

Tutorial 5. React CRUD

1. Passing data between components

Tutorial: Passing a message from parent to child

useEffect can be used to achieve this goal

```
import { useEffect } from "react"
import {useState} from "react"
export default function List(props){
  const [refetch, setRefetch] = useState(false)

  useEffect(() => {
    setRefetch(props.reload)
  })

  return (
    <div> {refetch? "True": "False"}</div>
  )
}
```

```
import { useState } from "react";
import List from "../List"

function App() {
  const [reload, setReload] = useState(false)
  return (
    <div>
      <List reload={reload}/>
    </div>
  )
}
```

```

    <button onClick={e=>setReload(true)}>Set reload true</button>
    <button onClick={e=>setReload(false)}>Set reload false</button>
  </div>
);
}

export default App;

```

Exercise: Pass a count value from App to List component

Tutorial: Passing a message from child to parent

We can achieve this by using a callback function(). For example f1 is a callback function

```

export default function Form({f1}) {

  return(
    <div>
      <button onClick={() => f1(true)}>I am happy</button>
      <button onClick={() => f1(false)}>I am sad</button>
    </div>
  )
}

```

```

function App() {
  const [reload, setReload] = useState(false)
  return (
    <div>
      {reload? "True": "False"}
      <Form f1={v=>setReload(v)} />
    </div>
  );
}

```

Exercise: Pass a count value from Form component to App component

Tutorial: Passing values between sibling components

We can combine both of the tricks above

```
function Form({f1}) {
  return (
    <div>
      <button onClick={e=>f1(true)}>Set True</button>
      <button onClick={e=>f1(false)}>Set False</button>
    </div>
  )
}
```

```
function List(props) {
  const [refetch, setRefetch] = useState(false)
  useEffect(() => {
    setRefetch(props.reload)
  })
}
```

```
return (
  <div> {refetch? "True": "False"}</div>
)
}
```

```
function App() {
  const [reload, setReload] = useState(false)
  return (
    <div>
      {reload? "True": "False"}
      <Form f1={reload=>setReload(reload)} />
    </div>
  )
}
```

```
<List reload={reload} />
```

```
</div>
);
}
```

Exercise: Pass a count value from Form component to List component

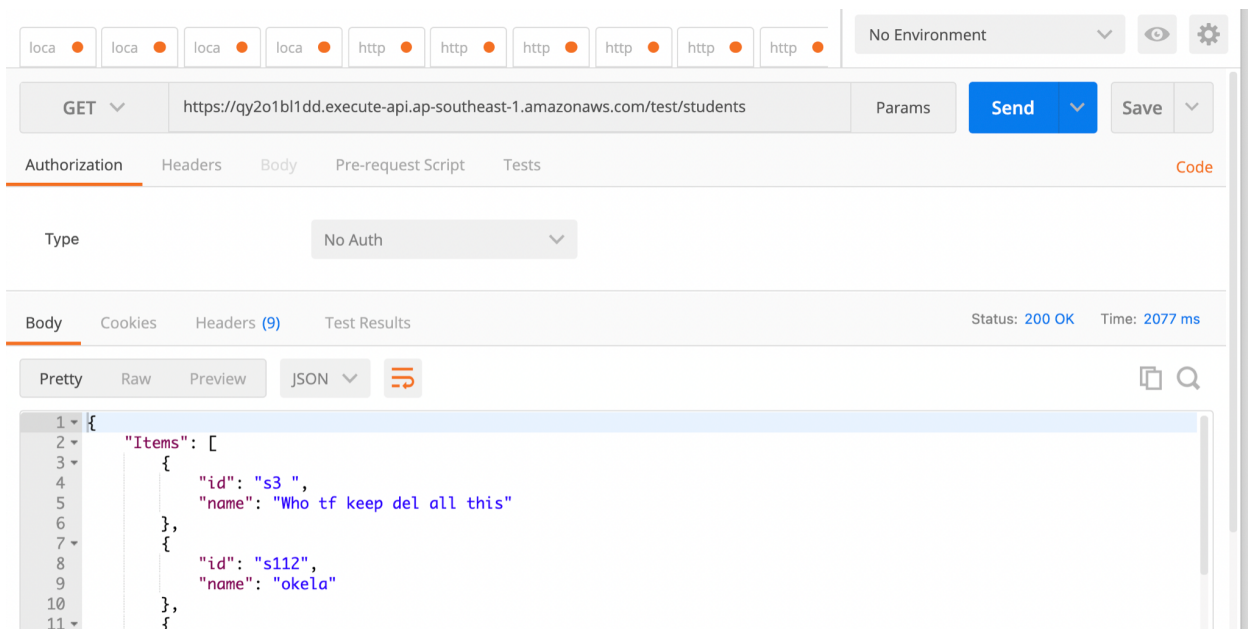
2. Restful API

Restful API is an architecture style that allows accessing and updating resources through standard HTTP protocol.

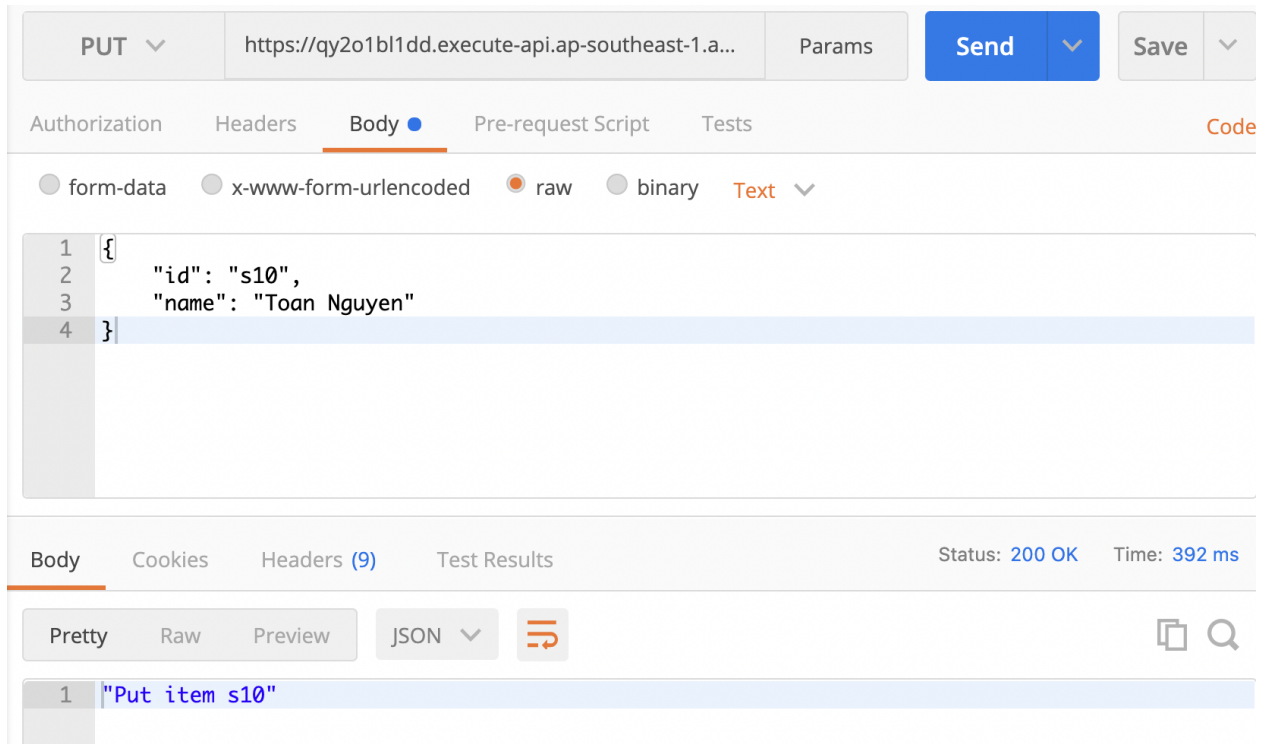
Please use Postman to test CRUD on the following REST endpoint:

<https://l3k2uf0o64.execute-api.ap-southeast-1.amazonaws.com/test/products>

a) GET:



b) PUT:



The image shows a Postman interface for a PUT request. The top bar includes a dropdown menu set to 'PUT', a URL field containing 'https://qy2o1bl1dd.execute-api.ap-southeast-1.a...', a 'Params' button, and 'Send' and 'Save' buttons. Below this, a tabbed interface shows 'Authorization', 'Headers', 'Body' (selected), 'Pre-request Script', and 'Tests'. The 'Body' tab has radio buttons for 'form-data', 'x-www-form-urlencoded', 'raw' (selected), and 'binary', with a 'Text' dropdown. The body content is a JSON object:

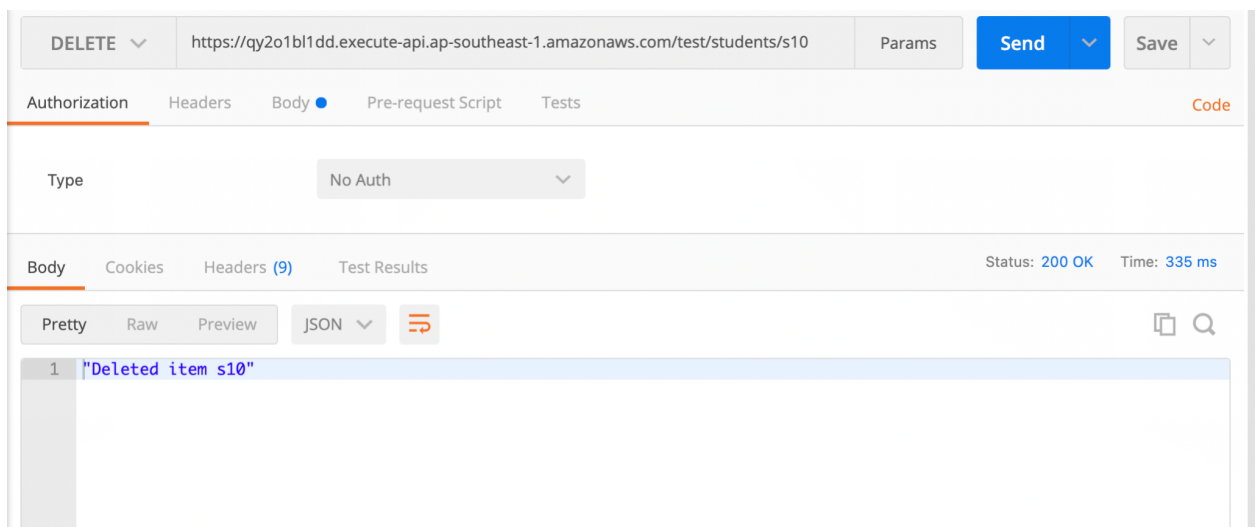
```
{
  "id": "s10",
  "name": "Toan Nguyen"
}
```

. The bottom section shows the response with tabs for 'Body', 'Cookies', 'Headers (9)', and 'Test Results'. The status is '200 OK' and the time is '392 ms'. The 'Body' tab has buttons for 'Pretty', 'Raw', and 'Preview', a 'JSON' dropdown, and a 'Send' button. The response body is a string:

```
"Put item s10"
```

.

c) DELETE:



The image shows a Postman interface for a DELETE request. The top bar includes a dropdown menu set to 'DELETE', a URL field containing 'https://qy2o1bl1dd.execute-api.ap-southeast-1.amazonaws.com/test/students/s10', a 'Params' button, and 'Send' and 'Save' buttons. Below this, a tabbed interface shows 'Authorization', 'Headers', 'Body' (selected), 'Pre-request Script', and 'Tests'. The 'Body' tab has a 'Type' dropdown set to 'No Auth'. The bottom section shows the response with tabs for 'Body', 'Cookies', 'Headers (9)', and 'Test Results'. The status is '200 OK' and the time is '335 ms'. The 'Body' tab has buttons for 'Pretty', 'Raw', and 'Preview', a 'JSON' dropdown, and a 'Send' button. The response body is a string:

```
"Deleted item s10"
```

.

Reference:

<https://blog.testproject.io/2020/07/15/the-ultimate-postman-tutorial-for-api-testing/>

3. Getting and posting data from/to Restful API

We can use JS function `fetch()` to get data from Restful endpoints.

Create a component called `ProductType` and put the following:

URL endpoint: <https://13k2uf0o64.execute-api.ap-southeast-1.amazonaws.com/test/products>

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

export default function StudentList() {
  const [data, setData] = useState([]);
  const endPoint = "https://13k2uf0o64.execute-api.ap-southeast-1.amazonaws.com/test/products"
  const [name, setName] = useState('')

  const save = () => {
    fetch(endPoint, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json'
      },
      body: JSON.stringify({ name: name })
    }).then(data => {

      fetch(endPoint)
        .then(response => response.json())
        .then(data => setData(data))

      data.json()
    })
  }

  //get data from api
  useEffect(() => {
    fetch(endPoint)
      .then(response => response.json())
      .then(data => setData(data));
  }, [])
}
```

```

return (
  <div>
    <h1>Form</h1>
    Name:<input type="text" value={name} onChange={ (e) => setName (e.target.value) } />
    <button onClick={ () => save () }>Save</button>

    <h1>List</h1>
    {data.map(a => (
      <li>{a.name} {a.id}</li>
    ))}

  </div>
);
}

```

4. Full CRUD

This example combines all of operations into one single component

You must change the endpoint to

<https://l3k2uf0o64.execute-api.ap-southeast-1.amazonaws.com/test/products>

```

import React, { useState, useEffect } from "react";
export default function StudentList() {
  const [data, setData] = useState([]);
  const endPoint = "https://l3k2uf0o64.execute-api.ap-southeast-1.amazonaws.com/test/products"
  const [name, setName] = useState('')

```

```

const [id, setId] = useState('')
const save = () => {
  if (id===''){
    fetch(endPoint, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json'
      },
      body: JSON.stringify({ name: name})
    }).then(data => load())
  }
  else{
    fetch(endPoint, {
      method: 'PUT',
      headers: {
        'Content-Type': 'application/json'
      },
      body: JSON.stringify({ id: id, name: name})
    }).then(data => load())
  }
}

const deleteStudent = (id) => {
  fetch(endPoint + "/" + id, {
    method: 'DELETE',
    headers: {
      'Content-Type': 'application/json'
    },
    body: JSON.stringify({ id: id})
  }).then(data => load())
}

const editStudent = (id, name) => {
  setId(id)
  setName(name)
}

//get data from api
const load = () => {

```



```

    fetch(endPoint)
      .then(response => response.json())
      .then(data => setData(data.Items));
  }

  //load data automatically
  useEffect(() => {
    load()
  }, [])

  return (
    <div>
      <h1>Form</h1>

      Id:<input type="text" value={id} onChange={ (e)=>setId(e.target.value) }/><br/>
      Name:<input type="text" value={name}
      onChange={ (e)=>setName(e.target.value) }/><br/>

      <button onClick={ () => save() }>Save</button>

      <h1>List</h1>
      <table border="1">
        <tr><td>Name</td><td>ID</td><td>Delete</td></tr>
        {data.map(a => (
          <tr>
            <td>{a.name}</td><td>{a.id}</td>
            <td><button onClick={ () => deleteStudent(a.id) }>Delete</button></td>
            <td><button onClick={ () => editStudent(a.id, a.name) }>Edit</button></td>
          </tr>
        ))}
      </table>
    </div>
  );
}

```

Important note:

If you see this problem, it means that your GET request not successful. Check the return json that you receive. In our case, it must be **data.Items**

```
const load = () => {  
  fetch(endPoint)  
    .then(response => response.json())  
    .then(data => setData(data.Items));  
}
```

TypeError: data.map is not a function

FullCRUD

src/FullCRUD.js:65

```
62 | <button onClick={() => save()}>Save</button>  
63 | <h1>List</h1>  
64 | <table border="1">  
> 65 | <tr><td>Name</td><td>ID</td><td>Delete</td></tr>  
66 | {data.map(a => (  
67 |   <tr>  
68 |     <td>{a.name}</td><td>{a.id}</td>
```