**Batch: D - 1 Roll No.: 16010122096**

**Experiment / assignment / tutorial No. 04**

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of the Staff In-charge with date**

|  |
| --- |
| Title: Demonstrate axios to Create Mock API Server |

**AIM:** To Implement the React Axios

**Problem Definition:**

Build a React application that interacts with a RESTful API using Axios to perform CRUD (Create, Read, Update, Delete) operations. The application should allow users to view, add, update, and delete data from the server. The application should allow users to view, add, update, and delete student data, with smooth navigation between different views using the useNavigate hook.

**Requirements:**

* Create a new React application using create-react-app.
* Install Axios using npm install axios.
* Install react-router-dom to handle navigation (npm install react-router-dom).

**Data Fetching:**

Create a component (StudentList.js) that fetches a list of students from a RESTful API endpoint (e.g., https://api.example.com/students) and displays them in a table or list. Handle loading states and errors during the fetch process.

**Adding a New Student:**

* Implement a form component (AddStudent.js) that allows users to add a new student record.
* Use Axios to send a POST request to the API with the new student data.
* Upon successful submission, navigate the user back to the student list view using useNavigate and display the newly added student in the list.

**Updating Student Data:**

* Implement an edit functionality in a separate component (EditStudent.js) that allows users to update an existing student's information.
* Use Axios to send a PUT request to the API with the updated student data.
* Upon successful submission, navigate the user back to the student list view using useNavigate, and reflect the updated student information in the list.

**Deleting a Student:**

* Add a delete button next to each student in the list.
* When the delete button is clicked, use Axios to send a DELETE request to the API.
* Upon successful deletion, the student should be removed from the list without requiring a page reload.

**Navigation:**

* Use useNavigate to smoothly navigate between different components/views (StudentList, AddStudent, EditStudent).
* Ensure that the browser’s back and forward buttons work correctly to navigate between the views.

**Resources used:**

**VS code, Json Server**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Expected OUTCOME of Experiment:**

**CO 2:**. Illustrate the concepts of various front-end, back-end web application development technologies & frameworks using different web development tools.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Books/ Journals/ Websites referred:**

1. Shelly Powers Learning Node O’ Reilly 2 nd Edition, 2016.

**Pre Lab/ Prior Concepts:**

**Write details about the following content**

* useNavigate

**useNavigate** is a hook provided by React Router that allows you to programmatically navigate between routes within a React application. It enables you to change the current location and redirect users to different parts of your app without the need for traditional anchor links or manual URL changes. By calling useNavigate, you obtain a function that you can use to push new locations onto the history stack, effectively updating the view based on route changes.

* Axios

**Axios** is a popular JavaScript library used for making HTTP requests. It simplifies the process of interacting with APIs by providing an easy-to-use API for sending requests and handling responses. Axios supports various request methods like GET, POST, PUT, and DELETE, and it can handle request and response transformations, including error handling. Its promise-based nature makes it a preferred choice for managing asynchronous operations in web applications.

* Routes in React

**Routes in React** refer to the way you manage and define different views or components that should be displayed based on the URL path in a React application. React Router is the primary library used for implementing routing, allowing you to map URL paths to specific components. This enables the creation of a single-page application (SPA) with multiple views or pages, where the content updates dynamically based on the user's navigation without a full page reload.

**Implementation Details:**

**1] Json-Server DB:**

{

  "students": [

    {

      "id": "1",

      "name": "Romil Lodaya",

      "age": "19"

    },

    {

      "id": "2",

      "name": "Siddhi Somaiya",

      "age": "21"

    },

    {

      "id": "3",

      "name": "Rahil Kothari",

      "age": "20"

    },

    {

      "name": "Aakriti Mehta",

      "age": "19",

      "id": "4baa"

    }

  ]

}

**2] StudentList.jsx**

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

import axios from "axios";

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

import "../studentlist.css";

const URL = "http://localhost:3000/students";

const StudentList = () => {

  const [students, setStudents] = useState([]);

  useEffect(() => {

    const fetchStudents = async () => {

      try {

        const response = await axios.get(URL);

        setStudents(response.data);

      } catch (error) {

        console.error("Error fetching students:", error);

      }

    };

    fetchStudents();

  }, []);

  const handleDelete = async (name) => {

    try {

      const response = await axios.get(URL);

      const student = response.data.find(student => student.name === name);

      if (student) {

        await axios.delete(`${URL}/${student.id}`);

        setStudents(students.filter((student) => student.name !== name));

      }

    } catch (error) {

      console.error("Error deleting student:", error);

    }

  };

  return (

    <>

      <div className="student-data">

        <h1>Student List</h1>

        {students.length > 0 ? (

          <table className="student-table">

            <thead>

              <tr>

                <th>Name</th>

                <th>Age</th>

                <th>Actions</th>

              </tr>

            </thead>

            <tbody>

              {students.map((student) => (

                <tr key={student.id}>

                  <td>{student.name}</td>

                  <td>{student.age}</td>

                  <td>

                    <Link

                      to="/update-student"

                      state={{ studentName: student.name }}

                      className="btn btn-secondary"

                    >

                      Edit

                    </Link>

                    <button

                      onClick={() => handleDelete(student.name)}

                      className="btn btn-danger"

                    >

                      Delete

                    </button>

                  </td>

                </tr>

              ))}

            </tbody>

          </table>

        ) : (

          <p>No students found.</p>

        )}

      </div>

      <div>

        <Link to="/add-student" className="btn btn-primary">

          Add New Student

        </Link>

      </div>

    </>

  );

};

export default StudentList;

**3] AddStudent.jsx**

import React, { useState } from 'react';

import axios from 'axios';

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

import "../updatestudent.css";

const URL = "http://localhost:3000/students";

const AddStudent = () => {

  const [name, setName] = useState('');

  const [age, setAge] = useState('');

  const navigate = useNavigate();

  const handleSubmit = async (e) => {

    e.preventDefault();

    try {

      await axios.post(URL, { name, age });

      navigate('/');

    } catch (error) {

      console.error('Error adding student:', error);

    }

  };

  return (

    <div>

      <h2>Add Student</h2>

      <form onSubmit={handleSubmit}>

        <div>

          <label>Name:</label>

          <input

            type="text"

            value={name}

            onChange={(e) => setName(e.target.value)}

            required

          />

        </div>

        <div>

          <label>Age:</label>

          <input

            type="number"

            value={age}

            onChange={(e) => setAge(e.target.value)}

            required

          />

        </div>

        <button type="submit">Add Student</button>

      </form>

    </div>

  );

};

export default AddStudent;

**4] UpdateStudent.jsx**

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

import axios from 'axios';

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

import "../updatestudent.css";

const URL = "http://localhost:3000/students";

const UpdateStudent = () => {

  const navigate = useNavigate();

  const location = useLocation();

  const { studentName } = location.state || {};

  const [name, setName] = useState('');

  const [age, setAge] = useState('');

  useEffect(() => {

    const fetchStudent = async () => {

      try {

        const response = await axios.get(URL);

        const student = response.data.find(student => student.name === studentName);

        if (student) {

          setName(student.name);

          setAge(student.age);

        }

      } catch (error) {

        console.error('Error fetching student data:', error);

      }

    };

    fetchStudent();

  }, [studentName]);

  const handleSubmit = async (e) => {

    e.preventDefault();

    try {

      const response = await axios.get(URL);

      const students = response.data;

      const student = students.find(student => student.name === studentName);

      if (student) {

        await axios.put(`${URL}/${student.id}`, { name, age });

        navigate('/');

      }

    } catch (error) {

      console.error('Error updating student:', error);

    }

  };

  return (

    <div>

      <h2>Update Student</h2>

      <form onSubmit={handleSubmit}>

        <div>

          <label>Name:</label>

          <input

            type="text"

            value={name}

            onChange={(e) => setName(e.target.value)}

            required

          />

        </div>

        <div>

          <label>Age:</label>

          <input

            type="number"

            value={age}

            onChange={(e) => setAge(e.target.value)}

            required

          />

        </div>

        <button type="submit">Update Student</button>

      </form>

    </div>

  );

};

export default UpdateStudent;

**5] DeleteStudent.jsx**

import React from 'react';

import axios from 'axios';

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

const URL = "http://localhost:3000/students";

const DeleteStudent = ({ studentId, onStudentDeleted }) => {

  const navigate = useNavigate();

  const handleDelete = async () => {

    try {

      await axios.delete(`${URL}/${studentId}`);

      onStudentDeleted(studentId);

      navigate('/');

    } catch (error) {

      console.error('Error deleting student:', error);

    }

  };

  return (

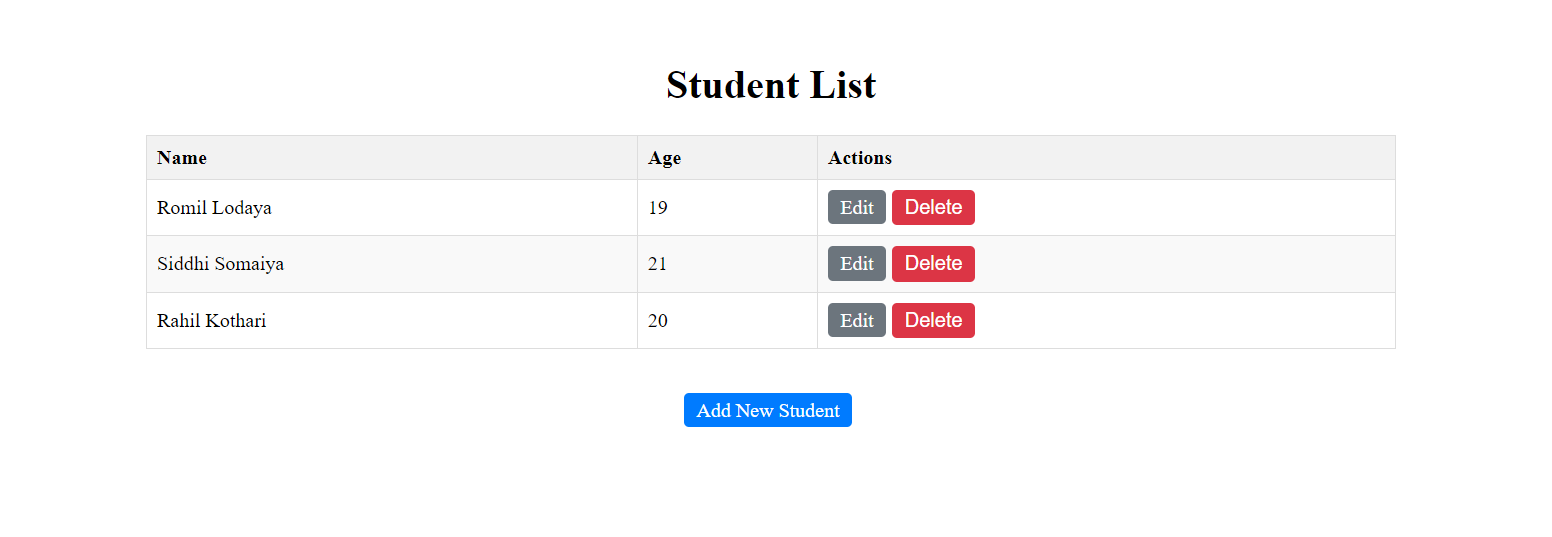
    <button onClick={handleDelete}>Delete</button>

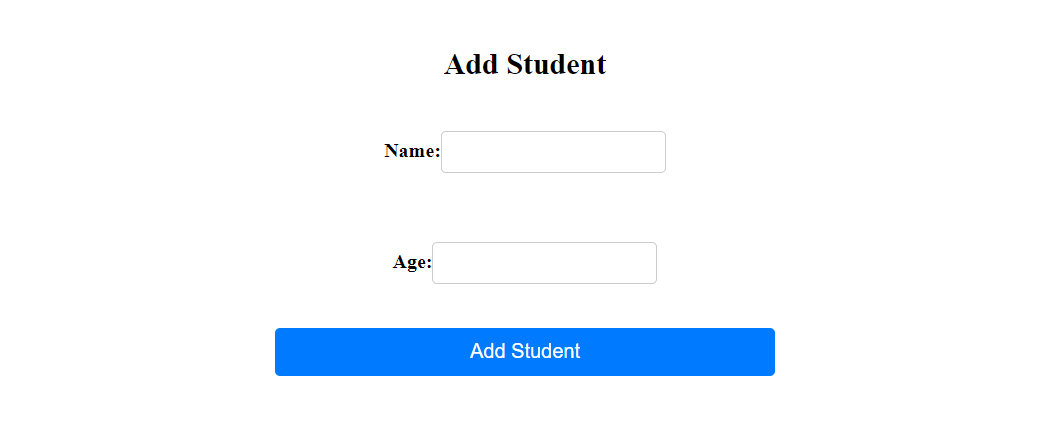
  );

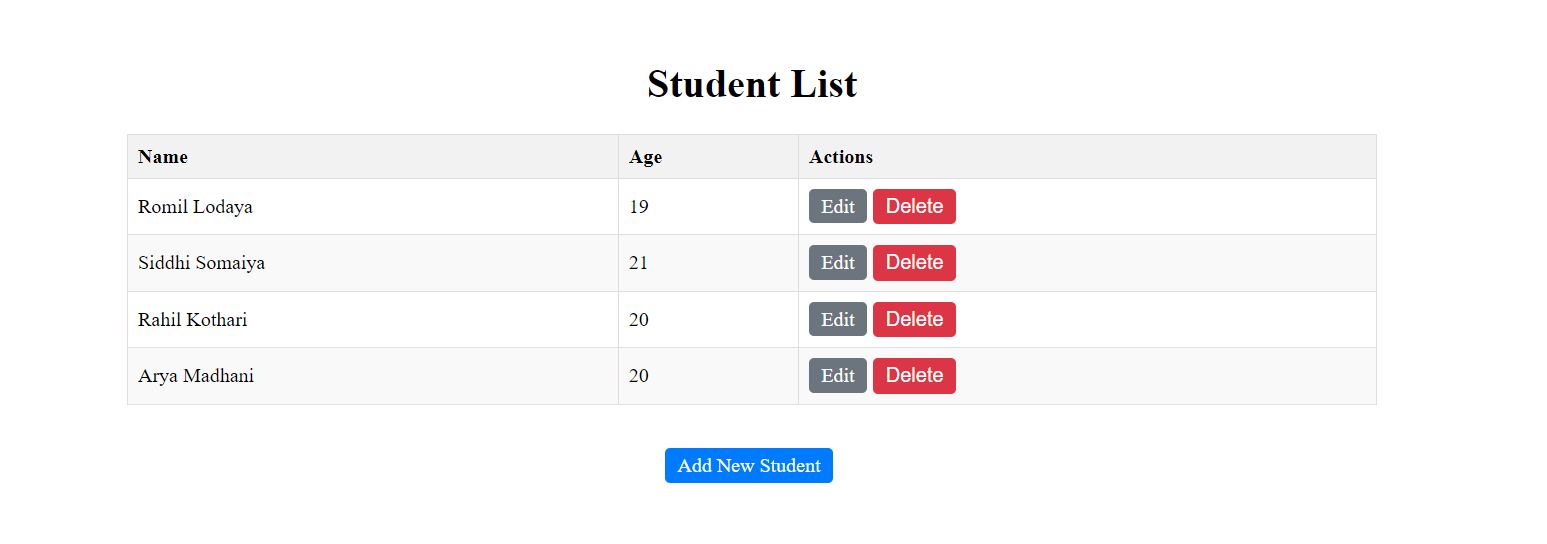
};

export default DeleteStudent;

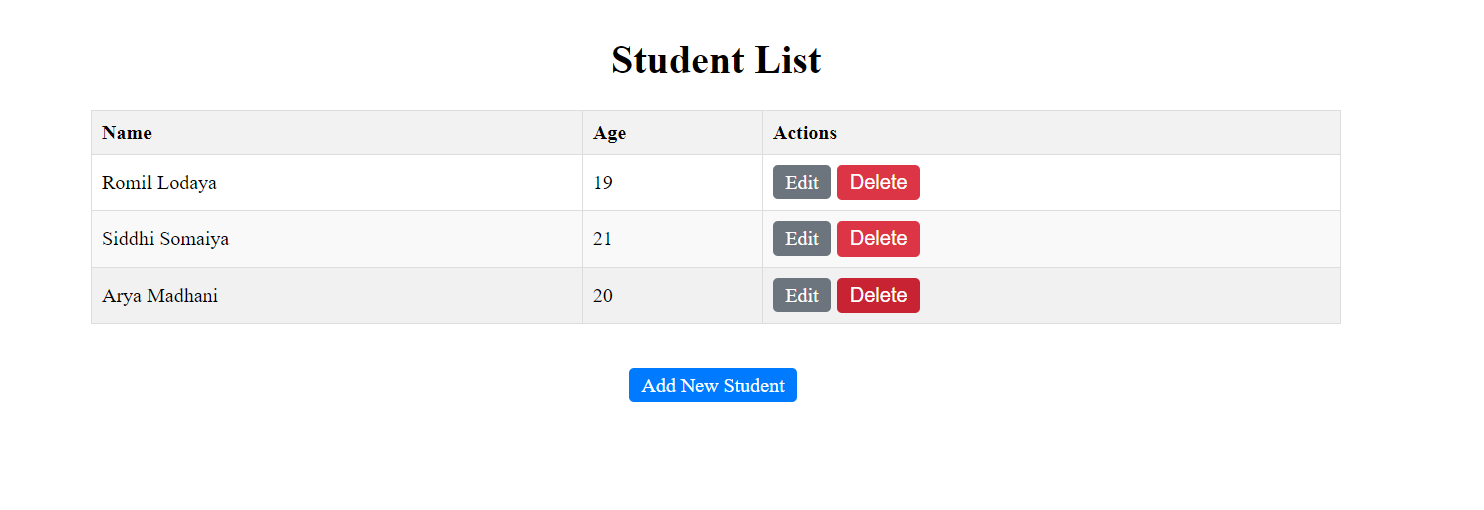
**Output:**













**Conclusion:**

Using Axios with React, we implemented CRUD operations for managing student data, demonstrating seamless navigation and data handling.

**Postlab questions:**

1) Different ways to Add Api in React/Javascript with example.

**1. Using fetch API**

fetch('https://api.example.com/users')

.then(response => response.json())

.then(data => console.log(data))

.catch(error => console.error('Error:', error));

**2. Using `axios` Library**

import axios from 'axios';

axios.get('https://api.example.com/users')

.then(response => console.log(response.data))

.catch(error => console.error('Error:', error));

**3. Using async/await with fetch**

async function fetchUsers() {

try {

const response = await fetch('https://api.example.com/users');

const data = await response.json();

console.log(data);

} catch (error) {

console.error('Error:', error);

}

}

fetchUsers()

**4. Using async/await with axios**

import axios from 'axios';

async function fetchUsers() {

try {

const response = await axios.get('https://api.example.com/users');

console.log(response.data);

} catch (error) {

console.error('Error:', error);

}

}

fetchUsers();