Project Idea: Movie Reviews

Project Description:
A website that allows users to view a list of movies depending on various categories, search for a specific movie, add movies to favourites list.

The website will be created using React for the frontend and it will be using an API provided by TMDB to retrieve movies.

Project Requirements:
The requirements will be split into multiple levels.

Level 1 (Setting up the project **):**

- Create a repository on Github and initialise it with a README.md file
- Clone the repository
- Create a new react project inside of the repository with the name of client
- Run the React app to check if everything is working as expected
- Clean the App component from the unnecessary imports and JSX (you could render a head saying movie reviews)
- Install the following npm packages: axios, react-router-dom
  - Axios is used to send HTTP requests to retrieve the data we want
  - React-router-dom is used to handle the routing in the fronten
- Create a .env file to add to it the API key later
- Go to TMDB documentation and sign up then generate an API key to use when sending requests
- Add the API key as an environment variable inside the .env file

Level 2 (Create basic file structure **):**
- Create a components folder in the src directory
- Create and export the following components (for now just return the name of the component inside a header)
  - MoviesList component: This component will be used to map over an array of movies and display a grid on the screen
  - MovieCard component: This component will display the movie information in a card format, it is going to display the image, title and rating
  - MovieDetails component: This component will display all the movie details and it will be shown on the screen when going to the corresponding path in the URL

Level 3 (Routing setup **):**

- Import `BrowserRouter` from the correct package and wrap it around the App component to enable routing in the App
- Inside the App add the following routes
  - For path '/' render the MoviesList so it shows on the base route when opening the web application (you should be able to see the heading that you have added)
  - For path '/movies/:id' render the MoviesDetails component, it should be displayed instead of the MoviesList when switching the path in the URL to the corresponding path (you should be able to see the heading that you have added after you change the path)
- Inside of MoviesDetails import useParams so you could access the movies id inside the path
  - Add the id to the rendered JSX so you could view it to make sure that you have accessed the id correctly from the path

Level 4 (Retrieving data from the API **):**
- Inside of MoviesList send a GET request using axios to retrieve the latest movies from the API (make sure to add that in the useEffect so it runs after the first render for the component then when you set the state with the response date it will re-render the app)
- Map over the movies to return a new array of MovieCard component to display it on the screen (make sure to pass the movie object to MovieCard component)

Level 5 (building the UI 1 **)**
- In MovieCard make sure that you can access the received prop from the parent component (MoviesList)
- Create a card that represents the movie, it should have an image, movie title, and rating if available (make sure to style it with CSS)
- In MoviesList create a grid using css to display the movie cards in a grid format (the map returns the grid children so the style must be applied to the parent element)

Level 6 (more routing **)**
- Use useNavigate to navigate to /movies/:id when clicking on view details in the MovieCard component (keep in mind the :id is a url parameter so you would navigate to something like /movies/1 assuming that component's id was equal to one)
-