<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>EnergizeAI - Smart Energy Management</title>

<!-- Tailwind CSS CDN -->

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

<!-- Chart.js for graphs -->

<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>

<!-- Inter Font -->

<link rel="stylesheet" href="https://rsms.me/inter/inter.css">

<style>

body {

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

background-color: #f7f9fc;

}

.shadow-custom {

box-shadow: 0 4px 12px rgba(0, 0, 0, 0.08);

}

.btn-primary {

background-color: #4a90e2;

transition: all 0.3s ease;

}

.btn-primary:hover {

background-color: #357bd8;

transform: translateY(-2px);

}

.gradient-bg {

background-image: linear-gradient(to right, #4a90e2, #6a85b6);

}

.chart-container {

width: 100%;

height: 300px;

}

/\* Custom scrollbar for better aesthetics \*/

.scrollbar-container::-webkit-scrollbar {

width: 8px;

}

.scrollbar-container::-webkit-scrollbar-track {

background: #e2e8f0;

border-radius: 10px;

}

.scrollbar-container::-webkit-scrollbar-thumb {

background: #94a3b8;

border-radius: 10px;

}

/\* Unique animated background \*/

.background-unique {

position: fixed;

top: 0;

left: 0;

width: 100%;

height: 100%;

background-color: #0d1117;

background-image: radial-gradient(circle, rgba(74, 144, 226, 0.1) 1px, transparent 1px);

background-size: 30px 30px;

animation: moveBackground 60s linear infinite;

z-index: -1;

}

@keyframes moveBackground {

from { background-position: 0 0; }

to { background-position: 100% 100%; }

}

</style>

</head>

<body class="bg-gray-100 text-gray-800">

<div class="background-unique"></div>

<!-- Main Container -->

<div class="min-h-screen flex flex-col items-center p-4 md:p-8 relative z-10">

<!-- Header and Title -->

<header class="w-full max-w-4xl flex justify-between items-center mb-6">

<h1 class="text-3xl md:text-4xl font-extrabold text-white drop-shadow-md">EnergizeAI</h1>

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

<span id="user-status" class="text-sm font-medium text-white">Connecting...</span>

</div>

</header>

<!-- Action Buttons -->

<div class="w-full max-w-4xl grid grid-cols-2 md:grid-cols-4 gap-4 mb-6">

<button id="start-all-btn" class="px-4 py-3 rounded-xl btn-primary text-white font-semibold transition-all shadow-custom hover:shadow-lg">

Start All

</button>

<button id="emergency-shutdown-btn" class="px-4 py-3 rounded-xl bg-red-600 text-white font-semibold transition-all shadow-custom hover:shadow-lg">

Emergency Shut Down

</button>

<button id="ai-optimize-btn" class="px-4 py-3 rounded-xl bg-purple-600 text-white font-semibold transition-all shadow-custom hover:shadow-lg">

AI Optimize

</button>

<button id="export-report-btn" class="px-4 py-3 rounded-xl bg-gray-700 text-white font-semibold transition-all shadow-custom hover:shadow-lg">

Export Report

</button>

</div>

<!-- Main Dashboard Content -->

<main class="w-full max-w-4xl space-y-6">

<!-- Real-time Energy Usage Card -->

<div class="bg-white p-6 rounded-2xl shadow-custom transition-transform transform hover:scale-105 duration-300">

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

<h2 class="text-2xl font-bold text-gray-900">Real-Time Usage</h2>

<span id="power-status" class="px-3 py-1 text-sm font-semibold rounded-full bg-green-200 text-green-800">Live</span>

</div>

<div class="text-center">

<p class="text-5xl md:text-6xl font-extrabold text-blue-600 mb-2" id="live-usage">0.0 kWh</p>

<p class="text-lg text-gray-500">Currently Consuming</p>

</div>

</div>

<!-- Dashboard Grid -->

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

<!-- Machine Usage Breakdown -->

<div class="bg-white p-6 rounded-2xl shadow-custom transition-transform transform hover:scale-105 duration-300">

<h2 class="text-xl font-bold text-gray-900 mb-4">Machine Usage</h2>

<div id="machine-list" class="space-y-4 text-sm font-medium">

<!-- Machine data will be populated here by JavaScript -->

</div>

</div>

<!-- Energy Saving Suggestions -->

<div class="bg-white p-6 rounded-2xl shadow-custom transition-transform transform hover:scale-105 duration-300">

<h2 class="text-xl font-bold text-gray-900 mb-4">Energy Saving Tips</h2>

<div id="suggestions-list" class="space-y-4 scrollbar-container max-h-64 overflow-y-auto">

<!-- Suggestions will be populated here -->

</div>

</div>

<!-- Historical Usage Graph -->

<div class="bg-white p-6 rounded-2xl shadow-custom md:col-span-2">

<h2 class="text-xl font-bold text-gray-900 mb-4">Historical Usage</h2>

<div class="chart-container">

<canvas id="historical-chart"></canvas>

</div>

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

<button id="daily-btn" class="px-4 py-2 text-sm font-medium rounded-full btn-primary text-white focus:outline-none">Daily</button>

<button id="weekly-btn" class="px-4 py-2 text-sm font-medium rounded-full bg-gray-200 text-gray-700 hover:bg-gray-300 transition-colors focus:outline-none">Weekly</button>

<button id="monthly-btn" class="px-4 py-2 text-sm font-medium rounded-full bg-gray-200 text-gray-700 hover:bg-gray-300 transition-colors focus:outline-none">Monthly</button>

</div>

</div>

</div>

<!-- Gamification Card -->

<div class="bg-white p-6 rounded-2xl shadow-custom transition-transform transform hover:scale-105 duration-300">

<h2 class="text-xl font-bold text-gray-900 mb-4">Energy Saving Leaderboard</h2>

<div id="leaderboard" class="space-y-3">

<!-- Leaderboard will be populated by JavaScript -->

</div>

</div>

</main>

</div>

<!-- Firebase SDKs -->

<script type="module">

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

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

import { getFirestore, doc, onSnapshot, collection, query, orderBy, getDocs, setDoc, getDoc, updateDoc } from "https://www.gstatic.com/firebasejs/11.6.1/firebase-firestore.js";

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

// Set Firebase debug level

setLogLevel('debug');

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

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

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

let db, auth;

let userId = 'anonymous';

let liveDataInterval;

let currentChart;

let isRunning = true;

const AI\_TIPS = [

"Based on historical data, turning off machines during lunch breaks can save up to 15% of daily energy.",

"Consider upgrading your lighting to LEDs. It's a low-cost change that provides significant long-term savings.",

"Schedule high-load machines to run during off-peak hours to reduce your overall electricity bill.",

"The \*\*Drilling Machine\*\* shows a high usage peak at 11:30 AM every day. Optimize its schedule.",

"We have detected a constant phantom load on the \*\*Welding Unit\*\*. Please check for standby power consumption.",

"Running a power factor correction check on your main electrical panel can prevent wastage and fines.",

"The \*\*CNC Router\*\* shows an unusual spike in energy. This could indicate a potential maintenance need.",

];

// Initialize Firebase

const initFirebase = async () => {

try {

if (Object.keys(firebaseConfig).length === 0) {

console.error("Firebase config not available. Cannot initialize.");

return;

}

const app = initializeApp(firebaseConfig);

db = getFirestore(app);

auth = getAuth(app);

if (initialAuthToken) {

await signInWithCustomToken(auth, initialAuthToken);

} else {

await signInAnonymously(auth);

}

userId = auth.currentUser.uid;

document.getElementById('user-status').textContent = `User: ${userId.substring(0, 8)}...`;

console.log("Firebase initialized successfully for user:", userId);

// Start listening for data

listenToData();

} catch (error) {

console.error("Error initializing Firebase:", error);

document.getElementById('user-status').textContent = 'Error';

}

};

const listenToData = () => {

if (!db) {

console.error("Firestore not initialized.");

return;

}

// Path to the user's private data

const userDocRef = doc(db, `/artifacts/${appId}/users/${userId}/energyData`, 'dashboard');

// Listen for real-time changes to the user's dashboard data

onSnapshot(userDocRef, (docSnap) => {

if (docSnap.exists()) {

const data = docSnap.data();

renderDashboard(data);

} else {

console.log("No data found for the current user. Initializing dummy data.");

initializeDummyData();

}

}, (error) => {

console.error("Error listening to user data:", error);

});

// Path to the public leaderboard data

const leaderboardCollectionRef = collection(db, `/artifacts/${appId}/public/data/leaderboard`);

const q = query(leaderboardCollectionRef, orderBy('total\_saved', 'desc'));

// Listen for real-time changes to the public leaderboard

onSnapshot(q, (querySnapshot) => {

const leaderboardData = [];

querySnapshot.forEach((doc) => {

leaderboardData.push(doc.data());

});

renderLeaderboard(leaderboardData);

}, (error) => {

console.error("Error listening to leaderboard data:", error);

});

// Start the dynamic data generation

startLiveUpdates();

};

// Dynamic Data Simulation

const startLiveUpdates = () => {

if (liveDataInterval) clearInterval(liveDataInterval);

isRunning = true;

document.getElementById('power-status').textContent = 'Live';

document.getElementById('power-status').classList.remove('bg-red-200', 'text-red-800');

document.getElementById('power-status').classList.add('bg-green-200', 'text-green-800');

liveDataInterval = setInterval(updateLiveData, 2000);

};

const stopLiveUpdates = () => {

clearInterval(liveDataInterval);

isRunning = false;

document.getElementById('power-status').textContent = 'Offline';

document.getElementById('power-status').classList.remove('bg-green-200', 'text-green-800');

document.getElementById('power-status').classList.add('bg-red-200', 'text-red-800');

};

const updateLiveData = async () => {

if (!db || !isRunning) return;

const docRef = doc(db, `/artifacts/${appId}/users/${userId}/energyData`, 'dashboard');

const leaderboardDocRef = doc(db, `/artifacts/${appId}/public/data/leaderboard`, userId);

const docSnap = await getDoc(docRef);

if (docSnap.exists()) {

const data = docSnap.data();

const newLiveUsage = parseFloat((Math.random() \* (7.0 - 4.5) + 4.5).toFixed(2));

const newMachines = data.machines.map(m => ({

...m,

usage: parseFloat((Math.random() \* (m.usage + 1)).toFixed(2))

}));

const newHistorical = { ...data.historicalData };

newHistorical.daily.values.push(newLiveUsage);

if (newHistorical.daily.values.length > 10) newHistorical.daily.values.shift();

newHistorical.daily.labels.push(new Date().toLocaleTimeString());

if (newHistorical.daily.labels.length > 10) newHistorical.daily.labels.shift();

const updatedSaved = (data.totalSaved || 0) + (Math.random() \* 0.5);

await setDoc(docRef, { ...data, liveUsage: newLiveUsage, machines: newMachines, historicalData: newHistorical, totalSaved: updatedSaved }, { merge: true });

await setDoc(leaderboardDocRef, { user\_id: userId, total\_saved: updatedSaved }, { merge: true });

}

};

const renderDashboard = (data) => {

// Update live usage display

const liveUsage = data.liveUsage || 0;

document.getElementById('live-usage').textContent = `${liveUsage.toFixed(2)} kWh`;

// Update machine list

const machineList = document.getElementById('machine-list');

machineList.innerHTML = '';

if (data.machines) {

data.machines.forEach(machine => {

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

machineEl.className = 'flex justify-between items-center';

machineEl.innerHTML = `

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

<span class="font-bold text-gray-900">${machine.usage.toFixed(2)} kWh</span>

`;

machineList.appendChild(machineEl);

});

}

// Update suggestions

const suggestionsList = document.getElementById('suggestions-list');

suggestionsList.innerHTML = '';

if (data.suggestions) {

data.suggestions.forEach(tip => {

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

tipEl.className = 'flex items-center space-x-2 p-3 bg-gray-50 rounded-lg';

tipEl.innerHTML = `

<svg class="w-6 h-6 text-yellow-500 flex-shrink-0" 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"></path>

</svg>

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

`;

suggestionsList.appendChild(tipEl);

});

}

// Update charts

if (currentChart) currentChart.destroy();

updateChart(data.historicalData.daily, 'daily');

};

const renderLeaderboard = (leaderboardData) => {

const leaderboardEl = document.getElementById('leaderboard');

leaderboardEl.innerHTML = '';

leaderboardData.forEach((user, index) => {

const isCurrentUser = user.user\_id === userId;

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

userEl.className = `flex justify-between items-center p-3 rounded-lg ${isCurrentUser ? 'gradient-bg text-white' : 'bg-gray-50'}`;

userEl.innerHTML = `

<span class="font-bold">${index + 1}. ${user.user\_id.substring(0, 8)}...</span>

<span class="font-semibold">${user.total\_saved.toFixed(2)} kWh Saved</span>

`;

leaderboardEl.appendChild(userEl);

});

};

const updateChart = (data, label) => {

const ctx = document.getElementById('historical-chart').getContext('2d');

if (currentChart) {

currentChart.destroy();

}

currentChart = new Chart(ctx, {

type: 'bar',

data: {

labels: data.labels,

datasets: [{

label: `Energy Usage (${label})`,

data: data.values,

backgroundColor: '#4a90e2',

borderColor: '#4a90e2',

borderWidth: 1,

borderRadius: 8,

}]

},

options: {

responsive: true,

maintainAspectRatio: false,

scales: {

y: {

beginAtZero: true,

title: {

display: true,

text: 'Energy (kWh)'

}

}

},

plugins: {

legend: {

display: false

}

}

}

});

};

// Dummy Data Initialization for unique project showcase

const initializeDummyData = async () => {

if (!db) return;

const userDocRef = doc(db, `/artifacts/${appId}/users/${userId}/energyData`, 'dashboard');

const leaderboardDocRef = doc(db, `/artifacts/${appId}/public/data/leaderboard`, userId);

const dummyData = {

liveUsage: 5.4,

machines: [

{ name: 'Drilling Machine', usage: 2.1 },

{ name: 'Welding Unit', usage: 1.5 },

{ name: 'CNC Router', usage: 1.0 },

{ name: 'Lighting System', usage: 0.8 },

],

suggestions: [

'Welcome to EnergizeAI! Your smart energy management dashboard.',

'Click "AI Optimize" to get personalized tips for your machines.'

],

historicalData: {

daily: {

labels: ['10 AM', '11 AM', '12 PM', '1 PM', '2 PM', '3 PM', '4 PM'],

values: [3.5, 4.2, 5.1, 4.8, 5.4, 6.0, 5.7],

},

weekly: {

labels: