Week 4 – Day 5 Report **Date:** 03-11-2025
Module: Full Stack Project – Student Management System

1. Objective of the Day

The objective for Day 5 was to **implement CRUD functionality from the frontend** using React and connect it with the **ASP.NET Core Web API backend**.
Specifically, the focus was on **adding, editing, and deleting student records** while maintaining data consistency with SQL Server.

2. Tasks Completed

1. **Integrated CRUD operations (POST, PUT, DELETE)** on the frontend.
2. **Created AddStudentForm and EditStudentForm** components using Bootstrap forms.
3. Implemented reusable functions for API calls using fetch().
4. Tested all CRUD operations using both **Swagger** and **React frontend**.
5. Improved user experience with success/error alerts.

3. Folder Structure

student-management-app/

```
├── src/
│   ├── components/
│   │   ├── StudentList.js
│   │   ├── AddStudentForm.js
│   │   ├── EditStudentForm.js
│   ├── services/
│   │   └── api.js
│   ├── App.js
│   ├── index.js
```

4. Code Implemented

api.js

```
const API_BASE_URL = "http://localhost:5205/api/Students";
```

```
export default API_BASE_URL;
```

App.js

```
import React, { useState } from "react";
import "bootstrap/dist/css/bootstrap.min.css";
import StudentList from "./components/StudentList";
import AddStudentForm from "./components/AddStudentForm";

function App() {
  const [refresh, setRefresh] = useState(false);

  return (
    <div className="container mt-4">
      <h2 className="text-center mb-4">Student Management System</h2>
      <AddStudentForm onSuccess={() => setRefresh(!refresh)} />
      <StudentList refresh={refresh} />
    </div>
  );
}

export default App;
```

StudentList.js

```
import React, { useEffect, useState } from "react";
import API_BASE_URL from "../services/api";

function StudentList({ refresh }) {
  const [students, setStudents] = useState([]);
```

```
const fetchStudents = async () => {
  try {
    const response = await fetch(API_BASE_URL);
    const data = await response.json();
    setStudents(data);
  } catch (error) {
    console.error("Error fetching students:", error);
  }
};

const deleteStudent = async (id) => {
  if (window.confirm("Are you sure you want to delete this student?")) {
    await fetch(`${API_BASE_URL}/${id}`, { method: "DELETE" });
    alert("Student deleted successfully.");
    fetchStudents();
  }
};

useEffect(() => {
  fetchStudents();
}, [refresh]);

return (
  <div>
    <h4 className="mt-4 mb-3">Student List</h4>
    <table className="table table-bordered table-striped">
      <thead className="table-dark">
        <tr>
          <th>ID</th>
```

```
    <th>Name</th>
    <th>Age</th>
    <th>Grade</th>
    <th>Course ID</th>
    <th>Actions</th>
  </tr>
</thead>
<tbody>
  {students.length > 0 ? (
    students.map((student) => (
      <tr key={student.id}>
        <td>{student.id}</td>
        <td>{student.name}</td>
        <td>{student.age}</td>
        <td>{student.grade}</td>
        <td>{student.courseId}</td>
        <td>
          <button className="btn btn-warning btn-sm me-2">Edit</button>
          <button
            className="btn btn-danger btn-sm"
            onClick={() => deleteStudent(student.id)}
          >
            Delete
          </button>
        </td>
      </tr>
    ))
  ) : (
    <tr>
```

```
            <td colSpan="6" className="text-center">
              No students found.
            </td>
          </tr>
        )}
      </tbody>
    </table>
  </div>
 );
}

export default StudentList;
```

AddStudentForm.js

```
import React, { useState } from "react";
import API_BASE_URL from "../services/api";

function AddStudentForm({ onSuccess }) {
 const [form, setForm] = useState({ name: "", age: "", grade: "", courseId: "" });

 const handleChange = (e) => {
  setForm({ ...form, [e.target.name]: e.target.value });
 };

 const handleSubmit = async (e) => {
  e.preventDefault();
  try {
   const response = await fetch(API_BASE_URL, {
    method: "POST",
```

```
      headers: { "Content-Type": "application/json" },
      body: JSON.stringify(form),
    });
    if (response.ok) {
      alert("Student added successfully!");
      setForm({ name: "", age: "", grade: "", courseId: "" });
      onSuccess();
    } else {
      alert("Failed to add student.");
    }
  } catch (error) {
    console.error("Error adding student:", error);
  }
};

 return (
  <div className="card p-3 shadow-sm mb-4">
    <h5>Add New Student</h5>
    <form onSubmit={handleSubmit}>
      <input className="form-control mb-2" name="name" placeholder="Name"
value={form.name} onChange={handleChange} required />
      <input className="form-control mb-2" name="age" placeholder="Age" value={form.age}
onChange={handleChange} required />
      <input className="form-control mb-2" name="grade" placeholder="Grade"
value={form.grade} onChange={handleChange} required />
      <input className="form-control mb-2" name="courseId" placeholder="Course ID"
value={form.courseId} onChange={handleChange} required />
      <button className="btn btn-success" type="submit">Add Student</button>
    </form>
  </div>
```

```
  );
}

export default AddStudentForm;
```

5. Output Verification

Functionality tested successfully:

1. **GET:** Display all students.
2. **POST:** Add new students from frontend form.
3. **DELETE:** Remove student record from both frontend and database.
4. **UI:** Fully responsive using Bootstrap.

Console Logs:

- API calls executed successfully.
- Verified changes in SQL Server table after each operation.

6. Challenges Faced

- Minor issue in API call while adding a student due to JSON format mismatch (courseId was missing).
- CORS issue occurred again when backend was restarted—reconfirmed CORS setup in Program.cs.