**React Setup**

### Step 1: Install Node.js and npm

You can download and install Node.js from the official website: [Node.js Downloads](https://nodejs.org/)

### Step 2: Create a React App

Open your terminal and run the following command to create a new React app using Create React App:

npx create-react-app your-app-name

Replace your-app-name with the desired name for your React application.

### Step 3: Navigate to the Project Directory

Change into the newly created project directory:

cd your-app-name

### Step 4: Start the Development Server

Start the development server to see your React app in action:

npm start

This command will open a new browser window/tab with your React app running at http://localhost:3000/.

### Step 5: Explore the Project Structure

The Create React App setup includes a predefined project structure. Key directories include:

* public: Contains the HTML file and assets that do not need to go through webpack.
* src: Contains your React application's source code.
* App.js: The main component.
* index.js: The entry point for your application.

### Step 6: Make Changes

Open the project in your preferred code editor and start making changes to the src directory, particularly App.js. As you save changes, the browser should automatically update with the latest version of your app.

### Step 7: Learn and Build

Explore the React documentation to learn more about React and how to build components, handle state, and manage your application's structure.

That's it! You've successfully set up a basic React app. Customize and extend it based on your project requirements.

### Task

### Task 1

Show the HTML Element in React

i)Table

ii)Form

iii)Form Elements

### Task 2

Add two numbers using html form input elements and button element

1)Enter only number value in input element

2)when click Add Button need to show the result after the Button

### Task 3:

Display the JSON Structure in HTML Table Format using react map

[

{

“name”:”Name1”,

“department”:”Engg”,

“dob”:”18/12/2000”

},

{

“name”:”Name2”,

“department”:”Engg”,

“dob”:”18/12/2000”

},

{

“name”:”Name3”,

“department”:”Engg”,

“dob”:”18/12/2000”

},

{

“name”:”Name4”,

“department”:”Engg”,

“dob”:”18/12/2000”

},

{

“name”:”Name5”,

“department”:”Engg”,

“dob”:”18/12/2000”

}

]

### Task 4:

Create one textbox and button when click the button

Call below api and and pass the q params and display output in page

const settings = {

async: true,

crossDomain: true,

url: 'https://weatherapi-com.p.rapidapi.com/current.json?q=London',

method: 'GET',

headers: {

'X-RapidAPI-Key': 'ea1736c136msh0f5c187e704eedbp15d852jsn9c8eb45d2ffd',

'X-RapidAPI-Host': 'weatherapi-com.p.rapidapi.com'

}

};

### Task 5:

react.js login and signup page using functional component

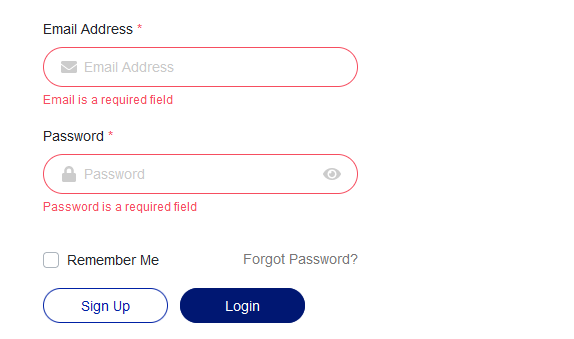
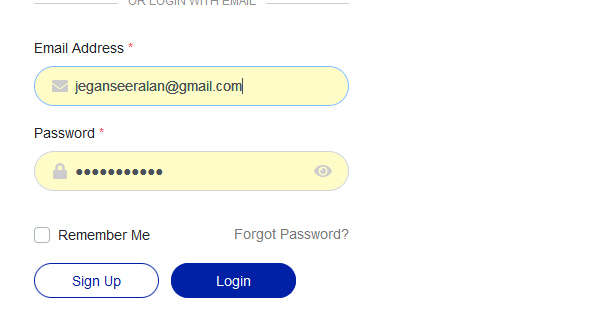
Validate all necessary validation

API URL : <https://dev-qualdo.eastus.cloudapp.azure.com/iam/login>

API METHOD:POST

API PARAMS:{"email":[MAILID@GMAIL.COM](mailto:MAILID@GMAIL.COM)","password":""}

HTML OUTPUT FORMAT



Task 6:

react.js login and signup page using class component

Validate all necessary validation

API URL : <https://dev-qualdo.eastus.cloudapp.azure.com/iam/login>

API METHOD:POST

API PARAMS:{"email":[MAILID@GMAIL.COM](mailto:MAILID@GMAIL.COM)","password":""}

HTML OUTPUT FORMAT

SAME AS ABOVE

Task 7:

React.js once successful login navigate to dashboard page add logout button in dashboard page, when click logout button navigate to login page, without login user cant see dashboard page - **class component**