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 string Ime { get; set; }

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

    [Required(ErrorMessage = "Morate uneti adresu")]
    [DataType(DataType.MultilineText)]
    [StringLength(60, ErrorMessage = "Maksimalno 60 karaktera")]
    public 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; }

}
```

Definisanje konekcionog stringa u appsettings.json fajlu

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "AllowedHosts": "*",
  "ConnectionStrings": {
    "DefaultConnection": "Data Source=GORAN-HP;Initial Catalog=OsobaDb2903;Integrated
Security=True;Encrypt=False"
  }
}
```

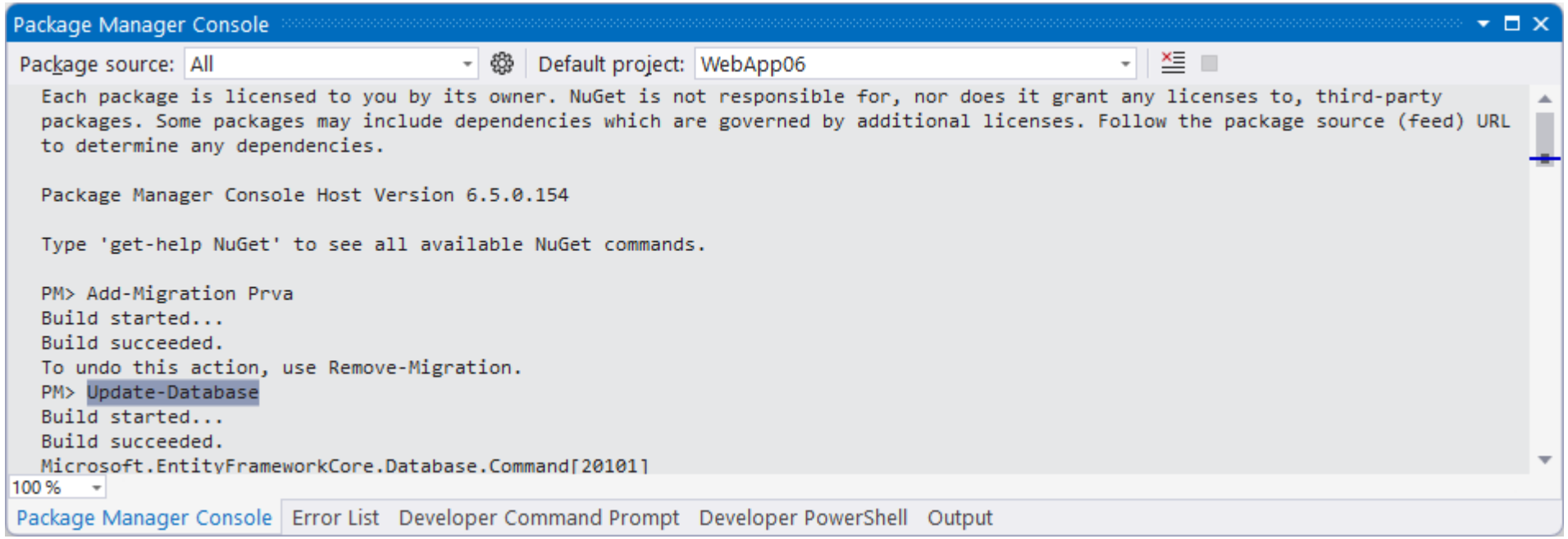
Registrujemo DbContext kao servis

```
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();
```

Kreiranje baze podataka primenom migracija

```
PM> Add-Migration Prva
```

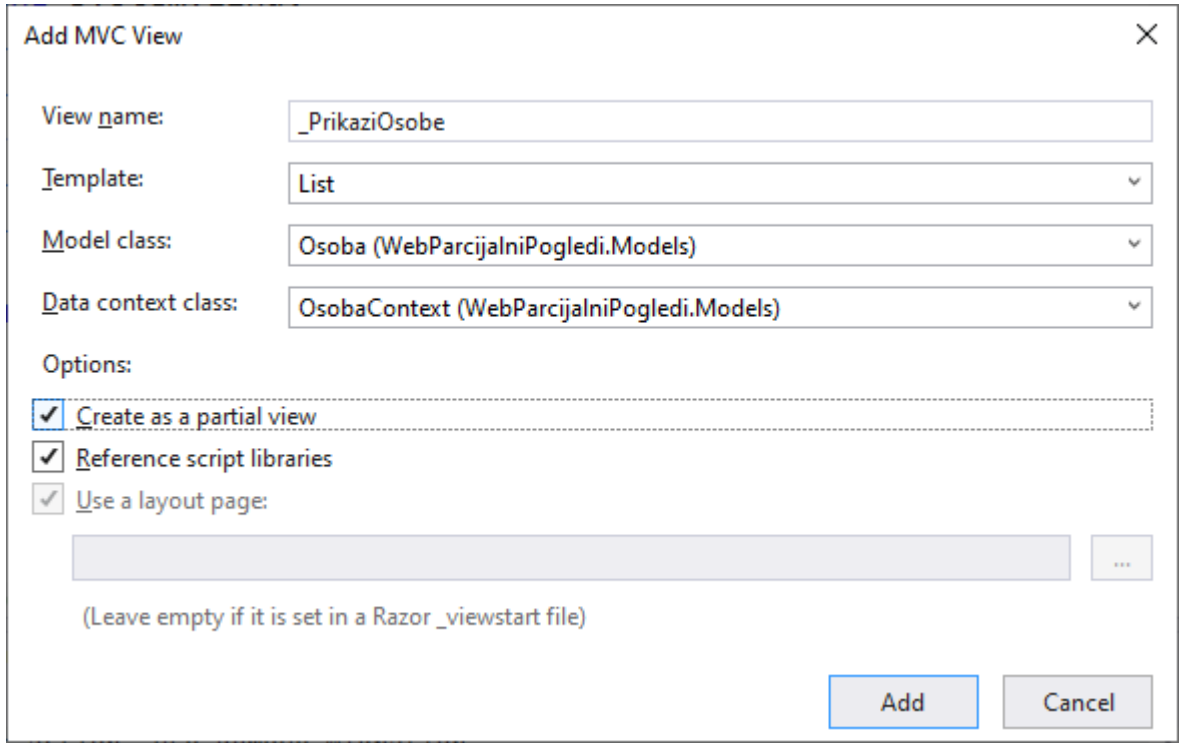
```
PM> Update-Database
```



Parcijalni pogledi

- Služi za renderovanje sličnog HTML na različitim lokacijama web aplikacije
- Kreira se kao i običan pogled ali se čekira opcija **Create as a Partial View**
- Unutar parcijalnog pogleda se ne ubacuju tagovi **<head>**, **<body>**
- Parcijalni pogled počinje donjom crtom **_PrikaziOsobe.cshtml**
- Tipizirani pogled ima **@model** direktivu na početku fajla

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());
    }
```

Poziv parcijalnog pogleda iz pogleda

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

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

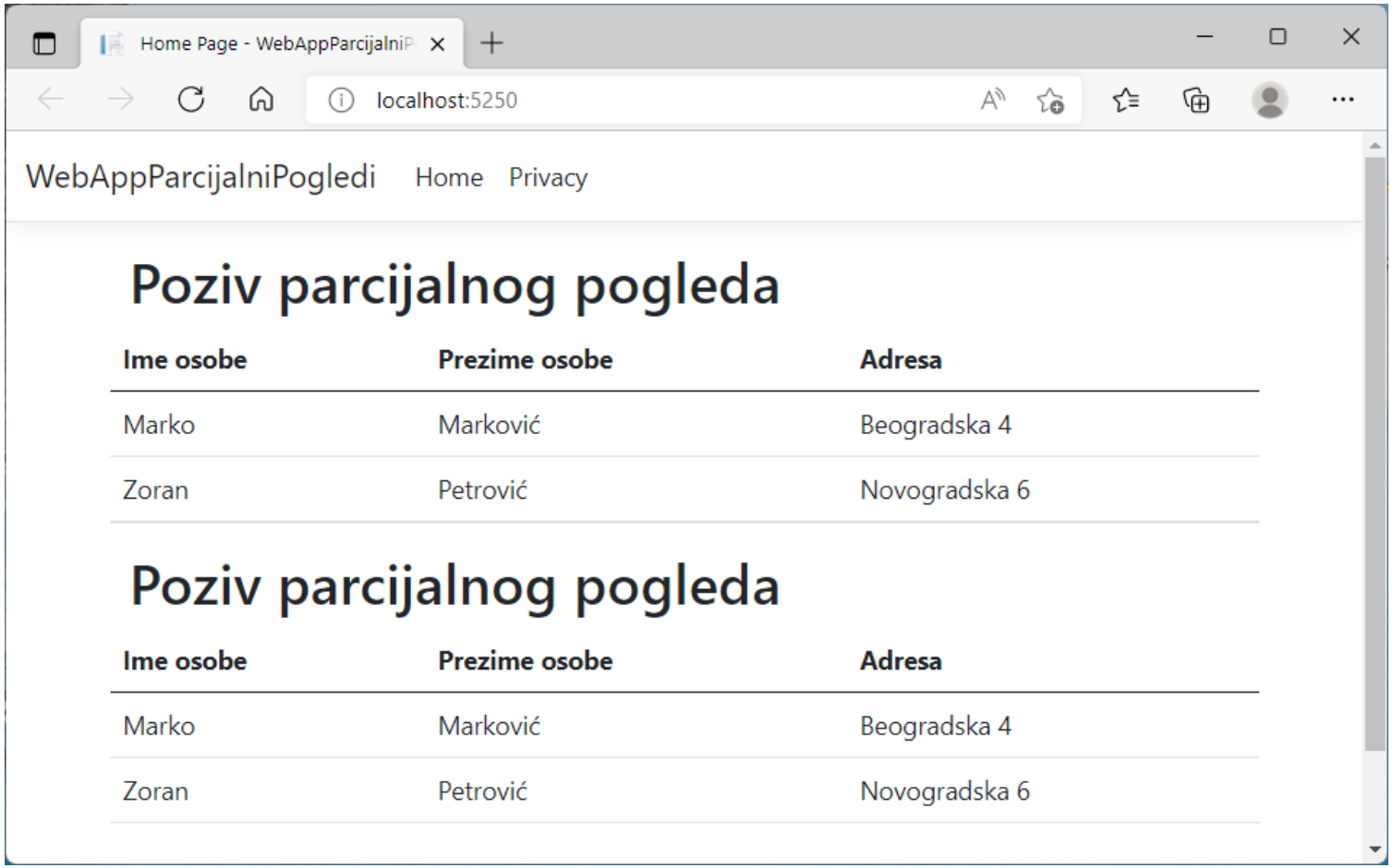
- PartialTagHelper je Tag Helper klasa u ASP.NET Core-u koja se koristi za uključivanje parcijalnog pogleda u HTML stranicu
- Upotreba partial tag helpera

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

Index.cshtml

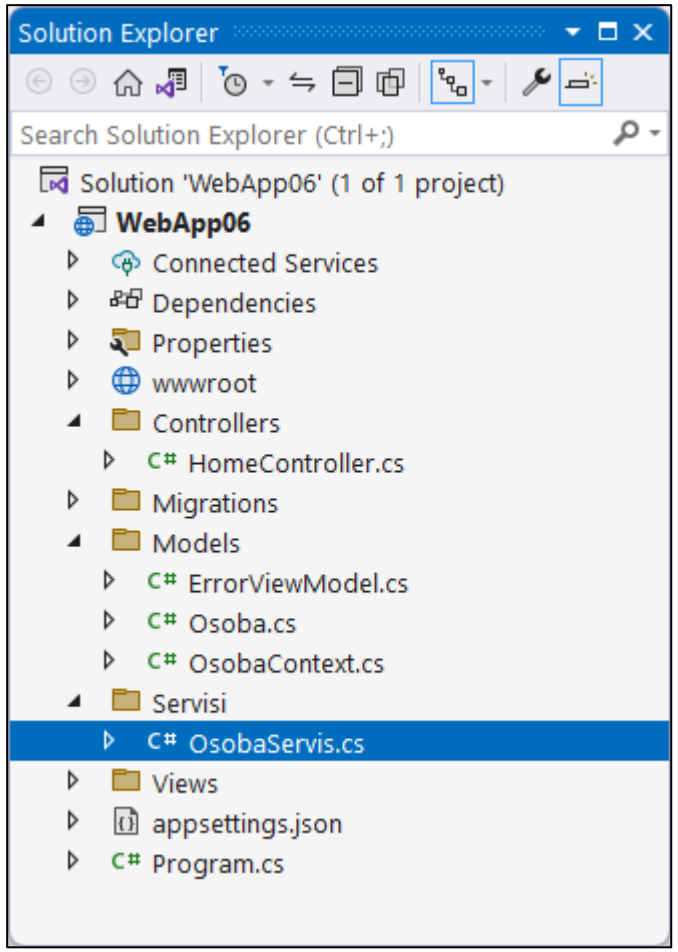
```
<div class="row">
    <h1>Poziv parcijalnog pogleda</h1>
    @await Html.PartialAsync("_PrikaziOsobe")
</div>
<div class="row">
    <h1>Poziv parcijalnog pogleda</h1>
    <partial name="_PrikaziOsobe" />
</div>
```

Parcijalni pogled prikazan u okviru Index pogleda



Klasa OsobaServis folder Servisi

```
public class OsobaServis
{
    private readonly OsobaContext _db;
    public OsobaServis(OsobaContext db)
    {
        _db = db;
    }

    public IEnumerable<Osoba> VratiOsobe()
    {
        return _db.Osobe;
    }

}
```



Registracija servisa OsobaServis u DI kontejneru

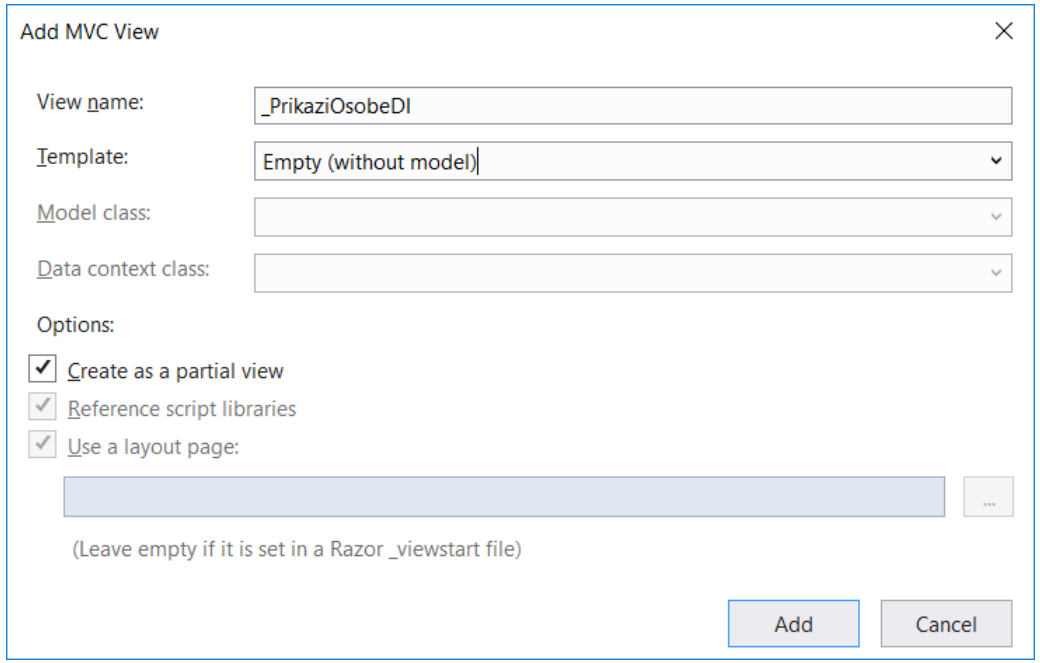
```
var builder = WebApplication.CreateBuilder(args);
var connectionString =
builder.Configuration.GetConnectionString("DefaultConnection");

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

// Add services to the container.
builder.Services.AddControllersWithViews();
builder.Services.AddTransient<OsobaServis>();
```

Kreira se nova instance klase OsobaServis kad za svaki zahtev u kome je servis OsobaServis tražen

Fajl _ViewImports.cshtml

```
@using WebApp06
@using WebApp06.Models
@addTagHelper *, Microsoft.AspNetCore.Mvc.TagHelpers
@using WebApp06.Servisi
```

Kreiranje pogleda unutar koga ubacujemo zavisnost



Ubacivanje zavisnosti u pogled

_PrikaziOsobeDI.cshtml

```
@inject OsobaServis oServis

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

Ubacivanje DbContext objekta u parcijalni pogled

_PrikaziOsobeDI.cshtml

```
@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>
```

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();
        }
```

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 oServis servis klase OsobaServis u pogled vrši se pomoću direktive

a. @import OsobaServis oServis
b. @inject OsobaServis oServis
c. @model OsobaServis oServis

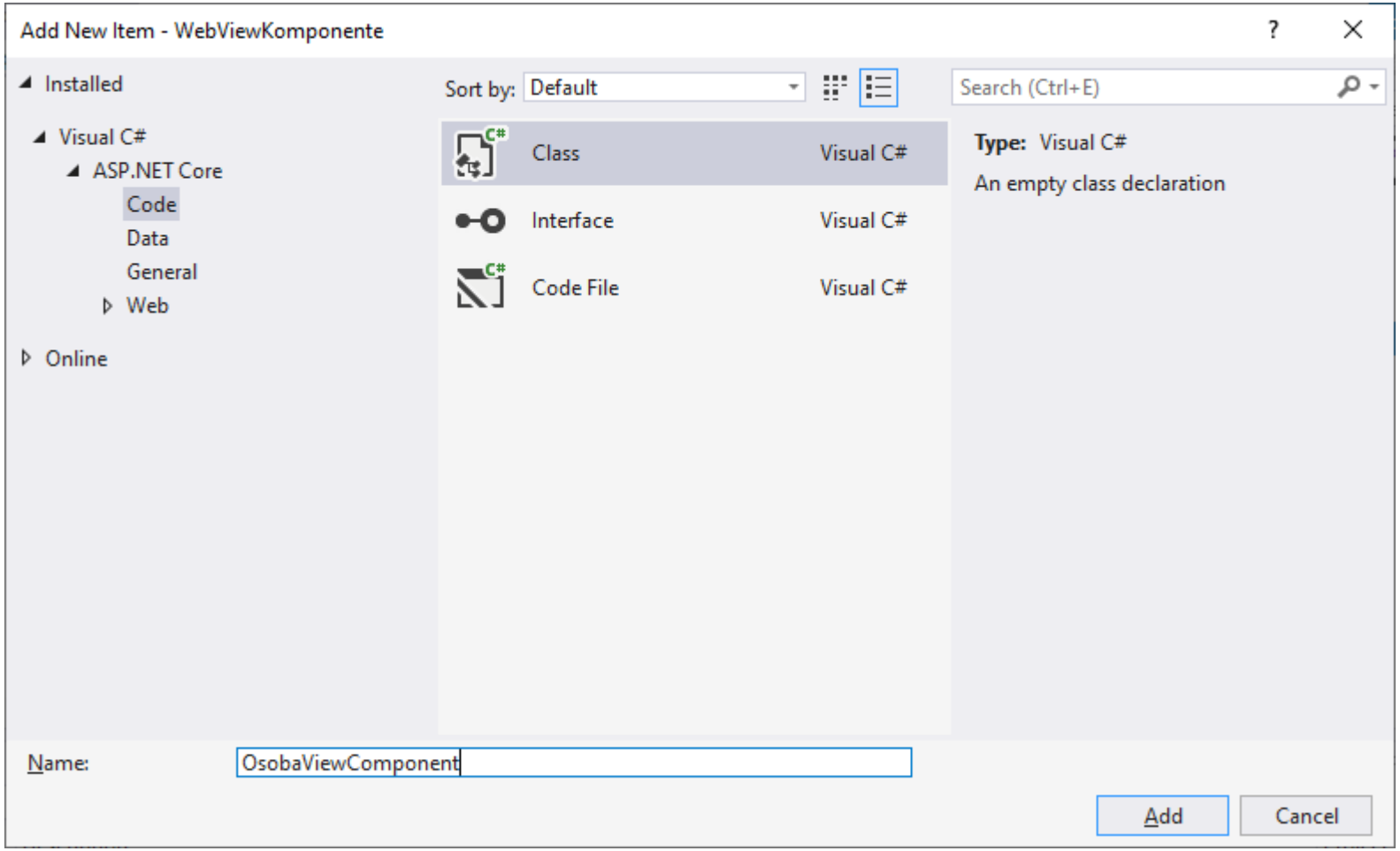
Odgovor: b

View komponente

View komponente

- View komponente ne koriste model binding kao parcijalni pogledi već zavise od podataka koji im se prosleđuju pri njihovom pozivu
- Često se komponenta poziva iz layout strane
- Klase komponente je obično izvedena iz klase ViewComponent i završava se sa **ViewComponent**
- Komponenta vraća pogled
- Logika View komponente se definiše u metodi **Invoke** ili **InvokeAsync** koja vraća **IViewComponentResult**
- Klasa može biti smeštena u bilo kom folderu aplikacije
- Obično se kreira folder **Components** gde se smeštaju klase komponenti

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()
    {
        return View(_db.Osobe.ToList());
    }

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

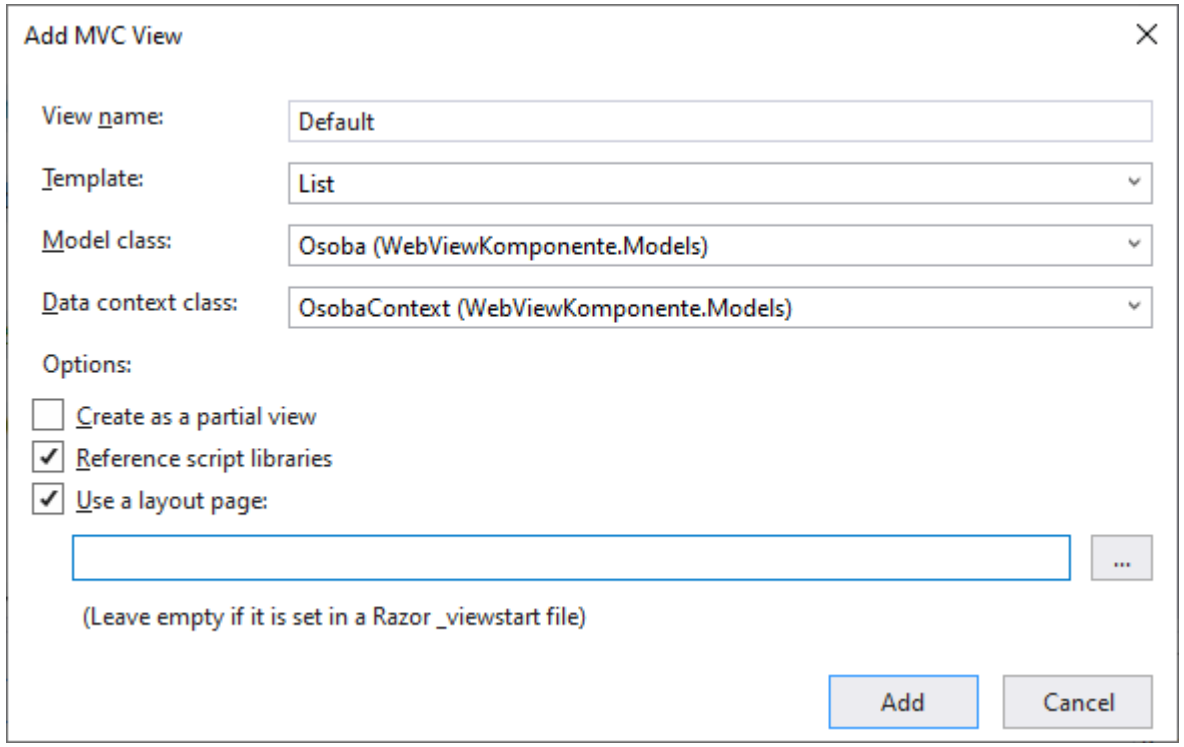
Poziv View komponente

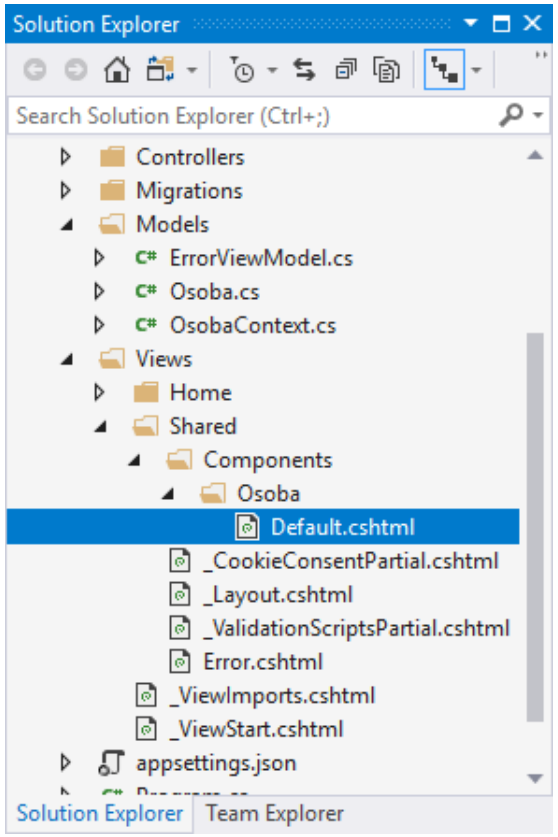
- Pogled koji odgovara View komponenti smešta se unutar foldera **Views/Shared/Components/Ime_Componente**
- Podrazumevano ime pogleda za komponentu je Default.cshtml
- View komponenta se može pozivati direktno iz kontrolera ali je uobičajeno pozivamo iz pogleda

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

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

Kreiranje pogleda za komponentu





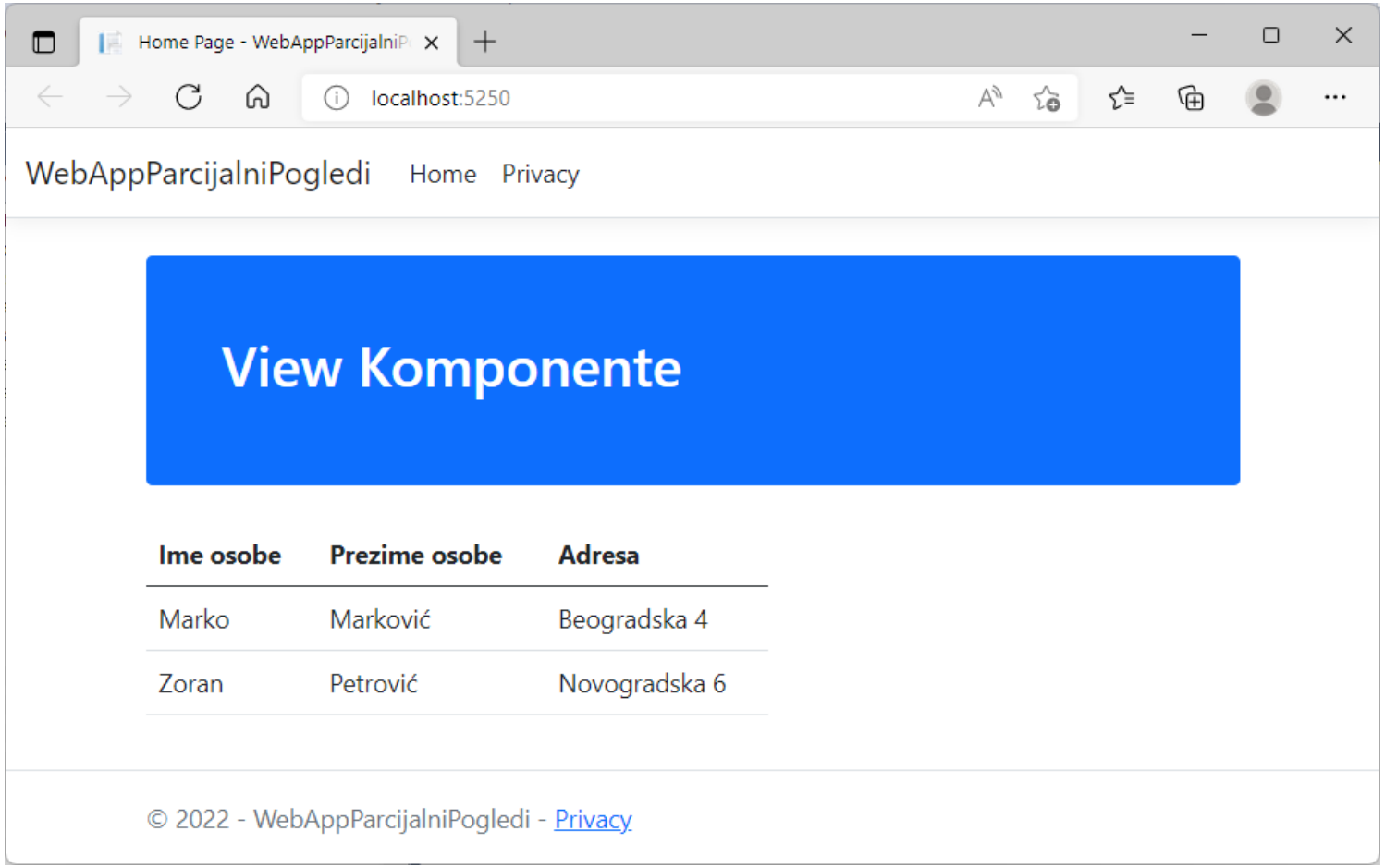
Default.cshtml

```
@model IEnumerable<WebViewKomponente.Models.Osoba>

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

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



_ViewImports.cshtml

```
@using WebApp06
@using WebApp06.Models
@addTagHelper *, Microsoft.AspNetCore.Mvc.TagHelpers
@addTagHelper *, WebApp06
@using WebApp06.Servisi
@using WebApp06.Components
```

Registracija View komponente kao tag helpera

Poziv komponente

```
<div class="jumbotron">
    <h1>View Komponente</h1>
</div>

<div class="row">
    <div class="col-md-7">
        <vc:osoba />
    </div>
</div>
```

Komponenta sa parametrom

```
public IViewComponentResult Invoke(int id=0)
{
    IEnumerable<Osoba> listaOsoba = db.Osobe;
    if (id != 0)
    {
        listaOsoba = listaOsoba.Where(os => os.OsobaId > id);
    }
    return View(listaOsoba);
}
```

Poziv komponente sa parametrima

```
<div class="row">
    <div class="col-md-7">
        @await Component.InvokeAsync("Osoba", new {id=1 })
    </div>
</div>
<div class="row">
    <div class="col-md-7">
        <vc:osoba id="1" />
    </div>
</div>
```

Pitanje 1

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 2

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 3

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 4

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 5

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