Text-Em-All Hybrid React Engineer Code Challenge – Documentation

Name: Lahari Devaraju

Challenge:

Build and deploy a React application that meets the following criteria:

- Consume the OMDB web API's **"By Search"** functionality to display information about at least ten movies in a format like a list or grid.
- Consume the OMDB web API's **"By ID"** functionality to allow a user to click on a movie item to view the details.

Link for a live demo of the challenge: https://superlative-tulumba-fe1e84.netlify.app

Setup and running the code:

- Take a clone from the git using the git clone.
- Then install node modules using **npm install** or **npm i.**
- Then run the application using npm start.

Concepts Used:

To build Movies React application, I typically utilize key technologies and concepts, such as Typescript for strong typing, MUI (Material-UI) for designing the app's visual components, Axios for fetching data and leveraging React's states and props to manage component behavior. I also tried the write the code using typescript and kept these files in a different folder.

Documentation for the coding challenge:

- **Describing the project:**
  - I have developed a React application that satisfies the specified criteria. Upon loading the application, a grid containing ten movies retrieved from page one of the OMDB API is displayed, alongside a search field located at the top right corner of the page. When a user enters the name of a movie or show in the search field, the grid updates to show the relevant results. Clearing the search field causes the initial ten movies from the API to be displayed once more.
  - The application also allows users to click on individual movie cards to view additional details. Upon doing so, a pop-up or modal appears, displaying pertinent information about the movie, such as its name, release year, director, and genre. In order to build this application, I leveraged multiple React components, employed both state and props, and demonstrated my understanding of component lifecycle.

Workflow of the project:

- This is a React application that displays a list of movie cards with their title, year, and poster image fetched from the OMDB API based on a search query entered in a search bar.
- The App component imports the CardDesign component, which is responsible for rendering the list of movie cards. The CardList component defines the list of movies and provides search functionality by implementing TextField from the Material-UI library.

- The CardList component fetches the data from the OMDB API using Axios and stores it in the state. The render method of the component filters the movies based on the search query and maps them to individual MovieCard components.
- The MovieCard component defines the structure of an individual movie card and uses the Card and CardMedia components from the Material-UI library. It also has a ModalButton component that enables the user to view more information about the movie in a modal.

Describing the individual code components:

This is a TypeScript code written in ReactJS library which has four different components: App.tsx, CardList.tsx, and MovieCard.tsx, ModalButton.tsx.

- **App.tsx**: This component renders the CardDesign component. CardDesign is a child component that is imported from "./components-TypeScript/CardDesign". CardDesign component is not shared in the code.
- **CardList.tsx**: This component fetches movies from the OMDB API based on a search query, and it returns a list of MovieCard components. The component has three interfaces CardListProps, CardListState, and Movie. CardListProps define props for the component, CardListState defines the state for the component, and Movie defines the movie type that is returned from the OMDB API.
- The component uses React lifecycle methods componentDidMount() and componentDidUpdate() to fetch the movies from the OMDB API. componentDidUpdate() is called when the state changes, and it updates the list of movies. The component uses axios to make HTTP requests to the OMDB API.
- The component receives searchQuery state as input from the user and passes it as a parameter to the apiCall function. The apiCall function fetches the data from the OMDB API based on the searchQuery, and it updates the component's state with the response from the API.
- CardList component has a handleSearchQueryChange function, which updates the searchQuery state when the user types something in the search bar.
- The component has a Grid layout from Material UI library that is used to render the MovieCard component for each movie received from the OMDB API. The component passes the movie details to the MovieCard component as props, which is rendered by the MovieCard component. If there are no movies in the list, the component will not render anything.
- **MovieCard.tsx**: This component receives movie details as props and renders them as a Card layout from Material UI library. The component has two interfaces, Props and State. Props define the props for the component, and State define the state for the component.
- **ModalButton.tsx:** The ModalButton component is imported in the MovieCard component and it is used to render a button that triggers a modal. The modal is used to display more information about the movie, such as the plot and the ratings.
- The component has a Card layout from Material UI library, which has a CardHeader, CardMedia, CardContent, and ModalButton components. The component receives movie details as props and renders them in the CardHeader and CardContent components. ModalButton component is used to display more details about the movie when clicked. The component sets the state of isModalVisible to false by default, which gets updated when the ModalButton component is clicked.

Screenshots of the output:

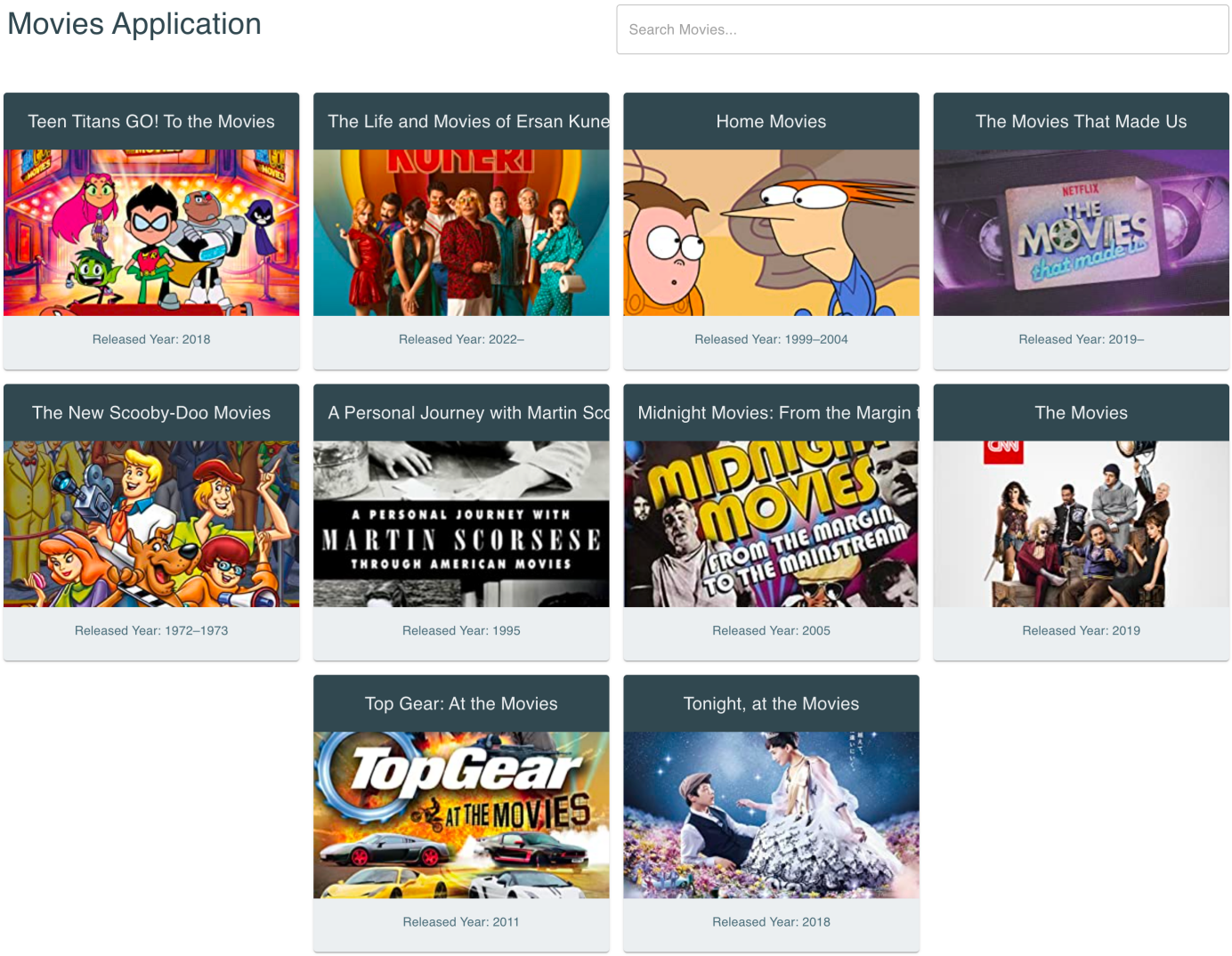

Fig 1: Initially Page

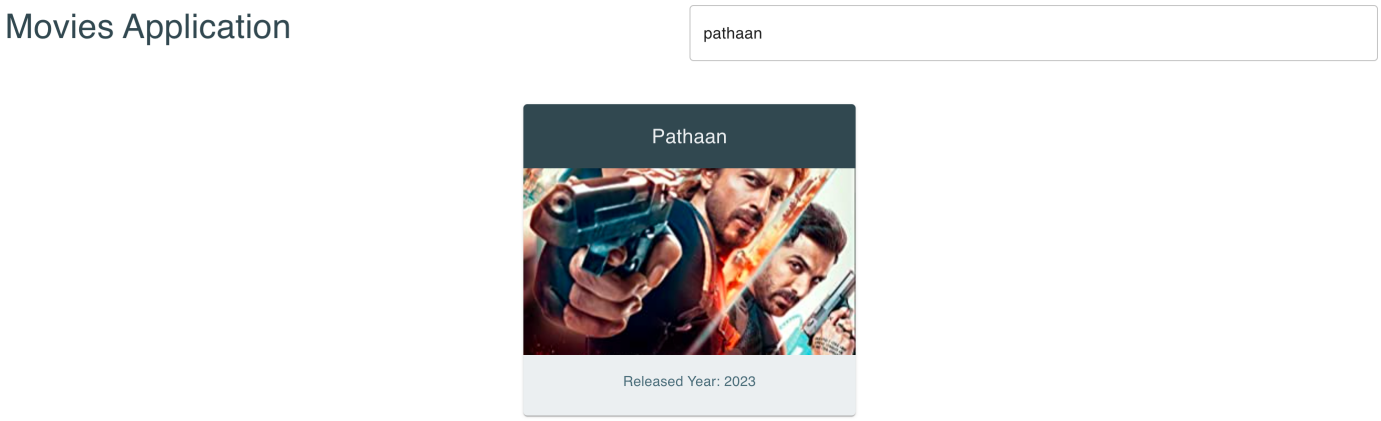

Fig 2: cards displayed on the search of a movie in the search field.

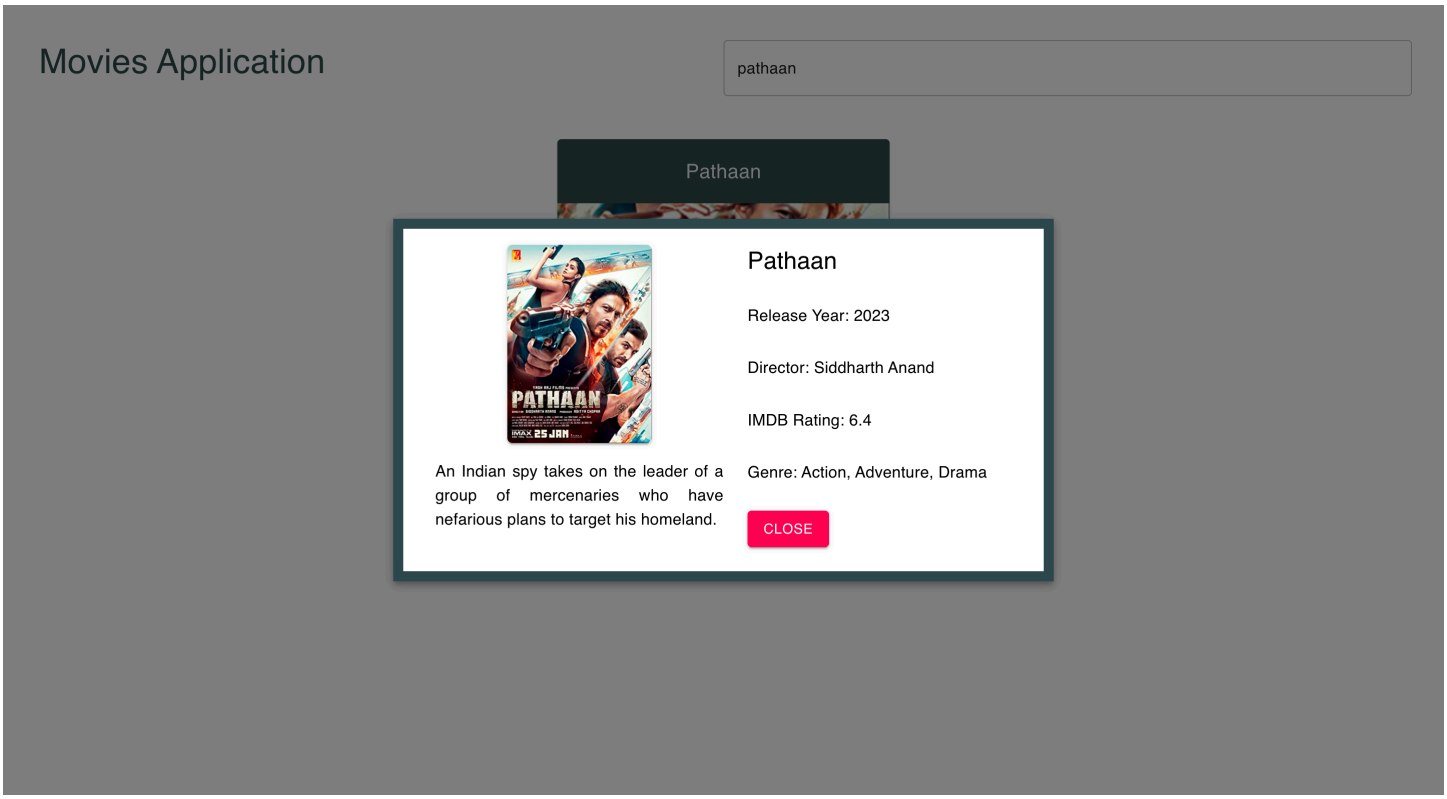

Fig 3: When the card is clicked, a modal will appear, displaying the movie details.

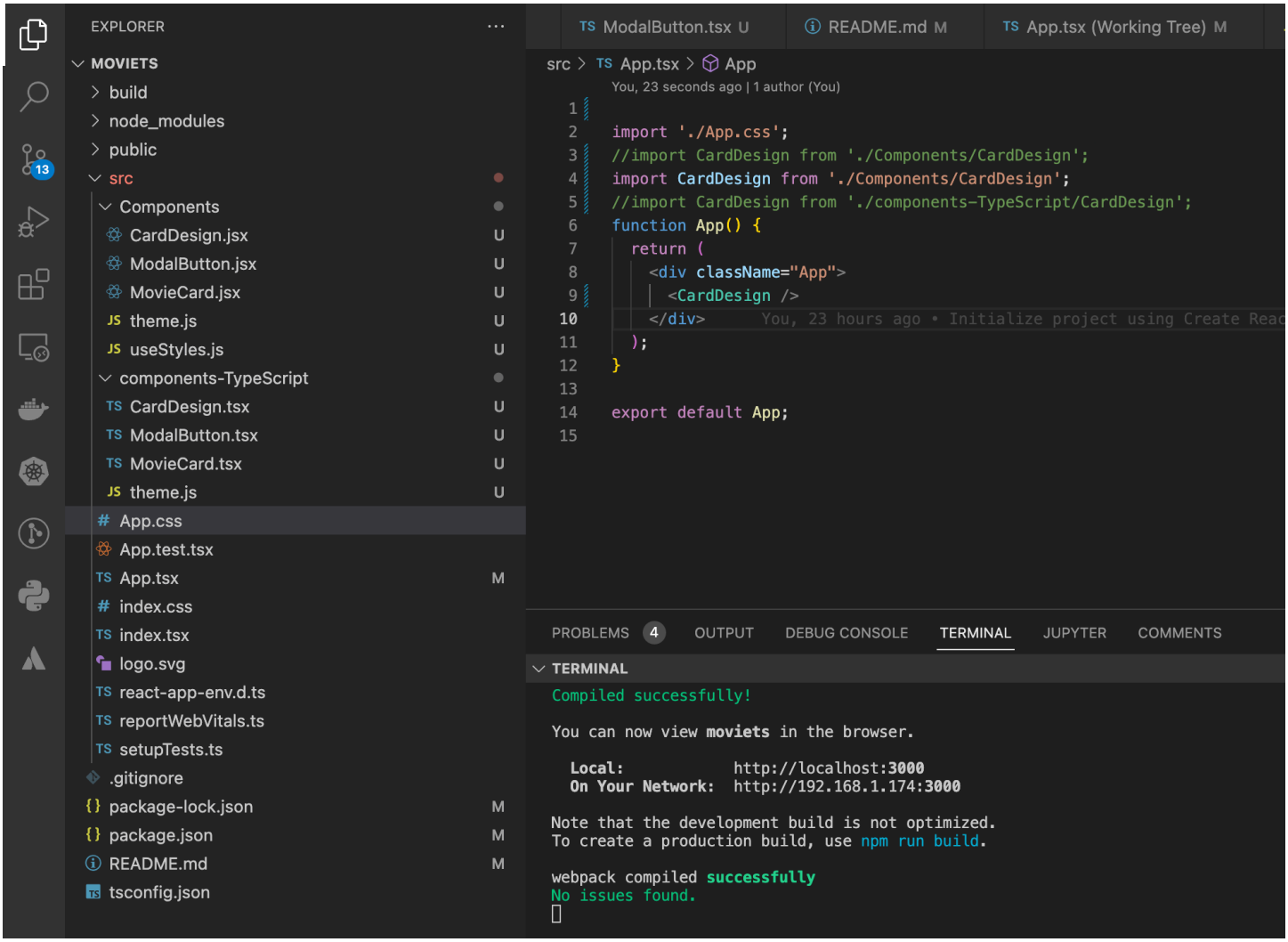

Fig: Folder structure of the application