1. Önsöz(Bu sitenin amacı nedir?)
2. Kullanılan Programlar
    a. Php
    b. Javascript
    c. SQL(mysql)
    d. HTML/CSS
3. Genel çalışma pirensipleri
    a. Veritabanı bağlantısı(PDO)
    b. Veritabanı yönetimi ve procedures
    c. Ilişkili veritabanı tabloları ve önemi
4. Bilgi çekme ve işleme
    a. Php action method
    b. Js ile php action method tetiklenmesi
    c. PHP include
5. Kullanıcı Rolleri ve Görevleri
    a. Müşteri
    b. Satıcı / Yönetici
    c. Misafir
6. Müşteri Ekranları
    a. Anasayfa
    b. Ürün Arama
    c. Ürün Detayı
    d. Kullanıcı sepeti
    e. Ödeme sayfası
    f. Kullanıcı Panosu
7. Satıcı/Yönetici Ekranları
    a. Katalog Ürün ekleme
    b. Ürüne Fiyat verme
    c. Ürünleri Listeleme/İndirim-Stok- Güncelleme
    d. Satışları Listeleme /Kargo Bekleyen-Gönderilen-Tamamlanan
    e. Kupon Ekleme
    f. Editor'ün Seçtiği Ürünler

1. Önsöz (Bu sitenin amacı nedir?)

Alışveriş artık sadece mağazalarda değil, internette de yapılabiliyor. İnsanlar evden çıkmadan ürünlere ulaşmak, fiyat karşılaştırmak ve indirimlerden yararlanmak isteyebiliyor. Dünyada Amazon, eBay, Alibaba gibi büyük e-ticaret siteleri de bu ihtiyacı görerek kuruldu ve zamanla milyonlarca kişiye hizmet veren dev pazar yerlerine dönüştü.

Benim geliştirdiğim bu proje, aynı fikrin küçük bir örneği olarak hazırlandı. Amacı;

Kullanıcıların ürünleri kolayca bulabilmesi,

Kupon ve indirimlerle daha uygun fiyata alışveriş yapabilmesi,

Satıcıların ürünlerini rahatça ekleyip yönetebilmesi,

Sipariş ve kargo sürecinin herkes için şeffaf ve basit olmasıdır.

Kullanıcıların ürünleri önce sepete ekleyip hemen satın alabilmesi ya da favorilere kaydedip daha sonra kolayca ulaşabilmesi,

Ürünlerin kategori ve alt kategorilerle düzenlenmesi, böylece aranan ürünün daha hızlı bulunabilmesi.

Kısacası bu site, hem müşteriye güvenli ve kolay bir alışveriş ortamı, hem de satıcıya ürünlerini tanıtma ve satma imkânı sunmayı hedeflemektedir.

Bu proje, tek başıma geliştirdiğim bir çalışma olup yazılım alanındaki yeteneklerimi göstermek ve e-ticaret mantığını anlayabildiğimi ortaya koymak amacıyla hazırlanmıştır.

https://modaway.42web.io/Mysqlecommerce/index.php ‘dan görünüşünü görebilirsiniz.

İlgili link tasarım kısmını göstermek içindir. Site ücretsiz olarak (procedurelere) izin vermediği için bu kadar yapabildim.

2. Kullanılan Programlar

a. PHP

PHP'yi bu projede sitenin arka planda çalışan motoru olarak kullandım. Kullanıcı ekranda sadece ürünleri, fiyatları veya sepeti görüyor ama aslında bunların hazırlanmasını perde arkasında PHP yapıyor.Mesela:

Bir kullanıcı giriş yapabilsin diye oturum (login) sistemini PHP ile kurdum. Böylece kullanıcı giriş yaptığında kendi adına özel içerikler görebiliyor.

Sepete ürün eklendiğinde, ürünün bilgilerini veritabanından çekip ekrana getirmesini PHP sağlıyor. Böylece sepet her kullanıcı için ayrı ayrı çalışıyor.

Sipariş verildiğinde, ürün fiyatlarını toplayıp kupon indirimini düşerek toplam tutarı hesaplayan ve alışverişi kaydeden yine PHP oluyor.

Kısacası ben PHP'yi, bu sitenin arka planında (Back-end) kullandım. Bütün hesaplamalar, veritabanı işlemleri, kullanıcıya özel içerikler ve siparişin tamamlanması gibi kritik işler PHP sayesinde gerçekleşiyor.

b. Sql (MySQL)

Bu projede MySQL'i verileri saklamak, düzenlemek ve saklananların gösterilmesi için kullandım. MySQL'i sitenin hafızası gibi düşünebilirsiniz; bütün bilgiler burada tutuluyor ve gerektiğinde buradan çağrılıyor.

Bu 3 adımı daha detaylandırmak gerekirse:

Saklamak: Kullanıcıların kayıt olurken girdiği bilgiler, ürünlerin adı, fiyatı ve açıklaması, verilen siparişler ve kargo detayları MySQL tablolarında kayıtlı duruyor. Yani sitede ne kadar bilgi varsa, hepsi burada güvenli şekilde saklanıyor.

Düzenlemek: Ürünlerin fiyatı değiştiğinde, stok azaldığında ya da siparişin durumu güncellendiğinde bu işlemler SQL sorguları ile yapılıyor. Kısacası bilgilerin her zaman doğru ve güncel kalması MySQL sayesinde oluyor.

Göstermek: Kullanıcı siteye girdiğinde gördüğü tüm ürünler, açıklamalar, sepetindeki içerikler aslında veritabanından geliyor. PHP, SQL'e bir sorgu göndererek "bu ürünün bilgilerini getir" ya da "bu kullanıcıya ait sipariş kayıtlarını seç" der. SQL, veritabanındaki ilgili tabloyu tarar, gerekli satırları bulur ve sonucu PHP'ye döndürür. PHP de aldığı bu verileri işleyerek kullanıcıya ekranda gösterir.

Kısacası MySQL, bu sitenin bütün bilgileri düzenli bir şekilde saklamasını, gerektiğinde değiştirilmesini ve en sonunda kullanıcıya gösterilmesini sağlayan temel yapı taşıdır.

c. Javascript

JavaScript'i bu projede sitenin ön yüzünde (Front-end) kullandım. En önemli özelliği, sayfayı yenilemeden dinamik bir biçimde değişiklik yapılmasına yardımcı olmasıdır.Mesela:

Kullanıcı bir ürünün adedini artırdığında toplam fiyatın anında güncellenmesini JavaScript ile sağladım.

"Sepete Ekle" butonuna basıldığında sayfa yenilenmeden ürünün sepete eklenmesini yine JavaScript ile yaptım.

Açılır menüler, kaydırılabilir ürün listeleri ve uyarı mesajlarının ekranda gösterilmesi gibi etkileşimler de JavaScript sayesinde oldu.

Ayrıca JavaScript'i, arka plandaki PHP ve veritabanı ile iletişim sağlamasında kullandım. Böylece kupon kontrolü, sepetin güncellenmesi veya sipariş bilgilerinin alınması gibi işlemler sayfa yenilenmeden gerçekleşti.

JavaScript sayesinde site, kullanıcı için daha hızlı ve akıcı bir kullanım deneyimi sunar; işlemler sayfa yenilenmeden gerçekleşir ve yapılan her değişiklik anında ekrana yansır.

d. HTML/CSS

Bu projede HTML ve CSS, sitenin temel yapısını ve görünümünü oluşturmak için ihtiyaç duyulan yapılardır. HTML, bir not defteri gibi sayfaya içerik eklerken; CSS bu içeriğin renk, boyut, yazı tipi ve düzenini ayarlamada yardımcı olur.

HTML ve CSS'i tasarlarken, kullanıcıya özgün, düzenli ve göze hoş gelen bir arayüz sunmayı hedefledim. Bunu yaparken başarılı e-ticaret sitelerinden ilham alarak, sitenin hem basit hem de anlaşılır görünmesini sağlamaya çalıştım.

3. Genel Çalışma Prensipleri

a. Veritabanı Bağlantısı (PDO)

(PHP Data Objects) PHP'nin veritabanı ile konuşmak için sunduğu sınıftır.

```php
<?php
$host = "localhost";
$dbname = "mysqltestdb";
$user = "root";
$pass = "abc123";

try {
    $dsn = "mysql:host=$host;dbname=$dbname;charset=utf8mb4";
    $pdo = new PDO($dsn, $user, $pass);
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $pdo->setAttribute(PDO::ATTR_DEFAULT_FETCH_MODE, PDO::FETCH_ASSOC);
} catch (PDOException $e) {
    error_log("PDO Connection failed: " . $e->getMessage());
    http_response_code(500);
    exit(json_encode(['success' => false, 'message' => 'DB connection failed']));
}
```

Bu projede veritabanı bağlantısı için PDO'yu kullandım. PDO, farklı SQL sunucularıyla ortak şekilde çalışabilen bir altyapıdır. MySQL, PostgreSQL, Oracle gibi pek çok veritabanını aynı yöntemle yönetmeye imkân tanır.

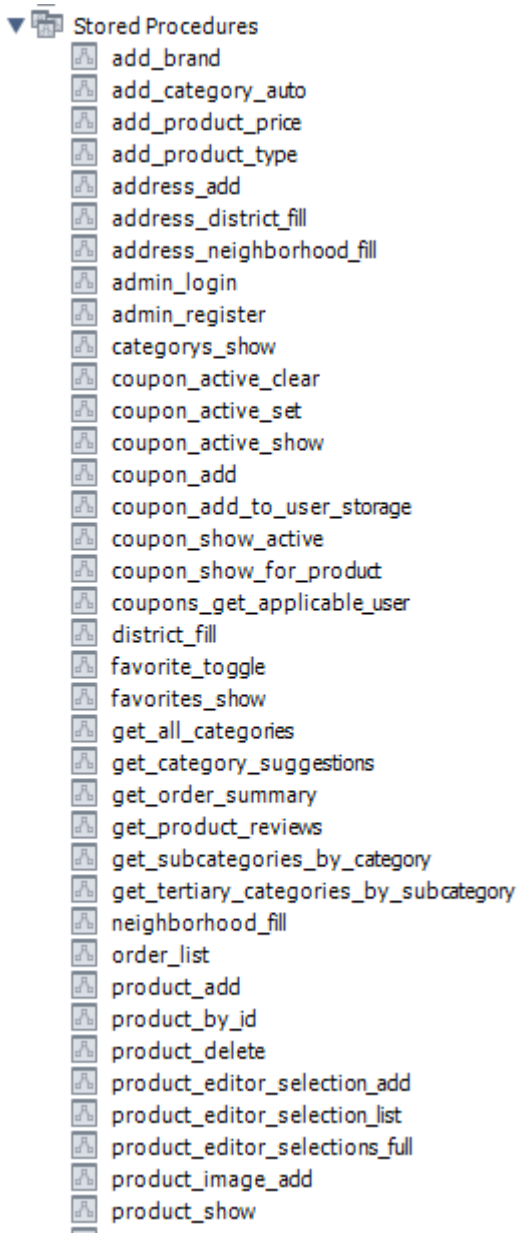
Araştırmalarım sonucunda PDO'nun güvenli, esnek ve modern bir yöntem olduğunu gördüm ve bu yüzden tercih ettim. Temel olarak yaptığı şey şudur: PHP'nin sunduğu sınıfları kullanarak, veritabanının adı ve şifresi gibi bağlantı bilgilerini PDO'ya iletiriz. Eğer bu iletişim başarılı olursa, veritabanına güvenli bir şekilde erişim sağlanmış olur.

Kısacası PDO, bu projede veritabanına bağlanırken kullandığım güvenli ve standart iletişim aracıdır.

Peki, "bu veritabanı nerede duruyor?" diye düşünebilirsiniz. Bunu şöyle hayal edelim: Bilgileri saklayan araçlarımız var (HDD/SSD gibi). Bu araçların evdeki modemimize bağlı olduğunu düşünün. Siz bilgisayarınızı ağa bağladığınızda modem, bu hafıza araçlarına erişir ve içindeki dosyaları size gösterebilir.

Veritabanı da aynı mantıkla çalışır. PDO sayesinde PHP, sunucudaki bu "hafıza aracına" bağlanır, gerekli bilgileri ister ve doğru şekilde geri alır.

b. Veritabanı Yönetimi(Procedure)



Bu projede procedure (saklı yordam) kullandım. Normal tablolar tek başına kullanıldığında, sistem büyüdükçe dallanarak kontrolden çıkacak bir dağınıklık oluşturabilir. Bu durum zaman kaybına ve kopukluklara yol açar. Procedure'ler sayesinde bu dağınıklığı toparlayıp işlemleri daha kolay, anlaşılır ve tekrar kullanılabilir hale getirmeyi amaçladım.

Kısaca procedure şunu sağlar:

İstediğim işlemi tek bir yerden, ekstra kod yazmadan çalıştırabilirim. Sonradan küçük bir değişiklik yapmam gerekirse, her şey düzenli çalıştığı için zorlanmadan güncelleyebilirim.

Aynı işlemi tekrar tekrar farklı yerlerde kullanmam gerektiğinde, tek sefer yazmam yeterlidir.

Proje büyüdükçe hem zaman kazandırır hem de anlaşılır bir yapı sağlar.

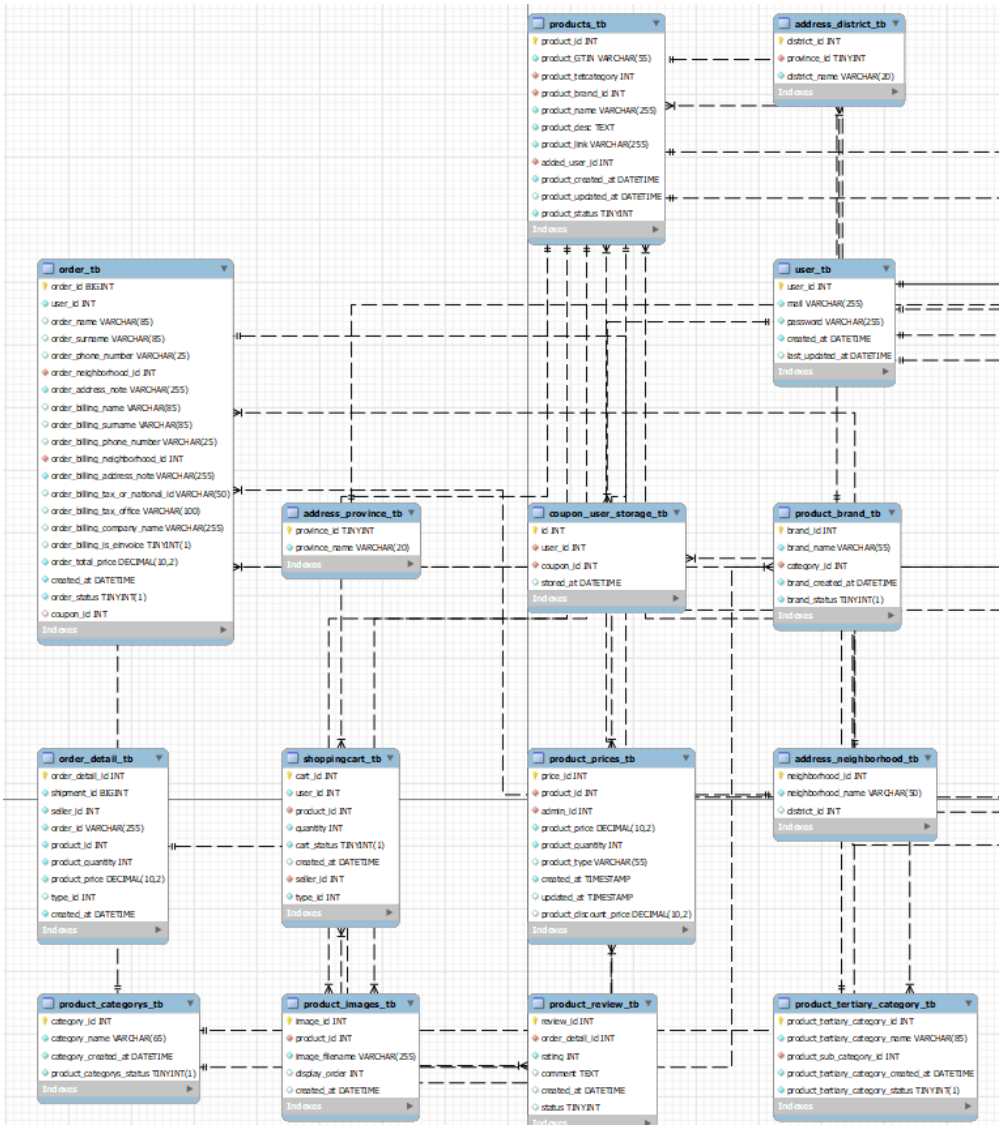
Bu sayede veritabanı işlemleri daha düzenli ve yönetilebilir hale gelmiş oldu.

Örnek olarak şöyle bir öngösterim yapabilirim:

```
$stmt = $pdo->prepare('CALL
favorites_show(:user_id)');
```

Bu kısım,PHP'den procedure çağırarak kullanıcının favorilerine eklediği ürünleri çekmesini sağlar. Burada prosedürün içindeki kod bloğunu buraya eklemiyorum; ancak devam eden kısımlarda neye benzediğinden bahsedeceğim.

c. İlişkili Veri Tabanı Ve Önemi



Bir veritabanı, farklı tabloların bir araya gelmesiyle oluşur. Tek bir tabloya her bilgiyi doldurmak yerine, ihtiyaç oldukça birden fazla tablo kullanılır ve bu tablolar birbirleriyle ilişkilendirilir. Böylece sistem daha düzenli çalışır ve ihtiyaç duyulan veriye kolayca ulaşılır.

Bunu bir örnekle anlatabiliriz: Bir öğrencinin çantasından kalemine ulaşmasını düşünelim. Öğrenci önce çantasına, sonra kalem kutusuna, oradan da kalemine uzanır. İlişkili veritabanı da aynı şekilde çalışır; veriye doğrudan değil, ilişkili tablolar üzerinden düzenli bir şekilde ulaşılır.

Ben de bu projede aynı mantığı kullandım. Örneğin bir kullanıcının adres bilgisine ulaşmak için, ayrı bir adres tablosu oluşturdum ve bunu kullanıcı tablosu ile ilişkilendirdim. Böylece kullanıcıya ait adres bilgisine, kullanıcı tablosu üzerinden düzenli ve doğru bir şekilde erişmek mümkün oldu.

İlişkili veritabanı, bilgilerin tek bir yerde üst üste yığılmasını engeller. Tablolar arasında kurulan bağlantılar sayesinde veriler hem daha düzenli saklanır hem de ihtiyaç duyulduğunda doğru ve hızlı bir şekilde bulunabilir. Bu yaklaşım, sistemin hem okunabilirliğini hem de sürdürülebilirliğini artırır.

4. Bilgi Çekme Ve İşlemi

a. PHP Action (İşlem Dosyası)

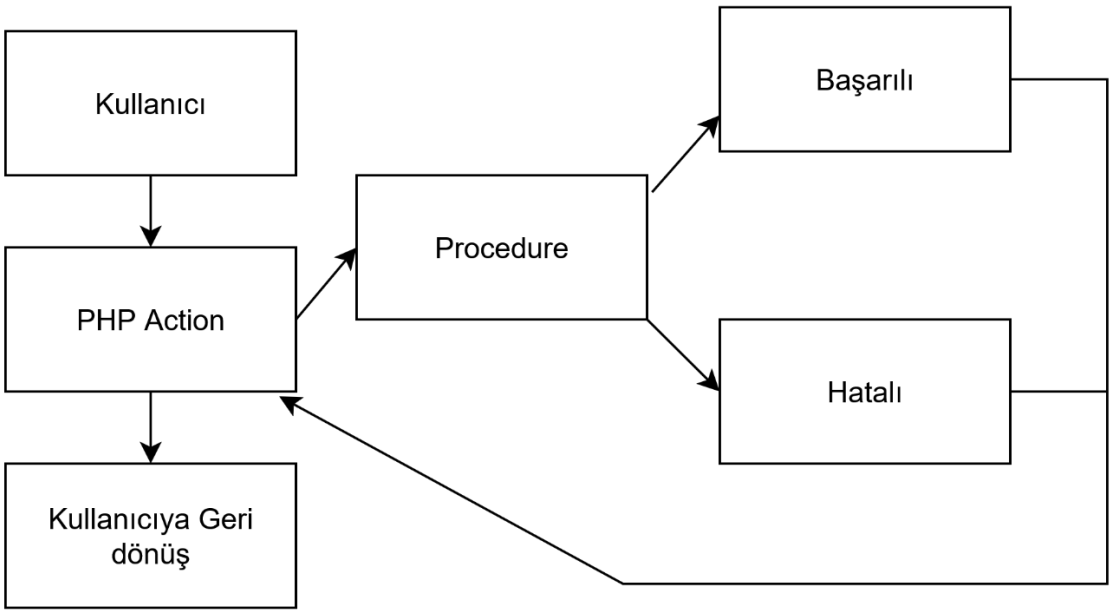
Kullanıcı site üzerinde bir işlem yaptığında (örneğin sepete ürün ekleme, kupon uygulama, adres kaydetme), bu istek doğrudan ilgili PHP action dosyasına gider. Action dosyası, kullanıcının gönderdiği veriyi alır, gerekli kontrol ve doğrulamaları yapar, ardından procedure çağırarak veritabanı ile iletişim kurar.Örneğin:

Kullanıcı “Sepete Ekle” butonuna bastığında, add_to_cart.php action dosyası çalışır.

Bu dosya, ürün bilgilerini alır ve ilgili procedure üzerinden veritabanına kaydeder.

Sonuç (başarılı/başarısız mesajı veya güncel sepet bilgisi) tekrar kullanıcıya gönderilir.

Bu yöntem sayesinde veritabanı işlemleri düzenli, güvenli ve tekrar kullanılabilir hale gelir. Action dosyaları, kullanıcı ile veritabanı arasındaki **köprü** görevini üstlenir.



b. JavaScript ile PHP Action'ın tetiklenmesi

Kullanıcı bir butona bastığında ya da bir form doldurduğunda, bu olay JavaScript tarafından yakalanır. JavaScript, **sayfanın yenilenmesine** gerek kalmadan ilgili dosyaya istek gönderir.

Bu işlem için JavaScript'in **fetch yapısı** kullanılır. Yani kullanıcıdan alınan bilgiler (örneğin ürün ID'si veya kupon kodu), fetch ile PHP action dosyasına gönderilir.Böylece:

Kullanıcı "Sepete Ekle" butonuna tıklayınca, JavaScript bu olayı algılar.

fetch **methodu** yardımıyla `add_to_cart.php` dosyasına gerekli bilgiler gönderilir.

PHP action dosyası, procedure çağırarak veritabanında işlemi yapar.

Sonuç tekrar JavaScript'e döner ve kullanıcıya ekranda "Ürün sepete eklendi" mesajı gösterilir.

Özetle: JavaScript, fetch üzerinden PHP action dosyalarını tetikler; bu da kullanıcı ile veritabanı arasında hızlı ve sorunsuz iletişim kurulmasını sağlar.

c. PHP Include

PHP'de include yapısı, bir projedeki ortak dosyaların tekrar tekrar yazılmadan farklı sayfalara eklenmesini sağlar. Örneğin veritabanı bağlantısı, üst menü (header) ya da alt kısım (footer) gibi her sayfada ortak kullanılan bölümler, tek bir dosyada tutulur. Daha sonra bu dosyalar ihtiyaç duyulan her sayfaya include edilerek projede düzen, bütünlük ve kolay yönetim sağlanır.

Bu yaklaşım sayesinde kod tekrarından kaçınılır, bakım süreci kolaylaşır ve projeye yeni bir parça eklendiğinde tek bir yerden değişiklik yapmak yeterli olur. **Örnek olarak:**

```php
<?php
// Sayfaya özel CSS ve JS dosyaları
$page_css = ["favorites.css", "product-preview.css"];
$page_js  = ["scripts.js"];

// Ortak üst kısım (header) ve kategori menüsü
include 'header.php';
include 'tools/action/categorynav.php';
?>

<!-- Sayfa içeriği buraya gelecek -->

<?php
// Ortak alt kısım (footer)
include 'footer.php';
?>
```

Sayfa İskeleti (Header / İçerik / Footer):

Bu yapı, hazırladığım her sayfanın ana şablonunu oluşturur.

header.php sayfanın üst bölümünü (logo, menü, ortak stiller) temsil eder.

footer.php alt bölümü (dip notlar, script yüklemeleri, telif vb.) kapsar.

Kısacası, sayfa bir sandviç gibidir: Üstte header, altta footer, ortada ise o sayfaya özgü içerik yer alır. Böylece üst ve alt kısımlara dokunmadan, yalnızca orta bölümü değiştirerek yeni sayfalar hazırlayabilirim.

Varlık yönetimi (CSS/JS):

page_css ve page_js dizileri, o sayfada kullanılacak stil(css) ve js dosyalarını tanımlar.

Bu dosyalar, CSS için header.php içinde; JS için footer.php içinde, önceden eklediğim küçük nüanslar sayesinde otomatik olarak sayfaya dahil edilir.

İhtiyaca göre bu listeleri kolayca düzenleyebilir, ekleyip çıkarabilir ve her sayfayı yalnızca gerektirdiği kaynaklarla çalıştırırım.

Bu şablon yaklaşımı, kod tekrarını azaltır, bakımı kolaylaştırır ve tüm sitede tutarlı bir görünüm sağlar.

5. Kullanıcı Rolleri ve Görevleri

a. Müşteri

Müşteri, sitenin en temel rolünü üstlenir. Görevleri ürünleri görüntülemek ve kendi sepetini yönetmek ile sınırlıdır.

Siteye üye olmadan da ürünleri farklı kategoriler altında inceleyebilir, açıklamalarını ve fiyatlarını görebilir.

Beğendiği ürünleri sepetine eklemek, çıkarmak veya adetlerini değiştirmek için üye olarak giriş yapması gerekir.

Sepetini düzenledikten sonra sipariş sürecine geçerek alışverişini tamamlayabilir.

Ayrıca kupon kullanma ya da favorilere ürün ekleme gibi ek işlevler de yine kendi hesabı üzerinden mümkündür.

Kısacası müşteri, siteyi ürünleri incelemek için özgürce kullanabilir; ancak alışveriş ve kişisel işlemlerini yönetebilmek için kendi hesabına ihtiyaç duyar.

b. Satıcı/Yönetici

Satıcı veya yönetici rolü, sitenin işleyişinde müşteriden daha aktif bir görev üstlenir.

Ürün Ekleme: Satıcı, sisteme yeni ürün eklerken katalog bilgilerini girer ve o ürüne fiyat ataması yapar.

Sipariş Takibi: Müşterilerin verdiği siparişleri görüntüleyebilir ve kargo süreçlerini yönetebilir.

Stok Kontrolü: Ürünlerin mevcut stok bilgilerini güncel tutarak müşterilerin doğru bilgiye ulaşmasını sağlar.

Müşteri İletişimi: Gerekli durumlarda müşterilerle iletişim kurarak sipariş sürecine destek verebilir.

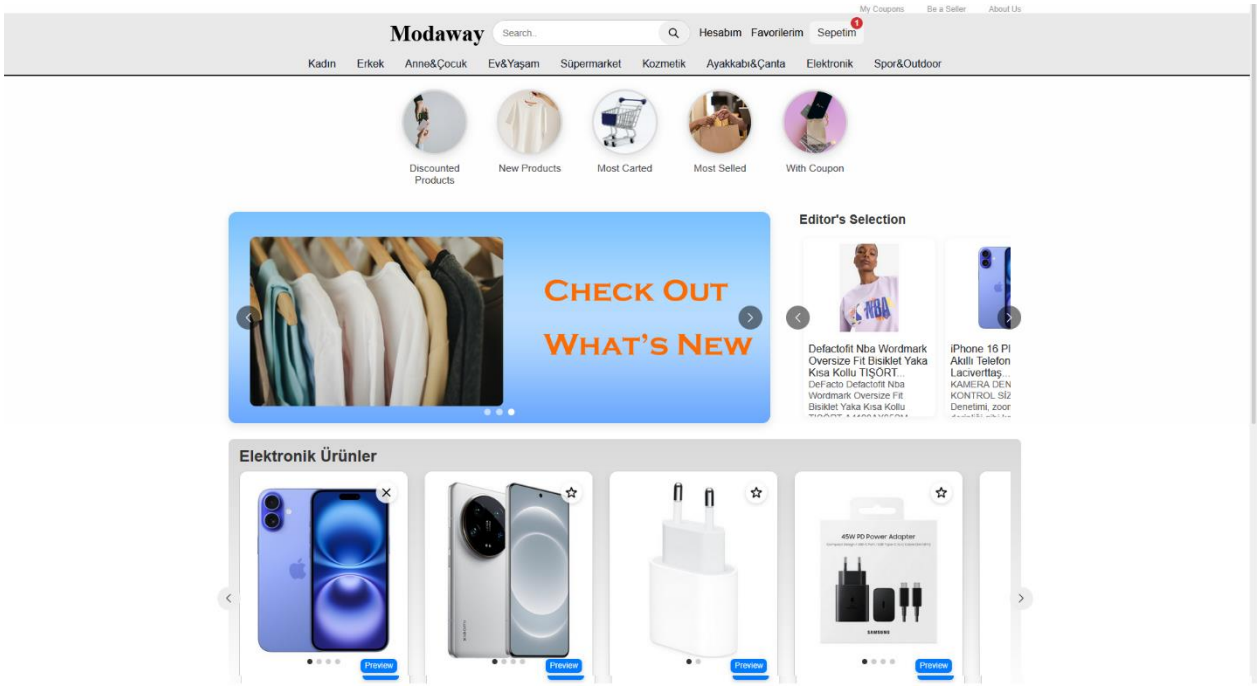
Bu projede satıcı ve yönetici ayrı roller olabilecekken, daha anlaşılır ve basit olması için tek bir başlık altında birleştirdim.

c. Misafir

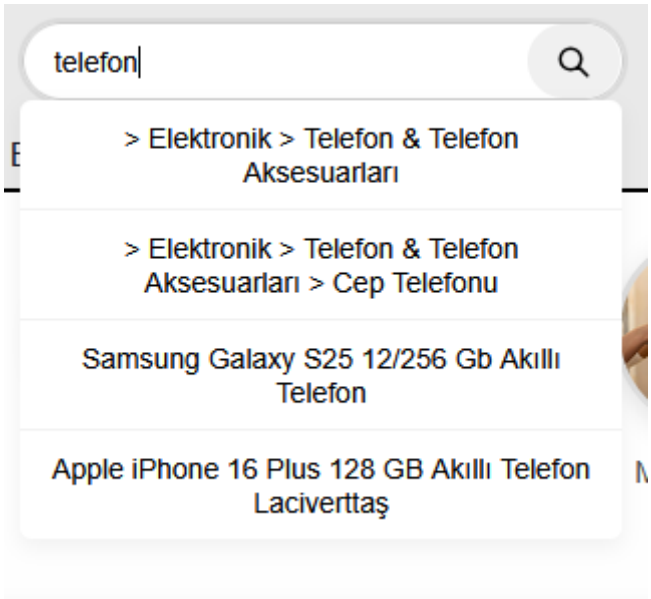
Misafir kullanıcı, siteye üye olmadan yalnızca ürünleri görüntüleyebilir. Sepet ekleme veya sipariş verme gibi işlemleri yapamaz.

6. Ekranlar

a. Anasayfa



Sayfanın en üst kısmında bizi ilk olarak arama kutusu karşılar. Bu kutuya yazılan terim; ürün adı, ürün kodu ya da kategori adı olabilir. Kullanıcı bir terim girdiğinde sistem, alttaki örnekte görüldüğü gibi en yakın sonuçları anında listeler. Eğer arama butonuna basılırsa, kullanıcı ürün arama sayfasına yönlendirilir ve yazılan terimle ilgili tüm sonuçlar detaylı olarak gösterilir. (Bu kısmı, arama sayfasını anlatırken daha ayrıntılı açıklayacağım.)

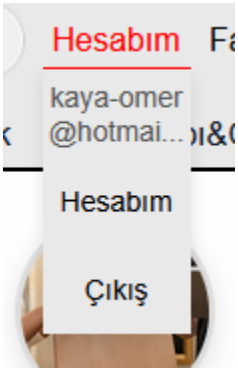


Arama şu mantıkla çalışır:
Kullanıcı bir cümle yazdığında (örneğin: **“Samsung S25 Telefon”**) metin boşluklara göre kelimelere ayrılır. Her kelime için sırayla arama yapılır:

- İlk olarak **“Samsung”** kelimesi geçen tüm kayıtlar bulunur.
- Daha sonra bu bulunan sonuçların içinde **“S25”** kelimesi aranır.
- Son olarak kalan sonuçlar arasında **“Telefon”** kelimesi aranır.

Böylece her adımda sonuçlar daraltılır ve sonunda yalnızca **tüm kelimeleri içeren** ürünler veya kategoriler kullanıcıya gösterilir.

Bu yöntem sayesinde tek kelimeyle yapılan aramaya göre daha **kesin ve alakalı sonuçlar** elde edilir.

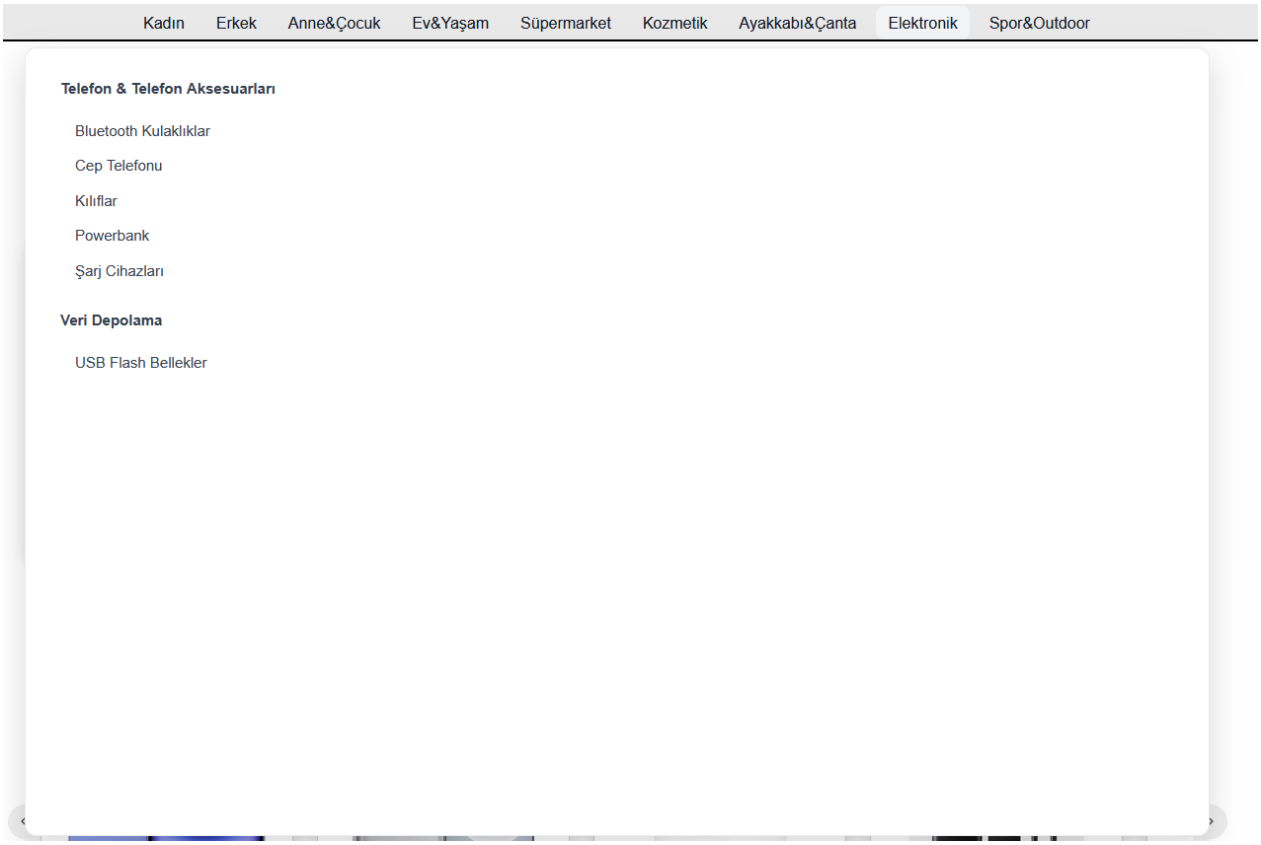


Arama kutusunun yanında yer alan Hesabım düğmesi kullanıcıyı kendi panel sayfasına götürür. Hemen yanında bulunan Favorilerim sekmesi, kullanıcının daha önce favoriye eklediği ürünlerin listesine kolayca ulaşmasını sağlar. Sepetim düğmesi ise doğrudan kullanıcının alışveriş sepetine yönlendirerek eklenen ürünleri görüntüleme ve düzenleme imkânı sunar.

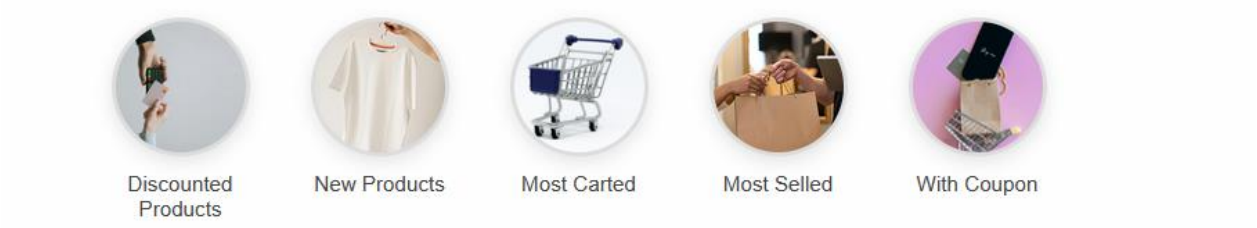
Kartlarda bulunun yıldız simgesiyle hızlı ve anlaşılır bir biçimde favoriye ekleme işlemi gerçekleştirilebilir.

Arama kutusunun hemen altında, kullanıcıyı kategori menüsü karşılar. Buradaki kategorilerden birine tıklanırsa, ilgili kategoriye ait ürünler listelenerek ürün arama sayfası açılır.

Eğer kullanıcı fareyi herhangi bir kategori üzerine getirirse, JavaScript yardımıyla alt kategorilerin bulunduğu bir açılır menü (pop-up) gösterilir. Böylece kullanıcı, ilgilendiği kategoriye daha detaylı şekilde erişebilir ve aramasını daha hızlı daraltabilir.



Kategorilerin alt kısmında özel arama seçenekleri yer alıyor. İlgili tüm ürünler ürün arama sayfasında listeleniyor.

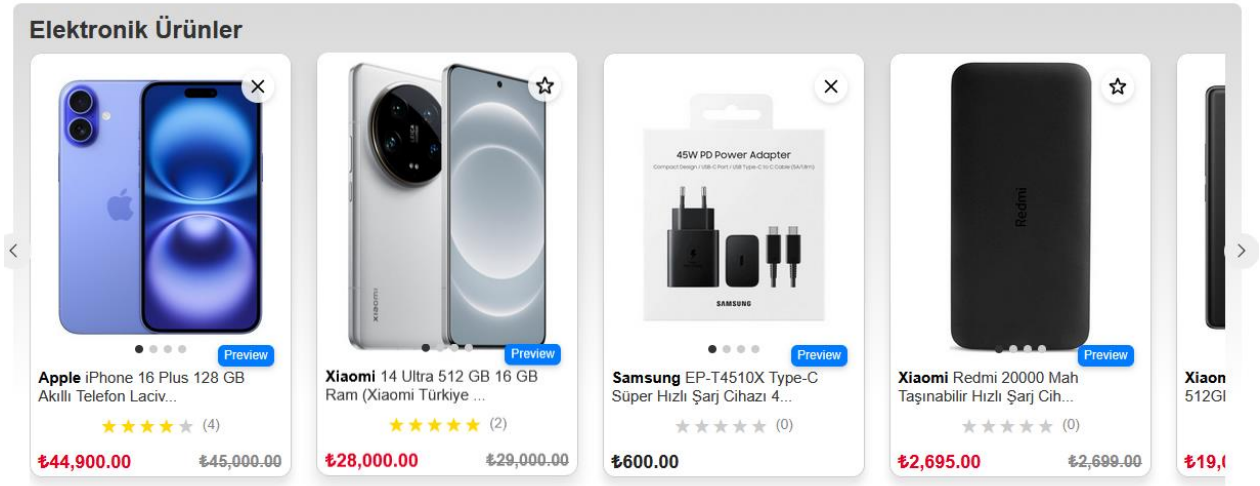


Kategori menüsünün hemen altında bizi bir slider (kayan görsel alanı) karşılar. Bu alandaki görsellere tıklandığında, ilgili kategoriye ait ürünler ürün arama ekranında listelenir.

Slider'ın sağ tarafında ise Editörün Seçtikleri bölümü yer alır. Burada, öne çıkarılmak istenen veya dikkat çekici bulunan ürünler kullanıcıya sunulur. Bu sayede ziyaretçiler yalnızca kategori bazlı değil, aynı zamanda editör önerileri üzerinden de ürün keşfedebilir.

Seçilen kategoriye ait ürünler, sayfanın devamında kullanıcıyı karşılar. Bu bölümde her ürün, ayrı bir dosya üzerinden çağrılan ürün kartı yapısıyla (product-cart.php) gösterilmektedir. Aynı kart yapısını sitenin farklı bölümlerinde de tekrar tekrar kullanıyorum.

Ürün kartının özellikleri:



Kartın üst kısmında, ürünü favorilere ekleme veya çıkarma işlevi gören bir buton bulunur. Kullanıcı bu butona tıkladığında ürün favorilere eklenir ya da listeden kaldırılır.

Ürün resminin üzerine fare ile gelindiğinde, eğer farklı görselleri varsa bunlar sırayla gösterilir. Görseller alanı dört parçaya bölünmüştür; fare hangi bölgeye gelirse o bölgeye ait resim görünür. Bu özellik maksimum 4 görselle sınırlandırılmıştır.

Kartın üzerinde yer alan Preview (ön izleme) butonu, ürünü hızlıca bir pop-up penceresinde açarak sepete ekleme imkânı sunar.

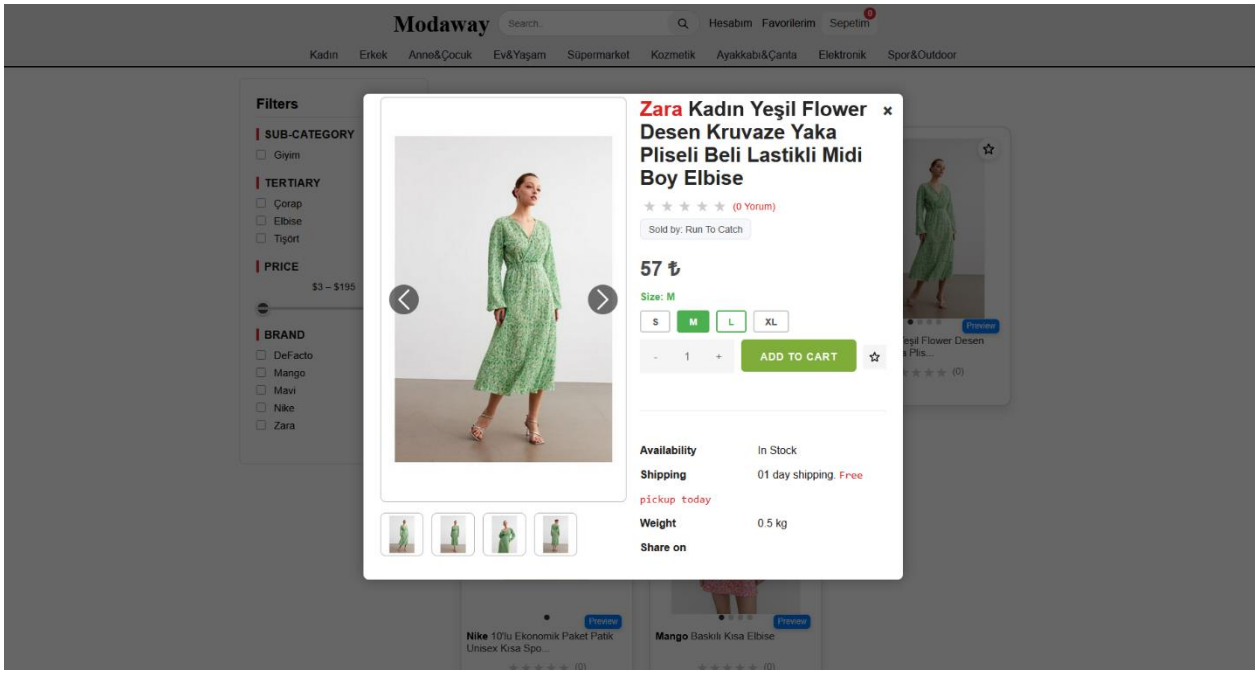
Kartın alt kısmında ürünün puanı görüntülenir. Ayrıca katalog fiyatı uygulaması kullanıldığı için, ürünün her zaman en ucuz fiyatı gösterilir. Ürün indirimdeyse, indirim bilgisi de burada belirtilir.

Bu yapı sayesinde ürün kartı hem detaylı bilgi sunar hem de kullanıcıya hızlı etkileşim imkânı verir.

Preview (Ön İzleme) butonu ile hızlı sepete ekleme işlemi sağlamayı amaçladım. Alttaki resimde bunun kullanıcıya nasıl görselleştirildiğini görebilirsiniz.

Preview butonu yerine farklı bir yol izleyip kart yapısına eklemeyi düşündüm; fakat beden seçme işlemi gerektiğinde bu yapı kafa karıştırıcı, küçük ve anlaşılmaz bir hâl aldı. Bu yüzden en sonunda böyle bir yapı kurmaya karar verdim.

Burada ürün detay sayfasının küçük bir modeli gibi çalışacak şekilde tasarımı tamamladım. Eğer ürünün bedeni varsa, beden seçilebilir; adet artırılabilir ve sepete ekleme işlemi kolayca gerçekleştirilebilir.

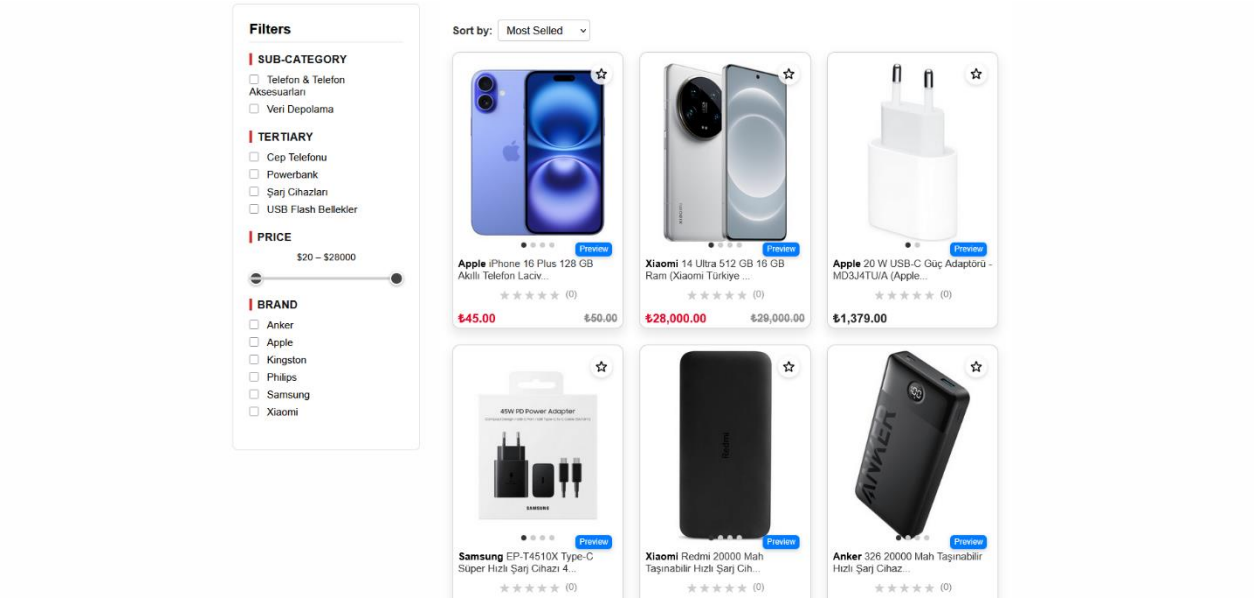


b. Ürün Arama

```
http://localhost/mysqlecommerce/product-grid.php?category=8
```

yukardaki gibi bir örnekle başlayalım:

Elektronik kategorine bağlı olan ürünlerin gösterilmesi istediğinde kullanıcı şu görsel ile karşılaşıyor.



Sol tarafta ürünlerle alakalı filtreleme bulunmaktadır. Sağ tarafta ile daha önceden bahsettiğim gibi kart sistemini kullandım.üst kısımda da sıralama bulunmaktadır.

Arama sayfasında üç farklı Procedure den biri seçilerek devam ediliyor. Böyle yaparak daha anlaşılır bir yapıya dönüştü.

Bu arama şekileri örneklendirmek gerekirse

```
http://localhost/Mysqlecommerce/product-grid.php?q=test
```

Aranan kelimeye özel,

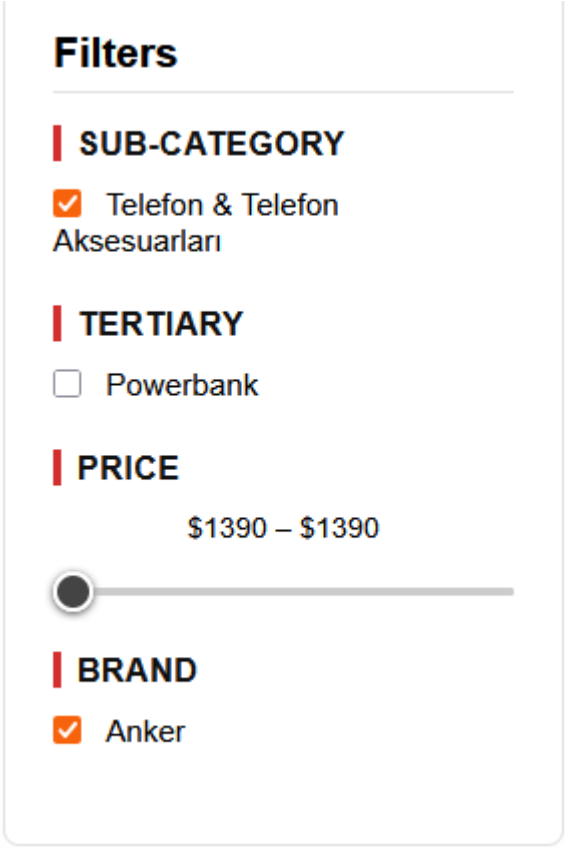
```
http://localhost/mysqlecommerce/product-grid.php?special=discounted
```

indirimde kuponlu veya yeni stoklara giren ürünler gibi arama seçmeleri,

```
http://localhost/mysqlecommerce/product-grid.php?category=8
```

sonuncusu ise kategori veya alt kategoriler yoluyla arama şeklidir.

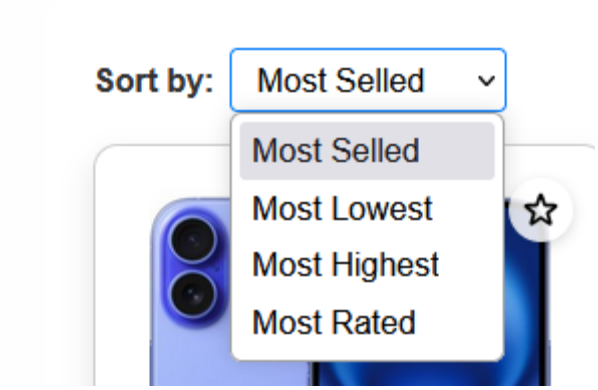
Kaldığımız yerden devam edelim: filtrelendire için js kullandım. Seçme alanlarına yada fiyat seçme butonları kullanıldığında aşağıdaki gibi seçim yapılarak istenilen listelere daha kolay erişme şansı sağladım.



Link kısmına bunları eklediği için kopyalayıp aynı yere ulaşma sağlanabiliyor.

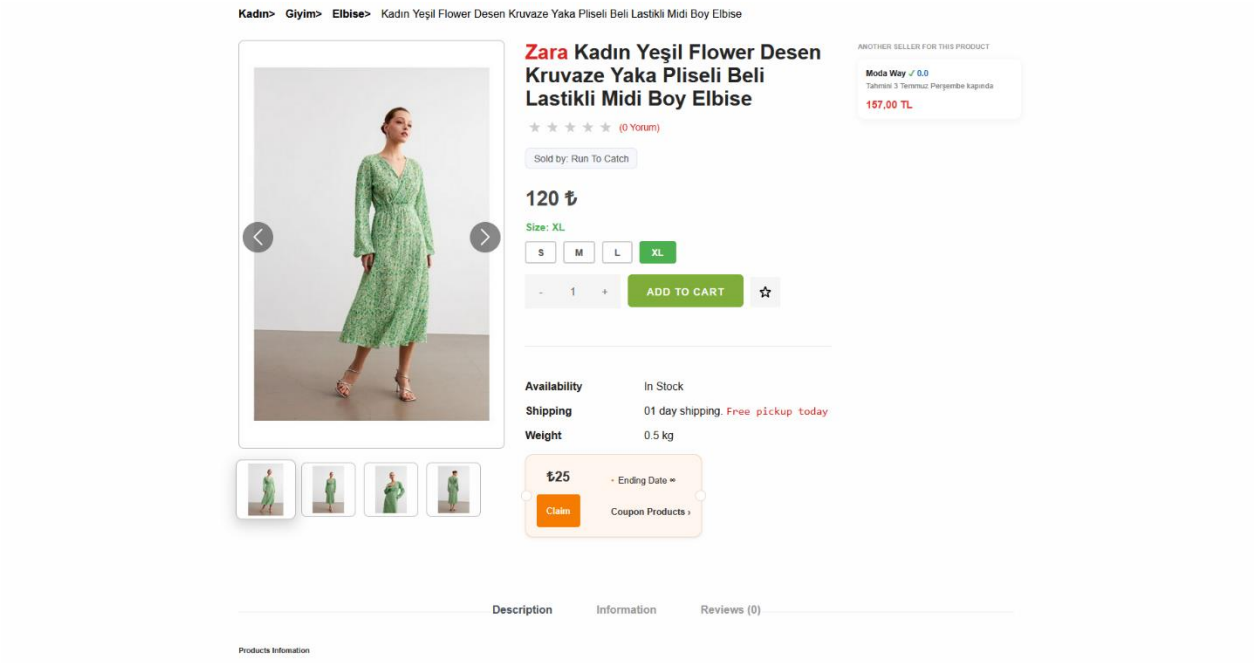
```
http://localhost/mysqlecommerce/product-grid.php?page=1&category=8&sub_category_ids[]=10&brand_ids[]=14&sort_by=most_selled
```

Sıralama ise bu seçimler le endeksli olmakla birlikte istenilen sıralama ile sonuçları getiriyor.



c. Ürün Detayı

Örnek bir sayfa olarak aşağı görsele göre anlatacağım.

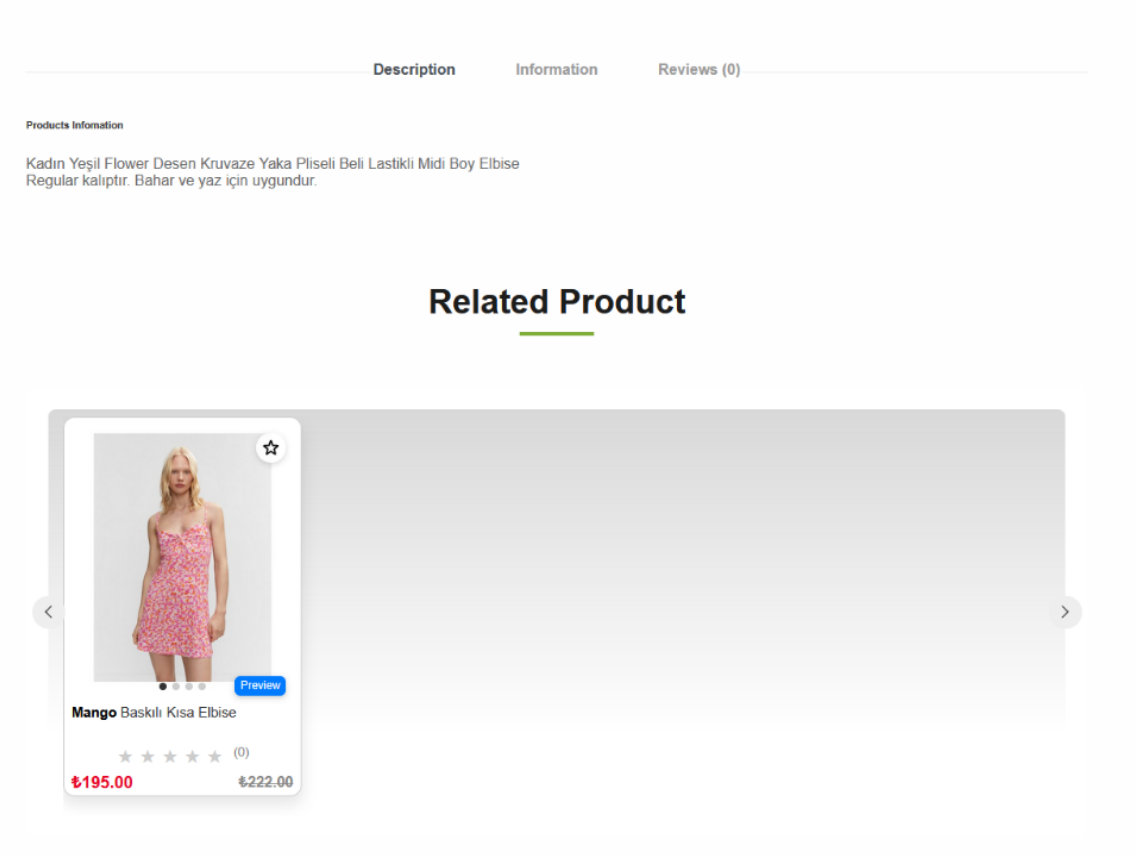


Karşımızda en ucuz seçili olarak geliyor.Kategori sistemi kullandığım için satıcıların adını görebilirsiniz.

Sağ kısımdaki diğer satıcılar kısmında daha pahalıya satan satıcılar gösterilmektedir. Tıklanarak o satıcı seçili olduğu sayfaya gitmektedir ve alım sağlanabilir.

Bedenler seçildiğinde de en ucuz satıcıyı seçmetedir. Sepete eklendiği zaman en üstteki sepetim sayısı artmakta ve sağ üst kısımda ürünün sepete eklendiğini belirten bir yazı bizi karşılıyor.al

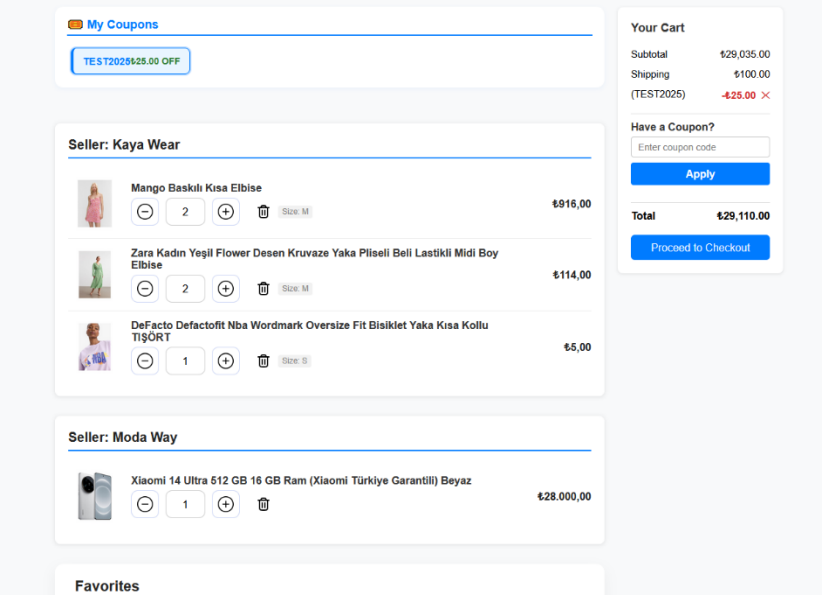
Sepete ekle butonun altında şayet kullanılabilir kategori kuponu varsa o gösteriliyor. Kazan a basılırsa kullanıcıya ait olan bir table da saklanıyor. Bu kısımda kullanıcı sepetinde daha anlaşılır olacak. O sayfada eklenmiş kuponlar şayet kullanılabiliyor listelenecektir.



Sayfanın devamında ürün detayı yorumlarının görüntülenebileceği ve alakalı farklı ürünlerin listelendiği bir alan bizi karşılıyor.

Kullanıcı alım işleminde sonra yorum yazabiliyor. Bu kısım kullanıcı panelinde yapılabiliyor.

d. Kullanıcı Sepeti



Bu sayfanın amacı, **sepete eklenen ürünleri kontrol etmek ve düzenlemek** içindir ve neredeyse tüm e-ticaret sitelerinde bulunur.

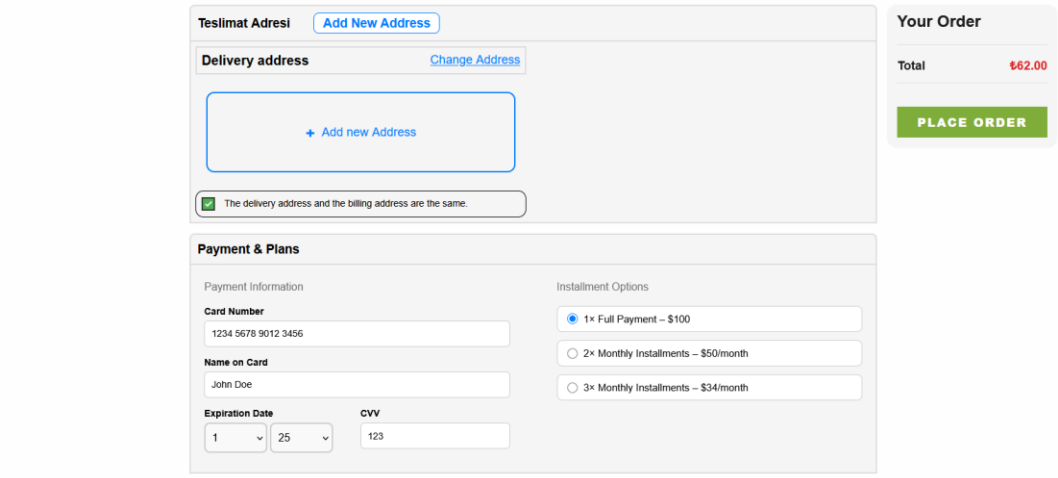
Üst kısımda **kupon deponuz** yer alır. Kuponlar, ürün detay sayfasındaki *"Kupon kazan"(claim)* alanından elde edilir. Burada **eklenebilir kuponlar görünür** ve bir kupona tıkladığınızda sistem otomatik olarak kuponu sepete uygular. Ayrıca sağ tarafta, **uygulanabilir kuponları** (eklenen ürünlerle ilişkili olanları) sepete manuel olarak ekleyebilirsiniz.

Orta kısımda, ürünler **satıcılara göre gruplanmış** şekilde listelenir. Böylece hangi satıcıdan ne kadar ürün aldığınız görünür ve buna göre **kargo ücreti** hesaplanır. Ürünlerin yanında bulunan "–" ve "+" simgeleri adet değiştirmenize, çöp kutusu simgesi ise ürünü tamamen kaldırmanıza yarar.

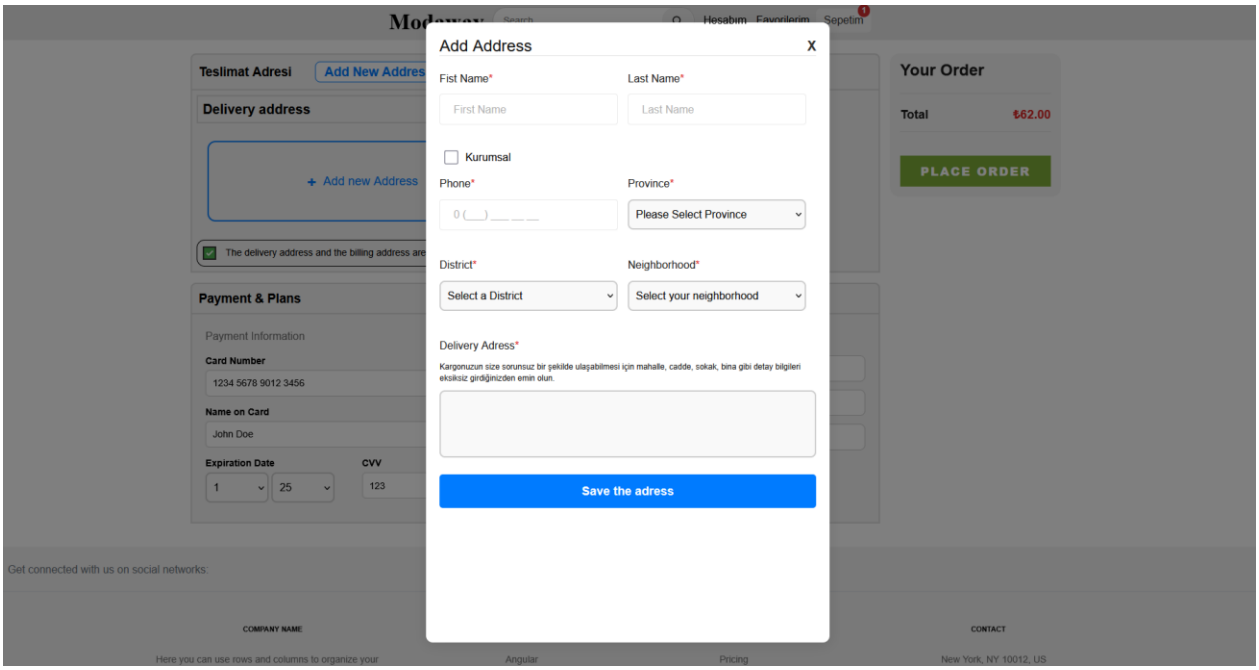
Alt kısımda, **favorileriniz** gösterilir. Burada, son kez göz atmanız ve unuttuğunuz bir ürünü hatırlamanız için küçük bir hatırlatıcı işlevi görür.

Son olarak, **siparişi tamamlama sayfasına gitmek için** sağ tarafta bulunan *Proceed to Checkout* butonuna tıklayabilirsiniz.

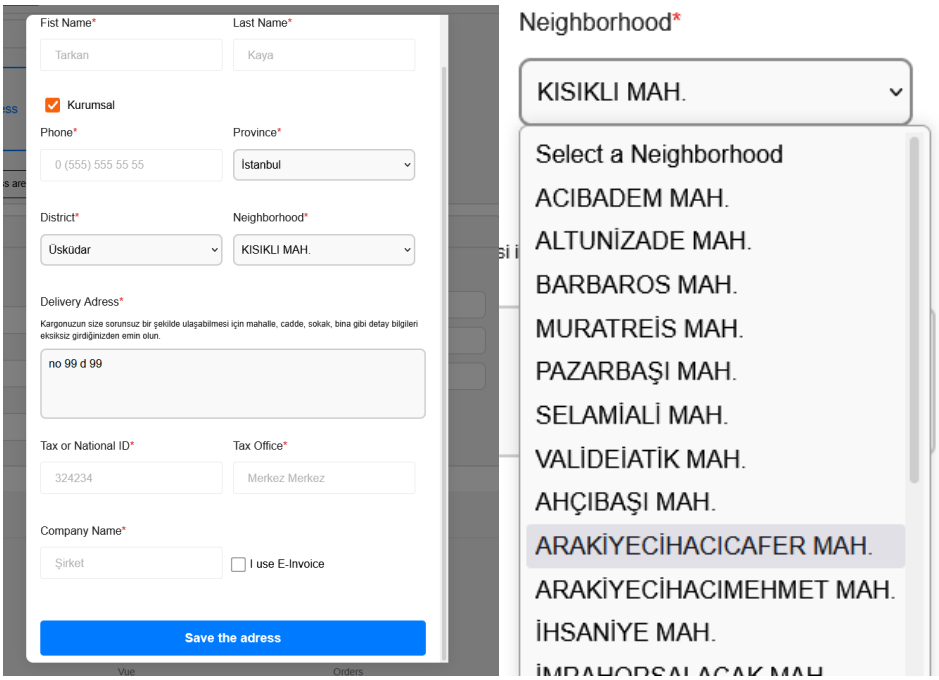
e. Ödeme Sayfası



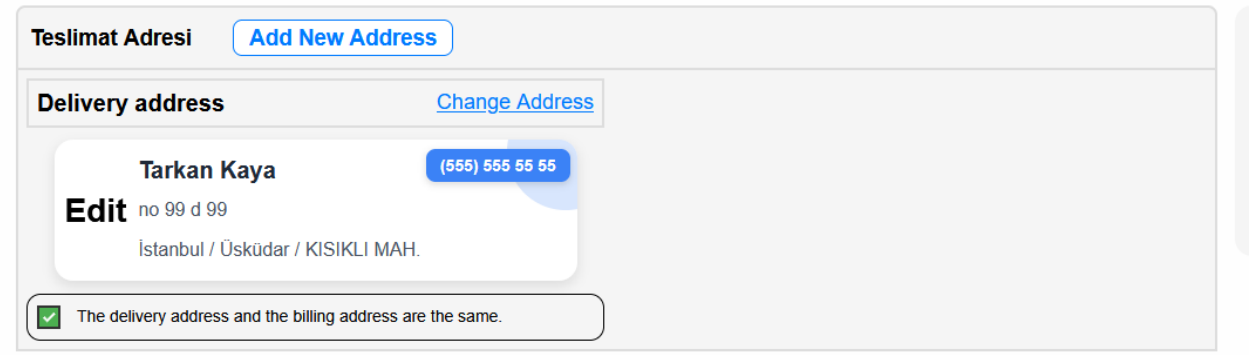
Yeni bir kullanıcı ödeme ekranında ilk olarak yeni bir adres eklemeli bu yüzden görünür olmasını için böyle bir düzen kulllandım.



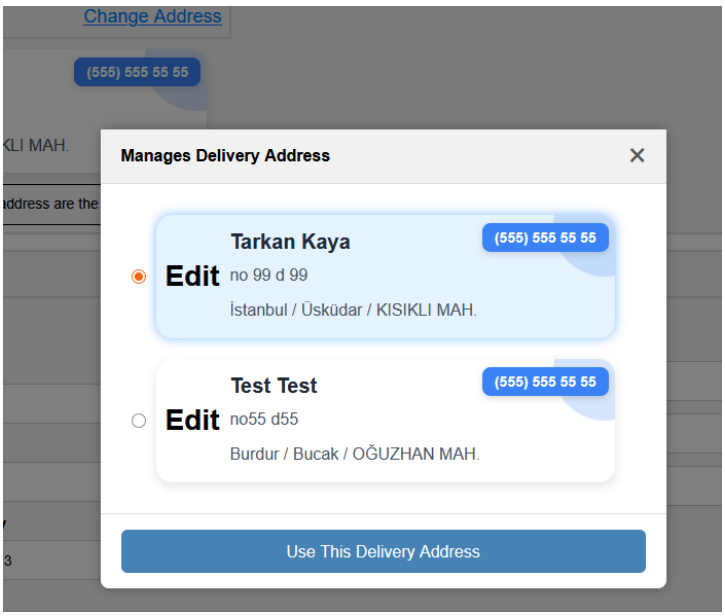
Yeni adres ekleme penceresinde adı soyadı ekledikten sonra kurumsal sa kurumsal butonuna basılarak açılan ek alanlar ortaya çıkar ve kullancı o alanlarıda doldurabilir. Şehir, ilçe ve Mahalle bilgilerini js yardımıyla dinamik bir şekilde eklenmesini sağladım. Türkiye adres bilgilerini ise internette buldum ve refine bir şekilde eklemeyi başardım.

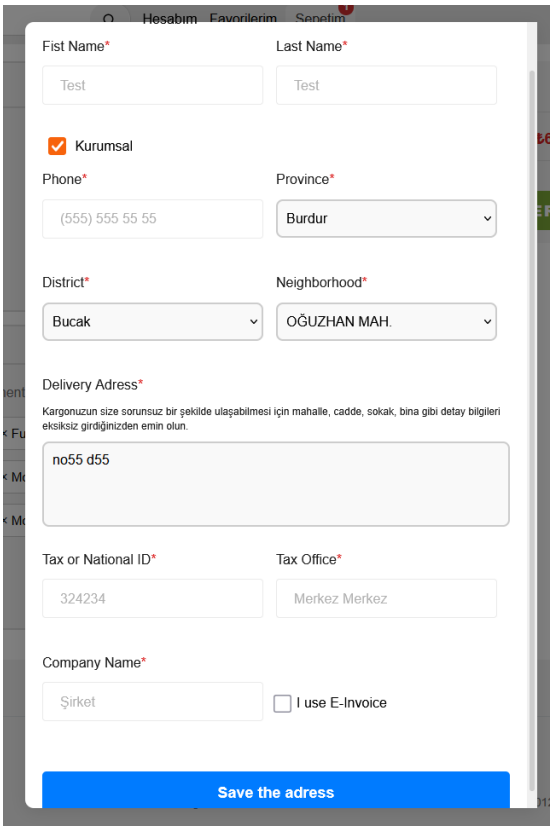


Kayıt ettikten sonra bizi şöyle bir alanlar karşılıyor. Eğer bu kullanıcının ilk adresi ise **otomatik olarak** seçiliyor.

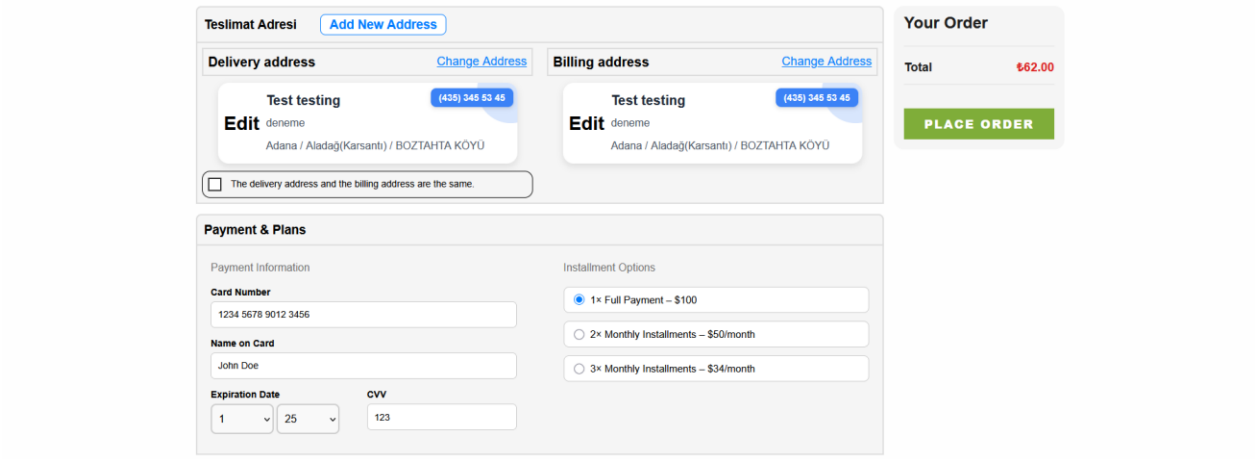


Dilersek alttaki buton yardımıyla fatura adresi seçebiliriz. Adres seçmek istersekte kutucuğun adres değiştir kısmıyla adresini değiştirebilir.





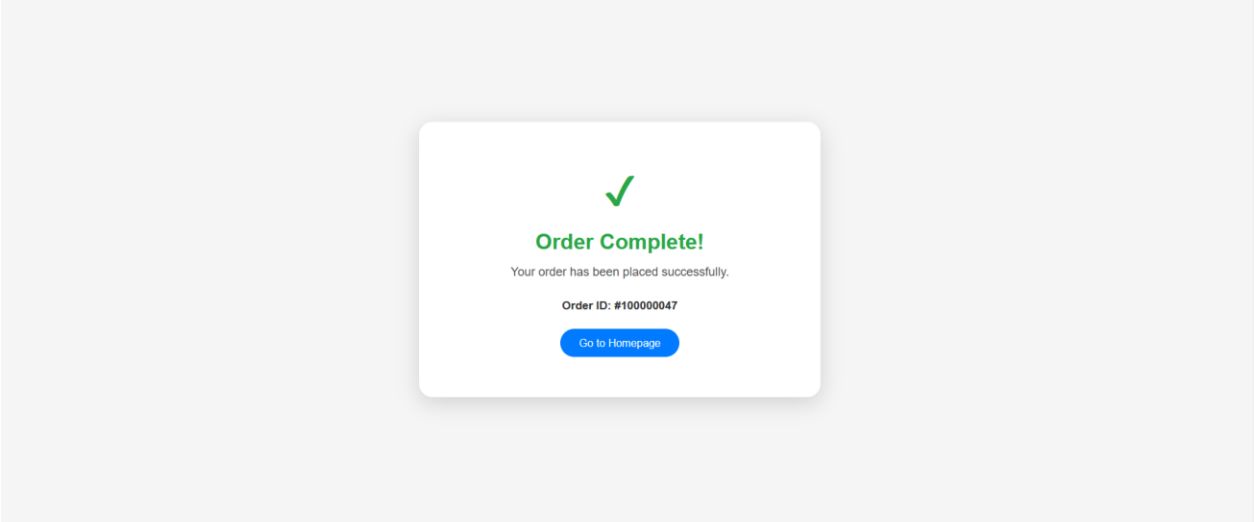
edit buttonuna basılırsa dolu bir şekilde pencere açılıp güncelleme gerçekleştirilebilir.



Özet olarak sayfa yüklendiğinde, PHP PDO sınıfı kullanılarak sayfa **kullanıcının kayıtlı adres bilgileriyle otomatik olarak doldurulur**. Yeni bir adres eklendiğinde “Change Address” butonları üzerinden seçimler arasında değişiklik sağlanabilir.

Herhangi bir değişiklik olduğunda, JavaScript aracılığıyla PHP PDO sınıfı çağrılır ve ilgili kart bölümü yenilenir. Böylece kullanıcıya güncel bilgiler anında yansıtılır. Js kullandığım için güncelleme kısmında sayfa yenilenmiyor. Buda estetik ve hız açısından gayet önemli.

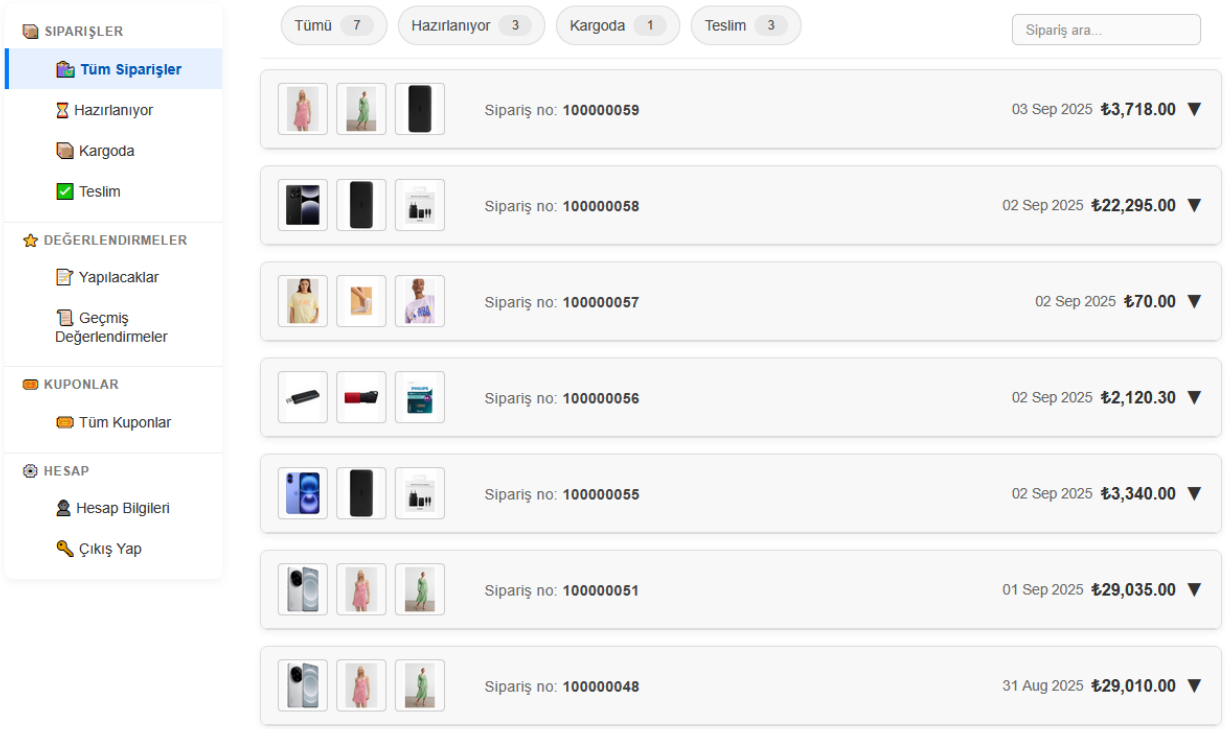
Seçilen veya değiştirilen tüm bilgiler doğrudan veritabanına işlendiği için, ekrana gelen her bilgi **doğru ve güncel** olur.



İşlem tamamlandıktan sonra kullanıcı, sipariş tamamlandı sayfasına yönlendirilir ve satın alma süreci başarıyla sona ermiş olur.

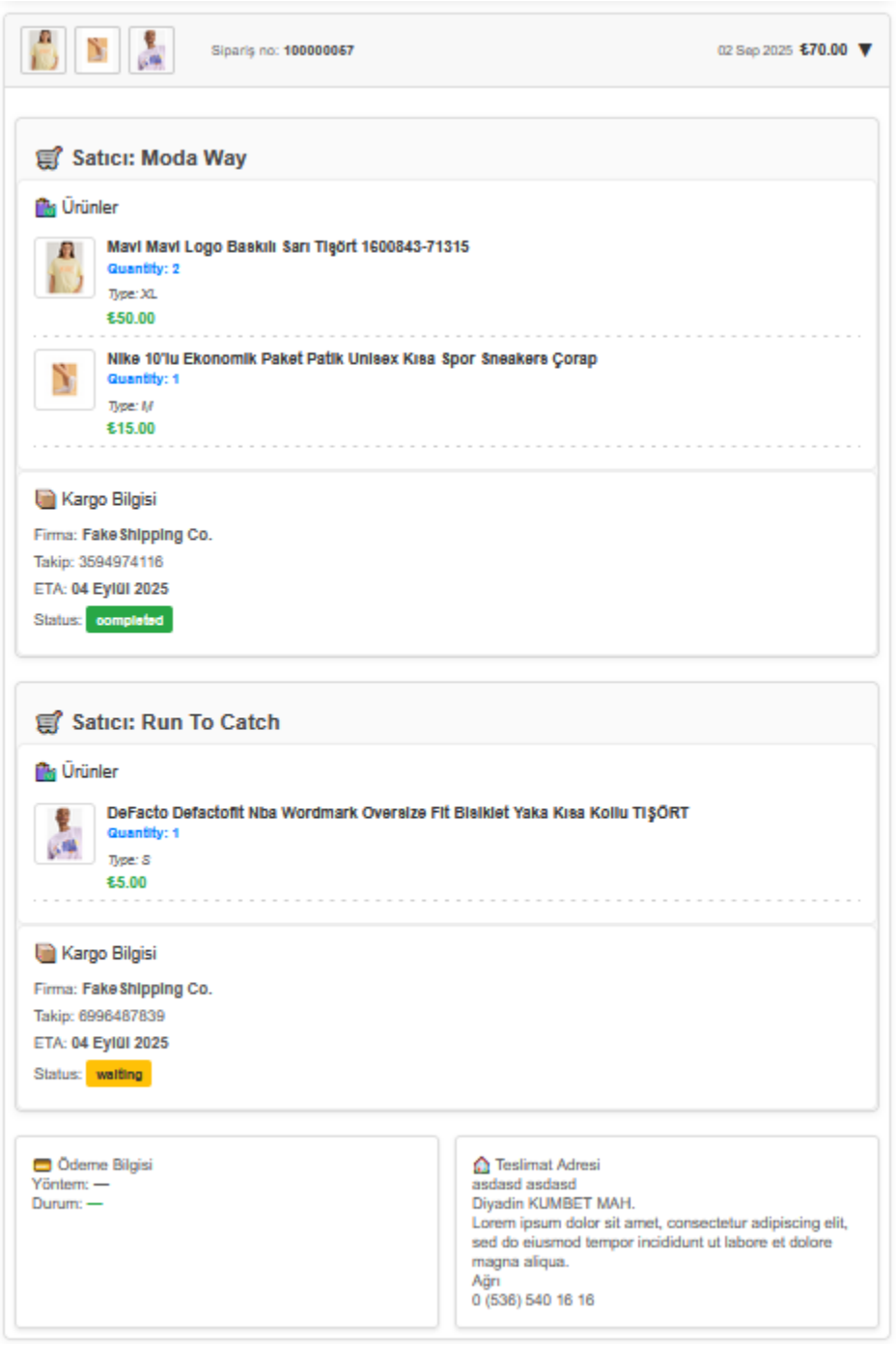
f. **Kullanıcı Paneli**

Kullanıcı paneli, alışveriş sonrasında kullanıcıya ait bilgileri içerir. Bu alanda; satın alınan ürünlerin kargo durumu, ürünler için yorum ve yıldızlı değerlendirme yapma imkânı ile kullanıcıya uygun kuponların görüntülendiği bölümler yer alır. Siparişlerden başlayalım:

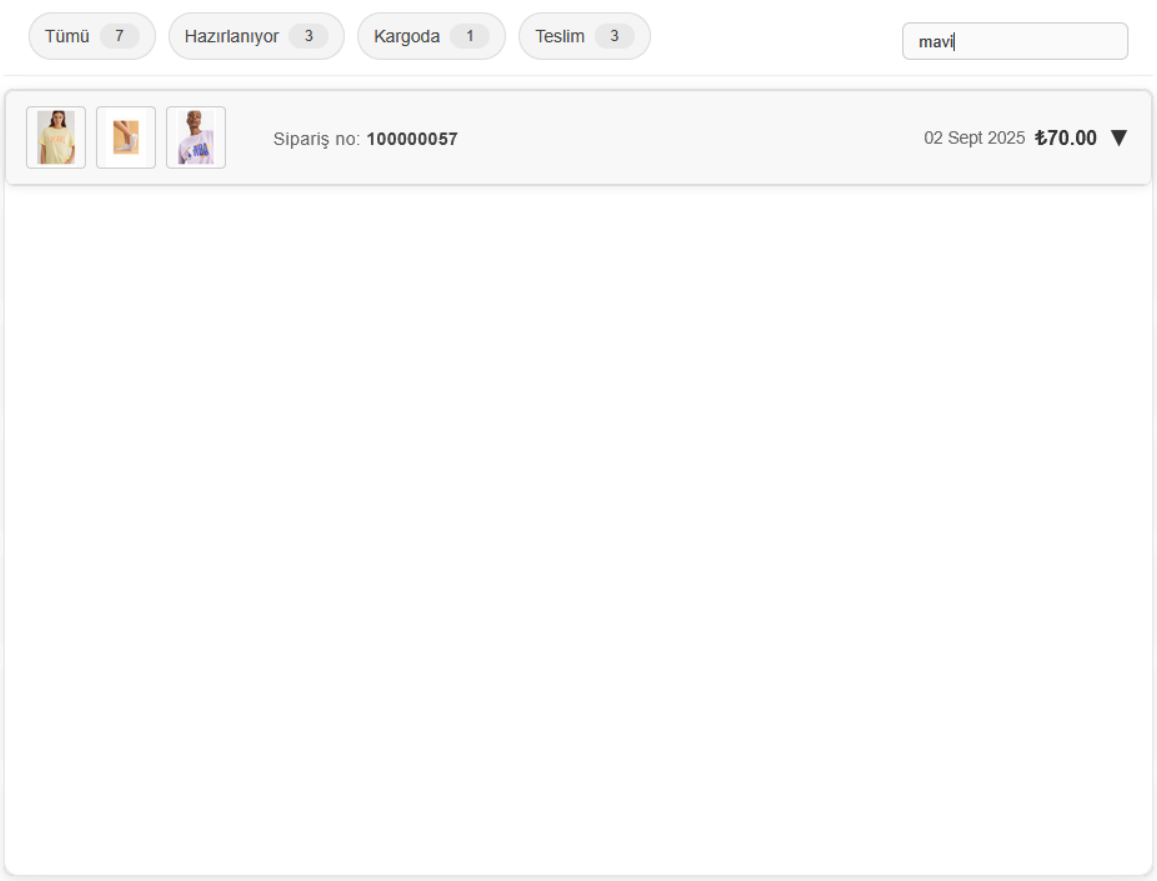


Arayüzde ilk olarak, verdiğimiz siparişlere göre sıralanmış bir alışveriş önizlemesi karşımıza çıkar. Önizlemede ilk üç ürünün resmi yer alır; ayrıca olası sorunlara karşı sipariş numarası, tarih ve ücret bilgileri de görüntülenir.

Tıklandığı zaman ilgili siparişin detayıyla karşılaşırız.



Katalog sistemi üzerinden çalıştığım için farklı kullanıcılar bulunuyor. Bu nedenle her satıcıya **özel** bir alanda sipariş bilgileri görüntüleniyor. Kullanıcı sipariş verdikten sonra ürünler satıcının ekranına düşüyor ve satıcı buradan siparişin durumunu güncelleyebiliyor. Bu işlemleri ayrıca satıcı kargo ekranında da göstereceğim. Sipariş verildiği anda o anki veriler çekildiği için, sonradan yapılan fiyat veya adres değişiklikleri siparişe **yansımıyor**. Bu tasarımı özellikle böyle planladım.

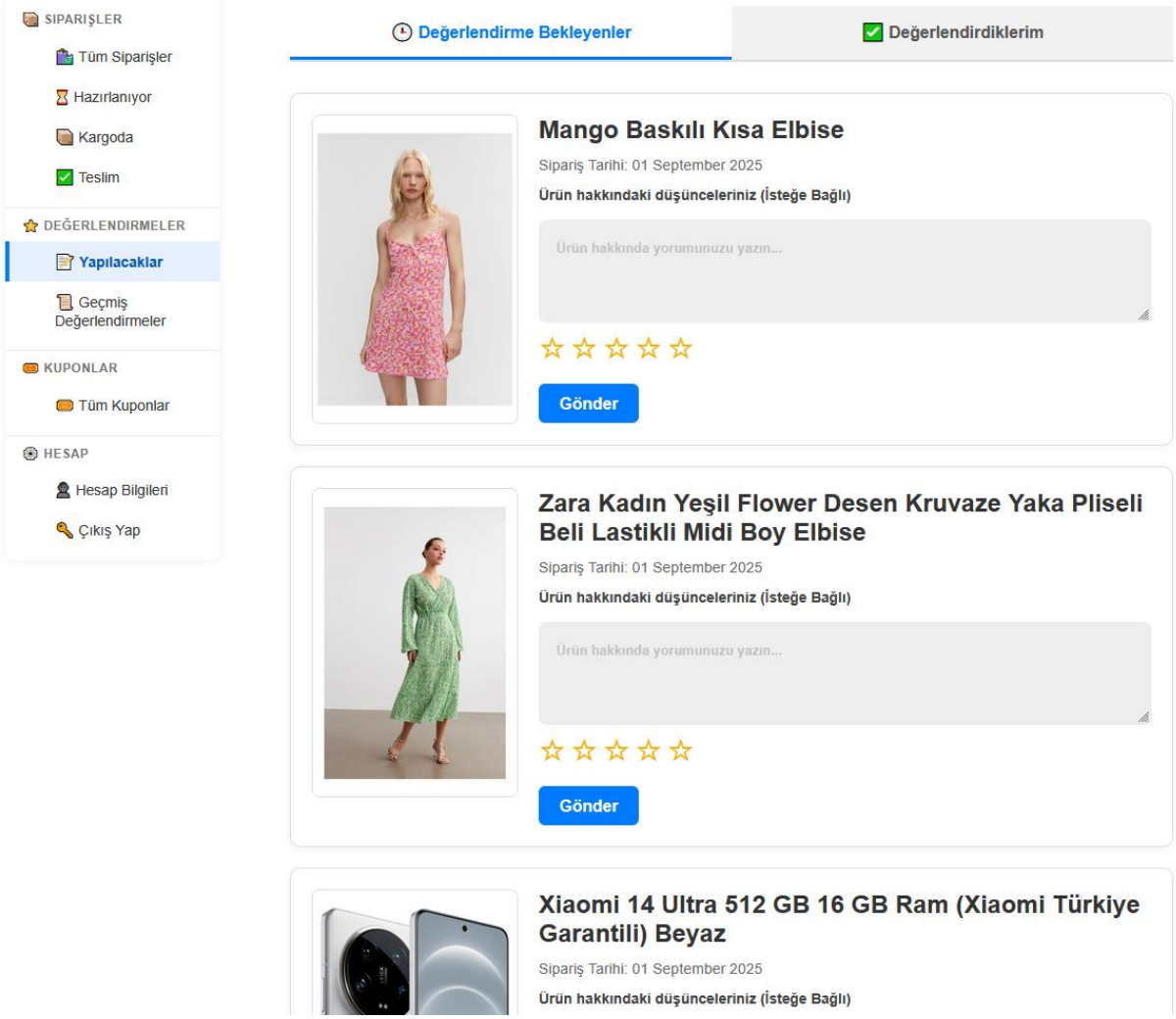


Sipariş arama kısmını da tasarladım. Bu bölümde kullanıcılar, sipariş numarası veya ürün adına göre geçmiş siparişlerini arayabiliyor. Bu işlevin **çalışmasın**ı JavaScript sağlıyor.

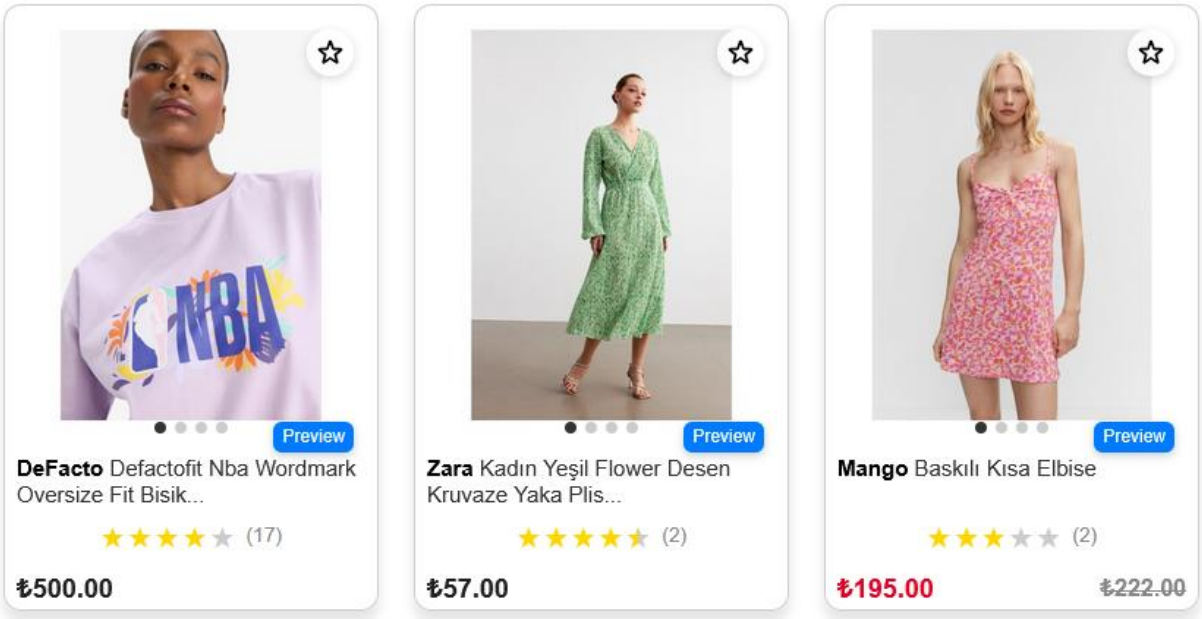
Kullanıcı siparişi verdikten sonra, sipariş ilk olarak “Bekliyor” durumunda oluyor. Satıcı, bu durumu “Kargolandı” veya “Tamamlandı” olarak güncelleyebiliyor. Bu akışı göstermek için toplamda dört farklı sayfa oluşturmam gerekti. Ancak her seferinde ayrı ayrı yazmak yerine, tek bir ana şablon yapısı tasarladım. Bu şablona boş değer gönderildiğinde tüm siparişler, “tamamlandı” değeri gönderildiğinde ise yalnızca tamamlanan siparişler listeleniyor. Örneğin:

```php
<?php
$status = 'completed';
include 'order-status-template.php';
?>
```

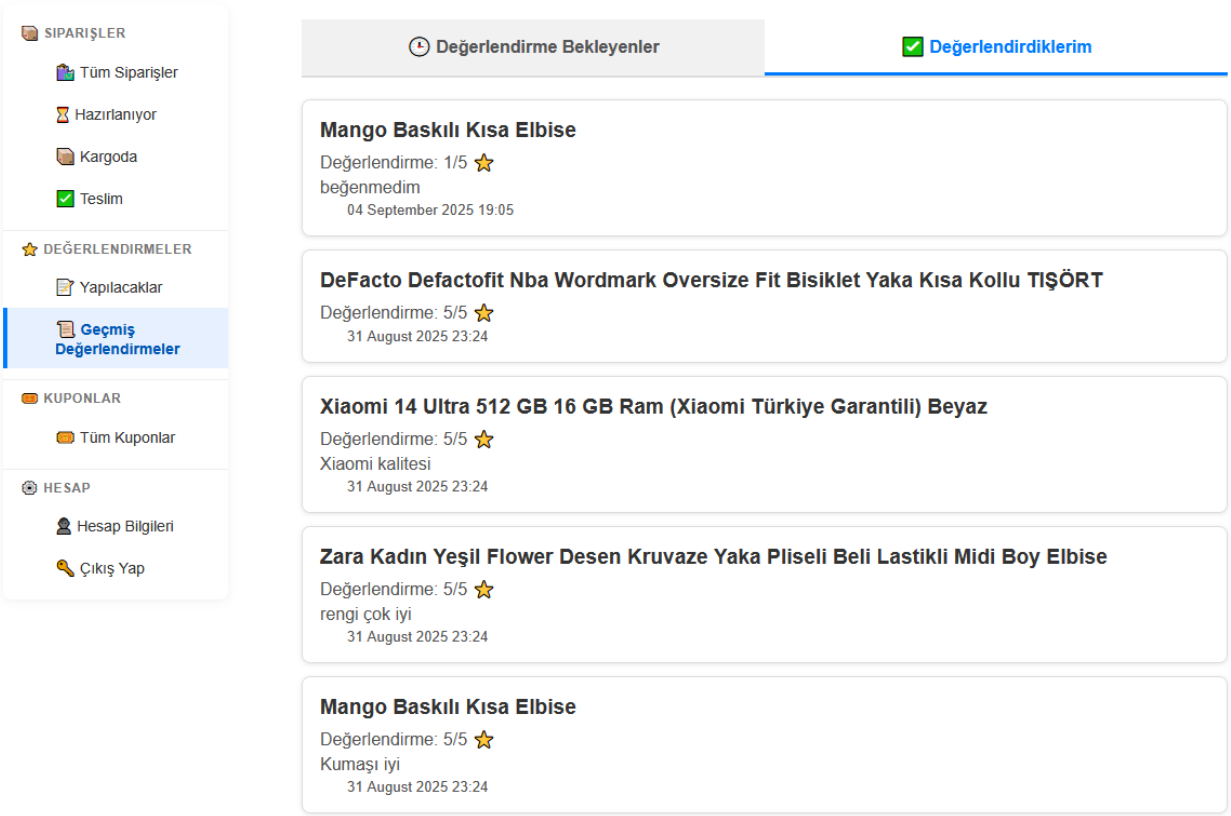
Sipariş detayları JavaScript ile doldurulуyor. Bu yapı hem arama yaparken tıklama ile, hem de normal şablonda aynı şekilde çalışıyor.



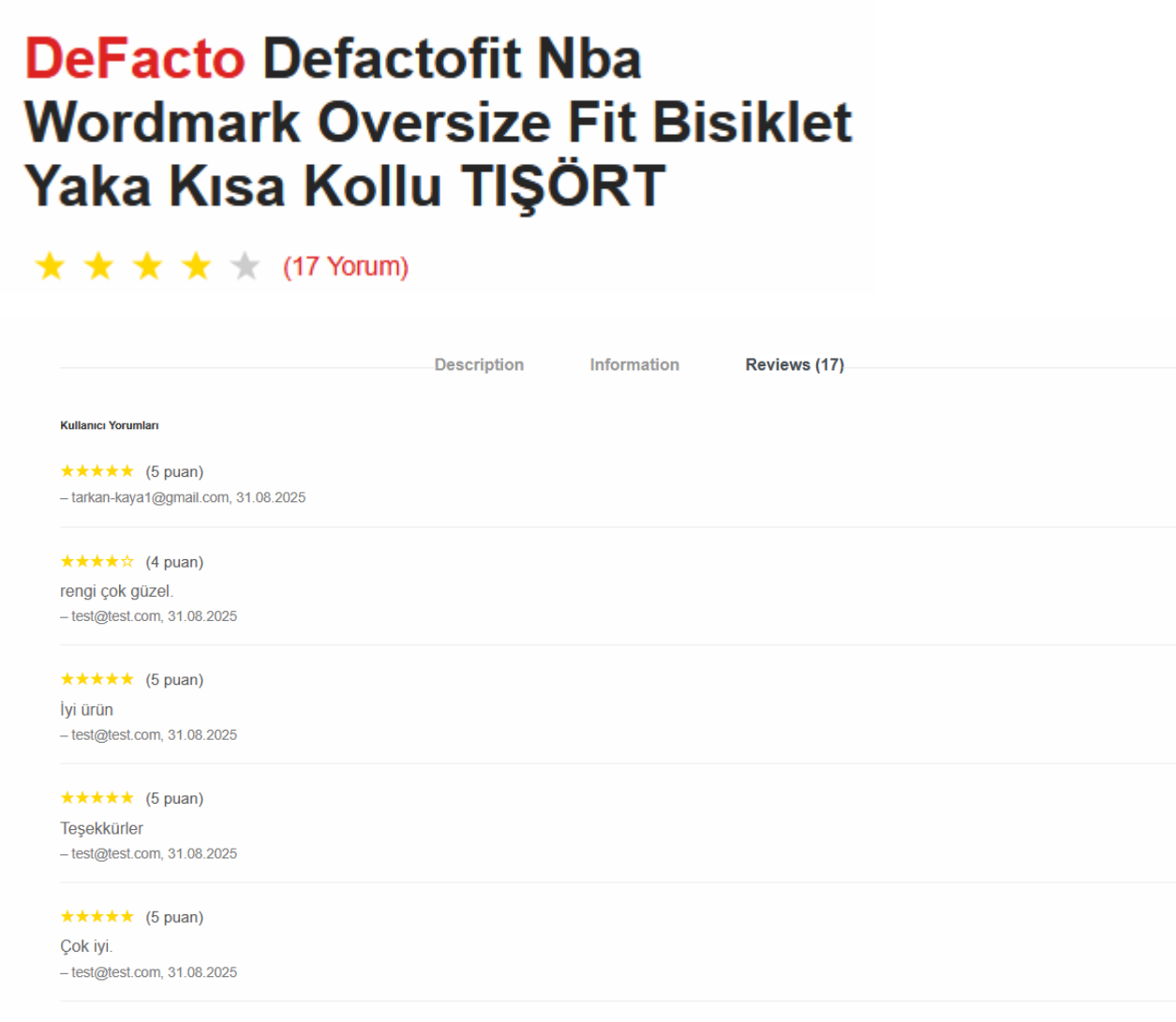
Sipariş tamamlandıktan sonra kullanıcıya yorum yapma imkânı verilmektedir. 1 ile 5 arasında yıldız verebilir ve isterse yorum ekleyebilir. Yapılan yorumlar ürün detay sayfasında görülebilir.



Böylece ürün kartı sadece şık bir görünüm kazanmakla kalmıyor, aynı zamanda kullanıcıya net bir bilgilendirme de sağlıyor. Yorumlar ve puanlama sistemi sayesinde alışveriş yapan kişiler ürünün kalitesi ve memnuniyet durumu hakkında kolayca fikir sahibi olabiliyor.

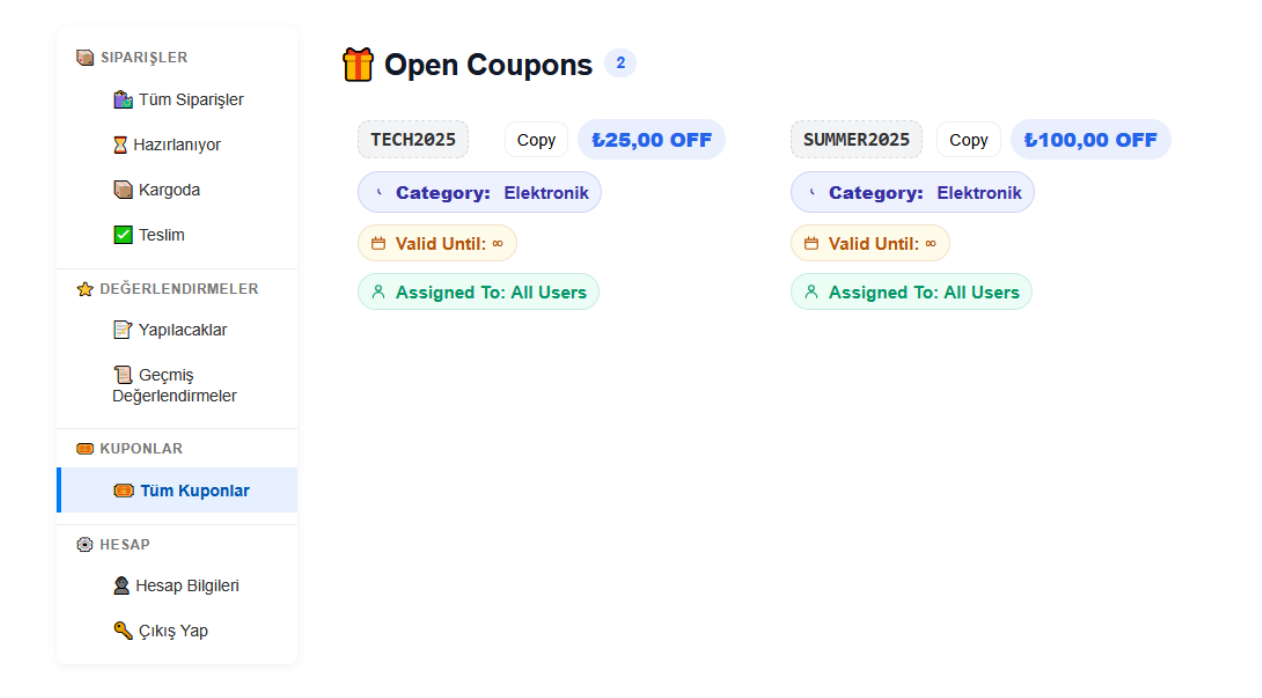


Kullanıcı yaptığı yorumlara bakabilmesi için böyle bir alana oluşturdum.



Ürün detay sayfasında, görünüm olarak ürün değerlendirmeleri bu şekilde yer almaktadır.

Yıldızlara yada ürün detayının bulunduğu yorumlar kısmına basılar tüm mesajlara odaklanılabilir. Bu işlem JavaScriptle dolduğu için sayfa yüklenirkenki **yoğunluğu** azaltmış olur.

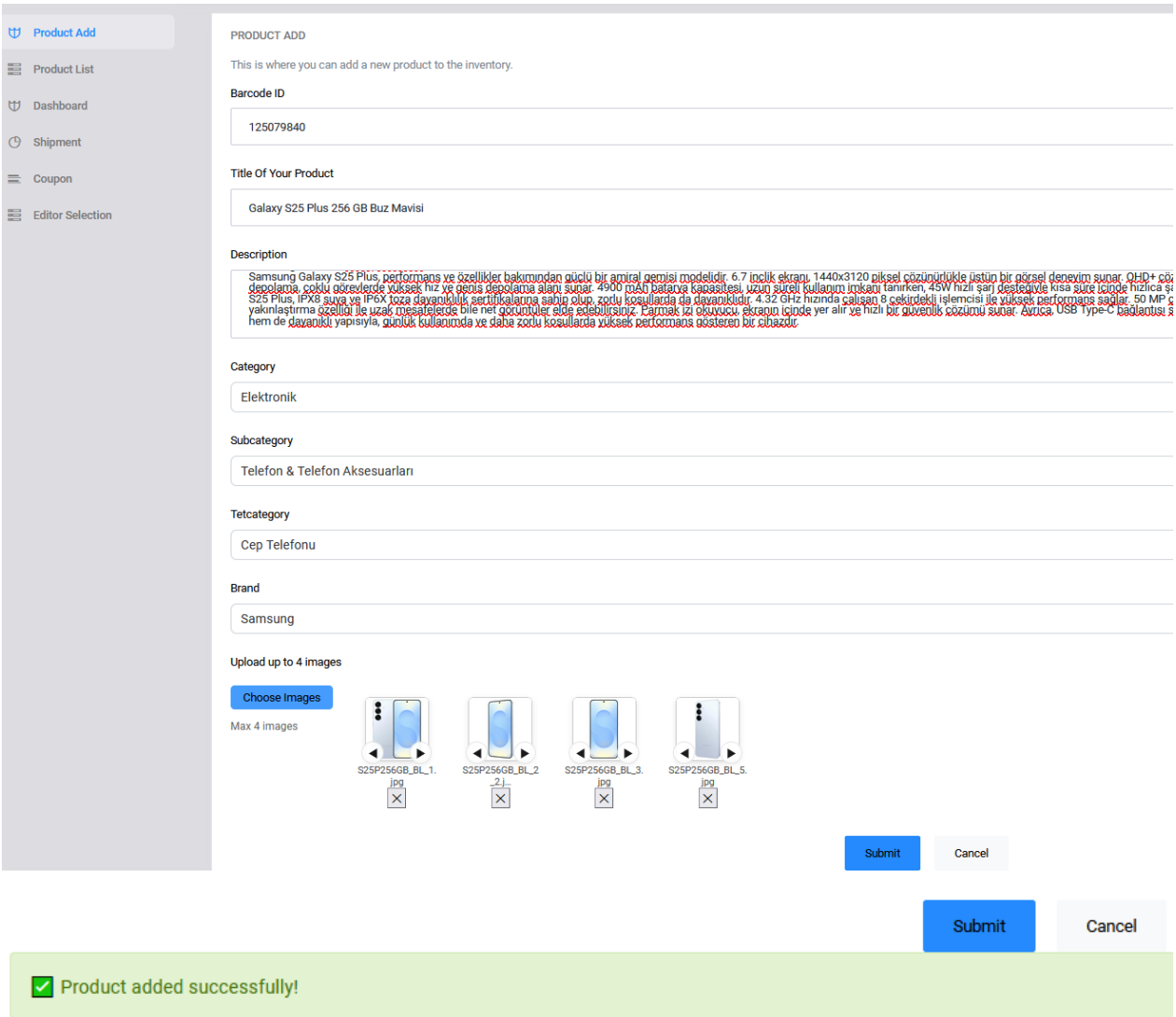


Son kullanıcı sayfasında, kullanıcının kullanabileceği tüm kuponlar görüntülenmektedir.
Kopyala-yapıştır özelliği de bulunduğu için, anlaşılır ve pratik bir yapı kurulduğunu düşünüyorum.
Kullanıcı daha önce kullandığı kuponları tekrar kullanamayacağı için, sonuçlar buna göre dönmektedir.
Eğer giriş yapılmadan bu sayfaya gelinirse, tüm aktif kuponlar görüntülenmektedir.

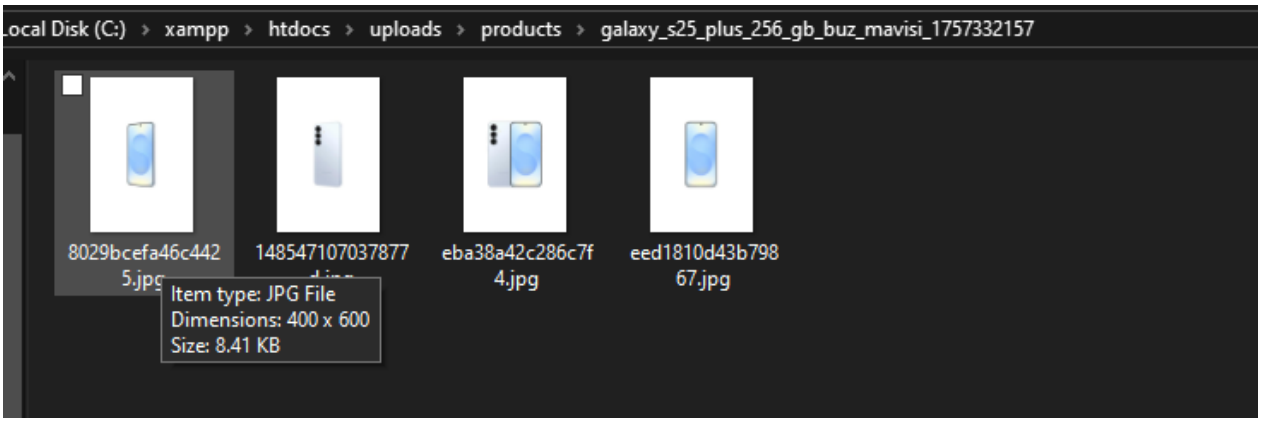
7. Satıcı/Yönetici ekranları

a. Katolog Ürün ekleme

Bir tane örnek ürün girerek örneklendireceğim.Ürünü eklerken başka bir sayfada kategori ve alt kategorilerin ve markaların doldurulduğu bir yapı tasarladım. Böylece düzenli bir altyapı yapabildim. Bu kategori ürünü olduğu için burda bir fiyatlandırma yada buna benzer bir şey yok çünkü fiyat verme işlemiyle ürünler giriliyor. Bir üst kullanıcı yapmadığım için herşeyi herkes yapabiliyormuş gibi ama ana mantık ortada ve projeyi bitirme sürem arttığı için belli başlı şeylere nokta koymam gerekti.

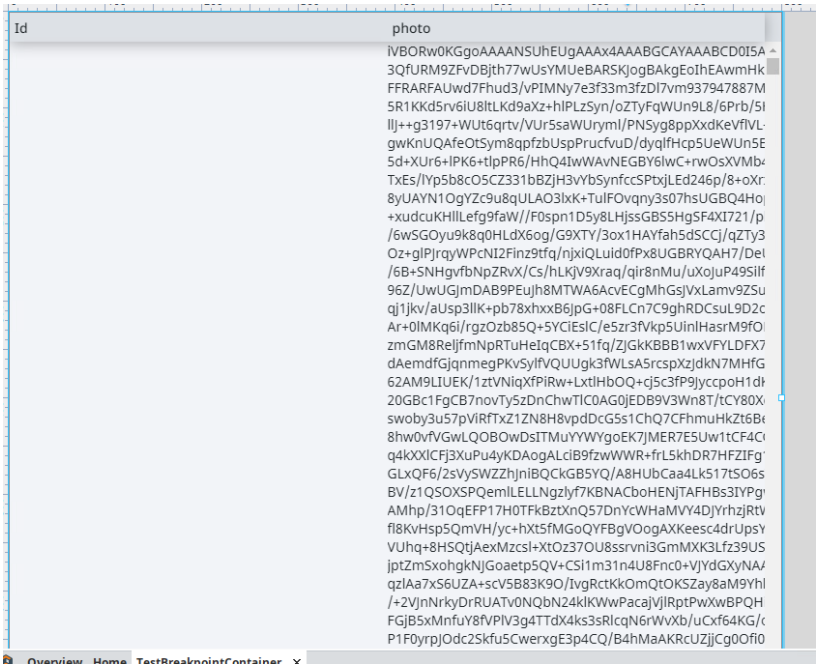


Ürün başarılı bir şekilde eklendiyse ürün işlemin başarılı geçtiği belirtiliyor. Sonra sürekli ekleme yapabilmek için herşey adım sıfırlanıyor.



Bu yapı, ürün eklerken yüklenen resimleri güvenli şekilde işliyor. Başlangıçta en fazla 4 resim kabul ediyorum (bu limit istenirse değiştirilebilir). Dosya türü MIME tipinden kontrol edildiği için sadece uzantısı `.jpg` yapılmış sahte dosyalar engelleniyor. Resimler 400x600 ölçüsünde yeniden boyutlandırılıyor, boş kalan kısımlar dizayna uysun diye beyazla dolduruluyor. Son olarak her dosya rastgele bir adla `.jpg` formatında kaydediliyor.

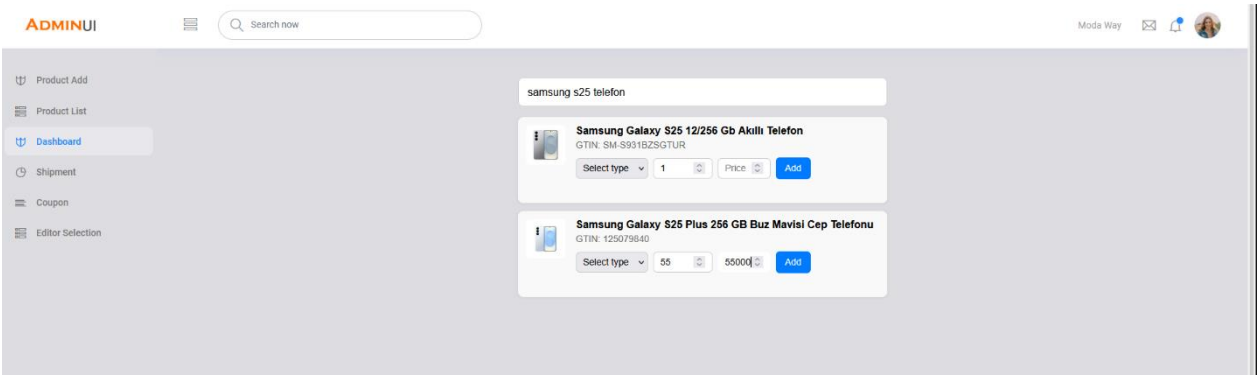
Bunu böyle yapmamamın sebebi şu: Resim ekleme olaylarında her zaman veritabanına resmin bulunduğu **uzantı (dosya yolu)** yazılır. Eğer resim aşırı küçük değilse dosyanın tamamı veritabanına eklenmez.Neden:



Çünkü veritabanı sürükle-bırak mantığında çalışan basit bir yapı değildir. Her veri bir **dize (string)** olarak veritabanına eklenmek zorundadır(Yukardaki resimdeki gibi). Yani resmi eklemek için önce bir veri dizesi dönüştürülmesi gerekir. Bu dize veritabanına kaydedilir, görüntülerken de çekilip tekrar resme dönüştürülür (dönüştürme işlemi tarayıcıda yapılır).

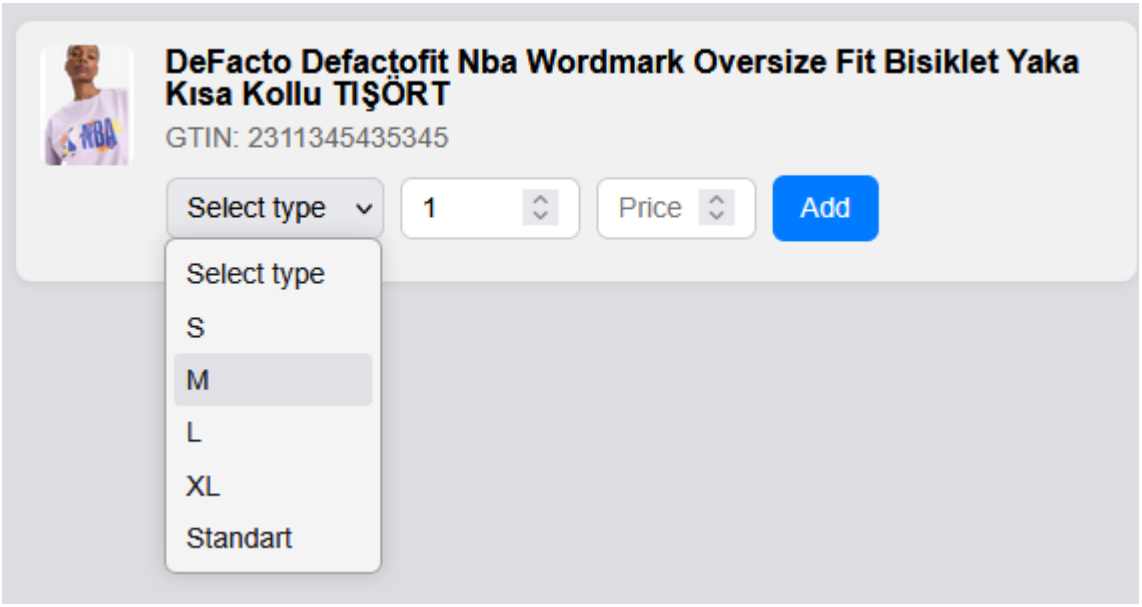
Bu yöntem veritabanını **zorlar**. Başta fark edilmese de birkaç kez tekrarlandığında gözle görülür bir yavaşlama ortaya çıkar. Kısacası sistemin çökmesine kadar gidebilir.

b. Katolog ürüne fiyat verme



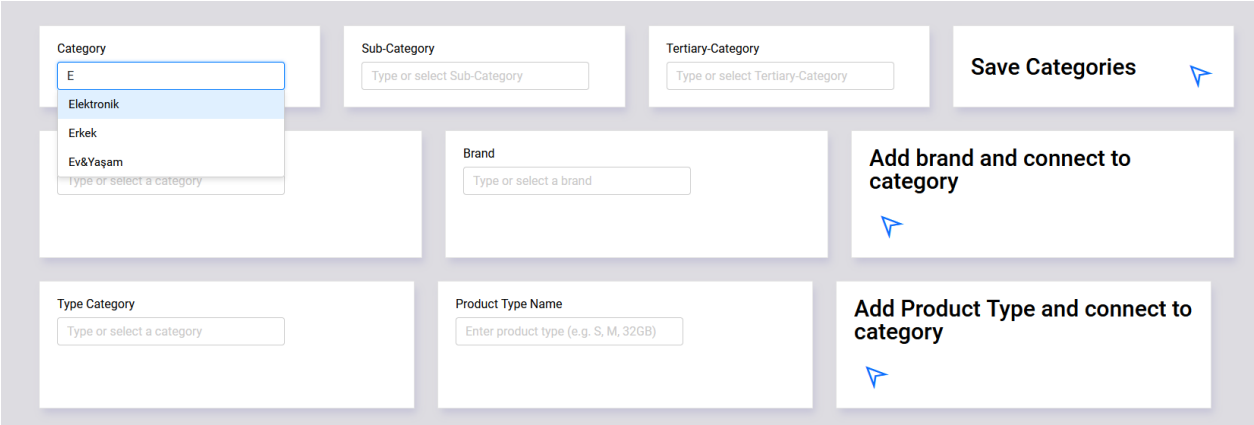
Burada istenilen kelimeler yazılarak elde edilen sonuçlarda ürünün **adet** ve **fiyat** bilgisi belirlenebilir. Böylece ürün, kullanıcı tarafından diğer e-ticaret sayfamızda satışa sunulabilir.

Arama kısmında kelimeleri tek tek sorguladım. Daha hızlı ve kesin sonuçlara ulaşmak için şu yöntemi kullandım: Diyelim üç kelime var(daha fazlada olabilir.). Aralarında boşluk olduğu için önce ilk kelimenin sonucunu ikinci kelimeyle eşleştirdim, ardından bu sorguyu üçüncü kelimeyle karşılaştırdım. Böylece daha doğru ve kesin sonuçlar elde ettim.

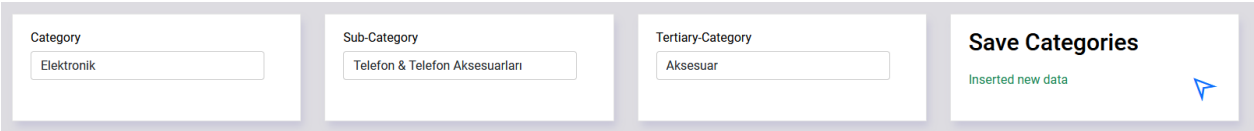


Renk kısmı yapmadım ama beden kısmını yaptım benzer şeyler olduğu için ve proje sürem arttığı için böyle uygun gördüm.

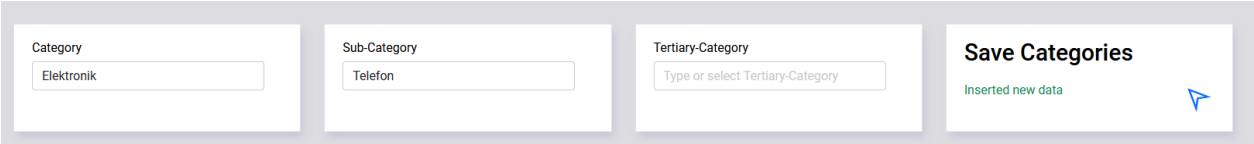
Sayfanın alt kısmında da kesin çözüm olarak düşündüğüm Kategorilerin eklenmesini, Kategoriye özel marka mesela kozmetik bölümünde nike,adidas gibi marka girilmiyor ve kategorilere beden bağlama burda Elektronik'e 256 GB yada Kadın Kategorisine S,M,L bağlanabiliyor.



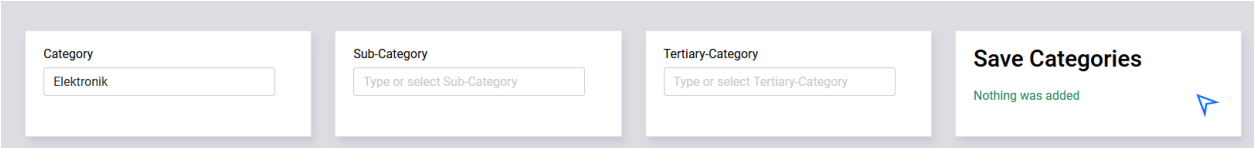
Ilk kısımın Category kısmında var olan kategoriler ortaya çıkıyor şayet yeni eklemek istenirse farklı bir şey yazılırsa yeni kategori olarak eklenir. Butona basıldığı zaman ekleme işlemi gerçekleşir. Örneğin:



yada



yada



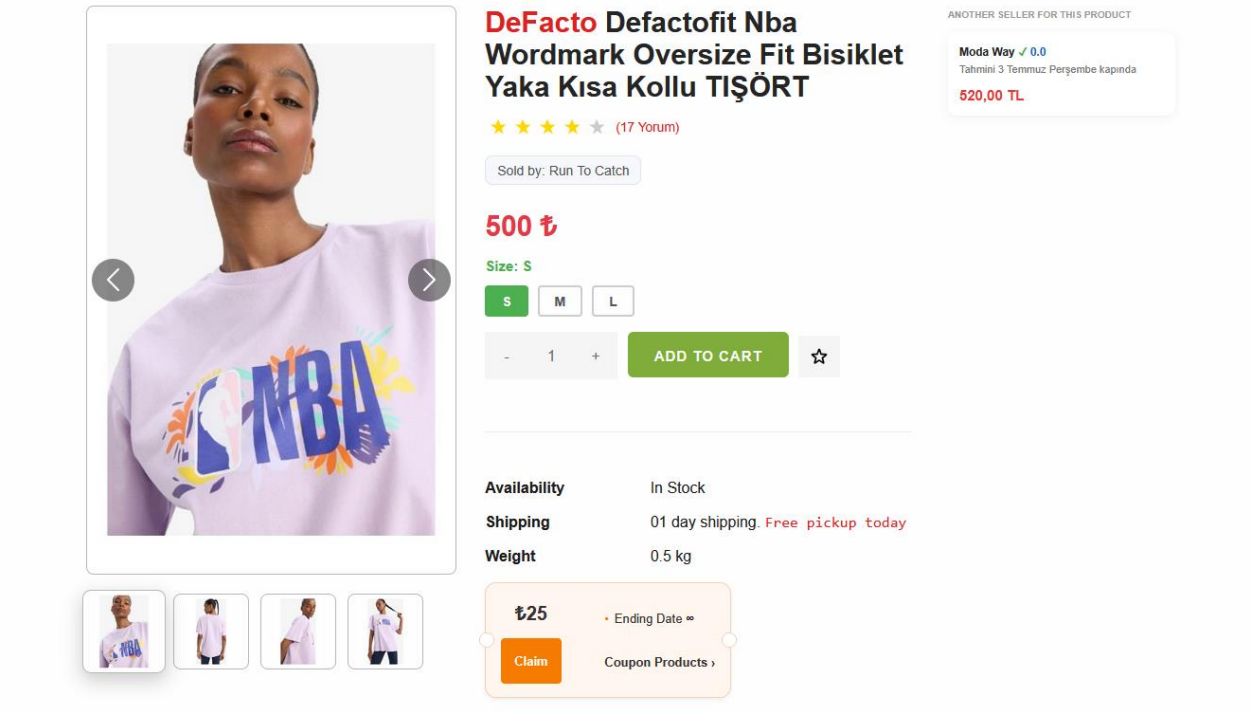
birincil ikincil ve üçüncül kategoriler vardır bu 3 dal arasında ilişkisi ile ürünler eklenir. Bu adımlar ilk kategori varsa ikinci sonra ilk ile ikinci arasına ilişki kurularak veritabanına eklenir. Yani olmayan veriler veritabanına girilerek kategoriler doldurulur. Seri girilebilmesi için tasarladığım ard arda ekleme işlemi yapılabilir. Yani:

Kategori

Kategori'nin -> ikincil Kategorisi

Kategori'nin -> ikincil Kategorisi'nin -> Üçüncül Kategorisi

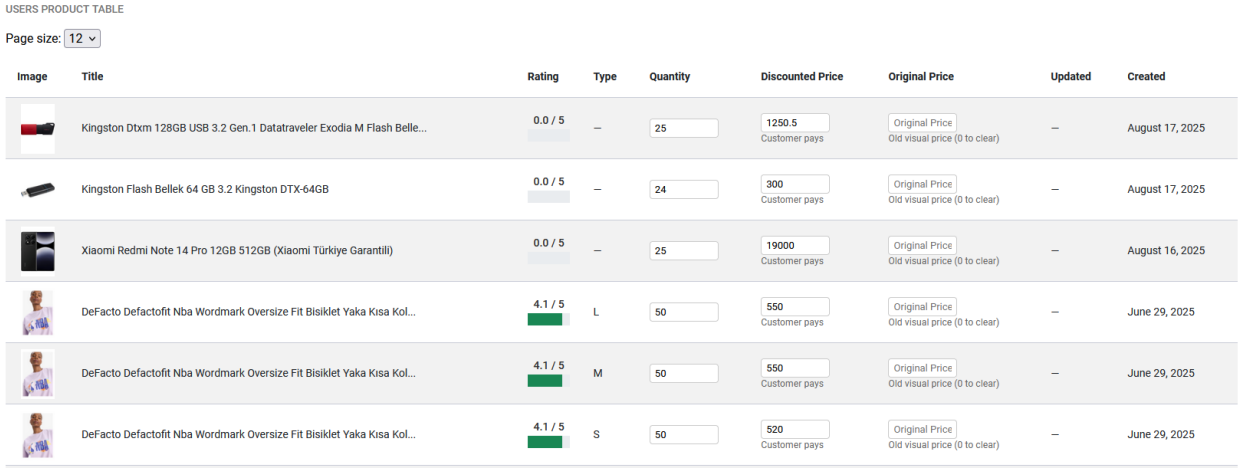
Şeklinde ekleme yapılabiliyor ve **yeni yada varolan** üzerine ekleme yapılabiliyor.



Ürün tipi kategoriye özel beden girilmesini sağlıyor, bunu bir alt kategori yapabilirdim ama böylede istediğim etkiyi veriyor. Kategoriye bağlamış olduğum tip sayesinde kadın kategorisine bedenler eklenebiliyor. Mesela Telefon bedensiz olarak eklenirse ürünse bedensizde devam edibilirim.

Önceden beden eklenmiş ise yada eklenmemişse bunun için küçük bir önlem aldım. Şayet beden eklendiyse bedenle şayet bedensiz se bedenli eklenmiyor. Tabiki bunun için farklı önlemler de alınması gerek ben küçük bir başlangıç yaparak burayı tamamladım.

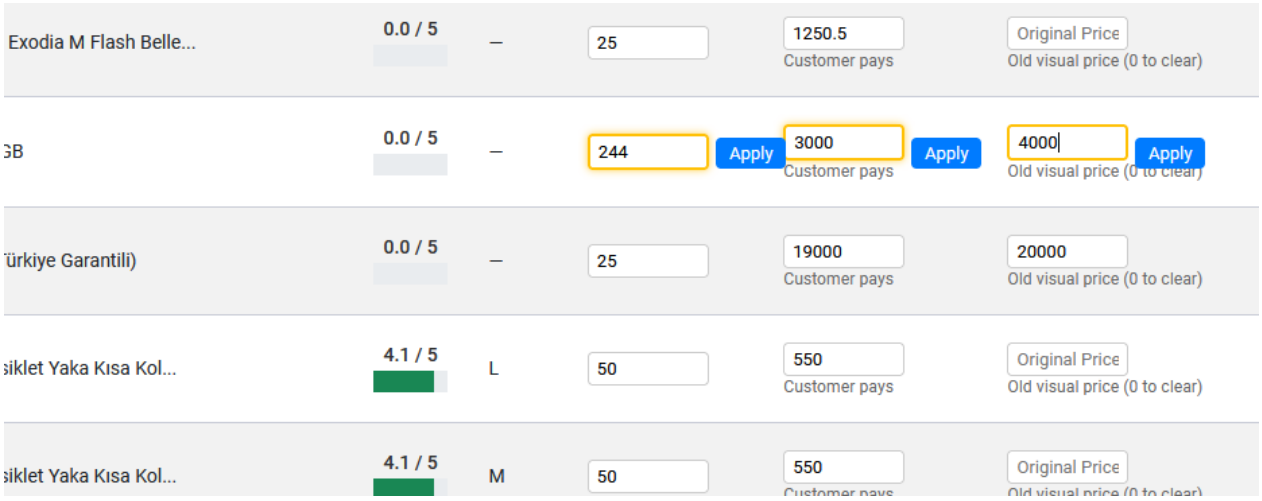
c. Admin ürün listeme/güncelleme


USERS PRODUCT TABLE

Page size: 12

Image	Title	Rating	Type	Quantity	Discounted Price	Original Price	Updated	Created
	Kingston Dtxm 128GB USB 3.2 Gen.1 Datatraveler Exodia M Flash Belle...	0.0 / 5	–	25	1250.5 Customer pays	Original Price Old visual price (0 to clear)	–	August 17, 2025
	Kingston Flash Bellek 64 GB 3.2 Kingston DTX-64GB	0.0 / 5	–	24	300 Customer pays	Original Price Old visual price (0 to clear)	–	August 17, 2025
	Xiaomi Redmi Note 14 Pro 12GB 512GB (Xiaomi Türkiye Garantili)	0.0 / 5	–	25	19000 Customer pays	Original Price Old visual price (0 to clear)	–	August 16, 2025
	DeFacto Defactofit Nba Wordmark Oversize Fit Bisiklet Yaka Kısa Kol...	4.1 / 5	L	50	550 Customer pays	Original Price Old visual price (0 to clear)	–	June 29, 2025
	DeFacto Defactofit Nba Wordmark Oversize Fit Bisiklet Yaka Kısa Kol...	4.1 / 5	M	50	550 Customer pays	Original Price Old visual price (0 to clear)	–	June 29, 2025
	DeFacto Defactofit Nba Wordmark Oversize Fit Bisiklet Yaka Kısa Kol...	4.1 / 5	S	50	520 Customer pays	Original Price Old visual price (0 to clear)	–	June 29, 2025

Fiyat verilmiş olan ürünler burda hem listeleniyor bende adet fiyat ve piyasa fiyatı olacak şekilde 3 farklı oldurulabilme alanı yaptım.

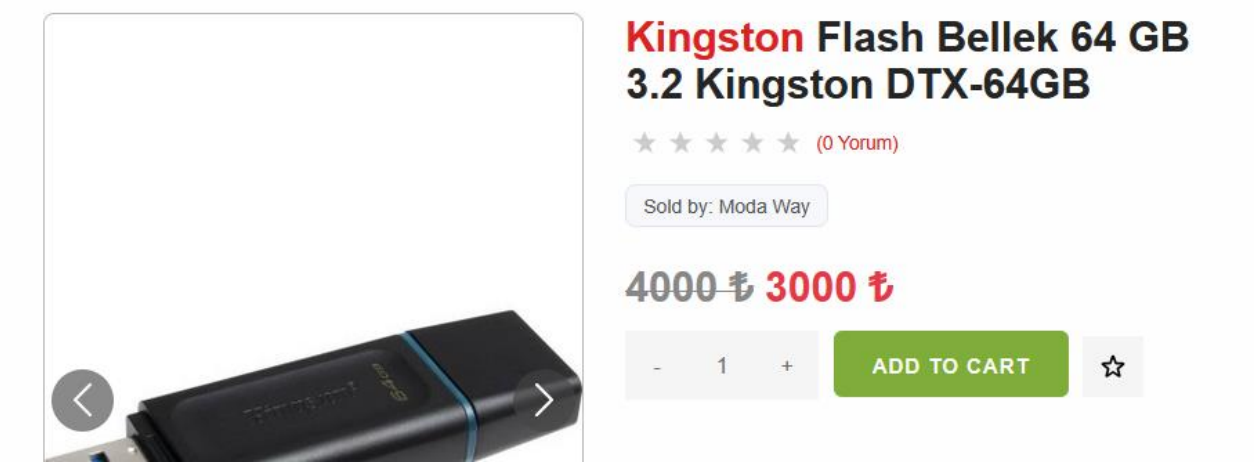


Bu alanlar Javascriptle eğer kutuda bir değişiklik olursa yanına buton belirerek güncelleme yapılabilme sağlanıyor.

Original set to ৳4,000.00 (active ৳3,000.00).

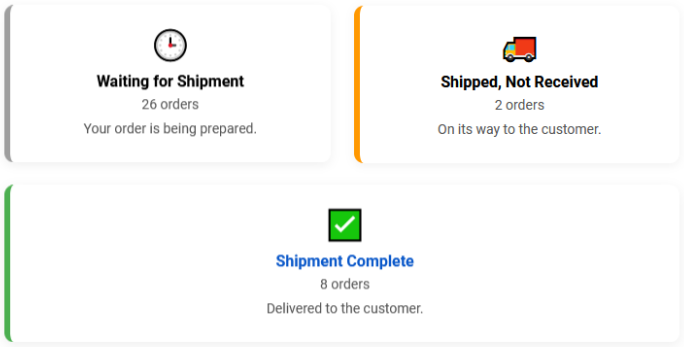
ting	Type	Quantity	Discounted Price	Original Price	Updated	Created
.0 / 5	–	25	1250.5 Customer pays	Original Price Old visual price (0 to clear)	–	August 17, 2025
.0 / 5	–	244	3000 Customer pays	4000 Old visual price (0 to clear)	–	August 17, 2025

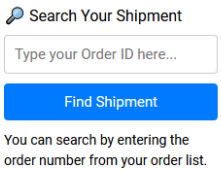
Şayet güncellenirse bu admine iletiliyor.



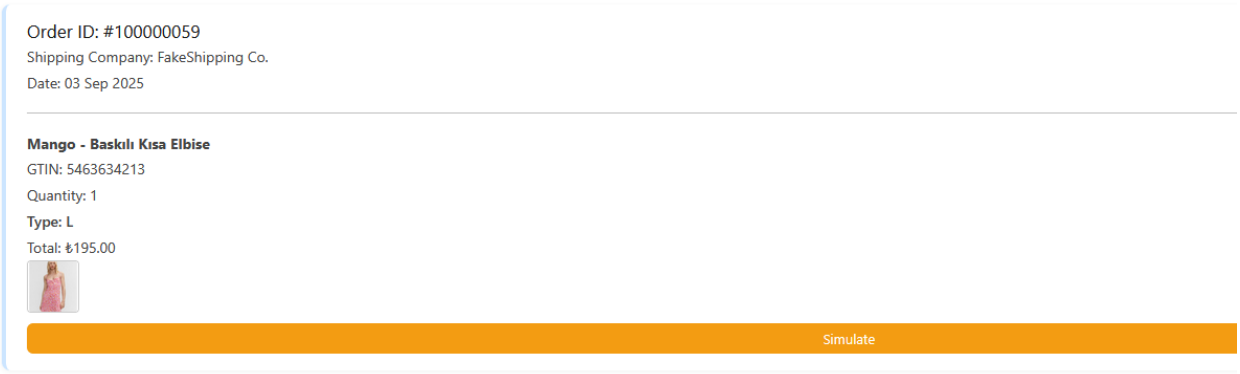
Ve sonuç anında yansıyor.

d. Admin Kargo durumu



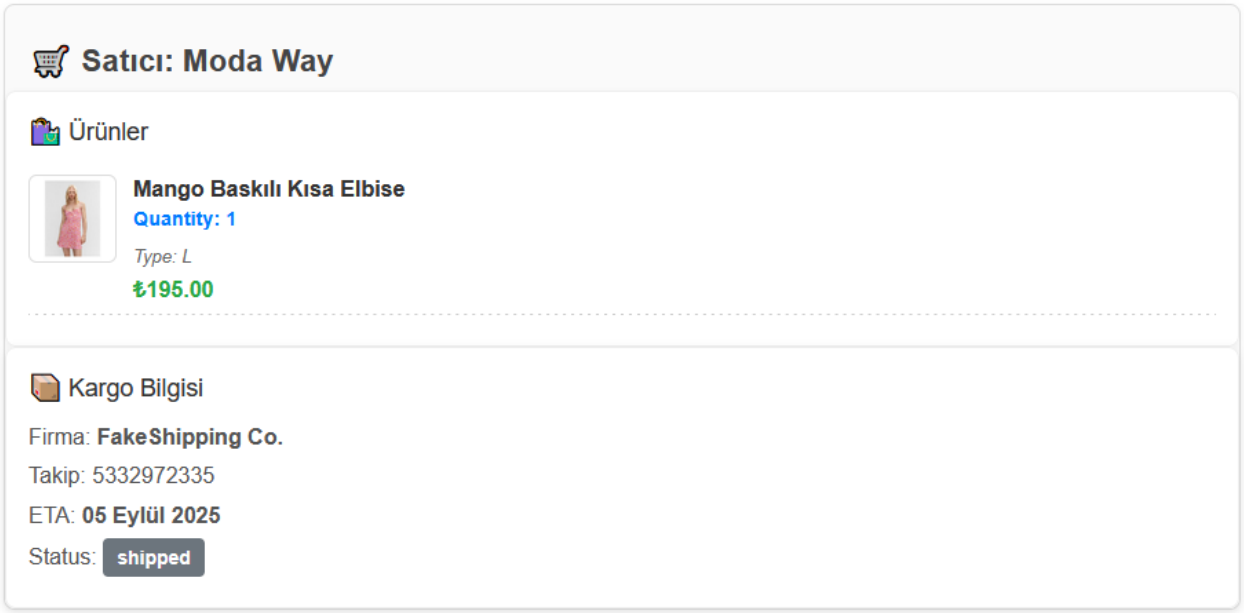


Burada admin, kullanıcı üzerinde olan kargoların sipariş durumunu güncelleyebiliyor. Sayfanın alt kısmında ise sipariş numarasına göre kargoları listelebiliyor.

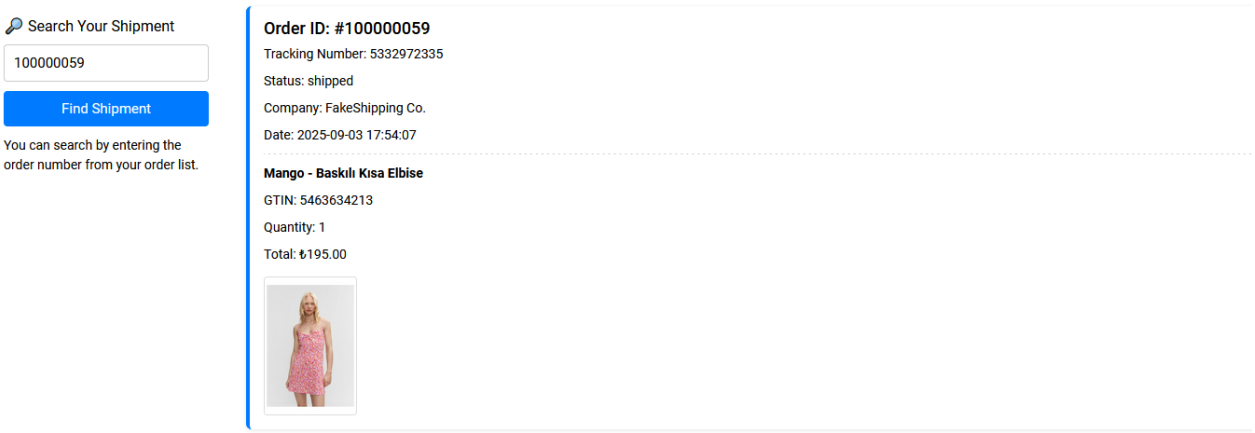


Bunu daha önceden de kullanmıştım. 3 sayfa tek bir sayfadan türetiliyor. Özel kelime olarak sadece ‘bekliyor’, ‘gönderildi’ ve ‘tamamlandı’ kelime gönderiliyor ve istenilen sayfa açılıyor.

```php
<?php
$status = 'completed';
include 'shipment-template.php';
?>
```

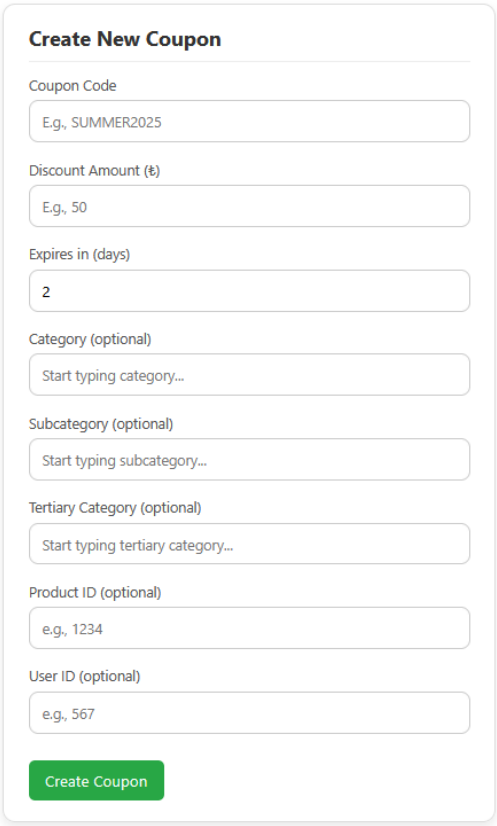


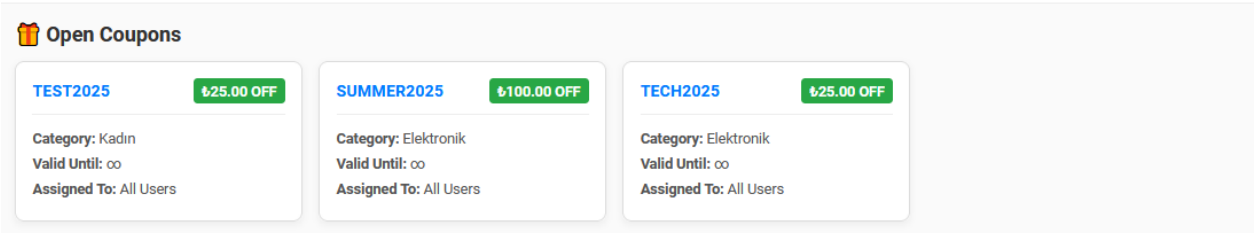
Kargo şiketini simule etmem gerekiyordu. Numarasına göre kargonun nerde olduğu göstermek istedim. Bunun için veritabanında ilgili bölgeleri açtım ve ihtiyacıma göre şekillendirdim. Kısa bir işlem yapmam gerekiyordu ve böyle devam ettim.



Sorgulayarak kargo durumunuda öğrenilebiliyor.

e. **Kupon Ekleme**





Burda istenilen koşullara göre kupon oluşturuluyor. Sayfanın alt kısmında ise hali hazırda açık olan kuponlar görülmektedir.
Kuponlar kişiye özel, ürüne özel veya kategorilere özel olacak şekilde oluşturulabilir.

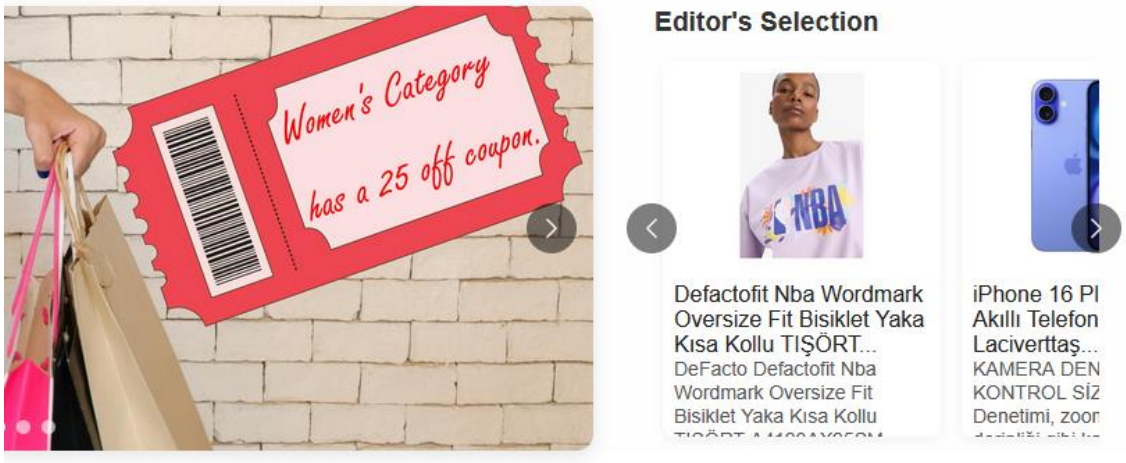
f. **Editor Seçimi**

All Products (Editor Selection)

#	Product	Status	Created At	Editor Pick
1	Galaxy S25 Plus 256 GB Buz Mavisi Cep Telefonu	0	—	☐
2	20 W USB-C Güç Adaptörü - MD3J4TU/A (Apple Türkiye Garantili)	0	—	☐
3	EP-T4510X Type-C Süper Hızlı Şarj Cihazı 45W Siyah	0	—	☐
4	Redmi 20000 Mah Taşınabilir Hızlı Şarj Cihazı - USB-C - 18W 2 Çıkışlı Powerbank - Siyah	0	—	☐
5	326 20000 Mah Taşınabilir Hızlı Şarj Cihazı - USB-C - 15W 2 Çıkışlı Powerbank - Siyah - A1367 (Anker Türkiye Garantili)	0	—	☐
6	Moon 64 GB USB 3.1 Ultra Speed Metal USB Flash Bellek	0	—	☐
7	Dtxm 128GB USB 3.2 Gen.1 Datatraveler Exodia M Flash Bellek DTXM/128	0	—	☐
8	Flash Bellek 64 GB 3.2 Kingston DTX-64GB	0	—	☐
9	Redmi Note 14 Pro 12GB 512GB (Xiaomi Türkiye Garantili)	0	—	☐
10	Defactofit Nba Wordmark Oversize Fit Bisiklet Yaka Kısa Kollu TIŞÖRT	1	2025-08-07 19:12	☑

Next »

Burda sağdaki kutucuğa basılarak istelen ürünün anasayfa görülmesine olanak sağlıyor.



İletişim

- **Ad Soyad:** Tarkan Kaya
- **Telefon:** 0(536) 450 16 33
- **İkamet:** [Üsküdar / İstanbul]
- **E-posta:** tarkan_kaya@ymail.com
- **LinkedIn:** https://www.linkedin.com/in/tarkan-kaya/
- **GitHub:** https://github.com/betsunohito

Diğer Projelerim

- 📌 Same File Finder – Bilgisayarınızdaki aynı dosyaları kolayca bulup temizlemenizi sağlar.
https://github.com/betsunohito/SameFileFinder
- 📌 My Shortcuts – Firefox için kısayol odaklı bir eklenti.
https://github.com/betsunohito/myShortcuts
- 📌 Coverage Cleaner – Chrome'un coverage çıktısını işleyip kullanılmayan kodları temizlemeye yarar.
https://github.com/betsunohito/CoverageCleaner