**MovieBunkers**

MovieBunkers is a movie collection management application built using MERN stack, using this application users can store, mark as [seen, unseen, favourite, star] titles and retrieve movies and TV shows with various filters.

This project has three modules:

1. MovieBunkers Backend: This module is the backend part of the application.

Technologies Used: TypeScript, Express JS, MongoDB

GitHub URL: <https://github.com/charan379/moviebunkers>

API Documentation: <https://moviebunkers-api.onrender.com/api-docs/>

1. React4Movies Frontend: This module is the frontend part of the application.

Technologies Used: ReactJS, Next JS, Redux, HTML, CSS, JavaScript

GitHub URL: <https://github.com/charan379/react4movies>

1. Tmdb-api-wrapper: This module is a serverless functions deployed on AWS lambda, fetches data from TMDB API and converts fetched data into our required format.

Technologies Used: JavaScript, AWS Serverless functions.

GitHub URL: <https://github.com/charan379/tmdb-api-wrapper>

# Features

* Responsive layout
* Add movies to collection from tmdb
* Search movies
* Know where movie is available for streaming
* Play youtube Trailers
* Filters
  + filter by language
  + filter by genre
  + filter by movie/tv
  + filter by Age/Film Board Certification
  + filter by seen/unseen
  + filter by favourite
  + filter by starred
* Sort
  + Sort by year
  + Sort by added date
  + Sort by title name
  + Sort by release year
  + Sort by ratting
* Set movie as seen, unseen, star, favourite for individual user
* Toggle between light/dark themes
* Role/Level based access control
* Links to external site like IMDB, TMDB
* User registration and activation using OTP verification through email
* Multi user view
  + User
  + Moderator
  + Admin

## Keyboard Shortcuts

* CTRL+Q => Opens Sidebar, If sidebar already opened focus search field
* CTRL+D => Clears all filters and search query
* Esc => closes sidebar, movie modal

# How it works

Backend and frontend communication

**MovieBunkers**

(Application Backend)

**React4Movies**

(Application Frontend)

API Response

API Request

API Request

API Response

**TMDB API**

(TMDB API wrapper, retrieves data of movies and tv shows from the movie data open API)

(External API)

# Backend System Design

API Request (GET, POST, UPDATE, DELETE)

**Repository**

(Logic to Create, Read, Update, Delete) data in database

Request with authorization header

Validation Exception

**Authentication / Authorization**

(Middleware)

Data to be validated

Request Body / Data validation by JOI Schema

Method Call

**Utility / Helper**

Functions

Perform Action, / Return data / Throws Exception

Method Call

Method Call

Perform Action, / Return data / Throws Exception

Perform Action, / Return data / Throws Exception

Success API Response from controller

Returns validated data

Exception caught in controller

**CRUD Operations**

**Database**

API Response

No

Invalid Route Exception

Authentication / Authorization Exception

Yes

If valid route, request will be passed to controller class

Valid Route

**Route Controllers**

(Bridge b/w API route and service logic)

**Service**

(Business Logic to handle request)

Error Response

**Exception Handler** (Middleware)

# Authentication

Moviebunkers uses JWT ( JSON Web Tokens ) for authentication and authorization purpose, when a user sends an login request from frontend to backend with correct credentials then backend will responds with a unique JSON web Token which will used in authorization header in every request that is being sent from frontend.

Frontend

React4Movies

MoviesBunkers API

Backend

Response with JWT Token

Login Request

Every subsequent request from frontend will have authorization header with this bearer token

## User Access Levels

There are three user roles in moviebunkers application USER, MODERATOR, ADMIN using these three roles I have designed three access levels

**Level One Access**: Every authenticated user will have level one access i.e. USER,MODERATOR,ADMIN.

**Level Two Access:** Authenticated users with role MODERATOR, ADMIN will have level two access.

**Level Three Access:** Authenticated users with role ADMIN will have

Level One

Roles: User, Moderator, Admin

All Authenticated Users

Access: Read

Level Two

Roles: Moderator, Admin

Access: Read, Write

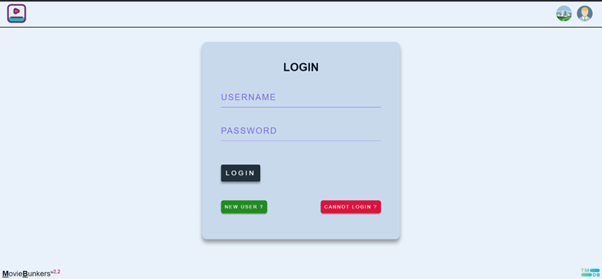
Level Three

Roles: Admin

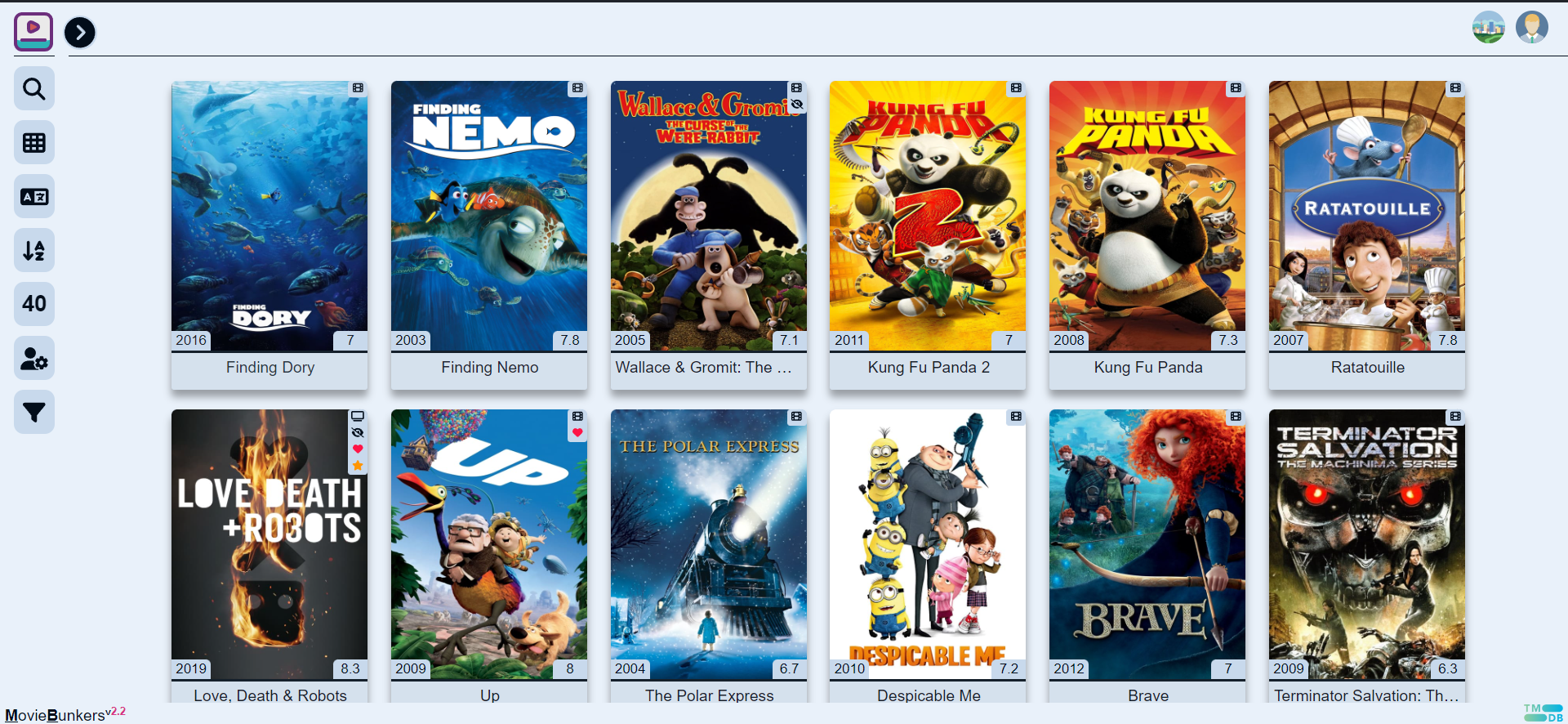
Access: Read, Write, Delete

# Application Frontend Screen Shots

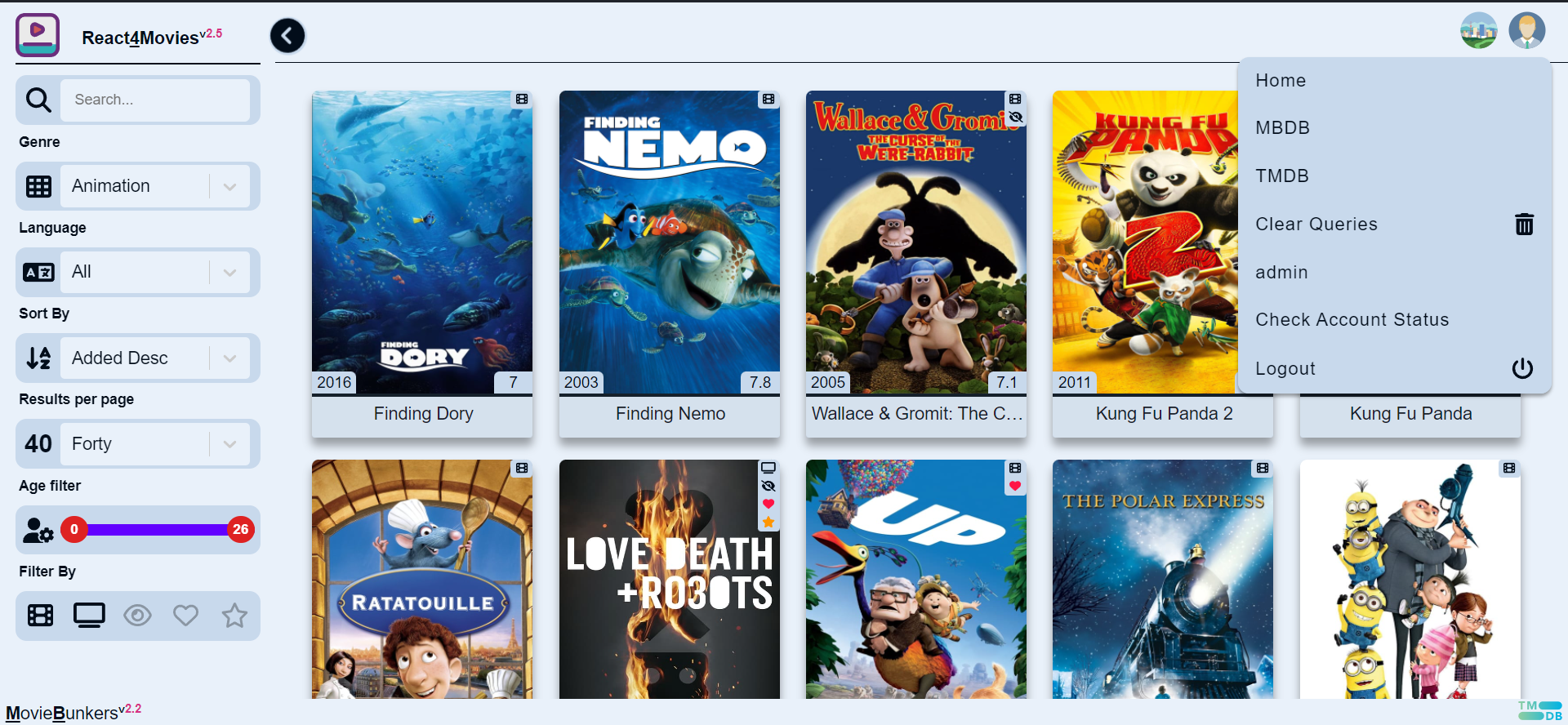
## Login Screen



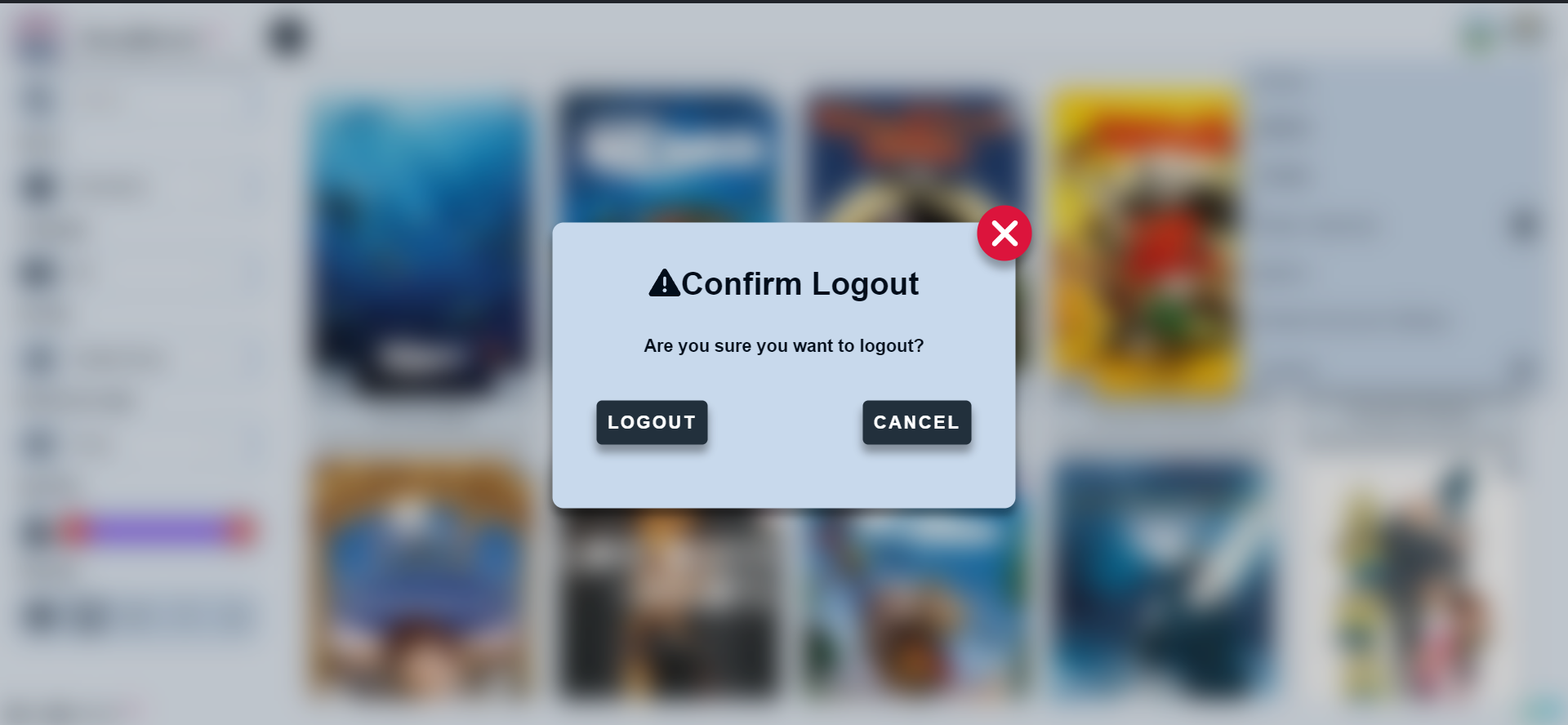
## Home Screen



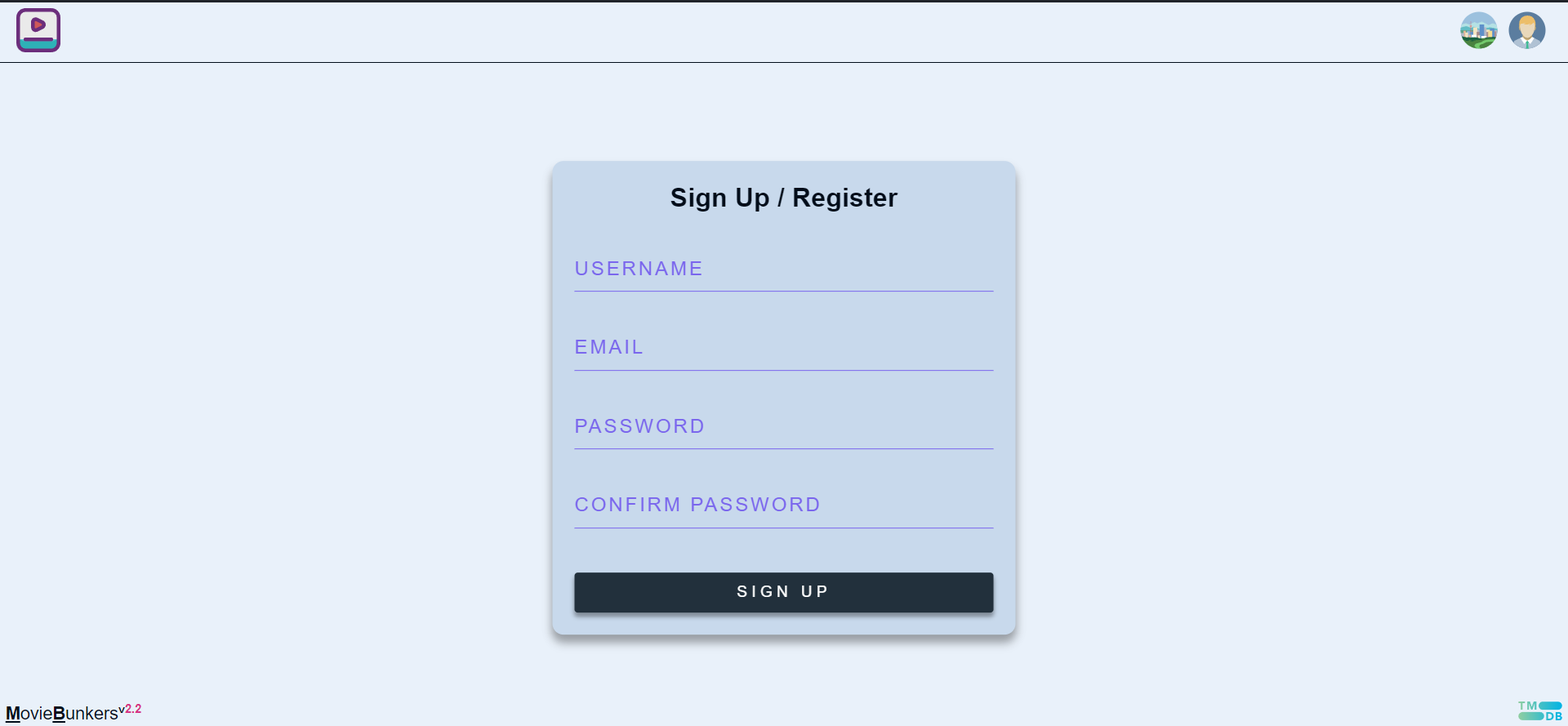
## Side Bar



## Logout



## Signup Form



## Password Reset Form

