

# YZM205 - NESNE TABANLI PROGRAMLAMA LAB.

---

## UYGULAMA – HAFTA 11

❖ *Eğitmen: Doç. Dr. Zafer CÖMERT*

*E-posta: [zcomert@samsun.edu.tr](mailto:zcomert@samsun.edu.tr)*

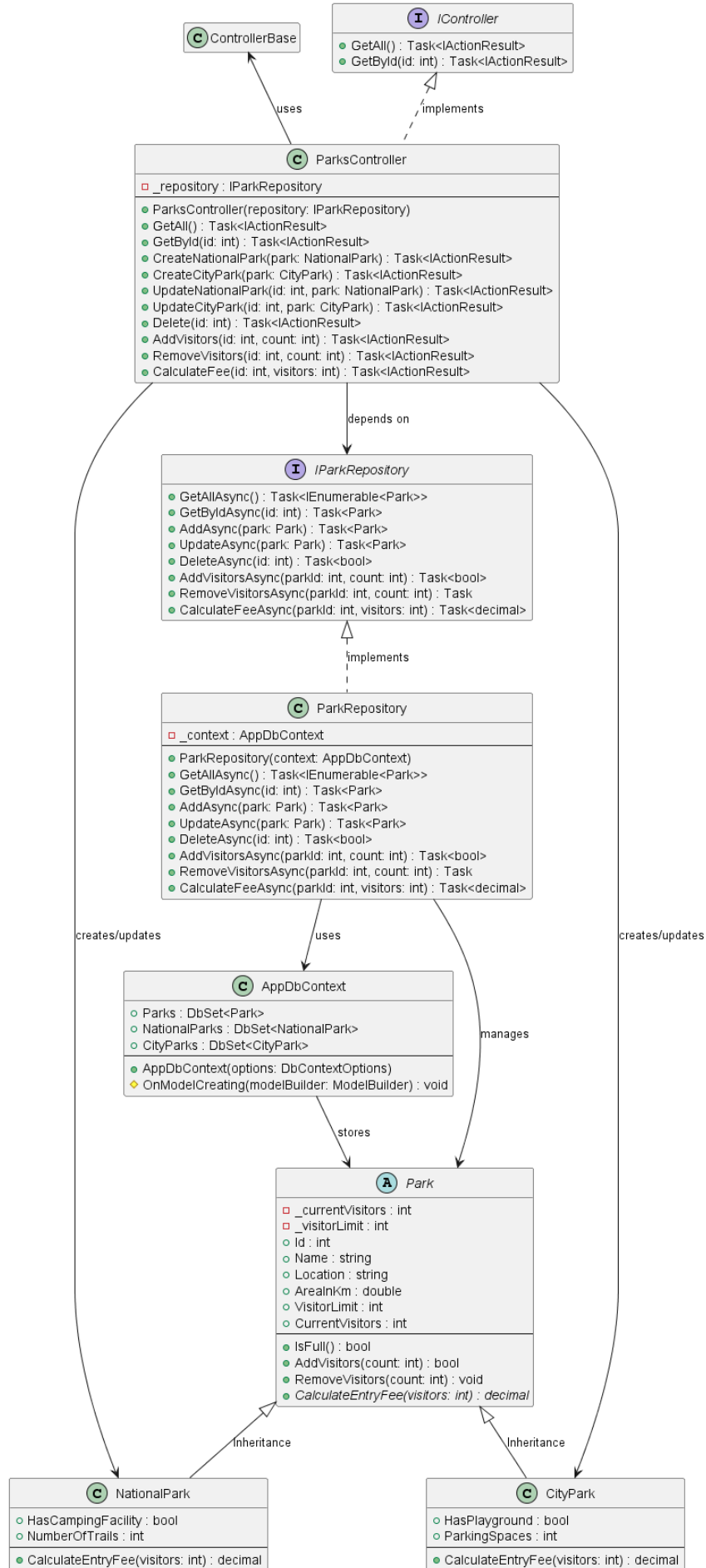
❖ *Yardımcı Öğretim Elemanı: Arş. Gör. Furkançan DEMİRCAN*

*E-posta: [furkancan.demircan@samsun.edu.tr](mailto:furkancan.demircan@samsun.edu.tr)*

### Adım 1: Uygulamanın oluşturulması

- İlk hafta oluşturduğumuz **Solution (Çözüm)** “YZM205” dosyasını açınız.
  - Gereken dosyaları **Github** üzerinden indirebilirsiniz.
- Solution dosyası içerisinde yeni bir “**ASP.NET Core Web API**” oluşturun.
  - Oluşturulan **ASP.NET Core Web API** projesine “**Week11**” ismi verin ve “**.Net 8**” üzerinden projeyi oluşturun.
  - “**Nuget Package Management (Nuget Paket Yönetici)**” kullanarak gerekli olan paketleri yükleyin.
    - EntityFrameworkCore
    - EntityFrameworkCore.Tools
    - EntityFrameworkCore.Design
    - EntityFrameworkCore.Sqlite

### Adım 2: Park Yönetim Projesi



Şekil 1. Proje sınıf diyagramı

- “Park” sınıfı için;
  - “\_currentVisitor” alanı park içerisindeki ziyaretçi sayısını göstermelidir.
  - “\_visitorLimit” alanı park içerisinde yer alabilecek maksimum ziyaretçi sayısını tutmalıdır.
  - “VisitorLimit” özelliği girilen sayının 0’dan büyük olduğunu kontrol etmelidir. Girilen sayı 0’dan küçük ise otomatik olarak 100 atamalıdır.
  - “CalculateEntryFee” metodu soyut olmalıdır.
  - “IsFull” metodu ziyaretçi sayısı limite ulaşınca True döndürmelidir.
  - “AddVisitor” metodu ziyaretçi sayısı limitten az ise “CurrentVisitor” özelliğini 1 artırmalı ve geriye True döndürmelidir. Öteki durumda geriye False döndürmelidir.
  - “RemoveVisitor” metodu ziyaretçi sayısını azaltmalıdır.
- “NationalPark” sınıfı için;
  - “CalculateEntryFee” metodu giriş ücretini hesaplamalıdır. Hesaplama işlemi;
    - Temel ücret: Ziyaretçiler \* 15 ABD doları
    - “HasCampingFacility” özelliği True ise: Ziyaretçiler \* 5 ABD doları ek ücret olarak ekleyin.
- “CityPark” sınıfı için;
  - “CalculateEntryFee” metodu giriş ücretini 0 olarak ayarlamalıdır.
- “ParkRepository” sınıfı için;
  - Tüm gerekli metotları EntityFrameworkCore operasyonlarına uygun olarak tanımlayın.
  - “AddVisitorAsync” ve “RemoveVisitorAsync” metotları “Park” sınıfı içerisinde yer alan metotları kullanmalıdır.
- “ParkController” sınıfı için;
  - “NationalPark” ve “CityPark” nesnelerinin oluşturulması için ayrı uç noktalar (endpoint) tanımlanmalıdır:
    - [HttpPost("national")] -> CreateNationalPark([FromBody] NationalPark park)
    - [HttpPost("city")] -> CreateCityPark([FromBody] CityPark park)
- “AppDbContext” sınıfı içerisinde;
  - “DbContext” implementasyonu gerçekleştirilmelidir.
  - “OnModelCreating” metodu geçersiz kılınarak Table-Per-Hierarchy (TPH) kalıtım stratejisi yapılandırılmalıdır.
  - “HasDiscriminator” yöntemi kullanılarak, “ParkType” alanı ayrıştırıcı (discriminator) sütun olarak belirtilmelidir.

```
protected override void OnModelCreating(ModelBuilder modelBuilder)
{
    modelBuilder.Entity<Park>()
        .HasDiscriminator<string>("ParkType")
        .HasValue<NationalPark>("NationalPark")
        .HasValue<CityPark>("CityPark");
}
```

Şekil 2. Table-Per-Hierarchy ekleme

- “Program” sınıfı içerisinde gerekli kod blokları eklenmelidir.

```
builder.Services.AddDbContext<AppDbContext>(options =>
    options.UseSqlite("Data Source=parks.db"));

// Register repository
builder.Services.AddScoped<IParkRepository, ParkRepository>();
```

Şekil 3. Veri tabanı bağlantısı

```
// Create database
using (var scope = app.Services.CreateScope())
{
    var db = scope.ServiceProvider.GetRequiredService<AppDbContext>();
    db.Database.EnsureCreated();
}
```

Şekil 4. Veri tabanı oluşturma

- Swagger içerisinde;
  - **National Park Oluşturma (POST /api/Parks/national)**

**Request Body:**

```
{
  "name": "Yellowstone",
  "location": "Wyoming, USA",
  "areaInKm": 8983,
  "visitorLimit": 500,
  "hasCampingFacility": true,
  "numberOfTrails": 12
}
```

**Response: 201 Created**

```
{
  "id": 1,
  "name": "Yellowstone",
  "currentVisitors": 0,
  "visitorLimit": 500
}
```

Şekil 5. Örnek 1

- **Ziyaretçi Ekleme (POST /api/Parks/1/visitors/add/50)**

**Response: 200 OK**

```
{
  "message": "Visitors added",
  "currentVisitors": 50
}
```

Şekil 6. Örnek 2

- **Giriş Ücreti Hesaplama (GET /api/Parks/1/fee/10)**

**Response: 200 OK**

```
{
  "parkId": 1,
  "visitors": 10,
  "entryFee": 200.00
}
```

Şekil 7. Örnek 3

**Github:** <https://github.com/FurkancanDemircan>