Proje Dokümantasyonu

1. Proje Adı:

JobStreamline

2. Proje Tanımı:

Bu proje, işverenlerin iş ilanlarını oluşturmasını ve kullanıcıların bu ilanları arayarak bulmasını sağlayan bir iş arama platformu için geliştirilen bir sistemin mimarisini göstermektedir. Sistem; PostgreSQL, Elasticsearch, Redis, Company API ve Job API gibi farklı bileşenlerden oluşmaktadır.

3. Kullanılan Teknolojiler:

- **PostgreSQL**: İşveren kayıt verilerini depolamak için kullanılır.
- **Elasticsearch**: İş ilanlarını indekslemek ve hızlı arama işlemleri için kullanılır.
- **Redis**: Yasaklı kelimeler (BlackWords) listesi için hızlı ve geçici bir depolama çözümü olarak kullanılır.
- **.NET 8:** Backend API geliştirmesi için.
- **Hangfire**: Zamana bağlı çalışarak süresi dolmuş ilanları pasife çekmek için kullanılır.
- **Docker Compose:** Servislerin konteynerleştirilmesi ve orkestrasyonu.

4. Kullanılan Temel Tasarım Desenleri

- **Repository Pattern:** Veri erişim mantığını soyutlamak ve veri tabanı işlemlerini izole etmek için kullanılır.
- **Dependency Injection (DI):** .NET Core'un yerleşik IoC (Inversion of Control) konteyneri kullanılarak bağımlılık yönetimi sağlanmıştır.
- **Unit of Work:** Karmaşık işlemler gerçekleştirilirken veri tutarlılığını sağlamak için kullanılır.
- **DTO Pattern:** Katmanlar arası iletişimi sağlamak amacıyla veri transfer nesneleri (DTO) kullanılmıştır.

5. Mimari Tasarım:

- **İşveren**: İşverenler sisteme kayıt olabilir ve iş ilanlarını girebilir.
- **Kullanıcı**: Kullanıcılar iş ilanlarını arayabilir ve görüntüleyebilir.
- **API'ler**: Sistem, iki ana API üzerinden çalışır: Company API ve Job API.
- **Veri Tabanları**: PostgreSQL işveren kayıtlarını tutarken, Elasticsearch iş ilanlarını aramak için kullanılır.
- **Redis**: Yasaklı kelimeleri (BlackWords) tutmak için kullanılır.
- **Log**: Sistem üzerindeki log yapısı APM üzerinden sağlanmıştır.

6. Veri Akışı:

1. **Kayıt İşlemi (Company API):**
    - İşveren sisteme kayıt olur. 2 adet ilan hakkı varsayılan olarak tanımlanır.
    - İşveren bilgileri Company API tarafından alınır ve PostgreSQL veritabanına kaydedilir.
2. **İlan Girişi (Job API):**
    - İşveren, bir iş ilanı oluşturmak için Job API'yi kullanır.
    - İşveren'in ilan verme hakkı kalmadı ise hata mesajı fırlatılır.
    - İlan verileri Redis'te bulunan yasaklı kelimelere karşı kontrol edilir.
    - İlan verileri, yasaklı kelimeler içermiyorsa Elasticsearch'e "jobs" indeksi altında kaydedilir.
3. **İlan Arama (Job API):**
    - Kullanıcı, Job API üzerinden bir arama sorgusu gönderir.
    - Job API, Elasticsearch'e sorgu gönderir ve sonuçları kullanıcıya iletir.

7.API Endpoint'leri

İşveren (Employer) Kaynakları

- POST /api/company: Yeni işveren kaydı oluşturur.
- GET /api/company/{id}: İşveren bilgilerini getirir.

İş İlanı (Job) Kaynakları

- POST /api/job: Yeni iş ilanı oluşturur.
- GET /api/job/{id}: İş ilanı bilgilerini getirir.
- GET /api/job/search?={text}: Yayında kalma süresine göre aktif iş ilanlarını elastic üzerinden full-text arama yapar.

8. Yüksek Erişilebilirlik ve Performans Optimizasyonu:

- **API Geliştirme**: RESTful servisler olarak geliştirilebilir.
- **Veritabanı Yönetimi**: PostgreSQL ve Elasticsearch için ayrı veritabanı yönetimi gereklidir. Elasticsearch, özellikle arama performansı için optimize edilmelidir.
- **Redis Entegrasyonu**: Redis, yasaklı kelime listesi için kullanılır ve hafızada tutulur. Bu, hızlı kontrol mekanizması sağlar.
- **Dağıtım**: Her bileşen Docker gibi konteyner teknolojileri kullanılarak izole edilebilir ve yönetilebilir.

9. Güvenlik:

- **Token**: API'ler için JWT Bearer token ile yetkilendirme yapılmıştır.

10. Test:

- **Unit**: Sistemde yer alan API'lerin test metotları hazırlanmıştır.

11. Ortam Kurulumu

Gereksinimler

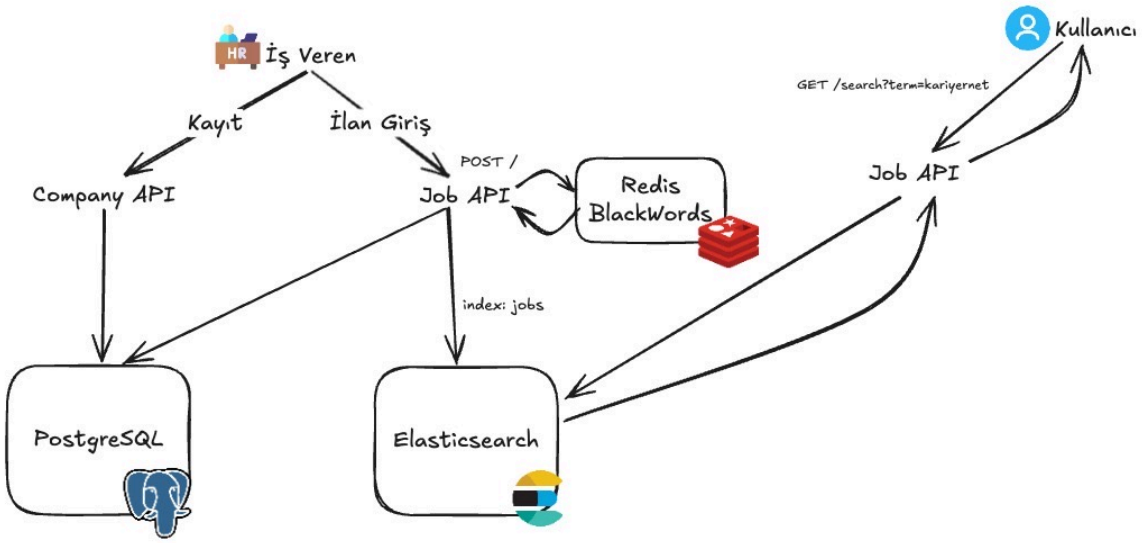
- Docker ve Docker Compose kurulumu yapılmalıdır.
- PostgreSQL, Redis ve Elasticsearch için sunucuda kullanılabilir portların açık olduğundan emin olunmalıdır.

Projenin Çalıştırılması

1. Depoyu klonlayın:
   git clone https://github.com/ceydaAcikgoz/JobStreamline.git
2. Proje dizinine gidin:
   cd JobStreamline/.devcontainer
3. Docker Compose ile servisleri başlatın:
   docker-compose up --build
4. VSCode ile Open-in-container ile açılıp debug modda run edilirse şu url'den proje çalışıyor;
   a. Swagger; http://localhost:5226/api/docs

12. Mimari Diagram:

Aşağıda projenin genel mimari yapısını gösteren bir diagram bulunmaktadır:



13. Sonuç

Bu backend sistemi, ilan yayınlama, arama ve kalite puanlama özellikleri ile güçlü ve ölçeklenebilir bir platform sunmaktadır. PostgreSQL, Redis ve Elasticsearch entegrasyonları, performansı ve ölçeklenebilirliği artırırken Docker Compose kullanımı ortam kurulumunu ve dağıtım sürecini basitleştirmiştir.