<!DOCTYPE html>

<html lang="fr">

<head>

 <meta charset="UTF-8">

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

 <title>Agencer mes réunions.fr</title>

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

 <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.3/css/all.min.css"></link>

 <link href="https://fonts.googleapis.com/css2?family=Roboto:wght@400;500;700&display=swap" rel="stylesheet">

 <style>

 body {

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

 }

 </style>

</head>

<body class="bg-gray-100">

 <div class="container mx-auto p-4">

 <header class="flex justify-between items-center py-4">

 <h1 class="text-3xl font-bold text-gray-800">Application de Prise de Notes</h1>

 <button class="bg-blue-500 text-white px-4 py-2 rounded hover:bg-blue-600" id="addNoteButton">

 <i class="fas fa-plus"></i> Ajouter une Note

 </button>

 <button class="bg-green-500 text-white px-4 py-2 rounded hover:bg-green-600" id="exportButton">

 <i class="fas fa-download"></i> Exporter les Notes

 </button>

 <button class="bg-teal-500 text-white px-4 py-2 rounded hover:bg-teal-600" id="importButton">

 <i class="fas fa-upload"></i> Importer .monrdv

 </button>

 </header>

 <!-- Zone de recherche de matière -->

 <div class="mb-4">

 <label for="searchMatiere" class="text-gray-700">Rechercher par matière :</label>

 <input type="text" id="searchMatiere" class="bg-white border-gray-300 border rounded p-2 w-full" placeholder="Tapez une matière...">

 </div>

 <!-- Sélecteurs de matière et couleur -->

 <div class="flex justify-between mb-4">

 <div>

 <label for="matiere" class="text-gray-700">Choisir le type :</label>

 <select id="matiere" class="bg-white border-gray-300 border rounded p-2">

 <option value="Réunion de département Feed and Food">Réunion de département Feed and Food</option>

 <option value="Conseil d’Administration de l’Association Française de Zootechnie">Conseil d’Administration de l’Association Française de Zootechnie</option>

 <option value="Académie de la Viande">Académie de la Viande</option>

 <option value="Comité Scientifique de l’ADIV">Comité Scientifique de l’ADIV</option>

 <option value="Conseil Scientifique du Réseau Territorial RetD ovin lait 64">Conseil Scientifique du Réseau Territorial RetD ovin lait 64</option>

<option value="Comité de Certification Qualisud">Comité de Certification Qualisud</option>

<option value="Comité Stratégique du Groupement d’Intérêt Scientifique (GIS) Avenir Élevage">Comité Stratégique du Groupement d’Intérêt Scientifique (GIS) Avenir Élevage</option>

<option value="Directoire Opérationnel du Groupement d’Intérêt Scientifique (GIS) Avenir Élevage">Directoire Opérationnel du Groupement d’Intérêt Scientifique (GIS) Avenir Élevage</option>

<option value="Commission Thématique Inter-filières Ressources Génétiques de FranceAgriMer">Commission Thématique Inter-filières Ressources Génétiques de FranceAgriMer</option>

<option value="Conseil de Département Sciences et Environnement">Conseil de Département Sciences et Environnement</option>

<option value="Conseil de Centre INRAE Nouvelle-Aquitaine">Conseil de Centre INRAE Nouvelle-Aquitaine</option>

<option value="Conseil Scientifique d’Agreenium">Conseil Scientifique d’Agreenium</option>

<option value="Conseil de l’École Doctorale">Conseil de l’École Doctorale</option>

<option value="Commission Recherche de l’Institut des Sciences de la Vigne et du Vin">Commission Recherche de l’Institut des Sciences de la Vigne et du Vin</option>

<option value="Commission Recherche">Commission Recherche</option>

<option value="Conseils Scientifiques">Conseils Scientifiques</option>

<option value="Processus Liaison Formation Recherches">Processus Liaison Formation Recherches</option>

<option value="Directoire">Directoire</option>

<option value="Comité de Direction">Comité de Direction</option>

<option value="Processus Stratégie et Gouvernance">Processus Stratégie et Gouvernance</option>

<option value="Processus Mise En Œuvre du ruban pédagogique">Processus Mise En Œuvre du ruban pédagogique</option>

<option value="Réforme du cursus">Réforme du cursus</option>

<option value="CA">CA</option>

<option value="CS">CS</option>

<option value="Réunion Conférence des Grandes Ecole">Réunion Conférence des Grandes Ecole</option>

<option value="Réunion DGER">Réunion DGER</option>

 </select>

 </div>

 <div>

 <label for="backgroundColor" class="text-gray-700">Choisir la couleur de fond :</label>

 <input type="color" id="backgroundColor" value="#f0f0f0" class="p-2 border rounded">

 </div>

 </div>

 <!-- Zone d'affichage des notes -->

 <main id="notesContainer" class="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-3 gap-4"></main>

 </div>

 <!-- Modal pour ajouter ou modifier une note -->

 <div id="noteModal" class="fixed inset-0 flex justify-center items-center bg-gray-800 bg-opacity-50 hidden">

 <div class="bg-white p-6 rounded shadow-xl w-1/2">

 <h2 class="text-xl font-semibold text-gray-800 mb-4" id="modalTitle">Ajouter une Réunion</h2>

 <textarea id="noteContent" class="w-full p-4 border rounded mb-4" placeholder="Racontez nous votre longue réunion..."></textarea>

 <div class="flex justify-between">

 <button id="saveNoteButton" class="bg-blue-500 text-white px-4 py-2 rounded hover:bg-blue-600">

 <i class="fas fa-save"></i> Sauvegarder

 </button>

 <button id="closeModalButton" class="bg-gray-500 text-white px-4 py-2 rounded hover:bg-gray-600">

 <i class="fas fa-times"></i> Fermer

 </button>

 </div>

 </div>

 </div>

 <script src="https://cdnjs.cloudflare.com/ajax/libs/jszip/3.10.1/jszip.min.js"></script>

 <script src="https://cdnjs.cloudflare.com/ajax/libs/FileSaver.js/2.0.5/FileSaver.min.js"></script>

 <script>

 let notes = JSON.parse(localStorage.getItem('notes')) || [];

 // Fonction pour afficher les notes

 function renderNotes(filter = '') {

 const notesContainer = document.getElementById('notesContainer');

 notesContainer.innerHTML = '';

 const filteredNotes = notes.filter(note => note.matiere.toLowerCase().includes(filter.toLowerCase()));

 filteredNotes.forEach((note, index) => {

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

 noteElement.className = 'bg-white p-4 rounded shadow';

 noteElement.innerHTML = `

 <h2 class="text-xl font-semibold text-gray-800">${note.matiere} - ${note.title}</h2>

 <p class="text-gray-600 mt-2">${note.content}</p>

 <div class="flex justify-between items-center mt-4">

 <button class="text-blue-500 hover:text-blue-600" onclick="editNote(${index})">

 <i class="fas fa-edit"></i> Modifier

 </button>

 <button class="text-red-500 hover:text-red-600" onclick="deleteNote(${index})">

 <i class="fas fa-trash"></i> Supprimer

 </button>

 </div>

 <div class="mt-4 text-sm text-gray-500">

 <p><strong>Créée le :</strong> ${note.createdAt}</p>

 <p><strong>Dernière modification :</strong> ${note.modifiedAt}</p>

 </div>

 `;

 notesContainer.appendChild(noteElement);

 });

 }

 // Fonction pour ajouter ou modifier une note

 function openNoteModal(index = -1) {

 const modal = document.getElementById('noteModal');

 const noteContent = document.getElementById('noteContent');

 const modalTitle = document.getElementById('modalTitle');

 const saveNoteButton = document.getElementById('saveNoteButton');

 if (index !== -1) {

 modalTitle.textContent = 'Modifier la Note';

 noteContent.value = notes[index].content;

 saveNoteButton.onclick = () => saveNote(index);

 } else {

 modalTitle.textContent = 'Ajouter une Note';

 noteContent.value = '';

 saveNoteButton.onclick = () => saveNote();

 }

 modal.classList.remove('hidden');

 }

 // Fonction pour sauvegarder la note

 function saveNote(index = -1) {

 const noteContent = document.getElementById('noteContent').value;

 const matiere = document.getElementById('matiere').value;

 if (noteContent.trim() === '') return;

 const currentDate = new Date().toLocaleString();

 if (index === -1) {

 notes.push({

 title: `Note ${notes.length + 1}`,

 content: noteContent,

 matiere: matiere,

 createdAt: currentDate,

 modifiedAt: currentDate

 });

 } else {

 notes[index].content = noteContent;

 notes[index].modifiedAt = currentDate;

 }

 localStorage.setItem('notes', JSON.stringify(notes));

 renderNotes();

 closeModal();

 }

 // Fonction pour fermer le modal

 function closeModal() {

 document.getElementById('noteModal').classList.add('hidden');

 }

 // Fonction pour supprimer une note

 function deleteNote(index) {

 notes.splice(index, 1);

 localStorage.setItem('notes', JSON.stringify(notes));

 renderNotes();

 }

 // Fonction pour éditer une note

 function editNote(index) {

 openNoteModal(index);

 }

 // Fonction pour exporter les notes en fonction du format choisi

 document.getElementById('exportButton').onclick = () => {

 const exportChoice = prompt("Choisissez un format d'export :\n1 - .monrdv\n2 - Word (.docx)\n3 - Texte (.txt)\n4 - PDF");

 switch(exportChoice) {

 case '1':

 exportMonrdv();

 break;

 case '2':

 exportWord();

 break;

 case '3':

 exportText();

 break;

 case '4':

 exportPDF();

 break;

 default:

 alert('Choix invalide');

 }

 };

 // Exporter en .monrdv (JSON)

 function exportMonrdv() {

 const blob = new Blob([JSON.stringify(notes, null, 2)], { type: 'application/json' });

 const link = document.createElement('a');

 link.href = URL.createObjectURL(blob);

 link.download = 'mon\_debrif.monrdv';

 link.click();

 }

 // Exporter en Word (.docx)

 function exportWord() {

 const doc = new Document();

 notes.forEach(note => {

 doc.addSection({

 children: [

 new Paragraph({

 children: [new TextRun(note.title)],

 }),

 new Paragraph({

 children: [new TextRun(note.content)],

 }),

 ],

 });

 });

 Packer.toBlob(doc).then(blob => {

 saveAs(blob, "mon\_debrif.docx");

 });

 }

 // Exporter en texte (.txt)

 function exportText() {

 const text = notes.map(note => `${note.title}\n${note.content}\n\n`).join('');

 const blob = new Blob([text], { type: 'text/plain' });

 const link = document.createElement('a');

 link.href = URL.createObjectURL(blob);

 link.download = 'mon\_debrif.txt';

 link.click();

 }

 // Exporter en PDF

 function exportPDF() {

 const doc = new jsPDF();

 notes.forEach((note, index) => {

 doc.text(`${note.title}`, 10, 10 + (index \* 20));

 doc.text(`${note.content}`, 10, 20 + (index \* 20));

 doc.addPage();

 });

 doc.save('mon\_debrif.pdf');

 }

 // Fonction d'importation d'un fichier .monrdv

 document.getElementById('importButton').onclick = () => {

 const input = document.createElement('input');

 input.type = 'file';

 input.accept = '.monrdv';

 input.onchange = e => {

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

 const reader = new FileReader();

 reader.onload = function(event) {

 const importedNotes = JSON.parse(event.target.result);

 notes = importedNotes;

 localStorage.setItem('notes', JSON.stringify(notes));

 renderNotes();

 };

 reader.readAsText(file);

 };

 input.click();

 };

 // Événements de la page

 document.getElementById('addNoteButton').onclick = () => openNoteModal();

 document.getElementById('closeModalButton').onclick = closeModal;

 // Changer la couleur de fond

 document.getElementById('backgroundColor').oninput = (e) => {

 document.body.style.backgroundColor = e.target.value;

 };

 // Sauvegarder la note avec Maj + Entrée

 document.getElementById('noteContent').addEventListener('keydown', function (event) {

 if (event.key === 'Enter' && event.shiftKey) {

 saveNote();

 }

 });

 // Filtrer les notes par matière

 document.getElementById('searchMatiere').addEventListener('input', function(event) {

 const searchTerm = event.target.value;

 renderNotes(searchTerm);

 });

 // Initialiser l'affichage des notes

 renderNotes();

 </script>

</body>

</html>