* **npx create-react-app netflix-gpt**

**# adding / Installing tailwindcss in project**

* + tailwindcss website> get started > framework guides > create react app
  + **Install Tailwind CSS :-** Install **tailwindcss** via npm, and then run the init command to generate your **tailwind.config.js** file.
    - **npm install -D tailwindcss**
    - **npx tailwindcss init**
  + **Configure your template paths :-** Add the paths to all of your template files in your **tailwind.config.js** file.
  + **Add the Tailwind directives to your CSS :-** Add the **@tailwind** directives for each of Tailwind’s layers to your **./src/index.css** file.
  + <https://tailwindcss.com/docs/guides/create-react-app>

**#Router Configuration**

* **npm i -D react-router-dom**
* appRouter = createBrowserRouter > path and element
* import RouterProvider 🡪 <RouterProvider router={appRouter}/>

**#Login Page / Sign-up Page**

* **Check Formik form validation also for big data validation**

**#Form Validation**

* **Regex for email & password validation**
* How can I validate an email address using a regular expression?
* <https://saturncloud.io/blog/how-can-i-validate-an-email-address-using-a-regular-expression/> 🡨 Email
* <https://regexr.com/3bfsi> 🡨 Password
* There is test function over regex and we want to test our email and password
* Validate User Full name
* 🡺 <https://regexr.com/3f8cm>

**export *const* checkValidData = (email, password) => {**

***// if this regex is pass then it will return TRUE inside it or FALSE inside "isEmailValid"***

***const* isEmailValid = /^([a-zA-Z0-9.\_%-]+@[a-zA-Z0-9.-]+.[a-zA-Z]{2,})$/.test(email);**

***// if this regex is pass then it will return TRUE inside it or FALSE inside "isPasswordValid"***

***const* isPasswordValid =/^(?=.\*\d)(?=.\*[a-z])(?=.\*[A-Z])(?=.\*[a-zA-Z]).{8,}$/.test(password);**

***// return's Error Message***

**if (!isEmailValid) return "Email ID is not valid";**

**if (!isPasswordValid) return "Password is not valid";**

***//if both are valid then return null i.e. No Error***

**return null;**

**};**

**useRef Hook**

* useRef is a React Hook that lets you reference a value that’s not needed for rendering.
* <https://react.dev/reference/react/useRef>
* <https://refine.dev/blog/react-useref-hook-and-ref/#introduction>

**#Authentication | Using Fire-base for Authentication**

* Using Google Fire base for Back-End
* Get Starte 🡺 Add Project 🡺 Project-Name 🡺 Default Account for Firebase
  + Get started by Adding Firebase to your web 🡺 Create a web-project
  + Register app
    - App-nickname / project name
    - ✔️ also set up Firebase Hosting for this app

* + Add Firebase SDK
    - npm install firebase
    - create firebase.js file in utils and paste that i.e. initialize firebase config
      * In your project go-to Project 🡺 overview 🡺 Enable Authentication
      * Get Started 🡺 sign in method 🡺 add sign in option 🡺 enable it
  + Install Firebase CLI
    - * To host your site with Firebase Hosting, you need the Firebase CLI (a command line tool).
      * Run the following [npm](https://www.npmjs.com/) command to install the CLI or update to the latest CLI version.
      * **npm install -g firebase-tools**
  + Deploy to firebase Hosting
    - You can deploy now or [later](https://firebase.google.com/docs/hosting/quickstart?hl=en&authuser=0). To deploy now, open a terminal window, then navigate to or create a root directory for your web app.
    - **Sign in to Google**
      * **firebase login**
    - **initiate your project**

Run this command from your app's root directory:

* + - * **firebase init**
    - **when you’re ready deploy your web app**

Put your static files (e.g., HTML, CSS, JS) in your app's deploy directory (the default is "public").

Then, run this command from your app's root directory:

* + - * **firebase deploy**
* After deploying, view your app at [netflixgpt-d91a2.web.app](https://netflixgpt-d91a2.web.app/)
* Need help? Check out the [Hosting docs](https://firebase.google.com/docs/hosting/quickstart?hl=en&authuser=0)

**#Firebase Documentation**

* <https://firebase.google.com/docs/auth/web/password-auth?hl=en&authuser=0>
* Firebase 🡺 Docs 🡺 Builds 🡺 Authentication – Web 🡺 Password Authentication
  + **Sign-Up** **a user with an email address and password**
    - Choose web Modular API
    - Create a password-based account 🡺 API – createUserWithEmailAndPassword **To create a user with email and password on firebase**
    - *import { getAuth,* ***createUserWithEmailAndPassword*** *} from "firebase/auth";*

**createUserWithEmailAndPassword(**

**auth,**

**email.current.value,**

**password.current.value**

**)**

* + **Sign-in** **a user with an email address and password**
    - *import { getAuth,* ***signInWithEmailAndPassword*** *} from "firebase/auth";*

**signInWithEmailAndPassword(**

**auth,**

**email.current.value,**

**password.current.value**

**)**

**#after building sign-in and signup feature**

* what will do now is try to push this user object we got / that user information we got we will put that info into our Redux Store

**#Redux Store for Sign in & Sign up**

* If the user sign up or a sign in we will got this user object and we will have to keep the user object with us because we need this user object anywhere in our app.
* So what we will do is as soon as the user sign in or sign up We will just add all that data to our redux store i.e. once user sign in or sign up we will add user to our redux store.
* **npm i -D @reduxjs/toolkit**
* **npm i react-redux**
  + utils > appStore.js 🡪 configureStore – reducer collection of different reducers
  + utils > userSlice.js 🡪 createSlice – name, initialState – reducers{add or remove} function
  + **userSlice**
* **import { createSlice } from "@reduxjs/toolkit";**
* ***const* userSlice = createSlice({**
* **name: "user",**
* **initialState: null,**
* **reducers: {**
* **addUser: (state, action) => {**
* **return action.payload**
* **},**
* **removeUser: (state, action) => {**
* **return null**
* **}**
* **}**
* **})**
* **export *const* { addUser, removeUser } = userSlice.actions**
* **export default userSlice.reducer**
  + **appStore**
* import { configureStore } from "@reduxjs/toolkit";
* import userReducer from "./userSlice"
* *const* appStore = configureStore({
* reducer: {
* user: userReducer
* }
* })
* **export default appStore**
  + **now providing Store to our App**
* *function* App() {
* return (
* <Provider *store*={appStore}>
* <Body />
* </Provider>
* );
* }
* Now what we will do next thing is as soon as my user sign in or sign up I want to update my store I want to update my userSlice with that user Information
* I will have to dispatch an action whenever user sign in and sign out, So instead of doing that dispatch again and again here we will use a utility which is given to us by firebase which is known as **onAuthStateChange**
* **Manage Users in Firebase** ~~(not using redux dispatch)~~ - **onAuthStateChange**
* <https://firebase.google.com/docs/auth/web/manage-users?hl=en>
* Firebase 🡺 Docs 🡺 Builds 🡺 Authentication – Web 🡺 Manage User
* Firebase gives us an amazing API that is known as onAuthStateChange this API is called whenever the user sign in, whenever the user sign up, whenever the user sign out and whenever there is an authentication state change/ Authentication happens. And you what to do on this auth change if you want to handle that auth then use this onAuthStateChange API. This is kind of like an Event Listener
* So where we add this code?
* You can write wherever you want to but mostly use Root level to write this code so I am writing it on Body component
* **#Introduction to GPT**
* We are going to build a separate page it is kind of like a Netflix GPT search page
* We will have search bar and suppose sometimes we don’t know what or which movie to watch right! and I just want to type something into the search bar and want that GPT APIs should give me the results i.e. what movie to watch. Its kind of like using GPT as movie recommendation

**GPT API**

* <https://platform.openai.com/> >> API Keys >>
* NetflixGPTProject API KEY ==> sk-proj-TmTn0rsr28VWhSYErR20dAT3BlbkFJI0NzwPB0A4mcjxiw17XY
* Install Library => **OpenAI Node API Library**
* **npm install –-save openai**

GPT API