.NET Web API Geliştirme Ödevi-1: Film Arşivi API'si

Kişisel bir film arşivini yönetmek için bir Web API geliştireceksiniz. Bu API, yeni filmler eklemeye, mevcut filmleri listelemeye, güncellemeye ve silmeye olanak tanıyacak.

Bir kaç bilgi:

1. Filmi temsil edecek tablonun aşağıdaki bilgileri tutması gerekmektedir:
    1. `Id`
    2. `Title`
    3. `Director`
    4. `ReleaseYear`
2. CRUD Endpointleri:
    1. **Tüm Filmleri Listeleme:**
        - `GET` isteklerini `/api/movies` adresine karşılamalıdır.
    2. **ID ile Tek Film Getirme:**
        - `GET` isteklerini `/api/movies/{id}` adresine karşılamalıdır.
    3. **Yeni Film Ekleme:**
        - `POST` isteklerini `/api/movies` adresine karşılamalıdır.
    4. **Filmi Güncelleme:**
        - `PUT` isteklerini `/api/movies/{id}` adresine karşılamalıdır.
    5. **Filmi Silme:**
        - `DELETE` isteklerini `/api/movies/{id}` adresine karşılamalıdır.
3. API testleri için projenizde Swagger kullanın.
4. Swagger üzerinden yukarıda geliştirdiğiniz tüm endpoint'leri test edin.
5. Hem başarılı senaryoları (bir filmi bulma, ekleme, silme) hem de hatalı senaryoları (var olmayan bir ID ile işlem yapma) deneyerek API'nızın doğru HTTP durum kodlarını döndürdüğünü doğrulayın.

Harika bir fikir! Verdiğiniz formatı kullanarak, öğrencilerin aynı temel becerileri farklı iş senaryoları üzerinde tekrar ederek pekiştirmelerini sağlayacak 4 adet yeni ödev hazırladım.

Acunmedya Akademi 13.Dönem Backend Eğitimi Sınıfı için Engin Niyazi Ergül tarafından hazırlanmıştır.

.NET Web API Geliştirme Ödevi-2: Blog Yönetim API'si

Basit bir blog platformu için arka uç (backend) hizmeti geliştireceksiniz. Geliştireceğiniz API, blog gönderilerinin (posts) oluşturulmasını, listelenmesini, güncellenmesini ve silinmesini yönetecek.

Bir kaç bilgi:

1. Blog gönderisini (Post) temsil edecek tablonun aşağıdaki bilgileri tutması gerekmektedir:
    1. `Id`
    2. `Title` (Başlık)
    3. `Content` (İçerik)
    4. `Author` (Yazar)
    5. `PublishedDate` (Yayınlanma Tarihi)
2. CRUD Endpointleri:
    1. **Tüm Gönderileri Listeleme:**
        - `GET` isteklerini `/api/posts` adresine karşılamalıdır.
    2. **ID ile Tek Gönderi Getirme:**
        - `GET` isteklerini `/api/posts/{id}` adresine karşılamalıdır.
    3. **Yeni Gönderi Ekleme:**
        - `POST` isteklerini `/api/posts` adresine karşılamalıdır.
    4. **Gönderiyi Güncelleme:**
        - `PUT` isteklerini `/api/posts/{id}` adresine karşılamalıdır.
    5. **Gönderiyi Silme:**
        - `DELETE` isteklerini `/api/posts/{id}` adresine karşılamalıdır.
3. API testleri için projenizde Swagger kullanın.
4. Swagger üzerinden yukarıda geliştirdiğiniz tüm endpoint'leri test edin.
5. Hem başarılı senaryoları (bir gönderiyi bulma, ekleme, silme) hem de hatalı senaryoları (var olmayan bir ID ile işlem yapma) deneyerek API'nızın doğru HTTP durum kodlarını döndürdüğünü doğrulayın.

Acunmedya Akademi 13.Dönem Backend Eğitimi Sınıfı için Engin Niyazi Ergül tarafından hazırlanmıştır.

.NET Web API Geliştirme Ödevi-3: Görev Yönetimi (To-Do) API'si

Kişisel bir görev yöneticisi veya "To-Do" uygulaması için bir API geliştireceksiniz. Bu API, kullanıcıların günlük görevlerini yönetmelerine olanak tanıyacak.

Bir kaç bilgi:

1. Görevi (Task) temsil edecek tablonun aşağıdaki bilgileri tutması gerekmektedir:
    1. `Id`
    2. `Title` (Başlık)
    3. `Description` (Açıklama)
    4. `IsCompleted` (Tamamlandı mı? - boolean)
    5. `DueDate` (Bitiş Tarihi)
2. CRUD Endpointleri:
    1. **Tüm Görevleri Listeleme:**
        - `GET` isteklerini `/api/tasks` adresine karşılamalıdır.
    2. **ID ile Tek Görev Getirme:**
        - `GET` isteklerini `/api/tasks/{id}` adresine karşılamalıdır.
    3. **Yeni Görev Ekleme:**
        - `POST` isteklerini `/api/tasks` adresine karşılamalıdır.
    4. **Görevi Güncelleme:**
        - `PUT` isteklerini `/api/tasks/{id}` adresine karşılamalıdır.
    5. **Görevi Silme:**
        - `DELETE` isteklerini `/api/tasks/{id}` adresine karşılamalıdır.
3. API testleri için projenizde Swagger kullanın.
4. Swagger üzerinden yukarıda geliştirdiğiniz tüm endpoint'leri test edin.
5. Hem başarılı senaryoları (bir görevi bulma, ekleme, silme) hem de hatalı senaryoları (var olmayan bir ID ile işlem yapma) deneyerek API'nızın doğru HTTP durum kodlarını döndürdüğünü doğrulayın.

Acunmedya Akademi 13.Dönem Backend Eğitimi Sınıfı için Engin Niyazi Ergül tarafından hazırlanmıştır.

.NET Web API Geliştirme Ödevi-4: Ürün Kataloğu API'si

Küçük bir e-ticaret sitesinin backendini yazdığınızı hayal edin. İlk göreviniz, ürün kataloğunu yönetmek için bir API oluşturmak.

Bir kaç bilgi:

1. Ürünü (Product) temsil edecek tablonun aşağıdaki bilgileri tutması gerekmektedir:
    1. `Id`
    2. `Name` (Ürün Adı)
    3. `Description` (Açıklama)
    4. `Price` (Fiyat - decimal)
    5. `StockQuantity` (Stok Adedi)
2. CRUD Endpointleri:
    1. **Tüm Ürünleri Listeleme:**
        - `GET` isteklerini `/api/products` adresine karşılamalıdır.
    2. **ID ile Tek Ürün Getirme:**
        - `GET` isteklerini `/api/products/{id}` adresine karşılamalıdır.
    3. **Yeni Ürün Ekleme:**
        - `POST` isteklerini `/api/products` adresine karşılamalıdır.
    4. **Ürünü Güncelleme:**
        - `PUT` isteklerini `/api/products/{id}` adresine karşılamalıdır.
    5. **Ürünü Silme:**
        - `DELETE` isteklerini `/api/products/{id}` adresine karşılamalıdır.
3. API testleri için projenizde Swagger kullanın.
4. Swagger üzerinden yukarıda geliştirdiğiniz tüm endpoint'leri test edin.
5. Hem başarılı senaryoları (bir ürünü bulma, ekleme, silme) hem de hatalı senaryoları (var olmayan bir ID ile işlem yapma) deneyerek API'nızın doğru HTTP durum kodlarını döndürdüğünü doğrulayın.

Acunmedya Akademi 13.Dönem Backend Eğitimi Sınıfı için Engin Niyazi Ergül tarafından hazırlanmıştır.

.NET Web API Geliştirme Ödevi-5: Müşteri Kayıt API'si

Bir CRM (Müşteri İlişkileri Yönetimi) uygulamasının temelini atacaksınız. Bu API, sisteme yeni müşteri kayıtları eklemek ve mevcut müşterileri yönetmekten sorumlu olacak.

Bir kaç bilgi:

1. Müşteriyi (Customer) temsil edecek tablonun aşağıdaki bilgileri tutması gerekmektedir:
    1. `Id`
    2. `FirstName` (Adı)
    3. `LastName` (Soyadı)
    4. `Email` (E-posta adresi)
    5. `PhoneNumber` (Telefon Numarası)
2. CRUD Endpointleri:
    1. **Tüm Müşterileri Listeleme:**
        - `GET` isteklerini `/api/customers` adresine karşılamalıdır.
    2. **ID ile Tek Müşteri Getirme:**
        - `GET` isteklerini `/api/customers/{id}` adresine karşılamalıdır.
    3. **Yeni Müşteri Ekleme:**
        - `POST` isteklerini `/api/customers` adresine karşılamalıdır.
    4. **Müşteriyi Güncelleme:**
        - `PUT` isteklerini `/api/customers/{id}` adresine karşılamalıdır.
    5. **Müşteriyi Silme:**
        - `DELETE` isteklerini `/api/customers/{id}` adresine karşılamalıdır.
3. API testleri için projenizde Swagger kullanın.
4. Swagger üzerinden yukarıda geliştirdiğiniz tüm endpoint'leri test edin.
5. Hem başarılı senaryoları (bir müşteriyi bulma, ekleme, silme) hem de hatalı senaryoları (var olmayan bir ID ile işlem yapma) deneyerek API'nızın doğru HTTP durum kodlarını döndürdüğünü doğrulayın.

Başarılar!