Bachelor of IT (Computer Science)
Assignment 2 - Client-Side React Application
CAB230 - Web Computing

Dane Madsen
n10983864@qut.edu.au

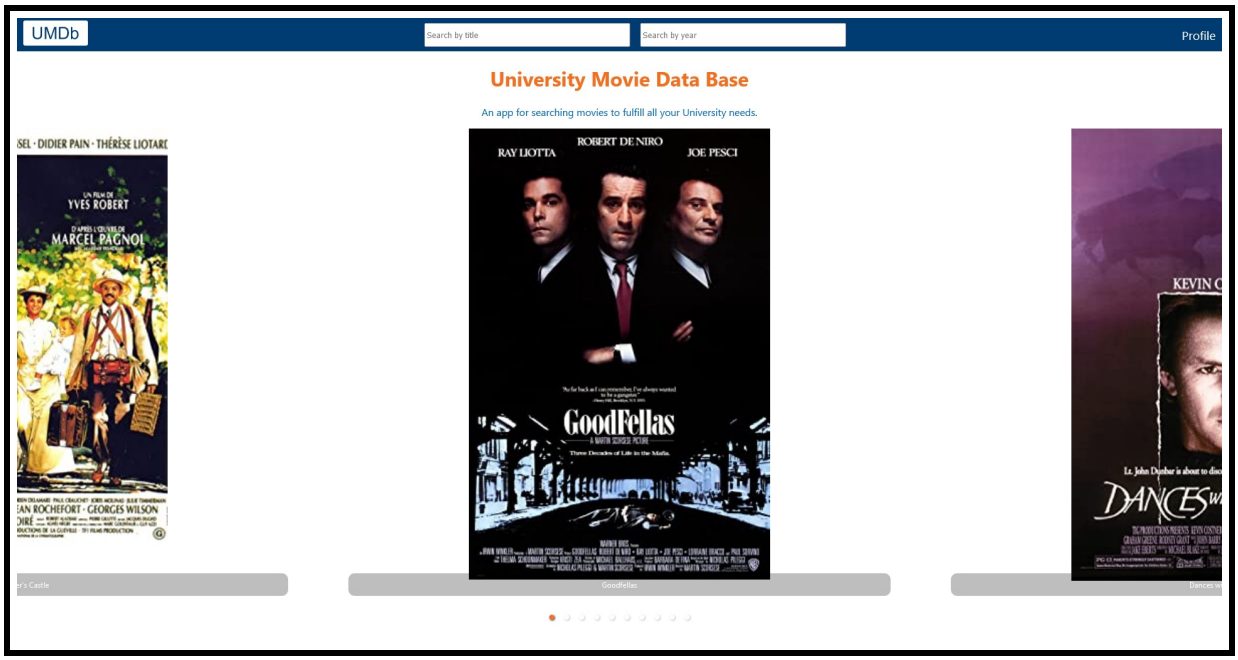
Contents

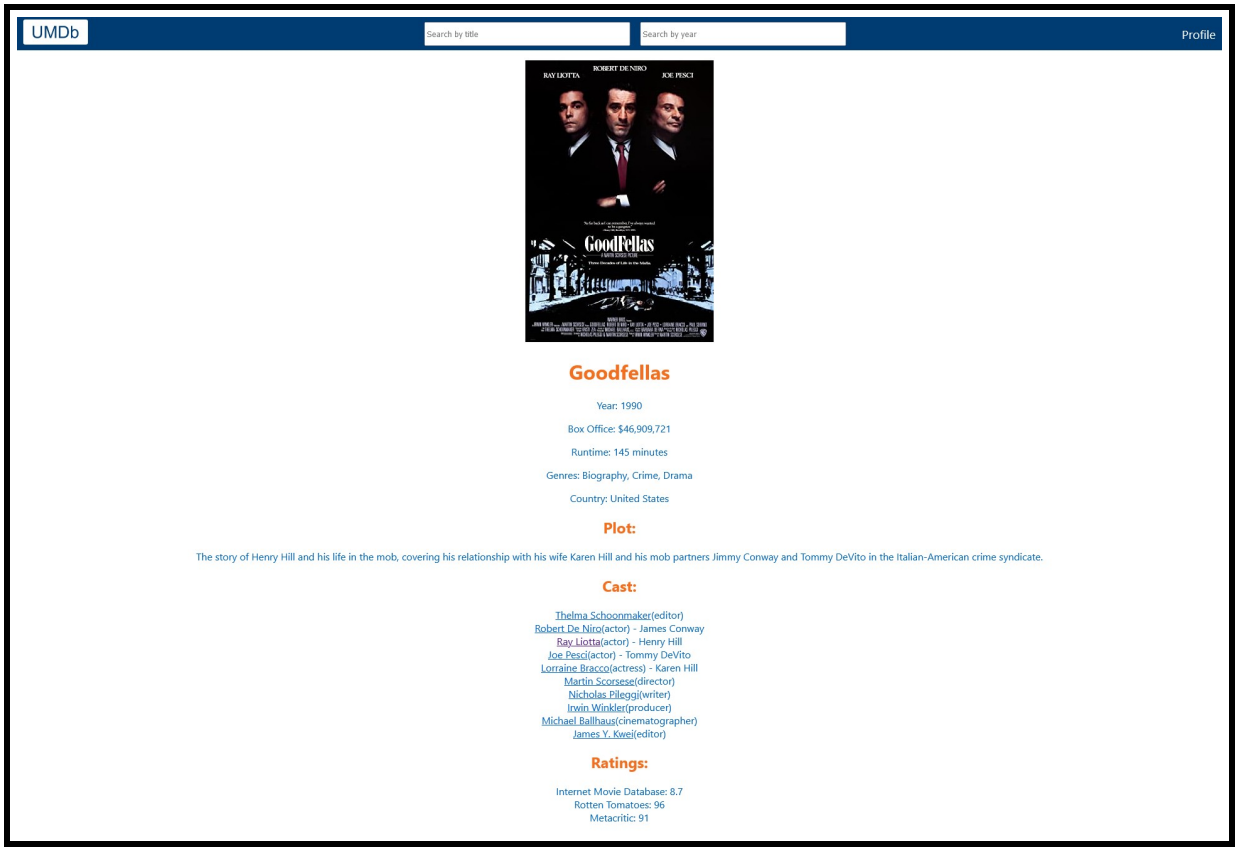
1 Introduction

1.1 Purpose and Description

The purpose of this react application is to collate and display information regarding movies, and cast members to the user in a responsive and accessible manner. The application should allow the user to search for movies by year or title and display the results in a list. When a movie is selected, the application should display all the details of the movie, including the year the movie was made, the plot of the movie and all the cast members involved with the movie. The application should also allow the user to visit individual pages of cast members where the user can view that cast members details, including their birth year, death year and all the movies they have been involved with.

It achieve this goal and to provide the user with the best experience I can, i have used a number of advanced react features, including the use of react router, IntesectionObserver and react-responsive-carousel. The use of these features has allowed me to create a professional looking application that is both easy to use and responsive for the user.





1.2 Completeness and Limitations

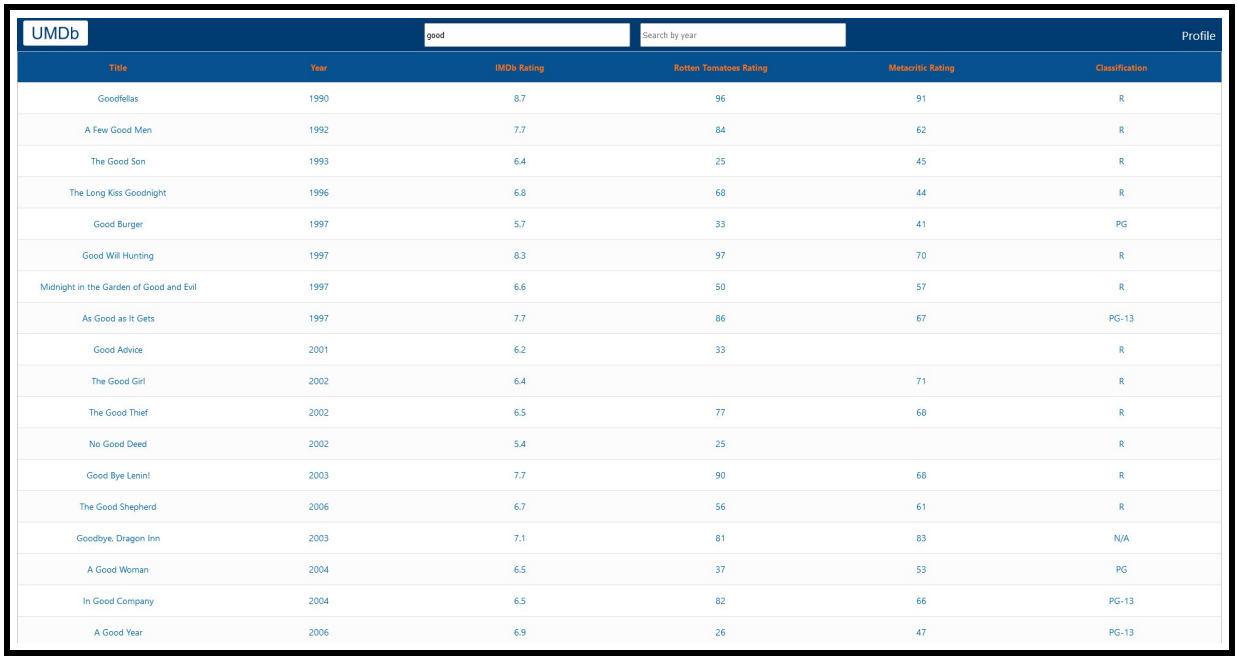
This implementation covers all the requirements of the assignment specification. Navigation is handled using react router, controlled forms are used for all user input, and every 10 minutes the refresh token is used to get a new access token. In addition to this, the application uses ag-grid to display the search results in an infinitly scrolling grid and react-responsive-carousel to display the highest scoring movies of all time on the home (landing) page. On the person page the movies the person is involved in are displayed in a grid with the IMDb Rating.

2 Use of End Points

The functionality for all the API endpoints is handled in API.js. This file contains all the functions that are used to make requests to the API. By containing all the API functionality in one file, it makes it easier to maintain and potentially update the application in the future.

2.1 /movies/search

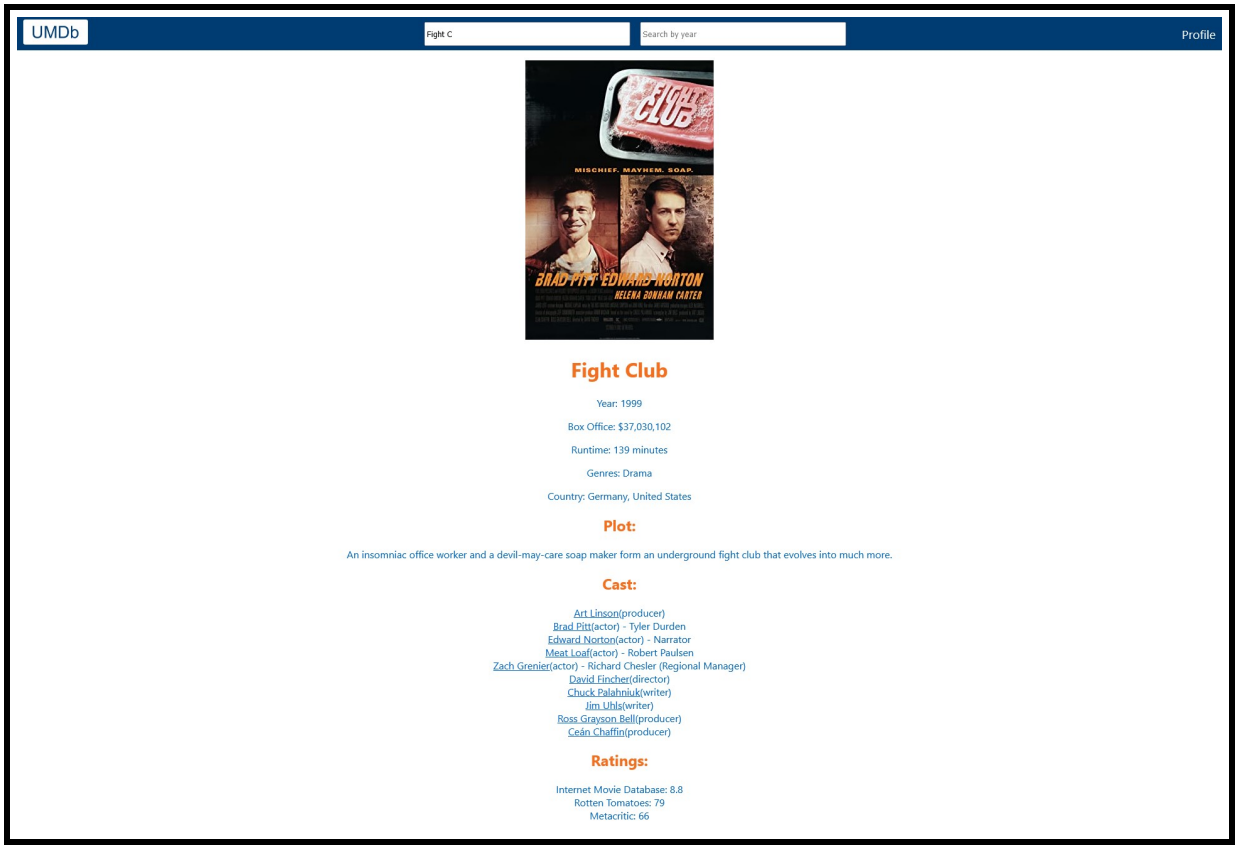
This endpoint is implemented as the getMovies function and is utilised by two search forms at the top of the application. The first search form is for the user to search for movies by title, and the second is for the user to search for movies by year. Both of these forms are controlled forms and can be accessed from any page in the application. Doing it this way removes an extra layer of complexity for the user and allows them to get into the function of the application right from the start of the application.



Title	Year	iMDb Rating	Rotten Tomatoes Rating	Metacritic Rating	Classification
Goodfellas	1990	8.7	96	91	R
A Few Good Men	1992	7.7	84	62	R
The Good Son	1993	6.4	25	45	R
The Long Kiss Goodnight	1996	6.8	68	44	R
Good Burger	1997	5.7	33	41	PG
Good Will Hunting	1997	8.3	97	70	R
Midnight in the Garden of Good and Evil	1997	6.6	50	57	R
As Good as It Gets	1997	7.7	86	67	PG-13
Good Advice	2001	6.2	33		R
The Good Girl	2002	6.4		71	R
The Good Thief	2002	6.5	77	68	R
No Good Deed	2002	5.4	25		R
Good Bye Lenin!	2003	7.7	90	68	R
The Good Shepherd	2006	6.7	56	61	R
Goodbye, Dragon Inn	2003	7.1	81	83	N/A
A Good Woman	2004	6.5	37	53	PG
In Good Company	2004	6.5	82	66	PG-13
A Good Year	2006	6.9	26	47	PG-13

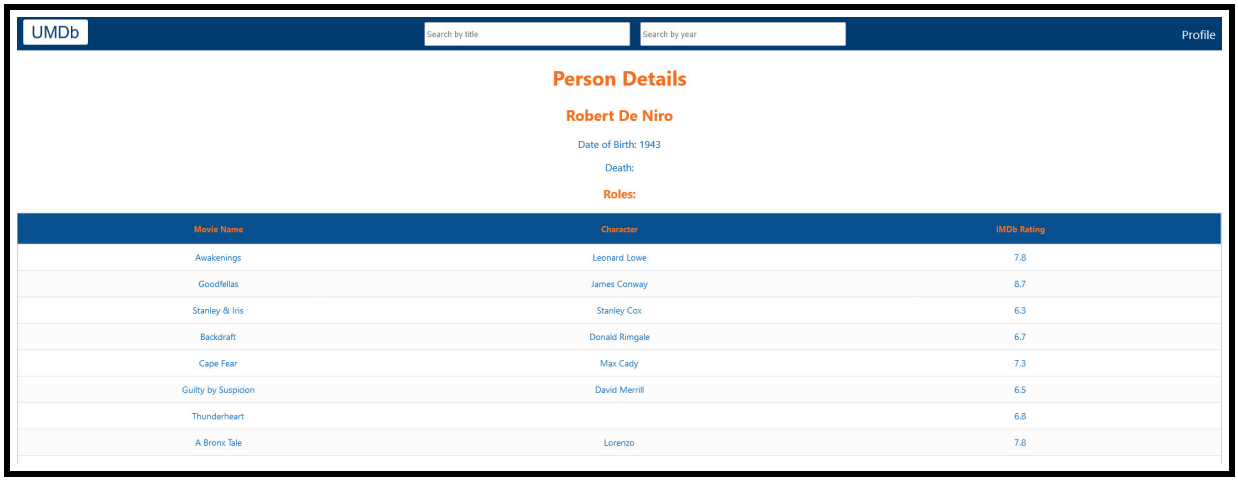
2.2 /movies/data/{imdbID}

This endpoint is implemented as the getMovie function and is utilised by the movie page to get the details of the movie the user has selected. The movie page is accessed by clicking on any of the movies in the search results. The movie page displays all the details of the movie, including the title, year, box office earnings, runtime, genre, country of origin, plot, cast and ratings. each cast member is a clickable link that will lead to that cast members respective page.



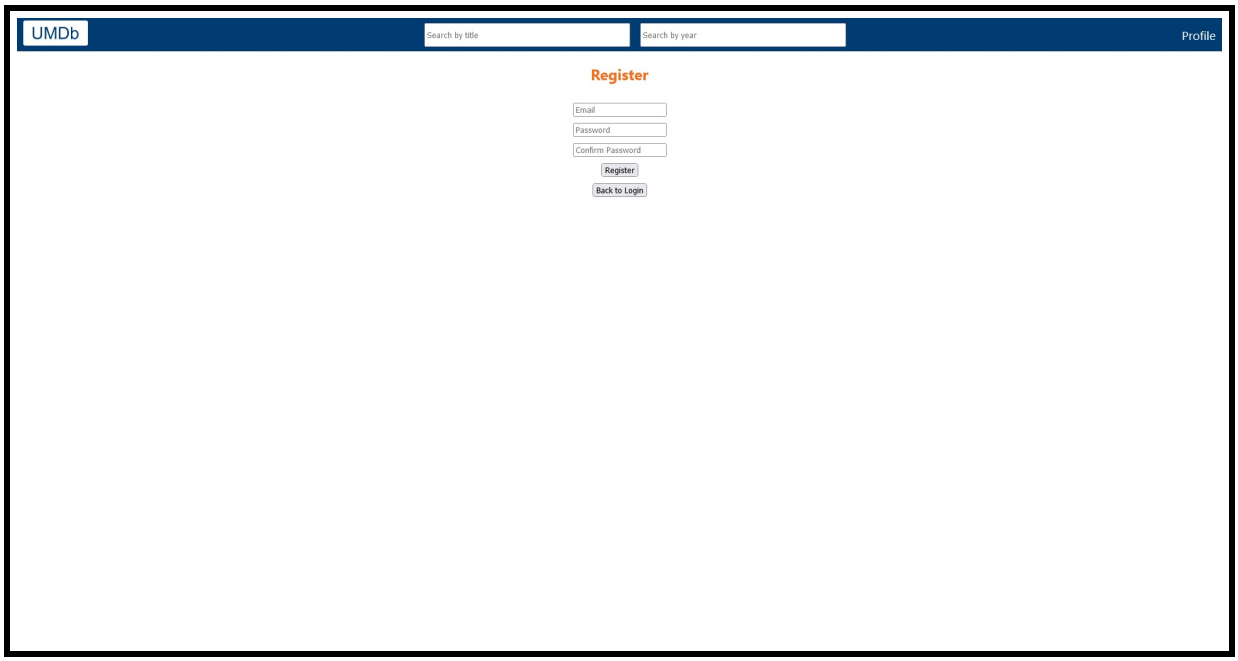
2.3 /people/{id}

This endpoint is implemented as the getPerson function and is utilised by the person page to get the details of the person the user has selected. The person page is accessed by clicking on any of the cast members on the movie page. The person page displays all the details of the person exposed by the API, including their name, birth year, death year, all the movies they have been involved with. In a grid the name of each movie, the name of the character the person played and the IMDb Rating is displayed for the users convenience.



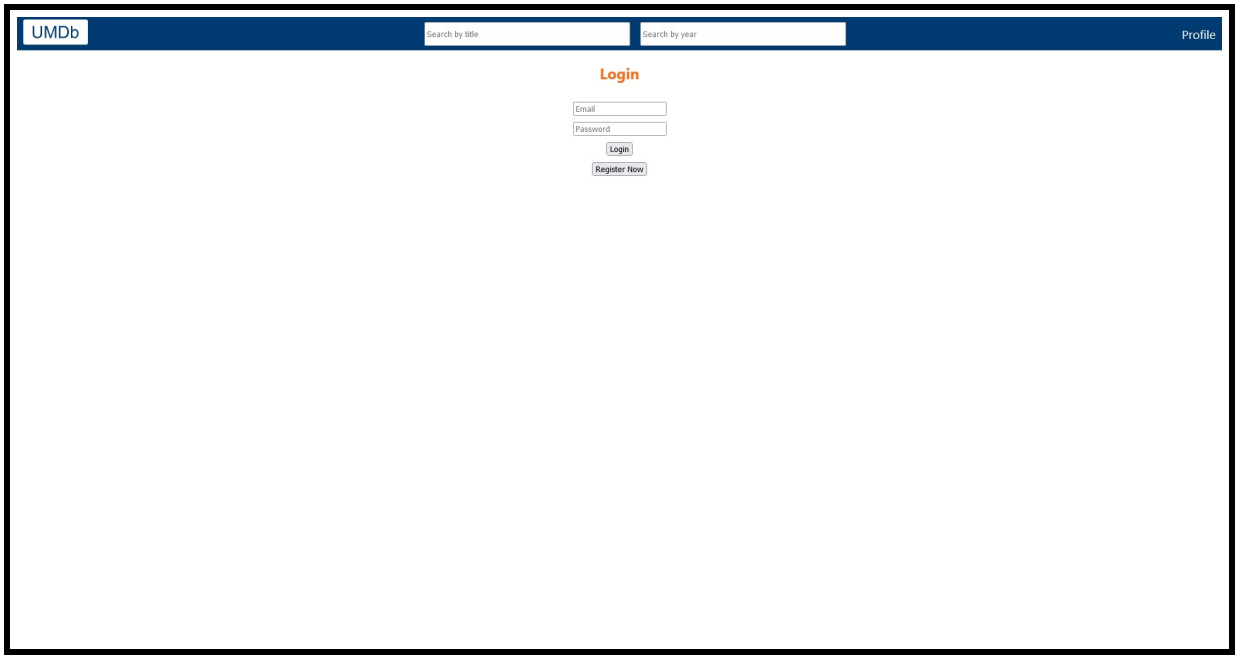
2.4 /user/register

This endpoint is implemented as the postRegister function and is utilised by the profile page to register new users. The profile page is accessed by clicking on the profile button in the navidation bar at the top of the page. When first launched, if the user isnt logged in the profile page will be in the login state, if the user presses the register button the profile page will be transitioned into the register state. The register state contains fields for the user to enter their email, password and confirm password. If the user presses the register button the /user/register endpoint will be utilised to register the user and the tokens will be stored in local storage. If the user presses back to login the page will be transitioned back into the login state.



2.5 /user/login

This endpoint is implemented as the postLogin function and is utilised by the profile page to login existing users. If the user accesses the the profile page while logged out they will be gretted with a login prompt with an email and password field, button to login and a button to register. If the user presses the register button the page will be transitioned into the register state. If the user presses the login button the /user/login endpoint will be utilised to login the user and the tokens will be stored in local storage.

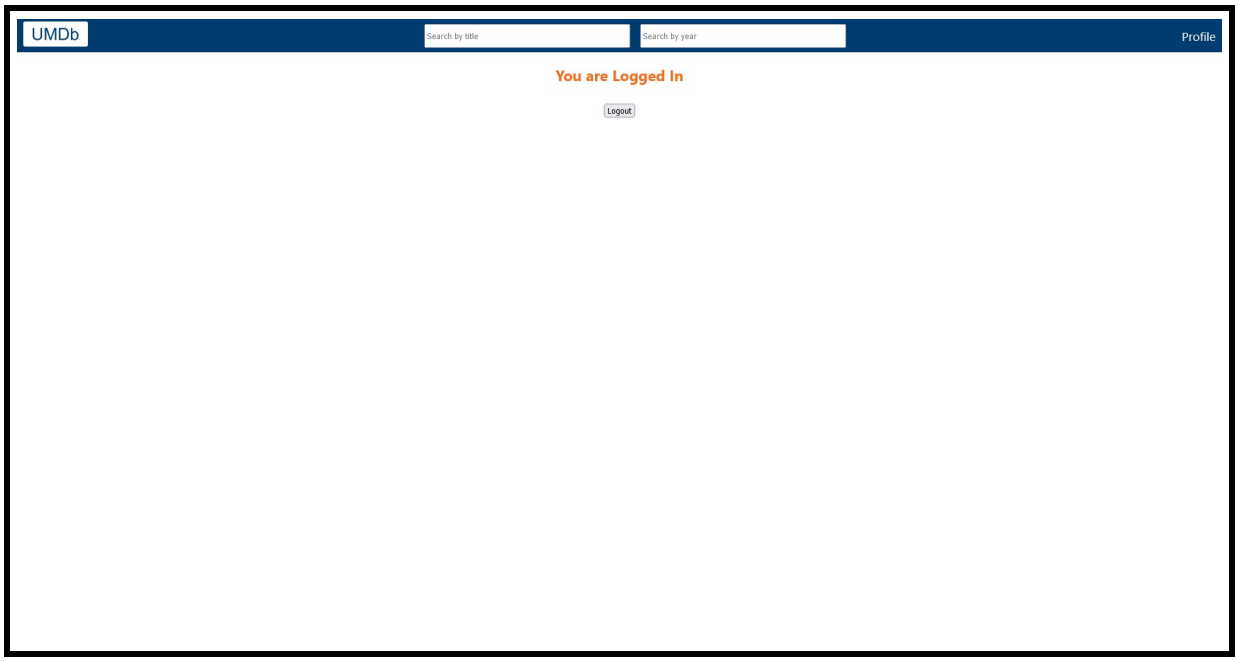


2.6 /user/refresh

This endpoint is implemented as the postRefresh function and is utilised in the back end in App.js to refresh the tokens when the app first starts and every 10 minutes after. This is done to ensure the user does not have to login every 10 minutes.

2.7 /user/logout

This endpoint is implemented as the postLogout function and is utilised by the profile page to logout the user. If the user accesses the the profile page while logged in they will be greeted with a message telling them they are logged in accompanied with a button to logout, if the user presses the logout button the /user/logout endpoint will be utilised to logout the user and the tokens will be deleted.



3 Modules Used

3.1 react-router-dom

This module was used to handle navigation within the application. It was used to create the navigation bar at the top of the page and to create the routes for the different pages in the application.

https://www.npmjs.com/package/react-router-dom

3.2 react-responsive-carousel

This module was used to create a carousel on the home page to display the movies with the highest IMDb Rating.

https://www.npmjs.com/package/react-responsive-carousel

3.3 ag-grid-react

This module was used in MoviesPage to create a grid to display the search results in an infinitely scrolling grid, and in PersonPage it is used to display the movies the person stars in.

https://www.npmjs.com/package/ag-grid-react

3.4 ag-grid-community

This module was used in both MoviesPage and PersonPage to utilise the CSS for the respective grids.

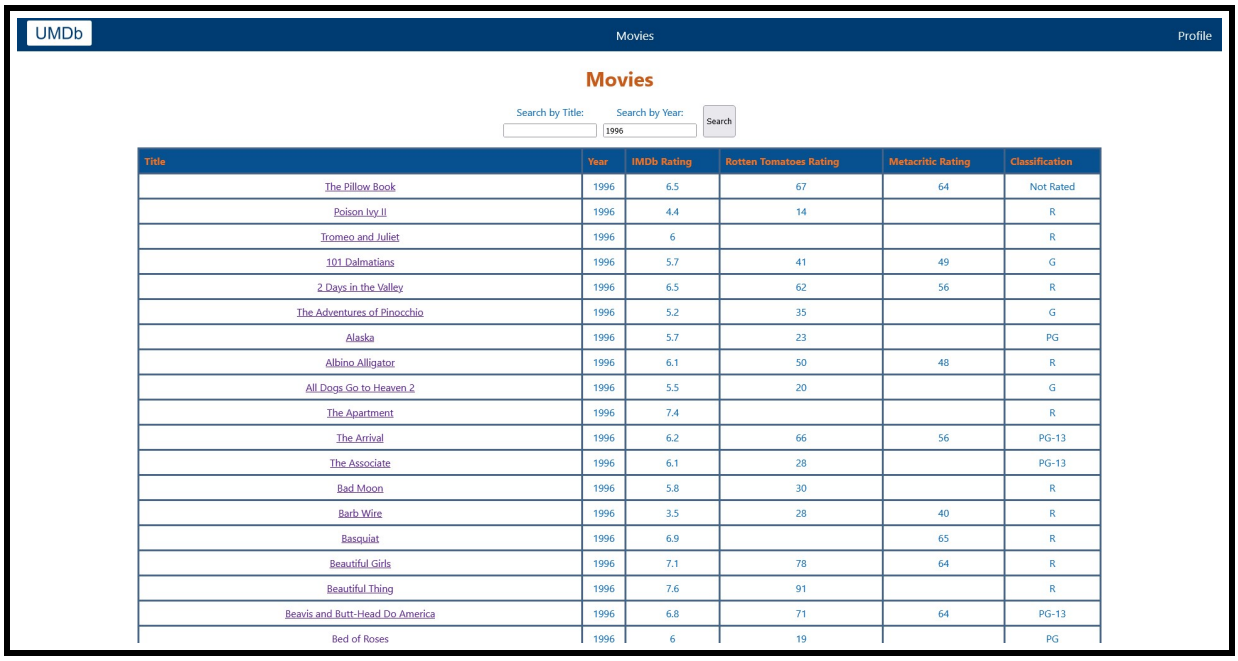
https://www.npmjs.com/package/ag-grid-community

4 Application Design

4.1 Navigation and Layout

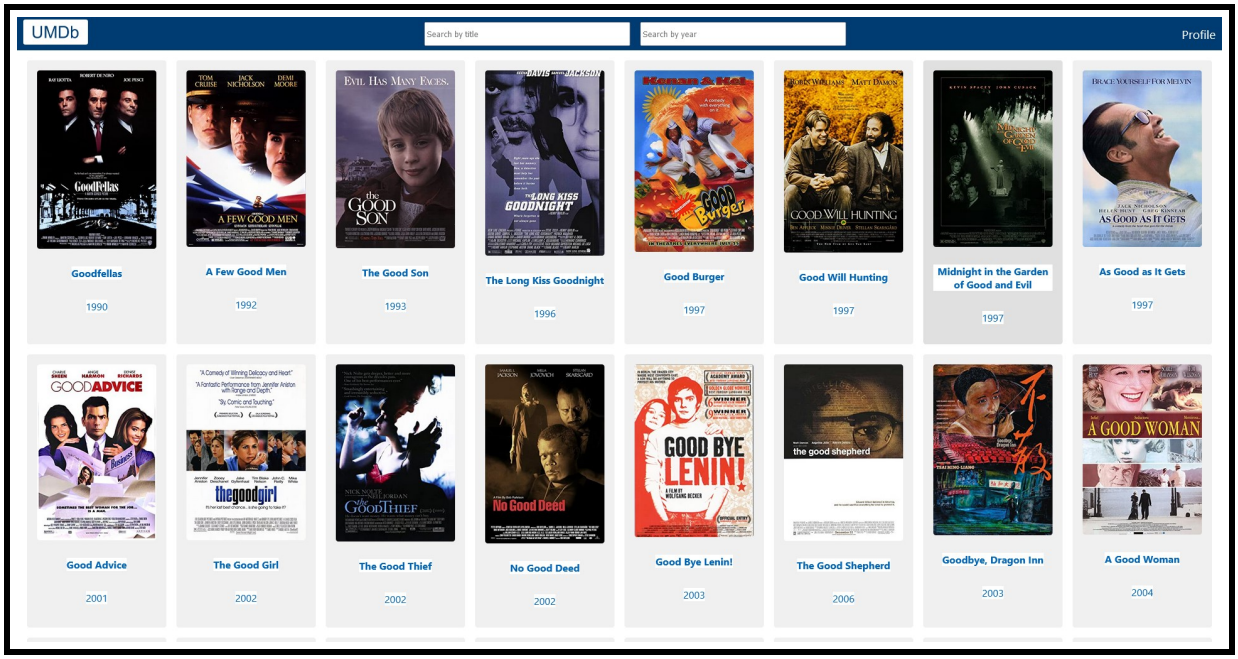
The application has been designed to be as simple to use as possible. The ultimate goal of the design was to develop a user experience that the user can understand and use without having to think about it.

A few design choices were made to achive this goal. The first major choice i made was to move the search bars to the navigation bar at the top of the page. This was done to make the search functionality more accessible to the user by allowing them to search from any page in the application. Moving the search bars to the navigation also freed up space on the MoviesPage allowing me to display more search results at once.



Title	Year	IMDb Rating	Rotten Tomatoes Rating	Metacritic Rating	Classification
The Pillow Book	1996	6.5	67	64	Not Rated
Poison Ivy II	1996	4.4	14		R
Tromeo and Juliet	1996	6			R
101 Dalmatians	1996	5.7	41	49	G
2 Days in the Valley	1996	6.5	62	56	R
The Adventures of Pinocchio	1996	5.2	35		G
Alaska	1996	5.7	23		PG
Albino Alligator	1996	6.1	50	48	R
All Dogs Go to Heaven 2	1996	5.5	20		G
The Apartment	1996	7.4			R
The Arrival	1996	6.2	66	56	PG-13
The Associate	1996	6.1	28		PG-13
Bad Moon	1996	5.8	30		R
Barb Wire	1996	3.5	28	40	R
Basquiat	1996	6.9		65	R
Beautiful Girls	1996	7.1	78	64	R
Beautiful Thing	1996	7.6	91		R
Beavis and Butt-Head Do America	1996	6.8	71	64	PG-13
Bed of Roses	1996	6	19		PG

During development i also investigated the possibility of having the poster image for each movie displayed in the search results. I decided against this as it excessively queries the API and often resulted in a rate limit. It also increased the complexity of being able to sort the search results which was something I wanted to implement.


UMDb
Search by title
Search by year
Profile
Goodfellas
1990
A Few Good Men
1992
The Good Son
1993
The Long Kiss Goodnight
1996
Good Burger
1997
Good Will Hunting
1997
Midnight in the Garden of Good and Evil
1997
As Good as It Gets
1997
Good Advice
2001
The Good Girl
2002
The Good Thief
2002
No Good Deed
2002
Good Bye Lenin!
2003
The Good Shepherd
2006
Goodbye, Dragon Inn
2003
A Good Woman
2004


UMDb
s
Search by year
Profile
The Other Side of the Wind
2018
IMDb Rating: 6.7
Rotten Tomatoes Rating:
Metacritic Rating: 78
Classification: R
A Tale of Springtime
1990
IMDb Rating: 7.1
Rotten Tomatoes Rating:
Metacritic Rating:
Classification: PG
The Adventures of Ford Fairlane
1990
IMDb Rating: 6.4
Rotten Tomatoes Rating:
Metacritic Rating: 24
Classification: R
Another 48 Hrs.
1990
IMDb Rating: 5.9
Rotten Tomatoes Rating:
Metacritic Rating: 23
Classification: R
Awakenings
1990
IMDb Rating: 7.8
Rotten Tomatoes Rating:
Metacritic Rating: 74
Classification: PG-13
Basket Case 2
1990
IMDb Rating: 5.4
Rotten Tomatoes Rating:
Metacritic Rating:
Classification: R
Blue Steel
1990
IMDb Rating: 5.7
Rotten Tomatoes Rating:
Metacritic Rating: 54
Classification: R
The Bonfire of the Vanities
1990
IMDb Rating: 5.6
Rotten Tomatoes Rating:
Metacritic Rating: 27
Classification: R
Child's Play 2
1990
IMDb Rating: 5.9
Rotten Tomatoes Rating:
Metacritic Rating: 37
Classification: R
My Mother's Castle
1990
IMDb Rating: 7.6
Rotten Tomatoes Rating:
Metacritic Rating:
Classification: PG
Class of 1999
1990
IMDb Rating: 5.9
Rotten Tomatoes Rating:
Metacritic Rating: 33
Classification: R
The Comfort of Strangers
1990
IMDb Rating: 6.3
Rotten Tomatoes Rating:
Metacritic Rating: 61
Classification: R
Dances with Wolves
1990
IMDb Rating: 8
Rotten Tomatoes Rating:
Metacritic Rating: 72
Classification: PG-13
Days of Thunder
1990
IMDb Rating: 6.1
Rotten Tomatoes Rating:
Metacritic Rating: 60
Classification: PG-13
Desperate Hours
1990
IMDb Rating: 5.4
Rotten Tomatoes Rating:
Metacritic Rating: 33
Classification: R
DuckTales the Movie: Treasure of the Lost Lamp
1990
IMDb Rating: 6.8
Rotten Tomatoes Rating:
Metacritic Rating:
Classification: G
Edward Scissorhands
1990
Ernest Goes to Jail
1990
The Exorcist III
1990
Fire Birds
1990
Flashback
1990
Flatliners
1990
The Freshman
1990
Ghost
1990

4.2 Usability and Quality of Design

As the application was designed with simplicity in mind, the usability of the application is very high. The application is very easy to use and the user can navigate the application with ease. The application is layed out in a way that is very clean, the only page that can at all be considered cluttered is the LandingPage because of the carousel but even then the carousel is very clean, easy to use and it doesnt seems to distract the user from the rest of the page.

The navigation of the app is very simple and easy for the user to understand. As previously mentioned, by having the search bars within the navigation bar at the top of the page the user can search from anywhere in the application. The only part of the navigation that may confuse the user is the home button using the logo of the application. It may not be clear to the user that this is a button that leads them home. But, this is quite a common design choice in web design and there isnt much use for the user to return to the home page as the user can search from any page in the application.

The design of the application is consistent throughout. The navigation bar is the same on every page, very little colours are used throughout the application and the colours that are used are consistent. Only the standard font is used throughout the application except in the ag-grids where a seperate style is used.

4.3 Accessibility

The application has attempted to be as accessible as possible. The application achieve most of the checkpoints of the W3C Web Content Accessibility Guidelines 2.0. The application has achived the following checkpoints:

- 1.1.1 All non-text content such as images have alternative text.
- 1.2.1 Information conveyed in colour can also be conveyed in black and white.
- 1.6.1 Page can be read without CSS.
- 1.6.2 Text equivalents update when content changes.
- 1.7.1 The screen doesnt flicker.
- 1.14.1 The application uses the clearest language appropriate.

5 Technical Description

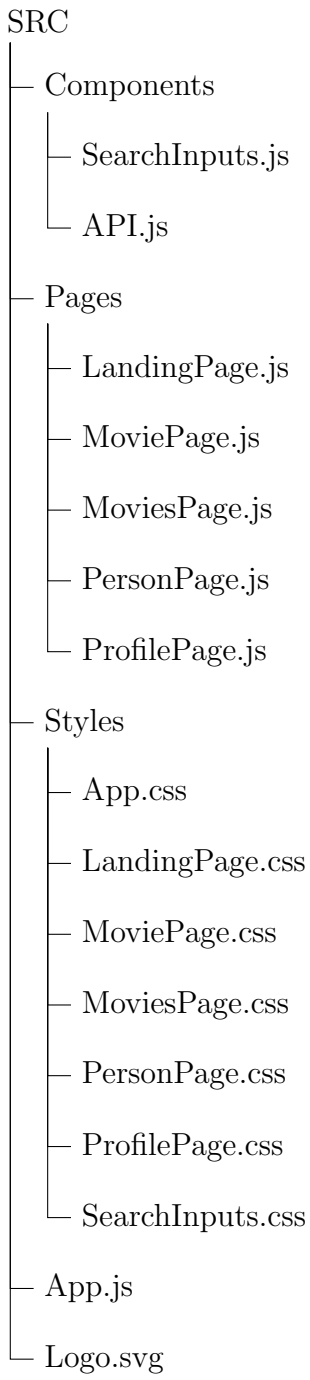
5.1 Architecture

- **Components:**
  This folder contains the components and the API handler for the application.
- **Pages:**
  This folder contains all the pages for the application.
- **Styles:**
  This folder contains all the CSS files for the application.

The diagram on the right shows the file structure of the application. As you can see, the application is structured in a very simple way. As previously mentioned, all the code for handling API requests is contained within the API.js file. This has made it easier to write the code for the application as it is all in one place and can be reused easily.

The only component of the application that has been seperated out into its own file is the search inputs. The only reason this was done was because I couldnt get it working within App.js.

Each page of the application has its own JS and CSS file. This was done to make the code easier to read and understand, and to cut down on the amount of code in each file.



5.2 Test Plan