CAREERFOUNDRY

Achievement 3 Project Brief: React App (myFlix)

Objective

Using React, build the client-side for an application called myFlix based on its existing server-side code (REST API and database).

Context

Client-side development hasn't always been so prominent. In the past, pages would be generated on the server-side and sent to the browser, resulting in a poor user experience. Thanks to modern browsers and libraries such as React, the client-side of an application is today considered to be just as important as the server-side. As a student of full-stack development, you need to be skilled in both the server-side and client-side.

In the previous Achievement, you built the server-side for a movie application called myFlix. The API and database that you built meet the information needs of myFlix users. Now, you need to create the interface they will use when making requests to, and receiving responses from, the server-side. The client-side of your myFlix application will include several interface views built using the React library that will handle data through the previously-defined REST API endpoints.

The code you write impacts both your users and your fellow developers. As you work through this Achievement, you'll need to consider, among other things, the readability and maintenance of your codebase, and the design and usability of your application.

By the end of the Achievement, you'll have a complete web application (client-side and server-side) built using full-stack JavaScript technologies, which you can showcase in your portfolio. This project will demonstrate your mastery of full-stack JavaScript development. The complete tech stack you'll master is known as the MERN (MongoDB, Express, React, and Node.js) stack.

The 5 W's

1. Who—The users of your myFlix application. They will be movie enthusiasts who enjoy reading information about different movies.
2. What—A single-page, responsive application with routing, rich interactions, several interface views, and a polished user experience. The client-side developed in this Achievement will support the existing server-side from Achievement 2 by facilitating user requests and rendering the response from the server-side via a number of different interface views.

3. When—myFlix users will be able to use it whenever they want to read information about different movies or update their user information—for instance, their list of “Favorite Movies.”
4. Where—The application will be hosted online. The myFlix application itself is responsive and can therefore be used anywhere and on any device, giving all users the same experience.
5. Why—Movie enthusiasts like to be able to access information about different movies, directors, and genres, whenever they want to. Having the ability to save lists of favorite movies will ensure users always have access to the films they want to watch or recommend to their peers.

Design Criteria

User Stories

- As a user, I want to be able to access information on movies, directors, and genres so that I can learn more about movies I’ve watched or am interested in.
- As a user, I want to be able to create a profile so I can save data about my favorite movies.

Features & Requirements

The feature requirements below were extracted from the user stories listed above. **Your project will only be approved if the following “essential” feature requirements are implemented in your Achievement project**.

Essential Views and Features

Main view

- Returns a list of ALL movies to the user (each listed item with an image, title, and description)
- Sorting and filtering
- Ability to select a movie for more details

Single movie view

- Returns data (description, genre, director, image) about a single movie to the user
- Allows users to add a movie to their list of favorites

Login view

- Allows users to log in with a username and password

Registration view

- Allows new users to register (username, password, email, birthday)

Genre view

- Returns data about a genre, with a name and description
- Displays example movies

Director view

- Returns data about a director (name, bio, birth year, death year)
- Displays example movies

Profile view

- Allows users to update their user info (username, password, email, date of birth)
- Allows existing users to deregister
- Displays favorite movies
- Allows users to remove a movie from their list of favorites

Optional Views and Features

Single movie view and all movies views

- Allow users to see which actors star in which movies
- Allow users to view more information about different movies, such as the release date and the movie rating

Actors view

- Allows users to view information about different actors

Profile view, single movie view, and all movies view

- Allow users to create a “To Watch” list in addition to their “Favorite Movies” list

Wireframes

You can download wireframes for each of the views for your project here: myFlix wireframe pack

Technical Requirements

- The application must be a single-page application (SPA)
- The application must use state routing to navigate between views and share URLs
- The application must give users the option to filter movies
- The application must give users the option to sort movies
- The application must initially use Parcel as its build tool
- The application must be written using the React library and in ES2015+
- The application must be written with React Redux (hence respecting the Flux pattern)
- The application must use Bootstrap as a UI library for styling and responsiveness
- The application must contain a mix of class components and function components
- The application may be hosted online