<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Admin Dashboard</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<!-- Sidebar -->

<div class="sidebar">

<h2>Admin Panel</h2>

<ul>

<li><a href="#dashboard" onclick="showSection('dashboard')">Dashboard</a></li>

<li><a href="#manage-classes" onclick="showSection('manage-classes')">Manage Classes</a></li>

<li><a href="#manage-teachers" onclick="showSection('manage-teachers')">Manage Teachers</a></li>

<li><a href="#manage-students" onclick="showSection('manage-students')">Manage Students</a></li>

<li><a href="#attendance" onclick="showSection('attendance')">Attendance</a></li>

<li><a href="#">Settings</a></li>

</ul>

</div>

<!-- Main Content -->

<div class="main-content">

<div class="header">

<h2>Administrator Dashboard</h2>

<button class="logout-btn" onclick="logout()">Logout</button>

</div>

<!-- Dashboard Section -->

<div id="dashboard">

<div class="dashboard-cards">

<div class="card">

<h3>Students</h3>

<p id="total-students">Loading...</p>

</div>

<div class="card">

<h3>Teachers</h3>

<p id="total-teachers">Loading...</p>

</div>

<div class="card">

<h3>Classes</h3>

<p id="total-classes">Loading...</p>

</div>

<div class="card">

<h3>Semesters</h3>

<p id="total-semesters">Loading...</p>

</div>

</div>

</div>

<!-- Manage Classes Section -->

<div id="manage-classes" class="manage-classes" style="display: none;">

<div class="class-management-container">

<div class="create-class-section">

<h2>Create Class</h2>

<input type="text" id="class-name" placeholder="Class Name">

<button onclick="addClass()">Save Class</button>

</div>

<div class="create-semester-section">

<h2>Add Semester to Class</h2>

<select id="semester-class-select" onchange="loadClassSemesters()">

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

</select>

<input type="text" id="semester-name" placeholder="Semester Name">

<button onclick="addSemester()">Add Semester</button>

</div>

</div>

<div class="class-list-container">

<h2>Saved Classes & Semesters</h2>

<div id="class-list">

<!-- Classes will be loaded here with their semesters -->

</div>

</div>

</div>

<!-- Manage Teachers Section -->

<div id="manage-teachers" class="manage-teachers" style="display: none;">

<h2>Create Teacher</h2>

<form id="teacher-form">

<input type="text" id="first-name" placeholder="First Name" required>

<input type="text" id="last-name" placeholder="Last Name" required>

<input type="email" id="email" placeholder="Email Address" required>

<input type="tel" id="phone" placeholder="Phone Number" required>

<select id="teacher-class" required>

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

</select>

<select id="teacher-semester" required>

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

</select>

<button type="submit">Save Teacher</button>

</form>

<div class="teacher-list" id="teacher-list">

<h2>Saved Teachers</h2>

</div>

</div>

<!-- Manage Students Section -->

<div id="manage-students" class="manage-students" style="display: none;">

<h2>Create Student</h2>

<form id="student-form">

<input type="text" id="student-first-name" placeholder="First Name" required>

<input type="text" id="student-last-name" placeholder="Last Name" required>

<input type="number" id="student-roll-no" placeholder="Roll No" required>

<input type="email" id="student-email" placeholder="Email Address" required>

<input type="tel" id="student-phone" placeholder="Phone Number" required>

<select id="student-class" required onchange="loadSemestersForStudent()">

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

</select>

<select id="student-semester" required>

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

</select>

<button type="submit">Save Student</button>

</form>

<div class="student-list" id="student-list">

<h2>Saved Students</h2>

</div>

</div>

<!-- Attendance Section -->

<div id="attendance" class="attendance" style="display: none;">

<h2>Mark Attendance</h2>

<div class="attendance-controls">

<label for="attendance-class">Select Class:</label>

<select id="attendance-class" onchange="loadSemestersForAttendance()">

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

</select>

<label for="attendance-semester">Select Semester:</label>

<select id="attendance-semester" onchange="fetchStudentsForAttendance()">

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

</select>

<label for="attendance-date">Select Date:</label>

<input type="date" id="attendance-date" required>

</div>

<div class="attendance-table">

<table>

<thead>

<tr>

<th>Student Name</th>

<th>Roll No</th>

<th>Status</th>

<th>Action</th>

</tr>

</thead>

<tbody id="attendance-student-list">

</tbody>

</table>

</div>

<button onclick="saveAttendance()">Save Attendance</button>

</div>

<!-- Settings Section -->

<div id="settings" class="settings" style="display: none;">

<h2>Settings</h2>

<p>Coming soon...</p>

</div>

</div>

<!-- Firebase SDK -->

<script type="module" src="firebase-config.js"></script>

<script type="module" src="app.js"></script>

</body>

</html>

import {

db,

collection,

getDocs,

doc,

getDoc,

deleteDoc,

updateDoc,

query,

where,

addDoc,

auth,

signOut,

onAuthStateChanged

} from './firebase-config.js';

// Global Functions

window.showSection = function(sectionId) {

document.querySelectorAll(".main-content > div").forEach((section) => {

section.style.display = "none";

});

const selectedSection = document.getElementById(sectionId);

if (selectedSection) {

selectedSection.style.display = "block";

}

if (sectionId === "dashboard") {

fetchDashboardData();

}

if (sectionId === "manage-classes") {

fetchClasses();

fetchClassesForDropdown("semester-class-select");

}

if (sectionId === "manage-teachers") {

fetchClassesForDropdown("teacher-class");

fetchTeachers();

}

if (sectionId === "manage-students") {

fetchClassesForDropdown("student-class");

fetchStudents();

}

if (sectionId === "attendance") {

fetchClassesForAttendanceDropdown();

}

};

window.onload = () => {

showSection("dashboard");

};

window.logout = async function() {

try {

await signOut(auth);

alert("Logged out successfully!");

window.location.href = "login.html";

} catch (error) {

console.error("Logout Error:", error);

alert("Error logging out: " + error.message);

}

};

// Class Management Functions

window.addClass = async function() {

const className = document.getElementById("class-name").value.trim();

if (className !== "") {

try {

await addDoc(collection(db, "classes"), {

name: className,

semesters: {} // Initialize empty semesters object

});

document.getElementById("class-name").value = "";

fetchClasses();

fetchClassesForDropdown("semester-class-select");

alert("Class added successfully!");

} catch (error) {

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

alert("Error adding class. Please try again.");

}

} else {

alert("Please enter a class name!");

}

};

async function fetchClasses() {

const classList = document.getElementById("class-list");

classList.innerHTML = "";

try {

const classesSnapshot = await getDocs(collection(db, "classes"));

if (classesSnapshot.empty) {

classList.innerHTML = "<p>No classes found.</p>";

} else {

classesSnapshot.forEach(async (doc) => {

const classData = doc.data();

const classItem = document.createElement("div");

classItem.className = "class-item";

// Class header with name and buttons

const classHeader = document.createElement("div");

classHeader.className = "class-header";

classHeader.innerHTML = `

<h3>${classData.name}</h3>

<div class="class-actions">

<button class="edit-btn" onclick="editClass('${doc.id}', '${classData.name}')">Edit</button>

<button class="delete-btn" onclick="deleteClass('${doc.id}')">Delete</button>

</div>

`;

// Semesters list

const semesterList = document.createElement("div");

semesterList.className = "semester-list";

if (classData.semesters && Object.keys(classData.semesters).length > 0) {

semesterList.innerHTML = "<h4>Semesters:</h4>";

for (const [semesterId, semesterData] of Object.entries(classData.semesters)) {

const semesterItem = document.createElement("div");

semesterItem.className = "semester-item";

semesterItem.innerHTML = `

<span>${semesterData.name}</span>

<div class="semester-actions">

<button class="edit-btn" onclick="editSemester('${doc.id}', '${semesterId}', '${semesterData.name}')">Edit</button>

<button class="delete-btn" onclick="deleteSemester('${doc.id}', '${semesterId}')">Delete</button>

</div>

`;

semesterList.appendChild(semesterItem);

}

} else {

semesterList.innerHTML = "<p>No semesters added yet.</p>";

}

classItem.appendChild(classHeader);

classItem.appendChild(semesterList);

classList.appendChild(classItem);

});

}

} catch (error) {

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

alert("Error fetching classes. Please try again.");

}

}

window.editClass = async function(id, currentName) {

const newName = prompt("Enter new class name:", currentName);

if (newName && newName.trim() !== "") {

try {

await updateDoc(doc(db, "classes", id), { name: newName });

fetchClasses();

alert("Class updated successfully!");

} catch (error) {

console.error("Error updating class:", error);

alert("Error updating class. Please try again.");

}

}

};

window.deleteClass = async function(id) {

if (confirm("Are you sure you want to delete this class and all its semesters?")) {

try {

await deleteDoc(doc(db, "classes", id));

fetchClasses();

alert("Class deleted successfully!");

} catch (error) {

console.error("Error deleting class:", error);

alert("Error deleting class. Please try again.");

}

}

};

// Semester Management Functions

window.addSemester = async function() {

const classId = document.getElementById("semester-class-select").value;

const semesterName = document.getElementById("semester-name").value.trim();

if (!classId) {

alert("Please select a class!");

return;

}

if (!semesterName) {

alert("Please enter a semester name!");

return;

}

try {

const classRef = doc(db, "classes", classId);

const classSnap = await getDoc(classRef);

if (classSnap.exists()) {

const classData = classSnap.data();

const newSemester = {

name: semesterName,

subjects: {}

};

// Update the semesters object in the class document

await updateDoc(classRef, {

[`semesters.${Date.now()}`]: newSemester // Using timestamp as temporary key

});

document.getElementById("semester-name").value = "";

fetchClasses();

alert("Semester added successfully!");

} else {

alert("Selected class not found!");

}

} catch (error) {

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

alert("Error adding semester. Please try again.");

}

};

window.editSemester = async function(classId, semesterId, currentName) {

const newName = prompt("Enter new semester name:", currentName);

if (newName && newName.trim() !== "") {

try {

const classRef = doc(db, "classes", classId);

await updateDoc(classRef, {

[`semesters.${semesterId}.name`]: newName

});

fetchClasses();

alert("Semester updated successfully!");

} catch (error) {

console.error("Error updating semester:", error);

alert("Error updating semester. Please try again.");

}

}

};

window.deleteSemester = async function(classId, semesterId) {

if (confirm("Are you sure you want to delete this semester?")) {

try {

const classRef = doc(db, "classes", classId);

await updateDoc(classRef, {

[`semesters.${semesterId}`]: deleteField()

});

fetchClasses();

alert("Semester deleted successfully!");

} catch (error) {

console.error("Error deleting semester:", error);

alert("Error deleting semester. Please try again.");

}

}

};

window.loadClassSemesters = async function() {

const classId = document.getElementById("semester-class-select").value;

if (!classId) return;

try {

const classRef = doc(db, "classes", classId);

const classSnap = await getDoc(classRef);

if (classSnap.exists()) {

const classData = classSnap.data();

// You can use this to populate a separate semester dropdown if needed

console.log("Available semesters:", classData.semesters);

}

} catch (error) {

console.error("Error loading semesters:", error);

}

};

// Teacher Management Functions

window.saveTeacher = async function(event) {

event.preventDefault();

const firstName = document.getElementById("first-name").value.trim();

const lastName = document.getElementById("last-name").value.trim();

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

const phone = document.getElementById("phone").value.trim();

const classId = document.getElementById("teacher-class").value;

const semesterId = document.getElementById("teacher-semester").value;

if (!firstName || !lastName || !email || !phone || !classId || !semesterId) {

alert("Please fill all fields!");

return;

}

try {

await addDoc(collection(db, "teachers"), {

firstName,

lastName,

email,

phone,

classId,

semesterId

});

document.getElementById("teacher-form").reset();

fetchTeachers();

alert("Teacher saved successfully!");

} catch (error) {

console.error("Error saving teacher:", error);

alert("Error saving teacher. Please try again.");

}

};

async function fetchTeachers() {

const teacherList = document.getElementById("teacher-list");

teacherList.innerHTML = "<h2>Saved Teachers</h2>";

try {

const teachersSnapshot = await getDocs(collection(db, "teachers"));

if (teachersSnapshot.empty) {

teacherList.innerHTML += "<p>No teachers found.</p>";

} else {

teachersSnapshot.forEach(async (teacherDoc) => {

const teacherData = teacherDoc.data();

// Get class name

let className = "Unknown Class";

if (teacherData.classId) {

const classDoc = await getDoc(doc(db, "classes", teacherData.classId));

if (classDoc.exists()) {

className = classDoc.data().name;

}

}

// Get semester name

let semesterName = "Unknown Semester";

if (teacherData.classId && teacherData.semesterId) {

const classDoc = await getDoc(doc(db, "classes", teacherData.classId));

if (classDoc.exists()) {

const classData = classDoc.data();

if (classData.semesters && classData.semesters[teacherData.semesterId]) {

semesterName = classData.semesters[teacherData.semesterId].name;

}

}

}

const teacherItem = document.createElement("div");

teacherItem.className = "teacher-item";

teacherItem.innerHTML = `

<div class="teacher-info">

<p><strong>${teacherData.firstName} ${teacherData.lastName}</strong></p>

<p>Email: ${teacherData.email}</p>

<p>Phone: ${teacherData.phone}</p>

<p>Class: ${className}</p>

<p>Semester: ${semesterName}</p>

</div>

<div class="teacher-actions">

<button class="edit-btn" onclick="editTeacher('${teacherDoc.id}')">Edit</button>

<button class="delete-btn" onclick="deleteTeacher('${teacherDoc.id}')">Delete</button>

</div>

`;

teacherList.appendChild(teacherItem);

});

}

} catch (error) {

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

alert("Error fetching teachers. Please try again.");

}

}

window.editTeacher = async function(id) {

const teacherDoc = await getDoc(doc(db, "teachers", id));

if (teacherDoc.exists()) {

const teacherData = teacherDoc.data();

const newFirstName = prompt("Enter new first name:", teacherData.firstName);

const newLastName = prompt("Enter new last name:", teacherData.lastName);

const newEmail = prompt("Enter new email:", teacherData.email);

const newPhone = prompt("Enter new phone number:", teacherData.phone);

if (newFirstName && newLastName && newEmail && newPhone) {

try {

await updateDoc(doc(db, "teachers", id), {

firstName: newFirstName,

lastName: newLastName,

email: newEmail,

phone: newPhone,

});

fetchTeachers();

alert("Teacher updated successfully!");

} catch (error) {

console.error("Error updating teacher:", error);

alert("Error updating teacher. Please try again.");

}

}

}

};

window.deleteTeacher = async function(id) {

if (confirm("Are you sure you want to delete this teacher?")) {

try {

await deleteDoc(doc(db, "teachers", id));

fetchTeachers();

alert("Teacher deleted successfully!");

} catch (error) {

console.error("Error deleting teacher:", error);

alert("Error deleting teacher. Please try again.");

}

}

};

// Student Management Functions

window.saveStudent = async function(event) {

event.preventDefault();

const firstName = document.getElementById("student-first-name").value.trim();

const lastName = document.getElementById("student-last-name").value.trim();

const rollNo = document.getElementById("student-roll-no").value.trim();

const email = document.getElementById("student-email").value.trim();

const phone = document.getElementById("student-phone").value.trim();

const classId = document.getElementById("student-class").value;

const semesterId = document.getElementById("student-semester").value;

if (!firstName || !lastName || !rollNo || !email || !phone || !classId || !semesterId) {

alert("Please fill all fields!");

return;

}

try {

await addDoc(collection(db, "students"), {

firstName,

lastName,

rollNo,

email,

phone,

classId,

semesterId

});

document.getElementById("student-form").reset();

fetchStudents();

alert("Student saved successfully!");

} catch (error) {

console.error("Error saving student:", error);

alert("Error saving student. Please try again.");

}

};

async function fetchStudents() {

const studentList = document.getElementById("student-list");

studentList.innerHTML = "<h2>Saved Students</h2>";

try {

const studentsSnapshot = await getDocs(collection(db, "students"));

if (studentsSnapshot.empty) {

studentList.innerHTML += "<p>No students found.</p>";

} else {

studentsSnapshot.forEach(async (studentDoc) => {

const studentData = studentDoc.data();

// Get class name

let className = "Unknown Class";

if (studentData.classId) {

const classDoc = await getDoc(doc(db, "classes", studentData.classId));

if (classDoc.exists()) {

className = classDoc.data().name;

}

}

// Get semester name

let semesterName = "Unknown Semester";

if (studentData.classId && studentData.semesterId) {

const classDoc = await getDoc(doc(db, "classes", studentData.classId));

if (classDoc.exists()) {

const classData = classDoc.data();

if (classData.semesters && classData.semesters[studentData.semesterId]) {

semesterName = classData.semesters[studentData.semesterId].name;

}

}

}

const studentItem = document.createElement("div");

studentItem.className = "student-item";

studentItem.innerHTML = `

<div class="student-info">

<p><strong>${studentData.firstName} ${studentData.lastName}</strong></p>

<p>Roll No: ${studentData.rollNo}</p>

<p>Email: ${studentData.email}</p>

<p>Phone: ${studentData.phone}</p>

<p>Class: ${className}</p>

<p>Semester: ${semesterName}</p>

</div>

<div class="student-actions">

<button class="edit-btn" onclick="editStudent('${studentDoc.id}')">Edit</button>

<button class="delete-btn" onclick="deleteStudent('${studentDoc.id}')">Delete</button>

</div>

`;

studentList.appendChild(studentItem);

});

}

} catch (error) {

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

alert("Error fetching students. Please try again.");

}

}

window.editStudent = async function(id) {

const studentDoc = await getDoc(doc(db, "students", id));

if (studentDoc.exists()) {

const studentData = studentDoc.data();

const newFirstName = prompt("Enter new first name:", studentData.firstName);

const newLastName = prompt("Enter new last name:", studentData.lastName);

const newRollNo = prompt("Enter new roll number:", studentData.rollNo);

const newEmail = prompt("Enter new email:", studentData.email);

const newPhone = prompt("Enter new phone number:", studentData.phone);

if (newFirstName && newLastName && newRollNo && newEmail && newPhone) {

try {

await updateDoc(doc(db, "students", id), {

firstName: newFirstName,

lastName: newLastName,

rollNo: newRollNo,

email: newEmail,

phone: newPhone,

});

fetchStudents();

alert("Student updated successfully!");

} catch (error) {

console.error("Error updating student:", error);

alert("Error updating student. Please try again.");

}

}

}

};

window.deleteStudent = async function(id) {

if (confirm("Are you sure you want to delete this student?")) {

try {

await deleteDoc(doc(db, "students", id));

fetchStudents();

alert("Student deleted successfully!");

} catch (error) {

console.error("Error deleting student:", error);

alert("Error deleting student. Please try again.");

}

}

};

// Attendance Management Functions

window.fetchStudentsForAttendance = async function() {

const classId = document.getElementById("attendance-class").value;

const semesterId = document.getElementById("attendance-semester").value;

const date = document.getElementById("attendance-date").value;

const studentList = document.getElementById("attendance-student-list");

studentList.innerHTML = "";

if (!classId || !semesterId || !date) {

return;

}

try {

const studentsQuery = query(

collection(db, "students"),

where("classId", "==", classId),

where("semesterId", "==", semesterId)

);

const studentsSnapshot = await getDocs(studentsQuery);

if (studentsSnapshot.empty) {

studentList.innerHTML = "<tr><td colspan='4'>No students found in this class/semester.</td></tr>";

} else {

for (const studentDoc of studentsSnapshot.docs) {

const studentData = studentDoc.data();

const studentId = studentDoc.id;

const attendanceQuery = query(

collection(db, "attendance"),

where("studentId", "==", studentId),

where("classId", "==", classId),

where("semesterId", "==", semesterId),

where("date", "==", date)

);

const attendanceSnapshot = await getDocs(attendanceQuery);

let status = "Unmarked";

if (!attendanceSnapshot.empty) {

status = attendanceSnapshot.docs[0].data().status;

}

const row = document.createElement("tr");

row.setAttribute("data-student-id", studentId);

row.innerHTML = `

<td>${studentData.firstName} ${studentData.lastName}</td>

<td>${studentData.rollNo}</td>

<td>${status}</td>

<td>

<button class="present-btn" onclick="markAttendance('${studentId}', 'present')">Present</button>

<button class="absent-btn" onclick="markAttendance('${studentId}', 'absent')">Absent</button>

</td>

`;

studentList.appendChild(row);

}

}

} catch (error) {

console.error("Error fetching students for attendance:", error);

alert("Error fetching students for attendance. Please try again.");

}

};

window.markAttendance = function(studentId, status) {

const row = document.querySelector(`tr[data-student-id="${studentId}"]`);

if (row) {

row.querySelector("td:nth-child(3)").innerText = status;

}

};

window.saveAttendance = async function() {

const classId = document.getElementById("attendance-class").value;

const semesterId = document.getElementById("attendance-semester").value;

const date = document.getElementById("attendance-date").value;

if (!classId || !semesterId || !date) {

alert("Please select a class, semester, and date!");

return;

}

const studentRows = document.querySelectorAll("#attendance-student-list tr");

if (studentRows.length === 0) {

alert("No students found to mark attendance!");

return;

}

try {

for (const row of studentRows) {

const studentId = row.getAttribute("data-student-id");

const status = row.querySelector("td:nth-child(3)").innerText;

const attendanceQuery = query(

collection(db, "attendance"),

where("studentId", "==", studentId),

where("classId", "==", classId),

where("semesterId", "==", semesterId),

where("date", "==", date)

);

const attendanceSnapshot = await getDocs(attendanceQuery);

if (attendanceSnapshot.empty) {

await addDoc(collection(db, "attendance"), {

studentId,

classId,

semesterId,

date,

status,

});

} else {

const attendanceId = attendanceSnapshot.docs[0].id;

await updateDoc(doc(db, "attendance", attendanceId), {

status,

});

}

}

alert("Attendance saved successfully!");

} catch (error) {

console.error("Error saving attendance:", error);

alert("Error saving attendance. Please try again.");

}

};

// Helper Functions

async function fetchDashboardData() {

try {

const studentsSnapshot = await getDocs(collection(db, "students"));

const totalStudents = studentsSnapshot.size;

const teachersSnapshot = await getDocs(collection(db, "teachers"));

const totalTeachers = teachersSnapshot.size;

const classesSnapshot = await getDocs(collection(db, "classes"));

const totalClasses = classesSnapshot.size;

// Calculate total semesters

let totalSemesters = 0;

classesSnapshot.forEach((doc) => {

const classData = doc.data();

if (classData.semesters) {

totalSemesters += Object.keys(classData.semesters).length;

}

});

document.getElementById("total-students").innerText = totalStudents;

document.getElementById("total-teachers").innerText = totalTeachers;

document.getElementById("total-classes").innerText = totalClasses;

document.getElementById("total-semesters").innerText = totalSemesters;

} catch (error) {

console.error("Error fetching dashboard data:", error);

alert("Error fetching dashboard data. Please try again.");

}

}

async function fetchClassesForDropdown(dropdownId) {

const classSelect = document.getElementById(dropdownId);

classSelect.innerHTML = "<option value=''>Select Class</option>";

try {

const classesSnapshot = await getDocs(collection(db, "classes"));

classesSnapshot.forEach((doc) => {

const option = document.createElement("option");

option.value = doc.id;

option.textContent = doc.data().name;

classSelect.appendChild(option);

});

} catch (error) {

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

alert("Error fetching classes. Please try again.");

}

}

async function fetchClassesForAttendanceDropdown() {

const classSelect = document.getElementById("attendance-class");

classSelect.innerHTML = "<option value=''>Select Class</option>";

try {

const classesSnapshot = await getDocs(collection(db, "classes"));

classesSnapshot.forEach((doc) => {

const option = document.createElement("option");

option.value = doc.id;

option.textContent = doc.data().name;

classSelect.appendChild(option);

});

} catch (error) {

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

alert("Error fetching classes. Please try again.");

}

}

// Load semesters when a class is selected (for student form)

window.loadSemestersForStudent = async function() {

const classId = document.getElementById("student-class").value;

const semesterSelect = document.getElementById("student-semester");

semesterSelect.innerHTML = "<option value=''>Select Semester</option>";

if (!classId) return;

try {

const classDoc = await getDoc(doc(db, "classes", classId));

if (classDoc.exists()) {

const classData = classDoc.data();

if (classData.semesters) {

for (const [semesterId, semesterData] of Object.entries(classData.semesters)) {

const option = document.createElement("option");

option.value = semesterId;

option.textContent = semesterData.name;

semesterSelect.appendChild(option);

}

}

}

} catch (error) {

console.error("Error loading semesters:", error);

}

};

// Load semesters when a class is selected (for attendance)

window.loadSemestersForAttendance = async function() {

const classId = document.getElementById("attendance-class").value;

const semesterSelect = document.getElementById("attendance-semester");

semesterSelect.innerHTML = "<option value=''>Select Semester</option>";

if (!classId) return;

try {

const classDoc = await getDoc(doc(db, "classes", classId));

if (classDoc.exists()) {

const classData = classDoc.data();

if (classData.semesters) {

for (const [semesterId, semesterData] of Object.entries(classData.semesters)) {

const option = document.createElement("option");

option.value = semesterId;

option.textContent = semesterData.name;

semesterSelect.appendChild(option);

}

}

}

} catch (error) {

console.error("Error loading semesters:", error);

}

};

// Event Listeners

document.getElementById("teacher-form").addEventListener("submit", window.saveTeacher);

document.getElementById("student-form").addEventListener("submit", window.saveStudent);