Kitabevi Otomasyon Sistemi - Proje Raporu

Proje Adı: Kitabevi Otomasyon Sistemi
Hazırlayan: Halit Mert Artun (211307005), Turan Asgarlı (211307101), Umar Muskiev (211307103)
Tarih: 05.05.2025

I. Problem Tanımı

Geleneksel kitabevleri, envanter takibi, satış işlemleri, müşteri yönetimi ve raporlama gibi operasyonel süreçleri genellikle manuel veya yarı otomatik yöntemlerle yürütmektedir. Bu durum aşağıdaki temel problemlere yol açmaktadır:

- **Verimsizlik:** Manuel veri girişi ve takibi zaman alıcıdır ve insan hatasına açıktır. Stok sayımı, satış kaydı gibi işlemler operasyonel yükü artırır.
- **Stok Yönetimi Zorlukları:** Anlık ve doğru stok bilgisine ulaşmak zordur. Hangi kitapların azaldığını, hangilerinin popüler olduğunu veya hangilerinin uzun süredir satılmadığını belirlemek güçtür. Bu durum, gereksiz stok maliyetlerine veya satış kayıplarına neden olabilir.
- **Satış Analizi Eksikliği:** Manuel kayıtlardan anlamlı satış verileri (örn. en çok satan kitaplar, kategoriler, belirli dönemlerdeki satış trendleri) çıkarmak ve analiz etmek zordur. Bu da stratejik kararlar almayı zorlaştırır.
- **Müşteri İlişkileri Yönetimi (CRM) Yetersizliği:** Müşteri bilgilerini ve satın alma geçmişlerini düzenli takip etmek, sadık müşterileri tanımak ve onlara özel kampanyalar sunmak manuel sistemlerde pratik değildir.
- **Raporlama Güçlüğü:** İşletme performansını değerlendirmek için gerekli olan finansal ve operasyonel raporların (örn. günlük/aylık ciro, stok değeri) hazırlanması zahmetlidir.

Bu proje, yukarıda belirtilen problemleri çözmek amacıyla geliştirilmiştir. **Kitabevi Otomasyon Sistemi**, kitapların, yazarların, yayınevlerinin, kategorilerin, müşterilerin ve satışların dijital ortamda yönetilmesini sağlayan, web tabanlı bir platform sunarak kitabevi operasyonlarını modernize etmeyi ve verimliliği artırmayı hedefler. Sistem, anlık stok takibi, kolay satış işlemi, otomatik raporlama ve veri analizi gibi özelliklerle işletme sahiplerine ve çalışanlarına destek olmayı amaçlamaktadır.

II. Yapılan Araştırmalar

Projenin geliştirilmesi sürecinde çeşitli teknolojik araçlar ve yöntemler araştırılmış, karşılaşılan sorunlara çözümler üretilmiştir:

A. Web Framework Seçimi

Projenin temelini oluşturmak için Python tabanlı web framework'leri araştırılmıştır. Esnekliği, modüler yapısı (*Blueprint* desteği) ve geniş eklenti ekosistemi nedeniyle **Flask** tercih edilmiştir [1].

B. Veritabanı ve ORM

İlişkisel verileri (kitaplar, yazarlar, satışlar vb.) yönetmek için bir veritabanı gerekliliği ortaya çıkmıştır. Python uygulamalarıyla kolay entegrasyon sağlayan ve veritabanı işlemlerini Python nesneleri üzerinden yapmayı mümkün kılan **SQLAlchemy ORM (Object-Relational Mapper)** aracı seçilmiştir [2]. Bu sayede SQL sorguları yazma ihtiyacı minimize edilmiştir.

C. Arayüz Tasarımı

Kullanıcı dostu ve modern bir arayüz oluşturmak için popüler CSS framework'leri incelenmiştir. Geniş bileşen kütüphanesi, grid sistemi ve responsive (duyarlı) tasarım yetenekleri nedeniyle **Bootstrap 5** kullanılmıştır [3]. Ek olarak, özel stil düzenlemeleri için *style.css*

dosyası oluşturulmuş ve CSS değişkenleri ile tema (açık/koyu) desteği sağlanmıştır.

D. Dashboard Geliştirmeleri

Ana sayfada işletme durumu hakkında hızlı bilgi verecek bir dashboard oluşturulması hedeflenmiştir.

- **İstatistikler:** Toplam kitap, yazar, müşteri, satış sayısı gibi temel metrikler `SQLAlchemy count()` fonksiyonu ile veritabanından çekilmiştir.
- **Listeler:** "Son Satışlar", "Stok Uyarısı", "Son Eklenen Kitaplar", "En Çok Satan Kitaplar" gibi listeler için `SQLAlchemy` sorguları kullanılmıştır. Bu sorgularda `filter()`, `order_by()`, `limit()`, `join()`, `group_by()` ve `func.sum()` gibi fonksiyonlardan yararlanılmıştır.
- **Hata Ayıklama:** Sorgu geliştirme sırasında `AttributeError` (örn. var olmayan sütun adı kullanımı) ve `NameError` (örn. db veya model import eksikliği) gibi hatalarla karşılaşılmış ve ilgili Python dosyalarında düzeltmeler yapılmıştır.
- **Grafikler:** Satış trendlerini ve dağılımları görselleştirmek için JavaScript tabanlı grafik kütüphaneleri araştırılmıştır. Kullanım kolaylığı ve esnekliği nedeniyle **Chart.js** tercih edilmiştir [4]. Aylık satış verileri için bar grafik, kategori bazlı satış dağılımı için pasta grafik implemente edilmiştir. Gerekli veriler Flask route'undan Jinja2 şablonuna `tojson` filtresi ile aktarılmıştır.

E. Arama ve Filtreleme

Kitap listesi sayfasında kullanıcıların istedikleri kitaplara kolayca ulaşabilmesi için arama ve filtreleme işlevselliği eklenmiştir.

- **Arama:** `Flask request.args` ile alınan arama terimi (`q`), `SQLAlchemy or_` ve `ilike()` fonksiyonları kullanılarak birden fazla alanda (Kitap Adı, ISBN, Yazar Adı/Soyadı vb.) arama yapacak şekilde sorguya entegre edilmiştir.
- **Filtreleme:** Kategori ve yayınevi bazında filtreleme için HTML `<select>` elemanları kullanılmıştır. Seçilen değerler yine `request.args` ile alınmış ve `SQLAlchemy filter()` koşulları ile sorguya eklenmiştir. Sayfalama linklerinin de filtre parametrelerini koruması sağlanmıştır.

F. Örnek Veri Oluşturma

Uygulamayı test etmek ve geliştirmek amacıyla çok sayıda örnek veriye ihtiyaç duyulmuştur. Manuel veri girişi yerine, rastgele ve anlamlı veriler üretebilen **Faker** kütüphanesi [5] kullanılarak *seed_data.py* adında bir script oluşturulmuştur. Bu script, Flask uygulama bağlamını kullanarak veritabanına otomatik olarak yazar, yayınevi, kategori ve kitap eklemektedir.

G. Tema Desteği (Açık/Koyu)

Kullanıcı deneyimini iyileştirmek amacıyla tema değiştirme özelliği eklenmiştir.

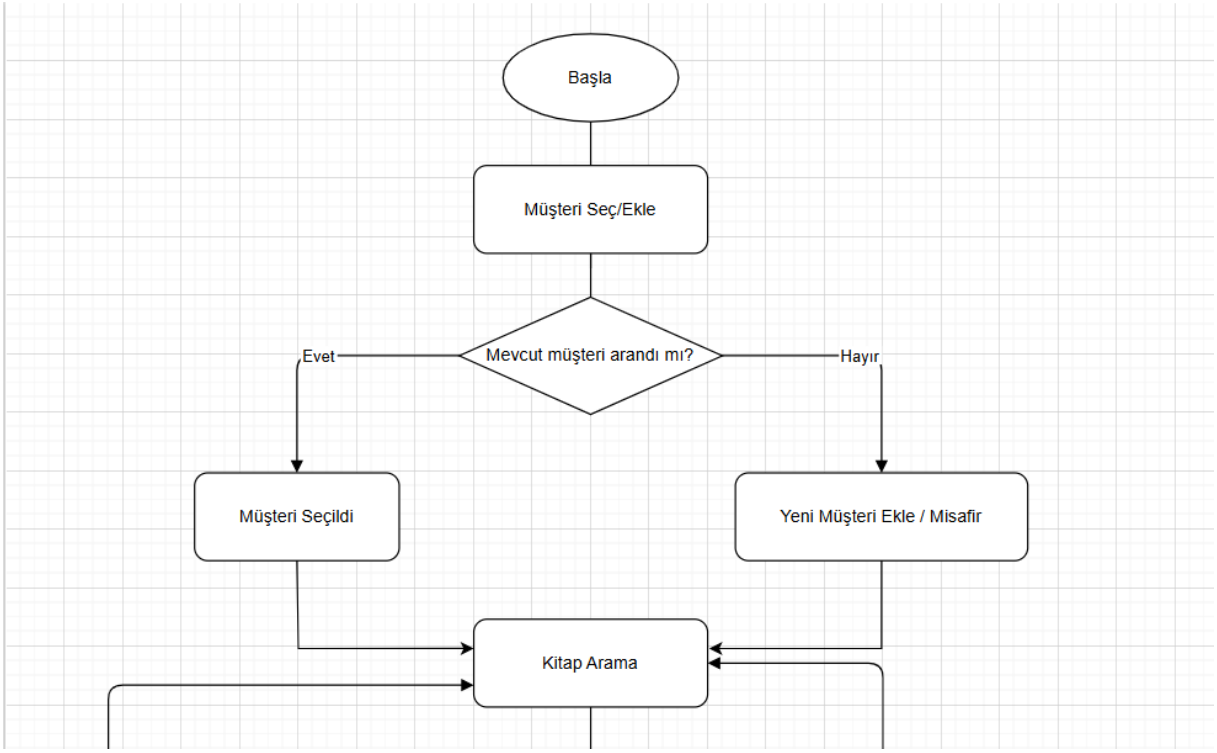
- **CSS Değişkenleri:** *style.css* dosyasında `:root` ve `[data-theme="dark"]` seçicileri altında renk paletleri tanımlanmıştır.
- **JavaScript & LocalStorage:** *base.html* şablonuna eklenen JavaScript kodu ile tema değiştirme düğmesine işlevsellik kazandırılmış, seçilen tema tarayıcının `localStorage`'ında saklanarak kalıcılık sağlanmıştır.

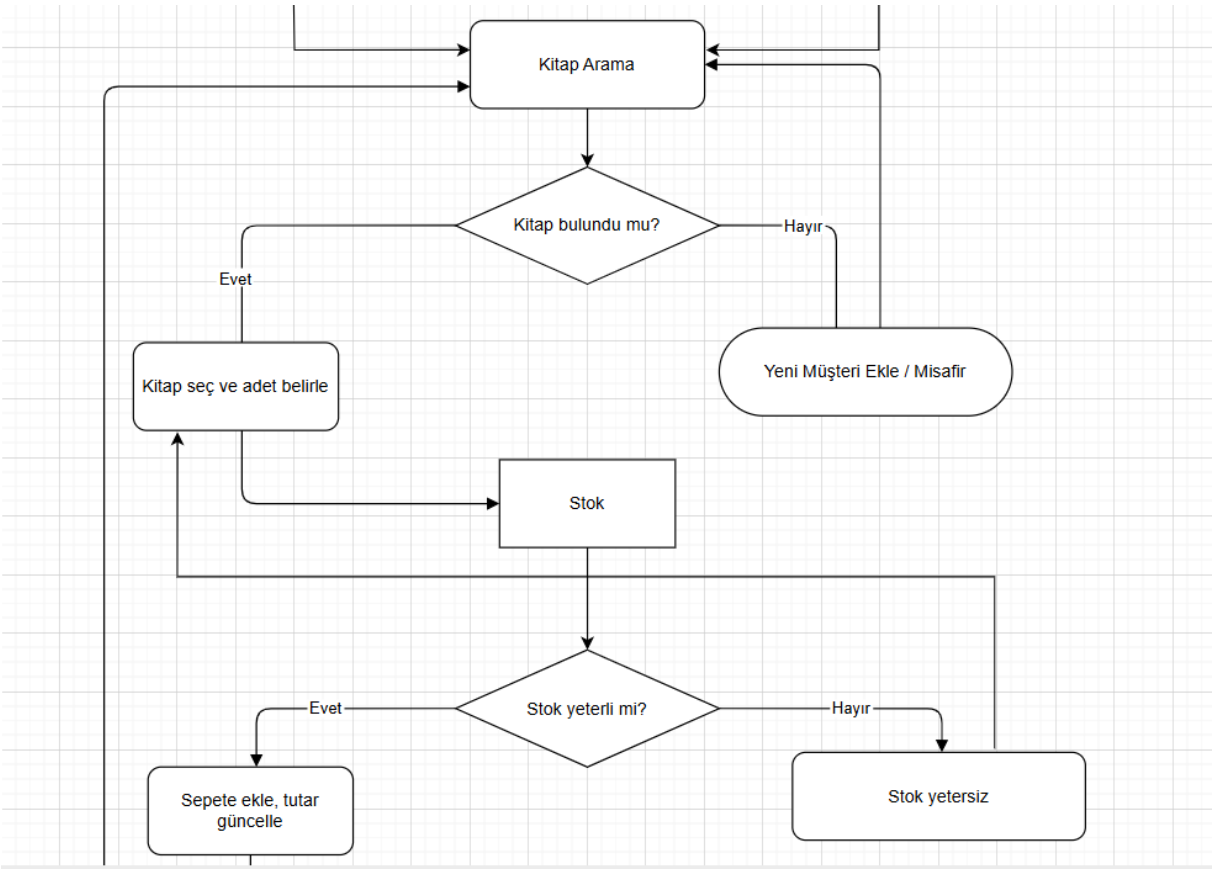
H. Diğer Araçlar

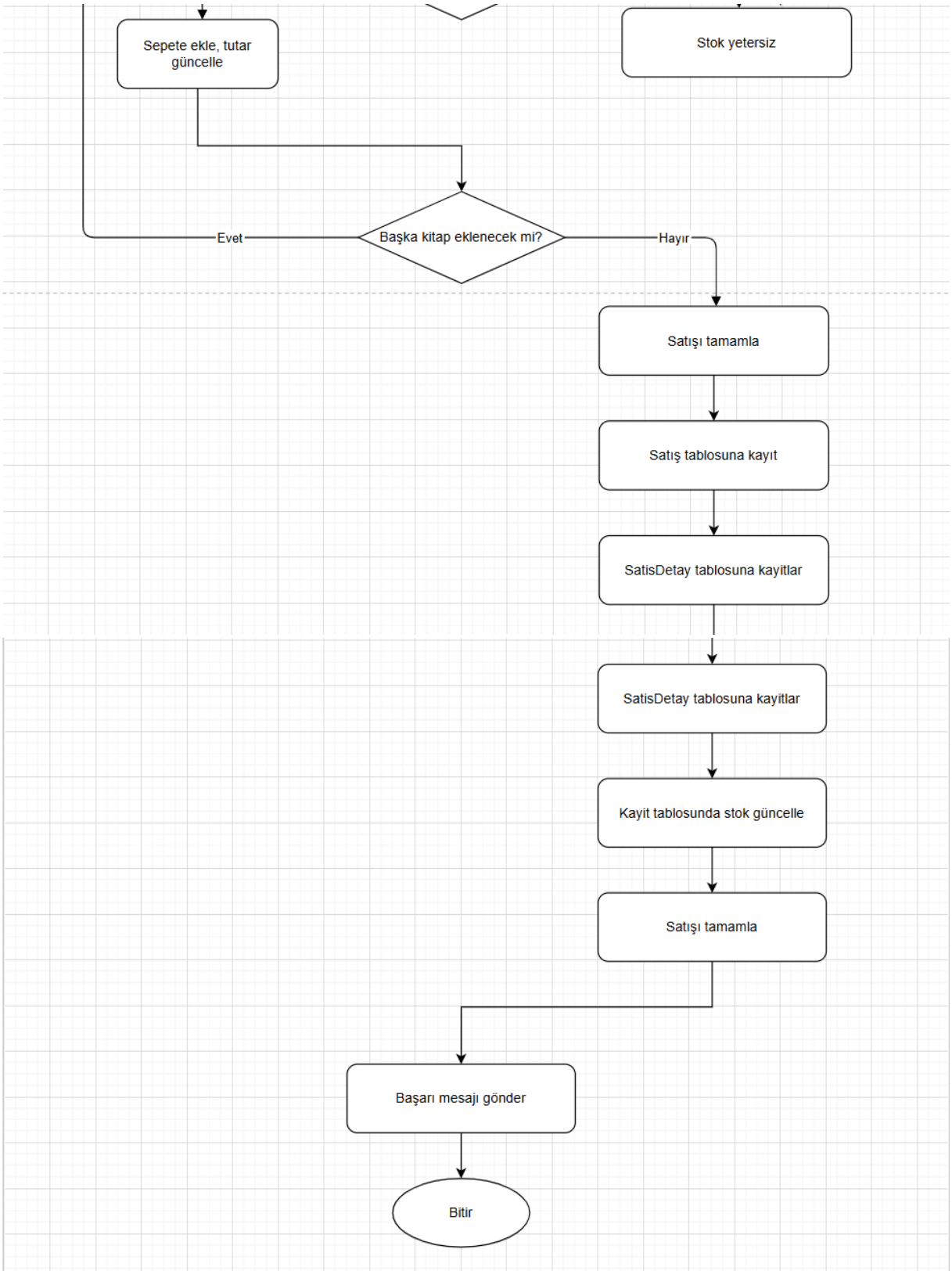
Projede ikonlar için **Font Awesome** [6] kullanılmıştır. Versiyon kontrolü için **Git** ve kod paylaşımı/takibi için **GitHub** platformu aktif olarak kullanılmıştır.

III. Akış Şeması

Bu bölümde projenin temel işlevlerinden biri olan “Yeni Satış Oluşturma” sürecinin akış şeması açıklanmaktadır.

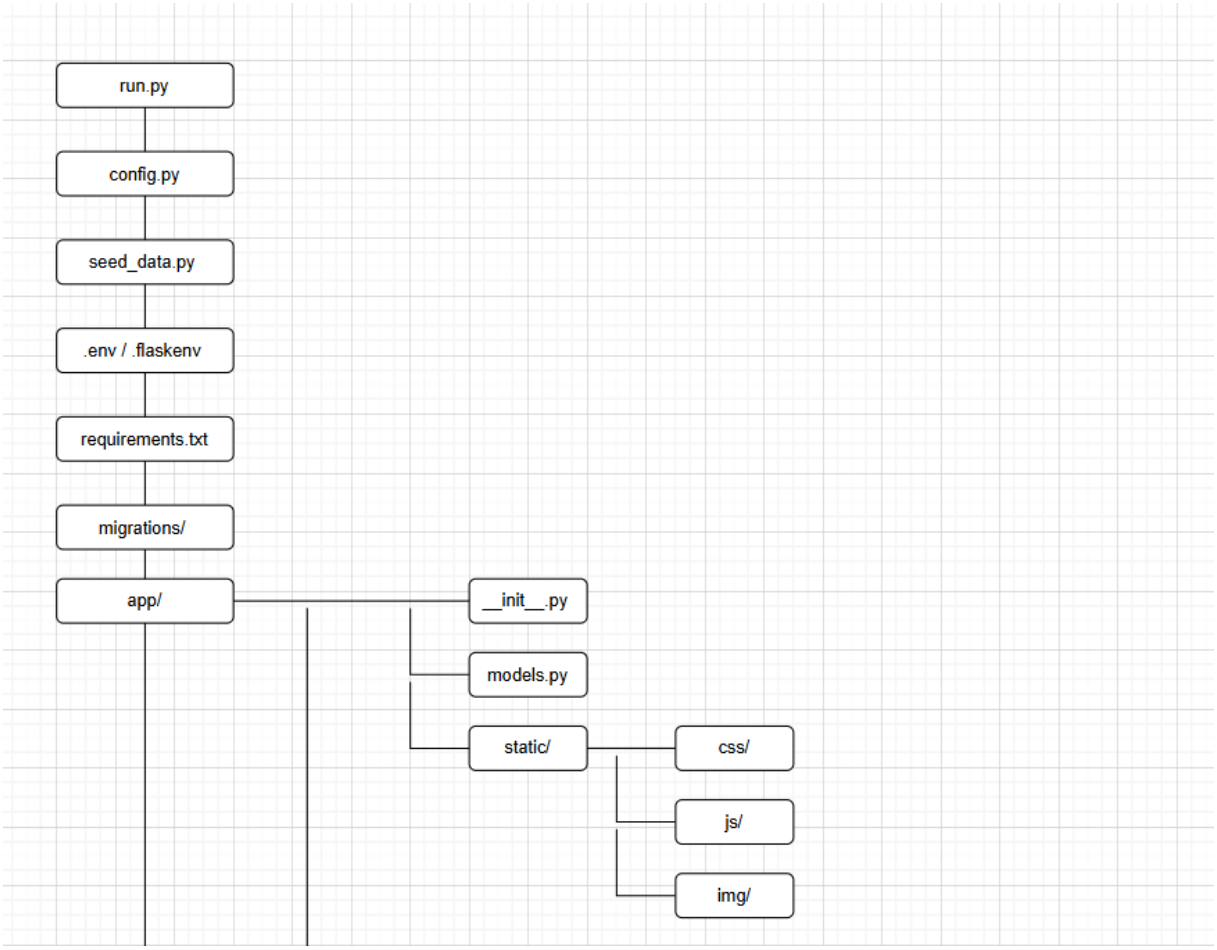


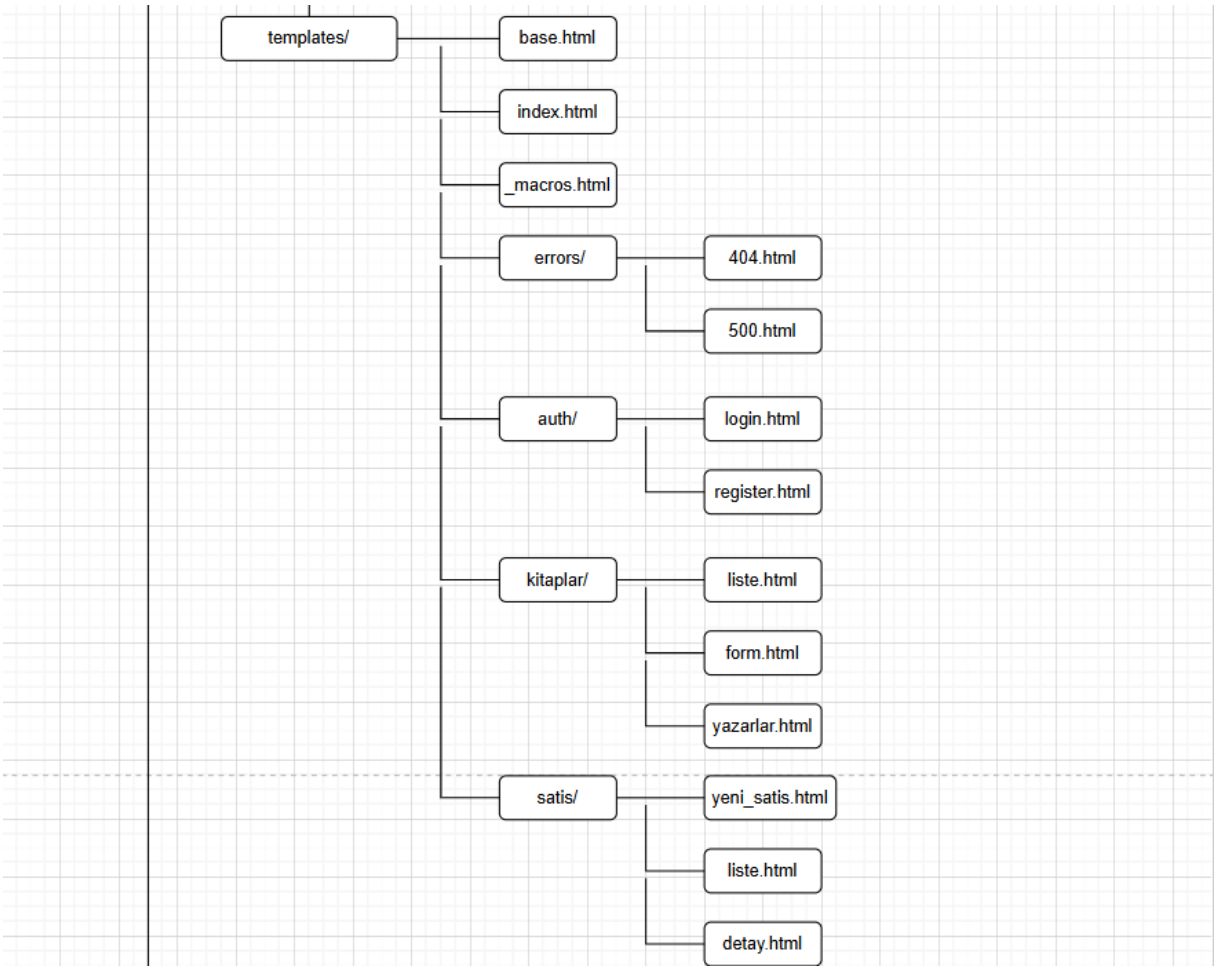

Kitap Arama
Kitap bulundu mu?
Hayır
Evet
Kitap seç ve adet belirle
Yeni Müşteri Ekle / Misafir
Stok
Stok yeterli mi?
Evet
Hayır
Sepete ekle, tutar güncelle
Stok yetersiz

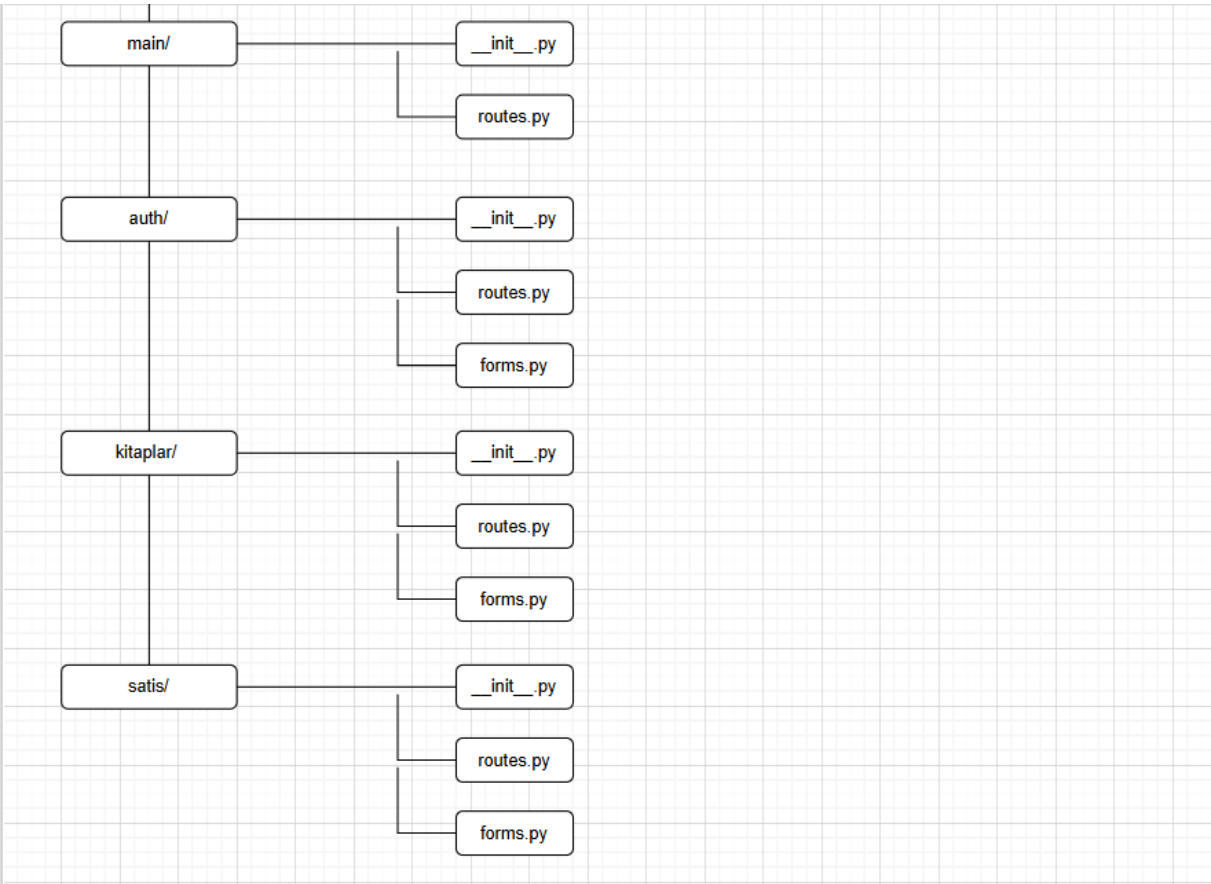

Sepete ekle, tutar güncelle
Stok yetersiz
Başka kitap eklenecek mi?
Evet
Hayır
Satışı tamamla
Satış tablosuna kayıt
SatisDetay tablosuna kayitlar
SatisDetay tablosuna kayitlar
Kayit tablosunda stok güncelle
Satışı tamamla
Başarı mesajı gönder
Bitir

IV. Yazılım Mimarisi

Proje, *Flask* web framework'ünün sağladığı *Blueprint* yapısı kullanılarak modüler bir mimaride geliştirilmiştir. Bu yaklaşım, uygulamanın farklı bölümlerini (kimlik doğrulama, kitap yönetimi, satış işlemleri vb.) ayrı modüller halinde organize etmeyi sağlar.


run.py
config.py
seed_data.py
.env / .flaskenv
requirements.txt
migrations/
app/
__init__.py
models.py
static/
css/
js/
img/


templates/
base.html
index.html
_macros.html
errors/
404.html
500.html
auth/
login.html
register.html
kitaplar/
liste.html
form.html
yazarlar.html
satis/
yeni_satis.html
liste.html
detay.html

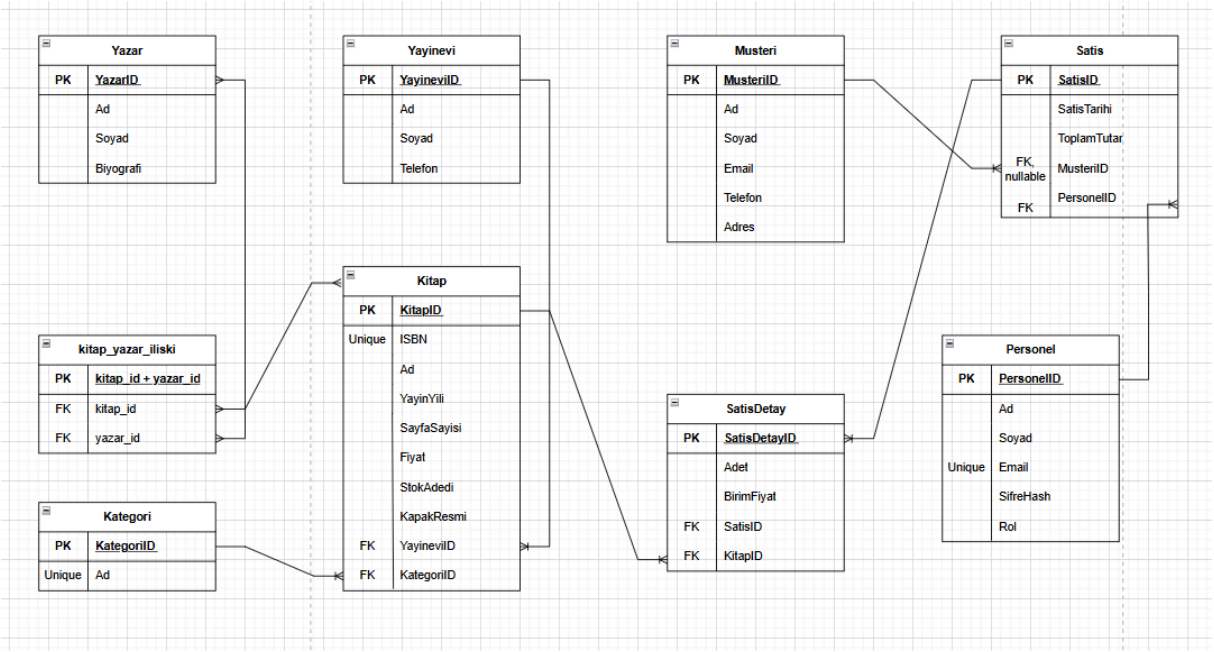


V. İşlem Akışı

1. Kullanıcı tarayıcıdan bir istek gönderir (örn. /kitaplar).
2. Flask, isteği ilgili Blueprint'teki (app/kitaplar) uygun route fonksiyonuna (index()) yönlendirir.
3. Route fonksiyonu, gerekirse request nesnesinden veri alır (örn. filtre parametreleri).
4. Fonksiyon, SQLAlchemy modelleri (Kitap, Kategori vb.) aracılığıyla veritabanı ile etkileşime girer (veri sorgulama, filtreleme).
5. Fonksiyon, alınan verileri ve diğer gerekli bilgileri render_template() fonksiyonu aracılığıyla ilgili Jinja2 şablonuna (kitaplar/liste.html) gönderir.
6. Jinja2 şablonu, verileri kullanarak HTML sayfasını oluşturur.
7. Oluşturulan HTML, Flask tarafından kullanıcıya yanıt olarak gönderilir.

VI. Veritabanı Diyagramı (ER Diagram)

Projenin veritabanı, SQLAlchemy ORM kullanılarak tasarlanmış ilişkisel bir yapıya sahiptir. Aşağıda tablolar ve aralarındaki ilişkiler açıklanmaktadır.



VII. Veritabanı Tasarımı

Tablolar (Varlıklar) ve Alanları:

1. **Yazarlar (Yazar)**
   - **YazarID** (Primary Key, Integer): Yazarın benzersiz kimliği.
   - **Ad** (String): Yazarın adı.
   - **Soyad** (String): Yazarın soyadı.
   - **Biyografi** (Text): Yazarın biyografisi (isteğe bağlı).
2. **Yayinevleri (Yayınevi)**
   - **YayineviID** (Primary Key, Integer): Yayınevinin benzersiz kimliği.
   - **Ad** (String, Unique): Yayınevinin adı (benzersiz olmalı).
   - **Adres** (Text): Yayınevinin adresi (isteğe bağlı).
   - **Telefon** (String): Yayınevinin telefonu (isteğe bağlı).
3. **Kategoriler (Kategori)**
   - **KategoriID** (Primary Key, Integer): Kategorinin benzersiz kimliği.
   - **Ad** (String, Unique): Kategori adı (benzersiz olmalı).
4. **Kitaplar (Kitap)**
   - **KitapID** (Primary Key, Integer): Kitabın benzersiz kimliği.
   - **ISBN** (String, Unique): Kitabın ISBN numarası (benzersiz olmalı).
   - **Ad** (String): Kitabın adı.
   - **YayinYili** (Integer): Kitabın yayın yılı (isteğe bağlı).
   - **SayfaSayisi** (Integer): Kitabın sayfa sayısı (isteğe bağlı).
   - **Fiyat** (Numeric): Kitabın fiyatı.
   - **StokAdedi** (Integer): Kitabın stok adedi (varsayılan 0).
   - **YayineviID** (Foreign Key → Yayinevleri.YayineviID): Kitabın yayınevi kimliği.
   - **KategoriID** (Foreign Key → Kategoriler.KategoriID): Kitabın kategori kimliği.
   - **KapakResmi** (String): Kitabın kapak resmi dosya

yolu (isteğe bağlı, varsayılan 'default_book_cover.jpg').

5. **Musteriler (Müşteri)**
    - **MusteriID** (Primary Key, Integer): Müşterinin benzersiz kimliği.
    - **Ad** (String): Müşterinin adı.
    - **Soyad** (String): Müşterinin soyadı.
    - **Email** (String, Unique): Müşterinin e-posta adresi (benzersiz, isteğe bağlı).
    - **Telefon** (String): Müşterinin telefonu (isteğe bağlı).
    - **KayitTarihi** (DateTime): Müşterinin kayıt tarihi (varsayılan o anki zaman).
6. **Personeller (Personel)** *(Aynı zamanda Kullanıcı (UserMixin))*
    - **PersonelID** (Primary Key, Integer): Personelin benzersiz kimliği.
    - **KullaniciAdi** (String, Unique): Personelin kullanıcı adı (benzersiz olmalı).
    - **SifreHash** (String): Personelin şifresinin hashlenmiş hali.
    - **Ad** (String): Personelin adı.
    - **Soyad** (String): Personelin soyadı.
    - **Rol** (String): Personelin rolü (örneğin, Admin, Kasiyer).
7. **Satislar (Satış)**
    - **SatisID** (Primary Key, Integer): Satış işleminin benzersiz kimliği.
    - **MusteriID** (Foreign Key → Musteriler.MusteriID): Satışın yapıldığı müşteri kimliği (isteğe bağlı, yani misafir satışı olabilir).
    - **SatisTarihi** (DateTime): Satışın yapıldığı tarih (varsayılan o anki zaman).
    - **ToplamTutar** (Numeric): Satışın toplam tutarı.
    - **PersonelID** (Foreign Key → Personeller.PersonelID): Satışı gerçekleştiren personel kimliği.
8. **SatisDetaylari (Satış Detayı)**
    - **SatisDetayID** (Primary Key, Integer): Satış detayının benzersiz kimliği.
    - **SatisID** (Foreign Key → Satislar.SatisID): Hangi satışa ait olduğu.
    - **KitapID** (Foreign Key → Kitaplar.KitapID): Satılan kitabın kimliği.
    - **Adet** (Integer): Satılan kitaptan kaç adet alındığı.
    - **BirimFiyat** (Numeric): Satış anındaki kitabın birim fiyatı.

İlişkiler:

- **Kitap ve Yazar (Çoka Çok):**
    - Kitaplar tablosu ile Yazarlar tablosu arasında *KitapYazarlari* adlı bir ara tablo (ilişki tablosu) üzerinden çoka çok (many-to-many) bir ilişki vardır.
    - KitapYazarlari tablosu **KitapID** ve **YazarID** alanlarını içerir ve bu iki alan birlikte birincil anahtardır.
    - Bir kitabın birden fazla yazarı olabilir ve bir yazar birden fazla kitap yazmış olabilir.
- **Kitap ve Yayınevi (Bire Çok):**
    - Yayinevleri tablosundan Kitaplar tablosuna doğru bire çok (one-to-many) bir ilişki vardır.
    - Bir yayınevi birden fazla kitap yayınlayabilir (*Yayinevi.kitaplar*).

    - Bir kitap sadece bir yayınevine aittir (*Kitap.yayinevi*).
- **Kitap ve Kategori (Bire Çok):**
    - Kategoriler tablosundan Kitaplar tablosuna doğru bire çok (one-to-many) bir ilişki vardır.
    - Bir kategori birden fazla kitaba sahip olabilir (*Kategori.kitaplar*).
    - Bir kitap sadece bir kategoriye aittir (*Kitap.kategori*).
- **Kitap ve Satış Detayı (Bire Çok):**
    - Kitaplar tablosundan SatisDetaylari tablosuna doğru bire çok (one-to-many) bir ilişki vardır.
    - Bir kitap birden fazla satış detayında yer alabilir (*Kitap.satis_detaylari*).
    - Bir satış detayı sadece bir kitaba aittir (*SatisDetay.kitap*).
- **Müşteri ve Satış (Bire Çok):**
    - Musteriler tablosundan Satislar tablosuna doğru bire çok (one-to-many) bir ilişki vardır.
    - Bir müşteri birden fazla satış işlemi yapabilir (*Musteri.satislar*).
    - Bir satış işlemi (eğer müşteri kayıtlıysa) bir müşteriye aittir (*Satis.musteri*).
- **Personel ve Satış (Bire Çok):**
    - Personeller tablosundan Satislar tablosuna doğru bire çok (one-to-many) bir ilişki vardır.
    - Bir personel birden fazla satış işlemi gerçekleştirebilir (*Personel.satislar*).
    - Bir satış işlemi bir personel tarafından gerçekleştirilir (*Satis.personel*).
- **Satış ve Satış Detayı (Bire Çok):**
    - Satislar tablosundan SatisDetaylari tablosuna doğru bire çok (one-to-many) bir ilişki vardır.
    - Bir satış işlemi birden fazla satış detayından (yani birden fazla farklı kitaptan veya aynı kitaptan farklı adetlerde) oluşabilir (*Satis.detaylar*).
    - Bir satış detayı sadece bir satış işlemine aittir (*SatisDetay.satis*).
    - Satış silindiğinde ilişkili SatisDetaylari da silinir *(cascade='all, delete-orphan')*.

VIII. Genel Yapı

Kitabevi Otomasyon Sistemi, Python tabanlı Flask framework'ü üzerine inşa edilmiş, modüler ve genişletilebilir bir web uygulamasıdır. SQLAlchemy ORM aracılığıyla ilişkisel veritabanı ile etkileşim kurar ve veritabanı şema yönetimi için Flask-Migrate kullanır. Kullanıcı arayüzü, Bootstrap 5 ve özel CSS stilleri ile oluşturulmuş olup, modern, duyarlı ve tema (açık/koyu) desteği sunan bir tasarıma sahiptir.

Temel Özellikler:

- **Kullanıcı Yönetimi:** Güvenli personel girişi (Flask-Login), parola hashleme ve rol tabanlı yetkilendirme (Admin rolü personel ekleyebilir).
- **Katalog Yönetimi:** Kitaplar, Yazarlar, Yayınevleri ve Kategoriler için tam CRUD (Oluşturma, Okuma, Güncelleme, Silme) işlemleri. Kitaplar ve yazarlar arasında çoktan çoğa ilişki desteği.
- **Stok Takibi:** Kitapların stok adetleri tutulur ve satış yapıldıkça otomatik olarak güncellenir. Kritik stok seviyesindeki kitaplar dashboard'da ve kitap listesinde belirtilir.
- **Satış Yönetimi:** Müşteri seçimi (veya misafir müşteri), sepet oluşturma (kitap arama ve ekleme), toplam tutar

hesaplama ve satış kaydı oluşturma işlevleri. Satış detayları ve geçmiş satışlar listelenebilir, PDF çıktısı alınabilir.
- **Müşteri Yönetimi:** Müşteri bilgilerinin kaydedilmesi ve listelenmesi.
- **Dashboard:** Sistemin genel durumu hakkında özet bilgiler sunar: temel istatistikler (toplam kitap, yazar vb.), son satışlar, stok uyarıları, son eklenen kitaplar, en çok satan kitaplar. Ayrıca, aylık satış trendlerini gösteren bar grafik ve kategori bazlı satış dağılımını gösteren pasta grafik içerir (Chart.js).
- **Arama ve Filtreleme:** Kitap listesi sayfasında kapsamlı arama (kitap adı, ISBN, yazar vb.) ve kategori/yayınevi bazında filtreleme imkanı.
- **Örnek Veri:** Uygulamayı test etmek için Faker kütüphanesi kullanılarak oluşturulan seed_data.py script'i.
- **Tema Desteği:** Kullanıcının tercihine göre açık ve koyu tema arasında geçiş yapma imkanı (CSS Değişkenleri ve LocalStorage).

Proje yapısı, Flask Blueprints kullanılarak modüler hale getirilmiştir, bu da gelecekte yeni özellikler eklemeyi veya mevcutları güncellemeyi kolaylaştırır. Kod okunabilirliği ve bakımı için PEP 8 standartlarına uyulmaya çalışılmıştır.

IX. Referanslar

Projenin geliştirilmesi sırasında aşağıdaki kaynaklardan ve teknolojilerden yararlanılmıştır:

1. **Flask Documentation:** https://flask.palletsprojects.com/
2. **SQLAlchemy Documentation:** https://www.sqlalchemy.org/
3. **Bootstrap Documentation:** https://getbootstrap.com/docs/5.2/getting-started/introduction/
4. **Chart.js Documentation:** https://www.chartjs.org/docs/latest/
5. **Faker Documentation:** https://faker.readthedocs.io/
6. **Font Awesome:** https://fontawesome.com/
7. **Jinja Documentation:** https://jinja.palletsprojects.com/
8. **Flask-Login Documentation:** https://flask-login.readthedocs.io/
9. **Flask-SQLAlchemy Documentation:** https://flask-sqlalchemy.palletsprojects.com/
10. **Flask-Migrate Documentation:** https://flask-migrate.readthedocs.io/
11. **MDN Web Docs (JavaScript, CSS):** https://developer.mozilla.org/
12. **Stack Overflow:** Çeşitli teknik sorunların çözümü için başvurulan topluluk forumu. https://stackoverflow.com/