

# **AKILLI KÜTÜPHANE YÖNETİM SİSTEMİ**

## **PROJE RAPORU**

**Üniversite:** Karadeniz Teknik Üniversitesi

**Fakülte:** Of Teknoloji Fakültesi

**Bölüm:** Yazılım Mühendisliği

**Ders Adı:** Veritabanı ve Yönetimi

**Proje Adı:** Akıllı Kütüphane Sistemi

**Öğrenci Adı:** Nehir Buket Polat

**Öğrenci No:** 445898

## **1-PROJENİN AMACI**

Bu projenin amacı, bir kütüphanede gerçekleşen kitap yönetimi, kullanıcı yönetimi, ödünç alma-iade işlemleri ve ceza hesaplamalarının bilgisayar ortamında güvenli, düzenli ve merkezi bir sistem üzerinden yürütülmesini sağlamaktır.

Geliştirilen sistem:

- **Admin** ve **User (Kullanıcı)** rollerini ayırrı
- Yetkilendirmeyi JWT (JSON Web Token) ile sağlar
- **RESTful API** mimarisini kullanır
- Frontend (HTML + JS) ile Backend (Flask) katmanlarını ayırrı
- **PostgreSQL** veritabanı üzerinde çalışır

Bu sayede hem gerçek hayatı kütüphane işleyişi modellenmiş hem de modern web uygulama mimarisi uygulanmıştır.

## **2-KULLANILAN TEKNOLOJİLER**

### **Backend**

- Python
- Flask
- Flask-JWT-Extended
- Flask-SQLAlchemy
- Flask-Migrate

### **Veritabanı**

- PostgreSQL
- İlişkisel veritabanı yapısı
- Foreign Key ilişkileri

## **Frontend**

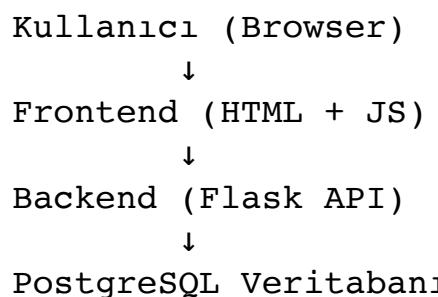
- HTML5
- CSS
- JavaScript (Fetch API)

## **Test Araçları**

- **Postman** (API testleri)
- **pgAdmin** (veritabanı kontrolü)

## **3-SİSTEM MİMARİSİ**

Proje **3 katmanlı mimari** ile geliştirilmiştir:



### **Katmanlar arası iletişim:**

- Frontend → Backend: `fetch()` + JSON
- Backend → Veritabanı: SQLAlchemy ORM
- Yetkilendirme: JWT Token

## 4-VERİTABANI TASARIMI

### Tablolar

#### kullanici

Alan	Açıklama
id	Kullanıcı ID
ad	Kullanıcı adı
email	Giriş için kullanılan e-posta
sifre	Hashlenmiş parola
rol	admin / user

#### kitap

Alan	Açıklama
id	Kitap ID
ad	Kitap adı
yazar_id	Yazar ilişkisi
kategori_id	Kategori ilişkisi
stok	Mevcut stok

#### odunc

Alan	Açıklama
id	Ödünç ID
kullanici_id	Kitabı alan kullanıcı
kitap_id	Alınan kitap
baslangic_tarih	Ödünç tarihi
iade_tarih	Son iade tarihi
teslim_edildi	Durum bilgisi

## **ceza**

<b>Alan</b>	<b>Açıklama</b>
id	Ceza ID
odunc_i_d	Hangi ödünçten doğduğu
miktar	Ceza tutarı
tarih	Ceza tarihi
aciklama	Ceza açıklaması

## **5-KİMLİK DOĞRULAMA ve GÜVENLİK (JWT)**

### **JWT Kullanımı**

- Kullanıcı giriş yaptığında /api/login endpoint'i çalışır
- Doğru bilgiler girilirse **JWT token** üretilir
- Token frontend'de **localStorage**'da saklanır
- Yetkili endpoint'lerde `@jwt_required()` kullanılır

### **Avantajları:**

- Stateless (sunucuda session tutulmaz)
- Güvenli
- Rol bazlı kontrol yapılabilir

## **6-ROL BAZLI SİSTEM**

### **Admin Yetkileri**

- Kitap ekleme
- Tüm ödüncleri görüntüleme
- Tüm cezaları görüntüleme
- Kullanıcıları listeleme

### **Kullanıcı Yetkileri**

- Kitapları görüntüleme
- Kitap ödünç alma
- Kendi ödünclerini görme
- Kendi cezalarını görme

Kullanıcılar sadece kendilerine ait verileri görebilir.

## **7-ÖDÜNÇ ALMA MEKANİZMASI**

1. Kullanıcı kitap listesinden kitap seçer
2. /api/oduncler → POST isteği gönderilir
3. Stok kontrolü yapılır
4. Stok varsa:
  - Ödünç kaydı oluşturulur
  - Kitap stoğu 1 azaltılır
  - İade tarihi otomatik hesaplanır (7 gün)

## **8-İADE ve CEZA SİSTEMİ**

- **Kullanıcı kitabı iade ettiğinde:**
  - Teslim durumu güncellenir
  - Kitap stoğu artırılır
- **Iade tarihi geçmişse:**
  - Gecikme gün sayısı hesaplanır
  - $\text{Gün} \times 5 \text{ TL}$  şeklinde ceza oluşturulur
  - Ceza tablosuna otomatik kayıt eklenir

Bu sistem tamamen backend tarafında otomatik çalışır.

## **9-FRONTEND SAYFALARI**

### **Login**

- Kullanıcı giriş ekranı
- JWT token alınır

### **User Paneli**

- Dashboard (Ödünç sayısı, ceza sayısı)
- Kitaplar
- Ödünçlerim
- Cezalarım

### **Admin Paneli**

- Kitap ekleme
- Genel ödüncüler
- Genel cezalar

## 10-API TESTLERİ (POSTMAN)

Projede tüm API uç noktaları **Postman ile test edilmiştir.**

Test edilen başlıca endpoint'ler:

- POST /api/login
- GET /api/kitaplar
- POST /api/odunceler
- GET /api/user/odunceler
- PUT /api/odunceler/{id}/iade
- GET /api/user/cezalar

Tüm test sonuçları Postman Collection olarak kaydedilmiştir ve sunumda gösterilmeye uygundur.

The screenshot shows the Postman application interface. At the top, there's a search bar and various navigation icons. Below it, a header bar indicates an 'Overview' section, a 'POST http://127.0.0.1:5000/api/login' request, and no environment selected. The main area shows a 'POST' method selected for a request to 'http://127.0.0.1:5000/api/login'. The 'Body' tab is active, showing raw JSON data:

```
1 {
2   "email": "user@test.com",
3   "sifre": "1234"
4 }
```

Below the body, the response section shows a status of '200 OK' with a response time of 230 ms and a response size of 615 B. The response body is displayed in JSON format:

```
1 {
2   "kullanici_id": 4,
3   "mesaj": "Giriş başarılı!",
4   "rol": "user",
5   "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJMcVzaCI6ZmFsc2UsImhdCI6MTc2NjQyMzQzOSwianRpIjoiNTFjNjYyZmYtMGFmMy00NTE3LWI0MTUtOWU0YzY3OTk4ZjM4IiwidHlwZSI6ImFjY2VzcylsInN1YiI6IjQ1LCJuYmYiOjE3NjY0MjM0Mzs1mNzcmYiOii2NjI2ZDdkMy1i0WRiLTRhNTkt0TY2Zi020GjzjNmNDlkZDQiLCJleHAiOjE3NjY0MjQzMzsInJvbCI6InVzZXIifQ.HqavCDqN45FbF1zBKfP302Dngt3DWlmFjUQ1AYyx9I"
6 }
```

At the bottom, there are buttons for 'Runner', 'Start Proxy', 'Cookies', 'Vault', 'Trash', and help icons.

network

Search Postman

Invite

Upgrade

Overview

POST http://127.0.0.1:5000/api/kitaplar • GET http://127.0.0.1:5000/api/kitaplar • + No environment

http://127.0.0.1:5000/api/kitaplar

Save Share

GET http://127.0.0.1:5000/api/kitaplar Send

Docs Params Authorization Headers (7) Body Scripts Settings Cookies

Auth Type: Bearer Token Token: 1zBKfP3O2Dngt3DWlmFjUQ1AYxe9l

The authorization header will be automatically generated when you send the request. Learn more about [Bearer Token](#) authorization.

Body Cookies Headers (5) Test Results

200 OK 10 ms 378 B

{ } JSON Preview Visualize

```
1 [  
2   {  
3     "ad": "Suç ve Ceza",  
4     "id": 1,  
5     "kategori_id": 1,  
6     "stok": 8,  
7     "yazar_id": 1  
8   },  
9   {  
10    "ad": "Test Kitap",  
11    "id": 10,  
12    "kategori_id": 1,  
13    "stok": 7,  
14    "yazar_id": 1  
15  }  
16 ]
```

final

Runner Start Proxy Cookies Vault Trash

network

Search Postman

Invite

Upgrade

Overview

POST http://127.0.0.1:5000/api/oduncler • GET http://127.0.0.1:5000/api/oduncler • POST http://127.0.0.1:5000/api/oduncler • + No environment

http://127.0.0.1:5000/api/oduncler

Save Share

POST http://127.0.0.1:5000/api/oduncler Send

Docs Params Authorization Headers (10) Body Scripts Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

1 {  
2 "kitap\_id": 1  
3 }  
4

Body Cookies Headers (5) Test Results

201 CREATED 20 ms 231 B

{ } JSON Preview Visualize

```
1 {  
2   "mesaj": "Kitap ödünç alındı"  
3 }
```

final

Runner Start Proxy Cookies Vault Trash

network

Search Postman

Invite Upgrade

Overview POST http://127.0.0.1:5000/api/user/oduncler GET http://127.0.0.1:5000/api/user/oduncler POST http://127.0.0.1:5000/api/user/oduncler GET http://127.0.0.1:5000/api/user/oduncler

No environment

http://127.0.0.1:5000/api/user/oduncler

Save Share Send

GET http://127.0.0.1:5000/api/user/oduncler

Docs Params Authorization Headers (7) Body Scripts Settings Cookies

Auth Type Bearer Token Token 1zBKfP3O2Dngt3DWlmlFjUQ1AYxe9I

Body Cookies Headers (5) Test Results

200 OK 10 ms 637 B

```
[{"id": 12, "kitap_adi": "Su\u0111 ve Ceza", "baslangic_tarih": "2025-12-21", "iade_tarih": "2025-12-28", "teslim_edildi": false}, {"id": 13, "kitap_adi": "Su\u0111 ve Ceza", "baslangic_tarih": "2025-12-21", "iade_tarih": "2025-12-28", "teslim_edildi": false}, {"id": 14, "kitap_adi": "Su\u0111 ve Ceza", "baslangic_tarih": "2025-12-22", "iade_tarih": "2025-12-29", "teslim_edildi": false}]
```

Runner Start Proxy Cookies Vault Trash

network

Search Postman

Invite Upgrade

Overview POST http://127.0.0.1:5000/api/user/oduncler/14/iade GET http://127.0.0.1:5000/api/user/oduncler/14/iade POST http://127.0.0.1:5000/api/user/oduncler/14/iade GET http://127.0.0.1:5000/api/user/oduncler/14/iade PUT http://127.0.0.1:5000/api/user/oduncler/14/iade

No environment

http://127.0.0.1:5000/api/user/oduncler/14/iade

Save Share Send

PUT http://127.0.0.1:5000/api/user/oduncler/14/iade

Docs Params Authorization Headers (8) Body Scripts Settings Cookies

Auth Type Bearer Token Token 1zBKfP3O2Dngt3DWlmlFjUQ1AYxe9I

The authorization header will be automatically generated when you send the request. Learn more about [Bearer Token](#) authorization.

Body Cookies Headers (5) Test Results

200 OK 48 ms 200 B

```
{"mesaj": "Kitap iade edildi"}
```

Runner Start Proxy Cookies Vault Trash

The screenshot shows the Postman application interface. At the top, there's a search bar with 'Search Postman' and a 'K' icon. To the right are various icons for inviting users, upgrading, and settings. Below the header, a navigation bar includes 'Overview', 'POST http://...', 'GET http://...', 'POST http://...', 'GET http://...', 'PUT http://...', 'GET http://...', and a '+' button. A dropdown menu says 'No environment'. On the left, a sidebar has tabs for 'Docs', 'Params', 'Authorization' (which is selected and highlighted in orange), 'Headers (7)', 'Body', 'Scripts', and 'Settings'. Under 'Authorization', it says 'Auth Type: Bearer Token' and shows a token value: '1zBKfP3O2Dngt3DWImFjUQ1AYxe9I'. A note explains that the authorization header will be automatically generated. The main area shows a 'GET' request to 'http://127.0.0.1:5000/api/user/cezalar'. The response status is '200 OK' with a duration of '9 ms' and a size of '167 B'. The response body is a JSON object with a single element '1': '[]'. At the bottom, there are buttons for 'Runner', 'Start Proxy', 'Cookies', 'Vault', 'Trash', and help.

```
Terminal Local + 
(venv) (base) nehirbuketpolat@Nehir-MacBook-Air AkilliKutuphane % python app.py
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 168-363-243
127.0.0.1 - - [22/Dec/2025 20:10:39] "POST /api/login HTTP/1.1" 200 -
127.0.0.1 - - [22/Dec/2025 20:12:26] "GET /api/kitaplar HTTP/1.1" 200 -
127.0.0.1 - - [22/Dec/2025 20:14:25] "POST /api/odunciler HTTP/1.1" 201 -
127.0.0.1 - - [22/Dec/2025 20:15:22] "GET /api/user/odunciler HTTP/1.1" 200 -
127.0.0.1 - - [22/Dec/2025 20:16:55] "PUT /api/odunciler/14/iade HTTP/1.1" 200 -
127.0.0.1 - - [22/Dec/2025 20:17:58] "GET /api/user/cezalar HTTP/1.1" 200 -
```

## 11-PROJENİN KAZANDIRDIKLARI

Bu proje sayesinde:

- Flask ile REST API geliştirme
- JWT ile güvenli kimlik doğrulama
- ORM (SQLAlchemy) kullanımı
- Gerçek hayat senaryosu modelleme
- Frontend–Backend entegrasyonu
- Postman ile profesyonel test süreci

konularında kapsamlı bir uygulama gerçekleştirilmiştir.

## 12-SONUÇ

Akıllı Kütüphane Yönetim Sistemi, modern yazılım geliştirme prensiplerine uygun, güvenli, modüler ve genişletilebilir bir yapıya sahiptir.

Sistem hem **akademik** hem de **gerçek dünya** ihtiyaçlarını karşılayacak şekilde tasarlanmıştır.