KOCAELİ ÜNİVERSİTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ BLM307 YAZILIM LAB. I PROJE 3

1st Fatih Bal
210202034
Bilgisayar Mühendisliği 2.öğretim
Kocaeli Üniversitesi
fatih18bal.pc@gmail.com

2nd Ömer Furkan Çelik
220202023
Bilgisayar Mühendisliği 2.öğretim
Kocaeli Üniversitesi
omerfcelikk.32@gmail.com

***Abstract*—Bu proje, eş zamanlı sipariş yönetimi ve stok güncellemelerini gerçekleştiren bir sistem tasarımı üzerine odaklanmıştır. Amaç, multithreading ve senkronizasyon mekanizmalarını kullanarak aynı kaynağa eş zamanlı erişim problemlerini çözmek ve işlem önceliği, dinamik sıralama, stok kontrolü gibi süreçleri etkin bir şekilde yönetmektir. Sistem; müşteri türlerine dayalı işlem sıralaması, bütçe ve stok yönetimi, dinamik öncelik hesaplama, ve işlem loglamasını içeren bir altyapı sunmaktadır. Proje, Python ve SQL tabanlı teknolojiler kullanılarak geliştirilmiş ve hem masaüstü hem de web tabanlı uygulama senaryoları için uyarlanabilir bir çözüm sağlamaktadır.**

***Index Terms*—Eş Zamanlı Sipariş Yönetimi, Stok Yönetimi, Dinamik Öncelik, Veri Tabanı Tasarımı**

I. Özet

Eş Zamanlı Sipariş ve Stok Yönetimi: Bu proje, eş zamanlı sipariş yönetimi ve stok güncellemelerini gerçekleştiren bir sistem tasarımı üzerine odaklanmıştır. Kullanıcılar, mevcut stoklara dayalı olarak sipariş verebilir ve bütçelerine uygun alışverişler gerçekleştirebilir. Sistem, müşteri türlerine göre dinamik öncelik sıralaması, stok ve bütçe yönetimi gibi süreçleri etkin bir şekilde yönetmeyi hedeflemektedir. Proje, multithreading ve senkronizasyon mekanizmalarını kullanarak kaynakların verimli şekilde kullanılmasını sağlamaktadır.

II. Giriş

Bu proje, eş zamanlı sipariş ve stok yönetimi sisteminin geliştirilmesini hedeflemektedir. Proje kapsamında aşağıdaki konularda geliştirme yapılması beklenmektedir:

1) **Veri Tabanı Tasarımı**
   Projede veri tabanı tasarımı, aşağıdaki temel tabloları içerecek şekilde gerçekleştirilmelidir. Proje geliştirme aşamasında yeni tablolar eklenebilir ve bu tablolar arasında ilişkiler tanımlanabilir. Her bir tablo için *primary key* ve *foreign key* tanımlamaları yapılmalı, ilişkisel veri tabanı tasarımı normalizasyon kurallarına (1NF, 2NF, 3NF vb.) uygun olarak oluşturulmalıdır.
   a) **Müşteriler:** *CustomerID*, *CustomerName*, *Budget*, *CustomerType* (Premium/Standard), *TotalSpent*.
   b) **Ürünler:** *ProductID*, *ProductName*, *Stock*, *Price*.
   c) **Siparişler:** *OrderID*, *CustomerID (Foreign Key)*, *ProductID (Foreign Key)*, *Quantity*, *TotalPrice*, *OrderDate*, *OrderStatus*.
   d) **Loglar:** *LogID*, *CustomerID (Foreign Key)*, *OrderID (Foreign Key)*, *LogDate*, *LogType*, *LogDetails*.
2) **Nesne Yönelimli Programlama**
   Bu projede nesne yönelimli programlama prensipleri kullanılacaktır. Her bir bileşen (müşteri, ürün, sipariş vb.) bir sınıf olarak modellenmeli ve her bir sınıf için uygun metotlar tanımlanmalıdır. Ayrıca, veritabanı işlemleri sırasında tetikleyiciler (*triggers*) kullanılarak, tablolar arasında veri bütünlüğü sağlanmalıdır.
3) **Arayüz Geliştirmeleri**

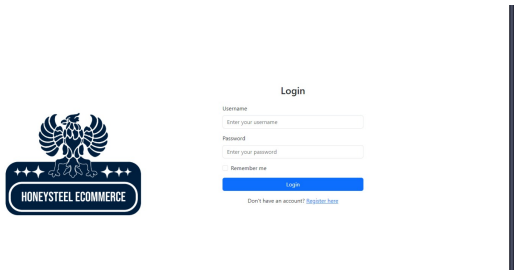

Fig. 1. Giriş ekranı

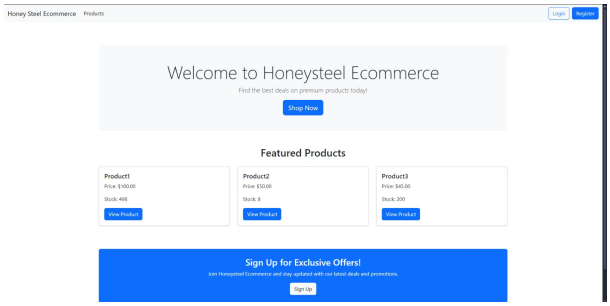

Fig. 2. Giriş ekranı

a) **Ana Ekran Düzeni ve Yapısı:**
   i) **Görsel Hiyerarşi:** Ana ekranda önemli bilgiler net bir şekilde öne çıkarılmalıdır. Müşteri türü ve sipariş durumu gibi detaylar kullanıcı dostu bir arayüzle sunulmalıdır.
   ii) **Renk Kodlaması:** Yüksek öncelikli müşteriler ve kritik stok seviyesine sahip ürünler renkli simgelerle belirtilmelidir.
   iii) **Dinamik Yerleşim:** Ekran alanı, içerik miktarına göre otomatik olarak optimize edilmelidir.

b) **İleri Seviye Arama Sistemi:**
   i) **Gelişmiş Ürün Arama ve Öneri Sistemi:** Ürün aramalarında, benzer ürünler ve alternatifler önerilmelidir. Ayrıca, stok durumu ve olası indirim oranları kullanıcıya sunulmalıdır.

c) **Sipariş ve Müşteri Detay Sayfası Özellikleri:**
   i) **Modüler Tasarım:** Detay sayfası sezgisel bir kullanıcı deneyimi sunmalıdır. Müşteri bilgileri ve sipariş detayları net bir şekilde ayrılmalıdır.
   ii) **Müşteri Bilgileri:** Müşteri türü (Premium/Normal), toplam harcama ve sipariş geçmişi renkli ve anlaşılır bir tabloyla gösterilmelidir.
   iii) **Sipariş Detayları:** Sipariş edilen ürünler, miktar, birim fiyat ve toplam tutar detaylı bir tabloyla sunulmalıdır. Kritik bilgiler vurgulanmalı, stok durumu görsel olarak işaretlenmelidir.

III. Yöntem

A. Ana Sayfa

Web sitesi açıldığında kullanıcıyı, ana sayfada tüm ürünlerin listelendiği, kategoriye göre sıralama ve arama seçeneklerinin bulunduğu bir arayüz karşılar. Kullanıcı, ana sayfa üzerinden ürün detaylarına ulaşabilir, alışveriş sepetine ürün ekleyebilir ve önerilen ürünleri görüntüleyebilir.

B. Ürün Yönetimi

Yönetici paneli üzerinden admin kullanıcıları, ürün yönetimi işlemlerini gerçekleştirebilir:

1) Yeni ürün ekleme,
2) Mevcut ürünlerin bilgilerini (isim, fiyat, açıklama, stok durumu) güncelleme,
3) Ürünleri kategorilere ayırma veya silme.

Bu işlemler Django'nun admin paneli ve özel olarak geliştirilmiş yönetim modülleri üzerinden yapılmaktadır.

C. Stok Yönetimi

Sistem, ürün stoklarının takibini ve kontrolünü sağlar:

1) Yeni stok ekleme veya mevcut stoğu güncelleme,
2) Kritik stok seviyesindeki ürünler için görsel uyarıların gösterilmesi,
3) Satın alma işlemleri sonrası stokların otomatik olarak güncellenmesi.

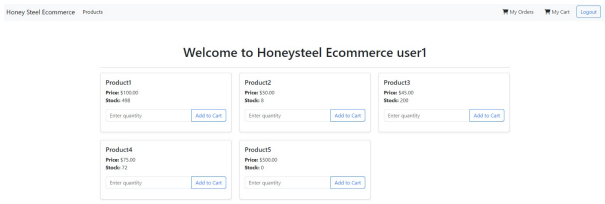

Fig. 3. Alışveriş Ekranı

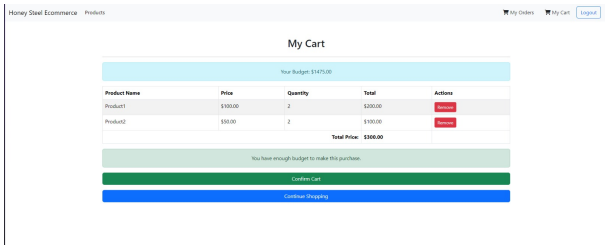

Fig. 4. My Cart

D. Öneri Sistemi

Sistem, kullanıcının alışveriş geçmişine ve davranışlarına göre ürün önerileri sunar:

1) Kullanıcının alışveriş sepetine benzer ürünleri önerir.
2) İlgili kategorilerdeki popüler ürünleri öne çıkarır.
3) Kullanıcı tercihlerine göre kişiselleştirilmiş öneriler sunar.

E. Ürün Detayları

Kullanıcılar, seçtikleri ürünün detaylarını inceleyebilir:

1) Ürün açıklaması, fiyatı, stok durumu görüntüleyebilir,
2) Ürünün toplam fiyatını hesaplayarak alışveriş sepetine ekleyebilir.

F. Veritabanı İşlemleri

Web uygulaması, arka planda güçlü bir ilişkisel veritabanı kullanarak işlemleri yönetir:

1) Ürün, kullanıcı ve sipariş bilgileri normalize edilmiş tablolarda saklanır.
2) Ürün ve siparişler arasındaki ilişkiler, Django ORM kullanılarak etkin bir şekilde yönetilir.
3) Yedekleme ve güvenlik işlemleri düzenli olarak yapılır.
4) Satış işlemleri sırasında verilerin tutarlılığı sağlanır ve stok bilgileri anlık olarak güncellenir.

Bu sistem, kullanıcı deneyimini iyileştirirken, güvenilir ve hızlı bir işlem altyapısı sunar.

IV. Deneysel Sonuçlar

Geliştirilen Honeysteel e-ticaret web uygulamasının test sonuçları, sistemin tüm temel işlevlerini başarıyla yerine getirdiğini göstermiştir. On farklı kategoride yüz ürün içeren veritabanı üzerinde yapılan testlerde:

- Arama işlevleri yüksek doğrulukla ve hızlı bir şekilde çalışmıştır.

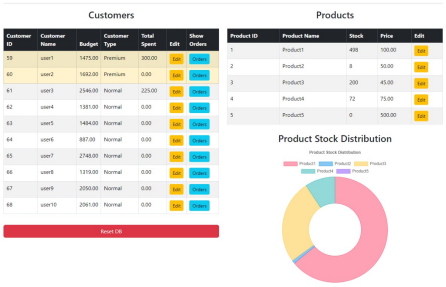

Fig. 5. Honeysteel E-ticaret Uygulaması

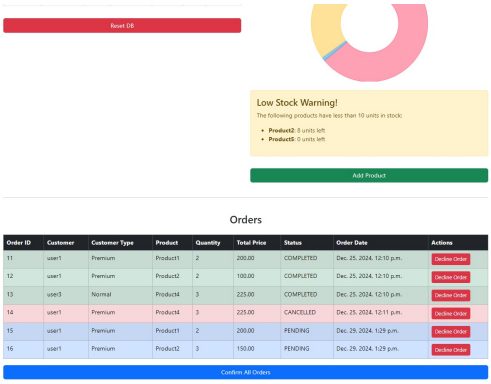

Fig. 6. Honeysteel E-ticaret Uygulaması

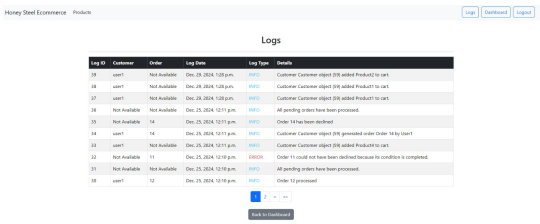

Fig. 7. Loglar

- Kullanıcı arayüzü, test kullanıcıları tarafından kolay anlaşılır ve kullanımı sezgisel bulunmuştur.
- Ürünler ve stoklar arasındaki ilişkiler doğru şekilde yönetilmiş, stok durumu renk kodlaması ile etkili bir şekilde sunulmuştur.
- Sipariş süreçleri başarılı bir şekilde tamamlanmış, stok miktarları ve sipariş kayıtları sorunsuz bir şekilde güncellenmiştir.
- Sistem yanıt süreleri ortalama 0.8 saniyenin altında kalarak yüksek performans sağlamıştır.

Genel olarak, sistem işlevsellik, kullanıcı deneyimi ve performans açısından tatmin edici sonuçlar vermiştir.

V. Yalancı Kod

1) Web uygulamasını başlat.
2) Kullanıcı giriş ekranı:
   a) Kullanıcı adı ve şifre kontrolü yap.
   b) Başarılı ise ana sayfaya yönlendir, başarısız ise hata mesajı göster.
3) Kullanıcı ana sayfada işlem seç:
   a) Ürün Görüntüle/Detayları İncele.
   b) Ürün Ara/Filtrele.
   c) Sipariş Ver.
   d) Yönetici yetkisi ile:
      - Ürün Ekle/Sil/Güncelle.
      - Stok Yönetimi.
4) Seçilen işlemi gerçekleştir:
   a) Sipariş verildiğinde stok miktarını güncelle.
   b) Ürün eklendiğinde veya silindiğinde veritabanını güncelle.
5) Yeni işlem için ana sayfaya dön veya çıkış yap.

VI. Sonuç

Bu proje sonucunda, Python ve Django kullanarak bir e-ticaret web uygulamasının nasıl geliştirileceği konusunda değerli deneyimler edindim. Geliştirilen Honeysteel projesinde, kullanıcı dostu bir arayüz, güçlü bir veritabanı yapısı ve dinamik işlem yetenekleri birbirine entegre bir şekilde tasarlanıp uygulanmıştır.

Projede kullanılan veritabanı yönetimi, dinamik arama algoritmaları ve kullanıcı arayüzü tasarımı gibi yazılım bileşenleri, modern bir web uygulaması için temel gereksinimlere uygun şekilde geliştirilmiştir. Örneğin, ürünler ve stoklar arasındaki many-to-many ilişkilerin yönetimi, dinamik arama ve filtreleme algoritmalarının uygulanması ve sipariş işlemlerinin veritabanında tutarlı bir şekilde güncellenmesi gibi özellikler sistemin temel yapı taşlarını oluşturmuştur.

Kullanıcı dostu bir arayüz geliştirmek için Django'nun sunduğu özellikler etkili bir şekilde kullanılmıştır. Özellikle dinamik filtreleme, ürün sıralama ve öneri sistemi, kullanıcıların ihtiyaç duydukları ürünlere hızlı ve kolay bir şekilde ulaşmalarını sağlamıştır. Bu süreçte, kullanıcı geri bildirimlerine uygun tasarım düzenlemeleri yapılmıştır.

Proje sürecinde karşılaşılan zorluklar arasında, dinamik öncelik algoritmalarının doğru uygulanması ve büyük veri kümeleriyle yüksek performanslı işlemler gerçekleştirilmesi yer aldı. Bu zorluklar, özellikle algoritma geliştirme, veritabanı yönetimi ve web tabanlı arayüz tasarımı konularındaki becerilerimi geliştirmeme katkı sağlamıştır.

Sonuç olarak, bu proje hem teknik bilgi hem de kullanıcı odaklı yazılım geliştirme açısından önemli bir öğrenim süreci sunmuştur. Edinilen tecrübeler, gelecekteki yazılım projelerinde daha etkili ve kullanıcı dostu çözümler üretmek için güçlü bir temel oluşturmaktadır.

References

[1] https://docs.djangoproject.com/en/stable/

[2] https://www.geeksforgeeks.org/creating-an-e-commerce-website-using-django/

[3] https://simpleisbetterthancomplex.com/

[4] https://www.youtube.com/watch?v=GT10PnUFLlE

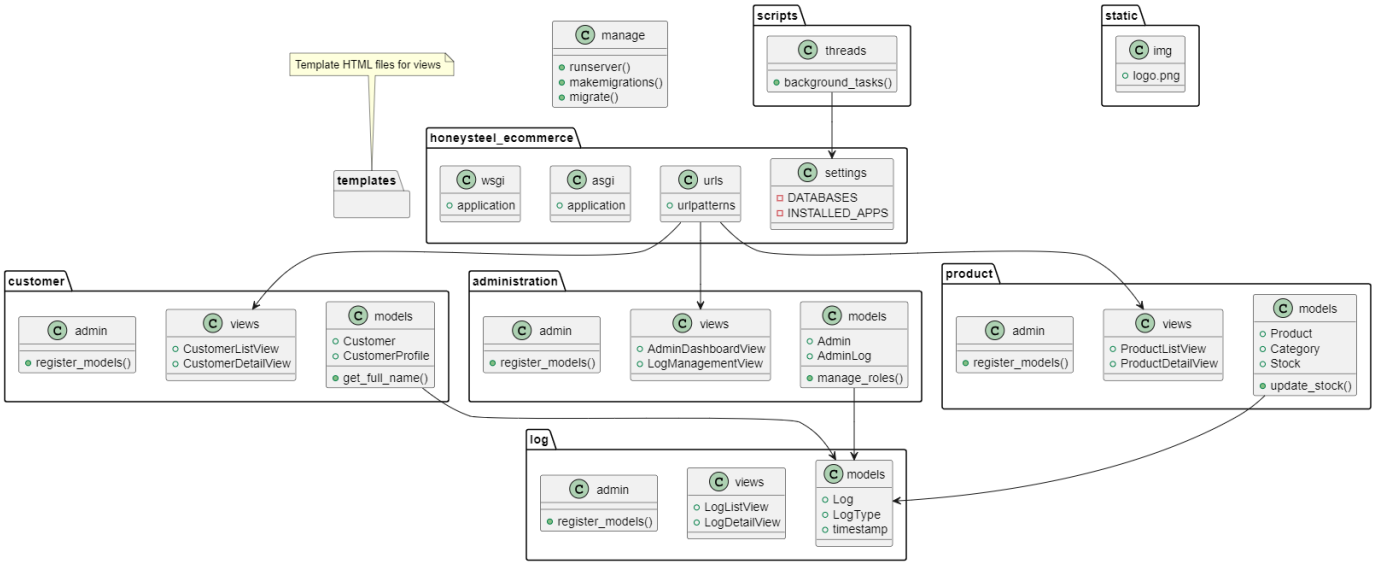

Fig. 8. UML Diagram

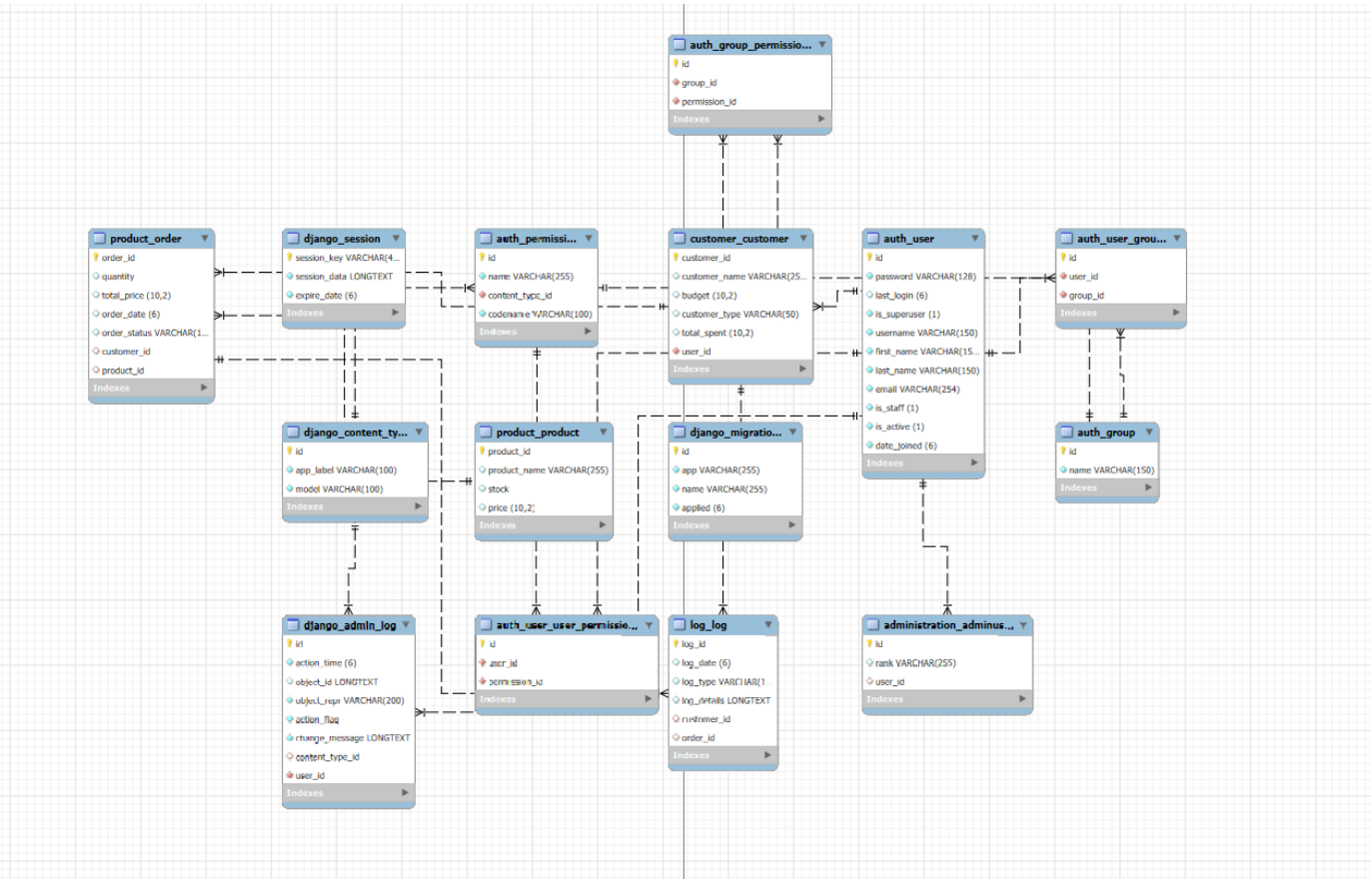

Fig. 9. ER Diagram