

Subject :





Name : Shivam Chaturvedi



Lib.ID

: 2426mca201



Section : D

Full stack development with java

|  |
| --- |
| 1.     Witte a CRUD Web Application using React JS or JavaScript frontend with Spring Boot as backend |

**React Js:**

import  { useEffect, useState } from "react";

function App() {

  const [items, setItems] = useState([]);

  const [newItemName, setNewItemName] = useState("");

  const [editId, setEditId] = useState(null);

  const [editName, setEditName] = useState("");

  const API\_URL = "http://localhost:8080/api";

  const fetchItems = async () => {

    const res = await fetch(API\_URL);

    const data = await res.json();

    setItems(data);

  };

  const addItem = async () => {

    if (!newItemName.trim()) return;

    await fetch(API\_URL, {

      method: "POST",

      headers: { "Content-Type": "application/json" },

      body: JSON.stringify({ name: newItemName }),

    });

    setNewItemName("");

    fetchItems();

  };

  const updateItem = async (id:number) => {

    if (!editName.trim()) return;

    await fetch(`${API\_URL}/${id}`, {

      method: "PUT",

      headers: { "Content-Type": "application/json" },

      body: JSON.stringify({ name: editName }),

    });

    setEditId(null);

    setEditName("");

    fetchItems();

  };

  const deleteItem = async (id:number) => {

    await fetch(`${API\_URL}/${id}`, { method: "DELETE" });

    fetchItems();

  };

  useEffect(() => {

    fetchItems();

  }, []);

  return (

    <div className="w-full flex flex-col items-center border-4 p-4">

      <h1 className="text-3xl font-bold mb-4">CRUD App Assignment 5  (Spring Boot + React)</h1>

      <div className="mb-4">

        <input

          className="border p-2 mr-2"

          type="text"

          value={newItemName}

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

          placeholder="Enter item name"

        />

        <button className="bg-blue-500 text-white px-4 py-2" onClick={addItem}>

          Add

        </button>

      </div>

      <ul className="w-full max-w-md">

        {items.map((item:object) => (

          <li

            key={item?.id}

            className="flex justify-between items-center border p-2 mb-2"

          >

            {editId === item?.id ? (

              <>

                <input

                  className="border p-1 mr-2"

                  value={editName}

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

                />

                <button

                  className="bg-green-500 text-white px-2 py-1 mr-1"

                  onClick={() => updateItem(item?.id)}

                >

                  Save

                </button>

                <button

                  className="bg-gray-400 text-white px-2 py-1"

                  onClick={() => setEditId(null)}

                >

                  Cancel

                </button>

              </>

            ) : (

              <>

                <span>{item.name}</span>

                <div>

                  <button

                    className="bg-yellow-500 text-white px-2 py-1 mr-1"

                    onClick={() => {

                      setEditId(item.id);

                      setEditName(item.name);

                    }}

                  >

                    Edit

                  </button>

                  <button

                    className="bg-red-500 text-white px-2 py-1"

                    onClick={() => deleteItem(item.id)}

                  >

                    Delete

                  </button>

                </div>

              </>

            )}

          </li>

        ))}

      </ul>

    </div>

  );

}

export default App;

Spring Boot Code:

package com.assignment5.SimpleApi.API;

import org.springframework.web.bind.annotation.\*;

import com.assignment5.SimpleApi.model.Item;

import java.util.\*;

import java.util.concurrent.atomic.AtomicLong;

*@RestController*

*@RequestMapping*("/api")

*@CrossOrigin*(origins = "http://localhost:5173")

public class Api {

private final List<Item> items = new ArrayList<>();

private final AtomicLong idGenerator = new AtomicLong();

*@GetMapping*

public List<Item> getItems() {

return items;

}

*@PostMapping*

public Item addItem(*@RequestBody* Item item) {

item.setId(idGenerator.incrementAndGet());

items.add(item);

return item;

}

*@PutMapping*("/{id}")

public Item updateItem(*@PathVariable* Long id, *@RequestBody* Item updatedItem) {

for (Item item : items) {

if (item.getId().equals(id)) {

item.setName(updatedItem.getName());

return item;

}

}

throw new NoSuchElementException("Item not found");

}

*@DeleteMapping*("/{id}")

public void deleteItem(*@PathVariable* Long id) {

items.removeIf(item -> item.getId().equals(id));

}

}

Model Class:

package com.assignment5.SimpleApi.model;

public class Item {

private Long id;

private String name;

public Item() {}

public Item(String name) {

this.name = name;

}

public void setId(Long id) {

this.id=id;

}

public Long getId() {

return id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

}

OUTPUT:

