<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>EduPlan Nigeria - Teacher Lesson Plan System</title>

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

<style>

.fade-in { animation: fadeIn 0.3s ease-in; }

@keyframes fadeIn { from { opacity: 0; transform: translateY(10px); } to { opacity: 1; transform: translateY(0); } }

.slide-in { animation: slideIn 0.3s ease-out; }

@keyframes slideIn { from { transform: translateX(-20px); opacity: 0; } to { transform: translateX(0); opacity: 1; } }

.pulse { animation: pulse 2s infinite; }

@keyframes pulse { 0%, 100% { opacity: 1; } 50% { opacity: 0.5; } }

.notification {

position: fixed;

top: 20px;

right: 20px;

z-index: 1000;

transform: translateX(400px);

transition: transform 0.3s ease-in-out;

}

.notification.show { transform: translateX(0); }

.loading-spinner {

border: 3px solid #f3f3f3;

border-top: 3px solid #3498db;

border-radius: 50%;

width: 20px;

height: 20px;

animation: spin 1s linear infinite;

}

@keyframes spin { 0% { transform: rotate(0deg); } 100% { transform: rotate(360deg); } }

.comment-bubble {

position: relative;

background: #fef3c7;

border: 1px solid #f59e0b;

border-radius: 8px;

padding: 12px;

margin: 8px 0;

}

.comment-bubble::before {

content: '';

position: absolute;

top: -8px;

left: 20px;

width: 0;

height: 0;

border-left: 8px solid transparent;

border-right: 8px solid transparent;

border-bottom: 8px solid #f59e0b;

}

.correction-highlight {

background-color: #fef2f2;

border-left: 4px solid #ef4444;

padding: 8px;

margin: 4px 0;

}

</style>

</head>

<body class="bg-gradient-to-br from-green-50 to-blue-50 min-h-screen">

<!-- Loading Screen -->

<div id="loadingScreen" class="fixed inset-0 bg-white z-50 flex items-center justify-center">

<div class="text-center">

<div class="w-16 h-16 bg-green-600 rounded-lg flex items-center justify-center mb-4 mx-auto">

<span class="text-white font-bold text-2xl">📚</span>

</div>

<h2 class="text-2xl font-bold text-gray-900 mb-2">EduPlan Nigeria</h2>

<div class="loading-spinner mx-auto"></div>

<p class="text-gray-600 mt-2">Loading lesson plan system...</p>

</div>

</div>

<!-- Authentication Screen -->

<div id="authScreen" class="fixed inset-0 bg-gradient-to-br from-green-600 to-blue-600 z-40 hidden">

<div class="flex items-center justify-center min-h-screen p-4">

<div class="bg-white rounded-2xl shadow-2xl max-w-md w-full p-8">

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

<div class="w-16 h-16 bg-green-600 rounded-lg flex items-center justify-center mb-4 mx-auto">

<span class="text-white font-bold text-2xl">📚</span>

</div>

<h1 class="text-2xl font-bold text-gray-900">EduPlan Nigeria</h1>

<p class="text-gray-600">Professional Lesson Plan Management</p>

</div>

<!-- Login Form -->

<div id="loginForm">

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

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">Email</label>

<input type="email" id="loginEmail" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500" placeholder="teacher@school.edu.ng">

</div>

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">Password</label>

<input type="password" id="loginPassword" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500" placeholder="Enter your password">

</div>

<button type="submit" class="w-full bg-green-600 hover:bg-green-700 text-white py-2 px-4 rounded-lg font-medium transition-colors flex items-center justify-center">

<span id="loginButtonText">Sign In</span>

<div id="loginSpinner" class="loading-spinner ml-2 hidden"></div>

</button>

</form>

<div class="mt-4 text-center">

<button onclick="showRegisterForm()" class="text-green-600 hover:text-green-700 text-sm">

Don't have an account? Register here

</button>

</div>

</div>

<!-- Register Form -->

<div id="registerForm" class="hidden">

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

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">Full Name</label>

<input type="text" id="registerName" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500" placeholder="John Doe">

</div>

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">Email</label>

<input type="email" id="registerEmail" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500" placeholder="teacher@school.edu.ng">

</div>

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">School Name</label>

<input type="text" id="registerSchool" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500" placeholder="Government Secondary School Lagos">

</div>

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">Role</label>

<select id="registerRole" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500">

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

<option value="teacher">Teacher</option>

<option value="admin">School Administrator</option>

<option value="principal">Principal</option>

</select>

</div>

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">Password</label>

<input type="password" id="registerPassword" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500" placeholder="Create a strong password">

</div>

<button type="submit" class="w-full bg-green-600 hover:bg-green-700 text-white py-2 px-4 rounded-lg font-medium transition-colors flex items-center justify-center">

<span id="registerButtonText">Create Account</span>

<div id="registerSpinner" class="loading-spinner ml-2 hidden"></div>

</button>

</form>

<div class="mt-4 text-center">

<button onclick="showLoginForm()" class="text-green-600 hover:text-green-700 text-sm">

Already have an account? Sign in

</button>

</div>

</div>

</div>

</div>

</div>

<!-- Notification System -->

<div id="notificationContainer"></div>

<!-- Main Application -->

<div id="mainApp" class="hidden">

<!-- Header -->

<header class="bg-white shadow-lg border-b-4 border-green-600">

<div class="max-w-7xl mx-auto px-4 sm:px-6 lg:px-8">

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

<div class="flex items-center space-x-3">

<div class="w-10 h-10 bg-green-600 rounded-lg flex items-center justify-center">

<span class="text-white font-bold text-lg">📚</span>

</div>

<div>

<h1 class="text-2xl font-bold text-gray-900">EduPlan Nigeria</h1>

<p class="text-sm text-gray-600">Professional Lesson Plan Management System</p>

</div>

</div>

<div class="flex items-center space-x-4">

<!-- User Info -->

<div class="text-right">

<p class="text-sm font-medium text-gray-900" id="currentUserName">Loading...</p>

<p class="text-xs text-gray-600" id="currentUserRole">Loading...</p>

</div>

<!-- Logout -->

<button onclick="logout()" class="bg-red-600 hover:bg-red-700 text-white px-4 py-2 rounded-lg text-sm font-medium transition-colors">

Logout

</button>

</div>

</div>

</div>

</header>

<!-- Navigation -->

<nav class="bg-white shadow-sm border-b">

<div class="max-w-7xl mx-auto px-4 sm:px-6 lg:px-8">

<div class="flex space-x-8">

<button onclick="showSection('dashboard')" class="nav-btn py-4 px-2 border-b-2 border-green-500 text-green-600 font-medium text-sm">

Dashboard

</button>

<button onclick="showSection('lesson-plans')" class="nav-btn py-4 px-2 border-b-2 border-transparent text-gray-500 hover:text-gray-700 font-medium text-sm">

My Lesson Plans

</button>

<button onclick="showSection('assessments')" class="nav-btn py-4 px-2 border-b-2 border-transparent text-gray-500 hover:text-gray-700 font-medium text-sm">

Assessments

</button>

<button id="adminNavBtn" onclick="showSection('admin-review')" class="nav-btn py-4 px-2 border-b-2 border-transparent text-gray-500 hover:text-gray-700 font-medium text-sm hidden">

Review Submissions

</button>

</div>

</div>

</nav>

<!-- Main Content -->

<main class="max-w-7xl mx-auto px-4 sm:px-6 lg:px-8 py-8">

<!-- Dashboard Section -->

<div id="dashboard" class="section fade-in">

<div class="mb-8">

<h2 class="text-3xl font-bold text-gray-900 mb-2">Dashboard</h2>

<p class="text-gray-600">Overview of your lesson plans and assessments</p>

</div>

<!-- Stats Cards -->

<div class="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-4 gap-6 mb-8">

<div class="bg-white rounded-xl shadow-md p-6 border-l-4 border-blue-500">

<div class="flex items-center">

<div class="flex-shrink-0">

<div class="w-8 h-8 bg-blue-100 rounded-lg flex items-center justify-center">

<span class="text-blue-600 text-lg">📝</span>

</div>

</div>

<div class="ml-4">

<p class="text-sm font-medium text-gray-600">Total Lesson Plans</p>

<p class="text-2xl font-bold text-gray-900" id="totalLessonPlans">0</p>

</div>

</div>

</div>

<div class="bg-white rounded-xl shadow-md p-6 border-l-4 border-green-500">

<div class="flex items-center">

<div class="flex-shrink-0">

<div class="w-8 h-8 bg-green-100 rounded-lg flex items-center justify-center">

<span class="text-green-600 text-lg">✅</span>

</div>

</div>

<div class="ml-4">

<p class="text-sm font-medium text-gray-600">Approved Plans</p>

<p class="text-2xl font-bold text-gray-900" id="approvedPlans">0</p>

</div>

</div>

</div>

<div class="bg-white rounded-xl shadow-md p-6 border-l-4 border-yellow-500">

<div class="flex items-center">

<div class="flex-shrink-0">

<div class="w-8 h-8 bg-yellow-100 rounded-lg flex items-center justify-center">

<span class="text-yellow-600 text-lg">⏳</span>

</div>

</div>

<div class="ml-4">

<p class="text-sm font-medium text-gray-600">Pending Review</p>

<p class="text-2xl font-bold text-gray-900" id="pendingPlans">0</p>

</div>

</div>

</div>

<div class="bg-white rounded-xl shadow-md p-6 border-l-4 border-purple-500">

<div class="flex items-center">

<div class="flex-shrink-0">

<div class="w-8 h-8 bg-purple-100 rounded-lg flex items-center justify-center">

<span class="text-purple-600 text-lg">📊</span>

</div>

</div>

<div class="ml-4">

<p class="text-sm font-medium text-gray-600">Assessments</p>

<p class="text-2xl font-bold text-gray-900" id="totalAssessments">0</p>

</div>

</div>

</div>

</div>

<!-- Recent Activity -->

<div class="bg-white rounded-xl shadow-md p-6">

<h3 class="text-lg font-semibold text-gray-900 mb-4">Recent Activity</h3>

<div class="space-y-4" id="recentActivity">

<div class="text-center text-gray-500 py-8">

<p>No recent activity</p>

</div>

</div>

</div>

</div>

<!-- Lesson Plans Section -->

<div id="lesson-plans" class="section hidden">

<div class="flex justify-between items-center mb-8">

<div>

<h2 class="text-3xl font-bold text-gray-900 mb-2">My Lesson Plans</h2>

<p class="text-gray-600">Create and manage your weekly lesson plans</p>

</div>

<button onclick="showCreateLessonPlan()" class="bg-green-600 hover:bg-green-700 text-white px-6 py-3 rounded-lg font-medium flex items-center space-x-2 transition-colors">

<span>➕</span>

<span>Create New Lesson Plan</span>

</button>

</div>

<!-- Lesson Plans List -->

<div class="grid gap-6" id="lessonPlansList">

<div class="text-center text-gray-500 py-8">

<div class="loading-spinner mx-auto mb-2"></div>

<p>Loading your lesson plans...</p>

</div>

</div>

</div>

<!-- Assessments Section -->

<div id="assessments" class="section hidden">

<div class="flex justify-between items-center mb-8">

<div>

<h2 class="text-3xl font-bold text-gray-900 mb-2">Assessments</h2>

<p class="text-gray-600">Create higher-order thinking assessments</p>

</div>

<button onclick="showCreateAssessment()" class="bg-blue-600 hover:bg-blue-700 text-white px-6 py-3 rounded-lg font-medium flex items-center space-x-2 transition-colors">

<span>📊</span>

<span>Create Assessment</span>

</button>

</div>

<!-- Assessments List -->

<div class="grid gap-6" id="assessmentsList">

<div class="text-center text-gray-500 py-8">

<div class="loading-spinner mx-auto mb-2"></div>

<p>Loading your assessments...</p>

</div>

</div>

</div>

<!-- Admin Review Section -->

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

<div class="mb-8">

<h2 class="text-3xl font-bold text-gray-900 mb-2">Review Submissions</h2>

<p class="text-gray-600">Review and provide feedback on teacher submissions</p>

</div>

<!-- Admin Stats -->

<div class="grid grid-cols-1 md:grid-cols-3 gap-6 mb-8">

<div class="bg-white rounded-xl shadow-md p-6 border-l-4 border-orange-500">

<div class="flex items-center">

<div class="flex-shrink-0">

<div class="w-8 h-8 bg-orange-100 rounded-lg flex items-center justify-center">

<span class="text-orange-600 text-lg">⏰</span>

</div>

</div>

<div class="ml-4">

<p class="text-sm font-medium text-gray-600">Pending Reviews</p>

<p class="text-2xl font-bold text-gray-900" id="adminPendingPlans">0</p>

</div>

</div>

</div>

<div class="bg-white rounded-xl shadow-md p-6 border-l-4 border-green-500">

<div class="flex items-center">

<div class="flex-shrink-0">

<div class="w-8 h-8 bg-green-100 rounded-lg flex items-center justify-center">

<span class="text-green-600 text-lg">✅</span>

</div>

</div>

<div class="ml-4">

<p class="text-sm font-medium text-gray-600">Approved Today</p>

<p class="text-2xl font-bold text-gray-900" id="approvedToday">0</p>

</div>

</div>

</div>

<div class="bg-white rounded-xl shadow-md p-6 border-l-4 border-blue-500">

<div class="flex items-center">

<div class="flex-shrink-0">

<div class="w-8 h-8 bg-blue-100 rounded-lg flex items-center justify-center">

<span class="text-blue-600 text-lg">👥</span>

</div>

</div>

<div class="ml-4">

<p class="text-sm font-medium text-gray-600">Total Teachers</p>

<p class="text-2xl font-bold text-gray-900" id="totalTeachers">0</p>

</div>

</div>

</div>

</div>

<!-- Pending Submissions -->

<div class="bg-white rounded-xl shadow-md p-6">

<h3 class="text-lg font-semibold text-gray-900 mb-6">Pending Submissions</h3>

<div class="space-y-4" id="adminSubmissionsList">

<div class="text-center text-gray-500 py-8">

<div class="loading-spinner mx-auto mb-2"></div>

<p>Loading pending submissions...</p>

</div>

</div>

</div>

</div>

</main>

<!-- Create Lesson Plan Modal -->

<div id="lessonPlanModal" class="fixed inset-0 bg-black bg-opacity-50 hidden z-50">

<div class="flex items-center justify-center min-h-screen p-4">

<div class="bg-white rounded-xl shadow-xl max-w-6xl w-full max-h-screen overflow-y-auto">

<div class="p-6 border-b">

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

<h3 class="text-xl font-bold text-gray-900">Create Weekly Lesson Plan</h3>

<button onclick="closeLessonPlanModal()" class="text-gray-400 hover:text-gray-600 text-2xl">×</button>

</div>

</div>

<form id="lessonPlanForm" class="p-6 space-y-6">

<!-- Basic Information -->

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

<h4 class="text-lg font-semibold text-gray-900 mb-4">Basic Information</h4>

<div class="grid grid-cols-1 md:grid-cols-3 gap-4">

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">Subject</label>

<select id="subject" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500">

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

<option value="Mathematics">Mathematics</option>

<option value="English Language">English Language</option>

<option value="Basic Science">Basic Science</option>

<option value="Social Studies">Social Studies</option>

<option value="Civic Education">Civic Education</option>

<option value="Computer Studies">Computer Studies</option>

<option value="Agricultural Science">Agricultural Science</option>

<option value="Home Economics">Home Economics</option>

<option value="French">French</option>

<option value="Yoruba">Yoruba</option>

<option value="Igbo">Igbo</option>

<option value="Hausa">Hausa</option>

</select>

</div>

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">Class</label>

<select id="class" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500">

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

<option value="JSS 1">JSS 1</option>

<option value="JSS 2">JSS 2</option>

<option value="JSS 3">JSS 3</option>

<option value="SS 1">SS 1</option>

<option value="SS 2">SS 2</option>

<option value="SS 3">SS 3</option>

</select>

</div>

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">Class Size</label>

<input type="number" id="classSize" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500" placeholder="e.g., 35" min="1">

</div>

</div>

<div class="grid grid-cols-1 md:grid-cols-2 gap-4 mt-4">

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">Duration</label>

<input type="text" id="duration" value="45 minutes" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500">

</div>

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">Week</label>

<input type="text" id="week" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500" placeholder="e.g., Week 5">

</div>

</div>

</div>

<!-- Lesson Hook -->

<div class="bg-blue-50 p-4 rounded-lg">

<h4 class="text-lg font-semibold text-gray-900 mb-4">Lesson Hook</h4>

<div class="mb-4">

<label class="block text-sm font-medium text-gray-700 mb-2">Hook Type</label>

<select id="hookType" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500">

<option value="">Select Hook Type</option>

<option value="storytelling">Story Telling</option>

<option value="drama">Drama/Role Play</option>

<option value="field\_trip">Going Out/Field Trip</option>

<option value="illustration">Using Illustration</option>

<option value="demonstration">Demonstration</option>

<option value="question">Thought-Provoking Question</option>

<option value="video">Video/Multimedia</option>

<option value="game">Educational Game</option>

</select>

</div>

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">Hook Description</label>

<textarea id="lessonHook" rows="3" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500" placeholder="Describe your engaging opening activity in detail..."></textarea>

</div>

</div>

<!-- Topic and Objectives -->

<div class="bg-green-50 p-4 rounded-lg">

<h4 class="text-lg font-semibold text-gray-900 mb-4">Topic & Learning Objectives</h4>

<div class="mb-4">

<label class="block text-sm font-medium text-gray-700 mb-2">Topic/Sub-topic</label>

<input type="text" id="topic" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500" placeholder="Enter the main topic or sub-topic">

</div>

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">Learning Objectives</label>

<textarea id="objectives" rows="4" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500" placeholder="By the end of this lesson, students should be able to...&#10;1. &#10;2. &#10;3."></textarea>

</div>

</div>

<!-- Entry Behaviour -->

<div class="bg-yellow-50 p-4 rounded-lg">

<h4 class="text-lg font-semibold text-gray-900 mb-4">Entry Behaviour/Previous Knowledge</h4>

<textarea id="entryBehaviour" rows="3" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500" placeholder="What should students already know before this lesson?"></textarea>

</div>

<!-- Direct Instruction -->

<div class="bg-purple-50 p-4 rounded-lg">

<h4 class="text-lg font-semibold text-gray-900 mb-4">Direct Instruction (Main Teaching)</h4>

<textarea id="directInstruction" rows="6" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500" placeholder="Detailed explanation of your main teaching content. Include step-by-step instructions, examples, and key concepts..."></textarea>

</div>

<!-- Differentiation -->

<div class="bg-indigo-50 p-4 rounded-lg">

<h4 class="text-lg font-semibold text-gray-900 mb-4">Differentiation</h4>

<div class="grid grid-cols-1 md:grid-cols-3 gap-4">

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">For Advanced Learners</label>

<textarea id="advancedLearners" rows="3" class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500" placeholder="Additional challenges or extensions..."></textarea>

</div>

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">For Average Learners</label>

<textarea id="averageLearners" rows="3" class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500" placeholder="Standard activities and support..."></textarea>

</div>

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">For Struggling Learners</label>

<textarea id="strugglingLearners" rows="3" class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500" placeholder="Additional support and scaffolding..."></textarea>

</div>

</div>

</div>

<!-- Teaching Methods and Resources -->

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

<div class="bg-red-50 p-4 rounded-lg">

<h4 class="text-lg font-semibold text-gray-900 mb-4">Teaching Methods</h4>

<div class="space-y-2">

<label class="flex items-center">

<input type="checkbox" name="teachingMethods" value="lecture" class="mr-2">

<span class="text-sm">Lecture Method</span>

</label>

<label class="flex items-center">

<input type="checkbox" name="teachingMethods" value="discussion" class="mr-2">

<span class="text-sm">Discussion Method</span>

</label>

<label class="flex items-center">

<input type="checkbox" name="teachingMethods" value="demonstration" class="mr-2">

<span class="text-sm">Demonstration</span>

</label>

<label class="flex items-center">

<input type="checkbox" name="teachingMethods" value="inquiry" class="mr-2">

<span class="text-sm">Inquiry-Based Learning</span>

</label>

<label class="flex items-center">

<input type="checkbox" name="teachingMethods" value="collaborative" class="mr-2">

<span class="text-sm">Collaborative Learning</span>

</label>

<label class="flex items-center">

<input type="checkbox" name="teachingMethods" value="project" class="mr-2">

<span class="text-sm">Project-Based Learning</span>

</label>

</div>

</div>

<div class="bg-teal-50 p-4 rounded-lg">

<h4 class="text-lg font-semibold text-gray-900 mb-4">Learning Resources & Materials</h4>

<textarea id="learningResources" rows="6" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500" placeholder="List all materials needed:&#10;- Textbooks&#10;- Visual aids&#10;- Technology tools&#10;- Manipulatives&#10;- etc."></textarea>

</div>

</div>

<!-- Summary and Assessment -->

<div class="bg-orange-50 p-4 rounded-lg">

<h4 class="text-lg font-semibold text-gray-900 mb-4">Summary & Assessment</h4>

<div class="mb-4">

<label class="block text-sm font-medium text-gray-700 mb-2">Lesson Summary</label>

<textarea id="lessonSummary" rows="3" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500" placeholder="Summarize the key points covered in this lesson..."></textarea>

</div>

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">Exit Ticket Questions (to assess class level of assimilation)</label>

<textarea id="exitTickets" rows="4" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500" placeholder="Write 3-5 quick assessment questions:&#10;1. &#10;2. &#10;3."></textarea>

</div>

</div>

<!-- Real-life Application -->

<div class="bg-pink-50 p-4 rounded-lg">

<h4 class="text-lg font-semibold text-gray-900 mb-4">Real-life Application & Relatable Activities</h4>

<textarea id="realLifeApplication" rows="4" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-green-500 focus:border-green-500" placeholder="How does this lesson connect to students' daily lives? What relatable activities will you include?"></textarea>

</div>

<div class="flex justify-end space-x-4 pt-6 border-t">

<button type="button" onclick="closeLessonPlanModal()" class="px-6 py-2 border border-gray-300 rounded-lg text-gray-700 hover:bg-gray-50 transition-colors">

Cancel

</button>

<button type="submit" class="px-6 py-2 bg-green-600 hover:bg-green-700 text-white rounded-lg transition-colors flex items-center">

<span id="saveLessonPlanText">Submit for Review</span>

<div id="saveLessonPlanSpinner" class="loading-spinner ml-2 hidden"></div>

</button>

</div>

</form>

</div>

</div>

</div>

<!-- Create Assessment Modal -->

<div id="assessmentModal" class="fixed inset-0 bg-black bg-opacity-50 hidden z-50">

<div class="flex items-center justify-center min-h-screen p-4">

<div class="bg-white rounded-xl shadow-xl max-w-4xl w-full max-h-screen overflow-y-auto">

<div class="p-6 border-b">

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

<h3 class="text-xl font-bold text-gray-900">Create Higher-Order Thinking Assessment</h3>

<button onclick="closeAssessmentModal()" class="text-gray-400 hover:text-gray-600 text-2xl">×</button>

</div>

</div>

<form id="assessmentForm" class="p-6 space-y-6">

<!-- Basic Information -->

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

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">Assessment Title</label>

<input type="text" id="assessmentTitle" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-500 focus:border-blue-500" placeholder="e.g., Critical Thinking in Mathematics">

</div>

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">Subject</label>

<select id="assessmentSubject" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-500 focus:border-blue-500">

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

<option value="Mathematics">Mathematics</option>

<option value="English Language">English Language</option>

<option value="Basic Science">Basic Science</option>

<option value="Social Studies">Social Studies</option>

<option value="Civic Education">Civic Education</option>

<option value="Computer Studies">Computer Studies</option>

</select>

</div>

</div>

<div class="grid grid-cols-1 md:grid-cols-3 gap-6">

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">Class</label>

<select id="assessmentClass" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-500 focus:border-blue-500">

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

<option value="JSS 1">JSS 1</option>

<option value="JSS 2">JSS 2</option>

<option value="JSS 3">JSS 3</option>

<option value="SS 1">SS 1</option>

<option value="SS 2">SS 2</option>

<option value="SS 3">SS 3</option>

</select>

</div>

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">Assessment Type</label>

<select id="assessmentType" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-500 focus:border-blue-500">

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

<option value="project">Project</option>

<option value="peer\_review">Peer Review</option>

<option value="story">Story/Narrative</option>

<option value="presentation">Presentation</option>

<option value="case\_study">Case Study</option>

<option value="portfolio">Portfolio</option>

</select>

</div>

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">Duration</label>

<select id="assessmentDuration" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-500 focus:border-blue-500">

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

<option value="1 week">1 Week</option>

<option value="2 weeks">2 Weeks</option>

<option value="3 weeks">3 Weeks</option>

<option value="1 month">1 Month</option>

</select>

</div>

</div>

<!-- Assessment Description -->

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">Assessment Description</label>

<textarea id="assessmentDescription" rows="4" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-500 focus:border-blue-500" placeholder="Describe the assessment task in detail..."></textarea>

</div>

<!-- Higher-Order Thinking Questions -->

<div class="bg-blue-50 p-4 rounded-lg">

<h4 class="text-lg font-semibold text-gray-900 mb-4">Higher-Order Thinking Questions</h4>

<p class="text-sm text-gray-600 mb-4">Create questions that test analysis, synthesis, evaluation, and creation (Bloom's Taxonomy)</p>

<textarea id="hotQuestions" rows="6" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-500 focus:border-blue-500" placeholder="Write questions that require students to:&#10;- Analyze information&#10;- Compare and contrast&#10;- Evaluate arguments&#10;- Create solutions&#10;- Apply knowledge to new situations"></textarea>

</div>

<!-- Real-life Application -->

<div class="bg-green-50 p-4 rounded-lg">

<h4 class="text-lg font-semibold text-gray-900 mb-4">Real-life Application</h4>

<textarea id="assessmentRealLife" rows="4" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-500 focus:border-blue-500" placeholder="How does this assessment connect to real-world situations? What practical applications will students explore?"></textarea>

</div>

<!-- Multiple Choice Test Section -->

<div class="bg-indigo-50 p-4 rounded-lg">

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

<h4 class="text-lg font-semibold text-gray-900">Multiple Choice Test Questions</h4>

<div class="flex items-center space-x-2">

<span class="text-sm text-gray-600">Questions: <span id="questionCount">0</span>/20</span>

<button type="button" onclick="addMultipleChoiceQuestion()" id="addQuestionBtn" class="bg-indigo-600 hover:bg-indigo-700 text-white px-3 py-1 rounded text-sm transition-colors">

+ Add Question

</button>

</div>

</div>

<p class="text-sm text-gray-600 mb-4">Create up to 20 multiple choice questions that test higher-order thinking skills</p>

<div id="multipleChoiceQuestions" class="space-y-4">

<!-- Questions will be added here dynamically -->

</div>

<div id="noQuestionsMessage" class="text-center text-gray-500 py-8 border-2 border-dashed border-gray-300 rounded-lg">

<span class="text-3xl mb-2 block">❓</span>

<p class="font-medium">No questions added yet</p>

<p class="text-sm">Click "Add Question" to create your first multiple choice question</p>

</div>

</div>

<!-- Rubric/Grading Criteria -->

<div class="bg-yellow-50 p-4 rounded-lg">

<h4 class="text-lg font-semibold text-gray-900 mb-4">Grading Criteria</h4>

<textarea id="gradingCriteria" rows="5" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-500 focus:border-blue-500" placeholder="Define clear criteria for assessment:&#10;- Excellent (90-100%)&#10;- Good (80-89%)&#10;- Satisfactory (70-79%)&#10;- Needs Improvement (Below 70%)"></textarea>

</div>

<div class="flex justify-end space-x-4 pt-6 border-t">

<button type="button" onclick="closeAssessmentModal()" class="px-6 py-2 border border-gray-300 rounded-lg text-gray-700 hover:bg-gray-50 transition-colors">

Cancel

</button>

<button type="submit" class="px-6 py-2 bg-blue-600 hover:bg-blue-700 text-white rounded-lg transition-colors flex items-center">

<span id="saveAssessmentText">Submit for Review</span>

<div id="saveAssessmentSpinner" class="loading-spinner ml-2 hidden"></div>

</button>

</div>

</form>

</div>

</div>

</div>

</div>

<script>

// Database simulation

class Database {

constructor() {

this.data = JSON.parse(localStorage.getItem('eduplan\_nigeria\_data')) || {

users: {},

lessonPlans: {},

assessments: {},

comments: {},

activities: []

};

this.listeners = {};

}

save() {

localStorage.setItem('eduplan\_nigeria\_data', JSON.stringify(this.data));

}

add(collection, data) {

const id = Date.now().toString();

data.id = id;

data.createdAt = new Date().toISOString();

data.updatedAt = new Date().toISOString();

if (!this.data[collection]) {

this.data[collection] = {};

}

this.data[collection][id] = data;

this.save();

this.triggerListeners(collection);

this.addActivity({

type: 'created',

collection: collection,

data: data,

user: currentUser.displayName || currentUser.email

});

return { id };

}

get(collection, callback) {

const docs = Object.values(this.data[collection] || {});

callback(docs);

}

update(collection, id, updates) {

if (this.data[collection] && this.data[collection][id]) {

this.data[collection][id] = { ...this.data[collection][id], ...updates, updatedAt: new Date().toISOString() };

this.save();

this.triggerListeners(collection);

this.addActivity({

type: 'updated',

collection: collection,

data: this.data[collection][id],

user: currentUser.displayName || currentUser.email

});

}

}

onSnapshot(collection, callback) {

if (!this.listeners[collection]) {

this.listeners[collection] = [];

}

this.listeners[collection].push(callback);

this.get(collection, callback);

}

triggerListeners(collection) {

if (this.listeners[collection]) {

this.listeners[collection].forEach(callback => {

this.get(collection, callback);

});

}

}

addActivity(activity) {

activity.id = Date.now().toString();

activity.timestamp = new Date().toISOString();

this.data.activities.unshift(activity);

if (this.data.activities.length > 50) {

this.data.activities = this.data.activities.slice(0, 50);

}

this.save();

this.triggerListeners('activities');

}

}

const db = new Database();

// Authentication system

class Auth {

constructor() {

this.currentUser = JSON.parse(localStorage.getItem('eduplan\_current\_user'));

}

async signInWithEmailAndPassword(email, password) {

await new Promise(resolve => setTimeout(resolve, 1000));

const users = Object.values(db.data.users || {});

const user = users.find(u => u.email === email && u.password === password);

if (user) {

this.currentUser = user;

localStorage.setItem('eduplan\_current\_user', JSON.stringify(user));

return { user };

} else {

throw new Error('Invalid email or password');

}

}

async createUserWithEmailAndPassword(email, password) {

await new Promise(resolve => setTimeout(resolve, 1500));

const users = Object.values(db.data.users || {});

if (users.find(u => u.email === email)) {

throw new Error('User already exists');

}

const user = {

uid: Date.now().toString(),

email: email,

password: password,

createdAt: new Date().toISOString()

};

db.data.users[user.uid] = user;

db.save();

this.currentUser = user;

localStorage.setItem('eduplan\_current\_user', JSON.stringify(user));

return { user };

}

async updateProfile(updates) {

if (this.currentUser) {

this.currentUser = { ...this.currentUser, ...updates };

db.data.users[this.currentUser.uid] = this.currentUser;

db.save();

localStorage.setItem('eduplan\_current\_user', JSON.stringify(this.currentUser));

}

}

signOut() {

this.currentUser = null;

localStorage.removeItem('eduplan\_current\_user');

}

}

const auth = new Auth();

let currentUser = null;

// Initialize application

async function initializeApp() {

document.getElementById('loadingScreen').style.display = 'flex';

await new Promise(resolve => setTimeout(resolve, 2000));

document.getElementById('loadingScreen').style.display = 'none';

if (auth.currentUser) {

currentUser = auth.currentUser;

showMainApp();

} else {

showAuthScreen();

}

}

function showAuthScreen() {

document.getElementById('authScreen').classList.remove('hidden');

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

}

function showMainApp() {

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

document.getElementById('mainApp').classList.remove('hidden');

document.getElementById('currentUserName').textContent = currentUser.displayName || currentUser.email;

document.getElementById('currentUserRole').textContent = (currentUser.role || 'teacher').charAt(0).toUpperCase() + (currentUser.role || 'teacher').slice(1);

if (currentUser.role === 'admin' || currentUser.role === 'principal') {

document.getElementById('adminNavBtn').classList.remove('hidden');

}

startRealtimeListeners();

loadDashboard();

}

function showLoginForm() {

document.getElementById('loginForm').classList.remove('hidden');

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

}

function showRegisterForm() {

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

document.getElementById('registerForm').classList.remove('hidden');

}

// Authentication handlers

document.getElementById('loginFormElement').addEventListener('submit', async function(e) {

e.preventDefault();

const email = document.getElementById('loginEmail').value;

const password = document.getElementById('loginPassword').value;

document.getElementById('loginButtonText').textContent = 'Signing In...';

document.getElementById('loginSpinner').classList.remove('hidden');

try {

const result = await auth.signInWithEmailAndPassword(email, password);

currentUser = result.user;

showNotification('Welcome back! Successfully signed in.', 'success');

showMainApp();

} catch (error) {

showNotification(error.message, 'error');

} finally {

document.getElementById('loginButtonText').textContent = 'Sign In';

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

}

});

document.getElementById('registerFormElement').addEventListener('submit', async function(e) {

e.preventDefault();

const name = document.getElementById('registerName').value;

const email = document.getElementById('registerEmail').value;

const school = document.getElementById('registerSchool').value;

const role = document.getElementById('registerRole').value;

const password = document.getElementById('registerPassword').value;

document.getElementById('registerButtonText').textContent = 'Creating Account...';

document.getElementById('registerSpinner').classList.remove('hidden');

try {

const result = await auth.createUserWithEmailAndPassword(email, password);

await auth.updateProfile({

displayName: name,

school: school,

role: role

});

currentUser = auth.currentUser;

showNotification('Account created successfully! Welcome to EduPlan Nigeria.', 'success');

showMainApp();

} catch (error) {

showNotification(error.message, 'error');

} finally {

document.getElementById('registerButtonText').textContent = 'Create Account';

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

}

});

function logout() {

auth.signOut();

currentUser = null;

showNotification('Successfully logged out. See you soon!', 'info');

showAuthScreen();

}

// Notification system

function showNotification(message, type = 'info') {

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

notification.className = `notification bg-white rounded-lg shadow-lg p-4 border-l-4 ${

type === 'success' ? 'border-green-500' :

type === 'error' ? 'border-red-500' :

type === 'warning' ? 'border-yellow-500' : 'border-blue-500'

}`;

notification.innerHTML = `

<div class="flex items-center">

<span class="text-lg mr-3">

${type === 'success' ? '✅' :

type === 'error' ? '❌' :

type === 'warning' ? '⚠️' : 'ℹ️'}

</span>

<p class="text-gray-900">${message}</p>

<button onclick="this.parentElement.parentElement.remove()" class="ml-4 text-gray-400 hover:text-gray-600">×</button>

</div>

`;

document.getElementById('notificationContainer').appendChild(notification);

setTimeout(() => notification.classList.add('show'), 100);

setTimeout(() => {

notification.classList.remove('show');

setTimeout(() => notification.remove(), 300);

}, 5000);

}

// Real-time listeners

function startRealtimeListeners() {

db.onSnapshot('lessonPlans', (plans) => {

const userPlans = plans.filter(plan => plan.userId === currentUser.uid);

updateLessonPlansList(userPlans);

updateStats();

if (currentUser.role === 'admin' || currentUser.role === 'principal') {

updateAdminDashboard(plans);

}

});

db.onSnapshot('assessments', (assessments) => {

const userAssessments = assessments.filter(assessment => assessment.userId === currentUser.uid);

updateAssessmentsList(userAssessments);

updateStats();

});

db.onSnapshot('activities', (activities) => {

updateRecentActivity(activities.slice(0, 5));

});

}

// Update functions

function updateLessonPlansList(plans) {

const container = document.getElementById('lessonPlansList');

if (plans.length === 0) {

container.innerHTML = `

<div class="text-center text-gray-500 py-8">

<span class="text-4xl mb-4 block">📝</span>

<p class="text-lg font-medium">No lesson plans yet</p>

<p class="text-sm">Create your first lesson plan to get started!</p>

</div>

`;

return;

}

container.innerHTML = '';

plans.forEach(plan => {

const statusColor = {

'approved': 'bg-green-100 text-green-800',

'pending': 'bg-yellow-100 text-yellow-800',

'needs\_revision': 'bg-red-100 text-red-800'

};

const statusText = {

'approved': 'Approved',

'pending': 'Pending Review',

'needs\_revision': 'Needs Revision'

};

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

planElement.className = 'bg-white rounded-xl shadow-md p-6 border-l-4 border-blue-500 slide-in';

planElement.innerHTML = `

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

<div>

<h3 class="text-lg font-semibold text-gray-900">${plan.subject} - ${plan.topic}</h3>

<p class="text-sm text-gray-600">${plan.class} • ${plan.week} • ${plan.classSize} students</p>

<p class="text-xs text-gray-500 mt-1">Created: ${new Date(plan.createdAt).toLocaleDateString()}</p>

</div>

<div class="flex items-center space-x-3">

<span class="px-3 py-1 rounded-full text-xs font-medium ${statusColor[plan.status || 'pending']}">

${statusText[plan.status || 'pending']}

</span>

<button onclick="viewLessonPlan('${plan.id}')" class="text-blue-600 hover:text-blue-800 text-sm">

View Details

</button>

</div>

</div>

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

<p><strong>Hook Type:</strong> ${plan.hookType || 'Not specified'}</p>

<p><strong>Objectives:</strong> ${plan.objectives ? plan.objectives.substring(0, 100) + '...' : 'Not specified'}</p>

</div>

${plan.adminComments ? `

<div class="mt-4 p-3 bg-yellow-50 border-l-4 border-yellow-400 rounded">

<p class="text-sm font-medium text-yellow-800">Admin Comments:</p>

<p class="text-sm text-yellow-700 mt-1">${plan.adminComments}</p>

</div>

` : ''}

`;

container.appendChild(planElement);

});

}

function updateAssessmentsList(assessments) {

const container = document.getElementById('assessmentsList');

if (assessments.length === 0) {

container.innerHTML = `

<div class="text-center text-gray-500 py-8">

<span class="text-4xl mb-4 block">📊</span>

<p class="text-lg font-medium">No assessments yet</p>

<p class="text-sm">Create your first assessment to get started!</p>

</div>

`;

return;

}

container.innerHTML = '';

assessments.forEach(assessment => {

const statusColor = assessment.status === 'approved' ? 'bg-green-100 text-green-800' : 'bg-yellow-100 text-yellow-800';

const statusText = assessment.status === 'approved' ? 'Approved' : 'Pending Review';

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

assessmentElement.className = 'bg-white rounded-xl shadow-md p-6 border-l-4 border-purple-500 slide-in';

assessmentElement.innerHTML = `

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

<div>

<h3 class="text-lg font-semibold text-gray-900">${assessment.title}</h3>

<p class="text-sm text-gray-600">${assessment.subject} • ${assessment.type}</p>

<p class="text-xs text-gray-500 mt-1">Created: ${new Date(assessment.createdAt).toLocaleDateString()}</p>

</div>

<div class="flex items-center space-x-3">

<span class="px-3 py-1 rounded-full text-xs font-medium ${statusColor}">

${statusText}

</span>

<button onclick="viewAssessment('${assessment.id}')" class="text-purple-600 hover:text-purple-800 text-sm">

View Details

</button>

</div>

</div>

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

<p><strong>Duration:</strong> ${assessment.duration}</p>

<p><strong>Class:</strong> ${assessment.class}</p>

${assessment.totalQuestions ? `<p><strong>Multiple Choice Questions:</strong> ${assessment.totalQuestions}</p>` : ''}

</div>

${assessment.adminComments ? `

<div class="mt-4 p-3 bg-yellow-50 border-l-4 border-yellow-400 rounded">

<p class="text-sm font-medium text-yellow-800">Admin Comments:</p>

<p class="text-sm text-yellow-700 mt-1">${assessment.adminComments}</p>

</div>

` : ''}

`;

container.appendChild(assessmentElement);

});

}

function updateStats() {

db.get('lessonPlans', (plans) => {

const userPlans = plans.filter(plan => plan.userId === currentUser.uid);

document.getElementById('totalLessonPlans').textContent = userPlans.length;

document.getElementById('approvedPlans').textContent = userPlans.filter(p => p.status === 'approved').length;

document.getElementById('pendingPlans').textContent = userPlans.filter(p => p.status === 'pending').length;

});

db.get('assessments', (assessments) => {

const userAssessments = assessments.filter(assessment => assessment.userId === currentUser.uid);

document.getElementById('totalAssessments').textContent = userAssessments.length;

});

}

function updateRecentActivity(activities) {

const container = document.getElementById('recentActivity');

if (activities.length === 0) {

container.innerHTML = `

<div class="text-center text-gray-500 py-4">

<p>No recent activity</p>

</div>

`;

return;

}

container.innerHTML = '';

activities.forEach(activity => {

const timeAgo = getTimeAgo(new Date(activity.timestamp));

const icon = activity.type === 'created' ? '➕' : '✏️';

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

activityElement.className = 'flex items-center space-x-3 p-3 bg-gray-50 rounded-lg';

activityElement.innerHTML = `

<span>${icon}</span>

<div class="flex-1">

<p class="text-sm font-medium">${activity.user} ${activity.type} a ${activity.collection.slice(0, -1)}</p>

<p class="text-xs text-gray-500">${timeAgo}</p>

</div>

`;

container.appendChild(activityElement);

});

}

function updateAdminDashboard(allPlans) {

const schoolPlans = allPlans.filter(plan => {

const teacher = Object.values(db.data.users || {}).find(u => u.uid === plan.userId);

return teacher && teacher.school === currentUser.school;

});

const pendingPlans = schoolPlans.filter(plan => plan.status === 'pending');

const approvedToday = schoolPlans.filter(plan => {

const today = new Date().toDateString();

return plan.status === 'approved' && new Date(plan.updatedAt).toDateString() === today;

});

document.getElementById('adminPendingPlans').textContent = pendingPlans.length;

document.getElementById('approvedToday').textContent = approvedToday.length;

const uniqueTeachers = new Set(schoolPlans.map(plan => plan.userId)).size;

document.getElementById('totalTeachers').textContent = uniqueTeachers;

updateAdminSubmissionsList(pendingPlans);

}

function updateAdminSubmissionsList(pendingPlans) {

const container = document.getElementById('adminSubmissionsList');

if (pendingPlans.length === 0) {

container.innerHTML = `

<div class="text-center text-gray-500 py-8">

<span class="text-4xl mb-4 block">✅</span>

<p class="text-lg font-medium">All caught up!</p>

<p class="text-sm">No submissions pending review</p>

</div>

`;

return;

}

container.innerHTML = '';

pendingPlans.forEach(plan => {

const teacher = Object.values(db.data.users || {}).find(u => u.uid === plan.userId);

const teacherName = teacher ? (teacher.displayName || teacher.email) : 'Unknown Teacher';

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

planElement.className = 'bg-white border border-gray-200 rounded-lg p-6 hover:shadow-md transition-shadow';

planElement.innerHTML = `

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

<div class="flex-1">

<div class="flex items-center space-x-2 mb-2">

<h3 class="text-lg font-semibold text-gray-900">${plan.subject} - ${plan.topic}</h3>

<span class="px-2 py-1 bg-yellow-100 text-yellow-800 text-xs rounded-full">Pending</span>

</div>

<p class="text-sm text-gray-600 mb-1"><strong>Teacher:</strong> ${teacherName}</p>

<p class="text-sm text-gray-600 mb-1"><strong>Class:</strong> ${plan.class} • <strong>Students:</strong> ${plan.classSize}</p>

<p class="text-sm text-gray-600"><strong>Hook Type:</strong> ${plan.hookType}</p>

<p class="text-xs text-gray-500 mt-2">Submitted: ${new Date(plan.createdAt).toLocaleDateString()}</p>

</div>

<div class="flex space-x-2 ml-4">

<button onclick="reviewSubmission('${plan.id}', 'approved')" class="bg-green-600 hover:bg-green-700 text-white px-4 py-2 rounded text-sm transition-colors">

Approve

</button>

<button onclick="showCommentModal('${plan.id}', 'lessonPlans')" class="bg-blue-600 hover:bg-blue-700 text-white px-4 py-2 rounded text-sm transition-colors">

Comment

</button>

<button onclick="reviewSubmission('${plan.id}', 'needs\_revision')" class="bg-red-600 hover:bg-red-700 text-white px-4 py-2 rounded text-sm transition-colors">

Request Revision

</button>

</div>

</div>

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

<p><strong>Objectives:</strong> ${plan.objectives ? plan.objectives.substring(0, 150) + '...' : 'Not specified'}</p>

</div>

`;

container.appendChild(planElement);

});

}

// Navigation

function showSection(sectionName) {

document.querySelectorAll('.section').forEach(section => {

section.classList.add('hidden');

});

document.getElementById(sectionName).classList.remove('hidden');

document.getElementById(sectionName).classList.add('fade-in');

document.querySelectorAll('.nav-btn').forEach(btn => {

btn.classList.remove('border-green-500', 'text-green-600');

btn.classList.add('border-transparent', 'text-gray-500');

});

event.target.classList.remove('border-transparent', 'text-gray-500');

event.target.classList.add('border-green-500', 'text-green-600');

if (sectionName === 'admin-review') {

loadAdminDashboard();

}

}

function loadDashboard() {

updateStats();

}

function loadAdminDashboard() {

db.get('lessonPlans', (plans) => {

updateAdminDashboard(plans);

});

}

// Modal functions

function showCreateLessonPlan() {

document.getElementById('lessonPlanForm').reset();

document.getElementById('lessonPlanModal').classList.remove('hidden');

}

function closeLessonPlanModal() {

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

}

function showCreateAssessment() {

document.getElementById('assessmentForm').reset();

document.getElementById('assessmentModal').classList.remove('hidden');

}

function closeAssessmentModal() {

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

resetMultipleChoiceQuestions();

}

function viewLessonPlan(id) {

db.get('lessonPlans', (plans) => {

const plan = plans.find(p => p.id === id);

if (plan) {

showLessonPlanDetails(plan);

}

});

}

function viewAssessment(id) {

db.get('assessments', (assessments) => {

const assessment = assessments.find(a => a.id === id);

if (assessment) {

showAssessmentDetails(assessment);

}

});

}

function showLessonPlanDetails(plan) {

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

modal.className = 'fixed inset-0 bg-black bg-opacity-50 z-50 flex items-center justify-center p-4';

modal.innerHTML = `

<div class="bg-white rounded-xl shadow-xl max-w-4xl w-full max-h-screen overflow-y-auto">

<div class="p-6 border-b">

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

<h3 class="text-xl font-bold text-gray-900">Lesson Plan Details</h3>

<button onclick="this.closest('.fixed').remove()" class="text-gray-400 hover:text-gray-600 text-2xl">×</button>

</div>

</div>

<div class="p-6 space-y-4">

<div class="grid grid-cols-2 gap-4">

<div><strong>Subject:</strong> ${plan.subject}</div>

<div><strong>Class:</strong> ${plan.class}</div>

<div><strong>Class Size:</strong> ${plan.classSize}</div>

<div><strong>Duration:</strong> ${plan.duration}</div>

<div><strong>Week:</strong> ${plan.week}</div>

<div><strong>Hook Type:</strong> ${plan.hookType}</div>

</div>

<div><strong>Topic:</strong> ${plan.topic}</div>

<div><strong>Learning Objectives:</strong><br>${plan.objectives || 'Not specified'}</div>

<div><strong>Lesson Hook:</strong><br>${plan.lessonHook || 'Not specified'}</div>

<div><strong>Entry Behaviour:</strong><br>${plan.entryBehaviour || 'Not specified'}</div>

<div><strong>Direct Instruction:</strong><br>${plan.directInstruction || 'Not specified'}</div>

<div><strong>Real-life Application:</strong><br>${plan.realLifeApplication || 'Not specified'}</div>

<div><strong>Exit Tickets:</strong><br>${plan.exitTickets || 'Not specified'}</div>

</div>

</div>

`;

document.body.appendChild(modal);

}

function showAssessmentDetails(assessment) {

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

modal.className = 'fixed inset-0 bg-black bg-opacity-50 z-50 flex items-center justify-center p-4';

let questionsHtml = '';

if (assessment.multipleChoiceQuestions && assessment.multipleChoiceQuestions.length > 0) {

questionsHtml = `

<div class="mt-6">

<h4 class="text-lg font-semibold text-gray-900 mb-4">Multiple Choice Questions (${assessment.multipleChoiceQuestions.length})</h4>

<div class="space-y-4 max-h-96 overflow-y-auto">

${assessment.multipleChoiceQuestions.map((question, index) => `

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

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

<h5 class="font-medium text-gray-900">Question ${index + 1}</h5>

<span class="px-2 py-1 text-xs rounded-full ${

question.difficulty === 'easy' ? 'bg-green-100 text-green-800' :

question.difficulty === 'medium' ? 'bg-yellow-100 text-yellow-800' :

'bg-red-100 text-red-800'

}">${question.difficulty}</span>

</div>

<p class="text-gray-700 mb-3">${question.text}</p>

<div class="grid grid-cols-2 gap-2 mb-3">

<div class="text-sm ${question.correctAnswer === 'A' ? 'font-semibold text-green-700' : 'text-gray-600'}">

A) ${question.options.A}

</div>

<div class="text-sm ${question.correctAnswer === 'B' ? 'font-semibold text-green-700' : 'text-gray-600'}">

B) ${question.options.B}

</div>

<div class="text-sm ${question.correctAnswer === 'C' ? 'font-semibold text-green-700' : 'text-gray-600'}">

C) ${question.options.C}

</div>

<div class="text-sm ${question.correctAnswer === 'D' ? 'font-semibold text-green-700' : 'text-gray-600'}">

D) ${question.options.D}

</div>

</div>

<div class="text-sm text-green-700">

<strong>Correct Answer:</strong> ${question.correctAnswer}

</div>

${question.explanation ? `

<div class="text-sm text-gray-600 mt-2">

<strong>Explanation:</strong> ${question.explanation}

</div>

` : ''}

</div>

`).join('')}

</div>

</div>

`;

}

modal.innerHTML = `

<div class="bg-white rounded-xl shadow-xl max-w-5xl w-full max-h-screen overflow-y-auto">

<div class="p-6 border-b">

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

<h3 class="text-xl font-bold text-gray-900">Assessment Details</h3>

<button onclick="this.closest('.fixed').remove()" class="text-gray-400 hover:text-gray-600 text-2xl">×</button>

</div>

</div>

<div class="p-6 space-y-4">

<div class="grid grid-cols-2 gap-4">

<div><strong>Title:</strong> ${assessment.title}</div>

<div><strong>Subject:</strong> ${assessment.subject}</div>

<div><strong>Class:</strong> ${assessment.class}</div>

<div><strong>Type:</strong> ${assessment.type}</div>

<div><strong>Duration:</strong> ${assessment.duration}</div>

<div><strong>Total Questions:</strong> ${assessment.totalQuestions || 0}</div>

</div>

<div><strong>Description:</strong><br>${assessment.description || 'Not specified'}</div>

<div><strong>Higher-Order Thinking Questions:</strong><br>${assessment.hotQuestions || 'Not specified'}</div>

<div><strong>Real-life Application:</strong><br>${assessment.realLifeApplication || 'Not specified'}</div>

<div><strong>Grading Criteria:</strong><br>${assessment.gradingCriteria || 'Not specified'}</div>

${questionsHtml}

</div>

</div>

`;

document.body.appendChild(modal);

}

function reviewSubmission(id, status) {

db.update('lessonPlans', id, {

status: status,

reviewedBy: currentUser.uid,

reviewedAt: new Date().toISOString()

});

const statusText = status === 'approved' ? 'approved' : 'marked for revision';

showNotification(`Lesson plan ${statusText} successfully!`, 'success');

}

function showCommentModal(id, collection) {

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

modal.className = 'fixed inset-0 bg-black bg-opacity-50 z-50 flex items-center justify-center p-4';

modal.innerHTML = `

<div class="bg-white rounded-xl shadow-xl max-w-2xl w-full">

<div class="p-6 border-b">

<h3 class="text-xl font-bold text-gray-900">Add Comments & Corrections</h3>

</div>

<div class="p-6">

<textarea id="adminComment" rows="6" class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-500 focus:border-blue-500" placeholder="Provide detailed feedback and suggestions for improvement..."></textarea>

<div class="flex justify-end space-x-4 mt-6">

<button onclick="this.closest('.fixed').remove()" class="px-6 py-2 border border-gray-300 rounded-lg text-gray-700 hover:bg-gray-50">

Cancel

</button>

<button onclick="saveComment('${id}', '${collection}'); this.closest('.fixed').remove();" class="px-6 py-2 bg-blue-600 hover:bg-blue-700 text-white rounded-lg">

Save Comment

</button>

</div>

</div>

</div>

`;

document.body.appendChild(modal);

}

function saveComment(id, collection) {

const comment = document.getElementById('adminComment').value;

if (comment.trim()) {

db.update(collection, id, {

adminComments: comment,

status: 'needs\_revision',

reviewedBy: currentUser.uid,

reviewedAt: new Date().toISOString()

});

showNotification('Comment added successfully!', 'success');

}

}

// Multiple Choice Questions Management

let questionCounter = 0;

const maxQuestions = 20;

function addMultipleChoiceQuestion() {

if (questionCounter >= maxQuestions) {

showNotification('Maximum of 20 questions allowed', 'warning');

return;

}

questionCounter++;

const questionId = `question\_${questionCounter}`;

const questionContainer = document.getElementById('multipleChoiceQuestions');

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

// Hide the no questions message

noQuestionsMessage.style.display = 'none';

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

questionElement.className = 'bg-white border border-indigo-200 rounded-lg p-4 fade-in';

questionElement.id = questionId;

questionElement.innerHTML = `

<div class="flex justify-between items-start mb-3">

<h5 class="text-md font-semibold text-gray-900">Question ${questionCounter}</h5>

<button type="button" onclick="removeQuestion('${questionId}')" class="text-red-600 hover:text-red-800 text-sm">

🗑️ Remove

</button>

</div>

<div class="mb-4">

<label class="block text-sm font-medium text-gray-700 mb-2">Question Text</label>

<textarea name="questionText\_${questionCounter}" rows="3" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-indigo-500 focus:border-indigo-500" placeholder="Write a higher-order thinking question that requires analysis, evaluation, or synthesis..."></textarea>

</div>

<div class="grid grid-cols-1 md:grid-cols-2 gap-3 mb-4">

<div>

<label class="block text-sm font-medium text-gray-700 mb-1">Option A</label>

<input type="text" name="optionA\_${questionCounter}" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-indigo-500 focus:border-indigo-500" placeholder="First option">

</div>

<div>

<label class="block text-sm font-medium text-gray-700 mb-1">Option B</label>

<input type="text" name="optionB\_${questionCounter}" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-indigo-500 focus:border-indigo-500" placeholder="Second option">

</div>

<div>

<label class="block text-sm font-medium text-gray-700 mb-1">Option C</label>

<input type="text" name="optionC\_${questionCounter}" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-indigo-500 focus:border-indigo-500" placeholder="Third option">

</div>

<div>

<label class="block text-sm font-medium text-gray-700 mb-1">Option D</label>

<input type="text" name="optionD\_${questionCounter}" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-indigo-500 focus:border-indigo-500" placeholder="Fourth option">

</div>

</div>

<div class="grid grid-cols-1 md:grid-cols-2 gap-4">

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">Correct Answer</label>

<select name="correctAnswer\_${questionCounter}" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-indigo-500 focus:border-indigo-500">

<option value="">Select correct answer</option>

<option value="A">Option A</option>

<option value="B">Option B</option>

<option value="C">Option C</option>

<option value="D">Option D</option>

</select>

</div>

<div>

<label class="block text-sm font-medium text-gray-700 mb-2">Difficulty Level</label>

<select name="difficulty\_${questionCounter}" required class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-indigo-500 focus:border-indigo-500">

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

<option value="easy">Easy</option>

<option value="medium">Medium</option>

<option value="hard">Hard</option>

</select>

</div>

</div>

<div class="mt-4">

<label class="block text-sm font-medium text-gray-700 mb-2">Explanation (Why this answer is correct)</label>

<textarea name="explanation\_${questionCounter}" rows="2" class="w-full px-3 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-indigo-500 focus:border-indigo-500" placeholder="Provide a clear explanation for the correct answer..."></textarea>

</div>

`;

questionContainer.appendChild(questionElement);

updateQuestionCount();

// Scroll to the new question

questionElement.scrollIntoView({ behavior: 'smooth', block: 'center' });

}

function removeQuestion(questionId) {

const questionElement = document.getElementById(questionId);

if (questionElement) {

questionElement.remove();

questionCounter--;

updateQuestionCount();

renumberQuestions();

// Show no questions message if no questions left

const questionContainer = document.getElementById('multipleChoiceQuestions');

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

if (questionContainer.children.length === 0) {

noQuestionsMessage.style.display = 'block';

}

}

}

function renumberQuestions() {

const questions = document.querySelectorAll('#multipleChoiceQuestions > div');

questions.forEach((question, index) => {

const newNumber = index + 1;

const questionTitle = question.querySelector('h5');

if (questionTitle) {

questionTitle.textContent = `Question ${newNumber}`;

}

});

}

function updateQuestionCount() {

document.getElementById('questionCount').textContent = questionCounter;

const addButton = document.getElementById('addQuestionBtn');

if (questionCounter >= maxQuestions) {

addButton.disabled = true;

addButton.classList.add('opacity-50', 'cursor-not-allowed');

addButton.textContent = 'Maximum Reached';

} else {

addButton.disabled = false;

addButton.classList.remove('opacity-50', 'cursor-not-allowed');

addButton.textContent = '+ Add Question';

}

}

function collectMultipleChoiceQuestions() {

const questions = [];

const questionElements = document.querySelectorAll('#multipleChoiceQuestions > div');

questionElements.forEach((element, index) => {

const questionNumber = index + 1;

const questionData = {

number: questionNumber,

text: element.querySelector(`textarea[name\*="questionText"]`).value,

options: {

A: element.querySelector(`input[name\*="optionA"]`).value,

B: element.querySelector(`input[name\*="optionB"]`).value,

C: element.querySelector(`input[name\*="optionC"]`).value,

D: element.querySelector(`input[name\*="optionD"]`).value

},

correctAnswer: element.querySelector(`select[name\*="correctAnswer"]`).value,

difficulty: element.querySelector(`select[name\*="difficulty"]`).value,

explanation: element.querySelector(`textarea[name\*="explanation"]`).value

};

questions.push(questionData);

});

return questions;

}

function resetMultipleChoiceQuestions() {

questionCounter = 0;

document.getElementById('multipleChoiceQuestions').innerHTML = '';

document.getElementById('noQuestionsMessage').style.display = 'block';

updateQuestionCount();

}

// Form handlers

document.getElementById('lessonPlanForm').addEventListener('submit', async function(e) {

e.preventDefault();

document.getElementById('saveLessonPlanText').textContent = 'Submitting...';

document.getElementById('saveLessonPlanSpinner').classList.remove('hidden');

try {

// Get selected teaching methods

const teachingMethods = Array.from(document.querySelectorAll('input[name="teachingMethods"]:checked'))

.map(checkbox => checkbox.value);

const lessonPlan = {

userId: currentUser.uid,

subject: document.getElementById('subject').value,

class: document.getElementById('class').value,

classSize: document.getElementById('classSize').value,

duration: document.getElementById('duration').value,

week: document.getElementById('week').value,

hookType: document.getElementById('hookType').value,

lessonHook: document.getElementById('lessonHook').value,

topic: document.getElementById('topic').value,

objectives: document.getElementById('objectives').value,

entryBehaviour: document.getElementById('entryBehaviour').value,

directInstruction: document.getElementById('directInstruction').value,

advancedLearners: document.getElementById('advancedLearners').value,

averageLearners: document.getElementById('averageLearners').value,

strugglingLearners: document.getElementById('strugglingLearners').value,

teachingMethods: teachingMethods,

learningResources: document.getElementById('learningResources').value,

lessonSummary: document.getElementById('lessonSummary').value,

exitTickets: document.getElementById('exitTickets').value,

realLifeApplication: document.getElementById('realLifeApplication').value,

status: 'pending'

};

await new Promise(resolve => setTimeout(resolve, 1000));

db.add('lessonPlans', lessonPlan);

closeLessonPlanModal();

showNotification('Lesson plan submitted for review successfully!', 'success');

} catch (error) {

showNotification('Error submitting lesson plan: ' + error.message, 'error');

} finally {

document.getElementById('saveLessonPlanText').textContent = 'Submit for Review';

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

}

});

document.getElementById('assessmentForm').addEventListener('submit', async function(e) {

e.preventDefault();

document.getElementById('saveAssessmentText').textContent = 'Submitting...';

document.getElementById('saveAssessmentSpinner').classList.remove('hidden');

try {

// Collect multiple choice questions

const multipleChoiceQuestions = collectMultipleChoiceQuestions();

const assessment = {

userId: currentUser.uid,

title: document.getElementById('assessmentTitle').value,

subject: document.getElementById('assessmentSubject').value,

class: document.getElementById('assessmentClass').value,

type: document.getElementById('assessmentType').value,

duration: document.getElementById('assessmentDuration').value,

description: document.getElementById('assessmentDescription').value,

hotQuestions: document.getElementById('hotQuestions').value,

realLifeApplication: document.getElementById('assessmentRealLife').value,

gradingCriteria: document.getElementById('gradingCriteria').value,

multipleChoiceQuestions: multipleChoiceQuestions,

totalQuestions: multipleChoiceQuestions.length,

status: 'pending'

};

await new Promise(resolve => setTimeout(resolve, 1000));

db.add('assessments', assessment);

closeAssessmentModal();

showNotification(`Assessment with ${multipleChoiceQuestions.length} questions submitted for review successfully!`, 'success');

} catch (error) {

showNotification('Error submitting assessment: ' + error.message, 'error');

} finally {

document.getElementById('saveAssessmentText').textContent = 'Submit for Review';

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

}

});

// Utility functions

function getTimeAgo(date) {

const now = new Date();

const diffInSeconds = Math.floor((now - date) / 1000);

if (diffInSeconds < 60) return 'Just now';

if (diffInSeconds < 3600) return `${Math.floor(diffInSeconds / 60)} minutes ago`;

if (diffInSeconds < 86400) return `${Math.floor(diffInSeconds / 3600)} hours ago`;

return `${Math.floor(diffInSeconds / 86400)} days ago`;

}

// Initialize application

document.addEventListener('DOMContentLoaded', function() {

initializeApp();

});

</script>

<script>(function(){function c(){var b=a.contentDocument||a.contentWindow.document;if(b){var d=b.createElement('script');d.innerHTML="window.\_\_CF$cv$params={r:'982c589a377d57a5',t:'MTc1ODQ4Njg5NS4wMDAwMDA='};var a=document.createElement('script');a.nonce='';a.src='/cdn-cgi/challenge-platform/scripts/jsd/main.js';document.getElementsByTagName('head')[0].appendChild(a);";b.getElementsByTagName('head')[0].appendChild(d)}}if(document.body){var a=document.createElement('iframe');a.height=1;a.width=1;a.style.position='absolute';a.style.top=0;a.style.left=0;a.style.border='none';a.style.visibility='hidden';document.body.appendChild(a);if('loading'!==document.readyState)c();else if(window.addEventListener)document.addEventListener('DOMContentLoaded',c);else{var e=document.onreadystatechange||function(){};document.onreadystatechange=function(b){e(b);'loading'!==document.readyState&&(document.onreadystatechange=e,c())}}}})();</script></body>

</html>