)

)

c)

)

)

a)

)

) #

)

)

(

( ))

)

)

)

d)

)

)

)

)

)

)

a) `ViewData` `ViewBag`

) `ModelData`

) `HtmlData` `HtmlCache`

) `CurrentContent` `PermanentContent`

)

`Mjesto` `Naziv`

)

```
@Model Mjesto
@model.Naziv
```

b)

```
@model Mjesto
@Model.Naziv
```

)

```
@model Mjesto
@Naziv
```

)

```
@Model Mjesto
@Naziv
```

)

```
@model Mjesto
@Mjesto.Naziv
```

```
try {
        ...
        throw new NullReferenceException(...)
}
catch (Exception exc) {
        Console.WriteLine("EXC");
}
catch (NullReferenceException nre) {
        Console.WriteLine("NRE");
}
finally {
        Console.WriteLine("F");
}
```

)

```
NRE
```

b)

EXC
F

)

NRE
F

)

EXC
NRE
F

)

F

)
)
)
d)
)

)
)
)
d)
)

)
)
)
)
)

a)
)
)

)

)

(

)

)

)

)

d)

)

( )

)

b)

)

)

)

( )

)

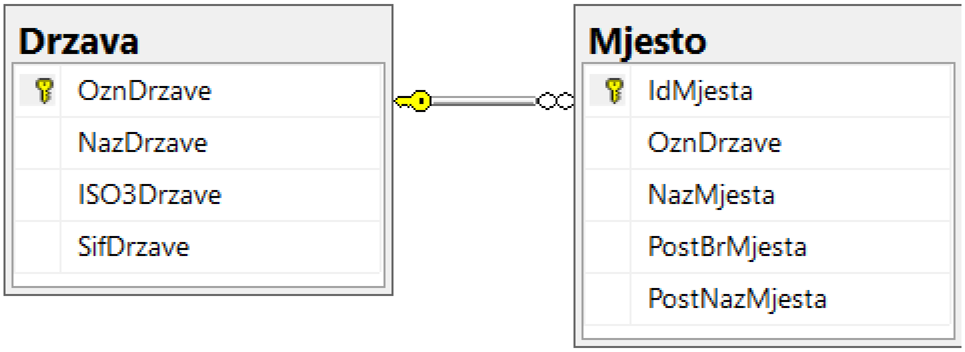
)

)

)

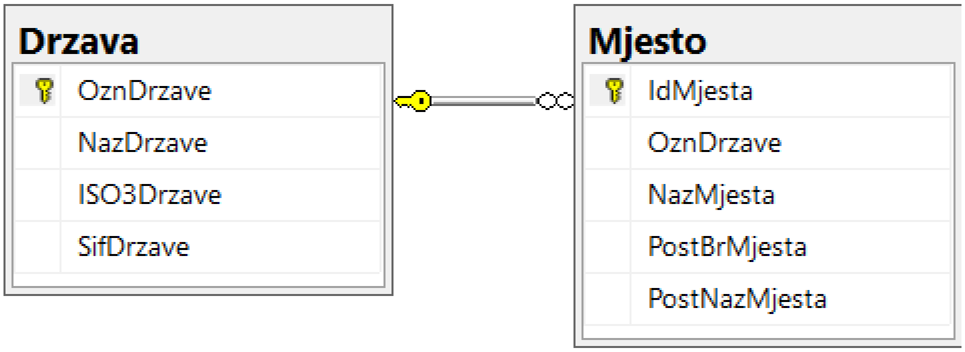
)

Drzava Mjesto



Drzava

)



Mjesto

a)

```
public partial class Mjesto
{
        ...
        public string OznDrzave { get; set; }
        ...
        public Drzava OznDrzaveNavigation { get; set; }
}
```

naziv=vrijednost (

p=1&q=7&user=student http://apps.zpr.fer.hr/myapp/run?p=1&q=7&user=student)

)

)

c)

)

)

)

b)

)

)

)

a)

)

)

)

)

) ( )

) ( )

) ( )

d) ( )

) ( )

) ( )

) ( )

) ( )

) ( )

) ( )

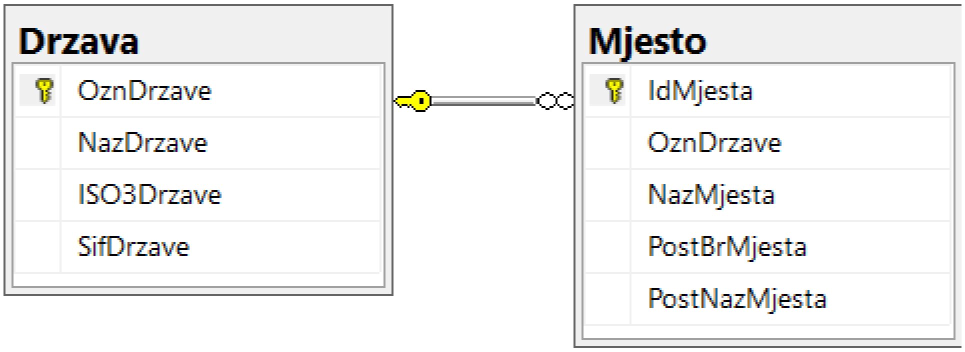
a) ( )

) ( )

) ( )

) ( )

) ( )



ctx

a)

```
var list = ctx.Mjesto
  .Select(m => new {
        m.NazivMjesta,
        Drzava = m.OznDrzaveNavigation.NazDrzave
  }).ToList();
```

a)

)

)

)

)

```
class Primjer
{
        private int index;
        public int Index
        {
                get { return index; }
                set {
                        if (value <= 0) { this.index = 0; }
                        else if (value > 100) { this.index = 100; }
                        else { this.index = value; }
                }
        }
}
```

)

)

)

)

)

"

"

)

)

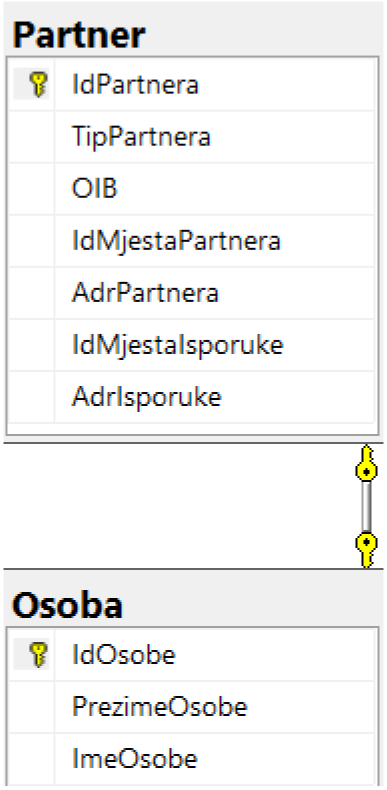
)

d)

)

a) `http://.../firma/artikl(8)`

`Partner` `Osoba`



Partner

- IdPartnera
- TipPartnera
- OIB
- IdMjestaPartnera
- AdrPartnera
- IdMjestaIsporuke
- AdrIsporuke

Osoba

- IdOsobe
- PrezimeOsobe
- ImeOsobe

Osoba

)

```
public partial class Osoba {
        ...
        public int IdPartnera { get; set; }
        ...
        public ICollection<Partner> IdOsobeNavigation { get; set; }
}
```

b)

```
public partial class Osoba {
```

)

```
public partial class Osoba
{
      ...
      public int IdOsobe { get; set; }
}
```

( )

a)

SELECT OznDrzave, NazDrzave FROM Drzava;

DbCommand IDataReader

Drzava

)

```
string oznaka = (string) reader["OznDrzave"];
string naziv = (string) reader["NazDrzave"];
```

)

```
string oznaka = reader.GetString(0);
string naziv = reader.GetString(1);
```

)

```
string oznaka = reader["OznDrzave"].ToString();
string naziv = reader["NazDrzave"].ToString();
```

d)

```
string oznaka = (string) reader.GetName("OznDrzave");
string naziv = (string) reader.GetName("NazDrzave");
```

)

)

c)

)

)

```
void IspisiVozila(IEnumerable<MotornoVozilo> vozila) {
      foreach(var vozilo in vozila)
            Console.WriteLine(vozilo.Model);
}
```

Automobil Motocikl MotornoVozilo IspisiVozilo

List<Automobil> IEnumerable<T>

a)

)

)

)

)

(

)

)

b)

)

)

)

)

)

)

d)

)

a) ApiController

) TypeFilter

) BadRequestFilter

) ControllerBase

) ExceptionFilter

" "

) " "

) " " " "

)

)

) " "

a)

(

)

)

)

)

)

)

a) `http://.../firma/artikl?$filter=CijArtikla gt 100 and CijArtikla lt 500` C

x

```
public string this[int a, int b] {
        get {
                ...
        }
        set {
                ...
        }
}
```

) `x["A", 5] = 1 + x["B", 3]`

b) `x[1, 2] = "Z" + x[2, 3]` n"

) `x(1, 2) = "Z"`

) `x.`

) ( )

)

a) `Func<int, int, double>`

) `Func<int<,>, double>`

) `Action<int, int, double>`

) `Func<Action<int, int>, double>`

) `Func<double, int, int>`

( )

)

b)

c)

d)

)

" "

a)

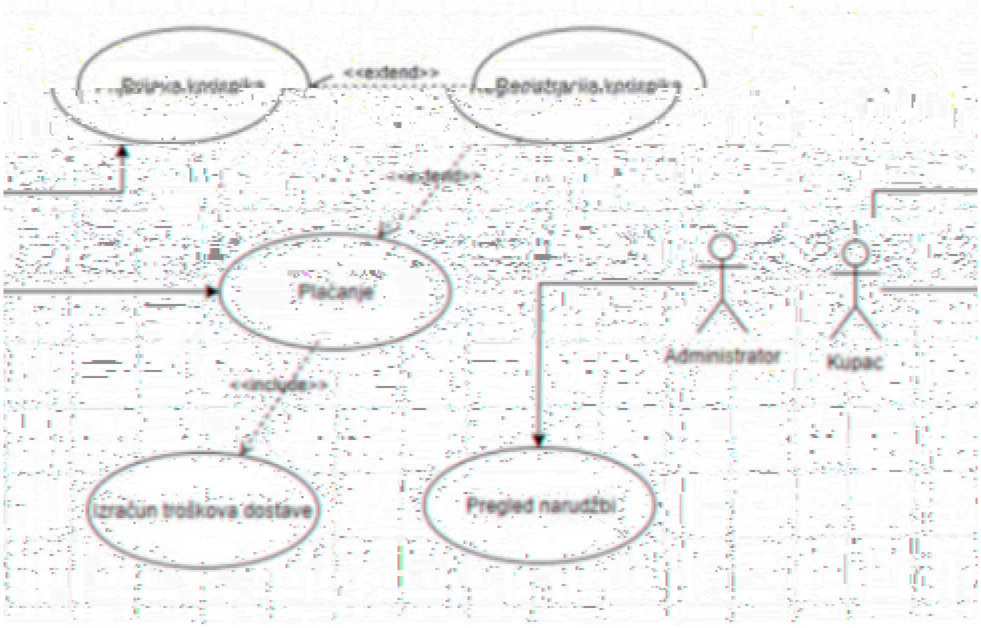
)

) ( )

c)

)

)



a)

(

)

)

b)

)

)

)

" "

a)

```
1. Introduction
      1.1. Project Overview
      1.2. Project Deliverables
      1.3. Evolution of the Software Project Management Plan
      1.4. Reference Materials
      1.5. Definitions and Acronyms
2. Project Organization
      2.1 Process Model
      2.2 Organizational Structure
      2.3 Organizational Boundaries and Interfaces
      2.4 Project Responsibilities
3. Managerial Process
      3.1 Management Objectives and Priorities
      3.2 Assumptions, Dependencies, and Constraints
      3.3 Resource Requirements
      3.4 Budget and Resource Allocation
      3.5 Schedule
6. Additional Components
7. Index
8. Appendices
```

a)

```
int a = 0, b = 0, c;
try {
      try {
            Console.Write("T1 ");
            c = b / a;
      }
      finally {
            Console.Write("F1 ");
      }
}
catch (Exception exc2) {
      Console.Write("E2 ");
}
finally {
      Console.Write("F2 ");
}
```

a) T1 F1 E2 F2

```
try {
      try {
            Console.Write("T1 ");
      }
      finally {
            Console.Write("F1 ");
      }
}
catch (Exception exc2) {
      Console.Write("E2 ");
}
finally {
      Console.Write("F2 ");
}
```

a) T1 F1 F2

%

a)

(bower.json packages.json)

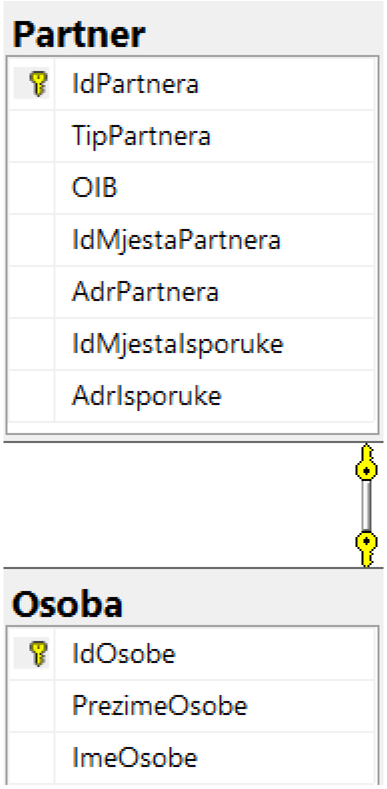
a) `"jQuery": "~3.3.1"`

a)

`private`

a)

Partner Osoba



Partner

- IdPartnera
- TipPartnera
- OIB
- IdMjestaPartnera
- AdrPartnera
- IdMjestaIsporuke
- AdrIsporuke

Osoba

- IdOsobe
- PrezimeOsobe
- ImeOsobe

Partner

a)

```
public partial class Partner
{
        ...
        public int IdOsobe { get; set; }
        ...
        public Osoba IdOsobeNavigation { get; set; }
}
```

)

```
public partial class Partner
{
       ...
       public int IdPartnera { get; set; }
       ...
       public Partner IdOsobe { get; set; }
}
```

a)

a)

%

a)

a)

```
class GenRazred<T> where T : class {
         ...
}
```

) `GenRazred<List<int>> x;`

) `GenRazred<Exception> x;`

) `GenRazred<string> x;`

d) `GenRazred<System.Int32> x;`

) `GenRazred<List<string>> x;`

(

)

a)

b)

c)

d)

)

#

a) `System.Object`

a)

)

)

)

)

a)

)

)

)

)

a) `@* komentar koji se neće pojaviti u generiranom html-u *@`

)

)

c)

)

)

"   "

a)

a) ( )

a)

(#)

a) ( )

a)

```
class Razred : IDisposable {
      private string Naziv;

      public Razred(string Naziv) {
            this.Naziv = Naziv;
      }

      public void Dispose() {
            Console.WriteLine(Naziv);
      }
}

class Program {
      static void Main(string[] args) {
            try {
                  Razred r1 = new Razred("A1");
                  using (Razred r2 = new Razred("B2")) {
                        Razred r3 = new Razred("C3");
                        r3 = null;
                        throw new Exception("Poruka");
                  }
                  r1.Dispose();
            }
            catch { }
      }
}
```

Razred Dispose

) C3

) B2 C3

c) B2

) B2 A1

) B2 C3 A1

a)

)

)

)

)

)

```
static void Main(string[] args) {
        Action<Func<int, int, int>> action = (f1) => Console.WriteLine(f1(5, 20));

        action += (f2) => Console.WriteLine(f2(3, 6));
        action((f1, f2) => f1 + f2);
}
```

a)

```
25
9
```

a) * (*)

a)

a)

Automobil Motocikl MotornoVozilo IComparer

public interface IComparer<in T>

```
void IspisiBoljiMotor(Motocikl a, Motocikl b, IComparer<Motocikl> comparer) {
        Console.WriteLine(comparer.Compare(a, b) < 0 ? a : b);
}
```

( )

a) IComparer<MotornoVozilo>

a)

)

)

)

)

) FromQuery

b) FromScript

) FromRoute

) FromServices

) FromForm

a)

" "

a)

a)

)

)

c)

)

)

Car Vehicle M

void M(List<Vehicle> list) M ( )

a) List<Vehicle>

) List<Vehicle> List<Car>

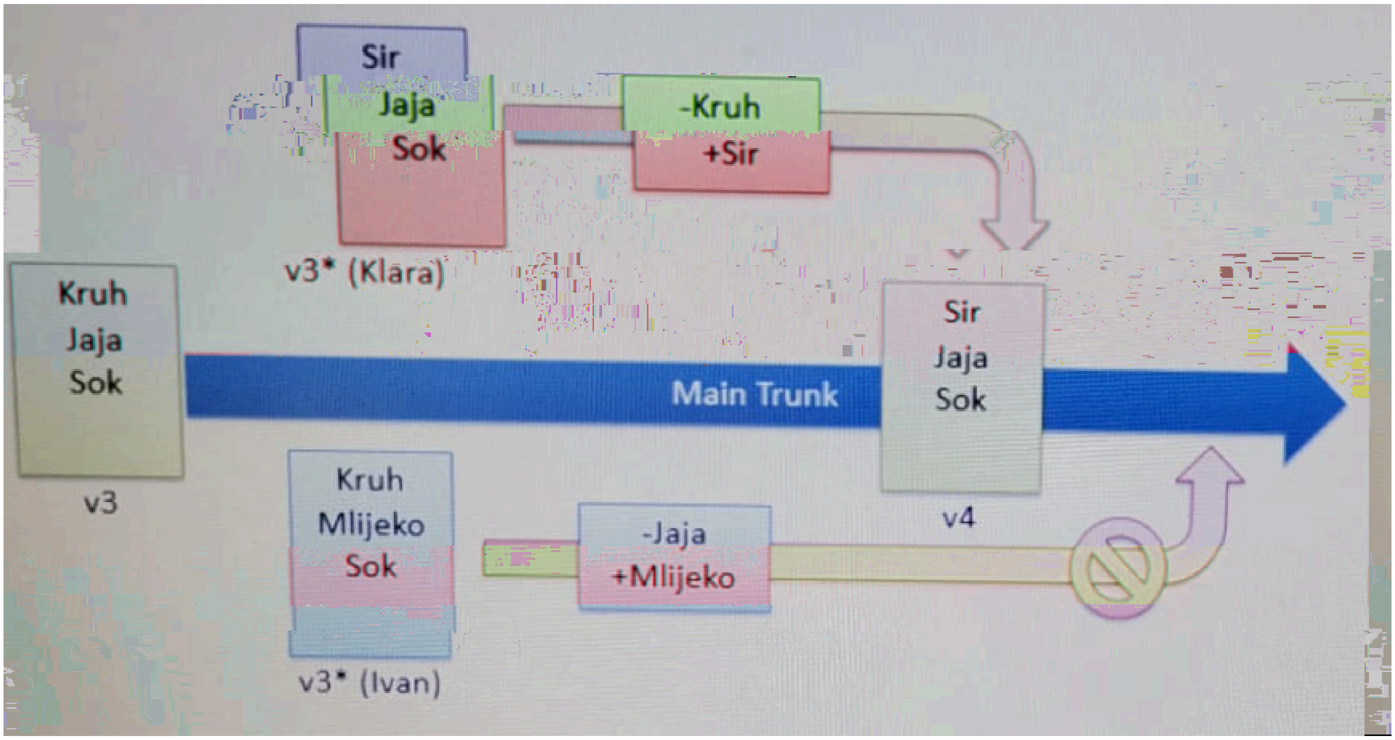
) IEnumerable<Vehicle> List<Vehicle>

) List<Vehicle> List<Car> IEnumerable<Vehicle> IEnumerable<Car>

) IEnumerable<Vehicle> IEnumerable<Car>

internal

a)



a)

a)

```
int a = 1, b = 2;
M(ref a, out b);
...
void M(ref int a, out int b) {
        b = 20;
}
```

ControllerBase

Tools Categories

)

```
[HttpPost]
public IActionResult
```

```
[HttpPost]
public IActionResult Assign([FromBody] Tools tools, [FromBody] Categories categories)
```

)

```
[HttpPost]
public IActionResult Assign([FromBody] Tools tools, [FromQuery] Categories categories)
```

)

```
[HttpPost]
public IActionResult Assign(Tools tools, [FromServices] Categories categories)
```

SifArtikla Artikl

Artikl

```
using (var context = new FirmaContext())
{
        for (int i = 1; i <= 10; i++)
        {
                Artikl artikl = new Artikl
                {
                        SifArtikla = i/2, CijArtikla = 10m,
                        JedMjere = "kom", NazArtikla = i.ToString()
                }
                context.Artikl.Add(artikl);
        }
        context.SaveChanges();
}
```

Artikl

a)

)

)

c)

)

)

out public interface IEnumerable<out T>

a)

a)

a) http://.../firma/artikl?$top=3&$orderby=CijArtikla desc

) http://.../firma/artikl(3)/By/CijArtikla

) http://.../firma/artikl/3?orderBy=CijArtikla&desc=true

) http://.../firma/artikl?$top=3&$filter=CijArtikla desc

) http://.../firma/3/artikl>$$by=CijArtiklaDesc

( )

a)

) FromBody

) FromRoute

c) FromApi

) FromForm

) FromHeader

```
static void Main(string[] args) {
        Action<int, string, Func<char, string, int>> action = (i, s, f) => Console.WriteLin
        action(6, "ispit iz RPPP-a", (x, s) => {
                int sum = 0;
                foreach (var c in s) {
                        if (x == c) {
                                ++sum;
                        }
                }
                return sum;
        });
}
```

a)

) ( )

)

)

)

) ( )

b) ( )

Create

a)

```
return RedirectToAction("Create");
```

a)

a)

a)

a)

Automobil Motocikl MotornoVozilo IEnumerable

```
public interface IEnumerable<out T>
```

```
void IspisiVozila(IEnumerable<MotornoVozilo> vozila) {
      foreach(var vozilo in vozila)
            Console.WriteLine(vozilo.Model);
}
```

a) List<Automobil>

) List<IEnumerable<Motocikl>>

) Motocikl

)

) Automobil

)

b) ( )

c)

)

)

a)

a)

)

)

)

d)

)

a)

a)

#

a)

)

```
dynamic s = 7;
```

)

```
dynamic s;
```

)

```
dynamic s;
s = 12;
s = "Pero";
```

)

```
dynamic s; s = "Pero";
s = 12;
```

a)

)

)

)

)

```
int a = 0, b = 0, c;
try {
        try {
                Console.Write("T1 ");
                c = b / a;
        }
        catch (Exception exc) {
                Console.Write("E1 ");
                c = a / b;
        }
        finally {
                Console.Write("F1 ");
        }
}
catch (Exception exc2) {
        Console.Write("E2 ");
}
finally {
        Console.Write("F2 ");
}
```

a) T1 E1 F1 E2 F2

(

)

a) TempData

( )

a)

)

)

)

)

)

)

)

)

)

)

a)

a)

```
public class MjestoController : ControllerBase {
        ...
        [HttpPost]
        public IActionResult Create(string s, int i) {
                ...
        }
}
```

a)

a)

( )

a)

a)

a)

a)

a)

R

```
class R {
        public int X { get; set; }
        public int Y { get; set; }
        public R() : this(5, 6) { }
        public R(int x) {
                Y = x;
        }
        public R(int a, int b) {
                X = a;
                Y = b;
        }
}
```

```
R r = new R {
        X = 7
};
Console.WriteLine(r.X + " " + r.Y);
```

a) 7 6

)

)

)

)

)

)

)

)

d)

)

a) ( )

a)

- Vozilo   Automobil   ElektricniAutomobil
- Vozilo   Motocikl
- Osoba   Djelatnik   Direktor

```
Djelatnik ZaduzenZa(Automobil a)
```

a)

```
Func<ElektricniAutomobil, Direktor> func = ZaduzenZa;
```

)

```
Func<ElektricniAutomobil, Osoba> func = ZaduzenZa;
```

c)

```
Func<Vozilo, Direktor> func = ZaduzenZa;
```

d)

```
Func<Vozilo, Osoba> func = ZaduzenZa;
```

)

```
Func<Vozilo, ElektricniAutomobil> func = ZaduzenZa;
```

- Vozilo   Automobil   ElektricniAutomobil
- Vozilo   Motocikl
- Osoba   Djelatnik   Direktor

```
void Zaduzi(Automobil a, Djelatnik D)
```

)

```
Action<Vozilo, Direktor> action = Zaduzi;
```

)

```
Func<ElektricniAutomobil, Osoba> action = Zaduzi;
```

)

```
Action<Vozilo, Osoba> action = Zaduzi;
```

d)

```
Action<ElektricniAutomobil, Direktor> action = Zaduzi;
```

)

```
Action<ElektricniAutomobil, Osoba> action = Zaduzi;
```

%

a)

( )

)

)

)

d)

)

) ( )

)

( )

)

)

)

)

)

) ( )

)

`Artikl`

```
using (var context = new FirmaContext())
{
        List<Artikl> artikli = context.Artikl
                .AsNoTracking()
                .Where(a => a.SifArtikla < 3)
                .ToList();
        artikli[0].CijArtikla += 10m;
        artikli[1].CijArtikla = artikli[0].CijArtikla;
        context.SaveChanges();
}
```

a) SELECT UPDATE

)

)

)

d)

)

)

b)

)

)

)

a)

)

)

)

)

IConnectionStringTool ConnectionStringTool

FirmaContext IConnectionStringTool

( ) FirmaContext

b)

```
private static IServiceProvider serviceProvider;
public static void Main(string[] args) {
        serviceProvider = new ServiceCollection()
                .AddSingleton<IConnectionStringTool, ConnectionStringTool>()
                .AddTransient<FirmaContext, FirmaContext>()
                .BuildServiceProvider();

        ...
        FirmaContext ctx = serviceProvider.GetService<FirmaContext>();
        ...
}
```

)

)

)

)

)

Stavka

- 1
- 2
- 3

```
int sum = 0;
var upit = context.Artikl
        .Include(a => a.Stavka)
        .AsNoTracking()
        .Where(a => a.SifArtikla >=1 && a.SifArtikla <= 3)
        .ToList();
foreach (Artikl artikl in upit) {
        sum += artikl.Stavka.Count();
}
```

d) SELECT

x

```
using(var context = new FirmaContext()) {
var x = context.Artikl
      .Select(a => new {
            a.SifArtikla, a.NazArtikla, a.CijArtikla,
      })
      .First();
}
```

) IEnumerable

) SifArtikla NazArtikla CijArtikla

d) NazArtikla CijArtikla SifArtikla

( )

)

)

)

)

)

a) ( )

)

)

)

)

a)

)

) (

)

)

)

( )

a)

)

)

c)

)

a)

) #

)

)

d)

)

Postupak()

a) base.Postupak();

)

return NotFound()

a)

a)

a) ( ) ( )

a)

a)

a)

a) ( )

a) ( )

a) ( )

a)

)

)

("   ")

)

)

("   ")

a)

a)

(   )

)

```
[ProducesResponseType((int) HttpStatusCode.NotFound)]
public async Task<IActionResult> Get(string sifArtikla)
```

)

```
[ProducesResponseType(typeof(IActionResult), (int) HttpStatusCode.OK)]
[ProducesResponseType((int) HttpStatusCode.NotFound)]
public async Task<IActionResult> Get(string sifArtikla)
```

c)

```
[ProducesResponseType(typeof(Artikl), (int) HttpStatusCode.OK)]
[ProducesResponseType((int) HttpStatusCode.NotFound)]
public async Task<IActionResult> Get(string sifArtikla)
```

)

)

)

)

)

a)

)

c)

d)

)

)

c)

)

)

)

)

c)

)

)

)

)

)

)

)

)

)

)

d)

)

( )

( )

a)

)

)

)

)

)

)

) '

)

)

in public interface IComparer<in T>

a)

)

)

)

)

)

)

)

)

)

```
static void Main(string[] args)
{
        Action<int, string, Func<char, string, int>> action = (i, s, f) => Console.WriteLin
        action(4, "This is not an easy question", (x, s) =>
        {
                int sum = 0;
                foreach (var c in s) {
                        if (x == c)
                                ++sum;
                }
                return sum;
        });
}
```

a) 5

) 4

a)

)

)

)

)

)

b)

)

)

)

)

)

)

)

)

) `ExceptionFilter`

) `BadRequest`

) `NotFound`

d) `ProblemDetails`

) `ActionResult`

( )

)

)

)

)

)

)

b)

)

)

)

a)

)

)

)

)

```
[ApiController]
[Route("[controller]")]
public class CitiesController : ControllerBase {
        ...
        [HttpDelete("{id}", Name = "DeleteCity")]
        public async Task<IActionResult> DeleteCity(int id)
        ...
}
```

) .../cities/deletecity/100

b) .../cities/100

) .../cities/delete?id=100

) .../cities/deletecity?id=100

) .../cities/delete/100

```
public class MjestoController : ControllerBase {
        ...
        [HttpPost]
        public IActionResult Create(MjestoViewModel model)
        ...
}
```

)

b)

)

)

https://server/app/mjesto   https://server/app/drzava

)

)

c)

)

)