AddReport.js

import React, { useState, useEffect } from "react";

import axios from "axios";

import "./AddReport.css";

const AddReport = () => {

    const [title, setTitle] = useState("");

    const [description, setDescription] = useState("");

    const [category, setCategory] = useState("");

    const [year, setYear] = useState("");

    const [month, setMonth] = useState("");

    const [document, setDocument] = useState(null);

    const [reports, setReports] = useState([]);

    // Handle File Selection

    const handleDocumentChange = (e) => {

        const file = e.target.files[0];

        setDocument(file);

    };

    // Handle Form Submission

    const handleSubmit = async (e) => {

        e.preventDefault();

        const formData = new FormData();

        formData.append("title", title);

        formData.append("description", description);

        formData.append("category", category);

        formData.append("year", year);

        formData.append("month", month);

        if (document) {

            formData.append("document", document);

        }

            const token = localStorage.getItem("adminToken");

           // const token = localStorage.getItem("adminToken");

            console.log("Token being sent in request:", token); // Log token

            try {

            await axios.post("http://localhost:5000/api/reports/add-report", formData, {

                headers: {

                    "Content-Type": "multipart/form-data",

                    Authorization: `Bearer ${token}`,

                },

            });

            alert("Report added successfully!");

            fetchReports();

        } catch (error) {

            alert("Failed to add report");

        }

    };

    // Fetch Reports

    const fetchReports = async () => {

        try {

            const token = localStorage.getItem("adminToken");

            const { data } = await axios.get("http://localhost:5000/api/reports/get-reports", {

                headers: { Authorization: `Bearer ${token}` },

            });

            setReports(data);

        } catch (error) {

            console.error("Error fetching reports:", error);

        }

    };

    useEffect(() => {

        fetchReports();

    }, []);

    return (

        <div className="reports-container">

            {/\* Add Report Section \*/}

            <div className="add-report-section">

                <h2>Add Report</h2>

                <form onSubmit={handleSubmit}>

                    <input type="text" placeholder="Title" value={title} onChange={(e) => setTitle(e.target.value)} required />

                    <textarea placeholder="Description" value={description} onChange={(e) => setDescription(e.target.value)} required />

                    <select value={category} onChange={(e) => setCategory(e.target.value)} required>

                        <option value="">Select Category</option>

                        <option value="Workshops">Workshops</option>

                        <option value="Alumni Meetups">Alumni Meetups</option>

                        <option value="Cultural Events">Cultural Events</option>

                        <option value="Others">Others</option>

                    </select>

                    <input type="number" placeholder="Year" value={year} onChange={(e) => setYear(e.target.value)} required />

                    <input type="number" placeholder="Month" value={month} onChange={(e) => setMonth(e.target.value)} required />

                    <input type="file" onChange={handleDocumentChange} accept="application/pdf" />

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

                </form>

            </div>

            {/\* Reports List Section \*/}

            <div className="reports-list-section">

                <h2>Reports List</h2>

                <ul className="reports-list">

                    {reports.map((report) => (

                        <li key={report.\_id} className="report-card">

                            <div className="report-content">

                                <h3 className="report-title">{report.title}</h3>

                                <p className="report-description">{report.description}</p>

                                {report.document && (

                                    <a href={`http://localhost:5000${report.document}`} target="\_blank" rel="noopener noreferrer">

                                        View Document

                                    </a>

                                )}

                            </div>

                        </li>

                    ))}

                </ul>

            </div>

        </div>

    );

};

export default AddReport;

Reports.js

import React, { useState, useEffect } from "react";

import axios from "axios";

import "./Reports.css";

const Reports = () => {

  const [reports, setReports] = useState([]);

  const [filteredReports, setFilteredReports] = useState([]);

  const [expandedId, setExpandedId] = useState(null);

  const [categories, setCategories] = useState([]);

  const [selectedCategory, setSelectedCategory] = useState("All");

  const [searchYear, setSearchYear] = useState("");

  useEffect(() => {

    fetchReports();

  }, []);

  const fetchReports = async () => {

    try {

      const response = await axios.get("http://localhost:5000/api/reports");

      setReports(response.data);

      setFilteredReports(response.data);

      // Extract unique categories

      const uniqueCategories = [...new Set(response.data.map((report) => report.category))];

      setCategories(uniqueCategories);

    } catch (error) {

      console.error("Error fetching reports:", error);

    }

  };

  // Handle category filter

  const handleCategoryChange = (event) => {

    setSelectedCategory(event.target.value);

    filterReports(event.target.value, searchYear);

  };

  // Handle search input

  const handleSearchChange = (event) => {

    setSearchYear(event.target.value);

    filterReports(selectedCategory, event.target.value);

  };

  // Filtering function

  const filterReports = (category, year) => {

    let filtered = reports;

    if (category !== "All") {

      filtered = filtered.filter((report) => report.category === category);

    }

    if (year.trim() !== "") {

      filtered = filtered.filter((report) => new Date(report.date).getFullYear().toString() === year);

    }

    setFilteredReports(filtered);

  };

  const toggleExpand = (id) => {

    setExpandedId(expandedId === id ? null : id);

  };

  return (

    <div className="reports-container">

      <h1 className="title">📜 Reports</h1>

      <div className="filters">

        <select className="filter-dropdown" value={selectedCategory} onChange={handleCategoryChange}>

          <option value="All">All Categories</option>

          {categories.map((category, index) => (

            <option key={index} value={category}>{category}</option>

          ))}

        </select>

        <input

          type="text"

          className="search-bar"

          placeholder="Search by Year (e.g. 2023)"

          value={searchYear}

          onChange={handleSearchChange}

        />

      </div>

      <div className="reports-grid">

        {filteredReports.length > 0 ? (

          filteredReports.map((report) => (

            <div

              key={report.\_id}

              className={`report-card ${expandedId === report.\_id ? "expanded" : ""}`}

              onClick={() => toggleExpand(report.\_id)}

            >

              <h2 className="report-title">{report.title}</h2>

              {expandedId === report.\_id && (

                <div className="report-details">

                  {report.document && (

                    <a href={`http://localhost:5000${report.document}`} target="\_blank" rel="noopener noreferrer">

                      View Document

                    </a>

                  )}

                  <p className="report-description">{report.description}</p>

                  <p className="report-category">📂 {report.category}</p>

                  <p className="report-date">📅 {new Date(report.date).toDateString()}</p>

                </div>

              )}

            </div>

          ))

        ) : (

          <p className="no-reports">No reports found 😕</p>

        )}

      </div>

    </div>

  );

};

export default Reports;

Model

const mongoose = require("mongoose");

const ReportSchema = new mongoose.Schema({

  title: String,

  description: String, // Detailed report content

  category: {

    type: String,

    enum: ["Workshops", "Alumni Meetups", "Cultural Events", "Others"], // Categories for filtering

  },

  year: Number, // Year for grouping

  month: Number, // Month for filtering

  document: String, // Store image URL

  date: { type: Date, default: Date.now }, // Original event date

});

module.exports = mongoose.model("Report", ReportSchema);

Adminreport/routes

const express = require("express");

const multer = require("multer");

const Report = require("../models/Report");

//const ActivityLog = require('../models/ActivityLog');  // Adjust the path to match your project structure

const adminAuthMiddleware = require("../middleware/adminAuthMiddleware");

const router = express.Router();

// Configure Multer for file uploads (storing images in memory)

//const storage = multer.memoryStorage();

const storage = multer.diskStorage({

    destination: function (req, file, cb) {

        cb(null, "uploads/");

    },

    filename: function (req, file, cb) {

        cb(null, Date.now() + "-" + file.originalname);

    }

});

const upload = multer({ storage });

// Add Report API

router.post("/add-report", adminAuthMiddleware, upload.single("document"), async (req, res) => {

    try {

        const { title, description, category, year, month } = req.body;

       // const image = req.file ? req.file.buffer : null; // Store as Buffer

        const newReport = new Report({

            title,

            description,

            category,

            year,

            month,

            document: req.file? `/uploads/${req.file.filename}` : null, // Store image as Buffer instead of Base64

        });

        await newReport.save();

        res.status(201).json({ message: "Report added successfully", report: newReport });

    } catch (error) {

        console.error("Error adding report:", error);

        res.status(500).json({ message: "Failed to add report" });

    }

});

// Get all reports

router.get("/get-reports", adminAuthMiddleware, async (req, res) => {

    try {

        const reports = await Report.find().sort({ date: -1 });

        res.status(200).json(reports);

    } catch (error) {

        console.error("Error fetching reports:", error);

        res.status(500).json({ message: "Failed to fetch reports" });

    }

});

router.get("/view-document/:id", async (req, res) => {

    try {

        const report = await Report.findById(req.params.id);

        if (!report || !report.image) {

            return res.status(404).json({ message: "Document not found" });

        }

        res.sendFile(report.image, { root: "uploads/" }); // Adjust storage path

    } catch (error) {

        res.status(500).json({ message: "Error fetching document", error });

    }

});

module.exports = router;

{groupedInternships[alumniName].internships.map((internship) => (

reportRoutes

const express = require("express");

const router = express.Router();

const Report = require("../models/Report");

// ✅ Get all reports (with optional filtering)

router.get("/", async (req, res) => {

  try {

    let query = {};

    if (req.query.category && req.query.category !== "All") {

      query.category = req.query.category;

    }

    const reports = await Report.find(query);

    res.json(reports);

  } catch (error) {

    res.status(500).json({ message: "Error fetching reports", error });

  }

});

// ✅ Add a new report (Only for Alumni Incharge)

router.post("/", async (req, res) => {

  try {

    const { title, description, category, image, date } = req.body;

    const newReport = new Report({ title, description, category, image, date });

    await newReport.save();

    res.status(201).json({ message: "Report added successfully" });

  } catch (error) {

    res.status(500).json({ message: "Error adding report", error });

  }

});

module.exports = router;