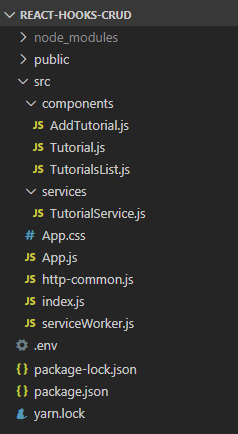
**Technology**

* React 17/16
* react-router-dom 6.2.2
* axios 0.26.1
* bootstrap 4.6.0

**Project Structure**

Now look at the project directory structure:



Let me explain it briefly.

– **package.json** contains 4 main modules: react, react-router-dom, axios & bootstrap.  
– App is the container that has Router & navbar.  
– There are 3 items using React hooks: TutorialsList, Tutorial, AddTutorial.  
– **http-common.js** initializes axios with HTTP base Url and headers.  
– TutorialDataService has functions for sending HTTP requests to the Apis.  
– **.env** configures *port* for this React Hooks CRUD App.

**Setup React.js Project**

Open cmd at the folder you want to save Project folder, run command:  
npx create-react-app react-hooks-crud

After the process is done. We create additional folders and files like the following tree:

 public

 src

 components

 AddTutorial.js

 TUtorial.js

 TutorialsList.js

 services

 TutorialService.js

 App.css

 App.js

 index.js

 package.json

**Install Bootstrap for React Hooks CRUD App**

Run command: npm install bootstrap.

Open **src**/*App.js* and modify the code inside it as following-

import React from "react";

import "bootstrap/dist/css/bootstrap.min.css";

function App() {

return (

// ...

);

}

export default App;

**Add React Router to React Hooks CRUD App**

– Run the command: npm install react-router-dom.  
– Open **src**/*index.js* and wrap App component by BrowserRouter object.

import React from "react";

import ReactDOM from "react-dom";

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

import App from "./App";

import \* as serviceWorker from "./serviceWorker";

ReactDOM.render(

<BrowserRouter>

<App />

</BrowserRouter>,

document.getElementById("root")

);

serviceWorker.unregister();

**Add Navbar to React Hooks CRUD App**

Open **src**/*App.js*, this App component is the root container for our application, it will contain a navbar, and also, a Routes object with several Route. Each Route points to a React Component.

import React from "react";

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

import "bootstrap/dist/css/bootstrap.min.css";

import "./App.css";

import AddTutorial from "./components/AddTutorial";

import Tutorial from "./components/Tutorial";

import TutorialsList from "./components/TutorialsList";

function App() {

return (

<div>

<nav className="navbar navbar-expand navbar-dark bg-dark">

<a href="/tutorials" className="navbar-brand">

bezKoder

</a>

<div className="navbar-nav mr-auto">

<li className="nav-item">

<Link to={"/tutorials"} className="nav-link">

Tutorials

</Link>

</li>

<li className="nav-item">

<Link to={"/add"} className="nav-link">

Add

</Link>

</li>

</div>

</nav>

<div className="container mt-3">

<Routes>

<Route path="/" element={<TutorialsList/>} />

<Route path="/tutorials" element={<TutorialsList/>} />

<Route path="/add" element={<AddTutorial/>} />

<Route path="/tutorials/:id" element={<Tutorial/>} />

</Routes>

</div>

</div>

);

}

export default App;

**Initialize Axios for React CRUD HTTP Client**

Let’s install *axios* with command: npm install axios.  
Under **src** folder, we create *http-common.js* file with following code:

import axios from "axios";

export default axios.create({

baseURL: "http://localhost:8080/api",

headers: {

"Content-type": "application/json"

}

});

You can change the baseURL that depends on REST APIs url that your Server configures.

For more details about ways to use Axios, please visit:  
[Axios request: Get/Post/Put/Delete example](https://www.bezkoder.com/axios-request/)

**Create Data Service**

In this step, we’re gonna create a service that uses axios object above to send HTTP requests.  
The service exports CRUD functions and finder method:

* CREATE: create
* RETRIEVE: getAll, get
* UPDATE: update
* DELETE: remove, removeAll
* FINDER: findByTitle

**services**/*TutorialService.js*

import http from "../http-common";

const getAll = () => {

return http.get("/tutorials");

};

const get = id => {

return http.get(`/tutorials/${id}`);

};

const create = data => {

return http.post("/tutorials", data);

};

const update = (id, data) => {

return http.put(`/tutorials/${id}`, data);

};

const remove = id => {

return http.delete(`/tutorials/${id}`);

};

const removeAll = () => {

return http.delete(`/tutorials`);

};

const findByTitle = title => {

return http.get(`/tutorials?title=${title}`);

};

const TutorialService = {

getAll,

get,

create,

update,

remove,

removeAll,

findByTitle

};

export default TutorialService;

We call axios (imported as http) get, post, put, delete method corresponding to HTTP Requests: GET, POST, PUT, DELETE to make CRUD Operations.

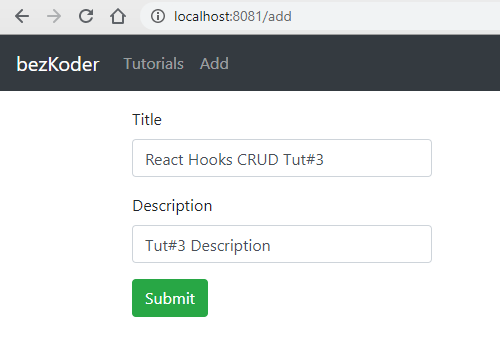
You can simplify import statement with:  
[Absolute Import in React](https://www.bezkoder.com/absolute-import-react/)

**Create React Components**

Now we’re gonna build 3 components corresponding to 3 Routes defined before.

**Add Object**

This component has a Form to submit new Tutorial with 2 fields: title & description.



**components**/*AddTutorial.js*

import React, { useState } from "react";

import TutorialDataService from "../services/TutorialService";

const AddTutorial = () => {

const initialTutorialState = {

id: null,

title: "",

description: "",

published: false

};

const [tutorial, setTutorial] = useState(initialTutorialState);

const [submitted, setSubmitted] = useState(false);

const handleInputChange = event => {

const { name, value } = event.target;

setTutorial({ ...tutorial, [name]: value });

};

const saveTutorial = () => {

var data = {

title: tutorial.title,

description: tutorial.description

};

TutorialDataService.create(data)

.then(response => {

setTutorial({

id: response.data.id,

title: response.data.title,

description: response.data.description,

published: response.data.published

});

setSubmitted(true);

console.log(response.data);

})

.catch(e => {

console.log(e);

});

};

const newTutorial = () => {

setTutorial(initialTutorialState);

setSubmitted(false);

};

return (

// ...

);

};

export default AddTutorial;

First, we define and set initial state: tutorial & submitted.

Next, we create handleInputChange() function to track the values of the input and set that state for changes. We also have a function to get tutorial state and send the POST request to the Web API. It calls TutorialDataService.create() method.

For return, we check the submitted state, if it is true, we show **Add** button for creating new Tutorial again. Otherwise, a Form with **Submit** button will display.

const AddTutorial = () => {

...

return (

<div className="submit-form">

{submitted ? (

<div>

<h4>You submitted successfully!</h4>

<button className="btn btn-success" onClick={newTutorial}>

Add

</button>

</div>

) : (

<div>

<div className="form-group">

<label htmlFor="title">Title</label>

<input

type="text"

className="form-control"

id="title"

required

value={tutorial.title}

onChange={handleInputChange}

name="title"

/>

</div>

<div className="form-group">

<label htmlFor="description">Description</label>

<input

type="text"

className="form-control"

id="description"

required

value={tutorial.description}

onChange={handleInputChange}

name="description"

/>

</div>

<button onClick={saveTutorial} className="btn btn-success">

Submit

</button>

</div>

)}

</div>

);

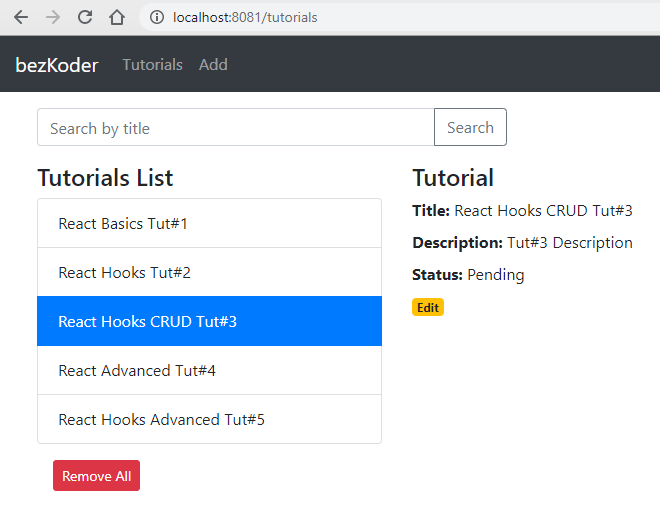
};

export default AddTutorial;

**List of Objects Component**

There will be:

* a search bar for finding Tutorials by *title*.
* a *tutorials* array displayed as a list on the left.
* a selected Tutorial which is shown on the right.



So we will have following state:

* searchTitle
* tutorials
* currentTutorial and currentIndex

We also need to use 3 TutorialDataService functions:

* getAll()
* removeAll()
* findByTitle()

We’re gonna use the Effect Hook: useEffect() to fetch the data from the Web API. This Hook tells React that the component needs to do something after render or performing the DOM updates. In this effect, we perform data fetching from API.

**components**/*TutorialsList.js*

import React, { useState, useEffect } from "react";

import TutorialDataService from "../services/TutorialService";

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

const TutorialsList = () => {

const [tutorials, setTutorials] = useState([]);

const [currentTutorial, setCurrentTutorial] = useState(null);

const [currentIndex, setCurrentIndex] = useState(-1);

const [searchTitle, setSearchTitle] = useState("");

useEffect(() => {

retrieveTutorials();

}, []);

const onChangeSearchTitle = e => {

const searchTitle = e.target.value;

setSearchTitle(searchTitle);

};

const retrieveTutorials = () => {

TutorialDataService.getAll()

.then(response => {

setTutorials(response.data);

console.log(response.data);

})

.catch(e => {

console.log(e);

});

};

const refreshList = () => {

retrieveTutorials();

setCurrentTutorial(null);

setCurrentIndex(-1);

};

const setActiveTutorial = (tutorial, index) => {

setCurrentTutorial(tutorial);

setCurrentIndex(index);

};

const removeAllTutorials = () => {

TutorialDataService.removeAll()

.then(response => {

console.log(response.data);

refreshList();

})

.catch(e => {

console.log(e);

});

};

const findByTitle = () => {

TutorialDataService.findByTitle(searchTitle)

.then(response => {

setTutorials(response.data);

console.log(response.data);

})

.catch(e => {

console.log(e);

});

};

return (

// ...

);

};

export default TutorialsList;

Let’s continue to implement UI elements:

...

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

const TutorialsList = () => {

...

return (

<div className="list row">

<div className="col-md-8">

<div className="input-group mb-3">

<input

type="text"

className="form-control"

placeholder="Search by title"

value={searchTitle}

onChange={onChangeSearchTitle}

/>

<div className="input-group-append">

<button

className="btn btn-outline-secondary"

type="button"

onClick={findByTitle}

>

Search

</button>

</div>

</div>

</div>

<div className="col-md-6">

<h4>Tutorials List</h4>

<ul className="list-group">

{tutorials &&

tutorials.map((tutorial, index) => (

<li

className={

"list-group-item " + (index === currentIndex ? "active" : "")

}

onClick={() => setActiveTutorial(tutorial, index)}

key={index}

>

{tutorial.title}

</li>

))}

</ul>

<button

className="m-3 btn btn-sm btn-danger"

onClick={removeAllTutorials}

>

Remove All

</button>

</div>

<div className="col-md-6">

{currentTutorial ? (

<div>

<h4>Tutorial</h4>

<div>

<label>

<strong>Title:</strong>

</label>{" "}

{currentTutorial.title}

</div>

<div>

<label>

<strong>Description:</strong>

</label>{" "}

{currentTutorial.description}

</div>

<div>

<label>

<strong>Status:</strong>

</label>{" "}

{currentTutorial.published ? "Published" : "Pending"}

</div>

<Link

to={"/tutorials/" + currentTutorial.id}

className="badge badge-warning"

>

Edit

</Link>

</div>

) : (

<div>

<br />

<p>Please click on a Tutorial...</p>

</div>

)}

</div>

</div>

);

};

export default TutorialsList;

If you click on **Edit** button of any Tutorial, the app will direct you to *Tutorial* page.  
We use React Router Link for accessing that page with url: /tutorials/:id.

You can add Pagination to this Page, just follow instruction in the post:  
[React Pagination using Hooks example](https://bezkoder.com/react-pagination-hooks/)

**Object details Component**

For getting data & update, delete the Tutorial, this component will use 3 TutorialDataService functions:

* get()
* update()
* remove()

We also use the Effect Hook useEffect() to get Tutorial by id in the URL (with the help of useParams() hook).

**components**/*Tutorial.js*

import React, { useState, useEffect } from "react";

import { useParams, useNavigate } from 'react-router-dom';

import TutorialDataService from "../services/TutorialService";

const Tutorial = props => {

const { id }= useParams();

let navigate = useNavigate();

const initialTutorialState = {

id: null,

title: "",

description: "",

published: false

};

const [currentTutorial, setCurrentTutorial] = useState(initialTutorialState);

const [message, setMessage] = useState("");

const getTutorial = id => {

TutorialDataService.get(id)

.then(response => {

setCurrentTutorial(response.data);

console.log(response.data);

})

.catch(e => {

console.log(e);

});

};

useEffect(() => {

if (id)

getTutorial(id);

}, [id]);

const handleInputChange = event => {

const { name, value } = event.target;

setCurrentTutorial({ ...currentTutorial, [name]: value });

};

const updatePublished = status => {

var data = {

id: currentTutorial.id,

title: currentTutorial.title,

description: currentTutorial.description,

published: status

};

TutorialDataService.update(currentTutorial.id, data)

.then(response => {

setCurrentTutorial({ ...currentTutorial, published: status });

console.log(response.data);

})

.catch(e => {

console.log(e);

});

};

const updateTutorial = () => {

TutorialDataService.update(currentTutorial.id, currentTutorial)

.then(response => {

console.log(response.data);

setMessage("The tutorial was updated successfully!");

})

.catch(e => {

console.log(e);

});

};

const deleteTutorial = () => {

TutorialDataService.remove(currentTutorial.id)

.then(response => {

console.log(response.data);

navigate("/tutorials");

})

.catch(e => {

console.log(e);

});

};

return (

// ...

);

};

export default Tutorial;

And this is the code inside return:

const Tutorial = props => {

...

return (

<div>

{currentTutorial ? (

<div className="edit-form">

<h4>Tutorial</h4>

<form>

<div className="form-group">

<label htmlFor="title">Title</label>

<input

type="text"

className="form-control"

id="title"

name="title"

value={currentTutorial.title}

onChange={handleInputChange}

/>

</div>

<div className="form-group">

<label htmlFor="description">Description</label>

<input

type="text"

className="form-control"

id="description"

name="description"

value={currentTutorial.description}

onChange={handleInputChange}

/>

</div>

<div className="form-group">

<label>

<strong>Status:</strong>

</label>

{currentTutorial.published ? "Published" : "Pending"}

</div>

</form>

{currentTutorial.published ? (

<button

className="badge badge-primary mr-2"

onClick={() => updatePublished(false)}

>

UnPublish

</button>

) : (

<button

className="badge badge-primary mr-2"

onClick={() => updatePublished(true)}

>

Publish

</button>

)}

<button className="badge badge-danger mr-2" onClick={deleteTutorial}>

Delete

</button>

<button

type="submit"

className="badge badge-success"

onClick={updateTutorial}

>

Update

</button>

<p>{message}</p>

</div>

) : (

<div>

<br />

<p>Please click on a Tutorial...</p>

</div>

)}

</div>

);

};

export default Tutorial;

**Add CSS style for React Components**

Open **src**/*App.css* and write CSS code as following:

.list {

text-align: left;

max-width: 750px;

margin: auto;

}

.submit-form {

max-width: 300px;

margin: auto;

}

.edit-form {

max-width: 300px;

margin: auto;

}

**Configure Port for React CRUD Client with Web API**

Because most of HTTP Server use CORS configuration that accepts resource sharing retrictted to some sites or ports, so we also need to configure port for our App.

In project folder, create *.env* file with following content:

PORT=8081

Now we’ve set our app running at port 8081.

**Run React Hooks CRUD App**

You can run our App with command: npm start.  
If the process is successful, open Browser with Url: http://localhost:8081/ and check it.