# ÖDEV 1: Movie Archive API (Proje İskeleti)

```
# 1. Klasör oluştur ve içine gir
mkdir MovieArchiveAPI
cd MovieArchiveAPI

# 2. Solution dosyasını oluştur
dotnet new sln -n MovieArchive

# 3. Projeleri oluştur
dotnet new webapi -n MovieArchive.API
dotnet new classlib -n MovieArchive.Business
dotnet new classlib -n MovieArchive.Data

# 4. Projeleri Solution'a ekle
dotnet sln add MovieArchive.API
dotnet sln add MovieArchive.Business
dotnet sln add MovieArchive.Data

# 5. Referansları (Bağımlılıkları) Ekle
# API -> Business'ı kullanır
dotnet add MovieArchive.API reference MovieArchive.Business
# Business -> Data'yı kullanır
dotnet add MovieArchive.Business reference MovieArchive.Data
```

---

# ÖDEV 2: Email Service API

```
mkdir EmailServiceAPI
cd EmailServiceAPI

dotnet new sln -n EmailService

dotnet new webapi -n EmailService.API
dotnet new classlib -n EmailService.Business
dotnet new classlib -n EmailService.Data

dotnet sln add EmailService.API
dotnet sln add EmailService.Business
dotnet sln add EmailService.Data

dotnet add EmailService.API reference EmailService.Business
dotnet add EmailService.Business reference EmailService.Data
```

---

# ÖDEV 3: Survey App API

```
mkdir SurveyAppAPI
cd SurveyAppAPI
```

```
dotnet new sln -n SurveyApp

dotnet new webapi -n SurveyApp.API
dotnet new classlib -n SurveyApp.Business
dotnet new classlib -n SurveyApp.Data

dotnet sln add SurveyApp.API
dotnet sln add SurveyApp.Business
dotnet sln add SurveyApp.Data

dotnet add SurveyApp.API reference SurveyApp.Business
dotnet add SurveyApp.Business reference SurveyApp.Data
```

---

# ÖDEV 4: Forum Platform API

```
mkdir ForumPlatformAPI
cd ForumPlatformAPI

dotnet new sln -n ForumPlatform

dotnet new webapi -n ForumPlatform.API
dotnet new classlib -n ForumPlatform.Business
dotnet new classlib -n ForumPlatform.Data

dotnet sln add ForumPlatform.API
dotnet sln add ForumPlatform.Business
dotnet sln add ForumPlatform.Data

dotnet add ForumPlatform.API reference ForumPlatform.Business
dotnet add ForumPlatform.Business reference ForumPlatform.Data
```

---

# ÖDEV 5: Event Manager API (Özel Doğrulamalar)

**Adım 1: Projeyi Oluşturma** Terminalde şu komutu çalıştırın:

```
dotnet new webapi -n EventManagerAPI
cd EventManagerAPI
```

*(Daha sonra projeyi VS Code veya Visual Studio ile açın)*

**Adım 2: Validation Attribute'larını Yazma** Proje içinde `ValidationAttributes` adında bir klasör açın ve içine aşağıdaki 3 sınıfı oluşturun.

*Dosya: ValidationAttributes/NoPastDateAttribute.cs*

```
using System.ComponentModel.DataAnnotations;

namespace EventManagerAPI.ValidationAttributes
```

```
{
    public class NoPastDateAttribute : ValidationAttribute
    {
        protected override ValidationResult? IsValid(object? value,
ValidationContext validationContext)
        {
            // Değer boşsa kontrol etme (Required attribute'u ayrı kullanılır)
            if (value is DateTime dateValue)
            {
                if (dateValue < DateTime.Now)
                {
                    return new ValidationResult("Etkinlik tarihi geçmişte
olamaz.");
                }
            }
            return ValidationResult.Success;
        }
    }
}
```

*Dosya: ValidationAttributes/MustStartWithUppercaseAttribute.cs*

```
using System.ComponentModel.DataAnnotations;

namespace EventManagerAPI.ValidationAttributes
{
    public class MustStartWithUppercaseAttribute : ValidationAttribute
    {
        protected override ValidationResult? IsValid(object? value,
ValidationContext validationContext)
        {
            if (value is string str && !string.IsNullOrEmpty(str))
            {
                // İlk harfi kontrol et
                if (!char.IsUpper(str[0]))
                {
                    return new ValidationResult("İsim büyük harfle
başlamalıdır.");
                }
            }
            return ValidationResult.Success;
        }
    }
}
```

*Dosya: ValidationAttributes/DescriptionMustNotContainNameAttribute.cs (Cross-Property Validation)*

```
using System.ComponentModel.DataAnnotations;
using EventManagerAPI.Models; // Modelin namespace'i

namespace EventManagerAPI.ValidationAttributes
{
    public class DescriptionMustNotContainNameAttribute : ValidationAttribute
    {
```

```
        protected override ValidationResult? IsValid(object? value,
ValidationContext validationContext)
        {
            // validationContext.ObjectInstance bize o anki Model'in tamamını
verir
            var eventItem = (Event)validationContext.ObjectInstance;

            // Null check yapalım
            if (!string.IsNullOrEmpty(eventItem.Description) &&
!string.IsNullOrEmpty(eventItem.Name))
            {
                // Description içinde Name geçiyor mu? (Büyük/küçük harf
duyarsız yaptım)
                if
(eventItem.Description.ToLower().Contains(eventItem.Name.ToLower()))
                {
                    return new ValidationResult("Açıklama alanı, etkinlik adını
içeremez.");
                }
            }

            return ValidationResult.Success;
        }
    }
}
```

**Adım 3: Modeli Oluşturma ve Attribute'ları Ekleme** Proje içinde `Models` klasörü açın (yoksa) ve `Event.cs` dosyasını oluşturun.

*Dosya: Models/Event.cs*

```
using System.ComponentModel.DataAnnotations;
using EventManagerAPI.ValidationAttributes; // Attribute'ların olduğu yer

namespace EventManagerAPI.Models
{
    public class Event
    {
        public int Id { get; set; }

        [Required]
        [MustStartWithUppercase] // 2. Kural
        public string Name { get; set; }

        [Required]
        [DescriptionMustNotContainName] // 3. Kural
        public string Description { get; set; }

        [Required]
        [NoPastDate] // 1. Kural
        public DateTime EventDate { get; set; }
    }
}
```

**Adım 4: Controller Oluşturma** `Controllers` klasörü altına `EventsController.cs` ekleyin.

*Dosya: Controllers/EventsController.cs*

```
using EventManagerAPI.Models;
using Microsoft.AspNetCore.Mvc;

namespace EventManagerAPI.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class EventsController : ControllerBase
    {
        [HttpPost]
        public IActionResult CreateEvent([FromBody] Event newEvent)
        {
            // [ApiController] attribute'u sayesinde ModelState.IsValid kontrolü
            // otomatik yapılır ve hata varsa 400 Bad Request döner.

            // Buraya geldi ise validation başarılıdır.
            return Ok($"Etkinlik başarıyla oluşturuldu: {newEvent.Name}");
        }
    }
}
```

## Test Etme (Swagger)

Projeyi çalıştırdığınızda (`dotnet run` veya F5), Swagger arayüzüne gidin ve POST isteği atarken şunları deneyin:

1. **Hata Testi 1:** `EventDate`'i dünkü bir tarih yapın. -> "Etkinlik tarihi geçmişte olamaz" hatası almalısınız.
2. **Hata Testi 2:** `Name` alanını "konser" (küçük harfle) yazın. -> "İsim büyük harfle başlamalıdır" hatası almalısınız.
3. **Hata Testi 3:** `Name`="Tarkan", `Description`="Tarkan konseri çok güzel olacak" yapın. -> "Açıklama alanı, etkinlik adını içeremez" hatası almalısınız.