**Model Binding**

**1. What is Model Binding in ASP.NET Core MVC?**

**Answer:**  
Model Binding in ASP.NET Core MVC automatically maps HTTP request data (such as form values, query strings, or route data) to action method parameters or properties of a model. It simplifies data handling by converting string-based request values into strongly typed objects.

**2. What are the different sources from which Model Binding fetches data?**

**Answer:**  
Model Binding in ASP.NET Core MVC fetches data from the following sources:

1. **Form Values** (e.g., Request.Form)
2. **Route Data** (e.g., {id} in /{controller}/{action}/{id})
3. **Query String Parameters** (e.g., ?name=John)
4. **Header Values** (e.g., Request.Headers)
5. **Custom Value Providers** (if implemented)

**3. What is the difference between Model Binding and Model Validation?**

**Answer:**

* **Model Binding** maps incoming HTTP request data to action parameters or model properties.
* **Model Validation** checks if the mapped data is valid based on validation attributes (e.g., [Required], [Range]). If validation fails, it adds errors to ModelState.

**4. How do you customize Model Binding in ASP.NET Core MVC?**

**Answer:**  
You can customize Model Binding by:

1. **Using [FromQuery], [FromBody], [FromRoute], [FromForm], [FromHeader] attributes.**

public IActionResult GetStudent([FromQuery] int id) { ... }

1. **Creating a custom Model Binder by implementing IModelBinder**

public class CustomBinder : IModelBinder

{

public Task BindModelAsync(ModelBindingContext bindingContext)

{

string value = bindingContext.ValueProvider.GetValue("customKey").FirstValue;

bindingContext.Result = ModelBindingResult.Success(value);

return Task.CompletedTask;

}

}

**Tag Helpers**

**5. What are Tag Helpers in ASP.NET Core MVC?**

**Answer:**  
Tag Helpers are server-side components that help generate HTML dynamically. They are similar to HTML elements but allow Razor syntax for improved readability and maintainability.

Example:

<label asp-for="Name"></label>

<input asp-for="Name" class="form-control" />

This will generate:

<label for="Name">Name</label>

<input id="Name" name="Name" class="form-control" />

**6. What are some commonly used Tag Helpers in ASP.NET Core MVC?**

**Answer:**

1. **asp-for** → Binds an input element to a model property.
2. **asp-action** → Generates the URL for an action method.
3. **asp-controller** → Specifies the controller for an action.
4. **asp-route-{value}** → Adds route values dynamically.
5. **asp-validation-for** → Displays validation messages for a property.
6. **asp-append-version** → Appends a version number to static files for cache busting.

**7. How do you enable Tag Helpers in a Razor View?**

**Answer:**  
By default, Tag Helpers are enabled in ASP.NET Core MVC. If needed, you can explicitly add them using:

@addTagHelper \*, Microsoft.AspNetCore.Mvc.TagHelpers

This registers built-in Tag Helpers from ASP.NET Core MVC.

**8. Can we create a custom Tag Helper in ASP.NET Core MVC?**

**Answer:**  
Yes, you can create a custom Tag Helper by inheriting from TagHelper.

Example:

public class EmailTagHelper : TagHelper

{

public string Email { get; set; }

public override void Process(TagHelperContext context, TagHelperOutput output)

{

output.TagName = "a";

output.Attributes.SetAttribute("href", $"mailto:{Email}");

output.Content.SetContent(Email);

}

}

Usage in a Razor view:

<email email="user@example.com"></email>

It generates:

<a href="mailto:user@example.com">user@example.com</a>

**Data Annotations**

**9. What are Data Annotations in ASP.NET Core MVC?**

**Answer:**  
Data Annotations are attributes that help validate model properties and define metadata. They are used in Model Validation and UI rendering.

Example:

public class Student

{

[Required(ErrorMessage = "Name is required")]

[StringLength(50, ErrorMessage = "Name cannot exceed 50 characters")]

public string Name { get; set; }

[Range(18, 60, ErrorMessage = "Age must be between 18 and 60")]

public int Age { get; set; }

[EmailAddress(ErrorMessage = "Invalid email format")]

public string Email { get; set; }

}

**10. What are some commonly used Data Annotation attributes?**

**Answer:**

1. [Required] → Ensures a field is not empty.
2. [StringLength(50)] → Limits the string length.
3. [Range(1, 100)] → Restricts values to a range.
4. [EmailAddress] → Validates email format.
5. [Compare("Password")] → Compares two properties (e.g., for password confirmation).
6. [RegularExpression("regex pattern")] → Validates format using regex.

**11. How do you create a custom Data Annotation?**

**Answer:**  
You can create a custom validation attribute by inheriting from ValidationAttribute.

Example:

public class EvenNumberAttribute : ValidationAttribute

{

public override bool IsValid(object value)

{

if (value is int number)

return number % 2 == 0;

return false;

}

}

Usage:

csharp

CopyEdit

public class NumberModel

{

[EvenNumber(ErrorMessage = "Number must be even")]

public int Value { get; set; }

}

**12. How do you enable client-side validation for Data Annotations in ASP.NET Core MVC?**

**Answer:**  
ASP.NET Core MVC supports client-side validation using jQuery Unobtrusive Validation. Ensure the following scripts are included in \_Layout.cshtml or view:

<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery-validate/1.19.3/jquery.validate.min.js"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery-validation-unobtrusive/3.2.12/jquery.validate.unobtrusive.min.js"></script>

This enables automatic validation for inputs bound with [Required], [Range], etc.

**13. How do you display validation messages in a Razor View?**

**Answer:**  
Use asp-validation-for and ValidationSummary:

<form asp-action="Submit">

<input asp-for="Name" class="form-control" />

<span asp-validation-for="Name" class="text-danger"></span>

<button type="submit">Submit</button>

</form>

<div asp-validation-summary="ModelOnly" class="text-danger"></div>

**Interview Questions**

**Model Binding**

**1. What is Model Binding?**  
Model Binding in ASP.NET Core MVC automatically maps HTTP request data (like form inputs, query strings, or route values) to action method parameters or model properties. It helps in handling user input easily without manual parsing.

**2. Where does Model Binding fetch data from?**  
It fetches data from form fields, query strings, route parameters, headers, and even request bodies in some cases.

**3. What’s the difference between Model Binding and Model Validation?**  
Model Binding maps request data to a model, while Model Validation ensures the data is correct using validation rules like required fields or value ranges.

**4. How can you customize Model Binding?**  
We can use attributes like [FromQuery], [FromRoute], [FromBody], and [FromForm] to specify where data should come from. If needed, we can also create a custom model binder.

**Tag Helpers**

**5. What are Tag Helpers?**  
Tag Helpers allow us to use HTML-like syntax while still having Razor’s server-side power. They make Razor views cleaner and easier to read.

**6. Can you give examples of commonly used Tag Helpers?**  
Some commonly used ones are:

* asp-for (binds input fields to model properties)
* asp-action and asp-controller (generate links for actions)
* asp-validation-for (shows validation messages)

**7. How do you enable Tag Helpers in a Razor View?**  
They are enabled by default, but if needed, we can explicitly add them using @addTagHelper \*, Microsoft.AspNetCore.Mvc.TagHelpers.

**8. Can we create custom Tag Helpers?**  
Yes, by inheriting from TagHelper and overriding the Process method, we can define our own logic.

**Data Annotations**

**9. What are Data Annotations?**  
Data Annotations are attributes used to define validation rules for model properties, like making a field required or setting a range for numbers.

**10. What are some commonly used Data Annotations?**  
Some commonly used ones are:

* [Required] (makes a field mandatory)
* [StringLength] (limits string length)
* [Range] (specifies a numeric range)
* [EmailAddress] (validates email format)

**11. Can we create custom Data Annotations?**  
Yes, we can create a custom validation attribute by inheriting from ValidationAttribute and overriding the IsValid method.

**12. How does client-side validation work in ASP.NET Core MVC?**  
Client-side validation is enabled using jQuery Unobtrusive Validation, which works automatically when Data Annotations are applied.

**13. How do we display validation messages in Razor Views?**  
We use asp-validation-for with form fields and asp-validation-summary to show all errors in one place.