

Kampüs Etkinlik ve Öğrenci Yönetim Sistemi – Proje Raporu

1. Projenin Amacı

Bu projenin amacı, kampüs içerisinde yer alan öğrencilerin, öğrenci kulüplerinin ve bu kulüpler tarafından düzenlenen etkinliklerin yönetimini sağlayan modüler bir Python uygulaması geliştirmektir. Sistem; öğrenci yönetimi, etkinlik yönetimi ve raporlama olmak üzere üç ana bileşenden oluşmaktadır.

Proje, veri bütünlüğünü sağlamak, çakışmaları engellemek ve raporlama yapabilmek amacıyla uygun veri yapıları kullanılarak tasarlanmıştır.

2. Dosya Yapısı ve Modüller

Proje aşağıdaki Python dosyalarından oluşmaktadır:

- **data_store.py** : Ortak veri yapılarının tutulduğu dosya
- **students.py** : Öğrenci işlemlerinin gerçekleştirildiği modül
- **events.py** : Etkinlik ve kayıt işlemlerinin gerçekleştirildiği modül
- **reports.py** : Raporlama ve istatistik işlemlerinin yapıldığı modül
- **demo.py** : Tüm fonksiyonların test edildiği ana dosya

Tüm dosyalar aynı dizin içerisinde yer almakta ve birbirleriyle `import` yoluyla haberleşmektedir.

3. Kullanılan Veri Yapıları

3.1 Öğrenci Veri Yapısı

Öğrenciler, anahtar olarak öğrenci numarası kullanılan bir sözlük (dictionary) içerisinde saklanmaktadır.

Her öğrenci için aşağıdaki bilgiler tutulmaktadır:

- Ad Soyad
- Bölüm
- Sınıf
- Üye olunan kulüpler (set)
- Katılınan etkinlikler (set)

Set veri yapısı kullanılarak tekrar eden kulüp veya etkinlik kayıtlarının önüne geçilmiştir.

3.2 Etkinlik Veri Yapısı

Etkinlikler, anahtar olarak etkinlik kodu kullanılan bir sözlük içerisinde saklanmaktadır.

Her etkinlik için aşağıdaki bilgiler tutulmaktadır:

- Etkinlik adı
 - Düzenleyen kulüp
 - Tarih ve saat bilgisi
 - Oda bilgisi
 - Kapasite
 - Katılımcılar (set)
-

3.3 Rezervasyon Veri Yapısı

Oda-tarih-saat çakışmalarını önlemek amacıyla `reservations` adlı bir set kullanılmaktadır.

Bu set içerisinde `(oda, tarih, saat)` şeklinde tuple değerler tutulmaktadır. Böylece aynı anda aynı odada birden fazla etkinlik oluşturulması engellenmiştir.

4. Fonksiyonların Açıklaması

4.1 Öğrenci Fonksiyonları (students.py)

- **add_student** : Yeni öğrenci ekler
 - **remove_student** : Var olan öğrenciyi siler
 - **get_student** : Belirli bir öğrencinin bilgilerini döndürür
 - **list_students** : Tüm öğrencileri liste halinde döndürür
-

4.2 Etkinlik Fonksiyonları (events.py)

- **add_event** : Yeni etkinlik oluşturur ve oda-tarih-saat çakışmasını kontrol eder
 - **register** : Öğrenciyi etkinliğe kaydeder (kapasite ve tekrar kontrolü yapar)
 - **cancel_registration** : Öğrencinin etkinlik kaydını iptal eder
 - **list_events** : Tüm etkinlikleri listeler
-

4.3 Raporlama Fonksiyonları (reports.py)

Bu modülde lambda fonksiyonları kullanılarak sıralama işlemleri gerçekleştirilmiştir.

- **popular_events** : En fazla katılımcıya sahip etkinlikleri sıralar
 - **active_students** : En fazla etkinliğe katılan öğrencileri sıralar
 - **club_statistics** : Kulüp bazlı etkinlik ve katılımcı sayılarını analiz eder
-

5. Demo ve Test Süreci (demo.py)

`demo.py` dosyasında:

- En az 3 öğrenci eklenmiştir
- En az 2 etkinlik oluşturulmuştur
- En az 3 öğrenci-etkinlik kaydı yapılmıştır
- Tüm raporlama fonksiyonları çağrılarak ekran çıktıları alınmıştır

Bu dosya, projenin doğru çalıştığını göstermek amacıyla hazırlanmıştır.

6. Sonuç

Bu projede modüler programlama yaklaşımı kullanılarak okunabilir, sürdürülebilir ve genişletilebilir bir sistem geliştirilmiştir. Veri bütünlüğü set yapıları ile sağlanmış, raporlama işlemleri lambda fonksiyonları ile gerçekleştirilmiştir.