1. client/dist/Module Assets & Logos/ has all the svg module files and platform Logo all are svg files.

2. client/dist/js/app.js

class SmartGenEduXApp {

constructor() {

this.currentUser = null;

this.currentRole = 'school\_admin';

this.apiBase = '/api';

this.templateVariablesMap = {};

this.init();

}

init() {

this.setupEventListeners();

this.checkAuthStatus();

}

setupEventListeners() {

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

if (loginForm) {

loginForm.addEventListener('submit', (e) => this.handleLogin(e));

}

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

if (e.target.classList.contains('modal')) this.closeModal();

});

document.addEventListener('keydown', (e) => {

if (e.key === 'Escape') this.closeModal();

});

// Arattai Send Button

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

if (sendButton) {

sendButton.addEventListener('click', () => this.handleArattaiSend());

}

}

async handleLogin(event) {

event.preventDefault();

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

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

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

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

loginText.style.display = 'none';

loginLoading.style.display = 'inline-block';

try {

await this.delay(1500); // Simulate API call delay

const user = this.getUserByEmail(email);

if (!user) throw new Error('Invalid credentials');

this.currentUser = user;

this.currentRole = user.role;

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

localStorage.setItem('smartgenedux\_role', user.role);

this.showDashboard();

} catch (err) {

alert('Login failed: ' + err.message);

loginText.style.display = 'inline';

loginLoading.style.display = 'none';

}

}

getUserByEmail(email) {

const users = {

'super@smartgenedux.org': {

id: 'super\_001',

name: 'Super Administrator',

email: 'super@smartgenedux.org',

role: 'super\_admin',

avatar: 'SA',

permissions: ['all'],

},

'admin@dps.edu': {

id: 'admin\_001',

name: 'Dr. Rajesh Kumar',

email: 'admin@dps.edu',

role: 'school\_admin',

school: 'Delhi Public School',

schoolId: 'school\_001',

avatar: 'RK',

permissions: ['school\_management', 'student\_management', 'teacher\_management'],

},

'priya@dps.edu': {

id: 'teacher\_001',

name: 'Ms. Priya Sharma',

email: 'priya@dps.edu',

role: 'teacher',

school: 'Delhi Public School',

schoolId: 'school\_001',

subjects: ['Mathematics', 'Science'],

classes: ['Class 1-A', 'Class 2-A'],

avatar: 'PS',

permissions: ['attendance', 'student\_records', 'assignments'],

},

};

return users[email.toLowerCase()] || null;

}

checkAuthStatus() {

const savedUser = localStorage.getItem('smartgenedux\_user');

const savedRole = localStorage.getItem('smartgenedux\_role');

if (savedUser && savedRole) {

this.currentUser = JSON.parse(savedUser);

this.currentRole = savedRole;

this.showDashboard();

}

}

showDashboard() {

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

document.getElementById('dashboard').classList.add('active');

this.updateUserInfo();

this.loadDashboardStats();

this.loadModules();

this.updateRoleButtons();

}

updateUserInfo() {

if (this.currentUser) {

document.getElementById('userName').textContent = this.currentUser.name;

document.getElementById('userRole').textContent = this.getRoleDisplayName(this.currentRole);

document.getElementById('userAvatar').textContent = this.currentUser.avatar;

}

}

getRoleDisplayName(role) {

const roleNames = {

super\_admin: 'Super Administrator',

school\_admin: 'School Administrator',

teacher: 'Teacher',

};

return roleNames[role] || role;

}

updateRoleButtons() {

const buttons = document.querySelectorAll('.role-btn');

buttons.forEach((btn) => {

btn.classList.remove('active');

if (btn.textContent.toLowerCase().replace(' ', '\_') === this.currentRole) {

btn.classList.add('active');

}

});

}

switchRole(role) {

this.currentRole = role;

localStorage.setItem('smartgenedux\_role', role);

this.updateUserInfo();

this.loadDashboardStats();

this.updateRoleButtons();

}

async loadDashboardStats() {

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

try {

const stats = await this.getStatsForRole(this.currentRole);

statsGrid.innerHTML = stats

.map(

(stat) => `

<div class="stat-card">

<div class="stat-icon">${stat.icon}</div>

<div class="stat-number">${stat.value}</div>

<div class="stat-label">${stat.label}</div>

</div>

`

)

.join('');

} catch (error) {

console.error('Failed to load stats:', error);

}

}

async getStatsForRole(role) {

await this.delay(500);

const statsData = {

super\_admin: [

{ icon: '🏫', value: '247', label: 'Total Schools' },

{ icon: '👥', value: '15,432', label: 'Total Students' },

{ icon: '👨‍🏫', value: '2,186', label: 'Total Teachers' },

{ icon: '💰', value: '₹24.7L', label: 'Monthly Revenue' },

{ icon: '📊', value: '94%', label: 'System Uptime' },

{ icon: '🎯', value: '98.2%', label: 'Client Satisfaction' },

],

school\_admin: [

{ icon: '👥', value: '1,247', label: 'Total Students' },

{ icon: '👨‍🏫', value: '89', label: 'Teaching Staff' },

{ icon: '📚', value: '47', label: 'Active Classes' },

{ icon: '💰', value: '₹89,500', label: 'Fee Collection' },

{ icon: '📅', value: '95%', label: 'Attendance Rate' },

{ icon: '🏆', value: '87%', label: 'Academic Performance' },

],

teacher: [

{ icon: '👥', value: '156', label: 'My Students' },

{ icon: '📚', value: '6', label: 'Classes Assigned' },

{ icon: '📝', value: '23', label: 'Pending Assessments' },

{ icon: '📅', value: '98%', label: 'Class Attendance' },

{ icon: '🎯', value: '92%', label: 'Assignment Completion' },

{ icon: '⭐', value: '4.8/5', label: 'Student Rating' },

],

};

return statsData[role] || [];

}

loadModules() {

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

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

academicModules.innerHTML = this.generateModuleCards(this.getAcademicModules());

premiumModules.innerHTML = this.generateModuleCards(this.getPremiumModules());

}

generateModuleCards(modules) {

return modules

.map(

(module) => `

<div class="module-card ${module.isPremium ? 'premium' : ''}" onclick="app.openModuleDetails('${module.id}')">

${module.isPremium ? '<div class="premium-badge">PREMIUM</div>' : ''}

<img src="./assets/${module.icon}" alt="${module.name}" class="module-icon" onerror="this.style.display='none'">

<div class="module-title">${module.name}</div>

<div class="module-description">${module.description}</div>

<ul class="module-features">

${module.features.slice(0, 3).map((feature) => `<li>${feature}</li>`).join('')}

</ul>

<div class="module-status">

<span class="status-badge">${module.status}</span>

<button class="module-btn" onclick="event.stopPropagation(); app.launchModule('${module.id}')">Launch</button>

</div>

</div>

`

)

.join('');

}

getAcademicModules() {

return [

// Example academic modules — replace with your real data

{

id: 'attendance',

icon: 'attendance.png',

name: 'Attendance Management',

description: 'Track daily attendance of students and staff.',

features: ['Daily attendance', 'Reports', 'Alerts'],

status: 'Active',

isPremium: false,

},

{

id: 'academic-performance',

icon: 'performance.png',

name: 'Academic Performance',

description: 'Manage student assessments and grading.',

features: ['Grades', 'Assessments', 'Reports'],

status: 'Active',

isPremium: false,

},

];

}

getPremiumModules() {

return [

// Example premium modules — replace with your real data

{

id: 'sms-integration',

icon: 'sms.png',

name: 'SMS/WhatsApp Integration',

description: 'Send automated alerts through SMS and WhatsApp.',

features: ['Automatic alerts', 'Bulk messaging', 'Templates'],

status: 'Premium',

isPremium: true,

},

{

id: 'fee-management',

icon: 'fees.png',

name: 'Fee Management',

description: 'Automate fee collection and reminders.',

features: ['Fee tracking', 'Payment reminders', 'Reports'],

status: 'Premium',

isPremium: true,

},

];

}

openModuleDetails(moduleId) {

// Show modal with module details

// You can implement this modal logic as needed

alert(`Open details for module: ${moduleId}`);

}

launchModule(moduleId) {

// Implement your module launch logic here, e.g. navigate to module page

alert(`Launching module: ${moduleId}`);

}

closeModal() {

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

if (modal) {

modal.style.display = 'none';

}

}

logout() {

localStorage.removeItem('smartgenedux\_user');

localStorage.removeItem('smartgenedux\_role');

this.currentUser = null;

this.currentRole = 'school\_admin';

document.getElementById('dashboard').classList.remove('active');

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

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

if (loginForm) loginForm.reset();

}

delay(ms) {

return new Promise((resolve) => setTimeout(resolve, ms));

}

// === Arattai Integration === //

async fetchArattaiTemplates() {

const res = await fetch(`${this.apiBase}/arattai-alert/templates`);

if (!res.ok) throw new Error('Failed to fetch templates');

return await res.json();

}

async fetchArattaiContacts() {

const res = await fetch(`${this.apiBase}/arattai-alert/contacts`);

if (!res.ok) throw new Error('Failed to fetch contacts');

return await res.json();

}

async fetchRecentArattaiMessages() {

const res = await fetch(`${this.apiBase}/arattai-alert/scheduled`);

if (!res.ok) throw new Error('Failed to fetch recent messages');

return await res.json();

}

async sendArattaiMessage(payload) {

const res = await fetch(`${this.apiBase}/arattai-alert/send`, {

method: 'POST',

headers: { 'Content-Type': 'application/json' },

body: JSON.stringify(payload),

});

return await res.json();

}

async setupArattaiPage() {

if (!window.location.pathname.includes('arattai.html')) return;

try {

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

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

const variablesContainer = document.getElementById('templateVariablesContainer');

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

const contacts = await this.fetchArattaiContacts();

recipientSelect.innerHTML =

'<option value="">Select a recipient</option>' +

contacts.map(

(c) =>

`<option value="${c.phoneNumber}">${c.parentName} (${c.studentName})</option>`

).join('');

const templates = await this.fetchArattaiTemplates();

this.templateVariablesMap = {};

templateSelect.innerHTML =

'<option value="">Select a template</option>' +

templates.map((t) => {

this.templateVariablesMap[t.id] = t.variables || [];

return `<option value="${t.id}">${t.name}</option>`;

}).join('');

templateSelect.addEventListener('change', (e) => {

variablesContainer.innerHTML = '';

const selectedId = e.target.value;

if (!selectedId || !this.templateVariablesMap[selectedId]) return;

this.templateVariablesMap[selectedId].forEach((variableName) => {

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

label.textContent = `${variableName.charAt(0).toUpperCase() + variableName.slice(1)}:`;

label.className = 'block text-sm font-medium mb-1';

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

input.type = 'text';

input.id = `var\_${variableName}`;

input.className = 'w-full border rounded px-3 py-2 mb-3';

variablesContainer.appendChild(label);

variablesContainer.appendChild(input);

});

});

const recentMessages = await this.fetchRecentArattaiMessages();

recentMessagesBody.innerHTML = recentMessages.map((msg) => `

<tr class="border-t">

<td class="px-4 py-2">${new Date(msg.scheduledFor || msg.sentAt).toLocaleString()}</td>

<td class="px-4 py-2">${msg.category || 'N/A'}</td>

<td class="px-4 py-2">${msg.recipientNumber}</td>

<td class="px-4 py-2">${msg.message}</td>

<td class="px-4 py-2">

<span class="${msg.status === 'sent' ? 'bg-green-100 text-green-800' : 'bg-red-100 text-red-800'} px-2 py-1 rounded text-xs">

${msg.status || 'pending'}

</span>

</td>

</tr>

`).join('');

} catch (error) {

console.error('Error loading Arattai data:', error);

}

}

async handleArattaiSend() {

const recipientNumber = document.getElementById('recipientSelect').value;

const templateId = document.getElementById('templateSelect').value;

const messageContent = document.getElementById('messageContent').value.trim();

if (!recipientNumber) {

alert('Please select a recipient');

return;

}

if (!templateId) {

alert('Please select a template');

return;

}

if (!messageContent) {

alert('Message cannot be empty');

return;

}

const variables = {};

(this.templateVariablesMap[templateId] || []).forEach((variableName) => {

const input = document.getElementById(`var\_${variableName}`);

if (input) variables[variableName] = input.value.trim();

});

try {

const response = await this.sendArattaiMessage({

recipientNumber,

templateId,

variables,

message: messageContent,

});

if (response.success) {

alert('Message sent successfully');

document.getElementById('messageContent').value = '';

document.getElementById('charCount').textContent = 'Characters: 0/1000';

// Optionally reload recent messages here

} else {

alert('Failed to send message');

}

} catch (error) {

alert('Error sending message');

console.error(error);

}

}

}

// Initialize app and setup Arattai page on DOMContentLoaded

const app = new SmartGenEduXApp();

window.app = app;

window.switchRole = (role) => app.switchRole(role);

window.logout = () => app.logout();

window.closeModal = () => app.closeModal();

document.addEventListener('DOMContentLoaded', () => {

app.setupArattaiPage();

});

3. client/dist/modules/Cbse\_Registration.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>CBSE Registration Management System</title>

<!-- Bootstrap 5 CDN -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css">

<style>

body { background-color: #e3f2fd; font-family: 'Inter', sans-serif; }

.container-fluid { max-width: 1200px; margin-top: 20px; }

.card-header { background-color: #1976d2; color: white; font-weight: bold; border-radius: 0.5rem 0.5rem 0 0; }

.section-card { margin-bottom: 25px; box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1); border-radius: 0.5rem; }

.alert-fixed { position: fixed; top: 20px; right: 20px; z-index: 1050; min-width: 300px; border-radius: 0.5rem; }

.nav-tabs .nav-link.active { background-color: #1976d2; color: white; border-color: #1976d2; }

.status-submitted { background-color: #6c757d; }

.status-verified { background-color: #ffc107; color: #333; }

.status-approved { background-color: #28a745; }

.status-rejected { background-color: #dc3545; }

.badge-status { color: white; }

.is-invalid-field { border: 1px solid #dc3545 !important; }

</style>

</head>

<body>

<div class="container-fluid">

<h1 class="text-center mb-4 text-primary"><i class="bi bi-file-earmark-text"></i> CBSE Board Registration</h1>

<!-- Notification Area -->

<div id="notification-area" class="alert-fixed" style="display: none;"></div>

<!-- Tab Navigation -->

<ul class="nav nav-tabs" id="cbseTab" role="tablist">

<li class="nav-item"><button class="nav-link active" id="list-tab" data-bs-toggle="tab" data-bs-target="#list-view" type="button"><i class="bi bi-list-task"></i> Registration List</button></li>

<li class="nav-item"><button class="nav-link" id="new-tab" data-bs-toggle="tab" data-bs-target="#new-form" type="button"><i class="bi bi-plus-circle"></i> New Registration</button></li>

<li class="nav-item"><button class="nav-link" id="audit-tab" data-bs-toggle="tab" data-bs-target="#audit" type="button"><i class="bi bi-box-arrow-down"></i> Export & Audit</button></li>

</ul>

<!-- Tab Content -->

<div class="tab-content pt-3" id="cbseTabContent">

<!-- 1. Registration List (GET /registrations) -->

<div class="tab-pane fade show active" id="list-view" role="tabpanel">

<div class="card section-card">

<div class="card-header">Manage Student Registrations (9th & 11th)</div>

<div class="card-body">

<!-- Filters -->

<form id="filterForm" class="row g-3 mb-4">

<div class="col-md-3">

<select class="form-select" id="filterClass" name="studentClass">

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

<option value="9th">9th</option>

<option value="11th">11th</option>

</select>

</div>

<div class="col-md-3">

<select class="form-select" id="filterStatus" name="status">

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

<option value="submitted">Submitted</option>

<option value="verified">Verified</option>

<option value="approved">Approved</option>

<option value="rejected">Rejected</option>

</select>

</div>

<div class="col-md-3">

<input type="text" class="form-control" id="filterYear" name="academicYear" placeholder="Academic Year (YYYY-YY)">

</div>

<div class="col-md-3">

<button type="submit" class="btn btn-primary w-100"><i class="bi bi-search"></i> Filter</button>

</div>

</form>

<!-- Registration Table -->

<div class="table-responsive">

<table class="table table-striped table-sm align-middle">

<thead>

<tr>

<th>Reg ID</th>

<th>Student</th>

<th>Class</th>

<th>Submitted At</th>

<th>Status</th>

<th>Action</th>

</tr>

</thead>

<tbody id="registrationsTableBody">

<tr><td colspan="6" class="text-center">Loading registrations...</td></tr>

</tbody>

</table>

</div>

<div id="paginationControls" class="d-flex justify-content-between mt-3">

<button class="btn btn-sm btn-outline-primary" id="prevPageBtn" disabled>Previous</button>

<span id="pageInfo" class="align-self-center small text-muted">Page 1</span>

<button class="btn btn-sm btn-outline-primary" id="nextPageBtn" disabled>Next</button>

</div>

</div>

</div>

</div>

<!-- 2. New Registration Form (POST /registrations) -->

<div class="tab-pane fade" id="new-form" role="tabpanel">

<div class="card section-card">

<div class="card-header">Submit New Student Registration Data</div>

<div class="card-body">

<form id="registrationForm" class="row g-3">

<!-- Section: Registration Details -->

<h5 class="mt-4 text-primary"><i class="bi bi-person-lines-fill"></i> Core Registration Details</h5>

<div class="col-md-4"><label for="studentId" class="form-label">Student System ID</label><input type="text" class="form-control" id="studentId" required></div>

<div class="col-md-4"><label for="studentName" class="form-label">Student Name (as per records)</label><input type="text" class="form-control" id="studentName" required></div>

<div class="col-md-2"><label for="studentClass" class="form-label">Class</label><select class="form-select" id="studentClass" required><option value="9th">9th</option><option value="11th">11th</option></select></div>

<div class="col-md-2"><label for="academicYear" class="form-label">Academic Year</label><input type="text" class="form-control" id="academicYear" placeholder="YYYY-YY" required></div>

<!-- Section: Personal Details -->

<h5 class="mt-4 text-primary"><i class="bi bi-person-fill"></i> Personal Details (CBSE Mandatory Fields)</h5>

<div class="col-md-4"><label for="fullName" class="form-label">Full Name (CBSE Format)</label><input type="text" class="form-control" id="fullName" required></div>

<div class="col-md-4"><label for="dateOfBirth" class="form-label">Date of Birth</label><input type="date" class="form-control" id="dateOfBirth" required></div>

<div class="col-md-4"><label for="gender" class="form-label">Gender</label><select class="form-select" id="gender" required><option value="Male">Male</option><option value="Female">Female</option><option value="Other">Other</option></select></div>

<div class="col-md-4"><label for="aadhaarNumber" class="form-label">Aadhaar Number (12 Digits)</label><input type="text" class="form-control" id="aadhaarNumber" minlength="12" maxlength="12" required></div>

<!-- NEW CBSE MANDATORY FIELDS -->

<div class="col-md-4">

<label for="community" class="form-label">Community/Category</label>

<select class="form-select" id="community" required>

<option value="GEN">General</option>

<option value="OBC">OBC</option>

<option value="SC">SC</option>

<option value="ST">ST</option>

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

</select>

</div>

<div class="col-md-4"><label for="udiseCode" class="form-label">UDISE Code</label><input type="text" class="form-control" id="udiseCode" required></div>

<div class="col-md-4"><label for="aparId" class="form-label">APAR ID (If Applicable)</label><input type="text" class="form-control" id="aparId"></div>

<!-- Section: Parent Details -->

<h5 class="mt-4 text-primary"><i class="bi bi-people-fill"></i> Parent Details</h5>

<div class="col-md-4"><label for="fatherName" class="form-label">Father's Name</label><input type="text" class="form-control" id="fatherName" required></div>

<div class="col-md-4"><label for="fatherPhone" class="form-label">Father's Phone</label><input type="tel" class="form-control" id="fatherPhone" placeholder="10-15 digits" required></div>

<div class="col-md-4"><label for="motherName" class="form-label">Mother's Name</label><input type="text" class="form-control" id="motherName" required></div>

<!-- Section: Submission -->

<h5 class="mt-4 text-primary"><i class="bi bi-upload"></i> Submission Settings</h5>

<div class="col-md-6">

<label for="submittedBy" class="form-label">Submission Source</label>

<select class="form-select" id="submittedBy" required><option value="admin">School Admin</option><option value="parent">Parent</option><option value="student">Student</option></select>

</div>

<div class="col-12 mt-4">

<button type="submit" class="btn btn-success btn-lg w-100" id="submitRegBtn"><i class="bi bi-check-circle"></i> Submit Registration</button>

</div>

<div class="col-12"><p class="text-danger small" id="formValidationSummary"></p></div>

</form>

</div>

</div>

</div>

<!-- 3. Export & Audit Tab -->

<div class="tab-pane fade" id="audit" role="tabpanel">

<div class="card section-card">

<div class="card-header">Data Export & Compliance Audit</div>

<div class="card-body">

<h5 class="text-primary"><i class="bi bi-file-earmark-arrow-down"></i> Data Export</h5>

<p class="text-muted">Export all registration records for final CBSE portal upload or internal review.</p>

<div class="row g-3">

<div class="col-md-4"><button class="btn btn-warning w-100"><i class="bi bi-filetype-csv"></i> Download Data (CSV)</button></div>

<div class="col-md-4"><button class="btn btn-warning w-100"><i class="bi bi-file-earmark-excel"></i> Download Data (XLSX)</button></div>

<div class="col-md-4"><button class="btn btn-warning w-100"><i class="bi bi-file-earmark-pdf"></i> Generate Printable Manifest</button></div>

</div>

<hr class="my-4">

<h5 class="text-primary"><i class="bi bi-journal-check"></i> Audit Trail (Simulated)</h5>

<p class="text-muted">Review system logs for verification, approval, and submission actions.</p>

<ul class="list-group list-group-flush small" id="auditLogList">

<li class="list-group-item">2025-10-10 14:30:00: User ID 111... triggered `CBSE\_VERIFICATION\_NEEDED` notification.</li>

<li class="list-group-item">2025-10-10 14:28:15: User ID 111... action `CBSE\_SUBMITTED` for Student ID STD12345.</li>

<li class="list-group-item">2025-10-09 09:00:00: System updated status of Reg ID 5 to 'Submitted'.</li>

</ul>

</div>

</div>

</div>

</div>

</div>

<!-- Registration Detail Modal -->

<div class="modal fade" id="detailModal" tabindex="-1">

<div class="modal-dialog modal-lg">

<div class="modal-content">

<div class="modal-header bg-primary text-white">

<h5 class="modal-title"><i class="bi bi-eye"></i> Registration Review - <span id="modalRegId"></span></h5>

<button type="button" class="btn-close btn-close-white" data-bs-dismiss="modal"></button>

</div>

<div class="modal-body">

<h6 class="text-success"><i class="bi bi-person-vcard"></i> Registration Data</h6>

<div class="row small mb-3">

<div class="col-md-4"><strong>Name:</strong> <span id="detailName"></span></div>

<div class="col-md-4"><strong>Class:</strong> <span id="detailClass"></span></div>

<div class="col-md-4"><strong>DOB:</strong> <span id="detailDob"></span></div>

<div class="col-md-4"><strong>Aadhaar:</strong> <span id="detailAadhaar"></span></div>

<div class="col-md-4"><strong>Community:</strong> <span id="detailCommunity"></span></div>

<div class="col-md-4"><strong>UDISE:</strong> <span id="detailUdise"></span></div>

<div class="col-md-4"><strong>APAR ID:</strong> <span id="detailAparId"></span></div>

<div class="col-md-4"><strong>Father:</strong> <span id="detailFather"></span></div>

<div class="col-md-4"><strong>Father Phone:</strong> <span id="detailFatherPhone"></span></div>

</div>

<h6 class="text-warning mt-4"><i class="bi bi-file-earmark-arrow-up"></i> Document Upload</h6>

<form id="documentUploadForm" data-reg-id="">

<div class="row g-2 mb-3">

<div class="col-md-4">

<select class="form-select form-select-sm" name="documentType" required>

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

<option value="birth\_certificate">Birth Certificate</option>

<option value="mark\_sheet">Previous Marksheet</option>

<option value="aadhaar\_copy">Aadhaar Copy</option>

</select>

</div>

<div class="col-md-5">

<input type="file" class="form-control form-control-sm" name="document" required>

</div>

<div class="col-md-3">

<button type="submit" class="btn btn-sm btn-primary w-100" id="uploadDocBtn">Upload</button>

</div>

</div>

</form>

<h6 class="text-warning mt-4"><i class="bi bi-file-earmark-arrow-up"></i> Document Status (Simulated Verification)</h6>

<ul class="list-group list-group-flush small mb-3" id="documentList">

<!-- Dynamic content will populate here -->

<li class="list-group-item d-flex justify-content-between align-items-center">

Aadhaar Copy

<span class="badge bg-warning text-dark">Awaiting Upload</span>

</li>

<li class="list-group-item d-flex justify-content-between align-items-center">

Previous Marksheet

<span class="badge bg-danger">Verification Pending</span>

<button class="btn btn-sm btn-outline-success ms-2 verify-doc-btn">Verify</button>

</li>

</ul>

<h6 class="text-danger"><i class="bi bi-arrow-right-square"></i> Workflow Actions</h6>

<div class="btn-group w-100" role="group">

<button type="button" class="btn btn-warning verify-action-btn" data-reg-id=""><i class="bi bi-person-check"></i> 1. Verify Details (Teacher/Staff)</button>

<button type="button" class="btn btn-success approve-action-btn" data-reg-id=""><i class="bi bi-check-circle-fill"></i> 2. Final Approval (Principal)</button>

<button type="button" class="btn btn-danger reject-action-btn" data-reg-id=""><i class="bi bi-x-circle-fill"></i> 3. Reject</button>

</div>

</div>

<div class="modal-footer">

<button type="button" class="btn btn-secondary" data-bs-dismiss="modal">Close</button>

</div>

</div>

</div>

</div>

<!-- Bootstrap Bundle with Popper -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>

<script>

const API\_BASE = '';

const MOCK\_JWT\_TOKEN = 'mock.jwt.token.for.auth';

let currentPage = 1;

const pageSize = 50;

const YEAR\_PATTERN = /^\d{4}-\d{2}$/;

const PHONE\_PATTERN = /^\+?\d{10,15}$/;

const AADHAAR\_LENGTH = 12;

/\*\* Helper function to show a temporary notification. \*/

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

const area = document.getElementById('notification-area');

area.innerHTML = `

<div class="alert alert-${type} alert-dismissible fade show" role="alert">

${message}

<button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>

</div>

`;

area.style.display = 'block';

setTimeout(() => { area.style.display = 'none'; }, 7000);

}

/\*\* Core Fetch Function with JWT Auth Header. \*/

async function apiFetch(endpoint, method = 'GET', body = null) {

try {

const options = {

method: method,

headers: {

'Content-Type': 'application/json',

'Authorization': `Bearer ${MOCK\_JWT\_TOKEN}`

},

};

if (body && method !== 'GET') {

options.body = JSON.stringify(body);

}

const response = await fetch(`${API\_BASE}${endpoint}`, options);

const data = await response.json();

if (!response.ok) {

throw new Error(data.error || `API error (${response.status})`);

}

return data;

} catch (error) {

console.error('API Fetch Error:', endpoint, error);

showNotification(error.message || 'Network error. Check console for details.', 'danger');

throw error;

}

}

// --- 1. REGISTRATION LIST LOGIC ---

function getStatusClass(status) {

switch(status) {

case 'submitted': return 'status-submitted';

case 'verified': return 'status-verified';

case 'approved': return 'status-approved';

case 'rejected': return 'status-rejected';

default: return 'bg-secondary';

}

}

async function fetchRegistrations(page = 1) {

currentPage = page;

const tableBody = document.getElementById('registrationsTableBody');

const filters = new URLSearchParams(new FormData(document.getElementById('filterForm')));

filters.set('limit', pageSize);

filters.set('offset', (page - 1) \* pageSize);

filters.forEach((value, key, parent) => {

if (!value) {

parent.delete(key);

}

});

tableBody.innerHTML = '<tr><td colspan="6" class="text-center"><div class="spinner-border spinner-border-sm"></div> Loading...</td></tr>';

try {

const data = await apiFetch(`/registrations?${filters.toString()}`);

tableBody.innerHTML = '';

if (data.data.length === 0) {

tableBody.innerHTML = '<tr><td colspan="6" class="text-center">No registrations found matching criteria.</td></tr>';

document.getElementById('nextPageBtn').disabled = true;

document.getElementById('prevPageBtn').disabled = true;

return;

}

data.data.forEach(reg => {

const statusClass = getStatusClass(reg.status);

const row = tableBody.insertRow();

const regData = reg.registrationData || {};

row.innerHTML = `

<td>${reg.id.substring(0, 8)}...</td>

<td>${regData.studentName || 'N/A'}</td>

<td>${regData.studentClass || 'N/A'}</td>

<td>${new Date(reg.submittedAt).toLocaleDateString()}</td>

<td><span class="badge badge-status ${statusClass}">${reg.status.toUpperCase()}</span></td>

<td><button class="btn btn-sm btn-info view-detail-btn" data-id="${reg.id}" data-details='${JSON.stringify(reg)}'><i class="bi bi-eye"></i> Review</button></td>

`;

});

// Pagination

document.getElementById('pageInfo').textContent = `Page ${currentPage} (Showing ${data.data.length} records)`;

document.getElementById('prevPageBtn').disabled = currentPage === 1;

document.getElementById('nextPageBtn').disabled = data.data.length < pageSize;

} catch (error) {

tableBody.innerHTML = '<tr><td colspan="6" class="text-center text-danger">Failed to load registrations.</td></tr>';

}

}

document.getElementById('filterForm').addEventListener('submit', (e) => {

e.preventDefault();

fetchRegistrations(1);

});

document.getElementById('prevPageBtn').addEventListener('click', () => {

if (currentPage > 1) fetchRegistrations(currentPage - 1);

});

document.getElementById('nextPageBtn').addEventListener('click', () => {

fetchRegistrations(currentPage + 1);

});

// --- 2. NEW REGISTRATION FORM LOGIC & VALIDATION ---

function validateRegistrationForm(form, payload) {

const errors = [];

const fields = form.querySelectorAll('input, select');

let isValid = true;

// Clear previous errors

fields.forEach(field => field.classList.remove('is-invalid-field'));

// Core Validation

if (!YEAR\_PATTERN.test(payload.academicYear)) {

errors.push("Academic Year must be in YYYY-YY format (e.g., 2024-25).");

form.academicYear.classList.add('is-invalid-field');

isValid = false;

}

if (payload.personalDetails.aadhaarNumber.length !== AADHAAR\_LENGTH) {

errors.push(`Aadhaar Number must be exactly ${AADHAAR\_LENGTH} digits.`);

form.aadhaarNumber.classList.add('is-invalid-field');

isValid = false;

}

if (!PHONE\_PATTERN.test(payload.parentDetails.fatherPhone)) {

errors.push("Father's Phone must be a valid 10-15 digit number.");

form.fatherPhone.classList.add('is-invalid-field');

isValid = false;

}

// Check all required fields (basic check)

fields.forEach(field => {

if (field.required && !field.value.trim()) {

field.classList.add('is-invalid-field');

isValid = false;

}

});

if (!isValid) {

document.getElementById('formValidationSummary').textContent = errors.length > 0 ? errors.join(' | ') : "Please fill out all required fields marked in red.";

} else {

document.getElementById('formValidationSummary').textContent = '';

}

return isValid;

}

document.getElementById('registrationForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const btn = document.getElementById('submitRegBtn');

const payload = {

studentId: form.studentId.value,

studentName: form.studentName.value,

studentClass: form.studentClass.value,

academicYear: form.academicYear.value,

submittedBy: form.submittedBy.value,

personalDetails: {

fullName: form.fullName.value,

dateOfBirth: form.dateOfBirth.value,

gender: form.gender.value,

aadhaarNumber: form.aadhaarNumber.value,

community: form.community.value,

udiseCode: form.udiseCode.value,

aparId: form.aparId.value,

},

parentDetails: {

fatherName: form.fatherName.value,

fatherPhone: form.fatherPhone.value,

motherName: form.motherName.value,

}

};

if (!validateRegistrationForm(form, payload)) {

showNotification("Please correct the highlighted errors in the form.", 'danger');

return;

}

btn.disabled = true;

btn.innerHTML = '<span class="spinner-border spinner-border-sm"></span> Submitting...';

try {

const data = await apiFetch('/registrations', 'POST', payload);

showNotification(`Registration successfully submitted! ID: ${data.registrationId}`, 'success');

form.reset();

fetchRegistrations(1);

} catch (error) {

// Handled by apiFetch

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-check-circle"></i> Submit Registration';

}

});

// --- 3. DETAIL MODAL & WORKFLOW SIMULATION ---

// Workflow Handler: Uses apiFetch to call simulated backend status update endpoints

async function handleWorkflowAction(regId, actionType) {

const btn = document.querySelector(`[data-reg-id="${regId}"].${actionType}-action-btn`);

if (!btn) return;

const originalText = btn.innerHTML;

const newStatus = actionType.replace('-action-btn', '');

btn.disabled = true;

btn.innerHTML = '<span class="spinner-border spinner-border-sm"></span> Processing...';

// Endpoint structure based on assumption: /registrations/{id}/{status\_action}

const endpoint = `/registrations/${regId}/${newStatus}`;

try {

const data = await apiFetch(endpoint, 'POST', { status: newStatus });

showNotification(`Registration ${regId.substring(0, 8)} status updated to ${newStatus.toUpperCase()}.`, 'success');

bootstrap.Modal.getInstance(document.getElementById('detailModal')).hide();

fetchRegistrations(currentPage);

} catch (error) {

// Error handled by apiFetch

} finally {

btn.disabled = false;

btn.innerHTML = originalText;

}

}

// Document Upload Handler

document.getElementById('documentUploadForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const regId = form.getAttribute('data-reg-id');

const docType = form.documentType.value;

const fileInput = form.document;

const btn = document.getElementById('uploadDocBtn');

if (!fileInput.files.length) {

showNotification("Please select a file to upload.", 'warning');

return;

}

const formData = new FormData();

formData.append('document', fileInput.files[0]);

formData.append('documentType', docType);

btn.disabled = true;

btn.innerHTML = '<span class="spinner-border spinner-border-sm"></span>';

try {

// NOTE: Using general fetch for multipart form data upload

const response = await fetch(`${API\_BASE}/applications/${regId}/upload-document`, {

method: 'POST',

headers: { 'Authorization': `Bearer ${MOCK\_JWT\_TOKEN}` },

body: formData

});

const data = await response.json();

if (response.ok && data.success) {

showNotification(`${docType} uploaded successfully!`, 'success');

// Simulate refresh of document list in modal

} else {

throw new Error(data.error || 'Upload failed.');

}

} catch (error) {

console.error("Upload error:", error);

showNotification(`Upload failed: ${error.message}`, 'danger');

} finally {

btn.disabled = false;

btn.innerHTML = 'Upload';

}

});

document.getElementById('registrationsTableBody').addEventListener('click', (e) => {

const btn = e.target.closest('.view-detail-btn');

if (btn) {

const reg = JSON.parse(btn.getAttribute('data-details'));

const regData = reg.registrationData;

const modal = new bootstrap.Modal(document.getElementById('detailModal'));

// Set Modal Data

document.getElementById('modalRegId').textContent = reg.id.substring(0, 15) + '...';

document.getElementById('detailName').textContent = regData.personalDetails.fullName;

document.getElementById('detailClass').textContent = regData.studentClass + ' (' + regData.academicYear + ')';

document.getElementById('detailDob').textContent = new Date(regData.personalDetails.dateOfBirth).toLocaleDateString();

document.getElementById('detailAadhaar').textContent = regData.personalDetails.aadhaarNumber;

// NEW PERSONAL DETAILS

document.getElementById('detailCommunity').textContent = regData.personalDetails.community || 'N/A';

document.getElementById('detailUdise').textContent = regData.personalDetails.udiseCode || 'N/A';

document.getElementById('detailAparId').textContent = regData.personalDetails.aparId || 'N/A';

document.getElementById('detailFather').textContent = regData.parentDetails.fatherName;

document.getElementById('detailFatherPhone').textContent = regData.parentDetails.fatherPhone;

// Set Registration ID for Document Upload Form

document.getElementById('documentUploadForm').setAttribute('data-reg-id', reg.id);

// Set Workflow Buttons and Permissions

document.querySelectorAll('.btn-group button').forEach(workflowBtn => {

workflowBtn.dataset.regId = reg.id;

});

document.querySelector('.verify-action-btn').disabled = reg.status !== 'submitted';

document.querySelector('.approve-action-btn').disabled = reg.status !== 'verified';

document.querySelector('.reject-action-btn').disabled = reg.status === 'rejected';

modal.show();

}

});

document.getElementById('detailModal').addEventListener('click', (e) => {

const btn = e.target.closest('.verify-action-btn') || e.target.closest('.approve-action-btn') || e.target.closest('.reject-action-btn');

if (btn) {

const regId = btn.dataset.regId;

let actionType = '';

if (btn.classList.contains('verify-action-btn')) actionType = 'verify';

else if (btn.classList.contains('approve-action-btn')) actionType = 'approve';

else if (btn.classList.contains('reject-action-btn')) actionType = 'reject';

if (actionType) {

handleWorkflowAction(regId, actionType);

}

}

});

// --- Initialization ---

document.addEventListener('DOMContentLoaded', () => {

const year = new Date().getFullYear();

document.getElementById('academicYear').value = `${year}-${String(year + 1).slice(-2)}`;

fetchRegistrations();

});

</script>

</body>

</html>

4. client/dist/modules/Documentation\_Hub.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Documentation & System Management Hub (Super Admin)</title>

<!-- Tailwind CSS -->

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

<link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css" rel="stylesheet">

<style>

body { background-color: #f8f9fa; font-family: 'Inter', sans-serif; }

.container { max-width: 1200px; margin-top: 20px; }

.tab-button.active { background-color: #3f51b5 !important; color: white !important; }

.card-header { background-color: #3f51b5; color: white; font-weight: bold; border-radius: 0.5rem 0.5rem 0 0; }

.section-card { box-shadow: 0 4px 8px rgba(0,0,0,0.05); border-radius: 0.5rem; margin-bottom: 20px; }

.code-block { background-color: #2d3748; color: #fff; padding: 10px; border-radius: 5px; font-family: monospace; overflow-x: auto; }

.warning-tag { background-color: #fef3c7; color: #b45309; padding: 2px 8px; border-radius: 4px; font-weight: 600; }

.bargain-card { border: 1px solid #dc2626; background-color: #fef2f2; }

</style>

</head>

<body>

<div class="container mx-auto">

<h1 class="text-3xl font-extrabold text-center mb-6 text-gray-800 flex items-center justify-center">

<i class="fas fa-user-shield text-red-600 mr-3"></i>Super Admin: Documentation & Integrity Hub

</h1>

<p class="text-center text-sm text-red-700 mb-6 font-semibold">This tool is restricted to Super Admin for global policy and system configuration.</p>

<!-- Notification Area -->

<div id="notification-area" class="fixed top-4 right-4 z-50"></div>

<!-- Tab Navigation -->

<div class="bg-white rounded-lg shadow-md mb-4">

<div class="flex border-b border-gray-200">

<button class="tab-button px-4 py-3 text-sm font-medium active flex-1 text-gray-700" onclick="switchTab('branding')" data-tab="branding">

<i class="fas fa-palette mr-2"></i>Branding & Compliance

</button>

<button class="tab-button px-4 py-3 text-sm font-medium flex-1 text-gray-700" onclick="switchTab('pricing')" data-tab="pricing">

<i class="fas fa-money-check-alt mr-2"></i>Global Pricing & Billing

</button>

<button class="tab-button px-4 py-3 text-sm font-medium flex-1 text-gray-700" onclick="switchTab('integrity')" data-tab="integrity">

<i class="fas fa-cogs mr-2"></i>Critical System Integrity

</button>

<button class="tab-button px-4 py-3 text-sm font-medium flex-1 text-gray-700" onclick="switchTab('reference')" data-tab="reference">

<i class="fas fa-file-alt mr-2"></i>Policy Editor

</button>

</div>

</div>

<!-- Tab Content -->

<div id="tabContent">

<!-- 1. Branding & Assets (COMPLIANCE & LOGO MANAGER) -->

<div data-tab-content="branding" class="tab-pane active">

<div class="section-card bg-white">

<div class="card-header bg-indigo-600">Compliance & Module Branding Asset Manager</div>

<form id="complianceForm" onsubmit="saveComplianceSettings(event)" class="p-6 space-y-6">

<!-- Compliance IDs -->

<h3 class="font-bold text-lg text-gray-700 mb-4 flex items-center border-b pb-2"><i class="fas fa-balance-scale mr-2 text-red-600"></i>Legal & Financial Compliance</h3>

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

<div>

<label class="block text-sm font-medium text-gray-700">MSME Registration No.</label>

<input type="text" id="msmeRegNo" name="msme\_reg\_no" placeholder="UDYAM-TN-34-XXXXXXX" class="mt-1 block w-full p-2 border rounded-md">

</div>

<div>

<label class="block text-sm font-medium text-gray-700">GST TIN (Tax ID)</label>

<input type="text" id="gstTinNo" name="gst\_tin\_no" placeholder="GSTIN-XXXXXXXXXXXX" class="mt-1 block w-full p-2 border rounded-md">

</div>

<div>

<label class="block text-sm font-medium text-gray-700">Main Platform Logo (File Upload)</label>

<input type="file" id="mainLogoFile" name="main\_logo\_file" accept="image/png, image/jpeg" class="mt-1 block w-full p-2 border rounded-md text-sm">

<p class="text-xs text-gray-500">Upload new image to replace current logo.</p>

</div>

</div>

<!-- Module Logos (Base64/Settings Keys) -->

<h3 class="font-bold text-lg text-gray-700 mb-4 flex items-center border-b pb-2 pt-4"><i class="fas fa-image mr-2 text-indigo-600"></i>Module Icon Base64 Injection</h3>

<p class="text-sm text-gray-600">The icons below are stored as Base64 strings in the database for non-breaking display in report cards/invoices.</p>

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

<!-- Injected Logo Inputs will appear here -->

</div>

<button type="submit" class="bg-green-600 text-white px-6 py-2 rounded-md hover:bg-green-700">

<i class="fas fa-save mr-2"></i>Save All Compliance & Branding

</button>

</form>

</div>

</div>

<!-- 2. Subscription & Pricing (Global Super Admin Control) -->

<div data-tab-content="pricing" class="tab-pane hidden">

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

<!-- Plan Management (Super Admin Control) -->

<div class="lg:col-span-1 section-card bg-white">

<div class="card-header bg-purple-600">Global Pricing Plans</div>

<div id="planList" class="p-4 space-y-4">

<p class="text-gray-500">Loading plans...</p>

</div>

<div class="p-4 border-t">

<button onclick="editGlobalPlans()" class="w-full bg-purple-600 text-white px-4 py-2 rounded-md hover:bg-purple-700">

<i class="fas fa-cloud-upload-alt mr-2"></i>Update Global Pricing (Super Admin)

</button>

</div>

</div>

<!-- Billing History & Negotiated Rate -->

<div class="lg:col-span-2 section-card bg-white">

<div class="card-header bg-blue-600">School Billing History & Pricing</div>

<div class="p-6">

<!-- NEGOTIATED PRICE CARD -->

<div class="p-4 bargain-card rounded-lg mb-6">

<h3 class="font-bold text-red-800 mb-2 flex items-center">

<i class="fas fa-handshake mr-2"></i>School-Specific Pricing Override

</h3>

<p class="text-sm text-gray-700 mb-3">Set the negotiated per-child/per-month rate for <span id="billingSchoolName" class="font-semibold text-red-800">Delhi Main Campus</span>.</p>

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

<div class="relative w-32">

<input type="number" id="negotiatedRate" value="10.00" step="0.01" class="w-full p-2 border border-red-400 rounded-md text-sm font-mono pl-6" min="0">

<span class="absolute left-2 top-2 text-sm font-bold text-red-600">₹</span>

</div>

<button onclick="saveNegotiatedPrice()" class="bg-red-600 text-white px-4 py-2 rounded-md text-sm hover:bg-red-700">

<i class="fas fa-save mr-2"></i>Save Negotiated Rate

</button>

</div>

<p class="text-xs mt-2 text-red-600">Global Rate: ₹10.00. Override will apply to all future invoices.</p>

</div>

<!-- END NEGOTIATED PRICE CARD -->

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

<h3 class="font-bold text-gray-700">Invoice History</h3>

<p class="text-sm text-gray-700">Total Platform Revenue: <span id="totalRevenue" class="font-bold text-green-600">₹0.00</span></p>

</div>

<div class="overflow-x-auto">

<table class="min-w-full divide-y divide-gray-200">

<thead class="bg-gray-50">

<tr>

<th class="px-6 py-3 text-left text-xs font-medium text-gray-500 uppercase tracking-wider">Invoice ID / Date</th>

<th class="px-6 py-3 text-left text-xs font-medium text-gray-500 uppercase tracking-wider">Plan / Term</th>

<th class="px-6 py-3 text-left text-xs font-medium text-gray-500 uppercase tracking-wider">Amount</th>

<th class="px-6 py-3 text-left text-xs font-medium text-gray-500 uppercase tracking-wider">Status</th>

</tr>

</thead>

<tbody id="invoiceTableBody" class="bg-white divide-y divide-gray-200">

<tr class="text-gray-500"><td colspan="4" class="px-6 py-4 text-center">No invoices loaded.</td></tr>

</tbody>

</table>

</div>

</div>

</div>

</div>

</div>

<!-- 3. System Integrity (Super Admin Only) -->

<div data-tab-content="integrity" class="tab-pane hidden">

<div class="section-card bg-white">

<div class="card-header bg-red-600 flex justify-between items-center">

Critical System Integrity Check

<button onclick="fetchSystemStatus()" class="bg-red-700 text-white px-4 py-1 rounded-full text-sm hover:bg-red-800">

<i class="fas fa-sync mr-2"></i>Run Check Now

</button>

</div>

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

<p class="text-red-700 font-semibold mb-4">This checks database security and operational readiness.</p>

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

<!-- RLS Status -->

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

<h3 class="font-bold text-lg mb-2 flex items-center text-gray-700"><i class="fas fa-shield-alt mr-2"></i>Row Level Security (RLS) Enforcement</h3>

<div id="rlsStatus" class="space-y-2 text-sm">

<p>Enforcement Status: <span class="font-bold" id="dbRlsStatus">---</span></p>

<p>Policies Count: <span class="font-bold" id="rlsPoliciesCount">---</span></p>

<p class="warning-tag mt-2">If RLS is not ON, data is NOT secure.</p>

</div>

</div>

<!-- Backup Status -->

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

<h3 class="font-bold text-lg mb-2 flex items-center text-gray-700"><i class="fas fa-database mr-2"></i>Database Maintenance</h3>

<div id="backupStatus" class="space-y-2 text-sm">

<p>Last Full Backup: <span class="font-bold" id="lastBackupTime">---</span></p>

<p>Next Audit Due: <span class="font-bold" id="nextAuditDue">---</span></p>

</div>

</div>

</div>

<!-- Database Policy Display (for reference) -->

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

<h3 class="font-bold mb-2 text-gray-700">Database Policy Reference</h3>

<p class="text-sm text-gray-600 mb-2">RLS Security Rule for Student Record Access:</p>

<div class="code-block">

CREATE POLICY "Students: Select" ON public.students FOR SELECT TO authenticated USING (school\_id = get\_user\_school\_id() AND (get\_user\_role() IN ('super\_admin', 'school\_admin', 'teacher') OR parent\_user\_profile\_id = get\_user\_id()));

</div>

</div>

</div>

</div>

</div>

<!-- 4. Policy Editor (Moved out of Branding tab) -->

<div data-tab-content="reference" class="tab-pane hidden">

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

<div class="lg:col-span-1 section-card bg-white">

<div class="card-header bg-green-600">Document Selection</div>

<div class="p-4 space-y-2">

<button class="w-full text-left p-3 bg-gray-100 rounded-md hover:bg-gray-200" onclick="viewDocument('terms\_and\_conditions')">

<i class="fas fa-gavel mr-2 text-red-600"></i>Terms & Conditions (T&C)

</button>

<button class="w-full text-left p-3 bg-gray-100 rounded-md hover:bg-gray-200" onclick="viewDocument('privacy\_policy')">

<i class="fas fa-user-secret mr-2 text-blue-600"></i>Privacy Policy

</button>

<button class="w-full text-left p-3 bg-gray-100 rounded-md hover:bg-gray-200" onclick="viewDocument('user\_manual')">

<i class="fas fa-book-open mr-2 text-purple-600"></i>System User Manual

</button>

</div>

</div>

<div class="lg:col-span-2 section-card bg-white">

<div class="card-header">Document Viewer: <span id="currentDocTitle">Welcome Guide</span></div>

<div class="p-6">

<textarea id="documentContent" rows="15" readonly class="w-full border p-3 rounded-md text-sm bg-gray-50">

Welcome to the SmartGenEduX Documentation Hub.

</textarea>

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

<p class="text-sm text-gray-600" id="docVersionInfo">Version: 1.0 | Last Updated: N/A</p>

<div class="space-x-3">

<button id="editDocBtn" onclick="openDocumentEditModal(currentEditingDocKey)" class="bg-yellow-600 text-white px-4 py-2 rounded-md hover:bg-yellow-700">

<i class="fas fa-edit mr-2"></i>Enable Edit

</button>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

<!-- Modal for Editing Document Content -->

<div id="docEditModal" class="fixed inset-0 bg-gray-900 bg-opacity-50 flex items-center justify-center z-50 hidden">

<div class="bg-white rounded-lg p-6 w-full max-w-3xl shadow-2xl">

<h3 class="text-xl font-bold mb-4">Editing: <span id="modalDocTitle"></span></h3>

<textarea id="modalDocumentContent" rows="20" class="w-full border p-3 rounded-md text-sm"></textarea>

<div class="mt-4 flex justify-end space-x-3">

<button type="button" onclick="closeDocEditModal()" class="bg-gray-300 text-gray-700 px-4 py-2 rounded-md">Cancel</button>

<button type="button" onclick="saveDocumentFromModal()" class="bg-green-600 text-white px-4 py-2 rounded-md">Save Changes</button>

</div>

</div>

</div>

<script>

const API\_BASE = '';

let currentEditingDocKey = null;

const MOCK\_SCHOOL\_ID = '00000000-0000-0000-0000-000000000001';

const MOCK\_USER\_ROLE = 'super\_admin';

// List of all modules that require a dedicated logo/icon for the Super Admin's config panel

const MODULE\_LOGOS = [

{ key: 'logo\_platform\_main', name: 'Platform Main Logo' },

{ key: 'logo\_fee\_management', name: 'Fee Management' },

{ key: 'logo\_attendance', name: 'Attendance System' },

{ key: 'logo\_whatsapp\_alert', name: 'WhatsApp/Arattai' },

{ key: 'logo\_pdf\_tools', name: 'PDF & Document Tools' },

{ key: 'logo\_admission', name: 'Admission Management' },

{ key: 'logo\_events', name: 'School Event Log' },

{ key: 'logo\_timetable', name: 'Timetable Management' },

{ key: 'logo\_substitution', name: 'Teacher Substitution' },

{ key: 'logo\_invigilation', name: 'Invigilation Duty' },

{ key: 'logo\_student\_distribution', name: 'Student Distribution' },

{ key: 'logo\_report\_tracker', name: 'Report Tracker' },

{ key: 'logo\_behavior\_tracker', name: 'Behavior Tracker' },

{ key: 'logo\_qpaper\_generator', name: 'Question Paper Gen' },

{ key: 'logo\_cbse\_registration', name: 'CBSE Registration' }

];

// --- Utility Functions ---

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

const colors = { success: 'bg-green-600', error: 'bg-red-600', info: 'bg-blue-600', warning: 'bg-yellow-600' };

const icons = { success: 'fas fa-check-circle', error: 'fas fa-exclamation-triangle', info: 'fas fa-info-circle', warning: 'fas fa-exclamation-triangle' };

const area = document.getElementById('notification-area');

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

messageDiv.className = `p-3 rounded-lg shadow-lg mb-2 text-white ${colors[type]}`;

messageDiv.innerHTML = `<i class="${icons[type]} mr-2"></i>${message}`;

area.appendChild(messageDiv);

setTimeout(() => messageDiv.remove(), 5000);

}

async function fetchAPI(endpoint, method = 'GET', body = null) {

const fullEndpoint = endpoint.replace(':schoolId', MOCK\_SCHOOL\_ID);

const options = {

method,

headers: { 'Content-Type': 'application/json' },

};

if (body && method !== 'GET') { options.body = JSON.stringify(body); }

try {

const response = await fetch(`${API\_BASE}${fullEndpoint}`, options);

const data = await response.json();

if (!response.ok) {

showNotification(data.error || `API Error: ${response.status}`, 'error');

throw new Error(data.error || 'API request failed');

}

return data;

} catch (error) {

console.error(`Fetch error on ${fullEndpoint}:`, error);

throw error;

}

}

function switchTab(tabName) {

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

btn.classList.remove('active');

});

document.querySelector(`[data-tab="${tabName}"]`).classList.add('active');

document.querySelectorAll('.tab-pane').forEach(pane => {

pane.classList.add('hidden');

});

document.querySelector(`[data-tab-content="${tabName}"]`).classList.remove('hidden');

if (tabName === 'reference') viewDocument('terms\_and\_conditions');

if (tabName === 'pricing') fetchPricingData();

if (tabName === 'integrity') fetchSystemStatus();

if (tabName === 'branding') initializeBrandingManager();

}

// --- Branding Manager Functions ---

function initializeBrandingManager() {

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

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

container.innerHTML = ''; // Clear previous inputs

// 1. Populate Compliance/Branding fields (Simulated fetch of existing settings)

fetchAPI(`/:schoolId`).then(settingsData => {

const settings = settingsData.settings || {};

document.getElementById('msmeRegNo').value = settings.msme\_reg\_no || '';

document.getElementById('gstTinNo').value = settings.gst\_tin\_no || '';

// 2. Render logo inputs based on the module list

MODULE\_LOGOS.forEach(logo => {

const existingValue = settings[logo.key] || '';

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

div.innerHTML = `

<label class="block text-sm font-medium text-gray-700">${logo.name} Logo (Base64)</label>

<textarea id="${logo.key}" data-key="${logo.key}" rows="1" placeholder="Paste Base64 data URI here..." class="mt-1 block w-full p-2 border rounded-md text-xs">${existingValue}</textarea>

`;

container.appendChild(div);

});

}).catch(() => {

// Render empty inputs if fetch fails

MODULE\_LOGOS.forEach(logo => {

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

div.innerHTML = `<label class="block text-sm font-medium text-gray-700">${logo.name} Logo (Base64)</label><textarea id="${logo.key}" data-key="${logo.key}" rows="1" placeholder="Paste Base64 data URI here..." class="mt-1 block w-full p-2 border rounded-md text-xs"></textarea>`;

container.appendChild(div);

});

});

}

async function saveComplianceSettings(event) {

event.preventDefault();

const form = event.target;

const btn = form.querySelector('button[type="submit"]');

btn.disabled = true;

const msme = form.msmeRegNo.value.trim();

const gst = form.gstTinNo.value.trim();

const payload = [

{ settingKey: 'msme\_reg\_no', settingValue: msme },

{ settingKey: 'gst\_tin\_no', settingValue: gst }

];

// Gather all logo Base64 inputs

MODULE\_LOGOS.forEach(logo => {

const input = document.getElementById(logo.key);

// NOTE: We trust the Super Admin to paste the full Base64 string.

// The backend will truncate for audit logs.

payload.push({

settingKey: logo.key,

settingValue: input.value.trim()

});

});

showNotification('Saving compliance and branding details...', 'info');

try {

// Hitting POST /:schoolId for batch update (compliance and branding assets)

const data = await fetchAPI(`/:schoolId`, 'POST', payload);

showNotification(data.message, 'success');

} catch (error) {

showNotification('Failed to save compliance settings.', 'error');

} finally {

btn.disabled = false;

}

}

// --- 1. Policy Editor Functions ---

async function viewDocument(docKey) {

currentEditingDocKey = docKey;

document.getElementById('currentDocTitle').textContent = docKey.replace(/\_/g, ' ').toUpperCase();

document.getElementById('documentContent').value = 'Loading...';

document.getElementById('documentContent').readOnly = true;

document.getElementById('editDocBtn').style.display = MOCK\_USER\_ROLE === 'super\_admin' ? 'block' : 'none';

try {

const data = await fetchAPI(`/docs/content/${docKey}`);

const doc = data.doc;

document.getElementById('documentContent').value = doc.content;

document.getElementById('docVersionInfo').textContent = `Version: ${doc.version} | Last Updated: ${new Date(doc.updated\_at).toLocaleString()}`;

} catch (error) {

document.getElementById('documentContent').value = `Error loading document: ${currentEditingDocKey}`;

}

}

function openDocumentEditModal(docKey) {

// Transfer current content to modal for editing

const content = document.getElementById('documentContent').value;

document.getElementById('modalDocTitle').textContent = docKey.replace(/\_/g, ' ').toUpperCase();

document.getElementById('modalDocumentContent').value = content;

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

document.getElementById('modalDocumentContent').focus();

showNotification('Document editing started in modal.', 'info');

}

function closeDocEditModal() {

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

showNotification('Editing session cancelled.', 'info');

}

async function saveDocumentFromModal() {

const content = document.getElementById('modalDocumentContent').value;

const modalBtn = document.querySelector('#docEditModal button[onclick="saveDocumentFromModal()"]');

modalBtn.disabled = true;

try {

const data = await fetchAPI(`/docs/content/${currentEditingDocKey}`, 'PUT', { content });

showNotification(data.message, 'success');

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

viewDocument(currentEditingDocKey); // Refresh main view with new content

} catch (error) {

// Error handled by fetchAPI

} finally {

modalBtn.disabled = false;

}

}

// --- 2. Pricing & Billing Functions ---

async function saveNegotiatedPrice() {

const rateInput = document.getElementById('negotiatedRate');

const rate = parseFloat(rateInput.value);

if (isNaN(rate) || rate < 0) {

return showNotification('Invalid price entered.', 'error');

}

rateInput.disabled = true;

showNotification(`Setting negotiated rate to ₹${rate.toFixed(2)}...`, 'info');

const payload = {

settingKey: 'academic\_negotiated\_rate',

settingValue: rate

};

try {

const data = await fetchAPI(`/config/school-pricing/${MOCK\_SCHOOL\_ID}`, 'PUT', payload);

showNotification(data.message, 'success');

document.getElementById('negotiatedRate').value = rate.toFixed(2);

} catch (error) {

showNotification('Failed to save negotiated price. Check Super Admin privileges.', 'error');

} finally {

rateInput.disabled = false;

}

}

async function fetchPricingData() {

const schoolId = MOCK\_SCHOOL\_ID;

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

try {

const [plansData, billingData] = await Promise.all([

fetchAPI('/config/plans'),

fetchAPI(`/billing/invoices/${schoolId}`)

]);

// Render Plans

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

planList.innerHTML = plansData.plans.map(p => `

<div class="p-3 ${p.isPremium ? 'bg-yellow-50 border-yellow-300' : 'bg-green-50 border-green-300'} border rounded-lg">

<span class="font-bold">${p.planName}</span> - ₹${p.monthlyPrice.toFixed(2)}/mo

<p class="text-xs text-gray-600">${p.features.join(', ')}</p>

<button onclick="editPlan('${p.id}')" class="mt-2 text-blue-600 text-xs hover:underline">Edit Plan</button>

</div>

`).join('');

// Render Invoices

document.getElementById('totalRevenue').textContent = `₹${billingData.totalRevenue.toFixed(2)}`;

invoiceTableBody.innerHTML = billingData.invoices.map(inv => `

<tr>

<td class="px-6 py-4 whitespace-nowrap text-sm text-gray-900">${inv.date} (ID: ${inv.id})</td>

<td class="px-6 py-4 whitespace-nowrap text-sm text-gray-500">${inv.plan}</td>

<td class="px-6 py-4 whitespace-nowrap text-sm font-medium text-green-600">₹${inv.amount.toFixed(2)}</td>

<td class="px-6 py-4 whitespace-nowrap text-sm">

<span class="px-2 inline-flex text-xs leading-5 font-semibold rounded-full bg-green-100 text-green-800">${inv.status}</span>

<button onclick="downloadInvoice('${inv.id}', '${inv.downloadUrl}')" class="ml-2 text-blue-500 hover:text-blue-700"><i class="fas fa-download"></i></button>

</td>

</tr>

`).join('');

} catch (error) {

invoiceTableBody.innerHTML = '<tr><td colspan="4" class="px-6 py-4 text-center text-red-600">Failed to load billing data.</td></tr>';

}

}

function editPlan(planId) {

showNotification(`Opening modal to edit plan ${planId}.`, 'info');

}

function downloadInvoice(invoiceId, url) {

showNotification(`Simulating download for Invoice ${invoiceId} from ${url}...`, 'success');

}

async function fetchSystemStatus() {

const rlsStatusDiv = document.getElementById('rlsStatus');

rlsStatusDiv.innerHTML = '<p class="text-gray-500"><i class="fas fa-spinner fa-spin"></i> Checking DB...</p>';

try {

const data = await fetchAPI('/system/status');

const integrity = data.integrityCheck;

const rlsStatus = integrity.databaseRls.toUpperCase();

document.getElementById('dbRlsStatus').textContent = rlsStatus;

document.getElementById('dbRlsStatus').className = `font-bold ${rlsStatus === 'ON' ? 'text-green-600' : 'text-red-600'}`;

document.getElementById('rlsPoliciesCount').textContent = integrity.rlsPoliciesCount;

document.getElementById('lastBackupTime').textContent = new Date(integrity.lastBackup).toLocaleString();

document.getElementById('nextAuditDue').textContent = integrity.nextAuditDue;

if (rlsStatus !== 'ON') {

showNotification('CRITICAL ALERT: RLS is OFF. Database security compromised!', 'error');

}

} catch (error) {

document.getElementById('rlsStatus').innerHTML = '<p class="text-red-600">Authentication Required / API Failed.</p>';

}

}

// --- Initialization ---

document.addEventListener('DOMContentLoaded', () => {

initializeBrandingManager();

switchTab('branding');

});

</script>

</body>

</html>

5. client/dist/modules/Exam\_seating\_planner.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Exam Seating Planner - Stratified Mixing</title>

<link href="https://cdn.jsdelivr.net/npm/tailwindcss@2.2.19/dist/tailwind.min.css" rel="stylesheet">

<link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css" rel="stylesheet">

<style>

body { font-family: 'Inter', sans-serif; }

.tab-button.active {

border-bottom: 3px solid #10b981; /\* Emerald green \*/

color: #10b981;

font-weight: 600;

}

.report-section { max-height: 500px; overflow-y: auto; }

</style>

</head>

<body class="bg-gray-50">

<div class="container mx-auto p-4 md:p-8">

<!-- Header -->

<div class="bg-white rounded-xl shadow-lg p-6 mb-8 border-t-4 border-emerald-500">

<h1 class="text-3xl font-extrabold text-gray-800 flex items-center">

<i class="fas fa-graduation-cap text-emerald-500 mr-4"></i>

Exam Seating Planner (Stratified Mixing)

</h1>

<p class="text-gray-500 mt-2">Design, generate, and export anti-cheating seating plans for exams across multiple classes and rooms.</p>

</div>

<!-- Global Notification Area -->

<div id="notification-area" class="fixed top-5 right-5 z-50"></div>

<!-- Main Content Area -->

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

<!-- Tabs -->

<div class="flex border-b mb-6 space-x-4">

<button id="tab-input" class="tab-button active pb-2 text-gray-600 hover:text-emerald-500 transition duration-150" onclick="switchTab('input')">

<i class="fas fa-edit mr-1"></i> 1. Setup & Generation

</button>

<button id="tab-reports" class="tab-button pb-2 text-gray-600 hover:text-emerald-500 transition duration-150" onclick="switchTab('reports')">

<i class="fas fa-print mr-1"></i> 2. Reports & Export

</button>

</div>

<!-- Tab Content -->

<div id="tab-content">

<!-- 1. Setup & Generation Tab -->

<div id="content-input" class="tab-content">

<p class="text-gray-700 mb-4 bg-yellow-50 p-3 rounded border border-yellow-200">

Enter classes and available rooms. The system will automatically perform \*\*stratified mixing\*\* to ensure students from the same section are separated for anti-cheating compliance.

</p>

<h3 class="text-xl font-semibold mb-3 text-gray-800">A. Classes & Sections</h3>

<div id="class-input-container" class="space-y-4 mb-6 p-4 border rounded-lg bg-gray-50">

<!-- Dynamic Class Inputs Go Here -->

<div class="class-group flex space-x-2 items-end border-b pb-4">

<div class="flex-1">

<label class="block text-sm font-medium text-gray-700">Class Name (e.g., IX, X)</label>

<input type="text" placeholder="Class Name" class="class-name w-full border border-gray-300 rounded-md shadow-sm p-2">

</div>

<div class="flex-1">

<label class="block text-sm font-medium text-gray-700">Sections (Comma Sep.)</label>

<input type="text" placeholder="A, B, C" class="class-sections w-full border border-gray-300 rounded-md shadow-sm p-2">

</div>

<button onclick="addClassGroup()" class="bg-blue-500 text-white p-2 rounded-md hover:bg-blue-600">

<i class="fas fa-plus"></i> Add Class

</button>

</div>

</div>

<button onclick="addClassGroup()" class="text-sm text-blue-600 hover:text-blue-800 mb-6"><i class="fas fa-plus-circle mr-1"></i> Add Another Class Group</button>

<h3 class="text-xl font-semibold mb-3 text-gray-800">B. Available Rooms</h3>

<div id="room-input-container" class="flex flex-wrap gap-3 mb-6 p-4 border rounded-lg bg-gray-50">

<input type="text" id="room-list" placeholder="Room X A1, Library, Terrace 1, IT Lab (Comma Separated)" class="w-full border border-gray-300 rounded-md shadow-sm p-2">

<input type="number" id="default-capacity" placeholder="Default Capacity per room (e.g., 40)" value="38" class="border border-gray-300 rounded-md shadow-sm p-2 w-full md:w-1/4">

</div>

<button onclick="generateSeatingPlan()" class="w-full md:w-1/2 mx-auto block bg-emerald-500 text-white py-3 rounded-lg font-bold text-lg hover:bg-emerald-600 transition duration-200">

<i class="fas fa-shuffle mr-2"></i> GENERATE SEATING PLAN

</button>

<div id="allocation-preview" class="mt-8"></div>

</div>

<!-- 2. Reports & Export Tab -->

<div id="content-reports" class="tab-content hidden">

<p class="text-gray-700 mb-6">Review the final distribution and generate printable reports for both the invigilator (room view) and the class teacher (student list view).</p>

<div id="report-message" class="mb-6">

<p class="text-red-500"><i class="fas fa-exclamation-circle mr-2"></i>Please generate a seating plan first in the Setup tab.</p>

</div>

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

<!-- Invigilator Report -->

<div class="card">

<h3 class="text-xl font-semibold mb-3 text-gray-800 border-b pb-2 flex items-center">

<i class="fas fa-clipboard-list mr-2 text-blue-500"></i> Invigilator's Reference

</h3>

<button onclick="generateInvigilatorReport()" id="btn-invigilator" class="w-full bg-blue-500 text-white py-2 rounded mb-4 hover:bg-blue-600" disabled>

Generate Printable View

</button>

<div id="invigilator-report-output" class="report-section border p-3 rounded bg-white">

<!-- Report content goes here -->

</div>

</div>

<!-- Class Teacher Report -->

<div class="card">

<h3 class="text-xl font-semibold mb-3 text-gray-800 border-b pb-2 flex items-center">

<i class="fas fa-user-check mr-2 text-purple-500"></i> Class Teacher's Reference

</h3>

<button onclick="generateClassTeacherReport()" id="btn-class-teacher" class="w-full bg-purple-500 text-white py-2 rounded mb-4 hover:bg-purple-600" disabled>

Generate Printable View

</button>

<div id="class-teacher-report-output" class="report-section border p-3 rounded bg-white">

<!-- Report content goes here -->

</div>

</div>

</div>

</div>

</div>

</div>

</div>

<!-- Modal for Dynamic Report Output -->

<div id="report-modal" class="hidden fixed inset-0 bg-gray-900 bg-opacity-75 z-50 flex items-center justify-center p-4">

<div class="bg-white rounded-lg shadow-2xl w-full max-w-4xl p-6">

<h2 id="modal-title" class="text-2xl font-bold mb-4 border-b pb-2"></h2>

<div id="modal-content-area" class="report-section max-h-96 overflow-y-auto mb-4 p-2 border"></div>

<button onclick="document.getElementById('report-modal').classList.add('hidden')" class="bg-gray-500 text-white py-2 px-4 rounded hover:bg-gray-600">Close</button>

<button onclick="printReport()" class="bg-emerald-500 text-white py-2 px-4 rounded ml-2 hover:bg-emerald-600">

<i class="fas fa-print mr-2"></i>Print Document

</button>

</div>

</div>

<script>

const API\_BASE = ''; // Replace with your actual API base URL if needed

let lastDistribution = null;

// --- Utility Functions ---

function showMessage(message, type) {

const colors = { success: 'bg-green-500', error: 'bg-red-500', info: 'bg-blue-500' };

const icons = { success: 'fas fa-check-circle', error: 'fas fa-exclamation-triangle', info: 'fas fa-info-circle' };

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

messageDiv.className = `p-4 rounded-lg shadow-xl text-white ${colors[type]} mb-3 transition-opacity duration-300`;

messageDiv.innerHTML = `<i class="${icons[type]} mr-2"></i>${message}`;

const area = document.getElementById('notification-area');

area.appendChild(messageDiv);

setTimeout(() => {

messageDiv.classList.add('opacity-0');

messageDiv.addEventListener('transitionend', () => messageDiv.remove());

}, 5000);

}

async function apiFetch(endpoint, method = 'GET', body = null) {

try {

const options = {

method: method,

headers: { 'Content-Type': 'application/json' },

};

if (body && method !== 'GET') {

options.body = JSON.stringify(body);

}

// Mocking authentication for demo purposes

options.headers['Authorization'] = 'Bearer mock\_jwt\_token';

const response = await fetch(`${API\_BASE}${endpoint}`, options);

const data = await response.json();

if (!response.ok || data.success === false) {

throw new Error(data.error || `API error (${response.status})`);

}

return data;

} catch (error) {

console.error('API Fetch Error:', endpoint, error);

showMessage(error.message || 'Network error occurred.', 'error');

throw error;

}

}

function switchTab(tabName) {

document.querySelectorAll('.tab-button').forEach(btn => btn.classList.remove('active'));

document.getElementById(`tab-${tabName}`).classList.add('active');

document.querySelectorAll('.tab-content').forEach(content => content.classList.add('hidden'));

document.getElementById(`content-${tabName}`).classList.remove('hidden');

if (tabName === 'reports' && !lastDistribution) {

document.getElementById('report-message').innerHTML = '<p class="text-red-500"><i class="fas fa-exclamation-circle mr-2"></i>Please generate a seating plan first in the Setup tab.</p>';

document.getElementById('btn-invigilator').disabled = true;

document.getElementById('btn-class-teacher').disabled = true;

} else if (tabName === 'reports') {

document.getElementById('report-message').innerHTML = '<p class="text-green-600"><i class="fas fa-check-circle mr-2"></i>Plan generated. Ready to create reports.</p>';

document.getElementById('btn-invigilator').disabled = false;

document.getElementById('btn-class-teacher').disabled = false;

}

}

// --- Core Application Functions ---

function addClassGroup() {

const container = document.getElementById('class-input-container');

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

newGroup.className = 'class-group flex space-x-2 items-end border-b pb-4';

newGroup.innerHTML = `

<div class="flex-1">

<label class="block text-sm font-medium text-gray-700">Class Name (e.g., IX)</label>

<input type="text" placeholder="Class Name" class="class-name w-full border border-gray-300 rounded-md shadow-sm p-2">

</div>

<div class="flex-1">

<label class="block text-sm font-medium text-gray-700">Sections (Comma Sep.)</label>

<input type="text" placeholder="A1, A2, B1 (with strength 1-38)" class="class-sections w-full border border-gray-300 rounded-md shadow-sm p-2">

</div>

<button onclick="this.parentNode.remove()" class="bg-red-500 text-white p-2 rounded-md hover:bg-red-600">

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

</button>

`;

container.appendChild(newGroup);

}

function collectInputData() {

const classGroups = document.querySelectorAll('.class-group');

const classData = [];

let isValid = true;

classGroups.forEach(group => {

const nameInput = group.querySelector('.class-name');

const sectionsInput = group.querySelector('.class-sections');

const className = nameInput.value.trim().toUpperCase();

const sections = sectionsInput.value.split(',').map(s => s.trim()).filter(s => s.length > 0);

if (className && sections.length > 0) {

// Create structured data for the backend, simulating student roll numbers 1-50

const sectionData = sections.map(secName => {

// Example: Class X, Section A1 has 50 students

return { name: secName, strength: 50, rollStart: 1, rollEnd: 50, id: `${className}\_${secName}` };

});

classData.push({ name: className, sections: sectionData });

} else if (nameInput.value || sectionsInput.value) {

isValid = false;

}

});

const roomsString = document.getElementById('room-list').value.trim();

const defaultCapacity = parseInt(document.getElementById('default-capacity').value);

if (!roomsString || isNaN(defaultCapacity) || defaultCapacity < 10) {

showMessage("Room list and valid capacity are required.", 'error');

isValid = false;

}

if (!isValid) {

showMessage("Please ensure all class names and sections are correctly entered.", 'error');

return null;

}

const rooms = roomsString.split(',').map(r => ({ name: r.trim().toUpperCase(), capacity: defaultCapacity }));

return { classData, rooms };

}

async function generateSeatingPlan() {

const inputData = collectInputData();

if (!inputData) return;

const btn = event.target;

btn.innerHTML = '<i class="fas fa-spinner fa-spin mr-2"></i>Generating... Please wait.';

btn.disabled = true;

try {

// Call the corrected backend endpoint

const data = await apiFetch('/student/distribute', 'POST', inputData);

lastDistribution = data.distribution;

displayAllocationPreview(data.distribution);

showMessage('Seating Plan generated successfully! Review the preview.', 'success');

switchTab('reports'); // Move to reports tab after success

} catch (error) {

showMessage(`Generation Failed: ${error.message}`, 'error');

} finally {

btn.innerHTML = '<i class="fas fa-shuffle mr-2"></i> GENERATE SEATING PLAN';

btn.disabled = false;

}

}

function displayAllocationPreview(distribution) {

const container = document.getElementById('allocation-preview');

let html = `

<h4 class="text-lg font-semibold mb-3 text-gray-700">Generated Allocation Summary:</h4>

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

`;

distribution.forEach(roomAssignment => {

const totalStudents = roomAssignment.assignments.reduce((sum, a) => sum + (a.rollEnd - a.rollStart + 1), 0);

// Truncate the list of sections for display

const sectionsList = roomAssignment.assignments.map(a => `${a.class} ${a.section}`).join(', ');

const displaySections = sectionsList.length > 50 ? sectionsList.substring(0, 50) + '...' : sectionsList;

html += `

<div class="bg-indigo-50 border border-indigo-200 p-4 rounded-lg shadow-sm">

<p class="font-bold text-indigo-800 text-xl">${roomAssignment.roomName}</p>

<p class="text-sm text-gray-600 mt-1">Total Seats: <span class="font-bold">${totalStudents}</span></p>

<p class="text-xs text-gray-500 mt-2">Sections: ${displaySections}</p>

<p class="text-xs text-red-500 font-medium">Mixing Ratio: Alternating</p>

</div>

`;

});

html += '</div>';

container.innerHTML = html;

}

// --- Report Generation Functions ---

function generateInvigilatorReport() {

if (!lastDistribution) return showMessage('Generate plan first.', 'error');

let html = '<div class="space-y-6">';

lastDistribution.forEach(roomAssignment => {

html += `

<div class="border-b pb-4">

<h4 class="text-2xl font-bold text-blue-800 mb-2">${roomAssignment.roomName}</h4>

<p class="text-sm text-gray-600 mb-3">Total Students: ${roomAssignment.assignments.reduce((sum, a) => sum + (a.rollEnd - a.rollStart + 1), 0)}</p>

<ul class="list-disc pl-5 space-y-1">

`;

roomAssignment.assignments.forEach(assignment => {

html += `<li class="text-gray-700 font-mono text-sm">

${assignment.class} ${assignment.section} [${assignment.rollStart} - ${assignment.rollEnd}]

</li>`;

});

html += '</ul></div>';

});

html += '</div>';

document.getElementById('invigilator-report-output').innerHTML = html;

// Show in modal for printing

document.getElementById('modal-title').textContent = "Invigilator's Reference (Print View)";

document.getElementById('modal-content-area').innerHTML = html;

document.getElementById('report-modal').classList.remove('hidden');

showMessage("Invigilator report generated.", 'success');

}

function generateClassTeacherReport() {

if (!lastDistribution) return showMessage('Generate plan first.', 'error');

// Structure: { 'Class X A1': [{ room: 'X A1', rollRange: '1-19' }, ...] }

const teacherReport = {};

lastDistribution.forEach(roomAssignment => {

roomAssignment.assignments.forEach(assignment => {

const classKey = `${assignment.class} ${assignment.section}`;

if (!teacherReport[classKey]) {

teacherReport[classKey] = [];

}

teacherReport[classKey].push({

room: roomAssignment.roomName,

rollRange: `${assignment.rollStart} - ${assignment.rollEnd}`

});

});

});

let html = '<div class="space-y-6">';

Object.entries(teacherReport).sort().forEach(([classKey, locations]) => {

html += `

<div class="border-b pb-4">

<h4 class="text-2xl font-bold text-purple-800 mb-3">${classKey}</h4>

<ul class="list-disc pl-5 space-y-1">

`;

locations.forEach(loc => {

html += `<li class="text-gray-700 text-base">

Roll Numbers <span class="font-semibold">${loc.rollRange}</span> must report to <span class="font-mono text-lg text-red-600">${loc.room}</span>.

</li>`;

});

html += '</ul></div>';

});

html += '</div>';

document.getElementById('class-teacher-report-output').innerHTML = html;

// Show in modal for printing

document.getElementById('modal-title').textContent = "Class Teacher's Reference (Print View)";

document.getElementById('modal-content-area').innerHTML = html;

document.getElementById('report-modal').classList.remove('hidden');

showMessage("Class Teacher report generated.", 'success');

}

function printReport() {

const printContent = document.getElementById('modal-content-area').innerHTML;

const title = document.getElementById('modal-title').textContent;

const originalBody = document.body.innerHTML;

const printWindow = window.open('', '', 'height=800,width=1200');

printWindow.document.write('<html><head><title>' + title + '</title>');

printWindow.document.write('<style>body { font-family: sans-serif; padding: 20px; } .text-red-600 { color: #dc2626; } .text-blue-800 { color: #1e40af; } .text-purple-800 { color: #5b21b6; } .font-mono { font-family: monospace; }</style>');

printWindow.document.write('</head><body>');

printWindow.document.write(`<h1>${title}</h1><hr class="mb-6"/>`);

printWindow.document.write(printContent);

printWindow.document.write('</body></html>');

printWindow.document.close();

printWindow.print();

}

// --- Initialization ---

document.addEventListener('DOMContentLoaded', () => {

// Add initial class input row

addClassGroup();

// Set mock initial classes

document.querySelector('.class-name').value = 'X';

document.querySelector('.class-sections').value = 'A1, A2, A3, A4, A5, A6, A7, A8, A9, A10, A11';

addClassGroup();

document.querySelector('.class-group:last-child .class-name').value = 'IX';

document.querySelector('.class-group:last-child .class-sections').value = 'A1, A2, A3, A4, A5, A6, A7, A8, A9, A10, A11';

document.getElementById('room-list').value = 'ROOM X A1, ROOM X A2, ROOM X A3, ROOM X A4, ROOM X A5, ROOM X A6, ROOM X A7, ROOM X A8, ROOM X A9, ROOM X A10, ROOM X A11, ROOM IX A1, ROOM IX A2, ROOM IX A3, ROOM IX A4, ROOM IX A5, ROOM IX A6, ROOM IX A7, ROOM IX A8, ROOM IX A9, ROOM IX A10, ROOM IX A11, LIBRARY, TERRACE 1, TERRACE 2, TERRACE 3, IT LAB';

switchTab('input');

});

</script>

</body>

</html>

6. client/dist/modules/Library\_Management.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>📚 Library Management System (LMS)</title>

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css">

<style>

body { background-color: #f4f7f6; }

.container-fluid { max-width: 1200px; margin-top: 20px; }

.card-header { background-color: #007bff; color: white; font-weight: bold; }

.section-card { margin-bottom: 25px; box-shadow: 0 4px 8px rgba(0,0,0,0.05); }

.alert-fixed {

position: fixed; top: 20px; right: 20px; z-index: 1050; min-width: 300px;

}

.overdue-row { background-color: #f8d7da; } /\* Red for overdue \*/

</style>

</head>

<body>

<div class="container-fluid">

<h1 class="text-center mb-4 text-primary">Library Compliance & Fines Module</h1>

<div id="notification-area" class="alert-fixed" style="display: none;"></div>

<ul class="nav nav-tabs" id="lmsTab" role="tablist">

<li class="nav-item" role="presentation">

<button class="nav-link active" id="inventory-tab" data-bs-toggle="tab" data-bs-target="#inventory" type="button" role="tab">📖 Inventory & Loan</button>

</li>

<li class="nav-item" role="presentation">

<button class="nav-link" id="loans-tab" data-bs-toggle="tab" data-bs-target="#loans" type="button" role="tab">⏳ Active Loans & Overdue</button>

</li>

<li class="nav-item" role="presentation">

<button class="nav-link" id="config-tab" data-bs-toggle="tab" data-bs-target="#config" type="button" role="tab">⚙️ Configuration</button>

</li>

<li class="nav-item" role="presentation">

<button class="nav-link" id="oversight-tab" data-bs-toggle="tab" data-bs-target="#oversight" type="button" role="tab">💰 Oversight (Principal)</button>

</li>

</ul>

<div class="tab-content pt-3" id="lmsTabContent">

<div class="tab-pane fade show active" id="inventory" role="tabpanel">

<div class="card section-card">

<div class="card-header">Book Inventory & Check-Out</div>

<div class="card-body">

<div class="row mb-3">

<div class="col-md-6">

<input type="text" class="form-control" id="inventorySearch" placeholder="Search by Title or Author...">

</div>

<div class="col-md-6 text-end">

<button class="btn btn-primary" id="searchInventoryBtn"><i class="bi bi-search"></i> Search</button>

<button class="btn btn-secondary" id="nextPageBtn" disabled>Next Page</button>

</div>

</div>

<div class="table-responsive">

<table class="table table-striped table-sm">

<thead>

<tr>

<th>Title</th>

<th>Author</th>

<th>ISBN</th>

<th>Available</th>

<th>Total</th>

<th>Action</th>

</tr>

</thead>

<tbody id="inventoryTableBody">

<tr><td colspan="6" class="text-center">Loading inventory...</td></tr>

</tbody>

</table>

</div>

<h5 class="mt-4">Check-Out Book</h5>

<form id="loanForm" class="row g-3">

<div class="col-md-6">

<label for="loanBookId" class="form-label">Book ID (from table)</label>

<input type="text" class="form-control" id="loanBookId" placeholder="e.g., 550e8400..." required readonly>

</div>

<div class="col-md-6">

<label for="loanStudentId" class="form-label">Student Profile ID</label>

<input type="text" class="form-control" id="loanStudentId" placeholder="e.g., 11111111..." required>

</div>

<div class="col-md-6">

<label for="loanDueDate" class="form-label">Due Date</label>

<input type="date" class="form-control" id="loanDueDate" required>

<div class="form-text text-danger" id="loanLimitInfo">Max loan days: 30</div>

</div>

<div class="col-12">

<button type="submit" class="btn btn-success" id="checkoutBtn"><i class="bi bi-box-arrow-up"></i> Check-Out Book</button>

</div>

</form>

</div>

</div>

</div>

<div class="tab-pane fade" id="loans" role="tabpanel">

<div class="card section-card">

<div class="card-header">Active Loans & Returns</div>

<div class="card-body">

<div class="row mb-3 align-items-center">

<div class="col-md-6">

<input type="text" class="form-control" id="loansSearch" placeholder="Search by Book/Student...">

</div>

<div class="col-md-3">

<div class="form-check form-switch">

<input class="form-check-input" type="checkbox" id="showOverdueOnly">

<label class="form-check-label" for="showOverdueOnly">Show Overdue Only</label>

</div>

</div>

<div class="col-md-3 text-end">

<button class="btn btn-primary" id="searchLoansBtn"><i class="bi bi-arrow-clockwise"></i> Refresh List</button>

</div>

</div>

<div class="table-responsive">

<table class="table table-sm table-hover">

<thead>

<tr>

<th>Loan ID</th>

<th>Title</th>

<th>Student</th>

<th>Due Date</th>

<th>Fines Incurred</th>

<th>Action</th>

</tr>

</thead>

<tbody id="activeLoansTableBody">

<tr><td colspan="6" class="text-center">Loading active loans...</td></tr>

</tbody>

</table>

</div>

</div>

</div>

</div>

<div class="tab-pane fade" id="config" role="tabpanel">

<div class="card section-card">

<div class="card-header">Library System Configuration (Manager/Admin Only)</div>

<div class="card-body">

<form id="configForm">

<div class="row g-3">

<div class="col-md-6">

<label for="maxLoanDays" class="form-label">Max Loan Days (7-90)</label>

<input type="number" class="form-control" id="maxLoanDays" min="7" max="90" required>

<div class="form-text">Maximum duration a book can be loaned.</div>

</div>

<div class="col-md-6">

<label for="finePerDay" class="form-label">Fine Per Day (₹)</label>

<input type="number" class="form-control" id="finePerDay" min="0" max="50" step="0.01" required>

<div class="form-text">Overdue fine rate (e.g., 5.00 INR).</div>

</div>

<div class="col-md-6">

<label for="maxConcurrentLoans" class="form-label">Max Concurrent Loans (1-10)</label>

<input type="number" class="form-control" id="maxConcurrentLoans" min="1" max="10" required>

<div class="form-text">Maximum books a student can hold at once.</div>

</div>

<div class="col-md-6">

<label for="overdueNotificationDays" class="form-label">Notification Reminder Days (1-14)</label>

<input type="number" class="form-control" id="overdueNotificationDays" min="1" max="14" required>

<div class="form-text">Days \*before\* due date to send an initial reminder. (Arattai/WhatsApp Integration)</div>

</div>

<div class="col-12 mt-4">

<button type="submit" class="btn btn-success" id="saveConfigBtn"><i class="bi bi-save"></i> Save Configuration</button>

</div>

</div>

</form>

</div>

</div>

</div>

<div class="tab-pane fade" id="oversight" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-danger">Principal Oversight Dashboard</div>

<div class="card-body">

<p class="alert alert-warning"><i class="bi bi-lock-fill"></i> Access restricted to Principal/Super Admin roles only.</p>

<div class="row text-center" id="oversightStats">

<div class="col-md-4">

<div class="p-3 border rounded bg-light">

<h4 class="text-primary" id="statActiveLoans">...</h4>

<p class="text-muted">Total Active Loans</p>

</div>

</div>

<div class="col-md-4">

<div class="p-3 border rounded bg-light">

<h4 class="text-danger" id="statOverdue">...</h4>

<p class="text-muted">Overdue Books</p>

</div>

</div>

<div class="col-md-4">

<div class="p-3 border rounded bg-light">

<h4 class="text-success" id="statAccruedFines">...</h4>

<p class="text-muted">Total Accrued Fines (Unpaid)</p>

</div>

</div>

</div>

<button class="btn btn-sm btn-info mt-3" id="refreshOversightBtn"><i class="bi bi-arrow-clockwise"></i> Refresh Stats</button>

</div>

</div>

</div>

</div>

</div>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>

<script>

const API\_BASE = '';

let currentOffset = 0;

const limit = 20;

let maxLoanDays = 30; // Default or fetched from config

/\*\* Helper function to show a temporary notification. \*/

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

const area = document.getElementById('notification-area');

area.innerHTML = `

<div class="alert alert-${type} alert-dismissible fade show" role="alert">

${message}

<button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>

</div>

`;

area.style.display = 'block';

setTimeout(() => {

area.style.display = 'none';

}, 7000);

}

// --- Core Fetch Function ---

async function apiFetch(endpoint, method = 'GET', body = null) {

try {

const options = {

method: method,

headers: { 'Content-Type': 'application/json' },

};

if (body && method !== 'GET') {

options.body = JSON.stringify(body);

}

const response = await fetch(`${API\_BASE}${endpoint}`, options);

const data = await response.json();

if (!response.ok) {

throw new Error(data.error || `API error (${response.status})`);

}

return data;

} catch (error) {

console.error('API Fetch Error:', endpoint, error);

showNotification(error.message || 'Network error.', 'danger');

throw error; // Re-throw to be caught by specific handler

}

}

// --- 1. INVENTORY & LOAN LOGIC ---

async function fetchInventory(resetOffset = true) {

if (resetOffset) currentOffset = 0;

const search = document.getElementById('inventorySearch').value;

const tableBody = document.getElementById('inventoryTableBody');

tableBody.innerHTML = '<tr><td colspan="6" class="text-center"><div class="spinner-border spinner-border-sm"></div> Loading...</td></tr>';

try {

const data = await apiFetch(`/books?limit=${limit}&offset=${currentOffset}&search=${search}`);

tableBody.innerHTML = '';

if (data.books.length === 0) {

tableBody.innerHTML = '<tr><td colspan="6" class="text-center">No books found matching the criteria.</td></tr>';

}

data.books.forEach(book => {

const row = tableBody.insertRow();

row.innerHTML = `

<td>${book.title}</td>

<td>${book.author}</td>

<td>${book.isbn}</td>

<td class="${book.available\_copies <= 2 ? 'text-danger fw-bold' : ''}">${book.available\_copies}</td>

<td>${book.total\_copies}</td>

<td><button class="btn btn-sm btn-info select-book-btn" data-book-id="${book.id}" ${book.available\_copies <= 0 ? 'disabled' : ''}>Check Out</button></td>

`;

});

document.getElementById('nextPageBtn').disabled = (currentOffset + limit) >= data.totalRecords;

} catch {

tableBody.innerHTML = '<tr><td colspan="6" class="text-center text-danger">Failed to load inventory.</td></tr>';

}

}

// Handle book selection for loan form

document.getElementById('inventoryTableBody').addEventListener('click', (e) => {

if (e.target.classList.contains('select-book-btn')) {

const bookId = e.target.getAttribute('data-book-id');

document.getElementById('loanBookId').value = bookId;

showNotification(`Ready to check out Book ID: ${bookId}`, 'info');

}

});

// Handle Loan Submission

document.getElementById('loanForm').addEventListener('submit', async (e) => {

e.preventDefault();

const btn = document.getElementById('checkoutBtn');

btn.disabled = true;

const payload = {

bookId: document.getElementById('loanBookId').value,

studentId: document.getElementById('loanStudentId').value,

dueDate: document.getElementById('loanDueDate').value

};

try {

const data = await apiFetch('/loan', 'POST', payload);

showNotification(`Success! ${data.message} Loan ID: ${data.loanId}`, 'success');

// Clear form and refresh lists

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

fetchInventory(false);

fetchActiveLoans();

} catch {

// Error handled by apiFetch

} finally {

btn.disabled = false;

}

});

document.getElementById('searchInventoryBtn').addEventListener('click', () => fetchInventory(true));

document.getElementById('nextPageBtn').addEventListener('click', () => {

currentOffset += limit;

fetchInventory(false);

});

// --- 2. ACTIVE LOANS & RETURN LOGIC ---

async function fetchActiveLoans() {

const search = document.getElementById('loansSearch').value;

const overdue = document.getElementById('showOverdueOnly').checked;

const tableBody = document.getElementById('activeLoansTableBody');

tableBody.innerHTML = '<tr><td colspan="6" class="text-center"><div class="spinner-border spinner-border-sm"></div> Loading...</td></tr>';

try {

const data = await apiFetch(`/loans/current?limit=${limit}&offset=0&search=${search}&overdue=${overdue}`);

tableBody.innerHTML = '';

if (data.activeLoans.length === 0) {

tableBody.innerHTML = '<tr><td colspan="6" class="text-center">No active loans found.</td></tr>';

}

data.activeLoans.forEach(loan => {

const row = tableBody.insertRow();

const isOverdue = new Date(loan.due\_date) < new Date();

if (isOverdue) row.classList.add('overdue-row');

row.innerHTML = `

<td>${loan.id.substring(0, 8)}...</td>

<td>${loan.title}</td>

<td>${loan.student\_name} (${loan.class\_id || 'N/A'})</td>

<td class="${isOverdue ? 'text-danger fw-bold' : ''}">${new Date(loan.due\_date).toLocaleDateString()}</td>

<td>${loan.fines\_incurred ? `₹${loan.fines\_incurred.toFixed(2)}` : 'N/A'}</td>

<td><button class="btn btn-sm btn-danger return-book-btn" data-loan-id="${loan.id}"><i class="bi bi-box-arrow-in-down"></i> Return</button></td>

`;

});

} catch {

tableBody.innerHTML = '<tr><td colspan="6" class="text-center text-danger">Failed to load active loans.</td></tr>';

}

}

// Handle Return Action

document.getElementById('activeLoansTableBody').addEventListener('click', async (e) => {

if (e.target.classList.contains('return-book-btn')) {

const btn = e.target;

const loanId = btn.getAttribute('data-loan-id');

btn.disabled = true;

btn.textContent = 'Processing...';

try {

const data = await apiFetch(`/return/${loanId}`, 'POST', {});

let msg = `${data.message} Loan: ${loanId.substring(0, 8)}...`;

if (data.fineCalculated > 0) {

msg += ` \*\*Fine incurred: ₹${data.fineCalculated.toFixed(2)}\*\* for ${data.daysOverdue} days overdue.`;

showNotification(msg, 'warning');

} else {

showNotification(msg, 'success');

}

fetchActiveLoans();

fetchInventory(false);

} catch {

// Error handled by apiFetch

btn.disabled = false;

btn.textContent = 'Return';

}

}

});

document.getElementById('searchLoansBtn').addEventListener('click', fetchActiveLoans);

document.getElementById('showOverdueOnly').addEventListener('change', fetchActiveLoans);

// --- 3. CONFIGURATION LOGIC ---

async function fetchConfig() {

try {

const data = await apiFetch('/config');

const settings = data.settings;

document.getElementById('maxLoanDays').value = settings.max\_loan\_days;

document.getElementById('finePerDay').value = settings.fine\_per\_day;

document.getElementById('overdueNotificationDays').value = settings.overdue\_notification\_days;

document.getElementById('maxConcurrentLoans').value = settings.max\_concurrent\_loans;

// Update client-side variable for loan form

maxLoanDays = settings.max\_loan\_days;

document.getElementById('loanLimitInfo').textContent = `Max loan days: ${maxLoanDays}`;

} catch (error) {

// If 403 (Auth denied) or 500 (DB error), config form will show default/empty.

// We keep the form as is, assuming the Manager/Admin will know the correct values.

}

}

document.getElementById('configForm').addEventListener('submit', async (e) => {

e.preventDefault();

const btn = document.getElementById('saveConfigBtn');

btn.disabled = true;

const payload = {

maxLoanDays: parseInt(document.getElementById('maxLoanDays').value),

finePerDay: parseFloat(document.getElementById('finePerDay').value),

overdueNotificationDays: parseInt(document.getElementById('overdueNotificationDays').value),

maxConcurrentLoans: parseInt(document.getElementById('maxConcurrentLoans').value)

};

try {

const data = await apiFetch('/config', 'PUT', payload);

showNotification(data.message, 'success');

fetchConfig(); // Re-fetch to confirm update

} catch {

// Error handled by apiFetch

} finally {

btn.disabled = false;

}

});

// --- 4. PRINCIPAL OVERSIGHT LOGIC ---

async function fetchOversightStats() {

document.getElementById('statActiveLoans').textContent = '...';

document.getElementById('statOverdue').textContent = '...';

document.getElementById('statAccruedFines').textContent = '...';

try {

const data = await apiFetch('/stats/oversight');

const stats = data.dashboard;

document.getElementById('statActiveLoans').textContent = stats.totalActiveLoans.toLocaleString();

document.getElementById('statOverdue').textContent = stats.totalOverdue.toLocaleString();

document.getElementById('statAccruedFines').textContent = `₹${stats.totalAccruedFines.toFixed(2).toLocaleString()}`;

showNotification('Principal oversight stats refreshed.', 'info');

} catch (error) {

// Handle 403 (Access Denied) explicitly

if (error.message && error.message.includes('403')) {

document.getElementById('oversightStats').innerHTML = '<div class="alert alert-danger">Access denied. Only Principal/Super Admin can view these statistics.</div>';

} else {

document.getElementById('oversightStats').innerHTML = '<div class="alert alert-danger">Failed to load statistics.</div>';

}

}

}

document.getElementById('refreshOversightBtn').addEventListener('click', fetchOversightStats);

// --- Initialization ---

document.addEventListener('DOMContentLoaded', () => {

fetchConfig();

fetchInventory(true);

fetchActiveLoans();

// Setup tab switch to load oversight on demand

document.getElementById('oversight-tab').addEventListener('shown.bs.tab', () => {

fetchOversightStats();

});

});

</script>

</body>

</html>

7. client/dist/modules/transportation.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>🚌 Transportation Management System</title>

<!-- Bootstrap 5 CDN -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css">

<style>

body { background-color: #f8f9fa; font-family: 'Inter', sans-serif; }

.container-fluid { max-width: 1200px; margin-top: 20px; }

.card-header { background-color: #4CAF50; color: white; font-weight: bold; }

.section-card { margin-bottom: 25px; box-shadow: 0 4px 8px rgba(0,0,0,0.05); border-radius: 0.5rem; }

.alert-fixed {

position: fixed; top: 20px; right: 20px; z-index: 1050; min-width: 300px; border-radius: 0.5rem;

}

.route-stop {

padding: 5px 10px;

margin-right: 5px;

border-radius: 5px;

background-color: #e9ecef;

font-size: 0.85rem;

}

.nav-tabs .nav-link.active {

background-color: #4CAF50;

color: white;

border-color: #4CAF50;

}

.gps-dot {

width: 10px;

height: 10px;

background-color: #17a2b8;

border-radius: 50%;

display: inline-block;

margin-right: 5px;

animation: pulse 1.5s infinite;

}

@keyframes pulse {

0% { box-shadow: 0 0 0 0 rgba(23, 162, 184, 0.7); }

70% { box-shadow: 0 0 0 10px rgba(23, 162, 184, 0); }

100% { box-shadow: 0 0 0 0 rgba(23, 162, 184, 0); }

}

</style>

</head>

<body>

<div class="container-fluid">

<h1 class="text-center mb-4 text-success"><i class="bi bi-bus-fill"></i> Transport Operations Dashboard</h1>

<!-- Notification Area -->

<div id="notification-area" class="alert-fixed" style="display: none;"></div>

<!-- Tab Navigation -->

<ul class="nav nav-tabs" id="transportTab" role="tablist">

<li class="nav-item" role="presentation">

<button class="nav-link active" id="routes-tab" data-bs-toggle="tab" data-bs-target="#routes-operations" type="button" role="tab"><i class="bi bi-geo-alt-fill"></i> Operations & Assets</button>

</li>

<li class="nav-item" role="presentation">

<button class="nav-link" id="assignment-attendance-tab" data-bs-toggle="tab" data-bs-target="#assignment-attendance" type="button" role="tab"><i class="bi bi-person-check"></i> Assignment & Attendance</button>

</li>

<li class="nav-item" role="presentation">

<button class="nav-link" id="billing-tab" data-bs-toggle="tab" data-bs-target="#billing" type="button" role="tab"><i class="bi bi-receipt"></i> Billing & Invoicing</button>

</li>

<li class="nav-item" role="presentation">

<button class="nav-link" id="config-tab" data-bs-toggle="tab" data-bs-target="#config" type="button" role="tab"><i class="bi bi-sliders"></i> Configuration</button>

</li>

<li class="nav-item" role="presentation">

<button class="nav-link" id="oversight-tab" data-bs-toggle="tab" data-bs-target="#oversight" type="button" role="tab"><i class="bi bi-currency-dollar"></i> Financial Oversight</button>

</li>

</ul>

<!-- Tab Content -->

<div class="tab-content pt-3" id="transportTabContent">

<!-- 1. Operations & Assets Tab (Routes CRUD, Vehicle CRUD, Stop CRUD, Live GPS) -->

<div class="tab-pane fade show active" id="routes-operations" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-success">Route Operations & Asset Management</div>

<div class="card-body">

<!-- Route Listing -->

<h5 class="mb-3"><i class="bi bi-list-columns"></i> Active Routes List</h5>

<div class="table-responsive mb-4">

<table class="table table-striped table-sm">

<thead>

<tr>

<th>Route No.</th>

<th>Name</th>

<th>Distance (km)</th>

<th>Vehicle</th>

<th>Driver</th>

<th>Actions</th>

</tr>

</thead>

<tbody id="routesTableBody">

<tr><td colspan="6" class="text-center">Loading routes...</td></tr>

</tbody>

</table>

</div>

<!-- Live Tracking & Manifest Export -->

<div class="row mb-4">

<div class="col-md-6">

<h5 class="text-info"><span class="gps-dot"></span> Live Vehicle Locations</h5>

<div id="liveLocationData" class="p-3 border rounded bg-light" style="max-height: 200px; overflow-y: auto;">

<p class="text-muted">Loading GPS data...</p>

</div>

<button class="btn btn-sm btn-info mt-2" onclick="fetchLiveLocations()"><i class="bi bi-broadcast"></i> Refresh GPS</button>

</div>

<div class="col-md-6">

<h5>Manifest Export</h5>

<p class="text-muted">Select a route above and click Export to generate the printable student list.</p>

<select class="form-select mb-2" id="manifestRouteId"></select>

<button class="btn btn-warning" id="exportManifestBtn" disabled><i class="bi bi-file-earmark-arrow-down"></i> Export Manifest (CSV)</button>

</div>

</div>

<hr class="my-4">

<!-- CRUD Forms (Accordions for cleaner view) -->

<div class="accordion" id="assetCrudAccordion">

<!-- Route Creation Form (POST /route) -->

<div class="accordion-item">

<h2 class="accordion-header"><button class="accordion-button collapsed bg-light" type="button" data-bs-toggle="collapse" data-bs-target="#collapseRoute" aria-expanded="false">

<i class="bi bi-plus-circle me-2"></i> Add New Route

</button></h2>

<div id="collapseRoute" class="accordion-collapse collapse" data-bs-parent="#assetCrudAccordion">

<div class="accordion-body">

<form id="newRouteForm" class="row g-3">

<div class="col-md-4"><label for="routeName" class="form-label">Route Name</label><input type="text" class="form-control" id="newRouteName" name="name" required></div>

<div class="col-md-4"><label for="routeNumber" class="form-label">Route Number</label><input type="text" class="form-control" id="routeNumber" name="route\_number" required></div>

<div class="col-md-4"><label for="routeDistance" class="form-label">Distance (km)</label><input type="number" class="form-control" id="routeDistance" name="route\_distance\_km" min="0" required></div>

<div class="col-md-6"><label for="startPoint" class="form-label">Start Point</label><input type="text" class="form-control" id="startPoint" name="start\_point" required></div>

<div class="col-md-6"><label for="endPoint" class="form-label">End Point</label><input type="text" class="form-control" id="endPoint" name="end\_point" required></div>

<div class="col-12"><button type="submit" class="btn btn-success" id="createRouteBtn">Create Route</button></div>

</form>

</div>

</div>

</div>

<!-- Vehicle CRUD (POST /vehicle, PUT/DELETE /vehicle/:id) -->

<div class="accordion-item">

<h2 class="accordion-header"><button class="accordion-button collapsed bg-light" type="button" data-bs-toggle="collapse" data-bs-target="#collapseVehicle" aria-expanded="false">

<i class="bi bi-truck me-2"></i> Vehicle Management (CRUD)

</button></h2>

<div id="collapseVehicle" class="accordion-collapse collapse" data-bs-parent="#assetCrudAccordion">

<div class="accordion-body">

<h6 class="mb-3">Add/Update Vehicle</h6>

<form id="vehicleCrudForm" class="row g-3 mb-4">

<input type="hidden" id="vehicleIdToEdit">

<div class="col-md-4"><label for="vehicleNumber" class="form-label">Vehicle Number</label><input type="text" class="form-control" id="vehicleNumber" name="vehicle\_number" required></div>

<div class="col-md-4"><label for="vehicleCapacity" class="form-label">Capacity</label><input type="number" class="form-control" id="vehicleCapacity" name="capacity" min="1" required></div>

<div class="col-md-4"><label for="vehicleDriverName" class="form-label">Driver Name</label><input type="text" class="form-control" id="vehicleDriverName" name="driver\_name" required></div>

<div class="col-md-6"><label for="vehicleDriverPhone" class="form-label">Driver Phone</label><input type="text" class="form-control" id="vehicleDriverPhone" name="driver\_phone" required></div>

<div class="col-md-6"><label for="vehicleRouteId" class="form-label">Assigned Route ID (Optional)</label><input type="text" class="form-control" id="vehicleRouteId" name="route\_id"></div>

<div class="col-12"><button type="submit" class="btn btn-primary" id="saveVehicleBtn">Add New Vehicle</button></div>

</form>

<h6 class="mt-4">Existing Vehicles</h6>

<div class="table-responsive">

<table class="table table-sm table-striped">

<thead><tr><th>Number</th><th>Capacity</th><th>Driver</th><th>Route ID</th><th>Actions</th></tr></thead>

<tbody id="vehicleTableBody"><tr><td colspan="5" class="text-center">No vehicles loaded.</td></tr></tbody>

</table>

</div>

</div>

</div>

</div>

<!-- Stop CRUD (POST /stop, PUT/DELETE /stop/:id) -->

<div class="accordion-item">

<h2 class="accordion-header"><button class="accordion-button collapsed bg-light" type="button" data-bs-toggle="collapse" data-bs-target="#collapseStop" aria-expanded="false">

<i class="bi bi-signpost-split me-2"></i> Stop Management (CRUD)

</button></h2>

<div id="collapseStop" class="accordion-collapse collapse" data-bs-parent="#assetCrudAccordion">

<div class="accordion-body">

<h6 class="mb-3">Add/Update Stop</h6>

<form id="stopCrudForm" class="row g-3 mb-4">

<input type="hidden" id="stopIdToEdit">

<div class="col-md-4"><label for="stopName" class="form-label">Stop Name</label><input type="text" class="form-control" id="stopName" name="stop\_name" required></div>

<div class="col-md-4"><label for="stopRouteId" class="form-label">Route ID</label><input type="text" class="form-control" id="stopRouteId" name="route\_id" required></div>

<div class="col-md-4"><label for="stopLatitude" class="form-label">Latitude</label><input type="number" class="form-control" id="stopLatitude" name="latitude" step="any" required></div>

<div class="col-md-4"><label for="stopLongitude" class="form-label">Longitude</label><input type="number" class="form-control" id="stopLongitude" name="longitude" step="any" required></div>

<div class="col-12"><button type="submit" class="btn btn-primary" id="saveStopBtn">Add New Stop</button></div>

</form>

<h6 class="mt-4">Existing Stops</h6>

<div class="table-responsive">

<table class="table table-sm table-striped">

<thead><tr><th>Name</th><th>Route ID</th><th>Lat/Lng</th><th>Actions</th></tr></thead>

<tbody id="stopTableBody"><tr><td colspan="4" class="text-center">No stops loaded.</td></tr></tbody>

</table>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

<!-- 2. Assignment & Attendance Tab (POST /assign, POST /attendance/barcode) -->

<div class="tab-pane fade" id="assignment-attendance" role="tabpanel">

<div class="row">

<div class="col-md-6">

<div class="card section-card">

<div class="card-header bg-success"><i class="bi bi-person-plus"></i> Student Route Assignment</div>

<div class="card-body">

<p class="alert alert-info">Assign a student to a route, checking for existing assignments and capacity limits.</p>

<form id="assignmentForm" class="row g-3">

<div class="col-md-6"><label for="assignStudentId" class="form-label">Student ID</label><input type="text" class="form-control" id="assignStudentId" required></div>

<div class="col-md-6"><label for="assignRouteId" class="form-label">Route ID</label><input type="text" class="form-control" id="assignRouteId" required></div>

<div class="col-md-6"><label for="assignPickupStopId" class="form-label">Pickup Stop ID</label><input type="text" class="form-control" id="assignPickupStopId" required></div>

<div class="col-md-6"><label for="assignDropoffStopId" class="form-label">Dropoff Stop ID (Optional)</label><input type="text" class="form-control" id="assignDropoffStopId"></div>

<div class="col-md-12">

<label for="assignFeePerTerm" class="form-label">Fee Per Term (INR)</label>

<div class="input-group">

<span class="input-group-text">₹</span>

<input type="number" class="form-control" id="assignFeePerTerm" min="0" required>

</div>

</div>

<div class="col-12"><button type="submit" class="btn btn-primary" id="assignStudentBtn"><i class="bi bi-check2-square"></i> Assign Student</button></div>

</form>

</div>

</div>

</div>

<div class="col-md-6">

<div class="card section-card">

<div class="card-header bg-warning text-dark"><i class="bi bi-barcode"></i> Transport Attendance Scanning</div>

<div class="card-body">

<p class="alert alert-warning">Use this form to mock a barcode scan event (On Board/Off Board) for real-time tracking.</p>

<form id="attendanceForm" class="row g-3">

<div class="col-md-6"><label for="barcodeUid" class="form-label">Student Barcode UID</label><input type="text" class="form-control" id="barcodeUid" required></div>

<div class="col-md-6"><label for="attendanceVehicleId" class="form-label">Vehicle ID</label><input type="text" class="form-control" id="attendanceVehicleId" required></div>

<div class="col-md-6"><label for="attendanceStatus" class="form-label">Status</label>

<select class="form-select" id="attendanceStatus" required>

<option value="on\_board">On Board (Pickup)</option>

<option value="off\_board">Off Board (Dropoff)</option>

<option value="missed">Missed (Manual)</option>

</select>

</div>

<div class="col-md-6"><label for="attendanceTimestamp" class="form-label">Timestamp</label><input type="datetime-local" class="form-control" id="attendanceTimestamp" required></div>

<div class="col-12"><button type="submit" class="btn btn-warning" id="recordAttendanceBtn"><i class="bi bi-cursor"></i> Record Scan</button></div>

</form>

</div>

</div>

</div>

</div>

</div>

<!-- 3. Billing & Invoicing Tab (POST /invoices/generate) -->

<div class="tab-pane fade" id="billing" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-primary">Monthly Invoice Generation</div>

<div class="card-body">

<p class="alert alert-info">Generate all student transportation invoices for the selected month based on assigned routes and configurations.</p>

<form id="invoiceForm">

<div class="mb-3">

<label for="billingMonth" class="form-label">Billing Month</label>

<input type="month" class="form-control" id="billingMonth" required>

<div class="form-text">Format: YYYY-MM (e.g., 2024-05)</div>

</div>

<button type="submit" class="btn btn-primary" id="generateInvoiceBtn"><i class="bi bi-file-earmark-spreadsheet"></i> Generate Invoices</button>

</form>

<h5 class="mt-5">Last Invoice Summary (Mock)</h5>

<div id="lastInvoiceSummary" class="p-3 border rounded bg-light">

<p class="text-muted">Generate a batch to see the summary here...</p>

</div>

</div>

</div>

</div>

<!-- 4. Configuration Tab (GET /config, PUT /config) -->

<div class="tab-pane fade" id="config" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-secondary">System Configuration (Manager Access)</div>

<div class="card-body">

<form id="configForm" class="row g-3">

<!-- Max Capacity Buffer -->

<div class="col-md-6">

<label for="maxCapacityBuffer" class="form-label">Max Capacity Buffer (%)</label>

<input type="number" class="form-control" id="maxCapacityBuffer" name="max\_capacity\_buffer" min="0" max="100" required>

<div class="form-text">Buffer percentage when calculating maximum safe seating capacity (e.g., 10 for 10% buffer).</div>

</div>

<!-- Fuel Price Per Liter -->

<div class="col-md-6">

<label for="fuelPricePerLiter" class="form-label">Fuel Price Per Liter (INR)</label>

<div class="input-group">

<span class="input-group-text">₹</span>

<input type="number" class="form-control" id="fuelPricePerLiter" name="fuel\_price\_per\_liter" min="1" step="0.01" required>

</div>

<div class="form-text">Used for calculating operational cost estimates.</div>

</div>

<!-- Billing Cycle -->

<div class="col-md-6">

<label for="billingCycle" class="form-label">Billing Cycle</label>

<select class="form-select" id="billingCycle" name="billing\_cycle" required>

<option value="monthly">Monthly</option>

<option value="quarterly">Quarterly</option>

</select>

<div class="form-text">How often students are billed for transport services.</div>

</div>

<!-- Emergency Contact -->

<div class="col-md-6">

<label for="emergencyContact" class="form-label">Emergency Contact Number</label>

<input type="text" class="form-control" id="emergencyContact" name="emergency\_contact" placeholder="+91..." required>

<div class="form-text">Primary contact for vehicle emergencies (Arattai/SMS target).</div>

</div>

<div class="col-12 mt-4">

<button type="submit" class="btn btn-secondary" id="saveConfigBtn"><i class="bi bi-save"></i> Save Configuration</button>

</div>

</form>

</div>

</div>

</div>

<!-- 5. Financial Oversight Tab (GET /invoices/overdue-summary) -->

<div class="tab-pane fade" id="oversight" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-danger">Financial Oversight (Principal/Admin Access)</div>

<div class="card-body">

<p class="alert alert-warning"><i class="bi bi-shield-fill-exclamation"></i> Only Principal or Super Admin can access this critical financial summary.</p>

<div class="row text-center" id="oversightStats">

<div class="col-md-4 mb-3">

<div class="p-4 border rounded bg-light">

<h4 class="text-primary" id="statTotalRoutes">...</h4>

<p class="text-muted">Total Active Routes</p>

</div>

</div>

<div class="col-md-4 mb-3">

<div class="p-4 border rounded bg-light">

<h4 class="text-danger" id="statOverdueAmount">...</h4>

<p class="text-muted">Total Overdue Fees</p>

</div>

</div>

<div class="col-md-4 mb-3">

<div class="p-4 border rounded bg-light">

<h4 class="text-success" id="statSafetyCompliance">...</h4>

<p class="text-muted">Safety Compliance (%)</p>

</div>

</div>

</div>

<button class="btn btn-sm btn-danger mt-3" id="refreshOversightBtn"><i class="bi bi-arrow-clockwise"></i> Refresh Financial Summary</button>

</div>

</div>

</div>

</div>

</div>

<!-- Bootstrap Bundle with Popper -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>

<script>

const API\_BASE = '';

let routesData = []; // Cache route data for manifest export dropdown

/\*\* Helper function to show a temporary notification. \*/

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

const area = document.getElementById('notification-area');

area.innerHTML = `

<div class="alert alert-${type} alert-dismissible fade show" role="alert">

${message}

<button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>

</div>

`;

area.style.display = 'block';

setTimeout(() => {

area.style.display = 'none';

}, 7000);

}

// Helper function to replace window.confirm (as alerts are blocked)

function customConfirm(message) {

return window.prompt(message + " (Type 'YES' to proceed)") === 'YES';

}

/\*\* Core Fetch Function with Error Handling. \*/

async function apiFetch(endpoint, method = 'GET', body = null) {

try {

const options = {

method: method,

headers: { 'Content-Type': 'application/json' },

};

if (body && method !== 'GET') {

options.body = JSON.stringify(body);

}

const response = await fetch(`${API\_BASE}${endpoint}`, options);

const data = await response.json();

if (!response.ok) {

throw new Error(data.error || `API error (${response.status})`);

}

return data;

} catch (error) {

console.error('API Fetch Error:', endpoint, error);

showNotification(error.message || 'Network error.', 'danger');

throw error;

}

}

// --- 1. OPERATIONS & ASSET LOGIC ---

// Fetch Routes (GET /routes)

async function fetchRoutes() {

const tableBody = document.getElementById('routesTableBody');

tableBody.innerHTML = '<tr><td colspan="6" class="text-center"><div class="spinner-border spinner-border-sm"></div> Loading Routes...</td></tr>';

try {

const data = await apiFetch('/routes');

routesData = data.routes;

tableBody.innerHTML = '';

if (routesData.length === 0) {

tableBody.innerHTML = '<tr><td colspan="6" class="text-center">No active routes found.</td></tr>';

document.getElementById('manifestRouteId').innerHTML = '<option value="">No Routes Available</option>';

document.getElementById('exportManifestBtn').disabled = true;

return;

}

// Populate Routes Table & Manifest Dropdown

const manifestSelect = document.getElementById('manifestRouteId');

manifestSelect.innerHTML = '<option value="">-- Select Route --</option>';

routesData.forEach(route => {

const statusClass = route.vehicle\_number ? 'bg-success' : 'bg-warning text-dark';

const safetyStatus = 'N/A'; // Mock safety status for simplicity

const row = tableBody.insertRow();

row.innerHTML = `

<td>${route.route\_number}</td>

<td>${route.name}</td>

<td>${route.route\_distance\_km || '0'}</td>

<td><span class="badge ${statusClass}">${route.vehicle\_number || 'Unassigned'}</span></td>

<td>${route.driver\_name || 'N/A'}</td>

<td>

<button class="btn btn-sm btn-warning safety-check-btn" data-route-id="${route.id}" data-route-name="${route.name}" title="Record Safety Check" disabled><i class="bi bi-shield-check"></i></button>

<button class="btn btn-sm btn-primary manifest-export-quick" data-route-id="${route.id}" title="Export Manifest"><i class="bi bi-download"></i></button>

</td>

`;

manifestSelect.innerHTML += `<option value="${route.id}">${route.route\_number} - ${route.name}</option>`;

});

document.getElementById('exportManifestBtn').disabled = false;

} catch {

tableBody.innerHTML = '<tr><td colspan="6" class="text-center text-danger">Failed to load routes.</td></tr>';

}

}

// Fetch Vehicle CRUD List (Mocked) - Updated to include data-attributes for full CRUD

async function fetchVehicles() {

const tableBody = document.getElementById('vehicleTableBody');

tableBody.innerHTML = '<tr><td colspan="5" class="text-center"><div class="spinner-border spinner-border-sm"></div> Loading Vehicles...</td></tr>';

// Mocking a fetch for the vehicle list

const mockVehicles = [

{ id: 'v1', vehicle\_number: 'TN01AZ1234', capacity: 40, driver\_name: 'Rajesh K', driver\_phone: '+919999912345', route\_id: 'r1' },

{ id: 'v2', vehicle\_number: 'TN02BY5678', capacity: 30, driver\_name: 'Priya S', driver\_phone: '+919876554321', route\_id: null },

];

tableBody.innerHTML = '';

mockVehicles.forEach(v => {

const row = tableBody.insertRow();

// Store the full vehicle object data as a string attribute for easy editing

const vehicleDataAttr = JSON.stringify(v);

row.innerHTML = `

<td>${v.vehicle\_number}</td>

<td>${v.capacity}</td>

<td>${v.driver\_name}</td>

<td>${v.route\_id || 'N/A'}</td>

<td>

<button class="btn btn-sm btn-info edit-vehicle-btn" data-vehicle-data='${vehicleDataAttr}'><i class="bi bi-pencil"></i> Edit</button>

<button class="btn btn-sm btn-danger delete-vehicle-btn" data-id="${v.id}" data-name="${v.vehicle\_number}"><i class="bi bi-trash"></i> Delete</button>

</td>

`;

});

}

// Fetch Stops CRUD List (Mocked) - Updated to include data-attributes for full CRUD

async function fetchStops() {

const tableBody = document.getElementById('stopTableBody');

tableBody.innerHTML = '<tr><td colspan="4" class="text-center"><div class="spinner-border spinner-border-sm"></div> Loading Stops...</td></tr>';

// Mocking stops

const mockStops = [

{ id: 's1', stop\_name: 'City Gate Park', route\_id: 'r1', latitude: 13.0, longitude: 80.2 },

{ id: 's2', stop\_name: 'Main Street Crossing', route\_id: 'r2', latitude: 13.1, longitude: 80.1 },

];

tableBody.innerHTML = '';

mockStops.forEach(s => {

const row = tableBody.insertRow();

const stopDataAttr = JSON.stringify(s);

row.innerHTML = `

<td>${s.stop\_name}</td>

<td>${s.route\_id}</td>

<td>${s.latitude.toFixed(3)}, ${s.longitude.toFixed(3)}</td>

<td>

<button class="btn btn-sm btn-info edit-stop-btn" data-stop-data='${stopDataAttr}'><i class="bi bi-pencil"></i> Edit</button>

<button class="btn btn-sm btn-danger delete-stop-btn" data-id="${s.id}" data-name="${s.stop\_name}"><i class="bi bi-trash"></i> Delete</button>

</td>

`;

});

}

// Fetch Live Locations (GET /vehicles/location)

async function fetchLiveLocations() {

const liveDiv = document.getElementById('liveLocationData');

liveDiv.innerHTML = '<p class="text-center"><div class="spinner-border spinner-border-sm"></div> Fetching...</p>';

try {

const data = await apiFetch('/vehicles/location');

liveDiv.innerHTML = '';

if (data.liveLocations.length === 0) {

liveDiv.innerHTML = '<p class="text-muted">No vehicle GPS data available.</p>';

return;

}

data.liveLocations.forEach(loc => {

const time = new Date(loc.timestamp).toLocaleTimeString();

liveDiv.innerHTML += `

<p class="mb-1"><span class="gps-dot"></span>

<strong>${loc.vehicle\_number}</strong> (Route: ${loc.route\_id || 'N/A'}) -

Lat: ${loc.latitude.toFixed(4)}, Lng: ${loc.longitude.toFixed(4)} @ ${time}

</p>

`;

});

} catch {

liveDiv.innerHTML = '<p class="text-danger">Failed to connect to live GPS service.</p>';

}

}

// --- 1.1 ROUTE CRUD HANDLERS ---

document.getElementById('newRouteForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const payload = {

route\_number: form.route\_number.value,

name: form.name.value,

route\_distance\_km: parseFloat(form.route\_distance\_km.value),

start\_point: form.start\_point.value,

end\_point: form.end\_point.value

};

const btn = document.getElementById('createRouteBtn');

btn.disabled = true;

try {

const data = await apiFetch('/route', 'POST', payload);

showNotification(data.message, 'success');

form.reset();

fetchRoutes();

} catch {

// Error handled by apiFetch

} finally {

btn.disabled = false;

}

});

document.getElementById('refreshRoutesBtn').addEventListener('click', () => {

fetchRoutes();

fetchLiveLocations();

});

// --- 1.2 MANIFEST EXPORT HANDLERS (GET /manifest/export/:routeId) ---

document.getElementById('exportManifestBtn').addEventListener('click', () => {

const routeId = document.getElementById('manifestRouteId').value;

if (!routeId) {

showNotification('Please select a route to export the manifest.', 'warning');

return;

}

// Directly trigger download via window location for CSV

window.location.href = `${API\_BASE}/manifest/export/${routeId}?format=csv`;

showNotification(`Generating manifest for Route ${routeId}... Check your downloads.`, 'info');

});

// Quick export from table row

document.getElementById('routesTableBody').addEventListener('click', (e) => {

if (e.target.classList.contains('manifest-export-quick')) {

const routeId = e.target.getAttribute('data-route-id');

window.location.href = `${API\_BASE}/manifest/export/${routeId}?format=csv`;

showNotification(`Quick generating manifest for Route ${routeId}...`, 'info');

}

});

// --- 1.3 VEHICLE CRUD HANDLERS (POST/PUT/DELETE /vehicle) ---

document.getElementById('vehicleCrudForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const vehicleId = document.getElementById('vehicleIdToEdit').value;

const method = vehicleId ? 'PUT' : 'POST';

const endpoint = vehicleId ? `/vehicle/${vehicleId}` : '/vehicle';

const payload = {

vehicle\_number: form.vehicleNumber.value,

capacity: parseInt(form.vehicleCapacity.value),

driver\_name: form.vehicleDriverName.value,

driver\_phone: form.vehicleDriverPhone.value,

route\_id: form.vehicleRouteId.value || null

};

const btn = document.getElementById('saveVehicleBtn');

btn.disabled = true;

try {

const data = await apiFetch(endpoint, method, payload);

showNotification(data.message, 'success');

// --- Cleanup ---

form.reset();

document.getElementById('vehicleIdToEdit').value = '';

document.getElementById('saveVehicleBtn').textContent = 'Add New Vehicle'; // Reset button text

// --- End Cleanup ---

fetchVehicles(); // Refresh list

fetchRoutes(); // Refresh routes list if route was assigned/unassigned

} catch {

// Error handled by apiFetch

} finally {

btn.disabled = false;

}

});

// --- 1.4 STOP CRUD HANDLERS (POST/PUT/DELETE /stop) ---

document.getElementById('stopCrudForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const stopId = document.getElementById('stopIdToEdit').value;

const method = stopId ? 'PUT' : 'POST';

const endpoint = stopId ? `/stop/${stopId}` : '/stop';

const payload = {

stop\_name: form.stopName.value,

route\_id: form.stopRouteId.value,

latitude: parseFloat(form.stopLatitude.value),

longitude: parseFloat(form.stopLongitude.value)

};

const btn = document.getElementById('saveStopBtn');

btn.disabled = true;

try {

const data = await apiFetch(endpoint, method, payload);

showNotification(data.message, 'success');

// --- Cleanup ---

form.reset();

document.getElementById('stopIdToEdit').value = '';

document.getElementById('saveStopBtn').textContent = 'Add New Stop'; // Reset button text

// --- End Cleanup ---

fetchStops(); // Refresh list

} catch {

// Error handled by apiFetch

} finally {

btn.disabled = false;

}

});

// --- 2. ASSIGNMENT & ATTENDANCE LOGIC ---

// Student Assignment (POST /assign)

document.getElementById('assignmentForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const payload = {

studentId: form.assignStudentId.value,

routeId: form.assignRouteId.value,

pickupStopId: form.assignPickupStopId.value,

dropoffStopId: form.assignDropoffStopId.value || null,

feePerTerm: parseFloat(form.assignFeePerTerm.value)

};

const btn = document.getElementById('assignStudentBtn');

btn.disabled = true;

try {

const data = await apiFetch('/assign', 'POST', payload);

showNotification(data.message, 'success');

form.reset();

} catch {

// Error handled by apiFetch

} finally {

btn.disabled = false;

}

});

// Attendance Scanning (POST /attendance/barcode)

document.getElementById('attendanceForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

// Format timestamp for ISO 8601

const timestamp = new Date(form.attendanceTimestamp.value).toISOString();

const payload = {

barcodeUid: form.barcodeUid.value,

vehicleId: form.attendanceVehicleId.value,

status: form.attendanceStatus.value,

timestamp: timestamp

};

const btn = document.getElementById('recordAttendanceBtn');

btn.disabled = true;

try {

const data = await apiFetch('/attendance/barcode', 'POST', payload);

showNotification(data.message, 'success');

// Keep form values for rapid scanning, just clear barcode field

form.barcodeUid.value = '';

} catch {

// Error handled by apiFetch

} finally {

btn.disabled = false;

}

});

// --- 3. BILLING & INVOICING LOGIC ---

// Invoice Generation (POST /invoices/generate - Mocked endpoint)

document.getElementById('invoiceForm').addEventListener('submit', async (e) => {

e.preventDefault();

const month = document.getElementById('billingMonth').value;

const btn = document.getElementById('generateInvoiceBtn');

btn.disabled = true;

const summaryDiv = document.getElementById('lastInvoiceSummary');

summaryDiv.innerHTML = '<p class="text-center"><div class="spinner-border spinner-border-sm"></div> Generating batch...</p>';

// NOTE: Mocking the endpoint response

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

showNotification(`Invoice generation triggered for ${month}.`, 'success');

summaryDiv.innerHTML = `

<p><strong>Batch Status: Completed</strong></p>

<p>Month: ${month}</p>

<p>Total Invoices Generated: 450</p>

<p class="text-success">Total Billed Amount: ₹2,560,000</p>

<p class="text-muted">Notifications queued to 450 parents.</p>

`;

btn.disabled = false;

});

// --- 4. CONFIGURATION LOGIC (GET/PUT /config) ---

async function fetchConfig() {

try {

const data = await apiFetch('/config');

const settings = data.settings;

// Populate form fields, using default values if settings are missing

document.getElementById('maxCapacityBuffer').value = settings.max\_capacity\_buffer || 10;

document.getElementById('fuelPricePerLiter').value = settings.fuel\_price\_per\_liter || 95.50;

document.getElementById('billingCycle').value = settings.billing\_cycle || 'monthly';

document.getElementById('emergencyContact').value = settings.emergency\_contact || '+919876543210';

} catch (error) {

// If 403 (Auth denied) or 500 (DB error), configuration page shows defaults/empty.

}

}

document.getElementById('configForm').addEventListener('submit', async (e) => {

e.preventDefault();

const btn = document.getElementById('saveConfigBtn');

btn.disabled = true;

const form = e.target;

const payload = {

max\_capacity\_buffer: parseInt(form.maxCapacityBuffer.value),

fuel\_price\_per\_liter: parseFloat(form.fuelPricePerLiter.value),

billing\_cycle: form.billingCycle.value,

emergency\_contact: form.emergencyContact.value

};

try {

// Real API call for PUT /config

const data = await apiFetch('/config', 'PUT', payload);

showNotification(data.message || 'Configuration updated successfully.', 'success');

fetchConfig();

} catch (error) {

// Error handled by apiFetch and shown via notification

} finally {

btn.disabled = false;

}

});

// --- 5. OVERSIGHT LOGIC (GET /invoices/overdue-summary - Mocked endpoint) ---

async function fetchOversightStats() {

document.getElementById('statTotalRoutes').textContent = '...';

document.getElementById('statOverdueAmount').textContent = '...';

document.getElementById('statSafetyCompliance').textContent = '...';

try {

// NOTE: This endpoint was not in the provided backend but derived from the previous context. Mocking stats.

await new Promise(r => setTimeout(r, 800));

const stats = {

totalActiveRoutes: routesData.length,

totalOverdueAmount: 45000.75, // Mock value

safetyCompliance: 92.5 // Mock value

};

document.getElementById('statTotalRoutes').textContent = stats.totalActiveRoutes;

document.getElementById('statOverdueAmount').textContent = `₹${stats.totalOverdueAmount.toFixed(2).toLocaleString()}`;

document.getElementById('statSafetyCompliance').textContent = `${stats.safetyCompliance.toFixed(1)}%`;

showNotification('Financial oversight stats refreshed.', 'info');

} catch (error) {

document.getElementById('oversightStats').innerHTML = '<div class="alert alert-danger">Access denied or failed to load statistics.</div>';

}

}

document.getElementById('refreshOversightBtn').addEventListener('click', fetchOversightStats);

// --- Delegation for Vehicle CRUD Actions ---

document.getElementById('vehicleTableBody').addEventListener('click', async (e) => {

if (e.target.closest('.edit-vehicle-btn')) {

const button = e.target.closest('.edit-vehicle-btn');

const vehicleData = JSON.parse(button.getAttribute('data-vehicle-data'));

document.getElementById('vehicleIdToEdit').value = vehicleData.id;

document.getElementById('vehicleNumber').value = vehicleData.vehicle\_number;

document.getElementById('vehicleCapacity').value = vehicleData.capacity;

document.getElementById('vehicleDriverName').value = vehicleData.driver\_name;

document.getElementById('vehicleDriverPhone').value = vehicleData.driver\_phone;

document.getElementById('vehicleRouteId').value = vehicleData.route\_id || '';

document.getElementById('saveVehicleBtn').textContent = 'Update Vehicle';

// Open the accordion

const collapseElement = document.getElementById('collapseVehicle');

const bsCollapse = new bootstrap.Collapse(collapseElement, { toggle: false });

bsCollapse.show();

collapseElement.scrollIntoView({ behavior: 'smooth' });

} else if (e.target.closest('.delete-vehicle-btn')) {

const button = e.target.closest('.delete-vehicle-btn');

const id = button.getAttribute('data-id');

const name = button.getAttribute('data-name');

if (customConfirm(`Are you sure you want to DELETE vehicle ${name}? This action cannot be undone and will unassign its route.`)) {

try {

await apiFetch(`/vehicle/${id}`, 'DELETE');

showNotification(`Vehicle ${name} deleted successfully.`, 'success');

fetchVehicles(); // Refresh list

fetchRoutes(); // Refresh routes list

} catch {

// Error handled by apiFetch

}

}

}

});

// --- Delegation for Stop CRUD Actions ---

document.getElementById('stopTableBody').addEventListener('click', async (e) => {

if (e.target.closest('.edit-stop-btn')) {

const button = e.target.closest('.edit-stop-btn');

const stopData = JSON.parse(button.getAttribute('data-stop-data'));

document.getElementById('stopIdToEdit').value = stopData.id;

document.getElementById('stopName').value = stopData.stop\_name;

document.getElementById('stopRouteId').value = stopData.route\_id;

document.getElementById('stopLatitude').value = stopData.latitude;

document.getElementById('stopLongitude').value = stopData.longitude;

document.getElementById('saveStopBtn').textContent = 'Update Stop';

// Open the accordion

const collapseElement = document.getElementById('collapseStop');

const bsCollapse = new bootstrap.Collapse(collapseElement, { toggle: false });

bsCollapse.show();

collapseElement.scrollIntoView({ behavior: 'smooth' });

} else if (e.target.closest('.delete-stop-btn')) {

const button = e.target.closest('.delete-stop-btn');

const id = button.getAttribute('data-id');

const name = button.getAttribute('data-name');

if (customConfirm(`Are you sure you want to DELETE stop "${name}"? This will affect routes and student assignments.`)) {

try {

await apiFetch(`/stop/${id}`, 'DELETE');

showNotification(`Stop "${name}" deleted successfully.`, 'success');

fetchStops(); // Refresh list

} catch {

// Error handled by apiFetch

}

}

}

});

// --- Initialization ---

document.addEventListener('DOMContentLoaded', () => {

// Set default attendance timestamp to current time

const now = new Date();

now.setMinutes(now.getMinutes() - now.getTimezoneOffset());

document.getElementById('attendanceTimestamp').value = now.toISOString().slice(0, 16);

fetchRoutes();

fetchVehicles();

fetchStops();

fetchConfig();

fetchLiveLocations();

// Setup tab switch to load oversight on demand

document.getElementById('oversight-tab').addEventListener('shown.bs.tab', fetchOversightStats);

});

</script>

</body>

</html>

8. client/dist/modules/Admission\_Management.html

<!DOCTYPE html>  
<html lang="en">  
<head>  
<meta charset="UTF-8">  
<meta name="viewport" content="width=device-width, initial-scale=1.0">  
<title>🏫 Admission Management System</title>  
<!-- Bootstrap 5 CDN -->  
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">  
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css">  
<style>  
body { background-color: #f0f4f8; font-family: 'Inter', sans-serif; }  
.container-fluid { max-width: 1200px; margin-top: 20px; }  
.card-header { background-color: #2c3e50; color: white; font-weight: bold; border-radius: 0.5rem 0.5rem 0 0; }  
.section-card { margin-bottom: 25px; box-shadow: 0 6px 12px rgba(0,0,0,0.1); border-radius: 0.5rem; }  
.alert-fixed {  
position: fixed; top: 20px; right: 20px; z-index: 1050; min-width: 300px; border-radius: 0.5rem;  
}  
.nav-tabs .nav-link.active {  
background-color: #3498db;  
color: white;  
border-color: #3498db;  
}  
.status-applied { background-color: #3498db; color: white; }  
.status-scheduled { background-color: #f1c40f; color: #333; }  
.status-admitted { background-color: #27ae60; color: white; }  
.status-rejected { background-color: #e74c3c; color: white; }  
.status-default { background-color: #bdc3c7; color: #333; }  
.history-item { font-size: 0.85rem; padding: 5px 0; border-bottom: 1px dashed #eee; }  
</style>  
</head>  
<body>  
  
<div class="container-fluid">  
<h1 class="text-center mb-4 text-primary"><i class="bi bi-person-fill-add"></i> Admission Management Dashboard</h1>  
  
<!-- Notification Area -->  
<div id="notification-area" class="alert-fixed" style="display: none;"></div>  
  
<!-- Tab Navigation -->  
<ul class="nav nav-tabs" id="admissionTab" role="tablist">  
<li class="nav-item" role="presentation">  
<button class="nav-link active" id="review-tab" data-bs-toggle="tab" data-bs-target="#review" type="button" role="tab"><i class="bi bi-list-check"></i> Application Review</button>  
</li>  
<li class="nav-item" role="presentation">  
<button class="nav-link" id="decision-tab" data-bs-toggle="tab" data-bs-target="#decision" type="button" role="tab"><i class="bi bi-check2-circle"></i> Decision Workflow</button>  
</li>  
<li class="nav-item" role="presentation">  
<button class="nav-link" id="analytics-tab" data-bs-toggle="tab" data-bs-target="#analytics" type="button" role="tab"><i class="bi bi-bar-chart-line"></i> Statistics & Bulk</button>  
</li>  
<li class="nav-item" role="presentation">  
<button class="nav-link" id="config-tab" data-bs-toggle="tab" data-bs-target="#config" type="button" role="tab"><i class="bi bi-gear-fill"></i> Configuration</button>  
</li>  
<li class="nav-item" role="presentation">  
<button class="nav-link" id="public-portal-tab" data-bs-toggle="tab" data-bs-target="#public-portal" type="button" role="tab"><i class="bi bi-door-open"></i> Public Status Check</button>  
</li>  
</ul>  
  
<!-- Tab Content -->  
<div class="tab-content pt-3" id="admissionTabContent">  
  
<!-- 1. Application Review Tab (GET /applications, File Upload) -->  
<div class="tab-pane fade show active" id="review" role="tabpanel">  
<div class="card section-card">  
<div class="card-header">Application List & Document Verification</div>  
<div class="card-body">  
  
<!-- Statistics Summary (Inline) -->  
<div id="review-stats-summary" class="row text-center mb-4">  
<!-- Stats loaded dynamically -->  
</div>  
  
<!-- Application Table -->  
<div class="table-responsive">  
<table class="table table-striped table-hover table-sm">  
<thead>  
<tr>  
<th>App No.</th>  
<th>Applicant</th>  
<th>Applied Class</th>  
<th>Status</th>  
<th>Submitted On</th>  
<th>Action</th>  
</tr>  
</thead>  
<tbody id="applicationsTableBody">  
<tr><td colspan="6" class="text-center">Loading applications...</td></tr>  
</tbody>  
</table>  
</div>  
</div>  
</div>  
</div>  
  
<!-- 2. Decision Workflow Tab (Document Verification, Scheduling, Final Decision) -->  
<div class="tab-pane fade" id="decision" role="tabpanel">  
<div class="card section-card">  
<div class="card-header">Application Workflow for: <span id="currentApplicantName" class="badge bg-light text-dark">Select Application</span></div>  
<div class="card-body">  
<div class="row">  
  
<!-- Document Verification Panel -->  
<div class="col-md-6 border-end">  
<h5>1. Document Verification & Upload</h5>  
<p class="text-muted">Review, upload, and verify required documents.</p>  
<div id="documentVerificationArea" class="mb-4">  
<p class="alert alert-warning">Please select an application from the 'Application Review' tab to begin the workflow.</p>  
<!-- Dynamic document list and verification buttons go here -->  
</div>  
  
<!-- Document Upload Form -->  
<h6 class="mt-4">Upload Supporting Document (Requires file selection)</h6>  
<form id="documentUploadForm" class="input-group">  
<input type="hidden" id="uploadAppId">  
<select class="form-select" id="documentTypeSelect" required>  
<option value="">Select Document Type</option>  
<!-- Document options dynamically generated from config -->  
</select>  
<input type="file" class="form-control" name="document" id="documentFileInput" required>  
<button class="btn btn-secondary" type="submit" id="uploadDocBtn" disabled><i class="bi bi-upload"></i> Upload</button>  
</form>  
</div>  
  
<!-- Scheduling & Final Decision Panel -->  
<div class="col-md-6">  
<h5>2. Scheduling & Final Decision</h5>  
  
<!-- Scheduling Form (POST /schedule) -->  
<h6 class="mt-3">Schedule Interview/Assessment</h6>  
<form id="scheduleForm" class="row g-2 mb-4">  
<input type="hidden" id="scheduleAppId">  
<div class="col-md-4"><select class="form-select" id="scheduleType" required><option value="interview">Interview</option><option value="assessment">Assessment</option></select></div>  
<div class="col-md-4"><input type="date" class="form-control" id="scheduleDate" required></div>  
<div class="col-md-4"><input type="time" class="form-control" id="scheduleTime" required></div>  
<div class="col-12"><button type="submit" class="btn btn-warning w-100" id="scheduleBtn" disabled><i class="bi bi-calendar-plus"></i> Schedule & Notify Parent</button></div>  
</form>  
  
<hr>  
  
<!-- Final Decision Form (POST /admission-decision) -->  
<h5 class="text-danger">3. Final Approval (Principal Only)</h5>  
<form id="decisionForm" class="row g-3">  
<input type="hidden" id="decisionAppId">  
<div class="col-md-6"><label for="decisionClass" class="form-label">Allocated Class</label><input type="text" class="form-control" id="decisionClass" placeholder="e.g., 5th Grade" required></div>  
<div class="col-md-6"><label for="decisionSection" class="form-label">Allocated Section</label><input type="text" class="form-control" id="decisionSection" placeholder="e.g., A" required></div>  
<div class="col-12 d-flex justify-content-between mt-4">  
<button type="button" class="btn btn-success btn-lg" id="admitBtn" data-decision="admitted" disabled><i class="bi bi-check-lg"></i> Admit</button>  
<button type="button" class="btn btn-danger btn-lg" id="rejectBtn" data-decision="rejected" disabled><i class="bi bi-x-lg"></i> Reject</button>  
</div>  
<div class="col-12 mt-3"><p id="decisionWarning" class="text-danger small">Final approval is a permanent action that creates a student record if Admitted.</p></div>  
</form>  
</div>  
</div>  
</div>  
</div>  
</div>  
  
<!-- 3. Statistics & Bulk Operations Tab -->  
<div class="tab-pane fade" id="analytics" role="tabpanel">  
<div class="card section-card">  
<div class="card-header">Admission Analytics & Bulk Tools</div>  
<div class="card-body">  
  
<!-- Statistics (GET /statistics) -->  
<h5 class="mb-3">Admission Funnel Summary</h5>  
<div id="statisticsArea" class="row text-center mb-5">  
<div class="col-md-4"><div class="p-3 border rounded bg-light"><h4 class="text-primary" id="statTotal">...</h4><p class="text-muted">Total Applications</p></div></div>  
<div class="col-md-4"><div class="p-3 border rounded bg-light"><h4 class="text-success" id="statAdmitted">...</h4><p class="text-muted">Admitted Students</p></div></div>  
<div class="col-md-4"><div class="p-3 border rounded bg-light"><h4 class="text-warning" id="statConversion">...</h4><p class="text-muted">Conversion Rate</p></div></div>  
</div>  
  
<hr>  
  
<!-- Bulk Operations (POST /bulk/upload, GET /bulk/download) -->  
<div class="row">  
<div class="col-md-6 border-end">  
<h5>Bulk Application Upload</h5>  
<p class="text-muted">Upload CSV/Excel file for batch application entry.</p>  
<form id="bulkUploadForm">  
<div class="mb-3"><input type="file" class="form-control" name="file" id="bulkFileInput" accept=".csv, .xlsx" required></div>  
<button type="submit" class="btn btn-primary" id="bulkUploadBtn"><i class="bi bi-file-earmark-arrow-up"></i> Start Bulk Upload</button>  
</form>  
</div>  
<div class="col-md-6">  
<h5>Data Export & Reporting</h5>  
<p class="text-muted">Download comprehensive data sets for offline analysis.</p>  
<button class="btn btn-secondary" id="bulkDownloadBtn"><i class="bi bi-file-earmark-arrow-down"></i> Export All Data</button>  
</div>  
</div>  
</div>  
</div>  
</div>  
  
<!-- 4. Configuration Tab (Integrated GET/PUT /config) -->  
<div class="tab-pane fade" id="config" role="tabpanel">  
<div class="card section-card">  
<div class="card-header bg-secondary">System Configuration (Admission Manager Setup)</div>  
<div class="card-body">  
<p class="alert alert-info"><i class="bi bi-gear-fill"></i> Adjust key admission policies for the current academic cycle.</p>  
  
<form id="admissionConfigForm">  
<h5 class="mt-4">Intake Cycle Setup</h5>  
<div class="row g-3 mb-4 border p-3 rounded">  
<div class="col-md-4">  
<label for="currentYear" class="form-label">Current Academic Year</label>  
<input type="text" class="form-control" id="currentYear" required>  
</div>  
<div class="col-md-4">  
<label for="maxIntake" class="form-label">Maximum Intake Capacity</label>  
<input type="number" class="form-control" id="maxIntake" required min="1">  
</div>  
<div class="col-md-4">  
<label for="admissionStatus" class="form-label">Global Admission Status</label>  
<select class="form-select" id="admissionStatus">  
<option value="open">Open for Applications</option>  
<option value="closed">Closed for Applications</option>  
<option value="waitlist\_only">Waitlist Only</option>  
</select>  
</div>  
</div>  
  
<h5 class="mt-4">Document Verification Checklist</h5>  
<div class="row g-3 mb-4 border p-3 rounded">  
<div class="col-md-6">  
<label class="form-label">Required Documents (Checked = Mandatory):</label>  
<div id="requiredDocsChecklist">  
<!-- Checklist populated by JS from config -->  
<p class="text-muted">Loading documents list...</p>  
</div>  
</div>  
<div class="col-md-6">  
<label for="verificationThreshold" class="form-label">Required Verification Threshold (%)</label>  
<input type="number" class="form-control" id="verificationThreshold" min="0" max="100" required>  
<div class="form-text">Percentage of documents that must be verified for workflow progression.</div>  
</div>  
</div>  
  
<h5 class="mt-4">Assessment & Notification Settings</h5>  
<div class="row g-3 mb-4 border p-3 rounded">  
<div class="col-md-6">  
<label for="minAssessmentScore" class="form-label">Min Assessment Score (%)</label>  
<input type="number" class="form-control" id="minAssessmentScore" min="0" max="100" required>  
<div class="form-text">Score below which a candidate is flagged for review.</div>  
</div>  
<div class="col-md-6">  
<label for="defaultNotificationPhone" class="form-label">Default Admission Notification Phone</label>  
<input type="text" class="form-control" id="defaultNotificationPhone" placeholder="+91..." required>  
<div class="form-text">Primary number used for Arattai/WhatsApp integration tests.</div>  
</div>  
</div>  
  
<button type="submit" class="btn btn-secondary mt-3" id="saveAdmissionConfigBtn"><i class="bi bi-save"></i> Save Configuration</button>  
</form>  
  
</div>  
</div>  
</div>  
  
<!-- 5. Public Status Check Portal (GET /portal/:applicationNumber) -->  
<div class="tab-pane fade" id="public-portal" role="tabpanel">  
<div class="card section-card">  
<div class="card-header bg-info">Application Status Portal</div>  
<div class="card-body">  
<p class="alert alert-info">Enter the application number to check the current status and history.</p>  
  
<form id="statusCheckForm" class="input-group mb-4">  
<input type="text" class="form-control" id="checkAppNumber" placeholder="Enter Application Number (e.g., ADM20241234)" required>  
<button class="btn btn-info" type="submit" id="checkStatusBtn"><i class="bi bi-search"></i> Check Status</button>  
</form>  
  
<div id="portalResultArea" class="mt-4 p-3 border rounded bg-light" style="min-height: 150px;">  
<h5 id="portalStatusDisplay" class="mb-3"></h5>  
<p id="portalApplicantName"></p>  
<h6>Application History:</h6>  
<div id="portalHistoryList"></div>  
</div>  
</div>  
</div>  
</div>  
  
</div>  
</div>  
  
<!-- Bootstrap Bundle with Popper -->  
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>  
  
<script>  
const API\_BASE = '';   
let currentSelectedApplication = null;  
let globalAdmissionConfig = {}; // Stores fetched configuration  
  
/\*\* Helper function to show a temporary notification. \*/  
function showNotification(message, type = 'success') {  
const area = document.getElementById('notification-area');  
area.innerHTML = `  
<div class="alert alert-${type} alert-dismissible fade show" role="alert">  
${message}  
<button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>  
</div>  
`;  
area.style.display = 'block';  
setTimeout(() => {  
area.style.display = 'none';  
}, 7000);  
}  
  
// Helper function to replace window.confirm (as alerts are blocked)  
function customConfirm(message) {  
return window.prompt(message + " (Type 'YES' to proceed)") === 'YES';  
}  
  
/\*\* Core Fetch Function with Error Handling. \*/  
async function apiFetch(endpoint, method = 'GET', body = null, isFormData = false) {  
try {  
const options = { method: method };  
  
if (!isFormData) {  
options.headers = { 'Content-Type': 'application/json' };  
if (body && method !== 'GET') { options.body = JSON.stringify(body); }  
} else if (body && method !== 'GET') {  
options.body = body;  
}  
  
const response = await fetch(`${API\_BASE}${endpoint}`, options);  
const data = await response.json();  
  
if (!response.ok) {  
throw new Error(data.error || `API error (${response.status})`);  
}  
return data;  
  
} catch (error) {  
console.error('API Fetch Error:', endpoint, error);  
showNotification(error.message || 'Network error.', 'danger');  
throw error;  
}  
}  
  
// --- STATUS & HISTORY HELPERS ---  
function getStatusBadge(status) {  
const normalizedStatus = status.toLowerCase().replace(/[^a-z]/g, '');  
const statusClass = normalizedStatus.includes('scheduled') ? 'scheduled' :   
normalizedStatus.includes('admitted') ? 'admitted' :  
normalizedStatus.includes('rejected') ? 'rejected' :   
normalizedStatus.includes('applied') ? 'applied' : 'default';  
  
return `<span class="badge status-${statusClass}">${status.toUpperCase()}</span>`;  
}  
  
function formatHistory(historyJson) {  
let history;  
try {  
history = JSON.parse(historyJson);  
} catch (e) {  
history = historyJson;   
}  
  
if (!Array.isArray(history)) return '<p class="text-muted">No history recorded.</p>';  
  
return history.map(item => {  
const date = new Date(item.timestamp).toLocaleString();  
let details = item.doc ? `(${item.doc})` : (item.date ? `(${item.date} @ ${item.time})` : '');  
  
return `<div class="history-item"><strong>${item.status.toUpperCase()}</strong> ${details} by ${item.actor} on ${date}</div>`;  
}).join('');  
}  
  
// --- CONFIGURATION LOGIC (GET/PUT /config) ---  
  
async function fetchConfig() {  
try {  
const data = await apiFetch('/config');  
const settings = data.settings;  
globalAdmissionConfig = settings;  
  
// 1. Populate form fields  
document.getElementById('currentYear').value = settings.currentYear || '';  
document.getElementById('maxIntake').value = settings.maxIntake || 0;  
document.getElementById('admissionStatus').value = settings.admissionStatus || 'open';  
document.getElementById('verificationThreshold').value = settings.verificationThreshold || 100;  
document.getElementById('minAssessmentScore').value = settings.minAssessmentScore || 70;  
document.getElementById('defaultNotificationPhone').value = settings.defaultNotificationPhone || '';  
  
// 2. Render Document Checklist  
renderDocumentChecklist(settings.requiredDocs);  
  
// 3. Update Document Upload Select Options  
renderDocumentUploadOptions(settings.requiredDocs);  
  
} catch (error) {  
console.error("Configuration fetch failed:", error);  
showNotification("Failed to load configuration. Using defaults.", 'danger');  
}  
}  
  
function renderDocumentChecklist(requiredDocs) {  
const listDiv = document.getElementById('requiredDocsChecklist');  
listDiv.innerHTML = '';  
  
const standardDocs = [  
{ key: 'birth\_certificate', label: 'Birth Certificate' },  
{ key: 'mark\_sheet', label: 'Previous Mark Sheet' },  
{ key: 'photo', label: 'Passport Size Photo' },  
{ key: 'transfer\_certificate', label: 'Transfer Certificate' }  
];  
  
standardDocs.forEach(doc => {  
const checked = requiredDocs.includes(doc.key) ? 'checked' : '';  
listDiv.innerHTML += `  
<div class="form-check">  
<input class="form-check-input required-doc-checkbox" type="checkbox" id="doc-${doc.key}" value="${doc.key}" ${checked}>  
<label class="form-check-label" for="doc-${doc.key}">${doc.label}</label>  
</div>  
`;  
});  
}  
  
function renderDocumentUploadOptions(requiredDocs) {  
const select = document.getElementById('documentTypeSelect');  
select.innerHTML = '<option value="">Select Document Type</option>';  
  
requiredDocs.forEach(docKey => {  
select.innerHTML += `<option value="${docKey}">${docKey.replace(/\_/g, ' ').toUpperCase()}</option>`;  
});  
}  
  
  
document.getElementById('admissionConfigForm').addEventListener('submit', async (e) => {  
e.preventDefault();  
const form = e.target;  
const btn = document.getElementById('saveAdmissionConfigBtn');  
btn.disabled = true;  
  
// Collect required documents from checked boxes  
const requiredDocs = Array.from(document.querySelectorAll('.required-doc-checkbox:checked')).map(cb => cb.value);  
  
const payload = {  
currentYear: form.currentYear.value,  
maxIntake: parseInt(form.maxIntake.value),  
admissionStatus: form.admissionStatus.value,  
verificationThreshold: parseInt(form.verificationThreshold.value),  
minAssessmentScore: parseInt(form.minAssessmentScore.value),  
defaultNotificationPhone: form.defaultNotificationPhone.value,  
requiredDocs: requiredDocs  
};  
  
try {  
const data = await apiFetch('/config', 'PUT', payload);  
showNotification(data.message, 'success');  
fetchConfig(); // Reload config to confirm changes  
} catch (error) {  
// Error handled by apiFetch  
} finally {  
btn.disabled = false;  
}  
});  
  
  
// --- 1. APPLICATION REVIEW LOGIC ---  
  
async function fetchApplications() {  
const tableBody = document.getElementById('applicationsTableBody');  
tableBody.innerHTML = '<tr><td colspan="6" class="text-center"><div class="spinner-border spinner-border-sm"></div> Loading applications...</td></tr>';  
  
try {  
const data = await apiFetch('/applications');  
  
tableBody.innerHTML = '';  
if (data.applications.length === 0) {  
tableBody.innerHTML = '<tr><td colspan="6" class="text-center">No admission applications found.</td></tr>';  
return;  
}  
  
data.applications.forEach(app => {  
const appliedClass = app.application\_data?.appliedForClass || 'N/A';  
const createdAt = new Date(app.created\_at).toLocaleDateString();  
  
const row = tableBody.insertRow();  
row.innerHTML = `  
<td>${app.application\_number}</td>  
<td>${app.applicant\_name}</td>  
<td>${appliedClass}</td>  
<td>${getStatusBadge(app.admission\_status)}</td>  
<td>${createdAt}</td>  
<td><button class="btn btn-sm btn-primary review-app-btn" data-id="${app.id}" data-app-data='${JSON.stringify(app)}'><i class="bi bi-arrow-right-circle"></i> Review</button></td>  
`;  
});  
  
fetchStatistics(); // Update stats immediately after fetching apps  
  
} catch {  
tableBody.innerHTML = '<tr><td colspan="6" class="text-center text-danger">Failed to load applications.</td></tr>';  
}  
}  
  
// Handle 'Review' button click to populate the Decision tab  
document.getElementById('applicationsTableBody').addEventListener('click', (e) => {  
const button = e.target.closest('.review-app-btn');  
if (button) {  
const appData = JSON.parse(button.getAttribute('data-app-data'));  
currentSelectedApplication = appData;  
  
// Set UI context  
document.getElementById('currentApplicantName').textContent = `${appData.applicant\_name} (${appData.application\_number})`;  
document.getElementById('uploadAppId').value = appData.id;  
document.getElementById('scheduleAppId').value = appData.id;  
document.getElementById('decisionAppId').value = appData.id;  
  
// Enable forms  
document.getElementById('uploadDocBtn').disabled = false;  
document.getElementById('scheduleBtn').disabled = false;  
document.getElementById('admitBtn').disabled = false;  
document.getElementById('rejectBtn').disabled = false;  
  
renderDocumentVerification(appData.id, appData.application\_data);  
  
// Switch to decision tab  
const decisionTabBtn = document.getElementById('decision-tab');  
const bsTab = new bootstrap.Tab(decisionTabBtn);  
bsTab.show();  
}  
});  
  
// --- 2. DECISION WORKFLOW LOGIC ---  
  
function renderDocumentVerification(appId, appData) {  
const area = document.getElementById('documentVerificationArea');  
const requiredDocs = globalAdmissionConfig.requiredDocs || [];  
const documentMap = {};  
  
requiredDocs.forEach(docType => {  
documentMap[docType] = appData?.application\_data?.documents?.[docType] || {};  
});  
  
let html = '<ul class="list-group">';  
  
if (Object.keys(documentMap).length === 0) {  
html = '<p class="alert alert-warning">No mandatory documents defined in configuration.</p>';  
} else {  
for (const docType in documentMap) {  
const doc = documentMap[docType];  
const uploaded = doc.uploaded || false;  
const verified = doc.verified || false;  
const statusText = uploaded ? (verified ? 'Verified' : 'Review Required') : 'Missing';  
const statusClass = verified ? 'text-success' : (uploaded ? 'text-warning' : 'text-danger');  
  
html += `  
<li class="list-group-item d-flex justify-content-between align-items-center">  
${docType.replace(/\_/g, ' ').toUpperCase()}  
<span class="${statusClass} small fw-bold">${statusText}</span>  
<div>  
${uploaded ? `<button class="btn btn-sm btn-outline-success verify-doc-btn" data-id="${appId}" data-type="${docType}" data-verified="true" ${verified ? 'disabled' : ''}>Verify</button> ` : ''}  
${uploaded && !verified ? `<button class="btn btn-sm btn-outline-danger verify-doc-btn" data-id="${appId}" data-type="${docType}" data-verified="false">Reject</button>` : ''}  
${uploaded && doc.path ? `<a href="${doc.path}" target="\_blank" class="btn btn-sm btn-info ms-2"><i class="bi bi-eye"></i> View</a>` : ''}  
</div>  
</li>  
`;  
}  
html += '</ul>';  
}  
area.innerHTML = html;  
}  
  
// POST Document Upload  
document.getElementById('documentUploadForm').addEventListener('submit', async (e) => {  
e.preventDefault();  
const form = e.target;  
const appId = document.getElementById('uploadAppId').value;  
const docType = document.getElementById('documentTypeSelect').value;  
  
const formData = new FormData(form);  
formData.append('documentType', docType);  
  
const btn = document.getElementById('uploadDocBtn');  
btn.disabled = true;  
  
try {  
const data = await apiFetch(`/applications/${appId}/upload-document`, 'POST', formData, true);  
showNotification(data.message, 'success');  
fetchApplications(); // Refresh table to get updated app data  
} catch {  
// Error handled by apiFetch  
} finally {  
btn.disabled = false;  
}  
});  
  
// POST Document Verification  
document.getElementById('documentVerificationArea').addEventListener('click', async (e) => {  
const button = e.target.closest('.verify-doc-btn');  
if (button) {  
const appId = button.getAttribute('data-id');  
const docType = button.getAttribute('data-type');  
const isVerified = button.getAttribute('data-verified') === 'true';  
  
try {  
const data = await apiFetch(`/applications/${appId}/verify-document`, 'POST', { documentType: docType, isVerified: isVerified });  
showNotification(data.message, 'success');  
fetchApplications(); // Refresh list to get updated status  
} catch {  
// Error handled by apiFetch  
}  
}  
});  
  
// POST Schedule Assessment/Interview  
document.getElementById('scheduleForm').addEventListener('submit', async (e) => {  
e.preventDefault();  
const form = e.target;  
const appId = document.getElementById('scheduleAppId').value;  
  
const payload = {  
scheduleType: form.scheduleType.value,  
date: form.scheduleDate.value,  
time: form.scheduleTime.value  
};  
  
const btn = document.getElementById('scheduleBtn');  
btn.disabled = true;  
  
try {  
const data = await apiFetch(`/applications/${appId}/schedule`, 'POST', payload);  
showNotification(data.message, 'success');  
fetchApplications(); // Refresh list to see status change  
} catch {  
// Error handled by apiFetch  
} finally {  
btn.disabled = false;  
}  
});  
  
// POST Final Admission Decision (Admit/Reject)  
document.getElementById('decisionForm').addEventListener('click', async (e) => {  
const decisionBtn = e.target.closest('button[data-decision]');  
if (!decisionBtn) return;  
  
const decision = decisionBtn.getAttribute('data-decision');  
const appId = document.getElementById('decisionAppId').value;  
const allocatedClass = document.getElementById('decisionClass').value;  
const allocatedSection = document.getElementById('decisionSection').value;  
  
if (decision === 'admitted' && (!allocatedClass || !allocatedSection)) {  
showNotification("Allocated Class and Section are required for admission.", 'warning');  
return;  
}  
  
if (!customConfirm(`Confirm final decision: ${decision.toUpperCase()}? This action is permanent and creates a student record if Admitted.`)) return;  
  
decisionBtn.disabled = true;  
  
const payload = { decision, allocatedClass, allocatedSection };  
  
try {  
const data = await apiFetch(`/applications/${appId}/admission-decision`, 'POST', payload);  
showNotification(data.message, 'success');  
fetchApplications(); // Refresh list  
} catch {  
// Error handled by apiFetch  
} finally {  
decisionBtn.disabled = false;  
}  
});  
  
  
// --- 3. STATISTICS & BULK OPERATIONS LOGIC ---  
  
async function fetchStatistics() {  
try {  
const data = await apiFetch('/statistics');  
const summary = data.summary;  
  
document.getElementById('statTotal').textContent = summary.totalApplications;  
document.getElementById('statAdmitted').textContent = summary.admitted;  
document.getElementById('statConversion').textContent = summary.conversionRate;  
  
// Update inline review stats summary (mocking a simple view here)  
document.getElementById('review-stats-summary').innerHTML = `  
<div class="col-md-3"><div class="p-2 border rounded bg-light"><strong>Total:</strong> ${summary.totalApplications}</div></div>  
<div class="col-md-3"><div class="p-2 border rounded status-admitted"><strong>Admitted:</strong> ${summary.admitted}</div></div>  
<div class="col-md-3"><div class="p-2 border rounded status-rejected"><strong>Rejected:</strong> ${summary.rejected}</div></div>  
<div class="col-md-3"><div class="p-2 border rounded status-applied"><strong>Pending:</strong> ${summary.totalApplications - summary.admitted - summary.rejected}</div></div>  
`;  
  
} catch (error) {  
console.error("Stats fetch error:", error);  
}  
}  
  
// POST Bulk Upload  
document.getElementById('bulkUploadForm').addEventListener('submit', async (e) => {  
e.preventDefault();  
const form = e.target;  
const formData = new FormData(form);  
const btn = document.getElementById('bulkUploadBtn');  
btn.disabled = true;  
  
try {  
const data = await apiFetch('/bulk/upload', 'POST', formData, true);  
showNotification(data.message, 'success');  
form.reset();  
} catch {  
// Error handled by apiFetch  
} finally {  
btn.disabled = false;  
}  
});  
  
// GET Bulk Download  
document.getElementById('bulkDownloadBtn').addEventListener('click', async () => {  
const btn = document.getElementById('bulkDownloadBtn');  
const originalText = btn.innerHTML;  
btn.disabled = true;  
btn.innerHTML = '<span class="spinner-border spinner-border-sm"></span> Initiating...';  
  
try {  
const data = await apiFetch('/bulk/download');  
showNotification(data.message, 'info');  
} catch {  
// Error handled by apiFetch  
} finally {  
btn.disabled = false;  
btn.innerHTML = originalText;  
}  
});  
  
// --- 5. PUBLIC STATUS CHECK LOGIC ---  
  
document.getElementById('statusCheckForm').addEventListener('submit', async (e) => {  
e.preventDefault();  
const appNumber = document.getElementById('checkAppNumber').value;  
const resultArea = document.getElementById('portalResultArea');  
const statusDisplay = document.getElementById('portalStatusDisplay');  
const historyList = document.getElementById('portalHistoryList');  
  
resultArea.classList.add('bg-light');  
statusDisplay.textContent = 'Searching...';  
historyList.innerHTML = '';  
  
try {  
const data = await apiFetch(`/portal/${appNumber}`);  
  
const currentStatus = data.status || 'N/A';  
const applicantName = data.details?.studentName || 'Applicant Details Unavailable';  
  
statusDisplay.innerHTML = `Current Status: ${getStatusBadge(currentStatus)}`;  
document.getElementById('portalApplicantName').textContent = `Applicant: ${applicantName}`;  
historyList.innerHTML = formatHistory(data.history);  
  
if (currentStatus === 'admitted') {  
resultArea.classList.remove('bg-light');  
resultArea.style.backgroundColor = '#ecf8e9'; // Light green for admitted  
} else {  
resultArea.style.backgroundColor = '';  
}  
  
} catch (error) {  
statusDisplay.innerHTML = `<span class="badge bg-danger">Error / Not Found</span>`;  
document.getElementById('portalApplicantName').textContent = `Error: ${error.message}`;  
historyList.innerHTML = '<p class="text-muted">Could not retrieve application history.</p>';  
}  
});  
  
  
// --- Initialization ---  
document.addEventListener('DOMContentLoaded', () => {  
// Set default date/time for scheduling form  
const today = new Date().toISOString().split('T')[0];  
document.getElementById('scheduleDate').value = today;  
document.getElementById('scheduleTime').value = '10:00';  
  
fetchConfig();  
fetchApplications();  
});  
</script>  
</body>  
</html>

9. client/dist/modules/Arattai.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>📱 Arattai Integration Dashboard</title>

<!-- Bootstrap 5 CDN -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css">

<style>

body { background-color: #e8f5e9; font-family: 'Inter', sans-serif; }

.container-fluid { max-width: 1300px; margin-top: 20px; }

.card-header { background-color: #38a169; color: white; font-weight: bold; border-radius: 0.5rem 0.5rem 0 0; }

.section-card { margin-bottom: 25px; box-shadow: 0 4px 10px rgba(0, 50, 0, 0.1); border-radius: 0.5rem; }

.alert-fixed {

position: fixed; top: 20px; right: 20px; z-index: 1050; min-width: 300px; border-radius: 0.5rem;

}

.nav-tabs .nav-link.active {

background-color: #38a169;

color: white;

border-color: #38a169;

}

.stat-box { transition: transform 0.2s; cursor: default; }

.stat-box:hover { transform: translateY(-3px); box-shadow: 0 4px 8px rgba(0, 0, 0, 0.15); }

</style>

</head>

<body>

<div class="container-fluid">

<h1 class="text-center mb-4 text-success"><i class="bi bi-whatsapp"></i> Arattai Communication Hub</h1>

<!-- Notification Area -->

<div id="notification-area" class="alert-fixed" style="display: none;"></div>

<!-- Tab Navigation -->

<ul class="nav nav-tabs" id="arattaiTab" role="tablist">

<li class="nav-item"><button class="nav-link active" id="analytics-tab" data-bs-toggle="tab" data-bs-target="#analytics" type="button"><i class="bi bi-speedometer2"></i> Dashboard & Analytics</button></li>

<li class="nav-item"><button class="nav-link" id="composer-tab" data-bs-toggle="tab" data-bs-target="#composer" type="button"><i class="bi bi-send"></i> Message Composer & Schedule</button></li>

<li class="nav-item"><button class="nav-link" id="broadcast-tab" data-bs-toggle="tab" data-bs-target="#broadcast" type="button"><i class="bi bi-megaphone"></i> Broadcast Campaigns</button></li>

<li class="nav-item"><button class="nav-link" id="automation-tab" data-bs-toggle="tab" data-bs-target="#automation" type="button"><i class="bi bi-robot"></i> Automation Rules</button></li>

<li class="nav-item"><button class="nav-link" id="contacts-tab" data-bs-toggle="tab" data-bs-target="#contacts" type="button"><i class="bi bi-people"></i> Contacts & Preferences</button></li>

<li class="nav-item"><button class="nav-link" id="config-tab" data-bs-toggle="tab" data-bs-target="#config" type="button"><i class="bi bi-gear"></i> System Configuration</button></li>

</ul>

<!-- Tab Content -->

<div class="tab-content pt-3" id="arattaiTabContent">

<!-- 1. Dashboard & Analytics Tab -->

<div class="tab-pane fade show active" id="analytics" role="tabpanel">

<div class="card section-card">

<div class="card-header">Communication Analytics & Reporting</div>

<div class="card-body">

<div class="row mb-4">

<div class="col-md-3"><button class="btn btn-outline-success w-100" id="refreshAnalyticsBtn"><i class="bi bi-arrow-clockwise"></i> Refresh Analytics</button></div>

<div class="col-md-9"><p class="alert alert-info py-2 m-0 small">Data below is simulated to represent the aggregation performed by the backend's `/analytics` endpoint, including real-time status distribution and cost calculation.</p></div>

</div>

<!-- Key Stats Row -->

<div class="row text-center g-3 mb-4" id="keyStatsArea">

<!-- Stats dynamically loaded -->

</div>

<div class="row g-4">

<!-- Category Breakdown -->

<div class="col-md-6">

<div class="card card-body h-100">

<h6>Message Breakdown by Category</h6>

<ul class="list-group list-group-flush" id="categoryBreakdownList"></ul>

</div>

</div>

<!-- Top Templates -->

<div class="col-md-6">

<div class="card card-body h-100">

<h6>Top 5 Templates by Read Rate</h6>

<table class="table table-sm table-striped">

<thead><tr><th>Template ID</th><th>Sent</th><th>Delivered Rate</th><th>Read Rate</th></tr></thead>

<tbody id="topTemplatesTableBody"></tbody>

</table>

</div>

</div>

</div>

</div>

</div>

</div>

<!-- 2. Message Composer & Schedule Tab -->

<div class="tab-pane fade" id="composer" role="tabpanel">

<div class="card section-card">

<div class="card-header">Immediate & Scheduled Messaging</div>

<div class="card-body">

<!-- Immediate Send Form (POST /send) -->

<h5 class="mb-3">New Message Composer</h5>

<form id="sendMessageForm" class="row g-3 border p-3 rounded mb-4">

<div class="col-md-4">

<label for="sendTemplateId" class="form-label">Template</label>

<select class="form-select" id="sendTemplateId" required>

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

<!-- Templates loaded dynamically -->

</select>

</div>

<div class="col-md-4">

<label for="recipientNumber" class="form-label">Recipient Phone</label>

<input type="text" class="form-control" id="recipientNumber" placeholder="+91..." required>

</div>

<div class="col-md-4">

<label for="sendVariables" class="form-label">Variables (JSON)</label>

<input type="text" class="form-control" id="sendVariables" placeholder='{"studentname": "Aarav"}' required>

<div class="form-text" id="templateVariablesHint">Variables required by template: N/A</div>

</div>

<div class="col-md-8">

<label for="scheduledFor" class="form-label">Scheduled For (Optional)</label>

<input type="datetime-local" class="form-control" id="scheduledFor">

</div>

<div class="col-md-4 d-grid">

<label class="form-label d-none d-md-block">&nbsp;</label>

<button type="submit" class="btn btn-success" id="sendMsgBtn"><i class="bi bi-arrow-right-circle"></i> Send/Schedule Message</button>

</div>

</form>

<hr>

<!-- Scheduled Messages List (GET /scheduled, DELETE /scheduled/:id) -->

<h5 class="mb-3">Active Scheduled Messages</h5>

<button class="btn btn-sm btn-outline-info mb-3" id="refreshScheduledBtn"><i class="bi bi-arrow-clockwise"></i> Refresh List</button>

<div class="table-responsive">

<table class="table table-sm table-striped">

<thead><tr><th>ID</th><th>Recipient</th><th>Scheduled Time</th><th>Category</th><th>Message Preview</th><th>Action</th></tr></thead>

<tbody id="scheduledMessagesTableBody"><tr><td colspan="6" class="text-center">No messages scheduled.</td></tr></tbody>

</table>

</div>

</div>

</div>

</div>

<!-- 3. Broadcast Campaigns Tab -->

<div class="tab-pane fade" id="broadcast" role="tabpanel">

<div class="card section-card">

<div class="card-header">Bulk Broadcast Management</div>

<div class="card-body">

<h5 class="mb-3">Create New Broadcast Campaign (POST /broadcast)</h5>

<form id="broadcastForm" class="row g-3 border p-3 rounded mb-4">

<div class="col-md-6">

<label for="campaignName" class="form-label">Campaign Name</label>

<input type="text" class="form-control" id="campaignName" required>

</div>

<div class="col-md-6">

<label for="broadcastTemplateId" class="form-label">Template</label>

<select class="form-select" id="broadcastTemplateId" required>

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

</select>

</div>

<div class="col-md-6">

<label for="targetAudience" class="form-label">Target Audience</label>

<select class="form-select" id="targetAudience" required>

<option value="all\_parents">All Parents (System Default)</option>

<option value="fee\_pending\_parents">Fee Pending Parents (System Default)</option>

<option value="custom\_list">Custom Phone List</option>

</select>

</div>

<div class="col-md-6">

<label for="recipientList" class="form-label">Custom Recipient List (Comma Separated)</label>

<textarea class="form-control" id="recipientList" rows="2" placeholder="+91..., +91..."></textarea>

<div class="form-text">Required only if 'Custom Phone List' is selected.</div>

</div>

<div class="col-md-6">

<label for="broadcastScheduledFor" class="form-label">Scheduled Time (Leave blank for immediate)</label>

<input type="datetime-local" class="form-control" id="broadcastScheduledFor">

</div>

<div class="col-12">

<button type="submit" class="btn btn-primary" id="createBroadcastBtn"><i class="bi bi-broadcast"></i> Initiate Broadcast</button>

</div>

</form>

<hr>

<h5 class="mb-3">Current & Past Campaigns</h5>

<div class="table-responsive">

<table class="table table-sm table-striped">

<thead><tr><th>Name</th><th>Status</th><th>Total Sent</th><th>Delivery Rate</th><th>Read Rate</th><th>Cost</th></tr></thead>

<tbody id="broadcastCampaignsTableBody"><tr><td colspan="6" class="text-center">No campaigns recorded.</td></tr></tbody>

</table>

</div>

</div>

</div>

</div>

<!-- 4. Automation Rules Tab -->

<div class="tab-pane fade" id="automation" role="tabpanel">

<div class="card section-card">

<div class="card-header">Event-Triggered Automation</div>

<div class="card-body">

<h5 class="mb-3">Create/Update Automation Rule (POST/PUT /automation)</h5>

<form id="automationForm" class="row g-3 border p-3 rounded mb-4">

<input type="hidden" id="automationRuleId">

<div class="col-md-4">

<label for="ruleName" class="form-label">Rule Name</label>

<input type="text" class="form-control" id="ruleName" required>

</div>

<div class="col-md-4">

<label for="triggerEvent" class="form-label">Trigger Event</label>

<select class="form-select" id="triggerEvent" required>

<option value="attendance\_absent">Attendance: Student Absent</option>

<option value="fee\_due">Finance: Fee Due Date Hit</option>

<option value="transport\_onboard">Transport: Onboard Scan</option>

<option value="exam\_schedule\_release">Academics: Exam Schedule Release</option>

</select>

</div>

<div class="col-md-4">

<label for="automationTemplateId" class="form-label">Template</label>

<select class="form-select" id="automationTemplateId" required>

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

</select>

</div>

<div class="col-md-10">

<label for="ruleConditions" class="form-label">Conditions (JSON Array)</label>

<textarea class="form-control" id="ruleConditions" rows="2" placeholder='[{"field": "class\_name", "operator": "eq", "value": "10-A"}]'></textarea>

<div class="form-text">e.g., Only run for 'Class 10-A' students.</div>

</div>

<div class="col-md-2 d-flex align-items-end">

<div class="form-check form-switch w-100">

<input class="form-check-input" type="checkbox" id="ruleActive" checked>

<label class="form-check-label" for="ruleActive">Active</label>

</div>

</div>

<div class="col-12">

<button type="submit" class="btn btn-success" id="saveAutomationRuleBtn"><i class="bi bi-save"></i> Save Rule</button>

<button type="button" class="btn btn-outline-danger" id="cancelEditAutomationBtn" style="display: none;">Cancel Edit</button>

</div>

</form>

<hr>

<h5 class="mb-3">Existing Automation Rules (GET /automation)</h5>

<div class="table-responsive">

<table class="table table-sm table-striped">

<thead><tr><th>Rule Name</th><th>Trigger</th><th>Template ID</th><th>Status</th><th>Action</th></tr></thead>

<tbody id="automationRulesTableBody"><tr><td colspan="5" class="text-center">No automation rules defined.</td></tr></tbody>

</table>

</div>

</div>

</div>

</div>

<!-- 5. Contacts & Preferences Tab -->

<div class="tab-pane fade" id="contacts" role="tabpanel">

<div class="card section-card">

<div class="card-header">Contact Management & Opt-Outs</div>

<div class="card-body">

<div class="row mb-3">

<div class="col-md-8">

<input type="text" class="form-control" id="contactSearch" placeholder="Search by Student/Parent Name or Phone...">

</div>

<div class="col-md-4 d-flex justify-content-end">

<button class="btn btn-outline-primary me-2" id="searchContactsBtn"><i class="bi bi-search"></i> Search</button>

<button class="btn btn-success" data-bs-toggle="modal" data-bs-target="#contactModal"><i class="bi bi-plus-circle"></i> Add Contact</button>

</div>

</div>

<div class="table-responsive">

<table class="table table-sm table-hover">

<thead><tr><th>Student/Parent</th><th>Phone</th><th>Verified</th><th>Consent</th><th>Preferences</th><th>Action</th></tr></thead>

<tbody id="contactsTableBody"><tr><td colspan="6" class="text-center">Loading contacts...</td></tr></tbody>

</table>

</div>

<!-- Opt-Out Section (POST /optout, GET /optout/:phone) -->

<h5 class="mt-4">Parent Opt-Out Management</h5>

<form id="optOutForm" class="input-group mb-3" style="max-width: 400px;">

<input type="text" class="form-control" id="optOutPhone" placeholder="Enter phone number to Opt-Out (+91...)" required>

<button class="btn btn-danger" type="submit" id="optOutBtn"><i class="bi bi-slash-circle"></i> Permanently Opt-Out</button>

</form>

<div id="optOutStatus"></div>

</div>

</div>

</div>

<!-- 6. System Configuration Tab (Comprehensive Setup) -->

<div class="tab-pane fade" id="config" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-secondary">Global Arattai Settings & Governance</div>

<div class="card-body">

<form id="systemConfigForm">

<h5 class="mt-4">General Settings & Business Info</h5>

<div class="row g-3 mb-4 border p-3 rounded">

<div class="col-md-4">

<label for="businessNumber" class="form-label">Business Phone Number</label>

<input type="text" class="form-control" id="businessNumber" required>

</div>

<div class="col-md-4">

<label for="businessName" class="form-label">Business Name (School)</label>

<input type="text" class="form-control" id="businessName" required>

</div>

<div class="col-md-4">

<label for="apiProvider" class="form-label">API Provider</label>

<input type="text" class="form-control" id="apiProvider" disabled>

</div>

</div>

<h5 class="mt-4">Usage Limits & Quota</h5>

<div class="row g-3 mb-4 border p-3 rounded">

<div class="col-md-4">

<label for="dailyLimit" class="form-label">Daily Message Limit</label>

<input type="number" class="form-control" id="dailyLimit" required min="100">

</div>

<div class="col-md-4">

<label for="monthlyLimit" class="form-label">Monthly Message Limit</label>

<input type="number" class="form-control" id="monthlyLimit" required min="1000">

</div>

<div class="col-md-4">

<label for="currentUsage" class="form-label">Current Usage (Monthly)</label>

<input type="text" class="form-control" id="currentUsage" disabled>

<div class="form-text">Remaining Quota: <span id="remainingQuotaText">...</span></div>

</div>

</div>

<h5 class="mt-4">Rate Limiting Controls</h5>

<div class="row g-3 mb-4 border p-3 rounded">

<div class="col-md-6">

<label for="messagesPerMinute" class="form-label">Messages Per Minute (API Level)</label>

<input type="number" class="form-control" id="messagesPerMinute" required min="1">

</div>

<div class="col-md-6">

<label for="messagesPerHour" class="form-label">Messages Per Hour (System Level)</label>

<input type="number" class="form-control" id="messagesPerHour" required min="1">

</div>

</div>

<h5 class="mt-4">Governance & Webhook Setup</h5>

<div class="row g-3 mb-4 border p-3 rounded">

<div class="col-md-6">

<div class="form-check form-switch">

<input class="form-check-input" type="checkbox" id="consentRequired">

<label class="form-check-label" for="consentRequired">Require Parent Consent before messaging</label>

</div>

<div class="form-check form-switch">

<input class="form-check-input" type="checkbox" id="automaticOptOut">

<label class="form-check-label" for="automaticOptOut">Automatic Opt-Out Handling</label>

</div>

<div class="form-check form-switch">

<input class="form-check-input" type="checkbox" id="contentFiltering">

<label class="form-check-label" for="contentFiltering">Enable Content Filtering</label>

</div>

<div class="form-check form-switch">

<input class="form-check-input" type="checkbox" id="deliveryReports">

<label class="form-check-label" for="deliveryReports">Enable Delivery Reports</label>

</div>

<div class="form-check form-switch">

<input class="form-check-input" type="checkbox" id="readReceipts">

<label class="form-check-label" for="readReceipts">Enable Read Receipts</label>

</div>

</div>

<div class="col-md-6">

<label for="webhookUrl" class="form-label">Webhook URL (For Delivery/Read Status)</label>

<input type="url" class="form-control" id="webhookUrl" required>

<label for="webhookSecret" class="form-label mt-2">Webhook Secret</label>

<input type="text" class="form-control" id="webhookSecret" required>

<label for="dataRetentionDays" class="form-label mt-2">Data Retention Days</label>

<input type="number" class="form-control" id="dataRetentionDays" required min="30">

</div>

</div>

<button type="submit" class="btn btn-secondary mt-3" id="saveSystemConfigBtn"><i class="bi bi-save"></i> Save Global Settings</button>

</form>

</div>

</div>

</div>

</div>

</div>

<!-- Contact Modal for Add/Edit -->

<div class="modal fade" id="contactModal" tabindex="-1">

<div class="modal-dialog">

<div class="modal-content">

<div class="modal-header">

<h5 class="modal-title" id="contactModalTitle">Add New Contact</h5>

<button type="button" class="btn-close" data-bs-dismiss="modal"></button>

</div>

<form id="contactForm">

<div class="modal-body">

<input type="hidden" id="contactId">

<div class="mb-3">

<label for="studentId" class="form-label">Student ID</label>

<input type="text" class="form-control" id="studentId" placeholder="e.g., STUDENT\_001" required>

</div>

<div class="mb-3">

<label for="parentName" class="form-label">Parent Name</label>

<input type="text" class="form-control" id="parentName" required>

</div>

<div class="mb-3">

<label for="phoneNumber" class="form-label">Phone Number</label>

<input type="text" class="form-control" id="phoneNumber" placeholder="+91..." required>

</div>

<div class="mb-3">

<label for="preferredLanguage" class="form-label">Preferred Language</label>

<select class="form-select" id="preferredLanguage">

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

<option value="tamil">Tamil</option>

<option value="hindi">Hindi</option>

</select>

</div>

<div class="form-check mb-3">

<input class="form-check-input" type="checkbox" id="consentGiven">

<label class="form-check-label" for="consentGiven">Consent Given</label>

</div>

<div class="form-check mb-3">

<input class="form-check-input" type="checkbox" id="arattaiVerified">

<label class="form-check-label" for="arattaiVerified">Arattai Verified</label>

</div>

<div class="mb-3">

<label class="form-label">Notification Preferences</label>

<div id="preferenceCheckboxes">

<div class="form-check form-check-inline"><input class="form-check-input" type="checkbox" name="pref" value="attendance" checked><label class="form-check-label">Attendance</label></div>

<div class="form-check form-check-inline"><input class="form-check-input" type="checkbox" name="pref" value="academic" checked><label class="form-check-label">Academic</label></div>

<div class="form-check form-check-inline"><input class="form-check-input" type="checkbox" name="pref" value="events" checked><label class="form-check-label">Events</label></div>

</div>

</div>

</div>

<div class="modal-footer">

<button type="button" class="btn btn-secondary" data-bs-dismiss="modal">Close</button>

<button type="submit" class="btn btn-primary" id="saveContactBtn">Save Contact</button>

</div>

</form>

</div>

</div>

</div>

<!-- Bootstrap Bundle with Popper -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>

<script>

const API\_BASE = ''; // Base path for the API

let globalTemplates = [];

let globalContacts = [];

/\*\* Helper function to show a temporary notification. \*/

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

const area = document.getElementById('notification-area');

area.innerHTML = `

<div class="alert alert-${type} alert-dismissible fade show" role="alert">

${message}

<button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>

</div>

`;

area.style.display = 'block';

setTimeout(() => {

area.style.display = 'none';

}, 7000);

}

/\*\* Core Fetch Function with Error Handling. \*/

async function apiFetch(endpoint, method = 'GET', body = null) {

try {

const options = {

method: method,

headers: { 'Content-Type': 'application/json' },

};

if (body && method !== 'GET') {

options.body = JSON.stringify(body);

}

const response = await fetch(`${API\_BASE}${endpoint}`, options);

const data = await response.json();

if (!response.ok) {

throw new Error(data.error || `API error (${response.status})`);

}

return data;

} catch (error) {

console.error('API Fetch Error:', endpoint, error);

showNotification(error.message || 'Network error.', 'danger');

throw error;

}

}

// --- TEMPLATE LOGIC ---

async function fetchTemplates() {

try {

const data = await apiFetch('/templates');

globalTemplates = data;

const selectIds = ['sendTemplateId', 'broadcastTemplateId', 'automationTemplateId'];

selectIds.forEach(selectId => {

const select = document.getElementById(selectId);

select.innerHTML = '<option value="">Select Template</option>';

data.forEach(t => {

select.innerHTML += `<option value="${t.id}" data-variables='${JSON.stringify(t.variables || [])}'>${t.name} (${t.id})</option>`;

});

});

// Set listener for Composer variables hint

document.getElementById('sendTemplateId').addEventListener('change', updateComposerHint);

} catch (error) {

console.error("Failed to load templates:", error);

showNotification("Failed to load templates for composer.", 'danger');

}

}

function updateComposerHint(e) {

const select = e.target;

const hint = document.getElementById('templateVariablesHint');

const selectedOption = select.options[select.selectedIndex];

if (selectedOption && selectedOption.value) {

try {

const variables = JSON.parse(selectedOption.getAttribute('data-variables'));

hint.textContent = `Variables required: {${variables.join('}, {')}}`;

} catch {

hint.textContent = 'Variables required: N/A';

}

} else {

hint.textContent = 'Variables required: N/A';

}

}

// --- 1. ANALYTICS & DASHBOARD LOGIC (MOCKED DATA) ---

async function fetchAnalytics() {

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

keyStatsArea.innerHTML = '<div class="col-12 text-center text-muted">Fetching analysis...</div>';

// MOCK DATA structure reflecting the backend's expected output

const mockAnalyticsData = {

totalMessagesSent: 1500,

deliveryRate: '98.5',

readRate: '82.3',

totalCost: 750.00,

avgCostPerMessage: 0.50,

categoryBreakdown: { attendance: 600, academic: 400, events: 300, finance: 200 },

topPerformingTemplates: [

{ templateId: 'template\_001', sent: 600, deliveryRate: '99.0', readRate: '88.5' },

{ templateId: 'fee\_alert', sent: 200, deliveryRate: '98.0', readRate: '85.1' },

]

};

// NOTE: In a real environment, you would use:

// const data = await apiFetch('/analytics');

// const analytics = data;

const analytics = mockAnalyticsData;

keyStatsArea.innerHTML = `

<div class="col-md-3"><div class="p-3 border rounded bg-light stat-box"><h4 class="text-primary">${analytics.totalMessagesSent.toLocaleString()}</h4><p class="text-muted">Total Messages Sent</p></div></div>

<div class="col-md-3"><div class="p-3 border rounded bg-light stat-box"><h4 class="text-success">${analytics.deliveryRate}%</h4><p class="text-muted">Delivery Rate</p></div></div>

<div class="col-md-3"><div class="p-3 border rounded bg-light stat-box"><h4 class="text-info">${analytics.readRate}%</h4><p class="text-muted">Read Rate</p></div></div>

<div class="col-md-3"><div class="p-3 border rounded bg-light stat-box"><h4 class="text-danger">₹${analytics.totalCost.toFixed(2)}</h4><p class="text-muted">Total Messaging Cost</p></div></div>

`;

// Category Breakdown

const breakdownList = document.getElementById('categoryBreakdownList');

breakdownList.innerHTML = '';

Object.entries(analytics.categoryBreakdown).forEach(([category, count]) => {

breakdownList.innerHTML += `<li class="list-group-item d-flex justify-content-between align-items-center">${category.toUpperCase()} <span class="badge bg-primary rounded-pill">${count}</span></li>`;

});

// Top Templates

const templatesTable = document.getElementById('topTemplatesTableBody');

templatesTable.innerHTML = '';

analytics.topPerformingTemplates.forEach(t => {

templatesTable.innerHTML += `<tr><td>${t.templateId}</td><td>${t.sent}</td><td>${t.deliveryRate}%</td><td>${t.readRate}%</td></tr>`;

});

document.getElementById('refreshAnalyticsBtn').addEventListener('click', fetchAnalytics);

}

// --- 2. MESSAGE COMPOSER LOGIC ---

document.getElementById('sendMessageForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const btn = document.getElementById('sendMsgBtn');

btn.disabled = true;

const isScheduled = form.scheduledFor.value !== '';

let variables;

try {

variables = JSON.parse(form.sendVariables.value);

} catch {

showNotification("Variables must be valid JSON object.", 'warning');

btn.disabled = false;

return;

}

const payload = {

templateId: form.sendTemplateId.value,

recipientNumber: form.recipientNumber.value,

variables: variables,

scheduledFor: isScheduled ? new Date(form.scheduledFor.value).toISOString() : undefined

};

const endpoint = isScheduled ? '/schedule' : '/send';

try {

const data = await apiFetch(endpoint, 'POST', payload);

showNotification(data.message || `Message successfully ${isScheduled ? 'scheduled.' : 'sent.'}`, 'success');

form.reset();

if (isScheduled) fetchScheduledMessages();

} catch (error) {

// Handled by apiFetch

} finally {

btn.disabled = false;

}

});

// Fetch Scheduled Messages

async function fetchScheduledMessages() {

const tableBody = document.getElementById('scheduledMessagesTableBody');

tableBody.innerHTML = '<tr><td colspan="6" class="text-center"><div class="spinner-border spinner-border-sm"></div> Loading scheduled messages...</td></tr>';

try {

const messages = await apiFetch('/scheduled');

tableBody.innerHTML = '';

if (messages.length === 0) {

tableBody.innerHTML = '<tr><td colspan="6" class="text-center">No messages scheduled.</td></tr>';

return;

}

messages.forEach(msg => {

const scheduledTime = new Date(msg.scheduledFor).toLocaleString();

const messagePreview = msg.message.substring(0, 50) + '...';

const row = tableBody.insertRow();

row.innerHTML = `

<td>${msg.id.substring(0, 10)}...</td>

<td>${msg.recipientNumber} (${msg.recipientName || 'N/A'})</td>

<td>${scheduledTime}</td>

<td><span class="badge bg-secondary">${msg.category.toUpperCase()}</span></td>

<td>${messagePreview}</td>

<td><button class="btn btn-sm btn-danger cancel-schedule-btn" data-id="${msg.id}" ${msg.status !== 'scheduled' ? 'disabled' : ''}><i class="bi bi-x-circle"></i> Cancel</button></td>

`;

});

} catch (error) {

tableBody.innerHTML = '<tr><td colspan="6" class="text-center text-danger">Failed to load scheduled messages.</td></tr>';

}

}

// Cancel Scheduled Message

document.getElementById('scheduledMessagesTableBody').addEventListener('click', async (e) => {

const btn = e.target.closest('.cancel-schedule-btn');

if (btn) {

const messageId = btn.getAttribute('data-id');

btn.disabled = true;

btn.innerHTML = '<span class="spinner-border spinner-border-sm"></span>';

try {

const data = await apiFetch(`/scheduled/${messageId}`, 'DELETE');

showNotification(data.message, 'success');

fetchScheduledMessages();

} catch {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-x-circle"></i> Cancel';

}

}

});

// --- 3. BROADCAST LOGIC ---

async function fetchBroadcastCampaigns() {

// NOTE: The backend only stores data in memory, so we simulate a fetch here

const mockCampaigns = [{

id: 'campaign\_1', name: 'Fee Reminder Q3', status: 'completed', totalRecipients: 500, sentCount: 500,

deliveredCount: 490, readCount: 400, cost: 250.00

},

{

id: 'campaign\_2', name: 'School Day Invite', status: 'processing', totalRecipients: 1200, sentCount: 800,

deliveredCount: 780, readCount: 0, cost: 400.00

}];

const tableBody = document.getElementById('broadcastCampaignsTableBody');

tableBody.innerHTML = '';

mockCampaigns.forEach(c => {

const deliveryRate = c.sentCount > 0 ? ((c.deliveredCount / c.sentCount) \* 100).toFixed(1) : '0.0';

const readRate = c.deliveredCount > 0 ? ((c.readCount / c.deliveredCount) \* 100).toFixed(1) : '0.0';

const statusClass = c.status === 'completed' ? 'bg-success' : (c.status === 'processing' ? 'bg-warning text-dark' : 'bg-secondary');

tableBody.innerHTML += `

<tr>

<td>${c.name}</td>

<td><span class="badge ${statusClass}">${c.status.toUpperCase()}</span></td>

<td>${c.sentCount} / ${c.totalRecipients}</td>

<td>${deliveryRate}%</td>

<td>${readRate}%</td>

<td>₹${c.cost.toFixed(2)}</td>

</tr>

`;

});

// NOTE: In production, fetchBroadcastCampaigns would hit GET /broadcasts (needs implementation)

}

document.getElementById('broadcastForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const target = form.targetAudience.value;

const btn = document.getElementById('createBroadcastBtn');

btn.disabled = true;

let recipientList = [];

if (target === 'custom\_list') {

const list = form.recipientList.value.split(',').map(n => n.trim()).filter(n => n.length > 0);

if (list.length === 0) {

showNotification("Custom list is empty.", 'warning');

btn.disabled = false; return;

}

recipientList = list;

}

const payload = {

name: form.campaignName.value,

templateId: form.broadcastTemplateId.value,

targetAudience: target,

recipientList: recipientList,

scheduledFor: form.broadcastScheduledFor.value || undefined,

variables: {} // Simplified for the frontend demo

};

try {

const data = await apiFetch('/broadcast', 'POST', payload);

showNotification(data.message, 'success');

form.reset();

fetchBroadcastCampaigns();

} catch {

// Handled by apiFetch

} finally {

btn.disabled = false;

}

});

// --- 4. AUTOMATION LOGIC ---

async function fetchAutomationRules() {

const tableBody = document.getElementById('automationRulesTableBody');

tableBody.innerHTML = '<tr><td colspan="5" class="text-center"><div class="spinner-border spinner-border-sm"></div> Loading automation rules...</td></tr>';

try {

const rules = await apiFetch('/automation');

tableBody.innerHTML = '';

if (rules.length === 0) {

tableBody.innerHTML = '<tr><td colspan="5" class="text-center">No automation rules found.</td></tr>';

return;

}

rules.forEach(rule => {

const statusClass = rule.active ? 'bg-success' : 'bg-danger';

const row = tableBody.insertRow();

row.innerHTML = `

<td>${rule.name}</td>

<td>${rule.trigger.replace(/\_/g, ' ')}</td>

<td>${rule.templateId}</td>

<td><span class="badge ${statusClass}">${rule.active ? 'Active' : 'Inactive'}</span></td>

<td><button class="btn btn-sm btn-info edit-rule-btn me-2" data-rule='${JSON.stringify(rule)}'><i class="bi bi-pencil"></i> Edit</button></td>

`;

});

} catch (error) {

tableBody.innerHTML = '<tr><td colspan="5" class="text-center text-danger">Failed to load automation rules.</td></tr>';

}

}

document.getElementById('automationRulesTableBody').addEventListener('click', (e) => {

const btn = e.target.closest('.edit-rule-btn');

if (btn) {

const rule = JSON.parse(btn.getAttribute('data-rule'));

document.getElementById('automationRuleId').value = rule.id;

document.getElementById('ruleName').value = rule.name;

document.getElementById('triggerEvent').value = rule.trigger;

document.getElementById('automationTemplateId').value = rule.templateId;

document.getElementById('ruleConditions').value = JSON.stringify(rule.conditions, null, 2);

document.getElementById('ruleActive').checked = rule.active;

document.getElementById('saveAutomationRuleBtn').textContent = 'Update Rule';

document.getElementById('cancelEditAutomationBtn').style.display = 'inline-block';

}

});

document.getElementById('cancelEditAutomationBtn').addEventListener('click', () => {

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

document.getElementById('automationRuleId').value = '';

document.getElementById('saveAutomationRuleBtn').textContent = 'Save Rule';

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

});

document.getElementById('automationForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const ruleId = form.automationRuleId.value;

const btn = document.getElementById('saveAutomationRuleBtn');

btn.disabled = true;

let conditions;

try {

conditions = JSON.parse(form.ruleConditions.value);

if (!Array.isArray(conditions)) throw new Error("Conditions must be an array.");

} catch {

showNotification("Conditions must be a valid JSON array.", 'warning');

btn.disabled = false; return;

}

const payload = {

name: form.ruleName.value,

trigger: form.triggerEvent.value,

templateId: form.automationTemplateId.value,

conditions: conditions,

active: form.ruleActive.checked,

createdBy: 'admin\_user'

};

const method = ruleId ? 'PUT' : 'POST';

const endpoint = ruleId ? `/automation/${ruleId}` : '/automation';

try {

const data = await apiFetch(endpoint, method, payload);

showNotification(data.message || `Rule ${method === 'POST' ? 'created' : 'updated'} successfully.`, 'success');

form.reset();

document.getElementById('automationRuleId').value = '';

document.getElementById('saveAutomationRuleBtn').textContent = 'Save Rule';

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

fetchAutomationRules();

} catch {

// Handled by apiFetch

} finally {

btn.disabled = false;

}

});

// --- 5. CONTACTS & PREFERENCES LOGIC ---

async function fetchContacts() {

const tableBody = document.getElementById('contactsTableBody');

tableBody.innerHTML = '<tr><td colspan="6" class="text-center"><div class="spinner-border spinner-border-sm"></div> Loading contacts...</td></tr>';

const search = document.getElementById('contactSearch').value;

try {

// NOTE: Search functionality requires backend implementation. Frontend only uses base GET /contacts.

const contacts = await apiFetch('/contacts');

globalContacts = contacts;

tableBody.innerHTML = '';

if (contacts.length === 0) {

tableBody.innerHTML = '<tr><td colspan="6" class="text-center">No contacts found.</td></tr>';

return;

}

contacts.forEach(c => {

const verifiedBadge = c.arattaiVerified ? '<span class="badge bg-success">Yes</span>' : '<span class="badge bg-danger">No</span>';

const consentBadge = c.consentGiven ? '<span class="badge bg-primary">Yes</span>' : '<span class="badge bg-warning text-dark">No</span>';

const prefs = (c.notificationPreferences || []).map(p => p.substring(0, 1).toUpperCase()).join(', ');

const row = tableBody.insertRow();

row.innerHTML = `

<td>${c.studentName} / ${c.parentName}</td>

<td>${c.phoneNumber}</td>

<td>${verifiedBadge}</td>

<td>${consentBadge}</td>

<td>${prefs}</td>

<td><button class="btn btn-sm btn-warning edit-contact-btn" data-id="${c.id}"><i class="bi bi-pencil"></i> Edit</button></td>

`;

});

} catch (error) {

tableBody.innerHTML = '<tr><td colspan="6" class="text-center text-danger">Failed to load contacts.</td></tr>';

}

}

document.getElementById('searchContactsBtn').addEventListener('click', fetchContacts);

// Populate modal for editing

document.getElementById('contactsTableBody').addEventListener('click', (e) => {

const btn = e.target.closest('.edit-contact-btn');

if (btn) {

const contactId = btn.getAttribute('data-id');

const contact = globalContacts.find(c => c.id === contactId);

document.getElementById('contactModalTitle').textContent = `Edit Contact: ${contact.parentName}`;

document.getElementById('contactId').value = contact.id;

document.getElementById('studentId').value = contact.studentId || '';

document.getElementById('parentName').value = contact.parentName || '';

document.getElementById('phoneNumber').value = contact.phoneNumber || '';

document.getElementById('preferredLanguage').value = contact.preferredLanguage || 'english';

document.getElementById('consentGiven').checked = contact.consentGiven || false;

document.getElementById('arattaiVerified').checked = contact.arattaiVerified || false;

// Preferences

const preferences = contact.notificationPreferences || [];

document.querySelectorAll('#preferenceCheckboxes input').forEach(input => {

input.checked = preferences.includes(input.value);

});

const modal = new bootstrap.Modal(document.getElementById('contactModal'));

modal.show();

}

});

// Handle Contact Form Submission (POST /contacts or PUT /contacts/:id)

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

e.preventDefault();

const form = e.target;

const contactId = form.contactId.value;

const btn = document.getElementById('saveContactBtn');

btn.disabled = true;

const preferences = Array.from(document.querySelectorAll('#preferenceCheckboxes input:checked')).map(input => input.value);

const payload = {

studentId: form.studentId.value,

parentName: form.parentName.value,

phoneNumber: form.phoneNumber.value,

preferredLanguage: form.preferredLanguage.value,

consentGiven: form.consentGiven.checked,

arattaiVerified: form.arattaiVerified.checked,

notificationPreferences: preferences

};

const method = contactId ? 'PUT' : 'POST';

const endpoint = contactId ? `/contacts/${contactId}` : '/contacts';

try {

const data = await apiFetch(endpoint, method, payload);

showNotification(data.message || `Contact successfully ${contactId ? 'updated' : 'added'}.`, 'success');

// Close modal and refresh

bootstrap.Modal.getInstance(document.getElementById('contactModal')).hide();

form.reset();

fetchContacts();

} catch {

// Handled by apiFetch

} finally {

btn.disabled = false;

}

});

// Opt-Out Logic

document.getElementById('optOutForm').addEventListener('submit', async (e) => {

e.preventDefault();

const phone = document.getElementById('optOutPhone').value;

const btn = document.getElementById('optOutBtn');

btn.disabled = true;

try {

const data = await apiFetch('/optout', 'POST', { phoneNumber: phone });

showNotification(data.message, 'success');

document.getElementById('optOutStatus').innerHTML = `<p class="text-danger fw-bold">Number ${phone} is now permanently Opted-Out.</p>`;

} catch {

// Handled by apiFetch

} finally {

btn.disabled = false;

}

});

// --- 6. CONFIGURATION LOGIC ---

async function fetchSystemConfig() {

try {

const settings = await apiFetch('/settings');

document.getElementById('businessNumber').value = settings.businessNumber || '';

document.getElementById('businessName').value = settings.businessName || '';

document.getElementById('apiProvider').value = settings.apiProvider || 'arattai\_business\_api';

// Limits

document.getElementById('dailyLimit').value = settings.dailyLimit || 0;

document.getElementById('monthlyLimit').value = settings.monthlyLimit || 0;

document.getElementById('currentUsage').value = settings.currentUsage || 0;

document.getElementById('remainingQuotaText').textContent = settings.remainingQuota || 0;

// Rate Limiting

document.getElementById('messagesPerMinute').value = settings.rateLimiting.messagesPerMinute || 0;

document.getElementById('messagesPerHour').value = settings.rateLimiting.messagesPerHour || 0;

// Governance

document.getElementById('consentRequired').checked = settings.consentRequired || false;

document.getElementById('automaticOptOut').checked = settings.automaticOptOut || false;

document.getElementById('contentFiltering').checked = settings.contentFiltering || false;

document.getElementById('deliveryReports').checked = settings.deliveryReports || false;

document.getElementById('readReceipts').checked = settings.readReceipts || false;

// Webhooks

document.getElementById('webhookUrl').value = settings.webhookUrl || '';

document.getElementById('webhookSecret').value = settings.webhookSecret || '';

document.getElementById('dataRetentionDays').value = settings.dataRetentionDays || 90;

} catch (error) {

showNotification("Failed to load system settings. Using defaults.", 'danger');

}

}

document.getElementById('systemConfigForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const btn = document.getElementById('saveSystemConfigBtn');

btn.disabled = true;

const payload = {

businessNumber: form.businessNumber.value,

businessName: form.businessName.value,

dailyLimit: parseInt(form.dailyLimit.value),

monthlyLimit: parseInt(form.monthlyLimit.value),

rateLimiting: {

messagesPerMinute: parseInt(form.messagesPerMinute.value),

messagesPerHour: parseInt(form.messagesPerHour.value)

},

consentRequired: form.consentRequired.checked,

automaticOptOut: form.automaticOptOut.checked,

contentFiltering: form.contentFiltering.checked,

deliveryReports: form.deliveryReports.checked,

readReceipts: form.readReceipts.checked,

webhookUrl: form.webhookUrl.value,

webhookSecret: form.webhookSecret.value,

dataRetentionDays: parseInt(form.dataRetentionDays.value)

};

try {

const data = await apiFetch('/settings', 'PUT', payload);

showNotification(data.message, 'success');

fetchSystemConfig();

} catch {

// Handled by apiFetch

} finally {

btn.disabled = false;

}

});

// --- Initialization ---

document.addEventListener('DOMContentLoaded', () => {

// Set default time for scheduling

const now = new Date();

now.setMinutes(now.getMinutes() + 15); // Default 15 mins in future

document.getElementById('scheduledFor').value = now.toISOString().substring(0, 16);

document.getElementById('broadcastScheduledFor').value = now.toISOString().substring(0, 16);

fetchTemplates();

fetchSystemConfig();

fetchAnalytics();

fetchScheduledMessages();

fetchBroadcastCampaigns();

fetchAutomationRules();

fetchContacts();

});

</script>

</body>

</html>

10. client/dist/modules/Attendance.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>✅ Attendance Management System - Ultimate Enterprise</title>

<!-- Bootstrap 5 CDN -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css">

<style>

body { background-color: #f5f8fa; font-family: 'Inter', sans-serif; }

.container-fluid { max-width: 1200px; margin-top: 20px; }

.card-header { background-color: #3f51b5; color: white; font-weight: bold; border-radius: 0.5rem 0.5rem 0 0; }

.section-card { margin-bottom: 25px; box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1); border-radius: 0.5rem; }

.alert-fixed {

position: fixed; top: 20px; right: 20px; z-index: 1050; min-width: 300px; border-radius: 0.5rem;

}

.nav-tabs .nav-link.active {

background-color: #3f51b5;

color: white;

border-color: #3f51b5;

}

.stat-box { transition: transform 0.2s; cursor: default; }

.stat-box:hover { transform: translateY(-3px); box-shadow: 0 4px 8px rgba(0, 0, 0, 0.15); }

.status-present { background-color: #4CAF50; }

.status-absent { background-color: #F44336; }

.status-late { background-color: #FFC107; color: #333; }

.btn-status { color: white; border: none; }

</style>

</head>

<body>

<div class="container-fluid">

<h1 class="text-center mb-4 text-primary"><i class="bi bi-calendar-check"></i> Attendance Module</h1>

<!-- Notification Area -->

<div id="notification-area" class="alert-fixed" style="display: none;"></div>

<!-- Tab Navigation -->

<ul class="nav nav-tabs" id="attendanceTab" role="tablist">

<li class="nav-item"><button class="nav-link active" id="daily-tab" data-bs-toggle="tab" data-bs-target="#daily" type="button"><i class="bi bi-clock"></i> Daily Operations</button></li>

<li class="nav-item"><button class="nav-link" id="teacher-tab" data-bs-toggle="tab" data-bs-target="#teacher" type="button"><i class="bi bi-person-badge"></i> Teacher Check-In</button></li>

<li class="nav-item"><button class="nav-link" id="barcode-tab" data-bs-toggle="tab" data-bs-target="#barcode" type="button"><i class="bi bi-qr-code-scan"></i> Student Attendance</button></li>

<li class="nav-item"><button class="nav-link" id="config-tab" data-bs-toggle="tab" data-bs-target="#config" type="button"><i class="bi bi-gear-fill"></i> Configuration & Payroll</button></li>

</ul>

<!-- Tab Content -->

<div class="tab-content pt-3" id="attendanceTabContent">

<!-- 1. Daily Operations Tab (GET /summary/today, POST /mark/student/:id) -->

<div class="tab-pane fade show active" id="daily" role="tabpanel">

<div class="card section-card">

<div class="card-header">Today's Attendance Overview</div>

<div class="card-body">

<!-- Summary Stats -->

<div class="row text-center g-3 mb-4" id="dailySummaryStats">

<!-- Stats dynamically loaded -->

</div>

<!-- Manual Student Marking -->

<h5 class="mt-4 mb-3">Manual Student Attendance Marking (Teacher/Admin)</h5>

<form id="manualMarkForm" class="row g-3 border p-3 rounded bg-light">

<div class="col-md-3">

<label for="manualStudentId" class="form-label">Student ID</label>

<input type="text" class="form-control" id="manualStudentId" placeholder="e.g., STD1001" required>

</div>

<div class="col-md-3">

<label for="manualClassId" class="form-label">Class ID</label>

<input type="text" class="form-control" id="manualClassId" placeholder="e.g., 10-A" required>

</div>

<div class="col-md-3">

<label for="manualDate" class="form-label">Date</label>

<input type="date" class="form-control" id="manualDate" required>

</div>

<div class="col-md-3">

<label for="manualTimeIn" class="form-label">Time In (HH:MM)</label>

<input type="time" class="form-control" id="manualTimeIn" required>

</div>

<div class="col-md-8">

<label for="manualNotes" class="form-label">Notes/Remarks</label>

<input type="text" class="form-control" id="manualNotes" placeholder="e.g., Absent due to fever">

</div>

<div class="col-md-4 d-flex align-items-end">

<div class="btn-group w-100" role="group">

<button type="button" class="btn btn-status status-present" data-status="present" data-action="mark">Present</button>

<button type="button" class="btn btn-status status-absent" data-status="absent" data-action="mark">Absent</button>

<button type="button" class="btn btn-status status-late" data-status="late" data-action="mark">Late</button>

</div>

</div>

<div class="col-12"><p class="text-danger small" id="manualMarkError"></p></div>

</form>

</div>

</div>

</div>

<!-- 2. Teacher Check-In Tab (POST /teachers/mark/:id) -->

<div class="tab-pane fade" id="teacher" role="tabpanel">

<div class="card section-card">

<div class="card-header">Teacher Secure Check-In (Biometric & GPS)</div>

<div class="card-body">

<p class="alert alert-success"><i class="bi bi-link-45deg"></i> \*\*Simulated Link Check-In:\*\* The system sends a secure link (30 mins before start/at end) to the teacher's registered mobile number. Opening the link triggers this process.</p>

<form id="teacherMarkForm" class="row g-3 border p-3 rounded bg-light">

<div class="col-md-4">

<label for="teacherProfileId" class="form-label">Teacher Profile ID</label>

<input type="text" class="form-control" id="teacherProfileId" placeholder="e.g., TEACHER\_005" required>

</div>

<div class="col-md-4">

<label for="teacherLocationSelect" class="form-label">Check-in Location</label>

<select class="form-select" id="teacherLocationSelect" required>

<option value="">Select Pre-set Location</option>

<!-- Options loaded from config -->

</select>

<div class="form-text">Choose the site where you are currently on duty.</div>

</div>

<div class="col-md-4">

<label for="biometricToken" class="form-label">Biometric Token (Simulated)</label>

<input type="password" class="form-control" id="biometricToken" placeholder="Enter OTP/Token" required>

<div class="form-text">Simulates Fingerprint/Face ID verification.</div>

</div>

<div class="col-12">

<label class="form-label">Simulated GPS Capture (Overridden by Selected Location)</label>

<div class="input-group">

<input type="text" class="form-control" id="teacherLat" placeholder="Latitude (Captured)" readonly>

<input type="text" class="form-control" id="teacherLon" placeholder="Longitude (Captured)" readonly>

</div>

<div class="form-text text-danger" id="geofenceStatus">Status: GPS coordinates locked to selected pre-set location.</div>

</div>

<div class="col-12 d-flex justify-content-center pt-3">

<button type="button" class="btn btn-lg btn-status status-present" data-status="present" data-action="mark-teacher"><i class="bi bi-fingerprint"></i> Check-In (Present)</button>

</div>

<div class="col-12 text-center pt-2">

<p class="text-danger small">Late status is determined automatically by time. Check-Out uses the same button sharp at end time.</p>

</div>

</form>

</div>

</div>

</div>

<!-- 3. Barcode Simulation Tab (POST /mark/barcode) -->

<div class="tab-pane fade" id="barcode" role="tabpanel">

<div class="card section-card">

<div class="card-header">Student Attendance Marking (Via Teacher/Admin Mobile)</div>

<div class="card-body">

<p class="alert alert-info"><i class="bi bi-mobile"></i> \*\*Teacher/Admin Action:\*\* Use your mobile device to scan the student's ID card barcode. The system records the student's ID, the time, and \*\*YOUR mobile location\*\*.</p>

<form id="barcodeMarkForm" class="row g-3 border p-3 rounded bg-light">

<div class="col-md-6">

<label for="barcodeUid" class="form-label">Barcode/QR UID (Student ID Card)</label>

<input type="text" class="form-control" id="barcodeUid" placeholder="e.g., BC456789" required>

<div class="form-text">Simulates the mobile camera scan.</div>

</div>

<div class="col-md-6">

<label for="barcodeTimeIn" class="form-label">Simulated Scan Time</label>

<input type="time" class="form-control" id="barcodeTimeIn" required>

<div class="form-text">Used to determine 'Present' or 'Late' status against school start time.</div>

</div>

<div class="col-12">

<h6 class="mt-3">Location Capture (Teacher/Admin Device GPS)</h6>

<div class="input-group">

<input type="text" class="form-control" id="barcodeLat" placeholder="Latitude (Current Location)" required>

<input type="text" class="form-control" id="barcodeLon" placeholder="Longitude (Current Location)" required>

<button type="button" class="btn btn-outline-secondary" id="getCurrentLocationBtn"><i class="bi bi-geo-alt"></i> Get GPS</button>

</div>

<div class="form-text">Records the location where the attendance was marked (e.g., the classroom).</div>

</div>

<div class="col-12 pt-3">

<button type="submit" class="btn btn-primary w-100" id="scanBarcodeBtn"><i class="bi bi-qr-code"></i> Record Student Attendance</button>

</div>

</form>

</div>

</div>

</div>

<!-- 4. Configuration & Payroll Tab (GET/PUT /config) -->

<div class="tab-pane fade" id="config" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-danger">Attendance Rules & Payroll Integration (Admin/Principal Only)</div>

<div class="card-body">

<form id="attendanceConfigForm">

<h5 class="mt-2">I. School Timing & Auto-Marking</h5>

<div class="row g-3 mb-4 border p-3 rounded">

<div class="col-md-4">

<label for="startTime" class="form-label">Official School Start Time</label>

<input type="time" class="form-control" id="startTime" required>

</div>

<div class="col-md-4">

<label for="lateThreshold" class="form-label">Late Threshold (Minutes)</label>

<input type="number" class="form-control" id="lateThreshold" required min="1" max="60">

</div>

<div class="col-md-4 d-flex align-items-center">

<div class="form-check form-switch pt-4">

<input class="form-check-input" type="checkbox" id="autoMarkAbsentEnabled">

<label class="form-check-label" for="autoMarkAbsentEnabled">Auto-Mark Absent at EOD</label>

</div>

</div>

</div>

<h5 class="mt-4">II. Teacher Flexible Location Setup (3 Pre-set Spots)</h5>

<div class="row g-3 mb-4 border p-3 rounded">

<div class="col-12"><div class="form-check form-switch"><input class="form-check-input" type="checkbox" id="geofenceEnabled"><label class="form-check-label" for="geofenceEnabled">Enable Geofence Enforcement (For all locations)</label></div></div>

<div class="col-md-4">

<label for="geofenceRadius" class="form-label">Geofence Radius (Meters)</label>

<input type="number" class="form-control" id="geofenceRadius" required min="10" max="500">

<div class="form-text">Radius around \*any\* selected location.</div>

</div>

<h6 class="mt-3">Location 1 (Default/Main Campus)</h6>

<div class="col-md-3"><input type="text" class="form-control" id="loc1Name" placeholder="Location Name" value="Main Campus"></div>

<div class="col-md-3"><input type="text" class="form-control" id="geofenceLat" placeholder="Latitude" required></div>

<div class="col-md-3"><input type="text" class="form-control" id="geofenceLon" placeholder="Longitude" required></div>

<h6 class="mt-3">Location 2 (e.g., Sports Ground)</h6>

<div class="col-md-3"><input type="text" class="form-control" id="loc2Name" placeholder="Location Name" value="Sports Ground"></div>

<div class="col-md-3"><input type="text" class="form-control" id="loc2Lat" placeholder="Latitude"></div>

<div class="col-md-3"><input type="text" class="form-control" id="loc2Lon" placeholder="Longitude"></div>

<h6 class="mt-3">Location 3 (e.g., Auditorium/Field Trip)</h6>

<div class="col-md-3"><input type="text" class="form-control" id="loc3Name" placeholder="Location Name" value="Auditorium"></div>

<div class="col-md-3"><input type="text" class="form-control" id="loc3Lat" placeholder="Latitude"></div>

<div class="col-md-3"><input type="text" class="form-control" id="loc3Lon" placeholder="Longitude"></div>

</div>

<h5 class="mt-4 text-danger">III. Payroll & Leave Conversion Rules</h5>

<p class="text-danger small">These rules integrate directly with the Accounts/Leave Management module to penalize frequent late marks.</p>

<div class="row g-3 mb-4 border border-danger p-3 rounded">

<div class="col-md-4">

<label for="maxLate" class="form-label">Maximum Allowed Late Entries</label>

<input type="number" class="form-control" id="maxLate" required min="1" max="30">

<div class="form-text">Total late marks allowed before conversion starts. (e.g., 6)</div>

</div>

<div class="col-md-4">

<label for="lateToPerm" class="form-label">Late to Permission Conversion Factor</label>

<input type="number" class="form-control" id="lateToPerm" required min="1" max="100">

<div class="form-text">X late marks = 1 Permission (e.g., 6 lates = 1 perm)</div>

</div>

<div class="col-md-4">

<label for="permToHalf" class="form-label">Permission to Half-Day Leave Factor</label>

<input type="number" class="form-control" id="permToHalf" required min="1" max="100">

<div class="form-text">Y Permissions = 0.5 Day Leave (e.g., 8 perms = 0.5 day)</div>

</div>

</div>

<button type="submit" class="btn btn-danger mt-3" id="saveConfigBtn"><i class="bi bi-save"></i> Save All Attendance & Payroll Settings</button>

</form>

</div>

</div>

</div>

</div>

</div>

<!-- Bootstrap Bundle with Popper -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>

<script>

const API\_BASE = '';

let globalLocations = []; // Store configured locations for use in the teacher check-in form

/\*\* Helper function to show a temporary notification. \*/

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

const area = document.getElementById('notification-area');

area.innerHTML = `

<div class="alert alert-${type} alert-dismissible fade show" role="alert">

${message}

<button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>

</div>

`;

area.style.display = 'block';

setTimeout(() => {

area.style.display = 'none';

}, 7000);

}

/\*\* Core Fetch Function with Error Handling. \*/

async function apiFetch(endpoint, method = 'GET', body = null) {

try {

const options = {

method: method,

headers: { 'Content-Type': 'application/json' },

};

if (body && method !== 'GET') {

options.body = JSON.stringify(body);

}

const response = await fetch(`${API\_BASE}${endpoint}`, options);

const data = await response.json();

if (!response.ok) {

throw new Error(data.error || `API error (${response.status})`);

}

return data;

} catch (error) {

console.error('API Fetch Error:', endpoint, error);

showNotification(error.message || 'Network error.', 'danger');

throw error;

}

}

/\*\* Helper to get actual GPS coordinates for the current device \*/

function getCurrentLocation(latInput, lonInput) {

if (navigator.geolocation) {

navigator.geolocation.getCurrentPosition(

(position) => {

latInput.value = position.coords.latitude.toFixed(6);

lonInput.value = position.coords.longitude.toFixed(6);

showNotification("GPS captured successfully!", 'info');

},

(error) => {

console.error("Geolocation error:", error);

showNotification("Failed to capture GPS. Please input manually.", 'warning');

latInput.value = '';

lonInput.value = '';

},

{ enableHighAccuracy: true, timeout: 5000, maximumAge: 0 }

);

} else {

showNotification("Geolocation is not supported by this browser.", 'warning');

}

}

// --- 1. DAILY OPERATIONS LOGIC ---

async function fetchAttendanceSummary() {

const statsArea = document.getElementById('dailySummaryStats');

statsArea.innerHTML = '<div class="col-12 text-center text-muted">Fetching summary...</div>';

try {

const summary = await apiFetch('/summary/today');

const absentPercent = summary.totalStudents > 0 ? ((summary.absent / summary.totalStudents) \* 100).toFixed(1) : 0;

statsArea.innerHTML = `

<div class="col-md-3"><div class="p-3 border rounded bg-light stat-box"><h4 class="text-primary">${summary.totalStudents.toLocaleString()}</h4><p class="text-muted">Total Enrolled Students</p></div></div>

<div class="col-md-3"><div class="p-3 border rounded bg-light stat-box"><h4 class="status-present text-white">${summary.present.toLocaleString()}</h4><p class="text-muted">Present (On-Time + Late)</p></div></div>

<div class="col-md-3"><div class="p-3 border rounded bg-light stat-box"><h4 class="status-absent text-white">${summary.absent.toLocaleString()}</h4><p class="text-muted">Absent Students</p></div></div>

<div class="col-md-3"><div class="p-3 border rounded bg-light stat-box"><h4 class="text-success">${summary.attendanceRate}%</h4><p class="text-muted">Attendance Rate (${absentPercent}% Absent)</p></div></div>

`;

} catch (error) {

statsArea.innerHTML = '<div class="col-12 text-center text-danger">Failed to load daily summary.</div>';

}

}

// Student Manual Marking Logic

document.getElementById('manualMarkForm').addEventListener('click', async (e) => {

const button = e.target.closest('button[data-action="mark"]');

if (!button) return;

const form = document.getElementById('manualMarkForm');

const studentId = form.manualStudentId.value;

const classId = form.manualClassId.value;

const date = form.manualDate.value;

const timeIn = form.manualTimeIn.value;

const status = button.getAttribute('data-status');

const notes = form.manualNotes.value;

document.getElementById('manualMarkError').textContent = '';

if (!studentId || !classId || !date || !timeIn) {

document.getElementById('manualMarkError').textContent = "Please fill in Student ID, Class ID, Date, and Time.";

return;

}

const payload = {

date,

status,

timeIn,

classId,

notes,

method: 'manual',

};

button.disabled = true;

button.innerHTML = '<span class="spinner-border spinner-border-sm"></span>';

try {

const data = await apiFetch(`/mark/student/${studentId}`, 'POST', payload);

showNotification(data.message, 'success');

form.reset();

fetchAttendanceSummary();

} catch (error) {

// Error handled by apiFetch

} finally {

button.disabled = false;

button.innerHTML = status.charAt(0).toUpperCase() + status.slice(1);

}

});

// --- 2. TEACHER CHECK-IN LOGIC ---

function populateTeacherLocations() {

const select = document.getElementById('teacherLocationSelect');

select.innerHTML = '<option value="">Select Pre-set Location</option>';

if (globalLocations.length === 0) {

select.innerHTML += '<option value="" disabled>No locations configured yet.</option>';

return;

}

globalLocations.forEach((loc, index) => {

if (loc.lat && loc.lon) {

select.innerHTML += `<option value="${index}" data-lat="${loc.lat}" data-lon="${loc.lon}">${loc.name}</option>`;

}

});

}

document.getElementById('teacherLocationSelect').addEventListener('change', (e) => {

const selectedOption = e.target.options[e.target.selectedIndex];

const lat = selectedOption.getAttribute('data-lat');

const lon = selectedOption.getAttribute('data-lon');

document.getElementById('teacherLat').value = lat || '';

document.getElementById('teacherLon').value = lon || '';

});

document.getElementById('teacherMarkForm').addEventListener('click', async (e) => {

const button = e.target.closest('button[data-action="mark-teacher"]');

if (!button) return;

const form = document.getElementById('teacherMarkForm');

const teacherProfileId = form.teacherProfileId.value;

const selectedLocationIndex = form.teacherLocationSelect.value;

const biometricToken = form.biometricToken.value;

const date = form.teacherDate.value;

const timeIn = form.teacherTimeIn.value;

const latitude = form.teacherLat.value;

const longitude = form.teacherLon.value;

const notes = 'Checked in from ' + form.teacherLocationSelect.options[form.teacherLocationSelect.selectedIndex].text; // Fixed note

if (!teacherProfileId || !selectedLocationIndex || !biometricToken) {

showNotification("Teacher ID, Location, and Biometric Token are required.", 'warning');

return;

}

if (!latitude || !longitude) {

showNotification("Selected location is missing GPS coordinates. Please check configuration.", 'danger');

return;

}

// NOTE: Status is simplified to 'present' initially, the backend will determine 'late' based on timeIn vs. startTime.

const status = 'present';

const payload = {

date,

status,

timeIn,

notes,

latitude,

longitude

};

button.disabled = true;

button.innerHTML = '<span class="spinner-border spinner-border-sm"></span>';

try {

const data = await apiFetch(`/teachers/mark/${teacherProfileId}`, 'POST', payload);

if (data.geofenceAlertTriggered) {

showNotification("Mark recorded, but \*\*GEOFENCE VIOLATION DETECTED!\*\* Principal/Admin alerted.", 'danger');

} else {

// NOTE: The backend logic for teachers determines late/present based on time, and triggers leave conversion.

const finalStatus = data.record.status || 'Present';

showNotification(`Teacher Check-In recorded as: \*\*${finalStatus.toUpperCase()}\*\*.`, 'success');

}

form.reset();

populateTeacherLocations(); // Reset dropdown

} catch (error) {

// Error handled by apiFetch

} finally {

button.disabled = false;

button.innerHTML = '<i class="bi bi-fingerprint"></i> Check-In (Present)';

}

});

// --- 3. BARCODE SIMULATION LOGIC ---

document.getElementById('getCurrentLocationBtn').addEventListener('click', () => {

getCurrentLocation(document.getElementById('barcodeLat'), document.getElementById('barcodeLon'));

});

document.getElementById('barcodeMarkForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const barcodeUid = form.barcodeUid.value;

const timeIn = form.barcodeTimeIn.value;

const latitude = form.barcodeLat.value;

const longitude = form.barcodeLon.value;

const btn = document.getElementById('scanBarcodeBtn');

btn.disabled = true;

if (!latitude || !longitude) {

showNotification("Please capture or input the Teacher/Admin device location.", 'warning');

btn.disabled = false;

return;

}

const payload = { barcode\_uid: barcodeUid, timeIn, latitude, longitude };

try {

const data = await apiFetch('/mark/barcode', 'POST', payload);

showNotification(data.message, 'success');

form.reset();

// Persist GPS data for quick successive scans

document.getElementById('barcodeLat').value = latitude;

document.getElementById('barcodeLon').value = longitude;

fetchAttendanceSummary();

} catch (error) {

// Error handled by apiFetch

} finally {

btn.disabled = false;

}

});

// --- 4. CONFIGURATION LOGIC ---

async function fetchAttendanceConfig() {

try {

const data = await apiFetch('/config');

const settings = data.settings;

const payrollRules = data.payrollRules;

// I. Timing Settings

document.getElementById('startTime').value = settings.school\_start\_time || '08:00';

document.getElementById('lateThreshold').value = settings.late\_threshold\_minutes || 10;

document.getElementById('autoMarkAbsentEnabled').checked = settings.auto\_mark\_absent || false;

// II. Geofence Settings and Flexible Locations

document.getElementById('geofenceEnabled').checked = settings.geofence\_enabled || false;

document.getElementById('geofenceRadius').value = settings.geofence\_radius\_meters || 100;

// Default Location 1 (Main Geofence Center)

document.getElementById('geofenceLat').value = settings.geofence\_center\_lat || '13.0827';

document.getElementById('geofenceLon').value = settings.geofence\_center\_lon || '80.2707';

// Load custom locations or use mock defaults for display

const customLocations = JSON.parse(settings.custom\_locations || '[]');

const locs = [

{ id: 1, name: settings.loc1\_name || 'Main Campus', lat: settings.geofence\_center\_lat, lon: settings.geofence\_center\_lon },

{ id: 2, name: settings.loc2\_name || 'Sports Ground', lat: settings.loc2\_lat || '', lon: settings.loc2\_lon || '' },

{ id: 3, name: settings.loc3\_name || 'Auditorium', lat: settings.loc3\_lat || '', lon: settings.loc3\_lon || '' },

];

// Set default/loaded values for config form

document.getElementById('loc1Name').value = locs[0].name;

document.getElementById('geofenceLat').value = locs[0].lat;

document.getElementById('geofenceLon').value = locs[0].lon;

document.getElementById('loc2Name').value = locs[1].name;

document.getElementById('loc2Lat').value = locs[1].lat;

document.getElementById('loc2Lon').value = locs[1].lon;

document.getElementById('loc3Name').value = locs[2].name;

document.getElementById('loc3Lat').value = locs[2].lat;

document.getElementById('loc3Lon').value = locs[2].lon;

globalLocations = locs.filter(l => l.lat && l.lon);

populateTeacherLocations(); // Update teacher check-in form

// III. Payroll Rules

document.getElementById('maxLate').value = payrollRules.max\_late\_entries || 6;

document.getElementById('lateToPerm').value = payrollRules.late\_to\_perm\_conversion || 6;

document.getElementById('permToHalf').value = payrollRules.perm\_to\_halfday\_conversion || 8;

} catch (error) {

// Fail silently or show default values if API fails/is mocked

console.error("Failed to fetch config:", error);

}

}

document.getElementById('attendanceConfigForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const btn = document.getElementById('saveConfigBtn');

btn.disabled = true;

// Payload for both attendance\_settings and teacher\_leave\_config

const payload = {

startTime: form.startTime.value,

lateThreshold: parseInt(form.lateThreshold.value),

autoMarkAbsentEnabled: form.autoMarkAbsentEnabled.checked,

geofenceEnabled: form.geofenceEnabled.checked,

geofenceLat: form.geofenceLat.value, // Location 1 GPS (Main Geofence Center)

geofenceLon: form.geofenceLon.value,

geofenceRadius: parseInt(form.geofenceRadius.value),

// New Flexible Locations data to be processed by backend

loc1Name: form.loc1Name.value,

loc2Name: form.loc2Name.value,

loc2Lat: form.loc2Lat.value,

loc2Lon: form.loc2Lon.value,

loc3Name: form.loc3Name.value,

loc3Lat: form.loc3Lat.value,

loc3Lon: form.loc3Lon.value,

// Payroll Rules

maxLate: parseInt(form.maxLate.value),

lateToPerm: parseInt(form.lateToPerm.value),

permToHalf: parseInt(form.permToHalf.value),

};

try {

const data = await apiFetch('/config', 'PUT', payload);

showNotification(data.message, 'success');

fetchAttendanceConfig(); // Refresh config display and update globalLocations

} catch (error) {

// Error handled by apiFetch

} finally {

btn.disabled = false;

}

});

// --- Initialization ---

document.addEventListener('DOMContentLoaded', () => {

const today = new Date().toISOString().split('T')[0];

const now = new Date();

const time = `${String(now.getHours()).padStart(2, '0')}:${String(now.getMinutes()).padStart(2, '0')}`;

document.getElementById('manualDate').value = today;

document.getElementById('teacherDate').value = today;

document.getElementById('manualTimeIn').value = time;

document.getElementById('teacherTimeIn').value = time;

document.getElementById('barcodeTimeIn').value = time;

// Add GPS capture event for the student marking tab (Teacher/Admin device location)

document.getElementById('getCurrentLocationBtn').addEventListener('click', () => {

getCurrentLocation(document.getElementById('barcodeLat'), document.getElementById('barcodeLon'));

});

fetchAttendanceSummary();

fetchAttendanceConfig();

});

</script>

</body>

</html>

11. client/dist/modules/Fee\_Management\_Admin.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>💰 Enterprise Fee Management System</title>

<!-- Bootstrap 5 CDN -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css">

<style>

body { background-color: #f8f9fa; font-family: 'Inter', sans-serif; }

.container-fluid { max-width: 1300px; margin-top: 20px; }

.header-bar {

background-color: #004d40; /\* Darker teal for contrast \*/

padding: 10px 20px;

border-radius: 0.5rem;

margin-bottom: 20px;

color: white;

}

.card-header { background-color: #00897b; color: white; font-weight: bold; border-radius: 0.5rem 0.5rem 0 0; }

.section-card { margin-bottom: 25px; box-shadow: 0 4px 12px rgba(0, 0, 0, 0.1); border-radius: 0.5rem; }

.alert-fixed { position: fixed; top: 20px; right: 20px; z-index: 1050; min-width: 300px; border-radius: 0.5rem; }

.nav-tabs .nav-link.active { background-color: #00897b; color: white; border-color: #00897b; }

.tab-content { border-radius: 0 0 0.5rem 0.5rem; }

.overdue-row { background-color: #fce4ec; }

.text-end { text-align: right !important; }

</style>

</head>

<body>

<div class="container-fluid">

<!-- GLOBAL HEADER BAR (Multi-Branch Selector) -->

<div class="header-bar d-flex justify-content-between align-items-center">

<h1 class="h3 mb-0"><i class="bi bi-bank"></i> Central Accounts Dashboard</h1>

<div class="d-flex align-items-center">

<label for="branchSelector" class="me-3 mb-0 fw-bold">Active Branch:</label>

<select id="branchSelector" class="form-select form-select-sm" style="width: 250px;"></select>

<h5 class="ms-4 mb-0"><span id="currentBranchName" class="badge bg-light text-dark">Loading...</span></h5>

</div>

</div>

<!-- Notification Area -->

<div id="notification-area" class="alert-fixed" style="display: none;"></div>

<!-- Tab Navigation -->

<ul class="nav nav-tabs" id="feeTab" role="tablist">

<li class="nav-item"><button class="nav-link active" id="collection-tab" data-bs-toggle="tab" data-bs-target="#collection" type="button"><i class="bi bi-wallet2"></i> Payment Collection</button></li>

<li class="nav-item"><button class="nav-link" id="defaulters-tab" data-bs-toggle="tab" data-bs-target="#defaulters" type="button"><i class="bi bi-person-x-fill"></i> Defaulters & Alerts</button></li>

<li class="nav-item"><button class="nav-link" id="config-tab" data-bs-toggle="tab" data-bs-target="#config" type="button"><i class="bi bi-sliders"></i> Configuration & Rules</button></li>

<li class="nav-item"><button class="nav-link" id="audit-tab" data-bs-toggle="tab" data-bs-target="#audit" type="button"><i class="bi bi-building-check"></i> Accounting & Integration</button></li>

</ul>

<!-- Tab Content -->

<div class="tab-content pt-3" id="feeTabContent">

<!-- 1. Payment Collection Tab (POST /payment) -->

<div class="tab-pane fade show active" id="collection" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-success">Log New Payment & Verification</div>

<div class="card-body">

<!-- Payment Logging Form -->

<form id="paymentForm" class="row g-3 mb-4">

<h5 class="text-primary">Record New Transaction (Branch: <span class="text-danger" id="formBranchName"></span>)</h5>

<div class="col-md-4">

<label for="studentId" class="form-label">Student ID (UUID)</label>

<input type="text" class="form-control" id="studentId" placeholder="e.g., uuid-std-123" required>

</div>

<div class="col-md-4">

<label for="amountPaid" class="form-label">Amount Paid (₹)</label>

<input type="number" class="form-control" id="amountPaid" min="1" step="0.01" required>

</div>

<div class="col-md-4">

<label for="feeStructureId" class="form-label">Fee Structure ID (Optional)</label>

<input type="text" class="form-control" id="feeStructureId" placeholder="e.g., Q1-2025" >

</div>

<div class="col-md-4">

<label for="paymentMethod" class="form-label">Method</label>

<select class="form-select" id="paymentMethod" required>

<option value="cash">Cash</option>

<option value="online">Online Transfer/UPI</option>

<option value="card">Card</option>

<option value="cheque">Cheque</option>

</select>

</div>

<div class="col-md-4">

<label for="transactionId" class="form-label">Transaction/Reference ID</label>

<input type="text" class="form-control" id="transactionId" placeholder="Required for non-cash methods">

</div>

<div class="col-md-4 d-flex align-items-end">

<button type="submit" class="btn btn-success w-100" id="logPaymentBtn"><i class="bi bi-plus-circle"></i> Log Payment</button>

</div>

</form>

<hr>

<!-- Payment History & Verification -->

<div class="row">

<div class="col-md-8">

<h5 class="text-primary mt-3">Recent Payments Logged</h5>

</div>

<div class="col-md-4 text-end">

<button class="btn btn-sm btn-info mt-3" id="triPartyBtn" data-bs-toggle="modal" data-bs-target="#verificationModal"><i class="bi bi-whatsapp"></i> Initiate Tri-Party Verification</button>

</div>

</div>

<div class="table-responsive mt-3">

<table class="table table-striped table-sm">

<thead>

<tr>

<th>Date</th>

<th>Student Name</th>

<th>Amount</th>

<th>Method</th>

<th>Txn ID</th>

<th>Status</th>

</tr>

</thead>

<tbody id="paymentsTableBody">

<tr><td colspan="6" class="text-center">Select a branch to view payments.</td></tr>

</tbody>

</table>

</div>

<div id="paymentPagination" class="d-flex justify-content-center">

<button class="btn btn-sm btn-outline-primary me-2" id="prevPaymentBtn" disabled>Prev</button>

<button class="btn btn-sm btn-outline-primary" id="nextPaymentBtn" disabled>Next</button>

</div>

</div>

</div>

</div>

<!-- 2. Defaulters & Alerts Tab (GET /defaulters, POST /defaulters/auto-alert) -->

<div class="tab-pane fade" id="defaulters" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-danger">Defaulters Management & Automated Alerts</div>

<div class="card-body">

<!-- Defaulter Filters & Actions -->

<form id="defaulterFilterForm" class="row g-3 mb-4">

<div class="col-md-3">

<input type="number" class="form-control" id="minOverdueDays" placeholder="Min Overdue Days" min="1">

</div>

<div class="col-md-3">

<input type="number" class="form-control" id="minAmount" placeholder="Min Overdue Amount (₹)" min="100">

</div>

<div class="col-md-3">

<button type="submit" class="btn btn-danger w-100"><i class="bi bi-filter-circle"></i> Filter Defaulters</button>

</div>

<div class="col-md-3">

<button type="button" class="btn btn-warning w-100" id="sendAlertsBtn"><i class="bi bi-bell"></i> Send Daily Reminders</button>

</div>

</form>

<!-- Defaulters List -->

<div class="table-responsive">

<table class="table table-striped table-sm">

<thead>

<tr>

<th>Student</th>

<th>Class</th>

<th>Overdue Amount</th>

<th>Days Overdue</th>

<th>Parent Phone</th>

<th>Last Alert</th>

</tr>

</thead>

<tbody id="defaultersTableBody">

<tr><td colspan="6" class="text-center">Select a branch to view defaulters.</td></tr>

</tbody>

</table>

</div>

<div class="d-flex justify-content-between mt-3">

<button class="btn btn-sm btn-outline-secondary" id="exportDefaultersCsv"><i class="bi bi-file-earmark-spreadsheet"></i> Export CSV</button>

<button class="btn btn-sm btn-outline-secondary" id="exportDefaultersPdf"><i class="bi bi-file-earmark-pdf"></i> Export PDF Report</button>

</div>

</div>

</div>

</div>

<!-- 3. Configuration & Rules Tab (GET/PUT /config) -->

<div class="tab-pane fade" id="config" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-info">Fee Configuration, Late Rules & Structures (Branch: <span id="configBranchName"></span>)</div>

<div class="card-body">

<form id="configForm" class="row g-3">

<h5 class="text-info"><i class="bi bi-wrench"></i> Core Fee Rules</h5>

<div class="col-md-4">

<label for="lateFeePercent" class="form-label">Late Fee Percentage (%)</label>

<input type="number" class="form-control" id="lateFeePercent" min="0" max="100" step="0.1" required>

<div class="form-text">Percentage applied to the outstanding balance after the grace period.</div>

</div>

<div class="col-md-4">

<label for="graceDays" class="form-label">Grace Period (Days)</label>

<input type="number" class="form-control" id="graceDays" min="0" required>

<div class="form-text">Number of days after the due date before late fees start accruing.</div>

</div>

<div class="col-md-4">

<label for="accountingSyncEnabled" class="form-label">Tally/Accounting Sync</label>

<select class="form-select" id="accountingSyncEnabled">

<option value="true">Enabled (Auto Sync)</option>

<option value="false">Disabled (Manual Sync Only)</option>

</select>

<div class="form-text">Controls automated fee data synchronization with external accounting software.</div>

</div>

<hr class="my-4">

<h5 class="text-info"><i class="bi bi-list-columns-reverse"></i> Fee Structure Management</h5>

<p class="small text-muted">Use the form to add new components (like Transportation, Olympiad, etc.) or remove existing ones below.</p>

<!-- New form for adding components -->

<form id="addFeeComponentForm" class="row g-3 mb-4 p-3 border rounded bg-light">

<div class="col-md-3">

<label class="form-label small">Component Name</label>

<input type="text" class="form-control form-control-sm" id="newComponentName" placeholder="e.g., Annual Day Fee" required>

</div>

<div class="col-md-3">

<label class="form-label small">Class Range</label>

<input type="text" class="form-control form-control-sm" id="newClassRange" placeholder="e.g., 9th-12th" required>

</div>

<div class="col-md-2">

<label class="form-label small">Amount (₹)</label>

<input type="number" class="form-control form-control-sm" id="newAmount" min="0" required>

</div>

<div class="col-md-2">

<label class="form-label small">Applicability</label>

<select class="form-select form-select-sm" id="newApplicability">

<option value="Mandatory">Mandatory</option>

<option value="Optional">Optional</option>

</select>

</div>

<div class="col-md-2 d-flex align-items-end">

<button type="submit" class="btn btn-sm btn-outline-success w-100"><i class="bi bi-plus-lg"></i> Add Component</button>

</div>

</form>

<!-- Fee Structure Table -->

<div class="col-12 table-responsive">

<table class="table table-bordered table-sm">

<thead>

<tr><th>Component</th><th>Class Range</th><th>Amount (₹)</th><th>Applicability</th><th>Action</th></tr>

</thead>

<tbody id="feeStructureBody">

<tr><td colspan="5" class="text-center">Select a branch to manage fee structure.</td></tr>

</tbody>

</table>

</div>

<hr class="my-4">

<div class="col-12">

<button type="submit" class="btn btn-info btn-lg w-100" id="saveConfigBtn"><i class="bi bi-save"></i> Save All Configuration Rules</button>

</div>

</form>

</div>

</div>

</div>

<!-- 4. Accounting & Integration Tab (POST /tally/sync-collection, POST /tally/webhook) -->

<div class="tab-pane fade" id="audit" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-dark">Accounting & Tally Integration (Branch: <span id="auditBranchName"></span>)</div>

<div class="card-body">

<h5 class="text-danger"><i class="bi bi-tally"></i> Tally Accounting Sync</h5>

<p class="text-muted">Initiate manual synchronization of fee collections to Tally for reconciliation.</p>

<div class="row g-3">

<div class="col-md-6">

<button class="btn btn-primary w-100" id="tallySyncBtn"><i class="bi bi-arrow-repeat"></i> Sync Fee Collection to Tally Now</button>

<div class="form-text mt-2" id="tallyStatus">Last sync: Never.</div>

</div>

<div class="col-md-6">

<button class="btn btn-secondary w-100" disabled><i class="bi bi-journal-text"></i> View Tally Sync Audit Log</button>

<div class="form-text mt-2">Check detailed success/failure log for reconciliation status.</div>

</div>

</div>

<hr class="my-4">

<h5 class="text-danger"><i class="bi bi-life-preserver"></i> Automated Reconciliation Response</h5>

<p class="small text-success fw-bold">This system automatically handles success, failure, and overpayment from Tally:</p>

<ul class="list-group list-group-flush small">

<li class="list-group-item text-success"><i class="bi bi-check-circle-fill"></i> \*\*Success:\*\* Logs confirmation.</li>

<li class="list-group-item text-warning"><i class="bi bi-exclamation-triangle-fill"></i> \*\*Failure:\*\* Instantly sends parent a new \*\*Payment Link\*\* to resolve the issue.</li>

<li class="list-group-item text-info"><i class="bi bi-cash-stack"></i> \*\*Overpayment:\*\* Instantly sends parent confirmation: "Your excess amount (₹X.XX) will be refunded within 1-2 business days."</li>

</ul>

</div>

</div>

</div>

</div>

</div>

<!-- Tri-Party Verification Modal (POST /verification/triparty-initiate) -->

<div class="modal fade" id="verificationModal" tabindex="-1">

<div class="modal-dialog">

<div class="modal-content">

<div class="modal-header bg-info text-dark">

<h5 class="modal-title"><i class="bi bi-patch-check"></i> Tri-Party Verification</h5>

<button type="button" class="btn-close" data-bs-dismiss="modal"></button>

</div>

<form id="verificationForm">

<div class="modal-body">

<p class="text-muted small">Select a payment ID to initiate a confirmation request to the paying parent via Arattai/WhatsApp.</p>

<div class="mb-3">

<label for="verificationPaymentId" class="form-label">Payment ID to Verify</label>

<input type="text" class="form-control" id="verificationPaymentId" placeholder="e.g., payment\_uuid\_123" required>

</div>

</div>

<div class="modal-footer">

<button type="button" class="btn btn-secondary" data-bs-dismiss="modal">Cancel</button>

<button type="submit" class="btn btn-info" id="initiateVerificationBtn"><i class="bi bi-send"></i> Initiate Verification</button>

</div>

</form>

</div>

</div>

</div>

<!-- Bootstrap Bundle with Popper -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>

<script>

const API\_BASE = '';

const MOCK\_JWT\_TOKEN = 'mock.jwt.token.for.auth';

let currentPaymentPage = 1;

const paymentPageSize = 50;

let currentDefaulterPage = 1;

const defaulterPageSize = 50;

// --- MULTI-BRANCH MOCK DATA AND STATE ---

const mockBranches = [

{ id: '00000000-0000-0000-0000-000000000001', name: 'HQ - Chennai Main' },

{ id: 'BRANCH-002', name: 'Branch 2 - Velachery' },

{ id: 'BRANCH-003', name: 'Branch 3 - Anna Nagar' },

{ id: 'BRANCH-004', name: 'Branch 4 - T. Nagar' },

{ id: 'BRANCH-005', name: 'Branch 5 - Coimbatore' },

{ id: 'BRANCH-006', name: 'Branch 6 - Madurai' },

{ id: 'BRANCH-007', name: 'Branch 7 - Trichy' },

{ id: 'BRANCH-008', name: 'Branch 8 - Salem' },

{ id: 'BRANCH-009', name: 'Branch 9 - Erode' },

{ id: 'BRANCH-010', name: 'Branch 10 - Tirunelveli' }

];

let currentSchoolId = mockBranches[0].id;

// Mock Fee Structures (Simulate different structures per branch)

const branchFeeStructures = {

'00000000-0000-0000-0000-000000000001': [

{ name: 'Tuition - Annual', range: 'All', amount: 55000, applicability: 'Mandatory' },

{ name: 'Transport - Qtr', range: 'All', amount: 15000, applicability: 'Optional' },

],

'BRANCH-002': [

{ name: 'Tuition - Half Year', range: 'All', amount: 28000, applicability: 'Mandatory' },

{ name: 'Exam Fee', range: 'All', amount: 1500, applicability: 'Mandatory' },

],

// Default structure for other branches

default: [

{ name: 'Tuition', range: 'All', amount: 45000, applicability: 'Mandatory' },

{ name: 'Olympiad Fee', range: '9th-12th', amount: 500, applicability: 'Optional' },

]

};

/\*\* Helper function to show a temporary notification. \*/

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

const area = document.getElementById('notification-area');

area.innerHTML = `

<div class="alert alert-${type} alert-dismissible fade show" role="alert">

${message}

<button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>

</div>

`;

area.style.display = 'block';

setTimeout(() => { area.style.display = 'none'; }, 7000);

}

/\*\* Core Fetch Function with JWT Auth Header and Branch Filter. \*/

async function apiFetch(endpoint, method = 'GET', body = null) {

let url = `${API\_BASE}${endpoint}`;

// Append schoolId filter for multi-branch operation

const sep = url.includes('?') ? '&' : '?';

url += `${sep}schoolId=${currentSchoolId}`;

try {

const options = {

method: method,

headers: {

'Content-Type': 'application/json',

'Authorization': `Bearer ${MOCK\_JWT\_TOKEN}`

},

};

if (body && method !== 'GET') {

options.body = JSON.stringify(body);

}

const response = await fetch(url, options);

const data = await response.json();

if (!response.ok) {

throw new Error(data.error || `API error (${response.status})`);

}

return data;

} catch (error) {

console.error('API Fetch Error:', endpoint, error);

showNotification(error.message || 'Network error. Check console for details.', 'danger');

throw error;

}

}

// --- GLOBAL BRANCH HANDLER ---

function updateUIForBranch(branchId) {

const branch = mockBranches.find(b => b.id === branchId) || mockBranches[0];

const branchName = branch.name;

document.getElementById('currentBranchName').textContent = branchName;

document.getElementById('formBranchName').textContent = branchName;

document.getElementById('configBranchName').textContent = branchName;

document.getElementById('auditBranchName').textContent = branchName;

}

async function handleBranchChange(event) {

const newBranchId = event.target.value;

currentSchoolId = newBranchId;

updateUIForBranch(newBranchId);

showNotification(`Switching to ${mockBranches.find(b => b.id === newBranchId).name}. Reloading data...`, 'secondary');

// Reload all operational tabs' data

await fetchConfig();

await fetchPayments(1);

await fetchDefaulters();

}

function initBranchSelector() {

const selector = document.getElementById('branchSelector');

mockBranches.forEach(branch => {

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

option.value = branch.id;

option.textContent = branch.name;

selector.appendChild(option);

});

selector.value = currentSchoolId;

updateUIForBranch(currentSchoolId);

selector.addEventListener('change', handleBranchChange);

}

// --- FEE STRUCTURE DYNAMIC MANAGEMENT (Updated to be branch-aware) ---

function addFeeComponentToTable(component) {

const tableBody = document.getElementById('feeStructureBody');

const row = tableBody.insertRow();

row.setAttribute('data-component', component.name);

const cleanAmount = parseFloat(component.amount).toFixed(2);

row.innerHTML = `

<td data-field="name">${component.name}</td>

<td data-field="range">${component.range}</td>

<td data-field="amount" class="text-end" data-raw-amount="${cleanAmount}">₹${parseFloat(cleanAmount).toLocaleString('en-IN')}</td>

<td data-field="applicability">${component.applicability}</td>

<td><button type="button" class="btn btn-sm btn-outline-danger delete-fee-component">Del</button></td>

`;

// Add event listener for the delete button

row.querySelector('.delete-fee-component').addEventListener('click', (e) => {

const componentName = e.target.closest('tr').getAttribute('data-component');

if (confirm(`Are you sure you want to remove the component: ${componentName}?`)) {

e.target.closest('tr').remove();

showNotification(`Fee component "${componentName}" removed from table. Click Save to finalize.`, 'warning');

}

});

}

function loadFeeStructureForBranch(branchId) {

const tableBody = document.getElementById('feeStructureBody');

tableBody.innerHTML = '';

// Mock fetching the correct structure based on branchId

const structure = branchFeeStructures[branchId] || branchFeeStructures.default;

if (structure.length === 0) {

tableBody.innerHTML = '<tr><td colspan="5" class="text-center">No fee structure defined for this branch. Add new components above.</td></tr>';

} else {

structure.forEach(addFeeComponentToTable);

}

}

document.getElementById('addFeeComponentForm').addEventListener('submit', (e) => {

e.preventDefault();

const form = e.target;

const newComponent = {

name: form.newComponentName.value.trim(),

range: form.newClassRange.value.trim(),

amount: parseFloat(form.newAmount.value),

applicability: form.newApplicability.value

};

if (newComponent.name && newComponent.range && !isNaN(newComponent.amount) && newComponent.amount >= 0) {

addFeeComponentToTable(newComponent);

showNotification(`Component "${newComponent.name}" added successfully.`, 'success');

form.reset();

} else {

showNotification('Please fill in all required fields correctly (Amount must be a non-negative number).', 'danger');

}

});

function harvestFeeStructureData() {

const rows = document.querySelectorAll('#feeStructureBody tr');

const updates = [];

rows.forEach(row => {

const rawAmountElement = row.querySelector('[data-raw-amount]');

if (rawAmountElement) {

updates.push({

name: row.querySelector('[data-field="name"]').textContent,

classRange: row.querySelector('[data-field="range"]').textContent,

amount: parseFloat(rawAmountElement.getAttribute('data-raw-amount')),

applicability: row.querySelector('[data-field="applicability"]').textContent,

});

}

});

return updates;

}

// --- 1. PAYMENT COLLECTION LOGIC (Updated to be branch-aware) ---

async function fetchPayments(page = 1) {

currentPaymentPage = page;

const offset = (page - 1) \* paymentPageSize;

const tableBody = document.getElementById('paymentsTableBody');

tableBody.innerHTML = '<tr><td colspan="6" class="text-center"><div class="spinner-border spinner-border-sm"></div> Loading payments for ' + currentSchoolId + '...</td></tr>';

try {

// API call now automatically sends the currentSchoolId as a query parameter

const data = await apiFetch(`/payments?limit=${paymentPageSize}&offset=${offset}`);

tableBody.innerHTML = '';

if (data.payments.length === 0) {

tableBody.innerHTML = '<tr><td colspan="6" class="text-center">No payment records found for this branch.</td></tr>';

document.getElementById('nextPaymentBtn').disabled = true;

document.getElementById('prevPaymentBtn').disabled = true;

return;

}

// MOCKING: Filtering mock data based on schoolId is assumed to happen on the backend

data.payments.forEach(p => {

const row = tableBody.insertRow();

row.innerHTML = `

<td>${new Date(p.payment\_date).toLocaleDateString()}</td>

<td>${p.first\_name || 'N/A'} ${p.last\_name || ''}</td>

<td>₹${p.amount\_paid.toFixed(2)}</td>

<td>${p.payment\_method}</td>

<td class="small">${p.transaction\_id || 'CASH'}</td>

<td><span class="badge bg-success">RECONCILED</span></td>

`;

});

document.getElementById('prevPaymentBtn').disabled = currentPaymentPage === 1;

document.getElementById('nextPaymentBtn').disabled = data.payments.length < paymentPageSize;

} catch (error) {

tableBody.innerHTML = '<tr><td colspan="6" class="text-center text-danger">Failed to load payments. API Error.</td></tr>';

}

}

// --- 2. DEFAULTERS LOGIC (Updated to be branch-aware) ---

async function fetchDefaulters() {

const tableBody = document.getElementById('defaultersTableBody');

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

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

let query = `/defaulters?limit=${defaulterPageSize}`;

if (minOverdueDays) query += `&minOverdueDays=${minOverdueDays}`;

if (minAmount) query += `&minAmount=${minAmount}`;

tableBody.innerHTML = '<tr><td colspan="6" class="text-center"><div class="spinner-border spinner-border-sm"></div> Loading defaulters for ' + currentSchoolId + '...</td></tr>';

try {

// API call now automatically sends the currentSchoolId as a query parameter

const data = await apiFetch(query);

tableBody.innerHTML = '';

if (data.defaulterList.length === 0) {

tableBody.innerHTML = '<tr><td colspan="6" class="text-center">No defaulters found for this branch.</td></tr>';

return;

}

// MOCKING: Assuming backend filters by schoolId

data.defaulterList.forEach(d => {

const row = tableBody.insertRow();

row.classList.add('overdue-row');

row.innerHTML = `

<td>${d.student\_name || 'Mock Student'}</td>

<td>${d.class\_name || 'X-A'}</td>

<td class="fw-bold">₹${(d.amount || 25000).toFixed(2)}</td>

<td>${d.days\_overdue || '30'}</td>

<td>${d.parent\_phone || '+91...'}</td>

<td>${d.last\_alert\_date || 'N/A'}</td>

`;

});

} catch (error) {

tableBody.innerHTML = '<tr><td colspan="6" class="text-center text-danger">Failed to load defaulters. API Error.</td></tr>';

}

}

// --- 3. CONFIGURATION LOGIC (Updated to be branch-aware) ---

async function fetchConfig() {

try {

// API call now automatically sends the currentSchoolId as a query parameter

const data = await apiFetch('/config');

const config = data.config;

document.getElementById('lateFeePercent').value = config.lateFeePercent || 2.0;

document.getElementById('graceDays').value = config.graceDays || 15;

document.getElementById('accountingSyncEnabled').value = String(config.accountingSyncEnabled || true);

// Load the fee structure specific to the current branch

loadFeeStructureForBranch(currentSchoolId);

showNotification(`Configuration for ${document.getElementById('currentBranchName').textContent} loaded successfully.`, 'secondary');

} catch (error) {

console.warn('Could not load configuration. Using defaults.');

loadFeeStructureForBranch(currentSchoolId); // Load mock structure if API fails

}

}

document.getElementById('configForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const btn = document.getElementById('saveConfigBtn');

const feeStructureUpdates = harvestFeeStructureData();

const payload = {

lateFeePercent: parseFloat(form.lateFeePercent.value),

graceDays: parseInt(form.graceDays.value, 10),

accountingSyncEnabled: form.accountingSyncEnabled.value === 'true',

feeStructureUpdates: feeStructureUpdates,

discountUpdates: [{ name: 'Sibling', value: 10 }]

};

btn.disabled = true;

btn.innerHTML = '<span class="spinner-border spinner-border-sm"></span> Saving...';

try {

// PUT request implicitly targets the current schoolId via apiFetch

const data = await apiFetch('/config', 'PUT', payload);

showNotification(data.message + ` (${feeStructureUpdates.length} components saved for ${document.getElementById('currentBranchName').textContent}!)`, 'success');

} catch (error) {

// Handled by apiFetch

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-save"></i> Save All Configuration Rules';

}

});

// --- 4. TALLY & VERIFICATION LOGIC (Updated to be branch-aware) ---

document.getElementById('tallySyncBtn').addEventListener('click', async () => {

const btn = document.getElementById('tallySyncBtn');

const statusDiv = document.getElementById('tallyStatus');

btn.disabled = true;

btn.innerHTML = '<span class="spinner-border spinner-border-sm"></span> Initiating Sync...';

statusDiv.textContent = 'Initiating sync with Tally... (Backend will handle auto-fail/refund notifications)';

try {

// API call now automatically sends the currentSchoolId as a query parameter

const data = await apiFetch('/tally/sync-collection', 'POST', {});

showNotification(data.message + ` (Branch: ${document.getElementById('currentBranchName').textContent})`, 'success');

statusDiv.textContent = `Sync initiated at: ${new Date().toLocaleTimeString()}. Status: ${data.status.toUpperCase()}`;

} catch (error) {

statusDiv.textContent = 'Sync failed. Check API error.';

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-arrow-repeat"></i> Sync Fee Collection to Tally Now';

}

});

// --- PAYMENT LOGGING ---

document.getElementById('paymentForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const btn = document.getElementById('logPaymentBtn');

const payload = {

studentId: form.studentId.value,

amount: parseFloat(form.amountPaid.value),

method: form.paymentMethod.value,

transactionId: form.transactionId.value,

feeStructureId: form.feeStructureId.value,

// schoolId is automatically added by apiFetch

};

btn.disabled = true;

btn.innerHTML = '<span class="spinner-border spinner-border-sm"></span> Processing...';

try {

const data = await apiFetch('/payment', 'POST', payload);

showNotification(data.message + ` Payment ID: ${data.paymentId}`, 'success');

form.reset();

fetchPayments(1); // Refresh list

} catch (error) {

// Handled by apiFetch

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-plus-circle"></i> Log Payment';

}

});

// --- TRI-PARTY VERIFICATION ---

document.getElementById('verificationForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const btn = document.getElementById('initiateVerificationBtn');

const paymentId = form.verificationPaymentId.value;

btn.disabled = true;

btn.innerHTML = '<span class="spinner-border spinner-border-sm"></span> Initiating...';

try {

const data = await apiFetch('/verification/triparty-initiate', 'POST', { paymentId });

showNotification(data.message, 'info');

form.reset();

bootstrap.Modal.getInstance(document.getElementById('verificationModal')).hide();

} catch (error) {

// Handled by apiFetch

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-send"></i> Initiate Verification';

}

});

// --- DEFAULTER EXPORTS ---

async function handleDefaulterExport(format) {

const btn = document.getElementById(`exportDefaulters${format === 'csv' ? 'Csv' : 'Pdf'}`);

btn.disabled = true;

const originalHtml = btn.innerHTML;

btn.innerHTML = `<span class="spinner-border spinner-border-sm"></span> Generating...`;

try {

const response = await apiFetch(`/defaulters?export=${format}`, 'GET');

if (format === 'csv') {

// In a real browser environment, the backend would stream the file.

// Here we simulate the download link message.

showNotification('CSV generation initiated. File download should start shortly.', 'success');

} else {

showNotification(response.message || 'PDF report generation queued successfully.', 'success');

}

} catch (error) {

// Handled by apiFetch

} finally {

btn.disabled = false;

btn.innerHTML = originalHtml;

}

}

// --- Initialization ---

document.addEventListener('DOMContentLoaded', () => {

initBranchSelector(); // Initialize the multi-branch logic first

// Initial data loads triggered by the branch handler

// Set up pagination buttons

document.getElementById('prevPaymentBtn').addEventListener('click', () => { if (currentPaymentPage > 1) fetchPayments(currentPaymentPage - 1); });

document.getElementById('nextPaymentBtn').addEventListener('click', () => { fetchPayments(currentPaymentPage + 1); });

document.getElementById('defaulterFilterForm').addEventListener('submit', (e) => { e.preventDefault(); fetchDefaulters(); });

// Set up export buttons

document.getElementById('exportDefaultersCsv').addEventListener('click', () => handleDefaulterExport('csv'));

document.getElementById('exportDefaultersPdf').addEventListener('click', () => handleDefaulterExport('pdf'));

// Set up automated alert button

document.getElementById('sendAlertsBtn').addEventListener('click', async () => {

const confirmAlert = confirm("Are you sure you want to trigger automated daily reminders to all filtered defaulters now?");

if (!confirmAlert) return;

const btn = document.getElementById('sendAlertsBtn');

btn.disabled = true;

btn.innerHTML = '<span class="spinner-border spinner-border-sm"></span> Sending...';

try {

const data = await apiFetch('/defaulters/auto-alert', 'POST', {});

showNotification(data.message, 'success');

} catch (error) {

// Handled by apiFetch

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-bell"></i> Send Daily Reminders';

}

});

});

</script>

</body>

</html>

12. client/dist/modules/Fee\_Tally\_Integration.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Fee Management with Tally Integration - SmartGenEduX</title>

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

<link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css" rel="stylesheet">

<style>

.gradient-header {

background-image: linear-gradient(to right, #10b981, #3b82f6); /\* Green-600 to Blue-600 \*/

}

.container {

max-width: 1400px;

}

</style>

</head>

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

<div class="container mx-auto px-4 py-8">

<!-- Header -->

<div class="gradient-header rounded-lg shadow-xl p-6 mb-6 text-white">

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

<div class="flex items-center">

<!-- Placeholder for logo image -->

<div class="w-12 h-12 mr-4 bg-white rounded-lg p-2 flex items-center justify-center">

<i class="fas fa-wallet text-2xl text-green-600"></i>

</div>

<div>

<h1 class="text-2xl font-bold">Fee Management with Tally Integration</h1>

<p class="text-blue-100">Advanced accounting integration with automated bookkeeping across all branches</p>

</div>

</div>

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

<span class="bg-yellow-500 text-yellow-900 px-3 py-1 rounded-full text-sm font-semibold">PREMIUM</span>

<button onclick="window.close()" class="bg-white bg-opacity-20 text-white px-4 py-2 rounded hover:bg-opacity-30">

<i class="fas fa-times mr-2"></i>Close

</button>

</div>

</div>

</div>

<!-- Tally Integration Status & Multi-Branch Selector -->

<div class="grid grid-cols-1 md:grid-cols-5 gap-6 mb-6">

<div class="md:col-span-1 bg-white rounded-lg shadow-md p-6">

<label for="branchSelector" class="block text-sm font-bold text-gray-700 mb-2">Select Branch</label>

<select id="branchSelector" onchange="fetchBranchData(this.value)" class="w-full border-2 border-blue-300 rounded-lg px-3 py-2 text-gray-800 focus:ring-blue-500 focus:border-blue-500">

<option value="BRANCH-001" selected>Branch 1 (HQ)</option>

<option value="BRANCH-002">Branch 2 (South)</option>

<option value="BRANCH-003">Branch 3 (North)</option>

<option value="BRANCH-004">Branch 4 (East)</option>

</select>

</div>

<div class="bg-gradient-to-br from-green-500 to-green-600 text-white rounded-lg p-6 flex items-center">

<i class="fas fa-link text-3xl mr-4"></i>

<div>

<div class="text-lg font-bold" id="statusTally">Connected</div>

<div class="text-green-100">Tally Integration</div>

</div>

</div>

<div class="bg-gradient-to-br from-blue-500 to-blue-600 text-white rounded-lg p-6 flex items-center">

<i class="fas fa-sync text-3xl mr-4"></i>

<div>

<div class="text-2xl font-bold" id="statusSynced">0</div>

<div class="text-blue-100">Auto Synced (Today)</div>

</div>

</div>

<div class="bg-gradient-to-br from-purple-500 to-purple-600 text-white rounded-lg p-6 flex items-center">

<i class="fas fa-calculator text-3xl mr-4"></i>

<div>

<div class="text-2xl font-bold" id="statusCollection">₹0.00</div>

<div class="text-purple-100">Today's Collection</div>

</div>

</div>

<div class="bg-gradient-to-br from-orange-500 to-orange-600 text-white rounded-lg p-6 flex items-center">

<i class="fas fa-chart-line text-3xl mr-4"></i>

<div>

<div class="text-2xl font-bold" id="statusAccuracy">--%</div>

<div class="text-orange-100">Sync Accuracy</div>

</div>

</div>

</div>

<!-- Tally Configuration & Advanced Features -->

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

<!-- Tally Connection Settings -->

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

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

<div class="w-12 h-12 bg-green-100 rounded-lg flex items-center justify-center mr-4">

<i class="fas fa-server text-green-600 text-xl"></i>

</div>

<div>

<h2 class="text-xl font-bold">Tally Server Configuration</h2>

<p class="text-gray-600">Configure Tally ERP integration settings</p>

</div>

</div>

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

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

<div class="flex items-center">

<i class="fas fa-check-circle text-green-600 mr-3"></i>

<div>

<h3 class="font-semibold text-green-900">Current Branch (B-1) Webhook URL:</h3>

<p class="text-sm text-green-700 break-all">/tally/webhook?schoolId=<span id="webhookBranchId">BRANCH-001</span></p>

</div>

</div>

</div>

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

<div>

<label class="block text-sm font-medium mb-2">Tally Server IP</label>

<input type="text" id="tallyIP" value="192.168.1.100" class="w-full border rounded px-3 py-2" placeholder="192.168.1.100">

</div>

<div>

<label class="block text-sm font-medium mb-2">Port</label>

<input type="text" id="tallyPort" value="9000" class="w-full border rounded px-3 py-2" placeholder="9000">

</div>

</div>

<div>

<label class="block text-sm font-medium mb-2">Company Database</label>

<select id="companyDB" class="w-full border rounded px-3 py-2">

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

<option value="school\_accounts\_2025" selected>School Accounts 2025</option>

<option value="previous\_year\_data">Previous Year Data</option>

</select>

</div>

<div class="flex space-x-3 border-t pt-4">

<button type="button" onclick="testConnection()" class="bg-blue-600 text-white px-4 py-2 rounded hover:bg-blue-700">

<i class="fas fa-plug mr-2"></i>Test Connection

</button>

<button type="button" onclick="syncNow()" class="bg-green-600 text-white px-4 py-2 rounded hover:bg-green-700">

<i class="fas fa-sync mr-2"></i>Sync Now (Manual)

</button>

</div>

</form>

</div>

<!-- Automated Accounting -->

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

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

<div class="w-12 h-12 bg-blue-100 rounded-lg flex items-center justify-center mr-4">

<i class="fas fa-robot text-blue-600 text-xl"></i>

</div>

<div>

<h2 class="text-xl font-bold">Automated Financial Logic</h2>

<p class="text-gray-600">Smart ledger, reconciliation failure, and refund management</p>

</div>

</div>

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

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

<h3 class="font-semibold text-red-900 mb-2"><i class="fas fa-exclamation-triangle mr-2"></i>Automated Failure/Refund Triggers</h3>

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

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

<span>Tally Fail → Auto Repayment Link to Parent</span>

<span class="text-green-600 font-bold">Enabled</span>

</label>

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

<span>Overpayment Detected → Auto Refund Notification</span>

<span class="text-green-600 font-bold">Enabled</span>

</label>

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

<span>Tri-Party Verification Status</span>

<span class="text-blue-600 font-bold">Integrated</span>

</label>

</div>

</div>

<div class="border-t pt-4">

<h3 class="font-semibold mb-2">Today's Auto Entries</h3>

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

<div class="flex justify-between">

<span>Fee collections processed</span>

<span class="font-bold" id="autoEntriesProcessed">0 entries</span>

</div>

<div class="flex justify-between">

<span>Vouchers generated</span>

<span class="font-bold" id="vouchersGenerated">0 vouchers</span>

</div>

<div class="flex justify-between">

<span>Reconciliation match status</span>

<span class="text-green-600 font-bold" id="reconciliationMatch">Pending</span>

</div>

</div>

</div>

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

<button onclick="reviewPending()" class="bg-orange-600 text-white px-4 py-2 rounded hover:bg-orange-700 text-sm">

<i class="fas fa-eye mr-2"></i>Review Pending Vouchers

</button>

<button onclick="bulkApprove()" class="bg-purple-600 text-white px-4 py-2 rounded hover:bg-purple-700 text-sm">

<i class="fas fa-check-double mr-2"></i>Bulk Approve Vouchers

</button>

</div>

</div>

</div>

</div>

<!-- Real-time Financial Reports & Audit -->

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

<!-- Financial Analytics -->

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

<h2 class="text-xl font-bold mb-4 flex items-center"><i class="fas fa-chart-bar text-purple-600 mr-2"></i> Financial Analytics & Reporting</h2>

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

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

<div class="bg-green-50 rounded-lg p-3 text-center">

<div class="text-lg font-bold text-green-600" id="monthlyCollection">₹0.00</div>

<div class="text-sm text-green-700">This Month Collection</div>

</div>

<div class="bg-blue-50 rounded-lg p-3 text-center">

<div class="text-lg font-bold text-blue-600" id="pendingDues">₹0.00</div>

<div class="text-sm text-blue-700">Total Pending Dues</div>

</div>

</div>

<div class="space-y-3">

<button onclick="generateTrialBalance()" class="w-full bg-purple-600 text-white py-2 rounded hover:bg-purple-700 text-sm">

<i class="fas fa-balance-scale mr-2"></i>Generate Trial Balance

</button>

<button onclick="defaultersList()" class="w-full bg-orange-600 text-white py-2 rounded hover:bg-orange-700 text-sm">

<i class="fas fa-exclamation-triangle mr-2"></i>Defaulters List (Multi-Branch)

</button>

<button onclick="reconciliationReport()" class="w-full bg-blue-600 text-white py-2 rounded hover:bg-blue-700 text-sm">

<i class="fas fa-check-double mr-2"></i>Full Reconciliation Report

</button>

</div>

</div>

</div>

<!-- Sync History & Audit Trail -->

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

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

<h2 class="text-xl font-bold">Sync History & Audit Trail</h2>

<button onclick="exportSyncLog()" class="bg-blue-600 text-white px-4 py-2 rounded hover:bg-blue-700 text-sm">

<i class="fas fa-download mr-2"></i>Export Log

</button>

</div>

<div class="overflow-x-auto h-64">

<table class="w-full text-sm">

<thead class="bg-gray-50">

<tr>

<th class="px-4 py-2 text-left">Sync Time</th>

<th class="px-4 py-2 text-left">Records</th>

<th class="px-4 py-2 text-left">Status</th>

<th class="px-4 py-2 text-left">Details</th>

</tr>

</thead>

<tbody id="syncHistoryBody">

<tr class="border-t">

<td class="px-4 py-2">5 minutes ago</td>

<td class="px-4 py-2">12 entries</td>

<td class="px-4 py-2"><span class="px-2 py-1 bg-green-100 text-green-800 rounded text-xs">Success</span></td>

<td class="px-4 py-2">Posted Vouchers to Tally</td>

</tr>

<tr class="border-t">

<td class="px-4 py-2">Yesterday 17:00</td>

<td class="px-4 py-2">78 entries</td>

<td class="px-4 py-2"><span class="px-2 py-1 bg-yellow-100 text-yellow-800 rounded text-xs">Failure Fix</span></td>

<td class="px-4 py-2">Auto-sent 3 repayment links</td>

</tr>

</tbody>

</table>

</div>

</div>

</div>

</div>

<!-- Script -->

<script>

const API\_BASE = '';

let currentBranchId = document.getElementById('branchSelector').value;

// --- Helper Functions ---

function showMessage(message, type) {

const colors = { 'success': 'bg-green-600', 'error': 'bg-red-600', 'info': 'bg-blue-600', 'warning': 'bg-yellow-600' };

const icons = { 'success': 'fas fa-check', 'error': 'fas fa-exclamation-triangle', 'info': 'fas fa-info-circle', 'warning': 'fas fa-bell' };

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

messageDiv.className = `fixed top-4 right-4 ${colors[type]} text-white px-6 py-3 rounded-lg shadow-lg z-50`;

messageDiv.innerHTML = `<i class="${icons[type]} mr-2"></i>${message}`;

document.body.appendChild(messageDiv);

setTimeout(() => { messageDiv.remove(); }, 4000);

}

// Simulates fetching data based on branch ID

function fetchBranchData(branchId) {

currentBranchId = branchId;

document.getElementById('webhookBranchId').textContent = branchId;

showMessage(`Switching to Branch ${branchId}...`, 'info');

// --- Simulate Real-time Data Fetch ---

const collection = Math.floor(Math.random() \* 500000);

const accuracy = (98 + Math.random() \* 2).toFixed(1);

const synced = Math.floor(Math.random() \* 100) + 20;

document.getElementById('statusTally').textContent = 'Connected';

document.getElementById('statusSynced').textContent = synced;

document.getElementById('statusCollection').textContent = `₹${collection.toLocaleString('en-IN')}`;

document.getElementById('statusAccuracy').textContent = `${accuracy}%`;

document.getElementById('monthlyCollection').textContent = `₹${(collection \* 5).toLocaleString('en-IN')}`;

document.getElementById('pendingDues').textContent = `₹${(collection \* 0.1).toLocaleString('en-IN')}`;

document.getElementById('autoEntriesProcessed').textContent = `${synced} entries`;

document.getElementById('vouchersGenerated').textContent = `${synced} vouchers`;

}

// --- Tally Actions ---

function testConnection() {

showMessage('Testing Tally server connection...', 'info');

setTimeout(() => {

showMessage('Connection successful! Tally ERP 9 detected.', 'success');

}, 2000);

}

async function syncNow() {

showMessage('Initiating manual sync with Tally...', 'info');

// This hits POST /tally/sync-collection?schoolId={currentBranchId}

try {

// Mock API call initiation

// const response = await fetch(`${API\_BASE}/tally/sync-collection?schoolId=${currentBranchId}`, { method: 'POST' });

setTimeout(() => {

const failed = Math.random() > 0.8;

if (failed) {

showMessage('Sync initiated. WAITING FOR TALLY WEBHOOK... (Failure scenario upcoming)', 'warning');

// Simulate delayed Tally failure webhook that triggers auto-repayment link

setTimeout(() => {

showMessage('Tally reports FAILURE! Auto-Repayment Link sent to parent.', 'error');

}, 5000);

} else {

showMessage('Sync initiated. WAITING FOR TALLY WEBHOOK... (Success scenario upcoming)', 'info');

// Simulate delayed Tally success webhook

setTimeout(() => {

showMessage('Tally confirms success! Vouchers reconciled.', 'success');

}, 4000);

}

}, 1000);

} catch (error) {

showMessage('Failed to initiate sync due to network error.', 'error');

}

}

// --- Voucher Management Actions ---

function reviewPending() {

showMessage('Opening voucher review panel for manual posting...', 'info');

}

function bulkApprove() {

showMessage('Bulk approving pending vouchers...', 'info');

setTimeout(() => {

showMessage('All pending vouchers approved and posted to Tally.', 'success');

}, 1500);

}

// --- Report Actions ---

function generateTrialBalance() {

showMessage('Generating Trial Balance for selected branch...', 'info');

setTimeout(() => { showMessage('Trial Balance generated and ready for download.', 'success'); }, 2000);

}

function defaultersList() {

showMessage('Generating Defaulters List across all branches...', 'info');

setTimeout(() => { showMessage('Defaulters list ready. Sent to Admin email.', 'success'); }, 2000);

}

function reconciliationReport() {

showMessage('Running Full Reconciliation Report...', 'info');

setTimeout(() => { showMessage('Reconciliation report generated. Check audit trail for details.', 'success'); }, 3000);

}

// --- Audit Actions ---

function exportSyncLog() {

showMessage('Exporting sync history log for Branch ' + currentBranchId + '...', 'success');

}

// --- Initialization ---

document.addEventListener('DOMContentLoaded', () => {

// Load initial data for the default branch (BRANCH-001)

fetchBranchData(currentBranchId);

});

</script>

</body>

</html>

13. client/dist/modules/Id\_card\_generator.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>🆔 ID Card Generator & Management</title>

<!-- Bootstrap 5 CDN -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css">

<style>

body { background-color: #f0f4f8; font-family: 'Inter', sans-serif; }

.container-fluid { max-width: 1200px; margin-top: 20px; }

.card-header { background-color: #3f51b5; /\* Indigo \*/ color: white; font-weight: bold; border-radius: 0.75rem 0.75rem 0 0; }

.section-card { margin-bottom: 25px; box-shadow: 0 4px 12px rgba(0, 0, 0, 0.1); border-radius: 0.75rem; }

.alert-fixed { position: fixed; top: 20px; right: 20px; z-index: 1050; min-width: 300px; border-radius: 0.5rem; }

.nav-tabs .nav-link.active { background-color: #3f51b5; color: white; border-color: #3f51b5; }

/\* ID Card Preview Styles \*/

.id-card {

width: 300px; height: 450px; background: linear-gradient(135deg, #e3f2fd, #bbdefb);

border-radius: 12px; padding: 15px; box-shadow: 0 4px 15px rgba(0,0,0,0.2);

position: relative; overflow: hidden; font-size: 14px;

}

.id-card-header { text-align: center; margin-bottom: 10px; border-bottom: 2px solid #3f51b5; padding-bottom: 5px; }

.id-photo { width: 100px; height: 100px; border-radius: 50%; object-fit: cover; border: 3px solid #3f51b5; margin: 0 auto 10px; display: block; }

.id-detail { margin-bottom: 5px; }

.id-barcode { text-align: center; margin-top: 15px; border-top: 1px dashed #999; padding-top: 10px; }

</style>

</head>

<body>

<div class="container-fluid">

<h1 class="text-center mb-4 text-primary"><i class="bi bi-person-badge"></i> ID Card Generation & Approval Portal</h1>

<!-- Notification Area -->

<div id="notification-area" class="alert-fixed" style="display: none;"></div>

<!-- Tab Navigation -->

<ul class="nav nav-tabs" id="idCardTab" role="tablist">

<li class="nav-item"><button class="nav-link active" id="request-tab" data-bs-toggle="tab" data-bs-target="#request-submission" type="button"><i class="bi bi-file-earmark-plus"></i> New Request</button></li>

<li class="nav-item"><button class="nav-link" id="workflow-tab" data-bs-toggle="tab" data-bs-target="#workflow-review" type="button"><i class="bi bi-list-check"></i> Approval Workflow</button></li>

<li class="nav-item"><button class="nav-link" id="config-tab" data-bs-toggle="tab" data-bs-target="#config-setup" type="button"><i class="bi bi-gear-fill"></i> Configuration Hub</button></li>

</ul>

<!-- Tab Content -->

<div class="tab-content pt-3" id="idCardTabContent">

<!-- 1. Request Submission Tab (POST /requests, POST /applications/:id/upload-document) -->

<div class="tab-pane fade show active" id="request-submission" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-success">Step 1: Data Submission & Photo Upload</div>

<div class="card-body">

<p class="alert alert-info small"><i class="bi bi-info-circle"></i> Submitting this form automatically \*\*generates and links a unique Barcode UID\*\* for the Attendance System, ensuring compliance for the new ID.</p>

<form id="requestForm" class="row g-3">

<div class="col-md-4">

<label for="userType" class="form-label">User Type</label>

<select class="form-select" id="userType" required>

<option value="student">Student</option>

<option value="staff">Staff / Teacher</option>

</select>

</div>

<div class="col-md-8">

<label for="targetEntityId" class="form-label">Target Profile ID (Student/Staff Internal ID)</label>

<input type="text" class="form-control" id="targetEntityId" placeholder="e.g., UUID or Internal ID (required for barcode link)" required>

</div>

<div class="col-md-6">

<label for="fullName" class="form-label">Full Name on Card</label>

<input type="text" class="form-control" id="fullName" required>

</div>

<div class="col-md-6">

<label for="cardTitle" class="form-label">Title/Designation (e.g., Class X-B, Physics Teacher)</label>

<input type="text" class="form-control" id="cardTitle" required>

</div>

<h6 class="mt-4">Required Data for Card</h6>

<div class="col-md-4"><input type="text" class="form-control form-control-sm" placeholder="Contact Phone" id="contactPhone" required></div>

<div class="col-md-4"><input type="text" class="form-control form-control-sm" placeholder="Emergency Contact" id="emergencyContact"></div>

<div class="col-md-4"><input type="date" class="form-control form-control-sm" id="dob" required></div>

<div class="col-12">

<button type="submit" class="btn btn-success" id="submitRequestBtn"><i class="bi bi-file-earmark-arrow-up"></i> Submit Data & Generate Barcode</button>

</div>

</form>

<hr class="my-4">

<!-- Photo Upload (Separate Step) -->

<h5 class="text-primary">Step 2: Upload Profile Photo</h5>

<form id="photoUploadForm" class="row g-3" enctype="multipart/form-data">

<div class="col-md-6">

<label for="requestIdUpload" class="form-label">Request ID (from Step 1)</label>

<input type="text" class="form-control" id="requestIdUpload" placeholder="Paste Request ID here" required>

</div>

<div class="col-md-6">

<label for="profilePhoto" class="form-label">Photo (JPEG/PNG, Max 10MB)</label>

<input type="file" class="form-control" id="profilePhoto" name="document" accept=".jpg,.jpeg,.png" required>

</div>

<div class="col-12">

<button type="submit" class="btn btn-primary" id="uploadPhotoBtn"><i class="bi bi-image"></i> Upload Photo & Finalize Request</button>

</div>

</form>

</div>

</div>

</div>

<!-- 2. Approval Workflow Tab (GET /requests, POST /requests/:id/generate, POST /requests/:id/approve, GET /requests/:id/print-data) -->

<div class="tab-pane fade" id="workflow-review" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-warning text-dark">ID Card Approval & Printing Gate (Principal Control)</div>

<div class="card-body">

<div class="d-flex justify-content-between mb-3">

<h5>Pending Requests</h5>

<button class="btn btn-sm btn-outline-primary" id="refreshRequestsBtn"><i class="bi bi-arrow-clockwise"></i> Refresh List</button>

</div>

<div class="table-responsive">

<table class="table table-striped table-sm">

<thead>

<tr>

<th>ID</th>

<th>Target Type</th>

<th>Name (Mock)</th>

<th>Status</th>

<th>Actions</th>

</tr>

</thead>

<tbody id="requestsTableBody">

<tr><td colspan="5" class="text-center">Loading requests...</td></tr>

</tbody>

</table>

</div>

</div>

</div>

<!-- Approval Modal (for Principal/Admin) -->

<div class="modal fade" id="approvalModal" tabindex="-1">

<div class="modal-dialog modal-lg">

<div class="modal-content">

<div class="modal-header bg-primary text-white">

<h5 class="modal-title"><i class="bi bi-card-checklist"></i> Review & Finalize Card Request</h5>

<button type="button" class="btn-close" data-bs-dismiss="modal"></button>

</div>

<div class="modal-body row">

<div class="col-md-6">

<h6>Request Details (ID: <span id="modalRequestId"></span>)</h6>

<ul class="list-unstyled small" id="modalRequestDetails"></ul>

<button class="btn btn-sm btn-info mt-3" id="finalizeGenerateBtn"><i class="bi bi-send"></i> Submit for Approval Gate</button>

</div>

<div class="col-md-6">

<h6>ID Card Preview</h6>

<div id="cardPreviewContainer" class="d-flex justify-content-center">

<!-- ID Card Preview will render here -->

<div class="id-card">

<div class="id-card-header">SCHOOL NAME / LOGO</div>

<img src="https://placehold.co/100x100/3f51b5/ffffff?text=PHOTO" alt="Profile" class="id-photo">

<div class="id-detail fw-bold text-center" id="previewName">N/A</div>

<div class="id-detail text-center text-muted" id="previewTitle">N/A</div>

<div class="id-detail mt-3">ID: <span id="previewEntityId">N/A</span></div>

<div class="id-detail">Phone: <span id="previewPhone">N/A</span></div>

<div class="id-barcode">

<small>Barcode UID:</small>

<div class="fw-bold" id="previewBarcode">N/A</div>

</div>

</div>

</div>

</div>

</div>

<div class="modal-footer justify-content-between">

<button class="btn btn-outline-success" id="principalApproveBtn" disabled><i class="bi bi-check-circle"></i> Principal Approve (Unlocks Print)</button>

<button class="btn btn-outline-danger" id="rejectBtn" disabled><i class="bi bi-x-circle"></i> Reject</button>

<button class="btn btn-secondary" data-bs-dismiss="modal">Close</button>

</div>

</div>

</div>

</div>

</div>

<!-- 3. Configuration Hub Tab (GET/PUT /config) -->

<div class="tab-pane fade" id="config-setup" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-danger">ID Card Configuration & Compliance</div>

<div class="card-body">

<p class="alert alert-warning small"><i class="bi bi-lock-fill"></i> Access to configuration is restricted to \*\*Admin/Principal\*\* roles only.</p>

<form id="configForm" class="row g-3">

<h5 class="text-danger">A. Design & Template Settings</h5>

<div class="col-md-6">

<label for="templateSelector" class="form-label">Card Template Version</label>

<select class="form-select" id="templateSelector" name="template">

<option value="modern\_student\_v1">Modern Student (v1)</option>

<option value="staff\_compliance\_v2">Staff Compliance (v2)</option>

<option value="minimal\_barcode\_v3">Minimal Barcode (v3)</option>

</select>

</div>

<div class="col-md-6">

<label for="barcodeStyle" class="form-label">Barcode Type for Attendance</label>

<select class="form-select" id="barcodeStyle" name="barcodeStyle">

<option value="QR">QR Code (Attendance + Info)</option>

<option value="Code128">Code 128 (Standard)</option>

</select>

<div class="form-text">Used by Attendance module for quick check-in.</div>

</div>

<hr class="my-3">

<h5 class="text-danger">B. Compliance & Expiration Rules</h5>

<div class="col-md-6">

<label for="expirationPolicy" class="form-label">Card Expiration Policy</label>

<select class="form-select" id="expirationPolicy" name="expirationPolicy">

<option value="12 months">12 Months (Standard)</option>

<option value="academic\_year\_end">End of Academic Year (March 31)</option>

<option value="24 months">24 Months (Staff)</option>

</select>

<div class="form-text">Sets the validity period before re-issuance is required.</div>

</div>

<div class="col-md-6">

<label for="requiredFields" class="form-label">Mandatory Fields on Card (e.g., Name, Class, DOB)</label>

<input type="text" class="form-control" id="requiredFields" name="requiredFields" placeholder="name, class, dob, emergency\_contact">

<div class="form-text">Comma-separated list used during data submission validation.</div>

</div>

<div class="col-12 mt-4">

<button type="submit" class="btn btn-danger btn-lg w-100" id="saveConfigBtn"><i class="bi bi-save"></i> Save Configuration</button>

</div>

</form>

</div>

</div>

</div>

<!-- Hidden Print Data Modal -->

<div class="modal fade" id="printModal" tabindex="-1">

<div class="modal-dialog">

<div class="modal-content">

<div class="modal-header bg-success text-white">

<h5 class="modal-title">Authorized Print Data</h5>

<button type="button" class="btn-close" data-bs-dismiss="modal"></button>

</div>

<div class="modal-body">

<p class="text-success fw-bold">This data is ready for the printer (Status: APPROVED)</p>

<pre id="printDataContainer" class="bg-light p-3 border rounded small"></pre>

</div>

<div class="modal-footer">

<button type="button" class="btn btn-secondary" data-bs-dismiss="modal">Close</button>

<button type="button" class="btn btn-success"><i class="bi bi-printer"></i> Print ID Card</button>

</div>

</div>

</div>

</div>

</div>

</div>

<!-- Bootstrap Bundle with Popper -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>

<script>

const API\_BASE = '';

const MOCK\_JWT\_TOKEN = 'mock.jwt.token.for.auth';

let currentRequests = [];

/\*\* Helper function to show a temporary notification. \*/

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

const area = document.getElementById('notification-area');

area.innerHTML = `

<div class="alert alert-${type} alert-dismissible fade show" role="alert">

${message}

<button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>

</div>

`;

area.style.display = 'block';

setTimeout(() => { area.style.display = 'none'; }, 7000);

}

/\*\* Core Fetch Function with JWT Auth Header. \*/

async function apiFetch(endpoint, method = 'GET', body = null, isFormData = false) {

try {

const options = {

method: method,

headers: {

'Authorization': `Bearer ${MOCK\_JWT\_TOKEN}`

},

};

if (body && method !== 'GET') {

if (!isFormData) {

options.headers['Content-Type'] = 'application/json';

options.body = JSON.stringify(body);

} else {

// For FormData, let the browser set Content-Type to boundary/multipart

options.body = body;

}

}

const response = await fetch(`${API\_BASE}${endpoint}`, options);

const data = await response.json();

if (!response.ok) {

throw new Error(data.error || `API error (${response.status})`);

}

return data;

} catch (error) {

console.error('API Fetch Error:', endpoint, error);

showNotification(error.message || 'Network error. Check console for details.', 'danger');

throw error;

}

}

// --- 1. REQUEST SUBMISSION LOGIC ---

document.getElementById('requestForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const btn = document.getElementById('submitRequestBtn');

const payload = {

targetEntityId: form.targetEntityId.value.trim(),

userType: form.userType.value,

name: form.fullName.value.trim(),

cardTitle: form.cardTitle.value.trim(),

contactPhone: form.contactPhone.value.trim(),

emergencyContact: form.emergencyContact.value.trim(),

dob: form.dob.value,

// Additional data (mocked fields to ensure request\_data is populated)

address: "Mock Address Line 1",

bloodGroup: "O+",

issueDate: new Date().toISOString().split('T')[0]

};

// NOTE: The entire payload object goes into request\_data in the DB

btn.disabled = true;

btn.innerHTML = '<span class="spinner-border spinner-border-sm"></span> Submitting...';

try {

const data = await apiFetch('/requests', 'POST', payload);

showNotification(`Request submitted. Barcode UID generated: ${data.barcodeUid}. Request ID: ${data.request.id}`, 'success');

// Automatically populate the photo upload form for the next step

document.getElementById('requestIdUpload').value = data.request.id;

form.reset();

} catch (error) {

// Handled by apiFetch

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-file-earmark-arrow-up"></i> Submit Data & Generate Barcode';

}

});

document.getElementById('photoUploadForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const btn = document.getElementById('uploadPhotoBtn');

const requestId = form.requestIdUpload.value.trim();

if (!requestId) {

showNotification('Please submit data first and paste the Request ID.', 'warning');

return;

}

if (!form.profilePhoto.files[0]) {

showNotification('Please select a photo file to upload.', 'warning');

return;

}

const formData = new FormData();

formData.append('document', form.profilePhoto.files[0]);

formData.append('documentType', 'profile\_photo'); // Used by backend for JSONB update

btn.disabled = true;

btn.innerHTML = '<span class="spinner-border spinner-border-sm"></span> Uploading...';

try {

// Mock endpoint POST /applications/:id/upload-document is used here

const data = await apiFetch(`/applications/${requestId}/upload-document`, 'POST', formData, true);

showNotification(`Photo uploaded successfully! The request is now ready for generation and Principal review.`, 'success');

form.reset();

form.requestIdUpload.value = requestId; // Keep ID present

// Trigger refresh on the workflow tab

fetchRequests();

} catch (error) {

// Handled by apiFetch

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-image"></i> Upload Photo & Finalize Request';

}

});

// --- 2. APPROVAL WORKFLOW LOGIC ---

async function fetchRequests() {

const tableBody = document.getElementById('requestsTableBody');

tableBody.innerHTML = '<tr><td colspan="5" class="text-center"><div class="spinner-border spinner-border-sm"></div> Fetching requests...</td></tr>';

currentRequests = [];

try {

const data = await apiFetch('/requests');

currentRequests = data.data;

tableBody.innerHTML = '';

if (currentRequests.length === 0) {

tableBody.innerHTML = '<tr><td colspan="5" class="text-center">No pending ID card requests.</td></tr>';

return;

}

currentRequests.forEach(req => {

const row = tableBody.insertRow();

const reqData = req.request\_data;

const statusClass = {

'pending\_data': 'bg-secondary',

'pending\_approval': 'bg-warning text-dark',

'approved': 'bg-success',

'rejected': 'bg-danger'

}[req.status] || 'bg-light text-dark';

row.innerHTML = `

<td class="small">${req.id.substring(0, 8)}...</td>

<td>${req.target\_entity\_type.toUpperCase()}</td>

<td>${reqData.name || 'N/A'}</td>

<td><span class="badge ${statusClass}">${req.status.toUpperCase()}</span></td>

<td>

<button class="btn btn-sm btn-outline-primary review-btn" data-bs-toggle="modal" data-bs-target="#approvalModal" data-request-id="${req.id}">Review</button>

${req.status === 'approved' ? `<button class="btn btn-sm btn-success print-btn" data-request-id="${req.id}"><i class="bi bi-printer"></i> Print</button>` : ''}

</td>

`;

});

} catch (error) {

tableBody.innerHTML = '<tr><td colspan="5" class="text-center text-danger">Failed to load requests.</td></tr>';

}

}

// Handle Review Button Click (Populate Modal)

document.getElementById('requestsTableBody').addEventListener('click', (e) => {

if (e.target.classList.contains('review-btn')) {

const requestId = e.target.getAttribute('data-request-id');

const request = currentRequests.find(r => r.id === requestId);

if (!request) return;

const data = request.request\_data;

document.getElementById('modalRequestId').textContent = requestId.substring(0, 12) + '...';

document.getElementById('modalRequestDetails').innerHTML = `

<li><strong>Status:</strong> <span class="badge bg-warning text-dark">${request.status.toUpperCase()}</span></li>

<li><strong>Name:</strong> ${data.name || 'N/A'}</li>

<li><strong>Title:</strong> ${data.cardTitle || 'N/A'}</li>

<li><strong>Profile ID:</strong> ${request.target\_entity\_id}</li>

<li><strong>Barcode UID:</strong> ${data.barcodeUid || 'N/A'}</li>

<li><strong>Phone:</strong> ${data.contactPhone || 'N/A'}</li>

<li><strong>Photo Status:</strong> ${data.documents?.profile\_photo?.uploaded ? '✅ Uploaded' : '❌ Pending'}</li>

`;

// Update Preview

document.getElementById('previewName').textContent = data.name || 'N/A';

document.getElementById('previewTitle').textContent = data.cardTitle || 'N/A';

document.getElementById('previewEntityId').textContent = request.target\_entity\_id;

document.getElementById('previewPhone').textContent = data.contactPhone || 'N/A';

document.getElementById('previewBarcode').textContent = data.barcodeUid || 'N/A';

// Set button states based on workflow status

document.getElementById('finalizeGenerateBtn').style.display = request.status === 'pending\_data' ? 'block' : 'none';

document.getElementById('principalApproveBtn').disabled = request.status !== 'pending\_approval';

document.getElementById('rejectBtn').disabled = request.status === 'approved' || request.status === 'rejected';

// Store current request ID on buttons

document.getElementById('principalApproveBtn').setAttribute('data-request-id', requestId);

document.getElementById('rejectBtn').setAttribute('data-request-id', requestId);

document.getElementById('finalizeGenerateBtn').setAttribute('data-request-id', requestId);

}

});

// Handle Finalize Generation Button (Admin Action)

document.getElementById('finalizeGenerateBtn').addEventListener('click', async (e) => {

const requestId = e.target.getAttribute('data-request-id');

const btn = e.target;

btn.disabled = true;

btn.textContent = 'Submitting...';

try {

const data = await apiFetch(`/requests/${requestId}/generate`, 'POST', {});

showNotification(data.message, 'success');

fetchRequests();

bootstrap.Modal.getInstance(document.getElementById('approvalModal')).hide();

} catch {

btn.disabled = false;

btn.textContent = 'Submit for Approval Gate';

}

});

// Handle Principal Approval Button

document.getElementById('principalApproveBtn').addEventListener('click', async (e) => {

const requestId = e.target.getAttribute('data-request-id');

const btn = e.target;

btn.disabled = true;

btn.textContent = 'Approving...';

try {

const data = await apiFetch(`/requests/${requestId}/approve`, 'POST', {});

showNotification(data.message, 'success');

fetchRequests();

bootstrap.Modal.getInstance(document.getElementById('approvalModal')).hide();

} catch {

btn.disabled = false;

btn.textContent = 'Principal Approve (Unlocks Print)';

}

});

// Handle Print Button Click (Gated Access)

document.getElementById('requestsTableBody').addEventListener('click', async (e) => {

if (e.target.classList.contains('print-btn')) {

const requestId = e.target.getAttribute('data-request-id');

const btn = e.target;

btn.disabled = true;

btn.textContent = 'Fetching...';

try {

const data = await apiFetch(`/requests/${requestId}/print-data`);

document.getElementById('printDataContainer').textContent = JSON.stringify(data, null, 2);

const printModal = new bootstrap.Modal(document.getElementById('printModal'));

printModal.show();

showNotification('Print data successfully retrieved (Status: APPROVED).', 'success');

} catch (error) {

// API handles 403 (Not Approved) response

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-printer"></i> Print';

}

}

});

// --- 3. CONFIGURATION LOGIC ---

async function fetchConfig() {

try {

const data = await apiFetch('/config');

const settings = data.settings;

document.getElementById('templateSelector').value = settings.template || 'modern\_student\_v1';

document.getElementById('barcodeStyle').value = settings.barcode\_style || 'QR';

document.getElementById('expirationPolicy').value = settings.expiration\_policy || '12 months';

document.getElementById('requiredFields').value = settings.required\_fields || 'name, class, dob, emergency\_contact';

} catch (error) {

// Assuming defaults if API fails or 403 (Auth denied)

showNotification('Configuration loaded with default fallback values (or access denied).', 'warning');

}

}

document.getElementById('configForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const btn = document.getElementById('saveConfigBtn');

const payload = {

template: form.templateSelector.value,

barcodeStyle: form.barcodeStyle.value,

expirationPolicy: form.expirationPolicy.value,

requiredFields: form.requiredFields.value.split(',').map(s => s.trim()).filter(s => s.length > 0),

complianceSettings: { photoFormat: 'jpeg', resolution: '300dpi' } // Mock compliance data

};

btn.disabled = true;

btn.innerHTML = '<span class="spinner-border spinner-border-sm"></span> Saving...';

try {

const data = await apiFetch('/config', 'PUT', payload);

showNotification(data.message + ` New template: ${data.newTemplate}`, 'success');

} catch (error) {

// Handled by apiFetch

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-save"></i> Save Configuration';

}

});

// --- Initialization ---

document.addEventListener('DOMContentLoaded', () => {

fetchRequests();

fetchConfig();

document.getElementById('refreshRequestsBtn').addEventListener('click', fetchRequests);

});

</script>

</body>

</html>

14. client/dist/modules/invigilation\_Duty.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Invigilation Duty Configuration & Management</title>

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">

<style>

body {

background-color: #f8f9fa;

}

.container {

max-width: 900px;

margin-top: 30px;

}

.card-header {

background-color: #007bff;

color: white;

font-weight: bold;

}

.section-card {

margin-bottom: 25px;

box-shadow: 0 4px 8px rgba(0,0,0,0.05);

}

.alert-fixed {

position: fixed;

top: 20px;

right: 20px;

z-index: 1050;

min-width: 300px;

}

</style>

</head>

<body>

<div class="container">

<h1 class="text-center mb-4 text-primary">🏫 Invigilation Management Dashboard</h1>

<div id="notification-area" class="alert-fixed" style="display: none;"></div>

<ul class="nav nav-tabs" id="myTab" role="tablist">

<li class="nav-item" role="presentation">

<button class="nav-link active" id="config-tab" data-bs-toggle="tab" data-bs-target="#config" type="button" role="tab">⚙️ Configuration</button>

</li>

<li class="nav-item" role="presentation">

<button class="nav-link" id="allocation-tab" data-bs-toggle="tab" data-bs-target="#allocation" type="button" role="tab">🔄 Auto Allocation</button>

</li>

<li class="nav-item" role="presentation">

<button class="nav-link" id="workflow-tab" data-bs-toggle="tab" data-bs-target="#workflow" type="button" role="tab">✅ Workflow & Export</button>

</li>

</ul>

<div class="tab-content pt-3" id="myTabContent">

<div class="tab-pane fade show active" id="config" role="tabpanel">

<div class="card section-card">

<div class="card-header">System Configuration</div>

<div class="card-body">

<form id="config-form">

<div class="mb-3">

<label for="maxDuties" class="form-label">Maximum Duties Per Cycle</label>

<input type="number" class="form-control" id="maxDuties" name="maxDuties" required min="1" max="50">

<div class="form-text">Sets the soft limit for workload balancing (Total Duties Assigned - TDA).</div>

</div>

<div class="mb-3">

<label for="compensationRate" class="form-label">Compensation Rate (Per Duty)</label>

<div class="input-group">

<span class="input-group-text">₹</span>

<input type="number" class="form-control" id="compensationRate" name="compensationRate" required min="0" step="10">

</div>

<div class="form-text">Financial compensation/honorarium for each invigilation duty.</div>

</div>

<div class="mb-3">

<label for="exemptionPolicy" class="form-label">Exemption Policy</label>

<select class="form-select" id="exemptionPolicy" name="exemptionPolicy">

<option value="none">No Exemptions</option>

<option value="subject\_match\_once">Exempt Subject Teacher (e.g., Maths during Maths exam)</option>

<option value="permanent">Permanent Exemption List</option>

</select>

<div class="form-text">Defines rules for teachers who cannot be assigned specific duties.</div>

</div>

<div class="form-check form-switch mb-4">

<input class="form-check-input" type="checkbox" role="switch" id="autoAllocation" name="autoAllocation">

<label class="form-check-label" for="autoAllocation">Enable Automatic Duty Allocation</label>

</div>

<button type="submit" class="btn btn-primary" id="saveConfigBtn">Save Configuration</button>

</form>

</div>

</div>

</div>

<div class="tab-pane fade" id="allocation" role="tabpanel">

<div class="card section-card">

<div class="card-header">Duty Auto Allocation</div>

<div class="card-body">

<p class="alert alert-info">This process runs the core algorithm to generate a new invigilation chart based on current workload (TDA) and configurations.</p>

<form id="auto-allocate-form">

<div class="mb-3">

<label for="examDate" class="form-label">Exam Date</label>

<input type="date" class="form-control" id="examDate" name="examDate" required>

<div class="form-text">The date for which duties need to be assigned.</div>

</div>

<div class="mb-3">

<label for="roomList" class="form-label">Rooms Requiring Invigilators (Comma Separated)</label>

<textarea class="form-control" id="roomList" name="roomList" rows="3" placeholder="e.g., Room 101, Room 102, Auditorium, Lab 3" required></textarea>

<div class="form-text">List of rooms that need an invigilator for the specified date.</div>

</div>

<button type="submit" class="btn btn-success" id="runAllocationBtn">Run Auto Allocation</button>

</form>

<h5 class="mt-4">Allocation Results</h5>

<div id="allocation-results" class="mt-2">

</div>

</div>

</div>

</div>

<div class="tab-pane fade" id="workflow" role="tabpanel">

<div class="card section-card">

<div class="card-header">Approval Workflow & Export</div>

<div class="card-body">

<div class="d-flex justify-content-between align-items-center mb-3">

<h5>Current Chart Status: <span id="current-status" class="badge bg-secondary">Loading...</span></h5>

</div>

<div class="mb-4 d-grid gap-2 d-md-block">

<button type="button" class="btn btn-warning me-2" id="submitApprovalBtn">1. Submit to Principal (Admin Action)</button>

<button type="button" class="btn btn-success me-2" id="approveChartBtn">2. Approve Chart (Principal Action)</button>

<button type="button" class="btn btn-info" id="exportChartBtn" disabled>3. Export Printable Chart</button>

</div>

<hr>

<h5>Printable Chart Data Preview</h5>

<div id="duty-chart-preview" class="table-responsive mt-3">

<p class="text-muted">Export data will appear here upon successful retrieval.</p>

</div>

</div>

</div>

</div>

</div>

</div>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>

<script>

const API\_BASE = ''; // Base path for the API, leave empty if running on same host/port

/\*\*

\* Helper function to show a temporary notification.

\*/

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

const area = document.getElementById('notification-area');

area.innerHTML = `

<div class="alert alert-${type} alert-dismissible fade show" role="alert">

${message}

<button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>

</div>

`;

area.style.display = 'block';

setTimeout(() => {

area.style.display = 'none';

}, 5000);

}

// --- Configuration Logic ---

async function fetchConfig() {

try {

const response = await fetch(`${API\_BASE}/config`);

const data = await response.json();

if (data.success) {

const settings = data.settings;

document.getElementById('maxDuties').value = settings.maxDutiesPerWeek || 8;

document.getElementById('compensationRate').value = settings.compensationRate || 200;

document.getElementById('exemptionPolicy').value = settings.exemptionPolicy || 'subject\_match\_once';

document.getElementById('autoAllocation').checked = settings.autoAllocation !== false;

// Also check and update the workflow status

document.getElementById('current-status').textContent = settings.approval\_status || 'Draft';

updateWorkflowButtons(settings.approval\_status || 'Draft');

} else {

showNotification(data.error || 'Failed to fetch configuration.', 'danger');

}

} catch (error) {

console.error('Fetch Config Error:', error);

// On real error, use the fallback values as a UI default

}

}

document.getElementById('config-form').addEventListener('submit', async (e) => {

e.preventDefault();

const formData = {

autoAllocation: document.getElementById('autoAllocation').checked,

maxDuties: parseInt(document.getElementById('maxDuties').value, 10),

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

compensationRate: parseInt(document.getElementById('compensationRate').value, 10),

};

document.getElementById('saveConfigBtn').disabled = true;

try {

const response = await fetch(`${API\_BASE}/config`, {

method: 'PUT',

headers: { 'Content-Type': 'application/json' },

body: JSON.stringify(formData)

});

const data = await response.json();

if (data.success) {

showNotification(data.message);

} else {

showNotification(data.error || 'Failed to update configuration.', 'danger');

}

} catch (error) {

showNotification('Network error during configuration update.', 'danger');

} finally {

document.getElementById('saveConfigBtn').disabled = false;

}

});

// --- Auto Allocation Logic ---

document.getElementById('auto-allocate-form').addEventListener('submit', async (e) => {

e.preventDefault();

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

const roomListString = document.getElementById('roomList').value;

const roomList = roomListString.split(',').map(r => r.trim()).filter(r => r.length > 0);

const payload = { examDate, roomList };

const btn = document.getElementById('runAllocationBtn');

btn.disabled = true;

btn.textContent = 'Running Allocation...';

document.getElementById('allocation-results').innerHTML = '<div class="spinner-border text-primary" role="status"></div>';

try {

const response = await fetch(`${API\_BASE}/auto-allocate`, {

method: 'POST',

headers: { 'Content-Type': 'application/json' },

body: JSON.stringify(payload)

});

const data = await response.json();

if (data.success) {

showNotification(data.message, 'success');

displayAllocationResults(data.allocations);

// Refresh status after allocation

fetchConfig();

} else {

showNotification(data.error || 'Allocation failed.', 'danger');

document.getElementById('allocation-results').innerHTML = `<p class="text-danger">${data.error || 'An unknown error occurred during allocation.'}</p>`;

}

} catch (error) {

showNotification('Network error during allocation.', 'danger');

document.getElementById('allocation-results').innerHTML = `<p class="text-danger">Failed to connect to the server for allocation.</p>`;

} finally {

btn.disabled = false;

btn.textContent = 'Run Auto Allocation';

}

});

function displayAllocationResults(allocations) {

const resultsDiv = document.getElementById('allocation-results');

if (allocations.length === 0) {

resultsDiv.innerHTML = '<p class="text-warning">No duties were assigned (perhaps not enough teachers or rooms were listed).</p>';

return;

}

let html = `<p class="text-success">Total ${allocations.length} duties assigned:</p><ul class="list-group">`;

allocations.forEach(a => {

html += `<li class="list-group-item"><strong>${a.teacherName}</strong> assigned to <strong>${a.room}</strong> (Duty ID: ${a.dutyId})</li>`;

});

html += '</ul>';

resultsDiv.innerHTML = html;

}

// --- Workflow Logic ---

/\*\*

\* Updates button states based on the current chart status.

\*/

function updateWorkflowButtons(status) {

const submitBtn = document.getElementById('submitApprovalBtn');

const approveBtn = document.getElementById('approveChartBtn');

const exportBtn = document.getElementById('exportChartBtn');

// Reset all buttons

submitBtn.disabled = false;

approveBtn.disabled = false;

exportBtn.disabled = true;

if (status === 'Pending') {

submitBtn.disabled = true; // Already submitted

document.getElementById('current-status').className = 'badge bg-warning';

} else if (status === 'Approved') {

submitBtn.disabled = true;

approveBtn.disabled = true;

exportBtn.disabled = false; // Unlock export

document.getElementById('current-status').className = 'badge bg-success';

} else if (status === 'Draft') {

approveBtn.disabled = true; // Cannot approve a draft

document.getElementById('current-status').className = 'badge bg-secondary';

}

}

// Submit for Approval Handler

document.getElementById('submitApprovalBtn').addEventListener('click', async () => {

await handleWorkflowAction('/submit-for-approval', 'Submitting...', 'submitApprovalBtn');

});

// Approve Chart Handler

document.getElementById('approveChartBtn').addEventListener('click', async () => {

await handleWorkflowAction('/approve-chart', 'Approving...', 'approveChartBtn');

});

// Export Chart Handler

document.getElementById('exportChartBtn').addEventListener('click', async () => {

await handleWorkflowAction('/export/duty-chart', 'Generating Export...', 'exportChartBtn', 'GET');

});

/\*\*

\* Generic handler for workflow POST actions.

\*/

async function handleWorkflowAction(endpoint, loadingText, buttonId, method = 'POST') {

const btn = document.getElementById(buttonId);

btn.disabled = true;

const originalText = btn.textContent;

btn.textContent = loadingText;

try {

const response = await fetch(`${API\_BASE}${endpoint}`, {

method: method,

headers: { 'Content-Type': 'application/json' },

// Body is only needed for POST/PUT if data is sent

...(method !== 'GET' ? { body: JSON.stringify({}) } : {})

});

const data = await response.json();

if (data.success) {

showNotification(data.message, 'success');

// Refresh status and buttons

fetchConfig();

if (endpoint === '/export/duty-chart') {

displayDutyChart(data.chartData, data.dates);

}

} else {

showNotification(data.error || 'Action failed.', 'danger');

}

} catch (error) {

showNotification('Network error during workflow action.', 'danger');

} finally {

btn.disabled = false;

btn.textContent = originalText;

}

}

function displayDutyChart(chartData, dates) {

const previewDiv = document.getElementById('duty-chart-preview');

if (chartData.length === 0) {

previewDiv.innerHTML = '<p class="text-warning">No invigilation duties found for export.</p>';

return;

}

// Create table headers

const headers = Object.keys(chartData[0]);

let tableHtml = '<table class="table table-sm table-bordered table-hover">';

tableHtml += '<thead class="table-light"><tr>';

headers.forEach(h => {

// Shorten date column names for better viewing

const displayHeader = dates.includes(h) ? h.substring(5) : h;

tableHtml += `<th>${displayHeader}</th>`;

});

tableHtml += '</tr></thead><tbody>';

// Create table rows

chartData.forEach(row => {

tableHtml += '<tr>';

headers.forEach(h => {

tableHtml += `<td>${row[h]}</td>`;

});

tableHtml += '</tr>';

});

tableHtml += '</tbody></table>';

previewDiv.innerHTML = tableHtml;

}

// --- Initialization ---

document.addEventListener('DOMContentLoaded', fetchConfig);

</script>

</body>

</html>

15. client/dist/modules/Pdf\_tool.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>PDF & Document Tools - SmartGenEduX</title>

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

<link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css" rel="stylesheet">

<style>

.drop-zone {

transition: border-color 0.2s, background-color 0.2s;

}

.drop-zone:hover, .drop-zone.drag-over {

border-color: #3b82f6;

background-color: #eff6ff;

}

.tool-card {

transition: transform 0.2s, box-shadow 0.2s;

border-radius: 0.75rem;

}

.tool-card:hover {

transform: translateY(-4px);

box-shadow: 0 10px 15px rgba(0, 0, 0, 0.1);

}

.file-upload-hidden {

opacity: 0;

width: 0.1px;

height: 0.1px;

position: absolute;

overflow: hidden;

z-index: -1;

}

</style>

</head>

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

<div class="container mx-auto px-4 py-8">

<!-- Notification Area -->

<div id="notification-area" class="fixed top-4 right-4 z-50"></div>

<!-- Header -->

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

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

<div class="flex items-center">

<!-- Placeholder for image, using icon for compliance -->

<div class="w-12 h-12 mr-4 bg-gray-200 rounded-lg flex items-center justify-center">

<i class="fas fa-file-invoice text-2xl text-gray-700"></i>

</div>

<div>

<h1 class="text-2xl font-bold text-gray-800">PDF & Document Tools</h1>

<p class="text-gray-600">Convert, compile, and manage documents with advanced processing</p>

</div>

</div>

<button onclick="window.close()" class="bg-gray-500 text-white px-4 py-2 rounded hover:bg-gray-600 transition duration-150">

<i class="fas fa-arrow-left mr-2"></i>Back to Dashboard

</button>

</div>

</div>

<!-- Tool Categories -->

<div class="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-3 gap-6 mb-6">

<!-- PDF to Word -->

<div class="bg-white tool-card shadow-md p-6">

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

<div class="w-16 h-16 bg-red-100 rounded-full flex items-center justify-center mx-auto mb-3">

<i class="fas fa-file-pdf text-2xl text-red-600"></i>

</div>

<h3 class="text-lg font-bold">PDF to Word</h3>

<p class="text-gray-600 text-sm">Convert PDF files to editable Word documents</p>

</div>

<div id="dropZonePdfToWord" class="drop-zone border-2 border-dashed border-gray-300 rounded-lg p-4 text-center cursor-pointer">

<input type="file" id="pdfToWord" accept=".pdf" class="file-upload-hidden" onchange="handleFileSelection('pdfToWord', this)">

<label for="pdfToWord">

<i class="fas fa-cloud-upload-alt text-3xl text-gray-400 mb-2"></i>

<p class="text-gray-600"><span id="fileNamePdfToWord">Drop PDF file or click to upload</span></p>

</label>

</div>

<button onclick="convertDocument('pdfToWord', 'Word')" class="w-full mt-4 bg-red-600 text-white py-2 rounded hover:bg-red-700 transition duration-150" id="btnPdfToWord">

<i class="fas fa-exchange-alt mr-2"></i>Convert to Word

</button>

</div>

<!-- Word to PDF -->

<div class="bg-white tool-card shadow-md p-6">

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

<div class="w-16 h-16 bg-blue-100 rounded-full flex items-center justify-center mx-auto mb-3">

<i class="fas fa-file-word text-2xl text-blue-600"></i>

</div>

<h3 class="text-lg font-bold">Word to PDF</h3>

<p class="text-gray-600 text-sm">Convert Word documents to PDF format</p>

</div>

<div id="dropZoneWordToPdf" class="drop-zone border-2 border-dashed border-gray-300 rounded-lg p-4 text-center cursor-pointer">

<input type="file" id="wordToPdf" accept=".doc,.docx" class="file-upload-hidden" onchange="handleFileSelection('wordToPdf', this)">

<label for="wordToPdf">

<i class="fas fa-cloud-upload-alt text-3xl text-gray-400 mb-2"></i>

<p class="text-gray-600"><span id="fileNameWordToPdf">Drop Word file or click to upload</span></p>

</label>

</div>

<button onclick="convertDocument('wordToPdf', 'PDF')" class="w-full mt-4 bg-blue-600 text-white py-2 rounded hover:bg-blue-700 transition duration-150" id="btnWordToPdf">

<i class="fas fa-exchange-alt mr-2"></i>Convert to PDF

</button>

</div>

<!-- Document Compiler (Merge) -->

<div class="bg-white tool-card shadow-md p-6">

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

<div class="w-16 h-16 bg-green-100 rounded-full flex items-center justify-center mx-auto mb-3">

<i class="fas fa-layer-group text-2xl text-green-600"></i>

</div>

<h3 class="text-lg font-bold">Document Compiler</h3>

<p class="text-gray-600 text-sm">Merge multiple documents into one</p>

</div>

<div id="dropZoneMultipleFiles" class="drop-zone border-2 border-dashed border-gray-300 rounded-lg p-4 text-center cursor-pointer">

<input type="file" id="multipleFiles" multiple accept=".pdf,.doc,.docx" class="file-upload-hidden" onchange="handleFileSelection('multipleFiles', this)">

<label for="multipleFiles">

<i class="fas fa-cloud-upload-alt text-3xl text-gray-400 mb-2"></i>

<p class="text-gray-600"><span id="fileNameMultipleFiles">Select multiple files to compile</span></p>

</label>

</div>

<button onclick="compileDocuments()" class="w-full mt-4 bg-green-600 text-white py-2 rounded hover:bg-green-700 transition duration-150" id="btnCompileDocuments">

<i class="fas fa-compress mr-2"></i>Compile Documents

</button>

</div>

</div>

<!-- Bulk Operations -->

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

<h2 class="text-xl font-bold mb-4">Bulk Operations</h2>

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

<div>

<h3 class="text-lg font-semibold mb-3">Batch Conversion</h3>

<div id="dropZoneBatchFiles" class="drop-zone border-2 border-dashed border-gray-300 rounded-lg p-6 text-center cursor-pointer">

<input type="file" id="batchFiles" multiple accept=".pdf,.doc,.docx" class="file-upload-hidden" onchange="handleFileSelection('batchFiles', this)">

<label for="batchFiles">

<i class="fas fa-files-medical text-4xl text-gray-400 mb-3"></i>

<p class="text-gray-600 mb-2"><span id="fileNameBatchFiles">Upload multiple files for batch processing</span></p>

<p class="text-sm text-gray-500">Supports PDF, DOC, DOCX formats</p>

</label>

</div>

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

<button onclick="batchConvert('PDF')" class="flex-1 bg-purple-600 text-white py-2 rounded hover:bg-purple-700 transition duration-150" id="btnBatchToPdf">

Convert All to PDF

</button>

<button onclick="batchConvert('Word')" class="flex-1 bg-orange-600 text-white py-2 rounded hover:bg-orange-700 transition duration-150" id="btnBatchToWord">

Convert All to Word

</button>

</div>

</div>

<div>

<h3 class="text-lg font-semibold mb-3">Document Templates</h3>

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

<button onclick="generateTemplate('Report')" class="w-full bg-gray-100 hover:bg-gray-200 p-3 rounded text-left transition duration-150">

<i class="fas fa-file-alt mr-2 text-blue-600"></i>

<span class="font-medium">Student Report Template</span>

</button>

<button onclick="generateTemplate('Certificate')" class="w-full bg-gray-100 hover:bg-gray-200 p-3 rounded text-left transition duration-150">

<i class="fas fa-certificate mr-2 text-yellow-600"></i>

<span class="font-medium">Certificate Template</span>

</button>

<button onclick="generateTemplate('Letter')" class="w-full bg-gray-100 hover:bg-gray-200 p-3 rounded text-left transition duration-150">

<i class="fas fa-envelope mr-2 text-green-600"></i>

<span class="font-medium">Official Letter Template</span>

</button>

<button onclick="generateTemplate('Invoice')" class="w-full bg-gray-100 hover:bg-gray-200 p-3 rounded text-left transition duration-150">

<i class="fas fa-receipt mr-2 text-purple-600"></i>

<span class="font-medium">Fee Invoice Template</span>

</button>

</div>

</div>

</div>

</div>

<!-- Recent Conversions (History) -->

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

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

<h2 class="text-xl font-bold">Recent Operations History</h2>

<button onclick="clearHistory()" class="text-red-600 hover:text-red-800 text-sm transition duration-150">

<i class="fas fa-trash-alt mr-1"></i>Clear History

</button>

</div>

<div class="overflow-x-auto">

<table class="w-full text-sm">

<thead class="bg-gray-50">

<tr>

<th class="px-4 py-2 text-left">File Name</th>

<th class="px-4 py-2 text-left">Operation</th>

<th class="px-4 py-2 text-left">Status</th>

<th class="px-4 py-2 text-left">Date</th>

<th class="px-4 py-2 text-left">Actions</th>

</tr>

</thead>

<tbody id="historyTableBody">

<!-- History will be populated here -->

</tbody>

</table>

</div>

</div>

</div>

<script>

let fileReferences = {}; // Stores files selected for conversion/compilation

// Utility function to display messages

function showMessage(message, type) {

const colors = { 'success': 'bg-green-600', 'error': 'bg-red-600', 'info': 'bg-blue-600', 'warning': 'bg-yellow-600' };

const icons = { 'success': 'fas fa-check-circle', 'error': 'fas fa-exclamation-triangle', 'info': 'fas fa-info-circle', 'warning': 'fas fa-bell' };

const area = document.getElementById('notification-area');

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

messageDiv.className = `p-3 mb-2 flex items-center ${colors[type]} text-white rounded-lg shadow-lg transition-all duration-300 transform translate-x-0`;

messageDiv.innerHTML = `<i class="${icons[type]} mr-3"></i><span>${message}</span>`;

area.appendChild(messageDiv);

setTimeout(() => { messageDiv.remove(); }, 5000);

}

// --- Core File Handling ---

function handleFileSelection(id, input) {

fileReferences[id] = input.files;

const fileNameSpan = document.getElementById(`fileName${id.charAt(0).toUpperCase() + id.slice(1)}`);

if (input.files.length === 1) {

fileNameSpan.innerHTML = `<span class="text-blue-600 font-medium">${input.files[0].name}</span> selected. Click Convert.`;

showMessage(`File "${input.files[0].name}" selected.`, 'info');

} else if (input.files.length > 1) {

fileNameSpan.innerHTML = `<span class="text-blue-600 font-medium">${input.files.length} files</span> ready for batch.`;

showMessage(`${input.files.length} files selected.`, 'info');

}

}

// --- History Management ---

function addHistoryEntry(fileName, operation, status) {

const body = document.getElementById('historyTableBody');

const row = body.insertRow(0); // Insert at top

let statusClass = '';

let statusText = '';

let actionHtml = '';

if (status === 'Completed') {

statusClass = 'bg-green-100 text-green-800';

statusText = 'Completed';

actionHtml = `<button class="text-blue-600 hover:text-blue-800 mr-2" title="Download"><i class="fas fa-download"></i></button>`;

} else if (status === 'Processing') {

statusClass = 'bg-yellow-100 text-yellow-800';

statusText = 'Processing';

actionHtml = `<button class="text-gray-400" title="Processing" disabled><i class="fas fa-spinner fa-spin"></i></button>`;

}

row.className = 'border-t';

row.innerHTML = `

<td class="px-4 py-2">${fileName}</td>

<td class="px-4 py-2">${operation}</td>

<td class="px-4 py-2"><span class="px-2 py-1 rounded text-xs ${statusClass}">${statusText}</span></td>

<td class="px-4 py-2">${new Date().toLocaleString()}</td>

<td class="px-4 py-2">${actionHtml}

<button class="text-red-600 hover:text-red-800 ml-2 delete-history-btn" title="Delete"><i class="fas fa-trash-alt"></i></button>

</td>

`;

// Attach delete listener (Simulated)

row.querySelector('.delete-history-btn').addEventListener('click', (e) => {

if (confirm(`Are you sure you want to delete the history for ${fileName}?`)) {

e.target.closest('tr').remove();

showMessage('History entry deleted.', 'info');

}

});

}

function clearHistory() {

if (confirm('Are you sure you want to clear ALL conversion history?')) {

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

showMessage('Conversion history cleared.', 'success');

}

}

// --- Conversion Logic ---

async function convertDocument(inputId, targetFormat) {

const files = fileReferences[inputId];

if (!files || files.length === 0) {

showMessage(`Please select a file to convert to ${targetFormat}`, 'error');

return;

}

const fileName = files[0].name;

const btn = document.getElementById(`btn${inputId.charAt(0).toUpperCase() + inputId.slice(1)}`);

const originalText = btn.innerHTML;

btn.innerHTML = '<i class="fas fa-spinner fa-spin mr-2"></i>Converting...';

btn.disabled = true;

addHistoryEntry(fileName, `${fileName.split('.').pop().toUpperCase()} → ${targetFormat}`, 'Processing');

await new Promise(resolve => setTimeout(resolve, 3000));

btn.innerHTML = originalText;

btn.disabled = false;

showMessage(`Conversion of "${fileName}" completed successfully!`, 'success');

// Update history status (simulated update)

const body = document.getElementById('historyTableBody');

body.rows[0].querySelector('.px-2.py-1').className = 'px-2 py-1 rounded text-xs bg-green-100 text-green-800';

body.rows[0].querySelector('.px-2.py-1').textContent = 'Completed';

// Reset input and file reference

document.getElementById(inputId).value = null;

fileReferences[inputId] = null;

document.getElementById(`fileName${inputId.charAt(0).toUpperCase() + inputId.slice(1)}`).textContent = `Drop ${targetFormat === 'Word' ? 'PDF' : 'Word'} file or click to upload`;

}

async function compileDocuments() {

const files = fileReferences['multipleFiles'];

if (!files || files.length < 2) {

showMessage('Select at least two files to compile.', 'error');

return;

}

const btn = document.getElementById('btnCompileDocuments');

const originalText = btn.innerHTML;

const count = files.length;

btn.innerHTML = '<i class="fas fa-spinner fa-spin mr-2"></i>Compiling...';

btn.disabled = true;

addHistoryEntry(`Compiled\_Document\_${count}x.pdf`, `Merge ${count} Files`, 'Processing');

await new Promise(resolve => setTimeout(resolve, 4000));

btn.innerHTML = originalText;

btn.disabled = false;

showMessage(`Compilation of ${count} documents completed successfully!`, 'success');

// Simulate history update

const body = document.getElementById('historyTableBody');

body.rows[0].querySelector('.px-2.py-1').className = 'px-2 py-1 rounded text-xs bg-green-100 text-green-800';

body.rows[0].querySelector('.px-2.py-1').textContent = 'Completed';

document.getElementById('multipleFiles').value = null;

fileReferences['multipleFiles'] = null;

document.getElementById('fileNameMultipleFiles').textContent = `Select multiple files to compile`;

}

// --- Batch Conversion Logic ---

async function batchConvert(targetFormat) {

const files = fileReferences['batchFiles'];

if (!files || files.length === 0) {

showMessage('Please upload files for batch processing.', 'error');

return;

}

const btnId = targetFormat === 'PDF' ? 'btnBatchToPdf' : 'btnBatchToWord';

const btn = document.getElementById(btnId);

const originalText = btn.innerHTML;

const count = files.length;

btn.innerHTML = '<i class="fas fa-spinner fa-spin mr-2"></i>Processing...';

btn.disabled = true;

addHistoryEntry(`Batch\_Convert\_${count}x.zip`, `Batch → ${targetFormat}`, 'Processing');

await new Promise(resolve => setTimeout(resolve, 5000));

btn.innerHTML = originalText;

btn.disabled = false;

showMessage(`Batch conversion of ${count} files to ${targetFormat} completed!`, 'success');

// Simulate history update

const body = document.getElementById('historyTableBody');

body.rows[0].querySelector('.px-2.py-1').className = 'px-2 py-1 rounded text-xs bg-green-100 text-green-800';

body.rows[0].querySelector('.px-2.py-1').textContent = 'Completed';

document.getElementById('batchFiles').value = null;

fileReferences['batchFiles'] = null;

document.getElementById('fileNameBatchFiles').textContent = `Upload multiple files for batch processing`;

}

// --- Template Generation Logic ---

function generateTemplate(type) {

showMessage(`Generating ${type} Template...`, 'info');

setTimeout(() => {

showMessage(`${type} Template generated and downloaded!`, 'success');

addHistoryEntry(`${type}\_Template.docx`, 'Template Generation', 'Completed');

}, 2000);

}

// --- Initialize History with Mock Data ---

document.addEventListener('DOMContentLoaded', () => {

// Add initial mock entries for realism

addHistoryEntry('Batch\_Certificates.zip', 'Batch Compile', 'Processing');

addHistoryEntry('Fee\_Structure\_2025.docx', 'Word → PDF', 'Completed');

addHistoryEntry('Student\_Report\_Class5.pdf', 'PDF → Word', 'Completed');

// Setup Drag and Drop functionality

['dropZonePdfToWord', 'dropZoneWordToPdf', 'dropZoneMultipleFiles', 'dropZoneBatchFiles'].forEach(id => {

const dropZone = document.getElementById(id);

dropZone.addEventListener('dragover', (e) => {

e.preventDefault();

dropZone.classList.add('drag-over');

});

dropZone.addEventListener('dragleave', (e) => {

e.preventDefault();

dropZone.classList.remove('drag-over');

});

dropZone.addEventListener('drop', (e) => {

e.preventDefault();

dropZone.classList.remove('drag-over');

const fileInput = dropZone.querySelector('input[type="file"]');

if (e.dataTransfer.files.length) {

fileInput.files = e.dataTransfer.files;

handleFileSelection(fileInput.id, fileInput);

}

});

});

});

</script>

</body>

</html>

16. client/dist/modules/Question\_Extractor.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>🧠 AI Question Extractor & Question Bank Builder</title>

<!-- Bootstrap 5 CDN -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css">

<style>

body { background-color: #e8f5e9; font-family: 'Inter', sans-serif; }

.container-fluid { max-width: 1200px; margin-top: 20px; }

.card-header { background-color: #388e3c; /\* Dark Green \*/ color: white; font-weight: bold; border-radius: 0.75rem 0.75rem 0 0; }

.section-card { margin-bottom: 25px; box-shadow: 0 4px 15px rgba(0, 0, 0, 0.1); border-radius: 0.75rem; }

.alert-fixed { position: fixed; top: 20px; right: 20px; z-index: 1050; min-width: 300px; border-radius: 0.5rem; }

.nav-tabs .nav-link.active { background-color: #4caf50; color: white; border-color: #4caf50; }

.question-item { border-left: 5px solid #ccc; padding-left: 10px; margin-bottom: 10px; }

.question-item.verified { border-left-color: #28a745; }

.question-item.needs-review { border-left-color: #ffc107; }

.status-badge { font-size: 0.75rem; vertical-align: middle; }

</style>

</head>

<body>

<div class="container-fluid">

<h1 class="text-center mb-4 text-success"><i class="bi bi-robot"></i> AI Question Extractor Dashboard</h1>

<!-- Notification Area -->

<div id="notification-area" class="alert-fixed" style="display: none;"></div>

<!-- Tab Navigation -->

<ul class="nav nav-tabs" id="extractorTab" role="tablist">

<li class="nav-item"><button class="nav-link active" id="upload-tab" data-bs-toggle="tab" data-bs-target="#upload-section" type="button"><i class="bi bi-file-earmark-arrow-up"></i> Upload & Process</button></li>

<li class="nav-item"><button class="nav-link" id="verify-tab" data-bs-toggle="tab" data-bs-target="#verification-section" type="button"><i class="bi bi-check2-square"></i> Verification Workflow</button></li>

<li class="nav-item"><button class="nav-link" id="analytics-tab" data-bs-toggle="tab" data-bs-target="#analytics-section" type="button"><i class="bi bi-graph-up"></i> Analytics</button></li>

<li class="nav-item"><button class="nav-link" id="config-tab" data-bs-toggle="tab" data-bs-target="#config-section" type="button"><i class="bi bi-sliders"></i> AI Configuration</button></li>

</ul>

<!-- Tab Content -->

<div class="tab-content pt-3" id="extractorTabContent">

<!-- 1. Upload & Process Tab (POST /upload, GET /documents) -->

<div class="tab-pane fade show active" id="upload-section" role="tabpanel">

<div class="card section-card">

<div class="card-header">Upload Document for Extraction</div>

<div class="card-body">

<form id="uploadForm" enctype="multipart/form-data" class="row g-3">

<div class="col-md-6">

<label for="documentFile" class="form-label">Upload Question Paper (PDF/DOCX)</label>

<input type="file" class="form-control" id="documentFile" name="documentFile" accept=".pdf,.doc,.docx" required>

<div class="form-text">Max file size: <span id="maxFileSizeText">5 MB</span>. Only PDF, DOCX allowed.</div>

</div>

<div class="col-md-3">

<label for="subjectId" class="form-label">Subject ID</label>

<select class="form-select" id="subjectId" name="subjectId" required>

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

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

<option value="PHY">Physics</option>

<option value="CHE">Chemistry</option>

</select>

</div>

<div class="col-md-3">

<label for="classId" class="form-label">Class ID</label>

<select class="form-select" id="classId" name="classId" required>

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

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

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

</select>

</div>

<div class="col-md-6">

<label for="extractionMethod" class="form-label">Extraction Method Preference</label>

<select class="form-select" id="extractionMethod" name="extractionMethod">

<option value="enhanced\_ocr">Enhanced OCR (Default)</option>

<option value="basic\_ocr">Basic OCR (Fast)</option>

<option value="ai\_enhanced">AI Enhanced (Slow, High Accuracy)</option>

</select>

</div>

<input type="hidden" name="uploadedBy" value="11111111-1111-1111-1111-111111111111">

<div class="col-12 mt-4">

<button type="submit" class="btn btn-success btn-lg" id="uploadBtn"><i class="bi bi-cloud-arrow-up"></i> Upload & Queue for AI Extraction</button>

</div>

</form>

<hr class="my-4">

<h5>Extraction History</h5>

<div class="table-responsive">

<table class="table table-striped table-sm">

<thead>

<tr>

<th>Doc ID</th>

<th>File Name</th>

<th>Uploaded</th>

<th>Status</th>

<th>Questions</th>

<th>Action</th>

</tr>

</thead>

<tbody id="documentHistoryTable">

<tr><td colspan="6" class="text-center">Loading history...</td></tr>

</tbody>

</table>

</div>

</div>

</div>

</div>

<!-- 2. Verification Workflow Tab (GET /documents/:id/questions, POST /questions/:id/verify, POST /questions/bulk-verify) -->

<div class="tab-pane fade" id="verification-section" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-warning text-dark">Human-in-the-Loop Verification</div>

<div class="card-body">

<div class="input-group mb-3">

<input type="text" class="form-control" id="targetDocumentId" placeholder="Enter Document ID to review questions">

<button class="btn btn-primary" type="button" id="loadQuestionsBtn"><i class="bi bi-arrow-right-square"></i> Load Questions</button>

</div>

<div id="bulkControls" class="d-none alert alert-light p-3">

<p class="fw-bold">Bulk Action on <span id="bulkCount">0</span> selected questions:</p>

<textarea class="form-control mb-2" id="bulkRemarks" placeholder="Optional remarks for bulk action..."></textarea>

<button class="btn btn-sm btn-success bulk-verify-btn" data-verified="true"><i class="bi bi-check-all"></i> Bulk Verify Selected</button>

<button class="btn btn-sm btn-danger bulk-verify-btn" data-verified="false"><i class="bi bi-x-octagon"></i> Bulk Reject/Review</button>

</div>

<div id="questionsList" class="mt-4">

<p class="text-muted text-center">Load a document ID above to see extracted questions awaiting verification.</p>

</div>

</div>

</div>

</div>

<!-- 3. Analytics Tab (GET /statistics) -->

<div class="tab-pane fade" id="analytics-section" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-info">Extraction Analytics & Status</div>

<div class="card-body">

<div class="row text-center">

<div class="col-md-4 mb-3">

<div class="p-4 border rounded bg-light">

<h2 class="text-primary" id="totalExtracted">...</h2>

<p class="text-muted">Total Questions Extracted</p>

</div>

</div>

<div class="col-md-4 mb-3">

<div class="p-4 border rounded bg-light">

<h2 class="text-success" id="verifiedCount">...</h2>

<p class="text-muted">Verified & Ready for QPG</p>

</div>

</div>

<div class="col-md-4 mb-3">

<div class="p-4 border rounded bg-light">

<h2 class="text-warning" id="processingCount">...</h2>

<p class="text-muted">Currently Processing</p>

</div>

</div>

</div>

<button class="btn btn-sm btn-outline-info mt-3" id="refreshStatsBtn"><i class="bi bi-arrow-clockwise"></i> Refresh Statistics</button>

</div>

</div>

</div>

<!-- 4. Configuration Tab (GET/PUT /config) -->

<div class="tab-pane fade" id="config-section" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-danger">AI Model & Extraction Configuration (Manager Only)</div>

<div class="card-body">

<p class="alert alert-danger small"><i class="bi bi-lock-fill"></i> Only School Admin/Principal can manage these critical AI and compliance settings.</p>

<form id="configForm" class="row g-3">

<h5 class="text-danger">A. AI Model & Quality Control</h5>

<div class="col-md-6">

<label for="aiModel" class="form-label">Preferred AI Extraction Model</label>

<select class="form-select" id="aiModel" name="aiModelPreference" required>

<option value="enhanced\_ocr">Enhanced OCR (Best Balance)</option>

<option value="basic\_ocr">Basic OCR (Cost Effective)</option>

<option value="ai\_enhanced">AI Enhanced (Highest Accuracy/Cost)</option>

</select>

</div>

<div class="col-md-6">

<label for="confidenceThreshold" class="form-label">AI Confidence Threshold (%)</label>

<input type="number" class="form-control" id="confidenceThreshold" name="confidenceThreshold" min="50" max="100" required>

<div class="form-text">Questions below this score are automatically flagged for human review.</div>

</div>

<hr class="my-3">

<h5 class="text-danger">B. Compliance & Auto-Pilot</h5>

<div class="col-md-4">

<label for="maxFileSize" class="form-label">Max File Size (MB)</label>

<input type="number" class="form-control" id="maxFileSize" name="maxFileSizeMb" min="1" max="50" required>

</div>

<div class="col-md-4">

<label for="supportedLanguages" class="form-label">Supported Languages (Comma Separated)</label>

<input type="text" class="form-control" id="supportedLanguages" name="supportedLanguages" placeholder="English, Hindi, Tamil" required>

</div>

<div class="col-md-4 d-flex align-items-end">

<div class="form-check form-switch mb-2">

<input class="form-check-input" type="checkbox" id="autoVerificationEnabled" name="autoVerificationEnabled">

<label class="form-check-label" for="autoVerificationEnabled">Enable Auto-Verification</label>

<div class="form-text">Automatically verifies questions above the confidence threshold.</div>

</div>

</div>

<div class="col-12 mt-4">

<button type="submit" class="btn btn-danger btn-lg w-100" id="saveConfigBtn"><i class="bi bi-save"></i> Save AI & Extraction Configuration</button>

</div>

</form>

</div>

</div>

</div>

</div>

</div>

<!-- Modal for Single Question Verification -->

<div class="modal fade" id="verifyModal" tabindex="-1">

<div class="modal-dialog">

<div class="modal-content">

<div class="modal-header bg-primary text-white">

<h5 class="modal-title">Verify Question <span id="modalQuestionId"></span></h5>

<button type="button" class="btn-close" data-bs-dismiss="modal"></button>

</div>

<form id="singleVerifyForm">

<div class="modal-body">

<input type="hidden" id="currentQuestionId">

<div class="mb-3">

<label class="form-label">Extracted Content:</label>

<textarea class="form-control" id="modalQuestionContent" rows="5" readonly></textarea>

<div class="form-text mt-2">Page: <span id="modalPageNumber"></span> | Status: <span id="modalCurrentStatus" class="badge"></span></div>

</div>

<div class="row mb-3">

<div class="col-md-6">

<label for="modalMarks" class="form-label">Marks</label>

<input type="number" class="form-control" id="modalMarks" min="1" max="10" required>

</div>

<div class="col-md-6">

<label for="modalDifficulty" class="form-label">Difficulty</label>

<select class="form-select" id="modalDifficulty" required>

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

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

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

</select>

</div>

</div>

<div class="mb-3">

<label for="modalRemarks" class="form-label">Verification Remarks</label>

<textarea class="form-control" id="modalRemarks" rows="2"></textarea>

</div>

</div>

<div class="modal-footer">

<button type="button" class="btn btn-secondary" data-bs-dismiss="modal">Close</button>

<button type="submit" class="btn btn-success" data-verification="true" id="saveVerificationBtn"><i class="bi bi-check-circle"></i> Mark Verified</button>

<button type="button" class="btn btn-warning" data-verification="false" id="flagForReviewBtn"><i class="bi bi-flag"></i> Flag for Review</button>

</div>

</form>

</div>

</div>

</div>

<!-- Bootstrap Bundle with Popper -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>

<script>

const API\_BASE = '';

const MOCK\_JWT\_TOKEN = 'mock.jwt.token.for.auth';

let currentQuestions = []; // Stores questions for verification tab

let currentDocumentId = null;

/\*\* Helper function to show a temporary notification. \*/

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

const area = document.getElementById('notification-area');

area.innerHTML = `

<div class="alert alert-${type} alert-dismissible fade show" role="alert">

${message}

<button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>

</div>

`;

area.style.display = 'block';

setTimeout(() => { area.style.display = 'none'; }, 7000);

}

/\*\* Core Fetch Function with JWT Auth Header. \*/

async function apiFetch(endpoint, method = 'GET', body = null, isFormData = false) {

try {

const options = {

method: method,

headers: {

'Authorization': `Bearer ${MOCK\_JWT\_TOKEN}`

},

};

if (body && method !== 'GET') {

if (!isFormData) {

options.headers['Content-Type'] = 'application/json';

options.body = JSON.stringify(body);

} else {

options.body = body;

}

}

const response = await fetch(`${API\_BASE}${endpoint}`, options);

const data = await response.json();

if (!response.ok) {

throw new Error(data.error || `API error (${response.status})`);

}

return data;

} catch (error) {

console.error('API Fetch Error:', endpoint, error);

showNotification(error.message || 'Network error. Check console for details.', 'danger');

throw error;

}

}

// --- 1. UPLOAD LOGIC ---

async function fetchDocumentHistory() {

const tableBody = document.getElementById('documentHistoryTable');

tableBody.innerHTML = '<tr><td colspan="6" class="text-center"><div class="spinner-border spinner-border-sm"></div> Loading...</td></tr>';

try {

const data = await apiFetch('/documents');

tableBody.innerHTML = '';

if (data.documents.length === 0) {

tableBody.innerHTML = '<tr><td colspan="6" class="text-center">No documents have been uploaded yet.</td></tr>';

return;

}

data.documents.forEach(doc => {

const statusClass = {

'queued': 'bg-secondary',

'processing': 'bg-primary',

'completed': 'bg-success',

'failed': 'bg-danger'

}[doc.processing\_status] || 'bg-dark';

const row = tableBody.insertRow();

row.innerHTML = `

<td class="small">${doc.id.substring(0, 8)}...</td>

<td>${doc.file\_name}</td>

<td>${new Date(doc.uploaded\_at).toLocaleDateString()}</td>

<td><span class="badge ${statusClass}">${doc.processing\_status.toUpperCase()}</span></td>

<td>${doc.questions\_extracted || 0}</td>

<td>

<button class="btn btn-sm btn-outline-warning review-doc-btn" data-doc-id="${doc.id}" ${doc.processing\_status !== 'completed' ? 'disabled' : ''}>Review</button>

</td>

`;

});

} catch (error) {

tableBody.innerHTML = '<tr><td colspan="6" class="text-center text-danger">Failed to load history.</td></tr>';

}

}

document.getElementById('uploadForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const btn = document.getElementById('uploadBtn');

const fileInput = document.getElementById('documentFile');

if (!fileInput.files[0]) return;

const formData = new FormData();

formData.append('documentFile', fileInput.files[0]);

formData.append('subjectId', form.subjectId.value);

formData.append('classId', form.classId.value);

formData.append('uploadedBy', form.uploadedBy.value);

formData.append('extractionMethod', form.extractionMethod.value);

btn.disabled = true;

btn.innerHTML = '<span class="spinner-border spinner-border-sm"></span> Processing & Uploading...';

try {

const data = await apiFetch('/upload', 'POST', formData, true);

showNotification(data.message + ` Document ID: ${data.documentId}`, 'success');

form.reset();

fetchDocumentHistory(); // Refresh history table

fetchStatistics(); // Update analytics

} catch (error) {

// Handled by apiFetch

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-cloud-arrow-up"></i> Upload & Queue for AI Extraction';

}

});

// Delegate click for Review button to automatically load verification tab

document.getElementById('documentHistoryTable').addEventListener('click', (e) => {

if (e.target.classList.contains('review-doc-btn')) {

const docId = e.target.getAttribute('data-doc-id');

document.getElementById('targetDocumentId').value = docId;

document.getElementById('loadQuestionsBtn').click();

// Switch tab to Verification

const verifyTab = new bootstrap.Tab(document.getElementById('verify-tab'));

verifyTab.show();

}

});

// --- 2. VERIFICATION WORKFLOW LOGIC ---

async function loadQuestions() {

const docId = document.getElementById('targetDocumentId').value.trim();

if (!docId) {

showNotification('Please enter a valid Document ID.', 'warning');

return;

}

const listContainer = document.getElementById('questionsList');

const btn = document.getElementById('loadQuestionsBtn');

listContainer.innerHTML = '<p class="text-center"><div class="spinner-border spinner-border-sm"></div> Loading questions...</p>';

btn.disabled = true;

currentDocumentId = docId;

try {

const data = await apiFetch(`/documents/${docId}/questions`);

currentQuestions = data.questions;

renderQuestionsList(data.questions);

} catch (error) {

listContainer.innerHTML = `<p class="text-danger text-center">Failed to load questions for Document ID: ${docId}. Check ID or status.</p>`;

document.getElementById('bulkControls').classList.add('d-none');

} finally {

btn.disabled = false;

}

}

function renderQuestionsList(questions) {

const listContainer = document.getElementById('questionsList');

if (questions.length === 0) {

listContainer.innerHTML = '<p class="text-center text-warning">No questions extracted for this document yet.</p>';

document.getElementById('bulkControls').classList.add('d-none');

return;

}

let html = '<ul class="list-unstyled">';

questions.forEach(q => {

const status = q.verified\_status || 'unverified';

const statusClass = status === 'verified' ? 'bg-success' : (status === 'needs\_review' ? 'bg-danger' : 'bg-secondary');

const itemClass = status === 'verified' ? 'verified' : (status === 'needs\_review' ? 'needs-review' : '');

html += `

<li class="question-item p-3 mb-3 border rounded ${itemClass}">

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

<div class="form-check">

<input class="form-check-input question-checkbox" type="checkbox" value="${q.id}" data-id="${q.id}">

<label class="form-check-label fw-bold">Q${q.page\_number}.${q.question\_number} </label>

<span class="badge ${statusClass} status-badge">${status.toUpperCase()}</span>

<span class="badge bg-info status-badge ms-2">Marks: ${q.marks || '?'}</span>

<span class="badge bg-dark status-badge">Diff: ${q.difficulty || '?' }</span>

</div>

<button class="btn btn-sm btn-outline-primary single-verify-btn" data-bs-toggle="modal" data-bs-target="#verifyModal" data-question-id="${q.id}">Edit / Verify</button>

</div>

<p class="mt-2 mb-0 small text-muted">${q.question\_text.substring(0, 150)}...</p>

</li>

`;

});

html += '</ul>';

listContainer.innerHTML = html;

document.getElementById('bulkControls').classList.remove('d-none');

}

document.getElementById('loadQuestionsBtn').addEventListener('click', loadQuestions);

document.getElementById('targetDocumentId').addEventListener('keypress', (e) => {

if (e.key === 'Enter') {

e.preventDefault();

loadQuestions();

}

});

// Delegation for Review/Edit button

document.getElementById('questionsList').addEventListener('click', (e) => {

if (e.target.classList.contains('single-verify-btn')) {

const qId = e.target.getAttribute('data-question-id');

const question = currentQuestions.find(q => q.id === qId);

if (!question) return;

document.getElementById('currentQuestionId').value = qId;

document.getElementById('modalQuestionId').textContent = qId.substring(0, 8);

document.getElementById('modalQuestionContent').value = question.question\_text;

document.getElementById('modalPageNumber').textContent = question.page\_number;

// Populate fields with existing data

document.getElementById('modalMarks').value = question.marks || 5;

document.getElementById('modalDifficulty').value = question.difficulty || 'medium';

document.getElementById('modalRemarks').value = question.verification\_notes || '';

const status = question.verified\_status || 'unverified';

const statusClass = status === 'verified' ? 'bg-success' : 'bg-warning text-dark';

document.getElementById('modalCurrentStatus').textContent = status.toUpperCase();

document.getElementById('modalCurrentStatus').className = `badge ${statusClass}`;

}

});

// Handle Single Verification Form Submission

document.getElementById('singleVerifyForm').addEventListener('click', async (e) => {

const target = e.target.closest('button');

if (!target || !(target.id === 'saveVerificationBtn' || target.id === 'flagForReviewBtn')) return;

e.preventDefault();

const qId = document.getElementById('currentQuestionId').value;

const isVerified = target.getAttribute('data-verification') === 'true';

const payload = {

isVerified: isVerified,

marks: parseInt(document.getElementById('modalMarks').value),

difficulty: document.getElementById('modalDifficulty').value,

remarks: document.getElementById('modalRemarks').value

};

const btnText = target.textContent;

target.disabled = true;

target.textContent = 'Saving...';

try {

const data = await apiFetch(`/questions/${qId}/verify`, 'POST', payload);

showNotification(data.message, 'success');

// Reload questions and close modal

loadQuestions();

bootstrap.Modal.getInstance(document.getElementById('verifyModal')).hide();

} catch (error) {

// Handled by apiFetch

} finally {

target.disabled = false;

target.textContent = btnText;

}

});

// Handle Bulk Verification Selection

document.getElementById('questionsList').addEventListener('change', () => {

const selected = document.querySelectorAll('.question-checkbox:checked');

document.getElementById('bulkCount').textContent = selected.length;

});

// Handle Bulk Verification Action

document.getElementById('bulkControls').addEventListener('click', async (e) => {

const target = e.target.closest('button.bulk-verify-btn');

if (!target) return;

const isVerified = target.getAttribute('data-verified') === 'true';

const selectedCheckboxes = document.querySelectorAll('.question-checkbox:checked');

const questionIds = Array.from(selectedCheckboxes).map(cb => cb.value);

if (questionIds.length === 0) {

showNotification('Select at least one question for bulk action.', 'warning');

return;

}

const payload = {

questionIds: questionIds,

isVerified: isVerified,

remarks: document.getElementById('bulkRemarks').value

};

const btnText = target.textContent;

target.disabled = true;

target.textContent = 'Processing...';

try {

const data = await apiFetch('/questions/bulk-verify', 'POST', payload);

showNotification(data.message, 'success');

loadQuestions();

} catch (error) {

// Handled by apiFetch

} finally {

target.disabled = false;

target.textContent = btnText;

}

});

// --- 3. ANALYTICS LOGIC ---

async function fetchStatistics() {

try {

const data = await apiFetch('/statistics');

const summary = data.summary;

document.getElementById('totalExtracted').textContent = summary.total\_extracted || 0;

document.getElementById('verifiedCount').textContent = summary.verified\_count || 0;

document.getElementById('processingCount').textContent = summary.processing\_count || 0;

} catch (error) {

// Display error message on the dashboard if needed

}

}

document.getElementById('refreshStatsBtn').addEventListener('click', fetchStatistics);

// --- 4. CONFIGURATION LOGIC ---

async function fetchConfig() {

try {

const data = await apiFetch('/config');

const settings = data.settings;

document.getElementById('aiModel').value = settings.ai\_model\_preference || 'enhanced\_ocr';

document.getElementById('confidenceThreshold').value = settings.confidence\_threshold || 90;

document.getElementById('autoVerificationEnabled').checked = settings.auto\_verification\_enabled || false;

document.getElementById('maxFileSize').value = settings.max\_file\_size\_mb || 5;

document.getElementById('supportedLanguages').value = (settings.supported\_languages || ['English']).join(', ');

// Update file size text in the upload tab

document.getElementById('maxFileSizeText').textContent = `${settings.max\_file\_size\_mb || 5} MB`;

} catch (error) {

showNotification('Configuration loaded with default fallback values (or access denied).', 'warning');

}

}

document.getElementById('configForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const btn = document.getElementById('saveConfigBtn');

const payload = {

ai\_model\_preference: form.aiModel.value,

confidence\_threshold: parseInt(form.confidenceThreshold.value),

auto\_verification\_enabled: form.autoVerificationEnabled.checked,

max\_file\_size\_mb: parseInt(form.maxFileSize.value),

supported\_languages: form.supportedLanguages.value.split(',').map(s => s.trim())

};

btn.disabled = true;

btn.innerHTML = '<span class="spinner-border spinner-border-sm"></span> Saving...';

try {

const data = await apiFetch('/config', 'PUT', payload);

showNotification(data.message, 'success');

fetchConfig(); // Refresh settings display

} catch (error) {

// Handled by apiFetch

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-save"></i> Save AI & Extraction Configuration';

}

});

// --- Initialization ---

document.addEventListener('DOMContentLoaded', () => {

fetchConfig();

fetchDocumentHistory();

fetchStatistics();

});

</script>

</body>

</html>

17. client/dist/modules/Question\_Paper.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>📝 Question Paper Generator (QPG) & Approval</title>

<!-- Bootstrap 5 CDN -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css">

<style>

body { background-color: #fafafa; font-family: 'Inter', sans-serif; }

.container-fluid { max-width: 1200px; margin-top: 20px; }

.card-header { background-color: #880e4f; /\* Dark Maroon \*/ color: white; font-weight: bold; border-radius: 0.75rem 0.75rem 0 0; }

.section-card { margin-bottom: 25px; box-shadow: 0 4px 15px rgba(0, 0, 0, 0.1); border-radius: 0.75rem; }

.alert-fixed { position: fixed; top: 20px; right: 20px; z-index: 1050; min-width: 300px; border-radius: 0.5rem; }

.nav-tabs .nav-link.active { background-color: #d81b60; color: white; border-color: #d81b60; }

.question-preview { border-left: 3px solid #ff4081; padding-left: 10px; margin-bottom: 5px; }

</style>

</head>

<body>

<div class="container-fluid">

<h1 class="text-center mb-4 text-pink-900" style="color: #d81b60;"><i class="bi bi-magic"></i> Automated Question Paper Generator</h1>

<!-- Notification Area -->

<div id="notification-area" class="alert-fixed" style="display: none;"></div>

<!-- Tab Navigation -->

<ul class="nav nav-tabs" id="qpgTab" role="tablist">

<li class="nav-item"><button class="nav-link active" id="generate-tab" data-bs-toggle="tab" data-bs-target="#generate-paper" type="button"><i class="bi bi-clipboard-plus"></i> Generate Paper</button></li>

<li class="nav-item"><button class="nav-link" id="bank-tab" data-bs-toggle="tab" data-bs-target="#question-bank" type="button"><i class="bi bi-database-fill"></i> Question Bank</button></li>

<li class="nav-item"><button class="nav-link" id="approval-tab" data-bs-toggle="tab" data-bs-target="#paper-approval" type="button"><i class="bi bi-shield-lock-fill"></i> HOD Approval</button></li>

<li class="nav-item"><button class="nav-link" id="config-tab" data-bs-toggle="tab" data-bs-target="#config-setup" type="button"><i class="bi bi-tools"></i> Configuration Hub</button></li>

</ul>

<!-- Tab Content -->

<div class="tab-content pt-3" id="qpgTabContent">

<!-- 1. Generate Paper Tab (POST /generate) -->

<div class="tab-pane fade show active" id="generate-paper" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-danger">Paper Generation Parameters</div>

<div class="card-body">

<p class="alert alert-warning small"><i class="bi bi-exclamation-triangle"></i> This tool automatically allocates questions from the verified bank based on Marks, Difficulty, and Chapter weightage set in the Configuration Hub.</p>

<form id="generatePaperForm" class="row g-3">

<h5 class="text-primary mb-3">Core Paper Details</h5>

<div class="col-md-6">

<label for="paperTitle" class="form-label">Exam Title</label>

<input type="text" class="form-control" id="paperTitle" placeholder="e.g., Mid-Term Examination" required>

</div>

<div class="col-md-3">

<label for="academicYear" class="form-label">Academic Year</label>

<input type="text" class="form-control" id="academicYear" placeholder="2024-25">

</div>

<div class="col-md-3">

<label for="classIdGenerate" class="form-label">Class</label>

<select class="form-select" id="classIdGenerate" required>

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

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

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

</select>

</div>

<div class="col-md-6">

<label for="subjectIdGenerate" class="form-label">Subject</label>

<select class="form-select" id="subjectIdGenerate" required>

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

<!-- Populated by JS -->

</select>

</div>

<div class="col-md-6">

<label for="chapters" class="form-label">Chapters Included (Select Multiple)</label>

<select class="form-select" id="chapters" multiple required size="3">

<option value="Ch1">Chapter 1: Basics</option>

<option value="Ch2">Chapter 2: Advanced</option>

<option value="Ch3">Chapter 3: Revision</option>

</select>

<div class="form-text">Algorithm pulls questions only from selected chapters.</div>

</div>

<div class="col-12 mt-4">

<label for="customInstructions" class="form-label">Custom Instructions / Header Text</label>

<textarea class="form-control" id="customInstructions" rows="2" placeholder="Time allowed: 3 Hours. All questions are compulsory."></textarea>

</div>

<div class="col-12 mt-4">

<button type="submit" class="btn btn-danger btn-lg w-100" id="generateBtn"><i class="bi bi-file-earmark-diff"></i> AUTO-GENERATE DRAFT PAPER</button>

</div>

</form>

</div>

</div>

</div>

<!-- 2. Question Bank Tab (GET /questions, POST /questions, POST /questions/:id/media) -->

<div class="tab-pane fade" id="question-bank" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-info">View & Contribute to Question Bank</div>

<div class="card-body">

<h5 class="text-info">Search Verified Questions</h5>

<form id="searchQuestionsForm" class="row g-3 mb-4">

<div class="col-md-4">

<select class="form-select" id="searchSubjectId" required>

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

<!-- Populated by JS -->

</select>

</div>

<div class="col-md-4">

<select class="form-select" id="searchClassId" required>

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

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

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

</select>

</div>

<div class="col-md-4">

<button type="submit" class="btn btn-info w-100"><i class="bi bi-search"></i> Load Questions</button>

</div>

</form>

<div id="bankList" class="my-3">

<p class="text-muted text-center">Use the search above to filter and load verified questions.</p>

</div>

<hr>

<h5 class="text-success">Add New Question (Manual Contribution)</h5>

<form id="newQuestionForm" class="row g-3">

<div class="col-md-4"><input type="text" class="form-control" id="newQChapterId" placeholder="Chapter ID" required></div>

<div class="col-md-4"><input type="number" class="form-control" id="newQMarks" placeholder="Marks (1-10)" min="1" max="10" required></div>

<div class="col-md-4">

<select class="form-select" id="newQDifficulty" required>

<option value="">Difficulty</option><option value="easy">Easy</option><option value="medium">Medium</option><option value="hard">Hard</option>

</select>

</div>

<div class="col-12">

<textarea class="form-control" id="newQText" rows="3" placeholder="Enter Question Text..." required></textarea>

</div>

<div class="col-md-6">

<select class="form-select" id="newQType" required>

<option value="">Type</option><option value="mcq">MCQ</option><option value="short\_answer">Short Answer</option><option value="long\_answer">Long Answer</option>

</select>

</div>

<div class="col-md-6">

<input type="text" class="form-control" id="newQAnswer" placeholder="Correct Answer (Optional)">

</div>

<div class="col-12">

<button type="submit" class="btn btn-success" id="addNewQBtn"><i class="bi bi-plus-circle"></i> Add Question to Bank</button>

<button type="button" class="btn btn-outline-primary ms-2" id="uploadMediaBtn" disabled><i class="bi bi-camera"></i> Upload Media</button>

</div>

</form>

</div>

</div>

</div>

<!-- 3. Approval Workflow Tab (GET /papers, POST /papers/:id/submit-for-approval, POST /papers/:id/approve, GET /papers/:id/export) -->

<div class="tab-pane fade" id="paper-approval" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-success">Generated Papers & Approval Status</div>

<div class="card-body">

<p class="alert alert-info small"><i class="bi bi-info-circle"></i> Only papers with \*\*APPROVED\*\* status can be printed/exported. Approving triggers a notification to the originating teacher.</p>

<div class="d-flex justify-content-end mb-3">

<button class="btn btn-sm btn-outline-success" id="refreshPapersBtn"><i class="bi bi-arrow-clockwise"></i> Refresh Papers List</button>

</div>

<div class="table-responsive">

<table class="table table-striped table-sm">

<thead>

<tr>

<th>Paper ID</th>

<th>Exam Name</th>

<th>Generated By</th>

<th>Status</th>

<th>Actions</th>

</tr>

</thead>

<tbody id="papersTableBody">

<tr><td colspan="5" class="text-center">Loading generated papers...</td></tr>

</tbody>

</table>

</div>

</div>

</div>

</div>

<!-- 4. Configuration Hub Tab (GET/PUT /config) -->

<div class="tab-pane fade" id="config-setup" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-primary">QPG Configuration & Layout Settings (Manager Only)</div>

<div class="card-body">

<p class="alert alert-danger small"><i class="bi bi-lock-fill"></i> Manager/Admin access required to modify core paper settings.</p>

<form id="configForm" class="row g-3">

<h5 class="text-primary">A. Core Generation Settings</h5>

<div class="col-md-4">

<label for="defaultTemplateId" class="form-label">Default Paper Template ID</label>

<input type="text" class="form-control" id="defaultTemplateId" required placeholder="e.g., TPL-MIDTERM-1">

<div class="form-text">Must match a valid entry in the `paper\_templates` table.</div>

</div>

<div class="col-md-4">

<label for="defaultTotalMarks" class="form-label">Default Total Marks</label>

<input type="number" class="form-control" id="defaultTotalMarks" min="10" required>

</div>

<div class="col-md-4">

<label for="defaultDuration" class="form-label">Default Duration (Mins)</label>

<input type="number" class="form-control" id="defaultDuration" min="30" max="240" required>

</div>

<div class="col-md-6">

<label for="passingMarksPercentage" class="form-label">Passing Marks (%)</label>

<input type="number" class="form-control" id="passingMarksPercentage" min="0" max="100" required>

</div>

<div class="col-md-6">

<label for="approvalRoles" class="form-label">Required Approval Roles (Comma separated)</label>

<input type="text" class="form-control" id="approvalRoles" placeholder="principal, hod\_maths" required>

</div>

<hr class="my-3">

<h5 class="text-primary">B. Layout & Compliance (Watermark)</h5>

<div class="col-md-4">

<label for="paperSize" class="form-label">Paper Size</label>

<select class="form-select" id="paperSize" required>

<option value="A4">A4</option>

<option value="Legal">Legal</option>

</select>

</div>

<div class="col-md-4">

<label for="orientation" class="form-label">Orientation</label>

<select class="form-select" id="orientation" required>

<option value="Portrait">Portrait</option>

<option value="Landscape">Landscape</option>

</select>

</div>

<div class="col-md-4 d-flex align-items-end">

<div class="form-check form-switch mb-2">

<input class="form-check-input" type="checkbox" id="watermarkEnabled" checked>

<label class="form-check-label" for="watermarkEnabled">Enable School Watermark</label>

</div>

</div>

<div class="col-12 mt-3">

<label class="form-label">Page Margins (mm)</label>

<div class="row g-2">

<div class="col-md-3"><input type="number" class="form-control" id="marginTop" placeholder="Top" min="0" required></div>

<div class="col-md-3"><input type="number" class="form-control" id="marginBottom" placeholder="Bottom" min="0" required></div>

<div class="col-md-3"><input type="number" class="form-control" id="marginLeft" placeholder="Left" min="0" required></div>

<div class="col-md-3"><input type="number" class="form-control" id="marginRight" placeholder="Right" min="0" required></div>

</div>

</div>

<!-- Template Structure Placeholder (JSON input for complex structures) -->

<div class="col-12 mt-3">

<label for="templateStructures" class="form-label">Template Structure JSON</label>

<textarea class="form-control" id="templateStructures" rows="3" required placeholder='[{"name": "Section A", "marks": 2, "count": 10}, ...] - Must be valid JSON'></textarea>

<div class="form-text">Defines question types and difficulty distribution rules.</div>

</div>

<div class="col-12 mt-4">

<button type="submit" class="btn btn-primary btn-lg w-100" id="saveConfigBtn"><i class="bi bi-save"></i> Save Configuration & Layout</button>

</div>

</form>

</div>

</div>

</div>

</div>

</div>

<!-- Print/Export Data Modal -->

<div class="modal fade" id="exportModal" tabindex="-1">

<div class="modal-dialog modal-lg">

<div class="modal-content">

<div class="modal-header bg-success text-white">

<h5 class="modal-title">Paper Export View (Gated)</h5>

<button type="button" class="btn-close" data-bs-dismiss="modal"></button>

</div>

<div class="modal-body">

<p class="alert alert-success small fw-bold">Status: <span id="exportStatus"></span> | Watermark: <span id="watermarkStatus"></span></p>

<h6>Layout & Metadata:</h6>

<ul id="exportMetadata" class="list-unstyled small"></ul>

<hr>

<h6>Paper Content (Simulated Printable View)</h6>

<div id="printableContent" class="p-3 border rounded">

<p class="text-muted text-center">The final, formatted question paper content would be rendered here, ready for printing, including header/footer and the school watermark.</p>

</div>

</div>

<div class="modal-footer">

<button type="button" class="btn btn-secondary" data-bs-dismiss="modal">Close</button>

<button type="button" class="btn btn-success"><i class="bi bi-download"></i> Download Final PDF</button>

</div>

</div>

</div>

</div>

<!-- Bootstrap Bundle with Popper -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>

<script>

const API\_BASE = '';

const MOCK\_JWT\_TOKEN = 'mock.jwt.token.for.auth';

let mockSubjects = [ { id: 'MATH', name: 'Mathematics' }, { id: 'PHY', name: 'Physics' }, { id: 'HIS', name: 'History' } ];

let currentPapers = [];

let currentBankQuestions = [];

/\*\* Helper function to show a temporary notification. \*/

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

const area = document.getElementById('notification-area');

area.innerHTML = `

<div class="alert alert-${type} alert-dismissible fade show" role="alert">

${message}

<button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>

</div>

`;

area.style.display = 'block';

setTimeout(() => { area.style.display = 'none'; }, 7000);

}

/\*\* Core Fetch Function with JWT Auth Header. \*/

async function apiFetch(endpoint, method = 'GET', body = null, isFormData = false) {

try {

const options = {

method: method,

headers: { 'Authorization': `Bearer ${MOCK\_JWT\_TOKEN}` },

};

if (body && method !== 'GET') {

if (!isFormData) {

options.headers['Content-Type'] = 'application/json';

options.body = JSON.stringify(body);

} else { options.body = body; }

}

const response = await fetch(`${API\_BASE}${endpoint}`, options);

const data = await response.json();

if (!response.ok) {

throw new Error(data.error || `API error (${response.status})`);

}

return data;

} catch (error) {

console.error('API Fetch Error:', endpoint, error);

showNotification(error.message || 'Network error. Check console for details.', 'danger');

throw error;

}

}

// --- Shared Utilities ---

function populateSubjectDropdowns() {

const dropdowns = ['subjectIdGenerate', 'searchSubjectId', 'newQSubjectId'];

dropdowns.forEach(id => {

const select = document.getElementById(id);

if (!select) return;

// Clear existing, keep "Select Subject" option

Array.from(select.options).slice(1).forEach(opt => select.removeChild(opt));

mockSubjects.forEach(sub => {

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

option.value = sub.id;

option.textContent = sub.name;

select.appendChild(option);

});

});

}

// NOTE: In a real system, you would fetch subjects via GET /subjects

// async function fetchSubjects() { try { const data = await apiFetch('/subjects'); mockSubjects = data.subjects; populateSubjectDropdowns(); } catch (e) {} }

// --- 1. GENERATE PAPER LOGIC (POST /generate) ---

document.getElementById('generatePaperForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const btn = document.getElementById('generateBtn');

// Collect selected chapter IDs

const chaptersSelect = document.getElementById('chapters');

const selectedChapters = Array.from(chaptersSelect.selectedOptions).map(option => option.value);

const payload = {

title: form.paperTitle.value,

subjectId: form.subjectIdGenerate.value,

classId: form.classIdGenerate.value,

academicYear: form.academicYear.value || undefined,

chapters: selectedChapters,

customInstructions: form.customInstructions.value || undefined

};

btn.disabled = true;

btn.innerHTML = '<span class="spinner-border spinner-border-sm"></span> Allocating Questions...';

try {

const data = await apiFetch('/generate', 'POST', payload);

showNotification(data.message + ` Paper ID: ${data.paperId}`, 'success');

form.reset();

fetchGeneratedPapers(); // Refresh the approval workflow list

} catch (error) {

// Handled by apiFetch

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-file-earmark-diff"></i> AUTO-GENERATE DRAFT PAPER';

}

});

// --- 2. QUESTION BANK LOGIC (GET /questions, POST /questions) ---

// Load questions for the bank view

document.getElementById('searchQuestionsForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

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

container.innerHTML = '<p class="text-center"><div class="spinner-border spinner-border-sm"></div> Loading questions...</p>';

const query = `subject=${form.searchSubjectId.value}&class=${form.searchClassId.value}`;

try {

const data = await apiFetch(`/questions?${query}`);

currentBankQuestions = data.questions;

if (data.questions.length === 0) {

container.innerHTML = '<p class="text-center text-warning">No verified questions found for this criteria.</p>';

return;

}

let html = '<ul class="list-unstyled">';

data.questions.forEach((q, index) => {

html += `

<li class="question-preview p-2 border rounded mb-2">

<span class="fw-bold text-danger">Q${index + 1}.</span>

${q.question\_text.substring(0, 100)}...

<span class="badge bg-primary ms-3">Marks: ${q.marks}</span>

<span class="badge bg-dark">Difficulty: ${q.difficulty}</span>

</li>

`;

});

html += '</ul>';

container.innerHTML = html;

} catch (error) {

container.innerHTML = '<p class="text-danger text-center">Failed to load questions. Check console.</p>';

}

});

// Add new question to bank

document.getElementById('newQuestionForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const btn = document.getElementById('addNewQBtn');

const payload = {

subjectId: document.getElementById('subjectIdGenerate').value, // Using the generate dropdown subject for context

chapterId: form.newQChapterId.value,

classId: document.getElementById('classIdGenerate').value, // Using the generate dropdown class for context

marks: parseInt(form.newQMarks.value),

difficulty: form.newQDifficulty.value,

type: form.newQType.value,

questionText: form.newQText.value,

correctAnswer: form.newQAnswer.value

};

btn.disabled = true;

btn.textContent = 'Adding...';

try {

const data = await apiFetch('/questions', 'POST', payload);

showNotification(data.message + ` ID: ${data.questionId}`, 'success');

form.reset();

document.getElementById('uploadMediaBtn').disabled = false; // Enable media upload for the new ID

document.getElementById('uploadMediaBtn').setAttribute('data-question-id', data.questionId);

} catch (error) {

// Handled by apiFetch

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-plus-circle"></i> Add Question to Bank';

}

});

// --- 3. APPROVAL WORKFLOW LOGIC (GET /papers, POST /papers/:id/approve) ---

async function fetchGeneratedPapers() {

const tableBody = document.getElementById('papersTableBody');

tableBody.innerHTML = '<tr><td colspan="5" class="text-center"><div class="spinner-border spinner-border-sm"></div> Loading papers...</td></tr>';

currentPapers = [];

try {

const data = await apiFetch('/papers');

currentPapers = data.papers;

tableBody.innerHTML = '';

if (currentPapers.length === 0) {

tableBody.innerHTML = '<tr><td colspan="5" class="text-center">No generated papers found.</td></tr>';

return;

}

currentPapers.forEach(paper => {

const status = paper.status;

const statusClass = {

'draft': 'bg-secondary',

'pending\_hod\_approval': 'bg-warning text-dark',

'approved': 'bg-success',

'rejected': 'bg-danger'

}[status] || 'bg-light text-dark';

const actions = [];

if (status === 'draft') {

actions.push(`<button class="btn btn-sm btn-outline-primary submit-btn" data-paper-id="${paper.id}">Submit to HOD</button>`);

} else if (status === 'pending\_hod\_approval') {

actions.push(`<button class="btn btn-sm btn-success approve-btn" data-paper-id="${paper.id}">Approve</button>`);

} else if (status === 'approved') {

actions.push(`<button class="btn btn-sm btn-danger export-btn" data-paper-id="${paper.id}"><i class="bi bi-file-earmark-pdf-fill"></i> Export/Print</button>`);

}

const row = tableBody.insertRow();

row.innerHTML = `

<td class="small">${paper.id.substring(0, 8)}...</td>

<td>${paper.exam\_name || 'N/A'}</td>

<td>${paper.generated\_by\_user\_profile\_id.substring(0, 8)}...</td>

<td><span class="badge ${statusClass}">${status.toUpperCase()}</span></td>

<td>${actions.join(' ')}</td>

`;

});

} catch (error) {

tableBody.innerHTML = '<tr><td colspan="5" class="text-center text-danger">Failed to load papers.</td></tr>';

}

}

document.getElementById('refreshPapersBtn').addEventListener('click', fetchGeneratedPapers);

// Delegation for workflow buttons

document.getElementById('papersTableBody').addEventListener('click', async (e) => {

const target = e.target.closest('button');

if (!target) return;

const paperId = target.getAttribute('data-paper-id');

if (target.classList.contains('submit-btn')) {

// Submit for Approval

target.disabled = true;

target.textContent = 'Submitting...';

try {

const data = await apiFetch(`/papers/${paperId}/submit-for-approval`, 'POST', {});

showNotification(data.message, 'success');

fetchGeneratedPapers();

} catch (e) {

target.disabled = false;

target.textContent = 'Submit to HOD';

}

} else if (target.classList.contains('approve-btn')) {

// HOD/Principal Approval

target.disabled = true;

target.textContent = 'Approving...';

try {

const data = await apiFetch(`/papers/${paperId}/approve`, 'POST', {});

showNotification(data.message, 'success');

fetchGeneratedPapers();

} catch (e) {

target.disabled = false;

target.textContent = 'Approve';

}

} else if (target.classList.contains('export-btn')) {

// Export/Print Gated Endpoint

target.disabled = true;

target.textContent = 'Exporting...';

try {

const data = await apiFetch(`/papers/${paperId}/export`);

document.getElementById('exportStatus').textContent = data.status.toUpperCase();

document.getElementById('watermarkStatus').textContent = data.layout.watermarkEnabled ? 'ENABLED' : 'DISABLED';

document.getElementById('exportMetadata').innerHTML = `

<li><strong>School:</strong> ${data.watermark.name || 'N/A'}</li>

<li><strong>Size:</strong> ${data.layout.paperSize} (${data.layout.orientation})</li>

<li><strong>Margins:</strong> T:${data.layout.marginTop}mm, B:${data.layout.marginBottom}mm, L:${data.layout.marginLeft}mm, R:${data.layout.marginRight}mm</li>

`;

const exportModal = new bootstrap.Modal(document.getElementById('exportModal'));

exportModal.show();

showNotification('Paper data fetched successfully for printing.', 'success');

} catch (e) {

// Handled by apiFetch (Will show 403 error if not approved)

} finally {

target.disabled = false;

target.innerHTML = '<i class="bi bi-file-earmark-pdf-fill"></i> Export/Print';

}

}

});

// --- 4. CONFIGURATION LOGIC (GET/PUT /config) ---

async function fetchConfig() {

try {

const data = await apiFetch('/config');

const settings = data.settings || {};

const s = settings.settings\_json || {};

const l = settings.layout\_settings || {};

// Core Settings

document.getElementById('defaultTemplateId').value = s.defaultTemplateId || 'TPL-MIDTERM-1';

document.getElementById('defaultTotalMarks').value = s.defaultTotalMarks || 50;

document.getElementById('defaultDuration').value = s.defaultDuration || 120;

document.getElementById('passingMarksPercentage').value = s.passingMarksPercentage || 35;

document.getElementById('approvalRoles').value = (s.approvalRoles || ['principal', 'hod']).join(', ');

document.getElementById('templateStructures').value = JSON.stringify(s.templateStructures || [], null, 2);

// Layout Settings

document.getElementById('paperSize').value = l.paperSize || 'A4';

document.getElementById('orientation').value = l.orientation || 'Portrait';

document.getElementById('watermarkEnabled').checked = l.watermarkEnabled !== false; // Default true

document.getElementById('marginTop').value = l.marginTop || 10;

document.getElementById('marginBottom').value = l.marginBottom || 10;

document.getElementById('marginLeft').value = l.marginLeft || 10;

document.getElementById('marginRight').value = l.marginRight || 10;

} catch (error) {

showNotification('Configuration loaded with default fallback values (or access denied).', 'warning');

}

}

document.getElementById('configForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const btn = document.getElementById('saveConfigBtn');

// --- Parse and Validate Inputs ---

let templateStructuresArray;

try {

templateStructuresArray = JSON.parse(form.templateStructures.value);

if (!Array.isArray(templateStructuresArray)) throw new Error("Must be an array.");

} catch (e) {

showNotification('Template Structure must be valid JSON array.', 'danger');

return;

}

const layoutSettings = {

paperSize: form.paperSize.value,

orientation: form.orientation.value,

watermarkEnabled: form.watermarkEnabled.checked,

marginTop: parseInt(form.marginTop.value),

marginBottom: parseInt(form.marginBottom.value),

marginLeft: parseInt(form.marginLeft.value),

marginRight: parseInt(form.marginRight.value),

};

const payload = {

defaultTemplateId: form.defaultTemplateId.value,

defaultTotalMarks: parseInt(form.defaultTotalMarks.value),

defaultDuration: parseInt(form.defaultDuration.value),

passingMarksPercentage: parseInt(form.passingMarksPercentage.value),

approvalRoles: form.approvalRoles.value.split(',').map(s => s.trim()),

templateStructures: templateStructuresArray,

layoutSettings: layoutSettings

};

btn.disabled = true;

btn.innerHTML = '<span class="spinner-border spinner-border-sm"></span> Saving...';

try {

const data = await apiFetch('/config', 'PUT', payload);

showNotification(data.message, 'success');

} catch (error) {

// Handled by apiFetch

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-save"></i> Save Configuration & Layout';

}

});

// --- Initialization ---

document.addEventListener('DOMContentLoaded', () => {

populateSubjectDropdowns();

fetchConfig();

fetchGeneratedPapers();

});

</script>

</body>

</html>

18. client/dist/modules/Reporttracker.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>📊 Student Report Card Tracker & Analytics</title>

<!-- Bootstrap 5 CDN for styling and responsiveness -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css">

<style>

body { background-color: #f0f2f5; font-family: 'Inter', sans-serif; }

.container-fluid { max-width: 1200px; margin-top: 20px; }

.card-header { background-color: #00796b; /\* Teal \*/ color: white; font-weight: bold; border-radius: 0.5rem 0.5rem 0 0; }

.section-card { margin-bottom: 25px; box-shadow: 0 4px 15px rgba(0, 0, 0, 0.1); border-radius: 0.5rem; }

.alert-fixed { position: fixed; top: 20px; right: 20px; z-index: 1050; min-width: 300px; border-radius: 0.5rem; }

.nav-tabs .nav-link.active { background-color: #00897b; color: white; border-color: #00897b; }

.marks-input { max-width: 80px; }

</style>

</head>

<body>

<div class="container-fluid">

<h1 class="text-center mb-4 text-primary"><i class="bi bi-journal-check"></i> Report Card Management & Compliance</h1>

<!-- Notification Area -->

<div id="notification-area" class="alert-fixed" style="display: none;"></div>

<!-- Tab Navigation -->

<ul class="nav nav-tabs" id="reportTab" role="tablist">

<li class="nav-item"><button class="nav-link active" id="entry-tab" data-bs-toggle="tab" data-bs-target="#mark-entry" type="button"><i class="bi bi-pencil-square"></i> Mark Entry & Finalization</button></li>

<li class="nav-item"><button class="nav-link" id="approval-tab" data-bs-toggle="tab" data-bs-target="#approval-dist" type="button"><i class="bi bi-patch-check-fill"></i> Principal Approval & Distribution</button></li>

<li class="nav-item"><button class="nav-link" id="analytics-tab" data-bs-toggle="tab" data-bs-target="#analytics-view" type="button"><i class="bi bi-graph-up"></i> Analytics & Oversight</button></li>

<li class="nav-item"><button class="nav-link" id="config-tab" data-bs-toggle="tab" data-bs-target="#config-setup" type="button"><i class="bi bi-gear-fill"></i> Configuration</button></li>

</ul>

<!-- Tab Content -->

<div class="tab-content pt-3" id="reportTabContent">

<!-- 1. Mark Entry & Finalization Tab (POST /reports/generate, POST /reports/:id/feed-marks, POST /reports/:id/finalize) -->

<div class="tab-pane fade show active" id="mark-entry" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-success">Step 1: Report Generation & Mark Entry (Teacher Action)</div>

<div class="card-body">

<!-- Report Generation Form -->

<h5 class="text-success">A. Generate Draft Reports (Bulk)</h5>

<form id="generateForm" class="row g-3 mb-4">

<div class="col-md-4">

<label for="reportType" class="form-label">Report Type</label>

<select class="form-select" id="reportType" required>

<option value="academic">Academic Report Card</option>

<option value="behavioral">Behavioral Assessment</option>

</select>

</div>

<div class="col-md-4">

<label for="term" class="form-label">Term/Cycle</label>

<input type="text" class="form-control" id="term" placeholder="e.g., Term 1, 2024-25" required>

</div>

<div class="col-md-4">

<label for="studentIds" class="form-label">Student IDs (Comma Separated)</label>

<input type="text" class="form-control" id="studentIds" placeholder="e.g., S101, S102, S103" required>

</div>

<div class="col-12">

<button type="submit" class="btn btn-success" id="generateBtn"><i class="bi bi-file-earmark-plus"></i> Generate Drafts</button>

</div>

</form>

<hr class="my-4">

<!-- Marks Feeding Form (Simulated per subject/student) -->

<h5 class="text-success">B. Feed Marks & Remarks</h5>

<form id="marksFeedForm" class="row g-3">

<div class="col-md-3">

<label for="feedReportId" class="form-label">Report ID</label>

<input type="text" class="form-control" id="feedReportId" placeholder="e.g., RPT-001" required>

</div>

<div class="col-md-3">

<label for="feedStudentId" class="form-label">Student ID</label>

<input type="text" class="form-control" id="feedStudentId" placeholder="e.g., S101" required>

</div>

<div class="col-md-3">

<label for="feedSubjectId" class="form-label">Subject ID</label>

<select class="form-select" id="feedSubjectId" required>

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

<option value="PHY">Physics</option>

<option value="HIS">History</option>

</select>

</div>

<div class="col-md-3">

<label for="feedMarks" class="form-label">Marks (Out of 100)</label>

<input type="number" class="form-control marks-input" id="feedMarks" min="0" max="100" required>

</div>

<div class="col-12">

<label for="feedRemarks" class="form-label">Remarks / Teacher Notes</label>

<textarea class="form-control" id="feedRemarks" rows="2"></textarea>

</div>

<div class="col-12">

<button type="submit" class="btn btn-primary me-2"><i class="bi bi-send-fill"></i> Feed Marks to Report</button>

<button type="button" class="btn btn-warning" id="finalizeBtn" disabled><i class="bi bi-lock-fill"></i> Finalize Report for Approval</button>

</div>

</form>

</div>

</div>

</div>

<!-- 2. Principal Approval & Distribution Tab (POST /reports/:id/distribute) -->

<div class="tab-pane fade" id="approval-dist" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-warning text-dark">Step 2: Principal Gated Workflow & Distribution</div>

<div class="card-body">

<p class="alert alert-danger small"><i class="bi bi-shield-fill-exclamation"></i> Distribution is \*\*CRITICALLY GATED\*\*. Only the Principal/Super Admin can approve and distribute reports to parents.</p>

<!-- Approval Pending List (Simulated) -->

<h5 class="text-warning">Reports Pending Principal Approval</h5>

<div class="table-responsive mb-4">

<table class="table table-striped table-sm">

<thead>

<tr>

<th>Report ID</th>

<th>Student</th>

<th>Term</th>

<th>Status</th>

<th>Action (Principal)</th>

</tr>

</thead>

<tbody id="approvalTableBody">

<tr><td colspan="5" class="text-center">No reports currently pending approval.</td></tr>

</tbody>

</table>

</div>

<!-- Distribution Action -->

<h5 class="text-success">C. Final Distribution</h5>

<div class="row g-3">

<div class="col-md-4">

<label for="distReportId" class="form-label">Approved Report ID</label>

<input type="text" class="form-control" id="distReportId" placeholder="RPT-001" required>

<div class="form-text">Must be a report with 'approved' status.</div>

</div>

<div class="col-md-4">

<label for="distMethod" class="form-label">Distribution Method</label>

<select class="form-select" id="distMethod">

<option value="digital">Digital (Arattai/Email)</option>

<option value="print">Print & Handout</option>

</select>

</div>

<div class="col-md-4">

<button type="button" class="btn btn-danger mt-4 w-100" id="distributeBtn"><i class="bi bi-bell-fill"></i> Approve & Distribute to Parents</button>

</div>

</div>

</div>

</div>

</div>

<!-- 3. Analytics & Oversight Tab (GET /analytics) -->

<div class="tab-pane fade" id="analytics-view" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-info">Comprehensive Performance Analytics (Admin/Principal View)</div>

<div class="card-body">

<h5 class="text-info">Hierarchy-Based Performance Analysis</h5>

<div class="row g-3 mb-4">

<div class="col-md-4">

<label for="hierarchySelect" class="form-label">Analyze By Hierarchy</label>

<select class="form-select" id="hierarchySelect">

<option value="class">Class (Overall Avg)</option>

<option value="section">Section</option>

<option value="wing">Wing/Department</option>

<option value="student">Individual Student (Requires ID)</option>

</select>

</div>

<div class="col-md-4">

<label for="hierarchyFilterId" class="form-label">Filter ID (Optional)</label>

<input type="text" class="form-control" id="hierarchyFilterId" placeholder="e.g., Class X-A, Student S101">

</div>

<div class="col-md-4">

<button type="button" class="btn btn-info mt-4 w-100" id="runAnalyticsBtn"><i class="bi bi-bar-chart-line-fill"></i> Run Analysis</button>

</div>

</div>

<div id="analyticsResults" class="mt-3">

<p class="text-muted text-center">Run an analysis to see aggregated results here.</p>

</div>

<hr class="my-4">

<h5 class="text-danger">Report List & Export</h5>

<div class="d-flex justify-content-between mb-3">

<input type="text" class="form-control me-2" id="reportSearch" placeholder="Search by Student ID or Term...">

<button class="btn btn-outline-danger me-2" id="exportCsvBtn"><i class="bi bi-filetype-csv"></i> Export CSV</button>

<button class="btn btn-outline-danger" id="exportPdfBtn"><i class="bi bi-file-pdf"></i> Export PDF</button>

</div>

<div class="table-responsive">

<table class="table table-striped table-sm">

<thead><tr><th>ID</th><th>Student</th><th>Term</th><th>Status</th><th>Date</th></tr></thead>

<tbody id="reportListBody"><tr><td colspan="5" class="text-center">Loading reports...</td></tr></tbody>

</table>

</div>

</div>

</div>

</div>

<!-- 4. Configuration Tab (GET/PUT /config) -->

<div class="tab-pane fade" id="config-setup" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-dark">Report Configuration & Grading Scales (Admin/Principal Only)</div>

<div class="card-body">

<p class="alert alert-danger small"><i class="bi bi-lock-fill"></i> Requires Admin/Principal access to maintain compliance and academic standards.</p>

<form id="configForm" class="row g-3">

<h5 class="text-dark">A. Subject Weightage & Grading Scale</h5>

<div class="col-md-6">

<label for="gradingScale" class="form-label">Grading Scale (JSON)</label>

<textarea class="form-control" id="gradingScale" rows="4" required placeholder='{"A1": [91, 100], "A2": [81, 90], ...}'></textarea>

<div class="form-text">Defines how percentage marks translate to grades.</div>

</div>

<div class="col-md-6">

<label for="subjectCategoryWeights" class="form-label">Subject Category Weights (JSON)</label>

<textarea class="form-control" id="subjectCategoryWeights" rows="4" required placeholder='{"main": 0.7, "co\_curricular": 0.3} - Sum must be 1.0'></textarea>

<div class="form-text">Used for calculating final overall performance index.</div>

</div>

<hr class="my-3">

<h5 class="text-dark">B. Report Template & Finalization Policy</h5>

<div class="col-md-6">

<label for="templatePolicy" class="form-label">Default Report Template ID</label>

<input type="text" class="form-control" id="templatePolicy" placeholder="e.g., CBSE\_T1\_V2" required>

<div class="form-text">Controls the visual and structural output of the final report.</div>

</div>

<div class="col-md-6">

<label for="finalizationPolicy" class="form-label">Finalization Policy</label>

<select class="form-select" id="finalizationPolicy" required>

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

<option value="hod">Requires HOD Approval</option>

</select>

<div class="form-text">Who must give final sign-off before distribution.</div>

</div>

<div class="col-12 mt-4">

<button type="submit" class="btn btn-dark btn-lg w-100" id="saveConfigBtn"><i class="bi bi-save"></i> Save Configuration</button>

</div>

</form>

</div>

</div>

</div>

</div>

</div>

<!-- Modal for Approval/Review -->

<div class="modal fade" id="reviewModal" tabindex="-1">

<div class="modal-dialog modal-lg">

<div class="modal-content">

<div class="modal-header bg-primary text-white">

<h5 class="modal-title">Report Review & Approval</h5>

<button type="button" class="btn-close" data-bs-dismiss="modal"></button>

</div>

<div class="modal-body">

<p>Review content for Report ID: <strong id="modalReportId"></strong></p>

<div id="modalReportContent" class="p-3 border rounded bg-light">

<p class="text-muted">Detailed marks, remarks, and overall performance metrics will load here for final review.</p>

<!-- Simulated detailed content -->

<h6>Student: Aarav Sharma | Class: X-A | Overall %: 88.5% (A1)</h6>

<ul class="list-unstyled small">

<li>Maths: 95/100 (Excellent work on Geometry.)</li>

<li>Science: 82/100 (Needs improvement in practical lab skills.)</li>

</ul>

</div>

</div>

<div class="modal-footer">

<button type="button" class="btn btn-secondary" data-bs-dismiss="modal">Close</button>

<button type="button" class="btn btn-danger" id="rejectReportBtn"><i class="bi bi-x-circle"></i> Reject & Send Back</button>

<button type="button" class="btn btn-success" id="approveReportBtn"><i class="bi bi-check-circle"></i> Approve Report</button>

</div>

</div>

</div>

</div>

<!-- Bootstrap Bundle with Popper -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>

<script>

const API\_BASE = '';

const MOCK\_JWT\_TOKEN = 'mock.jwt.token.for.auth';

// Mock data to simulate API responses

let mockReports = [

{ id: 'RPT-001', student\_id: 'S101', report\_type: 'Academic', generated\_date: '2025-09-01', status: 'pending\_approval' },

{ id: 'RPT-002', student\_id: 'S102', report\_type: 'Academic', generated\_date: '2025-08-25', status: 'approved' },

{ id: 'RPT-003', student\_id: 'S103', report\_type: 'Behavioral', generated\_date: '2025-08-20', status: 'draft' },

];

let mockAnalytics = [

{ hierarchy\_name: 'Class X', avg\_performance: '82.5' },

{ hierarchy\_name: 'Class XII', avg\_performance: '79.1' },

{ hierarchy\_name: 'Junior Wing', avg\_performance: '88.0' },

];

/\*\* Helper function to show a temporary notification. \*/

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

const area = document.getElementById('notification-area');

area.innerHTML = `

<div class="alert alert-${type} alert-dismissible fade show" role="alert">

${message}

<button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>

</div>

`;

area.style.display = 'block';

setTimeout(() => { area.style.display = 'none'; }, 7000);

}

/\*\* Core Fetch Function with JWT Auth Header. \*/

async function apiFetch(endpoint, method = 'GET', body = null) {

try {

const options = {

method: method,

headers: {

'Content-Type': 'application/json',

'Authorization': `Bearer ${MOCK\_JWT\_TOKEN}`

},

};

if (body && method !== 'GET') {

options.body = JSON.stringify(body);

}

const response = await fetch(`${API\_BASE}${endpoint}`, options);

const data = await response.json();

if (!response.ok) {

throw new Error(data.error || `API error (${response.status})`);

}

return data;

} catch (error) {

console.error('API Fetch Error:', endpoint, error);

showNotification(error.message || 'Network error. Check console for details.', 'danger');

throw error;

}

}

// --- 1. MARK ENTRY & FINALIZATION LOGIC ---

document.getElementById('generateForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const btn = document.getElementById('generateBtn');

const studentIds = form.studentIds.value.split(',').map(id => id.trim()).filter(id => id);

// Mocking UUIDs as required by backend schema

const validatedStudentIds = studentIds.map(id => `id-${id}-0000-0000-0000-000000000000`);

const payload = {

studentIds: validatedStudentIds,

reportType: form.reportType.value,

term: form.term.value,

templateId: 'CBSE\_T1\_V2'

};

btn.disabled = true;

btn.innerHTML = '<span class="spinner-border spinner-border-sm"></span> Generating Drafts...';

try {

const data = await apiFetch('/reports/generate', 'POST', payload);

showNotification(data.message, 'success');

fetchReportList(); // Refresh list to see new drafts

} catch (error) {

// Handled by apiFetch

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-file-earmark-plus"></i> Generate Drafts';

}

});

document.getElementById('marksFeedForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const btn = form.querySelector('.btn-primary');

const reportId = form.feedReportId.value;

// Mock student ID to UUID format for compliance

const studentId = `id-${form.feedStudentId.value}-0000-0000-0000-000000000000`;

const payload = {

studentId: studentId,

subjectId: form.feedSubjectId.value,

marks: parseInt(form.feedMarks.value),

remarks: form.feedRemarks.value

};

btn.disabled = true;

btn.innerHTML = '<span class="spinner-border spinner-border-sm"></span> Submitting...';

try {

const data = await apiFetch(`/reports/${reportId}/feed-marks`, 'POST', payload);

showNotification(data.message, 'success');

// Enable finalize button after marks are fed successfully (Simulated)

document.getElementById('finalizeBtn').disabled = false;

form.reset();

} catch (error) {

// Handled by apiFetch

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-send-fill"></i> Feed Marks to Report';

}

});

document.getElementById('finalizeBtn').addEventListener('click', async (e) => {

const reportId = document.getElementById('marksFeedForm').feedReportId.value;

if (!reportId) {

showNotification('Please enter a Report ID to finalize.', 'warning');

return;

}

const btn = e.target;

btn.disabled = true;

btn.textContent = 'Finalizing...';

try {

const data = await apiFetch(`/reports/${reportId}/finalize`, 'POST', {});

showNotification(data.message, 'success');

fetchApprovalList(); // Refresh list to show new pending report

} catch (error) {

// Handled by apiFetch

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-lock-fill"></i> Finalize Report for Approval';

}

});

// --- 2. PRINCIPAL APPROVAL & DISTRIBUTION LOGIC ---

async function fetchApprovalList() {

const tableBody = document.getElementById('approvalTableBody');

tableBody.innerHTML = '<tr><td colspan="5" class="text-center"><div class="spinner-border spinner-border-sm"></div> Loading pending reports...</td></tr>';

try {

// Simulating fetching only pending reports

// In production, this would be a filtered call to GET /reports?status=pending\_approval

const data = await apiFetch('/reports');

const pendingReports = data.reports.filter(r => r.status === 'pending\_approval');

tableBody.innerHTML = '';

if (pendingReports.length === 0) {

tableBody.innerHTML = '<tr><td colspan="5" class="text-center">No reports currently pending Principal approval.</td></tr>';

return;

}

pendingReports.forEach(report => {

const row = tableBody.insertRow();

row.innerHTML = `

<td>${report.id.substring(0, 8)}...</td>

<td>${report.student\_id.substring(0, 8)}...</td>

<td>${report.report\_type} - ${report.term || 'T1'}</td>

<td><span class="badge bg-warning text-dark">${report.status.toUpperCase()}</span></td>

<td>

<button class="btn btn-sm btn-outline-info review-btn me-2" data-bs-toggle="modal" data-bs-target="#reviewModal" data-report-id="${report.id}">Review</button>

<button class="btn btn-sm btn-success approve-btn-list" data-report-id="${report.id}">Approve</button>

</td>

`;

});

} catch (error) {

tableBody.innerHTML = '<tr><td colspan="5" class="text-center text-danger">Failed to load reports for approval.</td></tr>';

}

}

document.getElementById('approvalTableBody').addEventListener('click', (e) => {

const target = e.target.closest('button');

if (!target) return;

const reportId = target.getAttribute('data-report-id');

if (target.classList.contains('review-btn')) {

document.getElementById('modalReportId').textContent = reportId.substring(0, 8) + '...';

// In production, fetch detailed report data and populate modalReportContent

} else if (target.classList.contains('approve-btn-list')) {

// Directly trigger approval process

processApproval(reportId, true);

}

});

async function processApproval(reportId, isApproved) {

// NOTE: Assuming there is a separate /reports/:reportId/set-status endpoint for approval/rejection.

// Since the backend only shows /distribute, we simulate the approval status update for now.

const newStatus = isApproved ? 'approved' : 'rejected';

showNotification(`Simulating Report ID ${reportId} status update to ${newStatus}.`, 'info');

// Mock DB update for demonstration

const reportIndex = mockReports.findIndex(r => r.id === reportId);

if (reportIndex !== -1) {

mockReports[reportIndex].status = newStatus;

}

// In a real system, you'd call a dedicated PUT/POST endpoint here.

// const data = await apiFetch(`/reports/${reportId}/set-status`, 'PUT', { status: newStatus });

fetchApprovalList();

fetchReportList();

bootstrap.Modal.getInstance(document.getElementById('reviewModal'))?.hide();

}

document.getElementById('approveReportBtn').addEventListener('click', () => {

const reportId = document.getElementById('modalReportId').textContent.replace('...', '').trim();

processApproval(reportId, true);

});

document.getElementById('rejectReportBtn').addEventListener('click', () => {

const reportId = document.getElementById('modalReportId').textContent.replace('...', '').trim();

processApproval(reportId, false);

});

document.getElementById('distributeBtn').addEventListener('click', async (e) => {

const reportId = document.getElementById('distReportId').value;

const method = document.getElementById('distMethod').value;

if (!reportId) { showNotification('Enter a Report ID to distribute.', 'warning'); return; }

const btn = e.target;

btn.disabled = true;

btn.textContent = 'Distributing...';

try {

const data = await apiFetch(`/reports/${reportId}/distribute`, 'POST', { distributionMethod: method });

showNotification(data.message, 'success');

fetchReportList();

} catch (error) {

// Handled by apiFetch (Will catch 400 error if not approved)

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-bell-fill"></i> Approve & Distribute to Parents';

}

});

// --- 3. ANALYTICS & REPORTING LOGIC ---

async function runAnalytics() {

const hierarchy = document.getElementById('hierarchySelect').value;

const filterId = document.getElementById('hierarchyFilterId').value;

const btn = document.getElementById('runAnalyticsBtn');

const resultsDiv = document.getElementById('analyticsResults');

let endpoint = `/analytics?hierarchy=${hierarchy}`;

if (filterId) endpoint += `&filterId=${filterId}`;

btn.disabled = true;

btn.innerHTML = '<span class="spinner-border spinner-border-sm"></span> Analyzing...';

resultsDiv.innerHTML = '<p class="text-center text-muted">Running analysis...</p>';

try {

const data = await apiFetch(endpoint); // Data is mockAnalytics or similar structure

let html = `<p class="fw-bold">${data.message}</p><ul class="list-group">`;

data.results.forEach(item => {

const avg = parseFloat(item.avg\_performance || 0).toFixed(2);

html += `<li class="list-group-item d-flex justify-content-between align-items-center">

${item.hierarchy\_name}:

<span class="badge bg-primary rounded-pill">${avg}% Avg</span>

</li>`;

});

html += '</ul>';

resultsDiv.innerHTML = html;

} catch (error) {

resultsDiv.innerHTML = `<p class="text-danger">Failed to run analytics. ${error.message}</p>`;

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-bar-chart-line-fill"></i> Run Analysis';

}

}

document.getElementById('runAnalyticsBtn').addEventListener('click', runAnalytics);

async function fetchReportList() {

const tableBody = document.getElementById('reportListBody');

tableBody.innerHTML = '<tr><td colspan="5" class="text-center"><div class="spinner-border spinner-border-sm"></div> Loading history...</td></tr>';

try {

// Simulating fetching data from the backend GET /reports endpoint

// In production, this would be const data = await apiFetch('/reports');

const data = { reports: mockReports };

tableBody.innerHTML = '';

data.reports.forEach(report => {

const statusClass = report.status === 'approved' ? 'bg-success' : (report.status === 'pending\_approval' ? 'bg-warning text-dark' : 'bg-secondary');

const row = tableBody.insertRow();

row.innerHTML = `

<td>${report.id.substring(0, 8)}...</td>

<td>${report.student\_id.substring(0, 8)}...</td>

<td>${report.report\_type} - ${report.term || 'T1'}</td>

<td><span class="badge ${statusClass}">${report.status.toUpperCase()}</span></td>

<td>${new Date(report.generated\_date).toLocaleDateString()}</td>

`;

});

} catch (error) {

tableBody.innerHTML = '<tr><td colspan="5" class="text-center text-danger">Failed to load report history.</td></tr>';

}

}

document.getElementById('exportCsvBtn').addEventListener('click', async () => {

try {

const data = await apiFetch('/reports?export=csv');

showNotification(data.message, 'success');

} catch (e) {

// Handled by apiFetch

}

});

document.getElementById('exportPdfBtn').addEventListener('click', async () => {

try {

const data = await apiFetch('/reports?export=pdf');

showNotification(data.message, 'success');

} catch (e) {

// Handled by apiFetch

}

});

// --- 4. CONFIGURATION LOGIC ---

async function fetchConfig() {

try {

const data = await apiFetch('/config');

const settings = data.settings || {};

document.getElementById('gradingScale').value = JSON.stringify(settings.grading\_scale || {"A1": [91, 100], "B1": [71, 80]}, null, 2);

document.getElementById('subjectCategoryWeights').value = JSON.stringify(settings.subject\_category\_weights || {"main": 0.7, "co\_curricular": 0.3}, null, 2);

document.getElementById('templatePolicy').value = settings.default\_template\_id || 'CBSE\_T1\_V2';

document.getElementById('finalizationPolicy').value = settings.finalization\_approver || 'principal';

} catch (error) {

showNotification('Configuration loaded with default fallback values (or access denied).', 'warning');

}

}

document.getElementById('configForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const btn = document.getElementById('saveConfigBtn');

let gradingScaleData, subjectCategoryWeightsData;

try {

gradingScaleData = JSON.parse(form.gradingScale.value);

subjectCategoryWeightsData = JSON.parse(form.subjectCategoryWeights.value);

} catch (e) {

showNotification('Grading Scale and Subject Weights must be valid JSON.', 'danger');

return;

}

const payload = {

gradingScale: gradingScaleData,

subjectCategoryWeights: subjectCategoryWeightsData,

reportTemplates: [{ id: form.templatePolicy.value }], // Simulating template update

finalizationPolicy: form.finalizationPolicy.value

};

btn.disabled = true;

btn.innerHTML = '<span class="spinner-border spinner-border-sm"></span> Saving...';

try {

const data = await apiFetch('/config', 'PUT', payload);

showNotification(data.message, 'success');

} catch (error) {

// Handled by apiFetch

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-save"></i> Save Configuration';

}

});

// --- Initialization ---

document.addEventListener('DOMContentLoaded', () => {

fetchReportList();

fetchApprovalList();

fetchConfig();

// Initial call to run analytics when tab is opened

const analyticsTab = document.getElementById('analytics-tab');

analyticsTab.addEventListener('shown.bs.tab', runAnalytics);

});

</script>

</body>

</html>

19. client/dist/modules/school\_event.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>🗓️ School Event Log & Compliance</title>

<!-- Bootstrap 5 CDN for styling and responsiveness -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css">

<style>

body { background-color: #f0f2f5; font-family: 'Inter', sans-serif; }

.container-fluid { max-width: 1200px; margin-top: 20px; }

.card-header { background-color: #00796b; /\* Teal \*/ color: white; font-weight: bold; border-radius: 0.5rem 0.5rem 0 0; }

.section-card { margin-bottom: 25px; box-shadow: 0 4px 15px rgba(0, 0, 0, 0.1); border-radius: 0.5rem; }

.alert-fixed { position: fixed; top: 20px; right: 20px; z-index: 1050; min-width: 300px; border-radius: 0.5rem; }

.nav-tabs .nav-link.active { background-color: #00897b; color: white; border-color: #00897b; }

.event-badge {

font-size: 0.75rem;

margin-right: 5px;

}

</style>

</head>

<body>

<div class="container-fluid">

<h1 class="text-center mb-4 text-primary"><i class="bi bi-calendar-event"></i> School Event Log & Compliance Dashboard</h1>

<!-- Notification Area -->

<div id="notification-area" class="alert-fixed" style="display: none;"></div>

<!-- Tab Navigation -->

<ul class="nav nav-tabs" id="eventTab" role="tablist">

<li class="nav-item"><button class="nav-link active" id="create-tab" data-bs-toggle="tab" data-bs-target="#event-creation" type="button"><i class="bi bi-plus-square"></i> Calendar & Creation</button></li>

<li class="nav-item"><button class="nav-link" id="workflow-tab" data-bs-toggle="tab" data-bs-target="#event-workflow" type="button"><i class="bi bi-patch-check"></i> Approval & Notifications</button></li>

<li class="nav-item"><button class="nav-link" id="feed-tab" data-bs-toggle="tab" data-bs-target="#event-feed" type="button"><i class="bi bi-broadcast-pin"></i> Live Event Feed</button></li>

<li class="nav-item"><button class="nav-link" id="config-tab" data-bs-toggle="tab" data-bs-target="#event-config" type="button"><i class="bi bi-gear-fill"></i> Configuration</button></li>

</ul>

<!-- Tab Content -->

<div class="tab-content pt-3" id="eventTabContent">

<!-- 1. Calendar & Creation Tab (POST /, GET /) -->

<div class="tab-pane fade show active" id="event-creation" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-success">Event Planning & Venue Conflict Check</div>

<div class="card-body">

<h5 class="text-success">A. Create New Event</h5>

<form id="eventForm" class="row g-3 mb-4">

<div class="col-md-6">

<label for="eventTitle" class="form-label">Event Title</label>

<input type="text" class="form-control" id="eventTitle" required maxlength="255">

</div>

<div class="col-md-6">

<label for="eventType" class="form-label">Type</label>

<select class="form-select" id="eventType" required>

<option value="academic">Academic</option>

<option value="sports">Sports</option>

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

<option value="meeting">Meeting</option>

<option value="competition">Competition</option>

<option value="workshop">Workshop</option>

<option value="celebration">Celebration</option>

<option value="trip">Trip/Excursion</option>

</select>

</div>

<div class="col-md-3">

<label for="startDate" class="form-label">Start Date</label>

<input type="date" class="form-control" id="startDate" required>

</div>

<div class="col-md-3">

<label for="endDate" class="form-label">End Date</label>

<input type="date" class="form-control" id="endDate" required>

</div>

<div class="col-md-3">

<label for="startTime" class="form-label">Start Time</label>

<input type="time" class="form-control" id="startTime" required>

</div>

<div class="col-md-3">

<label for="endTime" class="form-label">End Time</label>

<input type="time" class="form-control" id="endTime" required>

</div>

<div class="col-md-4">

<label for="venue" class="form-label">Venue</label>

<input type="text" class="form-control" id="venue" placeholder="e.g., Auditorium, Ground A" required>

</div>

<div class="col-md-4">

<label for="organizer" class="form-label">Organizer/Incharge</label>

<input type="text" class="form-control" id="organizer" required>

</div>

<div class="col-md-4">

<label for="allocatedBudget" class="form-label">Allocated Budget (Optional)</label>

<input type="number" class="form-control" id="allocatedBudget" min="0" placeholder="₹">

</div>

<div class="col-md-6">

<label for="priority" class="form-label">Priority</label>

<select class="form-select" id="priority" required>

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

<option value="high">High (Requires Principal Alert)</option>

<option value="urgent">Urgent (Requires Immediate Principal Alert)</option>

<option value="low">Low</option>

</select>

</div>

<div class="col-12">

<label for="description" class="form-label">Description (Max 1000 chars)</label>

<textarea class="form-control" id="description" rows="2" maxlength="1000"></textarea>

</div>

<div class="col-12">

<button type="submit" class="btn btn-success" id="createEventBtn"><i class="bi bi-check-circle"></i> Create Event (Planning)</button>

</div>

</form>

<hr class="my-4">

<h5 class="text-primary">B. Upcoming Events (Live Calendar View)</h5>

<div class="table-responsive">

<table class="table table-striped table-sm">

<thead>

<tr>

<th>Title</th>

<th>Venue</th>

<th>Dates</th>

<th>Time</th>

<th>Status</th>

</tr>

</thead>

<tbody id="eventListBody">

<tr><td colspan="5" class="text-center">Loading events...</td></tr>

</tbody>

</table>

</div>

</div>

</div>

</div>

<!-- 2. Approval & Notifications Tab (POST /:eventId/approve, POST /:eventId/notify) -->

<div class="tab-pane fade" id="event-workflow" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-warning text-dark">Principal Approval & Communication</div>

<div class="card-body">

<!-- Approval Section -->

<h5 class="text-warning">A. Event Approval (Principal/Approver Only)</h5>

<form id="approvalForm" class="row g-3 mb-4">

<div class="col-md-8">

<label for="approveEventId" class="form-label">Event ID to Approve</label>

<input type="text" class="form-control" id="approveEventId" placeholder="e.g., EVT-123" required>

</div>

<div class="col-md-4">

<button type="submit" class="btn btn-warning mt-4 w-100" id="approveBtn"><i class="bi bi-patch-check-fill"></i> Approve Event</button>

</div>

</form>

<hr class="my-4">

<!-- Notification Section -->

<h5 class="text-info">B. Send Final Reminder / Notification</h5>

<form id="notifyForm" class="row g-3">

<div class="col-md-4">

<label for="notifyEventId" class="form-label">Event ID</label>

<input type="text" class="form-control" id="notifyEventId" placeholder="e.g., EVT-123" required>

</div>

<div class="col-md-4">

<label for="notifyRecipients" class="form-label">Recipients (Comma Separated)</label>

<select class="form-select" id="notifyRecipients" multiple required>

<option value="parents" selected>Parents</option>

<option value="students">Students</option>

<option value="staff">Staff</option>

</select>

</div>

<div class="col-md-4">

<label for="notifyMethod" class="form-label">Delivery Method</label>

<select class="form-select" id="notifyMethod" required>

<option value="whatsapp" selected>WhatsApp (Arattai)</option>

<option value="email">Email</option>

<option value="sms">SMS</option>

</select>

</div>

<div class="col-12">

<label for="notifyMessage" class="form-label">Custom Message (Template Message)</label>

<textarea class="form-control" id="notifyMessage" rows="3" required placeholder="Event starts soon! Don't forget your materials."></textarea>

</div>

<div class="col-12">

<button type="submit" class="btn btn-info" id="notifyBtn"><i class="bi bi-send-fill"></i> Send Notification</button>

</div>

</form>

</div>

</div>

</div>

<!-- 3. Live Event Feed Tab (Gated by Config) -->

<div class="tab-pane fade" id="event-feed" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-primary">Live Event Feed (Filtered by Configuration)</div>

<div class="card-body">

<p class="alert alert-info small">This feed dynamically monitors and displays events matching the \*\*Configured Priority\*\* and \*\*Types\*\* (set in the Configuration tab).</p>

<div id="liveFeedBody">

<p class="text-muted text-center">No live events matching the configured filters...</p>

</div>

</div>

</div>

</div>

<!-- 4. Configuration Tab (GET/PUT /config) -->

<div class="tab-pane fade" id="event-config" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-dark">Live Monitoring Configuration (Admin/Principal Only)</div>

<div class="card-body">

<form id="configForm" class="row g-3">

<p class="alert alert-warning small">These settings control which events appear on the \*\*Live Event Feed\*\* tab for real-time monitoring.</p>

<div class="col-md-6">

<label for="liveFeedTypes" class="form-label">Live Feed Event Types (Select Multiple)</label>

<select class="form-select" id="liveFeedTypes" multiple required>

<option value="academic">Academic</option>

<option value="sports">Sports</option>

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

<option value="meeting">Meeting</option>

<option value="competition">Competition</option>

<option value="workshop">Workshop</option>

<option value="celebration">Celebration</option>

<option value="trip">Trip/Excursion</option>

</select>

<div class="form-text">Events of these types will appear in the Live Feed.</div>

</div>

<div class="col-md-6">

<label for="liveFeedPriority" class="form-label">Minimum Live Feed Priority</label>

<select class="form-select" id="liveFeedPriority" required>

<option value="urgent">Urgent</option>

<option value="high" selected>High</option>

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

<option value="low">Low</option>

</select>

<div class="form-text">Only events at this priority level or higher will be monitored.</div>

</div>

<div class="col-12 mt-4">

<button type="submit" class="btn btn-dark btn-lg w-100" id="saveConfigBtn"><i class="bi bi-save"></i> Save Live Feed Configuration</button>

</div>

</form>

</div>

</div>

</div>

</div>

</div>

<!-- Bootstrap Bundle with Popper -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>

<script>

const API\_BASE = '';

const MOCK\_JWT\_TOKEN = 'mock.jwt.token.for.auth';

let mockEvents = [

{ id: 'EVT-123', title: 'Annual Sports Day', type: 'sports', startDate: '2025-10-20', endDate: '2025-10-20', startTime: '09:00', endTime: '16:00', venue: 'Main Ground', status: 'approved', priority: 'high', organizer: 'Mr. Sharma' },

{ id: 'EVT-124', title: 'PTA Meeting', type: 'meeting', startDate: '2025-11-05', endDate: '2025-11-05', startTime: '17:00', endTime: '18:00', venue: 'Auditorium', status: 'planning', priority: 'medium', organizer: 'Principal' },

{ id: 'EVT-125', title: 'Science Fair', type: 'academic', startDate: '2025-12-01', endDate: '2025-12-02', startTime: '10:00', endTime: '15:00', venue: 'Science Block', status: 'approved', priority: 'medium', organizer: 'Ms. Kapoor' },

];

let currentConfig = { liveFeedTypes: ['sports', 'meeting'], liveFeedPriority: 'high' };

/\*\* Helper function to show a temporary notification. \*/

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

const area = document.getElementById('notification-area');

area.innerHTML = `

<div class="alert alert-${type} alert-dismissible fade show" role="alert">

${message}

<button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>

</div>

`;

area.style.display = 'block';

setTimeout(() => { area.style.display = 'none'; }, 7000);

}

/\*\* Core Fetch Function with JWT Auth Header. \*/

async function apiFetch(endpoint, method = 'GET', body = null) {

try {

const options = {

method: method,

headers: {

'Content-Type': 'application/json',

'Authorization': `Bearer ${MOCK\_JWT\_TOKEN}`

},

};

if (body && method !== 'GET') {

options.body = JSON.stringify(body);

}

const response = await fetch(`${API\_BASE}${endpoint}`, options);

const data = await response.json();

if (!response.ok) {

throw new Error(data.error || `API error (${response.status})`);

}

return data;

} catch (error) {

console.error('API Fetch Error:', endpoint, error);

showNotification(error.message || 'Network error. Check console for details.', 'danger');

throw error;

}

}

const getStatusBadge = (status) => {

switch(status) {

case 'approved': return 'bg-success';

case 'planning': return 'bg-warning text-dark';

case 'cancelled': return 'bg-danger';

default: return 'bg-secondary';

}

};

const getPriorityBadge = (priority) => {

switch(priority) {

case 'urgent': return 'bg-danger';

case 'high': return 'bg-warning text-dark';

case 'medium': return 'bg-info';

default: return 'bg-light text-dark';

}

};

// --- 1. EVENT CREATION & CALENDAR LOGIC ---

async function fetchEventList() {

const tableBody = document.getElementById('eventListBody');

tableBody.innerHTML = '<tr><td colspan="5" class="text-center"><div class="spinner-border spinner-border-sm"></div> Loading events...</td></tr>';

try {

// In production, pass search/filter query params here

const data = await apiFetch('/');

tableBody.innerHTML = '';

if (data.events.length === 0) {

tableBody.innerHTML = '<tr><td colspan="5" class="text-center">No upcoming events scheduled.</td></tr>';

return;

}

data.events.forEach(event => {

const row = tableBody.insertRow();

row.innerHTML = `

<td>${event.title} <span class="badge ${getPriorityBadge(event.priority)} event-badge">${event.priority.toUpperCase()}</span></td>

<td>${event.venue}</td>

<td>${new Date(event.start\_date).toLocaleDateString()} - ${new Date(event.end\_date).toLocaleDateString()}</td>

<td>${event.start\_time} - ${event.end\_time}</td>

<td><span class="badge ${getStatusBadge(event.status)}">${event.status.toUpperCase()}</span></td>

`;

});

mockEvents = data.events; // Update mock data for live feed simulation

updateLiveFeed();

} catch (error) {

tableBody.innerHTML = '<tr><td colspan="5" class="text-center text-danger">Failed to load events.</td></tr>';

}

}

document.getElementById('eventForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const btn = document.getElementById('createEventBtn');

const payload = {

title: form.eventTitle.value,

type: form.eventType.value,

startDate: form.startDate.value,

endDate: form.endDate.value,

startTime: form.startTime.value,

endTime: form.endTime.value,

venue: form.venue.value,

organizer: form.organizer.value,

allocatedBudget: form.allocatedBudget.value ? parseFloat(form.allocatedBudget.value) : 0,

description: form.description.value,

priority: form.priority.value,

targetAudience: ['students', 'parents'] // Mock target audience

};

btn.disabled = true;

btn.innerHTML = '<span class="spinner-border spinner-border-sm"></span> Creating...';

try {

const data = await apiFetch('/', 'POST', payload);

showNotification(data.message, 'success');

form.reset();

fetchEventList();

} catch (error) {

// Handled by apiFetch (Will catch 409 conflict error)

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-check-circle"></i> Create Event (Planning)';

}

});

// --- 2. WORKFLOW LOGIC ---

document.getElementById('approvalForm').addEventListener('submit', async (e) => {

e.preventDefault();

const eventId = document.getElementById('approveEventId').value;

const btn = document.getElementById('approveBtn');

btn.disabled = true;

btn.textContent = 'Approving...';

try {

const data = await apiFetch(`/${eventId}/approve`, 'POST', {});

showNotification(data.message, 'success');

fetchEventList();

} catch (error) {

// Handled by apiFetch (Will catch 403 authorization error)

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-patch-check-fill"></i> Approve Event';

}

});

document.getElementById('notifyForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const btn = document.getElementById('notifyBtn');

const recipientsArray = Array.from(form.notifyRecipients.selectedOptions).map(opt => opt.value);

const payload = {

message: form.notifyMessage.value,

recipients: recipientsArray,

method: form.notifyMethod.value

};

const eventId = form.notifyEventId.value;

btn.disabled = true;

btn.innerHTML = '<span class="spinner-border spinner-border-sm"></span> Sending...';

try {

const data = await apiFetch(`/${eventId}/notify`, 'POST', payload);

showNotification(data.message, 'success');

form.reset();

} catch (error) {

// Handled by apiFetch

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-send-fill"></i> Send Notification';

}

});

// --- 3. LIVE EVENT FEED LOGIC ---

const priorityOrder = { 'low': 1, 'medium': 2, 'high': 3, 'urgent': 4 };

function updateLiveFeed() {

const feedDiv = document.getElementById('liveFeedBody');

const { liveFeedTypes, liveFeedPriority } = currentConfig;

if (!liveFeedTypes || liveFeedTypes.length === 0) {

feedDiv.innerHTML = '<p class="text-warning text-center">Live feed disabled: No event types are configured for monitoring.</p>';

return;

}

const minPriorityValue = priorityOrder[liveFeedPriority] || 0;

const liveEvents = mockEvents

.filter(event => new Date(event.startDate) >= new Date(new Date().setHours(0, 0, 0, 0))) // Filter for today or future events

.filter(event => liveFeedTypes.includes(event.type)) // Filter by configured type

.filter(event => priorityOrder[event.priority] >= minPriorityValue) // Filter by configured minimum priority

.sort((a, b) => priorityOrder[b.priority] - priorityOrder[a.priority]); // Sort by urgency

if (liveEvents.length === 0) {

feedDiv.innerHTML = '<p class="text-muted text-center">No events currently match the live monitoring filters.</p>';

return;

}

let html = '<ul class="list-group">';

liveEvents.forEach(event => {

html += `

<li class="list-group-item d-flex justify-content-between align-items-center list-group-item-${event.priority === 'urgent' ? 'danger' : (event.priority === 'high' ? 'warning' : 'light')}">

<div>

<strong>${event.title}</strong> (${event.organizer})

<span class="badge ${getPriorityBadge(event.priority)}">${event.priority.toUpperCase()}</span>

<span class="badge bg-secondary">${event.venue}</span>

</div>

<small class="text-muted">${new Date(event.startDate).toLocaleDateString()} @ ${event.startTime}</small>

</li>

`;

});

html += '</ul>';

feedDiv.innerHTML = html;

}

// --- 4. CONFIGURATION LOGIC ---

async function fetchConfig() {

try {

const data = await apiFetch('/config');

const settings = data.settings;

// Set global configuration variable

currentConfig = {

liveFeedTypes: settings.live\_feed\_types || [],

liveFeedPriority: settings.live\_feed\_priority || 'medium'

};

// Populate form fields

const selectTypes = document.getElementById('liveFeedTypes');

Array.from(selectTypes.options).forEach(option => {

option.selected = currentConfig.liveFeedTypes.includes(option.value);

});

document.getElementById('liveFeedPriority').value = currentConfig.liveFeedPriority;

// Update the live feed immediately after fetching config

updateLiveFeed();

} catch (error) {

showNotification('Could not fetch configuration. Using default Live Feed settings.', 'warning');

}

}

document.getElementById('configForm').addEventListener('submit', async (e) => {

e.preventDefault();

const form = e.target;

const btn = document.getElementById('saveConfigBtn');

const liveFeedTypes = Array.from(form.liveFeedTypes.selectedOptions).map(opt => opt.value);

const payload = {

liveFeedTypes: liveFeedTypes,

liveFeedPriority: form.liveFeedPriority.value

};

btn.disabled = true;

btn.innerHTML = '<span class="spinner-border spinner-border-sm"></span> Saving...';

try {

const data = await apiFetch('/config', 'PUT', payload);

showNotification(data.message, 'success');

// Update local config and refresh feed

currentConfig = data.newSettings;

updateLiveFeed();

} catch (error) {

// Handled by apiFetch

} finally {

btn.disabled = false;

btn.innerHTML = '<i class="bi bi-save"></i> Save Live Feed Configuration';

}

});

// --- Initialization ---

document.addEventListener('DOMContentLoaded', () => {

fetchConfig();

fetchEventList();

});

</script>

</body>

</html>

20. client/dist/modules/student\_behaviour.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>🌟 Student Behavior Tracker</title>

<!-- Bootstrap 5 CDN -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css">

<style>

body { background-color: #f4f7f6; font-family: 'Inter', sans-serif; }

.container-fluid { max-width: 1300px; margin-top: 20px; }

.card-header { background-color: #3498db; color: white; font-weight: bold; }

.section-card { margin-bottom: 25px; box-shadow: 0 4px 12px rgba(0,0,0,0.1); border-radius: 0.75rem; }

.alert-fixed { position: fixed; top: 20px; right: 20px; z-index: 1050; min-width: 350px; border-radius: 0.5rem; }

.level-badge { font-weight: bold; padding: 5px 10px; border-radius: 15px; }

.positive-btn { background-color: #2ecc71; color: white; border: none; }

.negative-btn { background-color: #e74c3c; color: white; border: none; }

.log-item.positive { border-left: 5px solid #2ecc71; }

.log-item.negative { border-left: 5px solid #e74c3c; }

.points-dial { font-size: 2.5rem; font-weight: 700; }

.config-category-card { cursor: pointer; transition: transform 0.2s; }

.config-category-card:hover { transform: translateY(-3px); box-shadow: 0 6px 15px rgba(0,0,0,0.15); }

</style>

</head>

<body>

<div class="container-fluid">

<h1 class="text-center mb-4 text-primary"><i class="bi bi-person-badge-fill"></i> Student Behavior & Intervention System</h1>

<!-- Notification Area -->

<div id="notification-area" class="alert-fixed" style="display: none;"></div>

<!-- Tab Navigation -->

<ul class="nav nav-tabs" id="behaviorTab" role="tablist">

<li class="nav-item" role="presentation">

<button class="nav-link active" id="dashboard-tab" data-bs-toggle="tab" data-bs-target="#dashboard" type="button" role="tab"><i class="bi bi-speedometer2"></i> Summary & Logger</button>

</li>

<li class="nav-item" role="presentation">

<button class="nav-link" id="detail-tab" data-bs-toggle="tab" data-bs-target="#detail" type="button" role="tab"><i class="bi bi-person-lines-fill"></i> Student Profile & Interventions</button>

</li>

<li class="nav-item" role="presentation">

<button class="nav-link" id="rewards-tab" data-bs-toggle="tab" data-bs-target="#rewards" type="button" role="tab"><i class="bi bi-trophy-fill"></i> Rewards & Redemption</button>

</li>

<li class="nav-item" role="presentation">

<button class="nav-link" id="analytics-tab" data-bs-toggle="tab" data-bs-target="#analytics" type="button" role="tab"><i class="bi bi-bar-chart-line-fill"></i> Analytics & Reports</button>

</li>

<li class="nav-item" role="presentation">

<button class="nav-link" id="config-tab" data-bs-toggle="tab" data-bs-target="#config" type="button" role="tab"><i class="bi bi-gear-fill"></i> Configuration</button>

</li>

</ul>

<!-- Tab Content -->

<div class="tab-content pt-3" id="behaviorTabContent">

<!-- 1. Dashboard & Logger -->

<div class="tab-pane fade show active" id="dashboard" role="tabpanel">

<div class="row">

<!-- Incident Logger (POST /log) -->

<div class="col-lg-4">

<div class="card section-card">

<div class="card-header bg-success">Log New Incident</div>

<div class="card-body">

<form id="incidentLoggerForm">

<div class="mb-3">

<label for="logStudentId" class="form-label">Student ID</label>

<select class="form-select" id="logStudentId" required>

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

<!-- Dynamically populated -->

</select>

</div>

<div class="mb-3">

<label for="logCategory" class="form-label">Category</label>

<select class="form-select" id="logCategory" required>

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

<!-- Dynamically populated from /categories -->

</select>

</div>

<div class="mb-3">

<label for="logTitle" class="form-label">Title</label>

<input type="text" class="form-control" id="logTitle" required>

</div>

<div class="mb-3">

<label for="logDescription" class="form-label">Details/Description</label>

<textarea class="form-control" id="logDescription" rows="2" required></textarea>

</div>

<div class="row mb-3">

<div class="col-md-6">

<label for="logPoints" class="form-label">Points Change</label>

<input type="number" class="form-control" id="logPoints" placeholder="Auto-filled or custom">

</div>

<div class="col-md-6">

<label for="logSeverity" class="form-label">Severity</label>

<select class="form-select" id="logSeverity" required>

<option value="low">Low</option>

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

<option value="high">High</option>

</select>

</div>

</div>

<div class="form-check form-switch mb-3">

<input class="form-check-input" type="checkbox" id="logParentNotify">

<label class="form-check-label" for="logParentNotify">Notify Parent (Immediate Arattai/WhatsApp)</label>

</div>

<button type="submit" class="btn btn-primary w-100"><i class="bi bi-check-circle"></i> Record Incident</button>

</form>

</div>

</div>

</div>

<!-- Student Summary Table (GET /summary) -->

<div class="col-lg-8">

<div class="card section-card">

<div class="card-header bg-primary">Student Behavior Overview</div>

<div class="card-body">

<div class="row mb-3">

<div class="col-md-4">

<select class="form-select" id="summaryFilterClass">

<option value="">Filter by Class</option>

<option value="Class 1-A">Class 1-A</option>

<option value="Class 2-A">Class 2-A</option>

<!-- Add more classes dynamically -->

</select>

</div>

<div class="col-md-4">

<select class="form-select" id="summaryFilterGrade">

<option value="">Filter by Grade Level</option>

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

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

<!-- Add more grades dynamically -->

</select>

</div>

<div class="col-md-4">

<button class="btn btn-info w-100" id="refreshSummaryBtn"><i class="bi bi-arrow-clockwise"></i> Refresh</button>

</div>

</div>

<div class="table-responsive">

<table class="table table-striped table-sm align-middle">

<thead>

<tr>

<th>Student</th>

<th>Points</th>

<th>Level</th>

<th>Trend</th>

<th>Logs (Pos/Neg)</th>

<th>Action</th>

</tr>

</thead>

<tbody id="summaryTableBody">

<tr><td colspan="6" class="text-center">Loading student data...</td></tr>

</tbody>

</table>

</div>

</div>

</div>

</div>

</div>

</div>

<!-- 2. Student Detail & Interventions -->

<div class="tab-pane fade" id="detail" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-secondary">Student Profile: <span id="detailStudentName">Select a Student</span></div>

<div class="card-body">

<div class="row">

<div class="col-lg-4">

<!-- Student Selector -->

<select class="form-select mb-3" id="studentDetailSelector">

<option value="">Select Student for Detail View</option>

</select>

<!-- Quick Stats -->

<div class="p-3 border rounded mb-4 text-center" id="detailQuickStats">

<p class="text-muted mb-1">Current Behavior Points</p>

<div id="detailPointsDial" class="points-dial text-primary">--</div>

<div id="detailBehaviorLevel" class="level-badge bg-secondary text-white">--</div>

<p class="mt-2 mb-0">Trend: <span id="detailTrend">--</span></p>

</div>

<!-- Intervention Scheduler (POST /intervention) -->

<h5>Schedule Follow-Up</h5>

<form id="interventionForm">

<input type="hidden" id="interventionStudentId" required>

<div class="mb-3">

<label for="interventionType" class="form-label">Intervention Type</label>

<select class="form-select" id="interventionType" required>

<option value="counseling\_session">Counseling Session</option>

<option value="parent\_meeting">Parent Meeting (P/T Conference)</option>

<option value="detention">Detention/Time-out</option>

<option value="reward\_privilege">Positive Reinforcement</option>

</select>

</div>

<div class="mb-3">

<label for="interventionScheduledDate" class="form-label">Scheduled Date</label>

<input type="date" class="form-control" id="interventionScheduledDate" required>

</div>

<div class="mb-3">

<label for="interventionAssignedTo" class="form-label">Assigned To (Counselor/Teacher ID)</label>

<input type="text" class="form-control" id="interventionAssignedTo" placeholder="e.g., counselor\_001" required>

</div>

<button type="submit" class="btn btn-warning w-100"><i class="bi bi-calendar-plus"></i> Schedule Intervention</button>

</form>

</div>

<div class="col-lg-8">

<!-- Intervention History (GET /interventions) -->

<h5>Intervention History & Status</h5>

<div id="interventionsList" class="list-group mb-4">

<!-- Dynamic list of interventions -->

<p class="text-muted text-center">No scheduled interventions found.</p>

</div>

<!-- Behavior Log History (GET /student/:studentId) -->

<h5>Recent Behavior Logs</h5>

<div id="studentLogHistory" class="list-group">

<!-- Dynamic list of logs -->

<p class="text-muted text-center">No recent logs.</p>

</div>

</div>

</div>

</div>

</div>

</div>

<!-- 3. Rewards & Redemption -->

<div class="tab-pane fade" id="rewards" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-info">Rewards System & Redemption</div>

<div class="card-body">

<div class="row">

<!-- Student Context -->

<div class="col-md-4">

<h5>Redemption Status</h5>

<select class="form-select mb-3" id="rewardStudentSelector">

<option value="">Select Student to View Rewards</option>

</select>

<div class="p-3 border rounded bg-light">

<p class="mb-1">Student Points: <strong id="rewardCurrentPoints">--</strong></p>

<p class="mb-1">Behavior Level: <span id="rewardBehaviorLevel" class="level-badge bg-secondary text-white">--</span></p>

</div>

</div>

<!-- Available Rewards (GET /rewards?studentId) -->

<div class="col-md-8">

<h5>Available Rewards for Redemption</h5>

<div id="availableRewardsList" class="row">

<!-- Dynamic list of available rewards -->

<p class="text-muted text-center">Select a student or check configuration.</p>

</div>

</div>

</div>

<hr class="my-4">

<!-- Rewards Management (GET /rewards) -->

<h5>All Defined Rewards (Configuration View)</h5>

<div class="table-responsive">

<table class="table table-striped table-sm">

<thead>

<tr>

<th>Reward</th>

<th>Type</th>

<th>Points Required</th>

<th>Description</th>

</tr>

</thead>

<tbody id="allRewardsTableBody">

<tr><td colspan="4" class="text-center">Loading rewards system...</td></tr>

</tbody>

</table>

</div>

</div>

</div>

</div>

<!-- 4. Analytics & Reports -->

<div class="tab-pane fade" id="analytics" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-dark">Behavior Analytics & Oversight</div>

<div class="card-body">

<div class="row mb-4">

<div class="col-md-4">

<label for="analyticsPeriod" class="form-label">Reporting Period</label>

<select class="form-select" id="analyticsPeriod">

<option value="week" selected>Last 7 Days</option>

<option value="month">Last 30 Days</option>

<option value="quarter">Last 90 Days</option>

</select>

</div>

<div class="col-md-4">

<button class="btn btn-dark mt-4 w-100" id="runAnalyticsBtn"><i class="bi bi-graph-up"></i> Run Analysis</button>

</div>

</div>

<!-- Overview Stats (GET /analytics) -->

<div class="row text-center mb-4" id="analyticsOverview">

<div class="col-md-3">

<div class="p-3 border rounded bg-light">

<h4 id="statTotalIncidents">--</h4><p class="text-muted">Total Incidents</p>

</div>

</div>

<div class="col-md-3">

<div class="p-3 border rounded bg-light">

<h4 id="statAvgPoints">--</h4><p class="text-muted">Avg Student Points</p>

</div>

</div>

<div class="col-md-3">

<div class="p-3 border rounded bg-light">

<h4 id="statPositiveRatio">--</h4><p class="text-muted">Positive Incidents</p>

</div>

</div>

<div class="col-md-3">

<div class="p-3 border rounded bg-light">

<h4 id="statImprovementRate">--</h4><p class="text-muted">Improvement Rate</p>

</div>

</div>

</div>

<div class="row">

<!-- Needs Attention -->

<div class="col-md-6">

<h5>Students Needing Attention</h5>

<div class="list-group" id="studentsNeedingAttentionList"></div>

</div>

<!-- Top Performers -->

<div class="col-md-6">

<h5>Top Behavior Performers</h5>

<div class="list-group" id="topPerformersList"></div>

</div>

</div>

</div>

</div>

</div>

<!-- 5. Configuration Page (Points System, Categories, Scale) -->

<div class="tab-pane fade" id="config" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-danger"><i class="bi bi-screwdriver"></i> System Configuration (Admin Only)</div>

<div class="card-body">

<!-- Points System Setup (Mock PUT/POST) -->

<h5>1. Core Points System & Reset</h5>

<form id="pointsSystemConfigForm" class="row g-3 p-3 border rounded mb-4">

<div class="col-md-3">

<label for="configStartingPoints" class="form-label">Starting Points</label>

<input type="number" class="form-control" id="configStartingPoints" required min="0" max="100">

</div>

<div class="col-md-3">

<label for="configMaxPoints" class="form-label">Max Points</label>

<input type="number" class="form-control" id="configMaxPoints" required min="1" max="100">

</div>

<div class="col-md-3">

<label for="configResetFrequency" class="form-label">Reset Frequency</label>

<select class="form-select" id="configResetFrequency" required>

<option value="monthly">Monthly</option>

<option value="quarterly">Quarterly</option>

<option value="annually">Annually</option>

</select>

</div>

<div class="col-md-3 d-flex align-items-end">

<button type="submit" class="btn btn-sm btn-success w-100">Save Points System</button>

</div>

</form>

<!-- Behavior Categories Management (Mock CRUD) -->

<h5>2. Behavior Categories (Positive/Negative)</h5>

<div id="categoryCardsContainer" class="row g-3 mb-4">

<!-- Dynamic category cards from backend/mock -->

</div>

<!-- Add/Edit Category Modal -->

<div class="modal fade" id="categoryModal" tabindex="-1">

<div class="modal-dialog">

<div class="modal-content">

<div class="modal-header">

<h5 class="modal-title" id="categoryModalTitle">Add New Category</h5>

<button type="button" class="btn-close" data-bs-dismiss="modal"></button>

</div>

<form id="categoryForm">

<div class="modal-body">

<input type="hidden" id="categoryId">

<div class="mb-3">

<label for="categoryName" class="form-label">Name</label>

<input type="text" class="form-control" id="categoryName" required>

</div>

<div class="mb-3">

<label for="categoryType" class="form-label">Type</label>

<select class="form-select" id="categoryType" required>

<option value="positive">Positive</option>

<option value="negative">Negative</option>

</select>

</div>

<div class="mb-3">

<label for="categoryPoints" class="form-label">Default Points Change</label>

<input type="number" class="form-control" id="categoryPoints" required>

</div>

<div class="mb-3">

<label for="categoryIcon" class="form-label">Icon (Emoji)</label>

<input type="text" class="form-control" id="categoryIcon" placeholder="e.g., 🎓, 🚫">

</div>

</div>

<div class="modal-footer">

<button type="button" class="btn btn-secondary" data-bs-dismiss="modal">Close</button>

<button type="submit" class="btn btn-primary" id="saveCategoryBtn">Save Changes</button>

</div>

</form>

</div>

</div>

</div>

<button class="btn btn-sm btn-primary" data-bs-toggle="modal" data-bs-target="#categoryModal" id="addNewCategoryBtn"><i class="bi bi-plus-circle"></i> Add New Category Rule</button>

</div>

</div>

</div>

</div>

</div>

<!-- Bootstrap Bundle with Popper -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>

<script>

const API\_BASE = '';

let studentList = []; // Global store for quick lookups

/\*\* Helper function to show a temporary notification. \*/

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

const area = document.getElementById('notification-area');

area.innerHTML = `

<div class="alert alert-${type} alert-dismissible fade show" role="alert">

${message}

<button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>

</div>

`;

area.style.display = 'block';

setTimeout(() => {

area.style.display = 'none';

}, 7000);

}

// --- Core Fetch Function ---

async function apiFetch(endpoint, method = 'GET', body = null) {

try {

const options = {

method: method,

headers: { 'Content-Type': 'application/json' },

};

if (body && method !== 'GET') {

options.body = JSON.stringify(body);

}

const response = await fetch(`${API\_BASE}${endpoint}`, options);

const data = await response.json();

if (!response.ok) {

throw new Error(data.error || `API error (${response.status})`);

}

return data;

} catch (error) {

console.error('API Fetch Error:', endpoint, error);

showNotification(error.message || 'Network error.', 'danger');

throw error;

}

}

// --- Data Population Helpers ---

function populateStudentSelectors(students) {

const selectors = ['logStudentId', 'studentDetailSelector', 'rewardStudentSelector'];

studentList = students; // Update global store

selectors.forEach(id => {

const select = document.getElementById(id);

const currentValue = select.value;

select.innerHTML = '<option value="">Select Student</option>'; // Reset

students.forEach(student => {

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

option.value = student.id;

option.textContent = `${student.name} (${student.rollNumber})`;

select.appendChild(option);

});

// Restore previous selection if possible

if (currentValue && students.some(s => s.id === currentValue)) {

select.value = currentValue;

}

});

}

async function populateCategories() {

try {

const categories = await apiFetch('/categories');

const select = document.getElementById('logCategory');

select.innerHTML = '<option value="">Select Category</option>';

categories.forEach(cat => {

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

option.value = cat.id;

option.textContent = `${cat.icon} ${cat.name} (${cat.defaultPoints > 0 ? '+' : ''}${cat.defaultPoints})`;

option.setAttribute('data-points', cat.defaultPoints);

option.setAttribute('data-type', cat.type);

select.appendChild(option);

});

// Update config cards

renderConfigCategories(categories);

} catch (error) {

console.error('Failed to load categories.');

}

}

// --- 1. Dashboard & Logger Logic ---

async function fetchSummary() {

const classId = document.getElementById('summaryFilterClass').value;

const grade = document.getElementById('summaryFilterGrade').value;

const tableBody = document.getElementById('summaryTableBody');

tableBody.innerHTML = '<tr><td colspan="6" class="text-center"><div class="spinner-border spinner-border-sm"></div> Loading...</td></tr>';

try {

const data = await apiFetch(`/summary?classId=${classId}&grade=${grade}`);

const students = data;

tableBody.innerHTML = '';

if (students.length === 0) {

tableBody.innerHTML = '<tr><td colspan="6" class="text-center">No students found for the current filter.</td></tr>';

return;

}

populateStudentSelectors(students);

students.forEach(student => {

const level = student.behaviorLevel;

const trendIcon = student.behaviorTrend === 'improving' ? 'bi-graph-up text-success' :

student.behaviorTrend === 'declining' ? 'bi-graph-down text-danger' : 'bi-dash-lg text-secondary';

const row = tableBody.insertRow();

row.innerHTML = `

<td><strong>${student.name}</strong><br><small class="text-muted">${student.class}</small></td>

<td class="fw-bold">${student.currentPoints}</td>

<td><span class="level-badge text-white" style="background-color: ${level.color};">${level.level.toUpperCase()}</span></td>

<td><i class="bi ${trendIcon}"></i> ${student.behaviorTrend}</td>

<td>${student.positiveIncidents} / ${student.negativeIncidents}</td>

<td><button class="btn btn-sm btn-outline-primary view-detail-btn" data-student-id="${student.id}"><i class="bi bi-eye"></i> Detail</button></td>

`;

});

} catch (error) {

tableBody.innerHTML = '<tr><td colspan="6" class="text-center text-danger">Failed to load summary.</td></tr>';

}

}

// Handle Log Incident Form Submission

document.getElementById('incidentLoggerForm').addEventListener('submit', async (e) => {

e.preventDefault();

const btn = e.submitter;

btn.disabled = true;

const selectedOption = document.getElementById('logCategory').options[document.getElementById('logCategory').selectedIndex];

const defaultPoints = parseInt(selectedOption.getAttribute('data-points'));

const payload = {

studentId: document.getElementById('logStudentId').value,

type: selectedOption.getAttribute('data-type'),

category: document.getElementById('logCategory').value,

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

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

points: parseInt(document.getElementById('logPoints').value) || defaultPoints,

recordedBy: 'teacher\_001', // Mock user ID

severity: document.getElementById('logSeverity').value,

parentNotified: document.getElementById('logParentNotify').checked,

followUpRequired: document.getElementById('logSeverity').value === 'high'

};

try {

const data = await apiFetch('/log', 'POST', payload);

showNotification(`Incident logged for ${studentList.find(s => s.id === payload.studentId).name}. Points: ${data.updatedPoints}`, 'success');

e.target.reset();

fetchSummary();

// Automatically switch to the detail view

document.getElementById('studentDetailSelector').value = payload.studentId;

fetchStudentDetail(payload.studentId);

new bootstrap.Tab(document.getElementById('detail-tab')).show();

} catch {

// Error handled by apiFetch

} finally {

btn.disabled = false;

}

});

// Auto-fill points based on category selection

document.getElementById('logCategory').addEventListener('change', () => {

const selectedOption = document.getElementById('logCategory').options[document.getElementById('logCategory').selectedIndex];

const pointsInput = document.getElementById('logPoints');

if (selectedOption.value) {

pointsInput.value = selectedOption.getAttribute('data-points');

} else {

pointsInput.value = '';

}

});

document.getElementById('refreshSummaryBtn').addEventListener('click', fetchSummary);

// --- 2. Student Detail & Interventions Logic ---

// Handle View Detail button from Summary

document.getElementById('summaryTableBody').addEventListener('click', (e) => {

if (e.target.classList.contains('view-detail-btn')) {

const studentId = e.target.getAttribute('data-student-id');

document.getElementById('studentDetailSelector').value = studentId;

fetchStudentDetail(studentId);

new bootstrap.Tab(document.getElementById('detail-tab')).show();

}

});

document.getElementById('studentDetailSelector').addEventListener('change', (e) => {

const studentId = e.target.value;

if (studentId) {

fetchStudentDetail(studentId);

}

});

async function fetchStudentDetail(studentId) {

if (!studentId) return;

try {

const data = await apiFetch(`/student/${studentId}`);

const student = data.student;

const stats = data.statistics;

document.getElementById('detailStudentName').textContent = `${student.name} (${student.class})`;

document.getElementById('detailPointsDial').textContent = student.currentPoints;

const levelBadge = document.getElementById('detailBehaviorLevel');

levelBadge.textContent = stats.behaviorLevel.level.toUpperCase();

levelBadge.style.backgroundColor = stats.behaviorLevel.color;

document.getElementById('detailTrend').textContent = student.behaviorTrend;

document.getElementById('interventionStudentId').value = studentId;

renderLogHistory(data.logs);

renderInterventions(data.interventions);

// Also trigger rewards update for this student

fetchRewards(studentId);

} catch (error) {

document.getElementById('detailStudentName').textContent = 'Error Loading Data';

console.error('Failed to load student detail:', error);

}

}

function renderLogHistory(logs) {

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

container.innerHTML = '';

if (logs.length === 0) {

container.innerHTML = '<p class="text-muted text-center">No logs found for this student.</p>';

return;

}

logs.forEach(log => {

const typeClass = log.type === 'positive' ? 'success' : 'danger';

const pointsSign = log.points > 0 ? '+' : '';

container.innerHTML += `

<div class="list-group-item log-item ${log.type} d-flex justify-content-between align-items-center">

<div>

<h6 class="mb-1">${log.title} (${log.category})</h6>

<small class="text-muted">${log.description.substring(0, 50)}...</small>

</div>

<span class="badge bg-${typeClass} fs-6">${pointsSign}${log.points}</span>

</div>

`;

});

}

// Handle Intervention Submission

document.getElementById('interventionForm').addEventListener('submit', async (e) => {

e.preventDefault();

const btn = e.submitter;

btn.disabled = true;

const studentId = document.getElementById('interventionStudentId').value;

const studentName = studentList.find(s => s.id === studentId)?.name;

const payload = {

studentId: studentId,

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

title: document.getElementById('interventionType').options[document.getElementById('interventionType').selectedIndex].textContent,

description: `Scheduled intervention for behavior review.`, // Simplified for mock

scheduledDate: document.getElementById('interventionScheduledDate').value,

assignedTo: document.getElementById('interventionAssignedTo').value,

parentInvolved: true // Default true for formality

};

try {

await apiFetch('/intervention', 'POST', payload);

showNotification(`Intervention scheduled for ${studentName}.`, 'success');

fetchStudentDetail(studentId); // Refresh interventions list

e.target.reset();

} catch {

// Error handled by apiFetch

} finally {

btn.disabled = false;

}

});

function renderInterventions(interventions) {

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

container.innerHTML = '';

if (interventions.length === 0) {

container.innerHTML = '<p class="text-muted text-center">No scheduled interventions found.</p>';

return;

}

interventions.forEach(i => {

const statusClass = i.status === 'scheduled' ? 'warning' : i.status === 'completed' ? 'success' : 'info';

container.innerHTML += `

<div class="list-group-item d-flex justify-content-between align-items-center">

<div>

<h6 class="mb-1">${i.title} (${i.type.replace('\_', ' ')})</h6>

<small class="text-muted">Due: ${i.scheduledDate} | Assigned: ${i.assignedTo}</small>

</div>

<span class="badge bg-${statusClass}">${i.status.toUpperCase()}</span>

</div>

`;

});

}

// --- 3. Rewards & Redemption Logic ---

document.getElementById('rewardStudentSelector').addEventListener('change', (e) => {

const studentId = e.target.value;

if (studentId) {

fetchRewards(studentId);

} else {

document.getElementById('rewardCurrentPoints').textContent = '--';

document.getElementById('rewardBehaviorLevel').textContent = '--';

document.getElementById('rewardBehaviorLevel').style.backgroundColor = '#6c757d';

document.getElementById('availableRewardsList').innerHTML = '<p class="text-muted text-center">Select a student to check available rewards.</p>';

}

});

async function fetchRewards(studentId = null) {

const listContainer = document.getElementById('availableRewardsList');

const tableBody = document.getElementById('allRewardsTableBody');

try {

// Fetch all rewards regardless of student for the table view

const allRewards = await apiFetch('/rewards');

tableBody.innerHTML = '';

allRewards.allRewards.forEach(reward => {

tableBody.innerHTML += `

<tr>

<td>${reward.icon || ''} ${reward.name}</td>

<td><span class="badge bg-secondary">${reward.type}</span></td>

<td>${reward.pointsRequired}</td>

<td>${reward.description}</td>

</tr>

`;

});

if (studentId) {

// Fetch rewards specific to the selected student

const studentRewardsData = await apiFetch(`/rewards?studentId=${studentId}`);

const student = studentList.find(s => s.id === studentId);

document.getElementById('rewardCurrentPoints').textContent = studentRewardsData.currentPoints;

const levelBadge = document.getElementById('rewardBehaviorLevel');

levelBadge.textContent = studentRewardsData.behaviorLevel.level.toUpperCase();

levelBadge.style.backgroundColor = studentRewardsData.behaviorLevel.color;

listContainer.innerHTML = '';

if (studentRewardsData.availableRewards.length === 0) {

listContainer.innerHTML = '<p class="text-warning">Insufficient points for any available rewards.</p>';

}

studentRewardsData.availableRewards.forEach(reward => {

listContainer.innerHTML += `

<div class="col-md-6 mb-3">

<div class="card h-100 p-2 shadow-sm">

<div class="card-body">

<h5 class="card-title">${reward.name}</h5>

<p class="card-text text-muted">${reward.description}</p>

<p class="fw-bold text-success">${reward.pointsRequired} Points</p>

<button class="btn btn-sm btn-info redeem-reward-btn" data-student-id="${studentId}" data-reward-id="${reward.id}">Redeem</button>

</div>

</div>

</div>

`;

});

}

} catch (error) {

console.error('Failed to fetch rewards:', error);

}

}

// Handle Redeem Reward

document.getElementById('availableRewardsList').addEventListener('click', async (e) => {

if (e.target.classList.contains('redeem-reward-btn')) {

const studentId = e.target.getAttribute('data-student-id');

const rewardId = e.target.getAttribute('data-reward-id');

const studentName = studentList.find(s => s.id === studentId)?.name;

try {

await apiFetch('/reward/redeem', 'POST', { studentId, rewardId });

showNotification(`${studentName} successfully redeemed reward!`, 'success');

fetchRewards(studentId); // Refresh points and available rewards

fetchSummary(); // Update summary table

} catch (error) {

console.error('Redemption failed:', error);

}

}

});

// --- 4. Analytics Logic ---

document.getElementById('runAnalyticsBtn').addEventListener('click', async () => {

const period = document.getElementById('analyticsPeriod').value;

try {

const data = await apiFetch(`/analytics?period=${period}`);

const overview = data.overview;

const trends = data.trends;

document.getElementById('statTotalIncidents').textContent = overview.totalIncidents;

document.getElementById('statAvgPoints').textContent = overview.averagePoints;

document.getElementById('statPositiveRatio').textContent = `${overview.positiveIncidents} / ${overview.negativeIncidents}`;

document.getElementById('statImprovementRate').textContent = `${trends.improvementRate}%`;

renderStudentsNeedingAttention(data.studentsNeedingAttention);

renderTopPerformers(data.topPerformers);

showNotification(`Analytics updated for the last ${period}.`, 'info');

} catch {

// Error handled by apiFetch

}

});

function renderStudentsNeedingAttention(students) {

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

container.innerHTML = '';

students.forEach(s => {

const level = s.behaviorLevel;

container.innerHTML += `

<div class="list-group-item d-flex justify-content-between align-items-center">

<span>${s.name} (${s.currentPoints} pts)</span>

<span class="level-badge text-white" style="background-color: ${level.color};">${level.level.toUpperCase()}</span>

</div>

`;

});

}

function renderTopPerformers(students) {

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

container.innerHTML = '';

students.forEach(s => {

const level = s.behaviorLevel;

container.innerHTML += `

<div class="list-group-item d-flex justify-content-between align-items-center">

<span>${s.name} (${s.currentPoints} pts)</span>

<span class="level-badge text-white" style="background-color: ${level.color};">${level.level.toUpperCase()}</span>

</div>

`;

});

}

// --- 5. Configuration Logic (Simulated CRUD) ---

// This function mocks fetching and populates the config forms

async function fetchConfig() {

// NOTE: In a real system, you would fetch these from /config endpoints.

const data = {

pointsSystem: { startingPoints: 75, maxPoints: 100, resetFrequency: 'monthly' },

behaviorCategories: (await apiFetch('/categories')) // Re-use categories endpoint for data

};

// 1. Points System Form

document.getElementById('configStartingPoints').value = data.pointsSystem.startingPoints;

document.getElementById('configMaxPoints').value = data.pointsSystem.maxPoints;

document.getElementById('configResetFrequency').value = data.pointsSystem.resetFrequency;

// 2. Categories Display

renderConfigCategories(data.behaviorCategories);

}

// Mock save for Points System

document.getElementById('pointsSystemConfigForm').addEventListener('submit', (e) => {

e.preventDefault();

// In production, this would be a PUT /config

showNotification('Points System configuration saved (Simulated PUT).', 'success');

});

// Render Configuration Category Cards

function renderConfigCategories(categories) {

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

container.innerHTML = '';

categories.forEach(cat => {

const cardHtml = `

<div class="col-md-4">

<div class="card config-category-card border-${cat.type === 'positive' ? 'success' : 'danger'}" data-category-id="${cat.id}" style="border-left: 5px solid ${cat.color};">

<div class="card-body d-flex justify-content-between align-items-center">

<div>

<h5 class="card-title mb-1">${cat.icon} ${cat.name}</h5>

<p class="card-text fw-bold text-${cat.type === 'positive' ? 'success' : 'danger'}">${cat.defaultPoints > 0 ? '+' : ''}${cat.defaultPoints} Points</p>

</div>

<div>

<button class="btn btn-sm btn-outline-info edit-category-btn" data-bs-toggle="modal" data-bs-target="#categoryModal" data-category-id="${cat.id}"><i class="bi bi-pencil"></i></button>

</div>

</div>

</div>

</div>

`;

container.innerHTML += cardHtml;

});

}

// Handle Edit/Add Category Modal

const categoryModal = new bootstrap.Modal(document.getElementById('categoryModal'));

// Logic to populate modal when editing

document.getElementById('categoryCardsContainer').addEventListener('click', (e) => {

if (e.target.closest('.edit-category-btn')) {

const categoryId = e.target.closest('.edit-category-btn').getAttribute('data-category-id');

const category = behaviorData.behaviorCategories.find(c => c.id === categoryId);

document.getElementById('categoryModalTitle').textContent = `Edit: ${category.name}`;

document.getElementById('categoryId').value = category.id;

document.getElementById('categoryName').value = category.name;

document.getElementById('categoryType').value = category.type;

document.getElementById('categoryPoints').value = category.defaultPoints;

document.getElementById('categoryIcon').value = category.icon;

}

});

document.getElementById('addNewCategoryBtn').addEventListener('click', () => {

document.getElementById('categoryModalTitle').textContent = 'Add New Category';

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

document.getElementById('categoryId').value = '';

});

// Mock save for Category

document.getElementById('categoryForm').addEventListener('submit', async (e) => {

e.preventDefault();

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

const action = categoryId ? 'UPDATE' : 'CREATE';

const payload = {

id: categoryId || 'new\_cat\_' + Date.now(),

name: document.getElementById('categoryName').value,

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

defaultPoints: parseInt(document.getElementById('categoryPoints').value),

icon: document.getElementById('categoryIcon').value,

color: payload.type === 'positive' ? '#2ecc71' : '#e74c3c' // Simplified color assignment

};

// In a real system, this would be POST or PUT /categories/:id

showNotification(`Category "${payload.name}" ${action} saved (Simulated API call).`, 'success');

categoryModal.hide();

await populateCategories(); // Refresh the category list on the UI

});

// --- Initialization ---

document.addEventListener('DOMContentLoaded', () => {

fetchSummary();

populateCategories();

fetchConfig();

// Auto-run analytics on first load

document.getElementById('runAnalyticsBtn').click();

});

</script>

</body>

</html>

21. client/dist/modules/Substitution.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Intelligent Substitution Manager</title>

<!-- Tailwind CSS (Emulation) -->

<link href="https://cdn.jsdelivr.net/npm/tailwindcss@2.2.19/dist/tailwind.min.css" rel="stylesheet">

<link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css" rel="stylesheet">

<style>

body { background-color: #f8f9fa; font-family: 'Inter', sans-serif; }

.container { max-width: 1100px; margin-top: 20px; }

.card-header { background-color: #004d40; color: white; font-weight: bold; border-radius: 0.5rem 0.5rem 0 0; }

.section-card { box-shadow: 0 4px 8px rgba(0,0,0,0.05); border-radius: 0.5rem; margin-bottom: 20px; }

.tab-button.active { background-color: #004d40 !important; color: white !important; }

.time-alert { background-color: #ffcc80; color: #4e342e; border-left: 4px solid #ff9800; }

</style>

</head>

<body>

<div class="container mx-auto">

<h1 class="text-3xl font-extrabold text-center mb-6 text-gray-800">🧑‍🏫 Intelligent Substitution Management</h1>

<!-- Notification Area -->

<div id="notification-area" class="fixed top-4 right-4 z-50"></div>

<!-- Global Context (Mock) -->

<div class="bg-blue-100 p-3 rounded-lg mb-4 text-sm">

<span class="font-semibold">Context:</span> Teacher ID <span id="currentTeacherId">T-1111</span> (Simulated) | Today: <span id="currentDate"></span>

</div>

<!-- Tab Navigation -->

<div class="bg-white rounded-lg shadow-md mb-4">

<div class="flex border-b border-gray-200">

<button class="tab-button px-4 py-3 text-sm font-medium active flex-1 text-gray-700" onclick="switchTab('leave')" data-tab="leave">

<i class="fas fa-calendar-times mr-2"></i>Leave Request & Auto-Assign

</button>

<button class="tab-button px-4 py-3 text-sm font-medium flex-1 text-gray-700" onclick="switchTab('mySubs')" data-tab="mySubs">

<i class="fas fa-clipboard-list mr-2"></i>My Substitutions (To Confirm/Complete)

</button>

<button class="tab-button px-4 py-3 text-sm font-medium flex-1 text-gray-700" onclick="switchTab('config')" data-tab="config">

<i class="fas fa-cogs mr-2"></i>Configuration Hub

</button>

</div>

</div>

<!-- Tab Content -->

<div id="tabContent">

<!-- 1. Leave Request & Auto-Assign (Teacher's Primary View) -->

<div data-tab-content="leave" class="tab-pane active">

<div class="section-card bg-white">

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

Teacher Leave Request & Automatic Substitution

</div>

<div class="p-6">

<!-- CRITICAL TIME GATE ALERT -->

<div id="timeGateAlert" class="p-4 rounded mb-4 text-sm time-alert">

<i class="fas fa-clock mr-2"></i>\*\*Time-Gated Submission:\*\* For same-day leave, requests must be submitted between \*\*1:00 AM and 7:20 AM\*\* (School Admin override exempts this rule).

</div>

<form id="leaveRequestForm" onsubmit="handleLeaveRequest(event)" class="space-y-4">

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

<div>

<label class="block text-sm font-medium text-gray-700">Teacher ID (Self-Populated)</label>

<input type="text" id="leaveTeacherId" value="T-1111" disabled class="mt-1 block w-full p-2 border bg-gray-100 rounded-md">

</div>

<div>

<label class="block text-sm font-medium text-gray-700">Date of Absence</label>

<input type="date" id="leaveDate" required class="mt-1 block w-full p-2 border border-gray-300 rounded-md">

</div>

<div class="col-span-1 md:col-span-2">

<label class="block text-sm font-medium text-gray-700">Type of Leave</label>

<select id="leaveType" required class="mt-1 block w-full p-2 border border-gray-300 rounded-md">

<option value="FULL\_DAY">FULL DAY (All Periods)</option>

<option value="ON\_DUTY">ON DUTY (All Periods)</option>

<option value="HALF\_DAY\_MORNING">HALF DAY MORNING (Periods 1-4)</option>

<option value="HALF\_DAY\_AFTERNOON">HALF DAY AFTERNOON (Periods 5-9)</option>

<option value="PERMISSION\_MORNING">PERMISSION MORNING (Periods 1-2)</option>

<option value="PERMISSION\_EVENING">PERMISSION EVENING (Periods 8-9)</option>

</select>

</div>

</div>

<div>

<label class="block text-sm font-medium text-gray-700">Reason for Absence</label>

<textarea id="leaveReason" rows="3" required class="mt-1 block w-full p-2 border border-gray-300 rounded-md"></textarea>

</div>

<button type="submit" class="bg-red-600 text-white px-6 py-2 rounded-md hover:bg-red-700 transition duration-150">

<i class="fas fa-paper-plane mr-2"></i>Request Leave & Auto-Generate Subs

</button>

</form>

</div>

</div>

</div>

<!-- 2. My Substitutions (To Confirm/Complete) -->

<div data-tab-content="mySubs" class="tab-pane hidden">

<div class="section-card bg-white">

<div class="card-header">

Substitutions Assigned To Me (Action Required)

</div>

<div class="p-6">

<p class="text-sm text-gray-500 mb-4">You must \*\*Confirm\*\* an assignment to finalize it and \*\*Complete\*\* it after the period finishes to reset your workload score.</p>

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

<!-- Mock Substitution Item -->

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

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

<span class="font-bold text-lg">Class 8-A Math (Period 4)</span>

<span class="px-3 py-1 text-sm rounded-full bg-yellow-200 text-yellow-800">Pending</span>

</div>

<p class="text-sm mt-1">Absent: Ms. Sinha | Date: <span class="font-medium">2025-01-15</span></p>

<p class="text-xs italic">Reason: Medical Leave</p>

<div class="mt-3 space-x-2">

<button class="bg-green-600 text-white px-4 py-1 rounded text-sm hover:bg-green-700" onclick="handleConfirmation('SUB-001', 'T-1111')">

<i class="fas fa-check"></i> Confirm Sub

</button>

<button class="bg-blue-600 text-white px-4 py-1 rounded text-sm hover:bg-blue-700" onclick="handleCompletion('SUB-001')">

<i class="fas fa-clipboard-check"></i> Mark Complete

</button>

<button class="bg-red-600 text-white px-4 py-1 rounded text-sm hover:bg-red-700" onclick="handleCancellation('SUB-001')">

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

</button>

</div>

</div>

</div>

</div>

</div>

</div>

<!-- 3. Configuration Hub -->

<div data-tab-content="config" class="tab-pane hidden">

<div class="section-card bg-white">

<div class="card-header">

Substitution Algorithm & Workload Rules (Admin/Principal Only)

</div>

<form id="configForm" onsubmit="handleConfigUpdate(event)" class="p-6 space-y-6">

<!-- Workload Limits -->

<div class="border-b pb-4">

<h3 class="text-xl font-semibold mb-3 text-gray-700">Workload & Fairness Limits</h3>

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

<div>

<label class="block text-sm font-medium text-gray-700">Min Subs Required (for experience)</label>

<input type="number" id="minSubstitutions" min="0" value="0" class="mt-1 block w-full p-2 border rounded-md">

</div>

<div>

<label class="block text-sm font-medium text-gray-700">Max Subs Allowed (Fairness)</label>

<input type="number" id="maxSubstitutions" min="1" value="3" class="mt-1 block w-full p-2 border rounded-md">

</div>

<div>

<label class="block text-sm font-medium text-gray-700">Max Daily Periods Exclusion (7+ Excluded)</label>

<input type="number" id="maxDailyPeriodsExclusion" min="5" value="6" class="mt-1 block w-full p-2 border rounded-md">

<p class="text-xs text-gray-500 mt-1">Teachers teaching > this count are excluded.</p>

</div>

</div>

</div>

<!-- Scoring Weights -->

<div class="border-b pb-4">

<h3 class="text-xl font-semibold mb-3 text-gray-700">Scoring Weights (Prioritization)</h3>

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

<div>

<label class="block text-sm font-medium text-gray-700">Weight: Subject Match Score</label>

<input type="number" id="weightSubjectMatch" min="0" value="40" class="mt-1 block w-full p-2 border rounded-md">

</div>

<div>

<label class="block text-sm font-medium text-gray-700">Weight: Class Teacher Preference</label>

<input type="number" id="weightClassTeacher" min="0" value="20" class="mt-1 block w-full p-2 border rounded-md">

</div>

</div>

</div>

<!-- Restriction Rules -->

<div>

<h3 class="text-xl font-semibold mb-3 text-gray-700">Restriction Rules (Anti-Overload)</h3>

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

<input type="checkbox" id="restrictionSameClassDay" checked class="form-checkbox h-5 w-5 text-red-600">

<span class="text-gray-700">Restrict Substitute from covering the \*Same Class\* twice on the same day.</span>

</label>

<div class="mt-4">

<label class="block text-sm font-medium text-gray-700">Permanent Exclusion List (Profile IDs/Names)</label>

<input type="text" id="permanentExclusionList" placeholder="T-005, Mrs. Geetha, T-012" class="mt-1 block w-full p-2 border rounded-md">

<p class="text-xs text-gray-500 mt-1">Teachers listed here are permanently excluded from the substitute pool.</p>

</div>

</div>

<button type="submit" class="bg-green-600 text-white px-6 py-2 rounded-md hover:bg-green-700 transition duration-150">

<i class="fas fa-save mr-2"></i>Save Configuration & Update Algorithm

</button>

</form>

</div>

</div>

</div>

</div>

<script>

// --- Utility Functions ---

const API\_BASE = '';

const CURRENT\_USER\_ID = 'T-1111'; // Mock Teacher ID

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

const colors = { success: 'bg-green-600', error: 'bg-red-600', info: 'bg-blue-600', warning: 'bg-yellow-600' };

const icons = { success: 'fas fa-check', error: 'fas fa-exclamation-triangle', info: 'fas fa-info-circle', warning: 'fas fa-exclamation-triangle' };

const area = document.getElementById('notification-area');

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

messageDiv.className = `p-3 rounded-lg shadow-lg mb-2 text-white ${colors[type]}`;

messageDiv.innerHTML = `<i class="${icons[type]} mr-2"></i>${message}`;

area.appendChild(messageDiv);

setTimeout(() => messageDiv.remove(), 5000);

}

async function fetchAPI(endpoint, method = 'GET', body = null) {

try {

const options = {

method,

headers: { 'Content-Type': 'application/json' },

};

if (body && method !== 'GET') { options.body = JSON.stringify(body); }

const response = await fetch(`${API\_BASE}${endpoint}`, options);

const data = await response.json();

if (!response.ok) {

showNotification(data.message || data.error || `API Error: ${response.status}`, 'error');

throw new Error(data.error || 'API request failed');

}

return data;

} catch (error) {

console.error(`Fetch error on ${endpoint}:`, error);

throw error;

}

}

function switchTab(tabName) {

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

btn.classList.remove('active');

});

document.querySelector(`[data-tab="${tabName}"]`).classList.add('active');

document.querySelectorAll('.tab-pane').forEach(pane => {

pane.classList.add('hidden');

});

document.querySelector(`[data-tab-content="${tabName}"]`).classList.remove('hidden');

if (tabName === 'leave') updateTimeGateAlert();

}

function isSubmissionWindowOpenClient() {

const now = new Date();

const currentHour = now.getHours();

const currentMinute = now.getMinutes();

// 1:00 AM (60 minutes) to 7:20 AM (440 minutes)

const currentTimeInMinutes = currentHour \* 60 + currentMinute;

const startWindowInMinutes = 60;

const endWindowInMinutes = 440;

return currentTimeInMinutes >= startWindowInMinutes && currentTimeInMinutes <= endWindowInMinutes;

}

function updateTimeGateAlert() {

const today = new Date().toISOString().split('T')[0];

document.getElementById('currentDate').textContent = today;

const alertDiv = document.getElementById('timeGateAlert');

if (isSubmissionWindowOpenClient()) {

alertDiv.classList.add('time-alert');

alertDiv.innerHTML = `<i class="fas fa-clock mr-2"></i>\*\*WINDOW OPEN:\*\* Same-day leave submission is currently active.`;

} else {

alertDiv.classList.remove('time-alert');

alertDiv.classList.add('bg-gray-200');

alertDiv.innerHTML = `<i class="fas fa-clock mr-2"></i>\*\*WINDOW CLOSED:\*\* Same-day leave window (1:00 AM - 7:20 AM) is currently closed. Requests must be for future dates.`;

}

}

// --- Core Functions ---

async function handleLeaveRequest(event) {

event.preventDefault();

const btn = event.target.querySelector('button');

btn.disabled = true;

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

const today = new Date().toISOString().split('T')[0];

// Client-side check for time gate (Backend is the final authority)

if (leaveDate === today && !isSubmissionWindowOpenClient()) {

showNotification('Error: The submission window for today is closed (7:20 AM).', 'error');

btn.disabled = false;

return;

}

const payload = {

teacherId: document.getElementById('leaveTeacherId').value,

date: leaveDate,

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

reason: document.getElementById('leaveReason').value

};

try {

const data = await fetchAPI('/leave/request', 'POST', payload);

showNotification(data.message, 'success');

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

// Optionally refresh substitution list for this teacher

} catch (error) {

// Backend error notification handled in fetchAPI

} finally {

btn.disabled = false;

}

}

async function handleConfigUpdate(event) {

event.preventDefault();

const btn = event.target.querySelector('button');

btn.disabled = true;

const exclusionList = document.getElementById('permanentExclusionList').value

.split(',').map(s => s.trim()).filter(s => s.length > 0);

const payload = {

minSubstitutions: parseInt(document.getElementById('minSubstitutions').value),

maxSubstitutions: parseInt(document.getElementById('maxSubstitutions').value),

weightSubjectMatch: parseInt(document.getElementById('weightSubjectMatch').value),

weightClassTeacher: parseInt(document.getElementById('weightClassTeacher').value),

restrictionSameClassDay: document.getElementById('restrictionSameClassDay').checked,

maxDailyPeriodsExclusion: parseInt(document.getElementById('maxDailyPeriodsExclusion').value),

permanentExclusionList: exclusionList

};

try {

const data = await fetchAPI('/config', 'PUT', payload);

showNotification(data.message, 'success');

} catch {

// Error notification handled in fetchAPI

} finally {

btn.disabled = false;

}

}

function handleConfirmation(subId, substituteId) {

if (substituteId !== CURRENT\_USER\_ID) {

return showNotification("Error: You can only confirm your own assigned duties.", 'error');

}

// Logic to hit POST /:id/confirm

showNotification(`Confirmation sent for ${subId}. Workload increased by 1.`, 'success');

}

function handleCompletion(subId) {

// Logic to hit POST /:id/complete (requires feedback data)

showNotification(`Completion marked for ${subId}. Workload reset.`, 'info');

}

function handleCancellation(subId) {

// Logic to hit POST /:id/cancel (requires reason data)

showNotification(`Cancellation request sent for ${subId}.`, 'warning');

}

// --- Initialization ---

document.addEventListener('DOMContentLoaded', () => {

document.getElementById('leaveTeacherId').value = CURRENT\_USER\_ID;

document.getElementById('currentTeacherId').textContent = CURRENT\_USER\_ID;

updateTimeGateAlert();

// Set the default date to today for convenience

document.getElementById('leaveDate').value = new Date().toISOString().split('T')[0];

// Simulate fetching and applying config on load (for the config page)

fetchAPI('/config').then(data => {

const settings = data.settings?.config\_data || {};

document.getElementById('minSubstitutions').value = settings.minSubstitutions || 0;

document.getElementById('maxSubstitutions').value = settings.maxSubstitutions || 3;

document.getElementById('weightSubjectMatch').value = settings.weightSubjectMatch || 40;

document.getElementById('weightClassTeacher').value = settings.weightClassTeacher || 20;

document.getElementById('restrictionSameClassDay').checked = settings.restrictionSameClassDay !== false;

document.getElementById('maxDailyPeriodsExclusion').value = settings.maxDailyPeriodsExclusion || 6;

document.getElementById('permanentExclusionList').value = (settings.permanentExclusionList || []).join(', ');

}).catch(() => {

// Use default values if fetch fails

});

});

</script>

</body>

</html>

22. client/dist/modules/timesubbehave.ai.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Timesubbehave AI Premium - SmartGenEduX</title>

<link href="https://cdn.jsdelivr.net/npm/tailwindcss@2.2.19/dist/tailwind.min.css" rel="stylesheet">

<link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css" rel="stylesheet">

</head>

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

<div class="container mx-auto px-4 py-8">

<!-- Header -->

<div class="bg-gradient-to-r from-purple-600 to-blue-600 rounded-lg shadow-md p-6 mb-6 text-white">

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

<div class="flex items-center">

<img src="../assets/Timesubbehave AI\_20250621\_132022\_0000\_1751087241308.png" alt="Timesubbehave AI" class="w-12 h-12 mr-4 bg-white rounded-lg p-2">

<div>

<h1 class="text-2xl font-bold">Timesubbehave AI Premium</h1>

<p class="text-blue-100">Advanced AI analytics for timetable optimization and behavior prediction</p>

</div>

</div>

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

<span class="bg-yellow-500 text-yellow-900 px-3 py-1 rounded-full text-sm font-semibold">PREMIUM</span>

<button onclick="window.close()" class="bg-white bg-opacity-20 text-white px-4 py-2 rounded hover:bg-opacity-30">

<i class="fas fa-times mr-2"></i>Close

</button>

</div>

</div>

</div>

<!-- AI Dashboard -->

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

<div class="bg-gradient-to-br from-blue-500 to-blue-600 text-white rounded-lg p-6">

<div class="flex items-center">

<i class="fas fa-brain text-3xl mr-4"></i>

<div>

<div class="text-2xl font-bold">97.3%</div>

<div class="text-blue-100">AI Prediction Accuracy</div>

</div>

</div>

</div>

<div class="bg-gradient-to-br from-green-500 to-green-600 text-white rounded-lg p-6">

<div class="flex items-center">

<i class="fas fa-clock text-3xl mr-4"></i>

<div>

<div class="text-2xl font-bold">23%</div>

<div class="text-green-100">Time Optimization</div>

</div>

</div>

</div>

<div class="bg-gradient-to-br from-purple-500 to-purple-600 text-white rounded-lg p-6">

<div class="flex items-center">

<i class="fas fa-chart-line text-3xl mr-4"></i>

<div>

<div class="text-2xl font-bold">89%</div>

<div class="text-purple-100">Behavior Prediction</div>

</div>

</div>

</div>

</div>

<!-- AI Modules Grid -->

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

<!-- Smart Timetable AI -->

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

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

<div class="w-12 h-12 bg-blue-100 rounded-lg flex items-center justify-center mr-4">

<i class="fas fa-calendar-alt text-blue-600 text-xl"></i>

</div>

<div>

<h2 class="text-xl font-bold">Smart Timetable AI</h2>

<p class="text-gray-600">AI-powered schedule optimization</p>

</div>

</div>

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

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

<h3 class="font-semibold text-blue-900 mb-2">AI Recommendations</h3>

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

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

<span>Optimal class scheduling</span>

<span class="text-green-600"><i class="fas fa-check"></i></span>

</div>

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

<span>Teacher workload balancing</span>

<span class="text-green-600"><i class="fas fa-check"></i></span>

</div>

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

<span>Room utilization optimization</span>

<span class="text-orange-600"><i class="fas fa-exclamation-triangle"></i></span>

</div>

</div>

</div>

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

<button onclick="optimizeTimetable()" class="bg-blue-600 text-white px-4 py-2 rounded hover:bg-blue-700">

<i class="fas fa-magic mr-2"></i>AI Optimize

</button>

<button onclick="viewAnalytics()" class="bg-green-600 text-white px-4 py-2 rounded hover:bg-green-700">

<i class="fas fa-chart-bar mr-2"></i>Analytics

</button>

</div>

</div>

</div>

<!-- Intelligent Substitution -->

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

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

<div class="w-12 h-12 bg-green-100 rounded-lg flex items-center justify-center mr-4">

<i class="fas fa-user-friends text-green-600 text-xl"></i>

</div>

<div>

<h2 class="text-xl font-bold">Intelligent Substitution</h2>

<p class="text-gray-600">AI teacher replacement system</p>

</div>

</div>

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

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

<h3 class="font-semibold text-green-900 mb-2">Smart Matching</h3>

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

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

<span>Subject expertise matching</span>

<span class="text-green-600">95%</span>

</div>

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

<span>Availability prediction</span>

<span class="text-green-600">92%</span>

</div>

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

<span>Workload optimization</span>

<span class="text-blue-600">87%</span>

</div>

</div>

</div>

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

<h3 class="font-semibold text-orange-900 mb-2">Leave Request Management</h3>

<p class="text-sm text-orange-700 mb-3">Submit and manage teacher leave requests</p>

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

<input type="date" class="w-full border rounded px-3 py-2 text-sm" placeholder="Leave Date">

<select class="w-full border rounded px-3 py-2 text-sm">

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

<option value="sick">Sick Leave</option>

<option value="personal">Personal Leave</option>

<option value="emergency">Emergency Leave</option>

</select>

<textarea placeholder="Reason for leave..." class="w-full border rounded px-3 py-2 text-sm" rows="2"></textarea>

<button onclick="submitLeaveRequest()" class="bg-orange-600 text-white px-4 py-2 rounded hover:bg-orange-700 text-sm w-full">

<i class="fas fa-calendar-alt mr-2"></i>Submit Leave Request

</button>

</div>

</div>

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

<button onclick="smartSubstitution()" class="bg-green-600 text-white px-4 py-2 rounded hover:bg-green-700">

<i class="fas fa-robot mr-2"></i>Auto Assign

</button>

<button onclick="predictLeaves()" class="bg-purple-600 text-white px-4 py-2 rounded hover:bg-purple-700">

<i class="fas fa-crystal-ball mr-2"></i>Predict

</button>

</div>

</div>

</div>

<!-- Behavior Analytics AI -->

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

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

<div class="w-12 h-12 bg-purple-100 rounded-lg flex items-center justify-center mr-4">

<i class="fas fa-user-graduate text-purple-600 text-xl"></i>

</div>

<div>

<h2 class="text-xl font-bold">Behavior Analytics AI</h2>

<p class="text-gray-600">Student behavior prediction and intervention</p>

</div>

</div>

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

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

<h3 class="font-semibold text-purple-900 mb-2">Risk Assessment</h3>

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

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

<span>High-risk students</span>

<span class="px-2 py-1 bg-red-100 text-red-800 rounded">7 students</span>

</div>

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

<span>Improving students</span>

<span class="px-2 py-1 bg-green-100 text-green-800 rounded">23 students</span>

</div>

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

<span>Early intervention needed</span>

<span class="px-2 py-1 bg-orange-100 text-orange-800 rounded">12 students</span>

</div>

</div>

</div>

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

<button onclick="behaviorAnalysis()" class="bg-purple-600 text-white px-4 py-2 rounded hover:bg-purple-700">

<i class="fas fa-search mr-2"></i>Analyze

</button>

<button onclick="generateReports()" class="bg-orange-600 text-white px-4 py-2 rounded hover:bg-orange-700">

<i class="fas fa-file-alt mr-2"></i>Reports

</button>

</div>

</div>

</div>

<!-- Leave Request AI -->

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

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

<div class="w-12 h-12 bg-orange-100 rounded-lg flex items-center justify-center mr-4">

<i class="fas fa-calendar-times text-orange-600 text-xl"></i>

</div>

<div>

<h2 class="text-xl font-bold">Smart Leave Management</h2>

<p class="text-gray-600">AI-powered leave request processing</p>

</div>

</div>

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

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

<h3 class="font-semibold text-orange-900 mb-2">Leave Request Portal</h3>

<p class="text-sm text-orange-700 mb-3">Submit and manage leave requests with AI approval workflow</p>

<button onclick="openLeaveForm()" class="bg-orange-600 text-white px-4 py-2 rounded hover:bg-orange-700 text-sm">

<i class="fas fa-external-link-alt mr-2"></i>Open Leave Form

</button>

</div>

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

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

<span>Pending requests</span>

<span class="font-bold">8</span>

</div>

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

<span>Auto-approved today</span>

<span class="font-bold text-green-600">5</span>

</div>

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

<span>Requires review</span>

<span class="font-bold text-orange-600">3</span>

</div>

</div>

</div>

</div>

</div>

<!-- AI Insights Dashboard -->

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

<h2 class="text-xl font-bold mb-4">AI Insights & Predictions</h2>

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

<div class="text-center">

<div class="w-16 h-16 bg-blue-100 rounded-full flex items-center justify-center mx-auto mb-3">

<i class="fas fa-clock text-2xl text-blue-600"></i>

</div>

<h3 class="font-semibold mb-2">Schedule Efficiency</h3>

<div class="text-2xl font-bold text-blue-600">87%</div>

<div class="text-sm text-gray-600">↑ 12% from last month</div>

</div>

<div class="text-center">

<div class="w-16 h-16 bg-green-100 rounded-full flex items-center justify-center mx-auto mb-3">

<i class="fas fa-users text-2xl text-green-600"></i>

</div>

<h3 class="font-semibold mb-2">Teacher Satisfaction</h3>

<div class="text-2xl font-bold text-green-600">92%</div>

<div class="text-sm text-gray-600">↑ 5% from last month</div>

</div>

<div class="text-center">

<div class="w-16 h-16 bg-purple-100 rounded-full flex items-center justify-center mx-auto mb-3">

<i class="fas fa-chart-line text-2xl text-purple-600"></i>

</div>

<h3 class="font-semibold mb-2">Behavior Improvement</h3>

<div class="text-2xl font-bold text-purple-600">76%</div>

<div class="text-sm text-gray-600">↑ 23% from last month</div>

</div>

<div class="text-center">

<div class="w-16 h-16 bg-orange-100 rounded-full flex items-center justify-center mx-auto mb-3">

<i class="fas fa-calendar-check text-2xl text-orange-600"></i>

</div>

<h3 class="font-semibold mb-2">Leave Management</h3>

<div class="text-2xl font-bold text-orange-600">94%</div>

<div class="text-sm text-gray-600">Auto-approval rate</div>

</div>

</div>

</div>

<!-- Recent AI Actions -->

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

<h2 class="text-xl font-bold mb-4">Recent AI Actions</h2>

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

<div class="flex items-center p-3 bg-blue-50 rounded-lg">

<div class="w-10 h-10 bg-blue-100 rounded-full flex items-center justify-center mr-4">

<i class="fas fa-magic text-blue-600"></i>

</div>

<div class="flex-1">

<div class="font-semibold">Timetable optimized for Class 9</div>

<div class="text-sm text-gray-600">AI reduced scheduling conflicts by 34%</div>

</div>

<div class="text-sm text-gray-500">2 minutes ago</div>

</div>

<div class="flex items-center p-3 bg-green-50 rounded-lg">

<div class="w-10 h-10 bg-green-100 rounded-full flex items-center justify-center mr-4">

<i class="fas fa-user-check text-green-600"></i>

</div>

<div class="flex-1">

<div class="font-semibold">Smart substitution assigned</div>

<div class="text-sm text-gray-600">Mrs. Sharma automatically assigned to Math class</div>

</div>

<div class="text-sm text-gray-500">15 minutes ago</div>

</div>

<div class="flex items-center p-3 bg-purple-50 rounded-lg">

<div class="w-10 h-10 bg-purple-100 rounded-full flex items-center justify-center mr-4">

<i class="fas fa-exclamation-triangle text-purple-600"></i>

</div>

<div class="flex-1">

<div class="font-semibold">Behavior alert triggered</div>

<div class="text-sm text-gray-600">Early intervention recommended for 3 students</div>

</div>

<div class="text-sm text-gray-500">1 hour ago</div>

</div>

<div class="flex items-center p-3 bg-orange-50 rounded-lg">

<div class="w-10 h-10 bg-orange-100 rounded-full flex items-center justify-center mr-4">

<i class="fas fa-calendar-check text-orange-600"></i>

</div>

<div class="flex-1">

<div class="font-semibold">Leave request auto-approved</div>

<div class="text-sm text-gray-600">Mr. Kumar's medical leave approved with cover arranged</div>

</div>

<div class="text-sm text-gray-500">3 hours ago</div>

</div>

</div>

</div>

</div>

<script>

function optimizeTimetable() {

showMessage('AI is analyzing current timetable and optimizing schedules...', 'info');

setTimeout(() => {

showMessage('Timetable optimized! Efficiency improved by 23%.', 'success');

}, 3000);

}

function viewAnalytics() {

showMessage('Opening detailed timetable analytics dashboard...', 'info');

}

function smartSubstitution() {

showMessage('AI is finding the best substitute teacher based on subject expertise and availability...', 'info');

setTimeout(() => {

showMessage('Perfect match found! Mrs. Priya Sharma assigned to Math class.', 'success');

}, 2000);

}

function predictLeaves() {

showMessage('AI analyzing patterns to predict upcoming leave requests...', 'info');

setTimeout(() => {

showMessage('Prediction complete: 3 teachers likely to request leave next week.', 'success');

}, 2500);

}

function behaviorAnalysis() {

showMessage('Running AI behavior analysis on student data...', 'info');

setTimeout(() => {

showMessage('Analysis complete: 7 students require attention, 23 showing improvement.', 'success');

}, 4000);

}

function generateReports() {

showMessage('Generating AI-powered behavior reports...', 'info');

setTimeout(() => {

showMessage('Comprehensive behavior reports generated and ready for download.', 'success');

}, 3000);

}

function openLeaveForm() {

showMessage('Opening smart leave request portal...', 'info');

// In real implementation, this would open: https://forms.gle/F9edY9SqSzkm2JKW8

setTimeout(() => {

window.open('https://forms.gle/F9edY9SqSzkm2JKW8', '\_blank');

}, 1000);

}

function showMessage(message, type) {

const colors = {

'success': 'bg-green-600',

'error': 'bg-red-600',

'info': 'bg-blue-600'

};

const icons = {

'success': 'fas fa-check',

'error': 'fas fa-exclamation-triangle',

'info': 'fas fa-info-circle'

};

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

messageDiv.className = `fixed top-4 right-4 ${colors[type]} text-white px-6 py-3 rounded-lg shadow-lg z-50`;

messageDiv.innerHTML = `<i class="${icons[type]} mr-2"></i>${message}`;

document.body.appendChild(messageDiv);

setTimeout(() => {

messageDiv.remove();

}, 3000);

}

function submitLeaveRequest() {

showMessage('Leave request submitted successfully! It will be processed by AI and assigned a substitute teacher.', 'success');

}

function smartSubstitution() {

showMessage('AI is finding the best substitute teacher match...', 'info');

setTimeout(() => {

showMessage('Substitute teacher assigned automatically! Ms. Priya will cover Math class.', 'success');

}, 2000);

}

function predictLeaves() {

showMessage('Analyzing patterns to predict future leaves...', 'info');

setTimeout(() => {

showMessage('Prediction complete! 3 teachers likely to request leave next week.', 'success');

}, 3000);

}

function behaviorAnalysis() {

showMessage('Running AI behavior analysis on student data...', 'info');

setTimeout(() => {

showMessage('Analysis complete! 7 students need early intervention support.', 'success');

}, 4000);

}

</script>

</body>

</html>

23. client/dist/modules/timetable.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>⏰ Timetable Management System</title>

<!-- Bootstrap 5 CDN -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css">

<style>

body { background-color: #f8f9fa; font-family: 'Inter', sans-serif; }

.container-fluid { max-width: 1300px; margin-top: 20px; }

.card-header { background-color: #007bff; color: white; font-weight: bold; }

.section-card { margin-bottom: 25px; box-shadow: 0 4px 12px rgba(0,0,0,0.1); border-radius: 0.75rem; }

.alert-fixed { position: fixed; top: 20px; right: 20px; z-index: 1050; min-width: 350px; border-radius: 0.5rem; }

/\* Timetable Grid Styles \*/

.timetable-grid {

display: grid;

grid-template-columns: 80px repeat(5, 1fr); /\* 5 days + Time \*/

border: 1px solid #dee2e6;

}

.grid-header, .grid-cell {

padding: 8px;

border: 1px solid #dee2e6;

text-align: center;

font-size: 0.85rem;

min-height: 50px;

display: flex;

align-items: center;

justify-content: center;

flex-direction: column;

}

.grid-header { font-weight: bold; background-color: #e9ecef; }

.time-cell { background-color: #f8f9fa; font-weight: 500; }

.lesson-cell { cursor: pointer; transition: background-color 0.2s; }

.lesson-cell:hover { background-color: #e0f7ff; }

.break-cell { background-color: #f7e0e0; font-style: italic; color: #dc3545; }

.conflict-cell { background-color: #ffcccc !important; border: 2px dashed #dc3545; }

/\* Configuration Styles \*/

.period-item { display: flex; align-items: center; padding: 8px 0; border-bottom: 1px dashed #ccc; }

.period-item:last-child { border-bottom: none; }

.period-time { font-weight: bold; width: 120px; }

</style>

</head>

<body>

<div class="container-fluid">

<h1 class="text-center mb-4 text-primary"><i class="bi bi-calendar-range-fill"></i> Timetable Manager & Version Control</h1>

<!-- Notification Area -->

<div id="notification-area" class="alert-fixed" style="display: none;"></div>

<!-- Tab Navigation -->

<ul class="nav nav-tabs" id="timetableTab" role="tablist">

<li class="nav-item" role="presentation">

<button class="nav-link active" id="schedule-tab" data-bs-toggle="tab" data-bs-target="#schedule-view" type="button" role="tab"><i class="bi bi-grid-3x3-gap-fill"></i> View & Edit Schedule</button>

</li>

<li class="nav-item" role="presentation">

<button class="nav-link" id="preferences-tab" data-bs-toggle="tab" data-bs-target="#preferences" type="button" role="tab"><i class="bi bi-clock-history"></i> Teacher Preferences</button>

</li>

<li class="nav-item" role="presentation">

<button class="nav-link" id="config-tab" data-bs-toggle="tab" data-bs-target="#config" type="button" role="tab"><i class="bi bi-gear-fill"></i> Configuration Hub</button>

</li>

<li class="nav-item" role="presentation">

<button class="nav-link" id="audit-tab" data-bs-toggle="tab" data-bs-target="#audit" type="button" role="tab"><i class="bi bi-shield-lock-fill"></i> Audit & Rollback</button>

</li>

</ul>

<!-- Tab Content -->

<div class="tab-content pt-3" id="timetableTabContent">

<!-- 1. View & Edit Schedule -->

<div class="tab-pane fade show active" id="schedule-view" role="tabpanel">

<div class="card section-card">

<div class="card-header">Class Schedule Builder: <span id="currentVersionDisplay">Loading...</span></div>

<div class="card-body">

<div class="row mb-3 align-items-center">

<div class="col-md-3">

<label for="classSelector" class="form-label">Select Class</label>

<select class="form-select" id="classSelector" required>

<option value="C1A">Class 1-A (Mock)</option>

<option value="C2B">Class 2-B (Mock)</option>

<option value="T9C">Class 9-C (Mock)</option>

</select>

</div>

<div class="col-md-3">

<label for="academicYear" class="form-label">Academic Year</label>

<input type="text" class="form-control" id="academicYear" value="2024-25" required>

</div>

<div class="col-md-3 d-flex align-items-end">

<button class="btn btn-outline-primary me-2" id="refreshScheduleBtn"><i class="bi bi-arrow-clockwise"></i> Refresh Grid</button>

</div>

<div class="col-md-3 text-end d-flex align-items-end justify-content-end">

<button class="btn btn-info" id="exportScheduleBtn"><i class="bi bi-download"></i> Export PDF/HTML</button>

</div>

</div>

<!-- Timetable Grid Display -->

<div class="table-responsive">

<div id="timetableGridContainer" class="timetable-grid">

<!-- Grid populated dynamically here -->

<div class="grid-header">Time</div>

<div class="grid-header">Monday</div>

<div class="grid-header">Tuesday</div>

<div class="grid-header">Wednesday</div>

<div class="grid-header">Thursday</div>

<div class="grid-header">Friday</div>

</div>

</div>

</div>

</div>

</div>

<!-- 2. Teacher Preferences -->

<div class="tab-pane fade" id="preferences" role="tabpanel">

<div class="card section-card">

<div class="card-header">Teacher Unavailability & Preferences</div>

<div class="card-body">

<p class="alert alert-info">Use this to record periods when a teacher is unavailable (e.g., non-teaching duty, fixed meeting, personal appointments). This data is used by the system's conflict checking algorithm.</p>

<form id="teacherPreferenceForm">

<div class="mb-3">

<label for="prefTeacherId" class="form-label">Select Teacher</label>

<select class="form-select" id="prefTeacherId" required>

<option value="T001">Teacher Jane Doe (T001)</option>

<option value="T002">Teacher John Smith (T002)</option>

</select>

</div>

<h5>Unavailable Periods (Maximum 5)</h5>

<div id="unavailablePeriodsContainer" class="p-3 border rounded mb-3">

<!-- Dynamic period inputs will be added here -->

<p class="text-muted text-center">Add unavailable time slots below.</p>

</div>

<button type="button" class="btn btn-sm btn-outline-secondary mb-3" id="addUnavailablePeriodBtn"><i class="bi bi-plus-circle"></i> Add Period</button>

<button type="submit" class="btn btn-primary w-100" id="savePreferencesBtn"><i class="bi bi-save"></i> Save Preferences</button>

</form>

</div>

</div>

</div>

<!-- 3. Configuration Hub -->

<div class="tab-pane fade" id="config" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-success">Configuration Hub: Periods, Wings & Holidays (Manager Access)</div>

<div class="card-body">

<form id="configForm">

<!-- Period Definition -->

<h5>1. Period Structure & Timings</h5>

<div class="table-responsive mb-4">

<table class="table table-bordered table-sm" id="periodConfigTable">

<thead>

<tr>

<th>#</th>

<th>Start Time</th>

<th>End Time</th>

<th>Name</th>

<th>Type</th>

<th>Action</th>

</tr>

</thead>

<tbody id="periodConfigBody">

<!-- Dynamic periods will be added here -->

</tbody>

</table>

<button type="button" class="btn btn-sm btn-outline-success" id="addPeriodBtn"><i class="bi bi-plus-circle"></i> Add Period</button>

</div>

<!-- Wings & Holidays -->

<div class="row mb-4">

<h5>2. System Settings</h5>

<div class="col-md-6 mb-3">

<label for="configWings" class="form-label">Defined Wings (Comma Separated)</label>

<input type="text" class="form-control" id="configWings" value="Primary,Middle,Secondary" required>

</div>

<div class="col-md-6 mb-3">

<label for="configHolidays" class="form-label">Holiday Dates (ISO Format, Comma Separated)</label>

<textarea class="form-control" id="configHolidays" rows="3" placeholder="2024-12-25, 2025-01-01"></textarea>

<div class="form-text">These dates are skipped during automatic schedule generation.</div>

</div>

</div>

<button type="submit" class="btn btn-success w-100" id="saveConfigBtn"><i class="bi bi-save"></i> Save Configuration & Create New Version</button>

</form>

</div>

</div>

</div>

<!-- 4. Audit & Rollback -->

<div class="tab-pane fade" id="audit" role="tabpanel">

<div class="card section-card">

<div class="card-header bg-danger">Timetable Version Audit & Rollback (Manager/Admin Only)</div>

<div class="card-body">

<p class="alert alert-warning">Review historical configurations. Rolling back deletes the current schedule and restores the selected version's settings and entries.</p>

<div class="table-responsive">

<table class="table table-striped table-sm">

<thead>

<tr>

<th>Version Name</th>

<th>Date Created</th>

<th>Created By</th>

<th>Holiday Count</th>

<th>Action</th>

</tr>

</thead>

<tbody id="auditHistoryTableBody">

<tr><td colspan="5" class="text-center">Loading audit history...</td></tr>

</tbody>

</table>

</div>

</div>

</div>

</div>

</div>

</div>

<!-- Modals -->

<!-- Schedule Entry Modal (POST /schedule) -->

<div class="modal fade" id="scheduleModal" tabindex="-1">

<div class="modal-dialog">

<div class="modal-content">

<div class="modal-header bg-primary text-white">

<h5 class="modal-title">Schedule Entry: <span id="modalDay"></span> - Period <span id="modalPeriod"></span></h5>

<button type="button" class="btn-close btn-close-white" data-bs-dismiss="modal"></button>

</div>

<form id="scheduleEntryForm">

<div class="modal-body">

<input type="hidden" id="entryDayOfWeek">

<input type="hidden" id="entryPeriodNumber">

<input type="hidden" id="entryClassId">

<div class="mb-3">

<label for="entrySubjectId" class="form-label">Subject ID</label>

<input type="text" class="form-control" id="entrySubjectId" placeholder="e.g., MATH101" required>

</div>

<div class="mb-3">

<label for="entryTeacherProfileId" class="form-label">Teacher ID</label>

<input type="text" class="form-control" id="entryTeacherProfileId" placeholder="e.g., T001" required>

</div>

<div class="mb-3">

<label for="entryRoomNumber" class="form-label">Room Number</label>

<input type="text" class="form-control" id="entryRoomNumber" placeholder="e.g., A-205" required>

</div>

<div class="alert alert-warning d-none" id="conflictAlert">

Conflict check failed! Cannot schedule here.

</div>

</div>

<div class="modal-footer">

<button type="button" class="btn btn-secondary" data-bs-dismiss="modal">Cancel</button>

<button type="submit" class="btn btn-success" id="saveScheduleEntryBtn">Save Entry</button>

</div>

</form>

</div>

</div>

</div>

<!-- Bootstrap Bundle with Popper -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>

<script>

const API\_BASE = '';

let configSettings = { periods: [], wings: [], holidayDates: [] };

let currentSchedule = [];

const dayMap = { 1: 'Monday', 2: 'Tuesday', 3: 'Wednesday', 4: 'Thursday', 5: 'Friday', 6: 'Saturday', 7: 'Sunday' };

/\*\* Helper function to show a temporary notification. \*/

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

const area = document.getElementById('notification-area');

area.innerHTML = `

<div class="alert alert-${type} alert-dismissible fade show" role="alert">

${message}

<button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>

</div>

`;

area.style.display = 'block';

setTimeout(() => {

area.style.display = 'none';

}, 7000);

}

// --- Core Fetch Function ---

async function apiFetch(endpoint, method = 'GET', body = null) {

try {

const options = {

method: method,

headers: { 'Content-Type': 'application/json' },

};

if (body && method !== 'GET') {

options.body = JSON.stringify(body);

}

const response = await fetch(`${API\_BASE}${endpoint}`, options);

const data = await response.json();

if (!response.ok) {

throw new Error(data.error || `API error (${response.status})`);

}

return data;

} catch (error) {

console.error('API Fetch Error:', endpoint, error);

showNotification(error.message || 'Network error.', 'danger');

throw error;

}

}

// --- 3. Configuration Hub Logic ---

async function fetchConfig() {

try {

const data = await apiFetch('/config');

configSettings = data.settings.settings || { periods: [], wings: ['Primary', 'Middle', 'Secondary'], holidayDates: [] };

// Update version display

document.getElementById('currentVersionDisplay').textContent = data.settings.version\_name ? `Version: ${data.settings.version\_name}` : 'Version: Draft';

// Populate Period Table

renderPeriodConfig(configSettings.periods);

// Populate System Settings

document.getElementById('configWings').value = configSettings.wings ? configSettings.wings.join(', ') : '';

document.getElementById('configHolidays').value = configSettings.holidayDates ? configSettings.holidayDates.map(d => new Date(d).toISOString().split('T')[0]).join(', ') : '';

} catch (error) {

console.error('Failed to load configuration. Using defaults.');

// Show notification only if it's not a 403 or simple not found error

}

}

function renderPeriodConfig(periods) {

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

container.innerHTML = '';

periods.sort((a, b) => a.id - b.id);

periods.forEach(p => {

const row = container.insertRow();

row.setAttribute('data-period-id', p.id);

row.innerHTML = `

<td>${p.id}</td>

<td><input type="time" class="form-control form-control-sm" value="${p.startTime}" required></td>

<td><input type="time" class="form-control form-control-sm" value="${p.endTime}" required></td>

<td><input type="text" class="form-control form-control-sm" value="${p.name}" required></td>

<td>

<select class="form-select form-select-sm" required>

<option value="academic" ${p.type === 'academic' ? 'selected' : ''}>Academic</option>

<option value="break" ${p.type === 'break' ? 'selected' : ''}>Break</option>

<option value="assembly" ${p.type === 'assembly' ? 'selected' : ''}>Assembly</option>

</select>

</td>

<td><button type="button" class="btn btn-sm btn-danger delete-period-btn"><i class="bi bi-x-circle"></i></button></td>

`;

});

}

// Add new blank period row

document.getElementById('addPeriodBtn').addEventListener('click', () => {

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

const newId = container.rows.length + 1; // Simple incrementing ID

const row = container.insertRow();

row.setAttribute('data-period-id', newId);

row.innerHTML = `

<td>${newId}</td>

<td><input type="time" class="form-control form-control-sm" value="09:00" required></td>

<td><input type="time" class="form-control form-control-sm" value="09:40" required></td>

<td><input type="text" class="form-control form-control-sm" value="New Period" required></td>

<td>

<select class="form-select form-select-sm" required>

<option value="academic">Academic</option>

<option value="break">Break</option>

<option value="assembly">Assembly</option>

</select>

</td>

<td><button type="button" class="btn btn-sm btn-danger delete-period-btn"><i class="bi bi-x-circle"></i></button></td>

`;

});

// Handle period deletion using delegation

document.getElementById('periodConfigTable').addEventListener('click', (e) => {

if (e.target.closest('.delete-period-btn')) {

e.target.closest('tr').remove();

}

});

// Handle Config Form Submission (PUT /config)

document.getElementById('configForm').addEventListener('submit', async (e) => {

e.preventDefault();

const btn = e.submitter;

btn.disabled = true;

const periods = [];

const rows = document.getElementById('periodConfigBody').rows;

for (let i = 0; i < rows.length; i++) {

const cells = rows[i].cells;

periods.push({

id: parseInt(cells[0].textContent),

startTime: cells[1].querySelector('input').value,

endTime: cells[2].querySelector('input').value,

name: cells[3].querySelector('input').value,

type: cells[4].querySelector('select').value,

});

}

const holidayDates = document.getElementById('configHolidays').value.split(',').map(d => d.trim()).filter(d => d);

const payload = {

periods: periods,

wings: document.getElementById('configWings').value.split(',').map(w => w.trim()).filter(w => w),

defaultTemplateId: 'default\_template\_v1', // Mock static ID

holidayDates: holidayDates

};

try {

const data = await apiFetch('/config', 'PUT', payload);

showNotification(data.message, 'success');

fetchConfig(); // Refresh config display and version number

} catch {

// Error handled by apiFetch

} finally {

btn.disabled = false;

}

});

// --- 1. View & Edit Schedule Logic ---

document.getElementById('refreshScheduleBtn').addEventListener('click', () => {

const classId = document.getElementById('classSelector').value;

if (classId) {

fetchSchedule(classId);

}

});

async function fetchSchedule(classId) {

try {

// 1. Fetch current schedule entries

const data = await apiFetch(`/schedule/${classId}`);

currentSchedule = data.schedule;

// 2. Render the grid

renderTimetableGrid(configSettings.periods, currentSchedule, classId);

} catch (error) {

showNotification('Could not load schedule. Check configuration and network.', 'danger');

document.getElementById('timetableGridContainer').innerHTML = 'Failed to load schedule grid.';

}

}

function renderTimetableGrid(periods, schedule, classId) {

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

container.innerHTML = '';

// Set grid columns (Time + 5 days)

container.style.gridTemplateColumns = `80px repeat(${Object.keys(dayMap).filter(d => d <= 5).length}, 1fr)`;

// Render Day Headers (1-5 for Mon-Fri)

container.innerHTML += `<div class="grid-header">Time</div>`;

for (const day in dayMap) {

if (day <= 5) {

container.innerHTML += `<div class="grid-header">${dayMap[day]}</div>`;

}

}

// Render Periods/Rows

periods.forEach(period => {

// Time Cell (Row Header)

const timeText = `${period.startTime} - ${period.endTime}`;

container.innerHTML += `<div class="grid-header time-cell">${period.name}<br><small>(${timeText})</small></div>`;

for (let day = 1; day <= 5; day++) {

const entry = schedule.find(s => s.day\_of\_week == day && s.period\_number == period.id);

let cellClass = 'lesson-cell';

let content = '+ Add Class';

if (period.type !== 'academic') {

cellClass = 'break-cell';

content = period.name;

} else if (entry) {

cellClass = 'lesson-cell bg-light'; // Lesson scheduled

content = `

<strong>${entry.subject\_name}</strong>

<small>${entry.teacher\_name}</small>

<small class="text-muted">Room: ${entry.room\_number}</small>

`;

}

// Add unique data attributes for the modal to use

container.innerHTML += `

<div class="${cellClass} grid-cell"

data-day="${day}"

data-day-name="${dayMap[day]}"

data-period="${period.id}"

data-class-id="${classId}"

data-bs-toggle="${period.type === 'academic' ? 'modal' : ''}"

data-bs-target="${period.type === 'academic' ? '#scheduleModal' : ''}">

${content}

</div>

`;

}

});

}

// Handle Modal Trigger

const scheduleModalElement = document.getElementById('scheduleModal');

const scheduleModal = new bootstrap.Modal(scheduleModalElement);

scheduleModalElement.addEventListener('show.bs.modal', (e) => {

const invoker = e.relatedTarget;

if (!invoker || invoker.classList.contains('break-cell')) return;

const day = invoker.getAttribute('data-day');

const period = invoker.getAttribute('data-period');

const classId = invoker.getAttribute('data-class-id');

document.getElementById('modalDay').textContent = dayMap[day];

document.getElementById('modalPeriod').textContent = period;

document.getElementById('entryDayOfWeek').value = day;

document.getElementById('entryPeriodNumber').value = period;

document.getElementById('entryClassId').value = classId;

document.getElementById('conflictAlert').classList.add('d-none');

});

// Handle Schedule Entry Submission (POST /schedule)

document.getElementById('scheduleEntryForm').addEventListener('submit', async (e) => {

e.preventDefault();

const btn = e.submitter;

btn.disabled = true;

const classId = document.getElementById('classSelector').value;

const payload = {

dayOfWeek: parseInt(document.getElementById('entryDayOfWeek').value),

classId: classId,

periodNumber: parseInt(document.getElementById('entryPeriodNumber').value),

subjectId: document.getElementById('entrySubjectId').value,

teacherProfileId: document.getElementById('entryTeacherProfileId').value,

roomNumber: document.getElementById('entryRoomNumber').value,

academicYear: document.getElementById('academicYear').value

};

try {

await apiFetch('/schedule', 'POST', payload);

showNotification('Schedule entry saved successfully.', 'success');

scheduleModal.hide();

fetchSchedule(classId); // Refresh grid

} catch (error) {

document.getElementById('conflictAlert').textContent = error.message.includes('Scheduling conflict') ? error.message : 'Failed to save entry.';

document.getElementById('conflictAlert').classList.remove('d-none');

} finally {

btn.disabled = false;

}

});

document.getElementById('exportScheduleBtn').addEventListener('click', async () => {

const classId = document.getElementById('classSelector').value;

if (!classId) return showNotification('Select a class to export.', 'warning');

try {

const data = await apiFetch(`/export/${classId}?format=PDF`);

showNotification(data.message, 'success');

} catch {

// Error handled by apiFetch

}

});

// --- 2. Teacher Preferences Logic ---

document.getElementById('addUnavailablePeriodBtn').addEventListener('click', () => {

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

const currentCount = container.querySelectorAll('.row').length;

if (currentCount >= 5) {

return showNotification('Maximum 5 unavailable periods allowed.', 'warning');

}

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

newRow.className = 'row g-2 mb-2';

newRow.innerHTML = `

<div class="col-md-4">

<select class="form-select form-select-sm unavailable-day" required>

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

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

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

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

<option value="4">Thursday</option>

<option value="5">Friday</option>

</select>

</div>

<div class="col-md-3">

<input type="number" class="form-control form-control-sm unavailable-period" placeholder="Period #" required min="1">

</div>

<div class="col-md-4">

<input type="text" class="form-control form-control-sm unavailable-reason" placeholder="Reason (Optional)">

</div>

<div class="col-md-1">

<button type="button" class="btn btn-sm btn-outline-danger remove-period-btn"><i class="bi bi-trash"></i></button>

</div>

`;

container.appendChild(newRow);

});

document.getElementById('unavailablePeriodsContainer').addEventListener('click', (e) => {

if (e.target.closest('.remove-period-btn')) {

e.target.closest('.row').remove();

}

});

document.getElementById('teacherPreferenceForm').addEventListener('submit', async (e) => {

e.preventDefault();

const btn = e.submitter;

btn.disabled = true;

const teacherId = document.getElementById('prefTeacherId').value;

const periods = [];

const rows = document.getElementById('unavailablePeriodsContainer').querySelectorAll('.row');

rows.forEach(row => {

const daySelect = row.querySelector('.unavailable-day');

const periodInput = row.querySelector('.unavailable-period');

const reasonInput = row.querySelector('.unavailable-reason');

if (daySelect && periodInput && daySelect.value && periodInput.value) {

periods.push({

dayOfWeek: parseInt(daySelect.value),

periodNumber: parseInt(periodInput.value),

reason: reasonInput.value

});

}

});

if (periods.length === 0) {

showNotification('Add at least one unavailable period.', 'warning');

btn.disabled = false;

return;

}

const payload = {

teacherId: teacherId,

unavailablePeriods: periods

};

try {

const data = await apiFetch('/teacher-preferences', 'POST', payload);

showNotification(data.message, 'success');

} catch {

// Error handled by apiFetch

} finally {

btn.disabled = false;

}

});

// --- 4. Audit & Rollback Logic ---

async function fetchAuditHistory() {

const tableBody = document.getElementById('auditHistoryTableBody');

tableBody.innerHTML = '<tr><td colspan="5" class="text-center"><div class="spinner-border spinner-border-sm"></div> Loading...</td></tr>';

try {

const data = await apiFetch('/audit/history');

const history = data.history;

tableBody.innerHTML = '';

if (history.length === 0) {

tableBody.innerHTML = '<tr><td colspan="5" class="text-center">No audit history found.</td></tr>';

return;

}

history.forEach(item => {

const row = tableBody.insertRow();

row.innerHTML = `

<td>${item.version\_name}</td>

<td>${new Date(item.created\_at).toLocaleString()}</td>

<td>${item.created\_by}</td>

<td>${item.holiday\_count || 'N/A'}</td>

<td>

<button class="btn btn-sm btn-danger rollback-btn" data-version="${item.version\_name}">Rollback</button>

</td>

`;

});

} catch {

tableBody.innerHTML = '<tr><td colspan="5" class="text-center text-danger">Failed to load audit history.</td></tr>';

}

}

document.getElementById('auditHistoryTableBody').addEventListener('click', async (e) => {

if (e.target.classList.contains('rollback-btn')) {

const versionName = e.target.getAttribute('data-version');

const confirmed = confirm(`WARNING: Are you sure you want to rollback to version ${versionName}? This will overwrite the current schedule.`);

if (!confirmed) return;

e.target.disabled = true;

e.target.textContent = 'Rolling back...';

try {

const data = await apiFetch(`/audit/rollback/${versionName}`, 'POST', {});

showNotification(data.message, 'success');

fetchConfig();

fetchAuditHistory();

document.getElementById('refreshScheduleBtn').click(); // Refresh the grid

} catch {

e.target.disabled = false;

e.target.textContent = 'Rollback';

}

}

});

// --- Initialization ---

document.addEventListener('DOMContentLoaded', () => {

fetchConfig().then(() => {

// Load default schedule after config is loaded

document.getElementById('refreshScheduleBtn').click();

});

fetchAuditHistory();

});

</script>

</body>

</html>

24. client/dist/modules/Vipu.ai.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Vipu AI - Principal & Super Admin Oversight</title>

<!-- Tailwind CSS (Emulation) -->

<link href="https://cdn.jsdelivr.net/npm/tailwindcss@2.2.19/dist/tailwind.min.css" rel="stylesheet">

<!-- Font Awesome Icons -->

<link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css" rel="stylesheet">

<style>

body { background-color: #f4f7f9; font-family: 'Inter', sans-serif; }

.container { max-width: 1200px; margin-top: 20px; }

.card-header { background-color: #2c3e50; color: white; font-weight: bold; }

.chat-container { height: 450px; overflow-y: auto; background: #f8f9fa; border-radius: 0.5rem; padding: 1rem; }

.chat-message { max-width: 80%; margin-bottom: 1rem; padding: 0.75rem 1rem; border-radius: 1.5rem; }

.chat-message.user { background-color: #3498db; color: white; margin-left: auto; border-bottom-right-radius: 0; }

.chat-message.assistant { background-color: #ecf0f1; color: #2c3e50; margin-right: auto; border-bottom-left-radius: 0; }

.tab-button.active { background-color: #3498db; color: white; }

.nav-link { transition: all 0.2s; }

.metric-value { font-size: 2.5rem; font-weight: 700; color: #3498db; }

.metric-label { color: #6b7280; }

</style>

</head>

<body>

<div class="container mx-auto">

<h1 class="text-3xl font-extrabold text-center mb-6 text-gray-800">🧠 Vipu AI Oversight Assistant</h1>

<!-- Content Area - Initially Hidden for Role Check -->

<div id="contentWrapper" style="display: none;">

<!-- Tab Navigation -->

<div class="bg-white rounded-lg shadow-md mb-4">

<div class="flex border-b border-gray-200">

<button class="tab-button nav-link px-6 py-3 text-sm font-medium active flex-1" onclick="switchTab('chat')" data-tab="chat">

<i class="fas fa-comments mr-2"></i>AI Chat Support

</button>

<button class="tab-button nav-link px-6 py-3 text-sm font-medium flex-1" onclick="switchTab('history')" data-tab="history">

<i class="fas fa-history mr-2"></i>Conversation History

</button>

<button class="tab-button nav-link px-6 py-3 text-sm font-medium flex-1" onclick="switchTab('knowledge')" data-tab="knowledge">

<i class="fas fa-book-open mr-2"></i>Knowledge Base

</button>

<button class="tab-button nav-link px-6 py-3 text-sm font-medium flex-1" onclick="switchTab('analytics')" data-tab="analytics">

<i class="fas fa-chart-line mr-2"></i>Usage Analytics

</button>

<button class="tab-button nav-link px-6 py-3 text-sm font-medium flex-1" onclick="switchTab('settings')" data-tab="settings">

<i class="fas fa-cog mr-2"></i>AI Configuration

</button>

</div>

</div>

<!-- Tab Content -->

<div id="tabContent">

<!-- 1. AI Chat Support -->

<div data-tab-content="chat" class="tab-pane active">

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

<!-- Chat Panel -->

<div class="lg:col-span-2 bg-white rounded-lg shadow-lg p-4 flex flex-col h-[600px]">

<div id="chatMessages" class="chat-container flex-grow">

<!-- Messages will be injected here -->

</div>

<div class="mt-4 flex space-x-3">

<input type="text" id="chatInput" placeholder="Ask Vipu AI for defaulter lists, exam schedules, or critical event status..." class="flex-1 border-2 border-gray-300 p-3 rounded-full focus:outline-none focus:border-blue-500" onkeypress="handleEnter(event)">

<button onclick="sendMessage()" class="bg-blue-600 text-white w-14 h-14 rounded-full hover:bg-blue-700 transition duration-150">

<i class="fas fa-paper-plane"></i>

</button>

</div>

<div class="mt-2 text-xs text-gray-500 text-center">

Vipu AI access level: <span class="font-bold text-red-600">PRINCIPAL/SUPER ADMIN</span> (<span id="currentUserId">Loading...</span>)

</div>

</div>

<!-- Capabilities and Quick Help -->

<div class="lg:col-span-1 space-y-6">

<!-- Quick Help / Related Topics (Management Focus) -->

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

<h3 class="text-lg font-bold mb-3 flex items-center text-gray-700">

<i class="fas fa-lightbulb mr-2 text-yellow-500"></i>Critical Oversight Queries

</h3>

<div id="quickHelpSuggestions" class="space-y-2 text-sm">

<button class="w-full text-left p-2 bg-red-100 hover:bg-red-200 rounded transition" onclick="quickHelp('defaulters')">

<i class="fas fa-money-bill-wave mr-2"></i>Get current Fee Defaulters List

</button>

<button class="w-full text-left p-2 bg-orange-100 hover:bg-orange-200 rounded transition" onclick="quickHelp('exam\_schedule')">

<i class="fas fa-calendar-alt mr-2"></i>Show final Exam Schedule & Invigilator Chart

</button>

<button class="w-full text-left p-2 bg-blue-100 hover:bg-blue-200 rounded transition" onclick="quickHelp('event\_status')">

<i class="fas fa-bullhorn mr-2"></i>What events are happening today?

</button>

</div>

</div>

<!-- Capabilities -->

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

<h3 class="text-lg font-bold mb-3 flex items-center text-gray-700">

<i class="fas fa-wrench mr-2 text-purple-500"></i>Vipu AI Management Capabilities

</h3>

<div id="capabilitiesList" class="space-y-2 text-sm">

<!-- Capabilities injected here -->

</div>

</div>

</div>

</div>

</div>

<!-- 2. Conversation History -->

<div data-tab-content="history" class="tab-pane hidden">

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

<h2 class="text-2xl font-bold mb-4 text-gray-700">Your Past Conversations</h2>

<div id="historyFilters" class="mb-4 space-x-4">

<select id="historyStatusFilter" class="p-2 border rounded" onchange="fetchUserHistory()">

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

<option value="active">Active</option>

<option value="completed">Completed</option>

</select>

<button class="bg-gray-200 p-2 rounded hover:bg-gray-300" onclick="fetchUserHistory()">

<i class="fas fa-sync-alt mr-1"></i>Refresh

</button>

</div>

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

<p class="text-gray-500">Loading history...</p>

</div>

</div>

</div>

<!-- 3. Knowledge Base -->

<div data-tab-content="knowledge" class="tab-pane hidden">

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

<h2 class="text-2xl font-bold mb-4 text-gray-700">Knowledge Base Explorer</h2>

<!-- Search and Filters -->

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

<input type="text" id="kbSearchInput" placeholder="Search policies, procedures, and management guidance..." class="flex-1 border-2 border-gray-300 p-3 rounded-lg focus:outline-none focus:border-blue-500">

<select id="kbCategoryFilter" class="p-3 border rounded-lg" onchange="fetchKnowledgeBase()">

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

<!-- Categories injected here -->

</select>

<button onclick="searchKnowledge()" class="bg-blue-600 text-white px-6 py-3 rounded-lg hover:bg-blue-700">

<i class="fas fa-search mr-2"></i>Search

</button>

</div>

<!-- Categories -->

<div id="kbCategories" class="flex flex-wrap gap-3 mb-6">

<!-- Category buttons injected here -->

</div>

<!-- Popular Questions -->

<h3 class="text-xl font-bold mb-3 text-gray-700">Popular FAQs</h3>

<div id="popularQuestionsList" class="space-y-3">

<p class="text-gray-500">Loading popular questions...</p>

</div>

</div>

</div>

<!-- 4. Analytics & Oversight (Admin/Principal Only) -->

<div data-tab-content="analytics" class="tab-pane hidden">

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

<h2 class="text-2xl font-bold mb-4 text-gray-700 flex items-center">

<i class="fas fa-chart-line mr-2 text-green-600"></i>System Analytics & Performance

</h2>

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

<!-- Usage Metrics -->

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

<div class="metric-value" id="totalConversations">0</div>

<div class="metric-label">Total Conversations</div>

</div>

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

<div class="metric-value" id="activeUsers">0</div>

<div class="metric-label">Active Staff Users</div>

</div>

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

<div class="metric-value" id="avgResponseTime">0s</div>

<div class="metric-label">Avg. Response Time</div>

</div>

<!-- Performance Metrics -->

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

<div class="bg-gray-100 p-4 rounded-lg">

<h3 class="font-semibold mb-2">Helpfulness Rate</h3>

<div class="text-3xl font-bold text-green-600" id="helpfulnessRate">0%</div>

<p class="text-sm text-gray-500">User marked as helpful (Assistant messages)</p>

</div>

<div class="bg-gray-100 p-4 rounded-lg">

<h3 class="font-semibold mb-2">Response Accuracy</h3>

<div class="text-3xl font-bold text-blue-600" id="responseAccuracy">0%</div>

<p class="text-sm text-gray-500">AI confidence score average</p>

</div>

<div class="bg-gray-100 p-4 rounded-lg">

<h3 class="font-semibold mb-2">Most Popular Topic</h3>

<div class="text-lg font-bold text-purple-600" id="mostPopularTopic">...</div>

<p class="text-sm text-gray-500">Category generating most traffic</p>

</div>

</div>

<!-- Usage Distribution -->

<div class="md:col-span-3 bg-gray-100 p-4 rounded-lg">

<h3 class="font-bold mb-2">Usage Distribution by Role</h3>

<div id="roleDistribution" class="text-sm space-y-1">

<!-- Distribution data injected here -->

</div>

</div>

</div>

</div>

</div>

<!-- 5. Configuration & Settings -->

<div data-tab-content="settings" class="tab-pane hidden">

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

<h2 class="text-2xl font-bold mb-4 text-gray-700 flex items-center">

<i class="fas fa-tools mr-2 text-orange-500"></i>AI System Settings

</h2>

<p class="text-red-600 mb-6">Settings changes affect Vipu AI's behavior and learning across the entire system. Requires management clearance.</p>

<form id="settingsForm" onsubmit="updateSettings(event)" class="space-y-6">

<!-- General Preferences -->

<div class="border-b pb-4">

<h3 class="text-xl font-semibold mb-3">General Preferences</h3>

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

<div>

<label class="block text-sm font-medium text-gray-700">Response Style</label>

<select id="responseStyle" class="mt-1 block w-full p-2 border border-gray-300 rounded-md">

<option value="friendly\_professional">Friendly & Professional</option>

<option value="formal">Formal & Detailed</option>

<option value="concise">Concise & Direct</option>

</select>

</div>

<div>

<label class="block text-sm font-medium text-gray-700">Data Retention (Days)</label>

<input type="number" id="dataRetention" min="30" max="365" class="mt-1 block w-full p-2 border border-gray-300 rounded-md">

</div>

</div>

</div>

<!-- Behavioral and Learning Controls -->

<div class="border-b pb-4">

<h3 class="text-xl font-semibold mb-3">Behavioral & Privacy Controls</h3>

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

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

<input type="checkbox" id="learningEnabled" class="form-checkbox h-5 w-5 text-blue-600">

<span class="text-gray-700">Enable System Learning (Improve AI over time)</span>

</label>

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

<input type="checkbox" id="personalizedResponses" class="form-checkbox h-5 w-5 text-blue-600">

<span class="text-gray-700">Enable Personalized Responses (Use user role/context)</span>

</label>

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

<input type="checkbox" id="contentFiltering" class="form-checkbox h-5 w-5 text-red-600">

<span class="text-gray-700">Strict Content Filtering/Moderation</span>

</label>

<div>

<label class="block text-sm font-medium text-gray-700">Privacy Mode</label>

<select id="privacyMode" class="mt-1 block w-full p-2 border border-gray-300 rounded-md">

<option value="enhanced">Enhanced (Recommended)</option>

<option value="standard">Standard</option>

</select>

</div>

</div>

</div>

<button type="submit" class="bg-orange-600 text-white px-6 py-3 rounded-lg hover:bg-orange-700 transition duration-150">

<i class="fas fa-save mr-2"></i>Save All AI Settings

</button>

</form>

</div>

</div>

</div>

</div>

</div>

<script>

const API\_BASE = ''; // Assumed base path for API calls

let currentConvId = null;

// --- CRITICAL CHANGE: ENFORCE PRINCIPAL/SUPER ADMIN ACCESS ---

const MOCK\_USER\_ID = 'principal\_001';

const MOCK\_USER\_NAME = 'Principal Sharma';

const MOCK\_USER\_ROLE = 'principal'; // Only 'principal' or 'super\_admin' allowed

const ALLOWED\_ROLES = ['principal', 'super\_admin'];

// --- Core Functions ---

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

const colors = { success: 'bg-green-500', error: 'bg-red-500', info: 'bg-blue-500', warning: 'bg-yellow-500' };

const icons = { success: 'fas fa-check-circle', error: 'fas fa-exclamation-circle', info: 'fas fa-info-circle', warning: 'fas fa-exclamation-triangle' };

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

area.className = `fixed top-4 right-4 ${colors[type]} text-white px-6 py-3 rounded-lg shadow-lg z-50`;

area.innerHTML = `<i class="${icons[type]} mr-2"></i>${message}`;

document.body.appendChild(area);

setTimeout(() => area.remove(), 4000);

}

async function fetchAPI(endpoint, method = 'GET', body = null) {

try {

const options = {

method,

headers: { 'Content-Type': 'application/json' },

};

if (body && method !== 'GET') { options.body = JSON.stringify(body); }

const response = await fetch(`${API\_BASE}${endpoint}`, options);

const data = await response.json();

if (!response.ok) {

showNotification(data.error || `API Error: ${response.status}`, 'error');

throw new Error(data.error || 'API request failed');

}

return data;

} catch (error) {

console.error(`Fetch error on ${endpoint}:`, error);

throw error;

}

}

function switchTab(tabName) {

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

btn.classList.remove('active', 'text-white', 'bg-blue-600');

btn.classList.add('text-gray-700');

});

document.querySelector(`[data-tab="${tabName}"]`).classList.add('active', 'text-white', 'bg-blue-600');

document.querySelectorAll('.tab-pane').forEach(pane => {

pane.classList.add('hidden');

});

document.querySelector(`[data-tab-content="${tabName}"]`).classList.remove('hidden');

// Load data when tab changes

if (tabName === 'history') fetchUserHistory();

if (tabName === 'knowledge') fetchKnowledgeBase();

if (tabName === 'analytics') fetchAnalytics();

if (tabName === 'settings') fetchSettings();

if (tabName === 'chat') {

if (!currentConvId) startNewConversation();

else scrollToBottom();

}

}

// --- Chat Functions ---

async function startNewConversation() {

if (!checkRoleAccess(false)) return; // Access already checked, this is now safe

currentConvId = null; // Resetting ID for a clean start

const chatBox = document.getElementById('chatMessages');

chatBox.innerHTML = '';

document.getElementById('currentUserId').textContent = MOCK\_USER\_ID;

// Send a mock greeting to start the conversation (simulates user joining chat)

const initialMessage = `Hello, I need critical oversight data.`;

try {

const data = await fetchAPI('/chat/start', 'POST', {

userId: MOCK\_USER\_ID,

userName: MOCK\_USER\_NAME,

userRole: MOCK\_USER\_ROLE,

initialMessage: initialMessage

});

currentConvId = data.conversation.id;

data.conversation.messages.forEach(msg => addMessageToChat(msg.content, msg.role));

showNotification('Oversight conversation started.', 'success');

scrollToBottom();

} catch (error) {

console.error("Failed to start chat:", error);

addMessageToChat("Error: Could not connect to Vipu AI for oversight data.", 'assistant');

}

}

async function sendMessage() {

const input = document.getElementById('chatInput');

const message = input.value.trim();

if (!message) return;

if (!currentConvId) {

return showNotification("Please wait for the conversation to initialize or refresh the page.", 'warning');

}

addMessageToChat(message, 'user');

input.value = '';

try {

const data = await fetchAPI(`/chat/${currentConvId}/message`, 'POST', { message });

const assistantMessage = data.assistantMessage;

addMessageToChat(assistantMessage.content, assistantMessage.role, assistantMessage.id, assistantMessage.confidence);

scrollToBottom();

} catch (error) {

addMessageToChat("Sorry, I encountered an error. Please try again.", 'assistant');

}

}

function addMessageToChat(text, role, msgId = null, confidence = null) {

const chatBox = document.getElementById('chatMessages');

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

messageDiv.className = `chat-message ${role}`;

let content = `<strong>${role === 'user' ? MOCK\_USER\_NAME : 'Vipu AI'}:</strong> ${text}`;

if (role === 'assistant' && confidence) {

content += `<div class="text-xs mt-1 opacity-75">Confidence: ${confidence}%</div>`;

content += `<div class="text-xs mt-1 space-x-2">

<button class="text-green-200 hover:text-white" onclick="rateMessage('${msgId}', true)"><i class="fas fa-thumbs-up"></i></button>

<button class="text-red-200 hover:text-white" onclick="rateMessage('${msgId}', false)"><i class="fas fa-thumbs-down"></i></button>

</div>`;

}

messageDiv.innerHTML = content;

chatBox.appendChild(messageDiv);

}

function scrollToBottom() {

const chatBox = document.getElementById('chatMessages');

chatBox.scrollTop = chatBox.scrollHeight;

}

function handleEnter(event) {

if (event.key === 'Enter') sendMessage();

}

async function rateMessage(msgId, helpful) {

try {

await fetchAPI(`/chat/message/${msgId}/feedback`, 'POST', { helpful });

showNotification(helpful ? 'Feedback recorded: Helpful!' : 'Feedback recorded: Not Helpful.', 'info');

} catch {

showNotification('Failed to record feedback.', 'error');

}

}

function quickHelp(type) {

let prompt = '';

if (type === 'defaulters') prompt = 'Provide the list of current fee defaulters and their total outstanding amount.';

if (type === 'exam\_schedule') prompt = 'Show the final approved seating chart and invigilator duties for the upcoming exam.';

if (type === 'event\_status') prompt = 'List all high-priority events happening today and their status.';

if (prompt) {

document.getElementById('chatInput').value = prompt;

sendMessage();

}

}

// --- Role Restriction and Initialization ---

function checkRoleAccess(notify = false) {

if (ALLOWED\_ROLES.includes(MOCK\_USER\_ROLE)) {

return true;

}

// Logic to display denial message now handled in DOMContentLoaded

return false;

}

// --- History Functions ---

async function fetchUserHistory() {

if (!checkRoleAccess(false)) return;

// Mock implementation for demonstration

const list = document.getElementById('conversationHistoryList');

list.innerHTML = '<p class="text-gray-500">History loading simulated...</p>';

}

function loadConversation(convId) {

if (!checkRoleAccess(false)) return;

// Mock implementation

showNotification(`Simulating load of conversation ${convId}.`, 'info');

switchTab('chat');

}

// --- Knowledge Base Functions ---

async function fetchKnowledgeBase() {

if (!checkRoleAccess(false)) return;

// Mock implementation for demonstration

const list = document.getElementById('popularQuestionsList');

list.innerHTML = '<p class="text-gray-500">Knowledge Base simulated to load based on management topics.</p>';

}

function searchKnowledge() {

if (!checkRoleAccess(false)) return;

// Mock implementation

showNotification(`Simulating search...`, 'info');

}

// --- Analytics Functions ---

async function fetchAnalytics() {

if (!checkRoleAccess(false)) return;

// Mock implementation for demonstration

document.getElementById('analyticsDashboard').innerHTML = '<p class="text-center text-gray-500">Analytics data simulated.</p>';

}

// --- Settings Functions ---

async function fetchSettings() {

if (!checkRoleAccess(false)) return;

// Mock implementation for demonstration

document.getElementById('responseStyle').value = 'formal';

}

async function updateSettings(event) {

event.preventDefault();

if (!checkRoleAccess(false)) return;

showNotification('Settings updated successfully (Simulated)', 'success');

}

// --- Initialization ---

document.addEventListener('DOMContentLoaded', () => {

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

// Check role and show content if authorized

if (checkRoleAccess(false)) {

contentWrapper.style.display = 'block';

startNewConversation();

switchTab('chat');

} else {

// Display denial message ONLY within this module's container

contentWrapper.innerHTML = `

<div class="bg-red-100 border border-red-400 text-red-700 px-4 py-6 rounded relative mt-8 mx-auto max-w-lg shadow-xl text-center">

<strong class="font-bold text-xl">🚫 Access Restricted!</strong>

<p class="mt-3">This Vipu AI Oversight Assistant is strictly limited to Principal and Super Admin dashboards for critical system oversight.</p>

<p class="text-sm mt-2">Your current simulated role is: <span class="font-bold">${MOCK\_USER\_ROLE.toUpperCase()}</span>. Access to other staff modules is unaffected.</p>

</div>

`;

contentWrapper.style.display = 'block';

}

});

</script>

</body>

</html>

25. client/dist/modules/Vipudev.ai.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>VipuDev AI - Super Admin Assistant | SmartGenEduX</title>

<link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css" rel="stylesheet">

<style>

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

}

body {

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);

min-height: 100vh;

padding: 20px;

}

.container {

max-width: 1200px;

margin: 0 auto;

background: white;

border-radius: 15px;

box-shadow: 0 10px 30px rgba(0,0,0,0.3);

overflow: hidden;

}

.header {

background: linear-gradient(135deg, #2c3e50, #3498db);

color: white;

padding: 30px;

text-align: center;

}

.header h1 {

font-size: 2.5em;

margin-bottom: 10px;

display: flex;

align-items: center;

justify-content: center;

gap: 15px;

}

.vipudev-logo {

width: 60px;

height: 60px;

border-radius: 50%;

border: 3px solid rgba(255,255,255,0.3);

display: flex; /\* Added for i element centering \*/

align-items: center;

justify-content: center;

font-size: 2em; /\* Added for icon size \*/

}

.header p {

font-size: 1.2em;

opacity: 0.9;

}

.main-content {

padding: 40px;

}

.ai-dashboard {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(300px, 1fr));

gap: 30px;

margin-bottom: 40px;

}

.ai-card {

background: linear-gradient(135deg, #f8f9fa, #e9ecef);

border-radius: 15px;

padding: 25px;

border-left: 5px solid #3498db;

transition: all 0.3s ease;

position: relative;

overflow: hidden;

}

.ai-card:hover {

transform: translateY(-5px);

box-shadow: 0 15px 35px rgba(0,0,0,0.1);

}

.ai-card::before {

content: '';

position: absolute;

top: 0;

right: 0;

width: 100px;

height: 100px;

background: linear-gradient(45deg, rgba(52, 152, 219, 0.1), transparent);

border-radius: 50%;

transform: translate(30px, -30px);

}

.ai-card h3 {

color: #2c3e50;

margin-bottom: 15px;

font-size: 1.3em;

display: flex;

align-items: center;

gap: 10px;

}

.ai-card p {

color: #666;

margin-bottom: 20px;

line-height: 1.6;

}

.btn {

background: linear-gradient(135deg, #3498db, #2980b9);

color: white;

border: none;

padding: 12px 24px;

border-radius: 8px;

cursor: pointer;

font-size: 1em;

transition: all 0.3s ease;

display: inline-flex;

align-items: center;

gap: 8px;

}

.btn:hover {

background: linear-gradient(135deg, #2980b9, #21618c);

transform: translateY(-2px);

}

.btn-success { background: linear-gradient(135deg, #27ae60, #229954); }

.btn-warning { background: linear-gradient(135deg, #f39c12, #e67e22); }

.btn-danger { background: linear-gradient(135deg, #e74c3c, #c0392b); }

.analytics-section {

background: #f8f9fa;

border-radius: 15px;

padding: 30px;

margin-top: 30px;

}

.analytics-grid {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));

gap: 20px;

margin-top: 20px;

}

.metric-card {

background: white;

padding: 20px;

border-radius: 10px;

text-align: center;

box-shadow: 0 5px 15px rgba(0,0,0,0.1);

}

.metric-value {

font-size: 2.5em;

font-weight: bold;

color: #3498db;

margin-bottom: 10px;

}

.metric-label {

color: #666;

font-size: 1.1em;

}

.chat-interface {

background: white;

border-radius: 15px;

padding: 30px;

margin-top: 30px;

border: 2px solid #e9ecef;

}

.chat-messages {

height: 300px;

overflow-y: auto;

border: 1px solid #dee2e6;

border-radius: 10px;

padding: 15px;

margin-bottom: 20px;

background: #f8f9fa;

}

.message {

margin-bottom: 15px;

padding: 10px 15px;

border-radius: 10px;

max-width: 80%;

}

.message.ai {

background: linear-gradient(135deg, #3498db, #2980b9);

color: white;

margin-left: auto;

}

.message.user {

background: #e9ecef;

color: #333;

}

.chat-input {

display: flex;

gap: 10px;

}

.chat-input input {

flex: 1;

padding: 12px;

border: 2px solid #dee2e6;

border-radius: 8px;

font-size: 1em;

}

.back-btn {

position: fixed;

top: 20px;

left: 20px;

background: rgba(255, 255, 255, 0.9);

color: #333;

padding: 12px 20px;

border-radius: 25px;

text-decoration: none;

font-weight: 600;

box-shadow: 0 5px 15px rgba(0,0,0,0.2);

transition: all 0.3s ease;

z-index: 1000;

}

.back-btn:hover {

background: white;

transform: translateY(-2px);

}

@media (max-width: 768px) {

.container { margin: 10px; }

.header { padding: 20px; }

.header h1 { font-size: 2em; }

.main-content { padding: 20px; }

.ai-dashboard { grid-template-columns: 1fr; }

}

</style>

</head>

<body>

<a href="../index.html" class="back-btn">

<i class="fas fa-arrow-left"></i> Back to Dashboard

</a>

<div class="container">

<div class="header">

<h1>

<i class="fas fa-microchip vipudev-logo"></i>

VipuDev AI

<i class="fas fa-code"></i>

</h1>

<p>Advanced AI Coding Assistant - Permission-Based Access | Mid to Advanced Level</p>

</div>

<div class="main-content">

<!-- Permission Gate -->

<div id="permissionGate" class="ai-card" style="margin-bottom: 30px; border-left-color: #e74c3c;">

<h3><i class="fas fa-shield-alt"></i> Access Control</h3>

<p>VipuDev AI requires your explicit permission for each coding operation. Enter authorization code to proceed. \*\*(Demo Code: VIPU2025)\*\*</p>

<div style="display: flex; gap: 10px; margin-top: 15px;">

<input type="password" id="authCode" placeholder="Enter authorization code" style="flex: 1; padding: 10px; border: 2px solid #dee2e6; border-radius: 5px;">

<button class="btn" onclick="requestPermission()">

<i class="fas fa-unlock"></i> Grant Access

</button>

</div>

</div>

<!-- AI Coding Features Dashboard -->

<div class="ai-dashboard" id="codingFeatures" style="display: none;">

<div class="ai-card">

<h3><i class="fas fa-code"></i> Code Generation</h3>

<p>Generate production-ready code for SmartGenEduX modules with best practices and security.</p>

<button class="btn" onclick="generateCode()">

<i class="fas fa-laptop-code"></i> Generate Code

</button>

</div>

<div class="ai-card">

<h3><i class="fas fa-bug"></i> Code Analysis & Debug</h3>

<p>Advanced code review, bug detection, and optimization recommendations.</p>

<button class="btn btn-warning" onclick="analyzeCode()">

<i class="fas fa-search"></i> Analyze Code

</button>

</div>

<div class="ai-card">

<h3><i class="fas fa-database"></i> Database Schema</h3>

<p>Generate optimized database schemas and migration scripts for new features.</p>

<button class="btn btn-success" onclick="generateSchema()">

<i class="fas fa-table"></i> Create Schema

</button>

</div>

<div class="ai-card">

<h3><i class="fas fa-rocket"></i> API Development</h3>

<p>Create robust REST APIs with authentication, validation, and error handling.</p>

<button class="btn" onclick="generateAPI()">

<i class="fas fa-link"></i> Build API

</button>

</div>

<div class="ai-card">

<h3><i class="fas fa-shield-check"></i> Security Audit</h3>

<p>Comprehensive security analysis and vulnerability assessment of your codebase.</p>

<button class="btn btn-danger" onclick="securityAudit()">

<i class="fas fa-lock"></i> Security Scan

</button>

</div>

<div class="ai-card">

<h3><i class="fas fa-magic"></i> AI Optimization</h3>

<p>Advanced AI algorithms for performance optimization and intelligent features.</p>

<button class="btn btn-success" onclick="aiOptimization()">

<i class="fas fa-brain"></i> AI Enhance

</button>

</div>

<div class="ai-card">

<h3><i class="fas fa-cloud"></i> Deployment Automation</h3>

<p>Automated deployment pipelines with CI/CD, monitoring, and rollback capabilities.</p>

<button class="btn" onclick="deploymentSetup()">

<i class="fas fa-upload"></i> Setup Deploy

</button>

</div>

<div class="ai-card">

<h3><i class="fas fa-cogs"></i> Module Generator</h3>

<p>Generate complete new modules for SmartGenEduX with full functionality.</p>

<button class="btn btn-warning" onclick="generateModule()">

<i class="fas fa-puzzle-piece"></i> Create Module

</button>

</div>

</div>

<!-- Code Editor Section -->

<div class="analytics-section" id="codeEditor" style="display: none;">

<h2><i class="fas fa-code"></i> Advanced Code Editor</h2>

<div style="display: grid; grid-template-columns: 1fr 1fr; gap: 20px; margin-top: 20px;">

<div>

<h4>Language Selection</h4>

<select id="languageSelect" style="width: 100%; padding: 10px; margin-bottom: 15px; border-radius: 5px;">

<option value="javascript">JavaScript</option>

<option value="typescript">TypeScript</option>

<option value="python">Python</option>

<option value="html">HTML</option>

<option value="css">CSS</option>

<option value="sql">SQL</option>

<option value="php">PHP</option>

<option value="java">Java</option>

</select>

<textarea id="codeInput" placeholder="Enter your code here or describe what you want me to build..." style="width: 100%; height: 300px; padding: 15px; font-family: 'Courier New', monospace; border: 2px solid #dee2e6; border-radius: 8px; resize: vertical;"></textarea>

<div style="margin-top: 10px;">

<button class="btn" onclick="analyzeInputCode()">

<i class="fas fa-search"></i> Analyze Code

</button>

<button class="btn btn-success" onclick="improveCode()">

<i class="fas fa-magic"></i> Improve Code

</button>

<button class="btn btn-warning" onclick="generateTests()">

<i class="fas fa-vial"></i> Generate Tests

</button>

</div>

</div>

<div>

<h4>AI Generated Output</h4>

<div id="codeOutput" style="background: #f8f9fa; border: 2px solid #dee2e6; border-radius: 8px; padding: 15px; height: 350px; overflow-y: auto; font-family: 'Courier New', monospace; white-space: pre-wrap;">

Ready to generate code... Enter your requirements and click a button to proceed.

</div>

<div style="margin-top: 10px;">

<button class="btn" onclick="copyCode()">

<i class="fas fa-copy"></i> Copy Code

</button>

<button class="btn btn-success" onclick="downloadCode()">

<i class="fas fa-download"></i> Download

</button>

<button class="btn btn-danger" onclick="clearOutput()">

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

</button>

</div>

</div>

</div>

</div>

<!-- Development Analytics -->

<div class="analytics-section" id="devAnalytics" style="display: none;">

<h2><i class="fas fa-chart-line"></i> Development Analytics</h2>

<div class="analytics-grid">

<div class="metric-card">

<div class="metric-value">19</div>

<div class="metric-label">Active Modules</div>

</div>

<div class="metric-card">

<div class="metric-value">247</div>

<div class="metric-label">Functions Created</div>

</div>

<div class="metric-card">

<div class="metric-value">15.8K</div>

<div class="metric-label">Lines of Code</div>

</div>

<div class="metric-card">

<div class="metric-value">98.3%</div>

<div class="metric-label">Code Quality</div>

</div>

<div class="metric-card">

<div class="metric-value">23</div>

<div class="metric-label">AI Optimizations</div>

</div>

<div class="metric-card">

<div class="metric-value">Zero</div>

<div class="metric-label">Security Issues</div>

</div>

</div>

<div style="margin-top: 30px;">

<h3>Recent AI Operations</h3>

<div style="background: white; border-radius: 10px; padding: 20px; margin-top: 15px;">

<div style="display: flex; justify-content: space-between; align-items: center; padding: 10px 0; border-bottom: 1px solid #eee;">

<span><i class="fas fa-code text-blue-500"></i> Generated Fee Management API</span>

<span style="color: #666;">2 hours ago</span>

</div>

<div style="display: flex; justify-content: space-between; align-items: center; padding: 10px 0; border-bottom: 1px solid #eee;">

<span><i class="fas fa-shield-check text-green-500"></i> Security Audit Completed</span>

<span style="color: #666;">5 hours ago</span>

</div>

<div style="display: flex; justify-content: space-between; align-items: center; padding: 10px 0; border-bottom: 1px solid #eee;">

<span><i class="fas fa-database text-purple-500"></i> Optimized Database Schema</span>

<span style="color: #666;">1 day ago</span>

</div>

<div style="display: flex; justify-content: space-between; align-items: center; padding: 10px 0;">

<span><i class="fas fa-magic text-orange-500"></i> AI Enhanced Student Module</span>

<span style="color: #666;">2 days ago</span>

</div>

</div>

</div>

<div class="chat-interface">

<h2><i class="fas fa-comments"></i> Chat with VipuDev AI</h2>

<div class="chat-messages" id="chatMessages">

<div class="message ai">

<strong>VipuDev AI:</strong> Hello! I'm your advanced AI assistant. I can help you with revenue analysis, performance optimization, user insights, and much more. What would you like to explore today?

</div>

</div>

<div class="chat-input">

<input type="text" id="chatInput" placeholder="Ask me anything about your platform..." onkeypress="handleEnter(event)">

<button class="btn" onclick="sendMessage()">

<i class="fas fa-paper-plane"></i> Send

</button>

</div>

</div>

</div>

</div>

</div>

<script>

let isAccessGranted = false;

const AUTHORIZATION\_CODE = "VIPU2025"; // Demo authorization code

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

const colors = {

success: 'bg-green-500',

error: 'bg-red-500',

info: 'bg-blue-500',

warning: 'bg-yellow-500'

};

const icons = {

success: 'fas fa-check-circle',

error: 'fas fa-exclamation-circle',

info: 'fas fa-info-circle',

warning: 'fas fa-exclamation-triangle'

};

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

messageDiv.className = `fixed top-4 right-4 ${colors[type]} text-white px-6 py-3 rounded-lg shadow-lg z-50`;

messageDiv.innerHTML = `<i class="${icons[type]} mr-2"></i>${message}`;

document.body.appendChild(messageDiv);

setTimeout(() => {

messageDiv.remove();

}, 4000);

}

function requestPermission() {

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

if (authCode === AUTHORIZATION\_CODE) {

isAccessGranted = true;

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

document.getElementById('codingFeatures').style.display = 'grid';

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

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

showMessage('Access granted! VipuDev AI coding features now available.', 'success');

} else {

showMessage('Invalid authorization code. Access denied.', 'error');

document.getElementById('authCode').value = '';

}

}

function requirePermission(action) {

if (!isAccessGranted) {

showMessage('Permission required! Enter authorization code first.', 'warning');

return false;

}

return true;

}

// Advanced Coding Functions

function generateCode() {

if (!requirePermission('generateCode')) return;

showMessage('AI is generating production-ready code with best practices...', 'info');

setTimeout(() => {

const codeOutput = `// Generated SmartGenEduX Module - Student Analytics

class StudentAnalytics {

constructor(databaseConnection) {

this.db = databaseConnection;

this.aiModel = new AIProcessor();

}

async analyzePerformance(studentId) {

try {

const studentData = await this.db.getStudentData(studentId);

const performanceMetrics = this.aiModel.processData(studentData);

return {

academicScore: performanceMetrics.calculateAcademicScore(),

behaviorPattern: performanceMetrics.getBehaviorInsights(),

recommendations: this.generateRecommendations(performanceMetrics),

riskFactors: this.identifyRiskFactors(performanceMetrics)

};

} catch (error) {

console.error('Analytics error:', error);

throw new Error('Failed to analyze student performance');

}

}

generateRecommendations(metrics) {

const recommendations = [];

if (metrics.academicScore < 70) {

recommendations.push('Schedule additional tutoring sessions');

recommendations.push('Implement personalized learning plan');

}

if (metrics.attendanceRate < 85) {

recommendations.push('Contact parents for attendance improvement');

}

return recommendations;

}

}

module.exports = StudentAnalytics;`;

document.getElementById('codeOutput').textContent = codeOutput;

showMessage('Code generation complete! Advanced student analytics module created.', 'success');

}, 3000);

}

function analyzeCode() {

if (!requirePermission('analyzeCode')) return;

const inputCode = document.getElementById('codeInput').value;

if (!inputCode.trim()) {

showMessage('Please enter code to analyze.', 'warning');

return;

}

showMessage('AI is performing deep code analysis...', 'info');

setTimeout(() => {

const analysis = `CODE ANALYSIS REPORT

===================

✅ STRENGTHS:

- Clean variable naming conventions

- Proper error handling implementation

- Good function separation and modularity

- Security considerations present

⚠️ IMPROVEMENTS:

- Add input validation for user data

- Implement rate limiting for API calls

- Consider adding unit tests

- Add JSDoc documentation

🔒 SECURITY:

- No SQL injection vulnerabilities found

- Proper authentication checks in place

- Sanitized user inputs detected

- Secure data handling patterns used

📊 PERFORMANCE:

- Efficient database queries

- Proper async/await usage

- Memory usage: Optimal

- Execution time: 23ms average

💡 RECOMMENDATIONS:

1. Add TypeScript for better type safety

2. Implement caching for frequently accessed data

3. Add monitoring and logging

4. Consider microservices architecture

OVERALL SCORE: 94/100 (Excellent)`;

document.getElementById('codeOutput').textContent = analysis;

showMessage('Code analysis complete! Found optimizations and security recommendations.', 'success');

}, 4000);

}

function generateSchema() {

if (!requirePermission('generateSchema')) return;

showMessage('Generating optimized database schema...', 'info');

setTimeout(() => {

const schema = `-- SmartGenEduX Advanced Analytics Schema

-- Generated by VipuDev AI

CREATE TABLE student\_analytics (

id SERIAL PRIMARY KEY,

student\_id VARCHAR(50) NOT NULL,

school\_id VARCHAR(50) NOT NULL,

academic\_score DECIMAL(5,2),

behavior\_score DECIMAL(5,2),

attendance\_rate DECIMAL(5,2),

risk\_level ENUM('low', 'medium', 'high'),

ai\_recommendations JSON,

performance\_trends JSON,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

INDEX idx\_student\_school (student\_id, school\_id),

INDEX idx\_risk\_level (risk\_level),

INDEX idx\_created\_at (created\_at),

FOREIGN KEY (student\_id) REFERENCES students(id) ON DELETE CASCADE,

FOREIGN KEY (school\_id) REFERENCES schools(id) ON DELETE CASCADE

);

CREATE TABLE ai\_insights (

id SERIAL PRIMARY KEY,

target\_type ENUM('student', 'teacher', 'class', 'school'),

target\_id VARCHAR(50) NOT NULL,

insight\_type VARCHAR(100) NOT NULL,

insight\_data JSON NOT NULL,

confidence\_score DECIMAL(3,2),

action\_required BOOLEAN DEFAULT FALSE,

processed\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

INDEX idx\_target (target\_type, target\_id),

INDEX idx\_insight\_type (insight\_type),

INDEX idx\_confidence (confidence\_score),

INDEX idx\_action\_required (action\_required)

);

-- Trigger for automatic analytics updates

DELIMITER //

CREATE TRIGGER update\_student\_analytics

AFTER INSERT ON attendance

FOR EACH ROW

BEGIN

INSERT INTO student\_analytics (student\_id, school\_id)

VALUES (NEW.student\_id, NEW.school\_id)

ON DUPLICATE KEY UPDATE updated\_at = CURRENT\_TIMESTAMP;

END//

DELIMITER ;`;

document.getElementById('codeOutput').textContent = schema;

showMessage('Database schema generated! Optimized for analytics and AI insights.', 'success');

}, 3500);

}

function generateAPI() {

if (!requirePermission('generateAPI')) return;

showMessage('Building robust REST API with authentication...', 'info');

setTimeout(() => {

const apiCode = `// SmartGenEduX API - Student Analytics Endpoint

// Generated by VipuDev AI with security and validation

const express = require('express');

const { body, validationResult } = require('express-validator');

const rateLimit = require('express-rate-limit');

const helmet = require('helmet');

const jwt = require('jsonwebtoken');

const router = express.Router();

// Security middleware

router.use(helmet());

router.use(rateLimit({

windowMs: 15 \* 60 \* 1000, // 15 minutes

max: 100 // limit each IP to 100 requests per windowMs

}));

// Authentication middleware

const authenticateToken = (req, res, next) => {

const authHeader = req.headers['authorization'];

const token = authHeader && authHeader.split(' ')[1];

if (!token) {

return res.sendStatus(401);

}

jwt.verify(token, process.env.JWT\_SECRET, (err, user) => {

if (err) return res.sendStatus(403);

req.user = user;

next();

});

};

// Get student analytics

router.get('/analytics/:studentId',

authenticateToken,

async (req, res) => {

try {

const { studentId } = req.params;

// Validate user has access to this student

const hasAccess = await checkStudentAccess(req.user.id, studentId);

if (!hasAccess) {

return res.status(403).json({ error: 'Access denied' });

}

const analytics = await StudentAnalytics.getAnalytics(studentId);

res.json({

success: true,

data: analytics,

timestamp: new Date().toISOString()

});

} catch (error) {

console.error('API Error:', error);

res.status(500).json({

success: false,

error: 'Internal server error'

});

}

}

);

// Create analytics report

router.post('/analytics/generate',

authenticateToken,

[

body('studentIds').isArray().withMessage('Student IDs must be an array'),

body('reportType').isIn(['performance', 'behavior', 'attendance']),

body('dateRange.start').isISO8601().withMessage('Invalid start date'),

body('dateRange.end').isISO8601().withMessage('Invalid end date')

],

async (req, res) => {

const errors = validationResult(req);

if (!errors.isEmpty()) {

return res.status(400).json({ errors: errors.array() });

}

try {

const { studentIds, reportType, dateRange } = req.body;

const report = await AnalyticsGenerator.generateReport({

studentIds,

reportType,

dateRange,

userId: req.user.id

});

res.json({

success: true,

reportId: report.id,

downloadUrl: \`/api/reports/download/\${report.id}\`,

estimatedTime: '2-3 minutes'

});

} catch (error) {

console.error('Report generation error:', error);

res.status(500).json({

success: false,

error: 'Failed to generate report'

});

}

}

);

module.exports = router;`;

document.getElementById('codeOutput').textContent = apiCode;

showMessage('API generated! Complete with authentication, validation, and rate limiting.', 'success');

}, 4000);

}

function securityAudit() {

if (!requirePermission('securityAudit')) return;

showMessage('Running comprehensive security audit...', 'info');

setTimeout(() => {

const auditReport = `SECURITY AUDIT REPORT - SmartGenEduX Platform

=============================================

🔒 AUTHENTICATION & AUTHORIZATION

✅ JWT token implementation secure

✅ Password hashing with bcrypt (12 rounds)

✅ Role-based access control implemented

✅ Session management secure

⚠️ Consider implementing 2FA for admin accounts

🛡️ DATA PROTECTION

✅ SQL injection prevention implemented

✅ XSS protection with content security policy

✅ CSRF tokens in place

✅ Input validation and sanitization

✅ Encrypted database connections

🌐 NETWORK SECURITY

✅ HTTPS enforced in production

✅ Rate limiting implemented

✅ Security headers configured

✅ CORS properly configured

⚠️ Consider DDoS protection for production

📊 VULNERABILITY SCAN

✅ No critical vulnerabilities found

✅ Dependencies up to date

✅ No exposed sensitive data

✅ Proper error handling (no info leakage)

🔐 ENCRYPTION

✅ Data at rest encrypted

✅ Data in transit encrypted

✅ API keys properly stored

✅ Environment variables secured

📋 COMPLIANCE

✅ GDPR compliance measures

✅ Data retention policies

✅ Audit logging implemented

✅ Privacy controls in place

OVERALL SECURITY SCORE: 96/100

RISK LEVEL: VERY LOW

RECOMMENDATIONS:

1. Implement 2FA for super admin accounts

2. Add DDoS protection service

3. Regular security training for staff

4. Quarterly penetration testing

Last scan: ${new Date().toLocaleString()}`;

document.getElementById('codeOutput').textContent = auditReport;

showMessage('Security audit complete! Platform security score: 96/100 - Very Low Risk.', 'success');

}, 5000);

}

function aiOptimization() {

if (!requirePermission('aiOptimization')) return;

showMessage('Implementing advanced AI optimizations...', 'info');

setTimeout(() => {

showMessage('AI optimization complete! Performance improved by 35%, predictive accuracy increased to 94%.', 'success');

}, 4000);

}

function deploymentSetup() {

if (!requirePermission('deploymentSetup')) return;

showMessage('Setting up automated deployment pipeline...', 'info');

setTimeout(() => {

showMessage('Deployment automation configured! CI/CD pipeline ready with automatic testing and rollback.', 'success');

}, 3500);

}

function generateModule() {

if (!requirePermission('generateModule')) return;

showMessage('Generating complete new module with full functionality...', 'info');

setTimeout(() => {

showMessage('Module generation complete! New "Student Counseling" module created with AI recommendations.', 'success');

}, 5000);

}

// Code Editor Functions

function analyzeInputCode() {

if (!requirePermission('analyzeInputCode')) return;

analyzeCode();

}

function improveCode() {

if (!requirePermission('improveCode')) return;

const inputCode = document.getElementById('codeInput').value;

if (!inputCode.trim()) {

showMessage('Please enter code to improve.', 'warning');

return;

}

showMessage('AI is improving your code with best practices...', 'info');

setTimeout(() => {

const improvedCode = `// IMPROVED CODE with AI optimizations

// Original code enhanced with error handling, type safety, and performance

${inputCode}

/\* AI IMPROVEMENTS APPLIED:

1. Added proper error handling

2. Implemented input validation

3. Added performance optimizations

4. Enhanced security measures

5. Improved code documentation

6. Added logging for debugging

7. Optimized database queries

8. Implemented caching where applicable

\*/`;

document.getElementById('codeOutput').textContent = improvedCode;

showMessage('Code improvement complete! Applied 8 optimizations and best practices.', 'success');

}, 3000);

}

function generateTests() {

if (!requirePermission('generateTests')) return;

showMessage('Generating comprehensive test suite...', 'info');

setTimeout(() => {

const tests = `// AUTOMATED TEST SUITE

// Generated by VipuDev AI

const { expect } = require('chai');

const request = require('supertest');

const app = require('../app');

describe('Student Analytics API', () => {

describe('GET /api/analytics/:studentId', () => {

it('should return analytics for valid student', async () => {

const response = await request(app)

.get('/api/analytics/STU001')

.set('Authorization', 'Bearer validToken')

.expect(200);

expect(response.body.success).to.be.true;

expect(response.body.data).to.have.property('academicScore');

});

it('should return 401 for unauthorized access', async () => {

await request(app)

.get('/api/analytics/STU001')

.expect(401);

});

it('should return 403 for insufficient permissions', async () => {

await request(app)

.get('/api/analytics/STU001')

.set('Authorization', 'Bearer limitedToken')

.expect(403);

});

});

describe('POST /api/analytics/generate', () => {

it('should generate report for valid data', async () => {

const reportData = {

studentIds: ['STU001', 'STU002'],

reportType: 'performance',

dateRange: {

start: '2025-01-01',

end: '2025-01-31'

}

};

const response = await request(app)

.post('/api/analytics/generate')

.set('Authorization', 'Bearer validToken')

.send(reportData)

.expect(200);

expect(response.body.success).to.be.true;

expect(response.body.reportId).to.exist;

});

});

});`;

document.getElementById('codeOutput').textContent = tests;

showMessage('Test suite generated! Complete with unit tests, integration tests, and edge cases.', 'success');

}, 3000);

}

function copyCode() {

const codeOutput = document.getElementById('codeOutput').textContent;

document.execCommand('copy');

showMessage('Code copied to clipboard!', 'success');

}

function downloadCode() {

const codeOutput = document.getElementById('codeOutput').textContent;

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

const url = URL.createObjectURL(blob);

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

a.href = url;

a.download = 'vipudev-ai-generated-code.txt';

a.click();

URL.revokeObjectURL(url);

showMessage('Code downloaded successfully!', 'success');

}

function clearOutput() {

document.getElementById('codeOutput').textContent = 'Ready to generate code... Enter your requirements and click a button to proceed.';

showMessage('Output cleared.', 'info');

}

function sendMessage() {

const input = document.getElementById('chatInput');

const message = input.value.trim();

if (message) {

addMessage(message, 'user');

input.value = '';

// Simulate AI response

setTimeout(() => {

const responses = [

"Based on the data analysis, I recommend focusing on the Fee Management module optimization for better revenue collection.",

"The attendance patterns show that Monday mornings have the highest absenteeism. Consider implementing targeted interventions.",

"Your platform usage analytics indicate that mobile access has increased by 340%. I suggest prioritizing mobile optimization.",

"Revenue forecasting shows a potential 25% growth opportunity in the premium module subscriptions.",

"System performance metrics indicate optimal operation. All modules are functioning within expected parameters."

];

const randomResponse = responses[Math.floor(Math.random() \* responses.length)];

addMessage(randomResponse, 'ai');

}, 1500);

}

}

function addMessage(text, sender) {

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

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

messageDiv.className = `message ${sender}`;

if (sender === 'ai') {

messageDiv.innerHTML = `<strong>VipuDev AI:</strong> ${text}`;

} else {

messageDiv.innerHTML = `<strong>You:</strong> ${text}`;

}

chatMessages.appendChild(messageDiv);

chatMessages.scrollTop = chatMessages.scrollHeight;

}

function handleEnter(event) {

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

sendMessage();

}

}

// Initialize with welcome message

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

showMessage('VipuDev AI Assistant activated! All systems ready.', 'success');

});

</script>

</body>

</html>

26. client.dist/modules/Voice-to-text.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Voice to Text Converter - SmartGenEduX</title>

<link href="https://cdn.jsdelivr.net/npm/tailwindcss@2.2.19/dist/tailwind.min.css" rel="stylesheet">

<link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css" rel="stylesheet">

</head>

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

<div class="container mx-auto px-4 py-8">

<!-- Header -->

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

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

<div class="flex items-center">

<img src="../assets/Voice to Text Converter\_20250621\_154004\_0000\_1751640161627.png" alt="Voice to Text" class="w-12 h-12 mr-4">

<div>

<h1 class="text-2xl font-bold text-gray-800">Voice to Text Converter</h1>

<p class="text-gray-600">AI-powered speech recognition for educational content</p>

</div>

</div>

<button onclick="window.close()" class="bg-gray-500 text-white px-4 py-2 rounded hover:bg-gray-600">

<i class="fas fa-times mr-2"></i>Close

</button>

</div>

</div>

<!-- Quick Stats -->

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

<div class="bg-blue-600 text-white rounded-lg p-6">

<div class="flex items-center">

<i class="fas fa-microphone text-3xl mr-4"></i>

<div>

<div class="text-2xl font-bold">234</div>

<div class="text-blue-100">Audio Files Processed</div>

</div>

</div>

</div>

<div class="bg-green-600 text-white rounded-lg p-6">

<div class="flex items-center">

<i class="fas fa-clock text-3xl mr-4"></i>

<div>

<div class="text-2xl font-bold">47h</div>

<div class="text-green-100">Total Audio Time</div>

</div>

</div>

</div>

<div class="bg-purple-600 text-white rounded-lg p-6">

<div class="flex items-center">

<i class="fas fa-language text-3xl mr-4"></i>

<div>

<div class="text-2xl font-bold">5</div>

<div class="text-purple-100">Languages Supported</div>

</div>

</div>

</div>

<div class="bg-orange-600 text-white rounded-lg p-6">

<div class="flex items-center">

<i class="fas fa-chart-line text-3xl mr-4"></i>

<div>

<div class="text-2xl font-bold">94%</div>

<div class="text-orange-100">Accuracy Rate</div>

</div>

</div>

</div>

</div>

<!-- Recording & Upload Interface -->

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

<!-- Live Recording -->

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

<h2 class="text-xl font-bold mb-4">Live Recording</h2>

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

<div class="text-center">

<div id="recordingStatus" class="w-32 h-32 bg-gray-100 rounded-full flex items-center justify-center mx-auto mb-4">

<i class="fas fa-microphone text-4xl text-gray-400"></i>

</div>

<div id="recordingTime" class="text-2xl font-bold text-gray-800 mb-2">00:00</div>

<div id="statusText" class="text-gray-600">Ready to record</div>

</div>

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

<button id="recordBtn" onclick="toggleRecording()" class="bg-red-600 text-white px-6 py-3 rounded-full hover:bg-red-700">

<i class="fas fa-microphone mr-2"></i>Start Recording

</button>

<button id="pauseBtn" onclick="pauseRecording()" class="bg-yellow-600 text-white px-6 py-3 rounded-full hover:bg-yellow-700" disabled>

<i class="fas fa-pause mr-2"></i>Pause

</button>

<button id="stopBtn" onclick="stopRecording()" class="bg-gray-600 text-white px-6 py-3 rounded-full hover:bg-gray-700" disabled>

<i class="fas fa-stop mr-2"></i>Stop

</button>

</div>

<div class="border-t pt-4">

<h3 class="font-semibold mb-2">Recording Settings</h3>

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

<div>

<label class="block text-sm font-medium mb-1">Language</label>

<select id="language" class="w-full border rounded px-3 py-2 text-sm">

<option value="en-US">English (US)</option>

<option value="en-IN">English (India)</option>

<option value="hi-IN">Hindi</option>

<option value="ta-IN">Tamil</option>

<option value="te-IN">Telugu</option>

</select>

</div>

<div>

<label class="block text-sm font-medium mb-1">Quality</label>

<select id="quality" class="w-full border rounded px-3 py-2 text-sm">

<option value="high">High Quality</option>

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

<option value="low">Low (Fast)</option>

</select>

</div>

</div>

<div class="mt-3">

<label class="flex items-center">

<input type="checkbox" id="autoSave" class="mr-2"> Auto-save transcriptions

</label>

<label class="flex items-center">

<input type="checkbox" id="realTime" class="mr-2"> Real-time transcription

</label>

</div>

</div>

</div>

</div>

<!-- File Upload -->

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

<h2 class="text-xl font-bold mb-4">Upload Audio Files</h2>

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

<div class="border-2 border-dashed border-gray-300 rounded-lg p-6 text-center">

<input type="file" id="audioUpload" accept="audio/\*" multiple class="hidden" onchange="handleAudioUpload(this)">

<label for="audioUpload" class="cursor-pointer">

<i class="fas fa-cloud-upload-alt text-4xl text-gray-400 mb-4"></i>

<p class="text-gray-600 mb-2">Drop audio files here or click to upload</p>

<p class="text-sm text-gray-500">Supports MP3, WAV, M4A, OGG formats</p>

</label>

</div>

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

<!-- Upload queue items will appear here -->

</div>

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

<h3 class="font-semibold text-blue-900 mb-2">Batch Processing</h3>

<p class="text-sm text-blue-700 mb-3">Upload multiple files for automatic batch transcription</p>

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

<button onclick="batchProcess()" class="bg-blue-600 text-white px-4 py-2 rounded hover:bg-blue-700 text-sm">

<i class="fas fa-layer-group mr-2"></i>Process All

</button>

<button onclick="clearQueue()" class="bg-gray-600 text-white px-4 py-2 rounded hover:bg-gray-700 text-sm">

<i class="fas fa-trash mr-2"></i>Clear Queue

</button>

</div>

</div>

</div>

</div>

</div>

<!-- Transcription Results -->

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

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

<h2 class="text-xl font-bold">Transcription Results</h2>

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

<button onclick="exportText()" class="bg-green-600 text-white px-4 py-2 rounded hover:bg-green-700">

<i class="fas fa-download mr-2"></i>Export

</button>

<button onclick="copyToClipboard()" class="bg-blue-600 text-white px-4 py-2 rounded hover:bg-blue-700">

<i class="fas fa-copy mr-2"></i>Copy

</button>

</div>

</div>

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

<div>

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

<h3 class="font-semibold">Live Transcription</h3>

<span id="confidenceScore" class="px-2 py-1 bg-gray-100 text-gray-800 rounded text-sm">Confidence: --</span>

</div>

<div id="liveTranscription" class="bg-gray-50 rounded-lg p-4 h-64 overflow-y-auto">

<p class="text-gray-500 text-center">Start recording to see live transcription...</p>

</div>

</div>

<div>

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

<h3 class="font-semibold">Formatted Text</h3>

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

<button onclick="formatText()" class="text-blue-600 hover:text-blue-800 text-sm">

<i class="fas fa-magic mr-1"></i>Auto Format

</button>

<button onclick="correctText()" class="text-purple-600 hover:text-purple-800 text-sm">

<i class="fas fa-spell-check mr-1"></i>Spell Check

</button>

</div>

</div>

<textarea id="formattedText" class="w-full h-64 border rounded-lg p-4 resize-none" placeholder="Formatted transcription will appear here..."></textarea>

</div>

</div>

</div>

<!-- Recent Transcriptions -->

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

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

<h2 class="text-xl font-bold">Recent Transcriptions</h2>

<button onclick="manageFiles()" class="bg-gray-600 text-white px-4 py-2 rounded hover:bg-gray-700">

<i class="fas fa-folder mr-2"></i>Manage Files

</button>

</div>

<div class="overflow-x-auto">

<table class="w-full text-sm">

<thead class="bg-gray-50">

<tr>

<th class="px-4 py-2 text-left">File Name</th>

<th class="px-4 py-2 text-left">Duration</th>

<th class="px-4 py-2 text-left">Language</th>

<th class="px-4 py-2 text-left">Accuracy</th>

<th class="px-4 py-2 text-left">Word Count</th>

<th class="px-4 py-2 text-left">Date</th>

<th class="px-4 py-2 text-left">Actions</th>

</tr>

</thead>

<tbody>

<tr class="border-t">

<td class="px-4 py-2">Math\_Lecture\_Chapter5.mp3</td>

<td class="px-4 py-2">45:23</td>

<td class="px-4 py-2">English</td>

<td class="px-4 py-2">

<span class="px-2 py-1 bg-green-100 text-green-800 rounded text-xs">96%</span>

</td>

<td class="px-4 py-2">4,567 words</td>

<td class="px-4 py-2">Jan 4, 2025</td>

<td class="px-4 py-2">

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

<button class="text-blue-600 hover:text-blue-800" title="View Text">

<i class="fas fa-eye"></i>

</button>

<button class="text-green-600 hover:text-green-800" title="Download">

<i class="fas fa-download"></i>

</button>

<button class="text-purple-600 hover:text-purple-800" title="Edit">

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

</button>

<button class="text-orange-600 hover:text-orange-800" title="Share">

<i class="fas fa-share"></i>

</button>

</div>

</td>

</tr>

<tr class="border-t">

<td class="px-4 py-2">Science\_Lab\_Experiment.wav</td>

<td class="px-4 py-2">28:15</td>

<td class="px-4 py-2">Hindi</td>

<td class="px-4 py-2">

<span class="px-2 py-1 bg-yellow-100 text-yellow-800 rounded text-xs">92%</span>

</td>

<td class="px-4 py-2">2,834 words</td>

<td class="px-4 py-2">Jan 3, 2025</td>

<td class="px-4 py-2">

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

<button class="text-blue-600 hover:text-blue-800" title="View Text">

<i class="fas fa-eye"></i>

</button>

<button class="text-green-600 hover:text-green-800" title="Download">

<i class="fas fa-download"></i>

</button>

<button class="text-purple-600 hover:text-purple-800" title="Edit">

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

</button>

<button class="text-orange-600 hover:text-orange-800" title="Share">

<i class="fas fa-share"></i>

</button>

</div>

</td>

</tr>

</tbody>

</table>

</div>

</div>

</div>

<script>

let isRecording = false;

let isPaused = false;

let recordingTimer;

let seconds = 0;

let mediaRecorder;

let audioChunks = [];

function toggleRecording() {

if (!isRecording) {

startRecording();

} else {

stopRecording();

}

}

async function startRecording() {

try {

const stream = await navigator.mediaDevices.getUserMedia({ audio: true });

mediaRecorder = new MediaRecorder(stream);

mediaRecorder.ondataavailable = event => {

audioChunks.push(event.data);

};

mediaRecorder.onstop = () => {

const audioBlob = new Blob(audioChunks, { type: 'audio/wav' });

processAudio(audioBlob);

};

mediaRecorder.start();

isRecording = true;

document.getElementById('recordBtn').innerHTML = '<i class="fas fa-stop mr-2"></i>Stop Recording';

document.getElementById('recordBtn').className = 'bg-red-600 text-white px-6 py-3 rounded-full hover:bg-red-700';

document.getElementById('pauseBtn').disabled = false;

document.getElementById('stopBtn').disabled = false;

document.getElementById('recordingStatus').innerHTML = '<i class="fas fa-microphone text-4xl text-red-500 animate-pulse"></i>';

document.getElementById('statusText').textContent = 'Recording...';

startTimer();

if (document.getElementById('realTime').checked) {

startRealTimeTranscription();

}

showMessage('Recording started successfully', 'success');

} catch (error) {

showMessage('Error accessing microphone. Please check permissions.', 'error');

}

}

function pauseRecording() {

if (isPaused) {

mediaRecorder.resume();

isPaused = false;

document.getElementById('pauseBtn').innerHTML = '<i class="fas fa-pause mr-2"></i>Pause';

document.getElementById('statusText').textContent = 'Recording...';

resumeTimer();

} else {

mediaRecorder.pause();

isPaused = true;

document.getElementById('pauseBtn').innerHTML = '<i class="fas fa-play mr-2"></i>Resume';

document.getElementById('statusText').textContent = 'Paused';

pauseTimer();

}

}

function stopRecording() {

if (mediaRecorder && isRecording) {

mediaRecorder.stop();

mediaRecorder.stream.getTracks().forEach(track => track.stop());

isRecording = false;

isPaused = false;

document.getElementById('recordBtn').innerHTML = '<i class="fas fa-microphone mr-2"></i>Start Recording';

document.getElementById('recordBtn').className = 'bg-green-600 text-white px-6 py-3 rounded-full hover:bg-green-700';

document.getElementById('pauseBtn').disabled = true;

document.getElementById('stopBtn').disabled = true;

document.getElementById('recordingStatus').innerHTML = '<i class="fas fa-microphone text-4xl text-gray-400"></i>';

document.getElementById('statusText').textContent = 'Processing...';

stopTimer();

audioChunks = [];

showMessage('Recording stopped. Processing audio...', 'info');

}

}

function startTimer() {

recordingTimer = setInterval(() => {

if (!isPaused) {

seconds++;

updateTimerDisplay();

}

}, 1000);

}

function pauseTimer() {

// Timer continues counting but display shows paused state

}

function resumeTimer() {

// Timer resumes normal operation

}

function stopTimer() {

clearInterval(recordingTimer);

seconds = 0;

updateTimerDisplay();

}

function updateTimerDisplay() {

const minutes = Math.floor(seconds / 60);

const secs = seconds % 60;

document.getElementById('recordingTime').textContent =

`${minutes.toString().padStart(2, '0')}:${secs.toString().padStart(2, '0')}`;

}

function startRealTimeTranscription() {

// Simulate real-time transcription

const liveDiv = document.getElementById('liveTranscription');

const sampleTexts = [

"Today we will discuss the fundamental concepts of mathematics...",

"The Pythagorean theorem states that in a right triangle...",

"Let me explain this concept with a practical example...",

"Can everyone see the diagram on the board clearly?",

"Please take notes as this will be important for your exam..."

];

let textIndex = 0;

const transcriptionInterval = setInterval(() => {

if (!isRecording) {

clearInterval(transcriptionInterval);

return;

}

if (textIndex < sampleTexts.length) {

liveDiv.innerHTML += `<p class="mb-2">${sampleTexts[textIndex]}</p>`;

textIndex++;

liveDiv.scrollTop = liveDiv.scrollHeight;

// Update confidence score

const confidence = Math.floor(Math.random() \* 10) + 85;

document.getElementById('confidenceScore').textContent = `Confidence: ${confidence}%`;

}

}, 3000);

}

function processAudio(audioBlob) {

// Simulate AI processing

setTimeout(() => {

const finalText = `This is a simulated transcription of your audio recording.

The AI has processed the audio and converted it to text with high accuracy.

The transcription includes proper punctuation and formatting.

Key features:

- Automatic speech recognition

- Multi-language support

- Real-time processing

- Export capabilities

Your recording has been successfully processed and is ready for export or further editing.`;

document.getElementById('formattedText').value = finalText;

document.getElementById('statusText').textContent = 'Transcription complete';

showMessage('Audio transcription completed successfully!', 'success');

}, 2000);

}

function handleAudioUpload(input) {

const files = input.files;

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

Array.from(files).forEach(file => {

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

fileItem.className = 'flex items-center justify-between p-3 border rounded';

fileItem.innerHTML = `

<div class="flex items-center">

<i class="fas fa-file-audio text-blue-500 mr-3"></i>

<div>

<div class="font-semibold">${file.name}</div>

<div class="text-sm text-gray-600">${(file.size / 1024 / 1024).toFixed(2)} MB</div>

</div>

</div>

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

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

<button onclick="this.parentElement.parentElement.remove()" class="text-red-600 hover:text-red-800">

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

</button>

</div>

`;

uploadQueue.appendChild(fileItem);

});

showMessage(`${files.length} file(s) added to processing queue`, 'success');

}

function batchProcess() {

const queueItems = document.querySelectorAll('#uploadQueue > div');

if (queueItems.length === 0) {

showMessage('No files in queue to process', 'error');

return;

}

queueItems.forEach(item => {

const statusSpan = item.querySelector('.bg-yellow-100');

statusSpan.className = 'px-2 py-1 bg-blue-100 text-blue-800 rounded text-xs';

statusSpan.textContent = 'Processing...';

setTimeout(() => {

statusSpan.className = 'px-2 py-1 bg-green-100 text-green-800 rounded text-xs';

statusSpan.textContent = 'Complete';

}, Math.random() \* 3000 + 2000);

});

showMessage('Batch processing started for all queued files', 'info');

}

function clearQueue() {

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

showMessage('Processing queue cleared', 'info');

}

function formatText() {

const text = document.getElementById('formattedText').value;

if (!text) {

showMessage('No text to format', 'error');

return;

}

showMessage('Text formatting applied', 'success');

}

function correctText() {

showMessage('Spell check completed. No errors found.', 'success');

}

function exportText() {

const text = document.getElementById('formattedText').value;

if (!text) {

showMessage('No text to export', 'error');

return;

}

showMessage('Text exported successfully', 'success');

}

function copyToClipboard() {

const text = document.getElementById('formattedText').value;

if (!text) {

showMessage('No text to copy', 'error');

return;

}

navigator.clipboard.writeText(text).then(() => {

showMessage('Text copied to clipboard', 'success');

});

}

function manageFiles() {

showMessage('Opening file management interface...', 'info');

}

function showMessage(message, type) {

const colors = {

'success': 'bg-green-600',

'error': 'bg-red-600',

'info': 'bg-blue-600'

};

const icons = {

'success': 'fas fa-check',

'error': 'fas fa-exclamation-triangle',

'info': 'fas fa-info-circle'

};

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

messageDiv.className = `fixed top-4 right-4 ${colors[type]} text-white px-6 py-3 rounded-lg shadow-lg z-50`;

messageDiv.innerHTML = `<i class="${icons[type]} mr-2"></i>${message}`;

document.body.appendChild(messageDiv);

setTimeout(() => {

messageDiv.remove();

}, 3000);

}

</script>

</body>

</html>

27. client/dist/modules/whatsapp\_alert.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Smart WhatsApp Alert Manager & Automation</title>

<!-- Tailwind CSS -->

<link href="https://cdn.jsdelivr.net/npm/tailwindcss@2.2.19/dist/tailwind.min.css" rel="stylesheet">

<!-- Font Awesome Icons -->

<link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css" rel="stylesheet">

<style>

body { background-color: #f8f9fa; font-family: 'Inter', sans-serif; }

.container { max-width: 1200px; margin-top: 20px; }

.card-header { background-color: #25D366; color: white; font-weight: bold; border-radius: 0.5rem 0.5rem 0 0; }

.tab-button.active { background-color: #25D366 !important; color: white !important; }

.section-card { box-shadow: 0 4px 8px rgba(0,0,0,0.05); border-radius: 0.5rem; margin-bottom: 20px; }

.status-badge { padding: 4px 8px; border-radius: 9999px; font-size: 0.75rem; font-weight: 600; }

</style>

</head>

<body>

<div class="container mx-auto">

<h1 class="text-3xl font-extrabold text-center mb-6 text-gray-800 flex items-center justify-center">

<i class="fab fa-whatsapp text-green-500 mr-3"></i>Smart WhatsApp Alert Center

</h1>

<!-- Notification Area -->

<div id="notification-area" class="fixed top-4 right-4 z-50"></div>

<!-- Tab Navigation -->

<div class="bg-white rounded-lg shadow-md mb-4">

<div class="flex border-b border-gray-200">

<button class="tab-button px-4 py-3 text-sm font-medium active flex-1 text-gray-700" onclick="switchTab('templates')" data-tab="templates">

<i class="fas fa-edit mr-2"></i>Template & Broadcast

</button>

<button class="tab-button px-4 py-3 text-sm font-medium flex-1 text-gray-700" onclick="switchTab('automation')" data-tab="automation">

<i class="fas fa-robot mr-2"></i>Automation Rules

</button>

<button class="tab-button px-4 py-3 text-sm font-medium flex-1 text-gray-700" onclick="switchTab('contacts')" data-tab="contacts">

<i class="fas fa-users mr-2"></i>Contact Preferences

</button>

<button class="tab-button px-4 py-3 text-sm font-medium flex-1 text-gray-700" onclick="switchTab('analytics')" data-tab="analytics">

<i class="fas fa-chart-bar mr-2"></i>Performance Analytics

</button>

<button class="tab-button px-4 py-3 text-sm font-medium flex-1 text-gray-700" onclick="switchTab('settings')" data-tab="settings">

<i class="fas fa-cog mr-2"></i>API & Compliance Config

</button>

</div>

</div>

<!-- Tab Content -->

<div id="tabContent">

<!-- 1. Templates & Broadcast (Template CRUD, Send Single/Broadcast) -->

<div data-tab-content="templates" class="tab-pane active">

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

<!-- Template List & Management -->

<div class="lg:col-span-2 section-card bg-white">

<div class="card-header">Template Gallery & Management (8 Templates Loaded)</div>

<div class="p-4">

<div id="templateList" class="space-y-3">

<p class="text-gray-500">Loading templates...</p>

</div>

</div>

</div>

<!-- Send/Broadcast Panel -->

<div class="lg:col-span-1 section-card bg-white">

<div class="card-header bg-blue-600">Send Single Message / Broadcast</div>

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

<h3 class="font-bold text-gray-700">Message Parameters</h3>

<select id="sendTemplateId" class="w-full p-2 border rounded-md text-sm" onchange="displayTemplateVariables()">

<option value="">Select Template...</option>

</select>

<input type="text" id="recipientNumber" placeholder="Recipient Phone (+91...)" class="w-full p-2 border rounded-md text-sm">

<div id="variableInputs" class="space-y-2 border p-2 rounded-md bg-gray-50">

<p class="text-sm text-gray-600 italic">Variables will appear here once a template is selected.</p>

</div>

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

<input type="datetime-local" id="scheduledFor" class="flex-1 p-2 border rounded-md text-sm">

<button onclick="sendSingleMessage(false)" class="bg-yellow-600 text-white px-4 py-2 rounded-md text-sm hover:bg-yellow-700">

<i class="fas fa-clock mr-2"></i>Schedule

</button>

</div>

<button onclick="sendSingleMessage(true)" class="w-full bg-green-600 text-white px-4 py-3 rounded-md hover:bg-green-700">

<i class="fas fa-paper-plane mr-2"></i>Send Now

</button>

</div>

</div>

</div>

</div>

<!-- 2. Automation Rules (CRUD Automation) -->

<div data-tab-content="automation" class="tab-pane hidden">

<div class="section-card bg-white">

<div class="card-header bg-purple-600">Automated Trigger Rules</div>

<div class="p-6">

<h3 class="font-bold text-lg mb-3">Current Active Rules</h3>

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

<p class="text-gray-500">Loading rules...</p>

</div>

</div>

</div>

</div>

<!-- 3. Contact Preferences (Consent, Opt-Out, Language) -->

<div data-tab-content="contacts" class="tab-pane hidden">

<div class="section-card bg-white">

<div class="card-header bg-orange-600">Parent Contact & Consent Management</div>

<div class="p-6">

<div id="contactFilter" class="mb-4 flex space-x-4">

<select id="contactVerifiedFilter" class="p-2 border rounded-md">

<option value="">Verification Status</option>

<option value="true">Verified (WhatsApp)</option>

<option value="false">Unverified</option>

</select>

<select id="contactConsentFilter" class="p-2 border rounded-md">

<option value="">Consent Status</option>

<option value="true">Consent Given</option>

<option value="false">Opted Out / No Consent</option>

</select>

<button onclick="fetchContacts()" class="bg-gray-200 p-2 rounded-md hover:bg-gray-300">

<i class="fas fa-sync-alt"></i> Refresh

</button>

</div>

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

<p class="text-gray-500">Loading contacts...</p>

</div>

</div>

</div>

</div>

<!-- 4. Performance Analytics (Reporting & Oversight) -->

<div data-tab-content="analytics" class="tab-pane hidden">

<div class="section-card bg-white">

<div class="card-header bg-red-600">Delivery & Performance Analytics</div>

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

<!-- Overview Metrics -->

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

<div class="p-4 bg-gray-100 rounded-lg">

<div class="text-3xl font-bold text-blue-600" id="totalMessagesSent">0</div>

<div class="text-sm text-gray-500">Total Sent (Monthly)</div>

</div>

<div class="p-4 bg-gray-100 rounded-lg">

<div class="text-3xl font-bold text-green-600" id="deliveryRate">0%</div>

<div class="text-sm text-gray-500">Delivery Rate</div>

</div>

<div class="p-4 bg-gray-100 rounded-lg">

<div class="text-3xl font-bold text-purple-600" id="readRate">0%</div>

<div class="text-sm text-gray-500">Read Rate</div>

</div>

<div class="p-4 bg-gray-100 rounded-lg">

<div class="text-3xl font-bold text-red-600" id="totalCost">₹0.00</div>

<div class="text-sm text-gray-500">Total Monthly Cost</div>

</div>

</div>

<!-- Top Templates by Performance -->

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

<h3 class="font-semibold mb-2">Top Performing Templates (By Read Rate)</h3>

<div id="topTemplates" class="space-y-2 text-sm">

<p class="text-gray-500">Loading...</p>

</div>

</div>

<!-- Engagement Breakdown -->

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

<h3 class="font-semibold mb-2">Engagement by Contact</h3>

<div id="contactEngagementList" class="space-y-2 text-sm">

<p class="text-gray-500">Loading...</p>

</div>

</div>

</div>

</div>

</div>

<!-- 5. API & Compliance Configuration -->

<div data-tab-content="settings" class="tab-pane hidden">

<div class="section-card bg-white">

<div class="card-header bg-gray-800">API, Quota & Compliance Settings</div>

<form id="settingsForm" onsubmit="updateSettings(event)" class="p-6 space-y-6">

<!-- API and Quota -->

<div class="border-b pb-4">

<h3 class="text-xl font-semibold mb-3 text-gray-700">API & Daily Quota</h3>

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

<div>

<label class="block text-sm font-medium text-gray-700">Business Phone Number</label>

<input type="text" id="businessNumber" class="mt-1 block w-full p-2 border rounded-md bg-gray-100" disabled>

</div>

<div>

<label class="block text-sm font-medium text-gray-700">Daily Message Limit</label>

<input type="number" id="dailyLimit" min="100" class="mt-1 block w-full p-2 border rounded-md">

<p class="text-xs text-gray-500 mt-1">Remaining: <span id="remainingQuota">...</span></p>

</div>

</div>

<div class="mt-4 p-3 bg-red-50 rounded-md">

<label class="block text-sm font-medium text-red-700">Webhook URL (For Delivery/Read Receipts)</label>

<input type="text" id="webhookUrl" class="mt-1 block w-full p-2 border border-red-300 rounded-md bg-white" placeholder="Requires secure URL for Tally/CRM Webhook">

</div>

</div>

<!-- Compliance and Opt-Out -->

<div class="border-b pb-4">

<h3 class="text-xl font-semibold mb-3 text-gray-700">Compliance & Opt-Out Rules</h3>

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

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

<input type="checkbox" id="consentRequired" class="form-checkbox h-5 w-5 text-red-600">

<span class="text-gray-700">Mandate Prior Consent for Non-Transactional Messages</span>

</label>

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

<input type="checkbox" id="automaticOptOut" class="form-checkbox h-5 w-5 text-red-600">

<span class="text-gray-700">Automatic Opt-Out Handling (STOP response)</span>

</label>

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

<input type="checkbox" id="contentFiltering" class="form-checkbox h-5 w-5 text-red-600">

<span class="text-gray-700">Enable Strict Content Filtering/Moderation</span>

</label>

<div>

<label class="block text-sm font-medium text-gray-700">Data Retention (Days)</label>

<input type="number" id="dataRetentionDays" min="30" class="mt-1 block w-full p-2 border rounded-md">

</div>

</div>

</div>

<button type="submit" class="bg-green-600 text-white px-6 py-3 rounded-lg hover:bg-green-700 transition duration-150">

<i class="fas fa-save mr-2"></i>Save All Settings

</button>

</form>

</div>

</div>

</div>

</div>

<script>

const API\_BASE = ''; // Assumed base path for API calls

let allTemplates = []; // Global store for templates

// --- Utility Functions ---

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

const colors = { success: 'bg-green-600', error: 'bg-red-600', info: 'bg-blue-600', warning: 'bg-yellow-600' };

const icons = { success: 'fas fa-check-circle', error: 'fas fa-exclamation-triangle', info: 'fas fa-info-circle', warning: 'fas fa-exclamation-triangle' };

const area = document.getElementById('notification-area');

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

messageDiv.className = `p-3 rounded-lg shadow-lg mb-2 text-white ${colors[type]}`;

messageDiv.innerHTML = `<i class="${icons[type]} mr-2"></i>${message}`;

area.appendChild(messageDiv);

setTimeout(() => messageDiv.remove(), 5000);

}

async function fetchAPI(endpoint, method = 'GET', body = null) {

try {

const options = {

method,

headers: { 'Content-Type': 'application/json' },

};

if (body && method !== 'GET') { options.body = JSON.stringify(body); }

const response = await fetch(`${API\_BASE}${endpoint}`, options);

const data = await response.json();

if (!response.ok) {

showNotification(data.error || `API Error: ${response.status}`, 'error');

throw new Error(data.error || 'API request failed');

}

return data;

} catch (error) {

console.error(`Fetch error on ${endpoint}:`, error);

throw error;

}

}

function switchTab(tabName) {

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

btn.classList.remove('active');

});

document.querySelector(`[data-tab="${tabName}"]`).classList.add('active');

document.querySelectorAll('.tab-pane').forEach(pane => {

pane.classList.add('hidden');

});

document.querySelector(`[data-tab-content="${tabName}"]`).classList.remove('hidden');

if (tabName === 'templates') fetchTemplates();

if (tabName === 'automation') fetchAutomationRules();

if (tabName === 'contacts') fetchContacts();

if (tabName === 'analytics') fetchAnalytics();

if (tabName === 'settings') fetchSettings();

}

// --- 1. Template Functions ---

async function fetchTemplates() {

const list = document.getElementById('templateList');

const sendSelect = document.getElementById('sendTemplateId');

list.innerHTML = '<p class="text-gray-500"><i class="fas fa-spinner fa-spin"></i> Loading templates...</p>';

sendSelect.innerHTML = '<option value="">Select Template...</option>';

try {

const data = await fetchAPI('/templates');

allTemplates = data;

list.innerHTML = data.map(t => `

<div class="p-4 border rounded-lg bg-gray-50 hover:bg-gray-100 transition">

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

<span class="font-bold text-lg">${t.name}</span>

<span class="status-badge ${t.active ? 'bg-green-100 text-green-800' : 'bg-red-100 text-red-800'}">${t.active ? 'Active' : 'Inactive'}</span>

</div>

<p class="text-sm text-gray-600 mt-1 italic">Trigger: ${t.triggerCondition} (${t.type})</p>

<p class="text-xs text-purple-600 mt-2">Variables: ${t.variables.map(v => `{{${v}}}`).join(', ')}</p>

<button class="mt-3 text-sm text-blue-600 hover:text-blue-800" onclick="editTemplate('${t.id}')">Edit / View Usage</button>

</div>

`).join('');

data.forEach(t => {

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

option.value = t.id;

option.textContent = `${t.name} (${t.category})`;

sendSelect.appendChild(option);

});

} catch (error) {

list.innerHTML = '<p class="text-red-600">Failed to load templates.</p>';

}

}

function displayTemplateVariables() {

const templateId = document.getElementById('sendTemplateId').value;

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

variableInputs.innerHTML = '';

if (!templateId) {

variableInputs.innerHTML = '<p class="text-sm text-gray-600 italic">Variables will appear here once a template is selected.</p>';

return;

}

const template = allTemplates.find(t => t.id === templateId);

if (template && template.variables) {

template.variables.forEach(v => {

const label = v.replace(/\_/g, ' ').toUpperCase();

variableInputs.innerHTML += `

<div>

<label class="block text-xs font-medium text-gray-700">${label}</label>

<input type="text" data-variable="${v}" placeholder="Value for {{${v}}}" class="w-full p-2 border rounded-md text-sm">

</div>

`;

});

}

}

async function sendSingleMessage(isImmediate) {

const templateId = document.getElementById('sendTemplateId').value;

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

const scheduledFor = isImmediate ? null : document.getElementById('scheduledFor').value;

const variableInputs = document.querySelectorAll('#variableInputs input[data-variable]');

if (!templateId || !recipientNumber) {

return showNotification('Please select a template and enter a recipient number.', 'warning');

}

if (!isImmediate && !scheduledFor) {

return showNotification('Please select a scheduled date and time.', 'warning');

}

const variables = {};

variableInputs.forEach(input => {

variables[input.dataset.variable] = input.value;

});

const payload = { templateId, recipientNumber, variables };

if (!isImmediate) payload.scheduledFor = scheduledFor;

const endpoint = '/send';

showNotification(`Sending request to ${isImmediate ? 'send now' : 'schedule'}...`, 'info');

try {

const data = await fetchAPI(endpoint, 'POST', payload);

if (data.scheduled) {

showNotification(`Message scheduled successfully for ${data.deliveryTime}!`, 'success');

} else {

showNotification(`Message sent successfully! Estimated delivery: 30 seconds.`, 'success');

}

} catch (error) {

// Error notification handled by fetchAPI

}

}

// --- 2. Automation Functions ---

async function fetchAutomationRules() {

const list = document.getElementById('automationRuleList');

list.innerHTML = '<p class="text-gray-500"><i class="fas fa-spinner fa-spin"></i> Loading rules...</p>';

try {

const data = await fetchAPI('/automation');

list.innerHTML = data.map(r => `

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

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

<span class="font-bold text-lg text-gray-800">${r.name}</span>

<span class="status-badge ${r.active ? 'bg-green-100 text-green-800' : 'bg-gray-100 text-gray-800'}">Status: ${r.active ? 'Active' : 'Inactive'}</span>

</div>

<p class="text-sm text-gray-600 mt-1">Trigger: <span class="font-medium">${r.trigger.toUpperCase().replace(/\_/g, ' ')}</span></p>

<p class="text-xs text-purple-600">Template: ${r.templateId} | Executed: ${r.executionCount} times</p>

<p class="text-xs mt-2 italic">Conditions: ${r.conditions.map(c => `${c.field} ${c.operator} ${c.value}`).join(' AND ')}</p>

</div>

`).join('');

} catch (error) {

list.innerHTML = '<p class="text-red-600">Failed to load automation rules.</p>';

}

}

// --- 3. Contact Functions ---

async function fetchContacts() {

const verified = document.getElementById('contactVerifiedFilter').value;

const consent = document.getElementById('contactConsentFilter').value;

const list = document.getElementById('contactList');

list.innerHTML = '<p class="text-gray-500"><i class="fas fa-spinner fa-spin"></i> Loading contacts...</p>';

try {

const data = await fetchAPI(`/contacts?verified=${verified}&consent=${consent}`);

list.innerHTML = data.map(c => `

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

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

<span class="font-bold text-gray-800">${c.parentName} (${c.parentType})</span>

<span class="text-sm text-blue-600">${c.phoneNumber}</span>

</div>

<p class="text-sm text-gray-600 mt-1">Student: ${c.studentName} (${c.class})</p>

<p class="text-xs mt-2">

<span class="mr-3 ${c.whatsappVerified ? 'text-green-600' : 'text-red-600'}">

<i class="fas fa-check-circle"></i> ${c.whatsappVerified ? 'Verified' : 'Unverified'}

</span>

<span class="mr-3 ${c.consentGiven ? 'text-green-600' : 'text-red-600'}">

<i class="fas fa-handshake"></i> Consent: ${c.consentGiven ? 'Yes' : 'No'}

</span>

<span class="mr-3 text-purple-600">Read Rate: ${c.readRate}%</span>

<button class="text-blue-600 text-xs hover:text-blue-800" onclick="updateContactPreferences('${c.id}')">Edit Preferences</button>

</p>

</div>

`).join('');

} catch (error) {

list.innerHTML = '<p class="text-red-600">Failed to load contacts.</p>';

}

}

// --- 4. Analytics Functions ---

async function fetchAnalytics() {

const dashboard = document.getElementById('analyticsDashboard');

dashboard.innerHTML = '<p class="text-center text-gray-500"><i class="fas fa-spinner fa-spin"></i> Fetching analytics...</p>';

try {

const data = await fetchAPI('/analytics');

const stats = data.monthlyStats;

const perf = data.performanceMetrics;

const realtime = data.realtimeStats;

document.getElementById('totalMessagesSent').textContent = stats.totalMessagesSent;

document.getElementById('deliveryRate').textContent = `${stats.deliveryRate}%`;

document.getElementById('readRate').textContent = `${stats.readRate}%`;

document.getElementById('totalCost').textContent = `₹${stats.totalCost.toFixed(2)}`;

// Top Templates

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

topTemplates.innerHTML = realtime.topPerformingTemplates.map(t => `

<div class="flex justify-between text-gray-700">

<span>${t.templateId}</span>

<span class="font-bold text-purple-600">${t.readRate}% Read</span>

</div>

`).join('');

// Contact Engagement

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

contactEngagementList.innerHTML = realtime.contactEngagement.slice(0, 5).map(c => `

<div class="flex justify-between text-gray-700">

<span>${c.studentName}</span>

<span class="font-bold">${c.readRate}% Read Rate</span>

</div>

`).join('');

} catch (error) {

// Dashboard remains visible with loading message if API fails

}

}

// --- 5. Settings Functions ---

async function fetchSettings() {

try {

const data = await fetchAPI('/settings');

const settings = data.settings || {};

document.getElementById('businessNumber').value = settings.businessNumber || '';

document.getElementById('dailyLimit').value = settings.dailyLimit || 1000;

document.getElementById('remainingQuota').textContent = settings.remainingQuota || 'N/A';

document.getElementById('webhookUrl').value = settings.webhookUrl || 'https://your-api.com/whatsapp/webhook';

document.getElementById('consentRequired').checked = settings.consentRequired !== false;

document.getElementById('automaticOptOut').checked = settings.automaticOptOut !== false;

document.getElementById('contentFiltering').checked = settings.contentFiltering !== false;

document.getElementById('dataRetentionDays').value = settings.dataRetentionDays || 90;

} catch (error) {

showNotification('Could not load current settings.', 'error');

}

}

async function updateSettings(event) {

event.preventDefault();

const btn = event.target.querySelector('button');

btn.disabled = true;

const payload = {

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

dailyLimit: parseInt(document.getElementById('dailyLimit').value),

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

consentRequired: document.getElementById('consentRequired').checked,

automaticOptOut: document.getElementById('automaticOptOut').checked,

contentFiltering: document.getElementById('contentFiltering').checked,

dataRetentionDays: parseInt(document.getElementById('dataRetentionDays').value),

// These are placeholder mock fields matching backend structure

apiProvider: 'whatsapp\_business\_api',

monthlyLimit: 25000

};

try {

const data = await fetchAPI('/settings', 'PUT', payload);

showNotification(data.message, 'success');

fetchSettings(); // Refresh settings after update

} catch {

// Error notification handled by fetchAPI

} finally {

btn.disabled = false;

}

}

// --- Initialization ---

document.addEventListener('DOMContentLoaded', () => {

fetchTemplates();

});

</script>

</body>

</html>

28. client/dist/404.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Page Not Found - SmartGenEduX</title>

<link href="https://cdn.jsdelivr.net/npm/tailwindcss@2.2.19/dist/tailwind.min.css" rel="stylesheet">

<link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css" rel="stylesheet">

</head>

<body class="bg-gradient-to-br from-blue-500 to-purple-600 min-h-screen flex items-center justify-center p-4">

<div class="text-center text-white">

<div class="mb-8">

<i class="fas fa-exclamation-triangle text-8xl text-yellow-300 mb-4"></i>

<h1 class="text-6xl font-bold mb-4">404</h1>

<h2 class="text-2xl font-semibold mb-4">Page Not Found</h2>

<p class="text-lg mb-8 max-w-md mx-auto">The page you're looking for doesn't exist or has been moved.</p>

</div>

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

<button onclick="window.history.back()" class="bg-white text-blue-600 px-6 py-3 rounded-lg font-semibold hover:bg-gray-100 transition-colors mr-4">

<i class="fas fa-arrow-left mr-2"></i>

Go Back

</button>

<a href="index.html" class="bg-yellow-500 text-white px-6 py-3 rounded-lg font-semibold hover:bg-yellow-600 transition-colors">

<i class="fas fa-home mr-2"></i>

Dashboard

</a>

</div>

</div>

</body>

</html>

29. client/dist/Dashboard\_new.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<title>SmartGenEduX Dashboard</title>

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

<style>

:root {

--sidebar-bg: #181c24;

--content-bg: #232837;

--tile-bg: #2d3247;

--tile-hover: #3a4060;

--accent: #00e6b8;

--premium: #ffb300;

--ai: #00b0ff;

--text: #f5f6fa;

--muted: #8a8fa3;

--border-radius: 18px;

}

body {

margin: 0;

font-family: 'Segoe UI', Arial, sans-serif;

background: var(--content-bg);

color: var(--text);

min-height: 100vh;

display: flex;

flex-direction: column;

}

.dashboard-container {

display: flex;

min-height: 100vh;

}

.sidebar {

width: 260px;

background: var(--sidebar-bg);

display: flex;

flex-direction: column;

align-items: center;

padding: 32px 0 0 0;

position: fixed;

height: 100vh;

box-shadow: 2px 0 12px rgba(0,0,0,0.12);

}

.sidebar .platform-logo {

width: 80px;

height: 80px;

margin-bottom: 32px;

display: flex;

align-items: center;

justify-content: center;

border-radius: 50%;

background: #fff;

overflow: hidden;

border: 2px solid var(--accent);

}

.sidebar .nav-icon {

width: 48px;

height: 48px;

margin: 18px 0;

background: var(--tile-bg);

border-radius: 12px;

display: flex;

align-items: center;

justify-content: center;

color: var(--muted);

font-size: 1.5em;

cursor: pointer;

transition: background 0.2s;

}

.sidebar .nav-icon:hover {

background: var(--tile-hover);

color: var(--accent);

}

.main-content {

margin-left: 260px;

flex: 1;

display: flex;

flex-direction: column;

min-height: 100vh;

}

.header {

background: var(--tile-bg);

padding: 24px 40px;

display: flex;

align-items: center;

justify-content: space-between;

border-bottom: 1px solid #232837;

}

.header .platform-logo {

width: 48px;

height: 48px;

border-radius: 12px;

background: #fff;

border: 2px solid var(--accent);

margin-right: 18px;

overflow: hidden;

display: flex;

align-items: center;

justify-content: center;

}

.header .branding {

display: flex;

align-items: center;

}

.header .user-role {

background: var(--tile-hover);

color: var(--accent);

padding: 8px 18px;

border-radius: 16px;

font-size: 1em;

font-weight: 500;

margin-left: 18px;

}

.header .quick-panel {

margin-left: 32px;

display: flex;

align-items: center;

gap: 18px;

}

.header .quick-panel .stat {

background: var(--sidebar-bg);

color: var(--text);

padding: 8px 14px;

border-radius: 10px;

font-size: 0.98em;

display: flex;

align-items: center;

gap: 6px;

}

.module-grid {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(260px, 1fr));

gap: 32px;

padding: 48px 40px;

background: url('https://www.transparenttextures.com/patterns/circuit-board.png'), var(--content-bg);

background-blend-mode: overlay;

}

.module-tile {

background: var(--tile-bg);

border-radius: var(--border-radius);

box-shadow: 0 2px 16px rgba(0,0,0,0.08);

padding: 32px 24px;

display: flex;

flex-direction: column;

align-items: center;

transition: transform 0.18s, box-shadow 0.18s;

position: relative;

cursor: pointer;

}

.module-tile:hover {

transform: translateY(-8px) scale(1.03);

box-shadow: 0 8px 32px rgba(0,0,0,0.18);

background: var(--tile-hover);

}

.module-tile .module-logo {

width: 56px;

height: 56px;

border-radius: 50%;

background: #fff;

display: flex;

align-items: center;

justify-content: center;

margin-bottom: 18px;

overflow: hidden;

border: 2px solid var(--accent);

}

.module-tile .module-name {

font-size: 1.2em;

font-weight: 600;

margin-bottom: 12px;

color: var(--text);

text-align: center;

}

.module-tile .feature-bullets {

color: var(--muted);

font-size: 0.98em;

margin-bottom: 12px;

text-align: center;

}

.module-tile .badge {

position: absolute;

top: 18px;

right: 18px;

padding: 6px 14px;

border-radius: 12px;

font-size: 0.9em;

font-weight: 600;

color: #fff;

background: var(--premium);

box-shadow: 0 2px 8px rgba(0,0,0,0.12);

}

.module-tile .badge.ai {

background: var(--ai);

}

.footer {

background: var(--sidebar-bg);

color: var(--muted);

text-align: center;

padding: 18px 0;

font-size: 0.98em;

margin-top: auto;

position: relative;

}

.footer .platform-logo-footer {

position: absolute;

right: 40px;

top: 10px;

width: 32px;

height: 32px;

border-radius: 8px;

background: #fff;

border: 2px solid var(--accent);

overflow: hidden;

display: flex;

align-items: center;

justify-content: center;

opacity: 0.2;

}

@media (max-width: 900px) {

.module-grid {

padding: 32px 10px;

gap: 18px;

}

.main-content {

margin-left: 0;

}

.sidebar {

display: none;

}

}

</style>

</head>

<body>

<div class="dashboard-container">

<aside class="sidebar">

<div class="platform-logo">

<img src="Module%20Assets%20%26%20Logos/smartgenedux-logo.svg" alt="SmartGenEduX Logo" width="64" height="64" />

</div>

<div class="nav-icon" title="Dashboard">🏠</div>

<div class="nav-icon" title="Modules">🧩</div>

<div class="nav-icon" title="Reports">📊</div>

<div class="nav-icon" title="Settings">⚙️</div>

</aside>

<main class="main-content">

<header class="header">

<div class="branding">

<div class="platform-logo">

<img src="Module%20Assets%20%26%20Logos/smartgenedux-logo.svg" alt="SmartGenEduX Logo" width="36" height="36" />

</div>

<span style="font-size:1.4em; font-weight:700; color:var(--text)">SmartGenEduX</span>

</div>

<div style="display:flex; align-items:center;">

<span class="user-role">Super Admin</span>

<div class="quick-panel">

<div class="stat"><span>🔔</span>3 Alerts</div>

<div class="stat"><span>📈</span>98% Uptime</div>

</div>

</div>

</header>

<section class="module-grid">

<!-- All modules below -->

<div class="module-tile">

<div class="module-logo">

<img src="Module%20Assets%20%26%20Logos/admission-management.svg" alt="Admission Management" width="40" height="40" />

</div>

<div class="module-name">Admission Management</div>

<div class="feature-bullets">

<ul><li>Online Admission Forms</li><li>Status Tracking</li></ul>

</div>

</div>

<div class="module-tile">

<div class="module-logo">

<img src="Module%20Assets%20%26%20Logos/fee-management.svg" alt="Fee Management" width="40" height="40" />

</div>

<div class="module-name">Fee Management</div>

<div class="feature-bullets">

<ul><li>Invoices & Receipts</li><li>Online Payments</li></ul>

</div>

</div>

<div class="module-tile">

<div class="module-logo">

<img src="Module%20Assets%20%26%20Logos/attendance.svg" alt="Attendance" width="40" height="40" />

</div>

<div class="module-name">Students/Teachers Attendance</div>

<div class="feature-bullets">

<ul><li>GPS Marking</li><li>Detailed Analytics</li></ul>

</div>

</div>

<div class="module-tile">

<div class="module-logo">

<img src="Module%20Assets%20%26%20Logos/arattai.svg" alt="Arattai Smart Alert" width="40" height="40" />

</div>

<div class="module-name">Arattai Smart Alert</div>

<div class="feature-bullets">

<ul><li>Smart Alerts</li></ul>

</div>

</div>

<div class="module-tile">

<div class="module-logo">

<img src="Module%20Assets%20%26%20Logos/whatsapp-integration.svg" alt="WhatsApp Integration" width="40" height="40" />

</div>

<div class="module-name">WhatsApp Integration</div>

<div class="feature-bullets">

<ul><li>Chat and Notifications</li></ul>

</div>

</div>

<div class="module-tile">

<div class="module-logo">

<img src="Module%20Assets%20%26%20Logos/documentation.svg" alt="Documentation" width="40" height="40" />

</div>

<div class="module-name">Documentation</div>

<div class="feature-bullets">

<ul><li>Comprehensive Guides</li></ul>

</div>

</div>

<div class="module-tile">

<div class="module-logo">

<img src="Module%20Assets%20%26%20Logos/id-card-generator.svg" alt="ID Card Generator" width="40" height="40" />

</div>

<div class="module-name">ID Card Generator</div>

<div class="feature-bullets">

<ul><li>Custom ID Cards</li></ul>

</div>

</div>

<div class="module-tile">

<div class="module-logo">

<img src="Module%20Assets%20%26%20Logos/school-events.svg" alt="School Events" width="40" height="40" />

</div>

<div class="module-name">School Events</div>

<div class="feature-bullets">

<ul><li>Event Scheduling</li></ul>

</div>

</div>

<div class="module-tile">

<div class="module-logo">

<img src="Module%20Assets%20%26%20Logos/school-transport.svg" alt="School Bus Transport" width="40" height="40" />

</div>

<div class="module-name">School Bus Transport</div>

<div class="feature-bullets">

<ul><li>Bus Routing</li></ul>

</div>

</div>

<div class="module-tile">

<div class="module-logo">

<img src="Module%20Assets%20%26%20Logos/cbse-registration.svg" alt="CBSE Registration" width="40" height="40" />

</div>

<div class="module-name">CBSE Registration Process</div>

<div class="feature-bullets">

<ul><li>Online CBSE Registrations</li></ul>

</div>

</div>

<div class="module-tile">

<div class="module-logo">

<img src="Module%20Assets%20%26%20Logos/timetable.svg" alt="Timetable Management" width="40" height="40" />

</div>

<div class="module-name">Timetable Management</div>

<div class="feature-bullets">

<ul><li>Auto Scheduling</li><li>Conflict Alerts</li></ul>

</div>

</div>

<div class="module-tile">

<div class="module-logo">

<img src="Module%20Assets%20%26%20Logos/library.svg" alt="Library Management" width="40" height="40" />

</div>

<div class="module-name">Library Management</div>

<div class="feature-bullets">

<ul><li>Book Management</li></ul>

</div>

</div>

<div class="module-tile">

<div class="module-logo">

<img src="Module%20Assets%20%26%20Logos/substitution.svg" alt="Substitution" width="40" height="40" />

</div>

<div class="module-name">Substitution</div>

<div class="feature-bullets">

<ul><li>Teacher Substitution</li></ul>

</div>

</div>

<div class="module-tile">

<div class="module-logo">

<img src="Module%20Assets%20%26%20Logos/student-distribution.svg" alt="Student Distribution" width="40" height="40" />

</div>

<div class="module-name">Student Distribution</div>

<div class="feature-bullets">

<ul><li>Class Allocation</li></ul>

</div>

</div>

<div class="module-tile">

<div class="module-logo">

<img src="Module%20Assets%20%26%20Logos/behaviour-tracker.svg" alt="Student Behaviour Tracker" width="40" height="40" />

</div>

<div class="module-name">Student Behaviour Tracker</div>

<div class="feature-bullets">

<ul><li>Behavioural Analysis</li></ul>

</div>

</div>

<div class="module-tile">

<div class="module-logo">

<img src="Module%20Assets%20%26%20Logos/report-tracker.svg" alt="Report Tracker" width="40" height="40" />

</div>

<div class="module-name">Report Tracker</div>

<div class="feature-bullets">

<ul><li>Performance Reports</li></ul>

</div>

</div>

<div class="module-tile">

<div class="module-logo">

<img src="Module%20Assets%20%26%20Logos/exam-seating.svg" alt="Exam Seating Planner" width="40" height="40" />

</div>

<div class="module-name">Exam Seating Planner</div>

<div class="feature-bullets">

<ul><li>Room Allocation</li></ul>

</div>

</div>

<div class="module-tile">

<div class="module-logo">

<img src="Module%20Assets%20%26%20Logos/voice-to-text.svg" alt="Voice To Text Converter" width="40" height="40" />

</div>

<div class="module-name">Voice To Text Converter</div>

<div class="feature-bullets">

<ul><li>Speech to Docs</li><li>AI Transcription</li></ul>

</div>

<div class="badge ai">AI</div>

</div>

<div class="module-tile">

<div class="module-logo">

<img src="Module%20Assets%20%26%20Logos/pdf-compilation.svg" alt="PDF Word Compilation" width="40" height="40" />

</div>

<div class="module-name">PDF Word Compilation</div>

<div class="feature-bullets">

<ul><li>Document Merging</li></ul>

</div>

</div>

<div class="module-tile">

<div class="module-logo">

<img src="Module%20Assets%20%26%20Logos/question-generator.svg" alt="Question Paper Generator" width="40" height="40" />

</div>

<div class="module-name">Question Paper Generator</div>

<div class="feature-bullets">

<ul><li>AI Paper Setter</li><li>Randomization</li></ul>

</div>

<div class="badge ai">AI</div>

</div>

<div class="module-tile">

<div class="module-logo">

<img src="Module%20Assets%20%26%20Logos/question-extractor.svg" alt="Question Extractor" width="40" height="40" />

</div>

<div class="module-name">Question Extractor</div>

<div class="feature-bullets">

<ul><li>PDF/Doc Parsing</li><li>AI Extraction</li></ul>

</div>

<div class="badge ai">AI</div>

</div>

<div class="module-tile">

<div class="module-logo">

<img src="Module%20Assets%20%26%20Logos/invigilation.svg" alt="Invigilation Duty Allocation" width="40" height="40" />

</div>

<div class="module-name">Invigilation Duty Allocation</div>

<div class="feature-bullets">

<ul><li>Exam Invigilation Scheduling</li></ul>

</div>

</div>

<div class="module-tile">

<div class="module-logo">

<img src="Module%20Assets%20%26%20Logos/fee-management-tally.svg" alt="Fee Management with Tally" width="40" height="40" />

</div>

<div class="module-name">Fee Management with Tally</div>

<div class="feature-bullets">

<ul><li>Tally Integration</li><li>Enterprise Finance</li></ul>

</div>

<div class="badge">Premium</div>

</div>

</section>

<footer class="footer">

&copy; 2025 SmartGenEduX. All rights reserved. Version 1.0

<div class="platform-logo-footer">

<img src="Module%20Assets%20%26%20Logos/smartgenedux-logo.svg" alt="SmartGenEduX Logo" width="24" height="24" />

</div>

</footer>

</main>

</div>

</body>

</html>

30. client/dist/Index.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1" />

<title>SmartGenEduX - Educational Platform</title>

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

<style>

/\* Your existing styles: circuit-pattern, glass-effect, animations, etc. \*/

body {

box-sizing: border-box;

}

.circuit-pattern {

background-image:

radial-gradient(circle at 25% 25%, rgba(59, 130, 246, 0.1) 0%, transparent 50%),

radial-gradient(circle at 75% 75%, rgba(147, 51, 234, 0.1) 0%, transparent 50%),

linear-gradient(135deg, rgba(59, 130, 246, 0.05) 0%, rgba(147, 51, 234, 0.05) 100%);

}

.circuit-overlay::before {

content: "";

position: absolute;

top: 0; left: 0; right: 0; bottom: 0;

background-image:

linear-gradient(90deg, rgba(59, 130, 246, 0.1) 1px, transparent 1px),

linear-gradient(rgba(59, 130, 246, 0.1) 1px, transparent 1px);

background-size: 50px 50px;

opacity: 0.3;

pointer-events: none;

}

.glass-effect {

background: rgba(15, 23, 42, 0.85);

backdrop-filter: blur(12px);

border: 1px solid rgba(59, 130, 246, 0.2);

box-shadow: 0 8px 32px rgba(0, 0, 0, 0.3);

}

.loading-spinner {

animation: spin 1s linear infinite;

}

@keyframes spin {

from { transform: rotate(0deg); }

to { transform: rotate(360deg); }

}

.fade-in {

animation: fadeIn 0.6s cubic-bezier(0.4, 0, 0.2, 1);

}

@keyframes fadeIn {

from { opacity: 0; transform: translateY(20px); }

to { opacity: 1; transform: translateY(0); }

}

</style>

</head>

<body class="min-h-full bg-slate-900 text-white circuit-pattern relative overflow-x-hidden">

<!-- Animated Background Blobs -->

<div class="fixed inset-0 overflow-hidden pointer-events-none">

<div class="absolute -top-40 -right-40 w-80 h-80 bg-blue-500 blob-shape animate-pulse"></div>

<div class="absolute -bottom-40 -left-40 w-80 h-80 bg-purple-500 blob-shape animate-pulse" style="animation-delay: 1s;"></div>

<div class="absolute top-1/2 left-1/2 transform -translate-x-1/2 -translate-y-1/2 w-60 h-60 bg-indigo-500 blob-shape animate-pulse" style="animation-delay: 2s;"></div>

</div>

<!-- Circuit Overlay -->

<div class="circuit-overlay fixed inset-0 pointer-events-none"></div>

<!-- Header -->

<header class="relative z-10 glass-effect border-b border-slate-700/50">

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

<div class="flex items-center justify-between h-16">

<!-- Logo -->

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

<div class="w-10 h-10 bg-gradient-to-br from-blue-500 to-purple-600 rounded-lg flex items-center justify-center shadow-lg">

<!-- Replace this svg with your logo SVG file if you have one -->

<svg class="w-6 h-6 text-white" fill="currentColor" viewBox="0 0 24 24">

<path d="M12 2L2 7v10c0 5.55 3.84 9.74 9 11 5.16-1.26 9-5.45 9-11V7l-10-5z"/>

<path d="M9 12l2 2 4-4" stroke="currentColor" stroke-width="2" fill="none"/>

</svg>

</div>

<div>

<h1 class="text-xl font-bold bg-gradient-to-r from-blue-400 to-purple-400 bg-clip-text text-transparent">

SmartGenEduX

</h1>

<p class="text-xs text-slate-400">Next-Gen Education Platform</p>

</div>

</div>

<!-- User Status & Auth Buttons -->

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

<div id="sessionInfo" class="hidden text-xs text-slate-400"></div>

<div id="userStatus" class="hidden items-center space-x-3">

<div class="w-8 h-8 bg-gradient-to-br from-green-400 to-blue-500 rounded-full flex items-center justify-center shadow-lg" id="userAvatar"></div>

<div class="text-sm">

<p class="font-medium">Welcome, <span id="userName"></span></p>

<p class="text-slate-400 text-xs" id="userRoleText"></p>

</div>

</div>

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

<button id="helpBtn" class="px-3 py-1 text-sm bg-slate-700/50 hover:bg-slate-600/50 rounded-lg transition-all duration-200 hover:scale-105">

Help

</button>

<button id="authBtn" class="px-4 py-1 text-sm bg-gradient-to-r from-blue-500 to-purple-600 hover:from-blue-600 hover:to-purple-700 rounded-lg transition-all duration-200 hover:scale-105 shadow-lg">

Login

</button>

</div>

</div>

</div>

</div>

</header>

<!-- Main Content -->

<main class="relative z-10 min-h-full flex items-center justify-center px-4 py-12">

<div class="max-w-lg w-full space-y-6">

<!-- Notification Area -->

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

<div id="notification" class="glass-effect rounded-xl p-4 border-l-4 fade-in">

<div class="flex items-center">

<div id="notificationIcon" class="flex-shrink-0 w-5 h-5 mr-3"></div>

<div class="flex-1">

<p id="notificationTitle" class="font-medium"></p>

<p id="notificationMessage" class="text-sm text-slate-300 mt-1"></p>

</div>

<button onclick="hideNotification()" class="ml-2 text-slate-400 hover:text-white" aria-label="Close notification">

<svg class="w-4 h-4" fill="currentColor" viewBox="0 0 20 20">

<path fill-rule="evenodd" d="M4.293 4.293a1 1 0 011.414 0L10 8.586l4.293-4.293a1 1 0 111.414 1.414L11.414 10l4.293 4.293a1 1 0 01-1.414 1.414L10 11.414l-4.293 4.293a1 1 0 01-1.414-1.414L8.586 10 4.293 5.707a1 1 0 010-1.414z" clip-rule="evenodd"/>

</svg>

</button>

</div>

</div>

</div>

<!-- Loading State -->

<div id="loadingState" class="text-center">

<div class="glass-effect rounded-2xl p-8 pulse-glow">

<div class="flex flex-col items-center space-y-6">

<div class="relative">

<div class="w-16 h-16 border-4 border-blue-500/30 border-t-blue-500 rounded-full loading-spinner"></div>

<div class="absolute inset-0 w-16 h-16 border-4 border-purple-500/20 border-b-purple-500 rounded-full loading-spinner" style="animation-delay: 0.5s; animation-direction: reverse;"></div>

</div>

<div>

<h2 class="text-xl font-semibold mb-2 bg-gradient-to-r from-blue-400 to-purple-400 bg-clip-text text-transparent">

Initializing SmartGenEduX

</h2>

<p class="text-slate-400" id="loadingMessage">Establishing secure connection...</p>

</div>

<div class="w-full bg-slate-700/50 rounded-full h-3 overflow-hidden">

<div id="progressBar" class="progress-shimmer h-3 rounded-full transition-all duration-500" style="width: 0%"></div>

</div>

<div id="securityInfo" class="text-xs text-slate-500 text-center space-y-1" aria-live="polite" aria-atomic="true">

<p>🔒 256-bit SSL Encryption Active</p>

<p>🛡️ Multi-Factor Authentication Ready</p>

</div>

</div>

</div>

</div>

<!-- Access Denied State -->

<div id="accessDeniedState" class="hidden text-center">

<div class="glass-effect rounded-2xl p-8 border-red-500/30 scale-in">

<div class="flex flex-col items-center space-y-6">

<div class="w-20 h-20 bg-red-500/20 rounded-full flex items-center justify-center relative">

<svg class="w-10 h-10 text-red-400" fill="none" stroke="currentColor" viewBox="0 0 24 24" aria-hidden="true">

<path stroke-linecap="round" stroke-linejoin="round" stroke-width="2" d="M12 15v2m0 0v2m0-2h2m-2 0H10m2-5V9m0 0V7m0 2h2m-2 0H10M21 12a9 9 0 11-18 0 9 9 0 0118 0z" />

</svg>

<div class="absolute inset-0 border-2 border-red-400/30 rounded-full animate-ping"></div>

</div>

<div>

<h2 class="text-2xl font-semibold mb-2 text-red-400">Access Denied</h2>

<p class="text-slate-400 mb-2">Authentication required to continue</p>

<p class="text-sm text-slate-500">Please log in with your SmartGenEduX credentials</p>

</div>

<div class="space-y-3 w-full">

<button onclick="showLogin()" class="w-full px-6 py-3 bg-gradient-to-r from-blue-500 to-purple-600 hover:from-blue-600 hover:to-purple-700 rounded-xl font-medium transition-all transform hover:scale-105 shadow-lg">

Login to Continue

</button>

</div>

</div>

</div>

</div>

<!-- Dashboard Redirect State -->

<div id="dashboardRedirectState" class="hidden text-center">

<div class="glass-effect rounded-2xl p-8 border-green-500/30 scale-in">

<div class="flex flex-col items-center space-y-6">

<div class="w-20 h-20 bg-green-500/20 rounded-full flex items-center justify-center relative">

<svg class="w-10 h-10 text-green-400" fill="none" stroke="currentColor" viewBox="0 0 24 24" aria-hidden="true">

<path stroke-linecap="round" stroke-linejoin="round" stroke-width="2" d="M9 12l2 2 4-4m6 2a9 9 0 11-18 0 9 9 0 0118 0z" />

</svg>

<div class="absolute inset-0 border-2 border-green-400/30 rounded-full animate-ping"></div>

</div>

<div>

<h2 class="text-2xl font-semibold mb-2 text-green-400">Welcome Back!</h2>

<p class="text-slate-400 mb-2">Authentication successful</p>

<p class="text-sm text-slate-500">Preparing your personalized dashboard...</p>

</div>

<div class="flex items-center space-x-2 text-sm text-slate-300" id="recentActivityPreview">

<!-- Recent activity dynamically inserted here -->

</div>

<div class="flex items-center space-x-2 text-sm text-slate-300 mt-2">

<div class="w-2 h-2 bg-blue-500 rounded-full animate-pulse"></div>

<span id="redirectMessage">Redirecting to dashboard...</span>

</div>

</div>

</div>

</div>

</div>

</main>

<!-- Footer -->

<footer class="relative z-10 glass-effect border-t border-slate-700/50 mt-auto">

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

<div class="flex flex-col sm:flex-row items-center justify-between space-y-4 sm:space-y-0">

<div class="text-sm text-slate-400">

© 2024 SmartGenEduX. All rights reserved.

</div>

<div class="text-sm text-slate-400">

Powered by <span class="bg-gradient-to-r from-blue-400 to-purple-400 bg-clip-text text-transparent font-medium">SmartGenEduX</span>

</div>

</div>

</div>

</footer>

<!-- Supabase SDK -->

<script src="https://cdn.jsdelivr.net/npm/@supabase/supabase-js"></script>

<script>

// Initialize Supabase client

const SUPABASE\_URL = 'https://waqttehjaifdkcbovund.supabase.co';

const SUPABASE\_ANON\_KEY = 'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZSIsInJlZiI6IndhcXR0ZWhqYWlmZGtjYm92dW5kIiwicm9sZSI6ImFub24iLCJpYXQiOjE3NDk2Mjk1ODIsImV4cCI6MjA2NTIwNTU4Mn0.nv1\_UIuXl-wSfkfTMvwCUsJTGO65TwZ2D8a571ELKJ0';

const supabase = supabase.createClient(SUPABASE\_URL, SUPABASE\_ANON\_KEY);

// DOM elements

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

const userNameEl = document.getElementById('userName');

const userRoleEl = document.getElementById('userRoleText');

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

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

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

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

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

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

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

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

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

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

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

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

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

// UI reset function

function resetUI() {

userStatus.classList.add('hidden');

sessionInfo.classList.add('hidden');

authBtn.textContent = 'Login';

authBtn.classList.remove('bg-red-500/80');

authBtn.classList.add('bg-gradient-to-r', 'from-blue-500', 'to-purple-600');

loadingState.classList.remove('hidden');

accessDeniedState.classList.add('hidden');

dashboardRedirectState.classList.add('hidden');

notificationArea.classList.add('hidden');

recentActivityPreview.innerHTML = '';

redirectMessage.textContent = 'Redirecting to dashboard...';

}

// Show notification helper

function showNotification(type, title, message) {

notificationTitle.textContent = title;

notificationMessage.textContent = message;

notificationArea.classList.remove('hidden');

if (type === 'success') {

notification.className = 'glass-effect rounded-xl p-4 border-l-4 border-green-500 fade-in';

notificationIcon.innerHTML = `<svg class="w-5 h-5 text-green-400" fill="currentColor" viewBox="0 0 20 20"><path fill-rule="evenodd" d="M10 18a8 8 0 100-16 8 8 0 000 16zm3.707-9.293a1 1 0 00-1.414-1.414L9 10.586 7.707 9.293a1 1 0 00-1.414 1.414l2 2a1 1 0 001.414 0l4-4z" clip-rule="evenodd"/></svg>`;

} else if (type === 'error') {

notification.className = 'glass-effect rounded-xl p-4 border-l-4 border-red-500 fade-in';

notificationIcon.innerHTML = `<svg class="w-5 h-5 text-red-400" fill="currentColor" viewBox="0 0 20 20"><path fill-rule="evenodd" d="M10 18a8 8 0 100-16 8 8 0 000 16zM8.707 7.293a1 1 0 00-1.414 1.414L8.586 10l-1.293 1.293a1 1 0 101.414 1.414L10 11.414l1.293 1.293a1 1 0 001.414-1.414L11.414 10l1.293-1.293a1 1 0 00-1.414-1.414L10 8.586 8.707 7.293z" clip-rule="evenodd"/></svg>`;

} else {

notification.className = 'glass-effect rounded-xl p-4 border-l-4 border-blue-500 fade-in';

notificationIcon.innerHTML = `<svg class="w-5 h-5 text-blue-400" fill="currentColor" viewBox="0 0 20 20"><path fill-rule="evenodd" d="M18 10a8 8 0 11-16 0 8 8 0 0116 0zm-7-4a1 1 0 11-2 0 1 1 0 012 0zM9 9a1 1 0 000 2v3a1 1 0 001 1h1a1 1 0 100-2v-3a1 1 0 00-1-1H9z" clip-rule="evenodd"/></svg>`;

}

setTimeout(() => {

notificationArea.classList.add('hidden');

}, 6000);

}

// Load user info and update UI

async function loadUser() {

resetUI();

loadingState.classList.remove('hidden');

const { data: { session }, error } = await supabase.auth.getSession();

if (error || !session) {

loadingState.classList.add('hidden');

accessDeniedState.classList.remove('hidden');

showNotification('error', 'Access Denied', 'Please log in to continue.');

return;

}

const user = session.user;

userNameEl.textContent = user.email || 'User';

userRoleEl.textContent = ''; // Add role logic if implemented

userAvatar.textContent = user.email?.slice(0, 2).toUpperCase() || 'U';

userStatus.classList.remove('hidden');

authBtn.textContent = 'Logout';

authBtn.classList.remove('bg-gradient-to-r', 'from-blue-500', 'to-purple-600');

authBtn.classList.add('bg-red-500/80');

loadingState.classList.add('hidden');

dashboardRedirectState.classList.remove('hidden');

setTimeout(() => {

window.location.href = '/dashboard'; // Adjust as per your routing setup

}, 4000);

}

// Auth button event

authBtn.addEventListener('click', async () => {

const { data: { session } } = await supabase.auth.getSession();

if (session) {

await supabase.auth.signOut();

showNotification('info', 'Logged Out', 'You have been successfully logged out.');

resetUI();

accessDeniedState.classList.remove('hidden');

} else {

window.location.href = '/login'; // Login page path

}

});

document.addEventListener('DOMContentLoaded', () => {

loadUser();

});

function hideNotification() {

notificationArea.classList.add('hidden');

}

</script>

</body>

</html>

31. client/dist/login.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<title>SmartGenEduX Login</title>

<meta name="viewport" content="width=device-width, initial-scale=1" />

<style>

body {

min-height: 100vh;

margin: 0;

font-family: 'Segoe UI', Arial, sans-serif;

background: linear-gradient(135deg, #1a237e 0%, #6a1b9a 100%);

position: relative;

overflow: hidden;

color: #fff;

}

body::before {

content: "";

position: absolute;

inset: 0;

background: url('data:image/svg+xml;utf8,<svg width="120" height="120" xmlns="http://www.w3.org/2000/svg"><g opacity="0.07"><rect x="20" y="20" width="80" height="80" rx="18" fill="none" stroke="white" stroke-width="2"/><circle cx="60" cy="60" r="12" fill="none" stroke="white" stroke-width="2"/><path d="M60 20v20M60 80v20M20 60h20M80 60h20" stroke="white" stroke-width="2" fill="none"/></g></svg>') repeat;

z-index: 0;

pointer-events: none;

}

.blur-shape {

position: absolute;

border-radius: 50%;

filter: blur(60px);

opacity: 0.32;

z-index: 1;

}

.blur1 { width: 320px; height: 320px; background: #7c4dff; top: 8%; left: 4%; }

.blur2 { width: 180px; height: 180px; background: #3949ab; bottom: 8%; right: 8%; }

.blur3 { width: 120px; height: 120px; background: #6a1b9a; top: 60%; left: 70%; }

.login-container {

position: relative;

z-index: 2;

min-height: 100vh;

display: flex;

align-items: center;

justify-content: center;

}

.login-card {

background: rgba(30, 32, 60, 0.68);

box-shadow: 0 8px 32px 0 rgba(31, 38, 135, 0.37);

backdrop-filter: blur(18px);

border-radius: 32px;

padding: 48px 36px 32px 36px;

width: 100%;

max-width: 400px;

display: flex;

flex-direction: column;

align-items: center;

border: 1.5px solid rgba(255,255,255,0.13);

margin: 32px 0;

}

.logo-container {

background: rgba(60, 60, 120, 0.7);

border-radius: 50%;

padding: 24px;

margin-bottom: 24px;

box-shadow: 0 4px 24px 0 #7c4dff55, 0 0 0 8px rgba(124,77,255,0.08);

display: flex;

align-items: center;

justify-content: center;

}

.logo-container img {

width: 64px;

height: 64px;

display: block;

}

.login-card h2 {

margin: 0 0 8px 0;

font-size: 2rem;

font-weight: 700;

letter-spacing: 0.02em;

text-align: center;

}

.login-card .tagline {

font-size: 1rem;

color: #b39ddb;

margin-bottom: 32px;

text-align: center;

}

.login-form {

width: 100%;

display: flex;

flex-direction: column;

gap: 20px;

}

.input-group {

position: relative;

width: 100%;

}

.input-group input {

width: 100%;

padding: 18px 16px 18px 48px;

background: rgba(255,255,255,0.09);

border: 1.5px solid rgba(255,255,255,0.18);

border-radius: 16px;

color: #fff;

font-size: 1.08rem;

outline: none;

transition: border 0.2s;

font-family: inherit;

}

.input-group input:focus {

border-color: #7c4dff;

}

.input-group label {

position: absolute;

left: 48px;

top: 18px;

color: #b39ddb;

font-size: 1.08rem;

pointer-events: none;

transition: 0.2s;

background: transparent;

}

.input-group input:focus + label,

.input-group input:not(:placeholder-shown) + label {

top: -12px;

left: 40px;

font-size: 0.88rem;

color: #7c4dff;

background: rgba(30,32,60,0.7);

padding: 0 6px;

border-radius: 4px;

}

.input-group .icon {

position: absolute;

left: 16px;

top: 50%;

transform: translateY(-50%);

color: #b39ddb;

font-size: 1.3rem;

pointer-events: none;

}

.form-options {

display: flex;

justify-content: space-between;

align-items: center;

font-size: 1rem;

margin-bottom: 8px;

}

.form-options label {

display: flex;

align-items: center;

gap: 6px;

cursor: pointer;

user-select: none;

}

.form-options input[type="checkbox"] {

accent-color: #7c4dff;

width: 16px;

height: 16px;

}

.form-options .forgot {

color: #7c4dff;

text-decoration: none;

font-weight: 500;

transition: text-decoration 0.2s;

}

.form-options .forgot:hover {

text-decoration: underline;

}

.sign-in-btn {

width: 100%;

padding: 15px 0;

background: linear-gradient(90deg, #7c4dff 0%, #3949ab 100%);

color: #fff;

font-size: 1.13rem;

font-weight: 700;

border: none;

border-radius: 14px;

box-shadow: 0 2px 12px 0 #7c4dff33;

cursor: pointer;

margin-top: 8px;

display: flex;

align-items: center;

justify-content: center;

gap: 12px;

transition: transform 0.1s, box-shadow 0.1s;

}

.sign-in-btn:hover, .sign-in-btn:focus {

transform: scale(1.03);

box-shadow: 0 4px 24px 0 #7c4dff55;

}

.login-footer {

margin-top: 28px;

width: 100%;

text-align: center;

font-size: 1rem;

color: #b39ddb;

}

.login-footer a {

color: #7c4dff;

text-decoration: none;

font-weight: 500;

margin-left: 4px;

}

.login-footer a:hover {

text-decoration: underline;

}

.microtext {

display: block;

font-size: 0.87rem;

color: #8e99f3;

margin-top: 6px;

}

.page-footer {

position: fixed;

bottom: 16px;

left: 0;

width: 100%;

text-align: center;

font-size: 0.95rem;

color: #b39ddb;

z-index: 2;

pointer-events: none;

}

.powered-by {

font-size: 0.85rem;

color: #8e99f3;

margin-top: 2px;

}

.watermark {

position: fixed;

bottom: 12px;

right: 18px;

opacity: 0.09;

z-index: 1;

width: 100px;

height: 100px;

pointer-events: none;

}

@media (max-width: 600px) {

.login-card {

padding: 28px 8px 18px 8px;

border-radius: 18px;

}

.logo-container {

padding: 12px;

}

.watermark {

width: 48px;

height: 48px;

}

.blur1, .blur2, .blur3 {

display: none;

}

}

</style>

</head>

<body>

<div class="blur-shape blur1"></div>

<div class="blur-shape blur2"></div>

<div class="blur-shape blur3"></div>

<div class="login-container">

<div class="login-card">

<div class="logo-container">

<img src="smartgenedux-logo.png" alt="SmartGenEduX Logo">

</div>

<h2>Welcome Back</h2>

<div class="tagline">Complete School Management ERP</div>

<form class="login-form" autocomplete="off" id="loginForm">

<div class="input-group">

<span class="icon">👤</span>

<input type="email" id="email" name="email" required placeholder=" " autocomplete="username" />

<label for="email">Email Address</label>

</div>

<div class="input-group">

<span class="icon">🔒</span>

<input type="password" id="password" name="password" required placeholder=" " autocomplete="current-password" />

<label for="password">Password</label>

</div>

<div class="form-options">

<label>

<input type="checkbox" name="remember" />

Remember me

</label>

<a href="#" class="forgot">Forgot password?</a>

</div>

<button type="submit" class="sign-in-btn">

<span>➤</span> Sign In

</button>

</form>

<div class="login-footer">

Don’t have an account?

<a href="#">Create Account</a>

<span class="microtext">Contact administrator for enterprise setup</span>

</div>

</div>

</div>

<div class="page-footer">

© 2025 SmartGenEduX. All rights reserved.

<div class="powered-by">Powered by Advanced School Management Technology</div>

</div>

<img src="smartgenedux-logo.png" alt="SmartGenEduX Watermark" class="watermark" />

<!-- Supabase SDK \*\*must\*\* be before your JS code -->

<script src="https://cdn.jsdelivr.net/npm/@supabase/supabase-js"></script>

<script>

const SUPABASE\_URL = 'https://waqttehjaifdkcbovund.supabase.co';

const SUPABASE\_ANON\_KEY = 'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZSIsInJlZiI6IndhcXR0ZWhqYWlmZGtjYm92dW5kIiwicm9sZSI6ImFub24iLCJpYXQiOjE3NDk2Mjk1ODIsImV4cCI6MjA2NTIwNTU4Mn0.nv1\_UIuXl-wSfkfTMvwCUsJTGO65TwZ2D8a571ELKJ0';

const supabase = supabase.createClient(SUPABASE\_URL, SUPABASE\_ANON\_KEY);

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

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

e.preventDefault();

const email = loginForm.email.value;

const password = loginForm.password.value;

try {

const { data, error } = await supabase.auth.signInWithPassword({ email, password });

if (error) {

alert('Login failed: ' + error.message);

return;

}

alert('Login successful! Redirecting...');

window.location.href = '/dashboard';

} catch (err) {

alert('Unexpected error: ' + err.message);

}

});

</script>

</body>

</html>

32. client/dist/Register.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1" />

<title>Register - SmartGenEduX</title>

<script src="https://cdn.jsdelivr.net/npm/@supabase/supabase-js@2/dist/supabase.min.js"></script>

<link href="https://cdn.jsdelivr.net/npm/tailwindcss@2.2.19/dist/tailwind.min.css" rel="stylesheet" />

<link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css" rel="stylesheet" />

<style>

body {

background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);

min-height: 100vh;

display: flex;

align-items: center;

justify-content: center;

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

}

.register-card {

background: rgba(255, 255, 255, 0.95);

backdrop-filter: blur(10px);

border-radius: 16px;

padding: 2rem;

width: 100%;

max-width: 400px;

box-shadow: 0 10px 30px rgba(0,0,0,0.1);

position: relative;

}

.input-group {

position: relative;

margin-bottom: 1.5rem;

}

.input-group input, .input-group select {

width: 100%;

padding: 0.75rem 1rem;

border: 1px solid #d1d5db;

border-radius: 8px;

outline: none;

font-size: 1rem;

transition: border-color 0.3s ease;

}

.input-group input:focus, .input-group select:focus {

border-color: #667eea;

box-shadow: 0 0 0 3px #c3dafe;

}

.input-group label {

position: absolute;

left: 1rem;

transform: translateY(-50%);

color: #9ca3af;

pointer-events: none;

transition: all 0.2s ease;

background: white;

padding: 0 0.25rem;

}

.input-group input:focus + label,

.input-group input:not(:placeholder-shown) + label,

.input-group select:focus + label,

.input-group select:valid + label {

top: -0.6rem;

font-size: 0.75rem;

color: #667eea;

}

.error-message, .success-message {

text-align: center;

margin-bottom: 1rem;

padding: 0.75rem;

border-radius: 8px;

font-weight: 600;

}

.error-message {

background-color: #fef2f2;

color: #dc2626;

border: 1px solid #f87171;

}

.success-message {

background-color: #f0fdf4;

color: #16a34a;

border: 1px solid #4ade80;

}

</style>

</head>

<body class="gradient-bg min-h-screen flex items-center justify-center p-4">

<!-- Background Pattern -->

<div class="absolute inset-0 overflow-hidden">

<div class="absolute -top-40 -right-40 w-80 h-80 bg-white opacity-10 rounded-full"></div>

<div class="absolute -bottom-40 -left-40 w-80 h-80 bg-white opacity-10 rounded-full"></div>

<div class="absolute top-1/2 left-1/4 w-60 h-60 bg-white opacity-5 rounded-full"></div>

</div>

<div class="relative z-10 w-full max-w-md">

<!-- Logo and Title -->

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

<div class="inline-flex items-center justify-center w-20 h-20 bg-white rounded-full shadow-lg mb-4 overflow-hidden">

<img src="Module%20Assets%20%26%20Logos/smartgenedux-logo.svg" alt="SmartGenEduX" class="w-full h-full object-cover" />

</div>

<h1 class="text-4xl font-bold text-white mb-2">SmartGenEduX</h1>

<p class="text-blue-100">Join Our Education Platform</p>

</div>

<!-- Registration Card -->

<div class="register-card rounded-2xl shadow-2xl p-8">

<div class="text-center mb-6">

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

<p class="text-gray-600">Get started with SmartGenEduX</p>

</div>

<!-- Message Area -->

<div id="messageArea"></div>

<!-- Registration Form -->

<form id="registerForm" class="space-y-6" novalidate>

<!-- Full Name -->

<div class="input-group">

<input type="text" id="name" name="fullName" placeholder=" " required minlength="2" />

<label for="name">Full Name</label>

</div>

<!-- Email -->

<div class="input-group">

<input type="email" id="email" name="email" placeholder=" " required />

<label for="email">Email Address</label>

</div>

<!-- Role -->

<div class="input-group">

<select id="role" name="role" required>

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

<option value="school\_admin">School Administrator</option>

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

</select>

<label for="role">Role</label>

</div>

<!-- School -->

<div class="input-group">

<input type="text" id="school" name="tenant" placeholder=" " required minlength="3" />

<label for="school">School Name</label>

</div>

<!-- Password -->

<div class="input-group">

<input type="password" id="password" name="password" placeholder=" " required minlength="6" />

<label for="password">Password (Min 6 Chars)</label>

</div>

<!-- Confirm Password -->

<div class="input-group">

<input type="password" id="confirmPassword" name="confirmPassword" placeholder=" " required />

<label for="confirmPassword">Confirm Password</label>

</div>

<!-- Terms -->

<div class="flex items-center">

<input type="checkbox" id="terms" class="rounded border-gray-300 text-blue-600 focus:ring-blue-500" required />

<label for="terms" class="ml-2 text-sm text-gray-600">

I agree to the

<a href="javascript:void(0)" onclick="openTerms()" class="text-blue-600 hover:text-blue-800">Terms of Service</a>

and

<a href="javascript:void(0)" onclick="openPrivacy()" class="text-blue-600 hover:text-blue-800">Privacy Policy</a>

</label>

</div>

<!-- Register Button -->

<button type="submit" class="w-full bg-gradient-to-r from-green-600 to-blue-600 text-white py-3 px-4 rounded-lg font-semibold hover:from-green-700 hover:to-blue-700 transform hover:scale-105 transition-all duration-300 shadow-lg">

<i class="fas fa-user-plus mr-2"></i> Create Account

</button>

</form>

<!-- Footer -->

<div class="text-center mt-6 pt-6 border-t border-gray-200">

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

Already have an account?

<a href="login.html" class="text-blue-600 hover:text-blue-800 font-medium">Sign In</a>

</p>

</div>

</div>

<div class="text-center mt-8 text-white text-sm">

<p>© 2024 SmartGenEduX. All rights reserved.</p>

<p class="mt-2 text-blue-200">Join thousands of educators worldwide</p>

</div>

</div>

<!-- Loading Overlay -->

<div id="loadingOverlay" class="fixed inset-0 bg-black bg-opacity-50 flex items-center justify-center hidden z-50">

<div class="bg-white rounded-lg p-6 text-center">

<div class="animate-spin w-8 h-8 border-4 border-blue-600 border-t-transparent rounded-full mx-auto mb-4"></div>

<p class="text-gray-600">Creating your account...</p>

</div>

</div>

<script>

const SUPABASE\_URL = 'https://waqttehjaifdkcbovund.supabase.co';

const SUPABASE\_ANON\_KEY = 'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZSIsInJlZiI6IndhcXR0ZWhqYWlmZGtjYm92dW5kIiwicm9sZSI6ImFub24iLCJpYXQiOjE3NDk2Mjk1ODIsImV4cCI6MjA2NTIwNTU4Mn0.nv1\_UIuXl-wSfkfTMvwCUsJTGO65TwZ2D8a571ELKJ0';

const supabase = supabase.createClient(SUPABASE\_URL, SUPABASE\_ANON\_KEY);

function showMessage(message, type) {

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

messageArea.innerHTML = '';

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

div.className = type === 'error' ? 'error-message' : 'success-message';

div.textContent = message;

messageArea.appendChild(div);

}

function openTerms() {

alert('Open Terms of Service content.');

}

function openPrivacy() {

alert('Open Privacy Policy content.');

}

document.getElementById('registerForm').addEventListener('submit', async (e) => {

e.preventDefault();

const name = e.target.fullName.value.trim();

const email = e.target.email.value.trim();

const password = e.target.password.value;

const confirmPassword = e.target.confirmPassword.value;

const role = e.target.role.value;

const tenant = e.target.tenant.value.trim();

const termsChecked = e.target.terms.checked;

// Validate inputs

if (!name || !email || !password || !confirmPassword || !role || !tenant) {

showMessage('Please fill all required fields.', 'error');

return;

}

if (password.length < 6) {

showMessage('Password must be at least 6 characters.', 'error');

return;

}

if (password !== confirmPassword) {

showMessage('Passwords do not match.', 'error');

return;

}

if (!termsChecked) {

showMessage('You must accept the terms and privacy policy.', 'error');

return;

}

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

try {

const { data, error } = await supabase.auth.signUp({

email,

password,

options: { data: { fullName: name, role, tenant } }

});

if (error) {

showMessage(error.message, 'error');

return;

}

showMessage('Account created successfully! Please check your email to confirm.', 'success');

setTimeout(() => {

window.location.href = 'login.html';

}, 3000);

} catch (err) {

showMessage('An unexpected error occurred: ' + err.message, 'error');

} finally {

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

}

});s

</script>

</body>

</html>