```
import React, { useState, useEffect } from "react";
import RuleModal from "../components/RuleModal";
import Notification from "../components/Notification";

const ReportsPage = () => {
  const API_BASE = "http://localhost:8080/api"; // adjust if backend runs elsewhere

  const [rules, setRules] = useState([]);
  const [showRuleModal, setShowRuleModal] = useState(false);
  const [currentRule, setCurrentRule] = useState({
    id: null,
    name: "",
    path: "",
    uploadedBy: "",
  });
  const [notification, setNotification] = useState(null);

  // 🔹 Fetch all reports
  const fetchReports = async () => {
    try {
      const res = await fetch(`${API_BASE}/fetch-all-report`);
      const data = await res.json();
      setRules(data);
    } catch (err) {
      console.error("Error fetching reports", err);
    }
  };

  // 🔹 Add new report
  const addReport = async (rule) => {
    try {
      const res = await fetch(`${API_BASE}/newreport`, {
        method: "POST",
        headers: { "Content-Type": "application/json" },
        body: JSON.stringify(rule),
      });
      if (res.ok) {
        fetchReports();
        setNotification({ type: "success", message: "Report added successfully!" });
      }
    } catch (err) {
      console.error("Error adding report", err);
    }
  };

  // 🔹 Update report
  const updateReport = async (rule) => {
    try {
```

```
    const res = await fetch(`${API_BASE}/updatereport/${rule.id}`, {
      method: "PUT",
      headers: { "Content-Type": "application/json" },
      body: JSON.stringify(rule),
    });
    if (res.ok) {
      fetchReports();
      setNotification({ type: "success", message: "Report updated successfully!" });
    }
  } catch (err) {
    console.error("Error updating report", err);
  }
};

// 🔹 Delete report
const deleteReport = async (id) => {
  try {
    const res = await fetch(`${API_BASE}/deletereport/${id}`, {
      method: "DELETE",
    });
    if (res.ok) {
      fetchReports();
      setNotification({ type: "success", message: "Report deleted successfully!" });
    }
  } catch (err) {
    console.error("Error deleting report", err);
  }
};

// 🔹 Handlers
const handleSaveRule = () => {
  if (currentRule.id) {
    updateReport(currentRule);
  } else {
    addReport(currentRule);
  }
  setShowRuleModal(false);
};

const handleDeleteRule = (ruleId) => {
  if (window.confirm("Are you sure you want to delete this report?")) {
    deleteReport(ruleId);
  }
};

const handleEditRule = (rule) => {
  setCurrentRule(rule);
  setShowRuleModal(true);
```

```
};

const handleAddRule = () => {
  setCurrentRule({ id: null, name: "", path: "", uploadedBy: "" });
  setShowRuleModal(true);
};

//  🔹 Load reports on mount
useEffect(() => {
  fetchReports();
}, []);

return (
  <div className="container mt-4">
    <h2>Reports</h2>
    <button className="btn btn-primary mb-3" onClick={handleAddRule}>
      Add Report
    </button>

    {notification && (
      <Notification
        type={notification.type}
        message={notification.message}
        onClose={() => setNotification(null)}
      />
    )}

    <table className="table table-bordered">
      <thead>
        <tr>
          <th>ID</th>
          <th>Report Name</th>
          <th>Path</th>
          <th>Uploaded By</th>
          <th>Actions</th>
        </tr>
      </thead>
      <tbody>
        {rules.map((rule) => (
          <tr key={rule.id}>
            <td>{rule.id}</td>
            <td>{rule.name}</td>
            <td>{rule.path}</td>
            <td>{rule.uploadedBy}</td>
            <td>
              <button
                className="btn btn-sm btn-warning me-2"
                onClick={() => handleEditRule(rule)}
```

```
              >
                Edit
              </button>
              <button
                className="btn btn-sm btn-danger"
                onClick={() => handleDeleteRule(rule.id)}
              >
                Delete
              </button>
            </td>
          </tr>
        ))}
      </tbody>
    </table>

    {showRuleModal && (
      <RuleModal
        rule={currentRule}
        setRule={setCurrentRule}
        onSave={handleSaveRule}
        onClose={() => setShowRuleModal(false)}
      />
    )}
  </div>
 );
};

export default ReportsPage;
```

```
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.web.servlet.config.annotation.CorsRegistry;
import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;

@Configuration
public class CorsConfig {
    @Bean
    public WebMvcConfigurer corsConfigurer() {
        return new WebMvcConfigurer() {
            @Override
            public void addCorsMappings(CorsRegistry registry) {
                registry.addMapping("/api/**")
                        .allowedOrigins("http://localhost:3000") // your React app
                        .allowedMethods("GET", "POST", "PUT", "DELETE");
            }
        };
    }
}
```