Step 1: Keep the Backend Intact

We'll use the backend as you originally had it. The backend will save student data to an in-memory array and allow the React frontend to fetch the data.

Backend (`server.js`)

The backend remains unchanged. It handles adding new students via a POST request and fetching the list of students via a GET request.

```
javascript
Copy code
// server.js (Express.js backend)
const express = require('express');
const cors = require('cors');
const bodyParser = require('body-parser');

const app = express();
const port = 5000;

// Middleware
app.use(cors()); // Enable CORS for frontend-backend communication
app.use(bodyParser.json()); // Parse JSON request bodies

// In-memory database for storing students (you can use a real DB later)
let students = [];

// Endpoint to add a new student
app.post('/api/students', (req, res) => {
  const newStudent = req.body;
  students.push(newStudent);
  res.status(201).json({ message: 'Student added successfully' });
});

// Endpoint to get all students
app.get('/api/students', (req, res) => {
  res.status(200).json(students);
});

// Start the server
app.listen(port, () => {
  console.log(`Server is running on http://localhost:${port}`);
});
```

Step 2: Update React Component to Interact with Backend

Now, in the React frontend, we will:

1. **Submit the student data to the backend** using a POST request.
2. **Fetch all student data from the backend** using a GET request.
3. **Display the student data below the form**.

Updated React Component (`StudentComponent.js`)

We'll modify the React component to use the backend API for both saving and fetching data.

```
javascript
Copy code
// src/StudentComponent.js
import React, { useState, useEffect } from 'react';
import axios from 'axios';

const StudentComponent = () => {
  const [students, setStudents] = useState([]); // To hold the student data
  const [newStudent, setNewStudent] = useState({ name: '', age: '' });

  // Fetch students from the backend when the component is mounted
  useEffect(() => {
    axios.get('http://localhost:5000/api/students')
      .then(response => {
        setStudents(response.data); // Update students state with data from
backend
      })
      .catch(error => {
        console.error('Error fetching students:', error);
      });
  }, []);

  // Handle form input change
  const handleChange = (e) => {
    const { name, value } = e.target;
    setNewStudent(prevState => ({ ...prevState, [name]: value }));
  };

  // Handle form submission
  const handleSubmit = (e) => {
    e.preventDefault();

    // Send the new student data to the backend
    axios.post('http://localhost:5000/api/students', newStudent)
      .then(response => {
        // After adding the student, fetch the updated student list from the
backend
        setStudents([...students, newStudent]);
        setNewStudent({ name: '', age: '' }); // Clear the form
      })
      .catch(error => {
        console.error('Error adding student:', error);
      });
  };

  return (
    <div>
      <h1>Student List</h1>

      {/* Input form to add new student */}
      <form onSubmit={handleSubmit}>
        <input
```

```
            type="text"
            name="name"
            value={newStudent.name}
            onChange={handleChange}
            placeholder="Name"
            required
          />
          <input
            type="number"
            name="age"
            value={newStudent.age}
            onChange={handleChange}
            placeholder="Age"
            required
          />
          <button type="submit">Add Student</button>
        </form>

        {/* Display the entered students below the form */}
        <div>
          <h2>Students:</h2>
          <ul>
            {students.map((student, index) => (
              <li key={index}>
                {student.name} - {student.age} years old
              </li>
            ))}
          </ul>
        </div>
      </div>
    );
};

export default StudentComponent;
```

Explanation of Changes:

1. **Fetching Data**:
   - We use the `useEffect` hook to **fetch all students** from the backend when the component mounts. This is done by making a `GET` request to `http://localhost:5000/api/students`.
   - The `students` state is updated with the data from the backend, which then gets displayed in the UI.
2. **Handling Form Submission**:
   - When the user submits the form, we send the data (name and age) to the backend via a `POST` request to `http://localhost:5000/api/students`.
   - After successfully adding the student, we **update the state** by adding the new student to the list of `students`. This will update the UI to show the new student.
3. **Displaying Students**:
   - The `students` array is mapped over to generate a list of students, which is displayed below the form.

Step 3: Ensure Backend and Frontend Are Running

1. **Run the backend** (Express server) in the `backend` folder:

   ```
   bash
   Copy code
   node server.js
   ```

   This will start the server on `http://localhost:5000`.

2. **Run the React frontend** in the `frontend` folder:

   ```
   bash
   Copy code
   npm start
   ```

   This will start the React app on `http://localhost:3000`.

Testing the Application:

- When you open your React app in the browser (typically `http://localhost:3000`), you will see the form to add a new student (name and age).
- When the user submits the form, the student data will be sent to the backend and stored there.
- After submitting the form, the student list will be **immediately updated** with the new student and displayed below the form (fetched from the backend).

Summary:

- The backend is **kept intact** and handles both the saving of data (via POST) and retrieving of data (via GET).
- The frontend **sends the data to the backend** when the user submits the form, and it **fetches the updated student list** from the backend and displays it immediately below the form.

This achieves your goal of adding and displaying student data using both the backend and frontend.