**Proje Raporu**

***Abstract –* Bu rapor, proje boyunca yapılan çalışmaları özetlemek ve açıklamak amacıyla oluşturulmuştur. Raporda projenin ne olduğu, hangi amaca hizmet ettiği, hangi yöntemlerin kullanıldığı, kullanılan geliştirme ortamı ve yalancı kod bilgisi bulunmaktadır.**

**I. GİRİŞ**

Ülkemizde ve tüm dünyada 2020 pandemisi ile beraber e-ticaret, satış hacimlerini ciddi derecede arttırmıştır. Bu artış beraberinde sitelerin gelişimini desteklemiş ve sayılarını arttırmıştır. Bu artış ile birlikte alıcının ürüne ulaşması için alternatif bir çok site varolmuştur.

Alıcılar daha ucuz ürüne ulaşabilmek için internet üzerinde bu alternatif sitelerde fazlasıyla zaman harcamakta. Bu proje kapsamında farklı e-ticaret sitelerden veriler çekilerek, kullanıcının bu acı noktasını hafifletmek amaçlanmıştır.

Proje kapsamında “Trendyol” ve “n11” e-ticaret sitelerinden veriler çekilmiş, PostgreSQL veritabanına Python programlama dili kullanılarak yazılmış ve yine Python’ ın Django modülü ile web sitesi projesine dönüştürülmüştür. Deployment için ise Heroku kullanılmıştır.

**2. YÖNTEM**

***2.1. Çalışma Ortamının Ayarlanması***

Öncelikle programlama dili olarak Python seçilmiştir. Veri kazıma (scraping), veritabanına kayıt işlemleri Python üzerinden yürütülmüştür. Veritabanı için PostgreSQL kullanılmış, PostgreSQL ve Python bağlantısı psycopg2 modülü yardımıyla sağlanmıştır. Proje dosyasının yönetimi Microsoft Visual Studio Code IDE’ si aracılığıyla sağlanmıştır.

**2.2. Proje Çalışma Mekanizması**

Web sitesi anlık olarak veritabanından beslenmekte olup, veritabanının güncelliği kullanıcının kararına bırakılmıştır. Proje dosyası içinde yer alan “scraper.py” aracılığı ile, ilgili e-ticaret sitelerindeki güncel veriler çekilebilmektedir. Eğer kullanıcı tarafından tercih edilirse, veritabanındaki tüm veriler “truncate cascade” yöntemi ile temizlenip sıfırdan da yazılabilir. Sitenin hızlı ve stabil çalışabilmesi adına, sistem bu şekilde kurulmuştur.

**2.3. Proje Ağacının Tanıtımı**

Proje, Python/Django modülü ile yazıldığı için, proje ağacı Django üzerinden tanıtılacaktır.

Şekil 1’ den görüleceği üzere proje ağacı 1 ana dosya ve 2 alt dosyadan oluşmaktadır.

İlk alt dosya olan eticaret dosyasında asgi.py ve wsgi.py dosyaları deployment için gerekli dosyalar olup, urls.py dosyası proje içerisindeki url’ leri yönetmek için kullanılırken, settings.py dosyası veribanı bilgilerinin, proje secret key’ inin ve statik dosyası lokasyonlarının bulunduğu dökümandır.

İkinci alt dosya olan scraping dosyasında ise veritabanına yapılan migration cacheleri, statik (img, css ve js dosyaları) dosyaları, hmtl templateleri, admin sayfası, veritabanı entegrasyon sayfası, scraping dosyası ve back-end dosyası bulunmaktadır.

Ana dosyanın direkt olarak içinde bulunan “manage.py” server ve local server üzerinde çalıştırmak için kullanılırken, “Procfile“deployment için ve “requirements.txt” ilgili kütüphanelerin sanal ortama kurulumu için kullanılmaktadır.

**Şekil 1. Proje Ağacı**

Graphical user interface

Description automatically generated with medium confidence

**2.4. scraper.py dökümanı**

***2.4.1. Veritabanı bağlantı fonksiyonu: dbConn()***

Bu fonksiyonda, psycopg2 kütüphanesinden faydalanılarak, veritabanı bağlantısı gerçekleştirilmiştir. Bu bağlantı ile scraper.py üzerinde insert, update ve select komutları kullanılmıştır.

*connection, cursor global tanımla*

*user, password, host, port, database değişkenlerini tanımla*

*connection cursor oluştur*

***2.4.2. Veri Kazıma Fonksiyonları: getTrendyolData(), getn11Data()***

veritabanı bağlantısını gerçekleştir

main\_url tanımla

1.sayfadan başlayarak 10.sayfaya kadar:

Sayfa içeriğini parse et ve sayfadaki ürün linklerini bul

parse ettiğin veriden görselleri bul

çektiğin veriler None değilse:

değişkenleri ata ve json verilerinden bilgileri çek

çektiğin verilere near duplicate kontrol yap

eğer veri veritabanında kayıtlıysa:

json verisetinden gelen veriye göre güncelleme yap

eğer kayıtlı değilse:

veritabanına kayıt at

**2.4.3. Django view fonksiyonu: getData()**

global değişkenleri tanımla

objects.values\_list ve objects.values fonksiyonları ile models.py’den veri çek

**2.4.3. Django view fonksiyonları: index(), lowPrice(), highPrice(), hplp()**

Bu fonksiyonlarda getData() üzerinden veritabanından okunan veriler, html templatelerine get request yapıldığında iletilmektedir.

Ayrıca filtreleme ve search fonksiyonları da bu fonksiyonlar içerisinde yer almaktadır.

getData() fonksiyonu ile verileri çek

eğer request POST request ise:

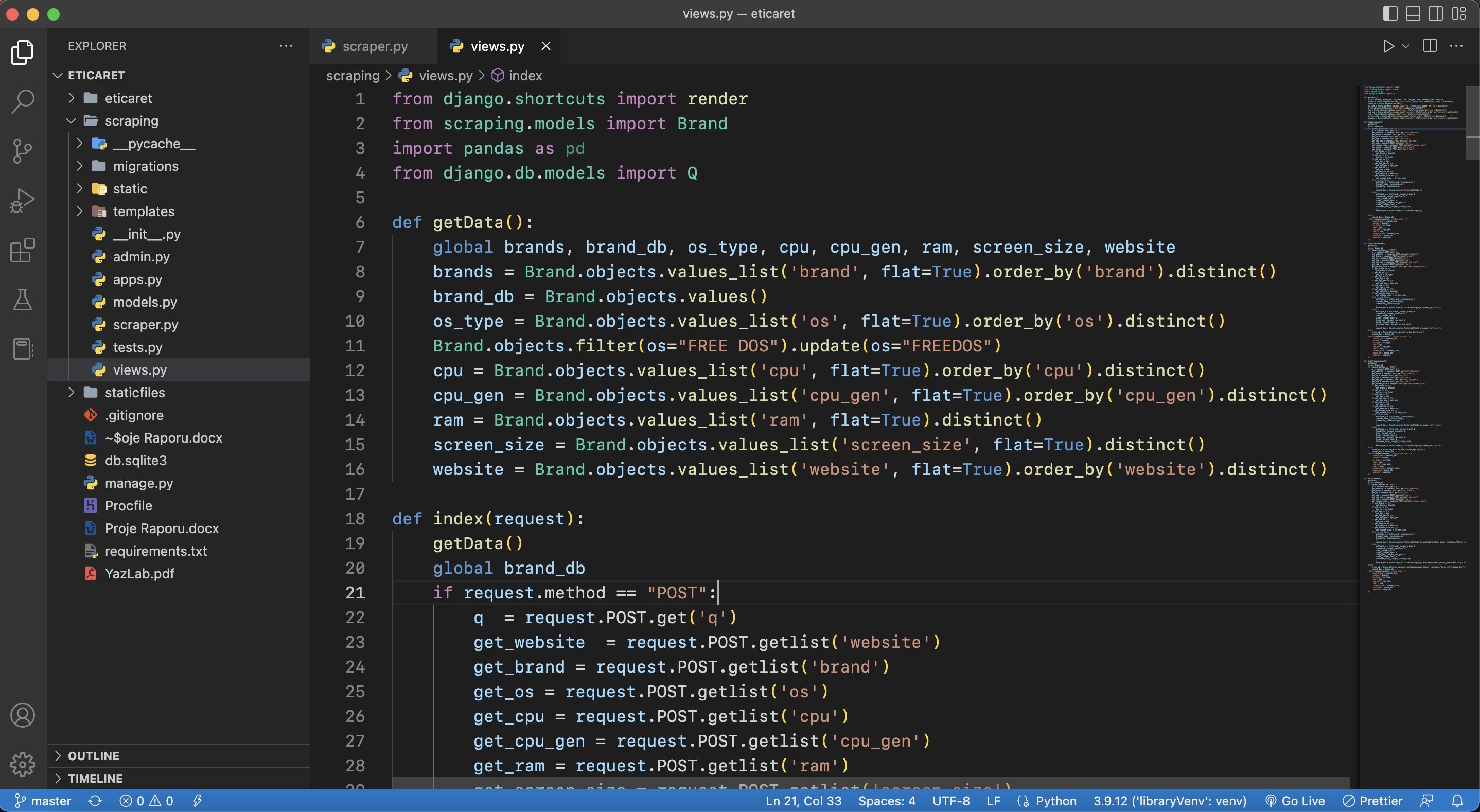
kullanıcı tarafından girilen filtre bilgilerini html name’leri üzerinden çek

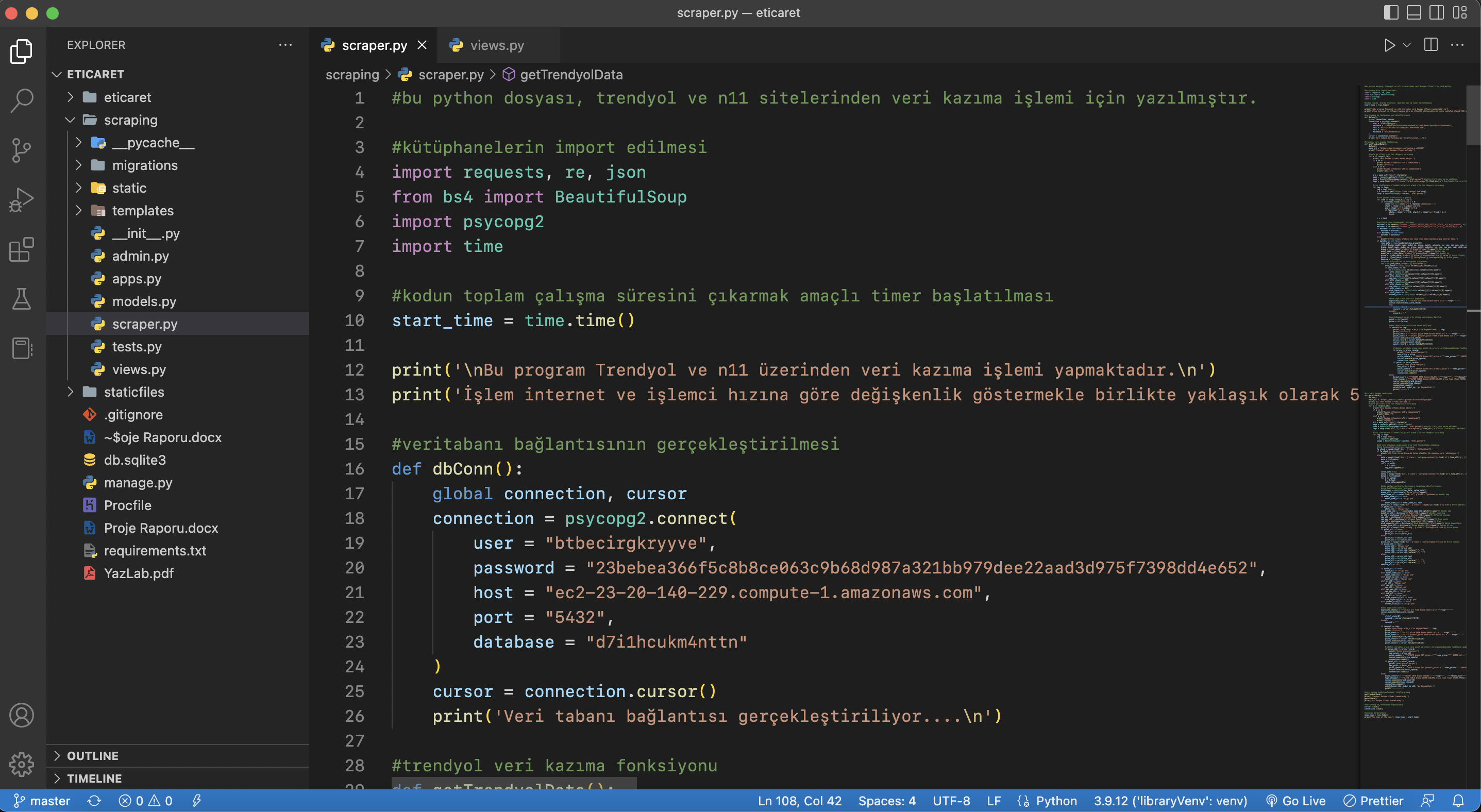
namelerden çektiğin verilerini models.py aracılığıyla veritabanında filtrele

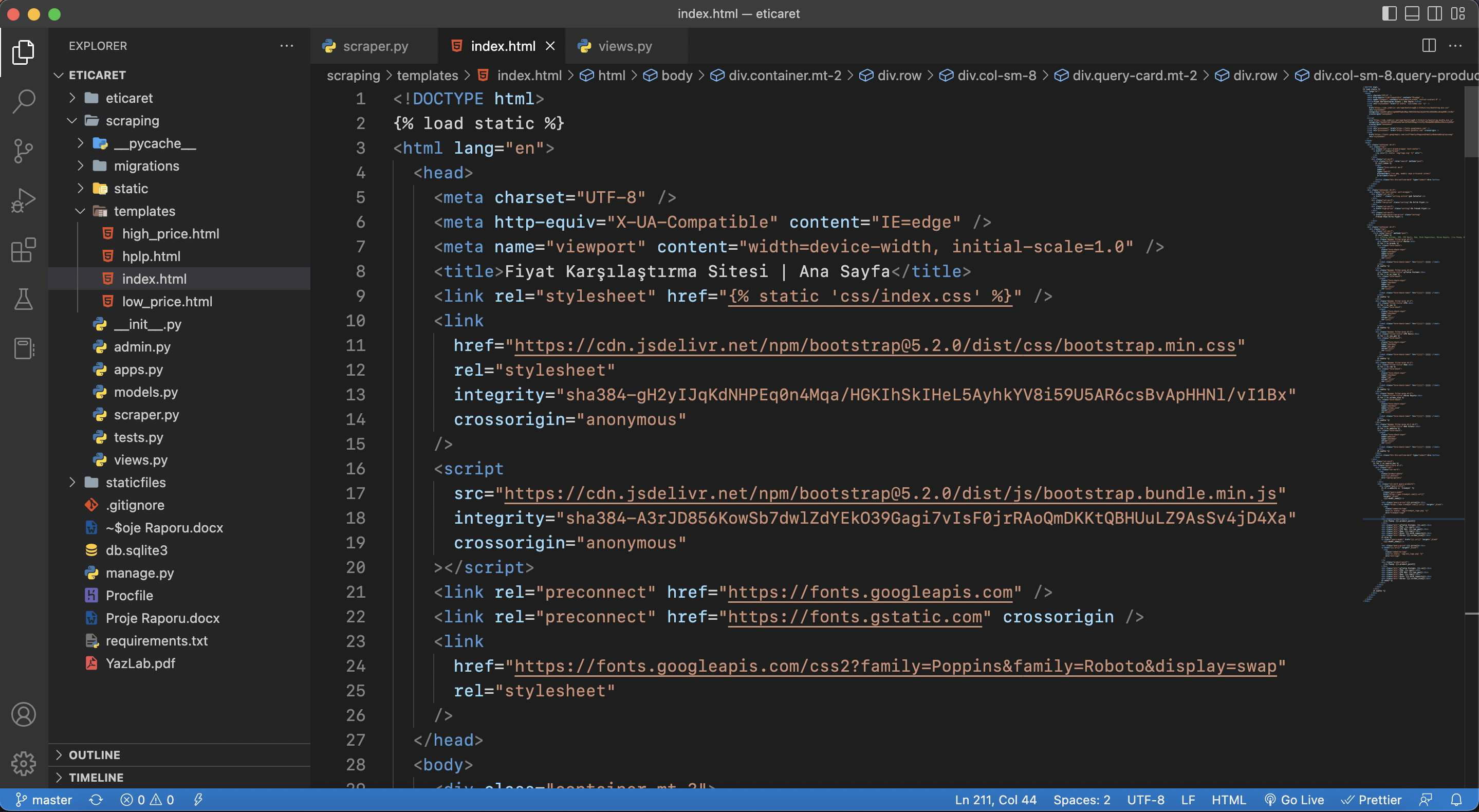
filtrelediğin veriyi değişkene ata ve ön-yüze gönder

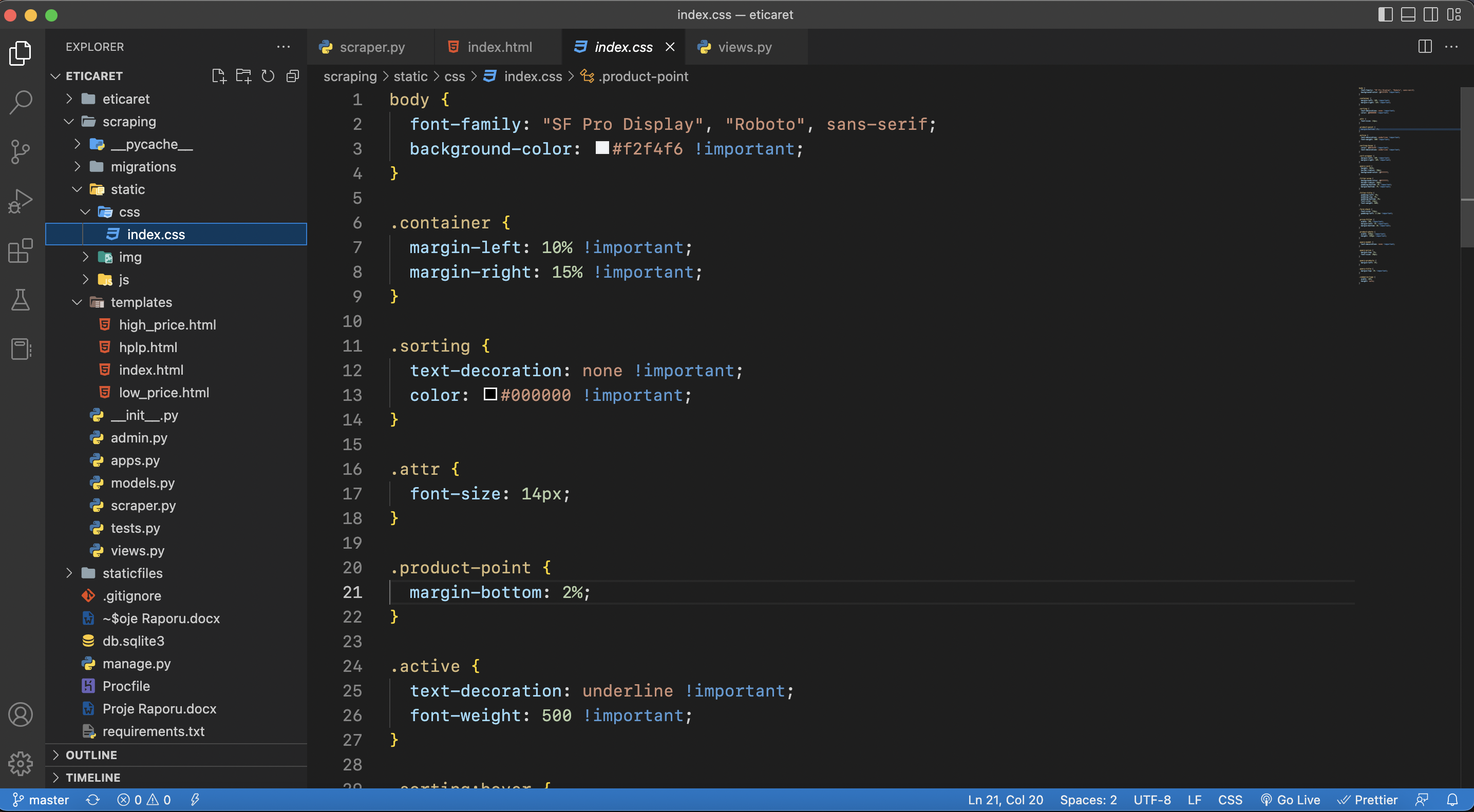
**3. DENEYSEL SONUÇLAR**

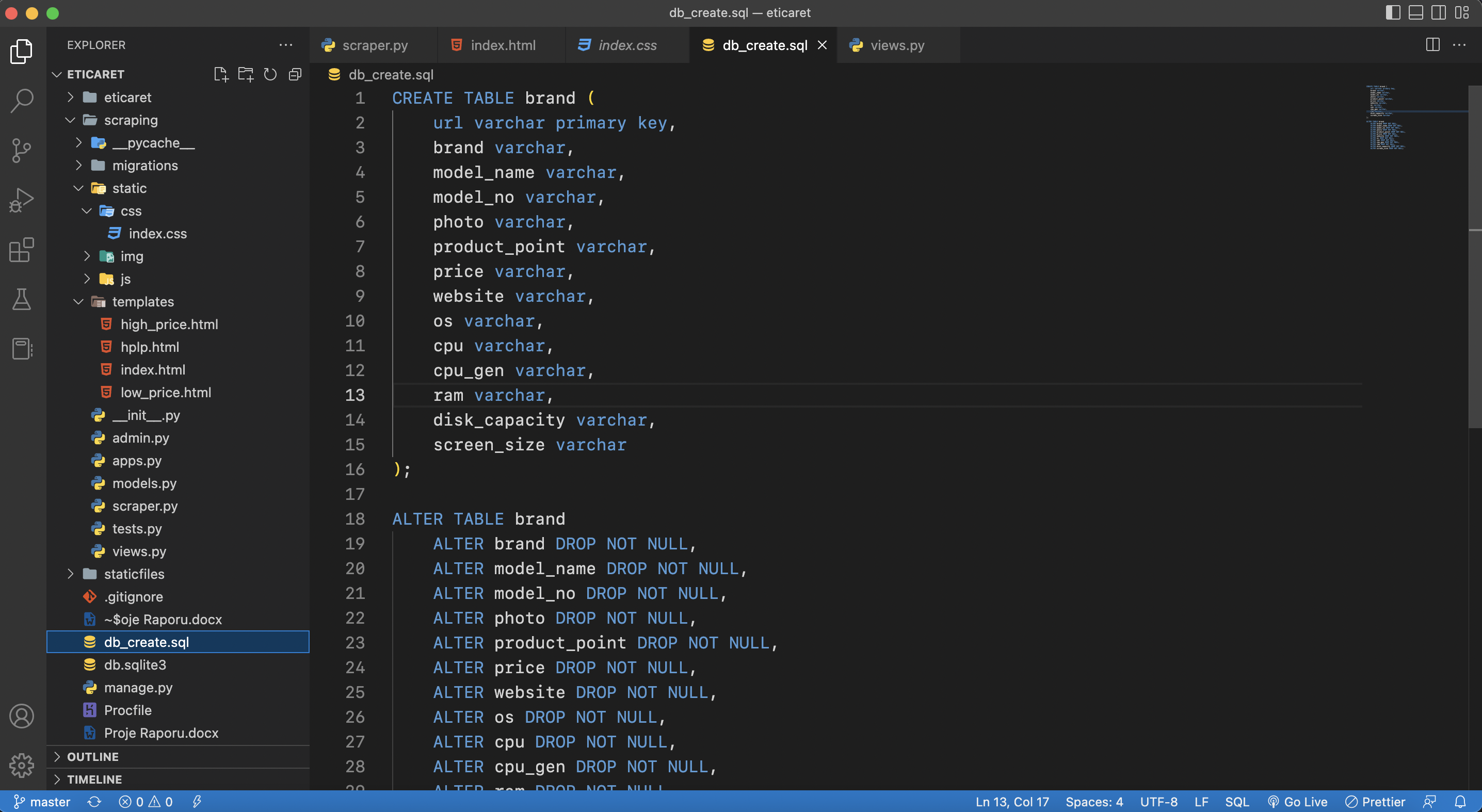
Bu başlıkta kod satırlarının ekran görüntüleri, web sitesinden görseller paylaşılmıştır.

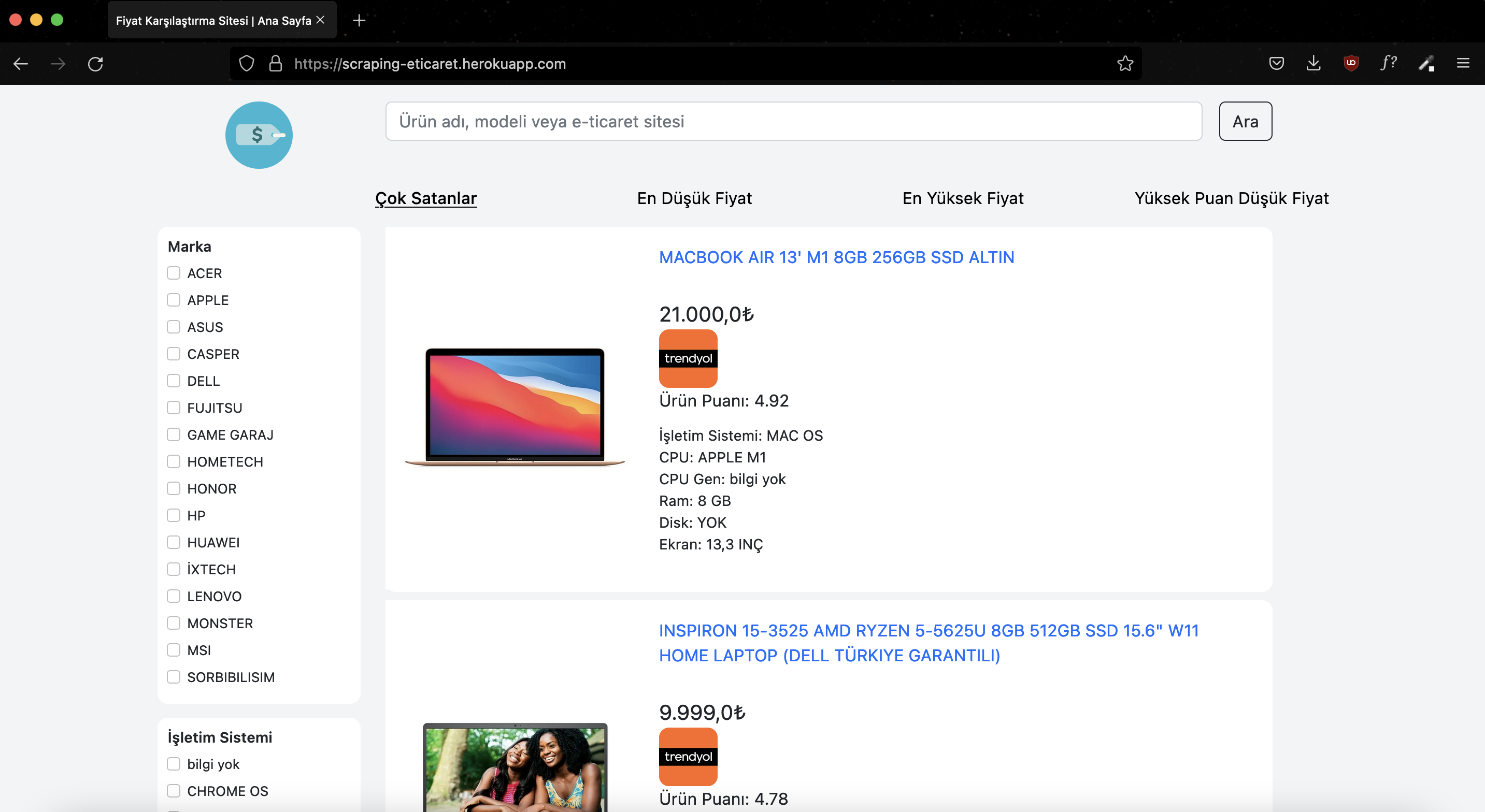












**4.SONUÇ**

Proje ile birlikte web siteleri üzerinden veri kazıma, python-veritabanı bağlantısı, front-end (html, css), back-end (python views.py), veritabanı oluşturma/güncelleme/silme işlemleri ve algoritma kurulması konularında bilgi edinilmiştir. Bunlara ek olarak github ve heroku gibi ortamlar ile deployment süreçleri öğrenilmiştir.

**5.KAYNAKÇA**

<https://www.medium.com>

<https://www.youtube.com>

<https://www.stackoverflow.com>

<https://www.heroku.com>

<https://towardsdatascience.com/>

<https://pandas.pydata.org/>

<https://www.w3schools.com>

<https://docs.djangoproject.com>