

## Sezione 13

# Paginare, ordinare e filtrare i dati

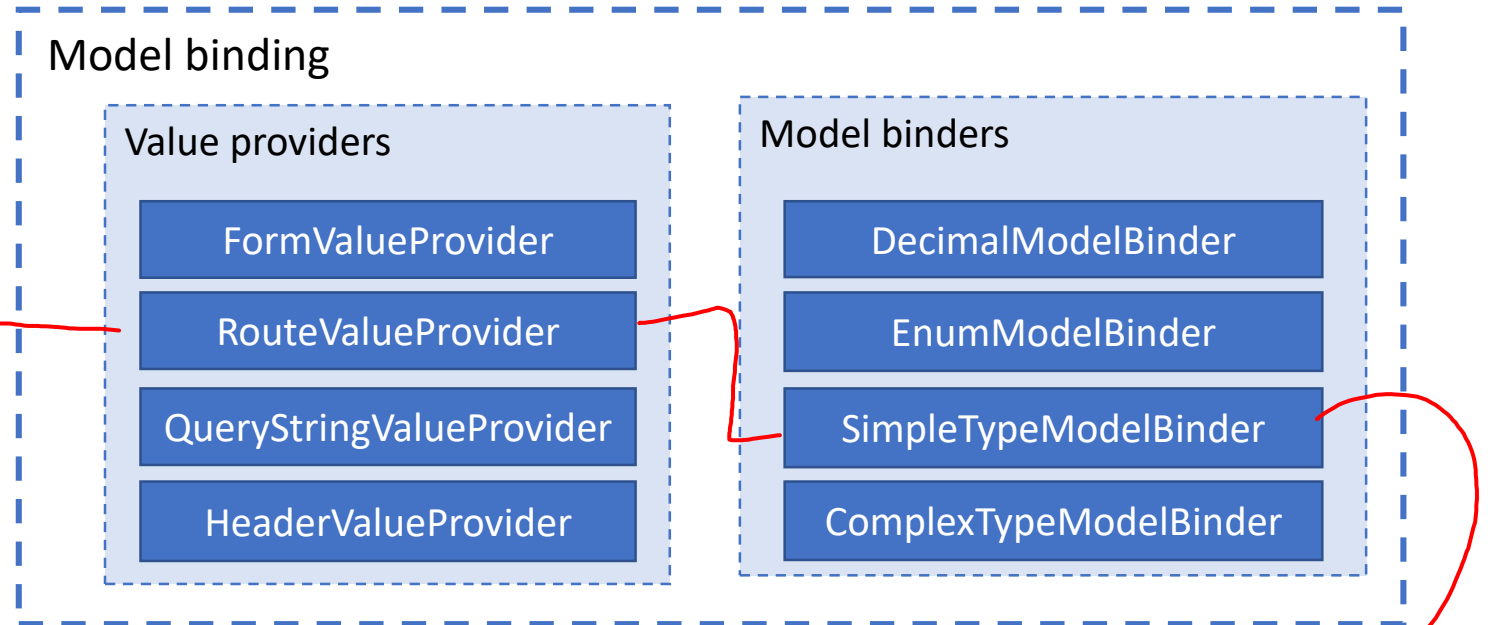
# Il model binding

- Mapping tra le informazioni della richiesta dell'utente e oggetti C#;
- Non dobbiamo fare niente per attivarlo: è già in funzione in MVC e WebAPI.



# II model binding

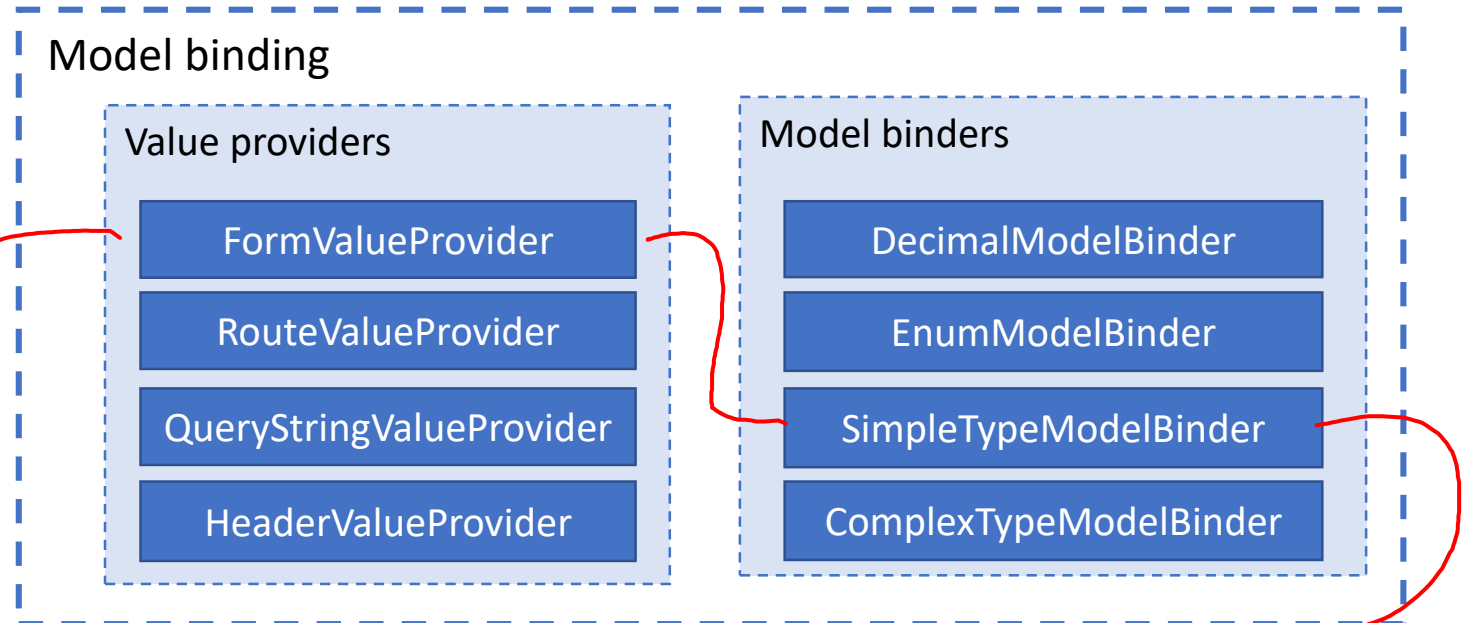
▼ General
Request URL: /Courses/Update/5?notify=1
Request Method: POST
Status Code: 200 OK
▼ Request Headers
Accept: text/html,application/xhtml+xml
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,it;q=0.8
▼ Query String Parameters
notify: 1
▼ Form Data
title: Nuovo Corso



```
public IActionResult Update(int id)
{
    // ...
}
```

# II model binding

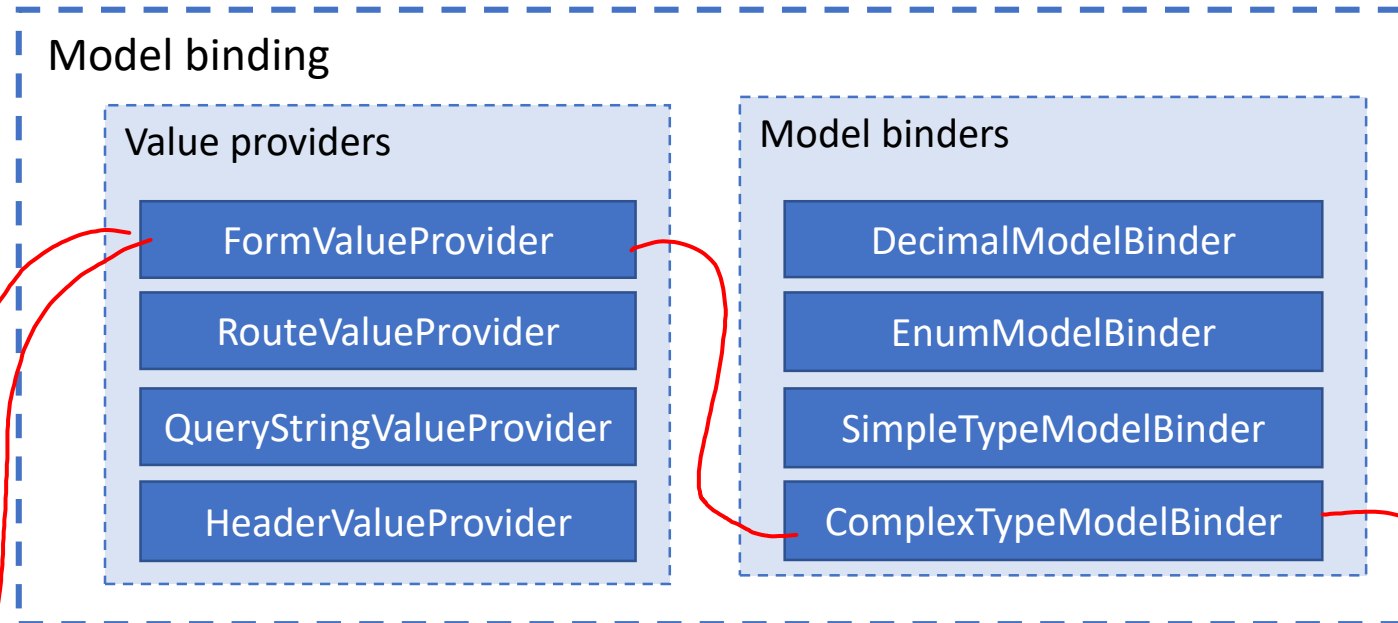
▼ General
Request URL: /Courses/Update/5?notify=1
Request Method: POST
Status Code: 200 OK
▼ Request Headers
Accept: text/html,application/xhtml+xml
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,it;q=0.8
▼ Query String Parameters
notify: 1
▼ Form Data
title: Nuovo Corso



```
public IActionResult Update(string title)
{
    // ...
}
```

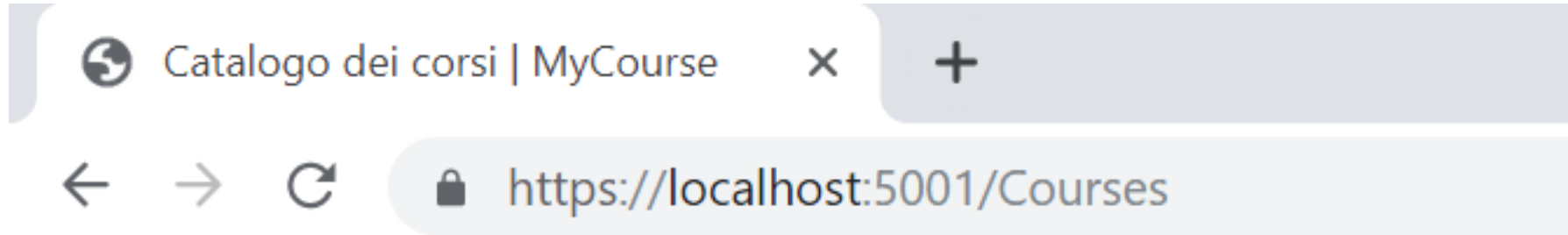
# II model binding

▼ General
Request URL: /Courses/Update/5?notify=1
Request Method: POST
Status Code: 200 OK
▼ Request Headers
Accept: text/html,application/xhtml+xml
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,it;q=0.8
▼ Query String Parameters
notify: 1
▼ Form Data
<u>title</u> : Nuovo Corso
<u>description</u> : Descrizione
<u>author</u> : Mario Rossi
<u>email</u> : info@example.org



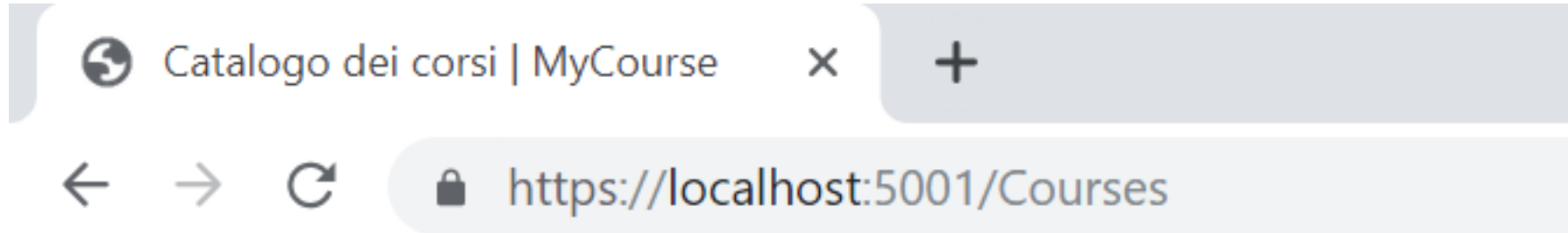
```
public IActionResult Update(MyViewModel model)
{
    // ...
}
```

# Valori di default



```
public async Task<IActionResult> Index(  
    string search, null  
    int page, 0  
    string orderby, null  
    bool ascending) false
```

# Valori di default



```
public async Task<IActionResult> Index(  
    string search = null,           null  
    int page = 1,                  1  
    string orderby = "price",      price  
    bool ascending = true)         true
```

# In certi casi usiamo gli attributi **From**...

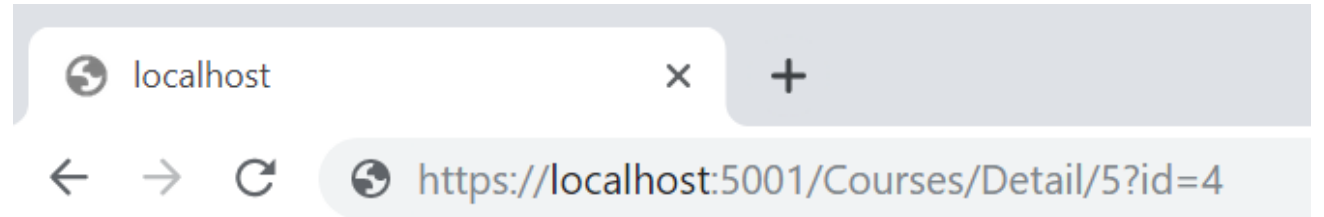
- [FromForm]
- [FromRoute]
- [FromQuery]
- [FromHeader]
- [FromServices]



# In certi casi usiamo gli attributi **From**...

- **[FromForm]**
- **[FromRoute]**
- **[FromQuery]**
- **[FromHeader]**
- **[FromServices]**

Se c'è ambiguità



```
public IActionResult Detail([FromQuery] int id)
{
    // ...
}
```

# In certi casi usiamo gli attributi **From...**

- [FromForm]
- [FromRoute]
- [FromQuery]
- **[FromHeader]**
- [FromServices]

Se il valore arriva dalle intestazioni

## ▼ Request Headers

**Accept:** text/html,application/xhtml+xml

**Accept-Encoding:** gzip, deflate, br

**Accept-Language:** en-US,en;q=0.9,it;q=0.8

```
public IActionResult Detail(  
    [FromHeader("Accept-Language")] string lang)  
{  
    // ...  
}
```

# In certi casi usiamo gli attributi **From...**





- [FromForm]
- [FromRoute]
- [FromQuery]
- [FromHeader]
- **[FromServices]**

Se vogliamo ricevere un servizio della grazie alla dependency injection

```
public IActionResult Detail(  
    [FromServices] IMemoryCache cache)  
{  
    // ...  
}
```

# Catalogo corsi

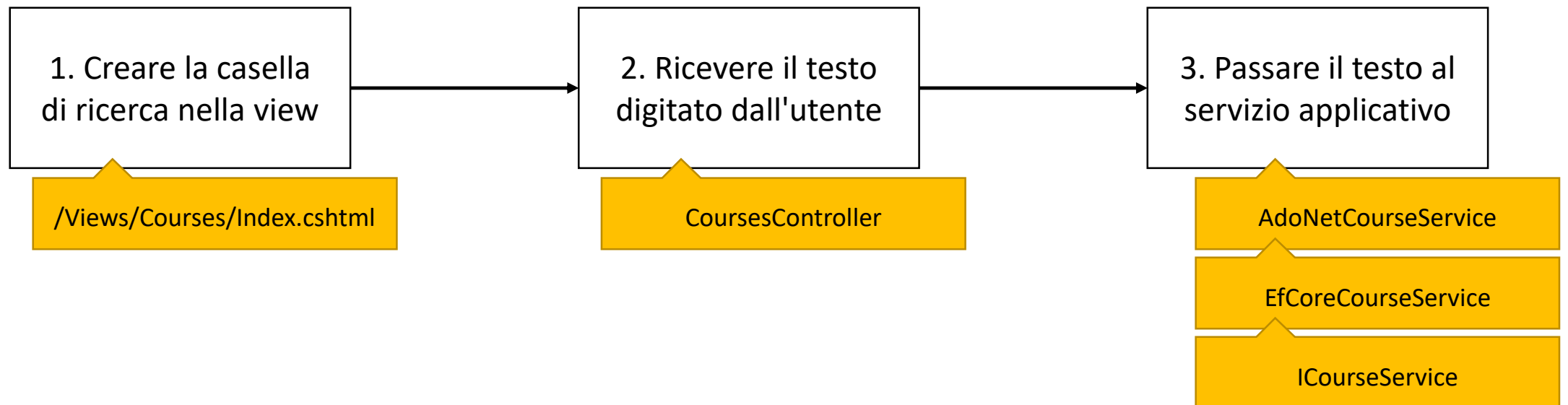


Titolo	Valutazione ▼	Prezzo	
 <b>Web marketing facile</b> <i>di Mario Rossi</i>	★★★★★	EUR 17.99 <del>EUR 19.99</del>	<a href="#">Dettagli</a>
 <b>L'ABC del fai da te</b> <i>di Mario Rossi</i>	★★★★★	EUR 7.99	<a href="#">Dettagli</a>
 <b>Come fare gli origami</b> <i>di Mario Rossi</i>	★★★★☆	EUR 30.99 <del>EUR 34.99</del>	<a href="#">Dettagli</a>
 <b>Corso di ceramica</b> <i>di Mario Rossi</i>	★★★★☆	USD 10.99 <del>USD 11.99</del>	<a href="#">Dettagli</a>

Punto n° 4

*Il catalogo dei corsi deve essere paginato (max 10 risultati per pagina) e deve supportare la ricerca per titolo e l'ordinamento per titolo, valutazione e prezzo corrente.*

# Il piano per implementare la ricerca



# Il tag helper form

*Controller e Action a cui inviare i dati*

```
<form asp-controller="Courses" asp-action="Index" method="get">  
  <input type="search" name="search">  
  <button>Cerca</button>  
</form>
```

*Causa l'invio  
del form*

*Determina il nome  
della chiave*

▼ General
Request URL: /Courses? <u>search</u> =Chitarra
Request Method: GET
Status Code: 🟢 200 OK
▼ Query String Parameters
<u>search</u> : Chitarra

*method  
get*

▼ General
Request URL: /Courses
Request Method: POST
Status Code: 🟢 200 OK
▼ Form Data
<u>search</u> : Chitarra

*method  
post*

# Quando scegliere method **get** o **post**?

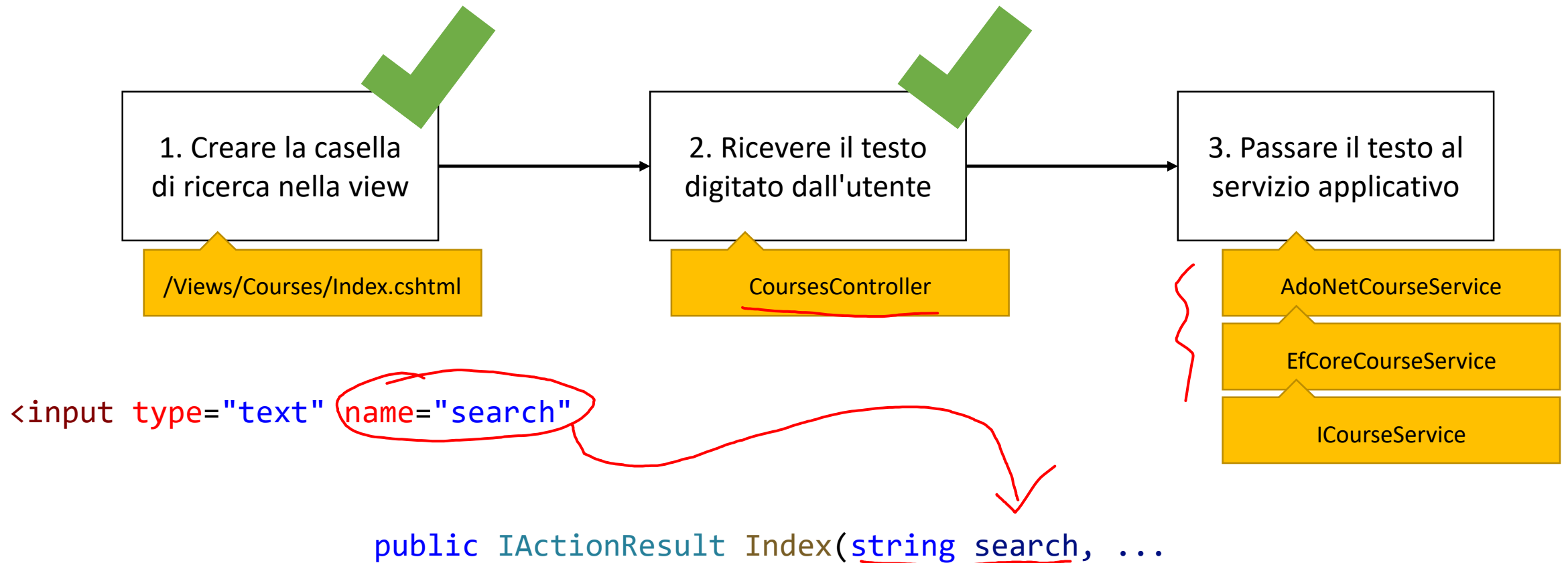
Scegliamo **get** quando vogliamo che le chiavi appaiano nell'URL:

- Così l'utente può facilmente condividere l'URL con altre persone;
- Oppure metterlo tra i preferiti per tornarci in seguito;
- Ma non possiamo passare troppi dati (massimo 2000 caratteri).

Scegliamo **post** negli altri casi:

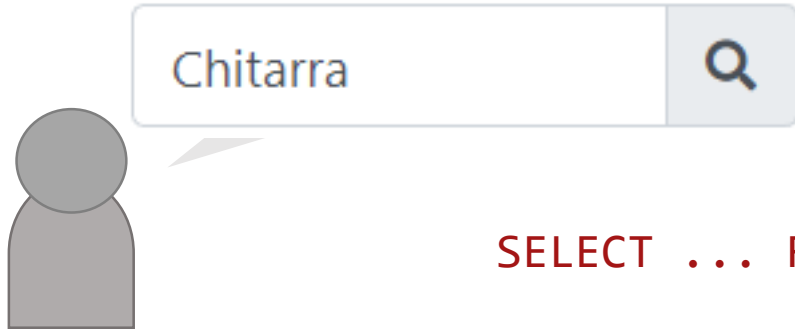
- Per inserire o modificare una certa mole di dati (es. la descrizione del corso);
- Non abbiamo il requisito della condivisione o dell'aggiunta ai preferiti.

# Il piano per implementare la ricerca

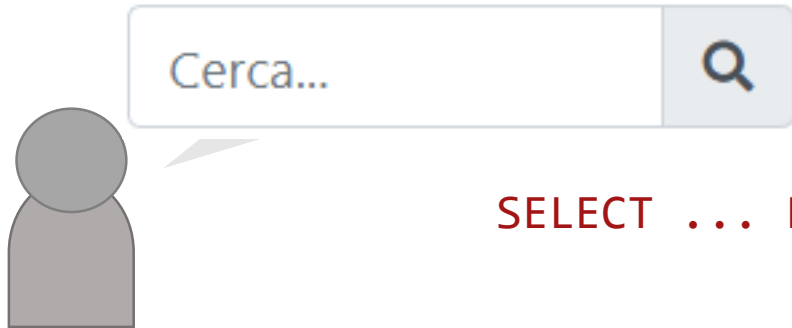




# Query di ricerca



```
SELECT ... FROM Courses WHERE Title LIKE '%Chitarra%'
```



```
SELECT ... FROM Courses WHERE Title LIKE '%'
```

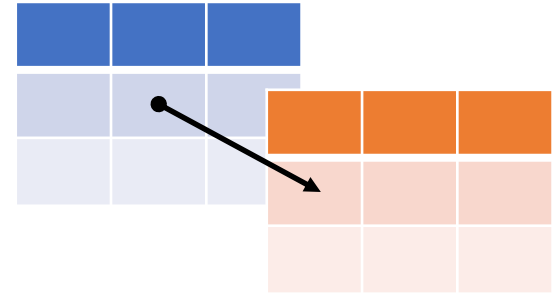
# Query optimizer

Database server

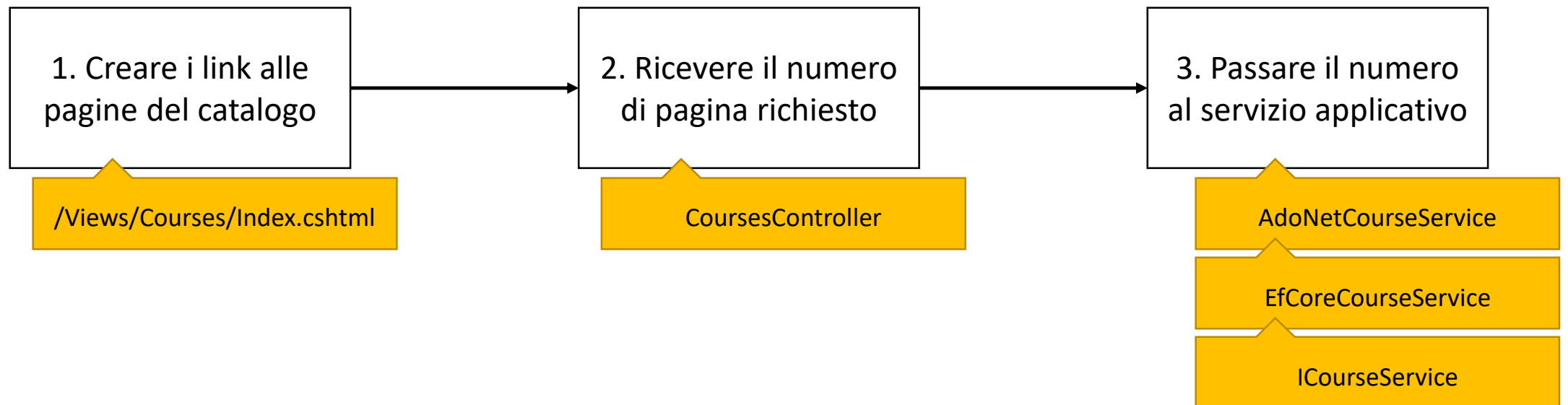
```
SELECT ... FROM Courses WHERE Title LIKE '%%'
```

Query  
optimizer

Query plan



# Il piano per implementare la paginazione



# Il tag helper <a> (*anchor*)

- Ci aiuta a creare link ad altre action o razor pages fornendo i nomi.

Ad esempio, questo tag helper <a>...

```
<a asp-action="Detail" asp-controller="Courses" asp-route-id="5">Corso 5</a>
```

...produce questo codice HTML.

```
<a href="/Courses/Detail/5">Corso 5</a>
```

# Il tag helper <a> (*anchor*)

- Ci aiuta a creare link ad altre action o razor pages fornendo i nomi.

Ad esempio, questo tag helper <a>...

```
<a asp-route-page="2">Pagina 2</a>
```

...produce questo codice HTML.

```
<a href="?page=2">Pagina 2</a>
```

# Paginazione con SQL: LIMIT e OFFSET

Esempio: ottenere la terza pagina di risultati, dove ogni pagina ne contiene 10

Tecnologia database	Esempio di query SQL di paginazione
SQLite MySQL PostgreSQL	SELECT * FROM Courses LIMIT 10 OFFSET 20
SQL Server Oracle	SELECT * FROM Courses ORDER BY Title OFFSET 20 ROWS FETCH NEXT 10 ROWS ONLY

*È importante indicare una clausola ORDER BY, altrimenti non è dato sapere in che ordine ci verranno fornite le righe.*

# Paginazione con LINQ: Skip e Take

Skip e Take sono due extension method:

- **Skip** serve a "saltare" i primi  $n$  risultati (equivale a OFFSET);
- **Take** serve a "prendere" i successivi  $m$  risultati (equivale a LIMIT).

```
IQueryable<CourseViewModel> queryLinq = dbContext.Courses  
    .Skip(20)  
    .Take(10)  
    .AsNoTracking()  
    .Select(course => CourseViewModel.FromEntity(course));
```

# Ordinamento con SQL: ORDER BY

Tecnologia database	Esempio di query SQL di ordinamento
SQLite MySQL PostgreSQL SQL Server Oracle ...	<pre>SELECT * FROM Courses ORDER BY Title ASC</pre> <p><i>ASC è il default e perciò può essere omissso</i></p>

Tecnologia database	Esempio di query SQL di ordinamento su più campi
SQLite MySQL PostgreSQL SQL Server Oracle ...	<pre>SELECT * FROM Courses ORDER BY Rating DESC, Title</pre>



# Sequenza delle clausole SQL

```
"SELECT ... WHERE ... ORDER BY ... LIMIT ...";
```

Deve essere  
sanitizzato

È un bool

`$"SELECT ... WHERE ... ORDER BY {orderby} {ascending} LIMIT ...";`

CurrentPrice... cioè?

Non possono diventare  
dei SqlParameter

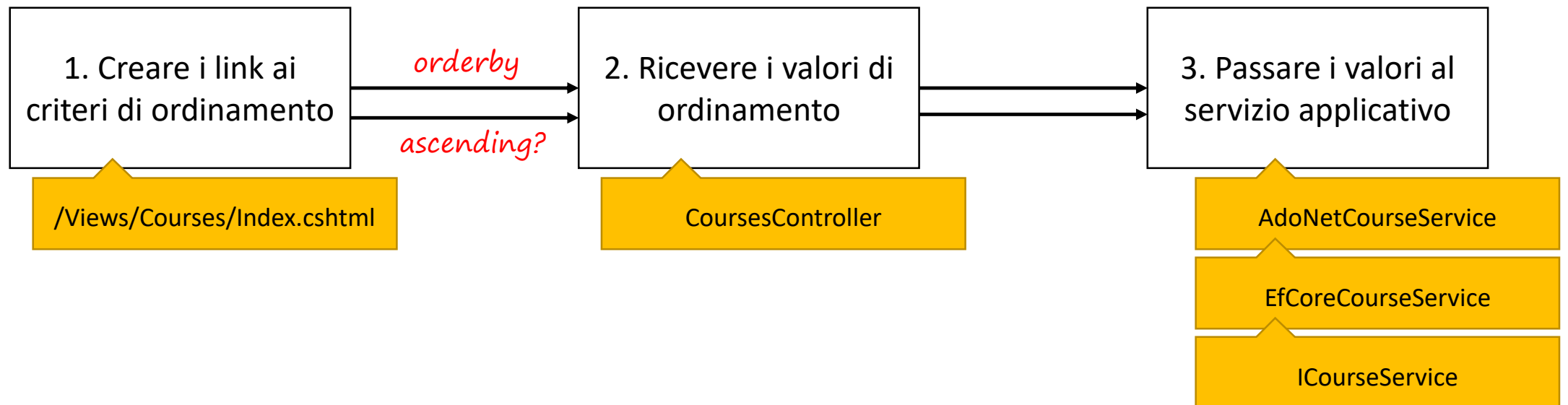
# Ordinamento con LINQ

Usiamo degli extension method:

- **OrderBy** ordina in senso ascendente;
- **OrderByDescending** ordina in senso discendente.
- Se abbiamo più criteri aggiungiamo anche **ThenBy** e **ThenByDescending**.


```
IQueryable<CourseViewModel> queryLinq = dbContext.Courses
    .OrderByDescending(course => course.Rating)
    .ThenBy(course => course.Title)
    .AsNoTracking()
    .Select(course => CourseViewModel.FromEntity(course));
```

# Il piano per implementare l'ordinamento



# Spostare la sanitizzazione in una classe

```
public class CourseListInputModel
{
    public string Search { get; set; }
    public int Page { get; set; }
    public string OrderBy { get; set; }
    public bool Ascending { get; set; }
}
```



```
public IActionResult Index(CourseListInputModel model)
{
    // ...
}
```

# Model binding di oggetti complessi

▼ General

**Request URL:** /Courses?search=Chitarra&page=2  
&orderby=Rating&ascending=true

**Request Method:** GET

**Status Code:** 200 OK

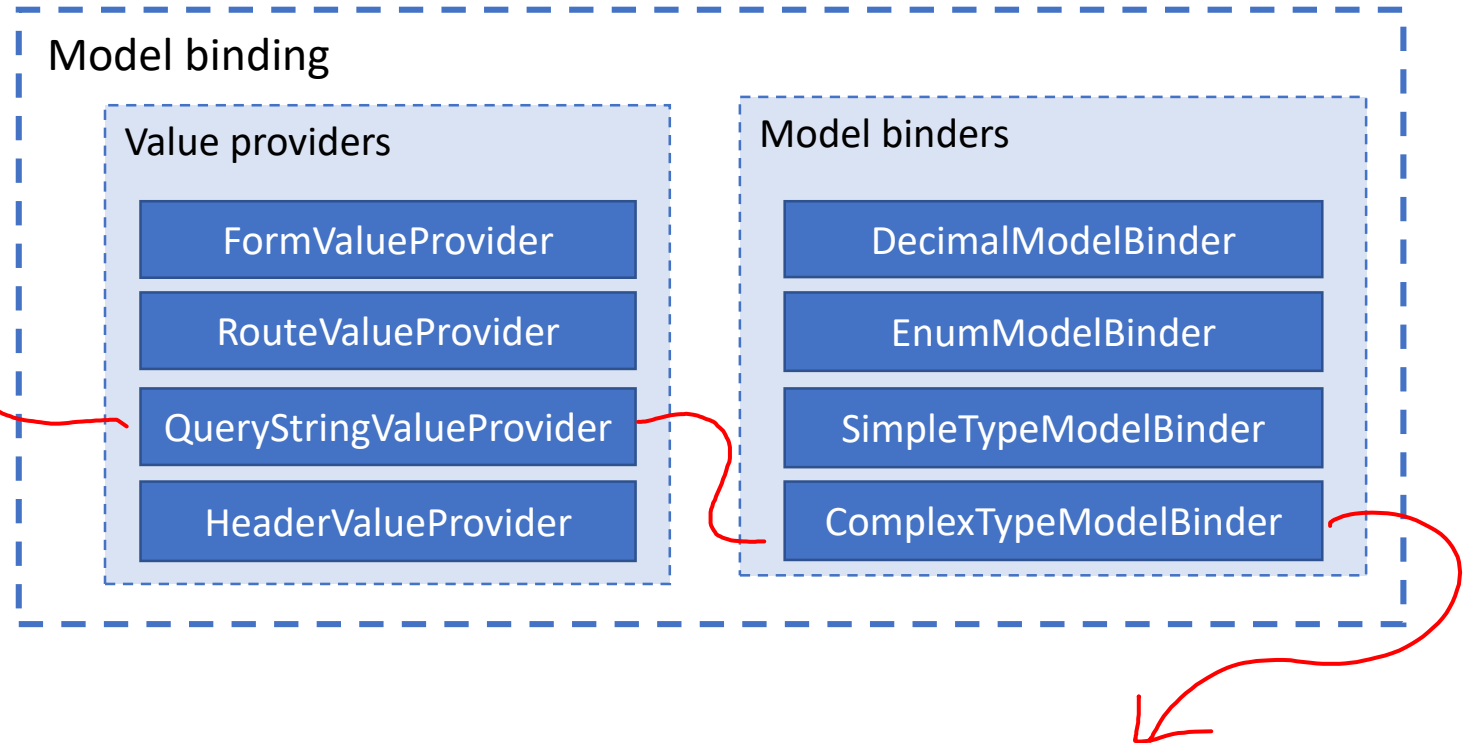
▼ Query String Parameters

**search:** Chitarra

**page:** 2

**orderby:** Rating

**ascending:** true

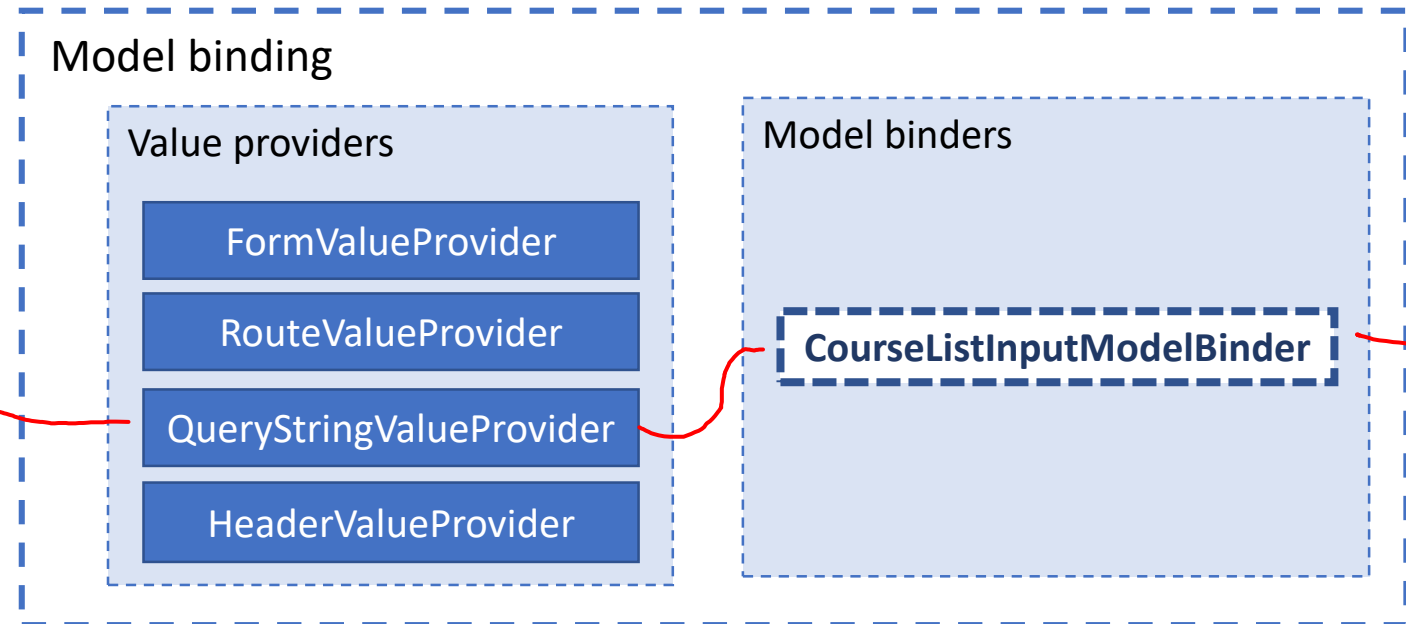


```
public IActionResult Index(CourseListInputModel model)
{
    // ...
}
```

# Un model binder personalizzato

**▼ General**  
**Request URL:** /Courses?search=Chitarra&page=2  
&orderby=Rating&ascending=true  
**Request Method:** GET  
**Status Code:** ● 200 OK

**▼ Query String Parameters**  
**search:** Chitarra  
**page:** 2  
**orderby:** Rating  
**ascending:** true



```
public IActionResult Index(CourseListInputModel model)
{
    // ...
}
```

# Per usare un model binder personalizzato...

- Decorare la classe model con l'attributo `ModelBinder`;

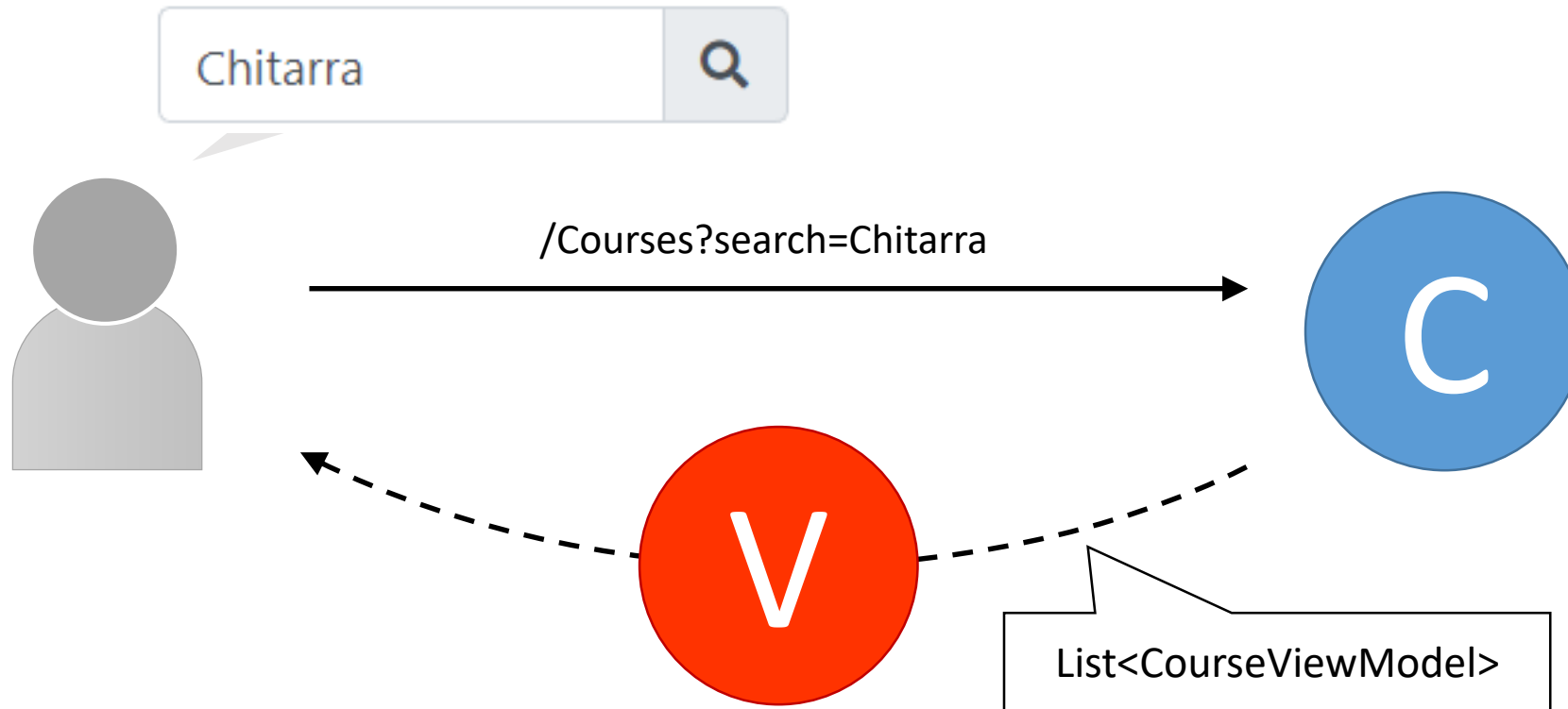
```
[ModelBinder(BinderType = typeof(CourseListInputModelBinder))]  
public class CourseListInputModel {  
    // ...  
}
```

- Il model binder deve implementare l'interfaccia `IModelBinder`.

```
public class CourseListInputModelBinder : IModelBinder {  
    public Task BindModelAsync(ModelBindingContext bindingContext) {  
        //Qui costruisco un'istanza di CourseListInputModel  
    }  
}
```



# Mantenere lo stato



*Non basta fornire alla view l'elenco di corsi!*

Mantenere lo stato

ViewData o ViewBag

...naaah

# Mantenere lo stato

```
@model CourseListViewModel
<div class="row">
  <div class="col-md-8">
    <h1>@ViewData["Title"]</h1>
  </div>
</div>
```

```
public class CourseListViewModel
{
    public List<CourseViewModel> Courses { get; set; }
    public CourseListInputModel Input { get; set; }
}
```

*Creiamo un ViewModel contenente tutto quel che serve alla view*

# Esempi di paginazione (Amazon)

Ispirato dalle tue visualizzazioni



Set Racchette Ping Pong  
Professionale STEXFIT, 2  
Racchette Con Borsa, 2 Racchette  
Ping Pong In Gomma Premium ...

★★★★★ ∨ 82

16,97€

Più venduto



Senston ITTF Set da pipistrello  
per racchetta da ping pong,  
paddle da ping pong con 2  
pipistrelli (impugnatura per le...

★★★★★ ∨ 12

26,49€



JOOLA 54200, Racchetta  
Sportivo Unisex – Adulto,  
Multicolore, Taglia Unica

★★★★★ ∨ 61

28,02€ ~~49,90€~~

Più venduto



Donic-Schildkröt T-One Training  
40Mm Confezione di 12 Palline  
per Tennis Tavolo, Multicolore

★★★★★ ∨ 37

5,99€

← Precedente

1

2

3

4

...

7

Avanti →

# Esempi di paginazione (Udemy)



## The Ultimate Web Developer How To Guide

283 lezioni • 22,5 ore • Tutti i livelli

Complete **web** developer Guide to websites working with HTML, CSS, JavaScript, PHP, Bootstrap, JQuery, MySQL and more | By Laurence Svekis

9,99 €

~~199,99 €~~

★★★★★ 4,5  
(1.055 valutazioni)



## Progressive Web Apps (PWA) - The Complete Guide

**BEST SELLER**

205 lezioni • 16 ore • Tutti i livelli

Build a Progressive **Web** App (PWA) that feels like an iOS & Android App, using Device Camera, Push Notifications and more | By Academind by Maximilian Schwarzmüller

9,99 €

~~149,99 €~~

★★★★★ 4,7  
(5.775 valutazioni)

# Esempi di paginazione (eBay)



One Piece Unlimited World Red Deluxe Edition Switch

**EUR 68,05**

*Compralo Subito*

Spedizione gratis

 Rapido e gratuito

- Consegna stimata entro mar. 27 ago.

 Affidabilità Top



4

5

6

7

8

9

10

11

12




13






Oggetti per pagina:

50 ▼

## Aggiunti di recente

	<b>L'arte dell'Ikebana</b> <i>di Mario Rossi</i>	★★★★☆	EUR 12.99 <del>EUR 15.99</del>	<a href="#">Dettagli</a>
	<b>Sommelier in 15 giorni</b> <i>di Mario Rossi</i>	★★★★☆	EUR 20.99 <del>EUR 24.99</del>	<a href="#">Dettagli</a>
	<b>Iniziare con Arduino e sensori</b> <i>di Mario Rossi</i>	★★★★☆	EUR 7.99	<a href="#">Dettagli</a>

## I migliori di sempre

	<b>Web marketing facile</b> <i>di Mario Rossi</i>	★★★★★	EUR 17.99 <del>EUR 19.99</del>	<a href="#">Dettagli</a>
	<b>L'ABC del fai da te</b> <i>di Mario Rossi</i>	★★★★★	EUR 7.99	<a href="#">Dettagli</a>
	<b>Come fare gli origami</b> <i>di Mario Rossi</i>	★★★★☆	EUR 30.99 <del>EUR 34.99</del>	<a href="#">Dettagli</a>

### Punto n° 1

*1. L'homepage deve presentare una selezione di contenuti:  
3 corsi aggiunti di recente e 3 corsi con valutazione più alta.*

# Progresso nella specifica

- Punto 1: elenchi di corsi in homepage;
- Punto 4: paginazione, ordinamento e ricerca.

## Requisiti funzionali

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23		

## Requisiti non funzionali

a	b	c	d
---	---	---	---