StokSohbet

Metin Huneli Erdamar

İçindekiler

1. Giriş: Proje Özeti

Bu proje, turizm sektöründe faaliyet gösteren otel işletmelerinin envanter stoğundaki ürünler, kodlar ve alternatif bilgiler hakkında, otel çalışanlarının doğal dilde sorduğu sorulara yanıt verebilen AI destekli dinamik bir sohbet uygulaması oluşturmayı amaçlamaktadır. Bu sayede, kullanıcılar bir balık türünü veya herhangi bir ürünü sorduğunda, yapay zeka botu ilgili bilgileri ve benzer alternatifleri, ürün kodlarıyla birlikte sunabilecektir.

2. Kullanılan Teknolojiler

2.1. Frontend

• Next.js: Modern, dinamik ve SEO dostu arayüzler oluşturmak, hem sunucu hem de istemci tarafı render işlemlerini desteklemek için tercih edilmiştir.

• React & TypeScript: UI bileşenleri ile state yönetimi ve tip güvenliği sağlanarak, geliştirmenin bakımını kolaylaştırmak ve hata oranını düşürmek hedeflenmiştir.

• Tailwind CSS: Hızlı, esnek ve responsive tasarımlar oluşturarak, açık ve koyu (dark mode) temaların kolayca uygulanmasına imkân tanımaktadır.

2.2. Backend

• Node.js & Next.js API Routes: Sunucu tarafı işlemler ve API endpoint'leri, mikroservis benzeri bir yaklaşımla modüler şekilde yönetilmektedir.

• TypeScript: Kodun bakımını kolaylaştırmak ve tip güvenliği sağlamak amacıyla tercih edilmiştir.

2.3. MongoDB İletişimi

• MongoDB Atlas: Otel stok verileri ve ilgili meta verilerin bulut üzerinde saklanması için kullanılmıştır.

• Node.js MongoDB Driver: lib/mongodb.ts dosyasında, .env dosyasında tanımlı MONGODB_URI üzerinden güvenli ve esnek bir bağlantı sağlanmıştır.

2.4. Chatbot API Entegrasyonu

• Gemini 2.0 Flash: Projede, kullanıcı sorgularını stok verileriyle birlikte analiz edebilecek şekilde konfigüre edilen temel AI modelidir.

2.5. Deploy ve TypeScript Revizyonu

• Vercel: Hızlı dağıtım, otomatik CI/CD ve serverless fonksiyon yönetimi için kullanılmıştır.

• TypeScript Revizyonları: Deploy sırasında Vercel üzerinde karşılaşılan tip ve ESLint hataları, Next.js API route'larında NextRequest kullanımı ve MongoDB bağlantısı gibi alanlarda giderilmiştir.

3. Süreçte AI Kullanımı

3.1. Kullanılan AI Araçları

Projede AI destekli sohbet motoru için:

• ChatGPT, Claude, Bolt.new, Cursor gibi yardımcı yapay zeka araçlarından yararlanılmıştır.

3.2. AI Araçlarının Kullanım Süreci

• Temel Mimarinin Oluşturulması:

Proje iskeleti Next.js, Node.js ve MongoDB tabanlı olarak hazırlanmış; prompt mühendisliği ve stok bilgisi entegrasyonu yapılarak AI uygulaması şekillendirilmiştir.

• Web Uygulamasının Özelleştirilmesi:

Yapay zeka yardımıyla temel yapı projenin ihtiyaçlarına uyarlanmış, gerekli debugging ve deploy aşamaları tamamlanmıştır.

4. Chatbot Finetuning Süreci

4.1. Prompt İkilileri ve Self Prompting Yapısı

• Prompt İkilileri:

Soru-cevap uyumunu artırmak için, Gemini API'ye gönderilen prompt metni detaylandırılmış ve örnek istemler hazırlanmıştır.

• Self Prompting:

Yapay zekanın eksik veya belirsiz bilgileri algılayarak netleştirici sorular sorması sağlanmış, böylece kullanıcıdan ek bilgi alarak daha doğru yanıtlar üretilmiştir.

4.2. RAG (Retrieval-Augmented Generation) Kullanımı

• Tokenların Gruplanması:

Kullanıcı sorguları, anahtar kelime analiziyle işlenerek ilgili veritabanı kayıtları çekilmiştir.

• Bağlam Bazlı Yanıt Üretimi:

Genel konular için geniş bağlam kullanılırken, stok odaklı sorgular için stok verileri temel alınarak cevaplar üretilmiştir.

5. Deploy Sürecinde Vercel için Debugging ve TypeScript Problemlerinin Çözümü

• Build Hatalarının Giderilmesi:

Vercel üzerinde karşılaşılan TypeScript, ESLint ve modül formatı sorunları düzeltilerek (örn. Next.js API route'larında NextRequest kullanımı, tailwind.config.js dosyasının ESM formatına dönüştürülmesi) sorunsuz bir dağıtım sağlanmıştır.

• Serverless Function İsimlendirme:

Proje adının sadeleştirilmesi ve package.json'daki "name" alanının özel karakterlerden arındırılması, Vercel fonksiyon isimlendirme kısıtlamalarına uyumluluk sağlamıştır.

• Environment Variables:

MONGODB_URI, GEMINI_API_KEY gibi değişkenler Vercel üzerinde doğru şekilde yapılandırılmıştır.

6. Test ve Kullanıcı Deneyimi Geri Bildirimlerine Dayalı Geliştirme

• Kullanıcı Testleri:

Uygulama, yerel geliştirme ortamında ve Vercel üzerindeki canlı dağıtımda test edilerek deneyim raporları alınmıştır.

• Geri Bildirim Toplama:

API logları ve MongoDB'deki "feedback" koleksiyonu incelenerek, sorgu yetersizliği veya belirsizliği tespit edilmiş, buna göre prompt ve sorgu sınıflandırması iyileştirilmiştir.

• İteratif Geliştirme:

Kullanıcı geri bildirimlerine göre yanıt kalitesi ve arayüz tasarımı sürekli olarak iyileştirilmiştir.

7. GitHub'da Sürüm Yönetimi ve Branching Stratejisi

• Git Kullanımı:

Tüm kaynak kod, GitHub'da sürüm kontrolü ile yönetilmiştir.

• Branching Stratejisi, Commit Mesajları ve Sürüm Numaralandırması:

Geliştirmeler anlaşılır commit mesajları ve düzenli sürüm numaralandırma ile takip edilmiş, feature branch'ler üzerinden ana dal (main) ile birleştirme işlemleri yürütülmüştür.

8. Sonuç

Bu proje, otel işletmelerinin stok yönetim sistemi kapsamında AI destekli bir sohbet uygulaması sunmaktadır. Next.js, Node.js, MongoDB ve Gemini API gibi modern teknolojilerin bütünleşik kullanımıyla hayata geçen proje, yapay zeka araçlarının kullanımı, prompt fine-tuning, RAG yaklaşımı, Vercel üzerinde dağıtım ve GitHub üzerinden sürüm kontrolü gibi aşamalarda detaylıca incelenmiş ve başarıyla tamamlanmıştır.