const express = require('express');

const router = express.Router();

const { createOrUpdateProfile, getProfile, updateProfile, upload } = require('../controllers/clientProfileController');

const protect  = require('../middleware/authMiddleware')

// Create or update profile (with file upload)

router.post('/', protect, upload.any('profilePicture','gallery'), createOrUpdateProfile);  // Create or update profile

router.get('/:id', getProfile);  // Get profile by ID

// Update profile (with file upload)

router.put('/edit',protect, upload.any('profilePicture','gallery'), updateProfile);

module.exports = router;

const ClientProfile = require('../models/ClientProfile');

const multer = require('multer');

const path = require('path');

const fs = require('fs');

// Ensure the 'uploads/profile-pictures' directory exists

if (!fs.existsSync('uploads/profile-pictures')) {

  fs.mkdirSync('uploads/profile-pictures', { recursive: true });

}

// Set up multer storage for profile pictures

const storage = multer.diskStorage({

  destination: (req, file, cb) => {

    cb(null, 'uploads/profile-pictures');

  },

  filename: (req, file, cb) => {

    cb(null, `${req.user.id}-${Date.now()}${path.extname(file.originalname)}`);

  },

});

const upload = multer({ storage });

// Create or update profile

const createOrUpdateProfile = async (req, res) => {

    const {

      fullName,

      email,

      phone,

      location,

      description,

      category,

      friendlyAddress,

      videoUrl,

      profileShow,

      website,

      foundedDate,

      employees,

      responseTime

    } = req.body;

    try {

      // Find profile by user ID (assuming user is logged in and req.user.id is available)

      let profile = await ClientProfile.findOne({ user: req.user.id });

      if (profile) {

        // Update profile fields

        profile.fullName = fullName || profile.fullName;

        profile.email = email || profile.email;

        profile.phone = phone || profile.phone;

        profile.location = location || profile.location;

        profile.description = description || profile.description;

        profile.category = category || profile.category;

        profile.friendlyAddress = friendlyAddress || profile.friendlyAddress;

        profile.videoUrl = videoUrl || profile.videoUrl;

        profile.profileShow = profileShow || profile.profileShow;

        profile.website = website || profile.website;

        profile.foundedDate = foundedDate || profile.foundedDate;

        profile.employees = employees || profile.employees;

        profile.responseTime = responseTime || profile.responseTime;

        if (req.file) {

          // If a new profile image is uploaded, update it

          profile.profilePicture = `/uploads/profile-pictures/${req.file.filename}`;

          profile.gallery = `/uploads/profile-pictures/${req.file.filename}`;

        }

        // Save updated profile

        await profile.save();

        return res.status(200).json(profile);

      } else {

        // Create a new profile if it doesn't exist

        const newProfile = new ClientProfile({

          user: req.user.id,

          fullName,

          email,

          phone,

          location,

          description,

          category,

          friendlyAddress,

          videoUrl,

          profileShow,

          website,

          foundedDate,

          employees,

          responseTime,

          profilePicture: req.file ? `/uploads/profile-pictures/${req.file.filename}` : '',

          gallery: req.file ? `/uploads/profile-pictures/${req.file.filename}` : ''

        });

        await newProfile.save();

        return res.status(201).json(newProfile);

      }

    } catch (error) {

      console.error('Error creating or updating profile:', error.message);

      res.status(500).json({ message: 'Error creating or updating profile' });

    }

  };

// Get profile by user ID

const getProfile = async (req, res) => {

  try {

    const profile = await ClientProfile.findOne({ user: req.params.id }).populate('user', ['fullName', 'email']);

    if (!profile) {

      return res.status(404).json({ message: 'Profile not found' });

    }

    res.json(profile);

  } catch (error) {

    console.error('Error fetching profile:', error);

    res.status(500).json({ message: 'Error fetching profile' });

  }

};

// Update profile

const updateProfile = async (req, res) => {

  const {

    fullName,

    email,

    phone,

    location,

    description,

    category,

    friendlyAddress,

    videoUrl,

    profileShow,

    website,

    foundedDate,

    employees,

    responseTime,

  } = req.body;

  try {

    const profilePicture = req.file ? `/uploads/profile-pictures/${req.file.filename}` : null;

    const gallery = req.file ? `/uploads/profile-pictures/${req.file.filename}` : null;

    const updatedProfile = await ClientProfile.findOneAndUpdate(

      { user: req.user.id },

      {

        fullName,

        email,

        phone,

        location,

        description,

        category,

        friendlyAddress,

        videoUrl,

        profileShow,

        website,

        foundedDate,

        employees,

        responseTime,

        profilePicture: profilePicture || undefined,

        gallery: gallery || undefined,

      },

      { new: true }

    );

    if (!updatedProfile) {

      return res.status(404).json({ message: 'Profile not found' });

    }

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

  } catch (error) {

    console.error('Error updating profile:', error);

    res.status(500).json({ message: 'Error updating profile' });

  }

};

module.exports = {

  createOrUpdateProfile,

  getProfile,

  updateProfile,

  upload,

};

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

import { Editor } from '@tinymce/tinymce-react';

import axios from '../utils/axios'; // Make sure you have axios configured for API calls

import { useParams } from 'react-router-dom';  // To get userId from URL

import { useSelector } from 'react-redux';  // To get userId from Redux

const ClientProfile = () => {

  // const { userId: paramUserId } = useParams();  // Get userId from route params

  const userId = useSelector((state) => state.auth.userId);  // Get userId from Redux

  // const userId = reduxUserId;  // Use paramUserId if it exists, else use reduxUserId

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

  const [profilePicture, setprofilePicture] = useState(null);

  const [fullName, setFullName] = useState('');

  const [email, setEmail] = useState('');

  const [phone, setPhone] = useState('');

  const [location, setLocation] = useState('');

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

  const [friendlyAddress, setFriendlyAddress] = useState('');

  const [gallery, setGallery] = useState(null);

  const [videoUrl, setVideoUrl] = useState('');

  const [profileShow, setProfileShow] = useState('show');

  const [website, setWebsite] = useState('');

  const [foundedDate, setFoundedDate] = useState('');

  const [employees, setEmployees] = useState('');

  const [responseTime, setResponseTime] = useState('');

  // Fetch profile data

  useEffect(() => {

    const fetchProfile = async () => {

      try {

        const { data } = await axios.get(`/clientprofile/${userId}`); // Endpoint to fetch the logged-in user's profile

        setFullName(data.fullName);

        setEmail(data.email);

        setPhone(data.phone);

        setLocation(data.location);

        setDescription(data.description);

        setCategory(data.category);

        setFriendlyAddress(data.friendlyAddress);

        setVideoUrl(data.videoUrl);

        setProfileShow(data.profileShow);

        setWebsite(data.website);

        setFoundedDate(data.foundedDate);

        setEmployees(data.employees);

        setResponseTime(data.responseTime);

        setprofilePicture(data.profilePicture);

        setGallery(data.gallery)

      } catch (error) {

        console.error('Error fetching profile:', error);

      }

    };

      if(userId){

        fetchProfile();

      }

  }, [userId]);

  const handleEditorChange = (content) => {

    setDescription(content);

  };

  const handleSaveProfile = async () => {

    const formData = new FormData();

    formData.append('profilePicture', profilePicture); // 'profilePicture' matches the backend schema

    formData.append('fullName', fullName);

    formData.append('email', email);

    formData.append('phone', phone);

    formData.append('location', location);

    formData.append('description', description);

    formData.append('category', category);

    formData.append('friendlyAddress', friendlyAddress);

    formData.append('gallery', gallery);

    formData.append('videoUrl', videoUrl);

    formData.append('profileShow', profileShow);

    formData.append('website', website);

    formData.append('foundedDate', foundedDate);

    formData.append('employees', employees);

    formData.append('responseTime', responseTime);

    try {

      const response = await axios.post('/clientprofile', formData, {

        headers: {

          'Content-Type': 'multipart/form-data'

        }

      });

      console.log('Profile saved successfully:', response.data);

    } catch (error) {

      console.error('Error saving profile:', error.response?.data || error.message);

    }

  };

  return (

    <div className="max-w-6xl mx-auto p-6">

      <h1 className="text-3xl font-bold mb-8 text-center">Edit Profile</h1>

      {/\* Profile Image \*/}

      <div className="mb-6">

        <label className="block text-lg font-semibold mb-2">Profile Image</label>

        <input type="file" onChange={(e) => setprofilePicture(e.target.files[0])} className="block w-full" />

      </div>

      {/\* Full Name, Email, Phone \*/}

      <div className="mb-6 grid grid-cols-1 md:grid-cols-2 gap-6">

        <div>

          <label className="block text-lg font-semibold mb-2">Full Name</label>

          <input

            type="text"

            value={fullName}

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

            className="w-full p-2 border border-gray-300 rounded-md"

          />

        </div>

        <div>

          <label className="block text-lg font-semibold mb-2">Email</label>

          <input

            type="email"

            value={email}

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

            className="w-full p-2 border border-gray-300 rounded-md"

          />

        </div>

        <div>

          <label className="block text-lg font-semibold mb-2">Phone Number</label>

          <input

            type="text"

            value={phone}

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

            className="w-full p-2 border border-gray-300 rounded-md"

          />

        </div>

        <div>

          <label className="block text-lg font-semibold mb-2">Response Time</label>

          <input

            type="text"

            value={responseTime}

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

            className="w-full p-2 border border-gray-300 rounded-md"

          />

        </div>

      </div>

      {/\* Show Profile, Category \*/}

      <div className="mb-6 grid grid-cols-1 md:grid-cols-2 gap-6">

        <div>

          <label className="block text-lg font-semibold mb-2">Show my Profile</label>

          <select

            value={profileShow}

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

            className="w-full p-2 border border-gray-300 rounded-md"

          >

            <option value="show">Show</option>

            <option value="hide">Hide</option>

          </select>

        </div>

        <div>

          <label className="block text-lg font-semibold mb-2">Category</label>

          <select

            value={category}

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

            className="w-full p-2 border border-gray-300 rounded-md"

          >

            <option value="business">Business</option>

            <option value="designer">Designer</option>

            <option value="digital-marketing">Digital Marketing</option>

            <option value="lifestyle">Lifestyle</option>

            <option value="programming-tech">Programming & Tech</option>

            <option value="project-managers">Project Managers</option>

            <option value="web-developers">Web Developers</option>

            <option value="writing-translation">Writing & Translation</option>

          </select>

        </div>

      </div>

      {/\* Website, Founded Date, Employees, Location \*/}

      <div className="mb-6 grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-2 gap-6">

        <div>

          <label className="block text-lg font-semibold mb-2">Website</label>

          <input

            type="text"

            value={website}

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

            className="w-full p-2 border border-gray-300 rounded-md"

          />

        </div>

        <div>

          <label className="block text-lg font-semibold mb-2">Founded Date</label>

          <input

            type="number"

            value={foundedDate}

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

            className="w-full p-2 border border-gray-300 rounded-md"

          />

        </div>

        <div>

          <label className="block text-lg font-semibold mb-2">Employees</label>

          <input

            type="number"

            value={employees}

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

            className="w-full p-2 border border-gray-300 rounded-md"

          />

        </div>

        <div>

          <label className="block text-lg font-semibold mb-2">Location</label>

          <input

            type="text"

            value={location}

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

            className="w-full p-2 border border-gray-300 rounded-md"

          />

        </div>

      </div>

      {/\* Friendly Address \*/}

      <div className="mb-6">

        <label className="block text-lg font-semibold mb-2">Friendly Address</label>

        <input

          type="text"

          value={friendlyAddress}

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

          className="w-full p-2 border border-gray-300 rounded-md"

        />

      </div>

      {/\* Description \*/}

      <div className="mb-6">

        <label className="block text-lg font-semibold mb-2">Description</label>

        <Editor

          apiKey='65gcyx9y0yezfpmk2p7xn336dpzjwn2walh2cekmrxk5o9k7'

          value={description}

          init={{

            height: 300,

            menubar: false,

            plugins: [

              'advlist autolink lists link image charmap print preview anchor',

              'searchreplace visualblocks code fullscreen',

              'insertdatetime media table paste code help wordcount',

            ],

            toolbar:

              'undo redo | formatselect | bold italic backcolor | ' +

              'alignleft aligncenter alignright alignjustify | ' +

              'bullist numlist outdent indent | removeformat | help',

          }}

          onEditorChange={handleEditorChange}

        />

      </div>

      {/\* Media Upload \*/}

      <div className="mb-6">

        <label className="block text-lg font-semibold mb-2">Gallery Image</label>

        <input type="file" onChange={(e) => setGallery(e.target.files[0])} className="block w-full" />

      </div>

      <div className="mb-6">

        <label className="block text-lg font-semibold mb-2">Introduction Video URL</label>

        <input

          type="text"

          value={videoUrl}

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

          className="w-full p-2 border border-gray-300 rounded-md"

        />

      </div>

      {/\* Save Button \*/}

      <button

        onClick={handleSaveProfile}

        className="px-6 py-3 bg-green-500 text-white font-semibold rounded-md hover:bg-green-600"

      >

        Save Profile

      </button>

    </div>

  );

};

export default ClientProfile;

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

import { FaMapMarkerAlt, FaClock, FaStar, FaEnvelope, FaPhone, FaUsers , FaUserCircle, FaBriefcase, FaCalendarAlt, FaLanguage, FaTransgender } from 'react-icons/fa';

// import Footer from '../Real DashBoard/Footer';

// import Navbar from '../Real DashBoard/Navbar';

import { useParams } from 'react-router-dom';  // To get userId from URL

import { useSelector } from 'react-redux';  // To get userId from Redux

import axios from '../utils/axios'

const ViewClientProfile = () => {

  const { userId: paramUserId } = useParams();  // Get userId from route params

  const reduxUserId = useSelector((state) => state.auth.userId);  // Get userId from Redux

  const userId = paramUserId || reduxUserId;  // Use paramUserId if it exists, else use reduxUserId

  const [profile, setProfile] = useState(null);

  useEffect(() => {

    const fetchProfile = async () => {

      try {

        const response = await axios.get(`/clientprofile/${userId}`);

        setProfile(response.data);

      } catch (error) {

        console.error('Error fetching profile:', error);

      }

    };

    if (userId) {

      fetchProfile();

    }

  }, [userId]);

  if (!profile) {

    return <p>Loading profile...</p>;

  }

  return (

    <>

    <div className="min-h-screen bg-gray-100 ">

      {/\* Header Section \*/}

      <div className="bg-orange-50 text-black py-10 px-6 lg:flex lg:items-center lg:justify-between">

        <div className="lg:flex lg:items-center">

          <img

            src="https://via.placeholder.com/150"

            alt="Freelancer"

            className="w-32 h-32 lg:w-48 lg:h-48 rounded-full object-cover mr-6"

          />

          <div>

            <h1 className="text-3xl font-bold">{profile.fullName}</h1>

            <p className="text-lg mt-2">Nursing Assistant</p>

            <div className="flex items-center space-x-6 mt-4">

              <span className="flex items-center">

                <FaEnvelope className="mr-2 text-gray-600" />

                {profile.email}

              </span>

              <span className="flex items-center">

                <FaMapMarkerAlt className="mr-2 text-gray-600" />

                Location: {profile.location}

              </span>

              <span className=" px-3 py-1 rounded-full text-sm">

                ⭐ 4.9 (52 Reviews)

              </span>

            </div>

          </div>

        </div>

      </div>

      <div className="container mx-auto mt-10 p-4 lg:p-10 lg:flex lg:space-x-8">

        {/\* Left Column: Freelancer Information \*/}

        <div className="lg:w-2/3 space-y-8">

          {/\* About Me Section \*/}

          <div className=" p-6">

            <h3 className="text-2xl font-bold mb-4">About Company</h3>

            <p>

              {profile.description}</p>

          </div>

          {/\* Posted Services Section \*/}

          <div className=" p-6">

            <h3 className="text-2xl font-bold mb-4">Posted Services</h3>

            <div className="grid grid-cols-1 md:grid-cols-2 gap-6">

              <div className="bg-white border border-gray-200 rounded-lg shadow-lg p-4">

                <img

                  src="https://freeio.blogdu.de/wp-content/uploads/2022/11/service11-768x576.jpg"

                  alt="Service Image"

                  className="w-full h-40 object-cover rounded-md mb-4"

                />

                <div className="text-gray-500 text-sm">Design & Creative</div>

                <h3 className="text-lg font-semibold text-gray-900">Developers drop the framework folder into a...</h3>

                <div className="flex items-center mt-2 mb-4">

                  <FaStar className="text-yellow-400" />

                  <span className="ml-1 text-gray-700 font-bold">4.5</span>

                  <span className="ml-2 text-gray-500">(2 Reviews)</span>

                </div>

                <div className="flex items-center justify-between">

                  <div className="text-sm text-gray-500">John Paul</div>

                  <div className="text-gray-900 font-bold">Starting at: $128.00</div>

                </div>

              </div>

              {/\* Add more services here if needed \*/}

            </div>

          </div>

         {/\* Reviews Section \*/}

        <div className="mt-6 p-4 sm:w-full md:w-full lg:w-full">

          <h3 className="text-xl font-bold mb-4">3 Reviews</h3>

          <div className="space-y-4">

            {/\* Review 1 \*/}

            <div className="bg-orange-100 p-4 rounded-lg">

              <p className="font-semibold">Admin</p>

              <p className="text-yellow-500">⭐⭐⭐⭐☆</p>

              <p className="text-gray-600">Lorem ipsum dolor sit amet, consectetur.</p>

            </div>

            {/\* Review 2 \*/}

            <div className="bg-orange-100 p-4 rounded-lg">

              <p className="font-semibold">Ali Tufan</p>

              <p className="text-yellow-500">⭐⭐⭐⭐⭐</p>

              <p className="text-gray-600 ">Vivamus vehicula sodales est, eu rhoncus urna semper eu.</p>

            </div>

          </div>

          {/\* Add a Review Form \*/}

          <div className="mt-6 p-4 ">

            <h4 className="font-bold mb-4">Add a Review</h4>

            <form className="space-y-6">

              {/\* Textarea for Review \*/}

              <div>

                <label className="block text-sm font-medium text-gray-700 mb-1">Your Review</label>

                <textarea

                  placeholder="Write your review..."

                  rows="4"

                  className="border p-3 w-full h-52 rounded-lg focus:ring-2 focus:ring-green-500 focus:outline-none"

                ></textarea>

              </div>

              {/\* Name and Email side by side \*/}

              <div className="flex space-x-4">

                <div className="flex-1">

                  <label className="block text-sm font-medium text-gray-700 mb-1">Your Name</label>

                  <input

                    type="text"

                    placeholder="Your Name"

                    className="border p-3 w-full rounded-lg focus:ring-2 focus:ring-green-500 focus:outline-none"

                  />

                </div>

                <div className="flex-1">

                  <label className="block text-sm font-medium text-gray-700 mb-1">Your Email</label>

                  <input

                    type="email"

                    placeholder="Your Email"

                    className="border p-3 w-full rounded-lg focus:ring-2 focus:ring-green-500 focus:outline-none"

                  />

                </div>

              </div>

              {/\* Rating Section \*/}

              <div className="flex items-center">

                <label className="block text-sm font-medium text-gray-700 mb-1 mr-4">Your Rating for this Listing</label>

                <div className="flex space-x-2">

                  {/\* Stars for rating \*/}

                  <span className="text-yellow-500">★</span>

                  <span className="text-yellow-500">★</span>

                  <span className="text-yellow-500">★</span>

                  <span className="text-yellow-500">★</span>

                  <span className="text-gray-300">★</span>

                </div>

              </div>

              {/\* Submit Button \*/}

              <button

                type="submit"

                className="bg-green-900 text-white py-2 px-6 rounded-lg hover:bg-green-950 transition duration-200"

              >

                Submit Review

              </button>

            </form>

          </div>

        </div>

        </div>

       <div className="lg:w-1/3 space-y-8">

  {/\* About the Seller Widget \*/}

  <div className="bg-white shadow-md rounded-lg p-6">

    <ul className="space-y-4">

      <li className="flex items-center">

        <FaBriefcase className="text-gray-500 mr-2" />

        Categories: {profile.category}

      </li>

      <li className="flex items-center">

        <FaUsers className="text-gray-500 mr-2" />

        Employees: {profile.employees}

      </li>

      <li className="flex items-center">

        <FaCalendarAlt className="text-gray-500 mr-2" />

        Founded Date: {profile.foundedDate}

      </li>

      <li className="flex items-center">

        <FaEnvelope className="text-gray-500 mr-2" />

        Email: {profile.email}

      </li>

      <li className="flex items-center">

        <FaPhone className="text-gray-500 mr-2" />

        Phone Number: {profile.phone}

      </li>

      <li className="flex items-center">

        <FaMapMarkerAlt className="text-gray-500 mr-2" />

        Location: {profile.location}

      </li>

    </ul>

  </div>

          {/\* Skills Section \*/}

        </div>

      </div>

    </div>

    {/\* <Footer /> \*/}

    </>

  );

};

export default ViewClientProfile;