<!DOCTYPE html>

<html lang="fr">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>JurisNota - Le Blog des Étudiants</title>

<script src="https://cdn.tailwindcss.com"></script>

<link href="https://fonts.googleapis.com/css2?family=Inter:wght@400;600;700&display=swap" rel="stylesheet">

<style>

body {

font-family: 'Inter', sans-serif;

}

.rating-scale {

display: flex;

justify-content: space-between;

align-items: center;

width: 100%;

padding: 0 10px;

position: relative;

}

.rating-line {

position: absolute;

left: 0;

top: 50%;

transform: translateY(-50%);

height: 4px;

width: 100%;

background-color: #e5e7eb;

border-radius: 2px;

}

.rating-point {

width: 24px;

height: 24px;

background-color: #d1d5db;

border-radius: 50%;

display: flex;

align-items: center;

justify-content: center;

font-size: 12px;

font-weight: bold;

color: #4b5563;

cursor: pointer;

z-index: 10;

transition: all 0.2s ease-in-out;

}

.rating-point.selected {

background-color: #10b981;

color: white;

}

#info-popup {

position: fixed;

top: 50%;

left: 50%;

transform: translate(-50%, -50%);

z-index: 50;

}

</style>

</head>

<body class="bg-gray-100 flex flex-col items-center justify-center min-h-screen">

<!-- Info Pop-up -->

<div id="info-popup" class="bg-white rounded-2xl shadow-2xl p-6 md:p-8 max-w-sm w-full text-center">

<h2 class="text-2xl font-bold text-gray-800 mb-4">Bienvenue sur JurisNota</h2>

<p class="text-sm text-gray-600 mb-4">

Ce site est anonyme. Nous ne collectons aucune donnée sur vous, à l'exception des notes et des commentaires que vous postez. Vos droits RGPD sont respectés, et vous pouvez à tout moment demander la suppression de votre contenu. En utilisant ce site, vous vous engagez à rester bienveillant. Tout commentaire insultant sera supprimé. Votre témoignage se doit d'être fidèle à la réalité. Votre adresse mail est demandée afin de pouvoir vous répondre efficacement. Elle sera supprimée dès qu'une réponse vous sera envoyée.

</p>

<button id="close-popup-btn" class="bg-green-500 text-white font-semibold py-2 px-6 rounded-full hover:bg-green-600 transition-colors duration-300 shadow-lg">

J'ai compris !

</button>

</div>

<!-- Main Content -->

<div id="main-content" class="hidden flex flex-col items-center justify-center min-h-screen w-full">

<header class="text-center my-4">

<h1 class="text-5xl md:text-6xl font-extrabold text-green-800 mb-2">JurisNota</h1>

<p class="text-md text-gray-600">Note tes cours - aide les autres</p>

</header>

<section class="bg-white rounded-2xl shadow-lg p-6 md:p-8 my-8 text-center max-w-xl w-full">

<p class="text-3xl text-gray-800 font-bold">Par les étudiants, pour les étudiants...</p>

<p class="text-sm text-gray-600">

<span class="font-bold">Pourquoi JurisNota ?</span> JurisNota est un espace collaboratif pour les étudiants en droit. L'objectif est de s'entraider en partageant des avis, des conseils et des notes sur les cours. Marre d'être constamment noté ? Sois l'acteur de l'information étudiante !

</p>

</section>

<main class="container mx-auto p-4 md:p-8 max-w-xl w-full">

<section id="level-selection-section" class="bg-white rounded-2xl shadow-lg p-6 md:p-8 my-8 text-center">

<h2 id="main-title" class="text-3xl italic text-gray-800 mb-6">Sélectionnez votre année</h2>

<div id="level-selector" class="flex flex-col items-center space-y-4 mb-8">

<button class="level-button w-full bg-green-500 text-white font-semibold py-2 px-6 rounded-full hover:bg-green-600 transition-colors duration-300 shadow-lg" data-level="L1">L1</button>

<button class="level-button w-full bg-green-500 text-white font-semibold py-2 px-6 rounded-full hover:bg-green-600 transition-colors duration-300 shadow-lg" data-level="L2">L2</button>

<button class="level-button w-full bg-green-500 text-white font-semibold py-2 px-6 rounded-full hover:bg-green-600 transition-colors duration-300 shadow-lg" data-level="L3">L3</button>

</div>

</section>

</main>

<section id="course-section" class="bg-white rounded-2xl shadow-lg p-6 md:p-8 my-8 hidden max-w-xl w-full">

<h2 id="course-title" class="text-3xl font-bold text-gray-800 mb-2"></h2>

<div class="flex items-center justify-center text-xl text-gray-600 mb-6">

<span class="font-semibold mr-2">Note moyenne :</span>

<span id="average-rating" class="font-bold">0</span> / 10

</div>

<div class="bg-gray-50 rounded-lg p-4 mb-8">

<h3 class="text-xl font-semibold text-gray-800 mb-4">Laisser une note et un commentaire</h3>

<form id="rating-form">

<div class="mb-4">

<label class="block text-gray-700 font-medium mb-2">Votre note (1-10) : <span class="text-red-500">\*</span></label>

<div id="rating-scale" class="rating-scale relative my-4">

<div class="rating-line"></div>

</div>

</div>

<div class="mb-4">

<label for="comment-input" class="block text-gray-700 font-medium mb-2">Votre commentaire : <span class="text-red-500">\*</span></label>

<textarea id="comment-input" name="comment" rows="4" required

class="w-full px-4 py-2 border border-gray-300 rounded-lg focus:outline-none focus:ring-2 focus:ring-green-500"></textarea>

</div>

<button type="submit" class="bg-green-500 text-white font-semibold py-2 px-6 rounded-full hover:bg-green-600 transition-colors duration-300 shadow-lg">

Publier

</button>

</form>

</div>

<div id="comments-section">

<h3 class="text-xl font-semibold text-gray-800 mb-4">Commentaires</h3>

<div id="comments-list" class="space-y-4">

</div>

</div>

<button id="back-from-course" class="mt-8 bg-gray-200 text-gray-800 font-semibold py-2 px-6 rounded-full hover:bg-gray-300 transition-colors duration-300">

Retour aux années

</button>

</section>

<!-- Contact Form Section -->

<section id="contact-form-section" class="bg-white rounded-2xl shadow-lg p-6 md:p-8 my-8 hidden max-w-xl w-full">

<h2 class="text-3xl font-bold text-gray-800 mb-6">Formulaire de contact</h2>

<form id="contact-form" class="space-y-4">

<div>

<label for="contact-topic" class="block text-gray-700 font-medium mb-2">Objet de la demande : <span class="text-red-500">\*</span></label>

<select id="contact-topic" name="topic" required

class="w-full px-4 py-2 border border-gray-300 rounded-lg focus:outline-none focus:ring-2 focus:ring-green-500">

<option value="" disabled selected>-- Sélectionnez un sujet --</option>

<option value="Suppression de commentaire">Suppression de commentaire</option>

<option value="Demande autre">Demande autre</option>

</select>

</div>

<div>

<label for="contact-message" class="block text-gray-700 font-medium mb-2">Développez votre demande : <span class="text-red-500">\*</span></label>

<textarea id="contact-message" name="message" rows="4" required

class="w-full px-4 py-2 border border-gray-300 rounded-lg focus:outline-none focus:ring-2 focus:ring-green-500"></textarea>

</div>

<div>

<label for="contact-email" class="block text-gray-700 font-medium mb-2">Votre adresse mail : <span class="text-red-500">\*</span></label>

<input type="email" id="contact-email" name="email" required

class="w-full px-4 py-2 border border-gray-300 rounded-lg focus:outline-none focus:ring-2 focus:ring-green-500">

</div>

<button type="submit" id="submit-contact-form" disabled

class="w-full bg-gray-400 text-white font-semibold py-2 px-6 rounded-full cursor-not-allowed">

Envoyer la demande

</button>

</form>

<button id="back-from-contact" class="mt-8 bg-gray-200 text-gray-800 font-semibold py-2 px-6 rounded-full hover:bg-gray-300 transition-colors duration-300">

Retour à la page principale

</button>

<div id="contact-message-status" class="mt-4 text-center"></div>

</section>

<!-- Contact Success Section -->

<section id="contact-success-section" class="bg-white rounded-2xl shadow-lg p-6 md:p-8 my-8 hidden max-w-xl w-full">

<div class="text-center p-8">

<h2 class="text-2xl font-bold text-gray-800 mb-4">Demande envoyée !</h2>

<p class="text-gray-600">Nous te remercions de ta demande ! Tu recevras une réponse dans un court délai.</p>

<button id="back-from-contact-success" class="mt-8 bg-green-500 text-white font-semibold py-2 px-6 rounded-full hover:bg-green-600 transition-colors duration-300">

Retour à la page principale

</button>

</div>

</section>

<!-- Admin Section (hidden by default) -->

<section id="admin-section" class="bg-white rounded-2xl shadow-lg p-6 md:p-8 my-8 hidden max-w-xl w-full text-center">

<h2 class="text-3xl font-bold text-gray-800 mb-6">Connexion Admin</h2>

<form id="admin-form" class="space-y-4">

<div>

<label for="admin-password" class="block text-gray-700 font-medium mb-2">Mot de passe</label>

<input type="password" id="admin-password" name="password" required

class="w-full px-4 py-2 border border-gray-300 rounded-lg focus:outline-none focus:ring-2 focus:ring-green-500 text-center">

</div>

<button type="submit" class="bg-green-500 text-white font-semibold py-2 px-6 rounded-full hover:bg-green-600 transition-colors duration-300 shadow-lg">

Se connecter

</button>

<div id="admin-message" class="text-red-500 mt-2 hidden"></div>

</form>

<button id="back-from-admin" class="mt-8 bg-gray-200 text-gray-800 font-semibold py-2 px-6 rounded-full hover:bg-gray-300 transition-colors duration-300">

Retour à la page principale

</button>

</section>

<!-- Admin Dashboard Section (hidden by default) -->

<section id="admin-dashboard-section" class="bg-white rounded-2xl shadow-lg p-6 md:p-8 my-8 hidden max-w-2xl w-full">

<div class="flex justify-between items-center mb-6">

<h2 class="text-3xl font-bold text-gray-800">Tableau de bord Admin</h2>

<button id="back-from-admin-dashboard" class="bg-gray-200 text-gray-800 font-semibold py-2 px-6 rounded-full hover:bg-gray-300 transition-colors duration-300">

Déconnexion

</button>

</div>

<div class="flex justify-center space-x-4 mb-8">

<button id="show-comments-tab" class="px-6 py-2 rounded-full font-semibold bg-green-500 text-white hover:bg-green-600 transition-colors duration-300">Commentaires</button>

<button id="show-troublemakers-tab" class="px-6 py-2 rounded-full font-semibold bg-gray-200 text-gray-800 hover:bg-gray-300 transition-colors duration-300">Gens casse couilles</button>

</div>

<div id="admin-comments-section">

<h3 class="text-xl font-bold text-gray-800 mb-4">Tous les commentaires</h3>

<div id="admin-comments-list" class="space-y-8">

<!-- Comments will be dynamically inserted here -->

</div>

</div>

<div id="admin-troublemakers-section" class="hidden">

<h3 class="text-xl font-bold text-gray-800 mb-4">Demandes de contact</h3>

<div id="admin-troublemakers-list" class="space-y-4">

<!-- Contact submissions will be dynamically inserted here -->

</div>

</div>

</section>

</div>

<footer class="bg-gray-800 text-white py-8 rounded-t-xl mt-8 w-full">

<div class="container mx-auto text-center">

<div class="flex justify-center space-x-6 mb-4">

<a href="#" id="admin-link" class="text-gray-400 hover:text-white transition-colors duration-300">Admin</a>

<a href="#" id="contact-link" class="text-gray-400 hover:text-white transition-colors duration-300">Formulaire de contact</a>

</div>

<p>&copy; 2024 JurisNota. Tous droits réservés.</p>

</div>

</footer>

<script type="module">

import { initializeApp } from "https://www.gstatic.com/firebasejs/11.6.1/firebase-app.js";

import { getAuth, signInAnonymously, signInWithCustomToken, onAuthStateChanged } from "https://www.gstatic.com/firebasejs/11.6.1/firebase-auth.js";

import { getFirestore, collection, onSnapshot, query, addDoc, doc, deleteDoc } from "https://www.gstatic.com/firebasejs/11.6.1/firebase-firestore.js";

import { setLogLevel } from "https://www.gstatic.com/firebasejs/11.6.1/firebase-firestore.js";

// Set log level for debugging

setLogLevel('debug');

// Retrieve global variables

const appId = typeof \_\_app\_id !== 'undefined' ? \_\_app\_id : 'default-app-id';

const firebaseConfig = typeof \_\_firebase\_config !== 'undefined' ? JSON.parse(\_\_firebase\_config) : {};

const initialAuthToken = typeof \_\_initial\_auth\_token !== 'undefined' ? \_\_initial\_auth\_token : null;

// Initialize Firebase

const app = initializeApp(firebaseConfig);

const db = getFirestore(app);

const auth = getAuth(app);

// Get DOM elements

const levelSelectorSection = document.getElementById('level-selection-section');

const mainTitle = document.getElementById('main-title');

const levelSelector = document.getElementById('level-selector');

const courseSection = document.getElementById('course-section');

const courseTitle = document.getElementById('course-title');

const averageRatingSpan = document.getElementById('average-rating');

const ratingForm = document.getElementById('rating-form');

const ratingScale = document.getElementById('rating-scale');

const commentInput = document.getElementById('comment-input');

const commentsList = document.getElementById('comments-list');

const backFromCourseButton = document.getElementById('back-from-course');

const contactFormSection = document.getElementById('contact-form-section');

const contactLink = document.getElementById('contact-link');

const contactForm = document.getElementById('contact-form');

const contactMessageStatus = document.getElementById('contact-message-status');

const backFromContactButton = document.getElementById('back-from-contact');

const contactTopicInput = document.getElementById('contact-topic');

const contactMessageInput = document.getElementById('contact-message');

const contactEmailInput = document.getElementById('contact-email');

const submitContactFormButton = document.getElementById('submit-contact-form');

const contactSuccessSection = document.getElementById('contact-success-section');

const adminSection = document.getElementById('admin-section');

const adminDashboardSection = document.getElementById('admin-dashboard-section');

const adminLink = document.getElementById('admin-link');

const adminForm = document.getElementById('admin-form');

const adminPasswordInput = document.getElementById('admin-password');

const adminMessageDiv = document.getElementById('admin-message');

const backFromAdminButton = document.getElementById('back-from-admin');

const backFromAdminDashboardButton = document.getElementById('back-from-admin-dashboard');

const showCommentsTab = document.getElementById('show-comments-tab');

const showTroublemakersTab = document.getElementById('show-troublemakers-tab');

const adminCommentsSection = document.getElementById('admin-comments-section');

const adminTroublemakersSection = document.getElementById('admin-troublemakers-section');

const adminCommentsList = document.getElementById('admin-comments-list');

const adminTroublemakersList = document.getElementById('admin-troublemakers-list');

const popup = document.getElementById('info-popup');

const closePopupBtn = document.getElementById('close-popup-btn');

const mainContent = document.getElementById('main-content');

let userId = null;

let allNotes = [];

let allContactSubmissions = [];

let currentCourseTitle = null;

let currentLevel = null;

let selectedRating = null;

// Static course data

const staticCourses = {

"L1": [

"Droit privé et théorie générale du droit",

"Droit constitutionnel 1",

"Histoire du droit et des institutions publiques",

"Économie politique",

"Institutions judiciaires",

"Droit constitutionnel 2",

"Droit de la famille",

"Institutions administratives",

"Relations internationales",

"Macroéconomie"

],

"L2": [

"Droit des obligations 1",

"Droit administratif 1",

"Droit institutionnel de l'Union européenne",

"Droit processuel",

"Finances publiques",

"Grands systèmes juridiques",

"Histoire du droit des obligations",

"Introduction to the English Legal System ",

"Droit des obligations 2",

"Droit administratif 2",

"Contrats spéciaux",

"Droit commercial",

"Droit des politiques et actions de l'Union Européenne",

"Histoire de l'Etat et des grands services publics",

"Introduction to UK Public Law",

"Vie politique contemporaine"

],

"L3": [

"Droit des obligations 3",

"Droit des sociétés 1",

"Droit du travail 1",

"Droit pénal",

"Libertés publiques",

"Droit de la responsabilité administrative",

"Introduction au droit international",

"English Tort Law",

"Droit allemand approfondi 1",

"Droit des cultes et des religions",

"Philosophie du droit",

"Droit des biens",

"Procédure civile",

"Droit des sociétés 2",

"Droit des sûretés",

"Procédure pénale",

"Droit du travail 2",

"Système juridique de l'Union Européenne",

"Droit administratif des biens",

"Droit fiscal",

"Droit de la fonction publique",

"Comptabilité",

"English Contract Law",

"Histoire de la propriété en Europe",

"Histoire du droit privé allemand",

"..."

]

};

// Pop-up logic

closePopupBtn.addEventListener('click', () => {

popup.classList.add('hidden');

mainContent.classList.remove('hidden');

});

// Authentication state listener

onAuthStateChanged(auth, async (user) => {

if (user) {

userId = user.uid;

console.log("User signed in with ID:", userId);

setupFirestoreListeners();

} else {

console.log("No user, signing in...");

try {

if (initialAuthToken) {

await signInWithCustomToken(auth, initialAuthToken);

console.log("Custom token sign-in successful.");

} else {

await signInAnonymously(auth);

console.log("Anonymous sign-in successful.");

}

} catch (error) {

console.error("Sign-in error:", error);

}

}

});

function setupFirestoreListeners() {

// Listener for course comments

const commentsCollectionPath = `artifacts/${appId}/public/data/comments`;

const commentsCollectionRef = collection(db, commentsCollectionPath);

onSnapshot(query(commentsCollectionRef), (snapshot) => {

allNotes = [];

snapshot.forEach((doc) => {

allNotes.push({ id: doc.id, ...doc.data() });

});

console.log("Notes loaded:", allNotes.length);

if (currentCourseTitle) {

displayCourseDetails(currentLevel, currentCourseTitle);

}

displayAdminDashboard();

}, (error) => {

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

});

// Listener for contact submissions

const contactCollectionPath = `artifacts/${appId}/public/data/contact\_submissions`;

const contactCollectionRef = collection(db, contactCollectionPath);

onSnapshot(query(contactCollectionRef), (snapshot) => {

allContactSubmissions = [];

snapshot.forEach((doc) => {

allContactSubmissions.push({ id: doc.id, ...doc.data() });

});

console.log("Contact submissions loaded:", allContactSubmissions.length);

displayAdminDashboard();

}, (error) => {

console.error("Error fetching contact submissions:", error);

});

}

// Populate year buttons with click-to-open dropdowns

function populateLevelButtons() {

levelSelector.innerHTML = '';

const levels = ['L1', 'L2', 'L3'];

levels.forEach(level => {

const levelContainer = document.createElement('div');

levelContainer.className = 'relative w-full';

const levelButton = document.createElement('button');

levelButton.className = 'level-button w-full bg-green-500 text-white font-semibold py-2 px-6 rounded-full hover:bg-green-600 transition-colors duration-300 shadow-lg';

levelButton.textContent = level;

levelButton.dataset.level = level;

const dropdownList = document.createElement('div');

dropdownList.className = 'course-list-dropdown absolute z-10 bg-white shadow-lg rounded-xl p-4 mt-2 w-full text-left max-h-60 overflow-y-auto hidden';

const courses = staticCourses[level] || [];

if (courses.length > 0) {

courses.forEach(courseTitle => {

const courseButton = document.createElement('button');

courseButton.className = 'block w-full text-left py-2 px-4 rounded-lg hover:bg-gray-100 transition-colors duration-200';

courseButton.textContent = courseTitle;

courseButton.addEventListener('click', () => {

currentLevel = level;

currentCourseTitle = courseTitle;

displayCourseDetails(level, courseTitle);

});

dropdownList.appendChild(courseButton);

});

} else {

const noCourseMessage = document.createElement('p');

noCourseMessage.className = 'text-sm text-gray-500 p-2';

noCourseMessage.textContent = 'Aucun cours disponible.';

dropdownList.appendChild(noCourseMessage);

}

levelButton.addEventListener('click', () => {

// Hide all other dropdowns

document.querySelectorAll('.course-list-dropdown').forEach(dd => {

if (dd !== dropdownList) {

dd.classList.add('hidden');

}

});

// Toggle the current dropdown's visibility

dropdownList.classList.toggle('hidden');

});

levelContainer.appendChild(levelButton);

levelContainer.appendChild(dropdownList);

levelSelector.appendChild(levelContainer);

});

}

function displayCourseDetails(level, title) {

mainTitle.classList.add('hidden');

levelSelectorSection.classList.add('hidden');

courseSection.classList.remove('hidden');

courseTitle.textContent = title;

const filteredComments = allNotes.filter(note => note.level === level && note.courseTitle === title);

const totalRating = filteredComments.reduce((sum, note) => sum + parseInt(note.rating, 10), 0);

const averageRating = filteredComments.length > 0 ? (totalRating / filteredComments.length).toFixed(1) : 0;

let avgRatingColorClass = 'text-green-500';

if (averageRating >= 0 && averageRating <= 4) {

avgRatingColorClass = 'text-red-500';

} else if (averageRating > 4 && averageRating <= 7) {

avgRatingColorClass = 'text-orange-500';

}

averageRatingSpan.textContent = averageRating;

averageRatingSpan.className = `font-bold ${avgRatingColorClass}`;

commentsList.innerHTML = '';

if (filteredComments.length === 0) {

commentsList.innerHTML = '<p class="text-center text-gray-500">Soyez le premier à noter ce cours !</p>';

} else {

filteredComments.sort((a, b) => b.createdAt - a.createdAt).forEach(comment => {

const commentElement = document.createElement('div');

let colorClass = 'text-green-500';

if (comment.rating >= 0 && comment.rating <= 4) {

colorClass = 'text-red-500';

} else if (comment.rating > 4 && comment.rating <= 7) {

colorClass = 'text-orange-500';

}

commentElement.className = 'bg-gray-100 rounded-xl p-4 shadow-sm';

commentElement.innerHTML = `

<div class="flex items-center justify-between mb-2">

<span class="font-semibold ${colorClass}">Note: ${comment.rating} / 10</span>

<span class="text-sm text-gray-500">Par: ${comment.userId.substring(0, 8)}...</span>

</div>

<p class="text-gray-700">${comment.comment}</p>

`;

commentsList.appendChild(commentElement);

});

}

}

// Add rating and comment to Firestore

ratingForm.addEventListener('submit', async (e) => {

e.preventDefault();

if (selectedRating === null) {

console.error("Veuillez sélectionner une note.");

return;

}

const comment = commentInput.value;

if (!userId) {

console.error("User not authenticated.");

return;

}

if (!currentCourseTitle || !currentLevel) {

console.error("No course selected.");

return;

}

try {

const commentsCollectionPath = `artifacts/${appId}/public/data/comments`;

await addDoc(collection(db, commentsCollectionPath), {

userId: userId,

level: currentLevel,

courseTitle: currentCourseTitle,

rating: selectedRating,

comment: comment,

createdAt: Date.now()

});

selectedRating = null;

commentInput.value = '';

updateRatingScale();

} catch (error) {

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

}

});

// Handle back button clicks

backFromCourseButton.addEventListener('click', () => {

courseSection.classList.add('hidden');

levelSelectorSection.classList.remove('hidden');

mainTitle.classList.remove('hidden');

});

// Contact form logic

contactLink.addEventListener('click', (e) => {

e.preventDefault();

levelSelectorSection.classList.add('hidden');

courseSection.classList.add('hidden');

adminSection.classList.add('hidden');

adminDashboardSection.classList.add('hidden');

contactFormSection.classList.remove('hidden');

contactSuccessSection.classList.add('hidden');

});

backFromContactButton.addEventListener('click', () => {

contactFormSection.classList.add('hidden');

levelSelectorSection.classList.remove('hidden');

mainTitle.classList.remove('hidden');

});

contactForm.addEventListener('input', () => {

const isFilled = contactTopicInput.value !== '' && contactMessageInput.value !== '' && contactEmailInput.value !== '';

if (isFilled) {

submitContactFormButton.disabled = false;

submitContactFormButton.classList.remove('bg-gray-400', 'cursor-not-allowed');

submitContactFormButton.classList.add('bg-green-500', 'hover:bg-green-600');

} else {

submitContactFormButton.disabled = true;

submitContactFormButton.classList.add('bg-gray-400', 'cursor-not-allowed');

submitContactFormButton.classList.remove('bg-green-500', 'hover:bg-green-600');

}

});

contactForm.addEventListener('submit', async (e) => {

e.preventDefault();

const topic = contactTopicInput.value;

const message = contactMessageInput.value;

const email = contactEmailInput.value;

try {

const contactCollectionPath = `artifacts/${appId}/public/data/contact\_submissions`;

await addDoc(collection(db, contactCollectionPath), {

topic,

message,

email,

createdAt: Date.now()

});

contactFormSection.classList.add('hidden');

contactSuccessSection.classList.remove('hidden');

contactForm.reset();

submitContactFormButton.disabled = true;

submitContactFormButton.classList.add('bg-gray-400', 'cursor-not-allowed');

submitContactFormButton.classList.remove('bg-green-500', 'hover:bg-green-600');

} catch (error) {

contactMessageStatus.textContent = "Erreur lors de l'envoi du message.";

contactMessageStatus.classList.remove('text-green-500');

contactMessageStatus.classList.add('text-red-500');

console.error("Error adding contact submission:", error);

}

});

document.getElementById('back-from-contact-success').addEventListener('click', () => {

contactSuccessSection.classList.add('hidden');

levelSelectorSection.classList.remove('hidden');

mainTitle.classList.remove('hidden');

});

// Admin section logic

adminLink.addEventListener('click', (e) => {

e.preventDefault();

levelSelectorSection.classList.add('hidden');

courseSection.classList.add('hidden');

contactFormSection.classList.add('hidden');

contactSuccessSection.classList.add('hidden');

adminDashboardSection.classList.add('hidden');

adminSection.classList.remove('hidden');

});

backFromAdminButton.addEventListener('click', () => {

adminSection.classList.add('hidden');

levelSelectorSection.classList.remove('hidden');

mainTitle.classList.remove('hidden');

});

// Admin login logic

adminForm.addEventListener('submit', (e) => {

e.preventDefault();

const password = adminPasswordInput.value;

const correctPassword = "NicolePD69";

if (password === correctPassword) {

adminSection.classList.add('hidden');

adminDashboardSection.classList.remove('hidden');

displayAdminDashboard();

} else {

adminMessageDiv.textContent = "Mot de passe incorrect.";

adminMessageDiv.classList.add('text-red-500');

adminMessageDiv.classList.remove('text-green-500');

adminMessageDiv.classList.remove('hidden');

}

});

// Admin Dashboard logic

function displayAdminDashboard() {

// Display comments tab by default

showCommentsTab.classList.add('bg-green-500', 'text-white');

showCommentsTab.classList.remove('bg-gray-200', 'text-gray-800');

showTroublemakersTab.classList.remove('bg-green-500', 'text-white');

showTroublemakersTab.classList.add('bg-gray-200', 'text-gray-800');

adminCommentsSection.classList.remove('hidden');

adminTroublemakersSection.classList.add('hidden');

displayCommentsInAdmin();

displayTroublemakersInAdmin();

}

function displayCommentsInAdmin() {

adminCommentsList.innerHTML = '';

if (allNotes.length === 0) {

adminCommentsList.innerHTML = '<p class="text-center text-gray-500">Aucun commentaire à afficher.</p>';

return;

}

const groupedComments = allNotes.reduce((acc, note) => {

const key = `${note.level} - ${note.courseTitle}`;

if (!acc[key]) {

acc[key] = {

level: note.level,

courseTitle: note.courseTitle,

comments: []

};

}

acc[key].comments.push(note);

return acc;

}, {});

for (const key in groupedComments) {

const courseData = groupedComments[key];

const courseGroup = document.createElement('div');

courseGroup.className = 'bg-gray-50 rounded-lg p-4 shadow-sm';

courseGroup.innerHTML = `

<h3 class="text-xl font-semibold text-gray-800 mb-4">${courseData.level} - ${courseData.courseTitle}</h3>

<div class="space-y-4"></div>

`;

const commentsContainer = courseGroup.querySelector('div');

courseData.comments.sort((a, b) => b.createdAt - a.createdAt).forEach(comment => {

const commentElement = document.createElement('div');

commentElement.className = 'bg-white rounded-xl p-4 shadow-md flex justify-between items-start';

let colorClass = 'text-green-500';

if (comment.rating >= 0 && comment.rating <= 4) {

colorClass = 'text-red-500';

} else if (comment.rating > 4 && comment.rating <= 7) {

colorClass = 'text-orange-500';

}

commentElement.innerHTML = `

<div>

<span class="font-semibold ${colorClass}">Note: ${comment.rating} / 10</span>

<p class="text-gray-700">${comment.comment}</p>

<span class="text-xs text-gray-500">ID utilisateur: ${comment.userId.substring(0, 8)}...</span>

</div>

<button class="delete-comment-btn text-red-500 hover:text-red-700 transition-colors duration-200" data-id="${comment.id}">

Supprimer

</button>

`;

commentsContainer.appendChild(commentElement);

});

adminCommentsList.appendChild(courseGroup);

}

}

function displayTroublemakersInAdmin() {

adminTroublemakersList.innerHTML = '';

if (allContactSubmissions.length === 0) {

adminTroublemakersList.innerHTML = '<p class="text-center text-gray-500">Aucun message de contact à afficher.</p>';

return;

}

allContactSubmissions.sort((a, b) => b.createdAt - a.createdAt).forEach(submission => {

const submissionElement = document.createElement('div');

submissionElement.className = 'bg-white rounded-xl p-4 shadow-md flex justify-between items-start';

submissionElement.innerHTML = `

<div>

<h4 class="font-bold text-gray-800">${submission.topic}</h4>

<p class="text-gray-700 mt-1">Message: ${submission.message}</p>

<p class="text-gray-700 mt-1">Email: ${submission.email}</p>

</div>

<button class="delete-submission-btn text-red-500 hover:text-red-700 transition-colors duration-200" data-id="${submission.id}">

Supprimer

</button>

`;

adminTroublemakersList.appendChild(submissionElement);

});

}

showCommentsTab.addEventListener('click', () => {

showCommentsTab.classList.add('bg-green-500', 'text-white');

showCommentsTab.classList.remove('bg-gray-200', 'text-gray-800');

showTroublemakersTab.classList.remove('bg-green-500', 'text-white');

showTroublemakersTab.classList.add('bg-gray-200', 'text-gray-800');

adminCommentsSection.classList.remove('hidden');

adminTroublemakersSection.classList.add('hidden');

});

showTroublemakersTab.addEventListener('click', () => {

showTroublemakersTab.classList.add('bg-green-500', 'text-white');

showTroublemakersTab.classList.remove('bg-gray-200', 'text-gray-800');

showCommentsTab.classList.remove('bg-green-500', 'text-white');

showCommentsTab.classList.add('bg-gray-200', 'text-gray-800');

adminTroublemakersSection.classList.remove('hidden');

adminCommentsSection.classList.add('hidden');

});

async function deleteComment(commentId) {

try {

const commentsCollectionPath = `artifacts/${appId}/public/data/comments`;

await deleteDoc(doc(db, commentsCollectionPath, commentId));

console.log("Document successfully deleted!");

} catch (error) {

console.error("Error removing document: ", error);

}

}

async function deleteSubmission(submissionId) {

try {

const contactCollectionPath = `artifacts/${appId}/public/data/contact\_submissions`;

await deleteDoc(doc(db, contactCollectionPath, submissionId));

console.log("Contact submission successfully deleted!");

} catch (error) {

console.error("Error removing contact submission: ", error);

}

}

adminCommentsList.addEventListener('click', (e) => {

const deleteButton = e.target.closest('.delete-comment-btn');

if (deleteButton) {

const commentId = deleteButton.dataset.id;

deleteComment(commentId);

}

});

adminTroublemakersList.addEventListener('click', (e) => {

const deleteButton = e.target.closest('.delete-submission-btn');

if (deleteButton) {

const submissionId = deleteButton.dataset.id;

deleteSubmission(submissionId);

}

});

backFromAdminDashboardButton.addEventListener('click', () => {

adminDashboardSection.classList.add('hidden');

levelSelectorSection.classList.remove('hidden');

mainTitle.classList.remove('hidden');

});

// Initialize level buttons

populateLevelButtons();

// --- Rating Scale Logic ---

function createRatingScale() {

for (let i = 1; i <= 10; i++) {

const point = document.createElement('div');

point.className = 'rating-point';

point.textContent = i;

point.dataset.value = i;

ratingScale.appendChild(point);

}

ratingScale.addEventListener('click', (e) => {

const target = e.target.closest('.rating-point');

if (target) {

selectedRating = parseInt(target.dataset.value, 10);

updateRatingScale();

}

});

}

function updateRatingScale() {

document.querySelectorAll('.rating-point').forEach(point => {

if (parseInt(point.dataset.value, 10) <= selectedRating) {

point.classList.add('selected');

} else {

point.classList.remove('selected');

}

});

}

createRatingScale();

</script>

</body>

</html>