Bölüm:Yazılım Mühendisliği

Ders Adı: Yazılım İnşası

Asst.Prof.Dr. AYŞE NUR ALTINTAŞ TANKÜL

teslim tarihi : 12/24

YAPAY ZEKA DESTEKLİ SOHBET SİSTEMİ (AI CHAT BOT)

PROJE TASARIM RAPORU

1	Abdulmouen Kanatri	2310238519
2	MOHAMED ABIYA	2410238560

İ

1. Sistemin Amacı

Bu proje, kullanıcıların gelişmiş Büyük Dil Modelleri (LLM) ile doğal dilde iletişim kurmasını sağlayan, modern ve ölçeklenebilir bir "AI Chat Bot" sistemi geliştirmeyi hedefler. Sistem, kullanıcıların sorularına anlık (real-time) yanıtlar almasını, yazılım kodları oluşturmasını ve geçmiş sohbetlerini kayıt altında tutmasını sağlar.

1.1 Tasarım Hedefleri

1.1.1. Kullanılabilirlik Sistemin arayüzü, teknik bilgisi olmayan kullanıcıların bile rahatça kullanabileceği sadelikte (Clean UI) tasarlanmıştır. "Sohbet et, kaydet ve yönet" prensibine odaklanılmıştır.

1.1.2. Performans Yapay zeka yanıtlarının gecikmesiz (Low Latency) sunulması kritik öneme sahiptir. Bu nedenle "Streaming" teknolojisi kullanılarak, yanıtın tamamının oluşması beklenmeden kelime kelime ekrana yansıtılması sağlanmıştır.

1.1.3. Veri Güvenliği Kullanıcıların sohbet geçmişi ve kimlik bilgileri, güvenli veritabanı bağlantıları ve şifreleme yöntemleri ile korunur. API anahtarları sunucu tarafında saklanır.

1.1.4. Ölçeklenebilirlik Next.js ve PostgreSQL altyapısı sayesinde sistem, artan kullanıcı sayısı ve mesaj trafiğini performans kaybı olmadan yönetebilecek şekilde tasarlanmıştır.

1.2 Ödünler ve Kriterler

1.2.1. Hız vs. Maliyet Daha hızlı yanıt süresi için Google Gemini Pro modeli tercih edilmiştir. Bu model, maliyet ve performans açısından en optimum dengeyi sunmaktadır.

1.2.2. Basitlik vs. Özellik Zenginliği Kullanıcı arayüzünde karmaşık ayarlardan kaçınılmış, bunun yerine odaklanmayı kolaylaştıran minimalist bir tasarım tercih edilmiştir.

2. Sistem Mimarisi

Sistem, **İstemci-Sunucu (Client-Server)** mimarisi üzerine kurulmuştur ve modern web standartlarına (T3 Stack benzeri) dayanır.

2.1 Alt Sistem Ayrıştırma

2.1.1. Kullanıcı Arayüzü (Frontend) Alt Sistemi

- **Teknoloji:** Next.js (React), Tailwind CSS.
- **Görev:** Kullanıcıdan girdi alma, mesajları listeleme, Markdown formatındaki yanıtları (kod blokları, tablolar) render etme.

2.1.2. İş Mantığı (Backend/API) Alt Sistemi

- **Teknoloji:** Next.js API Routes (Serverless Functions).
- **Görev:** İstemciden gelen mesajları doğrulama, veritabanına kaydetme ve AI servisine iletme.

2.1.3. Veri Yönetimi Alt Sistemi

- **Teknoloji:** Prisma ORM, PostgreSQL.
- **Görev:** Kullanıcı, Sohbet ve Mesaj verilerinin tutarlı bir şekilde saklanması ve sorgulanması.

2.2 Donanım/Yazılım Eşleme

2.2.1. Sunucu Tarafı

- Proje, Vercel veya benzeri bulut platformlarında çalışacak şekilde "Serverless" mimariye uygundur. Node.js çalışma zamanı (Runtime) gerektirir.

2.2.2. İstemci Tarafı

- Modern bir web tarayıcısı (Chrome, Edge, Safari) yeterlidir. Responsive tasarım sayesinde mobil cihazlarda da tam performansla çalışır.

2.3 Kalıcı Veri Yönetimi

Sistem, ilişkisel bir veritabanı (RDBMS) olan PostgreSQL kullanır. Veri bütünlüğü **Prisma ORM** aracılığıyla sağlanır.

- **Veri Saklama:** Sohbet geçmişi kalıcı olarak diskte saklanır.
- **Yedekleme:** Bulut tabanlı veritabanı sağlayıcısı (örn. Neon veya Supabase) tarafından otomatik yedekleme yapılır.

2.4 Erişim Kontrolü ve Güvenlik

2.4.1. Kimlik Doğrulama

- **NextAuth.js** kütüphanesi kullanılarak Google OAuth veya E-posta girişi sağlanır. Sadece giriş yapan kullanıcılar kendi verilerine erişebilir.

2.4.2. API Güvenliği

- Google Gemini API anahtarı sunucu tarafında (`.env` dosyasında) saklanır, istemci tarafına asla sızdırılmaz.

3. Alt Sistem Hizmetleri

3.1 Kullanıcı Arayüzü Alt Sistemi Hizmetleri

3.1.1. Sohbet Ekranı

- Mesajların balonlar (bubbles) içinde gösterilmesi.
- Kullanıcı mesajlarının sağda, bot yanıtlarının solda hizalanması.
- Yazıyor animasyonu (Typing indicator).

3.1.2. Sidebar (Yan Menü)

- Geçmiş sohbetlerin tarihe göre listelenmesi.
- Yeni sohbet başlatma butonu.
- Sohbet silme veya başlık düzenleme seçenekleri.

3.2 Yapay Zeka Entegrasyon Alt Sistemi

3.2.1. Prompt Mühendisliği (Prompt Engineering)

- Kullanıcı mesajı API'ye gönderilmeden önce, sistem tarafından özel bir "Sistem Mesajı" (System Instruction) eklenerek botun davranışları (örneğin: "Sen yardımsever bir asistansın") tanımlanır.

3.2.2. Akış Yönetimi (Streaming)

- API'den gelen veriler parça parça (chunk) alınır ve anında arayüze basılır, böylece kullanıcı beklemez.

4. Düşük Seviyeli Tasarım (Low-Level Design)

4.1 Son Nesne Tasarımı (Class Design)

4.1.1. ChatClient (Arayüz Bileşeni)

- **Açıklama:** Sohbet penceresini yöneten ana React bileşeni.
- **State (Durum):** `messages[]`, `input`, `isLoading`.
- **Metotlar:** `handleSubmit()`, `handleInputChange()`.

4.1.2. RouteHandler (API Yöneticisi)

- **Açıklama:** `/api/chat` adresine gelen POST isteklerini karşılar.
- **İşlev:** Gelen mesaj dizisini (Array) doğrular, Google API'sine gönderir ve dönen yanıtı Stream'e çevirir.

4.1.3. PrismaClient (Veritabanı Yöneticisi)

- **Açıklama:** Veritabanı ile iletişim kuran Singleton nesnesi.
- **Metotlar:** `prisma.conversation.create()`, `prisma.message.create()`.

4.2 Tasarım Kararları ve Desenleri

4.2.1. Optimistic UI (İyimser Arayüz)

- Kullanıcı mesajı gönderdiğinde, sunucudan yanıt gelmesi beklenmeden mesaj ekranda "gönderilmiş gibi" gösterilir. Bu, uygulamanın çok hızlı hissedilmesini sağlar.

4.2.2. Server-Side Rendering (SSR)

- İlk yüklemede sayfa içeriği sunucuda oluşturulur, bu da SEO ve ilk açılış hızı (FCP) için avantaj sağlar.

4.2.3. Component-Based Architecture

- Arayüz; `ChatList`, `ChatInput`, `MessageBubble` gibi küçük, yeniden kullanılabilir bileşenlere bölünmüştür. Bu, kodun bakımını ve test edilmesini kolaylaştırır.