

Project Goal

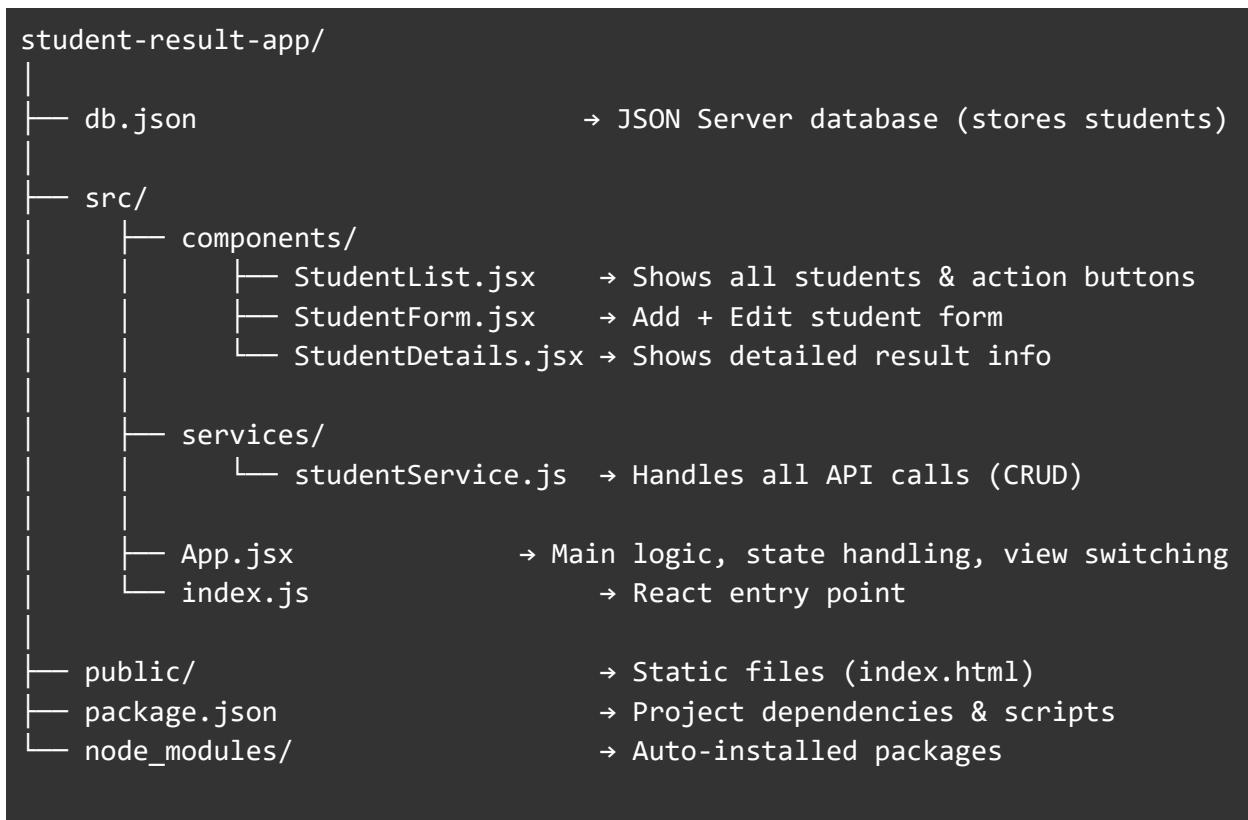
Build a React application that can **Add, View, Edit, Delete** student data using:

- React components
- useState for managing all data
- JSON Server as backend
- Fetch API for CRUD operations

The app manages:

Name, Section, Marks, Grade

Folder Structure



What Each Component Does

StudentList

- Shows all students in a table/list
- Contains buttons:
 - Load Students
 - Add Student
 - Edit
 - Delete
 - View Details

StudentForm

- Used for both Adding & Editing
- Contains input fields: name, section, marks, grade
- Uses only useState for form state

StudentDetails

- Shows complete info of one student
- Simple read-only view

studentService.js

- Handles communication with JSON Server
- Sends requests: GET, POST, PUT, DELETE

App.jsx

- Holds all main states (students, selected student, modes)
- Switches between:
 - List Screen
 - Add/Edit Form
 - Details Screen

How Students Should Handle Data (No `useEffect`)

Because they know only `useState`, all data fetching will happen through manual user actions.

- Use a “Load Students” button to fetch data
- After adding/editing/deleting → show an alert and let them click the “Load Students” button again
- All API calls happen only inside:
 - Button click handlers
 - Form submission

This keeps it simple and beginner-friendly.

CRUD Flow

1. Create (Add Student)

Fill the form → Submit → Student saved in JSON Server.

2. Read (View Student List)

Click “Load Students” → All students displayed in StudentList.

3. Update (Edit Student)

Click “Edit” → Form opens → Update information → Save changes.

4. Delete (Remove Student)

Click “Delete” → Student removed from JSON Server.

5. View Details

Click “View” → Show full result details (Marks, Grade, Section, Name).

Note for Students

This project guide covers the **minimum required features** you must implement:

- Add Student
- View Students
- Edit Student
- Delete Student
- View Student Details
- Use React components + useState
- Use JSON Server for storing data
- Perform all CRUD operations through button clicks and form submissions

You must complete all these basic requirements.

If you want to score better or practice more, you are **free to add extra features**, such as:

- Search or filter
- Sorting
- Pagination
- Form validation
- Better UI/Styling
- Loading indicators

Adding extra functionality is optional, but highly encouraged.