GEREKSİNİM ANALİZİ DOKÜMANI

1. Proje Adı:

Turhub – Github Türkiye Rapor

2. Proje Amacı:

Bu projenin amacı, Türkiye'deki GitHub kullanıcılarının yazılım geliştirme eğilimlerini ve popüler projelerini analiz ederek, bu verileri görsel ve anlamlı bir formatta sunmaktır. Python kullanılarak GitHub API'den veri toplanacak, PostgreSQL veritabanına kaydedilecek ve JavaScript ile dinamik bir web arayüzü üzerinden raporlar görselleştirilecektir.

3. Proje Kapsamı:

- Türkiye'deki GitHub kullanıcılarının tercih ettiği programlama dillerinin analiz edilmesi.
- Türkiye'de en çok yıldızlanmış projelerin ve en çok takip edilen kullanıcıların tespit edilmesi.
- Python ile verilerin GitHub API'sinden alınması.
- PostgreSQL kullanarak verilerin depolanması ve yönetilmesi.
- JavaScript (Node.js, React) ile dinamik web sayfası ve veri görselleştirme yapılması.

4. İşlevsel Gereksinimler (Functional Requirements):

4.1 Kullanıcı Arayüzü (UI) Gereksinimleri:

- Web sayfası, Türkiye'deki GitHub kullanıcıları ve projeleri hakkında veriler sunan ana sayfa içermelidir.
- **Görselleştirmeler**:
    - Popüler programlama dillerini bir **pasta grafiği** ile sunma.
    - En çok yıldızlanmış projeleri bir **sıralama tablosu** ve **bar grafiği** ile gösterme.
    - En çok takip edilen kullanıcıları bir **kullanıcı sıralama listesi** ile gösterme.
- Veriler **gerçek zamanlı** ya da düzenli olarak güncellenmeli ve anlık raporlar oluşturulmalıdır.

4.2 GitHub API ile Veri Çekme:

- **Python** kullanılarak GitHub API'ye istekler gönderilecek ve JSON formatında veri alınacak.
- Çekilecek veriler arasında şunlar bulunmalıdır:
    - Kullanıcı adı (username)
    - Konum (location)
    - Takipçi sayısı (followers_count)
    - Kullanıcının projeleri ve projelerin yıldız (star) ve fork sayıları
    - Projede kullanılan programlama dili
- Python'da veriyi işlemek için **requests**, **pandas**, ve **GitHub API Wrapper** kullanılabilir.

4.3 Veri Depolama (PostgreSQL):

- **PostgreSQL** kullanılarak çekilen veriler veritabanına kaydedilecektir.
- Veritabanı şeması:
    - Kullanıcı bilgileri, projeler, programlama dilleri ve takipçi ilişkileri tablolarda tutulacaktır.
    - İlişkisel veritabanı modeli, tablolar arasında birincil/harici anahtar ilişkileri içermelidir.
- Veritabanı, büyük veri kümelerini depolayacak şekilde optimize edilmelidir.

4.4 Veri Görselleştirme (JavaScript - React):

- **React.js** kullanılarak, kullanıcı dostu bir arayüzde veriler sunulacaktır.
- Görselleştirme için **D3.js** veya **Chart.js** gibi JavaScript kütüphaneleri kullanılacak:
    - Popüler programlama dillerini **pasta grafiği**.
    - En çok yıldızlanmış projeleri **bar grafiği**.
    - En popüler kullanıcıları bir **sıralama tablosu**.

4.5 Otomatik Veri Güncelleme:

- Veri toplama işlemi Python'da bir zamanlayıcı (örn. **cron job** veya **apscheduler**) ile düzenli olarak yapılacak.
- PostgreSQL veritabanı düzenli aralıklarla güncellenecek ve yeni veriler eklenecektir.

5. Performans Gereksinimleri (Performance Requirements):

- Web sayfasının yüklenme süresi **2 saniyeden az** olmalıdır.

- GitHub API istek sınırları dikkate alınmalı; bu sınırları aşmamak için **önbellekleme** (örn. Redis) kullanılmalıdır.
- PostgreSQL sorguları optimize edilmeli ve büyük veri setleri üzerinde hızlı sonuçlar verecek şekilde yapılandırılmalıdır.

6. Güvenlik Gereksinimleri (Security Requirements):

- GitHub API erişimi için kullanılacak **API anahtarları**, çevresel değişkenler içinde güvenli bir şekilde saklanmalıdır (örn. .env dosyaları).
- HTTPS protokolü ile güvenli bağlantı sağlanmalıdır.
- SQL Injection ve diğer potansiyel güvenlik açıklarına karşı koruma sağlanmalıdır.

7. Kullanılabilirlik Gereksinimleri (Usability Requirements):

- Web sayfası **responsive** olmalı ve mobil cihazlarda sorunsuz çalışmalıdır.
- Kullanıcı arayüzü basit ve anlaşılır olmalıdır.
- Web sayfası üzerinden **arama** ve **filtreleme** özellikleri bulunmalıdır (örn. belirli programlama dillerine göre arama yapılabilmeli).

8. Teknik Gereksinimler (Technical Requirements):

8.1 Yazılım Teknolojileri:

- **Frontend**:
  - **HTML5**, **CSS3**, **JavaScript** (React.js framework).
  - Veri görselleştirme için **Chart.js** veya **D3.js**.
- **Backend**:
  - **Node.js** veya **Express.js** ile API ve sunucu işlemleri yürütülecek.
  - Python kullanılarak veri GitHub API'dan çekilecek ve PostgreSQL veritabanına işlenecek.
  - **Flask** veya **FastAPI** framework'leri ile Python backend servisi kurulabilir.
- **Veritabanı**:
  - **PostgreSQL** veritabanı kullanılacak.
  - Veritabanı düzenli olarak güncellenecek ve ölçeklenebilirlik göz önünde bulundurulacak.

9. Framework ve Kütüphaneler:

- **React.js**: Kullanıcı arayüzü geliştirmede kullanılacak ve dinamik içerikler bu framework ile sağlanacak.
- **Chart.js / D3.js**: Veri görselleştirmeleri için kullanılacak grafik kütüphaneleri.
- **Node.js**: Backend işlemleri ve API sunucusu yönetimi için kullanılacak.
- **Express.js**: Node.js için basit ve hızlı bir web uygulama framework'ü.
- **Flask/FastAPI (Python)**: Python ile GitHub API'den veri çekme ve işleme için kullanılacak mikro framework'ler.
- **PostgreSQL**: İlişkisel veritabanı yönetim sistemi olarak kullanılacak.
- **pandas (Python)**: Verilerin işlenmesi ve temizlenmesi için kullanılacak Python kütüphanesi.
- **requests (Python)**: API'den veri çekmek için kullanılacak HTTP istemcisi.

10. Zaman Çizelgesi (Timeline):

Görev	Başlangıç Tarihi	Bitiş Tarihi	Sorumlu Kişi
Gereksinim Analizi	01 Kasım 2024	05 Kasım 2024	Proje Ekibi
Python API Entegrasyonu	06 Kasım 2024	12 Kasım 2024	Python Geliştirici
Veritabanı Tasarımı (PostgreSQL)	13 Kasım 2024	17 Kasım 2024	DB Uzmanı
Frontend Geliştirme (React.js)	18 Kasım 2024	25 Kasım 2024	Frontend Geliştirici
Veri Görselleştirme	26 Kasım 2024	03 Aralık 2024	Frontend Geliştirici
Test ve Son Düzenlemeler	04 Aralık 2024	10 Aralık 2024	Test Ekibi
Proje Yayına Alma	11 Aralık 2024	15 Aralık 2024	Tüm Ekip