Parcijalni pogledi i View komponente

Model klasa

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
```

```
[Table("Osoba")]

public class Osoba
{
    public int OsobaId { get; set; }

    [Display(Name = "Ime osobe")]
    [Required(ErrorMessage = "Morate uneti ime")]
    [StringLength(30, ErrorMessage = "Maksimalno 30 karaktera")]
    public required string Ime { get; set; }

    [Required(ErrorMessage = "Morate uneti prezime")]
    [Display(Name = "Prezime osobe")]
    [StringLength(30, ErrorMessage = "Maksimalno 30 karaktera")]
    public required string Prezime { get; set; }

    [Required(ErrorMessage = "Morate uneti adresu")]
    [DataType(DataType.MultilineText)]
    [StringLength(100, ErrorMessage = "Maksimalno 100 karaktera")]
    public required string Adresa { get; set; }
}
```

DbContext klasa

```
using Microsoft.EntityFrameworkCore;
```

```
public class OsobaContext : DbContext
{
    public OsobaContext(DbContextOptions<OsobaContext> opcije) : base(opcije)
    {
    }
    public DbSet<Osoba> Osobe { get; set; }

}
```

```
public class OsobaContext(DbContextOptions<OsobaContext> opcije) :
DbContext(opcije)
{
     public DbSet<Osoba> Osobe { get; set; }

}
```

Definisanje konekcionog stringa u appsettings.json fajlu

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "AllowedHosts": "*",
  "ConnectionStrings": {
    "DefaultConnection": "Data Source=GORAN-HP;Initial Catalog=dbParcijalniPogledi;Integrated
Security=True;Encrypt=False"
  }
}
```

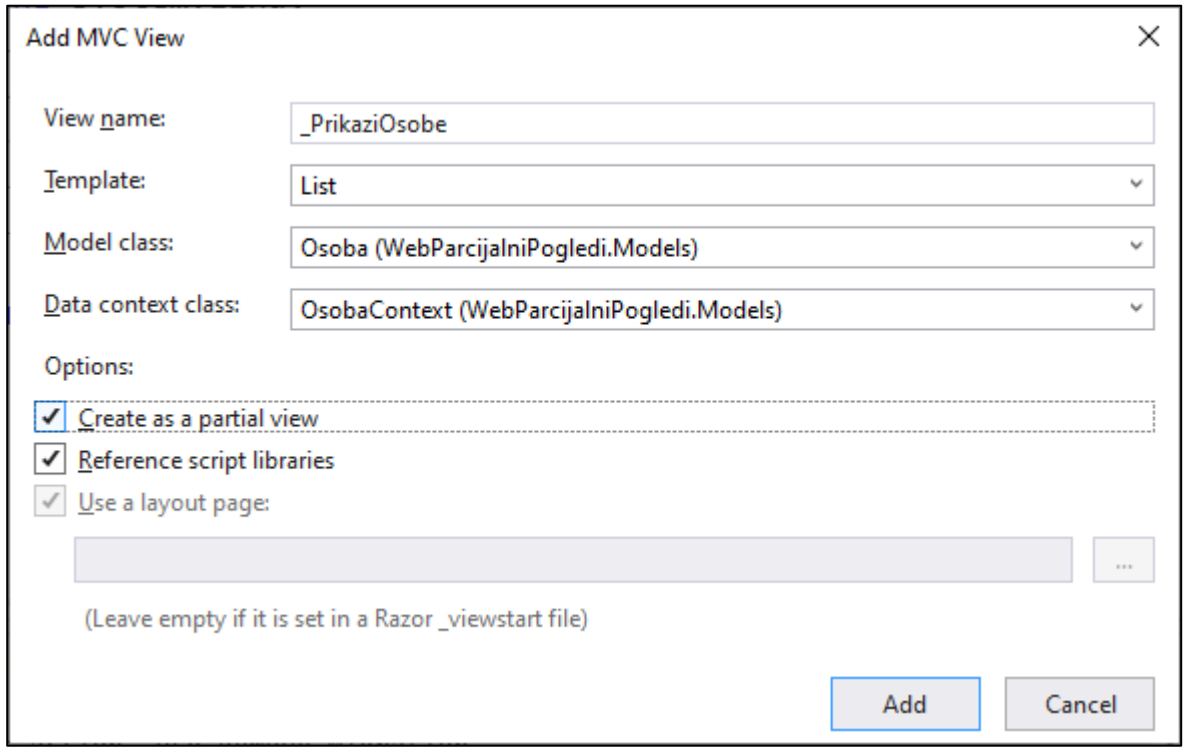
Registracija DbContext servisa

```
var builder = WebApplication.CreateBuilder(args);
var connectionString =
builder.Configuration.GetConnectionString("DefaultConnection");

builder.Services.AddDbContext<OsobaContext>(options =>
    options.UseSqlServer(connectionString));

// Add services to the container.
builder.Services.AddControllersWithViews();
```

Parcijalni pogledi

- Koriste se za renderovanje istog (ili sličnog) HTML-a na različitim mestima u web aplikaciji
- Kreiraju se kao i obični pogledi, uz čekiranje opcije **Create as a Partial View**
- U parcijalnom pogledu se ne koriste tagovi <head> i <body>
- Naziv parcijalnog pogleda počinje donjom crtom (npr. _PrikaziOsobe.cshtml)
- Tipizirani parcijalni pogled na početku fajla sadrži direktivu **@model**

Kreiranje parcijalnog pogleda u folderu Shared



_PrikaziOsobe.cshtml

```
@model IEnumerable<Osoba>


<table class="table">
    <thead>
        <tr>
                <th>
                    @Html.DisplayNameFor(model => model.Ime)
                </th>
                <th>
                    @Html.DisplayNameFor(model => model.Prezime)
                </th>
                <th>
                    @Html.DisplayNameFor(model => model.Adresa)
                </th>
            <th></th>
        </tr>
    </thead>
    <tbody>
@foreach (var item in Model) {
        <tr>
            <td>
                @Html.DisplayFor(modelItem => item.Ime)
            </td>
            <td>
                @Html.DisplayFor(modelItem => item.Prezime)
            </td>
            <td>
                @Html.DisplayFor(modelItem => item.Adresa)
            </td>

        </tr>
}
    </tbody>
</table>
```

Prosleđivanje podataka Index pogledu

```
public class HomeController : Controller
{

    private readonly OsobaContext _db;

    public HomeController(OsobaContext db)
    {
        _db = db;
    }
    public IActionResult Index()
    {
        return View(_db.Osobe.ToList());
    }
```

```
public class HomeController(OsobaContext _db) : Controller
{
     public IActionResult Index()
      {
          return View(_db.Osobe.ToList());
      }
```

Poziv parcijalnog pogleda iz pogleda

- Metoda **Html.PartialAsync()** služi za asinhroni poziv parcijalnog pogleda

```
@await Html.PartialAsync("_PrikaziOsobe")
```

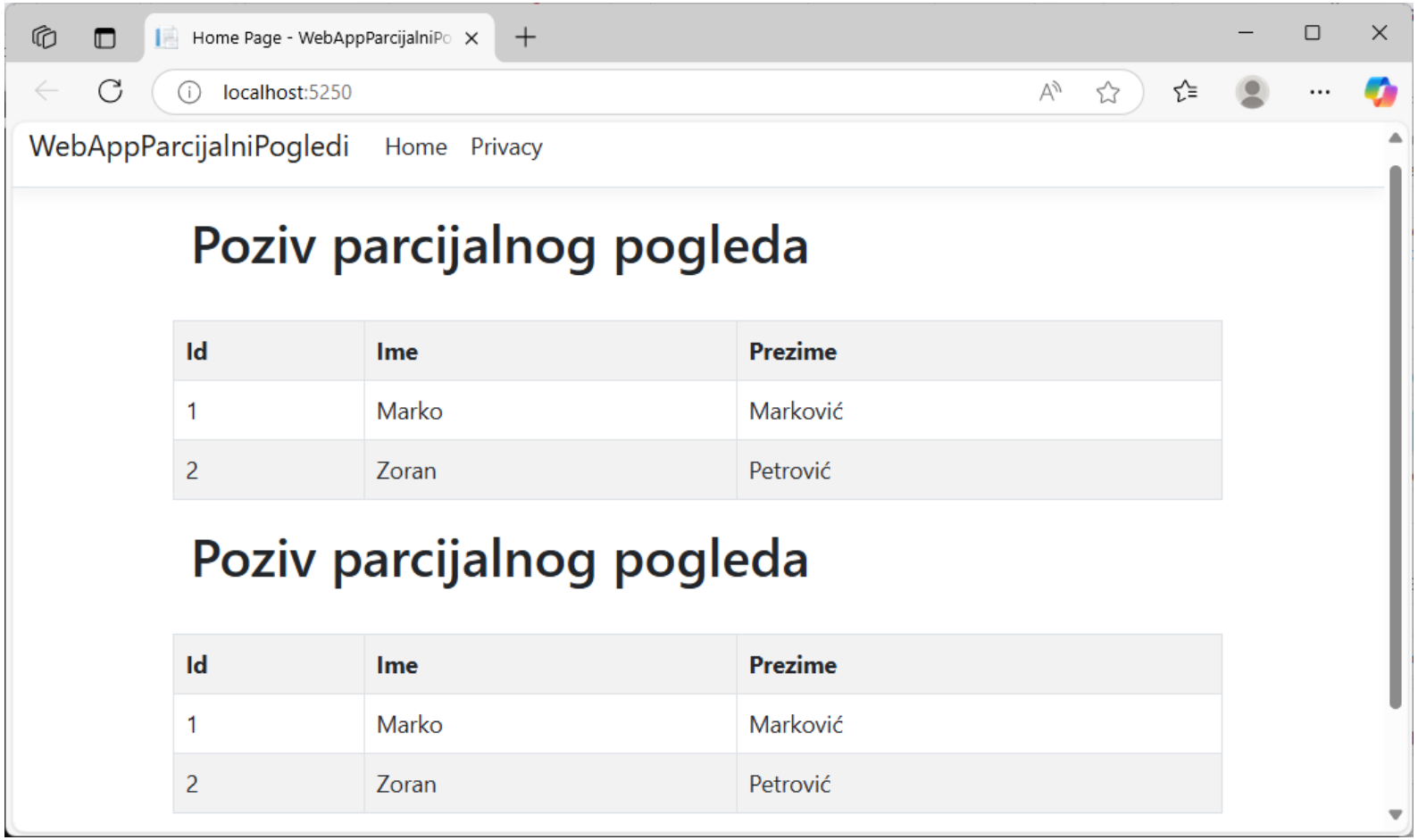
- **Partial Tag Helper** je element(tag) u ASP.NET Core-u koja omogućava uključivanje parcijalnog pogleda u .cshtml stranicu
- Upotreba Partial Tag Helper-a:

```
<partial name="_PrikaziOsobe" />
```

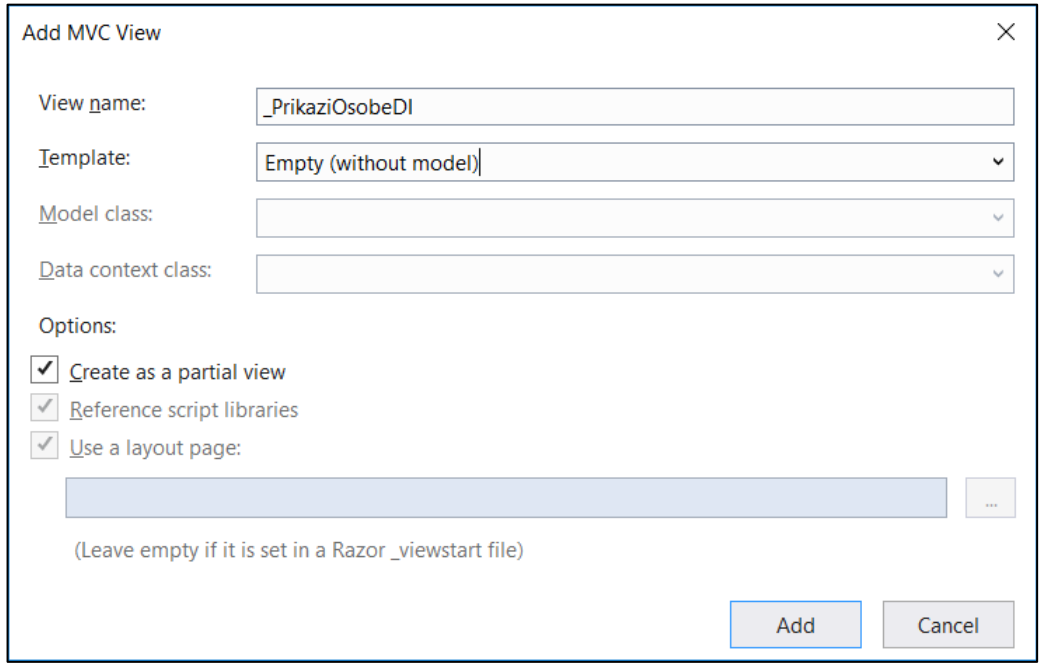
Index.cshtml

```
<div class="row">
    <h1>Poziv parcijalnog pogleda</h1>
    @await Html.PartialAsync("_PrikaziOsobe")
</div>
<div class="row">
    <h1>Poziv parcijalnog pogleda</h1>
    <partial name="_PrikaziOsobe" />
</div>
```

Parcijalni pogled prikazan u okviru Index pogleda



Kreiranje pogleda unutar koga ubacujemo zavisnost



Fajl _ViewImports.cshtml

```
@using WebAppParcijalniPogledi
@using WebAppParcijalniPogledi.Models
@addTagHelper *, Microsoft.AspNetCore.Mvc.TagHelpers
@addTagHelper *,WebAppParcijalniPogledi
```

- `@using WebAppParcijalniPogledi.Models`
  - Omogućava korišćenje klasa iz **Models** foldera uključujući i DbContext klasu u svim pogledim
- `@addTagHelper *,WebAppParcijalniPogledi`
  - Ova direktiva uključuje sve **Tag Helper-**e iz aplikacije
  - Omogućava korišćenje custom tagova bez dodatne konfiguracije u svakom pogledu
  - Važi globalno za sve .cshtml stranice
  - Bez ove linije, Tag Helper-i iz projekta ne bi bili prepoznati u Razor-u
  - Radi slično kao using, ali za Tag Helper komponente

Ubacivanje DbContext objekta u parcijalni pogled

```
@inject OsobaContext db
<table class="table table-bordered table-striped">
    <tr>
        <th>Id</th>
        <th>Ime</th>
        <th>Prezime</th>
    </tr>
    @foreach (var os in db.Osobe.ToList())
    {
        <tr>
            <td>@os.OsobaId</td>
            <td>@os.Ime</td>
            <td>@os.Prezime</td>
        </tr>
    }
</table>
```

_PrikaziOsobeDI.cshtml

- @inject OsobaContext db ubacuje instancu DbContext-a u Razor pogled
- Omogućava pristup bazi podataka direktno iz pogleda preko promenljive db
- @inject koristi Dependency Injection (DI) mehanizam ASP.NET Core-a
- Korišćenje DbContext-a direktno u pogledu ili parcijalnom pogledu nije preporučena praksa

Index strana HomeController

```
@{
    ViewData["Title"] = "Home Page";
}

<div class="row">
    <h1>Poziv parcijalnog pogleda</h1>
     <partial name="_PrikaziOsobeDI"/>
</div>
```

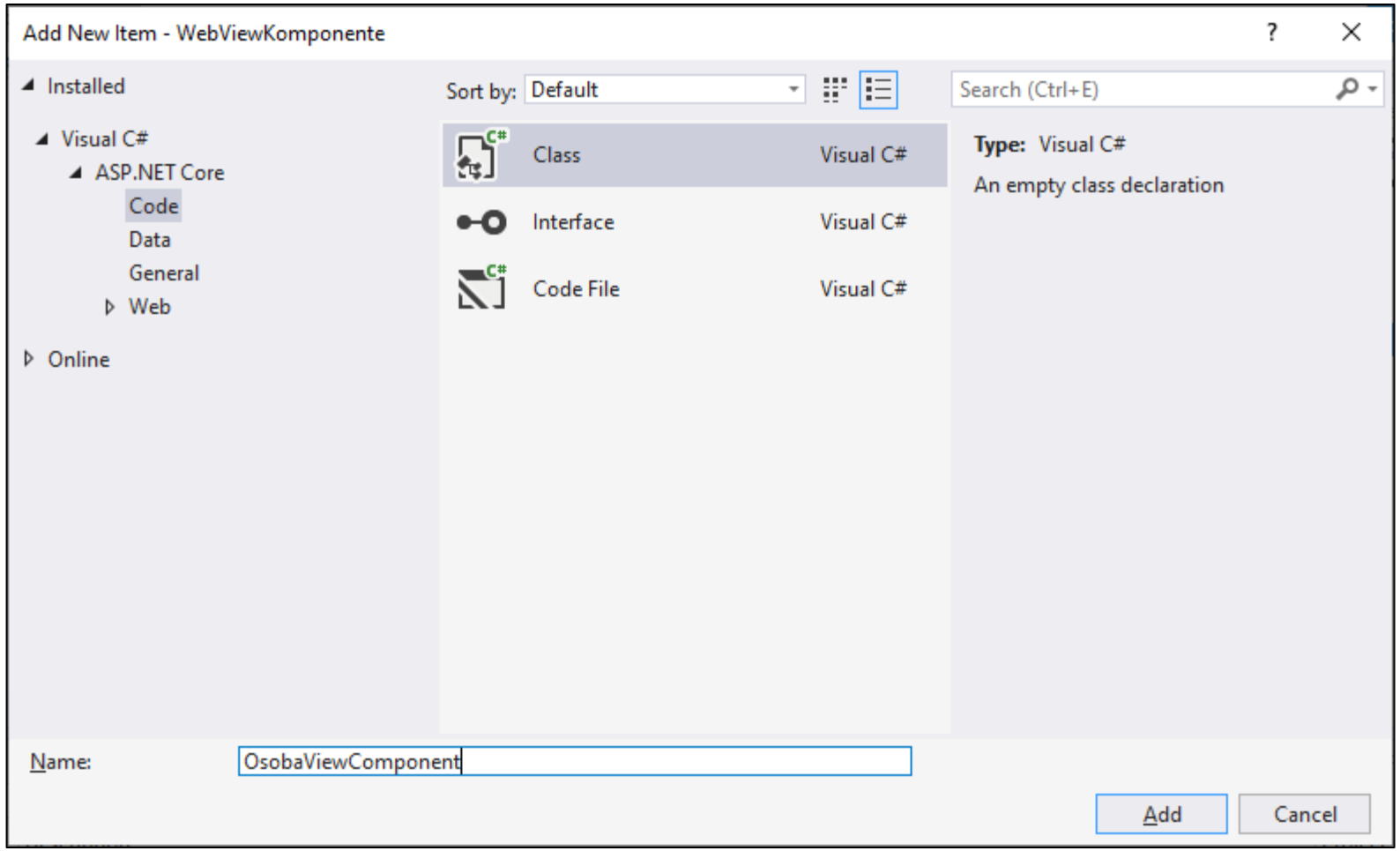
```
 public IActionResult Index()
    {
         // return View(db.Osobe.ToList());
         return View();
    }
```

View komponente

View komponente

- View komponente ne koriste model binding kao kontroleri i parcijalni pogledi
- Podaci se pribavljaju unutar same komponente ili se prosleđuju pri pozivu
- Često se pozivaju iz layout stranice
- Klasa komponente nasleđuje klasu **ViewComponent** i obično se završava sufiksom **ViewComponent**
- Komponenta vraća pogled (View)
- Logika se definiše u metodama **Invoke** ili **InvokeAsync** koje vraćaju **IViewComponentResult**
- Klasa može biti smeštena u bilo kom folderu aplikacije
- U praksi se često koristi folder **Components** za organizaciju

Klasa OsobaViewComponent

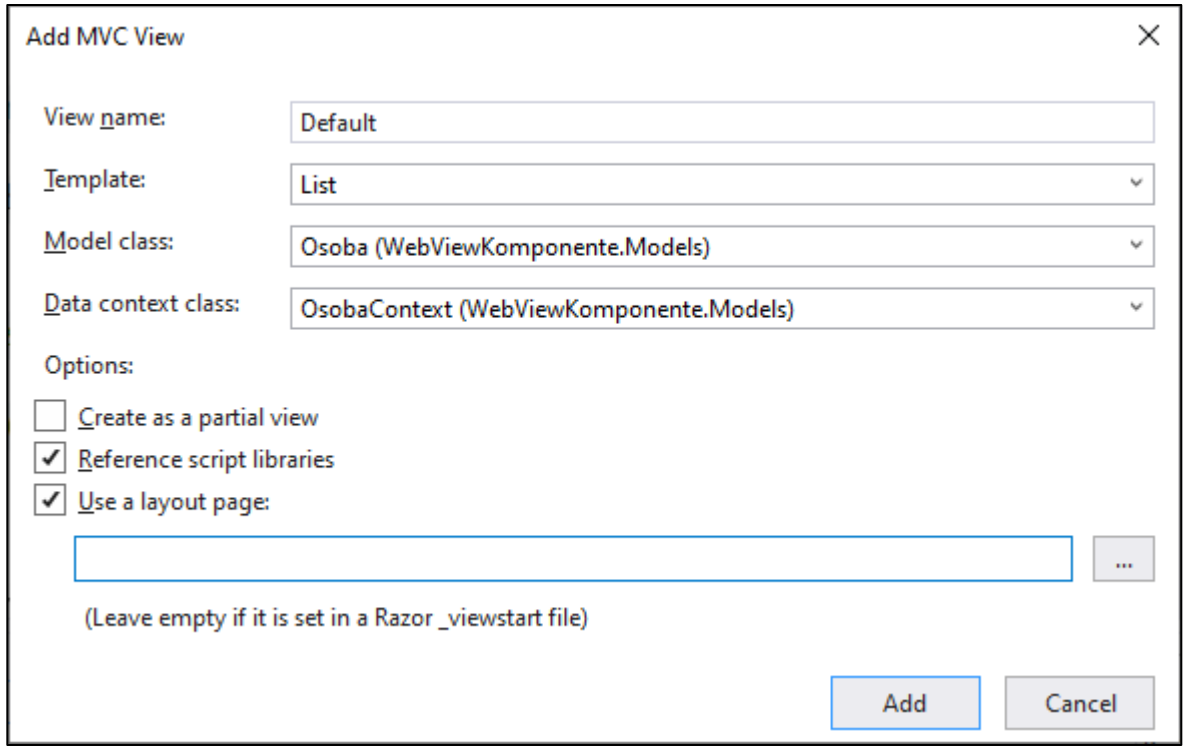


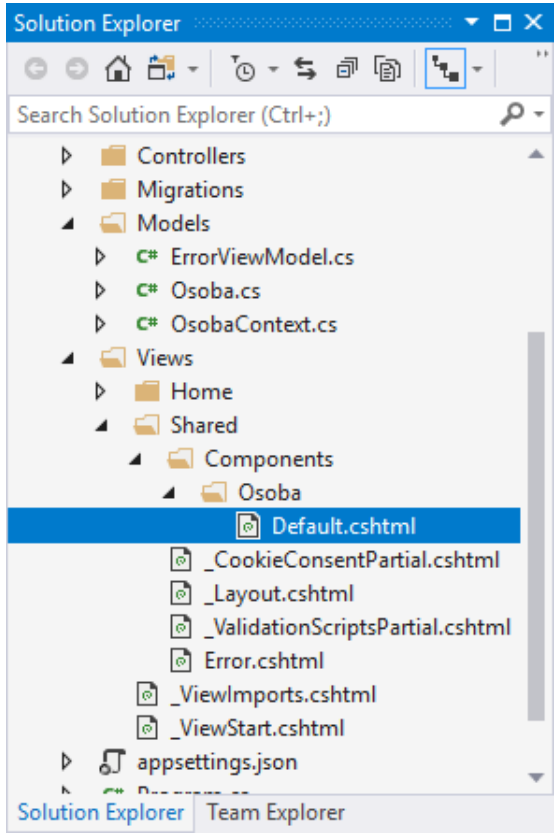
Klasa OsobaViewComponent

```
using Microsoft.AspNetCore.Mvc;
```

```
public class OsobaViewComponent: ViewComponent
{
    private readonly OsobaContext _db;

    public OsobaViewComponent(OsobaContext db)
    {
        _db = db;
    }

        public IViewComponentResult Invoke()
        {
            var osobe = _db.Osobe.ToList();
            return View(osobe);
        }
        //public async Task<IViewComponentResult> InvokeAsync()
        //{
        //    return View(await _db.Osobe.ToListAsync());
        //}
}
```

Kreiranje pogleda za komponentu





- Pogled koji odgovara View komponenti smešta se unutar foldera **Views/Shared/Components/Ime_Componente**
- Podrazumevano ime pogleda za komponentu **je Default.cshtml**

Default.cshtml

```
@model List<Osoba>

<table class="table table-bordered table-striped">
    <tr>
        <th>Id</th>
        <th>Ime</th>
        <th>Prezime</th>
        <th>Adresa</th>
    </tr>

    @foreach (var os in Model)
    {
        <tr>
            <td>@os.OsobaId</td>
            <td>@os.Ime</td>
            <td>@os.Prezime</td>
            <td>@os.Adresa</td>
        </tr>
    }
</table>
```

Poziv View komponente

- View komponenta se najčešće poziva iz pogleda (view-a)
- Može se pozvati i iz kontrolera, ali je to ređe u praksi
- Poziv se vrši metodom **Component.InvokeAsync**

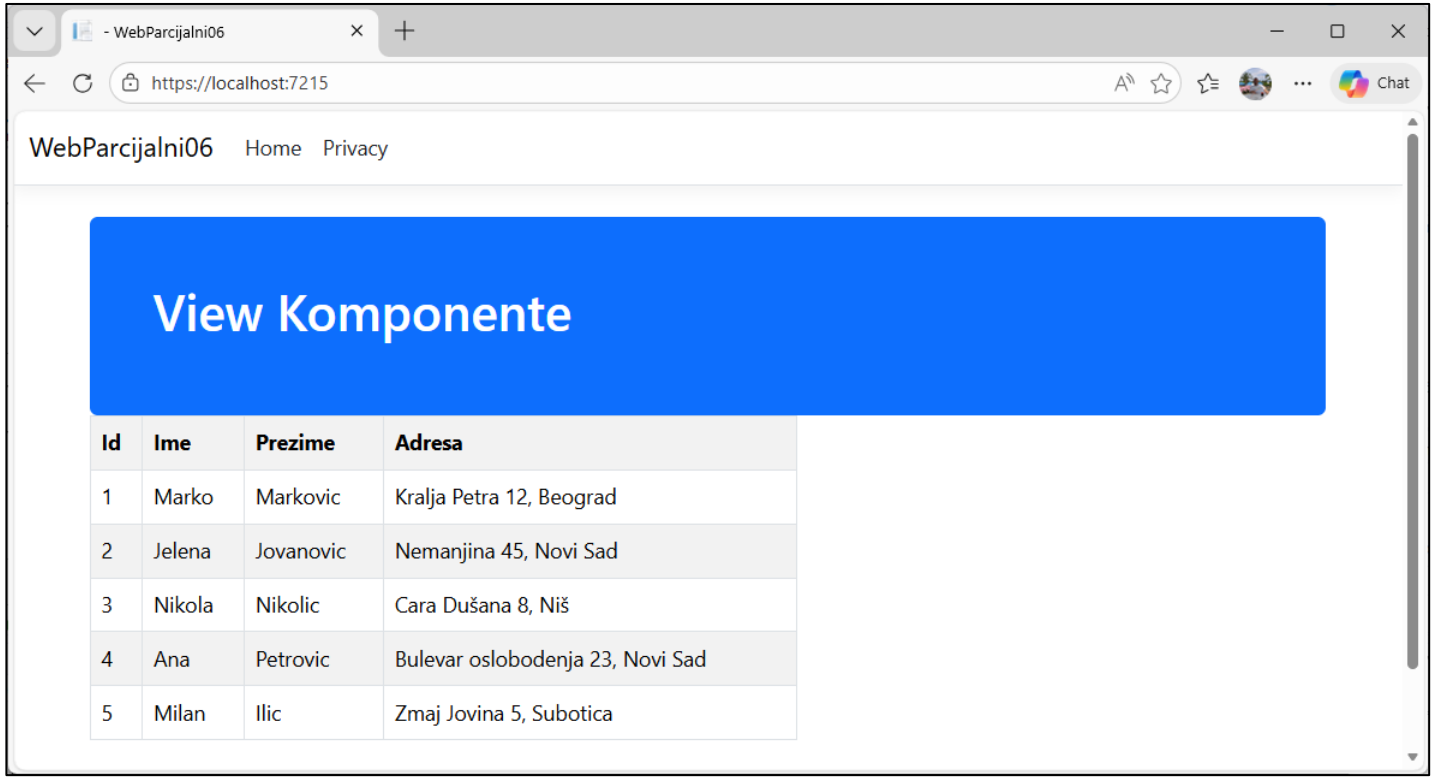
```
@await Component.InvokeAsync("Osoba")
```

```
@await Component.InvokeAsync("Ime_View_Komponente", {parametar1="vrednost1",
parametar2="vrednost"})
```

Index pogled klasa HomeController

```
<div class="mt-4 p-5 bg-primary text-white rounded">
    <h1>View Komponente</h1>
</div>

<div class="row">
    <div class="col-md-7">
        @await Component.InvokeAsync("Osoba")
    </div>
</div>
```

Index strana



Id	Ime	Prezime	Adresa
1	Marko	Markovic	Kralja Petra 12, Beograd
2	Jelena	Jovanovic	Nemanjina 45, Novi Sad
3	Nikola	Nikolic	Cara Dušana 8, Niš
4	Ana	Petrovic	Bulevar oslobodenja 23, Novi Sad
5	Milan	Ilic	Zmaj Jovina 5, Subotica

_ViewImports.cshtml

```
@using WebParcijalni06
@using WebParcijalni06.Models
@addTagHelper *, Microsoft.AspNetCore.Mvc.TagHelpers
@using WebParcijalni06.Components
@addTagHelper *, WebParcijalni06
```

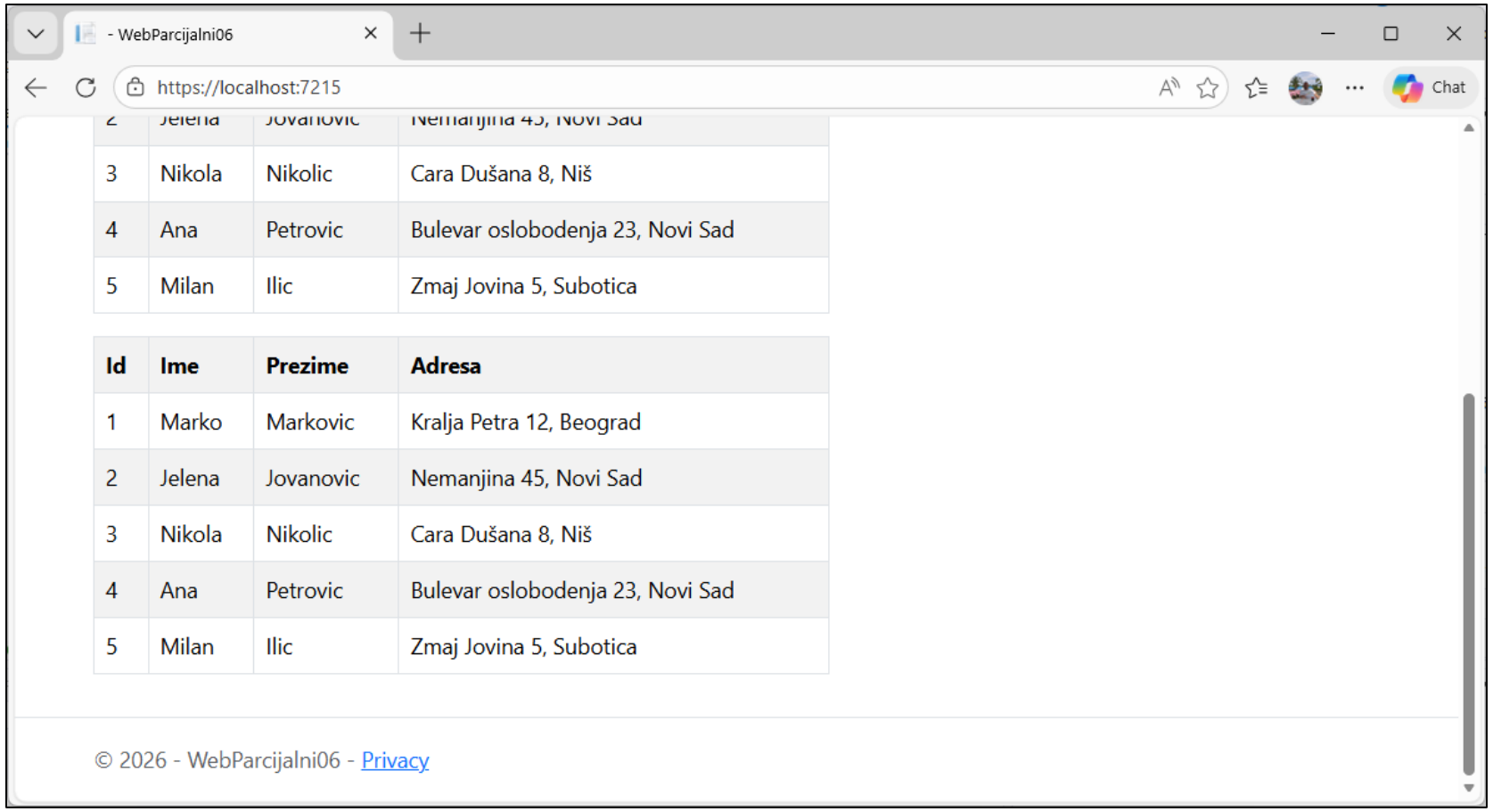
- @using WebParcijalni06.Components omogućava korišćenje View komponenti
- @addTagHelper *, WebParcijalni06 registruje sve Tag Helper-e iz aplikacije
- Omogućava korišćenje View komponente kao HTML taga
- Nije potrebno dodatno uključivanje u svakom pogledu

Poziv komponente

```
<div class="mt-4 p-5 bg-primary text-white rounded">
    <h1>View Komponente</h1>
</div>

<div class="row">
    <div class="col-md-7">
        @await Component.InvokeAsync("Osoba")
    </div>
</div>
<div class="row">
    <div class="col-md-7">
        <vc:osoba />
    </div>
</div>
```

- View komponenta se može pozvati na dva načina
- Prvi način: Component.InvokeAsync("Osoba") (C# sintaksa)
- Drugi način**: <vc:osoba />** (Tag Helper sintaksa)
- Oba načina daju isti rezultat
- Tag Helper sintaksa je čitljivija i češće se koristi u praksi

Index strana

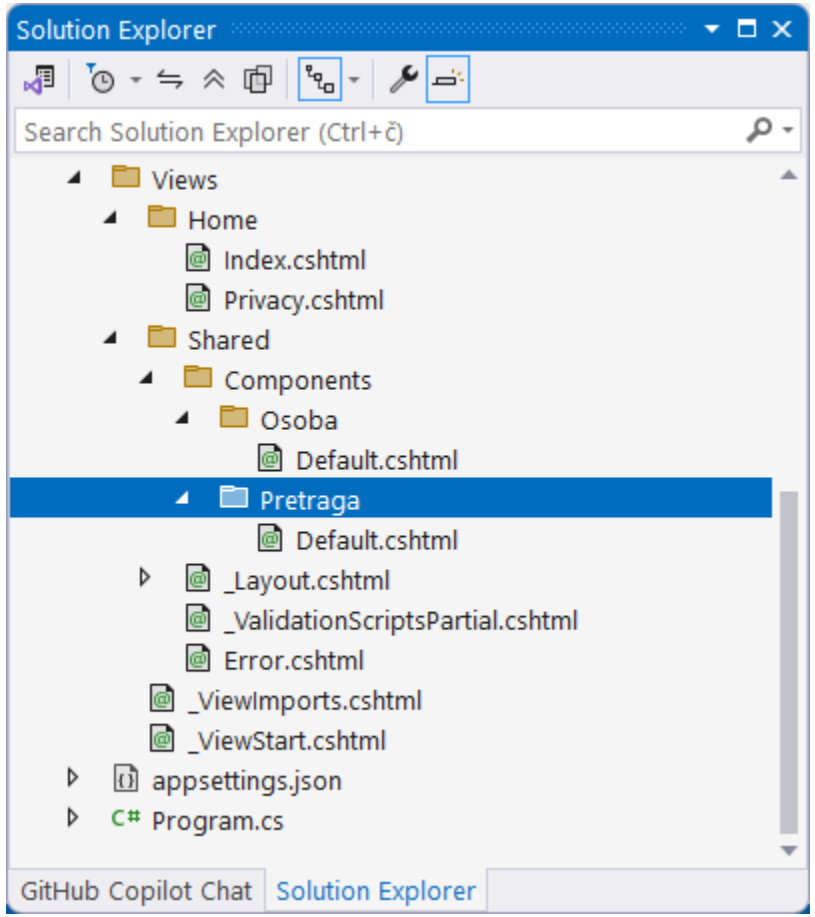


Komponenta sa parametrom

```
public class PretragaViewComponent : ViewComponent
{
    private readonly OsobaContext _db;

    public PretragaViewComponent(OsobaContext db)
    {
        _db = db;
    }

    public IViewComponentResult Invoke(string tekst)
    {
        var osobe = _db.Osobe
            .Where(o => o.Ime.Contains(tekst) || o.Prezime.Contains(tekst))
            .ToList();

        return View(osobe);
    }
}
```

Komponenta sa parametrom –šablon

```
@model List<Osoba>

<table class="table table-bordered">
    @foreach (var os in Model)
    {
        <tr>
            <td>@os.Ime</td>
            <td>@os.Prezime</td>
            <td>@os.Adresa</td>
        </tr>
    }
</table>
```

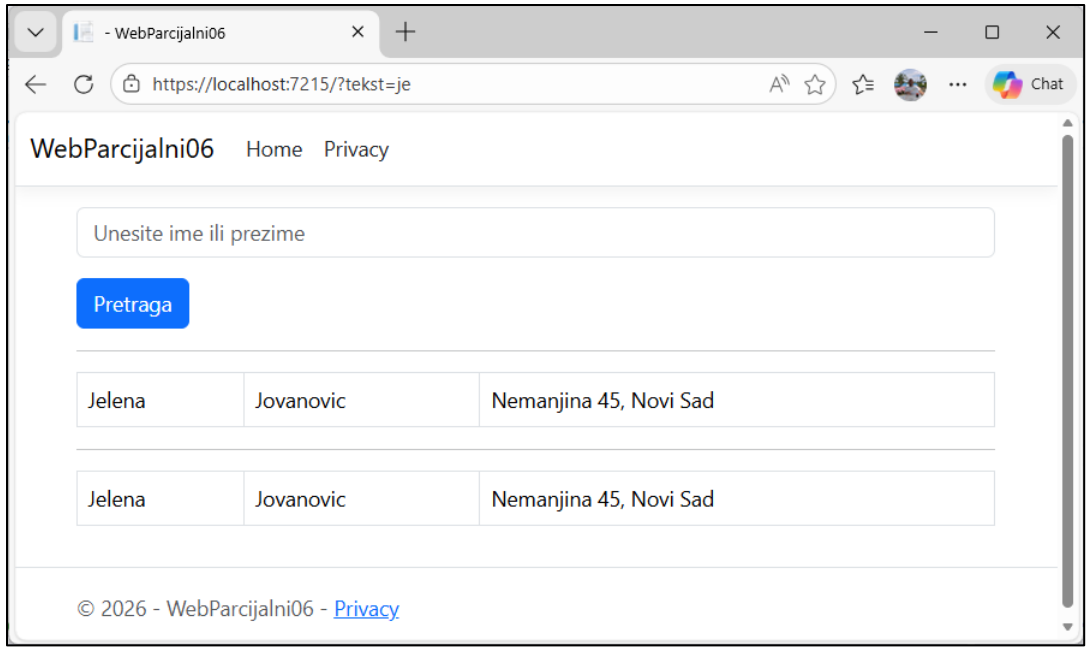


Poziv komponente sa parametrima

```
<form method="get">
    <div class="mb-3">
        <input type="text" name="tekst" class="form-control“
        placeholder="Unesite ime ili prezime" />
    </div>
    <button type="submit" class="btn btn-primary">Pretraga</button>
</form>

<hr />
<vc:pretraga tekst="@Context.Request.Query["tekst"]" />
<hr />
@await Component.InvokeAsync("Pretraga", new { tekst =
Context.Request.Query["tekst"].ToString() })
```

- Svojstvo **Context** u Razor pogledu služi za pristup **HttpContext** objektu
- HttpContext sadrži informacije o trenutnom HTTP zahtevu i odgovoru
- **Request** je svojstvo objekta HttpContext i predstavlja dolazni HTTP zahtev
- **Query** je kolekcija (IQueryCollection) koja sadrži parametre iz URL-a (query string)
- Context.Request.Query["tekst"] predstavlja pristup vrednosti parametra iz query string-a preko HttpContext-a.

Index strana



Pitanje 1

Parcijalni pogled ne može da sadrži sledeći html element:

a. <p>
b.
c. <head>

Odgovor: c

Pitanje 2

U ASP.NET Core MVC web aplikaciji kreiran je parcijalni pogled _PrikaziOsobe koji prikazuje kolekciju objekata klase Osoba u vidu tabele. Na koji način referenciramo ovaj pogled unutar Index pogleda

a. await Partial("_PrikaziOsobe ")
b. await ViewBag("_PrikaziOsobe ")
c. await Html.PartialAsync("_PrikaziOsobe")

Odgovor: c

Pitanje 3

Ako parcijalni pogled pozivamo iz pogleda koji pripadaju različitim kontrolerima smeštamo ga unutar foldera:

a. unutar foldera Views/Home
b. unutar foldera Views/Shared
c. unutar foldera Models

Odgovor: b

Pitanje 4

Ubacivanje objekta db servis klase OsobaContext u pogled vrši se pomoću direktive

a. @import OsobaContext db
b. @inject OsobaContext db
c. @model OsobaContext db

Odgovor: b

Pitanje 5

Šta su parcijalni pogledi u ASP.NET Core MVC aplikacijama?

a. To su delovi Razor koda koji se mogu iskoristiti na više stranica ili projekata u okviru aplikacije
b. To su kontroleri koji se koriste za obradu zahteva koje se odnose samo na deo stranice
c. To su vidljivi delovi MVC modela koji se koriste za validaciju podataka.

Odgovor: a

Pitanje 6

Koji od navedenih mehanizama se koristi za prikazivanje dinamičkog sadržaja u ASP.NET Core MVC aplikacijama kada se poziva metoda **InvokeAsync**?

a. Partial View
b. View Component
c. View

Odgovor: b

Pitanje 7

Poziv komponente koja je predstavljena klasom OsobaViewComponent unutar nekog cshtm fajla vrši se korišćenjem koda:

a. @await Component.InvokeAsync("Osoba")
b. @await Component.Invoke("OsobaViewComponent ")
c. @await Component("Osoba")

Odgovor: a

Pitanje 8

Koja metoda klase View komponente se koristi za definisanje logike View komponente u ASP.NET Core MVC aplikacijama?

a. InvokeComponent

b. View

c. InvokeAsync

Odgovor: c

Pitanje 9

Koja je uloga fajla Default.cshtml u ViewComponenti u ASP.NET Core MVC aplikacijama?

a. Definiše šablon koji se koristi za prikazivanje HTML-a u ViewComponenti
b. Definiše JavaScript funkcije koje se koriste u ViewComponenti
c. Definiše CSS stilove koji se primenjuju na ViewComponentu

Odgovor: a

Pitanje 10

Koji od sledećih primera ispravno poziva View komponentu **PretragaViewComponent** sa parametrom **tekst** postavljenim na vrednost Ma?

a. @Component.Invoke("Pretraga", tekst = "Ma")
b. @await Component.InvokeAsync("PretragaView", new { tekst = "Ma" })
c. <vc:pretraga tekst="Ma"/>

Odgovor: c