Lab: Fetch Data using useFetch

Estimated time needed: 40 minutes

What you will learn

In this lab, you will learn how to fetch data in React using a custom hook (useFetch) and a corresponding component (FetchData). By encapsulating data-fetching logic into reusable hooks, you can improve code maintainability. You will master React's useState and useEffect hooks for managing state and asynchronous operations efficiently. Additionally, you learn to dynamically render fetched data on the UI with JSX, enabling the creation of dynamic and interactive components.

Learning objectives

After completing this lab, you will be able to:

- Fetch data from an external API using the fetch API method within a React application

- Implement a custom React hook (useFetch) to encapsulate the logic to fetch data

- Render the data retrieved from the API onto the UI and display relevant information such as name, importance, benefits, and the best time to intake for each item

Prerequisites

- Basic Knowledge of HTML
- Intermediate knowledge of JavaScript
- Basic Knowledge of React hooks and custom hook

Setting up the Environment

1. From the menu on top of the lab, click on the **Terminal** tab at the top-right of the window shown at number 1 in the given screenshot, and then click **New Terminal** as shown at number 2.
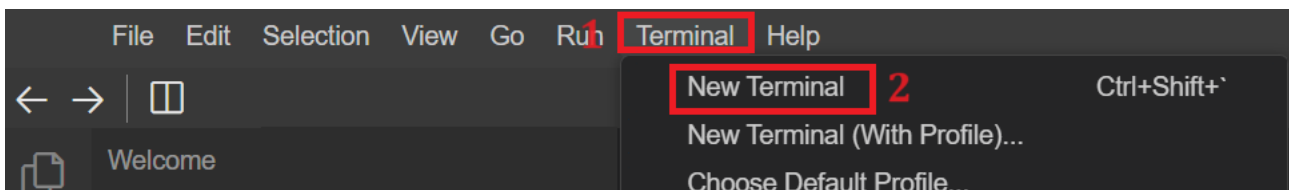
2. Write the following command in the terminal to clone the boiler template for this React application and hit Enter. The custom_hook application includes the class components named as **useFetch.jsx** and **FetchData.jsx** and a css file named as **FetchData.css**.

```
git clone https://github.com/ibm-developer-skills-network/custom_hook.git
```

3. This will create a folder named **custom_hook** under the **project** folder and the structure will be as shown in the given screenshot.
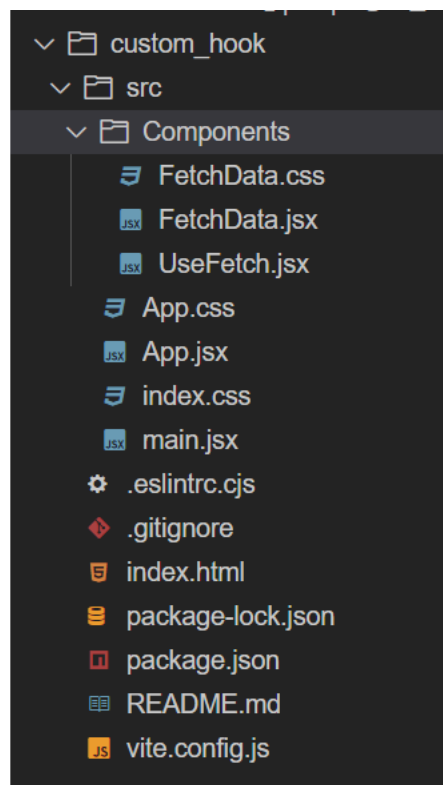
4. Write the command to enter the **custom_hook** folder in the terminal. Use the below command to navigate to the **custom_hook** folder in the terminal.

```
cd custom_hook
```

5. To make sure that the code you have cloned is working correctly, you need to follow the given steps:

   - Write the following command in the terminal and hit Enter to install all the necessary packages to execute the application:

     ```
     npm install
     ```

   - Then execute the following command to run the application and this will provide you with port number 4173:

     ```
     npm run preview
     ```

6. To view your React application, click the Skills Network icon on the left panel (refer to number 1). This action will open the **SKILLS NETWORK TOOLBOX**. Next, click **Launch Application** (refer to number 2). Enter port number **4173** in **Application Port** (refer to number 3) and click .
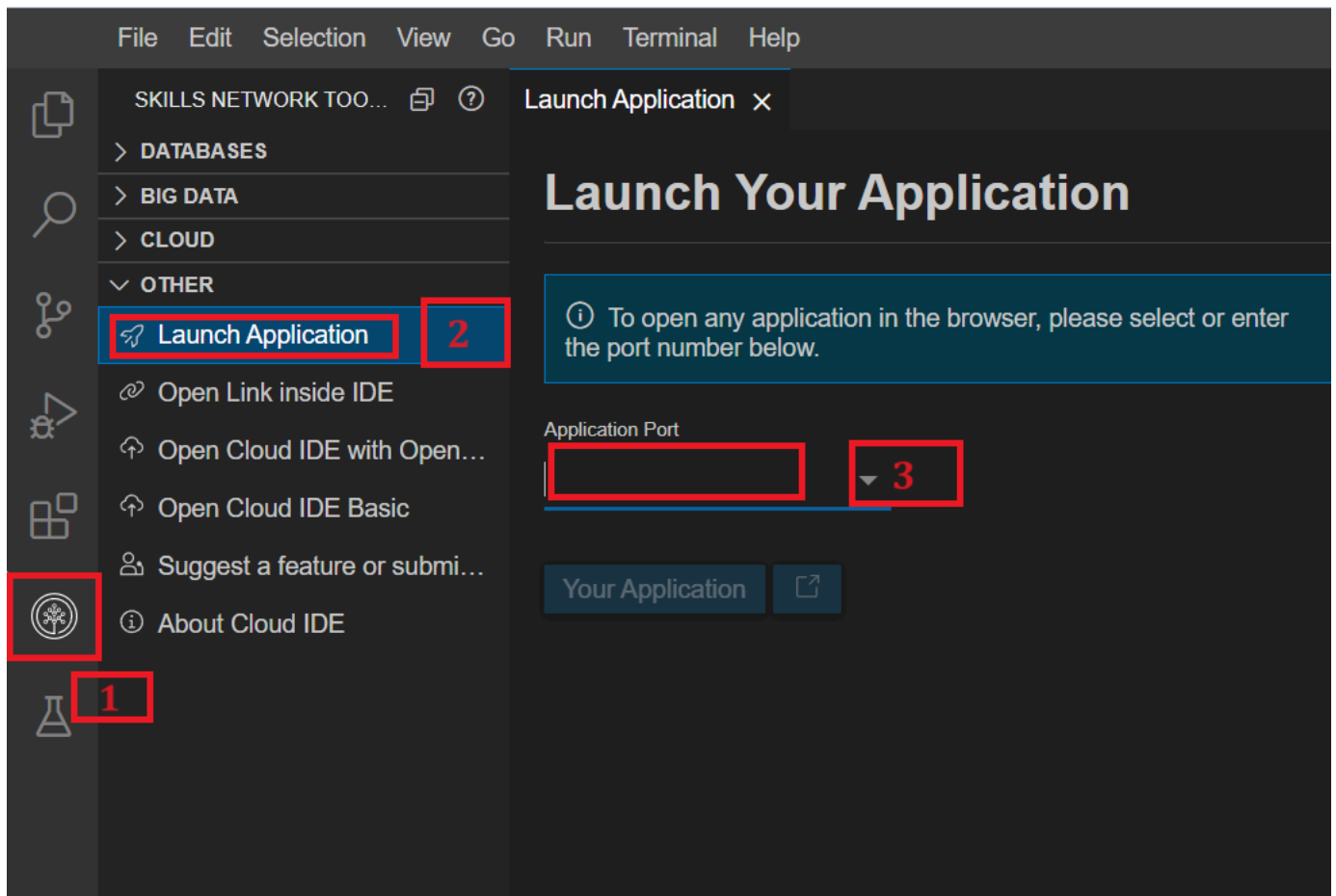
7. The output will display as shown in the given screenshot.

Use Fetch Custom Hook

Create Custom Hook

1. Next, navigate to the `FetchData.jsx` component to open which is located within the **Components** folder of the **src** directory in your cloned **custom_hook** folder.

2. The basic structure of `FetchData.jsx` component has been provided with a default function component having one `<h1>` tag with heading and a `<ul>` tag with class named as **list_data_main**.

```
import React from 'react'
const FetchData = () => {
  return (
    <>
      <ul className='list_data_main'>
        <h1 className='useFetch_heading'>Use Fetch Custom Hook</h1>
      </ul>
    </>
  )
}
export default FetchData
```

3. Now you need to fetch data from [API Data Link](). There are two approaches to fetch the data:

   - First, implement the logic using the fetch API method within each component that needs to fetch data. Suppose you have 5 components in which you need to load the data from external source. In this case, for each component you need to write the similar logic to fetch the details.
   - Second, create a reusable custom hook. This custom hook encapsulates the data fetching logic, allowing it to be reused across multiple components without the need to rewrite the entire code to fetch details in each component.

4. To create the custom hook, you need to navigate to the `useFetch.jsx` component under **Components** folder of **src** folder. The basic layout has been provided to you in the component as given:

```
const useFetch = (url) => {
}
export default useFetch
```

- In above one arrow function function is declared with one parameter passes with name as `url`.
- This method has been export as default.

5. In the basic layout, you need to implement the `useEffect` hook to create a template to fetch the data.

6. Create a variable named `data` using `useState` hook along with `setData` as the function. Include this code within arrow function component.

```
const[data,setData]=useState();
```

7. Also include given statement at the top `useFetch.jsx` component to ensure the working of **useState** hook.

```
import { useState } from "react";
```

6. Now implement the `useEffect` hook within arrow function in which you will create the logic to fetch data from any given url using the fetch api method and return data as an array. Include this code after variable declaration with **useState** hook.

```
useEffect(()=>{
        fetch(url).then((res)=>res.json())
        .then((data)=>setData(data))
    },[])
   return [data]
```

- `useEffect`: This is a React Hook used for performing side effects in functional components. It's typically used for fetching data, subscribing to events, or any other side effects that don't involve rendering.
- `fetch(url)`: This initiates an HTTP request to the specified url.
- `.then((res) => res.json())`: This converts the response from the server to JSON format.
- `.then((data) => setData(data))`: This sets the retrieved data to the state variable data. setData is a function that updates the state in React functional components.

7. Include given code at the top of the `useFetch.jsx` component to ensure working of **useEffect** hook.

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

8. The entire code should have the same structure as given below:

```
import React, { useEffect, useState } from 'react'
 const useFetch = (url) => {
     const[data,setData]=useState();
     useEffect(()=>{
        fetch(url).then((res)=>res.json())
        .then((data)=>setData(data))
    },[])
   return [data]
}
export default useFetch
```

The `url` variable is passed as a parameter in the `useFetch` arrow function to specify the exact URL of a specific website from which any component can easily fetch data. This function allows components using the custom hook to provide a URL dynamically, enabling seamless data fetching from various sources without the need for redundant code.
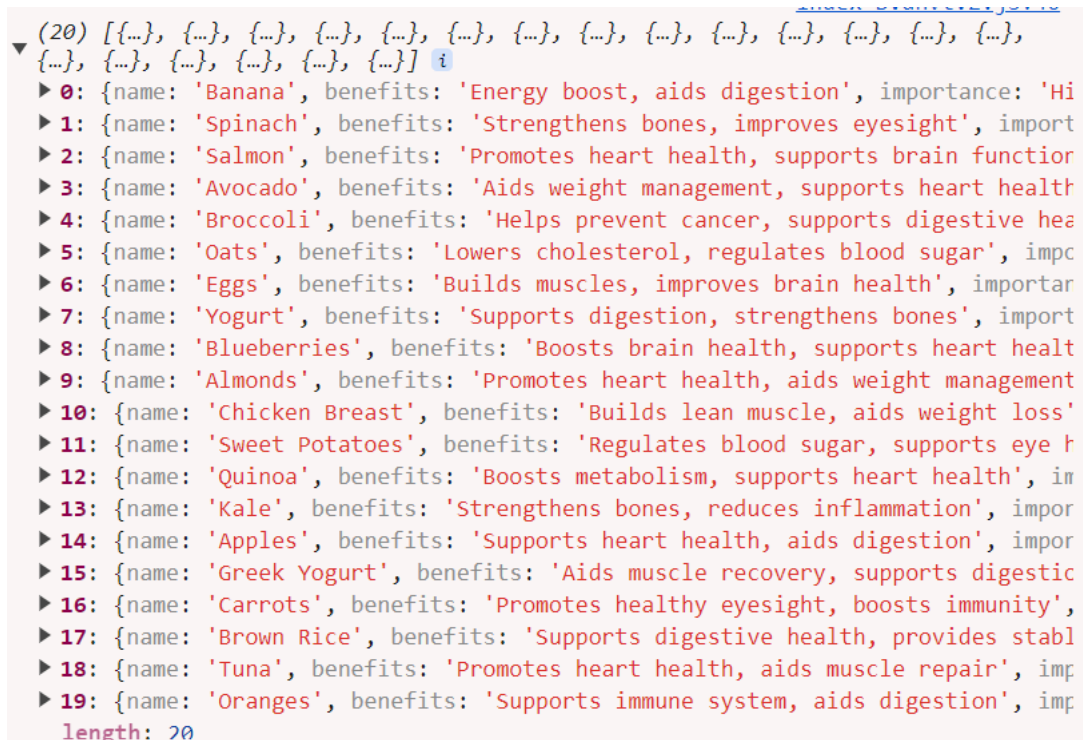
Fetch Data Using Custom Hook

1. Now again, navigate to the `FetchData.jsx` file located within the Components folder of the **src** directory in your cloned **custom_hook** folder.

2. Implement the custom hook by passing the specific url from which you want to fetch data inside the `FetchData` arrow function. Also include `console.log` to check whether data has been retrieved in the variable `data`. Include given code before `return` statement of `FetchData.jsx` function component.

```
const [data]=useFetch('https://api.npoint.io/9045c260b1565daa9e15');
console.log(data);
```

3. Import **useFetch** to ensure working of this component. for this include gievn code at the top of `FetchData.jsx` component.

```
import useFetch from './UseFetch';
```

4. To check the output rerun the React application again. In the browser right-click and select **inspect** and click the **Console** tab. The output should display as shown in the given screenshot.
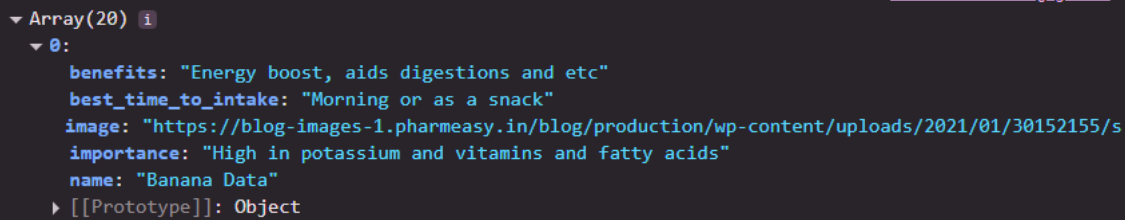


```
(20) [{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…},
{…}, {…}, {…}, {…}, {…}, {…}] ⓘ
 ▶ 0: {name: 'Banana', benefits: 'Energy boost, aids digestion', importance: 'Hi
 ▶ 1: {name: 'Spinach', benefits: 'Strengthens bones, improves eyesight', import
 ▶ 2: {name: 'Salmon', benefits: 'Promotes heart health, supports brain function
 ▶ 3: {name: 'Avocado', benefits: 'Aids weight management, supports heart health
 ▶ 4: {name: 'Broccoli', benefits: 'Helps prevent cancer, supports digestive hea
 ▶ 5: {name: 'Oats', benefits: 'Lowers cholesterol, regulates blood sugar', impc
 ▶ 6: {name: 'Eggs', benefits: 'Builds muscles, improves brain health', importar
 ▶ 7: {name: 'Yogurt', benefits: 'Supports digestion, strengthens bones', import
 ▶ 8: {name: 'Blueberries', benefits: 'Boosts brain health, supports heart healt
 ▶ 9: {name: 'Almonds', benefits: 'Promotes heart health, aids weight management
 ▶ 10: {name: 'Chicken Breast', benefits: 'Builds lean muscle, aids weight loss'
 ▶ 11: {name: 'Sweet Potatoes', benefits: 'Regulates blood sugar, supports eye h
 ▶ 12: {name: 'Quinoa', benefits: 'Boosts metabolism, supports heart health', in
 ▶ 13: {name: 'Kale', benefits: 'Strengthens bones, reduces inflammation', impor
 ▶ 14: {name: 'Apples', benefits: 'Supports heart health, aids digestion', impor
 ▶ 15: {name: 'Greek Yogurt', benefits: 'Aids muscle recovery, supports digestic
 ▶ 16: {name: 'Carrots', benefits: 'Promotes healthy eyesight, boosts immunity',
 ▶ 17: {name: 'Brown Rice', benefits: 'Supports digestive health, provides stabl
 ▶ 18: {name: 'Tuna', benefits: 'Promotes heart health, aids muscle repair', imp
 ▶ 19: {name: 'Oranges', benefits: 'Supports immune system, aids digestion', imp
   length: 20
```

Retrieve Data in Front End

1. To retrieve the data, you need to perform an iteration of the data array to fetch the entire data. You need to apply the `map` array method within the `<ul>` tag after `<h1>...</h1>` tag of `FetchData.jsx` component as follows:

```
{data && data.map((e)=>(
    <>
    </>
))}
```

2. To implement this, create the `<li>` tag with the class name `list_data` within fragments `<></>` of `<ul>` tag. Afterwards, inspect the `console.log` output to determine which data you wish to display on the frontend. For example, in the given screenshot, upon expanding the first index object of the array, you'll notice that there are 5 distinct types of information that can be accessed.

```
▼ Array(20) ℹ
  ▼ 0:
      benefits: "Energy boost, aids digestions and etc"
      best_time_to_intake: "Morning or as a snack"
      image: "https://blog-images-1.pharmeasy.in/blog/production/wp-content/uploads/2021/01/30152155/s
      importance: "High in potassium and vitamins and fatty acids"
      name: "Banana Data"
    ▶ [[Prototype]]: Object
```

3. These distinct types, then can fetch data using e variable that you have passed as parameter in the `map` method.

```
<li className='list_data'>
        <h3>{e.name}</h3>
        <p><strong>Importance: </strong>{e.importance}</p>
        <p><strong>Benefits: </strong>{e.benefits}</p>
        <p><strong>Time to eat: </strong>{e.best_time_to_intake}</p>
        </li>
```

4. Include css file inside `FetchData.jsx` component using import statement. Include give statement before function component along with other import statements.

```
import './FetchData.css'
```

5. The entire code will look like below:

```
import React from 'react'
import useFetch from './UseFetch'
import './FetchData.css'
 const FetchData = () => {
    const [data]=useFetch('https://api.npoint.io/9045c260b1565daa9e15');
    console.log(data);
  return (
    <>
    <ul className='list_data_main'>
      <h1 className='useFetch_heading'>Use Fetch Custom Hook</h1>
      {data && data.map(e=>(
        <>
        <li className='list_data'>
            <h3>{e.name}</h3>
            <p><strong>Importance: </strong>{e.importance}</p>
            <p><strong>Benefits: </strong>{e.benefits}</p>
            <p><strong>Time to eat: </strong>{e.best_time_to_intake}</p>
            </li>
        </>
     ))}
     </ul>
     </>
     )
}
export default  FetchData
```

Note: The e parameter contains the value of each array index for every iteration.

Check the Output

1. To stop the execution of the React application in the terminal perform `ctrl+c` to quit.

2. Then write the given command in the terminal and hit Enter.

```
npm run preview
```

3. To view your React application, refresh the already opened webpage for the React application on your browser. If it is not open then, click the Skills Network icon on the left panel. This action will open the "SKILLS NETWORK TOOLBOX." Next, select "Launch Application". Enter port number **4173** in "Application Port" and click [icon].

4. The output will be displayed as shown in the given screenshot.

Use Fetch Custom Hook

Banana Data

Importance: High in potassium and vitamins and fatty acids

Benefits: Energy boost, aids digestions and etc

Time to eat: Morning or as a snack

Spinach

Importance: Rich in iron and antioxidants

Benefits: Strengthens bones, improves eyesight

Time to eat: Lunch or dinner or breakf=fast

Salmon

Importance: High in omega-3 fatty acids

Benefits: Promotes heart health, supports brain function

Time to eat: Lunch or dinner

Note: To see the latest changes you need to execute `npm run preview` again in the terminal.

5. ▼ Click here to see the full solution for "FetchData.jsx".

```
import React from 'react';
import useFetch from './UseFetch';
import './FetchData.css'
const FetchData = () => {
    const [data]=useFetch('https://api.npoint.io/9045c260b1565daa9e15');
    console.log(data);
  return (
    <>
     <ul className='list_data_main'>
        <h1 className='useFetch_heading'>Use Fetch Custom Hook</h1>
        {data && data.map((e)=>(
      <>
       <li className='list_data'>
          <h3>{e.name}</h3>
          <p><strong>Importance: </strong>{e.importance}</p>
          <p><strong>Benefits: </strong>{e.benefits}</p>
          <p><strong>Time to eat: </strong>{e.best_time_to_intake}</p>
          </li>
      </>
    ))}
     </ul>
     </>
   )
 }
export default FetchData
```

6. ▼ Click here to see the full solution for "useFetch.jsx".

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

```
const useFetch = (url) => {
    const[data,setData]=useState();
    useEffect(()=>{
        fetch(url).then((res)=>res.json())
        .then((data)=>setData(data))
    },[])
return [data]
}
export default useFetch
```

7. ▼ Click here to see the full solution for "App.jsx" parent component.

```
import React from 'react'
import FetchData from './Components/FetchData'
function App() {
  return (
 <>
 <FetchData/>
 </>
  )
}
export default App
```

Practice Exercise

1. In this practice exercise you will see how custom hook `useFetch.jsx` component can be used in other component to fetch data.

2. You will need to fetch data for API related to **Yoga**. Click on this link [Yoga-Benefits](#) to see the api data.

3. First create one component by right clicking on **Components** folder after selecting it and then click on **New File**. Give the component named as `FetchYogaData.jsx`.

4. Then create basic layout for in `FetchYogaData.jsx` component.

   Hint: Use function component struture

▼ Click here for the sample solution

```
import React from 'react'
const FetchYogaData = () => {
  return (
   <>
   </>
  )
}
export default FetchYogaData
```

5. Now import custom hook component `useFetch.jsx` into `FetchYogaData.jsx` component and pass api link as url to it.

   API URL is: https://api.npoint.io/4459a9a10e43812e1152

```
import useFetch from './UseFetch';
```

   Hint: Use import statement to use custome hook component in current component and declare variable varibale to pass url.

▼ Click here for the sample solution

```
import useFetch from './UseFetch';
const [data]=useFetch('https://api.npoint.io/4459a9a10e43812e1152');
```

- Include import statement at the top of the component and data variable before the return statement of the function component.

6. Write `console.log` statement to check if data is coming in console tab in browser or not.

   Hint: write `console.log(data)` before return statement

7. Include `FetchYogaData.jsx` component in the `App.jsx` parent component and then Check the output by re-running the application in terminal again.

   Hint: use import statement.

8. Create `<ul>` tag with className as list_data_main. Within this, create logic to map through the fetched data and display each item inside `<li>` tags.

The screenshot below displays a sample output

Yoga Benefits

Lotus Pose (Padmasana)

Benefits: Calms the brain, stimulates the pelvis, spine, abdomen, and bladder, stretches the ankles and knees, eases menstrual

Duration: 5-10 minutes

Sukhasana (Easy Pose)

Benefits: Calms the brain, stretches the knees and ankles, strengthens the b

Duration: 5-10 minutes

Corpse Pose (Savasana)

Benefits: Reduces fatigue and headache, relieves mild depression, stress, and insomnia, relaxes th

Duration: 10-15 minutes

Half Lotus Pose (Ardha Padmasana)

Benefits: Stretches the knees and ankles, opens up the hips, helps to calm the mind, incr

Duration: 5-10 minutes

Child's Pose (Balasana)

Benefits: Stretches the hips, thighs, and ankles, helps to relieve stress and fatigue, gently

Duration: 1-5 minutes

▼ Click here to see the full solution for "FetchYogaData.jsx".

```
import React from 'react';
import UseFetch from './UseFetch';
import './FetchData.css';
const FetchYogaData = () => {
  // Use the custom hook to fetch data from the Yoga API.
  const [data] = UseFetch('https://api.npoint.io/4459a9a10e43812e1152');
  console.log(data);
  return (
    <ul className="list_data_main">
      <h1 className="usefetch_heading">Yoga Benefits</h1>
      {data && data.map((e) => (
        <li className="list_data">
          <h3>{e.name}</h3>
          <p><strong>Benefits: </strong>{e.benefits}</p>
          <p><strong>Duration: </strong>{e.time_duration}</p>
        </li>
      ))}
    </ul>
  );
```

```
  };
  export default FetchYogaData;
```

▼ Click here to see the full solution for "App.jsx".

```
import React from 'react'
import FetchData from './Components/FetchData';
import FetchYogaData from './Components/FetchYogaData'
function App() {
  return (
 <>
 {/* <FetchData/> */}
 <FetchYogaData/>
 </>
  )
}
export default App
```

Conclusion

Congratulations! You have created your React application to fetch the data using your own personalized custom hook!

- In this lab, you created a React component named **FetchData** responsible for rendering fetched data onto the user interface. It uses the **useFetch** custom hook to retrieve data from an API endpoint specified by a URL.

- The custom React hook named **useFetch** encapsulates the logic for fetching data from a specified URL. This hook uses **useState** and **useEffect** to manage the data fetching process, ensuring that data is retrieved asynchronously and stored in the component's state.

- You have learned to implement the map method through the fetched data array that renders each item onto the UI using JSX. It displays various attributes of each data item, such as name, importance, benefits, and the best time to intake, within styled list items.

- By separating the data fetching logic into a custom hook (useFetch), you have adopted a modular and reusable approach. This approach allows components throughout the application to easily fetch data from different API endpoints by simply providing the desired URL as an argument to the useFetch hook, promoting code efficiency and maintainability.

Author(s)

Richa Arora