**T.C.**

**SAKARYA ÜNİVERSİTESİ**

**BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ**

BSM 401 BİLGİSAYAR MÜHENDİSLİĞİ TASARIMI

“FROM MY EYES” BİR HOBİ PAYLAŞIM SİTESİ

G171210303 – Furkan KOÇ

|  |  |  |
| --- | --- | --- |
| Bölüm  Danışman | :  : | BİLGİSAYAR MÜHENDİSLİĞİ  Prof. Dr. Ümit KOCABIÇAK |

2020-2021 Güz Dönemi

**İÇİNDEKİLER**

|  |  |
| --- | --- |
| İÇİNDEKİLER…….......................................................................................... | 2 |
| ÖNSÖZ……….................................................................................................. | 3 |
| ÖZET………………………………………..................................................... | 3 |
| 1.GİRİŞ…………….......................................................................................... | 4 |
| 2.KULLANILAN TEKNOLOJİLER…............................................................ | 4 |
| 2.1. Java Spring Boot……………………………………………………… | 4 |
| 2.1.1. Controller………………………………………………………... | 5 |
| 2.1.2. Model……………………………………………………………. | 6 |
| 2.1.2.1. Entity……………………………………………………….. | 6 |
| 2.1.2.2. Dto………………………………………………………….. | 8 |
| 2.1.2.3. Mapper……………………………………………………... | 8 |
| 2.1.3. Repository……………………………………………………….. | 9 |
| 2.1.4. Service…………………………………………………………… | 10 |
| 2.2. Maven………………………………………………………………… | 11 |
| 2.3. Hibernate……………………………………………………………… | 12 |
| 2.4. Lombok……………………………………………………………….. | 12 |
| 2.5. Spring Data JPA……………………………………………………… | 13 |
| 2.6. RESTful………………………………………………………………. | 14 |
| 2.7. JavaScript……………………………………………………………... | 14 |
| 2.8. ReactJS………………………………………………………………... | 14 |
| 3. UYGULAMA……………………………………………………………… | 15 |
| 4. SONUÇ…………………………………………………………………….. | 19 |
| 5. KAYNAKÇA………………………………………………………………. | 20 |
| ÖZGEÇMİŞ | 21 |
| BSM 401 BİLGİSAYAR MÜHENDİSLİĞİ TASARIMI DEĞERLENDİRME VE SÖZLÜ SINAV TUTANAĞI…………………..… | 22 |

**ÖNSÖZ**

Bu projeye başlamadan önce zorlanacağımı biliyordum ama bu kadar zorlu olacağını tahmin edememiştim. Yeni teknolojiler öğrenmek ne kadar zevkli bir şey olsa da bazı zamanlar oldukça fazla can sıkıcı olabiliyor. Araştırmalar yapmak, makaleler okumak, eğitim videoları izlemek ve bunların sonunda öğrendiğimiz şeyleri bir araya getirip ortaya bir proje çıkartmak gerçekten zorlu ve sabır isteyen bir iş. Yeri geliyor 5 dakika da öğrenebildiğimiz şeyler oluyor yeri geliyor saatlerce ufacık bir yerde takılıp kalıyoruz. Ama ortaya çıkan projenin sonucuna baktığım zaman tam olarak istediğim yere gelemesem bile güzel bir iş çıkardığımı düşünmenin mutluluğunu yaşıyorum. Yeni teknolojiler öğrenmenin verdiği mutlulukta cabası. Anlıyorum ki öğrenecek daha çok şey var ve ben daha bu yolun çok başındayım. Bu projeden kazandığım deneyimlerle birlikte istediğim zaman çalışarak, çabalayarak ortaya güzel işler çıkartabileceğimi görmüş oldum. Artık daha kararlı ve istekli şekilde yoluma devam edeceğim.

**ÖZET**

Günümüzde birçok sosyal medya platformu bulunmaktadır. Biz bu platformlara üye oluruz ve kendimizle ilgili yazılar, fotoğraflar, videolar vb. paylaşımlar yaparız. Benim yaptığım “From My Eyes” sitesi de adından da anlaşılabileceği üzere kendi gözümüzden hobilerimizi paylaşabileceğimiz bir web sitesidir.

Hobilerimizin fotoğrafını koyup hobilerimizle ilgili tamamen kendi görüşlerimizi paylaştığımız bir site. Örneğin; Bir kitap okuduk ve bu kitabı arkadaşlarımızla paylaşmak istedik. Bu kitabın konusunu, bize neler kattığını, okurken nelerden etkilendiğini vb. şeyleri anlatabiliriz. Yapmış olduğum sitede hobilerimizi kategorilerine(Kitap, Film, Gezi, Oyun vb.) göre ayırdık bu kategorilerden birini seçerek paylaşım yapabiliyoruz. Başkalarının yaptığı paylaşımları beğenebiliyoruz eğer istersek biz de bu paylaşımlara yorumlar yapabiliyoruz. Arkadaşlarımızın profillerine girerek sadece onların paylaştığı gönderileri görebiliriz ve onların paylaştıkları şeylerden sürekli haberdar olmak istersek onları takip edebiliriz. Profil sayfasından kendimizle ilgili bilgiler verebilir ve profil fotoğrafımızı koyabiliriz bunları değiştirebiliriz.

1. **GİRİŞ**

“From My Eyes” hobilerimizi paylaşabileceğimiz bir internet sitesidir. Bu sitede hem hobilerimizi paylaşabilir hem de paylaşılan hobilere yorum yapabilir ve istersek bu paylaşımları beğenebiliriz.

Projemi Code First metoduyla yaptım, bu web sitesini yaparken database tarafında PostgreSQL kullandım. Backend kısmında ise Java:Spring Boot, Maven, Jpa, Hibernate, Lombok ve RESTful teknolojilerini kullandım. Frontend’i yaparken ise Javascript ve ReactJS teknolojilerini kullandım.

Bu projeyi yapmada belirttiğim teknolojileri seçmemin sebebi bu teknolojilerin birbirleriyle çok verimli ve uyumlu bir şekilde kullanıcı dostu olarak çalışabilmesidir. İçlerindeki fonksiyonlar sayesinde yazmak için çok fazla uğraşmamız gereken fonksiyonları metotlar halinde çağırabiliyor ve kullanabiliyoruz.

Bundan sonraki bölümlerde kullanılan teknolojiler detaylı bir şekilde anlatılacak ve projenin bir uygulaması gösterilecektir.

1. **KULLANILAN TEKNOLOJİLER**
   1. Java: Spring Boot

Resmi proje sayfasındaki tanıma göre Spring Boot bağımsız(stand-alone) Spring tabanlı uygulamalar geliştirmenizi kolaylaştırmaya yarar. Spring Boot‘u bu kadar popüler yapan şey ise otomatik konfigürasyon özelliği. Çok az Spring konfigürasyonu ile çoğu Spring Boot projesini hayata geçirebilirsiniz.

Spring Boot’un bazı temel özellikleri

* Bağımsız(stand-alone) Spring uygulamaları oluşturabilmek.
* Gömülü bir web sunucusu(Tomcat, Jetty, Undertow) ile gelmesi.
* Build konfigürasyonunu kolaylaştırmak için sağladığı starter’lar.
* Otomatik konfigürasyon.
* Kod üretimi(code generation) ve XML konfigürasyona ihtiyaç duymaması.

Yukarıda kısaca Spring Boot‘dan bahsettik. Şimdi şunu söylemeliyim ki Spring Boot da bir proje oluşturmak çok kolay. <https://start.spring.io/> sitesinden istediğimiz konfigürasyonlara göre bir spring projesi oluşturabiliriz. Burada bir Maven projesi yapmak istediğimizi Java dilini kullanabileceğimizi ve istediğimiz bağımlılıkları(dependency) seçebiliyoruz. Bağımlılıklar kısmında kullanacağımız teknolojileri yani PostgreSQL, Lombok, JPA, Hibernate ve RESTful’u seçerek bir projeyi hemen hayata geçirebiliriz.

Spring Boot içerisinde kullandığım katmanlar ise şöyle:

* Controller
* Model
  + Entity
  + Dto
  + Mapper
* Repository
* Service

Şimdi gelelim bu katmanların detaylı anlatımına.

* + 1. **Controller**

Frontend’den gelen ilk istek controllerımızda karşılanır ve buradan service katmanına yönlendirilir.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Fotoğrafta da göründüğü gibi bazı annotationlarımız bulunmaktadır. @RestController bu class’ın bir controller olduğunu belirtiyor. @RequiredArgsConstructor class’ımızın içerisindeki constructorları otomatik olarak oluşturuyor. @CrossOrigin ise erişim izni verilmesini sağlar. Göründüğü gibi @PostMapping ile belirlediğimiz path üzerinden isteklerimiz ilk olarak controller’a düşüyor. Burada önemli olan şey ise gelen istek ile bizim belirlediğimiz istek aynı değişkenlere sahip olmalıdır buna UserDto örneğini verebiliriz ve UserDto’yu yazımın ilerleyen bölümlerinde anlatacağım. Controller içerisinde Service’imizi tanımlayıp sonra bu Service’e istek atarak bize gelen bilgileri Service katmanına iletiyoruz. Burada BaseControllerdan kalıtım alıyoruz ve BaseController içerisinde path adreslerimiz yer alıyor.

* + 1. **Model**

Model katmanının entity kısmında projemizde kullanacağımız veritabanını gerçekliyoruz ve dto kısmında ise frontendden gelecek istekte bulunacak olan verilerin neler olacağını belirtiyoruz. Daha sonrasında Mapperlar ile entity ve dto arasında dönüşüm işlemi yaptırabiliyoruz.

* + - 1. **Entity**

Bahsettiğim gibi veritabanı modelimizi bu katmanda gerçekliyoruz.

Fotoğrafta görüleceği üzere yine bazı annotationlarımız var. @Entity’yi bu class’ın database de tablosunun oluşmasını sağlamak için kullanıyoruz. @Getter ve @Setter ise veritabanında olan kayıtlarımızı getlemek ve setlemek için kullanılır. @Id o değişkenin tablonun ID’si olduğunu gösterir. @GeneretadValue ise ID’nin nasıl üretileceği işlemine yarar burada birer birer arttırmasını istiyoruz. @Size veritabanında ayrılacak karakter uzunluğunu ayarlamamızı sağlar. @OneToOne, @OneToMany, @ManyToOne ve @ManyToMany ise tablolar arasındaki bağlantıların nasıl olduğunu ayarlamamıza yarar.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

* + - 1. **Dto**

Dto katmanında frontendden gelecek olan bilgilerin tutulacağı değişkenleri belirtiyoruz. Aynı zamanda geri dönüş değerini de dto olarak belirtirsek bu değerleri isteğin geldiği yere geri döndürebiliyoruz.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Fotoğrafta göründüğü üzere User için frontendden gelen bilgiler bu şekilde olmalı. @Getter ve @Setter ise service katmanında bunlara setleme ve getleme yapmaya yarar.

* + - 1. **Mapper**

Mapperlarımız ise dto ve entity katmanları arasında dönüşüm yapmamızı sağlar. Entityden bir bilgi getirip dto ya dönüştürüp istek atan yere bu verimizi döndürebiliriz veya dto ile gelen bilgileri entity’ye çevirip veritabanımıza kaydedebiliriz.

**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

* + 1. **Repository**

Repository katmanında database’imizle ilgili işlemleri yaptırıyoruz.

Kalıtım aldığımız JPA Repository sayesinde kullanabildiğimiz hazır metotlar ile fazladan kod yazmaya gerek kalmadan CRUD (Create, Read, Update ve Delete) işlemlerini yaptırabiliyoruz. Eğer daha özel sorgulara ihtiyacımız var ise tabi ki bu sorguları da yazıp işlemlerimizi yaptırabiliriz.

metin, ekran görüntüsü, siyah, kapat içeren bir resim

Açıklama otomatik olarak oluşturuldu

Fotoğrafta da göründüğü gibi kendimize özel sorguları @Query annotation’ını kullanarak yazabiliyoruz. Dönüş tipi(Fotoğrafta User ve Integer olarak belirtmişim) olarakta istediğimize göre ayarlayabiliriz.

* + 1. **Service**

Service katmanı aslında en çok işi yaptığımız kısım diyebilirim. Bütün katmanları harmanlayarak kullandığımız yer Service katmanıdır. Controllerdan gelen isteğe service katmanında istediğimiz işlemleri yapıp sonrasında controller’a geri gönderiyoruz.

@Service annotation’ı bu class’ın bir service olduğunu belirtiyor.

Fotoğrafta gördüğümüz gibi createNewUser metoduna gelen UserDto önce daha önce öyle bir kayıt olup olmadığı kontrol edilip eğer veritabanında aynı username ile kayıt yok ise kaydının yapılması sağlanıyor. Eğer aynı username ile bir kayıt varsa veritabanına kayıt yapılmayıp geriye null değeri döndürülüyor. getUser metodu ise kullanıcı adına göre veritabanından aldığımız kullanıcı bilgisini UserDto’ya dönüştürüp geri dönderiyor.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Spring Boot’un katmanlarını bu şekilde özetleyebiliriz. Şimdi ise gelelim Springte kullandığımız teknolojileri kısaca anlatmaya.

* 1. **Maven**

Maven, POM(Project Object Model)’a dayanan güçlü bir proje yönetim aracıdır. Proje oluşturma, bağımlılık ve dokümantasyon için kullanılır. ANT gibi derleme işlemini basitleştirir. Ancak ANT’den çok daha ileri bir teknolojidir.

Kısacası, Maven için herhangi bir Java tabanlı projeyi oluşturmak ve yönetmek için kullanılabilecek bir araçtır demek çokta yanlış olmayacaktır. Maven, Java geliştiricilerinin günlük işlerini kolaylaştırır ve genellikle herhangi bir Java tabanlı projenin anlaşılmasına yardımcı olur. Peki Maven’ın yardımcı olduğu temel noktalar nelerdi?

* Projeleri kolayca build edebiliriz
* Hiç zorlanmadan JAR, WAR gibi istediğimiz paket formatında projeyi derleyebiliriz
* Projenin bağımlılıklarını kolayca yönetebiliriz
* Proje hakkında meta bilgilerini tutabiliriz
* Projeleri Kaynak Kodu Yönetim Sistemlerine(Git, SVN) kolayca entegre edebiliriz
  1. **Hibernate**

Hibernate bir ORM Firework’udür.

Nesne tabanlı programlarda veritabanımızda kullanacağımız bir yapıdır Hibernate. Hibernate veritabanımızla iletişime geçen programımızdan her türlü işlemi yapmamızı sağlayan bir aracıdır. Neden Hibernate kullanmalıyız sorusuna ise yazım kolaylığı cevabını verebiliriz. Örneğin;

JDBC ile veritabanına kayıt eklerken kullandığımız yapı;

stmt.executeUpdate( "INSERT INTO musteri VALUES ('burak', 'kutbay')");

Hibernate ile kayıt eklerken ile şöyle yazmamız yeterli;

session.save(musteri);

Yazılım karmaşıklığından kurtarmakla birlikte düzgün bir yazılım yapısı kullanmamıza olanak sağlamaktadır.

* 1. **Lombok**

Lombok, Java projesi geliştirirken IDE’ye entegre edilebilen bir anotasyon ile kod üretme (code generation) kütüphanesidir. Lombok ile daha temiz ve daha az kod yazmış oluruz.

Java’da proje geliştirirken yaygın olarak yapmamız gereken bazı işlemler bulunmakta. Bunlar projemizin iş tarafına gerçek bir değer getirmez iken kodumuzun çok fazla ayrıntı barındırmasını zorunlu hale getiriyor.

Örnek olarak Bir POJOda (Plain Old Java Object) Sürekli Get-Set, Constructor, toString vb. methodları yazmamız gerekiyor. Birçok sınıfta olması gereken bu methodlar o kadar uzuyor ki sınıfların uzunluğu ve kod karmaşıklığı artıyor, clean code’dan uzaklaşıyoruz. Lombok sayesinde küçük anotasyonlar ile kodumuzu oldukça kısaltmış oluyoruz.

* 1. **Spring Data JPA**

Spring Data, Spring kütüphanesinin bir parçasıdır. Spring Data JPA bir JPA sağlayıcısı değildir. JPA sağlayıcımızın (Hibernate gibi) üstüne ekstra bir soyutlama katmanı ekleyen bir kütüphane / çerçevedir.

Hibernate bir JPA uygulamasıdır, Spring Data JPA ise bir JPA veri erişimi soyutlamasıdır. Spring Data, GenericDAO uygulamalarına bir çözüm sunar. Spring Data JPA bir uygulama veya JPA sağlayıcısı değildir, veri erişim katmanlarını uygulamak için gerekli olan kaynak kodu miktarını önemli ölçüde azaltmak için kullanılan bir soyutlamadır. Hibernate, loose coupling avantajlarıyla ORM aracı olarak mükemmel bir seçimdir. Unutmayın, Spring Data JPA her zaman Hibernate veya Eclipse Lınk gibi JPA sağlayıcılarını gerektirir.

Spring Data JPA ile CRUD operasyonları için herhangi bir implementasyon yazmanıza gerek yoktur. Örneğin;

repo.findAll();

Bu kod bloğu ile tablodaki tüm kayıtlar listelenecektir. Eğer Hibernate kullanıyorsak, daha sonra repo.findAll () metodu için, aşağıdaki gibi bir implementasyon yazmaları gerekir. Örneğin;

Session session = this.sessionFactory.getCurrentSession();  
List listUsers = session.createQuery("from Customer").list();  
return listUsers;

* 1. **RESTful**

REST, client-server arasındaki haberleşmeyi sağlayan HTTP protokolü üzerinden çalışan bir mimaridir. REST ,servis yönelimli mimari üzerine oluşturulan yazılımlarda kullanılan bir transfer yöntemidir.İstemci ve sunucu arasında XML ve JSON verilerini taşıyarak uygulamanın haberleşmesini sağlar.REST mimarisini kullanan servislere ise RESTful servis denir. Bizim projemizde frontend ile backend arasındaki haberleşmeyi sağlar.

* 1. **JavaScript**

JavaScript’in rakipleriyle karşılaştırıldığında çeşitli avantajları bulunuyor, özellikle de belirli alanlarda. JavaScript avantajlarından bazıları şu şekildedir:

* Bir compiler (derleyici) kullanmanız gerekmez çünkü web tarayıcıları HTML ile yorumlar;
* Diğer yazılım dillerine nazaran öğrenmesi daha kolaydır;
* Hataları bulmak dolayısıyla çözmek daha kolaydır;
* Belirli web sayfası öğelerine veya özel durumlara ayarlanabilir, örneğin fare tıklaması veya imleci üzerine getirme gibi;
* JS birden fazla platformda, tarayıcıda çalışabilir;
* JavaScript kullanarak input değerlendirebilir ve manuel very kontrollerini azaltabilirsiniz;
* Web siteleri daha interaktif yapar ve ziyaretçilerin dikkatini çeker;
* Diğer yazılım dillerinden daha hızlı ve hafiftir.
  1. **ReactJS**

React’ın resmi sitesindeki tanımı:

Interactive UI tasarımlarını acısız bir şekilde yapmanızı sağlar.

Gerçek zamanlı web sitesi geliştirmek için birçok javascript frameworkleri var. Angular, Ember gibi birçok gelişmiş faremwork var. Facebook’un ise bu problemleri cozmek için gelistirdigi sistem React.

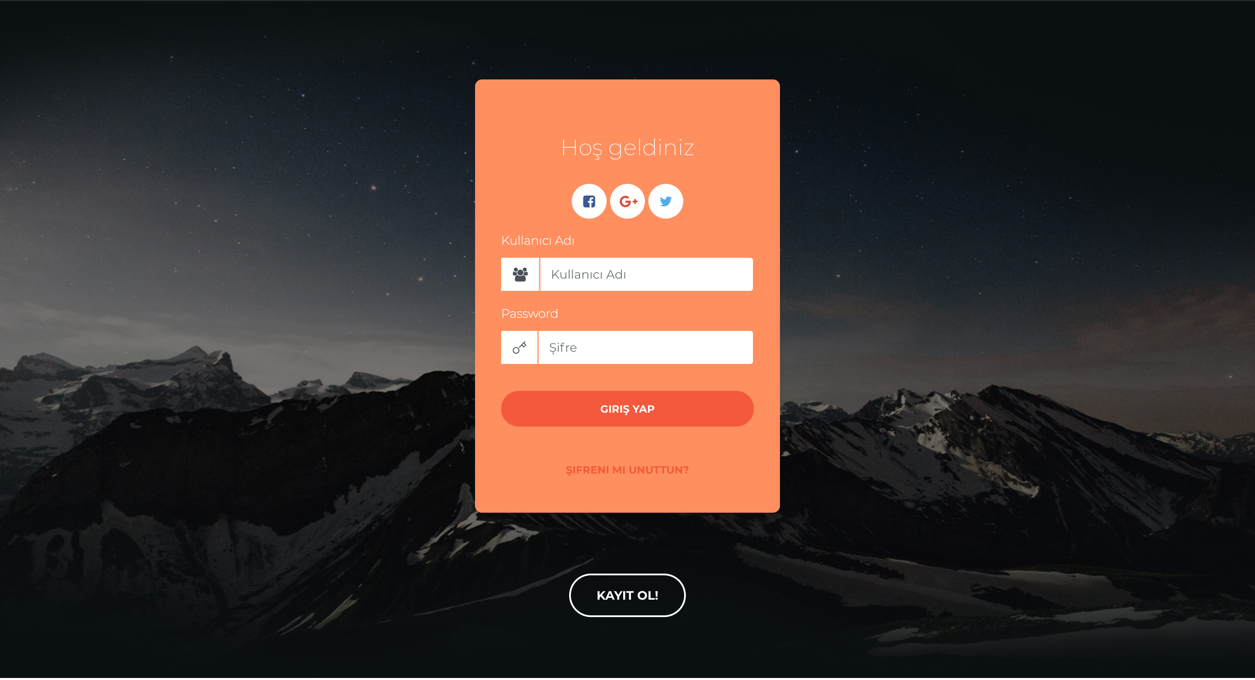
Açık kaynaklı bir JavaScript GUI kütüphanesi olan React’in en büyük odak noktası vardır; UI görevlerini verimli bir şekilde tamamlamaktır. MVC kalıbındaki (Model-View-Controller – Model Görünüm Kontrolcüsü) “V” olarak sınıflandırılabilir.

Java’da yeniden kullanılabilir bileşenleri destekler. React aynı fonksiyonu kullanan başka uygulamalara geliştirilen bileşenleri yeniden kullanmanıza izin verir. Bileşen yeniden kullanabilme özelliği geliştiriciler için sıradışı bir özelliktir.

React bileşeni, HTML ile JavaScript’i birleştirmenizi sağlayan tercihe bağlı bir sözdizimi uzantısı olan JSX kullandığından yazması daha kolaydır. JSX JavaScript ile HTML‘in mükemmel bir karışımıdır. Bütün web site yapı yazma sürecini açıklar. Ek olarak, kullanılan uzantı birden çok fonksiyonu işlemeyi daha kolay hale getirir. JSX en popüler sözdizimi uzantısı olmasa da özel bileşenler veya yüksek hacimli uygulamalar geliştirmekte oldukça verimli olduğunu kanıtlamıştır.

1. **UYGULAMA**

Web sitemizde kullanıcı ve şifre ile giriş yapabildiğimiz bir kullanıcı girişi bulunmaktadır. Bu sayfada bulunan Kayıt Ol butonu ile kayıt olma sayfasına gidebiliriz.

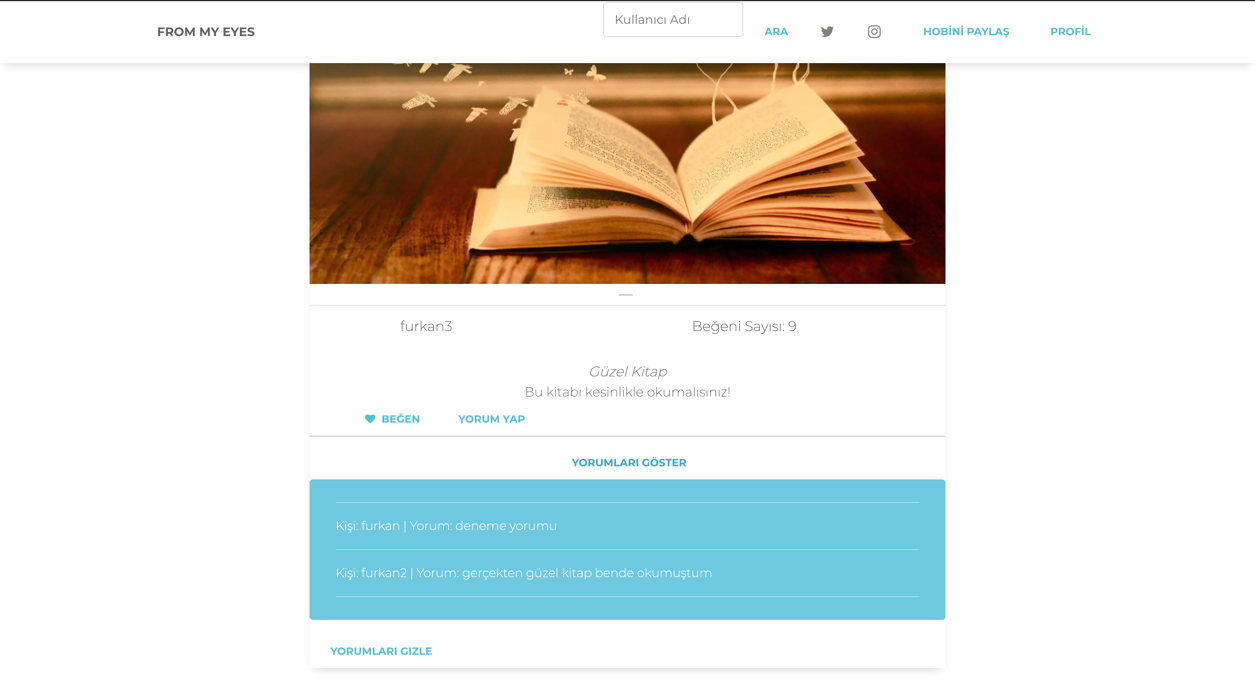
****

Kayıt Ol sayfasında kullanıcı adı ve şifre bilgileri girilerek kayıt olabiliriz.

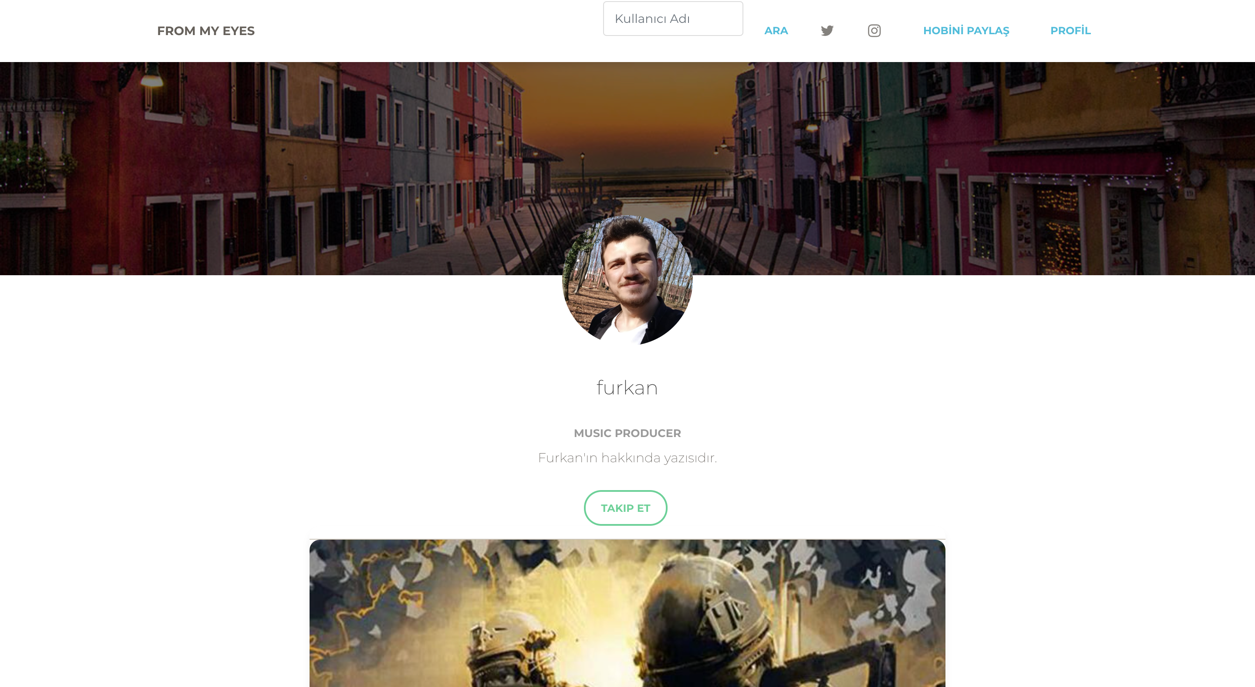
metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

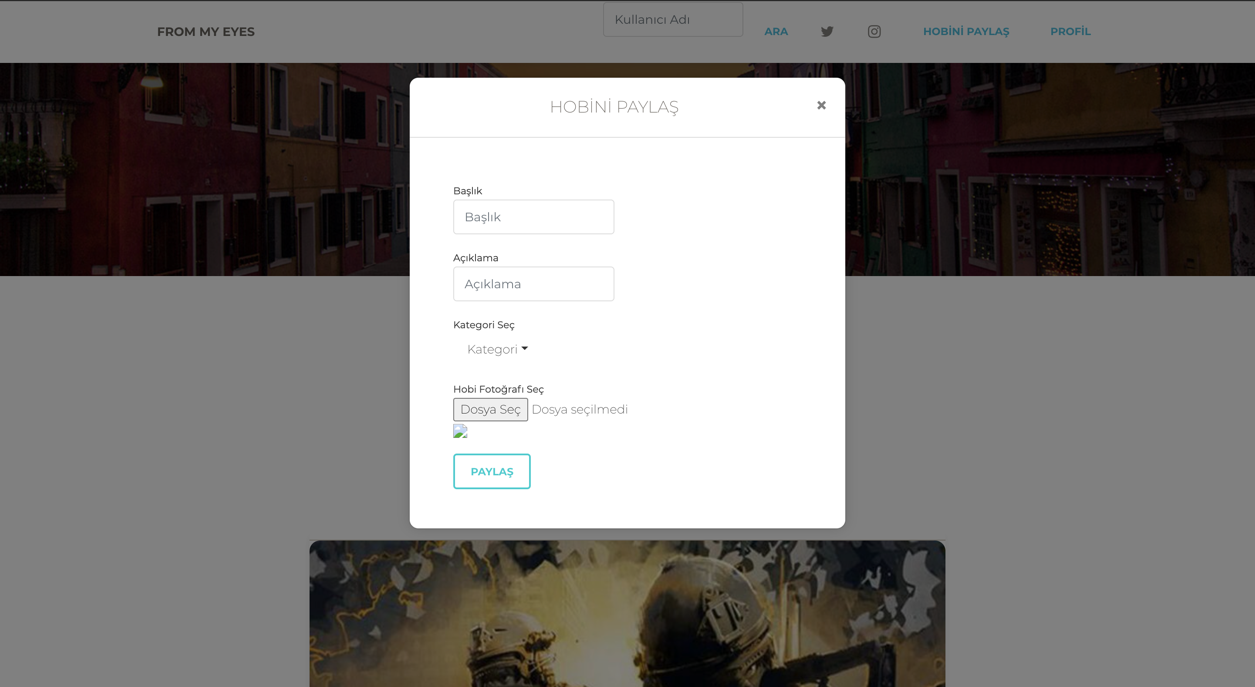
Giriş yaptıktan sonra karşımıza ana sayfa geliyor. Ana sayfada paylaşılan hobileri görebilir, istersek onlara yorum yapabilir ve beğenebiliriz.



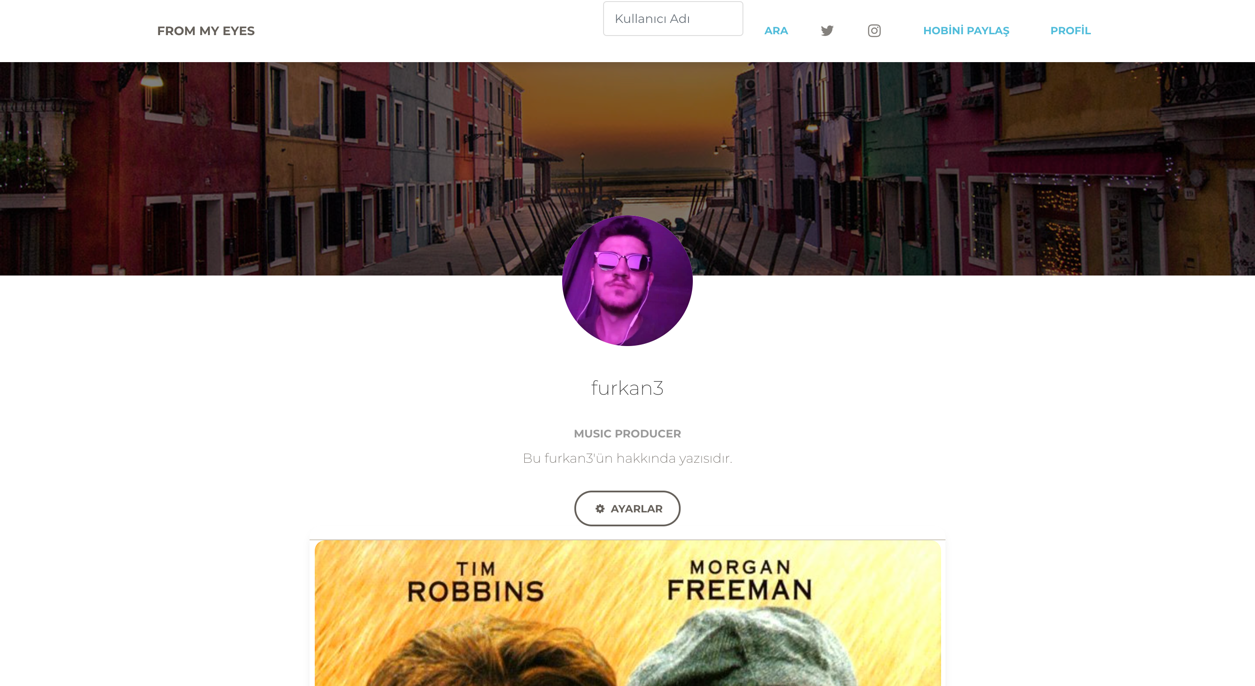
Menüde bulunan arama alanı ile başka kullanıcıları arayabilir ve onların profil sayfalarını görebiliriz. Profilini gördüğümüz kişiyi takip edebilir, onların paylaştıklarını görebilir, beğenebilir ve yorum yapabiliriz.



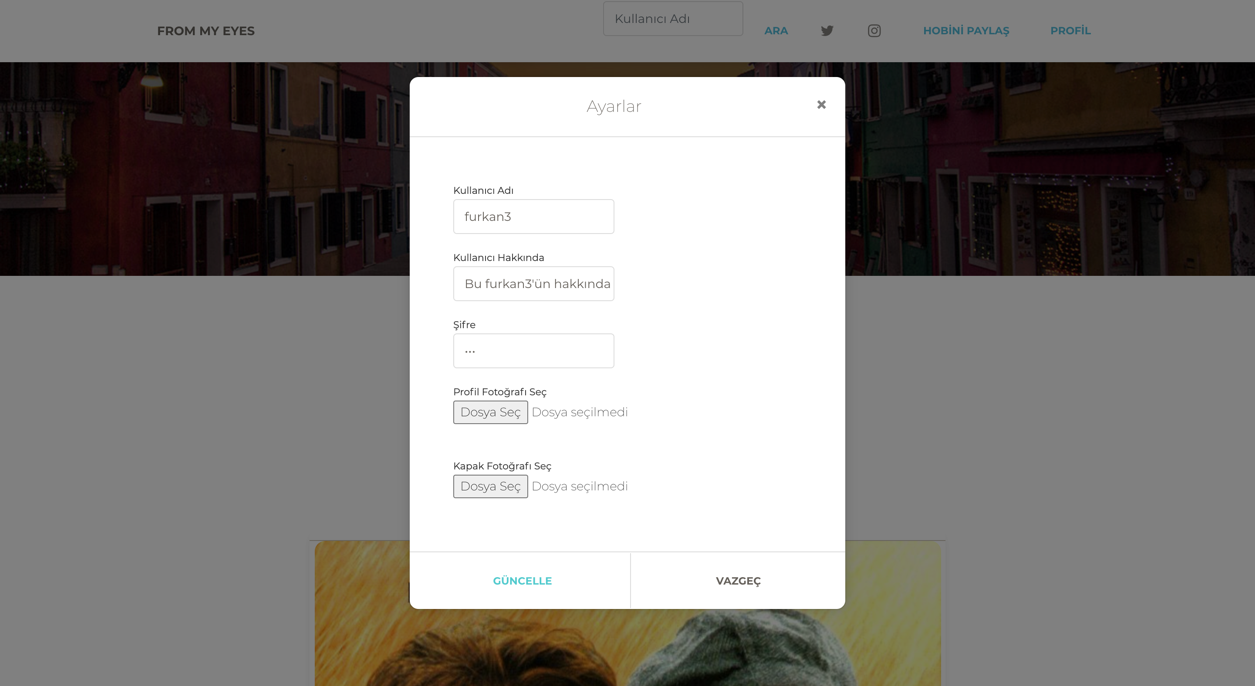
Hobini paylaş butonu ile hobilerimizi paylaşabiliriz.



Profil butonuna basarak kendi profil sayfamıza gidebiliriz. Kendi paylaşımlarımızı görebiliriz.



Profilimizde bulunan Ayarlar butonu ile bilgilerimizi güncelleyebiliriz.



1. **SONUÇ**

Sonuç olarak From My Eyes sitesinde hobilerimizi paylaşabiliriz ve paylaşılan hobileri beğenebilir, yorum yapabiliriz. Başka insanların profillerine gidip onların hobilerini görebiliriz.

Kullanmış olduğum teknolojiler Java:Spring Boot, Maven, Hibernate, JPA, RESTful, Lombok birbirleriyle çok uyumlu çalışan bir backend yapısı oluşturuyor ve backendi kolayca yazmamızı sağlıyor. Frontend kısmında kullanmış olduğum ReactJS ise dinamik bir web sitesini yapmayı kolaylaştıran frameworklerinden birisi olduğunu kanıtlıyor.

Ben projemde kullanmış olduğum teknolojileri uzun bir araştırma sürecinden sonra tam anlamıyla öğrenebildim. Bu raporda anlattığım şekilde basit bir yapı kurup siz de bu teknolojiler ile bir proje ortaya çıkartabilirsiniz.

Hedeflediğim yere tam olarak ulaşamasam da ana hatlarıyla bu projeyi bitirebildiğimi düşünüyorum. Eksik olan kısımları ise ilerleyen süreçlerde tamamlanabilir.

1. **KAYNAKÇA**

* <https://medium.com/yazilim-vip/bu-yaz%C4%B1n%C4%B1n-amac%C4%B1-maven-ile-siz-okurlar%C4%B1-tan%C4%B1%C5%9Ft%C4%B1rmakt%C4%B1r-aab0f6ff91f4>
* <https://medium.com/@furkanbegen/spring-boot-nedir-4cc3f41eb7de>
* <https://blog.burakkutbay.com/hibernate-nedir-hibernate-dersleri.html/>
* <https://medium.com/kodgemisi/project-lombok-6d2490df8adf#:~:text=Lombok%2C%20Java%20projesi%20geli%C5%9Ftirirken%20IDE,%C3%BCretme%20(code%20generation)%20k%C3%BCt%C3%BCphanesidir.&text=%C3%96rnek%20olarak%20Bir%20POJOda%20(Plain,toString%20vb.%20methodlar%C4%B1%20yazmam%C4%B1z%20gerekiyor>.
* <https://medium.com/@sangaibisi/hibernate-ve-spring-data-jpa-aras%C4%B1ndaki-fark-nedir-aab8274c41e4>
* <https://medium.com/@bsrutmn/rest-ve-restful-web-servi%CC%87s-nedi%CC%87r-7258b7db7f66>
* <https://www.hostinger.web.tr/rehberler/javascript-nedir-yeni-baslayanlar-icin-temel-javascript-rehberi/>
* https://www.hostinger.web.tr/rehberler/react-nedir/

**ÖZGEÇMİŞ**

Furkan KOÇ 01.03.1997 yılında istanbul’da doğdu. Lise hayatından beri bilgisayar üzerine okumakta ve çalışmaktadır. Lisede Handan Hayrettin Yelkikanat Anadolu Teknik Lisesinde bilgisayar bölümünde okumuştur. Daha sonrasında Marmara Üniversitesi Bilgisayar Programcılığı bölümünde okumuş ve mezun olmuştur. Mezun olduktan sonra bir süre Bilgi İşlem Sorumlusu olarak çalışan Furkan bu süreçte DGS sınavına hazırlandı ve sınav sonucunda Sakarya Üniversitesi Bilgisayar Mühendisliği bölümünü kazandı. Burada okurken Bileşim A.Ş. firmasında staj yapmış ve şuan 4. Sınıfta öğrenciliği devam etmektedir. 4. Sınıfın başında Sakarya Teknokentte faaliyetlerine devam eden CMV Teknoloji firmasında çalışmaya başlamıştır ve hala çalışmaya devam etmektedir..

**BSM 401 BİLGİSAYAR MÜHENDİSLİĞİ TASARIMI**

**Değerlendİrme ve Sözlü Sınav Tutanağı**

KONU :From My Eyes Bir Hobi Paylaşım Sitesi

ÖĞRENCİLER (G171210303/Furkan/KOÇ):

|  |  |  |  |
| --- | --- | --- | --- |
| Değerlendirme Konusu | İstenenler | Not Aralığı | Not |
| **Yazılı Çalışma** |  |  |  |
| **Çalışma klavuza uygun olarak hazırlanmış mı?** | x | 0-5 |  |
| **Teknik Yönden** |  |  |  |
| **Problemin tanımı yapılmış mı?** | x | 0-5 |  |
| Geliştirilecek yazılımın/donanımın mimarisini içeren blok şeması (yazılımlar için veri akış şeması (dfd) da olabilir) çizilerek açıklanmış mı? |  |  |  |
| Blok şemadaki birimler arasındaki bilgi akışına ait model/gösterim var mı? |  |  |  |
| Yazılımın gereksinim listesi oluşturulmuş mu? |  |  |  |
| Kullanılan/kullanılması düşünülen araçlar/teknolojiler anlatılmış mı? |  |  |  |
| Donanımların programlanması/konfigürasyonu için yazılım gereksinimleri belirtilmiş mi? |  |  |  |
| UML ile modelleme yapılmış mı? |  |  |  |
| Veritabanları kullanılmış ise kavramsal model çıkarılmış mı? (Varlık ilişki modeli, noSQL kavramsal modelleri v.b.) |  |  |  |
| Projeye yönelik iş-zaman çizelgesi çıkarılarak maliyet analizi yapılmış mı? |  |  |  |
| Donanım bileşenlerinin maliyet analizi (prototip-adetli seri üretim vb.) çıkarılmış mı? |  |  |  |
| Donanım için gerekli enerji analizi (minimum-uyku-aktif-maksimum) yapılmış mı? |  |  |  |
| Grup çalışmalarında grup üyelerinin görev tanımları verilmiş mi (iş-zaman çizelgesinde belirtilebilir)? |  |  |  |
| Sürüm denetim sistemi (Version Control System; Git, Subversion v.s.) kullanılmış mı? |  |  |  |
| Sistemin genel testi için uygulanan metotlar ve iyileştirme süreçlerinin dökümü verilmiş mi? |  |  |  |
| Yazılımın sızma testi yapılmış mı? |  |  |  |
| Performans testi yapılmış mı? |  |  |  |
| Tasarımın uygulamasında ortaya çıkan uyumsuzluklar ve aksaklıklar belirtilerek çözüm yöntemleri tartışılmış mı? |  |  |  |
| **Yapılan işlerin zorluk derecesi?** | x | 0-25 |  |
| **Sözlü Sınav** |  |  |  |
| **Yapılan sunum başarılı mı?** | x | 0-5 |  |
| **Soruları yanıtlama yetkinliği?** | x | 0-20 |  |
| **Devam Durumu** |  |  |  |
| **Öğrenci dönem içerisindeki raporlarını düzenli olarak hazırladı mı?** | x | 0-5 |  |
| **Diğer Maddeler** |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| **Toplam** |  |  |  |

Danışman :Prof. dr. ümit kocabıçak

danışman imzası: