



**Code
Academy**



Dėstytojas

Robertas Ūselis

Request'ų validavimas ir Darbas su nuotraukom

Data



Šiandien išmoksime

01

Kaip validuoti užklausa?

02

Kaip uploadinti, download'inti nuotrauką iš db?

03

Kaip pavaliduoti uploadinamos nuotraukos savybes?



Vartotojo įvesties validavimo svarba.

Saugumas

Duomenų vientisumas

Vartotojo patirtis

Teisiniai ir atitikties įsipareigojimai

Našumo optimizavimas

Reputacija ir pasitikėjimas



Data Annotation Validation

```
public class CreateToDoDto
{
    [Required]
    2 references | 2/2 passing
    public string Type { get; set; }

    [Required]
    [StringLength(100, MinimumLength = 3)]
    2 references | 2/2 passing
    public string Content { get; set; }

    [Range(0, 10)]
    2 references | 2/2 passing
    public int Difficulty { get; set; }

    [RegularExpression(@"^\d{4}-\d{2}-\d{2}$", ErrorMessage = "Date must be in YYYY-MM-DD format")]
    2 references | 2/2 passing
    public DateTime? EndDate { get; set; }


    [Required]
    2 references | 2/2 passing
    public bool IsCompleted { get; set; }
}
```



Kaip vyksta validavimas

Kai gaunama užklausa validacija vyksta „po kapišonu“ pagal nurodytas taisykles prieš perduodant dto kontrolieriui

```
[Route("api/[controller]")]
[ApiController]
0 references
public class ValidationController : ControllerBase
{
    [HttpPost]
    0 references
    public IActionResult Post([FromBody] CreateToDoDto dto)
    {
        // Create todo logic
        return Ok();
    }
}
```





Custom Validation. return bool

```
[AttributeUsage(AttributeTargets.Property, AllowMultiple = false)]  
1 reference  
public class ValidTodoTypeAttribute: ValidationAttribute  
{  
    private readonly string[] _validTypes = {"work", "home", "other"};  
  
    0 references  
    public override bool IsValid(object value)  
    {  
        var type = value as string;  
        if (type == null)  
        {  
            return false;  
        }  
  
        return _validTypes.Contains(type.ToLower());  
    }  
}
```



Custom Validation. return ValidationResult

```
public class NoWeekendsAttribute : ValidationAttribute
{
    0 references
    protected override ValidationResult IsValid(object value, ValidationContext validationContext)
    {
        if (value is DateTime)
        {
            var dateValue = (DateTime)value;
            if (dateValue.DayOfWeek == DayOfWeek.Saturday || dateValue.DayOfWeek == DayOfWeek.Sunday)
            {
                return new ValidationResult("Date cannot be on a weekend.");
            }
        }
        else
        {
            return new ValidationResult("Invalid date format.");
        }



        return ValidationResult.Success;
    }
}
```



Custom Validation. AttributeUsage

AttributeTargets: nurodo programos elementus, kuriems gali būti taikomas atributas. Pavyzdžiui, galite apriboti, kad atributas būtų taikomas tik klasėms, metodams, savybėms ir t. t.

AllowMultiple: nustato, ar atributas gali būti taikomas daugiau nei vieną kartą vienam programos elementui.



```
[AttributeUsage(AttributeTargets.Property, AllowMultiple = false)]  
1 reference  
public class ValidTodoTypeAttribute: ValidationAttribute  
{  
    private readonly string[] _validTypes = {"work", "home", "other"};  
  
    0 references  
    public override bool IsValid(object value)  
    {  
        var type = value as string;  
        if (type == null)  
        {  
            return false;  
        }  
  
        return _validTypes.Contains(type.ToLower());  
    }  
}
```




Custom Validation

```
public class CreateToDoDto
{
    [Required]
    [ValidTodoType]
    2 references | 2/2 passing
    public string Type { get; set; }

    [Required]
    [StringLength(100, MinimumLength = 3)]
    2 references | 2/2 passing
    public string Content { get; set; }

    [Range(0, 10)]
    2 references | 2/2 passing
    public int Difficulty { get; set; }

    [NoWeekends(ErrorMessage = "Todo end date cannot be on a weekend.")]
    [RegularExpression(@"^\d{4}-\d{2}-\d{2}$", ErrorMessage = "Date must be in YYYY-MM-DD format")]
    2 references | 2/2 passing
    public DateTime? EndDate { get; set; }

    [Required]
    2 references | 2/2 passing
    public bool IsCompleted { get; set; }
}
```



Užduotis nr. 1

Sugalvokite bei sudėkite buld-in ir custom validacijas CreateCarDto klasei

```
public class CreateCarDto
{
    0 references
    public string Make { get; set; }
    0 references
    public string Model { get; set; }
    0 references
    public int Year { get; set; }
    0 references
    public string LicensePlate { get; set; }
    0 references
    public double Mileage { get; set; }
    0 references
    public string Type { get; set; }
    0 references
    public string ImageUrl { get; set; }
}
```



Darbas su failais



Susikuriame objektą, kurį naudosime priimdami nuotraukos upload

Pirma mums reikia pavaliduoti, ar upload'inamas objektas yra nuotraukos failų tipo ir ar atitinka mūsų norimą dydį

```
public class ImageUploadRequest
{
    public IFormFile Image { get; set; }
}
```



Requesto validavimas

Norint validuoti failo dydį ir tipą galima susikurti du atributus.

Pirmasis atributas bus atsakingas už dydį, o antrasis už tipą



MaxFileSizeAttribute

```
public class MaxFileSizeAttribute : ValidationAttribute
{
    private readonly int _maxFileSize;
    public MaxFileSizeAttribute(int maxFileSize)
    {
        _maxFileSize = maxFileSize;
    }

    protected override ValidationResult IsValid(
        object value, ValidationContext validationContext)
    {
        if (value is IFormFile file)
        {
            if (file.Length > _maxFileSize)
            {
                return new ValidationResult(GetErrorMessage());
            }
        }

        return ValidationResult.Success;
    }

    public string GetErrorMessage()
    {
        return $"Maximum allowed file size is { _maxFileSize} bytes.";
    }
}
```



AllowedExtensionsAttribute

```
public class AllowedExtensionsAttribute : ValidationAttribute
{
    private readonly string[] _extensions;
    public AllowedExtensionsAttribute(string[] extensions)
    {
        _extensions = extensions;
    }

    protected override ValidationResult IsValid(
        object value, ValidationContext validationContext)
    {
        if (value is IFormFile file)
        {
            var extension = Path.GetExtension(file.FileName);
            if (!_extensions.Contains(extension.ToLower()))
            {
                return new ValidationResult(GetErrorMessage());
            }
        }

        return ValidationResult.Success;
    }

    public string GetErrorMessage()
    {
        return $"This photo extension is not allowed!";
    }
}
```



Panaudojus atributus, klasė dabar atrodo taip

Failų dydį reguliuojames patys, extension'ų tipų taipogi yra daugiau, tai pasirenkame pagal poreikį.

```
public class ImageUploadRequest
{
    [MaxFileSize(5 * 1024 * 1024)]
    [AllowedExtensions(new string[] { ".png", ".jpg" })]
    public IFormFile Image { get; set; }
}
```




Upload endpoint

Endpoint'as atrodytų taip, verta būtų pagalvoti konvertavimą į bitus perkelti į servisą, kuris savo viduje pakonvertavęs iškvieštų Save repozitorijos metodą

```
[ApiController]
[Route("[controller]")]
public class ImageController : ControllerBase
{
    [HttpPost("Upload")]
    public ActionResult UploadImage([FromForm] ImageUploadRequest imageRequest)
    {
        using var memoryStream = new MemoryStream();
        imageRequest.Image.CopyTo(memoryStream);
        // you can save this to database now
        var imageBytes = memoryStream.ToArray();
        return Ok();
    }
}
```



Download endpoint

Download endpoint'as turėtų pasiimti bitus iš duomenų bazės ir grąžinti juos kaip File

Pradžioj susikuriame repozitoriją, kuri paimtų duomenis iš duomenų bazės

```
public class ImageRepository : IImageRepository
{
    private readonly ApplicationDbContext _context;
    public Image GetImage(Guid id)
    {
        return _context.Images.First(x => x.Id == id);
    }
}
```

```
public class ImageService : IImageService
{
    private readonly IImageRepository _imageRepository;
    public Image GetImage(Guid id)
    {
        return _imageRepository.GetImage(id);
    }
}
```

Tad controller'yje grąžiname rezultatą

```
[HttpGet("Download")]
public ActionResult DownloadImage([FromQuery] Guid id)
{
    var image = _imageService.GetImage(id);
    return File(image.ImageBytes, $"image/{image.ContentType}");
}
```



Užduotis nr. 1

- Sukurkite image upload/download API su validacijomis.
- Naujas funkcionalumas bus sukūrimas thumbnail tipo nuotraukos iš originalo.
- Thumbnail'as turės būt saugojamas kitoje lentelėje.
- Thumbnail'o esmė yra mažesnis dydis negu originalas.
- Šio funkcionalumo reikės pasieškoti patiems;)
- Turi būti galimybė get'inti individualiai tiek didelę nuotrauką, tiek thumbnail