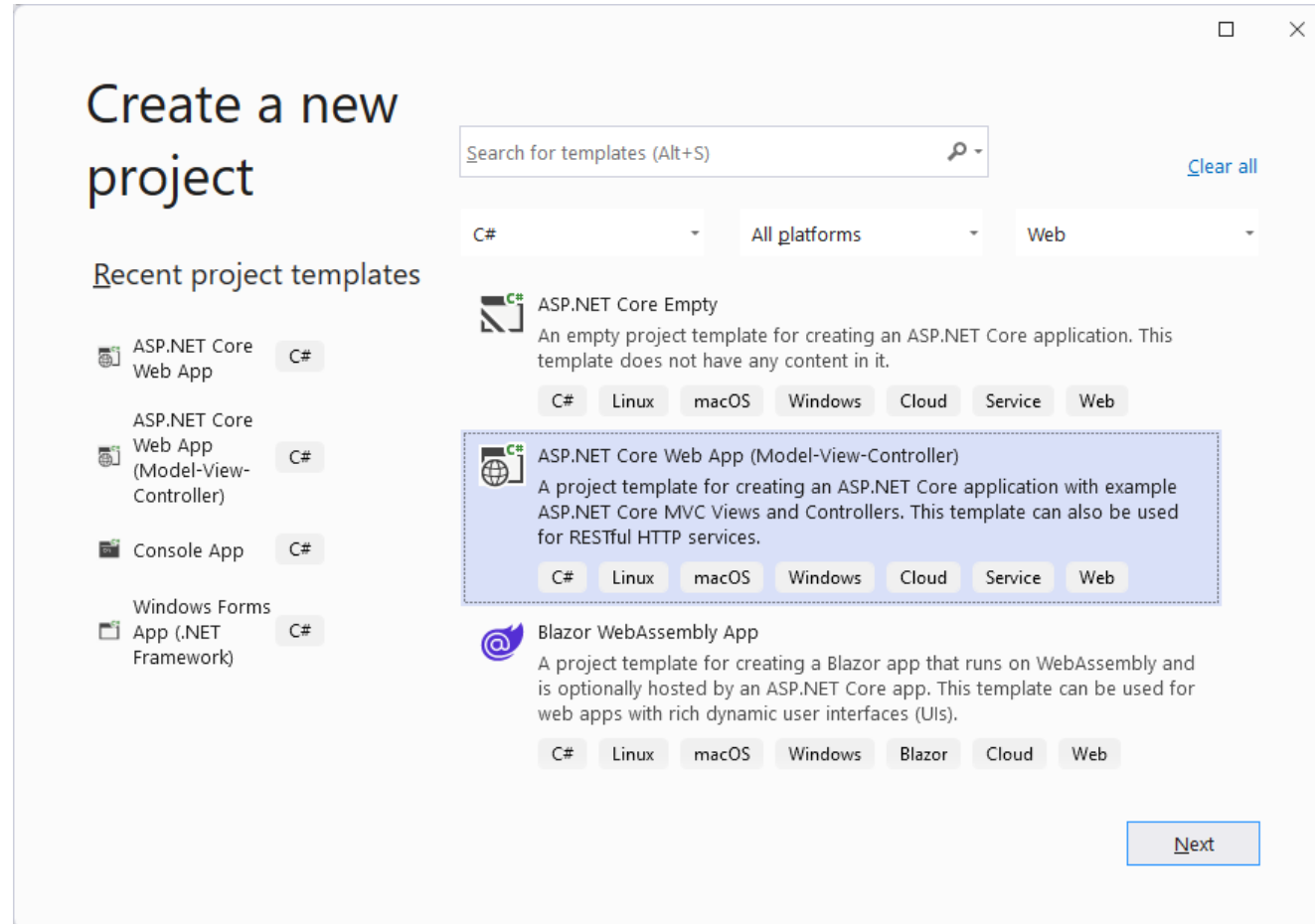
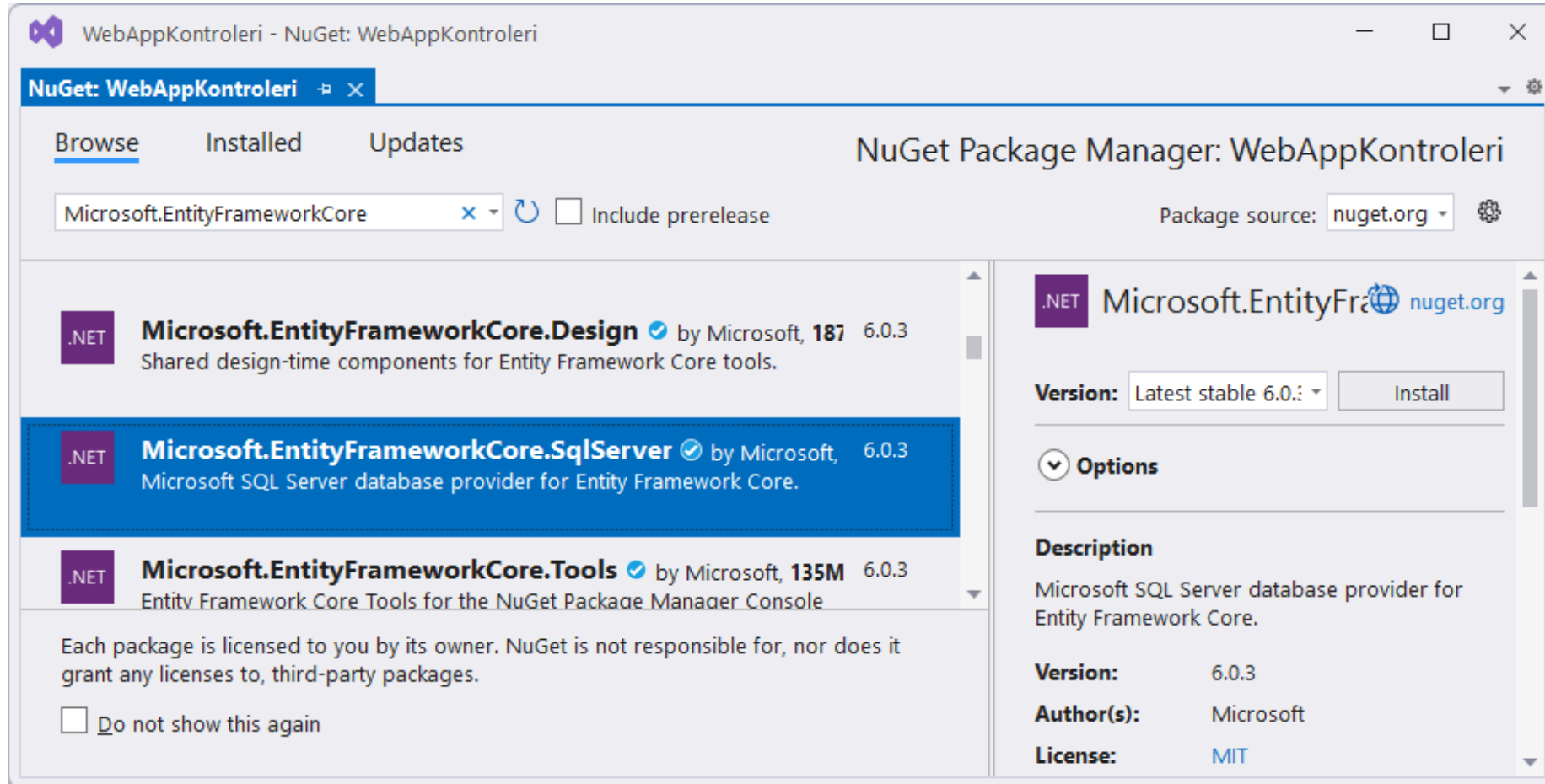


Kreiranje MVC kontrolera

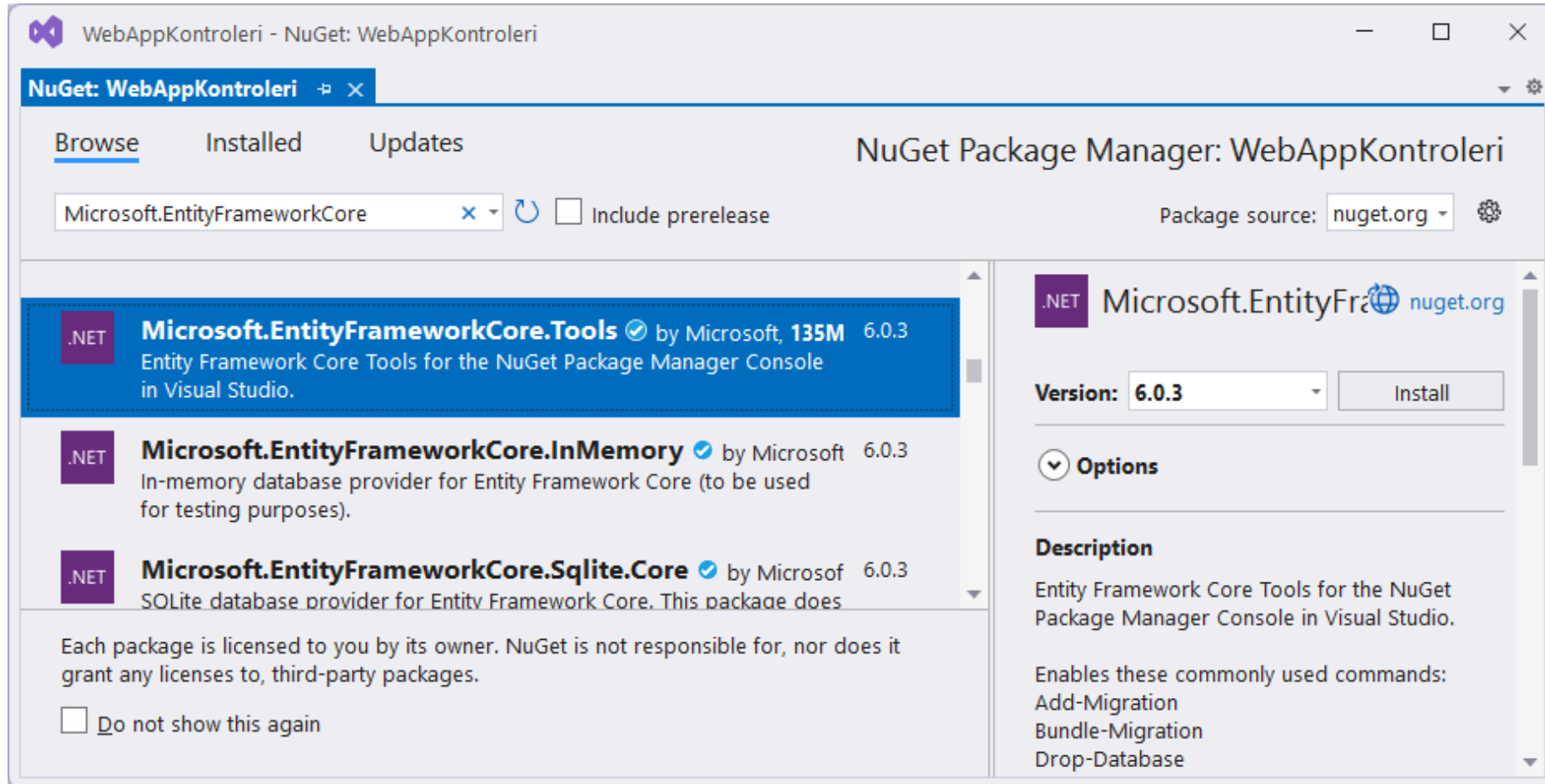
Kreiranje ASP.NET Core Web aplikacije



Microsoft.EntityFrameworkCore.SqlServer



Microsoft.EntityFrameworkCore.Tools



Pojam kontrolera

- Kontroler je C# klasa čije su javne metode odgovorne za obradu HTTP zahteva kod ASP.NET Core MVC web aplikacija
- Ime kontrolerske klase se završava sa Controller
- Kontroler je klasa koja je obično izvedena iz bazne klase Controller
- Metoda kontrolera koja obrađuje HTTP zahtev naziva se akciona metoda ili akcija
- Svaki kontroler sadrži jednu ili više akcionih metoda
- Akciona metoda kontrolera mora biti javna, ne može biti statička, i ne može se overloadovati
- Akciona metoda može vratiti objekat bilo kog tipa
- Uobičajeno je da akciona metoda vraća objekat koji implementira IActionResult interfejs
- Akcija kontrolera prosleđuje podatke odgovarajućem pogledu

Rutiranje

- Rute opisuju kako se URL adrese zahteva mapiraju u akcije kontrolera
- Rute se definišu unutar middleware komponente za rutiranje unutar fajla Program.cs
- Metoda **MapControllerRoute** klase **WebApplication** se koristi za kreiranje default rute
- Ako želimo da pozovemo **Details** akcionu metodu **OsobaController** klase i da toj metodi prosledimo id parametar vrednosti 3 potrebno je uneti sledeću adresu u browseru:
`http://<app>/osoba/details/3`

```
app.MapControllerRoute(  
    name: "default",  
    pattern: "{controller=Home}/{action=Index}/{id?}");
```

HomeController

```
public class HomeController : Controller
{
    public IActionResult Index()
    {
        return View();
    }
}
```

Interfejs IActionResult

- Ovim interfejsom definiše se ugovor koji predstavlja rezultat izvršavanja akcione metode
- Obično akcione metode vraćaju objekat koji implementira IActionResult interfejs
- Metoda View() bazne klase Controller vraća instancu klase ViewResult
- Metoda Json() vraća objekat klase JsonResult i koristi se kada akciona metoda treba da generiše json string

Model klasa Osoba i DbContext klasa OsobaContext

```
[Table("Osoba")]
public class Osoba
{
    public int OsobaId { get; set; }

    [Required]
    [StringLength(30)]
    public string Ime { get; set; }

    [Required]
    [StringLength(30)]
    public string Prezime { get; set; }

    [Required]
    [StringLength(100)]
    public string Adresa { get; set; }
}
```

```
public class OsobaContext : DbContext
{
    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=OsobaDb1303;Integrated Security=True;Encrypt=False"
  }
}
```

Registrujemo DbContext kao servis

```
using WebMvcKontroler.Models;  
using Microsoft.EntityFrameworkCore;
```

```
// Add services to the container.  
var connectionString =  
builder.Configuration.GetConnectionString("DefaultConnection");  
builder.Services.AddDbContext<OsobaContext>(opcije =>  
    opcije.UseSqlServer(connectionString));  
builder.Services.AddControllersWithViews();
```

Promenimo DbContext klasu

```
using Microsoft.EntityFrameworkCore;
```

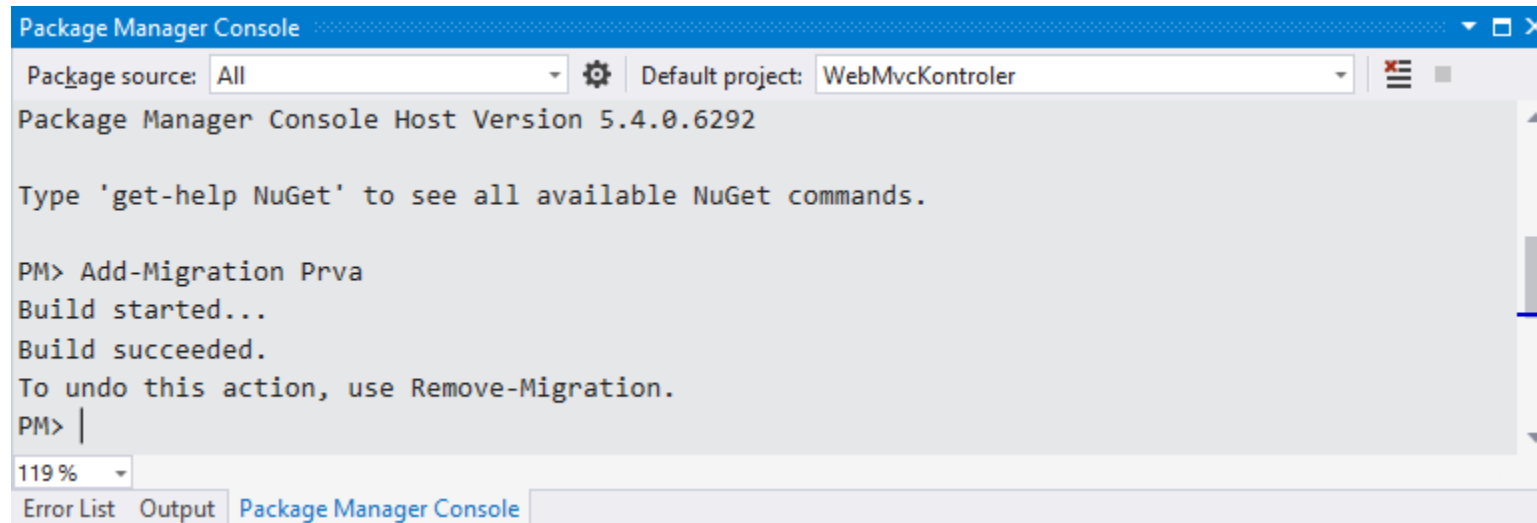
```
public class OsobaContext : DbContext
{
    public OsobaContext(DbContextOptions<OsobaContext> opcije) : base(opcije)
    {
    }

    public DbSet<Osoba> Osobe { get; set; }
}
```

Kreiranje baze podataka primenom migracija

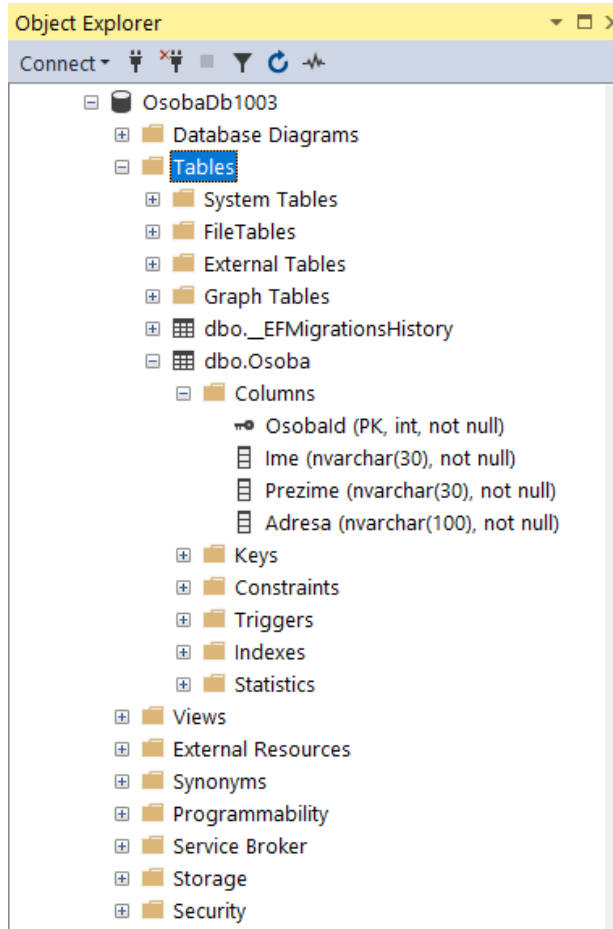
```
PM> Add-Migration Prva
```

```
PM> Update-Database
```

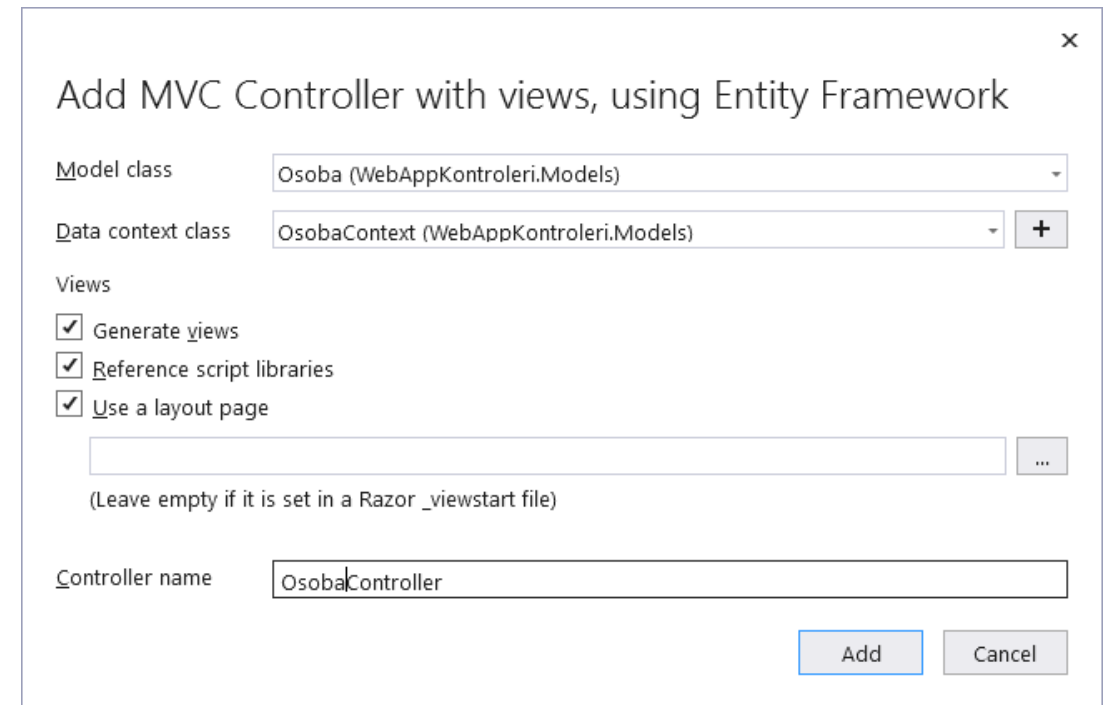
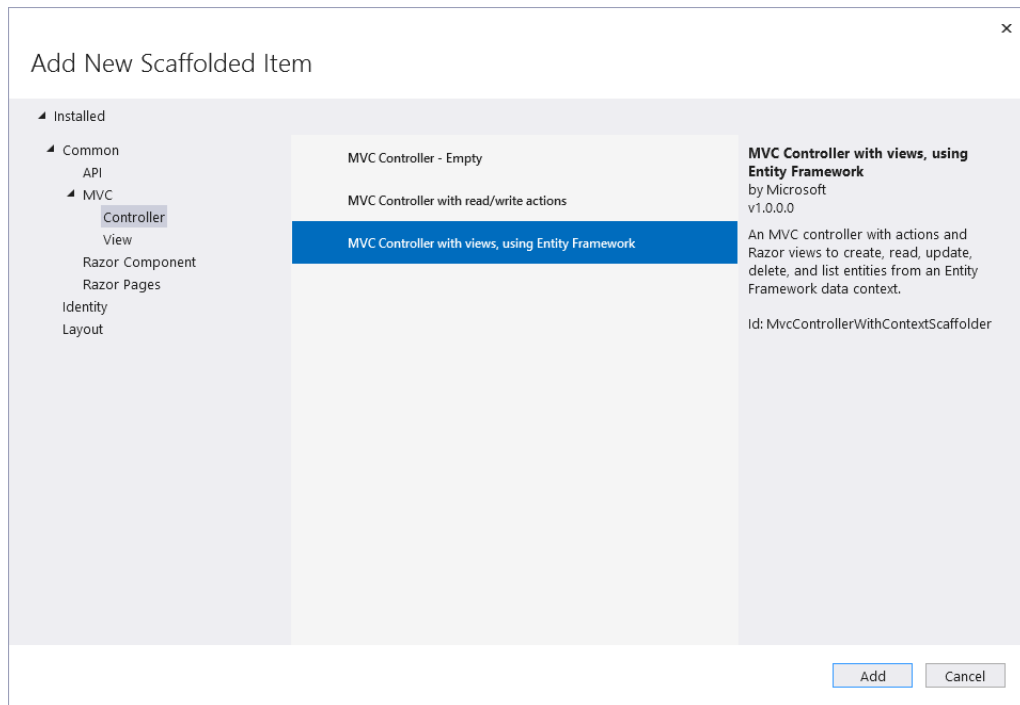


```
Package Manager Console
Package source: All
Default project: WebMvcKontroler
Package Manager Console Host Version 5.4.0.6292
Type 'get-help NuGet' to see all available NuGet commands.
PM> Add-Migration Prva
Build started...
Build succeeded.
To undo this action, use Remove-Migration.
PM> |
```

Kreirana baza podataka



Kreiranje kontrolera



Klasa OsobaController

```
public class OsobaController : Controller
{
    private readonly OsobaContext _context;

    public OsobaController(OsobaContext context)
    {
        _context = context;
    }
}
```


Akcija Index asinhrona verzija

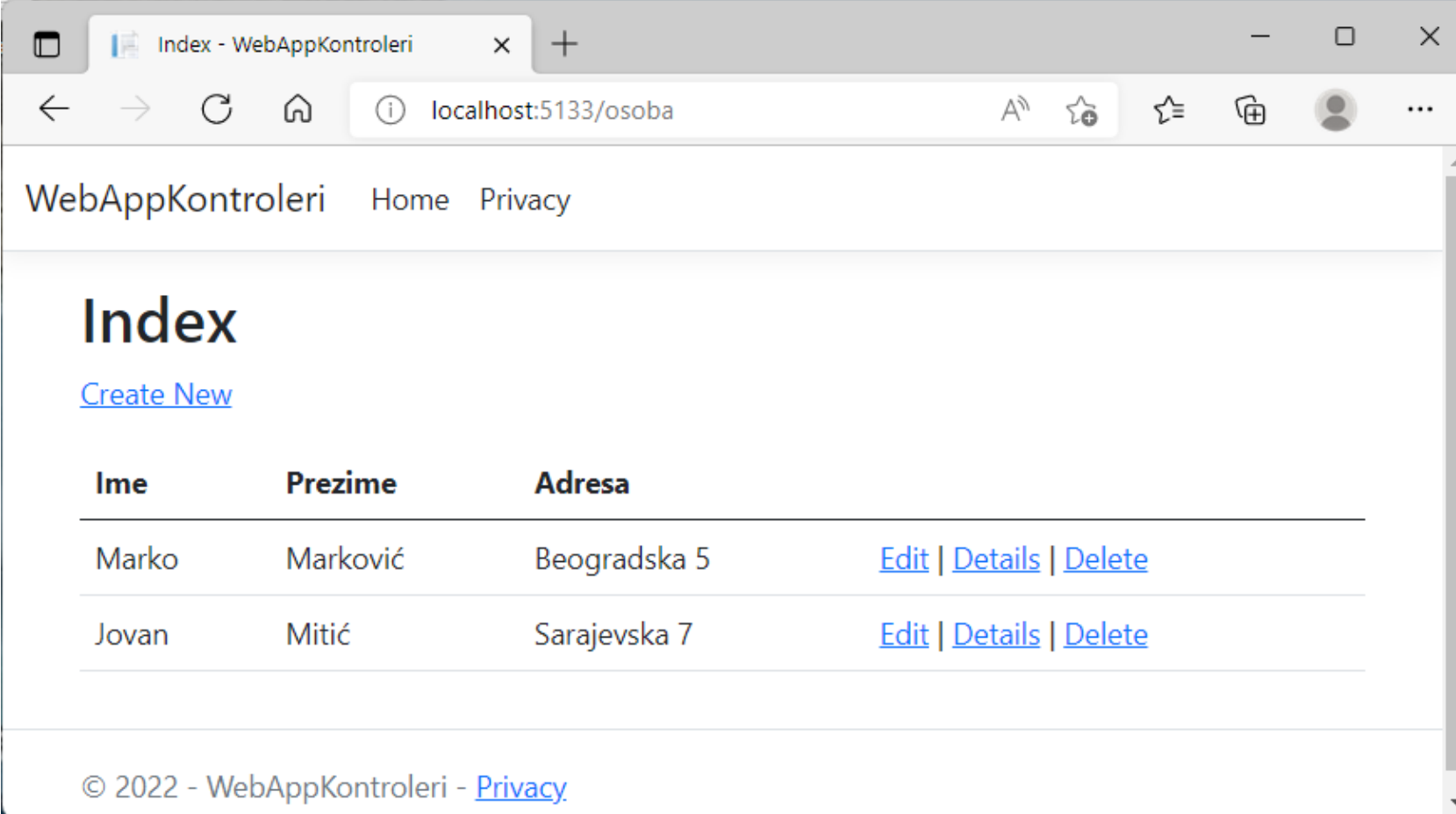
```
using System.Threading.Tasks;  
using Microsoft.AspNetCore.Mvc;  
using WebMvcKontroler.Models;  
using Microsoft.EntityFrameworkCore;
```

```
public async Task<IActionResult> Index()  
{  
    return View(await _context.Osobe.ToListAsync());  
}
```

Akcija Index sinhrona verzija

```
// GET: Osoba
public IActionResult Index()
{
    return View(_context.Osobe.ToList());
}
```

Poziv Index akcije kontrolera Osoba



The screenshot shows a web browser window with the following content:

- Browser tab: Index - WebAppKontroleri
- Address bar: localhost:5133/osoba
- Navigation links: WebAppKontroleri, Home, Privacy
- Section title: Index
- Link: [Create New](#)
- Table with columns: Ime, Prezime, Adresa, and actions (Edit, Details, Delete).
- Footer: © 2022 - WebAppKontroleri - [Privacy](#)

Ime	Prezime	Adresa	
Marko	Marković	Beogradska 5	Edit Details Delete
Jovan	Mitić	Sarajevska 7	Edit Details Delete

Akciona metoda Details

```
// GET: Osoba/Details/5
public async Task<IActionResult> Details(int? id)
{
    if (id == null)
    {
        return NotFound();
    }

    var osoba = await _context.Osobe
        .FirstOrDefaultAsync(m => m.OsobaId == id);
    if (osoba == null)
    {
        return NotFound();
    }
    return View(osoba);
}
```

Metoda NotFound() kreira objekat klase NotFoundResult koja proizvodi Not Found (404) odziv

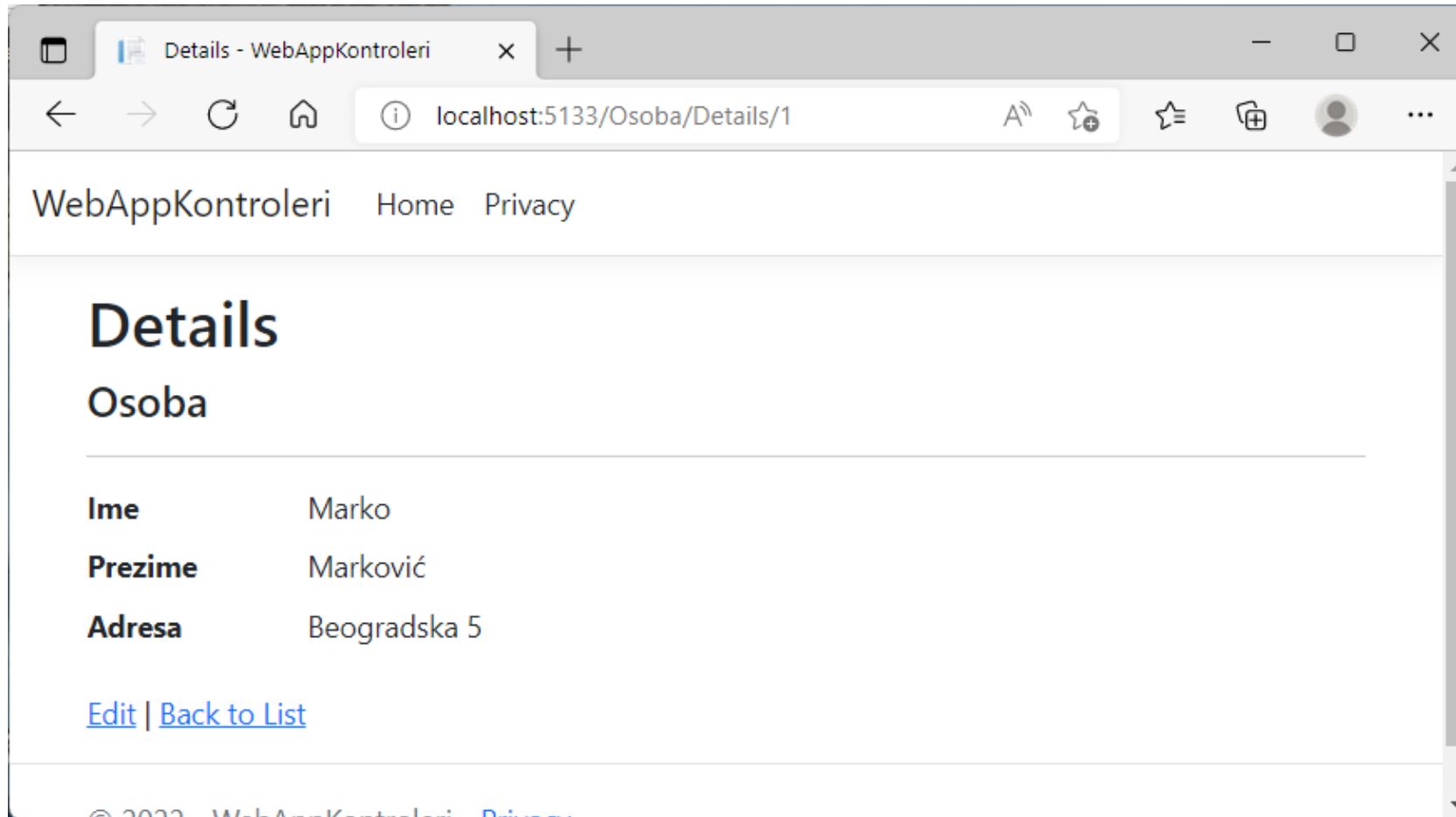
Metoda Details sinhrona verzija

```
public IActionResult Details(int? id)
{
    if (id == null || _context.Osobe == null)
    {
        return NotFound();
    }

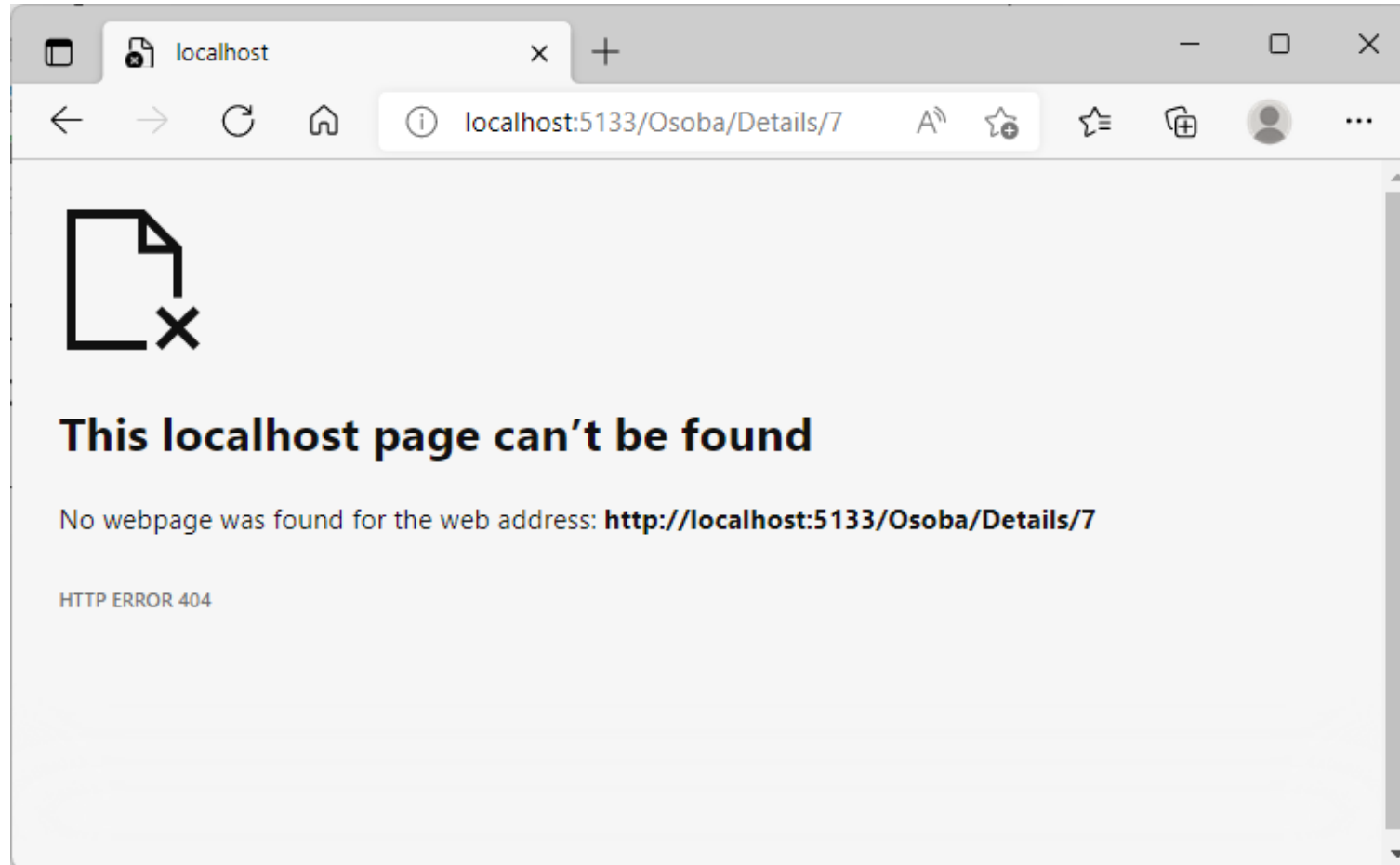
    var osoba = _context.Osobe
        .FirstOrDefault(m => m.OsobaId == id);
    if (osoba == null)
    {
        return NotFound();
    }

    return View(osoba);
}
```

Poziv Details akcije kontrolera Osoba



Prosleđivanje nepostojećeg id-a



Dodavanje middleware komponente za obradu odgovara sa kodnim statusom od 400 do 599

```
app.UseStatusCodePages();
app.UseStaticFiles();

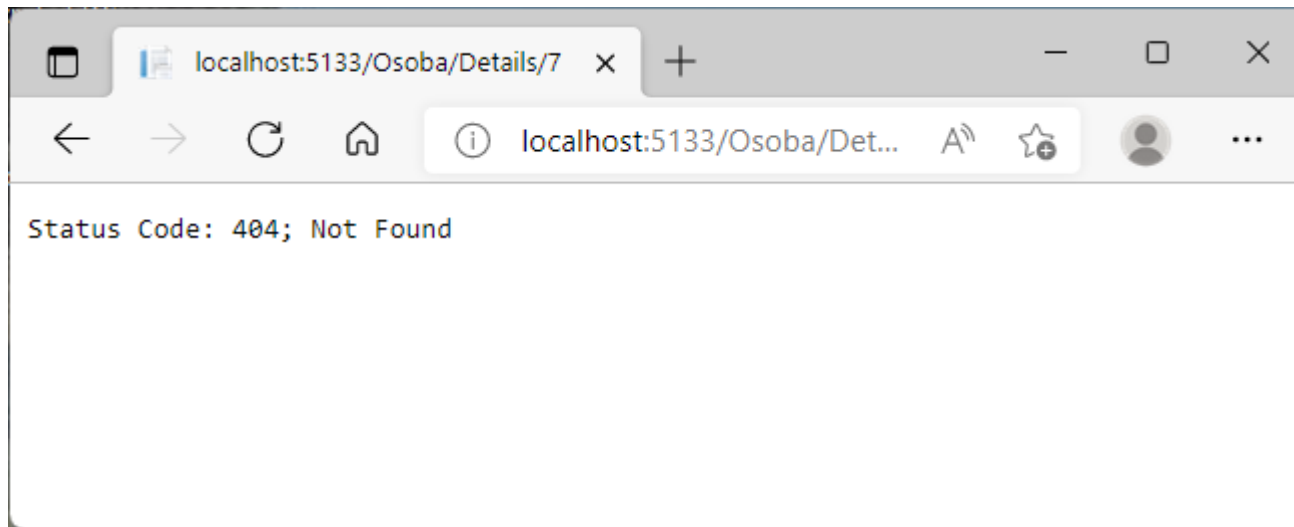
app.UseRouting();

app.UseAuthorization();

app.MapControllerRoute(
    name: "default",
    pattern: "{controller=Home}/{action=Index}/{id?}");

app.Run();
```


Loš poziv Details akcione metode



Redirekcija na korisnički kreiranu stranu greške

```
public IActionResult Details(int? id)
{
    if (id == null || _context.Osobe == null)
    {
        return RedirectToAction("Error");
    }

    var osoba = _context.Osobe
        .FirstOrDefault(m => m.OsobaId == id);
    if (osoba == null)
    {
        return RedirectToAction("Error");
    }

    return View(osoba);
}
```

Akcija Error()

```
public IActionResult Error()  
{  
    return View();  
}
```

Pogled Error.cshtml

```
@{  
    ViewData["Title"] = "Greska";  
}  
  
<h1>Greska!</h1>  
<p>Doslo je do greske!</p>
```

GET i POST akciona metoda Create

```
// GET: Osoba/Create
public IActionResult Create()
{
    return View();
}

[HttpPost]
[ValidateAntiForgeryToken]
public async Task<IActionResult> Create([Bind("OsobaId,Ime,Prezime,Adresa")] Osoba osoba)
{
    if (ModelState.IsValid)
    {
        _context.Add(osoba);
        await _context.SaveChangesAsync();
        return RedirectToAction(nameof(Index));
    }
    return View(osoba);
}
```

Akciona metoda Create sinhrona verzija

```
[HttpPost]
[ValidateAntiForgeryToken]
public IActionResult Create([Bind("OsobaId, Ime, Prezime, Adresa")] Osoba
osoba)
{
    if (ModelState.IsValid)
    {
        _context.Add(osoba);
        _context.SaveChanges();
        return RedirectToAction(nameof(Index));
    }
    return View(osoba);
}
```

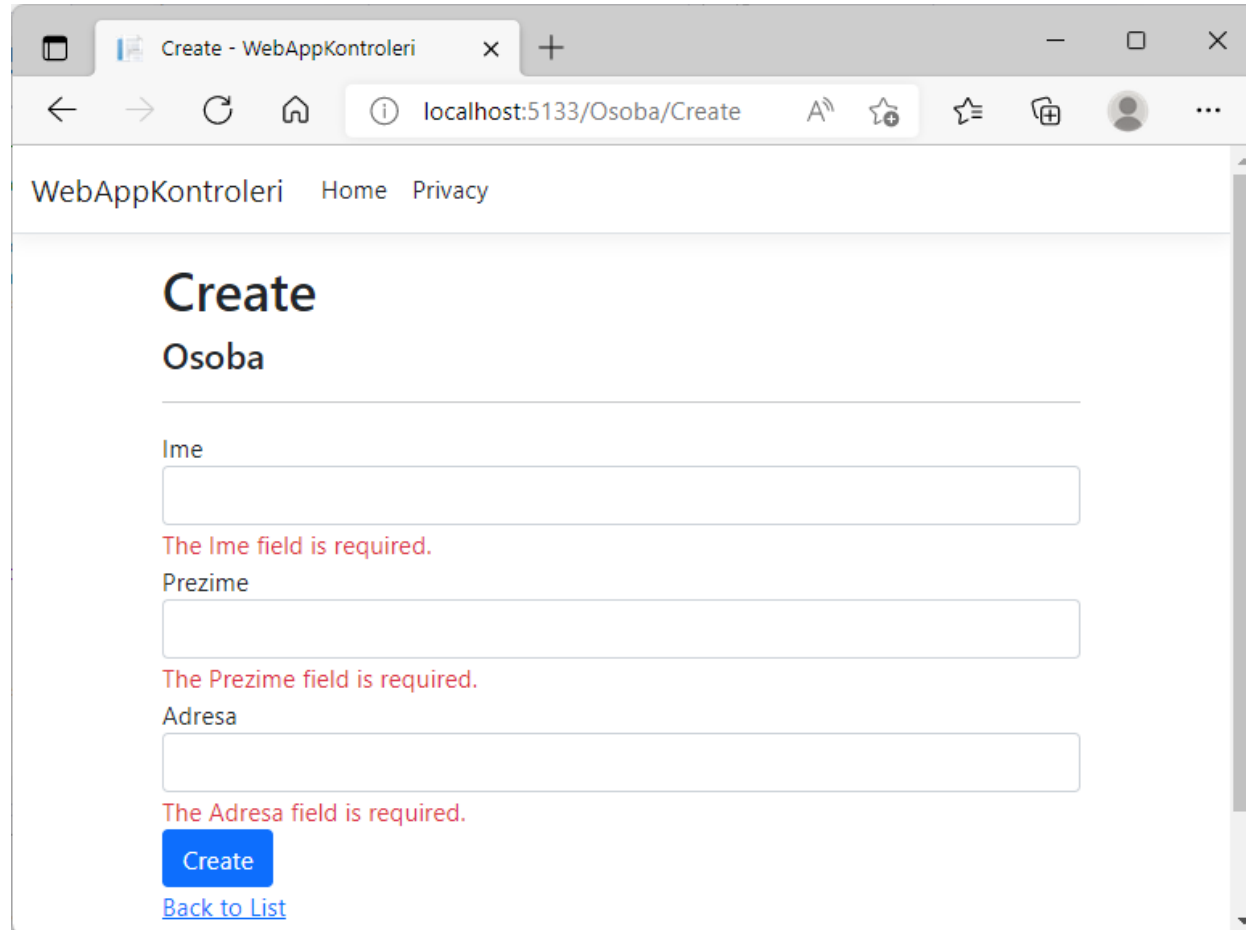
Model binding

- Model binding je process kreiranja .NET objekata korišćenjem podataka iz HTTP zahteva
- Kreirani objekti se prosleđuju akcionim metodama kontrolera
- Model binder je komponenta MVC aplikacije koja obezbeđuje podatke za akcione metode iz različitih delova HTTP zahteva
- Model binder koristi tri izvora podataka da bi mapirao HTTP zahteve u podatke neophodne akcionim metodama
 - vrednosti polja forme
 - vrednosti parametara iz rute
 - vrednosti parametara iz query stringova

Svojstva i atributske klase koje koristi metoda Create

- **ValidateAntiForgeryToken** predstavlja atributsku klasu koja sprečava falsifikovanje zahteva
- **ModelState** je property kontrolera i sadrži kolekciju (name,value) parova koju su prosleđeni u POST zahtevu do servera, takođe sadrži i kolekciju grešaka za svaku vrednost koja je poslata do servera
- **ModelState.IsValid** ukazuje da su greške u modelu dodate u ModelState
- Pomoću atributske klase **Bind** se specificiraju koja svojstva model klase se koriste u kreiranju .NET objekata od strane model bindera
- Metoda **RedirectToAction** poziva akciju kontrolera koja se specificira kao string parameter metode
- Ključna reč **nameof** se koristi da se dobije ime promenljive, tipa ili metode

Poziv akcione metode Create kontrolera Osoba



The screenshot shows a web browser window with the address bar displaying 'localhost:5133/Osoba/Create'. The page title is 'WebAppKontroleri' and the navigation menu includes 'Home' and 'Privacy'. The main heading is 'Create Osoba'. Below the heading are three input fields: 'Ime', 'Prezime', and 'Adresa'. Each field has a red error message below it: 'The Ime field is required.', 'The Prezime field is required.', and 'The Adresa field is required.'. At the bottom of the form is a blue 'Create' button and a blue link labeled 'Back to List'.

WebAppKontroleri Home Privacy

Create Osoba

Ime

The Ime field is required.

Prezime

The Prezime field is required.

Adresa

The Adresa field is required.

[Create](#)

[Back to List](#)

Edit GET metoda – asinhrona verzija

```
public async Task<IActionResult> Edit(int? id)
{
    if (id == null)
    {
        return NotFound();
    }

    var osoba = await _context.Osobe.FindAsync(id);
    if (osoba == null)
    {
        return NotFound();
    }
    return View(osoba);
}
```

Edit GET metoda – sinhrona verzija

```
// GET: Osoba/Edit/5
public IActionResult Edit(int? id)
{
    if (id == null || _context.Osobe == null)
    {
        return RedirectToAction("Error");
    }

    var osoba = _context.Osobe.Find(id);
    if (osoba == null)
    {
        return RedirectToAction("Error");
    }
    return View(osoba);
}
```

Edit POST metoda - asinhrona verzija

```
[HttpPost]
[ValidateAntiForgeryToken]
public async Task<IActionResult> Edit(int id,
[Bind("OsobaId,Ime,Prezime,Adresa")] Osoba osoba)
{
    if (id != osoba.OsobaId)
    {
        return NotFound();
    }

    if (ModelState.IsValid)
    {
        try
        {
            _context.Update(osoba);
            await _context.SaveChangesAsync();
        }
        catch (DbUpdateConcurrencyException)
        {
            if (!OsobaExists(osoba.OsobaId))
            {
                return NotFound();
            }
            else
            {
                throw;
            }
        }
        return RedirectToAction(nameof(Index));
    }
    return View(osoba);
}
```

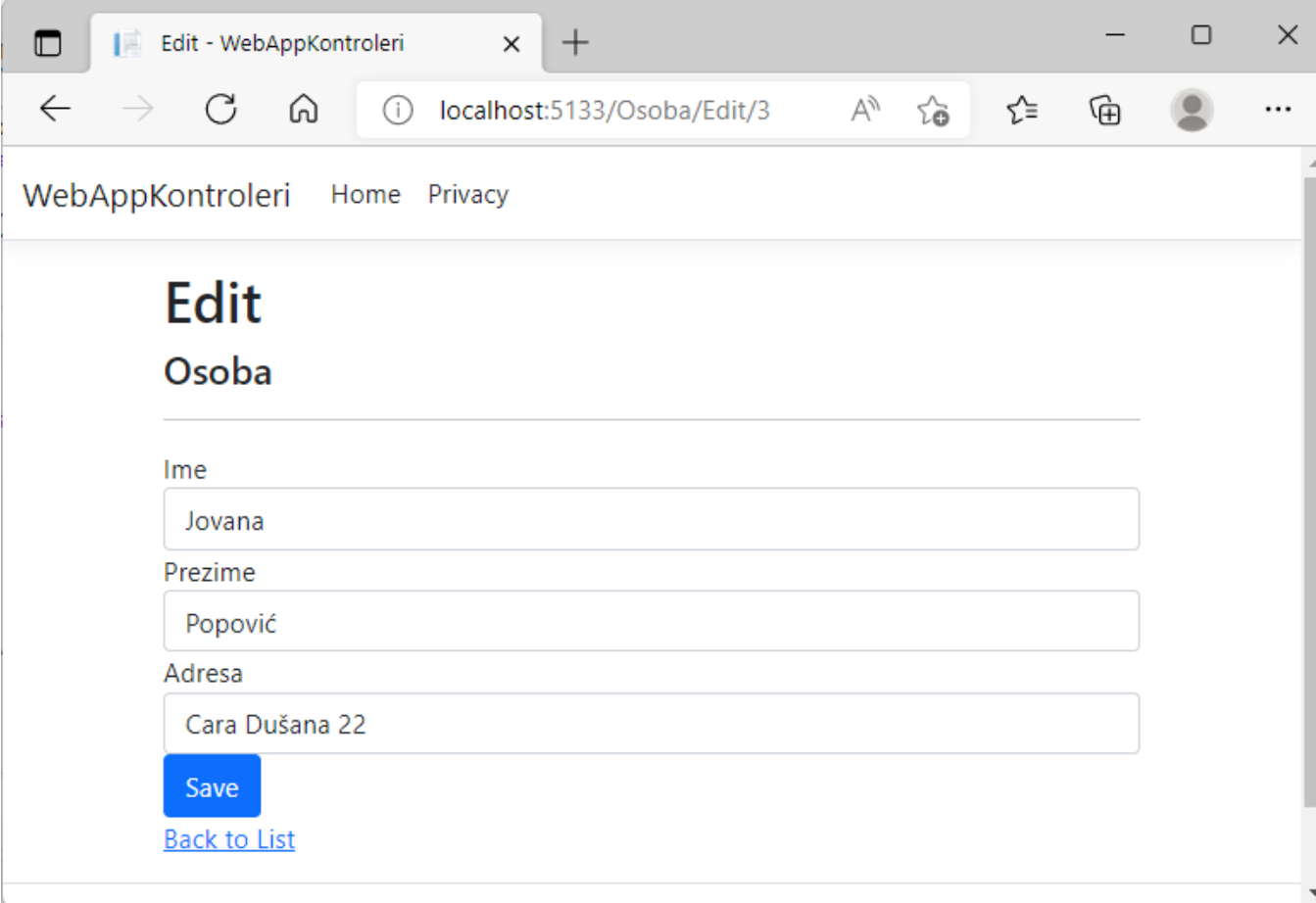
```
private bool OsobaExists(int id)
{
    return _context.Osobe.Any(e => e.OsobaId == id);
}
```

Edit POST metoda - sinhrona verzija

```
[HttpPost]
[ValidateAntiForgeryToken]
public IActionResult Edit(int id, [Bind("OsobaId,Ime,Prezime,Adresa")] Osoba osoba)
{
    if (id != osoba.OsobaId)
    {
        return RedirectToAction("Error");
    }

    try
    {
        if (ModelState.IsValid)
        {
            _context.Update(osoba);
            _context.SaveChanges();
            return RedirectToAction(nameof(Index));
        }
        return View(osoba);
    }
    catch (DbUpdateConcurrencyException)
    {
        if (!OsobaExists(osoba.OsobaId))
        {
            return RedirectToAction("Error");
        }
        return RedirectToAction("Error");
    }
    catch (Exception)
    {
        // Log the exception or do something else with it.
        return RedirectToAction("Error");
    }
}
```

Poziv Edit GET metode kontrolera Osoba



The screenshot shows a web browser window with the following details:

- Tab: Edit - WebAppKontroleri
- Address bar: localhost:5133/Osoba/Edit/3
- Page title: WebAppKontroleri
- Navigation: Home, Privacy
- Section: **Edit Osoba**
- Form fields:
 - Ime: Jovana
 - Prezime: Popović
 - Adresa: Cara Dušana 22
- Buttons: Save (blue), [Back to List](#) (blue link)

Delete GET metoda – asinhrona verzija

```
public async Task<IActionResult> Delete(int? id)
{
    if (id == null)
    {
        return NotFound();
    }

    var osoba = await _context.Osobe
        .FirstOrDefaultAsync(m => m.OsobaId == id);
    if (osoba == null)
    {
        return NotFound();
    }

    return View(osoba);
}
```

Delete GET metoda – sinhrona verzija

```
public async Task<IActionResult> Delete(int? id)
{
    if (id == null || _context.Osobe == null)
    {
        return RedirectToAction("Error");
    }

    var osoba = await _context.Osobe
        .FirstOrDefaultAsync(m => m.OsobaId == id);
    if (osoba == null)
    {
        return RedirectToAction("Error");
    }

    return View(osoba);
}
```


Delete POST metoda – asinhrona verzija

ActionName klasa ukazuje da se radi o akciji Delete

```
[HttpPost, ActionName("Delete")]
[ValidateAntiForgeryToken]
public async Task<IActionResult> DeleteConfirmed(int id)
{
    if (_context.Osobe == null)
    {
        return RedirectToAction("Error");
    }
    var osoba = await _context.Osobe.FindAsync(id);
    if (osoba != null)
    {
        _context.Osobe.Remove(osoba);
    }

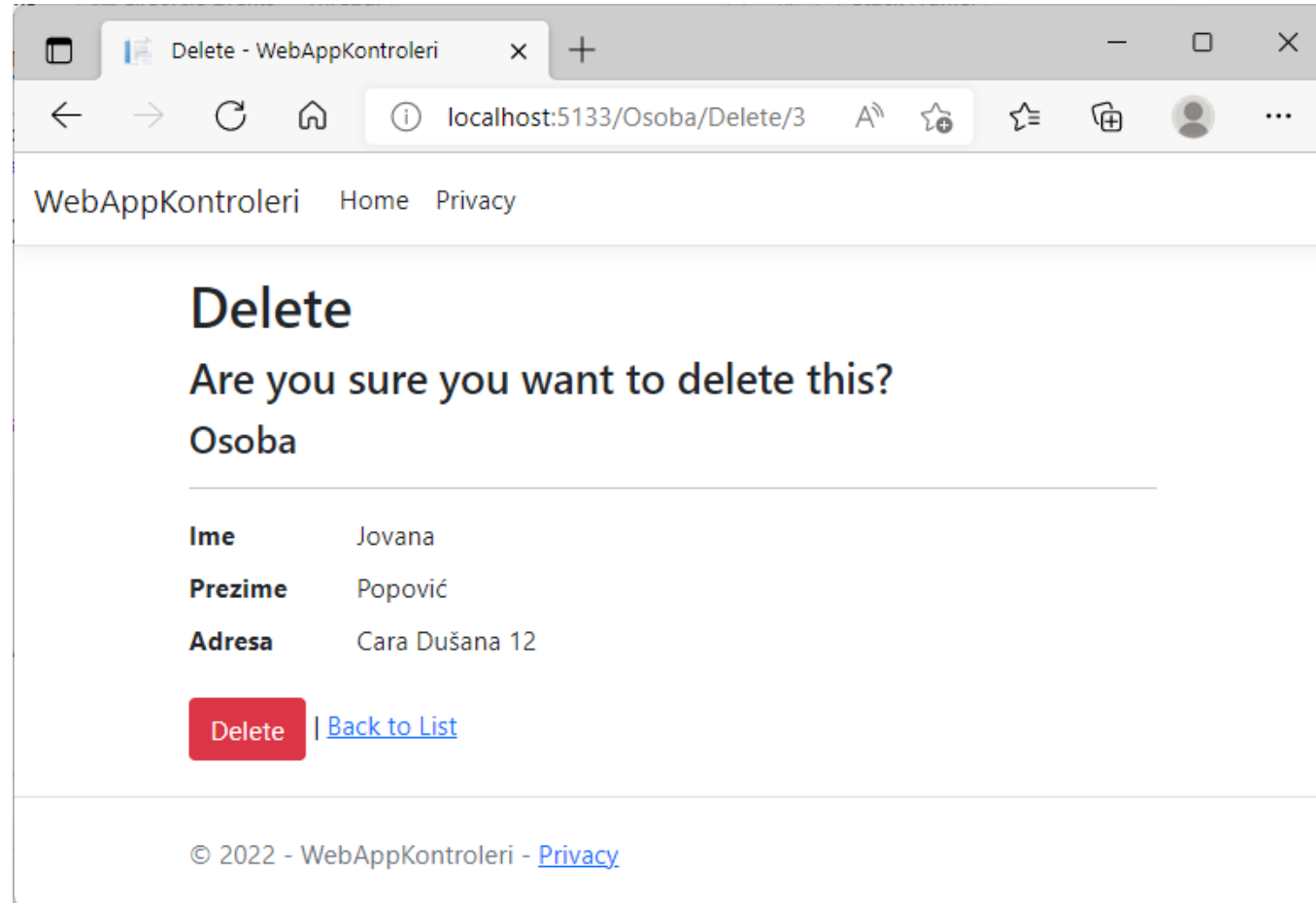
    await _context.SaveChangesAsync();
    return RedirectToAction(nameof(Index));
}
```

Delete POST metoda – sinhrona verzija

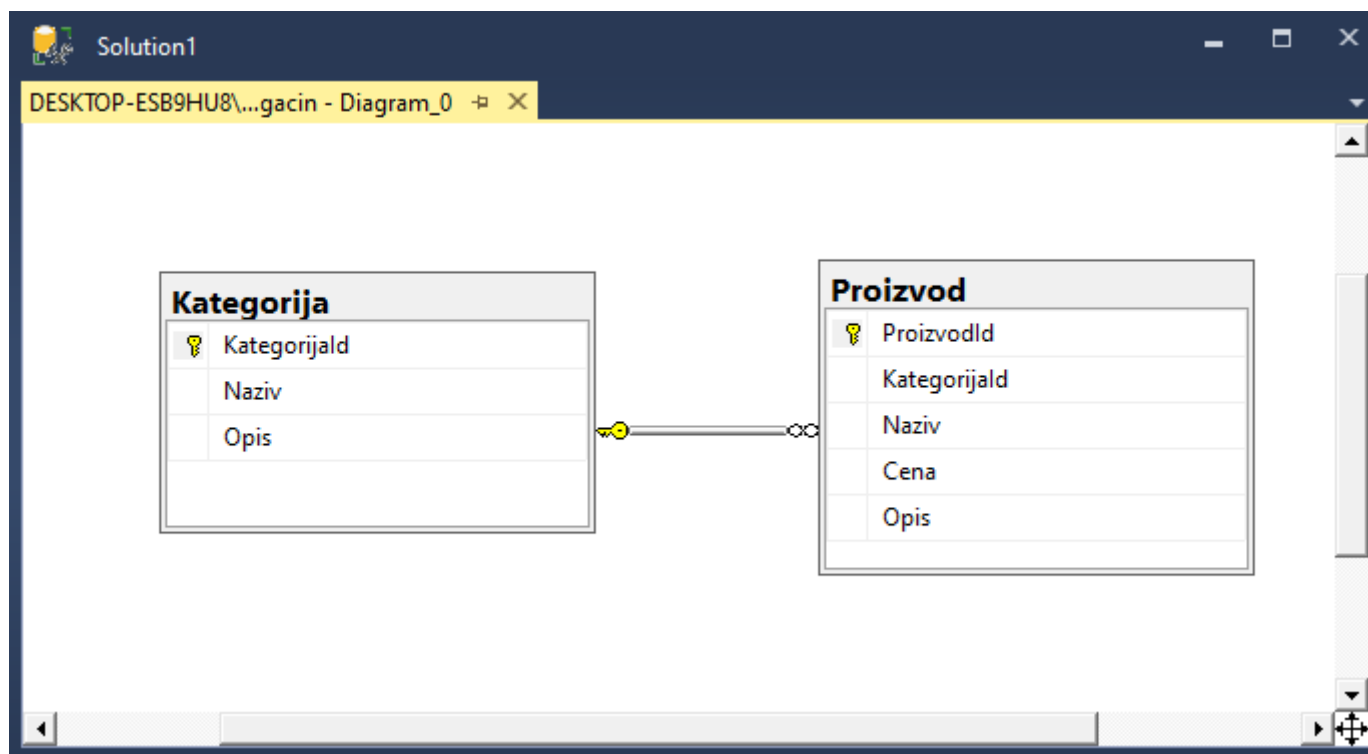
```
[HttpPost, ActionName("Delete")]
[ValidateAntiForgeryToken]
public IActionResult DeleteConfirmed(int id)
{
    if (_context.Osobe == null)
    {
        return RedirectToAction("Error");
    }
    var osoba = _context.Osobe.Find(id);
    if (osoba != null)
    {
        _context.Osobe.Remove(osoba);
    }

    _context.SaveChanges();
    return RedirectToAction(nameof(Index));
}
```

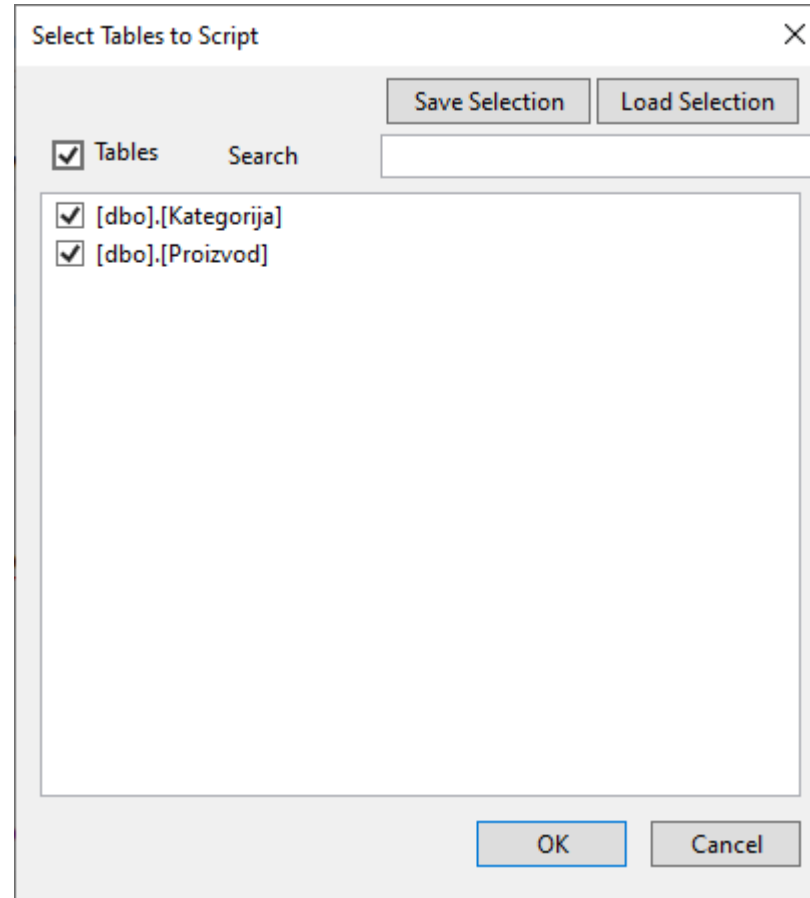
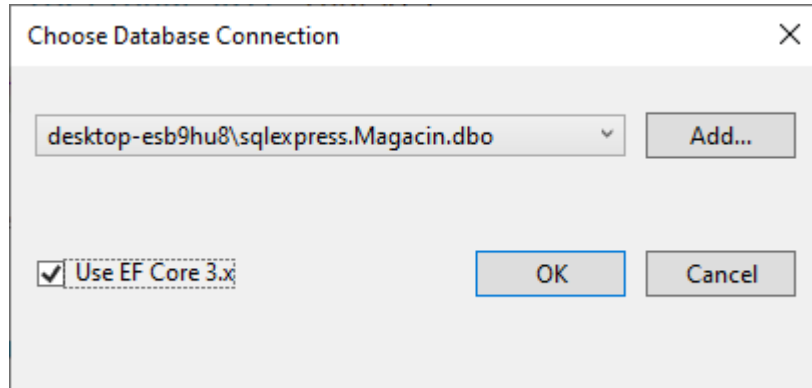
Poziv akcije Delete kontrolera Osoba



Primer filtriranja podataka



EF Core Power Tools



EF Core Power Tools

Generate EF Core Model in Project WebCoreFiltriranje1

Context name
MagacinContext

Namespace
WebCoreFiltriranje1

EntityTypes path (f.ex. Models) - optional
Models

EntityTypes sub-namespace (overrides path) - optional

DbContext path (f.ex. Data) - optional

DbContext sub-namespace (overrides path) - optional

What to generate
EntityTypes & DbContext

Naming

- Pluralize or singularize generated object names (English)
- Use table and column names directly from the database

Use DataAnnotation attributes to configure the model

Customize code using Handlebars templates C#

Include connection string in generated code

Install the EF Core provider package in the project

OK Cancel

Model klasa Kategorija

```
[Table("Kategorija")]
public partial class Kategorija
{
    public Kategorija()
    {
        Proizvodi = new HashSet<Proizvod>();
    }

    public int KategorijaId { get; set; }

    [Required]
    [StringLength(70)]
    public string Naziv { get; set; }

    [StringLength(120)]
    public string Opis { get; set; }

    public virtual ICollection<Proizvod> Proizvodi { get; set; }
}
```

Model klasa Proizvod

```
public partial class Proizvod
{
    public int ProizvodId { get; set; }

    public int KategorijaId { get; set; }

    [Required]
    [StringLength(120)]
    public string Naziv { get; set; }

    [Column(TypeName = "decimal(10, 2)")]
    public decimal Cena { get; set; }

    [StringLength(120)]
    public string Opis { get; set; }
    public virtual Kategorija Kategorija { get; set; }
}
```


DbContext klasa

```
public partial class MagacinContext : DbContext
{
    public MagacinContext(DbContextOptions<MagacinContext> opcije)
        : base(opcije)
    {
    }

    public virtual DbSet<Kategorija> Kategorije { get; set; }
    public virtual DbSet<Proizvod> Proizvodi { get; set; }
}
```

appsettings.json

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "AllowedHosts": "*",
  "ConnectionStrings": {
    "DefaultConnection": "Server=GORAN-HP;Initial Catalog=Magacin;Integrated Security=true"
  }
}
```

DbContext kao servis

```
var connectionString = builder.Configuration.GetConnectionString("DefaultConnection");
builder.Services.AddDbContext<MagacinContext>(options =>
    options.UseSqlServer(connectionString));
// Add services to the container.
builder.Services.AddControllersWithViews();
```

Klasa HomeController

```
public class HomeController : Controller
{
    private readonly MagacinContext _db;
    public HomeController(MagacinContext db)
    {
        _db = db;
    }

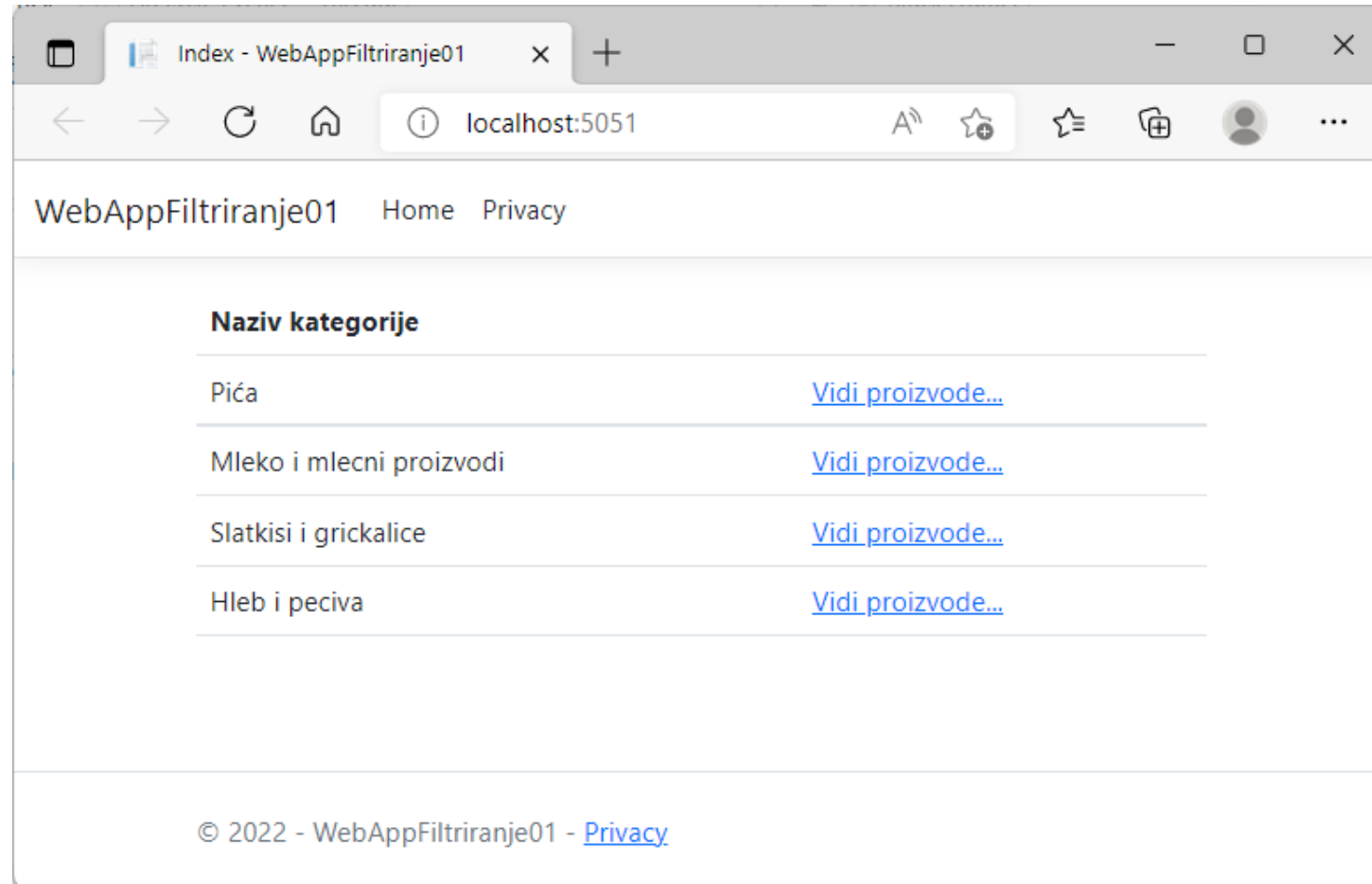
    public IActionResult Index()
    {
        return View(_db.Kategorije.ToList());
    }
}
```

```
@model IEnumerable<Kategorija>

@{
    ViewData["Title"] = "Index";
}

<table class="table">
    <thead>
        <tr>
            <th>Naziv kategorije</th>
            <th></th>
        </tr>
    </thead>
    <tbody>
        @foreach(Kategorija k in Model)
        {
            <tr>
                <td>
                    @k.Naziv
                </td>
                <td>
                    <a href="@($"/Home/VratiProizvode/{@k.KategorijaId}")">Vidi proizvode...</a>
                </td>
            </tr>
        }
    </tbody>
</table>
```

Index pogled



Metoda za filtriranje

```
public IActionResult VratiProizvode(int id=0)
{
    IQueryable<Proizvod> listaProizvoda = db.Proizvodi;
    if (id != 0)
    {
        Kategorija k1 = db.Kategorije.Find(id);
        if (k1 != null)
        {
            ViewBag.Kategorija = k1.Naziv;
            listaProizvoda = listaProizvoda.Where(p => p.KategorijaId == id);
        }
        else
        {
            ViewBag.Kategorija = "";
            return View();
        }
    }
    else
    {
        ViewBag.Kategorija = "Svi proizvodi";
    }

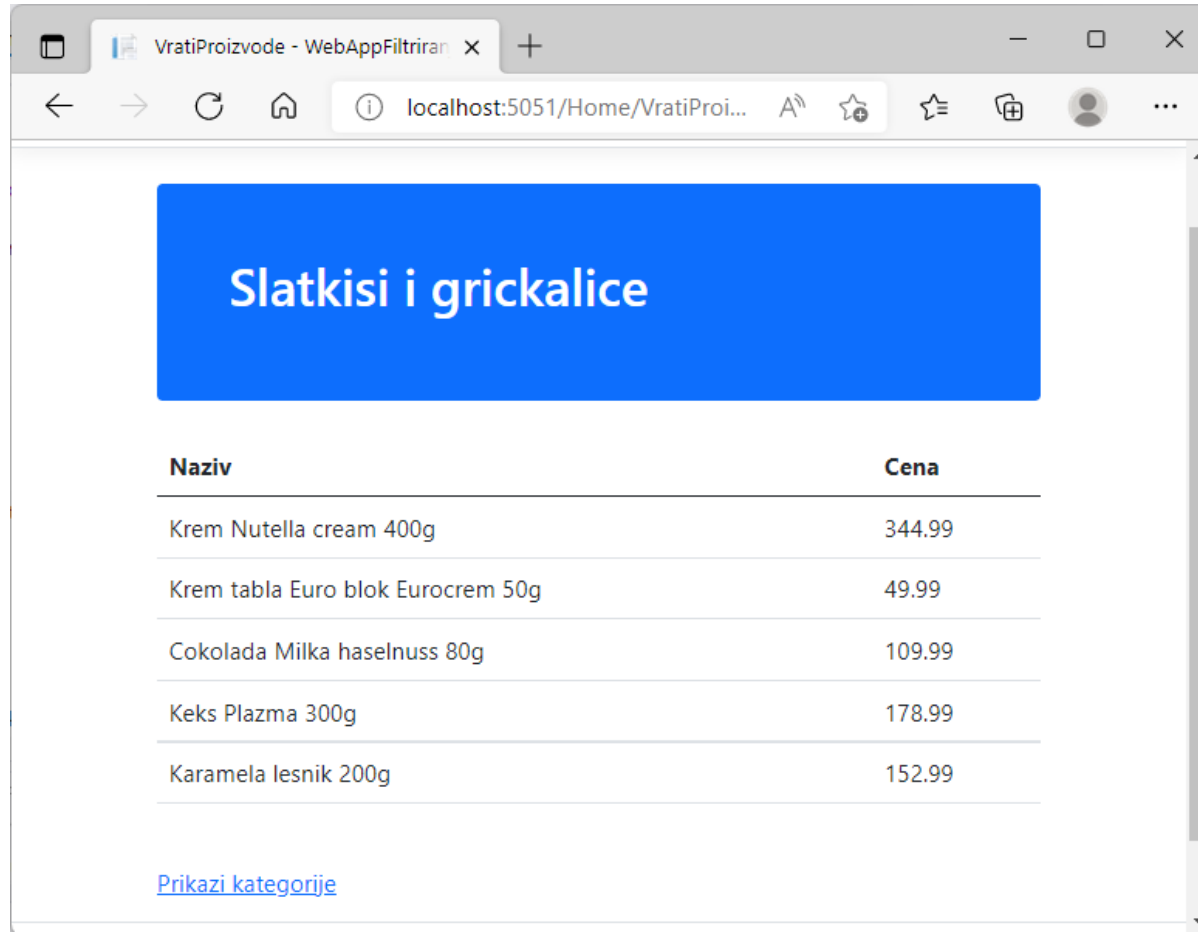
    return View(listaProizvoda);
}
```

VratiProizvode.cshtml

```
@model IEnumerable<WebAppFiltriranje01.Models.Proizvod>
@{
    ViewData["Title"] = "VratiProizvode";
}
<div class="mt-4 p-5 bg-primary text-white rounded">
    <h1>
        @ViewBag.Kategorija
    </h1>
</div>
@if (ViewBag.Kategorija != "")
{
    <table class="table">
        <thead>
            <tr>
                <th>
                    Naziv
                </th>
                <th>
                    Cena
                </th>
            </tr>
        </thead>
```

```
        <tbody>
            @foreach (Proizvod p in Model)
            {
                <tr>
                    <td>
                        @p.Naziv
                    </td>
                    <td>
                        @p.Cena
                    </td>
                </tr>
            }
        </tbody>
    </table>
}
else
{
    <p>Ne postoji kategorija</p>
}
<br>
<a href="/Home/Index">Prikazi kategorije</a>
```


Prikaz proizvoda



The screenshot shows a web browser window with the title 'VratiProizvode - WebAppFiltriran'. The address bar shows 'localhost:5051/Home/VratiProi...'. The main content area features a blue header with the text 'Slatkisi i grickalice'. Below this is a table with two columns: 'Naziv' and 'Cena'. The table lists five products with their respective prices. At the bottom of the table, there is a link labeled 'Prikazi kategorije'.

Naziv	Cena
Krem Nutella cream 400g	344.99
Krem tabla Euro blok Eurocrem 50g	49.99
Cokolada Milka haselnuss 80g	109.99
Keks Plazma 300g	178.99
Karamela lesnik 200g	152.99

[Prikazi kategorije](#)

Pitanje 1

MVC web aplikacija se izvršava na adresi `http://localhost:5000`. Ako želimo da pozovemo **Details** akcionu metodu **OsobaController** klase i da toj metodi prosledimo vrednost 3 kao ulazni parametar ove metode, potrebno je uneti sledeću adresu u browseru:

- a. `http://localhost:5000/Osoba/Details/3`
- b. `http://localhost:5000/OsobaController/details/3`
- c. `http://localhost:5000/Osoba/details/?=3`

Odgovor: a

Pitanje 2

Komponenta ASP.NET MVC web aplikacije koja je odgovorna za obradu zahteva naziva se:

- a. model
- b. pogled
- c. kontroler

Odgovor: c

Pitanje 3

Zaokruži netačno tvrđenje. Kontrolerska klasa:

- a. Ima ime koje se završava sa rečju Controller
- b. Može biti izvedena iz klase Controller
- c. Mora biti opisana atributskom klasom [Authorize]

Odgovor: c

Pitanje 4

Interfejs pomoću koga se specificira povratni tip akcione metode kontrolera je :

- a. IDbContext
- b. IActionResult
- c. IJson

Odgovor: b

Pitanje 5

Ako akciona metoda kontrolera nije opisana atributskom klasom onda se ona može pozvati:

- a. GET zahtev
- b. POST zahtev
- c. I GET i POST zahtevom

Odgovor: c

Pitanje 6

Proces kreiranja .NET objekata na osnovu parametara HTTP zahteva u metodama kontrolera naziva se

- a. Model binding
- b. Controller binding
- c. Pogled binding

Odgovor: a

Pitanje 7

Validnost podataka koji su prosleđeni do akcione metode kontrolera proverava se korišćenjem izraza

- a. `ModeBinding.IsValid`
- b. `ModelState.IsValid`
- c. `Model.IsValid`

Odgovor: b