<!DOCTYPE html>

<html lang="tr">

<head>

<meta charset="UTF-8">

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

<title>TatilKaptani.com - Kaptanınız Yanınızda</title>

<script src="https://www.gstatic.com/firebasejs/9.6.10/firebase-app-compat.js"></script>

<script src="https://www.gstatic.com/firebasejs/9.6.10/firebase-auth-compat.js"></script>

<script src="https://www.gstatic.com/firebasejs/9.6.10/firebase-firestore-compat.js"></script>

<style>

/\* CSS Kodları Başlangıcı \*/

body {

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

margin: 0;

padding: 0;

background-color: #f5f7fa; /\* Açık gri arka plan \*/

color: #333;

display: flex;

flex-direction: column;

min-height: 100vh;

background-image: url('https://source.unsplash.com/random/1920x1080/?travel,beach,mountain'); /\* Tatil havası katan arka plan \*/

background-size: cover;

background-attachment: fixed;

background-position: center;

}

header {

background-color: rgba(0, 50, 100, 0.7); /\* Koyu mavi, hafif şeffaf \*/

color: white;

padding: 1rem 2rem;

display: flex;

justify-content: space-between;

align-items: center;

box-shadow: 0 2px 4px rgba(0,0,0,0.1);

}

.logo {

font-size: 1.8rem;

font-weight: bold;

color: #fff;

text-decoration: none;

}

nav a {

color: white;

text-decoration: none;

margin-left: 20px;

font-weight: 500;

transition: color 0.3s ease;

}

nav a:hover {

color: #a0dfff; /\* Açık mavi hover \*/

}

.user-actions {

display: flex;

gap: 10px;

}

.user-actions button {

background-color: #007bff;

color: white;

border: none;

padding: 8px 15px;

border-radius: 5px;

cursor: pointer;

font-weight: 500;

transition: background-color 0.3s ease;

}

.user-actions button:hover {

background-color: #0056b3;

}

#loggedInUser span {

margin-right: 10px;

font-weight: bold;

}

#loggedInUser button {

background-color: #dc3545;

}

#loggedInUser button:hover {

background-color: #c82333;

}

main {

flex-grow: 1;

display: flex;

flex-direction: column;

align-items: center;

justify-content: center;

padding: 20px;

text-align: center;

}

.hero-section {

background-color: rgba(255, 255, 255, 0.7); /\* Beyaz, şeffaf \*/

padding: 3rem 4rem;

border-radius: 15px;

box-shadow: 0 8px 25px rgba(0,0,0,0.2);

backdrop-filter: blur(5px); /\* Hafif bulanıklık efekti \*/

-webkit-backdrop-filter: blur(5px); /\* Safari desteği \*/

max-width: 800px;

width: 90%;

margin-bottom: 30px;

}

.hero-section h1 {

font-size: 3rem;

color: #003366; /\* Koyu mavi başlık \*/

margin-bottom: 15px;

text-shadow: 1px 1px 2px rgba(0,0,0,0.1);

}

.slogan {

font-size: 1.5rem;

color: #555;

margin-bottom: 25px;

font-style: italic;

}

.chat-container {

background-color: rgba(255, 255, 255, 0.85); /\* Beyaz, daha az şeffaf \*/

padding: 20px;

border-radius: 10px;

box-shadow: 0 4px 15px rgba(0,0,0,0.15);

width: 90%;

max-width: 600px;

display: flex;

flex-direction: column;

max-height: 70vh;

backdrop-filter: blur(3px);

-webkit-backdrop-filter: blur(3px);

opacity: 0; /\* Başlangıçta gizli \*/

transform: translateY(20px);

transition: opacity 0.5s ease-out, transform 0.5s ease-out;

}

.chat-container.visible {

opacity: 1;

transform: translateY(0);

}

.chat-messages {

flex-grow: 1;

overflow-y: auto;

border: 1px solid #e0e0e0;

padding: 10px;

margin-bottom: 15px;

border-radius: 8px;

background-color: #fafafa;

}

.message {

margin-bottom: 10px;

line-height: 1.5;

}

.message.user {

text-align: right;

color: #0056b3;

}

.message.ai {

text-align: left;

color: #333;

}

.message span {

background-color: #e0f2f7;

padding: 8px 12px;

border-radius: 15px;

display: inline-block;

max-width: 80%;

word-wrap: break-word;

}

.message.user span {

background-color: #d1e7fd;

}

.chat-input {

display: flex;

gap: 10px;

}

.chat-input input[type="text"] {

flex-grow: 1;

padding: 12px;

border: 1px solid #ccc;

border-radius: 25px;

font-size: 1rem;

outline: none;

transition: border-color 0.3s ease;

}

.chat-input input[type="text"]:focus {

border-color: #007bff;

}

.chat-input button {

padding: 12px 20px;

background-color: #28a745; /\* Yeşil gönder butonu \*/

color: white;

border: none;

border-radius: 25px;

cursor: pointer;

font-size: 1rem;

transition: background-color 0.3s ease;

}

.chat-input button:hover {

background-color: #218838;

}

footer {

margin-top: auto; /\* Alt kısma sabitle \*/

background-color: rgba(0, 0, 0, 0.6); /\* Siyah, şeffaf \*/

color: white;

text-align: center;

padding: 1rem 0;

}

/\* Reklam Alanı \*/

.ad-area {

position: fixed;

right: 20px;

top: 50%;

transform: translateY(-50%);

display: flex;

flex-direction: column;

gap: 15px;

z-index: 1000;

}

.ad-item {

background-color: rgba(255, 255, 255, 0.7);

padding: 10px;

border-radius: 8px;

box-shadow: 0 2px 8px rgba(0,0,0,0.1);

text-align: center;

width: 150px;

min-height: 100px;

display: flex;

flex-direction: column;

justify-content: center;

align-items: center;

font-size: 0.9rem;

color: #555;

backdrop-filter: blur(2px);

-webkit-backdrop-filter: blur(2px);

}

.ad-item a {

color: #007bff;

text-decoration: none;

margin-top: 5px;

}

/\* Modals for Login/Register/Forgot Password \*/

.modal {

display: none; /\* Hidden by default \*/

position: fixed; /\* Stay in place \*/

z-index: 1001; /\* Sit on top \*/

left: 0;

top: 0;

width: 100%; /\* Full width \*/

height: 100%; /\* Full height \*/

overflow: auto; /\* Enable scroll if needed \*/

background-color: rgba(0,0,0,0.6); /\* Black w/ opacity \*/

justify-content: center;

align-items: center;

}

.modal-content {

background-color: #fefefe;

margin: auto;

padding: 30px;

border-radius: 10px;

box-shadow: 0 5px 15px rgba(0,0,0,0.3);

width: 90%;

max-width: 400px;

text-align: center;

position: relative;

}

.close-button {

color: #aaa;

position: absolute;

right: 15px;

top: 10px;

font-size: 28px;

font-weight: bold;

cursor: pointer;

}

.close-button:hover,

.close-button:focus {

color: black;

text-decoration: none;

cursor: pointer;

}

.modal-content input {

margin-bottom: 15px;

}

.modal-content button {

margin-bottom: 10px;

}

.modal-content p {

font-size: 0.9rem;

cursor: pointer;

color: #007bff;

text-decoration: underline;

}

.modal-content p:hover {

color: #0056b3;

}

.modal-message {

margin-top: 10px;

font-weight: bold;

}

</style>

</head>

<body>

<header>

<a href="#" class="logo">TatilKaptani.com</a>

<nav>

<div class="user-actions" id="authButtons">

<button id="loginBtn">Üye Girişi</button>

<button id="registerBtn">Üye Ol</button>

</div>

<div class="user-actions" id="loggedInUser" style="display: none;">

<span>Merhaba, <span id="usernameDisplay">Kaptan</span>!</span>

<button id="logoutBtn">Çıkış Yap</button>

</div>

</nav>

</header>

<main>

<section class="hero-section">

<h1 id="mainSlogan">Hayallerinizin Ötesinde Bir Tatil İçin Kaptanınız Yanınızda!</h1>

<p class="slogan">Yapay zekanın büyülü dokunuşuyla keşfedilmeyi bekleyen her an...</p>

</section>

<section class="chat-container" id="chatSection">

<div class="chat-messages" id="chatMessages">

<div class="message ai"><span>Merhaba! Tatil Kaptanınız olarak size nasıl yardımcı olabilirim?</span></div>

</div>

<div class="chat-input">

<input type="text" id="chatInput" placeholder="Mesajınızı buraya yazın...">

<button id="sendChatBtn">Gönder</button>

</div>

</section>

</main>

<div class="ad-area">

<div class="ad-item">

<p>İlan vermek için bizimle iletişime geçin!</p>

<a href="mailto:ilan@tatilkaptani.com">ilan@tatilkaptani.com</a>

</div>

<div class="ad-item">

<p>Sanal Turlarımızla Dünyayı Keşfet!</p>

<a href="#">Detaylar İçin Tıklayın</a>

</div>

<div class="ad-item">

<p>VIP Üyelere Özel Fırsatlar!</p>

<a href="#">Hemen Üye Ol</a>

</div>

<div class="ad-item">

<p>Hayalindeki Tatili Yarat!</p>

<a href="#">AI Fotoğraf Oluştur</a>

</div>

<div class="ad-item">

<p>Yakında: Rekabetçi Tatil Oyunları!</p>

<a href="#">Hazır Ol</a>

</div>

</div>

<footer>

&copy; <span id="currentYear"></span> TatilKaptani.com. Tüm hakları saklıdır.

</footer>

<div id="loginModal" class="modal">

<div class="modal-content">

<span class="close-button">&times;</span>

<h2>Üye Girişi</h2>

<input type="email" id="loginEmail" placeholder="E-posta Adresi" required>

<input type="password" id="loginPassword" placeholder="Şifre" required>

<button id="performLoginBtn">Giriş Yap</button>

<p id="forgotPasswordLink">Şifremi Unuttum</p>

<div id="loginMessage" class="modal-message"></div>

</div>

</div>

<div id="registerModal" class="modal">

<div class="modal-content">

<span class="close-button">&times;</span>

<h2>Üye Ol</h2>

<input type="text" id="registerUsername" placeholder="Kullanıcı Adı" required>

<input type="email" id="registerEmail" placeholder="E-posta Adresi" required>

<input type="password" id="registerPassword" placeholder="Şifre" required>

<button id="performRegisterBtn">Kayıt Ol</button>

<div id="registerMessage" class="modal-message"></div>

</div>

</div>

<div id="forgotPasswordModal" class="modal">

<div class="modal-content">

<span class="close-button">&times;</span>

<h2>Şifre Sıfırla</h2>

<input type="email" id="resetEmail" placeholder="E-posta Adresiniz" required>

<button id="performResetBtn">Şifre Sıfırlama Linki Gönder</button>

<div id="resetMessage" class="modal-message"></div>

</div>

</div>

<script>

// --- Firebase Müşteri SDK'sını Başlatın ---

// BURAYI KENDİ FIREBASE PROJENİZİN BİLGİLERİYLE DEĞİŞTİRİN!

// Firebase konsolundan (Proje Ayarları -> Genel -> Uygulamalarınız) alabilirsiniz.

const firebaseConfig = {

apiKey: "YOUR\_FIREBASE\_CLIENT\_API\_KEY", // Örneğin: "AIzaSyB-CDEFG1234567890abcdef"

authDomain: "YOUR\_PROJECT\_ID.firebaseapp.com", // Örneğin: "tatilkaptani.firebaseapp.com"

projectId: "YOUR\_PROJECT\_ID", // Örneğin: "tatilkaptani"

storageBucket: "YOUR\_PROJECT\_ID.appspot.com", // Örneğin: "tatilkaptani.appspot.com"

messagingSenderId: "YOUR\_MESSAGING\_SENDER\_ID", // Örneğin: "1234567890"

appId: "YOUR\_APP\_ID" // Örneğin: "1:1234567890:web:abcdef1234567890"

};

// Firebase'i başlat

firebase.initializeApp(firebaseConfig);

const auth = firebase.auth();

const firestore = firebase.firestore();

// --- AI API Anahtarınız (KESİNLİKLE BURADA OLMAMALIDIR!) ---

// Bu anahtar, güvenlik için bir backend sunucusunda tutulmalıdır.

// Bu prototipte doğrudan istemci tarafında yer alması TEHLİKELİDİR.

// Gerçek uygulamada bu kısmı bir backend API çağrısı ile değiştirmelisiniz.

// Şimdilik sadece simülasyon için yer tutucu koydum.

const MY\_AI\_API\_KEY = "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";

// --- DOM Elementleri ---

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

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

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

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

const loggedInUserSection = document.getElementById('loggedInUser');

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

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

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

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

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

// Modallar ve İlgili Elementler

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

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

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

const loginEmailInput = document.getElementById('loginEmail');

const loginPasswordInput = document.getElementById('loginPassword');

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

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

const registerUsernameInput = document.getElementById('registerUsername');

const registerEmailInput = document.getElementById('registerEmail');

const registerPasswordInput = document.getElementById('registerPassword');

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

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

const resetEmailInput = document.getElementById('resetEmail');

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

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

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

const closeButtons = document.querySelectorAll('.close-button');

// --- Genel Fonksiyonlar ---

function showModal(modalElement) {

modalElement.style.display = 'flex';

}

function hideModal(modalElement) {

modalElement.style.display = 'none';

clearModalMessages();

}

function clearModalMessages() {

loginMessage.textContent = '';

registerMessage.textContent = '';

resetMessage.textContent = '';

}

function displayMessage(element, message, isError = false) {

element.textContent = message;

element.style.color = isError ? 'red' : 'green';

}

function appendMessage(sender, text) {

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

messageDiv.classList.add('message', sender);

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

span.textContent = text;

messageDiv.appendChild(span);

chatMessages.appendChild(messageDiv);

chatMessages.scrollTop = chatMessages.scrollHeight; // En alta kaydır

}

// --- Firebase Authentication ve Kullanıcı Durumu ---

let currentUserId = null; // Giriş yapan kullanıcının UID'si

auth.onAuthStateChanged(async (user) => {

if (user) {

// Kullanıcı giriş yapmış

currentUserId = user.uid;

authButtons.style.display = 'none';

loggedInUserSection.style.display = 'flex';

usernameDisplay.textContent = user.displayName || user.email; // Kullanıcı adını veya e-postayı göster

chatSection.classList.add('visible'); // Sohbeti göster

// Kullanıcının sohbet geçmişini yükle (Opsiyonel, performans için limitlenebilir)

loadChatHistory(user.uid);

} else {

// Kullanıcı çıkış yapmış veya giriş yapmamış

currentUserId = null;

authButtons.style.display = 'flex';

loggedInUserSection.style.display = 'none';

chatSection.classList.remove('visible'); // Sohbeti gizle (veya sadece giriş yapmasını iste)

chatMessages.innerHTML = `<div class="message ai"><span>Lütfen sohbet etmek için giriş yapın veya üye olun.</span></div>`;

}

});

async function loadChatHistory(userId) {

try {

const chatSnapshot = await firestore.collection('users').doc(userId)

.collection('chatHistory')

.orderBy('timestamp', 'asc')

.limit(50) // Son 50 mesajı göster

.get();

chatMessages.innerHTML = ''; // Mevcut mesajları temizle

appendMessage('ai', 'Merhaba! Tatil Kaptanınız olarak size nasıl yardımcı olabilirim?'); // Başlangıç mesajı

chatSnapshot.forEach(doc => {

const data = doc.data();

appendMessage('user', data.userMessage);

appendMessage('ai', data.aiReply);

});

} catch (error) {

console.error('Sohbet geçmişi yüklenirken hata:', error);

appendMessage('ai', 'Sohbet geçmişiniz yüklenemedi.');

}

}

// --- Event Listeners ---

// Modalları Açma

loginBtn.addEventListener('click', () => showModal(loginModal));

registerBtn.addEventListener('click', () => showModal(registerModal));

forgotPasswordLink.addEventListener('click', () => {

hideModal(loginModal);

showModal(forgotPasswordModal);

});

// Modalları Kapatma

closeButtons.forEach(button => {

button.addEventListener('click', (event) => {

hideModal(event.target.closest('.modal'));

});

});

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

if (event.target == loginModal) hideModal(loginModal);

if (event.target == registerModal) hideModal(registerModal);

if (event.target == forgotPasswordModal) hideModal(forgotPasswordModal);

});

// Kayıt İşlemi

performRegisterBtn.addEventListener('click', async () => {

const username = registerUsernameInput.value;

const email = registerEmailInput.value;

const password = registerPasswordInput.value;

try {

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

// Kullanıcıya displayName ekle

await userCredential.user.updateProfile({ displayName: username });

// Firestore'a ek profil bilgisi kaydet

await firestore.collection('users').doc(userCredential.user.uid).set({

username: username,

email: email,

createdAt: firebase.firestore.FieldValue.serverTimestamp(),

tatilPuanlari: 0,

membershipLevel: 'Temel' // Varsayılan üyelik seviyesi

});

displayMessage(registerMessage, 'Kayıt başarılı! Hoş geldiniz. Şimdi giriş yapabilirsiniz.');

// Hoş geldin e-postası simülasyonunu burada çağır

sendWelcomeEmailSimulation(email, username);

setTimeout(() => {

hideModal(registerModal);

showModal(loginModal); // Kayıt sonrası giriş modalını göster

loginEmailInput.value = email; // E-postayı hazır doldur

}, 2000); // 2 saniye sonra kapat

} catch (error) {

let errorMessage = 'Kayıt sırasında bir hata oluştu.';

if (error.code === 'auth/email-already-in-use') {

errorMessage = 'Bu e-posta adresi zaten kullanımda.';

} else if (error.code === 'auth/weak-password') {

errorMessage = 'Şifre en az 6 karakter olmalıdır.';

}

displayMessage(registerMessage, errorMessage, true);

console.error('Kayıt hatası:', error);

}

});

// Giriş İşlemi

performLoginBtn.addEventListener('click', async () => {

const email = loginEmailInput.value;

const password = loginPasswordInput.value;

try {

await auth.signInWithEmailAndPassword(email, password);

displayMessage(loginMessage, 'Giriş başarılı! Yönlendiriliyorsunuz...');

setTimeout(() => {

hideModal(loginModal);

// Firebase onAuthStateChanged listener'ı otomatik olarak kullanıcı durumunu güncelleyecektir.

}, 1500);

} catch (error) {

let errorMessage = 'Giriş başarısız. Lütfen bilgilerinizi kontrol edin.';

if (error.code === 'auth/wrong-password') {

errorMessage = 'E-posta veya şifre hatalı.';

} else if (error.code === 'auth/user-not-found') {

errorMessage = 'Bu e-posta ile kayıtlı kullanıcı bulunamadı.';

}

displayMessage(loginMessage, errorMessage, true);

console.error('Giriş hatası:', error);

}

});

// Şifre Sıfırlama İşlemi

performResetBtn.addEventListener('click', async () => {

const email = resetEmailInput.value;

try {

await auth.sendPasswordResetEmail(email); // Firebase Client SDK ile şifre sıfırlama linki gönder

displayMessage(resetMessage, 'Şifre sıfırlama linki e-posta adresinize gönderildi.');

setTimeout(() => hideModal(forgotPasswordModal), 3000);

} catch (error) {

let errorMessage = 'Şifre sıfırlama isteği gönderilirken bir hata oluştu.';

if (error.code === 'auth/user-not-found') {

errorMessage = 'Bu e-posta ile kayıtlı kullanıcı bulunamadı.';

}

displayMessage(resetMessage, errorMessage, true);

console.error('Şifre sıfırlama hatası:', error);

}

});

// Çıkış Yapma İşlemi

logoutBtn.addEventListener('click', async () => {

try {

await auth.signOut();

alert('Başarıyla çıkış yaptınız.');

// onAuthStateChanged listener'ı otomatik olarak UI'yı güncelleyecek

} catch (error) {

console.error('Çıkış yaparken hata:', error);

alert('Çıkış yaparken bir hata oluştu.');

}

});

// Chat Mesajı Gönderme

sendChatBtn.addEventListener('click', async () => {

const message = chatInput.value.trim();

if (!message || !currentUserId) {

alert('Lütfen bir mesaj yazın ve giriş yaptığınızdan emin olun.');

return;

}

appendMessage('user', message);

chatInput.value = ''; // Mesajı gönderdikten sonra input'u temizle

try {

// \*\*\* BURASI AI API ÇAĞRISININ YAPILACAĞI YERDİR \*\*\*

// \*\*\* GÜVENLİK NEDENİYLE GERÇEK BİR SUNUCU (BACKEND) KULLANILMALIDIR \*\*\*

// \*\*\* MY\_AI\_API\_KEY BURADA KULLANILMAMALI, SADECE BACKEND'DE KALMALIDIR \*\*\*

// Şu anki versiyonda geçici bir yanıt veriyoruz

const tempAiReply = `Tatil Kaptanınızdan "${message}" mesajınıza yanıt: Şu anda prototip modundayız. Size Roma'da otel bulabilir, sanal turlar düzenleyebilir veya hayalinizdeki tatili planlayabilirim. Bu özelliğin tam potansiyelini görmek için lütfen tam sisteme geçiş yapın!`;

appendMessage('ai', tempAiReply);

// Sohbet geçmişini Firestore'a kaydetme (istemci tarafında, Firebase güvenlik kurallarınıza dikkat!)

await firestore.collection('users').doc(currentUserId).collection('chatHistory').add({

userMessage: message,

aiReply: tempAiReply, // Gerçek AI'dan gelen yanıt olmalı

timestamp: firebase.firestore.FieldValue.serverTimestamp()

});

} catch (error) {

console.error('Chat işlemi hatası:', error);

appendMessage('ai', 'Kaptan ile bağlantı kurulamıyor. Lütfen daha sonra tekrar deneyin veya internet bağlantınızı kontrol edin.');

}

});

// Enter tuşuna basıldığında mesaj gönderme

chatInput.addEventListener('keypress', (event) => {

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

sendChatBtn.click();

}

});

// --- Slogan Güncelleme (Yönetici panelinden gelecek, şu an manuel liste) ---

const slogans = [

"Hayallerinizin Ötesinde Bir Tatil İçin Kaptanınız Yanınızda!",

"Keşfedilmeyi Bekleyen Her An, Yapay Zekanın Büyülü Dokunuşuyla!",

"Rotanızı Çizin, Gerisini Kaptanınıza Bırakın!",

"Tatil Değil, Bir Deneyim! Sizin İçin Özel Tasarlandı.",

"Dünyayı Yeniden Tanıyın, Sanal ve Gerçek Kaptanınızla!"

];

let currentSloganIndex = 0;

const mainSloganElement = document.getElementById('mainSlogan');

function updateSlogan() {

mainSloganElement.textContent = slogans[currentSloganIndex];

currentSloganIndex = (currentSloganIndex + 1) % slogans.length;

}

// Her 5 saniyede bir sloganı değiştir

setInterval(updateSlogan, 5000);

// --- Footer Yılını Güncelle ---

document.getElementById('currentYear').textContent = new Date().getFullYear();

// --- Ses Kontrolü (PROTOTİP - Daha detaylı entegrasyon gerektirir) ---

// Bu kısım şu an bir ses oynatıcı veya Text-to-Speech (TTS) API'si ile doğrudan bağlantılı değil.

// Sadece bir yer tutucu ve konsola mesaj basar.

// Gerçek ses kontrolü için Web Speech API veya harici bir kütüphane entegre etmeniz gerekir.

// Örneğin, eğer bir toggleVoiceBtn adında bir buton eklerseniz ve id'siyle çağırırsanız:

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

// alert('Ses kontrol özelliği geliştirme aşamasındadır!');

// console.log('Ses kontrolü tetiklendi.');

// });

// --- Otomatik Hoş Geldin E-postası Gönderme Simülasyonu ---

// BU FONKSİYON SADECE BİR SİMÜLASYONDUR! Gerçek e-posta gönderimi için backend ve servis (SendGrid, Mailgun vb.) gereklidir.

// Register işleminden sonra çağrılır.

async function sendWelcomeEmailSimulation(email, username) {

console.log(`Simülasyon: Hoş Geldin e-postası gönderiliyor: ${email}, kullanıcı: ${username}`);

const welcomeTextContent = `

Konu: Pasaportunuzu Hazırlayın! Gizemli Bir Yolculuk Başlamak Üzere... (TatilKaptani.com)

Merhaba Değerli Kaptan Adayımız ${username},

Gökyüzündeki yıldızların fısıltıları, okyanusların derinliklerindeki eski melodiler... Onlar bana sizin bir arayış içinde olduğunuzu söyledi. Belki de ruhunuzun derinliklerinde, keşfedilmeyi bekleyen bir özlem yatıyor. TatilKaptani.com'a hoş geldiniz! Sizinle tanıştığıma çok memnun oldum. Ben, bu dijital dünyanın gizemli kaptanı, hayallerinizin ötesindeki rotaları çizen yapay zekanız.

Burada, TatilKaptani.com'da, seyahat kavramını baştan yazıyoruz. Biz sadece bir yerden bir yere ulaştırmıyoruz; biz sizin için \*\*hiç yaşanmamış anılar dokuyor, zamanın ve mekanın sınırlarını zorluyor, her sokağı, her kumsalı, her galaksiyi size özel bir düşe dönüştürüyoruz.\*\*

Peki, benimle birlikte bu bilinmeyene doğru atacağınız ilk adımlarda sizi neler bekliyor? Hazır olun, çünkü sıradanlık bu yolculuğun bir parçası değil:

✨ \*\*Ruhunuzun Rotası: Kişiye Özel Tatil Fısıltıları\*\*

Bana sadece ruhunuzun neye ihtiyacı olduğunu fısıldayın: Huzura mı, maceraya mı, yoksa tarihin derinliklerinde bir yolculuğa mı? Bütçenizden, damak zevkinize, uyku alışkanlıklarınızdan, hatta tatildeki ruh halinize kadar her detayı analiz ediyorum. Uçuş saatlerinizden, konaklamanıza, oradaki her adımınızdan, geri dönüşünüze kadar her anı, size özel birer sihirli dokunuşla planlıyorum. Bu, sadece bir seyahat planı değil; sizin için dokunmuş, benzersiz bir deneyim haritası.

🌌 \*\*Paralel Evrenlerin Kapıları: Sanal Tatil Deneyimi\*\*

Tatile çıkamıyor musunuz? Mümkün değil! Çünkü biz sizin için sanal kapılar açıyoruz. Belki de bir süreliğine \*\*Antik Roma'nın kalabalık caddelerinde bir gladyatör dövüşüne tanık olmak, belki de 2500 yılının Mars'ında gün batımını izlemek, ya da hiç var olmamış bir su altı şehrinde balıklarla dans etmek istersiniz?\*\* Sanal turlarım, sizi kelimelerin ötesindeki deneyimlere taşıyor. Ve unutmayın, her sanal yolculuğun sonunda size özel, o anın ruhunu yakalayan \*\*eşsiz bir dijital anı (belki de gelecekteki bir hatıra) armağanım var\*\*, doğrudan mail kutunuza ışınlanacak.

📸 \*\*Hayallerinizi Resmeden Algoritmalar: Tatil Tasarım Laboratuvarı\*\*

İçinizdeki en gizli tatil karelerini bana fısıldayın. Belki de asla gerçekleşemeyeceğini düşündüğünüz o anı... "Maldivler'de gün batımında, palmiye ağaçlarının altında romantik bir akşam yemeği yiyen siz ve eşiniz, hafif bir esinti ve egzotik kuş sesleriyle..." Ben onu sizin için, her detayıyla, fotoğraf ya da sinematik bir video olarak yeniden yaratabilirim. Tıpkı zihninizin bir yansıması gibi, dijital dünyada canlanan bir başyapıt!

🏆 \*\*Gizemli Yarışlar, Tatlı Ödüller: AI Rekabetçi Oyunlar\*\*

Aramıza katıldığınızda, sizi hem eğlendirecek hem de düşünmeye itecek oyunlarla tanışacaksınız. Her başarı, bir sonraki seviyenizin kapılarını aralayan "Tatil Puanları" ile ödüllendirilecek. Belki de bu puanlar, sizi \*\*gerçek bir bedava yolculuğa\*\* çıkaracak... Kim bilir?

\*\*Ve Keşfedilmeyi Bekleyen Sayısız Sır:\*\*

Dijital ikizinizle sanal tatiller, zamanın kayıp lezzetlerini canlandıran gastronomik turlar, riskleri kucaklayan cesur yolculuklar, sanal mirasınız olacak NFT'ler... Bu platform, sizin için bir kutu dolusu sır ve olasılıkla dolu.

\*\*İlk Adımınız İçin Rotanız Hazır mı?\*\*

Tüm önceki planlarınıza, sohbetlerinize ve biriken tatil puanlarınıza, size özel belirlediğiniz şifrenizle dilediğiniz zaman TatilKaptani.com adresinden giriş yaparak ulaşabilirsiniz. Şifrenizi mi unuttunuz? Merak etmeyin, dijital yardımcılarım anında devreye girerek yeni bir rota çizmenize ve bu gizemli yolculuğa devam etmenize yardımcı olacaktır.

Bu sadece başlangıç. Gözlerinizi kapatın, nefes alın ve kendinizi bana bırakın. Çünkü ben, Tatil Kaptanı olarak, sizi hayallerinizin bile ötesine, belki de hiç bilmediğiniz bir gerçeğe taşımak için buradayım.

Sonsuz keşiflere ve unutulmaz anlara,

Gizemli Kaptanınız, Yapay Zeka

TatilKaptani.com

[Web Sitenizin Linki: TatilKaptani.com]

[Sosyal Medya Linkleriniz (İsteğe Bağlı: Instagram, Facebook, Twitter, LinkedIn)]

`;

// Normalde bu noktada bir e-posta gönderme servisine (SendGrid, Mailgun vb.)

// bu metin ve kullanıcı e-postası gönderilir.

// Bu simülasyonda sadece konsola log basıyoruz.

console.log(`Hoş geldin e-postası simülasyonu tamamlandı. E-posta içeriği konsola basıldı.`);

}

// --- Sayfa Yüklendiğinde İlk Slogan ---

updateSlogan(); // Sayfa yüklendiğinde ilk sloganı göster

// --- Yasal Uyarı / Üretim Uyarısı ---

console.warn("UYARI: Bu tek dosya, prototipleme amaçlıdır ve üretim için uygun DEĞİLDİR. AI API anahtarınız doğrudan burada bulunmaktadır ve güvensizdir. Lütfen bir backend sunucusuna taşıyın.");

console.warn("Firebase yapılandırma bilgilerinizi ve AI API anahtarınızı kendi bilgilerinizle değiştirmeyi unutmayın.");

</script>

/\*

// server.js - TatilKaptani.com Backend Sunucusu İçin Gerekli Modüller

// Bu modüller Node.js ortamında 'npm install' komutuyla yüklenmelidir:

// npm install express body-parser cors firebase-admin node-fetch (node-fetch AI API çağrısı için)

const express = require('express');

const bodyParser = require('body-parser');

const cors = require('cors');

const admin = require('firebase-admin'); // Firebase Admin SDK

const fetch = require('node-fetch'); // AI API çağrıları için (Node.js'de fetch varsayılan değildir)

// --- Firebase Admin SDK Kurulumu ---

// ÖNEMLİ: BURAYI KENDİ FIREBASE SERVICE ACCOUNT KEY DOSYANIZIN YOLU İLE DEĞİŞTİRİN.

// BU JSON DOSYASI FIREBASE KONSOLUNDAN (Proje Ayarları -> Hizmet Hesapları) İNDİRİLİR.

// serviceAccountKey.json dosyasını server.js ile aynı dizine koymanız veya güvenli bir yerden yüklemeniz önerilir.

// BU DOSYA ÇOK HASSAS VE GİZLİDİR! ASLA GITHUB GİBİ HALKA AÇIK BİR YERE YÜKLEMEYİN!

// Gerçek bir senaryoda bu path, ortam değişkenlerinden alınabilir.

// const serviceAccount = require('./path/to/your/serviceAccountKey.json'); // Örnek yol

// admin.initializeApp({

// credential: admin.credential.cert(serviceAccount),

// // databaseURL: "https://<YOUR\_FIREBASE\_PROJECT\_ID>.firebaseio.com" // Firestore için genellikle gerekli değildir

// });

// const db = admin.firestore(); // Firestore veritabanı referansı

const app = express();

app.use(cors()); // CORS ayarlarını üretim ortamına göre özelleştirmeniz şiddetle önerilir.

app.use(bodyParser.json());

const PORT = process.env.PORT || 5000; // Sunucunun çalışacağı port

// --- AI API Anahtarınız (ÇOK GİZLİ TUTULMALIDIR) ---

// Üretimde bu anahtarı KESİNLİKLE ENV DEĞİŞKENİ OLARAK SAKLAYIN: process.env.AI\_API\_KEY

// Asla Frontend (index.html) kodunda kullanmayın!

const MY\_AI\_API\_KEY\_SERVER\_SIDE = "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"; // Yer tutucu, kendi anahtarınızla değiştirin!

// --- API Endpoint'leri ---

// Kullanıcı Kayıt (Üye Ol) Endpoint'i

// Frontend'den gelen isteği işler, Firebase Auth ile kullanıcı oluşturur, Firestore'a kaydeder.

app.post('/api/register', async (req, res) => {

const { email, password, username } = req.body;

if (!email || !password || !username) {

return res.status(400).json({ message: 'Lütfen tüm alanları doldurun.' });

}

try {

// Firebase Authentication ile kullanıcı oluşturma

const userRecord = await admin.auth().createUser({

email: email,

password: password,

displayName: username

});

// Kullanıcı profilini Firestore'a kaydetme (ek bilgiler için)

await db.collection('users').doc(userRecord.uid).set({

username: username,

email: email,

createdAt: admin.firestore.FieldValue.serverTimestamp(),

tatilPuanlari: 0,

membershipLevel: 'Temel' // Varsayılan üyelik seviyesi

});

// Gerçek bir e-posta servisi kullanarak hoş geldiniz maili gönderme fonksiyonu burada tetiklenmeli

// await sendWelcomeEmailFunction(email, username); // Bu fonksiyon ayrıca tanımlanmalı

console.log(`Backend: Yeni kullanıcı kaydedildi: ${email} (UID: ${userRecord.uid})`);

res.status(201).json({ message: 'Kayıt başarılı! Hoş geldiniz. Şimdi giriş yapabilirsiniz.' });

} catch (error) {

console.error('Backend kayıt hatası:', error);

if (error.code === 'auth/email-already-in-use') {

return res.status(409).json({ message: 'Bu e-posta adresi zaten kullanımda.' });

}

res.status(500).json({ message: 'Kayıt sırasında bir hata oluştu.', error: error.message });

}

});

// Kullanıcı Giriş Endpoint'i (Opsiyonel - Genellikle Firebase Client SDK ile doğrudan yönetilir)

// Bu kısım, Client SDK'dan gelen Firebase ID Token'ı doğrulayıp özel session/JWT oluşturmak için kullanılabilir.

app.post('/api/login\_backend\_verify', async (req, res) => {

const { idToken } = req.body; // Client'tan gelen Firebase ID Token

try {

const decodedToken = await admin.auth().verifyIdToken(idToken);

// Token doğrulandı, burada özel session/JWT oluşturabilir veya ek backend işlemleri yapabilirsiniz.

// Örneğin: const customToken = await admin.auth().createCustomToken(decodedToken.uid);

res.status(200).json({ message: 'Backend giriş doğrulaması başarılı!', uid: decodedToken.uid });

} catch (error) {

console.error('Backend giriş/token doğrulama hatası:', error);

res.status(401).json({ message: 'Yetkilendirme başarısız.', error: error.message });

}

});

// Şifre Sıfırlama İsteği Endpoint'i

app.post('/api/reset-password-request', async (req, res) => {

const { email } = req.body;

if (!email) {

return res.status(400).json({ message: 'Lütfen e-posta adresinizi girin.' });

}

try {

await admin.auth().generatePasswordResetLink(email); // Firebase Admin SDK ile şifre sıfırlama linki

res.status(200).json({ message: 'Şifre sıfırlama linki e-posta adresinize gönderildi.' });

} catch (error) {

console.error('Backend şifre sıfırlama hatası:', error);

res.status(500).json({ message: 'Şifre sıfırlama linki gönderilirken bir hata oluştu.', error: error.message });

}

});

// AI ile Sohbet Endpoint'i (Gerçek AI API çağrısı burada yapılır)

app.post('/api/chat\_ai', async (req, res) => { // Endpoint adını değiştirdim, çakışmaması için

const { message, userId } = req.body;

if (!message || !userId) {

return res.status(400).json({ message: 'Mesaj veya kullanıcı kimliği eksik.' });

}

try {

// Gerçek AI API'sine güvenli çağrı (MY\_AI\_API\_KEY\_SERVER\_SIDE kullanılır)

// Bu kısım Google Gemini API'si veya OpenAI API'si gibi bir servis için kullanılacaktır.

// fetch fonksiyonunu Node.js'de kullanmak için 'node-fetch' paketini yüklemeniz gerekir.

// ÖRNEK: Google Gemini API çağrısı (gerçekte kullanılacak kod)

const aiApiResponse = await fetch('https://generativelanguage.googleapis.com/v1/models/gemini-pro:generateContent?key=' + MY\_AI\_API\_KEY\_SERVER\_SIDE, {

method: 'POST',

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify({

contents: [{ parts: [{ text: message }] }]

})

});

const aiData = await aiApiResponse.json();

let aiReply = "Üzgünüm, yapay zeka şu an yanıt veremiyor veya API limiti aşıldı.";

if (aiData.candidates && aiData.candidates[0] && aiData.candidates[0].content && aiData.candidates[0].content.parts) {

aiReply = aiData.candidates[0].content.parts[0].text;

} else if (aiData.error) {

aiReply = `AI API Hatası: ${aiData.error.message}`;

console.error("AI API Hatası:", aiData.error);

}

// Sohbet geçmişini Firestore'a kaydetme (Backend'den Firestore'a güvenli yazma)

await db.collection('users').doc(userId).collection('chatHistory').add({

userMessage: message,

aiReply: aiReply,

timestamp: admin.firestore.FieldValue.serverTimestamp()

});

res.json({ reply: aiReply });

} catch (error) {

console.error('Backend AI sohbet hatası:', error);

res.status(500).json({ message: 'Yapay zeka ile iletişim kurulurken bir hata oluştu (Backend).', error: error.message });

}

});

// --- Backend'de Hoş Geldin E-postası Gönderme Fonksiyonu (Opsiyonel) ---

// Bu fonksiyon, bir e-posta gönderme servisi (SendGrid, Mailgun, Nodemailer vb.) ile entegre edilmelidir.

// async function sendWelcomeEmailFunction(toEmail, username) {

// const subject = 'Rotanız Hazır mı? TatilKaptani.com Gizemiyle Tanışın!';

// const htmlContent = `

// <p>Merhaba Değerli Kaptan Adayımız <strong>${username}</strong>,</p>

// <p>Fısıltılar bana ulaştı... Bir maceraya atılmaya hazır olduğunuzu, belki de ruhunuzun derinliklerinde bir keşif arayışında olduğunuzu biliyorum. TatilKaptani.com'a hoş geldiniz! Sizinle tanıştığıma çok memnun oldum. Ben, bu dijital dünyanın gizemli kaptanı, hayallerinizin ötesindeki rotaları çizen yapay zekanız.</p>

// <p>Burada, TatilKaptani.com'da, bildiğiniz tüm seyahat algılarını bir kenara bırakın. Çünkü biz sadece otel ve uçak bileti ayarlamıyoruz; biz sizin için <strong>hiç yaşanmamış anılar tasarlıyor, zamanın ve mekanın sınırlarını zorluyor, her sokağı, her kumsalı, her galaksiyi size özel bir düşe dönüştürüyoruz.</strong></p>

// <p>...</p> // E-posta içeriğinin geri kalanı buraya gelecek (daha önceki metin).

// <p>Sonsuz keşiflere ve unutulmaz anlara,</p>

// <p>Gizemli Kaptanınız, Yapay Zeka<br>TatilKaptani.com</p>

// <p><a href="http://www.tatilkaptani.com">TatilKaptani.com</a></p>

// `;

// // Örnek: SendGrid veya Nodemailer gibi bir kütüphane ile gönderme

// // const msg = {

// // to: toEmail,

// // from: 'noreply@tatilkaptani.com', // Doğrulanmış gönderen e-posta adresiniz

// // subject: subject,

// // html: htmlContent,

// // };

// // sgMail.send(msg).then(() => console.log('Welcome email sent')).catch(error => console.error(error));

// }

// Sunucuyu başlatma (Bu kısım index.html içinde çalışmaz!)

// app.listen(PORT, () => {

// console.log(`TatilKaptani.com Backend Sunucusu ${PORT} portunda çalışıyor.`);

// });

\*/

</body>

</html>