Bilişim Sistemleri Mühendisliği Fabrika Otomasyon Projesi

1Yunus Emre Gölbaşı, 2Serhat Akçadağ, 3Murat Öztürk

1yunusemregolbasi0@gmail.com, 2serhat.akcadag@gmail.com, 3murat5506furkan@gmail.com

Özet: PHP, Laravel, MySQL, React.js, TailwindCSS kullanılarak yazılmış bir fabrika otomasyonu. Fabrika otomasyonları, endüstriyel üretimi yönetmek, verimliliğini artırmak, kontrol etmek ve işletme maliyetlerini düşürmek için kullanılan bir yaklaşımdır. Bu çalışmanın amacı, bir fabrikada kullanılan otomasyonun minimalist bir taslağını oluşturmaktır.

Keywords-PHP, Laravel, React.js, TailwindCSS, MySQL, Fabrika, Otomasyon

1.Giriş

Proje fabrika içindeki üretim, personel yönetimi ve raporlamaya modernist bir bakış açısı sunar. Proje içersinde fabrika içindeki üretim hatları mevcuttur. Bu üretim hatlarında üretilen ürünlerin yanı sıra ayrıca üretim hatlarına bağlı makineler mevcuttur. Fabrikada bulunan personellerin yönetimine gelidiğinde ise personellerin mevcut tüm bilgileri yöneticiye sunulur. Ayrıca yönetici personellerin iş emirlerini control edebilir ve raporlayabilir.

Raporlama kısmına gelindiğinde ise günlük üretim raporlama bir üretim hattında günlük üretime bağlı, günlük aylık haftalık ve yıllık miktar grafik şeklinde raporlanır. Personel raporlamaysa personelin tamamladığı iş emirlerinin miktarına göre yüksek performanslı, normal, düşük performansı olarak raporlanır.

**2.Problem Tanımı**

Bir fabrika içerisinde çalışanların dijtal ortamda tutulması, üretim emirleri verilmesi, bir üretim hattında üretilen ürünlerin takibi, makinelerin takibi ve bu personellerin yaptığı işlerin takibi başlıca sorunlardandır. Geliştirilen projede bu sorunların tümü web ortamında cevaplanmaya çalışılmıştır.

**3.Yapılan araştırmalar**

Projede herkesin ilgili teknolojiler ile daha önce tecrübesi bulunduğundan araştırma süreci bu proje özelind eçok kapsamlı olmamıştır. Yine de zorlandığımız birtakım konular aşağıdaki maddelerde bulunur.

3.1 Backend

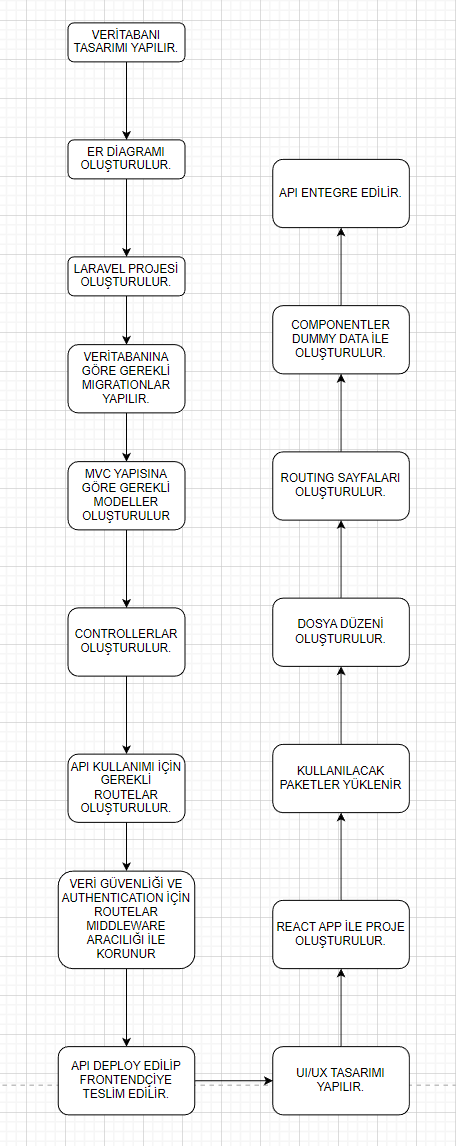
Migration işlemlerinde constraint verilirken Laravel’in varsayılan isimlendirilmesine uyulmaması yüzünden verdiğim foreign ve primary keylerden dolayı on delete için constraintlerimde hata olmuştur. Ayrıca bir model nesnesi geri döndürülürken modele ekstradan bir alan verilmesi ve Laravel’in faker kütüphanesinde eşsiz bir şekilde date üretilememesi karşılaştığım başlıca sorunlardır. Bu sorunların birçoğunun cevabı Laravel’in original dokümantasyonunda cevabı bulunup, eşsiz bir şekilde date döndüren fonksiyonu DateTime.php adlı dosyaya kendi elimle manuel bir şekilde ekelemek zorunda kaldım.

3.2 Frontend

Routing işlemleri için kullandığımız React Routerin güncellenmesi sebebiyle yönlendirme işlemi redirect doğru şekilde çalışmadı. Bunun yerine React Router’in useNavigate hook’u kullanıldı.

Ayrıca Laravel ile geliştirilen API’yi entegre ederken localde kurulu PHP sürümünün Laravel’in yazıldığı PHP sürümüne güncellenmesini gerçekleştirdik, API’ya frontend’den istek gönderirken headers bölümünde ‘content-type’ ın body’de gönderilen datanın türünü API’ye bildirmesi gerektiğini HTTP request protokülünden öğrendik.

**4.Akış Şeması**



**5.Yazılım mimarisi**

5.1 Frontend

React.js tarafında daha önce de söylediğimiz gibi stil işlemleri için TailwindCSS ve Material UI kullandık. Tailwind config dosyasında proje klasörü içindeki bütün .jsx uzantılı dosyaları tarayıp anında projenin geçerli stil dosyasına işleyen Tailwind clı ile geliştirme ortamında rahat ettiğimizi söyleyebilirim. Birkaç hazır stil işlemleri için bazı Material UI componentleri kullandık örneğin; modal (ekrana pop-up olarak çıkan alert), snackbar (sayfanın sol altında anlık info vermek için), navbar, tabpanel (sayfanın sol kısmındaki sekme görünümü) ve tabel componentlerini kullandık.

Projemizi single page application olarak geliştirdik. Bu projemizin ön yüzünde daha akıcı ve hızlı bir deneyim sunmamızı sağladı. Grafik için “apexChart” kullandık.

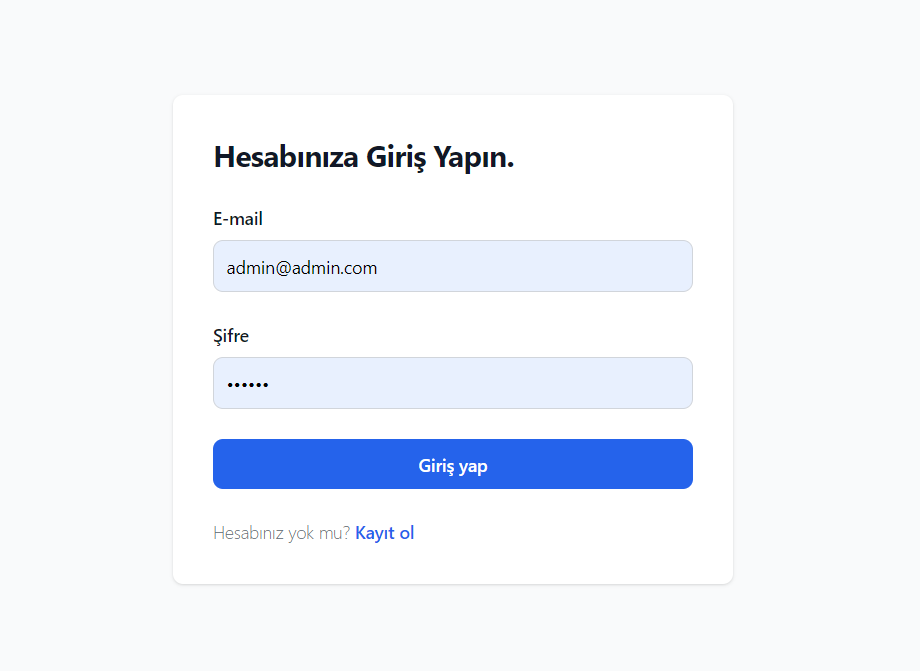
API entegrasyonu hakkındaysa API url’ini .env dosyasında tuttuk ve global state management ( redux veya context) kullanmadığımız için her componentte .env dosyasından çektik. Kullanıcı giriş yaptıktan sonra gelen tokeni propslar yardımıyla alt componentlere taşıdık ve aynı zamanda kullanıcı tarayıcıyı kapatıp tekrar açtığında tekrar login olmasına gerek kalmaması için tokeni tarayıcının localStorage’inde tuttuk. API isteklerini yönetmek için ayrıca bir dosya oluşturmadan datanın kullanılacağı component’te isteği gönderdik.

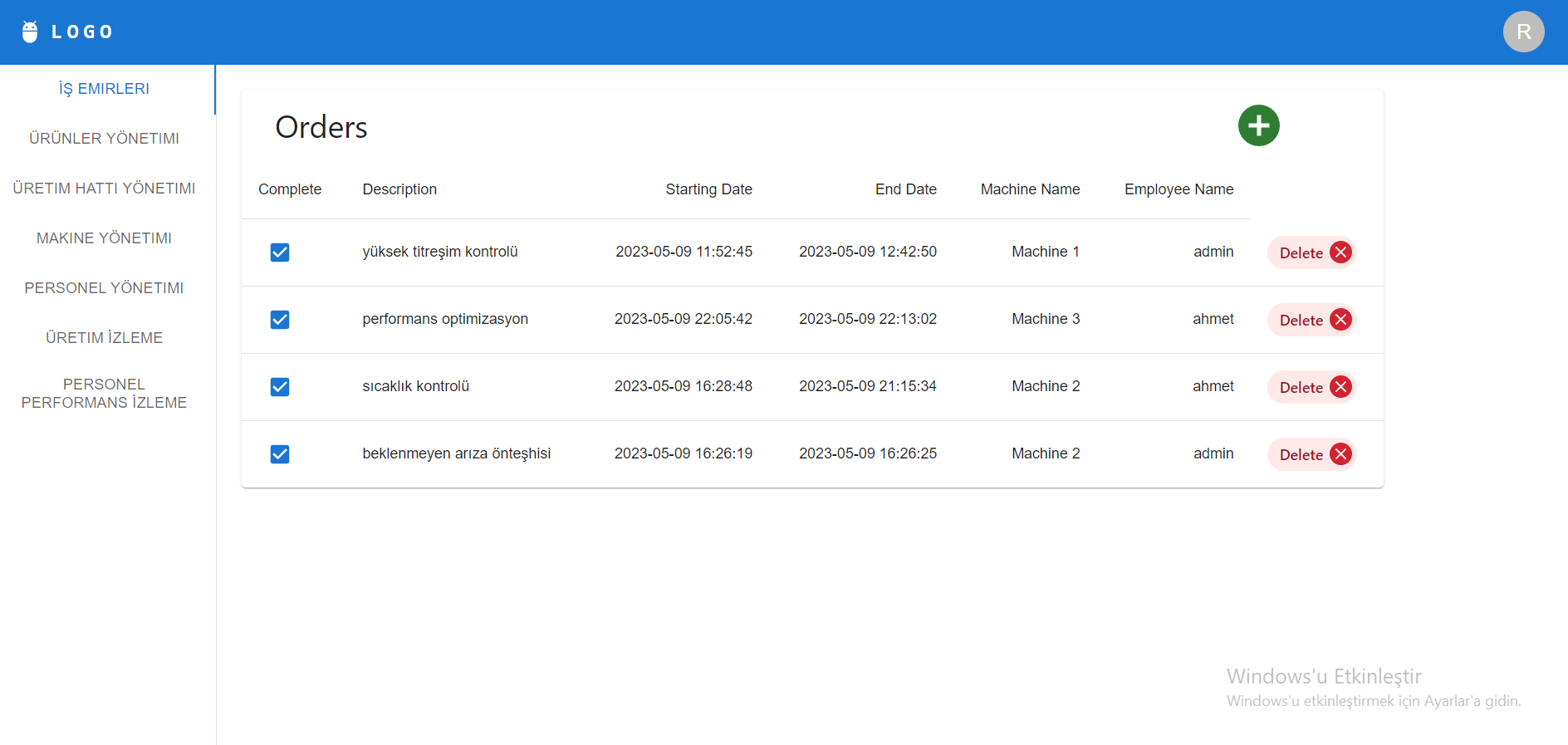
5.2 Backend

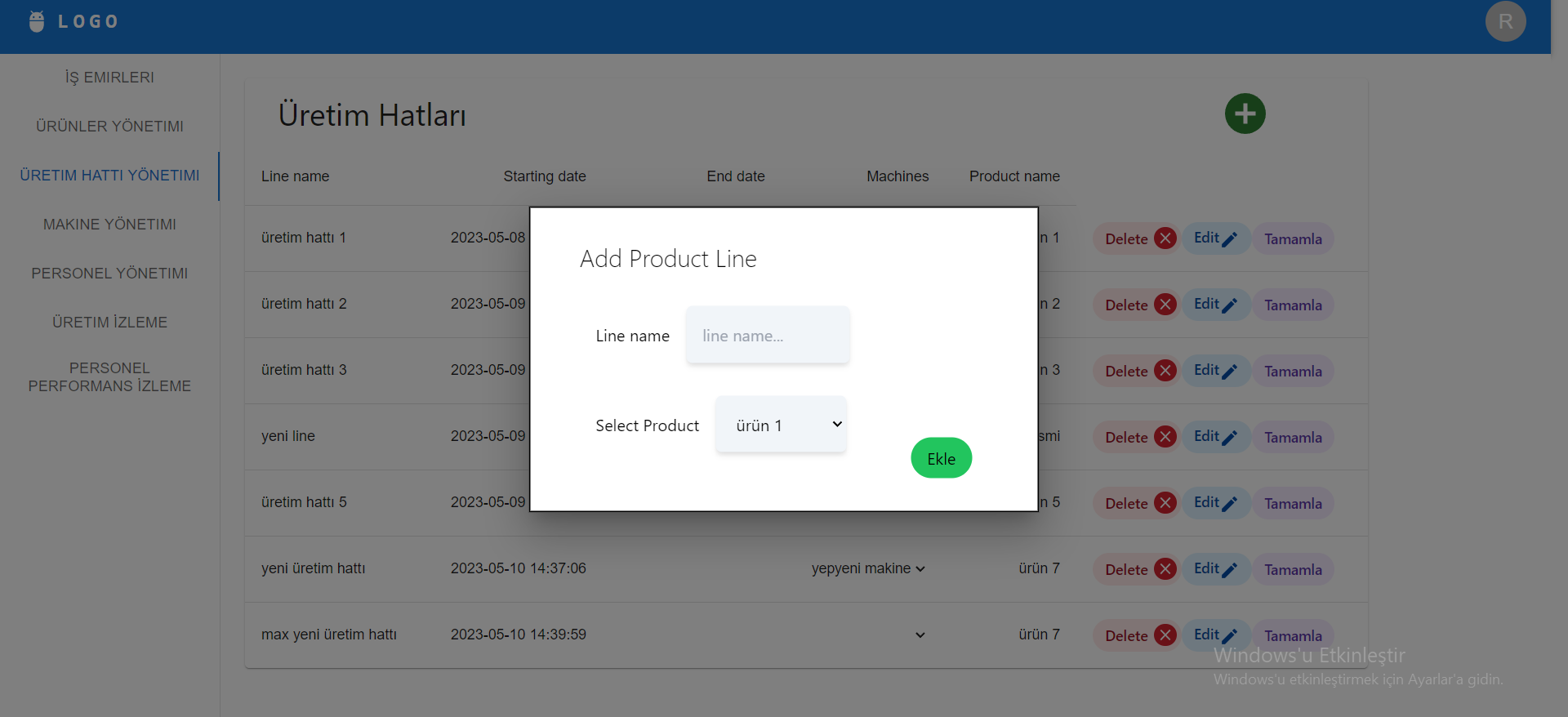
Backend yazılırken MVC yapısı kullanılmıştır. Frontendde SPA kullanacağımız için bir API geliştirilmesi uygun görülmüştür. Yazılan API'da veritabanındaki her bir tabloya eşdeğer bir Model bulunmaktadır. Bu veritabanındaki her bir tablonun CRUD işlemleri ilgili Controllerlara yazılmıştır. Bunun yanı sıra bazı özelleştirilmiş fonksiyonlar yazılmıştır. Örneğin günlük üretim raporunun günlük, haftalık ve yıllık bazda döndürülmesi gibi. Oluşturulan controllerlar ve methodları Routers klasörü altındaki api.php dosyasında kullanılmıştır. Çalışanlara admin rolü verilmiş ve adminin login, register ve logout fonksiyonları yazılmıştır. Bu fonksiyonlardan register ve login fonksiyonları geriye birer token döndürür. Bu tokenların geçerlilik süresi 3 saattir. Her 24 saatte bir ise veritabanından geçerlilik süresi dolmuş tokenlar tamamen silinir. Bu döndürülen tokenlar ve kullanıcıların şifreleri ise güvenlik açısından veritabanında hashlenmiş bir şekilde tutulur. Yazılan ilgili routelar Sanctum Authentication aracılığı ile korunur ve çalışabilmesi için API'a istek atılırken Headers kısmında Authentication headerına berarer token olarak verilmesi gerekmektedir. İlgili API localde çalışır ve frontendciye API ile çalışabilmesi için teslim edilir.

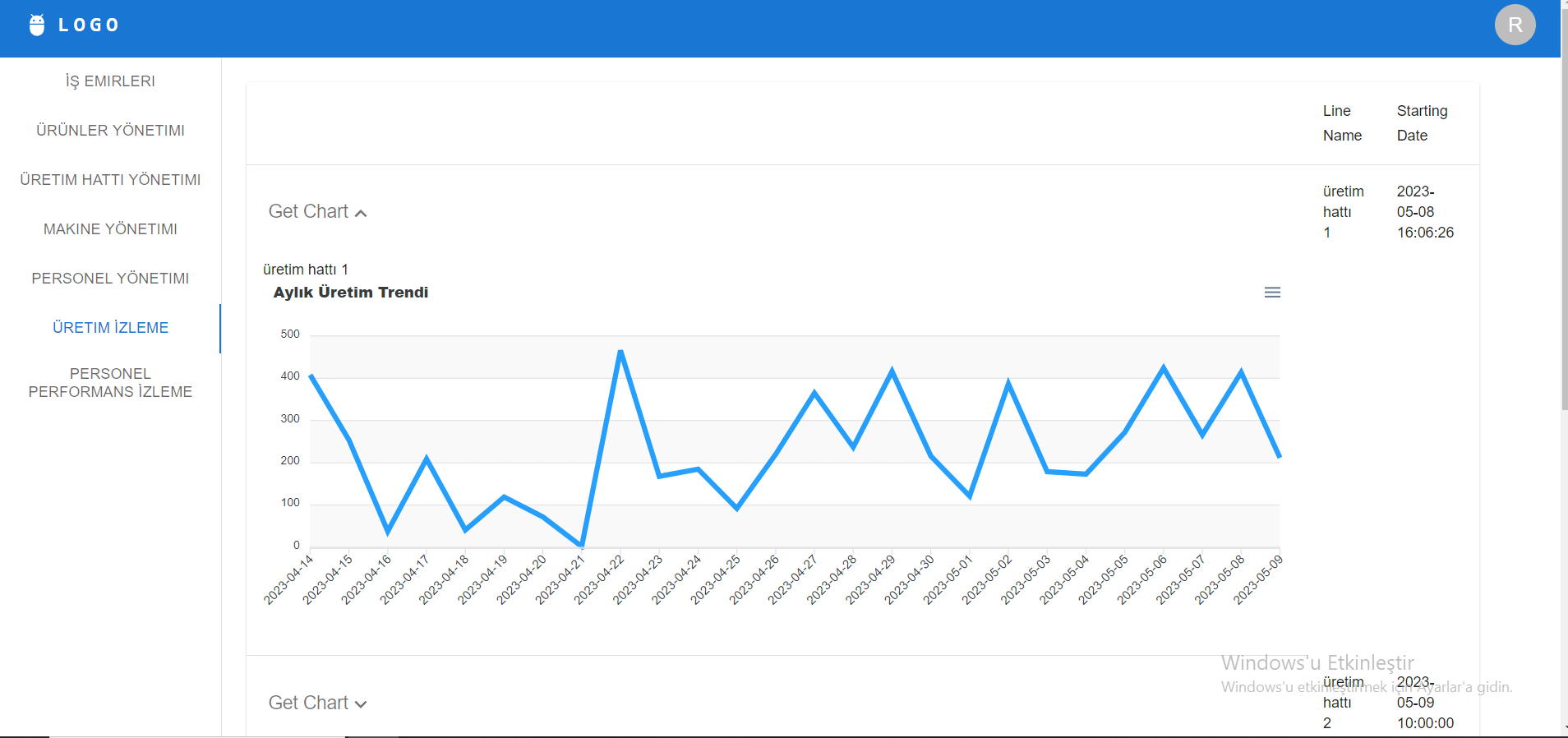
**6. Genel Yapısı**

Otomasyon projemizde genel olarak Laravel API’den gelen .json dataları kullanıcı arayüzünde React.js ile etkileşimli hale getirdik. Projemizde kullanıcı oturum açma işlemleri, tüm CRUD işlemleri, kullanıcı yetkilendirme, personel izleme, grafikle üretim raporları özellikleri bulunmaktadır. Yönetici yeni bir ürün, makine, personel, üretim hattı ekleyebilir. Bunları silip, düzenleyebilir. Çalışanlarına iş emri verebilir. Makinelerin son bakım tarihini görebilir. Projemizden bazı ekran görüntüleri:









References:

<https://laravel.com/docs/9.x>

<https://github.com/fzaninotto/Faker/issues/1878>

<https://stackoverflow.com/questions/47741526/laravel-api-token-authentication-not-working>

<https://stackoverflow.com/questions/41014735/how-to-change-the-user-model-in-laravel-to-use-other-table-for-authentication-th>

<https://stackoverflow.com/questions/65372739/laravel-sanctum-how-to-configure-the-expiration-date-of-the-token>

<https://react.dev/blog/2023/03/16/introducing-react-dev>

<https://reactrouter.com/en/main>

<https://tailwindcss.com/>

<https://mui.com/material-ui>

<https://apexcharts.com/>

<https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API>