KARADENİZ TEKNİK ÜNİVERSİTESİ
OF TEKNOLOJİ FAKÜLTESİ
YAZILIM MÜHENDİSLİĞİ

KÜTÜPHANE YÖNETİM SİSTEMİ SAVUNMA RAPORU

HAZIRLAYAN

ADI SOYADI: Halil KAYA
ÖĞRENCİ NUMARASI: 445891
TESLİM TARİHİ: 29 / 12 / 2025

DERS ADI: Veri Tabanı Yönetim Sistemi
DERS YÜRÜTÜCÜSÜ: Hakan AYDIN

İÇİNDEKİLER

1.PROJENİN AMACI ve KAPSAMI

Bu projenin amacı, bir kütüphanede gerçekleştirilen kitap işlemleri ve ödünç alma işlemlerinin dijital bir ortamda yapılmasını sağlayan bir kütüphane yönetim sistemi geliştirilmektir. Geliştirilen sistemle birlikte kütüphanelerde kitap arama, ödünç alma gibi işlemlerin daha verimli bir şekilde gerçekleştirilmesi, verilerin doğru ve tutarlı şekilde saklanması ardından kolayca takip edilebilmesi hedeflenmiştir. Ayrıca, geç iade edilen kitaplar için ceza hesaplaması sistem tarafından otomatik bir şekilde yapılması ve kayıt altına alması amaçlanmıştır. Yetkili kullanıcılar ise kitap, yazar, kategori gibi verilerin eklenme, düzenlenme ve silme işlemlerini yapabilmektedir.

Bu proje kapsamında, ilişkisel veri tabanı tasarımı yapılmış, MySQL tabanlı CRUD işlemleri uygulanmış, JWT tabanlı kimlik doğrulama mekanizması kullanılmış ve backend tarafında REST API geliştirilmiştir. Kullanıcıların ve adminler, kütüphane işlemlerini web sitesi üzerinden güvenli bir şekilde gerçekleştirebilmektedir.

2.KULLANILAN TEKNOLOJİLER

2.1 Backend

Backend tarafında proje, Python kütüphanesi olan Flask framework'ü kullanılarak geliştirilmiştir. Backend, REST mimarisi prensipleriyle tasarlanarak sistemin daha güvenli, kontrollü ve okunur olması hedeflenmiştir.

Geliştirilen backend yapısında katmanlı mimari benimsenmiştir. Mimari yapı kapsamında Model katmanı, veri tabanında yer alan verileri nesneye dönüştürmekte; Repository katmanı, veri tabanı işlemlerini gerçekleştirmektedir. Service katmanı, iş kurallarının ve uygulama mantığının yürütüldüğü katman olup Routes katmanı ise istemciden gelen endpoint isteklerini çağırarak uygun servislere yönlendirmektedir.

Routes katmanında, temel CRUD işlemlerini gerçekleştiren API uç noktaları oluşturulmuştur (Örneğin: /kitaplarim-veri, /odunc-al vb.). Bu uç noktalar sayesinde veriler üzerinde işlemler gerçekleştirilmektedir. Ayrıca, sistemde güvenli erişim sağlamak aracıyla JWT tabanlı kimlik doğrulama mekanizması kullanılmıştır.

```
185  @adminBp.route("/admin/cezalar/veri")
186  @jwt_required()
187  def getCezalar():
188      fines = DelayFineRepository.getAll()
189      return jsonify({"success": True, "data": [{
190          "ID": f.ID,
191          "userID": f.userID,
192          "bookID": f.bookID,
193          "delay": f.delay,
194          "sum": f.sum,
195          "firstDay": f.firstDay
196      } for f in fines]})
```

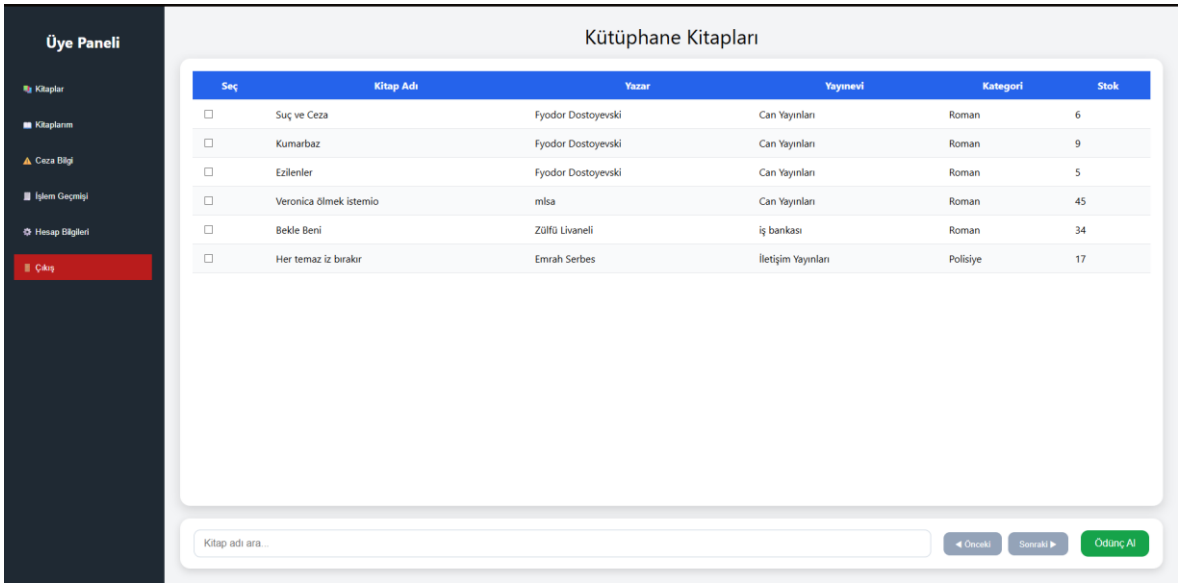
Görsel2.1: adminRoutes katmanında örnek bir Route kodu.

2.2 Frontend

Kullanıcıların sistemle etkileşimde bulunabilmesini sağlamak amacıyla web tabanlı bir kullanıcı arayüz geliştirilmiştir. Frontend katmanı, backend tarafında geliştirilen REST API uç noktaları ile sistemin işlevlerini kullanıcıya görsel ve etkileşimli bir şekilde sunulması sağlanmıştır.

Frontend geliştirme sürecinde HTML, CSS ve JavaScript teknolojileri kullanılmıştır. Bu kapsamda kullanıcıların sisteme giriş yapabilmesi, kitap ödünç alma ve iade işlemleri gibi birçok işlem için kullanıcı arayüzleri tasarlanmıştır. Ayrıca admin yetkisine sahip kullanıcılar için kütüphane işlemlerini gerçekleştirebilmeleri olanak tanıyan ek sayfalar oluşturulmuştur (Örneğin: yazarlar.html, kategoriler.html vb.).

Frontend katmanı, backend servislerinden alınan verileri dinamik olarak ekrana yansıtarak sistemin gerçek zamanlı ve etkileşimli bir şekilde çalışmasını sağlamaktadır.



Görsel 2.2: kitaplarim.html sayfası örnek kullanıcı arayüzü (admin yetkisi olmayan)

2.3 Veri Tabanı

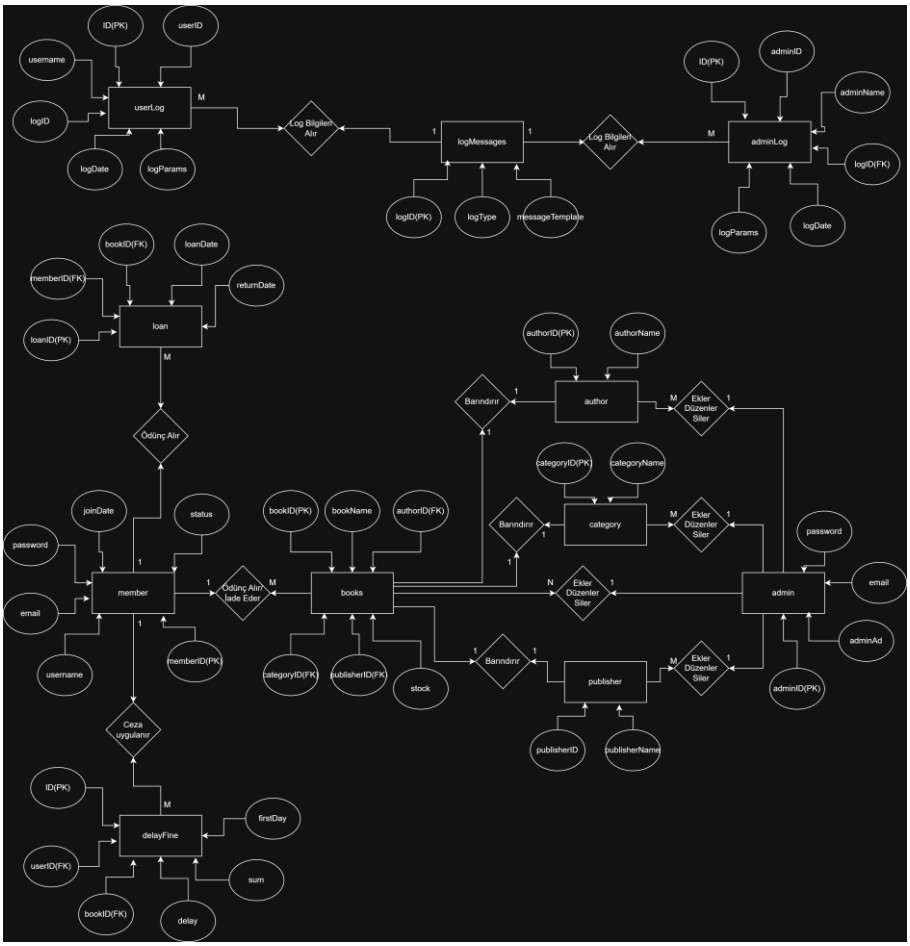
Bu projede, verilerin kalıcı, güvenli ve düzenli bir şekilde saklanabilmesi amacıyla MySQL veri tabanı kullanılmıştır. Veri tabanı tasarımı; sistemde yer alan kitaplar, kullanıcılar, yazarlar, kategoriler, yayınevleri, ödünç alma, ceza işlemleri ve log bilgilerini kapsayacak şekilde oluşturulmuştur.

Veri tabanı içerisinde tablolar arası ilişkiler tanımlanarak veri bütünlüğünün sağlanması hedeflenmiştir. Bu kapsamda, birincil anahtar (Primary Key) ve yabancı anahtar (Foreign Key) yapıları kullanılarak tablolar arasında bağlantılar kurulmuştur. Ayrıca verilerin yönetimi ve işlenmesi amacıyla backend tarafında CRUD işlemleri gerçekleştirilmiştir.

Sistem içerisinde, ödünç alınan kitapların iade sürelerinin takip edilebilmesi ve geç iade durumlarında ceza işlemlerinin otomatik uygulanabilmesi amacıyla TRIGGER ve STORED PROCEDURE kullanılmıştır (Örneğin: tryDelayFineCal). Bu sayede veri tabanı işlemleri otomatikleştirilmiş, sistemin daha stabil, tutarlı ve güvenilir bir şekilde çalışmasını sağlanmıştır.

Geliştirilen veri tabanı yapısı, backend ile uyumlu bir şekilde çalışarak uygulamanın ihtiyaç duyduğu verilerin güvenli ve hızlı bir biçimde erişilmesini mümkün kılmaktadır.

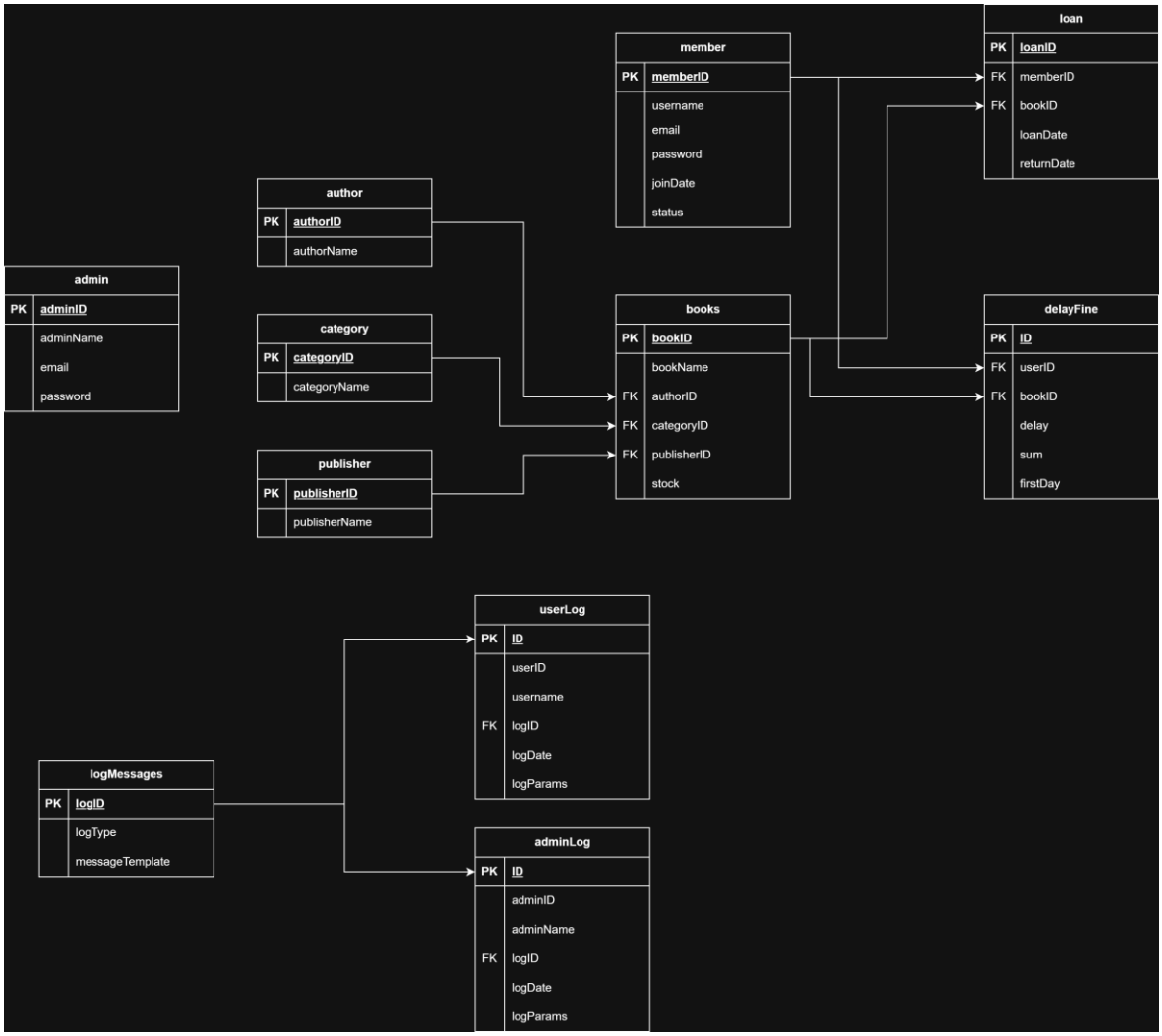
Aşağıda, veri tabanına ait ER diyagramları ve tablo ilişkileri sunulmuştur.



Görsel 2.3: ER diyagramı

Görselde sunulan ER diyagramında, tablo ve ilişkiler sistemin işleyişine uygun bir biçimde modellenmiş olup, veri tabanı üzerindeki işlemler açık ve anlaşılır şekilde gösterilmiştir. logMessages tablosunda yer alan userLog ve adminLog alanında bulunan userID ve adminID verileri için yabancıl anahtar (Foreign Key) kısıtlaması tanımlanmamıştır. Bu tasarım yaklaşımı ile, kullanıcı veya yönetici kayıtları silinse dahi log verilerinin korunması ve veri kaybının önlenmesi amaçlanmıştır.

Oluşturulan bu ER diyagramı sayesinde veri tabanındaki yapısının daha anlaşılır hale getirilmesi hedeflenmiştir.



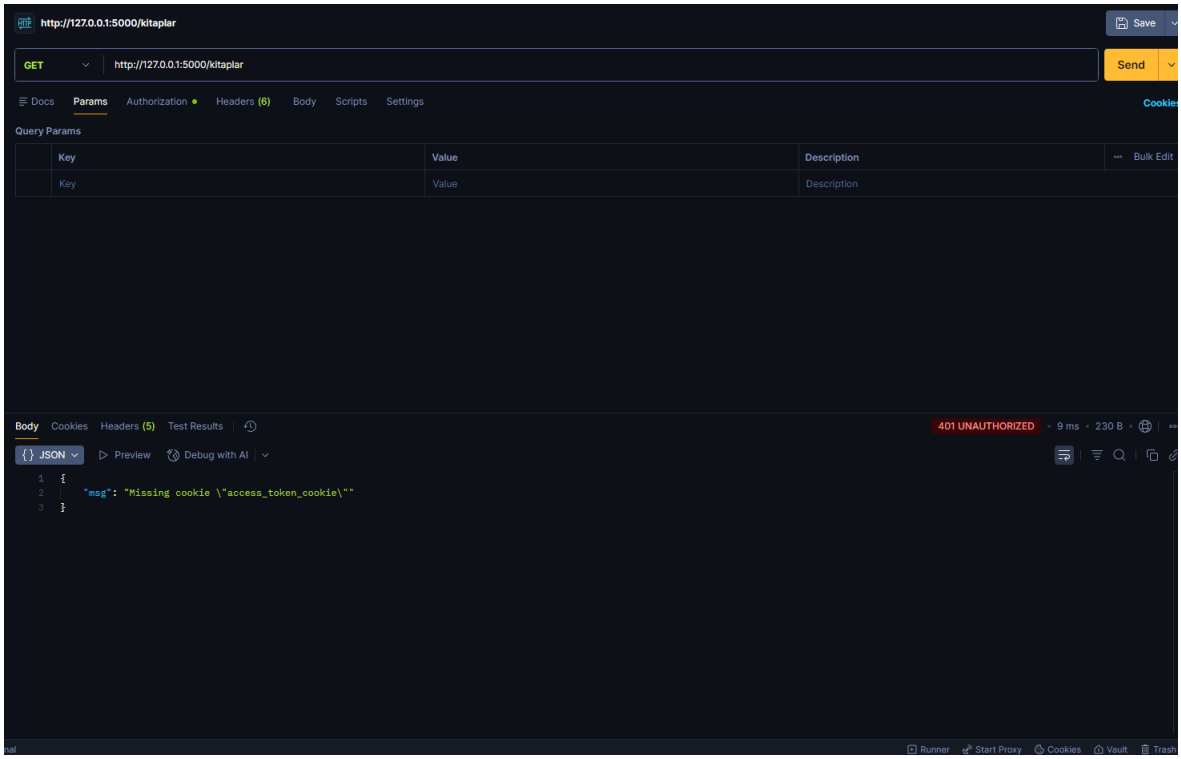
Görsel 2.4: ER diyagramı tablo şeklinde gösterilmiş hali

3. TEST SÜRECİ (POSTMAN)

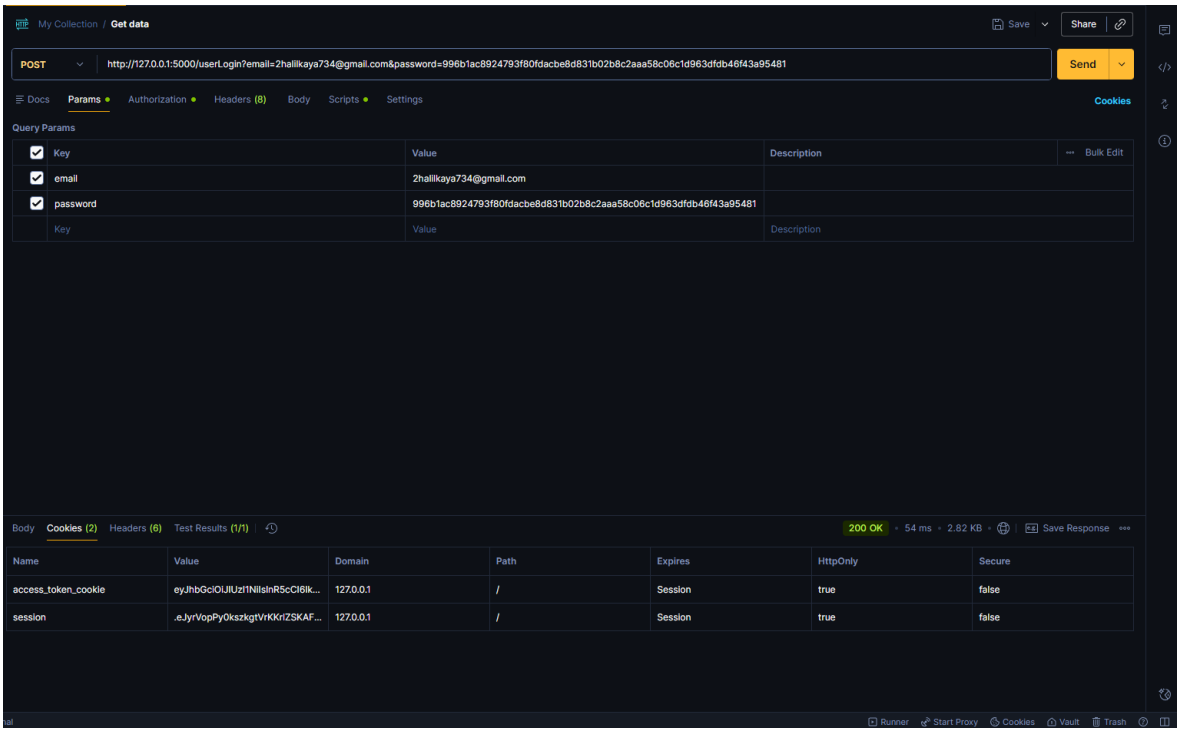
Geliştirilen backend servislerinin doğru ve beklenen bir şekilde çalıştığının doğrulanması amacıyla, REST API uç noktaları Postman aracı kullanılarak test edilmiştir. Postman üzerinden yapılan testlerde, sistemde yer alan tüm temel işlemler için ayrı ayrı endpoint istekleri gönderilmiştir.

Test süreci kapsamında; kitap, kullanıcı, yazar ve kategori gibi varlıklar için GET ve POST istekleri kullanılarak CRUD işlemlerinin doğru şekilde gerçekleştirildiği gözlemlenmiştir. Ayrıca kullanıcı giriş işlemleri test edilerek, kimlik doğrulama mekanizmasının doğru çalıştığı ve yetkisiz erişimlerin engellendiği doğrulanmıştır.

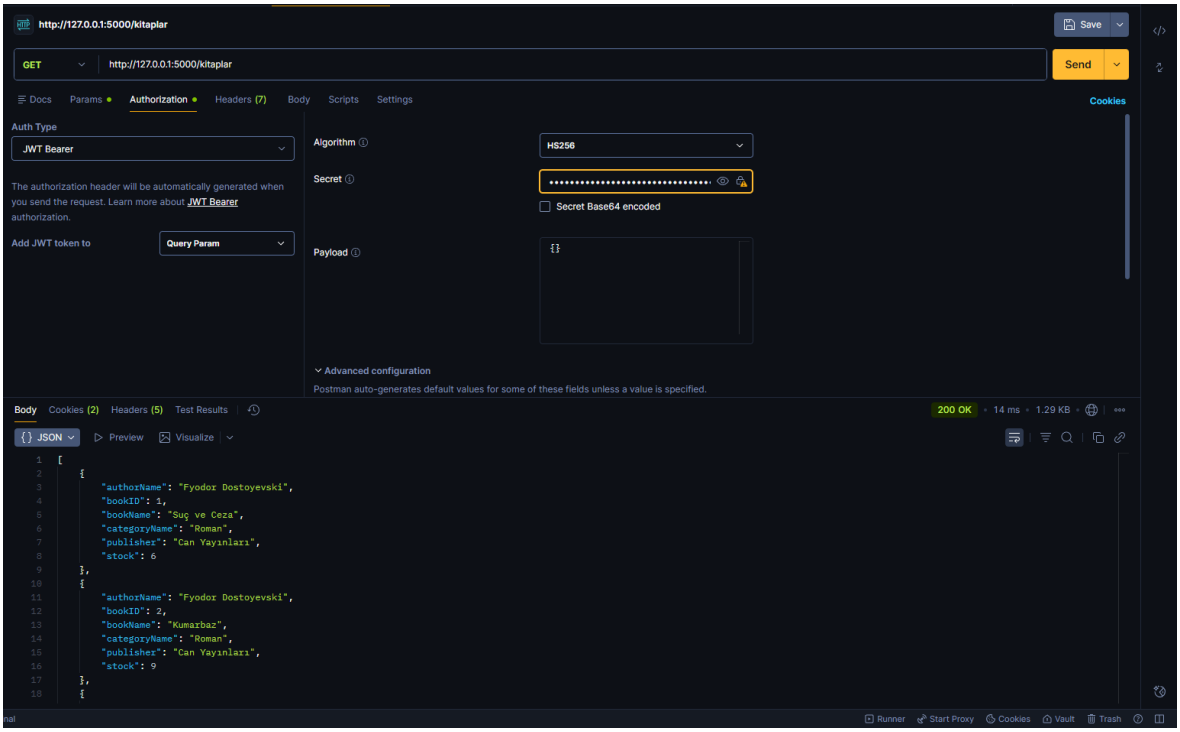
Gerçekleştirilen bu testler sonucunda, uygulamanın fonksiyonel gereksinimleri karşılandığı ve sistemin stabil bir şekilde çalıştığı teyit edilmiştir. Aşağıda, Postman üzerinden gerçekleştirilen testlere ait örnek ekran görüntüleri sunulmuştur.



Görsel 3.1: *JWT token bulunmadığı için kullanıcı girişi yapılmadan kitap verilerine erişim engellenmiştir.*



Görsel 3.2: Kullanıcı girişi sonrası JWT token, cookie içerisinde başarıyla alınması.



Görsel 3.3: JWT doğrulaması sonrası kitap verilerinin başarılı şekilde alınması.