Üst Seviye Tasarım Dökümanı

OBSS HRIS PROJESİ

Yazan: Mehmet Tunahan Kanpolat

# Giriş

Bu doküman, OBSS HRIS yazılımının üst seviye tasarımını açıklamak amacıyla hazırlanmıştır. Dokümanda, sistem gereksinimlerini karşılamak için kullanılacak tasarım ve mimari yaklaşımları detaylı olarak açıklanmıştır.

# Sisteme Genel Bakış

OBSS HRIS, IK Uzmanları ve Adaylar (İş Başvurusu Yapan Kişiler) arasında etkileşim sağlayan bir uygulamadır. Sistem aşağıdaki ana bileşenleri içerir:

* LDAP Authentication
* LinkedIn Authentication
* LinkedIn Bilgileri Çekme
* İlan Yönetimi
* İş Başvurusu Yapma
* İş Başvurularını Görüntüleme (Aday)
* İş Başvurularını Görüntüleme (IK Uzmanı)
* Mail Servis
* Kara Listeye Alma
* Free Format Text Bazlı Arama
* Başvuruları Uygunluğa Göre Sıralama

# Kullanıcı Profilleri

IK Uzmanları ve Adaylar olmak üzere iki tip kullanıcı profili desteklenir. İki kullanıcı tipinin de farklı yetkileri vardır.

IK Uzmanlarının sahip olduğu yetkiler:

* Adayları banlama
* İş ilanı oluşturma, güncelleme, silme
* Adaylar üzerinde free format text bazlı arama

Adayların sahip olduğu yetkiler:

* İş başvurusu yapabilme, silebilme
* Aktif ilanları görütüleme
* İş başvurularını görüntüleme

# [High-Level Design](https://softwaredominos.com/home/software-design-development-articles/high-level-solution-design-documents-what-is-it-and-when-do-you-need-one/)

## Mantıksal Tasarım Diyagramları

* Linkedin Authentication

A diagram of a diagram

Description automatically generated

* LDAP Authentication

A diagram of a computer

Description automatically generated

* İş ilanı oluşturma

A diagram of a computer

Description automatically generated

* İş ilanlarını görüntüleme

A diagram of a computer

Description automatically generated with medium confidence

* İş ilanına başvuru

A diagram with text and words

Description automatically generated

* İnsan kaynaklarına ait iş ilanlarını görüntüleme

A diagram of a computer

Description automatically generated

* İnsan kaynaklarına ait iş ilanlarına başvuranları görütüleme

A computer screen shot of a diagram

Description automatically generated

* İş ilanından bağımsız aday arama

A screenshot of a computer

Description automatically generated

* İş başvurusunun statüsü güncelleme

A computer diagram with a mouse

Description automatically generated with medium confidence

* Adayın kendi profilini görüntülemesi

A screenshot of a computer

Description automatically generated

* İnsan kaynaklarının adayın profilini görüntülemesi

A diagram of a data flow

Description automatically generated

* Adayın sistemden çıkış yapması

A screenshot of a computer

Description automatically generated

* Adayın kendi iş başvurularının görüntülenmesi

A close-up of a document

Description automatically generated

* İnsan kaynaklarının adayın iş başvurularını görüntülemesi

A diagram of a diagram

Description automatically generated

* Adayın linkedin profilinin scrape edilmesi

A diagram of a diagram

Description automatically generated

* Adayın banlanması

A diagram of a flowchart

Description automatically generated

## Fiziksel Deployment Diyagramı

**A diagram of a computer

Description automatically generated**

## Çözüm Mimarisi

1. Teknoloji Yığını:

* Sunucu: Java Spring
* İstemci: React
* Veritabanı: PostgreSQL, Open LDAP, Elasticsearch

1. Uygulama Katmanları:

* Controller Katmanı: Kullanıcı arayüzü ve istemci tarafı işlemleri gerçekleştirir.
* Service Katmanı: Controllera gelen istekleri yapılmasını sağlar.
* Repository Katmanı: Veritabanı işlemleri için kullanılır.

1. Veritabanı Tasarımı:

* candidates: id, first\_name, last\_name, linkedin\_id, email, about, is\_banned, profile\_picture
* human\_resources: id, firstname, lastname, username
* job\_posts:id, code, company, location, title, description, hr\_id, activation\_time, closure\_time
* job\_applications: id, candidate\_id, job\_post\_id, application\_time, status, eligibility
* blacklist: id, candidate\_id, reason
* candidate\_skills: candidate\_id, skill
* job\_post\_required\_skills: job\_post\_id, required\_skill

## Yapılan Seçimler ve Alınan Kararlar

* Frontend de React ve Tailwind css kullanmaya karar verdim. React’a biraz hakim olduğum için daha kolay adapte olabileceğimi düşündüm. Tailwind css kullanma sebebim de esneklik sağlaması. Daha önce hazır componentlerin bulunduğu Semantic UI React kullanmıştım, bir componentte değişiklik yapmak istediğim de zorlanıyordum.
* Linkedin Authorization’ı stateless yapabilmek için cookie kullandım. Sebebiyse spring oauth2 clienti stateless yapmaya çalıştığımda authorization request not found hatası alıyordum. Bu durumu araştırdığımda cookie kullanılarak çözüldüğüne ulaştım. Bu sayede de frontendim de callback kullanarak jwt tokeni alabildim.
* Linkedin authentication api üzerinden adayın profil bilgilerine tam olarak ulaşılamadığı için Selenium kullanarak scraping yaptım. Bu kısımda chrome tarayıcısının her seferinde yeniden açılıp login için Captcha’ya takılmamak için bean olarak tanımladım. Bu yapı da aynı anda gelen isteklerde sorun yapabilir, bu yapı deneysel olduğu için görmezden geldim.
* Scraping uygulamasından oluşacak hatalardan dolayı, ana projenin etkilenmemesi için farklı bir projede oluşturdum.
* Mail gönderme servisinde thread kullanarak, statü değişitirme işlemlerinde insan kaynaklarına bildirimin hemen gitmesini sağladım. Burada thread poolu sabit olarak 10 adet tanımladım, uygulamanın kullanıma sunulmayacağını düşünerek bir insiyatif aldım.
* Free text bazlı arama kısmında kullanıcı aramak istediği kelimenin 1 harfini yanlış girdiğinde de sonuç alabilmesi için fuzziness yapısını, birden fazla kelime arattığında sistemde bu kelimeler ard arda olmadığında da sonuç alabilmesi için de slop yapısını kullandım.
* Bütün liste şeklinde dönülen isteklerde sayfalama kullanarak sistemi daha efektif kullanmayı sağladım.