Case Study

MyMDB - My Movie Database

MyMDb

My Movie Database

(Client Side / Angular)

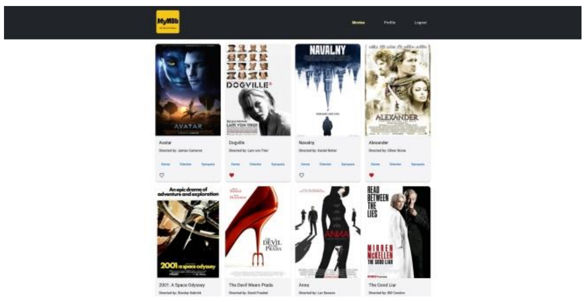
GitHub Repository: https://github.com/ibxibx/mymdb-angular-client/

Live Demo: https://ibxibx.github.io/mymdb-angular-client/welcome

Project Description

MyMDB is a single-page, responsive movie database application built with Angular that interfaces with a RESTful movie API. The application allows users to browse a collection of movies, view detailed information about movies, directors, and genres, as well as create and manage their own user profile and favourite movies list.

This client-side application was developed to complement a previously built server-side API, providing users with an intuitive and engaging interface to interact with the movie database. The project demonstrates the implementation of modern Angular practices, including component architecture, services, routing, and integration with Angular Material for a consistent user experience. This study project was my first project built with TypeScript and Angular and advanced my front-end skills.



Movie View Screen

Project Objectives

- Create a client-side application using Angular to provide an interface for the existing MyMDB API

- Implement user registration and authentication functionality

- Allow users to view and interact with movie data through an intuitive interface

- Enable users to maintain a profile and manage their favourite movies list

- Design a responsive application that works across various devices

- Apply Angular Material design principles for a consistent and attractive user interface

- Ensure proper documentation and code organization

- Deploy the application to GitHub Pages for public access

Tech Stack and Specs

- **Frontend Framework**: Angular 17
- **UI Components**: Angular Material
- **Styling**: SCSS
- **State Management**: RxJS
- **Authentication**: JWT (JSON Web Tokens)
- **HTTP Communication**: Angular HttpClient
- **Form Management**: Angular Reactive Forms
- **Development Tools**:
  - TypeScript
  - Angular CLI
  - npm
- **Documentation**: TypeDoc for technical documentation
- **Deployment:** GitHub Pages
- **Version Control:** Git & GitHub

User Stories

⇒ **As a user, I want to be able to register and log in to the application, so that I can access personalized features and maintain my movie preferences across sessions.**

⇒ **As a user, I want to be able to browse and search for movies, so that I can discover new films and access information about movies I'm interested in.**

⇒ **As a user, I want to be able to view detailed information about movies, directors, and genres, so that I can learn more about the films and make informed decisions about what to watch.**

⇒ **As a user, I want to be able to maintain a list of favourite movies, so that I can keep track of films I enjoy and want to revisit.**

⇒ **As a user, I want to be able to update my profile information, so that I can keep my account details current and personalized.**

Project Structure

The MyMDB Angular client application follows a modular component-based architecture that aligns with Angular best practices. The main components include:

- **Welcome Component**: Serves as the entry point, handling user registration and login

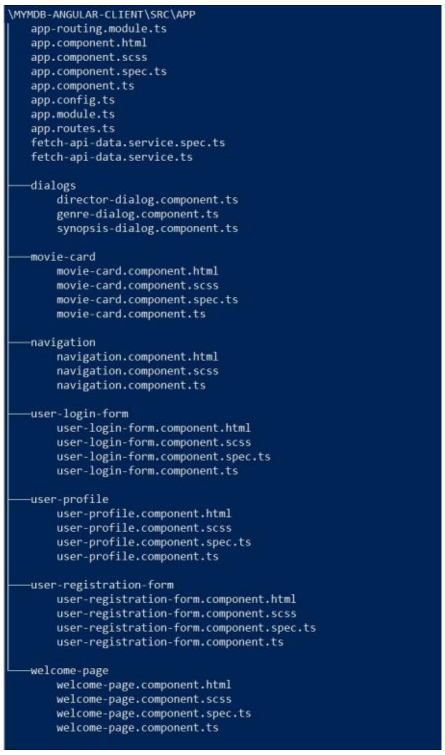
- **Movie Card Component**: Displays the movie collection with interactive cards

- **Movie Details Component**: Shows comprehensive information about a selected movie

- **Director Component**: Provides biographical information about movie directors

- **Genre Component**: Offers details about specific movie genres

- **User Profile Component**: Allows users to view and edit their profile information



```
\MYMDB-ANGULAR-CLIENT\SRC\APP
│   app-routing.module.ts
│   app.component.html
│   app.component.scss
│   app.component.spec.ts
│   app.component.ts
│   app.config.ts
│   app.module.ts
│   app.routes.ts
│   fetch-api-data.service.spec.ts
│   fetch-api-data.service.ts
│
├───dialogs
│       director-dialog.component.ts
│       genre-dialog.component.ts
│       synopsis-dialog.component.ts
│
├───movie-card
│       movie-card.component.html
│       movie-card.component.scss
│       movie-card.component.spec.ts
│       movie-card.component.ts
│
├───navigation
│       navigation.component.html
│       navigation.component.scss
│       navigation.component.ts
│
├───user-login-form
│       user-login-form.component.html
│       user-login-form.component.scss
│       user-login-form.component.spec.ts
│       user-login-form.component.ts
│
├───user-profile
│       user-profile.component.html
│       user-profile.component.scss
│       user-profile.component.spec.ts
│       user-profile.component.ts
│
├───user-registration-form
│       user-registration-form.component.html
│       user-registration-form.component.scss
│       user-registration-form.component.spec.ts
│       user-registration-form.component.ts
│
└───welcome-page
        welcome-page.component.html
        welcome-page.component.scss
        welcome-page.component.spec.ts
        welcome-page.component.ts
```

App Components /src/app Folder

- **Navigation Component**: Facilitates movement between different views of the application

These components are supported by a service layer that handles API communication, user authentication, and data management. The application uses Angular's routing module to navigate between different views while maintaining a single-page application architecture.

Building Steps

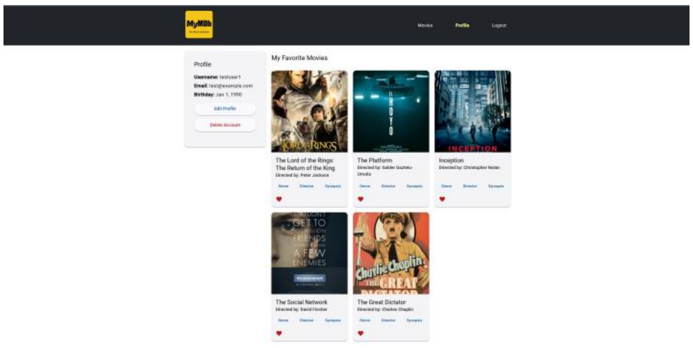
1. Initial Setup

Created a new Angular project using Angular CLI with the command `ng new mymdb-angular-client`. This established the project structure and installed necessary dependencies. Generated the API service using `ng generate service fetch-apidata` to implement the logic for making API calls to the backend endpoints.

2. API Service Implementation

Implemented the core service functionality to communicate with the backend API. This included methods for user registration, login, movie retrieval, and favourite movie management. Set up the HTTP interceptors and error handling to ensure robust API communication.

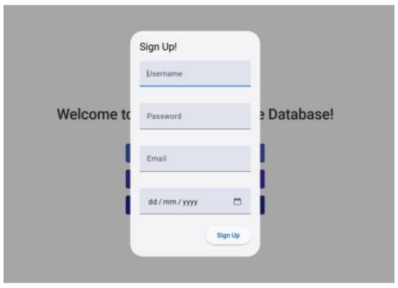
3. Angular Material Integration Added Angular Material to the project for consistent UI components. Configured the theme and imported necessary modules. This provided access to pre-built components like cards, buttons, dialogs, and form controls that maintain Material Design principles.



Profile View with Favourite Movies List

4. User Registration Implementation

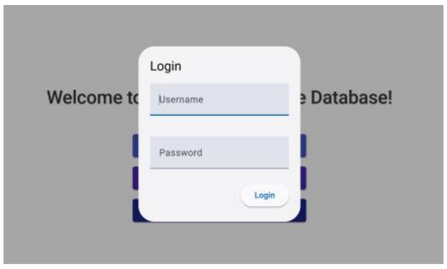
Developed the user registration functionality with Angular Material forms. Created a sign-up form component with validation for username, password, email, and birthday fields. Integrated this with the API service to allow new users to create accounts.



Sign Up Modal

5. Login Functionality

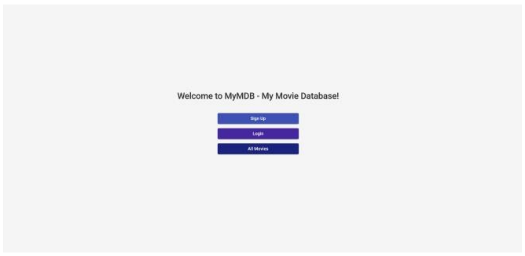
Built the login component (mymdb-angular-client\src\app\user-login-form\user-login-form.component.ts) with form validation and error handling. Implemented JWT token storage in localStorage for maintaining user sessions. Added redirection to the movie view upon successful authentication.



Login Modal

6. Welcome Page Development

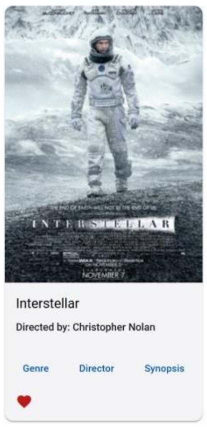
Designed and implemented a welcoming landing page (pic.X) that introduces the application and provides options for users to register or log in. Added responsive styling to ensure a good experience across device sizes.



Welcome Screen

7. Movie Card Component

Created the movie card component to display the collection of movies in a grid layout. Each card shows the movie title, an image, and basic information. Implemented actions for viewing details and adding to favorites.



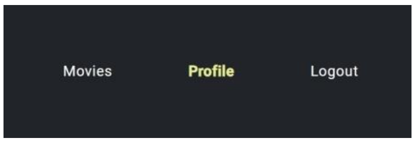
Movie Card

8. GitHub Pages Deployment

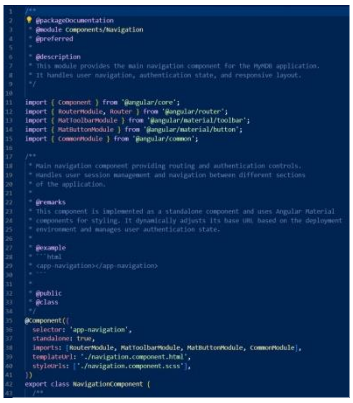
Set up the application for deployment on GitHub Pages by configuring the base href and build options. This made the application publicly accessible for demonstration purposes.

9. Navigation and User Profile

Added a navigation bar for easy movement between application views. Developed the user profile component that displays user information and allows for profile editing, including username, password, and email changes.



Header Navigation Menu



```
/**
 * @packageDocumentation
 * @module Components/Navigation
 * @preferred
 *
 * @description
 * This module provides the main navigation component for the MyMDB application.
 * It handles user navigation, authentication state, and responsive layout.
 */

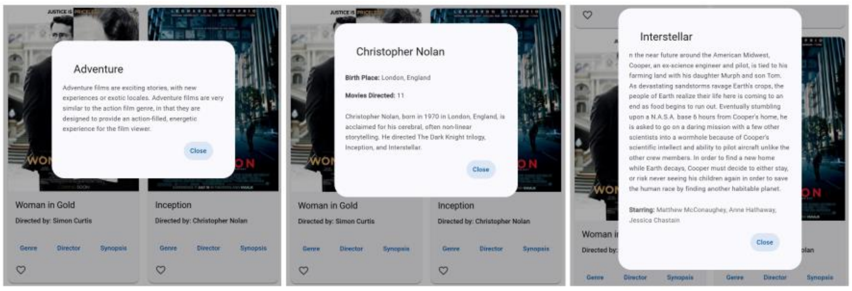
import { Component } from '@angular/core';
import { RouterModule, Router } from '@angular/router';
import { MatToolbarModule } from '@angular/material/toolbar';
import { MatButtonModule } from '@angular/material/button';
import { CommonModule } from '@angular/common';

/**
 * Main navigation component providing routing and authentication controls.
 * Handles user session management and navigation between different sections
 * of the application.
 *
 * @remarks
 * This component is implemented as a standalone component and uses Angular Material
 * components for styling. It dynamically adjusts its base URL based on the deployment
 * environment and manages user authentication state.
 *
 * @example
 * ```html
 * <app-navigation></app-navigation>
 * ```
 *
 * @public
 * @class
 */
@Component({
  selector: 'app-navigation',
  standalone: true,
  imports: [RouterModule, MatToolbarModule, MatButtonModule, CommonModule],
  templateUrl: './navigation.component.html',
  styleUrls: ['./navigation.component.scss'],
})
export class NavigationComponent {
  /**
```

navigation.component.ts
code fragment

10. Feature Completion

Implemented the remaining components for movie details, director information, and genre descriptions using dialog modals. Added the favorite movies functionality to allow users to maintain a list of preferred films. Polished the welcome menu options for better user experience.



Adventure

Adventure films are exciting stories, with new experiences or exotic locales. Adventure films are very similar to the action film genre, in that they are designed to provide an action-filled, energetic experience for the film viewer.

Christopher Nolan

Birth Place: London, England

Movies Directed: 11

Christopher Nolan, born in 1970 in London, England, is acclaimed for his cerebral, often non-linear storytelling. He directed The Dark Knight trilogy, Inception, and Interstellar.

Interstellar

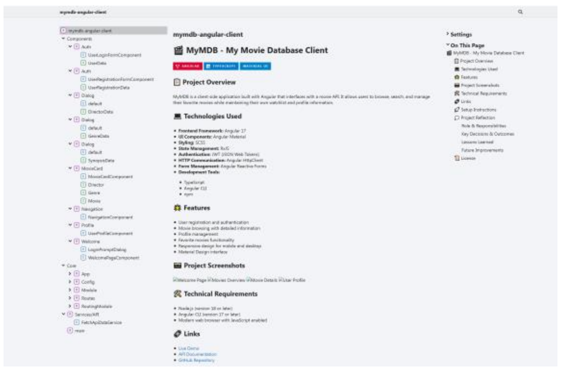
n the near future around the American Midwest, Cooper, an ex-science engineer and pilot, is tied to his farming land with his daughter Murph and son Tom. As devastating sandstorms ravage Earth's crops, the people of Earth realize their life here is coming to an end as food begins to run out. Eventually stumbling upon a N.A.S.A. base 6 hours from Cooper's home, he is asked to go on a daring mission with a few other scientists into a wormhole because of Cooper's scientific intellect and ability to pilot aircraft unlike the other crew members. In order to find a new home while Earth decays, Cooper must decide to either stay, or risk never seeing his children again in order to save the human race by finding another habitable planet.

Starring: Matthew McConaughey, Anne Hathaway, Jessica Chastain

Genre, Director and Synopsis Modals

11. Documentation and Finalization

Added comprehensive JSDoc and TypeDoc comments throughout the codebase to generate technical documentation. Created a detailed README with project information, screenshots, and setup instructions. Made final UI adjustments and tested all functionality. The documentation made with JSDoc is accessible via the file: https://github.com/ibxibx/mymdb-angular-client/blob/main/docs/index.html



mymdb-angular-client

🎬 MyMDB - My Movie Database Client

📋 Project Overview

[illegible]

💻 Technologies Used

- **Frontend Framework:** Angular 17
- **UI Components:** Angular Material
- **Styling:** SCSS
- **State Management:** RxJS
- **Authentication:** JWT (JSON Web Tokens)
- **HTTP Communication:** Angular HttpClient
- **Form Management:** Angular Reactive Forms
- **Development Tools:**
  - TypeScript
  - Angular CLI
  - npm

✨ Features

- User registration and authentication
- Movie browsing with detailed information
- Profile management
- Favorite movies functionality
- Responsive design for mobile and desktop
- Material Design interface

📷 Project Screenshots

Welcome Page Movies Overview Movie Details User Profile

🛠 Technical Requirements

- Node.js (version 18 or later)
- Angular CLI (version 17 or later)
- Modern web browser with JavaScript enabled

🔗 Links

- Live Demo
- API Documentation
- GitHub Repository

Technical Challenges and Solutions

Challenge 1: TypeScript Type Safety

Challenge: As a first-time Angular and TypeScript developer, adapting to TypeScript's strict typing system required a significant mindset shift from JavaScript's more flexible approach.

Solution: I invested time in learning TypeScript fundamentals and leveraged Angular's generated interfaces. Created specific interfaces for all data models (User, Movie, Director, Genre) which enhanced code quality and reduced runtime errors through compile-time checking.

Challenge 2: JWT Authentication Flow

Challenge: Implementing a secure authentication flow with JWT tokens while maintaining state across component navigation presented complexity in token storage and validation.

Solution: Developed a comprehensive authentication service that handles token storage in localStorage, automatic token inclusion in API requests through HTTP interceptors, and token expiration checking. This provided a secure but seamless authentication experience.

Challenge 3: Dialog Component Communication

Challenge: Passing data between parent components and dialog modals (for movie details, director info, and genre information) while maintaining proper component isolation was difficult.

Solution: Utilized Angular Material's MatDialog service with data injection, allowing parent components to pass specific data to dialog components. Implemented dialog references to handle return values when dialogs close, creating a clean communication channel between components.

Challenge 4: Responsive Design Implementation

Challenge: Ensuring the application looked and functioned well across various device sizes without duplicating style code was challenging.

Solution: Leveraged Angular Material's built-in responsive features combined with custom SCSS media queries. Created a flexible grid system for movie cards that automatically adjusts based on screen width. Utilized Angular's flex-layout module for responsive container management.

Challenge 5: Form Validation

Challenge: Creating user-friendly forms with proper validation feedback for registration and profile editing required handling multiple validation scenarios.

Solution: Implemented Angular's Reactive Forms with custom validators and error messages. Created reusable validation patterns and error display components that provide immediate visual feedback, improving the user experience during form completion.

Skills Acquired

Through the development of the MyMDB Angular client application, I gained proficiency in several key areas:

- **Angular Framework**: Learned Angular's component architecture, services, dependency injection, and module system.

- **TypeScript Programming**: Developed strong typing skills, interface definitions, and leveraged TypeScript's object-oriented features.

- **Angular Material**: Gained experience with Material Design principles and implementing pre-built UI components for a professional look and feel.

- **RxJS and Observables**: Learned to work with reactive programming paradigms for handling asynchronous operations and data streams.

- **Routing and Navigation**: Implemented Angular's router for creating a seamless single-page application experience.

- **Form Handling**: Mastered both template-driven and reactive forms, including validation strategies and custom validators.

- **API Integration**: Developed skills in connecting frontend applications with RESTful APIs using Angular's HttpClient.

- **Authentication**: Implemented JWT-based authentication and protected routes.

- **Documentation**: Learned to use TypeDoc for generating technical documentation from code comments.

- **Deployment**: Gained experience with deploying Angular applications to GitHub Pages.

Final Thoughts and Reflections

Developing the MyMDB Angular client application was a significant learning experience that pushed me to expand my frontend development skills beyond my previous knowledge. Starting with little prior experience in Angular or TypeScript, this project provided a comprehensive introduction to the Angular ecosystem and its approach to building modern web applications.

The transition from React to Angular presented an interesting challenge, as the frameworks have different philosophies and patterns. Angular's more opinionated structure, with its emphasis on TypeScript and comprehensive tooling, initially felt restrictive but ultimately proved beneficial for maintaining a consistent and organized codebase. The enforced discipline of TypeScript's type system caught numerous potential bugs during development that might have only appeared at runtime in a JavaScript-based project.

Angular Material proved to be an excellent choice for UI components, allowing me to focus on functionality rather than spending excessive time on styling. The pre-built components integrated seamlessly with Angular's form system, creating a polished user experience with relatively little custom CSS required.

One of the most valuable lessons from this project was understanding the importance of proper state management in a component-based architecture. Using services effectively to share data between components and handling asynchronous operations significantly improved the application's performance and maintainability.

If I were to approach this project again, I would spend more time planning the component structure before beginning implementation. Some components required refactoring as the project progressed and requirements became clearer. I would also explore NgRx for more robust state management, as the application complexity grew beyond what simple services could efficiently handle.

For developers interested in testing the application without creating a new account, you can use these credentials:
- Username: testuser1
- Password: Test123!

This project has been an invaluable addition to my portfolio, demonstrating my ability to learn and apply new technologies in a practical context. The experience gained with Angular will serve as a solid foundation for future frontend development work, and the patterns learned—component architecture, services, reactive programming—are applicable across modern JavaScript frameworks.