# SyncBranch Proje Dokümantasyonu

## 1. Giriş

### Proje Arka Planı

SyncBranch, müzik dinleme alışkanlıklarını analiz ederek kullanıcıların kişisel profillerini ortaya çıkaran, kullanıcıları bu veriler üzerinden eşleştiren ve sosyal bir platform oluşturan yenilikçi bir mobil uygulamadır. Projenin başlangıç noktası, müziğin insan yaşamındaki önemli rolünden ilham alarak, insanların müzik zevkleri ve tercihleri üzerinden kendilerini keşfetmelerini sağlamak olmuştur.

### Amaç ve Hedefler

SyncBranch’ın temel amacı, kullanıcıların dinleme geçmişlerini analiz ederek kişilik özelliklerini ortaya çıkarmak ve bu özellikler doğrultusunda kullanıcıları bir araya getirmektir. Hedeflenen sonuç, kullanıcıların müzik tabanlı sosyal bir topluluk oluşturarak, müzikle ilham bulan ve eğlenen bir platformda etkileşim kurabilmelerini sağlamaktır.

#### Projenin Spesifik Hedefleri:

* **Müzik verisi analizi**: Kullanıcıların dinleme alışkanlıkları üzerinden veri toplamak ve analiz yapmak.
* **Sosyal eşleştirme**: Kullanıcıların ortak müzik zevklerine göre eşleştirilmesini sağlamak.
* **Kullanıcı deneyimi**: Basit, sezgisel ve kullanıcı dostu bir arayüz ile kullanıcıların uygulamayı kolayca kullanabilmesini sağlamak.
* **Güvenlik**: Kullanıcı verilerinin güvenliğini sağlamak için güçlü veri koruma önlemleri almak.

### Paydaşlar ve Hedef Kullanıcılar

* **Paydaşlar**:
  + **Uygulama Geliştirme Ekibi**: Backend ve frontend geliştiriciler, veri mühendisleri ve UX/UI tasarımcıları.
  + **Proje Yönetimi**: Scrum Master ve proje yöneticileri.
* **Hedef Kullanıcılar**: Müzik tutkunları, yeni müzik keşfetmeyi seven kişiler, müzikle kendini ifade eden kullanıcılar.

## 2. Teknoloji Yığını

### Programlama Dilleri ve Araçlar

* **Backend**: Python (FastAPI) — Modern web uygulamaları için yüksek performanslı ve esnek bir framework.
* **Veritabanı**: Microsoft SQL Server — Güvenilir ve ölçeklenebilir veri yönetim çözümleri sunar.
* **Frontend**: Flutter — Mobil uygulama geliştirme için güçlü ve dinamik bir araç.
* **API Entegrasyonu**: Spotify API — Müzik verilerini çekmek ve analiz etmek için kullanılan birinci sınıf çözüm.

### Frameworkler ve Kütüphaneler

* **FastAPI**: Hızlı gelişim süreçleri ve performans odaklı yapısıyla modern backend geliştirme.
* **Pyodbc**: Veritabanı bağlantısı ve sorguları için kullanılır.
* **Pandas ve NumPy**: Büyük veri setlerinin işlenmesi ve analiz edilmesi için.
* **Matplotlib/Seaborn**: Veri görselleştirme ve analiz çıktılarının sunumu.

### Diğer Araçlar

* **Git ve GitHub**: Kod versiyon kontrolü ve iş birliği.
* **JIRA**: Proje yönetimi ve görev takibi.
* **Postman**: API testleri ve doğrulama.

## 3. Proje Genel Görünümü

### Uygulama Tanımı

SyncBranch, müzikseverlerin müzik dinleme geçmişlerini analiz eden ve bu analizler üzerinden kişisel içgörüler sunan bir mobil uygulamadır. Kullanıcılar, uygulama üzerinden diğer kullanıcılarla iletişim kurabilir, önerilerde bulunabilir ve yeni müzikler keşfederek müzikle dolu bir deneyim yaşayabilir.

### Temel Özellikler ve Fonksiyonlar

1. **Kullanıcı Doğrulama**:
   * Güvenli giriş ve çok faktörlü kimlik doğrulama (MFA) özellikleri.
2. **Müzik Verisi Toplama ve Analiz**:
   * Spotify API üzerinden veri çekme ve analizler için işleme.
3. **Sosyal Etkileşim**:
   * Kullanıcılar arasında anlık mesajlaşma ve kullanıcıları engelleme/bildirme fonksiyonları.
4. **Bildirim Sistemi**:
   * Eşleşme ve sohbet güncellemeleri için bildirim gönderme.

### Kullanıcı Hikayeleri ve Kullanım Durumları

**Kullanım Durumu Örneği: Dinleme Geçmişini Görüntüleme**

* **Kullanıcı**: Müzik dinleme alışkanlıklarını görüntüleyerek analiz sonuçlarını inceler.
* **Sistem**: Spotify API’den veri çeker ve kullanıcı arayüzünde analiz sonuçlarını gösterir.

**Kullanım Durumu Örneği: Kişilik Analizi ve Sosyal Eşleştirme**

* **Kullanıcı**: Dinleme alışkanlıklarına göre uygulama üzerinde kişilik analizini görüntüler ve öneriler alır.
* **Sistem**: Kullanıcının analizini yapar ve benzer profillere sahip kullanıcıları önerir.

## 4. Yazılım Geliştirme Yaşam Döngüsü (SDLC)

### Planlama ve Gereksinim Toplama

Proje başlangıcında detaylı bir gereksinim toplama süreci gerçekleştirildi. Kullanıcı ihtiyaçları belirlendi ve işlevsel gereksinimler dokümante edildi.

### Sistem Tasarımı ve Mimarisi

Proje, modüler ve ölçeklenebilir bir yapı hedeflenerek tasarlandı. **Yüksek Seviye Mimarisi** üç ana bileşenden oluşur:

1. **Backend**: Hızlı veri işleme ve API entegrasyonu.
2. **Veritabanı**: Kullanıcı ve müzik verilerinin güvenli şekilde saklanması.
3. **Frontend**: Kullanıcıların sezgisel bir deneyim yaşamasını sağlayan mobil arayüz.

### Geliştirme ve Uygulama

Tüm modüller, modern yazılım geliştirme prensiplerine uygun olarak geliştirildi. Test odaklı bir yaklaşım benimsendi ve kod kalitesi yüksek tutuldu.

## 5. Veri Yönetimi

### Veri Depolama Çözümleri

Veriler, SQL Server’da güvenli ve yedeklenebilir bir biçimde saklanır. Veritabanı yapısı, performansı artırmak için optimize edildi.

### Veri İşleme Teknikleri

* **Veri Rafine Etme**: Spotify API’den çekilen veriler işlenir ve analiz için hazır hale getirilir.
* **Veri Analizi**: Kullanıcıların dinleme alışkanlıkları analiz edilerek kişisel içgörüler sunulur.

## 6. Kullanıcı Arayüzü (UI) ve Kullanıcı Deneyimi (UX)

### Tasarım İlkeleri

* **Kullanıcı Odaklı Tasarım**: Kolay navigasyon, basit ve anlaşılır arayüzler.
* **Responsive Tasarım**: Her cihazda sorunsuz kullanım.
* **Renk Paleti ve Tipografi**: Kullanıcı dostu renkler ve okunaklı yazı tipleri.

### Anahtar UI Bileşenleri

* **Ana Sayfa**: Kullanıcılar için öneriler ve analiz sonuçlarının gösterildiği panel.
* **Profil Sayfası**: Kullanıcı bilgileri ve kişisel analiz raporları.
* **Bildirim Paneli**: Yeni eşleşme ve mesaj bildirimlerinin görüldüğü alan.

## 7. Kodlama ve En İyi Uygulamalar

### Kodlama Yönergeleri

* **Yorumlama ve Dokümantasyon**: Tüm fonksiyonlar ve metodlar detaylı yorumlarla açıklanır.
* **Kod Düzeni**: Python PEP 8 standartlarına uygun yazım ve formatlama.
* **Error Handling**: Hata yakalama mekanizmaları ile sistemin kararlı çalışması sağlanır.

## 8. Sistem İş Akışı ve Mimari

### Yüksek Seviye Sistem Mimarisi

Kullanıcıların uygulama ile etkileşimleri, verilerin Spotify API üzerinden çekilmesi, analizlerin yapılması ve kullanıcıya sunulması şeklinde bir akış izler.

1. **Veri İşleme Akışı**: Kullanıcı -> Spotify API -> Veritabanı -> Kullanıcı Arayüzü.
2. **Sosyal Etkileşim Akışı**: Kullanıcı öneri alır, iletişim kurar ve yeni kullanıcılar keşfeder.

### Bileşen ve Veri Akış Diyagramları

Sistemin bileşenleri arasındaki veri akışı şematik olarak sunulmuştur ve bu akış diyagramları tüm veri etkileşimlerini detaylıca gösterir.

## 9. Test ve Kalite Güvencesi

### Hata Takibi ve Yönetimi

* Hata yönetimi için JIRA kullanıldı.
* Loglama mekanizmaları, sorunların hızlı çözümü için entegre edildi.

### Test Aşamaları

* **Birim Testleri**: Her modül bağımsız olarak test edildi.
* **Entegrasyon Testleri**: Modüller arası entegrasyon test edildi.
* **Sistem Testleri**: Genel sistem performansı test edildi.
* **Kullanıcı Kabul Testi (UAT)**: Kullanıcılar üzerinde yapılan testler ile son onay alındı.

## 10. Gelecek Yol Haritası

### Potansiyel İyileştirmeler

* **Performans İyileştirmeleri**: Daha büyük veri kümelerinde hızlı işlem.
* **Gelişmiş AI Kullanımı**: Kullanıcı analizlerinde makine öğrenmesi ve yapay zeka entegrasyonu.
* **Yeni Özellikler**: Öneri motoru geliştirmeleri, kişiselleştirilmiş müzik önerileri.

### Entegrasyon Planları

* **Yapay Zeka**: Müzik analizi için yeni AI algoritmalarının entegrasyonu.
* **Üçüncü Parti Entegrasyonlar**: Farklı müzik platformları ile entegrasyon.

## İletişim Bilgileri

* **İletişim**: <mailto:cagatayalkan@yggbranch.dev>
* **GitHub Repo**: <https://github.com/cagatay-softgineer/SyncBranch>