**2024-2025 YILI BAHAR DÖNEMİ**

**BULUT BİLİŞİMİ ARASINAV RAPORU**

****

**AD – SOYAD ÖĞRENCİ NUMARASI**

**HÜSEYİN TINAZTEPE: 21290360**

**YUSUF TUNÇ: 22290071**

**BORA KOCABIYIK: 21290270**

PROJE 1 GitHub Linki: **Çift Katmanlı Web Uygulaması (Web API + Frontend)**

<https://github.com/brckfrc/skynotes.git>

PROJE 2 GitHub Linki: **Akıllı Veri Analitiği ve Makine Öğrenmesi Uygulaması**

https://github.com/hsyntinaztepe/mlproject

**PROJE 1: Çift Katmanlı Web Uygulaması (Web API + Frontend)**

**SkyNotes Tam Yığın Not Yönetim Uygulaması Proje Raporu**

**1. Proje Genel Bakış**

**1.1 Proje Amacı**

SkyNotes, kullanıcıların dijital not yönetimini kolaylaştırmak için geliştirilmiş bir web uygulamasıdır. Proje, modern web geliştirme teknolojilerini kullanarak kullanıcı dostu bir not yönetim sistemi oluşturmayı hedeflemektedir.

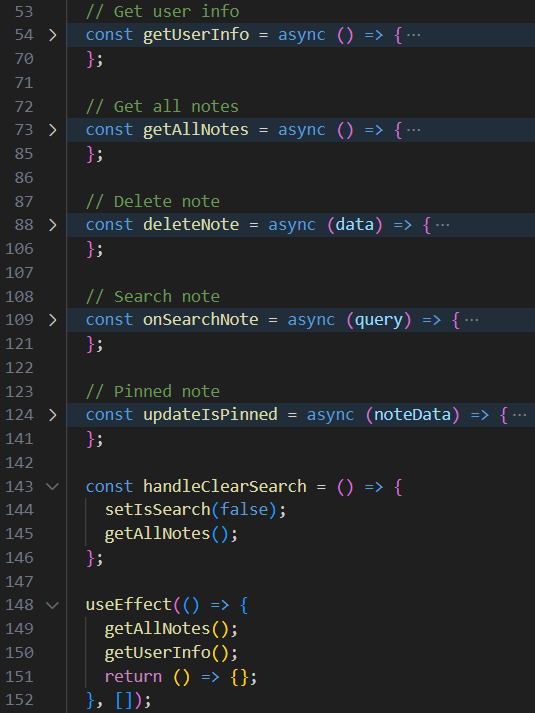
**2. Teknoloji Yığını**

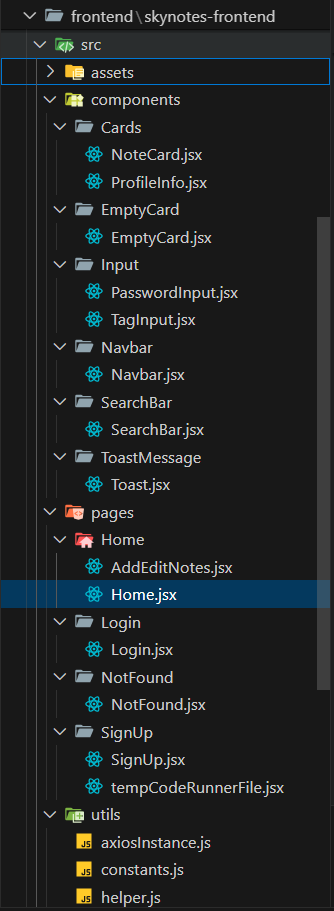
**2.1 Frontend Teknolojileri**

SkyNotes'un frontend geliştirme süreci, kullanıcı deneyimini merkeze alan bir yaklaşımla ilerletildi. Proje başlangıcında Vite ile hızlı ve modern bir React projesi kurularak temeller atıldı. İlk olarak, uygulamanın genel mimari yapısı ve bileşen hiyerarşisi planlandı. React Router DOM kullanılarak sayfa navigasyonu için sağlam bir altyapı oluşturuldu.

Tailwind CSS ile stil oluşturma süreci, hızlı ve responsive bir tasarım sağlamak üzere yapılandırıldı. Her bileşen, kullanıcı deneyimini optimize edecek şekilde detaylı olarak tasarlandı. Geliştirme sürecinde sürekli olarak bileşenlerin yeniden kullanılabilirliği ve kod tekrarından kaçınma prensipleri gözetildi.

* React: Kullanıcı arayüzü geliştirmek için modern JavaScript kütüphanesi
* React Router DOM: Sayfa yönlendirme ve navigasyon için
* Tailwind CSS: Hızlı ve esnek stil oluşturma için utility-first CSS framework'ü
* React Icons: Kullanıcı arayüzü için zengin icon seti



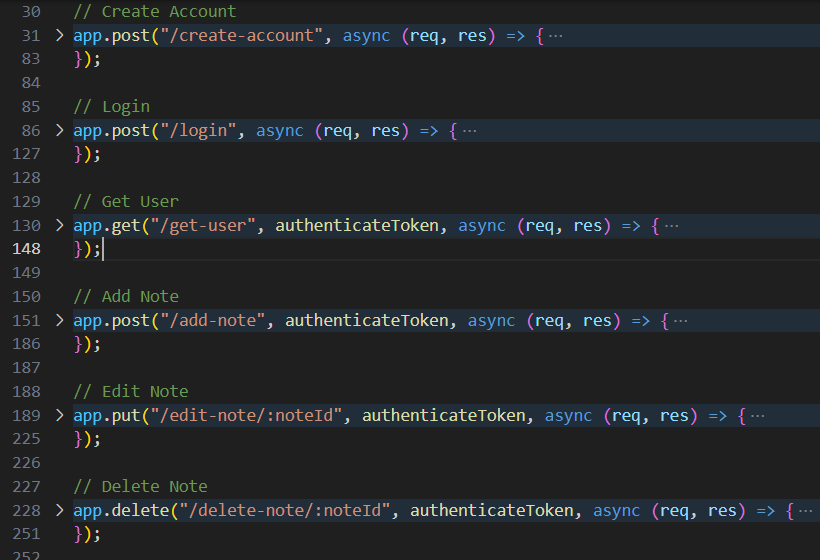
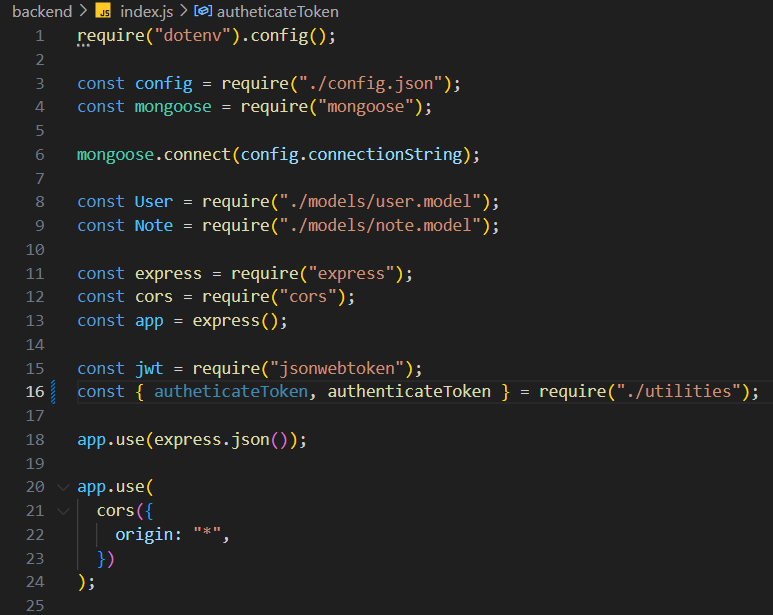


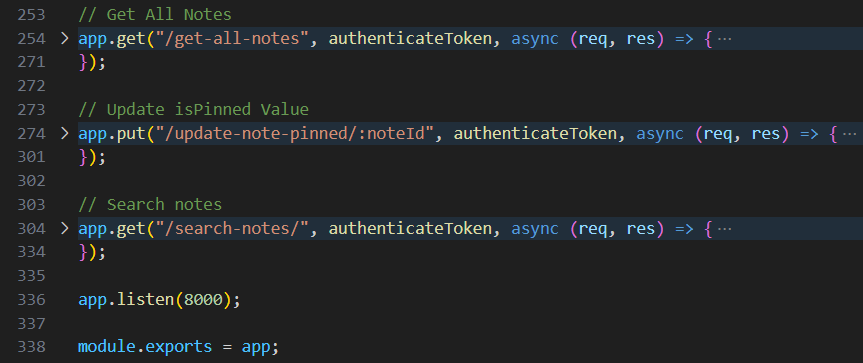
**2.2 Backend Teknolojileri**

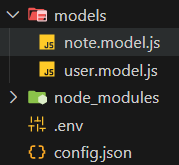
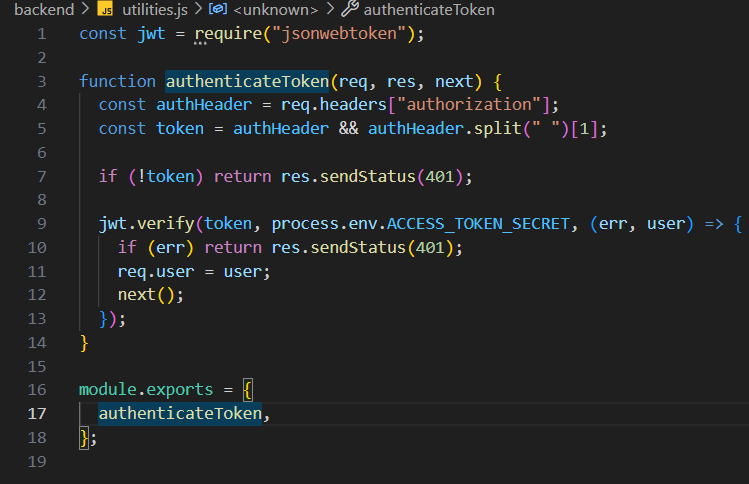
* Node.js: Sunucu tarafı JavaScript çalışma ortamı
* Express.js: Web uygulaması ve API geliştirme framework'ü
* Mongoose: MongoDB ile etkileşim için Object Data Modeling (ODM) kütüphanesi
* JSON Web Token (JWT): Güvenli kullanıcı kimlik doğrulama mekanizması
* dotenv: Ortam değişkenlerini yönetme
* cors: Cross-Origin Resource Sharing yönetimi

Backend geliştirme süreci, güvenli ve ölçeklenebilir bir API mimarisi oluşturmaya odaklandı. Node.js ve Express.js kombinasyonu, hızlı ve esnek bir sunucu tarafı çözümü sundu. Mongoose ORM kullanılarak MongoDB ile olan veri etkileşimi optimize edildi ve veri modellemesi için güçlü bir altyapı kuruldu.

Kullanıcı ve not modellerinin tasarımında, veri bütünlüğü ve güvenliği ön planda tutuldu. JWT (JSON Web Token) ile kimlik doğrulama mekanizması uygulandı, bu sayede kullanıcı oturumları güvenli bir şekilde yönetildi. Her API endpoint'i, hem güvenlik hem de performans açısından detaylı olarak test edildi.

****

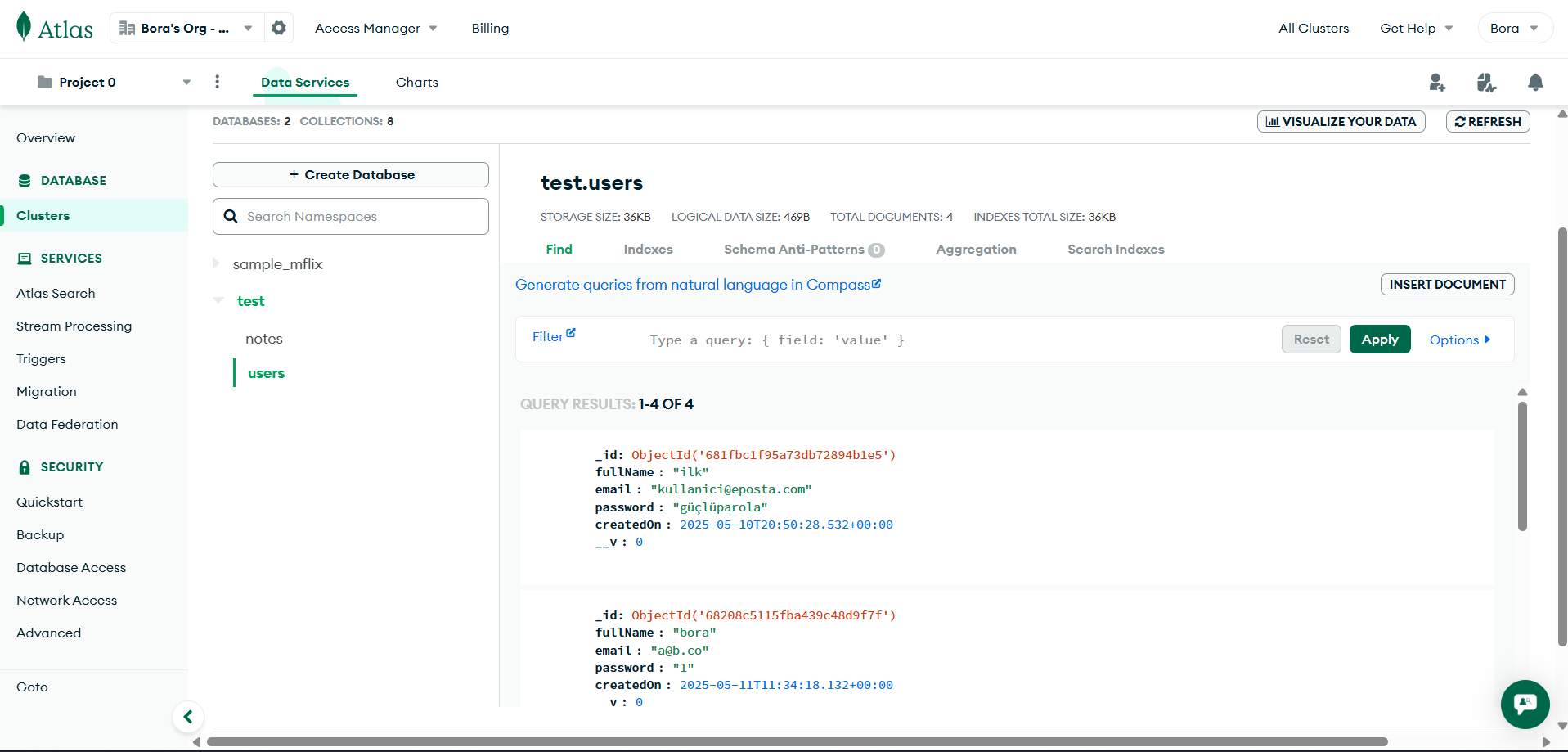
****

****

**2.3 Veritabanı**

* MongoDB: NoSQL, esnek ve ölçeklenebilir bulut tabanlı veritabanı





**3. Uygulama Özellikleri**

**3.1 Kullanıcı İşlevselliği**

* Kullanıcı kayıt ve giriş sistemi
* Not oluşturma, düzenleme ve silme
* Not arama fonksiyonu
* Notları sabitleme (pin) özelliği
* Okunabilir tarih formatı
* Bilgilendirme mesajları (Toast)
* Boş not listesi için özel kullanıcı arayüzü

**4. Geliştirme Süreci**

**4.1 Proje Kurulumu**

* Proje npm create vite@latest kullanılarak oluşturuldu
* Tailwind CSS resmi dokümantasyonuna göre entegre edildi
* Google Fonts'dan Noto Sans fontu kullanıldı
* Tarih biçimlendirme için moment kütüphanesi entegre edildi

**4.2 Frontend Geliştirme Aşamaları**

* Sayfa ve bileşen tasarımı
* React Router DOM ile navigasyon
* Tailwind CSS ile stil oluşturma
* Özel bileşenler geliştirildi:
* Navbar
* Giriş ekranı
* Şifre girişi bileşeni
* Arama çubuğu
* Not ekleme/düzenleme modalı
* Etiket girişi bileşeni

**4.3 Backend Geliştirme Aşamaları**

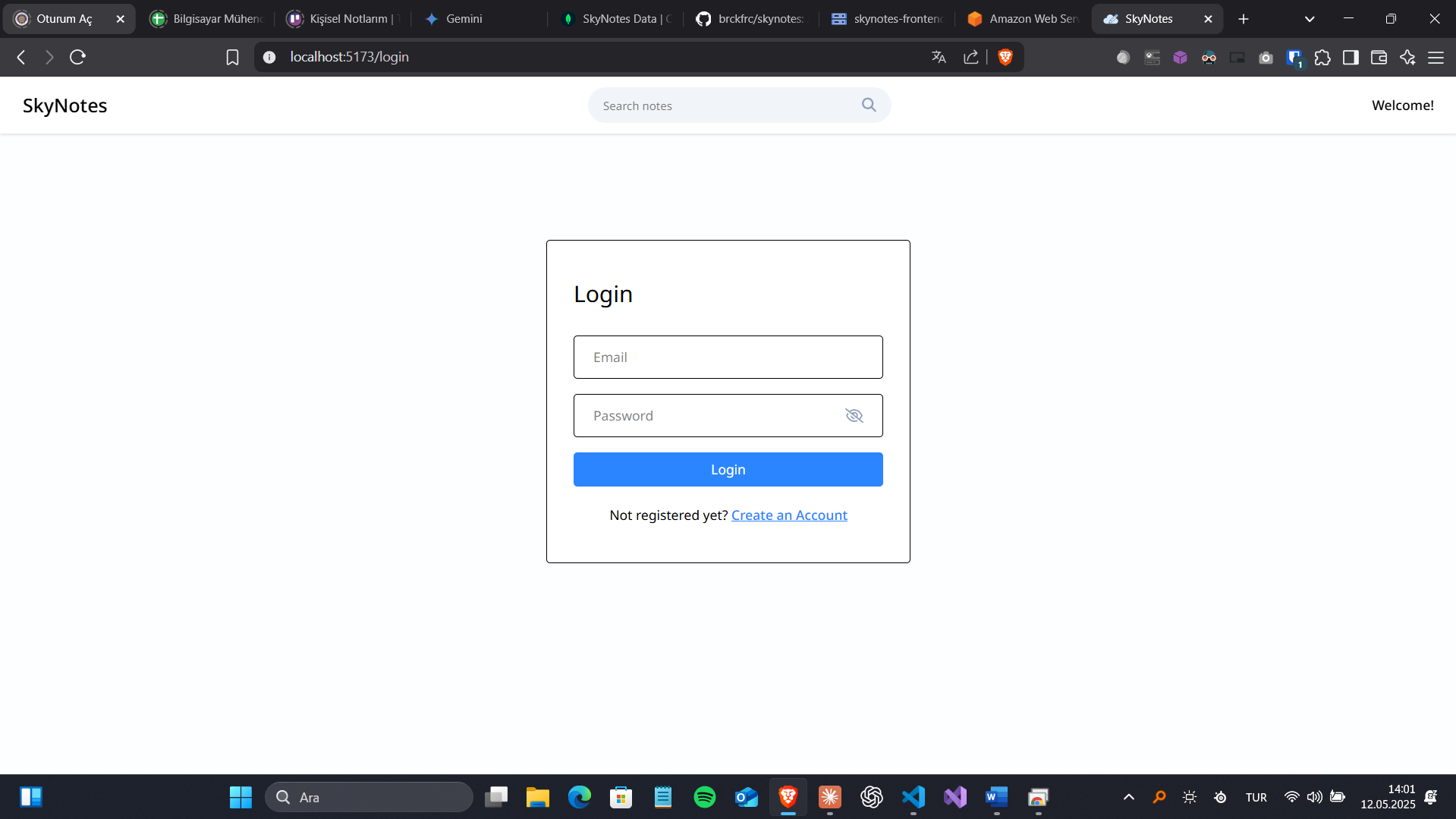
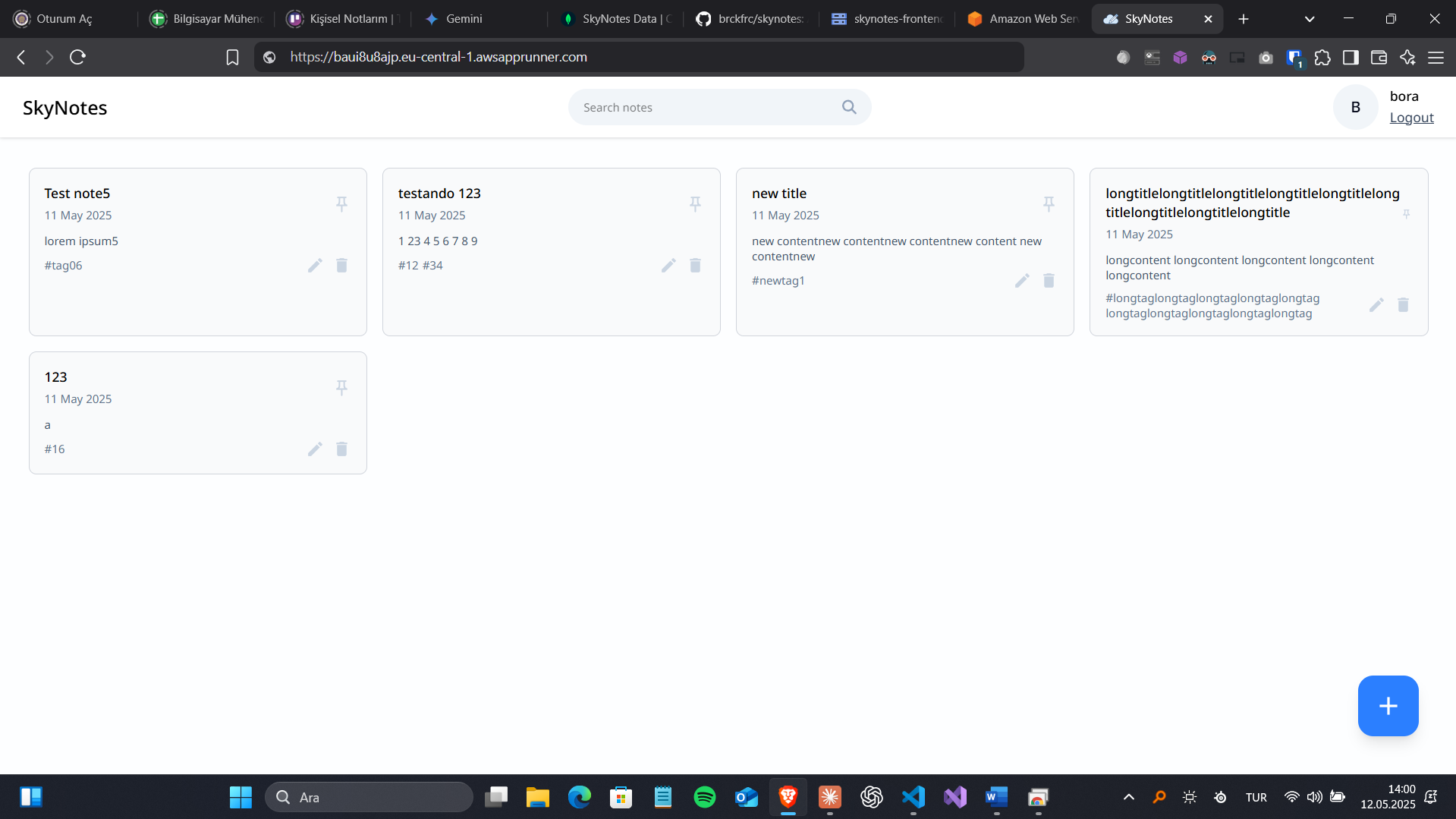
* Node.js, Express.js ve Mongoose ile API geliştirme
* Kullanıcı ve Not modelleri oluşturma
* API endpoint'leri hazırlama:
* Kullanıcı kayıt
* Kullanıcı girişi
* Not oluşturma
* Not düzenleme
* Not silme
* Not getirme
* Not sabitleme

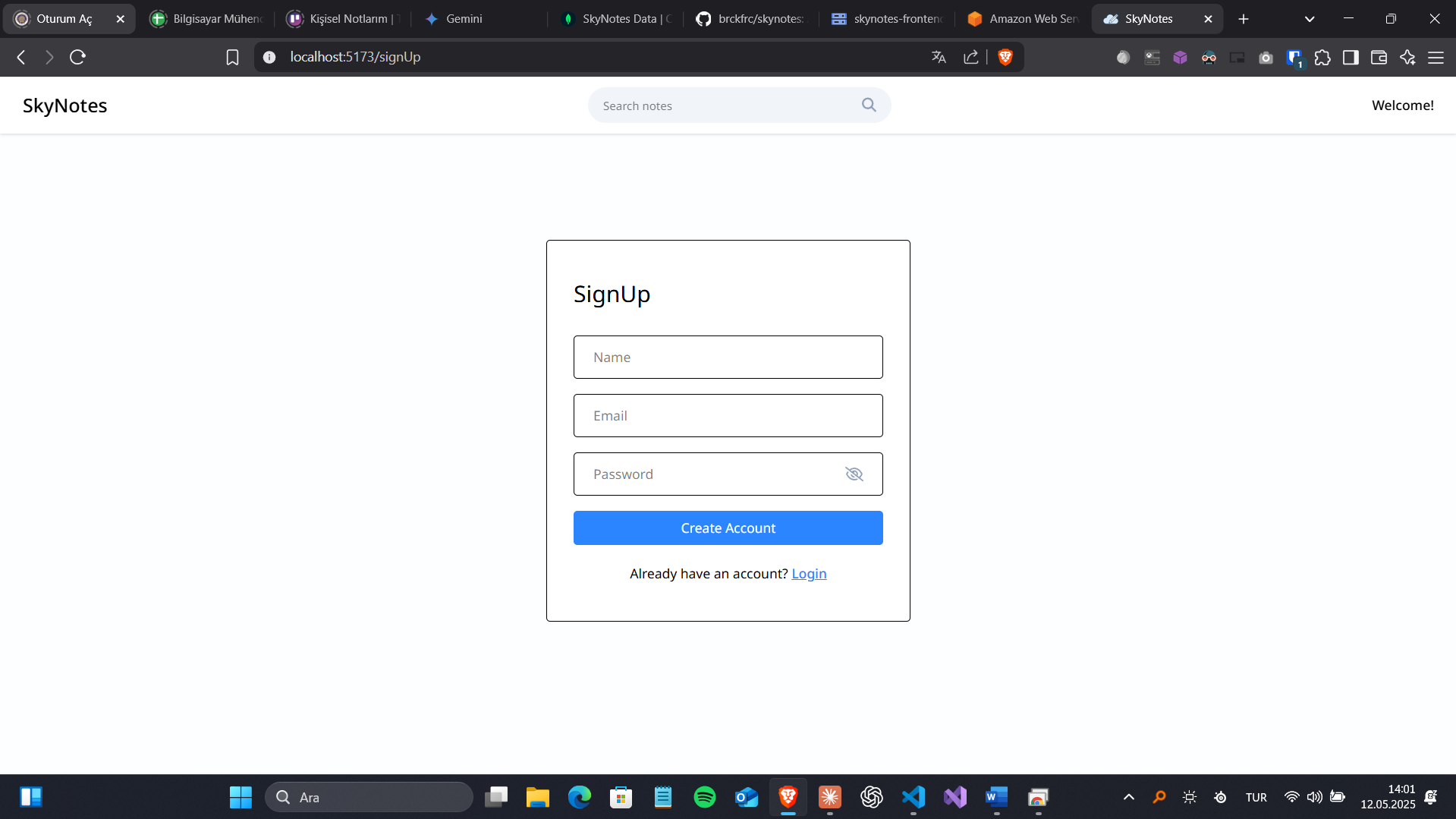
**4.4 Entegrasyon**

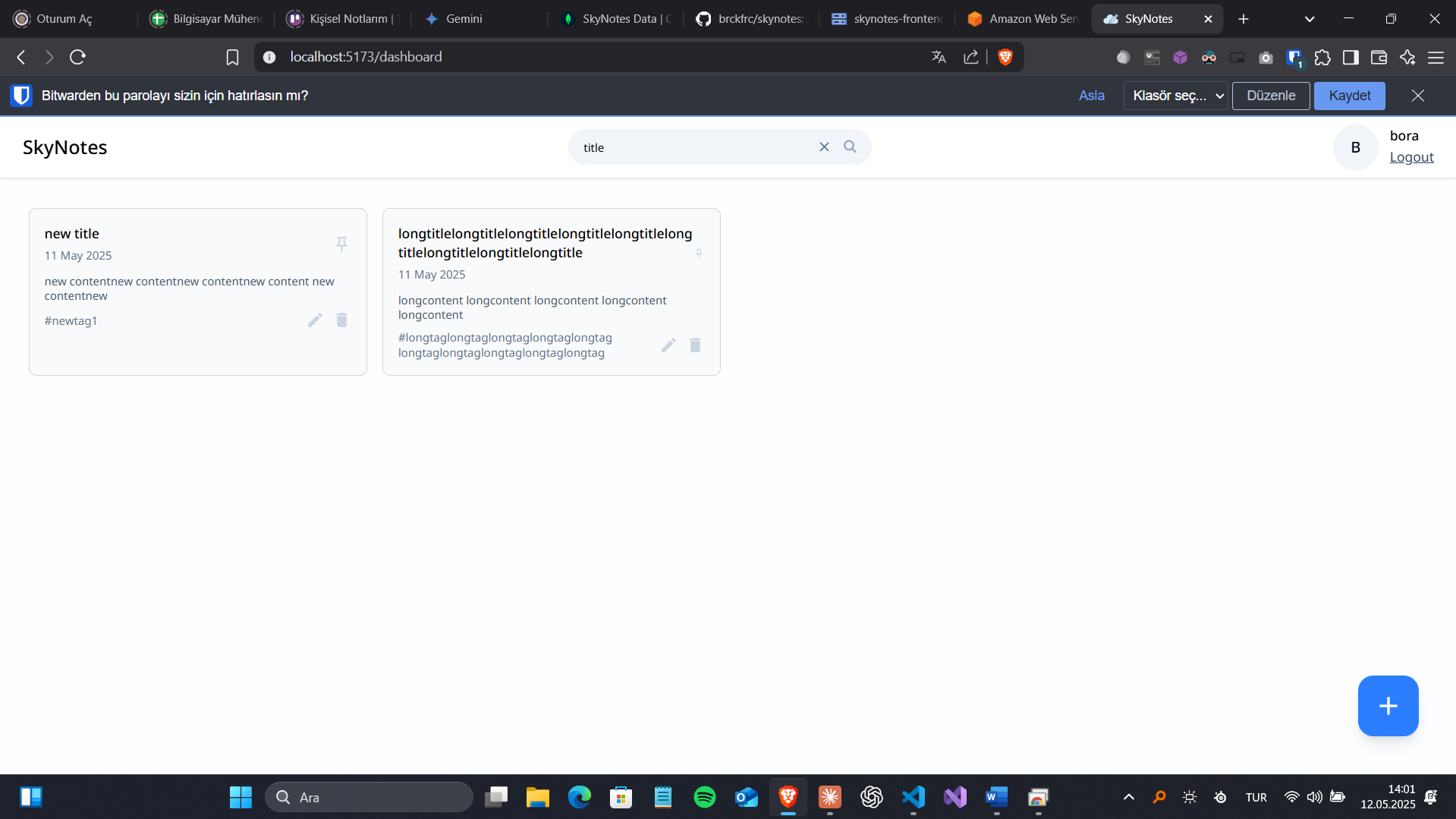
* Frontend ve backend arasında API bağlantıları kuruldu
* Kullanıcı bilgileri backend'den çekilip görüntülendi
* Kullanıcıya özel notlar listelendi
* Arama fonksiyonu entegre edildi
* Bilgilendirme mesajları (toast) eklendi

**5. Sonuç**

SkyNotes projesi, modern web geliştirme teknolojilerini kullanarak kullanıcı merkezli, esnek bir not yönetim uygulaması oluşturmayı başarmıştır. Proje, frontend ve backend teknolojilerinin entegrasyonunu gösteren kapsamlı bir örnek teşkil etmektedir.

****

****

****

**PROJE 2: Akıllı Veri Analitiği ve Makine Öğrenmesi Uygulaması**

**Proje Konusu: AWS kullanarak telefon özelliklerini bulunduran veri setini eğiterek fiyat aralık tahmini yapan modeli buluta deploy ettik ve buradan API oluşturarak veritabanı uygulamasında veri analizi gerçekleştirdik.**

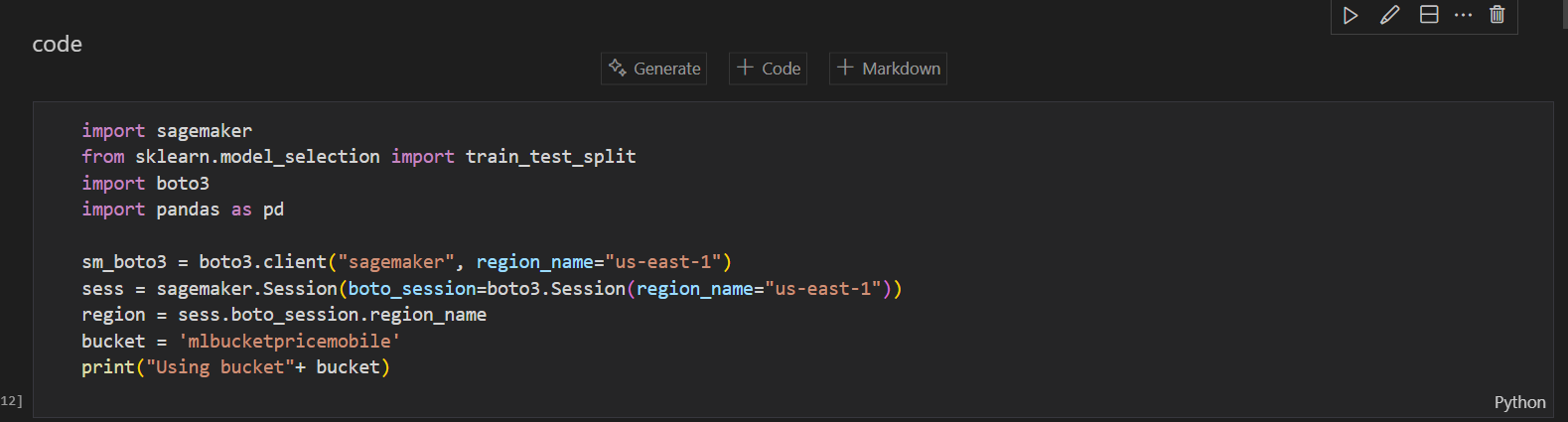
**Bu projede;**

**Backend dili: Python, Node.js**

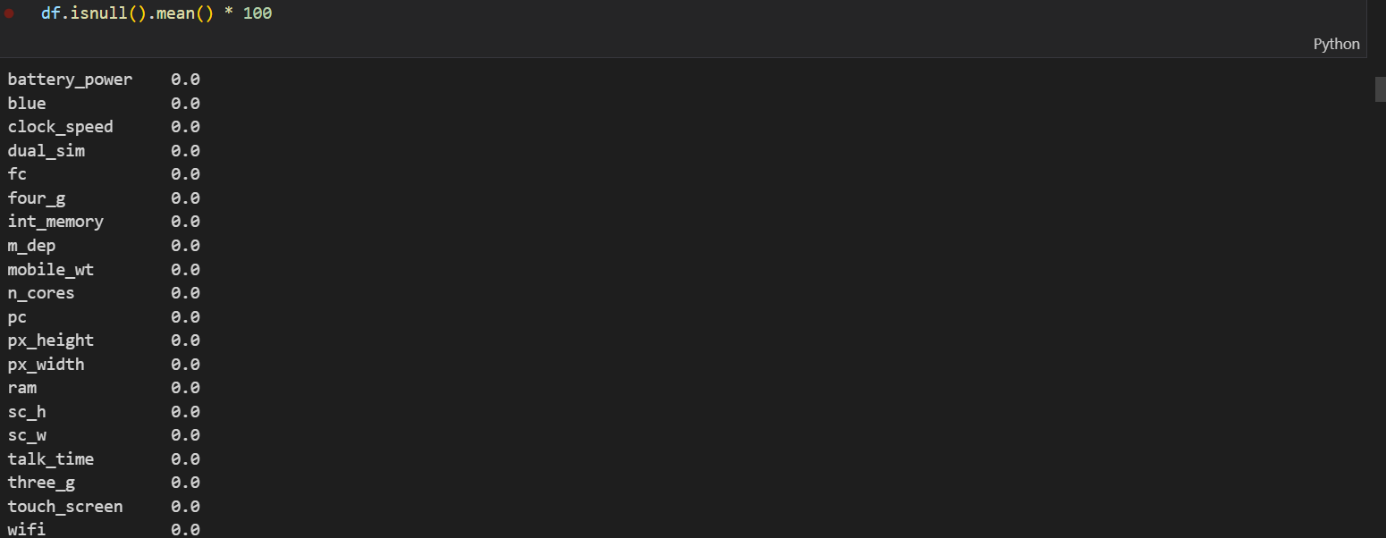
**Makine Öğrenmesi Kütüphaneleri,**

**Veri Tabanı: MongoDB,**

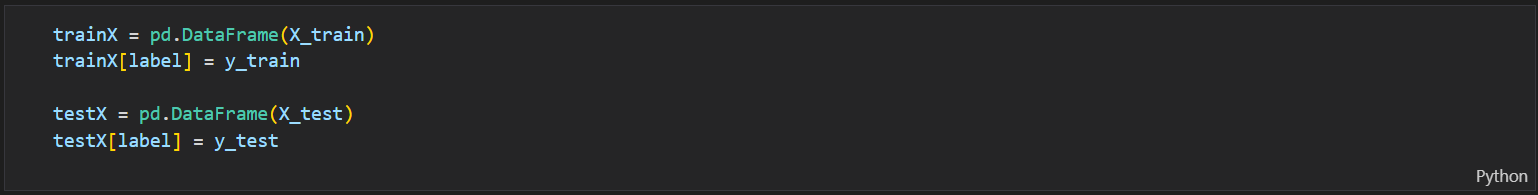
**Bulut Platformu: AWS (S3 Bucket, Sagemaker, Lambda, API Gateway)**

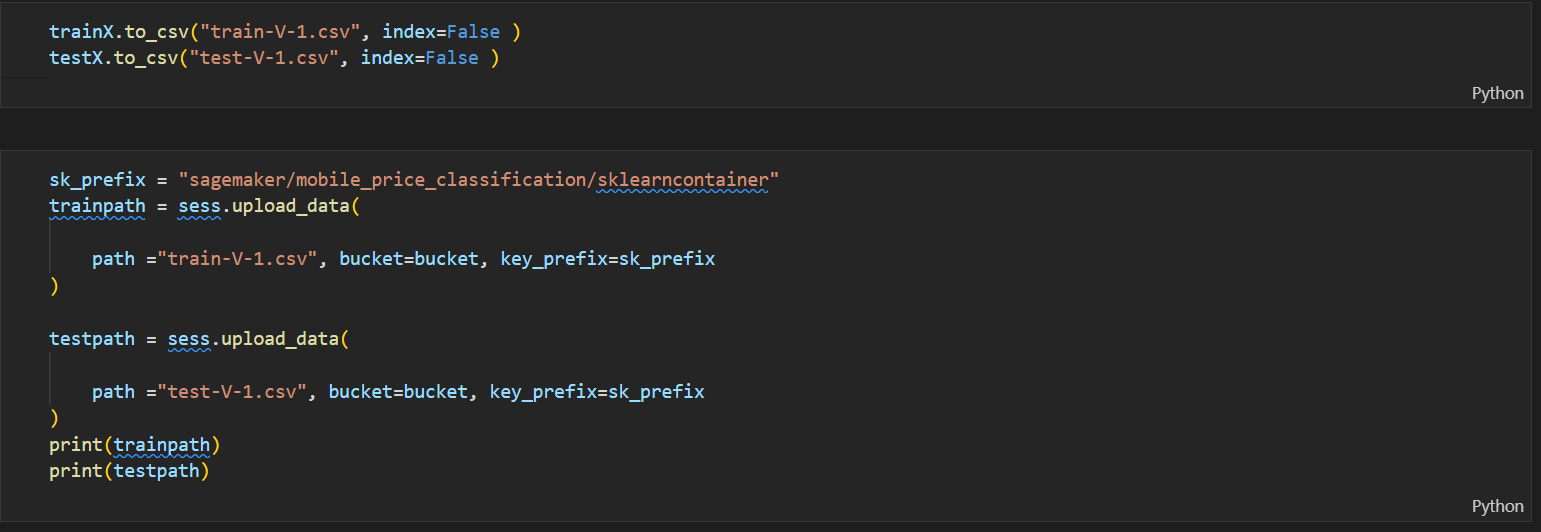
****

S3 Bucket oluşturarak localdeki kod ile bağlantı sağlandı.

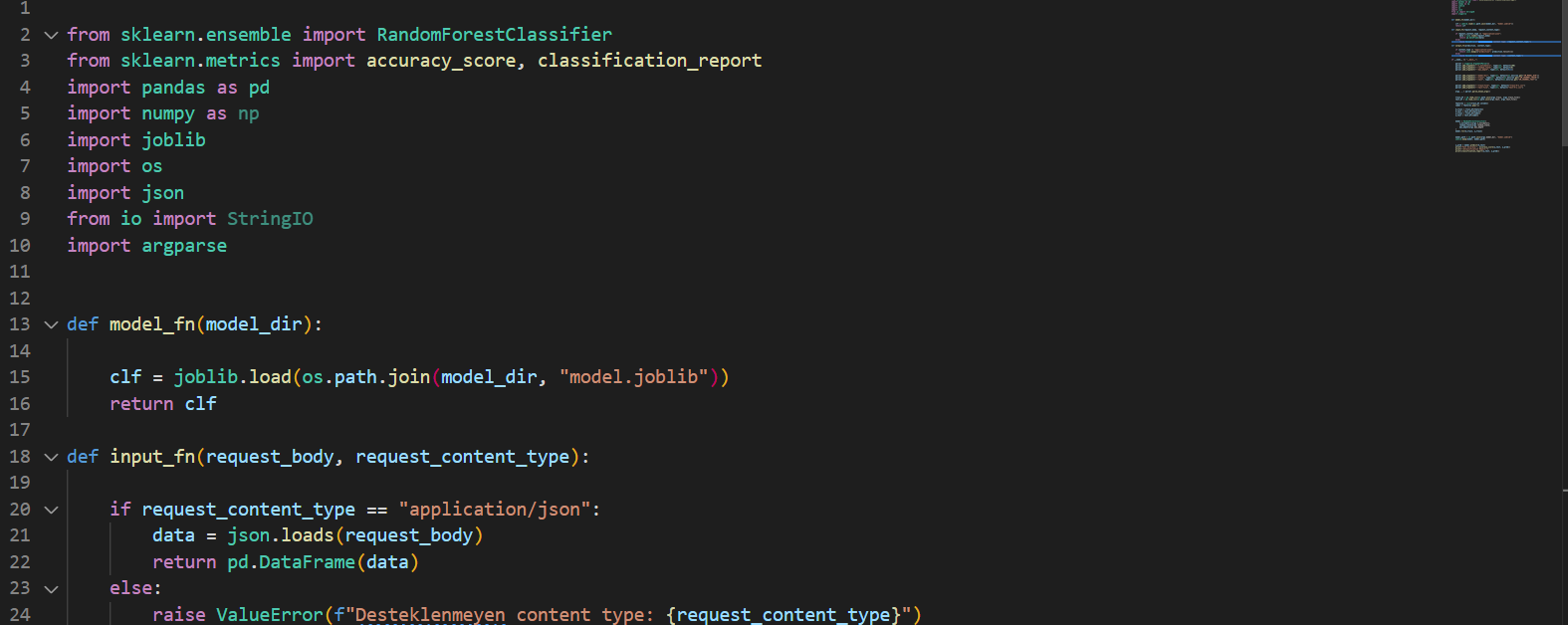


Elimizde bulunan dataset’i analiz edilerek eksiklikler gözlemlendi.

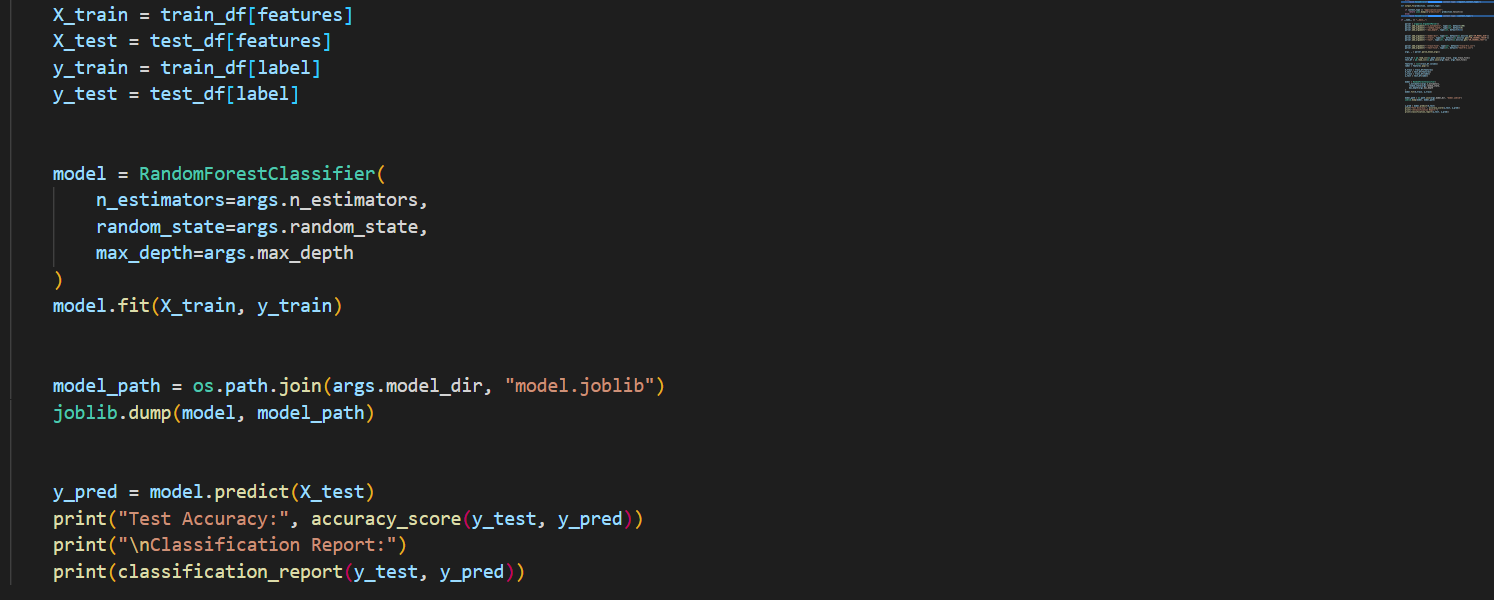




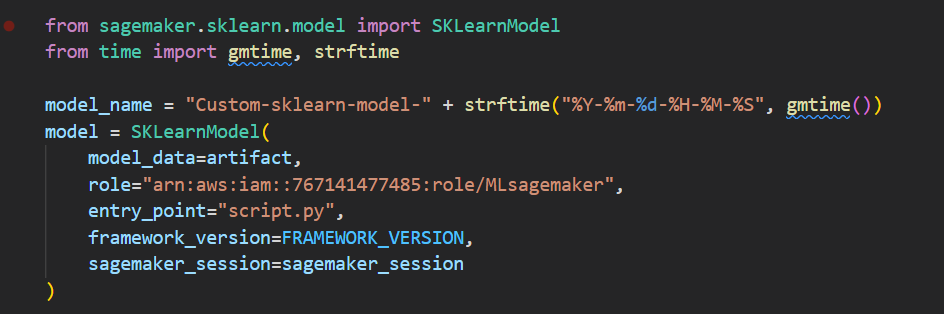
Bucket’ a var olan train.csv ile test.csv dosyaları yüklendi.



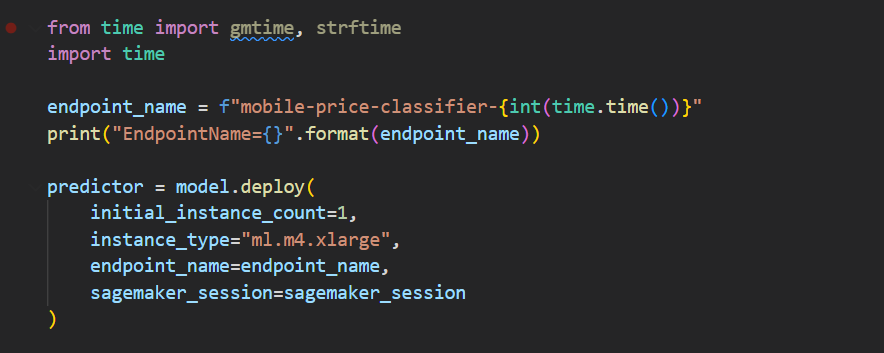




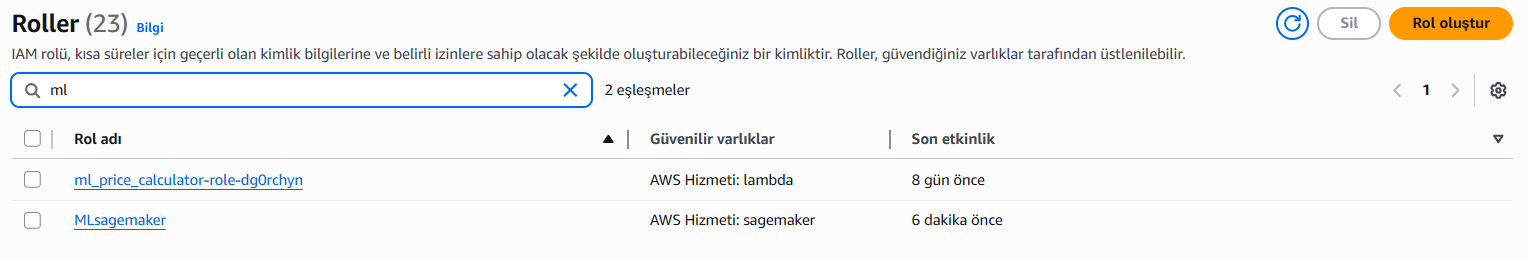
Randomforest ile elimizde bulunan dataseti eğitmek için ve daha sonra model eğitiminde kullanılması için script.py üzerine overwrite edildi.

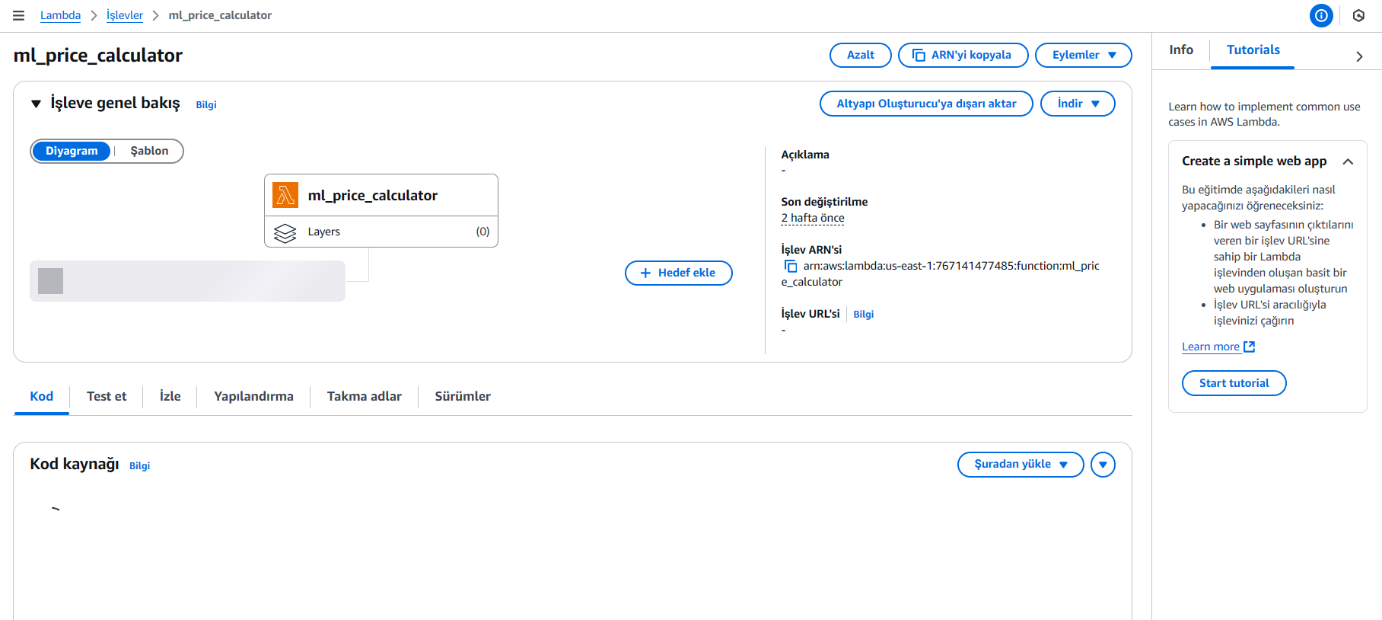


Aws üzerinde IAM rolü belirlendi ve bağlantı kurması için koda eklendi. Arından script.py ile model eğitildi.

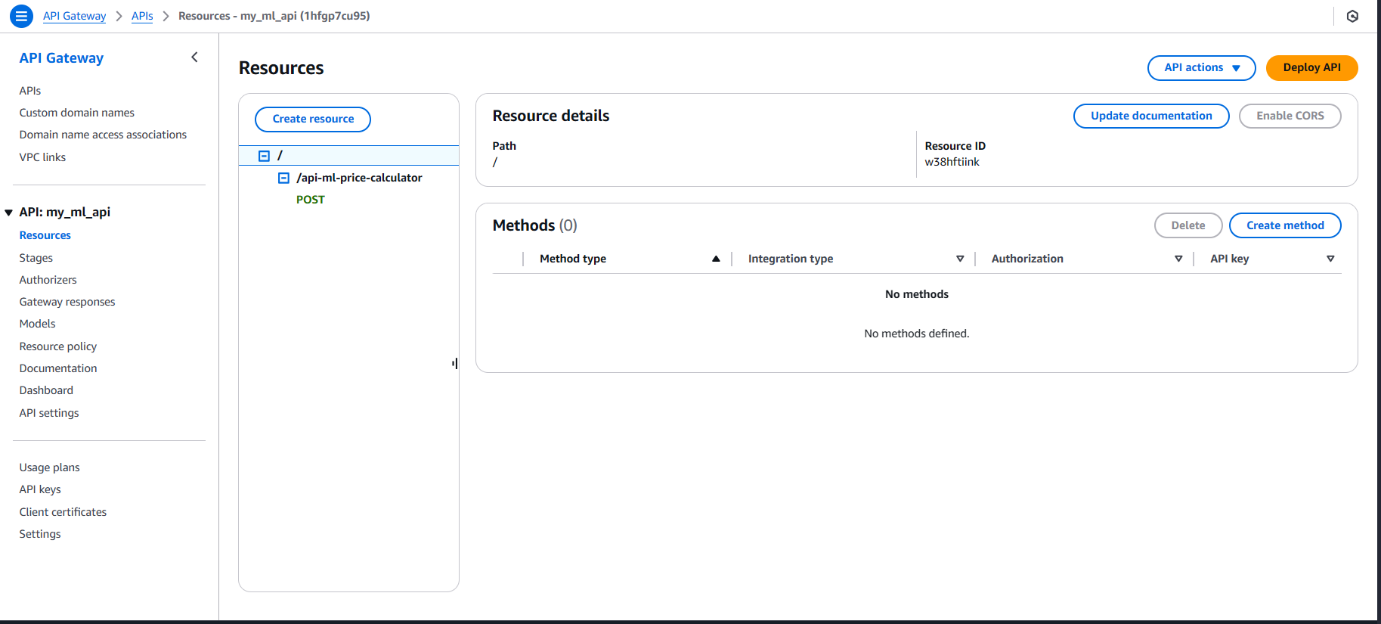


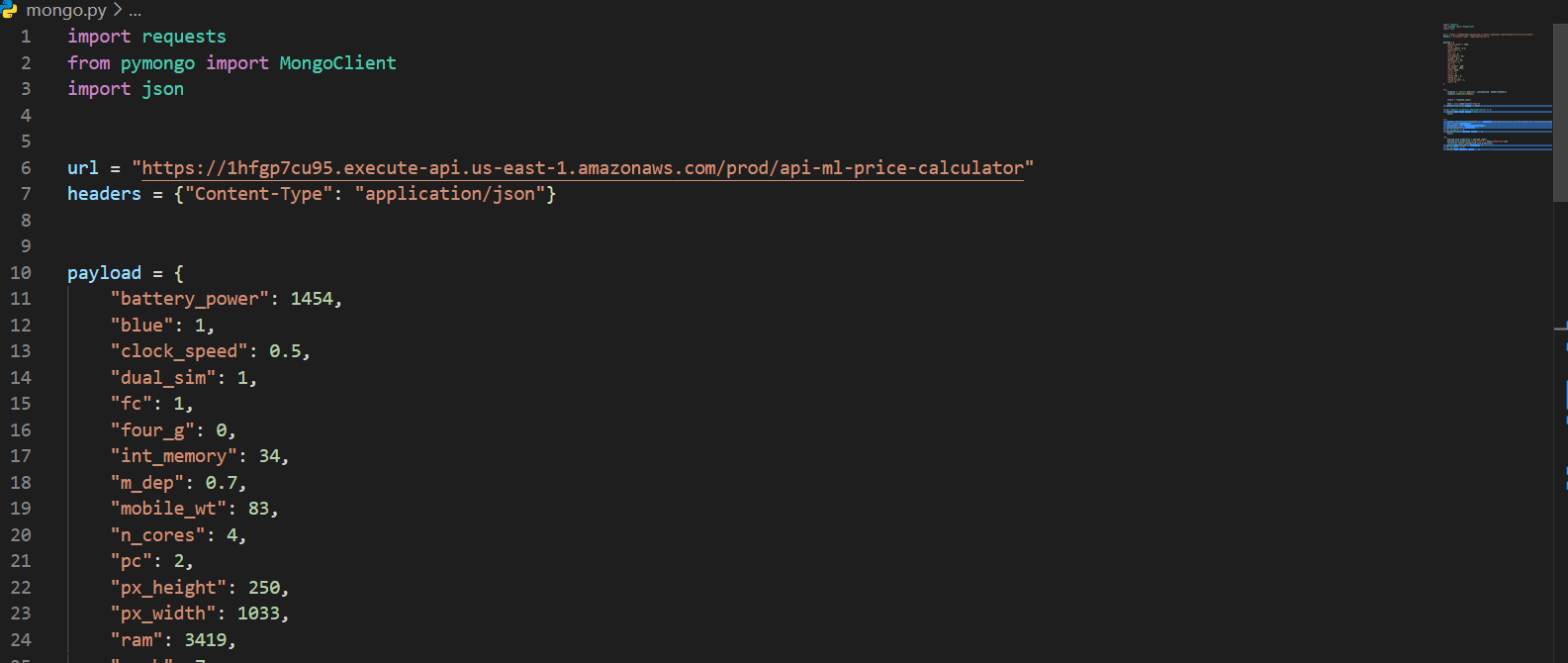
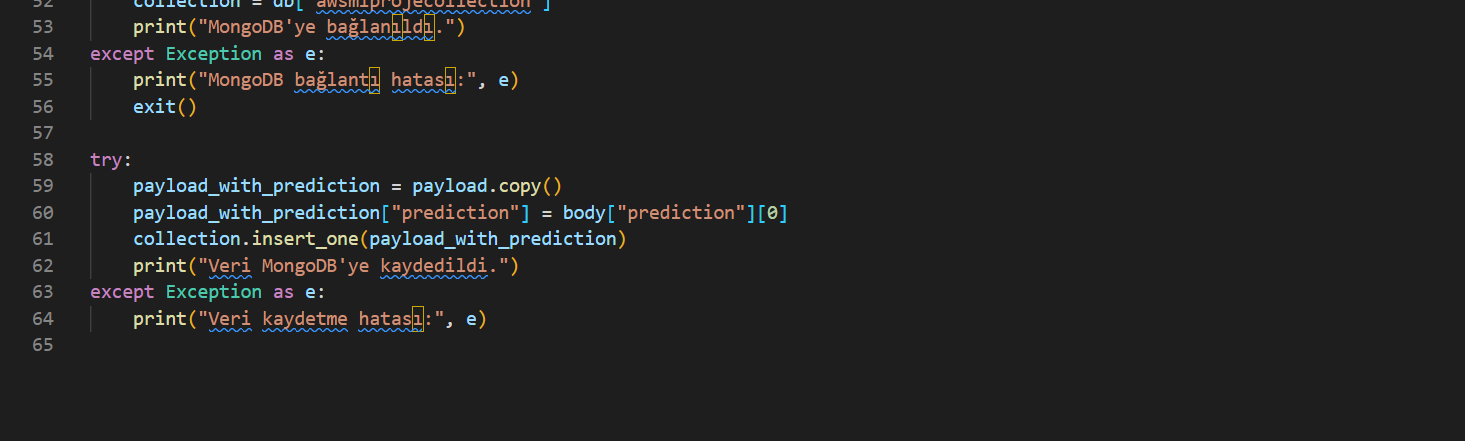
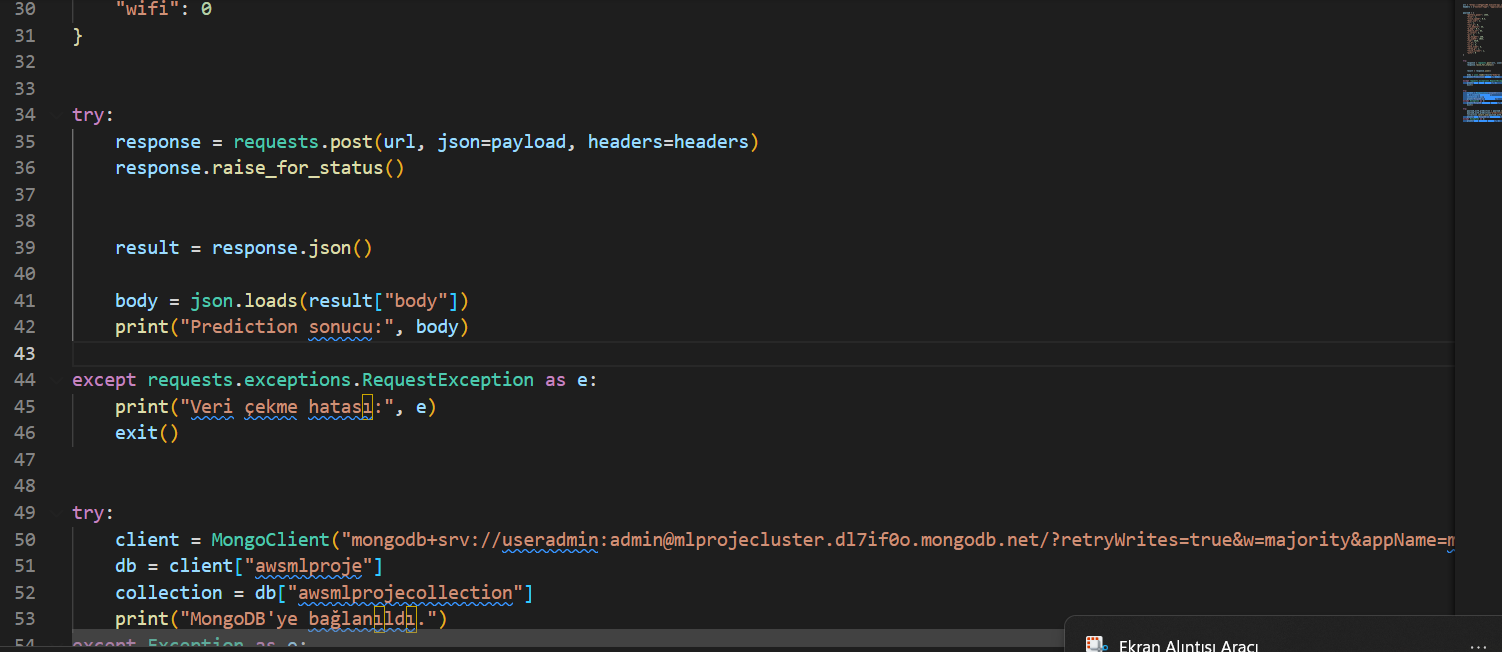
Bu kısımda Sagemaker için endpoint oluşturuldu.



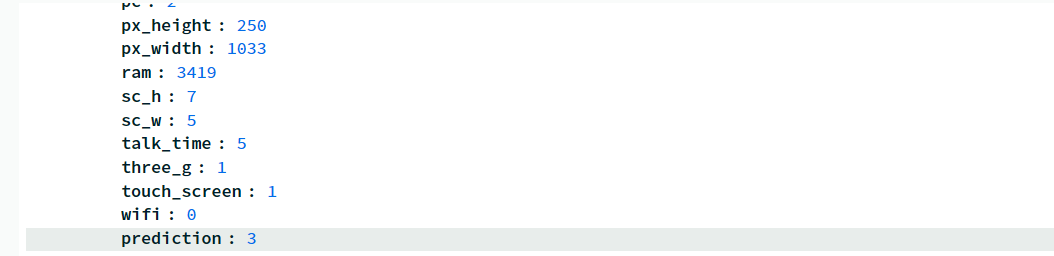
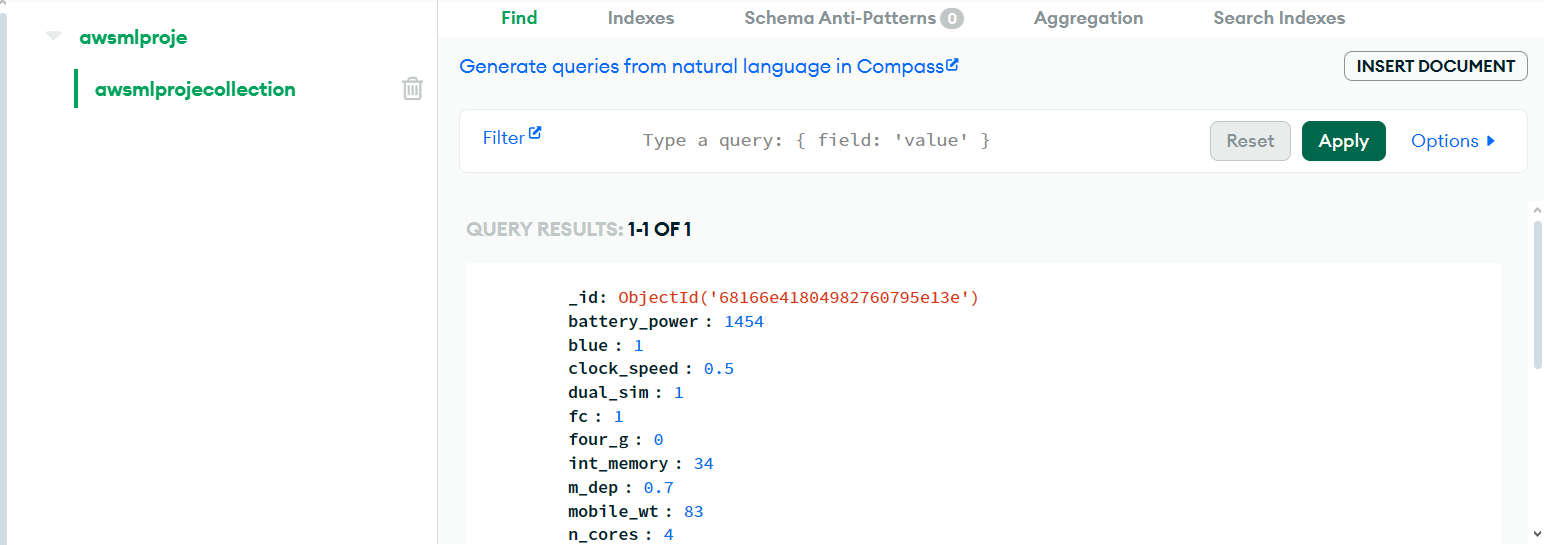


Ardından, Lambdaya IAM üzerinden sagemaker erişim rolü verildi. Ardından bu endpoint Lambda’ya bağlandı.



API Gateway kullanılarak MongoDB için RESTAPI oluşturuldu ve deploy edildi.

Bu kısımda MongoDB ile AWS API bağlantısı sağlandı ardından local kodda bu bağlantı gösterildi. Test verisi için veri yollandı ve ardından kod çalıştırılarak API kullanıldı



Sonuç olarak, MongoDB’ye Tahmin ve test verisi gönderildi ve gözlemlendi.