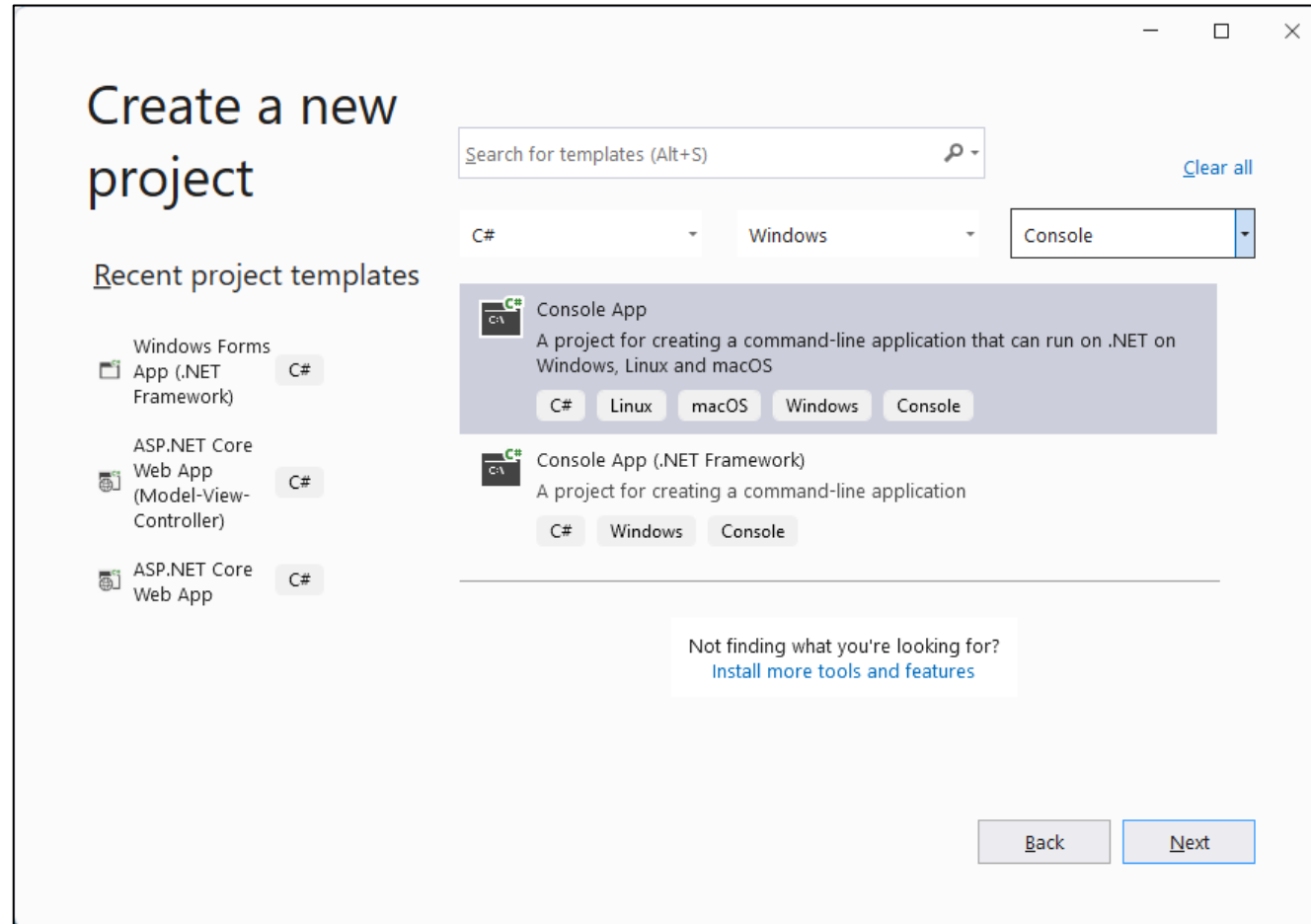


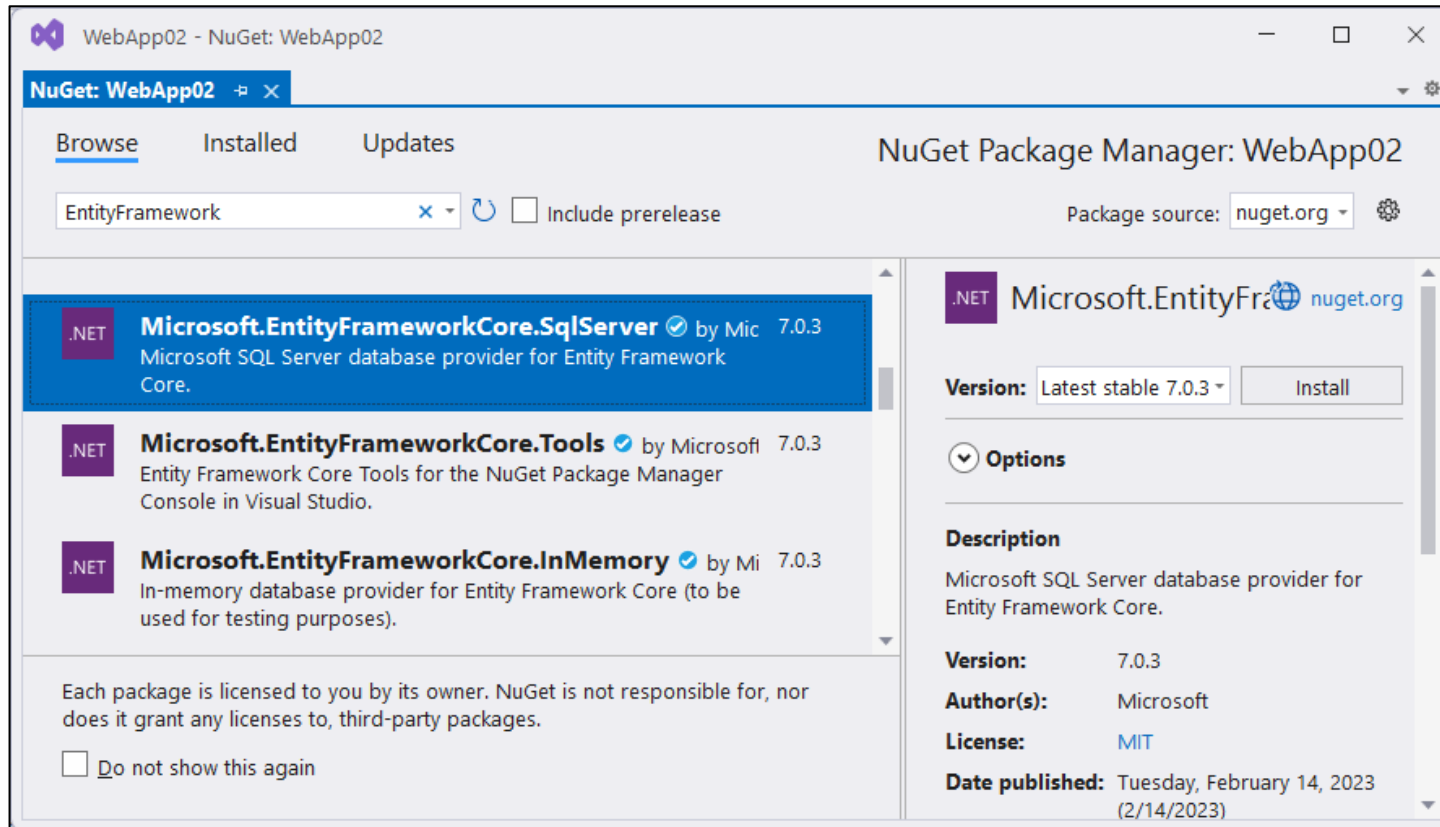
Entity Framework Core

.NET Core konzolna aplikacija



Microsoft.EntityFrameworkCore.SqlServer

- Project->Manage Nuget Packages



EF Core Power Tools

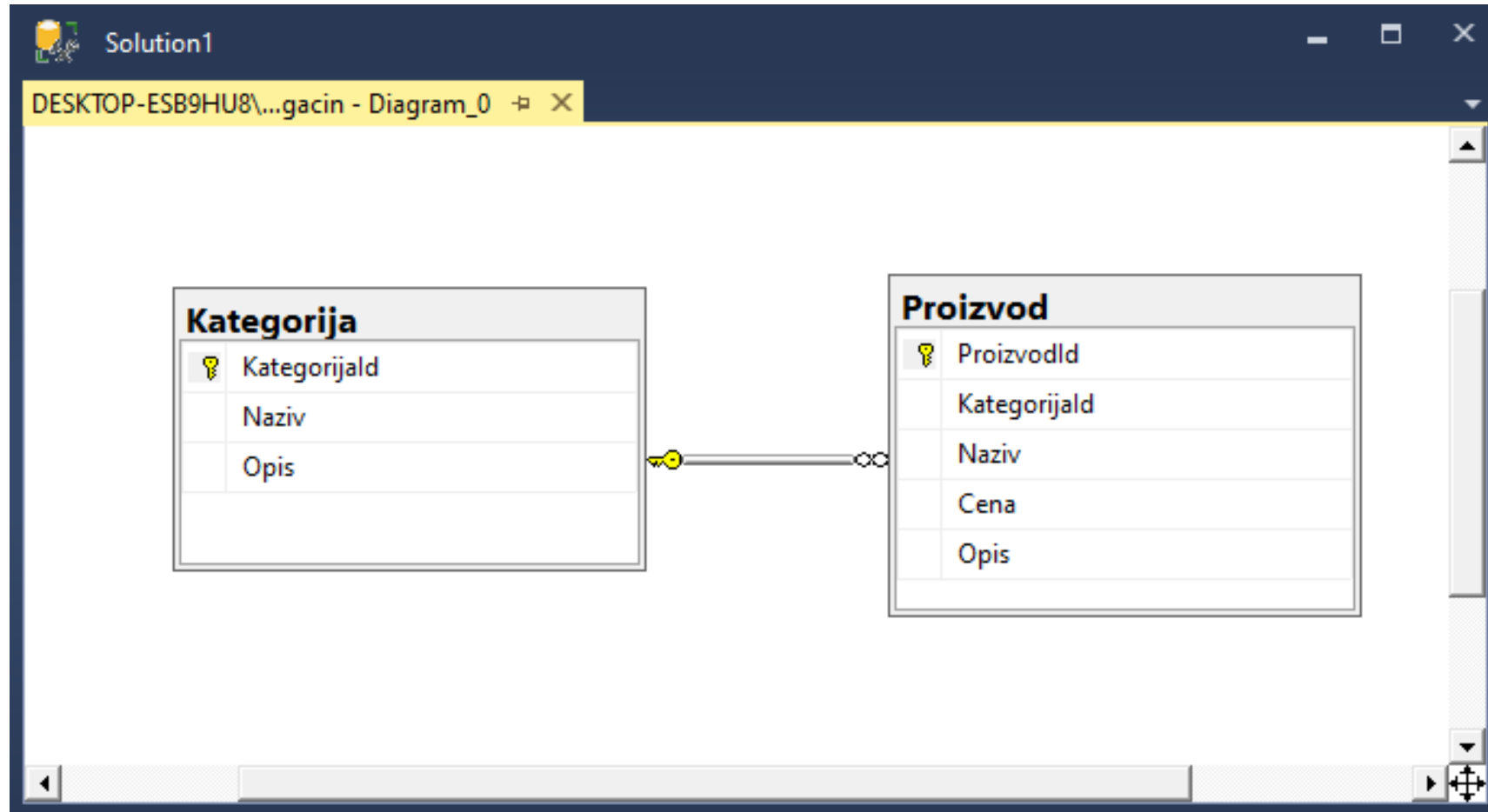
- Extensios -> Manage Extensios

The screenshot shows the 'Manage Extensions' window in Visual Studio. The search bar contains 'ef core'. The results list includes:

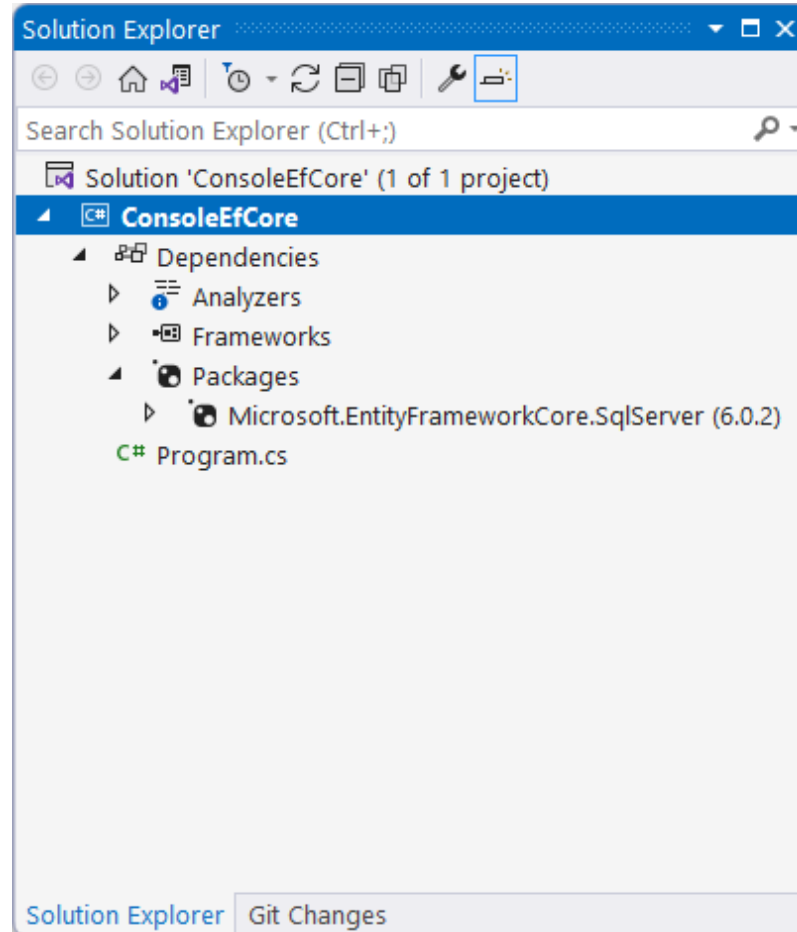
- EF Core Power Tools** (highlighted): Useful design-time DbContext features, added to the Visual Studio Solution Explorer context menu. Created By: ErikEJ, Version: 2.5.1296, Installs: 285135, Pricing Category: Free, Rating: ★★★★★ (413 Votes).
- EF Core Power Pack**: A package that contains EF Core Power Tools and some DDEX providers that improve your experience when using EF Core...
- Angular 14/ASP.NET Core 6 Project Template**: Free ASP.NET Core 6 / Angular 14 startup project template with complete login, user and role management. Plus other u...
- ASP.NET AJAX, MVC, ASP.NET Core and Bootstrap...**: Over 110 ASP.NET WebForms Controls. Over 70 MVC Extensions. Over 40 ASP.NET Bootstrap Controls. Includes AS... Trial
- EntityFramework Core Scaffolding**: A VS tool to run the `dotnet ef dbcontext scaffold` command.

At the bottom, a yellow notification bar states: 'Some extensions were automatically updated. Updates will be applied the next time you start Visual Studio.' A 'Close' button is visible next to the notification.

Baza Magacin



Solution Explorer



Desni klik na projekat pa opcija Edit

```
<Project Sdk="Microsoft.NET.Sdk">

  <PropertyGroup>
    <OutputType>Exe</OutputType>
    <TargetFramework>net7.0</TargetFramework>
    <ImplicitUsings>enable</ImplicitUsings>
    <Nullable>enable</Nullable>
  </PropertyGroup>

  <ItemGroup>
    <PackageReference Include="Microsoft.EntityFrameworkCore.SqlServer" Version="7.0.3" />
  </ItemGroup>

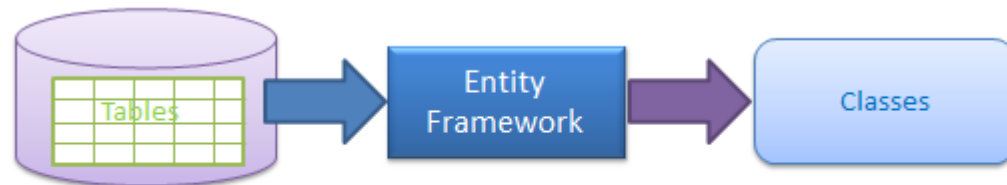
</Project>
```

Entity Framework Core

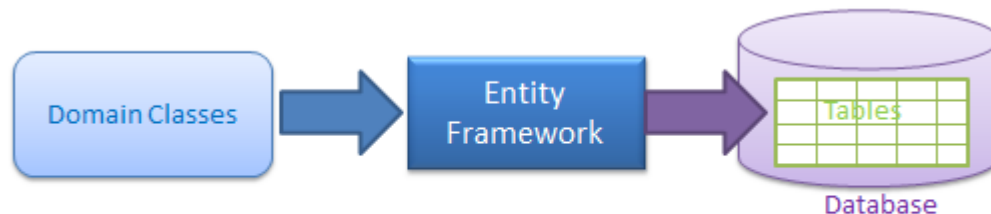
- EF Core je međuplatformska verzija Entity Frameworka
- EF Core je objektno-relacioni mapper (O/RM) koji omogućava .NET programerima da rade sa bazom korišćenjem .NET objekata.

Strategije dizajna modela

- **Database-first design**, najpre se dizajnira i kreira baza podataka a zatim se generišu entiteti aplikacije
- **Model-first design**, najpre se dizajniraju entiteti za aplikaciju a zatim se kreira baza podataka na osnovu tih entiteta

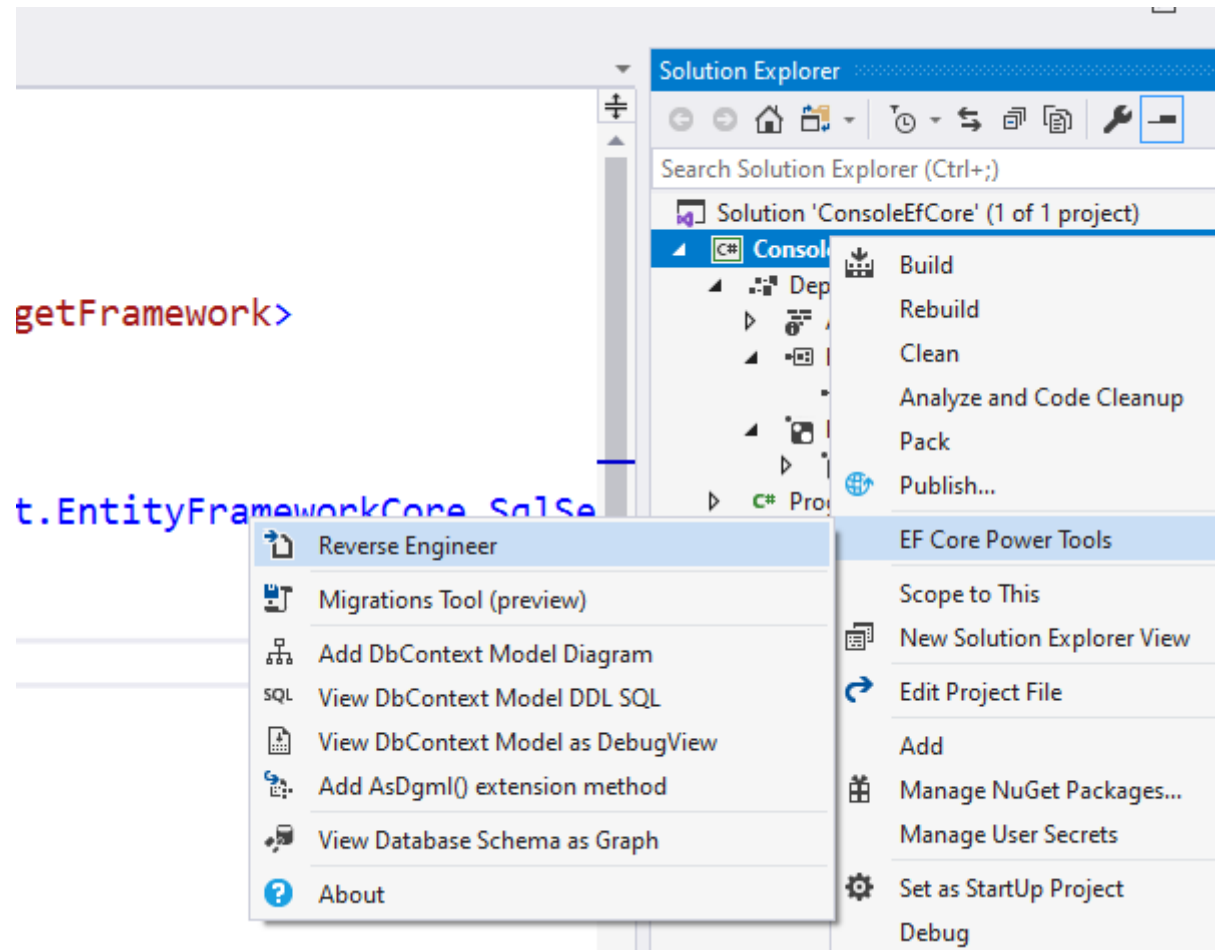


Generate Data Access Classes for Existing Database



Create Database from the Domain Classes

Pokreni EF Core Power Tools



Odabir konekcije

The 'Connection Properties' dialog box is shown with the following fields and options:

- Data source:** Microsoft SQL Server (SqlClient) with a 'Change...' button.
- Server name:** GORAN-HP with a 'Refresh' button.
- Log on to the server:**
 - Authentication:** Windows Authentication
 - User name:** (empty text box)
 - Password:** (empty text box)
 - Save my password
- Connect to a database:**
 - Select or enter a database name: Magacin
 - Attach a database file: (empty text box) with a 'Browse...' button
 - Logical name: (empty text box)
- Buttons:** Test Connection, OK, Cancel, and an 'Advanced...' button.

The 'Choose Database Connection' dialog box is shown with the following fields and options:

- Database name:** GORAN-HP.Magacin (highlighted with a dashed border) with a minus sign button and an 'Add...' button.
- Title:** Choose SQL Server Database project (.dacpac)
- Buttons:** Add... (greyed out)
- Filter schemas
- Buttons:** Add... (greyed out)
- Buttons:** EF Core 7 (dropdown), 2.5.1296 (text), OK, and Cancel.

Odabir tabela i kreiranje modela

Generate EF Core Model in Project ConsoleEfCore

Context name:

Namespace:

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

EntityTypes sub-namespace (overrides path) - optional:

DbContext path (f.ex. Data) - optional:

DbContext sub-namespace (overrides path) - optional:

What to generate:

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:

Include connection string in generated code

Install the EF Core provider package in the project

Choose Database Objects

Search:

- Tables
 - dbo
 - Kategorija
 - Proizvod

Entitetska klasa Kategorija

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

        [Key]
        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; }
    }
}
```

Entitetska klasa Proizvod

```
public partial class Proizvod
{
    [Key]
    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; }

    [ForeignKey(nameof(KategorijaId))]
    public virtual Kategorija Kategorija { get; set; }
}
```

DbContext klasa

```
public partial class MagacinContext : DbContext
{
    public MagacinContext()
    {
    }

    public MagacinContext(DbContextOptions<MagacinContext> options)
        : base(options)
    {
    }

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

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

    protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
        => optionsBuilder.UseSqlServer("Data Source=GORAN-HP;Initial Catalog=Magacin;Integrated Security=True;Encrypt=False");
}
```

DbContext klasa

- DbContext klasa je most za interakciju entitetskih klasa i baze podataka
- DbSet <TEntity> predstavlja kolekciju entitetskih objekata koji odgovaraju redovima tabele u bazi podataka
- DbContext sadrži svojstva za pristup DbSet-ovima **DbSet<TEntity>** za sve entitetske klase koje su mapirane u tabele baze
- DbContext klasa:
 - upravlja konekcijom sa odgovarajućom bazom podataka
 - prati promene u objektima unutar DbSet ova
 - čuva promene iz DbSetova u bazi podataka

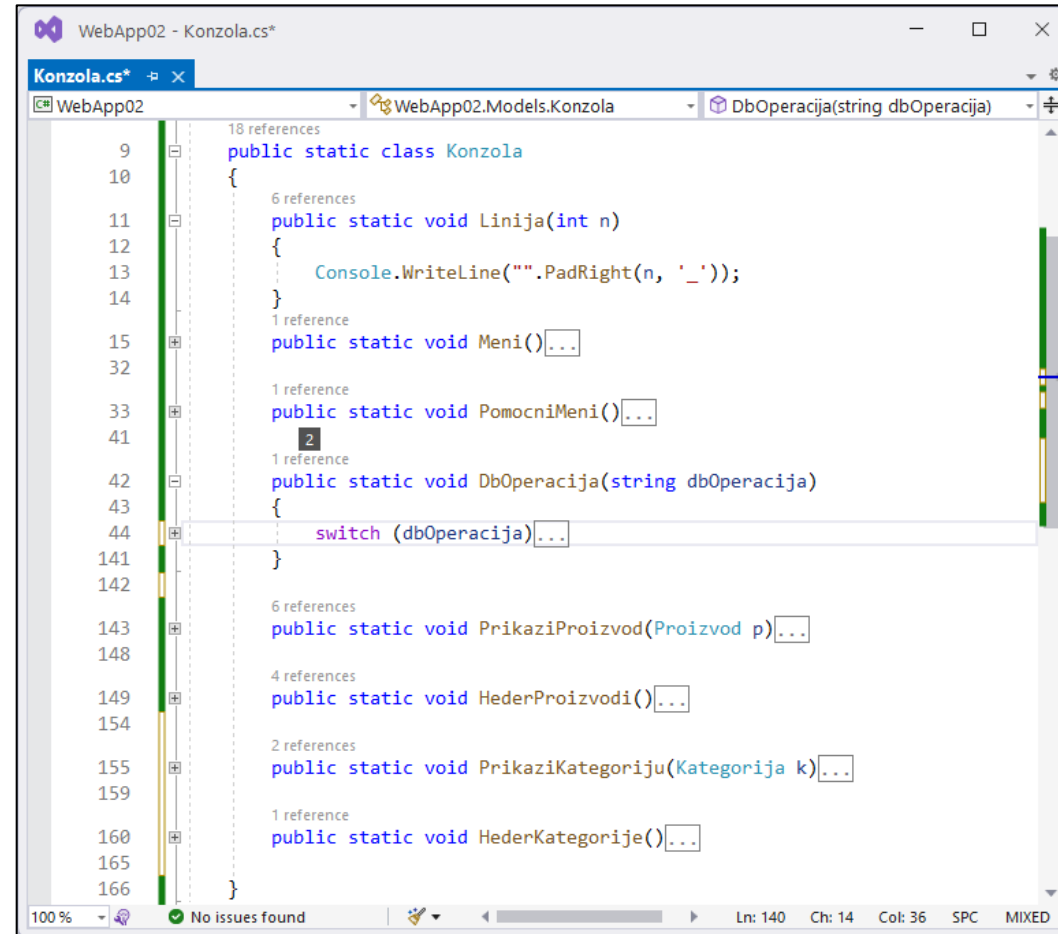
Klasa za komunikaciju sa bazom MagacinDal

```
using Microsoft.EntityFrameworkCore;

namespace WebApp02.Models{
    public static class MagacinDal
    {

    }
}
```

Klasa za komunikaciju sa konzolom - Konzola



```
WebApp02 - Konzola.cs*
Konzola.cs*
WebApp02
WebApp02.Models.Konzola
DbOperacija(string dbOperacija)
18 references
9 public static class Konzola
10 {
11     6 references
12     public static void Linija(int n)
13     {
14         Console.WriteLine("").PadRight(n, '_');
15     }
16     1 reference
17     public static void Meni()...
18
19     1 reference
20     public static void PomocniMeni()...
21
22     1 reference
23     public static void DbOperacija(string dbOperacija)
24     {
25         switch (dbOperacija)...
26     }
27
28     6 references
29     public static void PrikaziProizvod(Proizvod p)...
30
31     4 references
32     public static void HederProizvodi()...
33
34     2 references
35     public static void PrikaziKategoriju(Kategorija k)...
36
37     1 reference
38     public static void HederKategorije()...
39 }
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
100% No issues found Ln: 140 Ch: 14 Col: 36 SPC MIXED
```

Klasa Konzola metoda DbOperacija

```
public static void DbOperacija(string dbOperacija)
{
    switch (dbOperacija)
    {
        case "1":
            //Prikazi proizvode sa odlozenim izvravanjem upita
            MagacinDal.PrikaziProizvode1();
            break;
        case "2":
            //Prikazi proizvode sa materijalizacijom upita
            MagacinDal.PrikaziProizvode2();
            break;
        ...
        default:
            //ne postoji Db operacija
            Console.WriteLine("Ne postojeća Db operacija");
            break;
    }
}
```

Fajl Program.cs

```
using WebApp02.Models;

string novaDbOperacija = "" ;
string dbOperacija;

do
{
    Konzola.Meni();
    Console.WriteLine("Odaberite DB operaciju:");
    dbOperacija = Console.ReadLine();
    Konzola.DbOperacija(dbOperacija);
    if (dbOperacija != "12")
    {
        Konzola.PomocniMeni();
        novaDbOperacija = Console.ReadLine();
    }
    else
    {
        novaDbOperacija = "n";
    }
} while (novaDbOperacija == "d") ;
```

Početni prozor aplikacije

```
C:\Users\goran\source\rep x + v - □ ×  
-----  
1 -> Prikazi proizvode sa odlozenim izvršavanjem upita  
2 -> Prikazi proizvode sa materijalizacijom upita  
3 -> Prikazi kategorije proizvoda  
4 -> Proizvodi iz kategorije  
5 -> Proizvodi iz kategorije - navigaciona svojstva  
6 -> Prikaz proizvoda na osnovu id -a SingleOrDefault  
7 -> Prikaz proizvoda na osnovu id -a Find  
8 -> Prikaz kategorije na osnovu id -a Find  
9 -> Unos nove kategorije  
10 -> Promena postojece kategorije  
11 -> Brisanje kategorije  
12 -> izađi  
-----  
Odaberite DB operaciju:  
|
```

Čitanje podataka sa odloženim izvršavanjem upita

```
public static void PrikaziProizvode1()
{
    using (MagacinContext db = new MagacinContext())
    {
        DbSet<Proizvod> listaProizvoda = db.Proizvodi;

        Konzola.HederProizvodi();
        foreach (Proizvod p in listaProizvoda)
        {
            Konzola.PrikaziProizvod(p);
        }
    }
}
```

Metoda PrikaziProizvode1(), klasa MagacinDal

Prikaz proizvoda

```
C:\Users\goran\source\rep x + v
Odaberite DB operaciju:
1
ID| Naziv | Cena
-----|-----|-----
1 | Coca-Cola 1l PET | 79.99
2 | Sok pomorandza Happy day 1l | 154.99
3 | Kafa mlevena Grand Gold 200g | 239.99
4 | Sok pomorandza Next Classic 1l | 124.99
5 | Fanta limenka 0,33l | 50.00
6 | Jogurt Balans +probiotik 1.5kg Pet | 152.00
7 | Mleko ster.2.8%mm Moja kravica BPslim 1L | 99.99
8 | Sir President Somborska 500g | 259.99
9 | Kiselo mleko 2.8%mm Moja kravica 180g | 24.99
10 | Dukat SenSia 1kg | 113.99
11 | Krem Nutella cream 400g | 344.99
12 | Krem tabla Euro blok Eurocrem 50g | 49.99
13 | Cokolada Milka haselnuss 80g | 109.99
14 | Keks Plazma 300g | 178.99
15 | Karamela lesnik 200g | 152.99
16 | Beskvasni integralni hleb Maxi 500g | 99.99
17 | Somun svezi Tvojih 5 minuta 190g | 39.99
18 | Strudla mak Kikindska pekara 110g | 99.99
19 | Tost tamni Hleb&Kifle 500g | 144.99
20 | Integralni dvopek Tvojih 5 minuta 210g | 111.99
-----|-----|-----
Da li zelite novu db operaciju?
Da -> Pritisni taster d
Ne -> Pritisni taster n
|
```

Čitanje podataka uz materijalizaciju upita

```
public static void PrikaziProizvode2()
{
    using (MagacinContext db = new MagacinContext())
    {
        List<Proizvod> listaProizvoda = db.Proizvodi.ToList();
        Konzola.HederProizvodi();
        foreach (Proizvod p in listaProizvoda)
        {
            Konzola.PrikaziProizvod(p);
        }
    }
}
```


Prikaz kategorija proizvoda

```
public static void PrikaziKategorije()
{
    using (MagacinContext db = new MagacinContext())
    {

        DbSet<Kategorija> listaKategorija = db.Kategorije;
        Konzola.HederKategorije();

        foreach (Kategorija k in listaKategorija)
        {
            Konzola.PrikaziKategoriju(k);
        }
    }
}
```

Prikaz kategorija

```
C:\Users\goran\source\rep x + v
4 -> Proizvodi iz kategorije
5 -> Proizvodi iz kategorije - navigaciona svojstva
6 -> Prikaz proizvoda na osnovu id -a SingleOrDefault
7 -> Prikaz proizvoda na osnovu id -a Find
8 -> Prikaz kategorije na osnovu id -a Find
9 -> Unos nove kategorije
10 -> Promena postojece kategorije
11 -> Brisanje kategorije
12 -> izađi

-----
Odaberite DB operaciju:
3
ID|      Naziv
-----
1   Sokovi
2   Mleko i mlecni proizvodi
3   Slatkisi i grickalice
4   Hleb i peciva
7   Test1
8   Test2
-----
Da li zelite novu db operaciju?
Da -> Pritisni taster d
Ne -> Pritisni taster n
-----
|
```

Filtriranje podataka

```
public static void ProizvodiIzKategorije1(int id)
{
    using (MagacinContext db = new MagacinContext())
    {
        IQueryable<Proizvod> listaProizvoda = db.Proizvodi
            .Where(p => p.KategorijaId == id);
        Konzola.HederProizvodi();
        foreach (Proizvod p in listaProizvoda)
        {
            Konzola.PrikaziProizvod(p);
        }
    }
}
```

Filtriranje podataka

```
case "4":  
    //Proizvodi iz kategorije  
    Console.WriteLine("Unesite id kategorije proizvoda (1 do 4)");  
    int id = int.Parse(Console.ReadLine());  
    MagacinDal.ProizvodiIzKategorije1(id);  
    break;  
  
case "5":  
    //Proizvodi iz kategorije - navigaciona svojstva  
    Console.WriteLine("Unesite id kategorije proizvoda (1 do 4)");  
    id = int.Parse(Console.ReadLine());  
    MagacinDal.ProizvodiIzKategorije2(id);  
    break;
```

Prikaz proizvoda iz kategorije

```
C:\Users\goran\source\rep x + v
3 -> Prikazi kategorije proizvoda
4 -> Proizvodi iz kategorije
5 -> Proizvodi iz kategorije - navigaciona svojstva
6 -> Prikaz proizvoda na osnovu id -a SingleOrDefault
7 -> Prikaz proizvoda na osnovu id -a Find
8 -> Prikaz kategorije na osnovu id -a Find
9 -> Unos nove kategorije
10 -> Promena postojece kategorije
11 -> Brisanje kategorije
12 -> izađi

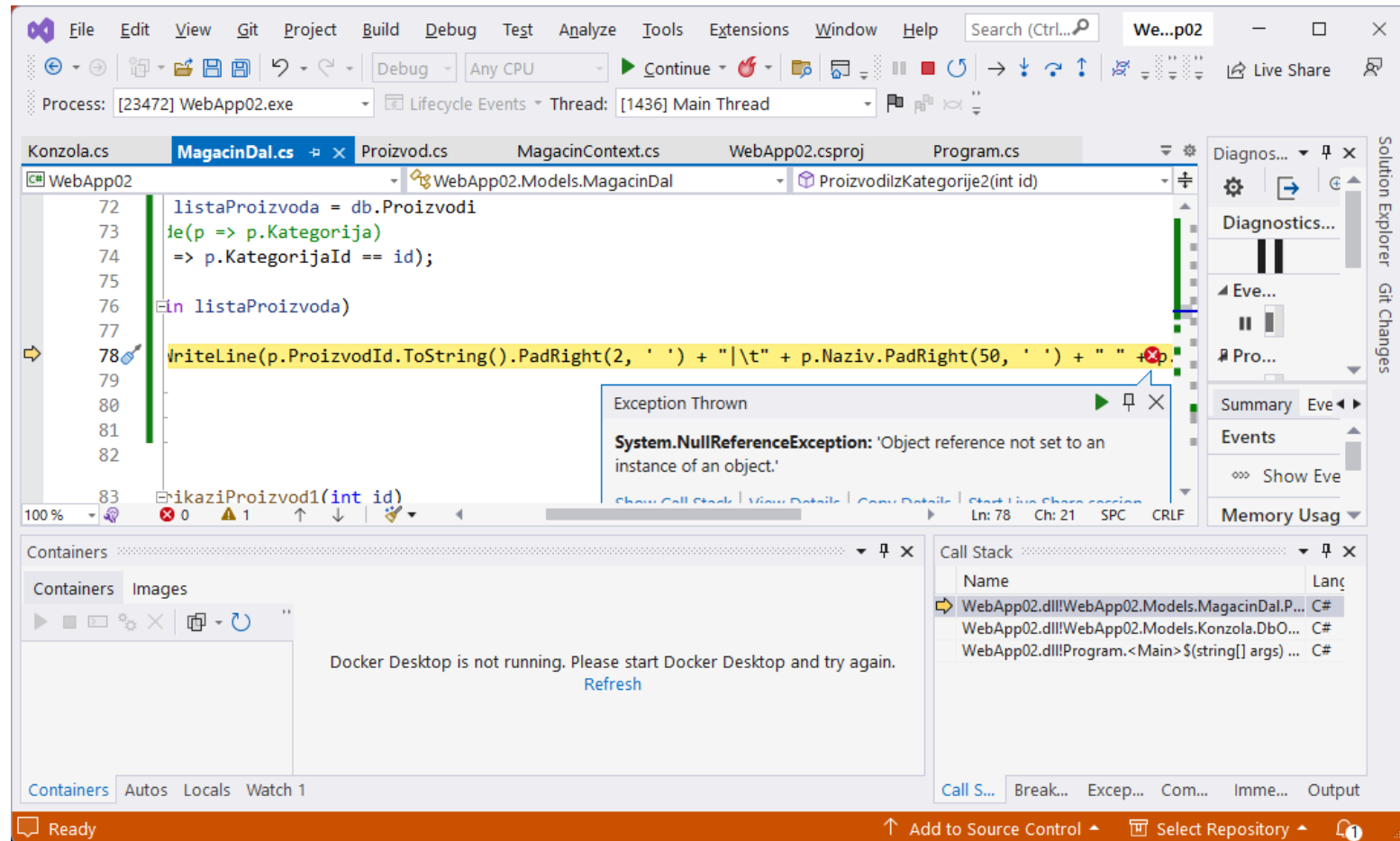
-----
Odaberite DB operaciju:
4
Unesite id kategorije proizvoda (1 do 4)
2
ID| Naziv | Cena
-----
6 | Jogurt Balans +probiotik 1.5kg Pet | 152.00
7 | Mleko ster.2.8%mm Moja kravica BPslim 1L | 99.99
8 | Sir President Somborska 500g | 259.99
9 | Kiselo mleko 2.8%mm Moja kravica 180g | 24.99
10 | Dukat SenSia 1kg | 113.99
-----
Da li zelite novu db operaciju?
Da -> Pritisni taster d
Ne -> Pritisni taster n
-----
```

Upotreba navigacionog svojstva

```
public static void ProizvodiIzKategorije2(int id)
{
    using (MagacinContext db = new MagacinContext())
    {
        IQueryable<Proizvod> listaProizvoda = db.Proizvodi
            .Where(p => p.KategorijaId == id);

        foreach (Proizvod p in listaProizvoda)
        {
            Console.WriteLine(p.ProizvodId.ToString().PadRight(2, ' ') +
                "\t" + p.Naziv.PadRight(50, ' ') + " " + p.Kategorija.Naziv);
        }
    }
}
```

Upotreba navigacionog svojstva



Prikaz podataka iz povezane tabele – Include()

```
public static void ProizvodiIzKategorije2(int id)
{
    using (MagacinContext db = new MagacinContext())
    {
        IQueryable<Proizvod> listaProizvoda = db.Proizvodi
            .Include(p => p.Kategorija)
            .Where(p => p.KategorijaId == id);

        foreach (Proizvod p in listaProizvoda)
        {
            Console.WriteLine(p.ProizvodId.ToString().PadRight(2, ' ') +
                "\t" + p.Naziv.PadRight(50, ' ') + " " + p.Kategorija.Naziv);
        }
    }
}
```


Prikaz podataka iz povezane tabele – Include()

```
C:\Users\goran\source\rep x + v
-----
1 -> Prikazi proizvode sa odlozenim izvravanjem upita
2 -> Prikazi proizvode sa materijalizacijom upita
3 -> Prikazi kategorije proizvoda
4 -> Proizvodi iz kategorije
5 -> Proizvodi iz kategorije - navigaciona svojstva
6 -> Prikaz proizvoda na osnovu id -a SingleOrDefault
7 -> Prikaz proizvoda na osnovu id -a Find
8 -> Prikaz kategorije na osnovu id -a Find
9 -> Unos nove kategorije
10 -> Promena postojece kategorije
11 -> Brisanje kategorije
12 -> izadji
-----
Odaberite DB operaciju:
5
Unesite id kategorije proizvoda (1 do 4)
2
6 |      Jogurt Balans +probiotik 1.5kg Pet      Mleko i mlecni proizvodi
7 |      Mleko ster.2.8%mm Moja kravica BPslim 1L  Mleko i mlecni proizvodi
8 |      Sir President Somborska 500g            Mleko i mlecni proizvodi
9 |      Kiselo mleko 2.8%mm Moja kravica 180g    Mleko i mlecni proizvodi
10|      Dukat SenSia 1kg                        Mleko i mlecni proizvodi
-----
Da li zelite novu db operaciju?
Da -> Pritisni taster d
Ne -> Pritisni taster n
-----
```

Čitanje jednog reda

```
public static void PrikaziProizvod1(int id)
{
    using (MagacinContext db = new MagacinContext())
    {
        Proizvod p1 = db.Proizvodi
            .SingleOrDefault(p => p.ProizvodId == id);
        if (p1 != null)
        {
            Konzola.PrikaziProizvod(p1);
        }
        else
        {
            Console.WriteLine($"Ne postoji proizvod ciji je id = {id}");
        }
    }
}
```

Čitanje jednog reda

```
public static void PrikaziProizvod2(int id)
{
    using (MagacinContext db = new MagacinContext())
    {
        Proizvod p1 = db.Proizvodi.Find(id);

        if (p1 != null)
        {
            Konzola.PrikaziProizvod(p1);
        }
        else
        {
            Console.WriteLine($"Ne postoji proizvod ciji je id = {id}");
        }
    }
}
```

Čitanje jednog reda

```
case "6":
    //Prikaz proizvoda na osnovu id - a SingleOrDefault
    Console.WriteLine("Unesite id proizvoda (1 do 20)");
    id = int.Parse(Console.ReadLine());
    MagacinDal.PrikaziProizvod1(id);
    break;

case "7":
    //Prikaz proizvoda na osnovu id -a Find
    Console.WriteLine("Unesite id proizvoda (1 do 20)");
    id = int.Parse(Console.ReadLine());
    MagacinDal.PrikaziProizvod2(id);
    break;
```

Čitanje jednog proizvoda

```
C:\Users\goran\source\rep x + v
1 -> Prikazi proizvode sa odlozenim izvravanjem upita
2 -> Prikazi proizvode sa materijalizacijom upita
3 -> Prikazi kategorije proizvoda
4 -> Proizvodi iz kategorije
5 -> Proizvodi iz kategorije - navigaciona svojstva
6 -> Prikaz proizvoda na osnovu id -a SingleOrDefault
7 -> Prikaz proizvoda na osnovu id -a Find
8 -> Prikaz kategorije na osnovu id -a Find
9 -> Unos nove kategorije
10 -> Promena postojece kategorije
11 -> Brisanje kategorije
12 -> izadji

-----
Odaberite DB operaciju:
6
Unesite id proizvoda (1 do 20)
12
12|      Krem tabla Euro blok Eurocrem 50g      |      49.99
-----
Da li zelite novu db operaciju?
Da -> Pritisni taster d
Ne -> Pritisni taster n
|
```

Čítanje jedne kategorije

```
public static void PrikaziKategoriju(int id)
{
    using (MagacinContext db = new MagacinContext())
    {
        Kategorija k1 = db.Kategorije.Find(id);

        if (k1 != null)
        {
            Konzola.PrikaziKategoriju(k1);
        }
        else
        {
            Console.WriteLine("Ne postoji kategorija");
        }
    }
}
```

Čítanje jedne kategorije

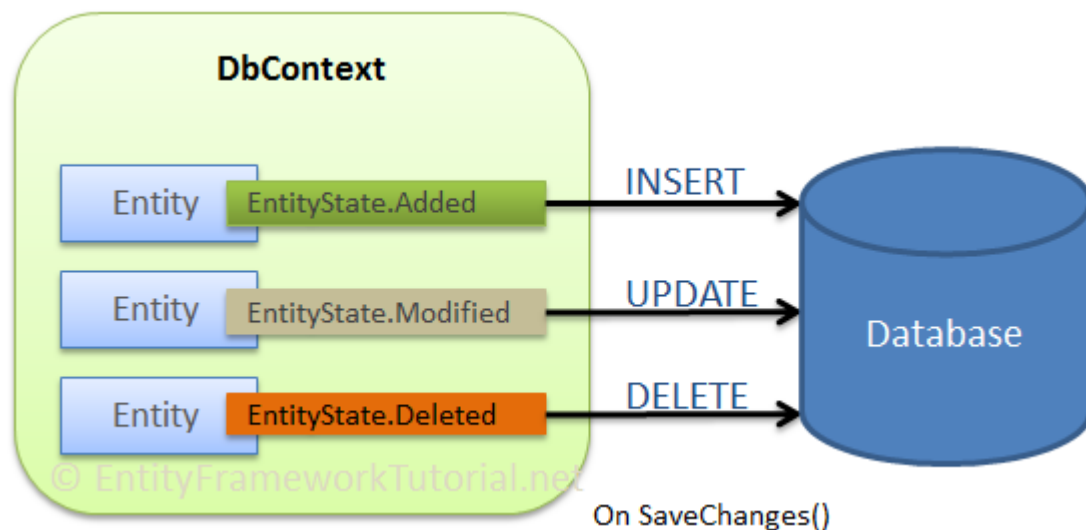
```
case "8":  
    //Prikaz kategorije na osnovu id -a Find  
    Console.WriteLine("Unesite id kategorije (1 do 4)");  
    id = int.Parse(Console.ReadLine());  
    MagacinDal.PrikaziKategoriju(id);  
    break;
```

Čitanje jedne kategorije

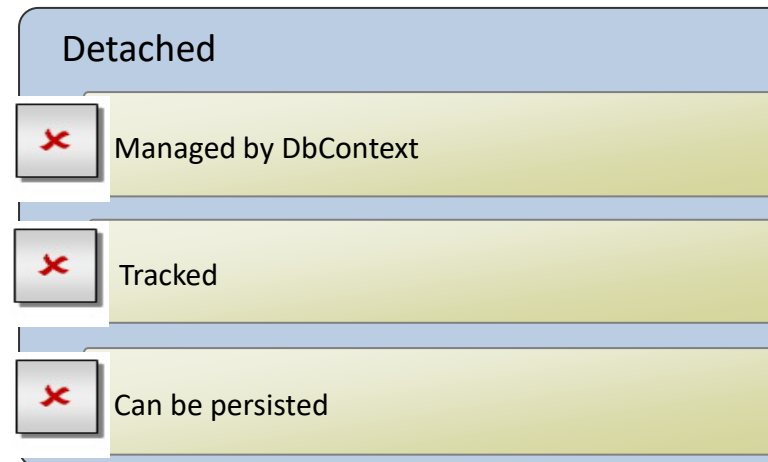
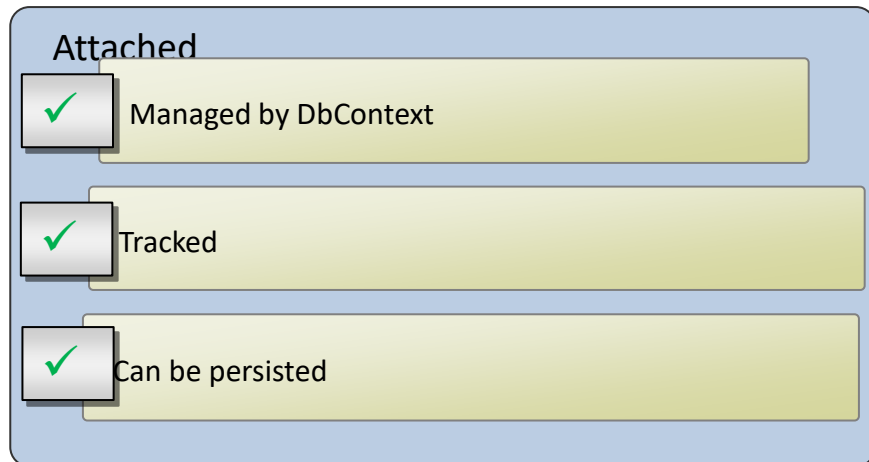
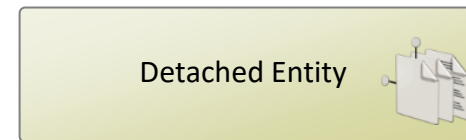
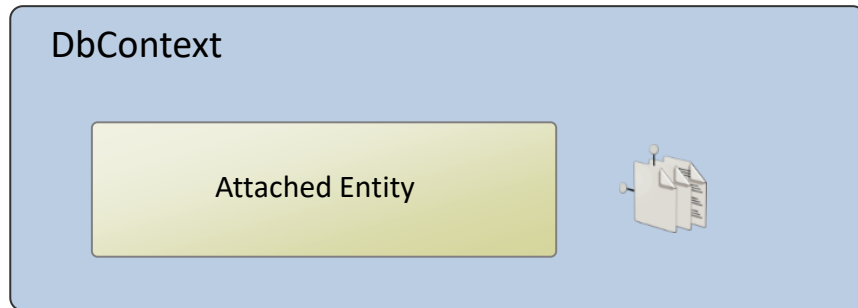
```
C:\Users\goran\source\rep x + v
-----
1 -> Prikazi proizvode sa odlozenim izvršavanjem upita
2 -> Prikazi proizvode sa materijalizacijom upita
3 -> Prikazi kategorije proizvoda
4 -> Proizvodi iz kategorije
5 -> Proizvodi iz kategorije - navigaciona svojstva
6 -> Prikaz proizvoda na osnovu id -a SingleOrDefault
7 -> Prikaz proizvoda na osnovu id -a Find
8 -> Prikaz kategorije na osnovu id -a Find
9 -> Unos nove kategorije
10 -> Promena postojeće kategorije
11 -> Brisanje kategorije
12 -> izađi
-----
Odaberite DB operaciju:
8
Unesite id kategorije (1 do 4)
2
2      Mleko i mlečni proizvodi
-----
Da li želite novu db operaciju?
Da -> Pritisni taster d
Ne -> Pritisni taster n
-----
|
```


Ažuriranje podataka korišćenjem EF

- DbContext objekat sadrži entity objekte koji predstavljaju podatke u odgovarajućoj bazi podataka
- Ukoliko želimo da modifikujemo podatke u bazi prvo se vrši promena sadržaja DbContext objekta a zatim se od njega traži da sačuva promene u bazi podataka



Atačovani i detačovani objekti



Unos podataka

```
public static int UbaciKategoriju(Kategorija k)
{
    using (MagacinContext db = new MagacinContext())
    {
        try
        {
            db.Add(k);
            db.SaveChanges();
            return k.KategorijaId;
        }
        catch (Exception xcp)
        {
            Console.WriteLine(xcp.Message);
            return -1;
        }
    }
}
```

Unos nove kategorije

```
case "9":  
    //Unos nove kategorije  
    Console.WriteLine("Unesite naziv nove kategorije");  
    string naziv = Console.ReadLine();  
    Console.WriteLine("Unesite opis nove kategorije");  
    string opis = Console.ReadLine();  
    Kategorija k1 = new Kategorija { Naziv = naziv, Opis = opis };  
    id = MagacinDal.UbaciKategoriju(k1);  
    Console.WriteLine("Ubacena kategorija ciji je id: " + id);  
    break;
```

Unos nove kategorije

```
C:\Users\goran\source\rep x + v
-----
1 -> Prikazi proizvode sa odlozenim izvravanjem upita
2 -> Prikazi proizvode sa materijalizacijom upita
3 -> Prikazi kategorije proizvoda
4 -> Proizvodi iz kategorije
5 -> Proizvodi iz kategorije - navigaciona svojstva
6 -> Prikaz proizvoda na osnovu id -a SingleOrDefault
7 -> Prikaz proizvoda na osnovu id -a Find
8 -> Prikaz kategorije na osnovu id -a Find
9 -> Unos nove kategorije
10 -> Promena postojece kategorije
11 -> Brisanje kategorije
12 -> izardji
-----
Odaberite DB operaciju:
9
Unesite naziv nove kategorije
Test3
Unesite opis nove kategorije
Opis3
Ubacena kategorija ciji je id: 9
-----
Da li zelite novu db operaciju?
Da -> Pritisni taster d
Ne -> Pritisni taster n
-----
|
```

Prikaz unetog reda

```
C:\Users\goran\source\rep x + v
-----
1 -> Prikazi proizvode sa odlozenim izvravanjem upita
2 -> Prikazi proizvode sa materijalizacijom upita
3 -> Prikazi kategorije proizvoda
4 -> Proizvodi iz kategorije
5 -> Proizvodi iz kategorije - navigaciona svojstva
6 -> Prikaz proizvoda na osnovu id -a SingleOrDefault
7 -> Prikaz proizvoda na osnovu id -a Find
8 -> Prikaz kategorije na osnovu id -a Find
9 -> Unos nove kategorije
10 -> Promena postojece kategorije
11 -> Brisanje kategorije
12 -> izardji
-----
Odaberite DB operaciju:
8
Unesite id kategorije (1 do 4)
9
9 Test3
-----
Da li zelite novu db operaciju?
Da -> Pritisni taster d
Ne -> Pritisni taster n
-----
|
```

Metoda VratiKategoriju()

```
public static Kategorija? VratiKategoriju(int id)
{
    using (MagacinContext db = new MagacinContext())
    {
        Kategorija k1 = db.Kategorije.Find(id);
        return k1;
    }
}
```

Ažuriranje reda

```
public static int PromeniKategoriju(Kategorija k)
{
    using (MagacinContext db = new MagacinContext())
    {
        try
        {
            db.Update(k);
            db.SaveChanges();
            return 0;
        }
        catch (Exception )
        {
            return -1;
        }
    }
}
```


Promena postojeće kategorije

```
case "10":  
    //Promena postojece kategorije  
    Console.WriteLine("Unesite id kategorije koju menjate");  
    id = int.Parse(Console.ReadLine());  
    MagacinDal.PrikaziKategoriju(id);  
    k1 = MagacinDal.VratiKategoriju(id);  
    int r = 0;  
    if (k1 != null)  
    {  
        Console.WriteLine("Unesite novi naziv kategorije");  
        naziv = Console.ReadLine();  
        Console.WriteLine("Unesite novi opis kategorije");  
        opis = Console.ReadLine();  
        k1.Naziv = naziv;  
        k1.Opis = opis;  
        r = MagacinDal.PromeniKategoriju(k1);  
        if (r == 0)  
        {  
            Console.WriteLine("Promenjena kategorija");  
        }  
        else  
        {  
            Console.WriteLine("Greska pri promeni");  
        }  
    }  
  
    break;
```

Promena postojeće kategorije

```
C:\Users\goran\source\rep. x + v
-----
1 -> Prikazi proizvode sa odlozenim izvrsavanjem upita
2 -> Prikazi proizvode sa materijalizacijom upita
3 -> Prikazi kategorije proizvoda
4 -> Proizvodi iz kategorije
5 -> Proizvodi iz kategorije - navigaciona svojstva
6 -> Prikaz proizvoda na osnovu id -a SingleOrDefault
7 -> Prikaz proizvoda na osnovu id -a Find
8 -> Prikaz kategorije na osnovu id -a Find
9 -> Unos nove kategorije
10 -> Promena postojece kategorije
11 -> Brisanje kategorije
12 -> izardji
-----
Odaberite DB operaciju:
10
Unesite id kategorije koju menjate
9
          Test3
Unesite novi naziv kategorije
Test33
Unesite novi opis kategorije
Opis33
Promenjena kategorija
-----
Da li zelite novu db operaciju?
Da -> Pritisni taster d
Ne -> Pritisni taster n
-----
```

Brisanje reda

```
public static int ObrisiKategoriju(int id)
{
    using (MagacinContext db = new MagacinContext())
    {
        try
        {
            Kategorija k = db.Kategorije.Find(id);
            db.Kategorije.Remove(k);
            db.SaveChanges();
            return 0;
        }
        catch (Exception)
        {
            return -1;
        }
    }
}
```

Brisanje reda

```
case "11":  
    //Brisanje kategorije  
    Console.WriteLine("Unesite id kategorije koju brisete");  
    id = int.Parse(Console.ReadLine());  
    r = MagacinDal.ObrisiKategoriju(id);  
    if (r == 0)  
    {  
        Console.WriteLine("Obrisana kategorija ciji je id:" + id);  
    }  
    else  
    {  
        Console.WriteLine("Greska pri brisanju kategorije");  
    }  
    MagacinDal.PrikaziKategoriju(id);  
    break;
```

Brisanje reda

```
C:\Users\goran\source\rep x + v
Da -> Pritisni taster d
Ne -> Pritisni taster n
-----
d
-----
1 -> Prikazi proizvode sa odlozenim izvsavanjem upita
2 -> Prikazi proizvode sa materijalizacijom upita
3 -> Prikazi kategorije proizvoda
4 -> Proizvodi iz kategorije
5 -> Proizvodi iz kategorije - navigaciona svojstva
6 -> Prikaz proizvoda na osnovu id -a SingleOrDefault
7 -> Prikaz proizvoda na osnovu id -a Find
8 -> Prikaz kategorije na osnovu id -a Find
9 -> Unos nove kategorije
10 -> Promena postojece kategorije
11 -> Brisanje kategorije
12 -> izađi
-----
Odaberite DB operaciju:
11
Unesite id kategorije koju brisete
9
Obrisana kategorija ciji je id:9
Ne postoji kategorija
-----
Da li zelite novu db operaciju?
Da -> Pritisni taster d
Ne -> Pritisni taster n
-----
|
```

Neuspešno brisanje reda

```
C:\Users\goran\source\rep. x + v
Da -> Pritisni taster d
Ne -> Pritisni taster n

-----
d
-----
1 -> Prikazi proizvode sa odlozenim izvršavanjem upita
2 -> Prikazi proizvode sa materijalizacijom upita
3 -> Prikazi kategorije proizvoda
4 -> Proizvodi iz kategorije
5 -> Proizvodi iz kategorije - navigaciona svojstva
6 -> Prikaz proizvoda na osnovu id -a SingleOrDefault
7 -> Prikaz proizvoda na osnovu id -a Find
8 -> Prikaz kategorije na osnovu id -a Find
9 -> Unos nove kategorije
10 -> Promena postojece kategorije
11 -> Brisanje kategorije
12 -> izađi

-----
Odaberite DB operaciju:
11
Unesite id kategorije koju brisete
1
Greska pri brisanju kategorije
1 Sokovi

-----
Da li zelite novu db operaciju?
Da -> Pritisni taster d
Ne -> Pritisni taster n

-----
```

Klase Task i Task<T>

- Klasa Task se nalazi u biblioteci **System.Threading.Tasks**
- Klasa Task se koristi za izvršavanje više poslova istovremeno, svaki u posebnoj programskoj niti
- Obično se vremenski zahtevni zadaci izvršavaju u posebnoj programskoj niti
- Klasa **Task** predstavlja metodu koja ne vraća vrednost, izvršava se u posebnoj programskoj niti i to izvršavanje je asinhrono
- Klasa **Task<T>** predstavlja asinhronu operaciju koja vraća rezultat

Vraćanje vrednosti iz zadatka

- Koristi se Task<TResult> klasa
- Koristi se Result svojstvo

```
private void button1_Click(object sender, EventArgs e)
{
    Task<string> t1 = Task.Run(
        () =>
        {
            Thread.Sleep(5000);
            return DateTime.Now.ToLongTimeString();
        }
    );

    // UI je blokiran
    label1.Text = t1.Result;
}
```


Korišćenje operatora `async` i `await`

- Modifikator `async` označava da je metoda asinhrona
- Unutar asinhrone metode se koristi operator `await` koji suspenduje izvršavanje metode sve dok se zadatak ne završi
- Omogućavaju poziv asinhrone metode i čekanje rezultata bez blokiranja programske niti

Izvršavanje asinhronone operacije u UI programskoj niti

```
private async void button1_Click(object sender, EventArgs e)
{
    Task<string> t1 = Task.Run(
        () =>
        {
            Thread.Sleep(5000);
            return DateTime.Now.ToLongTimeString();
        }
    );

    // linija koda se izvrsava kada je rezultat raspoloziv
    // UI nije blokiran

    label1.Text = await t1;
}
```

Kreiranje Awaitable metode

- Ako sinhrona metoda vraća **void** asinhroni ekvivalent vraća **Task**
- Ako sinhrona metoda vraća tip **T**, asinhroni ekvivalent vraća **Task<T>**

Asinhrona metoda

```
private async Task<string> CitajAsinhrono()
{
    Task<string> task1 = Task.Run(() =>
    {
        //simulacija dugotrajnog procesa
        Thread.Sleep(3000);
        return DateTime.Now.ToLongTimeString();
    });

    return await task1;
}
```

```
private async void button2_Click(object sender, EventArgs e)
{
    label1.Text = await CitajAsinhrono();
}
```

Pitanje 1

Kod Entity Frameworka Core tehnologije osnovna klasa koja se koristi kao bazna klasa za manipulaciju sa entitetskim objektima naziva se:

- a. DbContext klasa
- b. DataSet klasa
- c. EntityContext klasa

Odgovor: a

Pitanje 2

Ako je **MagacinContext** klasa izvedena iz **DbContext** klase. Neka je **db** instanaca **MagacinContext** klase. Klasa **MagacinContext** ima svojstvo **Kategorije** pomoću koga se pristupa DbSet objektu tipa **DbSet<Kategorija>** koji se sastoji od objekata entitetske klase **Kategorija**. Kako se pronalazi atačovani objekat čiji id ima vrednost 1:

- a. `Kategorija k1 = db.Kategorije.Find(k=>k.id=1);`
- b. `Kategorija k1 = db.Kategorije.Select(1);`
- c. `Kategorija k1 = db.Kategorije.Find(1);`

Odogovor: c

Pitanje 3

Da bi se promene unutar tipiziranog DbContext objekta označenog sa db sačuvale u bazi podataka pišemo sledeću liniju koda:

- a. `db.Save();`
- b. `db. SaveToDatabase();`
- c. `db.SaveChanges();`

Odogovor: c

Pitanje 4

Za kreiranje operacije koja se izvršava u posebnoj programskoj niti i koja ne vraća vrednost koristi se klasa:

- a. Thread
- b. Task
- c. Task<T>

Odgovor: b

Pitanje 5

Za asinhronu metodu koja treba da vrati string kao povratni tip se specificira:

- a. `Thread<string>`
- b. `Task`
- c. `Task<string>`

Odogovor: c

Pitanje 6

Kreirana je metoda klasa sledećeg potpisa:

private async Task<string> Citaj()

Na koji način pozivamo ovu metodu:

- a. Citaj()
- b. async Citaj()
- c. await Citaj()

Odogovor: c