

# TEKNİK ŞARTNAME

T.C. Ziraat Bankası A.Ş. ile Sakarya Uygulamalı Bilimler Üniversitesi (SAYEM) arasında imzalanan protokol çerçevesinde, bankanın mobil uygulaması ve üniversitenin uzaktan eğitim sistemi arasında entegrasyon sağlamak amacıyla geliştirilecek RESTful API'nin detaylı teknik özelliklerini ve geliştirme süreçlerini içermektedir. Bu API, bankanın müşterilerine eğitim sistemine güvenli erişim sağlama, eğitim materyallerine ulaşım, katılım belgesi üretimi ve e-Devlet entegrasyonu gibi işlevleri yerine getirecektir. Proje Laravel framework üzerinde geliştirilecek ve API test süreçleri Postman uygulaması ile yapılacaktır.

## Teknik Şartname

### 1. Projenin Amacı

Bu projenin amacı, bankanın müşterilerinin mobil uygulama üzerinden üniversitenin uzaktan eğitim sistemine (LMS) şifresiz ve güvenli bir şekilde erişimini sağlamak amacıyla bir RESTful API geliştirmektir. API, kullanıcı doğrulaması, token üretimi, eğitim içeriklerine erişim, katılım belgesi üretimi ve kullanıcıya bildirim gönderme süreçlerini kapsayacaktır. Proje, PHP Laravel framework üzerinde yapılandırılacak ve modern yazılım geliştirme standartlarına uygun olarak tasarlanacaktır.

### 2. API Genel Özellikleri

#### 1. RESTful API Tasarımı:

- **Framework:** API, **Laravel framework** kullanılarak geliştirilecektir.
- **Endpoint Yapısı:** Tüm endpoint'ler, RESTful mimariye uygun olarak tasarlanacak ve JSON formatında veri alışverişi yapılacaktır.
- **HTTP Status Kodları:** API, aşağıdaki HTTP status kodlarını kullanacaktır:
  - **200 (OK):** İstek başarılı olduğunda kullanılır.
  - **201 (Created):** Yeni bir kaynak oluşturulduğunda kullanılır.
  - **400 (Bad Request):** İstek geçersiz veya eksik olduğunda kullanılır.
  - **401 (Unauthorized):** Kullanıcı doğrulaması yapılamadığında kullanılır.
  - **403 (Forbidden):** Yetkisiz erişim denemesi yapıldığında kullanılır.
  - **404 (Not Found):** İstenen kaynak bulunamadığında kullanılır.
  - **500 (Internal Server Error):** Sunucu hatalarında kullanılır.

#### 2. Token Tabanlı Yetkilendirme:

- **JWT (JSON Web Token):** Kullanıcı doğrulaması sonrası JWT kullanılarak yetkilendirme sağlanacaktır.
- **Token Yönetimi:** Her API çağrısında, Authorization başlığı altında geçerli bir JWT token ile istek yapılacaktır. Token süresi dolduğunda, refresh token kullanılarak yeni bir token sağlanacaktır.
- **Güvenlik Mekanizmaları:** Token revocation ve blacklisting mekanizmaları yetkisiz erişimlere karşı korunma sağlayacaktır.

### 3. Endpoint Detayları:

- **/api/authenticate (POST)**: Kullanıcı doğrulaması ve token üretimi için kullanılır. Kullanıcı bilgileri doğrulandıktan sonra geçerli bir JWT token üretilir.
- **/api/education/content (GET)**: Kullanıcıya ait eğitim materyalleri, video dersler ve notlar bu endpoint üzerinden sağlanır.
- **/api/education/record (POST)**: Kullanıcının eğitim sürecindeki ilerlemeleri ve izleme süreleri gibi veriler bu endpoint aracılığıyla kaydedilir.
- **/api/certificate/generate (POST)**: Kullanıcı için otomatik olarak katılım belgesi oluşturulur ve belge e-Devlet sistemine entegre edilir.
- **/api/certificate/status (GET)**: Katılım belgesinin durumu sorgulanır. SMS bildirimi bu işlemle entegre çalışır.

### 4. API Test Süreci:

- **Postman Kullanımı**: API test süreçlerinde **Postman** kullanılarak endpoint'ler manuel test edilecektir. Her bir endpoint için test senaryoları oluşturulacak ve JSON formatındaki isteklere karşı API'nin verdiği yanıtlar test edilecektir.
  - Postman'de oluşturulacak test adımları:
    - **Authenticate Endpoint Testi**: Kullanıcı doğrulama ve token üretiminin düzgün çalışıp çalışmadığını kontrol etmek.
    - **Education Content Testi**: Kullanıcının token'ı ile eğitim materyallerine erişimin sağlandığını test etmek.
    - **Certificate Generate Testi**: Eğitim süreci tamamlandığında katılım belgesi oluşturulmasını test etmek.
- **Laravel Test Araçları**: API, Laravel'in yerleşik **PHPUnit** test araçları ile de kapsamlı bir şekilde test edilecektir. Unit ve Integration test'ler her bir fonksiyon için ayrı ayrı yapılacaktır.

## 3. Geliştirme Süreci

### 1. Proje Yapılandırması:

- **Framework ve Sürüm**: Geliştirme süreci Laravel 10.x sürümü üzerinde yapılacaktır.
- **MVC Mimarisi**: Proje, Model-View-Controller (MVC) mimarisi kullanılarak modüler ve ölçeklenebilir bir yapıda tasarlanacaktır.
- **Veritabanı Yönetimi**: MySQL veya PostgreSQL kullanılacak, Eloquent ORM ile veri yönetimi sağlanacaktır.
- **Git Versiyon Kontrolü**: Proje yönetimi Git kullanılarak yapılacak ve geliştirme aşamaları branch'ler üzerinden yürütülecektir.

### 2. API Güvenliği:

- **CORS Politikası**: Belirlenen domain'ler dışında API erişimi engellenecektir.
- **Güvenlik Tehditlerine Karşı Önlemler**: Laravel'in yerleşik güvenlik özellikleri kullanılarak SQL enjeksiyon, XSS ve CSRF saldırılarına karşı koruma sağlanacaktır.
- **Şifreleme**: Kullanıcı bilgileri AES-256 gibi güçlü şifreleme algoritmaları kullanılarak şifrelenmiş olarak saklanacaktır.

## 4. Entegrasyon ve Test Süreci

### 1. API Entegrasyonu:

- **JWT Entegrasyonu:** Laravel'in **Sanctum** veya **Passport** paketleri kullanılarak JWT tabanlı token üretimi ve yönetimi sağlanacaktır.
- **Webhook Entegrasyonu:** Katılım belgeleri ve SMS bildirimleri için webhook entegrasyonu yapılacaktır.

### 2. Postman ile Test Süreci:

- Postman testleri yapılırken, her bir endpoint'in doğru çalıştığı ve doğru HTTP status kodlarının döndüğü kontrol edilecektir. Her bir API çağrısı için test senaryoları şu şekilde yapılandırılacaktır:
  - **Başarılı İstek Testi:** Doğru kullanıcı bilgilerinin doğrulanıp doğrulanmadığını kontrol edin.
  - **Yetkilendirilmemiş İstek Testi:** JWT token olmadan yapılan isteklerin 401 Unauthorized ile reddedildiğinden emin olun.
  - **Yanlış Veri Testi:** Geçersiz veriyle yapılan isteklerin 400 Bad Request döndüğünden emin olun.

### 3. Test Driven Development (TDD):

- Proje, TDD yaklaşımıyla geliştirilecek olup, her fonksiyon için önce testler yazılacak ve ardından kodlama yapılacaktır. Testler, **PHPUnit** ile entegre olarak çalıştırılacaktır.

### 5. Performans Optimizasyonu ve Raporlama

- **Eager Loading ve Caching:** API'nin hızlı yanıt vermesi için Eager Loading, Query Caching ve Redis Cache gibi performans artırıcı teknikler kullanılacaktır.
- **API İzleme:** API'nin performansı **New Relic** veya **Datadog** gibi araçlar ile izlenecek ve gerekli optimizasyonlar yapılacaktır.
- **Raporlama:** Kullanıcıların eğitim materyalleri ile olan etkileşimleri `/api/education/record` endpoint'i üzerinden kaydedilecek ve raporlanacaktır.

### 6. Güvenlik ve Gizlilik

- **KVKK'ya Uygunluk:** API, 6698 sayılı Kişisel Verilerin Korunması Kanunu'na (KVKK) uygun olarak geliştirilecek ve veri gizliliği sağlanacaktır.
- **Penetration Testleri:** API, düzenli olarak penetrasyon testlerine tabi tutulacak ve güvenlik açıkları kapatılacaktır.