# Connecting React JS application with node JS

## React JS:

React JS is an open-source JavaScript library for creating single-page applications with a focus on user interfaces. It has a strong developer community and is extensively used for creating SPAs (Single Page Applications).

## Node JS:

Because of its single-threaded nature, NodeJS is best suited for non-blocking, event-driven servers. It was created with real-time, push-based architectures in mind and is utilized for traditional websites and back-end API applications.

We'll learn how to use NodeJS as a backend and ReactJS as a frontend in this tutorial. Let’s started with any further delay.

## Initializing Project:

1. Create an empty directory, name it as fun-app
2. Now create another directory, and name it as client
3. Again, create directory name it as server.

A screenshot of a computer

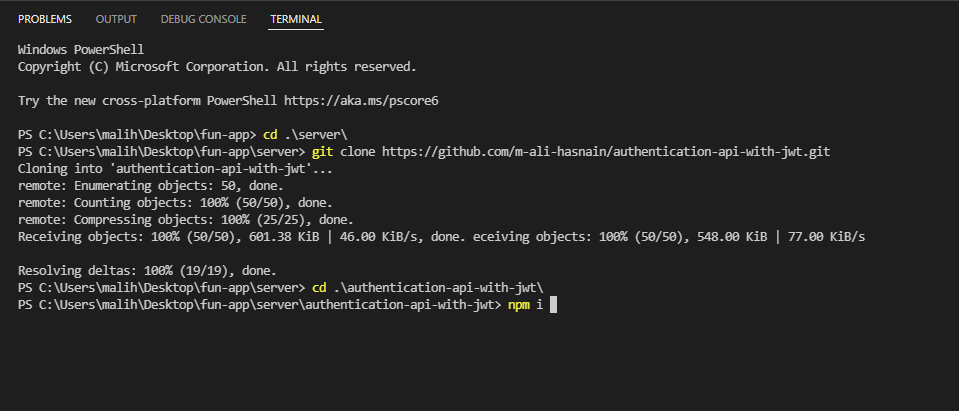
Description automatically generated with medium confidence

1. Open it with editor of your choice. I would like to prefer [VS CODE](https://code.visualstudio.com/)
2. Open terminal by pressing ***ctrl+shift+~***
3. Type ***cd client*** and hit enter. Now, you are in client dir
4. Type ***npx create-react-app ./*** to create react app with same name as “client”

Graphical user interface, text

Description automatically generated

1. When app is created, type ***npm install axios*** and hit enter
2. Now, go back to root directory by typing ***cd ../***
3. Now, type ***cd server***
4. For server, I am not going to create application from scratch instead I will clone repository from GitHub. This repository is basically users api which performs signup, login and authentication with jwt
5. Type ***git clone*** [***https://github.com/m-ali-hasnain/authentication-api-with-jwt.git***](https://github.com/m-ali-hasnain/authentication-api-with-jwt.git)and hit enter
6. again type ***cd authentication-api-with-jwt*** and hit enter
7. Now type ***npm install*** and hit enter.



1. You will need to create .env file under authentication-api-with-jwt directory and put following content in it

PORT = 5000

MONGO\_URI = your mongo db url here

JSON\_WEB\_TOKEN\_SECRET=pain is gain

Graphical user interface, text

Description automatically generated

1. Type ***npm start*** and hit enter in authentication-api-with-jwt directory. Your server will start.
2. Now go to package.json file in client and add “proxy” field like this

Text

Description automatically generated

1. Go to client folder and then src folder and create a folder component. Then inside components create following files
   1. Login.jsx
   2. Signup.jsx
   3. Welcome.jsx

A screenshot of a computer

Description automatically generated with medium confidence

1. Now in signup.jsx put following code

import React from "react";

import axios from "axios";

const Signup = () => {

  const [userName, setUserName] = React.useState("");

  const [userEmail, setUserEmail] = React.useState("");

  const [userPassword, setUserPassword] = React.useState("");

  const register = async (event) => {

    event.preventDefault();

    //now we will send request to backend

    if (!userName && !userEmail && !userPassword) {

      alert("please fill out all details");

    } else {

      var userDetails = {

        userName,

        userEmail,

        userPassword,

      };

      try {

        const { data } = await axios.post("/users/register", userDetails);

        alert(`${data.userName} you are successfully registered`);

        //storing token in localstorage

        localStorage.setItem("token", JSON.stringify(data.token));

      } catch (error) {

        console.log(error);

      }

    }

  };

  return (

    <div>

      <form>

        <label htmlFor="userName">User Name</label>

        <input

          type="text"

          id="userName"

          value={userName}

          onChange={(e) => {

            setUserName(e.target.value);

          }}

        />

        <br />

        <label htmlFor="userEmail">User Email:</label>

        <input

          type="email"

          id="userEmail"

          required

          value={userEmail}

          onChange={(e) => {

            setUserEmail(e.target.value);

          }}

        />

        <br />

        <label htmlFor="userPassword">User Password</label>

        <input

          type="password"

          id="userPassword"

          value={userPassword}

          onChange={(e) => {

            setUserPassword(e.target.value);

          }}

        />

        <br />

        <input type="submit" value="signup" onClick={register} />

      </form>

    </div>

  );

};

export default Signup;

And in login.jsx put the following code

import React from "react";

import axios from "axios";

import { useNavigate } from "react-router-dom";

const Login = () => {

  const [userEmail, setUserEmail] = React.useState("");

  const [userPassword, setUserPassword] = React.useState("");

  const navigate = useNavigate();

  const login = async (event) => {

    event.preventDefault();

    //now we will send request to backend

    if (!userEmail && !userPassword) {

      alert("Please enter all fields");

    } else {

      try {

        const { data } = await axios.post("/users/login", {

          userEmail: userEmail,

          userPassword: userPassword,

        });

        console.log(data);

        localStorage.clear();

        //storing token in localstorage

        localStorage.setItem("token", JSON.stringify(data.token));

        //redirecting to welcome page

        navigate("/welcome", { state: { userEmail: data.userEmail } });

      } catch (error) {

        console.log(error);

      }

    }

  };

  return (

    <div>

      <form>

        <label htmlFor="userEmail">User Email:</label>

        <input

          type="email"

          id="userEmail"

          required

          value={userEmail}

          onChange={(e) => {

            setUserEmail(e.target.value);

          }}

        />

        <br />

        <label htmlFor="userPassword">User Password</label>

        <input

          type="password"

          id="userPassword"

          value={userPassword}

          onChange={(e) => {

            setUserPassword(e.target.value);

          }}

        />

        <br />

        <input type="submit" value="Login" onClick={login} />

      </form>

    </div>

  );

};

export default Login;

In Welcome.js put following code

import React from "react";

import { useLocation } from "react-router-dom";

const Welcome = () => {

  const { state } = useLocation();

  const { userEmail } = state;

  return <div>Welcome {userEmail}</div>;

};

export default Welcome;

In App.js put following Code:

import logo from "./logo.svg";

import "./App.css";

import React from "react";

import { Routes, Route } from "react-router-dom";

import Login from "./components/Login";

import Signup from "./components/Signup";

import Welcome from "./components/Welcome";

function App() {

  return (

    <div className="App">

      <Routes>

        <Route path="login" exact element={<Login />}></Route>

        <Route path="signup" exact element={<Signup />}></Route>

        <Route path="/welcome" exact element={<Welcome />}></Route>

      </Routes>

      React-App

    </div>

  );

}

export default App;

In Index.js Put following Code:

import React from "react";

import ReactDOM from "react-dom";

import "./index.css";

import App from "./App";

import reportWebVitals from "./reportWebVitals";

import { BrowserRouter as Router } from "react-router-dom";

ReactDOM.render(

  <React.StrictMode>

    <Router>

      <App />

    </Router>

  </React.StrictMode>,

  document.getElementById("root")

);

// If you want to start measuring performance in your app, pass a function

// to log results (for example: reportWebVitals(console.log))

// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals

reportWebVitals();

Let’s Try Now:

A picture containing table

Description automatically generated

Graphical user interface, application

Description automatically generated Graphical user interface

Description automatically generated with medium confidence

Graphical user interface, application

Description automatically generated Graphical user interface, application

Description automatically generated

Cool😊 ♥, Our frontend application is now connected to backend. I hope you enojyed this tutorial and stay tuned for more.