

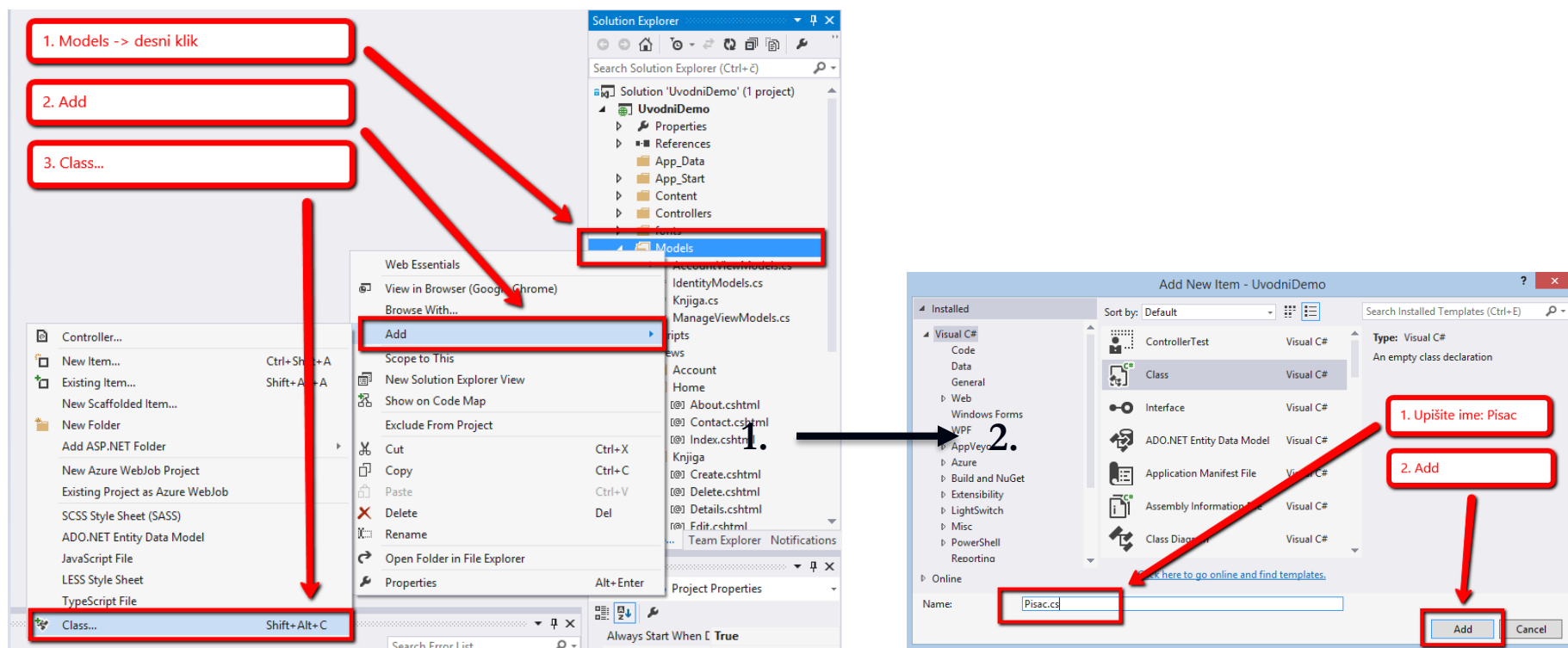
MVC 5

(Model-View-Controller)

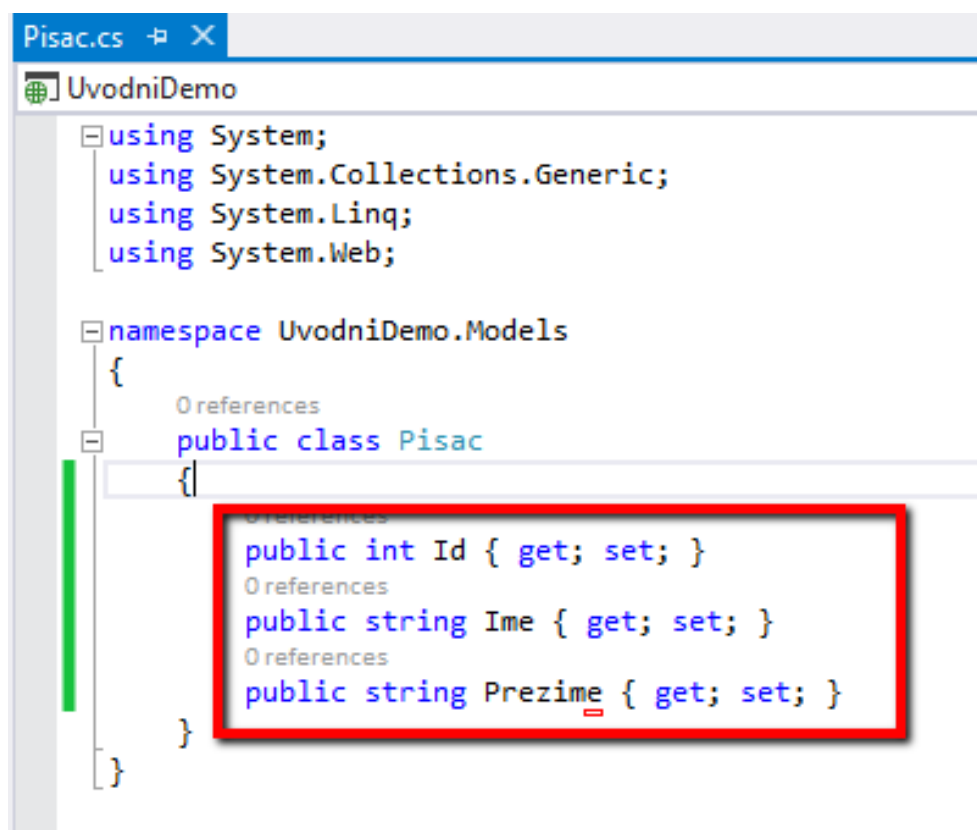
Kreiranje glavnih
objekata - Scaffolding

- Koraci koji slijede su rješenje 2. zadatka iz dokumenta – 01 MVC Zadaci - Osnove.pdf
- Također, sam kod (source code) je dio primjera – UvodniDemo, i tamo možete provjeriti ovo rješenje i sav pripadajući kod

- Kreirajte projekt tipa ASP.NET Web Application
- Kreirajte model – Pisac – unutar direktorija Models



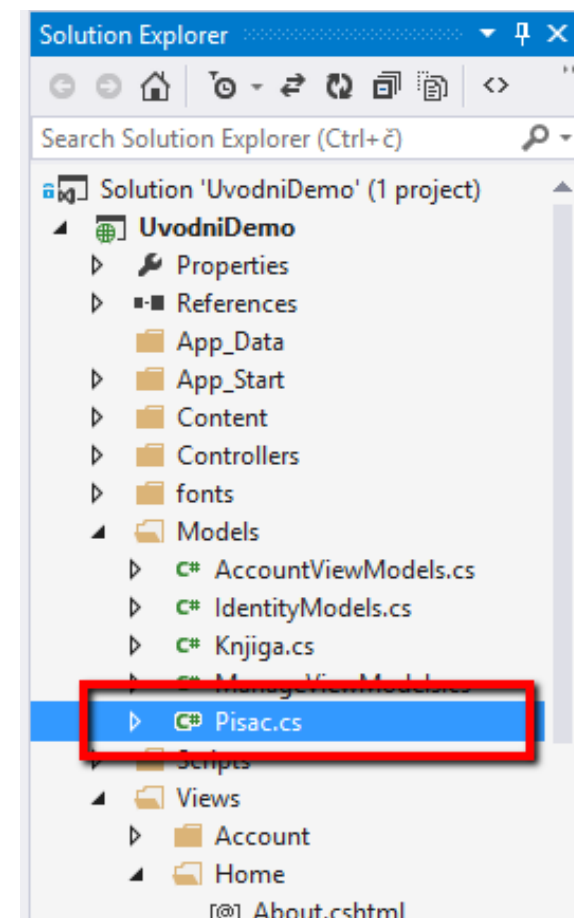
- Definirajte model – Pisac – kao klasu sa slijedećim svojstvima:



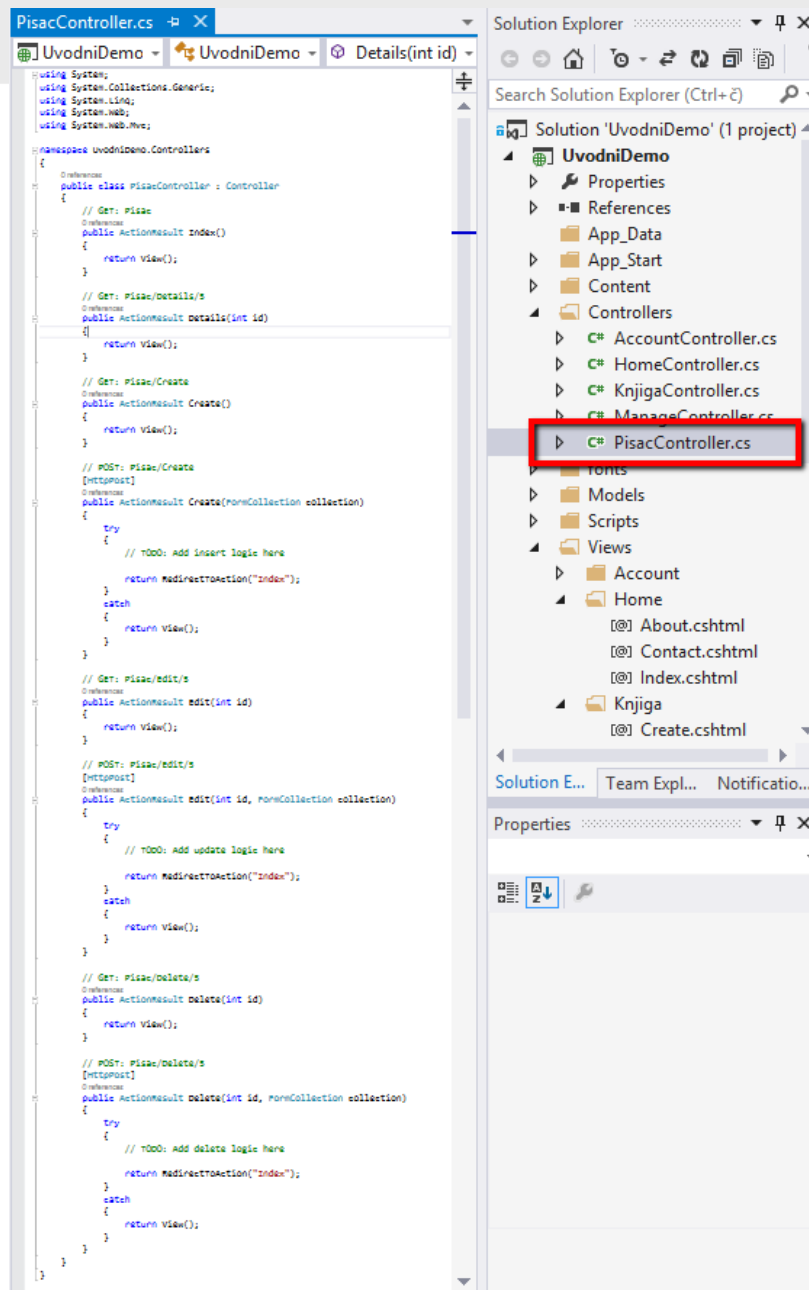
```
Pisac.cs
UvodniDemo

using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace UvodniDemo.Models
{
    public class Pisac
    {
        public int Id { get; set; }
        public string Ime { get; set; }
        public string Prezime { get; set; }
    }
}
```



- Dobili smo Controller sa slijedećim metodama (akcijske metode):



■ Kreirajte view za Index akcijsku metodu

1. Desni klik na View

2. Add View...

1. Ctrl+M, Ctrl+G

2.

Odabrati slijedeće opcije ...

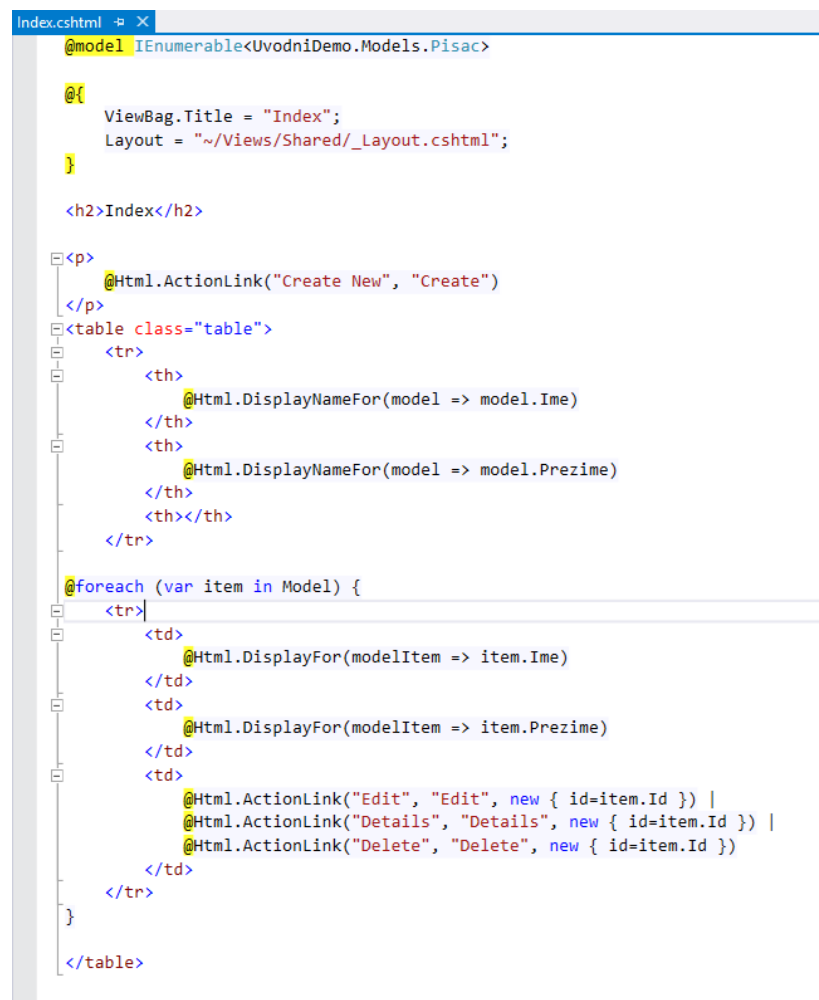
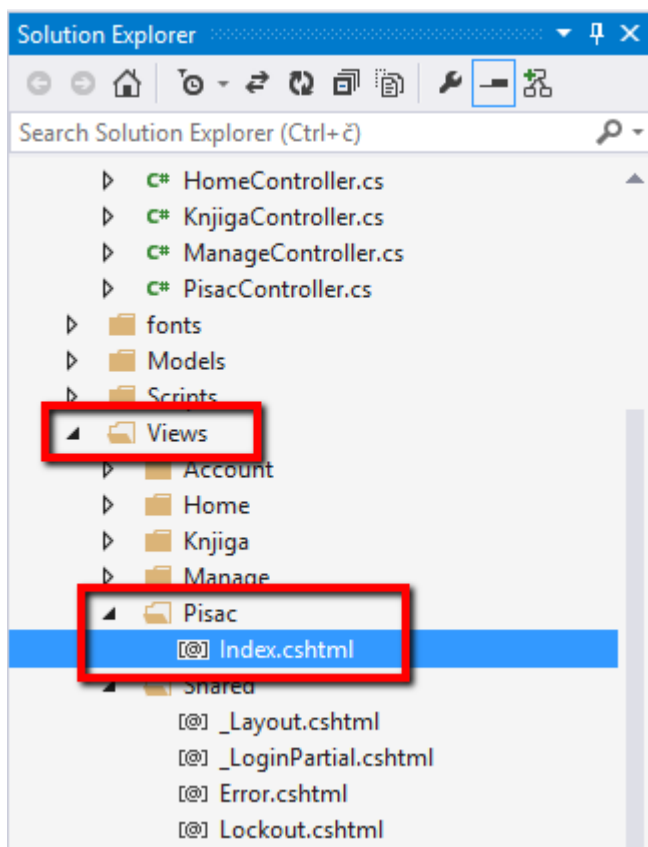
Kliknite Add

Add

Cancel

Metoda	View name	Template	Model class	Use a layout page
Index	Index	List	Pisac	✓

- Dobili smo view Index unutar direktorija Views -> Pisac

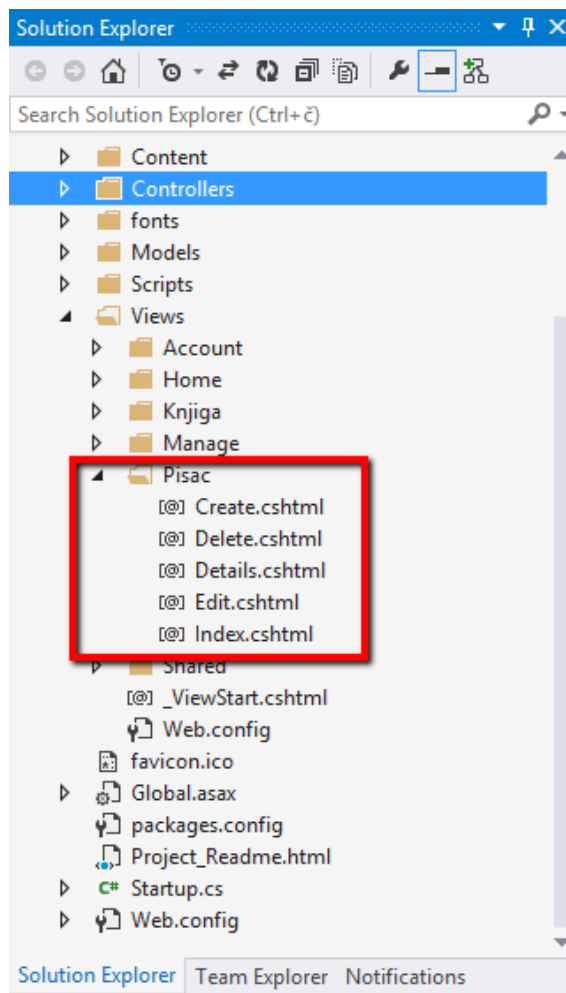


- Na isti način kreirajte ostale View-ove za ostale akcijske metode controllera – PisacController

Metoda	View name	Template	Model class	Use a layout page
Details	Details	Details	Pisac	✓
Create	Create	Create	Pisac	✓
Edit	Edit	Edit	Pisac	✓
Delete	Delete	Delete	Pisac	✓

- Spremite i zatvorite sve kreirane datoteke
- Neke metode se dupliraju (Create, Edit, Delete) pa je u tom slučaju dovoljno napraviti View samo jednom.

- Na kraju trebate imati View-ove kao na slici:



- Slijedi programski kod (source code) akcijskih metoda controllera – PisacController
- Globalna varijabla na nivou samog Controllera je deklarirana ovom linijom:

```
static List<Pisac> pisci = new List<Pisac>();
```

- Metoda za inicijalno punjenje liste pisaca
 - Ovako simuliramo podatke dohvaćene iz baze i kasnije ćemo to stvarno i zamijeniti realnim objektima koji dohvaćaju podatke iz baze

```
private void NapuniPisce()
{
    pisci.Add(new Pisac() { Id = 1, Ime = "Dobriša", Prezime = "Cesarić" });
    pisci.Add(new Pisac() { Id = 2, Ime = "Tin", Prezime = "Ujević" });
    pisci.Add(new Pisac() { Id = 3, Ime = "Jure", Prezime = "Kaštelan" });
}
```

- Akcijska metoda – Index
 - Punimo listu ako je prazna
 - Prikazujemo podatke

```
public ActionResult Index()
{
    if (pisci.Count == 0)
        NapuniPisce();

    return View(pisci);
}
```

- Akcijska metoda – Details
 - Dohvaćamo pisca sa određenim Id-jem
 - Prikazujemo podatke

```
public ActionResult Details(int id)
{
    Pisac pisac = pisci.FirstOrDefault(item => item.Id == id);

    return View(pisac);
}
```

- Akcijska metoda – Create (GET)
 - Ova metoda ostaje onakva kakva je i generirana, ostaje nepromijenjena
 - Prikazujemo prazan obrazac za unos novog pisca
 - Korisnik će unijeti podatke koji će spremiti

```
public ActionResult Create()
{
    return View();
}
```

■ Akcijska metoda – Create (POST)

- Ova metoda će primiti podatke sa forme i spremiti novog pisca

[HttpPost]

```
public ActionResult Create(FormCollection collection) {  
    try {  
        // TODO: Add insert logic here  
        int maxId = pisci.Max(item => item.Id);  
  
        pisci.Add(new Pisac() { Id = maxId + 1, Ime = collection["Ime"],  
                                Prezime = collection["Prezime"]});  
  
        return RedirectToAction("Index");  
    } catch {  
        return View();  
    }  
}
```


- Akcijska metoda – Edit (GET)
 - Dohvaćamo pisca sa određenim Id-jem
 - Omogućujemo uređivanje podataka

```
public ActionResult Edit(int id)
{
    Pisac pisac = pisci.FirstOrDefault(item => item.Id == id);

    return View(pisac);
}
```

■ Akcijska metoda – Edit (POST)

- Ova metoda će primiti podatke sa forme i spremiti postojećeg pisca

[HttpPost]

```
public ActionResult Edit(int id, FormCollection collection)
{
    try {
        // TODO: Add update logic here
        Pisac pisac = pisci.FirstOrDefault(item => item.Id == id);
        pisac.Id = id;
        pisac.Ime = collection["Ime"];
        pisac.Prezime = collection["Prezime"];

        return RedirectToAction("Index");
    } catch {
        return View();
    }
}
```

- Akcijska metoda – Delete (GET)
 - Dohvaćamo pisca sa određenim Id-jem
 - Omogućujemo njegovo brisanje

```
public ActionResult Delete(int id)
{
    Pisac pisac = pisci.FirstOrDefault(item => item.Id == id);

    return View(pisac);
}
```

■ Akcijska metoda – Delete (POST)

- Ova metoda će primiti podatke sa forme i obrisati postojećeg pisca

[HttpPost]

```
public ActionResult Delete(int id, FormCollection collection) {  
    try {  
        // TODO: Add delete logic here  
        foreach (var item in pisci) {  
            if (item.Id == id) {  
                pisci.Remove(item);  
            }  
        }  
  
        return RedirectToAction("Index");  
    } catch {  
        return View();  
    }  
}
```

- Ako želite sve isprobati od početka, pobrišite slijedeće objekte i krenite ponovo:

