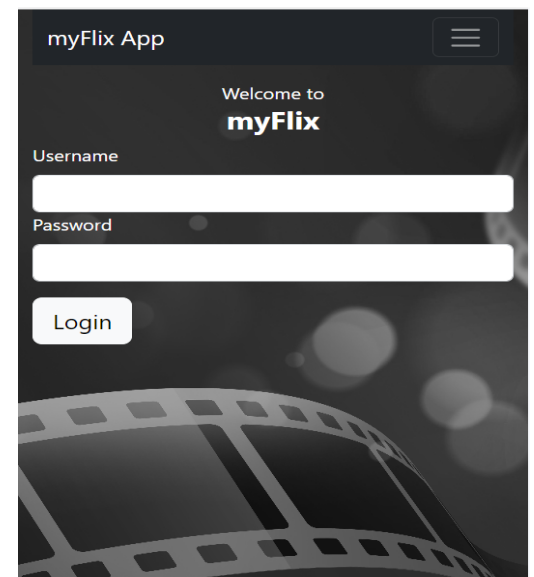
Tutor: Timothy Nyabongo

Mentor: Stephen Barungi

Implementation Details

I. Frontend setup

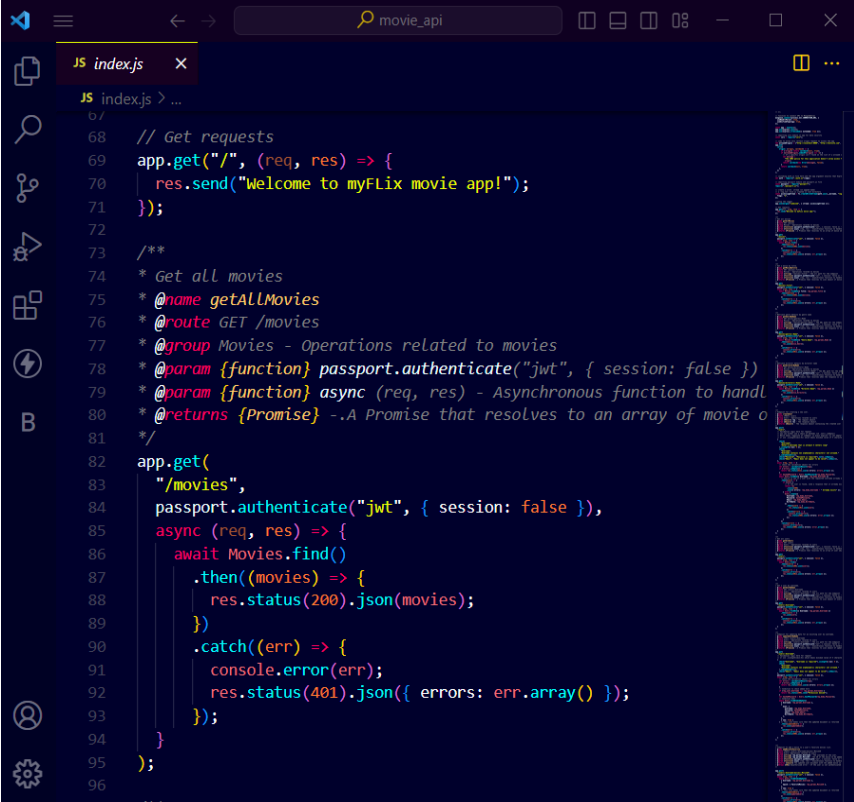
The React.js application is set up with a responsive and user-friendly interface. It provides an intuitive design that allows users to search for movies easily. The UI includes components such as a movie search bar, movie details page, user login/register functionality, adding movies to favorites and user profile page.



II. Backend setup

The Node.js and Express.js server is set up with a RESTful API that interacts with the MongoDB database. Routes are defined for fetching movie data from an external API, storing movies and users details in the database, adding movies to favorites, and managing user accounts.

[View Endpoints](#)



```
68 // Get requests
69 app.get("/", (req, res) => {
70   res.send("Welcome to myFLix movie app!");
71 });
72
73 /**
74  * Get all movies
75  * @name getAllMovies
76  * @route GET /movies
77  * @group Movies - Operations related to movies
78  * @param {function} passport.authenticate("jwt", { session: false })
79  * @param {function} async (req, res) - Asynchronous function to handle
80  * @returns {Promise} - A Promise that resolves to an array of movie objects
81  */
82 app.get(
83   "/movies",
84   passport.authenticate("jwt", { session: false }),
85   async (req, res) => {
86     await Movies.find()
87       .then((movies) => {
88         res.status(200).json(movies);
89       })
90       .catch((err) => {
91         console.error(err);
92         res.status(401).json({ errors: err.array() });
93       });
94   }
95 );
96
```

III. Database Management

MongoDB is used as the database management system for myFlix. It stores movie information such as title, genre, director etc. The database is accessed through the backend API endpoints to retrieve relevant movie data based on user queries. Additionally, the database stores user information and their favorite movies.

IV. Authentication and Security

To ensure secure access to myFlix, HTTP authentication has been implemented for user login. Users are required to provide valid login credentials to access the application. Additionally, JWT tokens are used for authorized access, allowing users to securely interact with the application and perform actions such as adding movies to their favorite list.

V. Error handling and debugging

During the development process, error handling and debugging techniques were employed to identify and rectify any issues that arose.

Challenges

Throughout the project, I faced challenges in implementing secure authentication and authorization mechanisms, debugging API requests, JWT tokens, and database operations, as well as optimizing the frontend UI for performance and user experience. However, through diligent effort and collaboration with my tutor and mentor, I successfully overcame these challenges to achieve the project goals.



Lessons Learned

This project taught me the importance of understanding and utilizing the full MERN stack for development. I realized that having a comprehensive knowledge of MongoDB, Express.js, React.js, and Node.js is crucial for building robust web applications. Additionally, I learned the significance of implementing secure and efficient user authentication and authorization mechanisms to protect user data. Furthermore, I recognized the importance of well-structured code, efficient API requests and optimized user interface performance. These lessons have provided me with valuable insights that will guide my future projects towards success.





Conclusion

myFlix project successfully demonstrated proficiency in full-stack JavaScript development using the MERN technology stack. Throughout the development process, valuable lessons were learned and improvements were made to enhance the project's functionality as well as overall user experience. myFlix project serves as a solid foundation for my future learning and development in full-stack JavaScript programming.