ANKARA ÜNİVERSİTESİ

Mühendislik Fakültesi

Bilgisayar Mühendisliği Bölümü

PROJE RAPORU

BiLet

Etkinlik Bilet Yönetim Sistemi

Hazırlayan: Yasin Turan Öcal

Öğrenci No: 21290318

Proje Linki (Drive):
https://drive.google.com/drive/folders/1NEQyDQJJPogMSxQo4cUP6hO3IKzSeQE_?usp=sharing

GitHub Linki:
https://github.com/2run66/bilet_app

ANKARA - 2026

İÇİNDEKİLER

1. GİRİŞ

Günümüzde etkinlik sektörü hızla dijitalleşmekte ve geleneksel kağıt bilet sistemleri yerini dijital çözümlere bırakmaktadır. Bu dönüşüm, hem organizatörler hem de katılımcılar için çeşitli avantajlar sunmaktadır. Dijital bilet sistemleri; sahtecilik riskini azaltmakta, bilet dağıtım maliyetlerini düşürmekte ve kullanıcı deneyimini iyileştirmektedir.

BiLet projesi, bu ihtiyaçlara cevap vermek üzere geliştirilmiş kapsamlı bir etkinlik bilet yönetim sistemidir. Proje, modern yazılım geliştirme teknolojileri kullanılarak tasarlanmış olup, hem mobil uygulama hem de backend API bileşenlerini içermektedir.

Bu rapor, BiLet projesinin teknik altyapısını, kullanılan teknolojileri, sistem mimarisini ve uygulama özelliklerini detaylı bir şekilde açıklamaktadır.

2. PROJE AMACI VE KAPSAMI

2.1. Proje Amacı

BiLet projesinin temel amacı, etkinlik biletlerinin dijital ortamda satın alınması, saklanması ve doğrulanması için kullanıcı dostu bir platform sunmaktır. Sistem, iki ana kullanıcı grubuna hizmet vermektedir:

- **Son Kullanıcılar:** Etkinlikleri keşfedebilen, bilet satın alabilen ve dijital biletlerini yönetebilen bireysel kullanıcılar.
- **Organizatörler:** Etkinlik oluşturabilen, bilet satışlarını takip edebilen ve QR kod ile katılımcı girişlerini doğrulayabilen etkinlik düzenleyicileri.

2.2. Proje Kapsamı

Proje kapsamında aşağıdaki bileşenler geliştirilmiştir:

- Cross-platform mobil uygulama (iOS ve Android)
- RESTful API backend servisi
- MongoDB bulut veritabanı entegrasyonu
- JWT tabanlı kimlik doğrulama sistemi
- QR kod oluşturma ve okuma modülleri
- Swagger API dokümantasyonu

3. KULLANILAN TEKNOLOJİLER

3.1. Frontend Teknolojileri

Mobil uygulama geliştirme sürecinde aşağıdaki teknolojiler kullanılmıştır:

Teknoloji	Versiyon	Kullanım Amacı
Flutter	SDK ^3.8.1	Cross-platform mobil uygulama framework'ü
Dart	3.x	Programlama dili
GetX	4.6.6	State management ve route yönetimi
Dio	5.4.0	HTTP client kütüphanesi
GetStorage	2.1.1	Yerel veri depolama
qr_flutter	4.1.0	QR kod oluşturma
qr_code_scanner	1.0.1	QR kod okuma

3.2. Backend Teknolojileri

Sunucu tarafı geliştirmede kullanılan teknolojiler aşağıda listelenmiştir:

Teknoloji	Versiyon	Kullanım Amacı
Python	3.x	Backend programlama dili
Flask	3.0.0	Web framework
MongoDB	Atlas Cloud	NoSQL veritabanı
MongoEngine	0.28.2	MongoDB ODM
Flask-JWT-Extended	4.6.0	JWT authentication
Flask-CORS	4.0.0	Cross-origin resource sharing
Flasgger	0.9.7.1	Swagger API dokümantasyonu

4. SİSTEM MİMARİSİ

BiLet projesi, MVC (Model-View-Controller) tasarım desenini takip eden modüler bir yapıya sahiptir. Sistem, istemci-sunucu mimarisi üzerine kurulmuş olup, mobil uygulama ile backend API arasında RESTful iletişim sağlanmaktadır.

4.1. Proje Dizin Yapısı

Frontend (Flutter) Yapısı:

```
lib/
├── controllers/         # İş mantığı katmanı (11 controller)
│   ├── auth_controller.dart
│   ├── home_controller.dart
│   ├── checkout_controller.dart
│   ├── tickets_controller.dart
│   └── ...
├── models/              # Veri modelleri (3 model)
│   ├── user_model.dart
│   ├── event_model.dart
│   └── ticket_model.dart
├── views/               # Kullanıcı arayüzü (17+ sayfa)
│   ├── auth/
│   ├── home/
│   ├── events/
│   ├── tickets/
│   └── organizer/
├── services/            # API servisleri (7 servis)
│   ├── api_service.dart
│   ├── auth_service.dart
│   └── ...
├── routes/              # Navigasyon yönetimi
└── main.dart            # Uygulama giriş noktası
```

Backend (Flask) Yapısı:

```
backend/
├── models/              # MongoDB modelleri
│   ├── user.py
│   ├── event.py
│   └── ticket.py
├── routes/              # API endpoint'leri
│   ├── auth_routes.py
│   ├── event_routes.py
│   ├── ticket_routes.py
│   ├── user_routes.py
```

```
│   └── organizer_routes.py
├── utils/                # Yardımcı fonksiyonlar
│   └── qr_generator.py
├── config.py             # Konfigürasyon
├── app.py                # Flask uygulaması
└── requirements.txt      # Bağımlılıklar
```

4.2. Katmanlı Mimari

Uygulama, aşağıdaki katmanlardan oluşmaktadır:

- **Sunum Katmanı (View):** Flutter widget'ları ile kullanıcı arayüzü
- **İş Mantığı Katmanı (Controller):** GetX controller'ları ile state yönetimi
- **Servis Katmanı:** API iletişimi ve veri işleme
- **Veri Katmanı (Model):** Veri yapıları ve veritabanı modelleri

5. VERİTABANI TASARIMI

Proje, MongoDB NoSQL veritabanı kullanmaktadır. MongoDB Atlas üzerinde barındırılan veritabanı, üç ana koleksiyondan oluşmaktadır.

5.1. Users Koleksiyonu

Kullanıcı bilgilerini ve kimlik doğrulama verilerini saklar.

```
{
    "_id": ObjectId,
    "email": String (unique, required),
    "password_hash": String (required),
    "name": String (required),
    "phone": String,
    "role": String ["user", "organizer", "admin"],
    "created_at": DateTime
}
```

5.2. Events Koleksiyonu

Etkinlik bilgilerini ve organizatör referansını içerir.

```
{
    "_id": ObjectId,
    "title": String (required),
    "description": String (required),
    "category": String (required),
    "location": String (required),
    "date": DateTime (required),
    "price": Float (required),
    "image_url": String (required),
    "available_tickets": Integer (required),
    "organizer_name": String (required),
    "organizer_id": Reference(User),
    "created_at": DateTime
}
```

5.3. Tickets Koleksiyonu

Satın alınan biletleri ve QR kod verilerini saklar.

```
{
    "_id": ObjectId,
    "event_id": Reference(Event),
    "user_id": Reference(User),
    "event_title": String,
    "event_location": String,
    "event_date": DateTime,
    "status": String ["active", "used", "expired", "pending"],
    "purchase_date": DateTime,
    "price": Float,
    "qr_code": String (base64),
    "seat_number": String
}
```

5.4. İndeksler

Veritabanı performansını artırmak için aşağıdaki indeksler tanımlanmıştır:

- Users: email (unique index)
- Events: category, date
- Tickets: event_id, user_id, event_date, status

6. API TASARIMI

Backend, RESTful API prensiplerine uygun olarak tasarlanmıştır. Tüm endpoint'ler /api prefix'i altında gruplandırılmıştır.

6.1. Authentication Endpoints

Method	Endpoint	Açıklama
POST	/api/auth/register	Yeni kullanıcı kaydı
POST	/api/auth/login	Kullanıcı girişi, JWT token döner
GET	/api/auth/me	Mevcut kullanıcı bilgisi

6.2. Event Endpoints

Method	Endpoint	Açıklama
GET	/api/events	Etkinlikleri listele (filtreleme destekli)
GET	/api/events/{id}	Etkinlik detayını getir
POST	/api/events	Yeni etkinlik oluştur (Auth gerekli)
PUT	/api/events/{id}	Etkinlik güncelle (Auth gerekli)
DELETE	/api/events/{id}	Etkinlik sil (Auth gerekli)
GET	/api/events/categories	Tüm kategorileri listele

6.3. Ticket Endpoints

Method	Endpoint	Açıklama
GET	/api/tickets	Kullanıcının biletlerini listele
GET	/api/tickets/{id}	Bilet detayını getir
POST	/api/tickets	Bilet satın al
PUT	/api/tickets/{id}/activate	Bileti aktifleştir
PUT	/api/tickets/{id}/use	Bileti kullanıldı olarak işaretle
DELETE	/api/tickets/{id}	Bileti iptal et

6.4. Organizer Endpoints

Method	Endpoint	Açıklama
GET	/api/organizer/events	Organizatörün etkinliklerini listele
GET	/api/organizer/events/{id}/attendees	Etkinlik katılımcılarını listele

7. UYGULAMA ÖZELLİKLERİ

7.1. Kullanıcı Modülü

Standart kullanıcılar için sunulan özellikler aşağıda listelenmiştir:

- **Kayıt ve Giriş:** E-posta ve şifre ile güvenli kimlik doğrulama
- **Etkinlik Keşfi:** Kategorilere göre filtreleme ve arama fonksiyonu
- **Etkinlik Detayları:** Etkinlik bilgileri, konum ve fiyat görüntüleme
- **Bilet Satın Alma:** Seçilen etkinlik için bilet satın alma işlemi
- **Dijital Bilet:** QR kodlu dijital biletin görüntülenmesi
- **Bilet Yönetimi:** Aktif ve geçmiş biletlerin kategorize edilmesi
- **Profil Yönetimi:** Kullanıcı bilgilerinin güncellenmesi

7.2. Organizatör Modülü

Etkinlik organizatörleri için sunulan özellikler:

- **Etkinlik Oluşturma:** Yeni etkinlik kaydı ve detay girişi
- **Etkinlik Düzenleme:** Mevcut etkinliklerin güncellenmesi
- **Satış Takibi:** Bilet satış istatistiklerinin görüntülenmesi
- **QR Kod Doğrulama:** Kamera ile bilet QR kodlarının taranması
- **Katılımcı Listesi:** Etkinliğe kayıtlı kullanıcıların görüntülenmesi
- **Giriş Onayı:** Biletlerin kullanıldı olarak işaretlenmesi

7.3. QR Kod Sistemi

Bilet doğrulama için QR kod sistemi implementasyonu:

- Her bilet için benzersiz QR kod üretimi (Base64 formatında)
- qr_flutter kütüphanesi ile QR kod görüntüleme
- qr_code_scanner kütüphanesi ile gerçek zamanlı QR kod okuma
- Backend'de bilet durumu doğrulama ve güncelleme

7.4. State Management

Uygulama genelinde GetX framework'ü ile reaktif state management uygulanmıştır. Bu yaklaşım şu avantajları sağlamaktadır:

- Reaktif programlama ile otomatik UI güncellemesi
- Dependency injection ile servis yönetimi
- Named routes ile kolay navigasyon
- Minimal boilerplate kod

8. GÜVENLİK

Proje geliştirme sürecinde güvenlik konularına özel önem verilmiştir. Uygulanan güvenlik önlemleri aşağıda açıklanmaktadır:

8.1. Kimlik Doğrulama

- **JWT (JSON Web Token):** Stateless authentication için JWT token kullanımı
- **Token Süresi:** Güvenlik için token yaşam süresi sınırlandırması
- **Refresh Token:** Oturum yenileme mekanizması

8.2. Şifre Güvenliği

- **Hashing:** Werkzeug kütüphanesi ile şifrelerin hash'lenerek saklanması
- **Salt:** Her şifre için benzersiz salt değeri
- **Minimum Uzunluk:** Şifre politikası (minimum 6 karakter)

8.3. API Güvenliği

- **CORS:** Cross-Origin Resource Sharing koruması
- **Rate Limiting:** API isteklerinin sınırlandırılması
- **Input Validation:** Gelen verilerin doğrulanması
- **Rol Tabanlı Yetkilendirme:** Endpoint erişim kontrolü

8.4. Veritabanı Güvenliği

- **MongoDB Atlas:** Güvenli bulut veritabanı hizmeti
- **SSL/TLS:** Şifreli veritabanı bağlantısı
- **IP Whitelist:** Erişim izni verilen IP adresleri

9. SONUÇ

BiLet projesi, modern yazılım geliştirme teknolojilerini kullanarak eksiksiz bir etkinlik bilet yönetim sistemi sunmaktadır. Flutter ile cross-platform mobil uygulama, Python Flask ile RESTful API ve MongoDB ile esnek veritabanı çözümü başarıyla entegre edilmiştir.

Proje, MVC tasarım deseni, modüler kod yapısı ve yazılım geliştirme best

practice'lerini takip ederek sürdürülebilir ve ölçeklenebilir bir mimari üzerine inşa edilmiştir. GetX ile state management, JWT ile güvenli authentication ve QR kod sistemi ile pratik bilet doğrulama özellikleri başarıyla implementasyon edilmiştir.

Geliştirilen sistem, hem son kullanıcılar hem de organizatörler için kullanıcı dostu bir deneyim sunmakta olup, gelecekte ödeme sistemi entegrasyonu, bildirim servisi ve analitik dashboard gibi özelliklerle genişletilebilir yapıdadır.

Proje İstatistikleri

Frontend Controller Sayısı	11
Frontend Servis Sayısı	7
Sayfa/View Sayısı	17+
Backend Route Modülü	6
Veritabanı Koleksiyonu	3

KAYNAKÇA

[1] Flutter Documentation. (2024). Flutter - Build apps for any screen. https://flutter.dev/docs

[2] Dart Programming Language. (2024). Dart Overview. https://dart.dev/overview

[3] GetX Documentation. (2024). GetX - Flutter State Management. https://pub.dev/packages/get

[4] Flask Documentation. (2024). Flask Web Development. https://flask.palletsprojects.com/

[5] MongoDB Documentation. (2024). MongoDB Manual. https://www.mongodb.com/docs/manual/

[6] MongoEngine Documentation. (2024). MongoEngine User Documentation. https://docs.mongoengine.org/

[7] Flask-JWT-Extended. (2024). JWT Authentication for Flask. https://flask-jwt-extended.readthedocs.io/

[8] Dio HTTP Client. (2024). Dio - Powerful HTTP client for Dart/Flutter. https://pub.dev/packages/dio

[9] QR Flutter. (2024). QR code generation for Flutter. https://pub.dev/packages/qr_flutter

[10] Flasgger. (2024). Easy Swagger UI for Flask. https://github.com/flasgger/flasgger