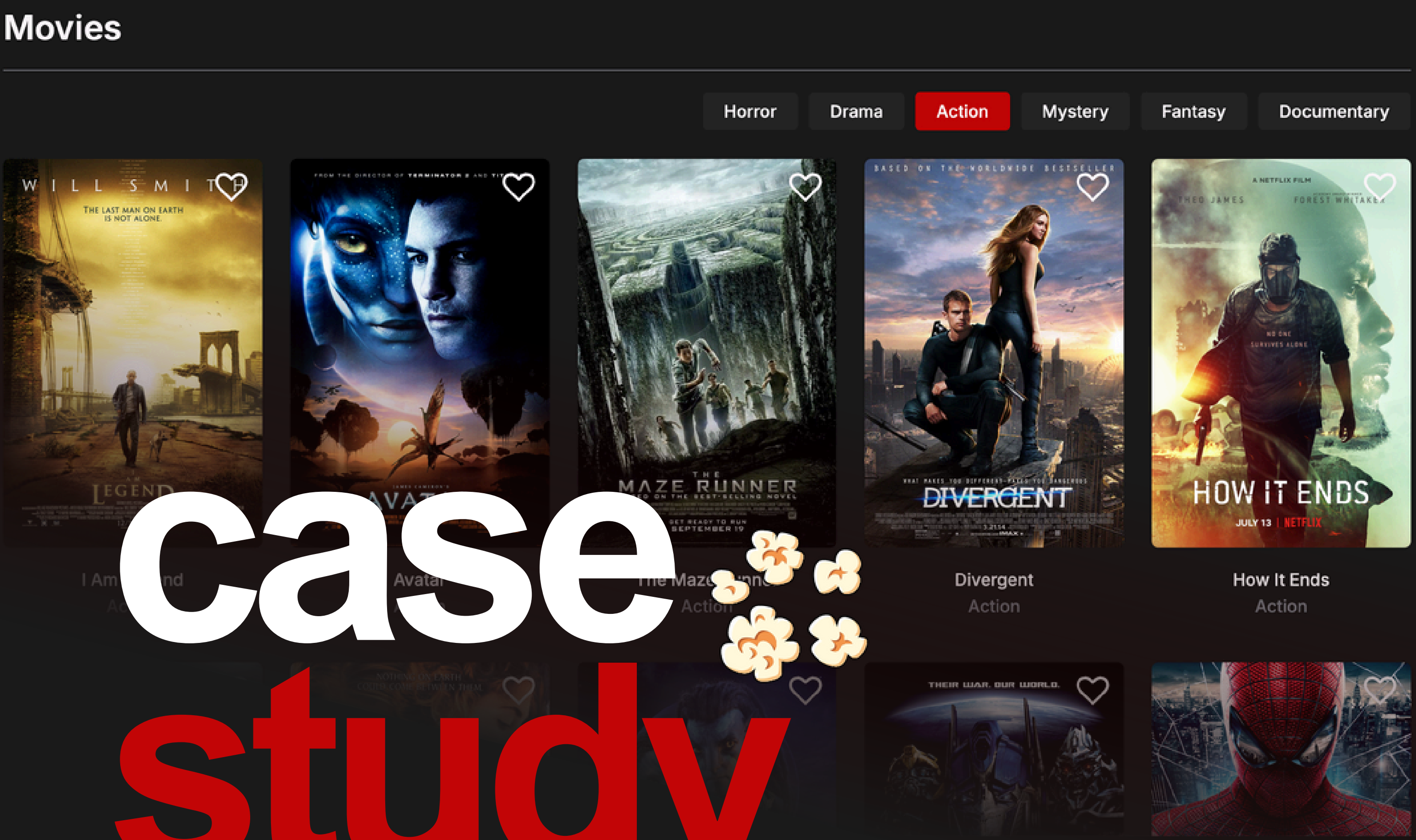


case study

POPCORNPAL

PopcornPal is a full-stack web application that allows users to browse and view detailed information about movies. Users can create accounts, manage their profiles, and add movies to their "My Favorites" list. The frontend is built with React, while the back end integrates a custom RESTful API for fetching movie data and authenticating users. Custom-built CSS ensures a smooth experience across all devices.

OBJECTIVES

Client-side

Several interface views (using React) that handle data through REST API endpoints and display movies to users

Rest API and Database

Build a database to store and manage movie and user data

FEATURES

Browse Movies

Browse through a wide collection of movies and access movie information

User Registration

Sign up and log in securely, powered by JSON web token

Favorites List

Personalized list to add and remove favorite movies

Profile Management

Registered users can modify their profile

TECHNOLOGIES

React, MongoDB, Node.js, Express, Mongoose, Framer, Parcel, Bcrypt, Passport.js, Netlify, Heroku, JWT, REST API, and CSS

monicaalyssa

popcornpal.netlify.app

Frontend (Client-side)

React.js was used as the core framework for building the user interface using components. It handled dynamic states for elements like the movie list, user favorites, profile data, and dropdown menus, making it easier to manage the interactions and updates to the UI based on changing data fetched from the API.

React-router-dom was used to handle navigation between different views, such as the homepage, individual movie pages, and the user profile page.

Custom CSS was used to style the application and align it with the desired aesthetic for every component. A red, very dark grey, and yellow color scheme was used to create a distinctive look. Using media queries the CSS was fine-tuned for the user interface making it accessible on all devices.

Framer Motion was used to enhance the user experience by adding animations. Specifically, it animated transitions when users clicked through different genres to display an engaging interaction for movies of that specific genre.

Hosting

Netlify was used to deploy and host the client-side React application.

Heroku was used to host the backend API, which was built with Node.js and Express.

Backend (Rest API and Database)

MongoDB Atlas was used as a cloud-based NoSQL database to store movie and use data.

Mongoose was used as a Object Data Modeling tool to interact with MongoDB to define movie and user schemas ensuring consistent data structure and validation.

Node.js served as a runtime environment for executing JavaScript on the server. It handled HTTP requests and sent responses back to the client. It also was responsible for managing API endpoints.

Express worked alongside Node.js to handle HTTP requests and define routes for the API. It managed middleware such as authentication, error handling, and parsing of incoming requests.

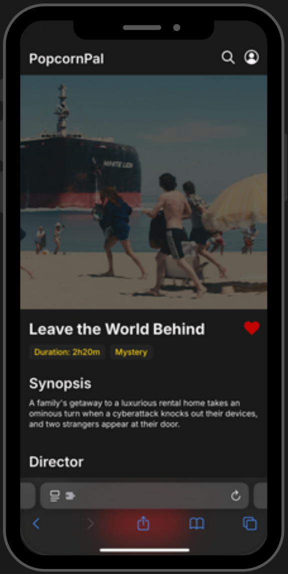
JWT/JSON Web Token was implemented for secure, stateless user authentication. When users logged in, the server generated a token that includes their user information, to then authenticate future requests.

Passport.js was used alongside JWT to implement user authentication by integrating a passport strategy that verifies credentials (username/password) and matches them against stored user data in the database.

Bcrypt was used to enhance security and hash user passwords before storing them in the database. This ensures user passwords are never stored in plaintext.

React Components

```
├──── components
│     ├──── main-view.jsx
│     ├──── login-nav-bar.jsx
│     ├──── login-view.jsx
│     ├──── signup-view.jsx
│     ├──── nav-bar.jsx
│     ├──── hero.jsx
│     ├──── movie-grid.jsx
│     ├──── movie-card.jsx
│     ├──── genre-dropdown.jsx
│     ├──── genre-item.jsx
│     ├──── user-menu.jsx
│     └──── profile-view.jsx
└──── index.jsx
```



A family's getaway to a luxurious rental home takes an ominous turn when a cyberattack knocks out their devices, and two strangers appear at their door.

CONCLUSION

What I've Learned

PopcornPal demonstrates my ability to build a full-stack web application. Through this project, I learned React and backend development, enhancing my skills in front-end frameworks, state management, secure user authentication, API creation, and UI design. Additionally, I gained hands-on experience with hosting and deployment platforms, using Heroku to host the backend API and Netlify for client-side deployment.

popcornpal.netlify.app

monicaalyssa