**StudentResultManagement.jsx**

import React, { useState } from 'react';

import './style.css';

const StudentResultManagement = () => {

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

const [editingStudent, setEditingStudent] = useState(null);

const addStudent = (student) => {

setStudents([...students, { ...student, id: Date.now() }]);

};

const editStudent = (updatedStudent) => {

setStudents(

students.map((student) =>

student.id === updatedStudent.id ? updatedStudent : student

)

);

setEditingStudent(null);

};

const deleteStudent = (id) => {

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

};

const calculateStudentAverage = (scores) => {

const total = scores.reduce((sum, score) => sum + score, 0);

return (total / scores.length).toFixed(2);

};

const calculateClassAverage = () => {

if (students.length === 0) return 0;

const total = students.reduce((sum, student) => {

const studentAverage = calculateStudentAverage([

student.math,

student.science,

student.english,

]);

return sum + parseFloat(studentAverage);

}, 0);

return (total / students.length).toFixed(2);

};

return (

<div className="student-result-management">

<h1>Student Result Management System</h1>

<AddStudentForm addStudent={addStudent} />

{editingStudent && (

<EditStudentForm

editingStudent={editingStudent}

editStudent={editStudent}

/>

)}

<StudentList

students={students}

deleteStudent={deleteStudent}

setEditingStudent={setEditingStudent}

calculateStudentAverage={calculateStudentAverage}

/>

<div className="average-score">

<h3>Class Average: {calculateClassAverage()}</h3>

</div>

</div>

);

};

const StudentList = ({

students,

deleteStudent,

setEditingStudent,

calculateStudentAverage,

}) => {

return (

<div className="student-list">

<h2>Student Results</h2>

<table>

<thead>

<tr>

<th>Name</th>

<th>Roll Number</th>

<th>Math</th>

<th>Science</th>

<th>English</th>

<th>Average</th>

<th>Actions</th>

</tr>

</thead>

<tbody>

{students.map((student) => (

<tr key={student.id}>

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

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

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

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

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

<td>

{calculateStudentAverage([

student.math,

student.science,

student.english,

])}

</td>

<td>

<button onClick={() => setEditingStudent(student)}>Edit</button>

<button onClick={() => deleteStudent(student.id)}>Delete</button>

</td>

</tr>

))}

</tbody>

</table>

</div>

);

};

const AddStudentForm = ({ addStudent }) => {

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

const [rollNumber, setRollNumber] = useState('');

const [math, setMath] = useState('');

const [science, setScience] = useState('');

const [english, setEnglish] = useState('');

const handleSubmit = (e) => {

e.preventDefault();

addStudent({

name,

rollNumber,

math: parseFloat(math),

science: parseFloat(science),

english: parseFloat(english),

});

setName('');

setRollNumber('');

setMath('');

setScience('');

setEnglish('');

};

return (

<div className="add-student-form">

<h2>Add New Student</h2>

<form onSubmit={handleSubmit}>

<label for="nm">Student Name</label>

<input

type="text"

placeholder="Name"

id="nm"

value={name}

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

required

/>

<label for="rn">Roll number</label>

<input

type="text"

id="rn"

placeholder="Roll Number"

value={rollNumber}

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

required

/>

<label for="s1">Subject 1</label>

<input

type="number"

id="s1"

value={math}

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

required

/>

Subject2

<input

type="number"

placeholder="Science Score"

value={science}

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

required

/>

Subject3

<input

type="number"

placeholder="English Score"

value={english}

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

required

/>

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

</form>

</div>

);

};

const EditStudentForm = ({ editingStudent, editStudent }) => {

const [name, setName] = useState(editingStudent.name);

const [rollNumber, setRollNumber] = useState(editingStudent.rollNumber);

const [math, setMath] = useState(editingStudent.math);

const [science, setScience] = useState(editingStudent.science);

const [english, setEnglish] = useState(editingStudent.english);

const handleSubmit = (e) => {

e.preventDefault();

editStudent({

...editingStudent,

name,

rollNumber,

math: parseFloat(math),

science: parseFloat(science),

english: parseFloat(english),

});

};

return (

<div className="edit-student-form">

<h2>Edit Student</h2>

<form onSubmit={handleSubmit}>

<input

type="text"

placeholder="Name"

value={name}

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

required

/>

<input

type="text"

placeholder="Roll Number"

value={rollNumber}

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

required

/>

<input

type="number"

placeholder="Math Score"

value={math}

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

required

/>

<input

type="number"

placeholder="Science Score"

value={science}

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

required

/>

<input

type="number"

placeholder="English Score"

value={english}

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

required

/>

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

</form>

</div>

);

};

export default StudentResultManagement;

**Style.css**

body {

font-family: Arial, sans-serif;

margin: 0;

padding: 0;

}

.student-result-management {

max-width: 800px;

margin: 20px auto;

padding: 20px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

}

h1, h2, h3 {

color: #ffffff;

}

/\* Forms \*/

.add-student-form, .edit-student-form {

margin-bottom: 20px;

}

form {

display: flex;

flex-direction: column;

gap: 10px;

}

input {

padding: 8px;

/\* border: 1px solid #ccc; \*/

border-radius: 4px;

}

button {

padding: 8px 12px;

background-color

: #28a745;

color: white;

border: none;

border-radius: 4px;

cursor: pointer;

}

button:hover {

background-color: #218838;

}

/\* Table \*/

.student-list table {

width: 100%;

border-collapse: collapse;

margin-bottom: 20px;

}

.student-list th, .student-list td {

padding: 10px;

border: 1px solid #ffffff;

text-align: center;

}

/\* Actions \*/

.actions button {

margin: 0 5px;

}

.actions button:first-child {

background-color: #ffc107;

}

.actions button:first-child:hover {

background-color: #e0a800;

}

.actions button:last-child {

background-color: #dc3545;

}

.actions button:last-child:hover {

background-color: #c82333;

}

/\* Average Score \*/

.average-score {

text-align: center;

font-size: 1.2em;

color: #333;

}

**OUTPUT** :

