CRUD operations:

CREATE:

```
import React, { useState } from 'react'
export default function CRUDexplain() {


//initial input filed useState
const [inputDetails,setInput]=useState([
    {
        empRoll:'',
        empName:'',
        empAge:'',
        empEmail:'',
        empCountry:'',
        empGender:'',
        empfeedback:''
    }
])
```

- You initialize state using the **useState** hook for **inputDetails**, which represents the user details entered in the form.
- **empRoll**, **empName**, **empAge**, **empEmail**, **empCountry**, **empGender**, and **empfeedback**. Initially, all fields are empty.

//initial input filed useState

```
const initialData=[
{   empRoll:1,
    empName:'karthick',
    empAge:23,
    empEmail:'karthick@gmail.com',
    empCountry:'india',
    empGender:'Male',
    empfeedback:'good'
},
{
    empRoll:2,
    empName:'kumar',
    empAge:20,
    empEmail:'kumar@gmail.com',
    empCountry:'USA',
    empGender:'Male',
    empfeedback:'good'

},
```

```
{
    empRoll:3,
    empName:'Arun',
    empAge:21,
    empEmail:'Arun@gmail.com',
    empCountry:'india',
    empGender:'Male',
    empfeedback:'good'

}
]

//alreay some values in useState
const [empDetails,setDetails]=useState(initialData);
```

initialize state for **empDetails**, which stores an array of user details objects. This represents the data displayed in the table. Initially,

READ:

```
       <tbody>

        { empDetails.length>0?(empDetails.map((value,index)=>

            <tr key={index}>
                <td>{value?.empRoll}</td>
                <td>{value?.empName}</td>
                <td>{value?.empAge}</td>
                <td>{value?.empEmail} </td>
                <td>{value?.empCountry} </td>
                <td>{value?.empGender} </td>
                <td>{value?.empfeedback} </td>
               <td><button onClick={()=>onHandleEdit(index)}
                id='edit' >EDIT</button>
                <button onClick={()=>onHandleDelte(index)}
id='delete'>DELETE</button>
                </td>
              </tr>
            )):(
            <tr>
                <td colSpan={8}>NO USERS IN THE TABLE </td>
            </tr>)}
            </tbody>
```

// Inside JSX <tbody> we used conditional rendering to render user details in rows. Otherwise, it displays a message indicating "NO USERS IN THE TABLE."

//The key={index} attribute is set to ensure React can efficiently update and re-render the list of users when changes occur.

//key attribute is used to uniquely identify each row when rendering a list of elements in React.

- In this case, key={index} assigns a unique identifier to each row based on its index in the array of user details (empDetails).
- This helps React efficiently update and re-render rows when changes occur.
- value?.empRoll: This is using optional chaining (?.).
- It allows you to **access the empRoll property** of the value object without causing an error if value is null or undefined.
- **<button onClick={() => onHandleEdit(index)} id='edit'>EDIT</button>**
  onHandleEdit function when clicked, passing the index of the user being edited as a parameter. This allows you to edit the user's details.

- **(<button onClick={() => onHandleDelte(index)} id='delete'>DELETE</button>)** the onHandleDelte function when clicked, passing the index of the user being deleted as a parameter.
- This allows you to delete the user.

UPDATE:

```
//onchange to set input values to inputDetails

const onHandleChange=(e)=>{
    const {name,value}=e.target;
    setInput((prev)=>({
        ...prev,
        [name]:value

    }))
}
```

- onHandleChange that handles changes in the input fields. It takes an **event (e) as a parameter.**
- Inside the function, **destructure the name and value properties from the input element** that triggered the change.
- You update the **inputDetails state using the spread operator (...prev**) to retain the existing state and only update the field specified by name.

```
//submit button function
const onHandleSubmit =()=>{
    if(!inputDetails.empName||!inputDetails.empAge||!inputDetails.empEmail||!input
Details.empCountry||!inputDetails.empGender)return;
    const
finIndex=empDetails.findIndex(y=>Number(y.empRoll)===Number(inputDetails.empRoll))
    if (finIndex>-1){
empDetails[finIndex]={
    ...empDetails[finIndex],
    empName:inputDetails.empName,
    empAge:inputDetails.empAge,
    empEmail:inputDetails.empEmail,
    empCountry:inputDetails.empCountry,
    empGender:inputDetails.empGender,
    empfeedback:inputDetails.empfeedback
}
setDetails([...empDetails])
setInput({ empRoll: '', empName: '', empAge: '', empEmail:
'',empCountry:'',empGender:'',empfeedback:'' });
setIsEdit(false);

    }else{
        const neData=[...empDetails]
        neData.push({
            empRoll:inputDetails.empRoll,
            empName:inputDetails.empName,
            empAge:inputDetails.empAge,
            empEmail:inputDetails.empEmail,
            empCountry:inputDetails.empCountry,
            empGender:inputDetails.empGender,
            empfeedback:inputDetails.empfeedback
        });
        setDetails([...neData])
        setInput({empRoll:'',empName:'',empAge:'',empEmail:'',empCountry:'',empGen
der:'',empfeedback:''}) //input
    }
    }
```

- You use the **findIndex method to check if the user with the same empRoll as in inputDetails already exists in the empDetails array**. This helps determine if you're updating an existing user.

- If the `finIndex` (index of the existing user) is greater than -1 (meaning the user exists), you update the user's details in the `empDetails` array with the values from `inputDetails`.
- If the user is new (not found in the array), you create a new user object and push it into the `empDetails` array.

- Finally, you reset the `inputDetails` state and set `isEdit` to `false` to indicate that you're no longer in edit mode.

DELETE:

```
//delete button
const onHandleDelte=(index)=>{
    setDetails((prev)=>{
        const preData=[...prev]
        preData.splice(index,1)
        return preData

    })
    setInput({ empRoll: '', empName: '', empAge: '', empEmail:
'',empCountry:'',empGender:'',empfeedback:'' });

}
```

- onHandleDelte that handles deleting a user from the empDetails array.
- Inside the function, create a copy of the empDetails array (preData) **using the spread operator and remove the user at the specified index using splice**.
- You then update the empDetails state with the modified array and **reset the inputDetails state.**

//EDIT:

```
//edit button
const onHandleEdit=(index)=>{
    const allData=empDetails[index]
    setInput({
        empRoll:allData.empRoll,
        empName:allData.empName,
        empAge:allData.empAge,
        empEmail:allData.empEmail,
        empCountry:allData.empCountry,
        empGender:allData.empGender,
        empfeedback:allData.empfeedback
    })

    setIsEdit(true);

}
```

- `onHandleEdit` that handles editing a user.
- Inside the function, retrieve all the details of the user at the specified index and set the `inputDetails` state with those details. This allows you to populate the form fields with the user's data for editing.
- You also set `isEdit` to `true` to indicate that you are in edit mode.

```
        <button id='submit' onClick={()=>onHandleSubmit()} >{isEdit ? 'Update' : 'Submit'}</button> <br />
```

- "Submit" button that has dynamic text based on whether you are in edit mode (`isEdit`). If in edit mode, the button text is "Update"; otherwise, it's "Submit."
- When clicked, it triggers the `onHandleSubmit` function for adding or updating a user

FULL CODE:

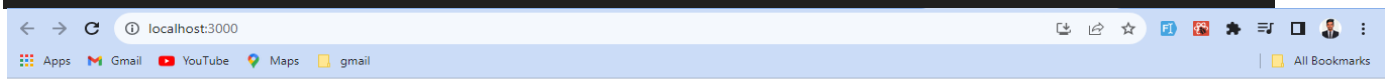
```
import React, { useState } from 'react'
import './App.css'
export default function CRUDexplain() {


//initial input filed useState
const [inputDetails,setInput]=useState([
    {
        empRoll:'',
        empName:'',
        empAge:'',
        empEmail:'',
        empCountry:'',
        empGender:'',
        empfeedback:''
    }
])

//change button
const [isEdit, setIsEdit] = useState(false);



//initial table values  shows in table

const initialData=[
{   empRoll:1,
    empName:'karthick',
    empAge:23,
    empEmail:'karthick@gmail.com',
    empCountry:'india',
    empGender:'Male',
    empfeedback:'good'
},
{
    empRoll:2,
    empName:'kumar',
    empAge:20,
    empEmail:'kumar@gmail.com',
    empCountry:'USA',
```

```
    empGender:'Male',
    empfeedback:'good'

},
{
    empRoll:3,
    empName:'Arun',
    empAge:21,
    empEmail:'Arun@gmail.com',
    empCountry:'india',
    empGender:'Male',
    empfeedback:'good'

}
]
//alreay some values in useState
const [empDetails,setDetails]=useState(initialData);


//onchange to set input values to inputDetails
const onHandleChange=(e)=>{
    const {name,value}=e.target;
    setInput((prev)=>({
        ...prev,
        [name]:value

    }))
}



//submit button function
const onHandleSubmit =()=>{
    if(!inputDetails.empName||!inputDetails.empAge||!inputDetails.empEmail||!input
Details.empCountry||!inputDetails.empGender)return;
    const
finIndex=empDetails.findIndex(y=>Number(y.empRoll)===Number(inputDetails.empRoll))
    if (finIndex>-1){
empDetails[finIndex]={
    ...empDetails[finIndex],
    empName:inputDetails.empName,
    empAge:inputDetails.empAge,
    empEmail:inputDetails.empEmail,
    empCountry:inputDetails.empCountry,
    empGender:inputDetails.empGender,
    empfeedback:inputDetails.empfeedback
}
setDetails([...empDetails])
setInput({ empRoll: '', empName: '', empAge: '', empEmail:
'',empCountry:'',empGender:'',empfeedback:'' });
setIsEdit(false);
```

```
    }else{
        const neData=[...empDetails]
        neData.push({
            empRoll:inputDetails.empRoll,
            empName:inputDetails.empName,
            empAge:inputDetails.empAge,
            empEmail:inputDetails.empEmail,
            empCountry:inputDetails.empCountry,
            empGender:inputDetails.empGender,
            empfeedback:inputDetails.empfeedback
        });
        setDetails([...neData])
        setInput({empRoll:'',empName:'',empAge:'',empEmail:'',empCountry:'',empGen
der:'',empfeedback:''}) //input
    }
    }




//delete button
const onHandleDelte=(index)=>{
    setDetails((prev)=>{
        const preData=[...prev]
        preData.splice(index,1)
        return preData

    })
    setInput({ empRoll: '', empName: '', empAge: '', empEmail:
'',empCountry:'',empGender:'',empfeedback:'' });

}

//edit button
const onHandleEdit=(index)=>{
    const allData=empDetails[index]
    setInput({
        empRoll:allData.empRoll,
        empName:allData.empName,
        empAge:allData.empAge,
        empEmail:allData.empEmail,
        empCountry:allData.empCountry,
        empGender:allData.empGender,
        empfeedback:allData.empfeedback
    })

    setIsEdit(true);

}
  return (
    <div>
```

```
        <h2>Register form</h2>
        <input name='empRoll' placeholder='User
Roll'  value={inputDetails.empRoll} onChange={onHandleChange} /> <br/>
        <input name='empName' placeholder='use Name'  value={inputDetails.empName}
onChange={onHandleChange} /> <br/>
        <input name='empAge' value={inputDetails.empAge} onChange={onHandleChange}
placeholder='user Age' /><br/>
        <input name='empEmail' value={inputDetails.empEmail}
onChange={onHandleChange} placeholder='user Email' /><br/>
        <label>Country</label><select name="empCountry" onChange={onHandleChange}>
            <option value="">Select</option>
            <option value="India">India</option>
            <option value="USA">USA</option>
            <option value="Uk">Uk</option>
            <option value="Europe">Eroupe</option>
            <option value="Canada">Canada</option>
        </select> <br/>
         <br/>
        <label >Gender</label>
        Male<input type="radio" name="empGender" value="Male"
checked={inputDetails.empGender === "Male"} onChange={onHandleChange} />
        FeMale<input type="radio" name="empGender" value="FeMale"
checked={inputDetails.empGender === "FeMale"}  onChange={onHandleChange} />
        <br />
        <br />
           <label>Feedback</label> <br/>
            <textarea name='empfeedback'
placeholder='feedback'  value={inputDetails.empfeedback}
onChange={onHandleChange}>

            </textarea> <br/><br/>
        <button id='submit' onClick={()=>onHandleSubmit()} >{isEdit ? 'Update' :
'Submit'}</button> <br />
        <br />
        <br />
        <br />
    <div className='container'>
    <table style={{border: '2px solid black'}}>
        <thead>
          <tr>
          <th colSpan={8}>User Details</th>
          </tr>
          <tr>
            <th>Roll</th>
            <th>Name</th>
            <th>Age</th>
            <th>Email</th>
            <th>Country</th>
            <th>Gender</th>
            <th>feedback</th>
            <th>Action</th>
          </tr>
```

```
        </thead>
        <tbody>
        { empDetails.length>0?(empDetails.map((value,index)=>

            <tr key={index}>
                <td>{value?.empRoll}</td>
                <td>{value?.empName}</td>
                <td>{value?.empAge}</td>
                <td>{value?.empEmail} </td>
                <td>{value?.empCountry} </td>
                <td>{value?.empGender} </td>
                <td>{value?.empfeedback} </td>
               <td><button onClick={()=>onHandleEdit(index)} id='edit'
>EDIT</button>
                <button onClick={()=>onHandleDelte(index)}
id='delete'>DELETE</button>
                </td>
              </tr>

            )):(

            <tr>

                <td colSpan={8}>NO USERS IN THE TABLE </td>
            </tr>)}
            </tbody>
    </table>
    </div>
    </div>

  )
}
```



Register form

User Roll

use Name

user Age

user Email

Country Select

Gender Male FeMale

Feedback

feedback

Submit

User Details

Roll	Name	Age	Email	Country	Gender	feedback	Action
1	karthick	23	karthick@gmail.com	india	Male	good	EDIT DELETE
2	kumar	20	kumar@gmail.com	USA	Male	good	EDIT DELETE
3	Arun	21	Arun@gmail.com	india	Male	good	EDIT DELETE