

Øvelse 20 – Inputvalidering med model binding & C# attributter

Information

Formålet med denne øvelse er at stifte bekendtskab med model binding og validering ved hjælp af denne tilgang. Model binding betyder, at rammeværktøjet (ASP.NET Core) automatisk mapper modtaget data til en bestemt datatype.

I projektet `ValidationModelBinding` skal I arbejde med controllerklassen `NameController.cs`. Her vil I se, at metoden `ValidateName` ikke indeholder traditionel valideringslogik, men i stedet anvender `ModelState.IsValid` til at afgøre, om det modtagne objekt overholder valideringsreglerne.

Valideringen baseres på attributter, der er sat på egenskaberne i modellen. I denne øvelse skal I tilføje flere valideringsattributter til `Firstname` og `Lastname` i klassen `PersonDTO` (findes i projektmappen `Models`).

Brug følgende dokumentation til hjælp med attributter:

- [Regex Attribute](#)
- [String Minimum Length](#)
- [String Maximum Length](#)
- [String length](#)

Fokus på konceptet, ikke teknologistakken

Bemærk:

Denne øvelse handler om *inputvalidering som sikkerhedsprincip* i webapplikationer – ikke om webudvikling eller ASP.NET specifikt.

Øvelsen fokuserer på **sikkerhedsdesign og håndtering af input ved tillidsgrænser**, hvilket gælder på tværs af teknologier og programmeringssprog.

Eksemplerne er lavet i en .NET-webapplikation, men principperne er **generelle** og overførbare til andre teknologier som Java, Python eller JavaScript.

Øvelserne tager ikke nødvendigvis højde for best practices eller konventioner i det anvendte sprog eller framework.

Hvis du vil forstå, hvordan inputvalidering implementeres korrekt og sikkert i netop den teknologistak, du arbejder med, bør du konsultere den officielle dokumentation.

□ Instruktioner

1. Tilføj valideringsattributter, så `Firstname` og `Lastname` ikke kan være længere end 20 tegn.
 2. Tilføj valideringsattributter, så `Firstname` og `Lastname` ikke kan være kortere end 3 tegn.
 3. Tilføj en Regex-attribut, der sikrer, at `Firstname` og `Lastname` kun indeholder alfabetiske karakterer.
 4. Test valideringen ved at køre applikationen og afprøve forskellige inputs.
-

□ Refleksionsspørgsmål

1. Hvad er fordelene ved at anvende model binding og attributter til inputvalidering frem for manuel validering?
 2. Hvordan kan model binding forbedre sikkerheden i en webapplikation?
 3. Hvordan håndteres fejl og feedback til brugeren, når en model ikke validerer?
 4. Kan model binding alene sikre applikationen mod alle typer angreb? Hvorfor/hvorfor ikke?
-

□ Forslag til videre undersøgelse (*frivillig opgave*)

□ Bonus: Er rækkefølgen af validering garanteret i ASP.NET Core?

Undersøg i den officielle dokumentation for ASP.NET Core, **om der findes garanti for rækkefølgen**, hvormed valideringsattributter (fx `[Required]`, `[StringLength]`, `[RegularExpression]`) evalueres.

Forslag til undersøgelse: - Find og læs i dokumentationen om *model validation* og *data annotations* - Er det beskrevet, hvordan eller i hvilken rækkefølge attributterne bliver brugt? - Er rækkefølgen vigtig at kende i praksis?

□ *Hint:* Du kan også overveje, hvordan man med en brugerdefineret valideringsattribut (`ValidationAttribute`) kan påvirke valideringsflowet – og hvilke muligheder det giver i forhold til fx sikkerhedsdesign.

▮ *Bonus info:* Man kan også begrænse det maksimale *Http content body* størrelse som API'et håndter, Se F.eks [Dette eksempe](#)

Kobling til relevante standarder og principper

Når I arbejder med inputvalidering via model binding og attributter i denne øvelse, relaterer det sig til flere centrale principper og standarder for software- og informationssikkerhed:

Element i øvelsen	Relateret standard / princip	Kommentar
Inputvalidering	ISO/IEC 27002:2022, afsnit 8.26 (applikationssikkerhed)	Anbefaler validering og integritetstjek af input til en applikation.
	OWASP ASVS 5.0.0, kontrol 2.2. (Input Validation)	Krav om validering af data
	Secure by Design (Johnsson et al.)	Understreger vigtigheden af at beskytte tillidsgrænser og sikre korrekt inputvalidering.

▮ *Brug af model binding og attributter er et eksempel på at automatisere og integrere sikkerhed direkte i applikationsdesignet, hvilket understøtter principperne i Secure by Design.*

Last update: 2025-10-30 12:07:19