

ASP.NET

Web stranice mogu da budu:

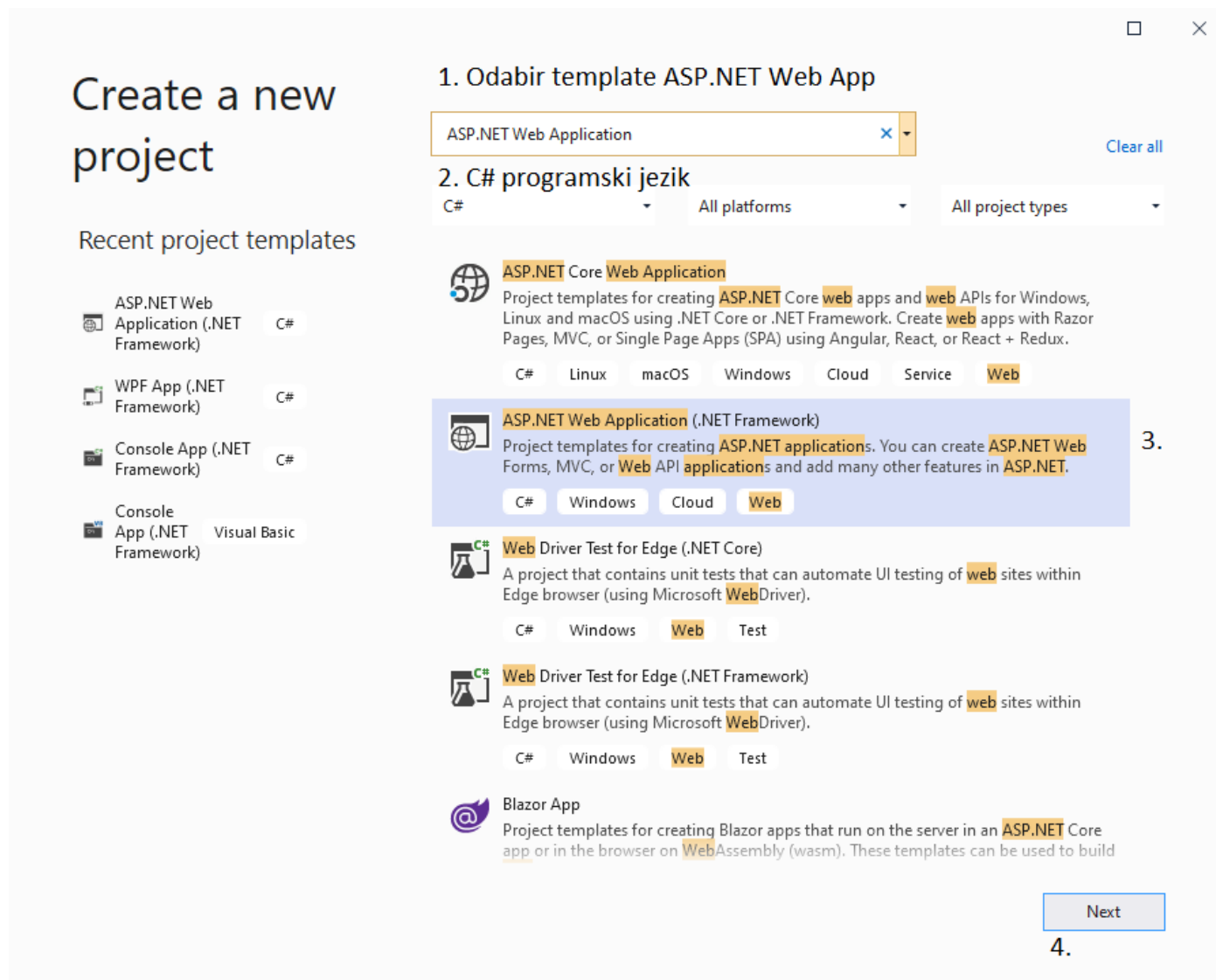
- statičkog sadržaja - Stranice koje se ne menjaju, na njima su uglavnom prikazane forme za unos. Sadržaj koji fizički postoji na serveru.
- dinamičkog sadržaja - Stranice ciji se sadržaj generiše na osnovu klijentkog zahteva, npr. odgovor uspešnosti. Sadržaj koji fizički ne postoji na serveru.

Razor stranice

Razor stranice sadrže HTML i C# kod. Dinamički generisan sadržaj koda započinje sa znakom @.

Kreiranje inicijalnog projekta

Prilikom kreiranja inicijalno prazne Web Aplikacije potrebno je odabrati opciju File>New Project (Slika 1). Iz Create a new project prozora odabere template ASP.NET Web Application (.NET Framework) C#.



Slika 1.

U sledećem prozoru (Slika 2.) Configure your new project zadajete Project name, lokaciju gde da se kreira vaš projekat ili odaberete da se kreira u postojećem Solution. Odaberete verziju .NET Framework (za projekat <= 4.7.0) i nakon toga Create.

□

×

Configure your new project

ASP.NET Web Application (.NET Framework) C# Windows Cloud Web

Project name

1. Naziv projekta

Location

...

Solution

Solution name ⓘ

2. Postojeći Solution

Framework

.NET Framework 2.0

.NET Framework 3.0

.NET Framework 3.5

.NET Framework 4

.NET Framework 4.5

.NET Framework 4.5.1

.NET Framework 4.5.2

.NET Framework 4.6

.NET Framework 4.6.1

.NET Framework 4.7.2

3. Projekat: <=v4.7.0
Rad od kuće: mogu verzije i preko v4.7.0

Back


Create

Slika 2.


U poslednjem prozoru (Slika 3) Create a new ASP.NET Web Application, odabrati opciju Empty i potom Create.

Create a new ASP.NET Web Application


1.


Empty


An empty project template for creating ASP.NET applications. This template does not have any content in it.


Web Forms


A project template for creating ASP.NET Web Forms applications. ASP.NET Web Forms lets you build dynamic websites using a familiar drag-and-drop, event-driven model. A design surface and hundreds of controls and components let you rapidly build sophisticated, powerful UI-driven sites with data access.


MVC

A project template for creating ASP.NET MVC applications. ASP.NET MVC allows you to build applications using the Model-View-Controller architecture. ASP.NET MVC includes many features that enable fast, test-driven development for creating applications that use the latest standards.


Web API

A project template for creating RESTful HTTP services that can reach a broad range of clients including browsers and mobile devices.


Single Page Application

A project template for creating rich client side JavaScript driven HTML5 applications using ASP.NET Web API. Single Page Applications provide a rich user experience which includes client-side interactions using HTML5, CSS3, and JavaScript.

Authentication
No Authentication
[Change](#)

Add folders & core references
☐ Web Forms
☐ MVC
☐ Web API

Advanced
☒ Configure for HTTPS
☐ Docker support
(Requires [Docker Desktop](#))
☐ Also create a project for unit tests

Vezbe05-zadatak.Tests

Back

Create

2.

Slika 3.

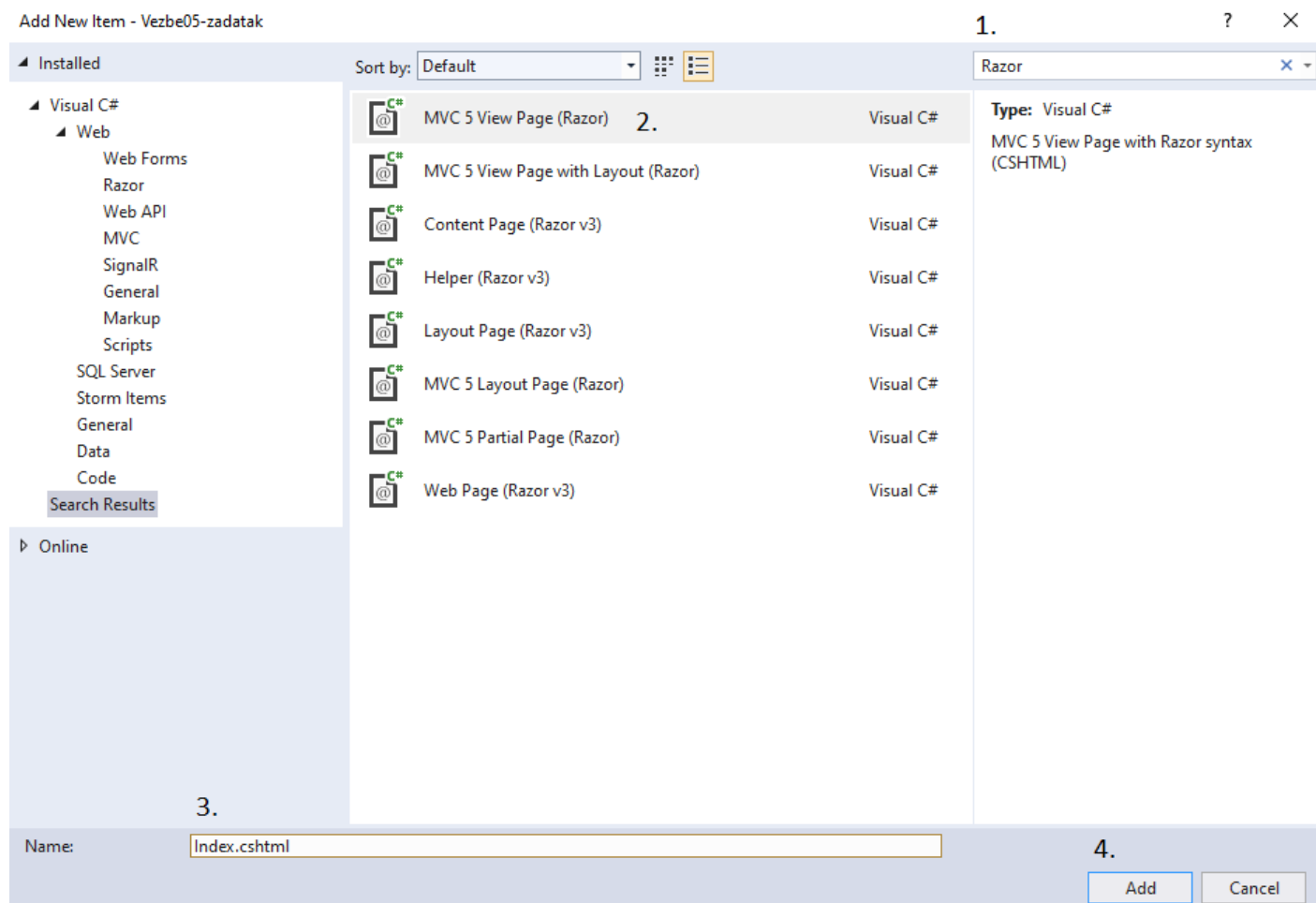
Struktura projekta

- Models - sadrži C# klase sa kojim predstavljamo entiteta u aplikaciji
- Views - sadrži .cshtml fajlove koje služe za prikaz
- packages.config - sadrži listu paketa koji se koriste u projektu. Omogućava NuGet package manageru da uvuče listu paketa ukoliko npr. projekat premestimo na drugi računar
- Web.config datoteka - sastoji se od opcija za podešavanja web aplikacije. Baziran na xmlu

Kreiranje Razor stranice

Prvo ćemo kreirati Index.cshtml stranicu. Potrebno je uraditi sledeće:

1. desni klik na projekat Vezbe05-zadatak>Add>New Item
2. u novom prozoru, pretražiti Razor i odabrati opciju MVC 5 View Page (Razor) Slika 4. Zadati naziv Index.cshtml i odabrati opciju Add.



Slika 4.

Predefinisane promenljive

Request

Objekat klase Request je povezan sa parametrima koji se prosleđuju zahtevom. Vrednostima parametra se pristupa putem operatora []. Parametri se

```
In [12]: @{
    Layout = null;
}

<!DOCTYPE html>

<html>
<head>
    <meta name="viewport" content="width=device-width" />
    <title></title>
</head>
<body>
    @{
        var message_get = Request["message_get"] != null ? Request["message_get"] : "";

        var message_post = Request["message_post"] != null ? Request["message_post"] : "";
        if (message_get != "")
        {
            <p style="color:green">Message @Request.HttpMethod: @message_get</p>
        }
        if (message_post != "")
        {
            <p style="color:green">Message @Request.HttpMethod: @message_post</p>
        }
        if (message_get == "" && message_post == "")
        {
            <p style="color:red">No new message</p>
        }
    }

    <div>
        Hello, World!

        <form method="post" action="~/Index.cshtml">
            <label>Type message:</label>
            <input name="message_post" />
            <input type="submit"/>
        </form>

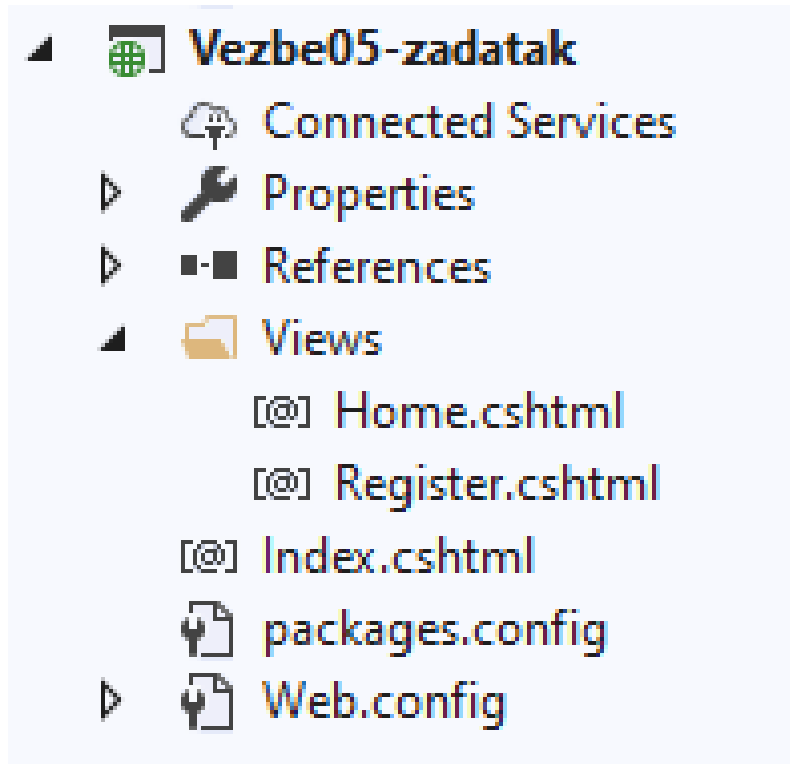
        <a href="~/Index.cshtml?message_get=Hello">Send message</a>
    </div>
```

```
</body>  
</html>
```

Session

Session objekat je povezan sa zahtevom i sadrži asocijativnu mapu objekata koji će pratiti ulogovanog korisnika.

Kreirati novi direktorijum u projektu Vezbe05-zadatak i dati mu naziv Views. Unutar ovog prozora kreirati Register.cshtml i Home.cshtml(Slika 5)



Slika 5.

Prekopirati sledeći kod u Register.cshtml

```
In [6]: @{\n    Layout = null;\n}\n\n<!DOCTYPE html>\n\n<html>\n<head>\n    <meta name="viewport" content="width=device-width" /\n    <title></title>\n</head>\n<body>\n    <div>\n        <h2>Register</h2>\n        <form action="../Views/Home.cshtml" method="post">\n            <table>\n                <tr>\n                    <td>Username:</td>\n                    <td><input type="text" name="username" /\n                </tr>\n                <tr>\n                    <td colspan="2">\n                        <input type="submit" value="Register" /\n                    </td>\n                </tr>\n            </table>\n        </form>\n    </div>\n</body>\n</html>
```

Kreirati Home.cshtml u View direktorijum i prekopirati sledeći kod:


```
In [8]: @{\n    Layout = null;\n}\n\n<!DOCTYPE html>\n\n<html>\n<head>\n    <meta name="viewport" content="width=device-width" /\n    <title></title>\n</head>\n<body>\n\n    @{\n        var loginUser = (string)Session["LOGGEDIN"];\n\n        if (loginUser == null)\n        {\n            Session["LOGGEDIN"] = String.Empty;\n            loginUser = String.Empty;\n        }\n\n        var user = Request["username"] != null && Request["username"] != "" ? Request["username"] : String.Empty;\n\n        /*\n        TODO: Dopuniti logiku za proveru prilikom logovanja\n        */\n        if (loginUser == String.Empty && user != String.Empty)\n        {\n            loginUser = user;\n            Session["LOGGEDIN"] = loginUser;\n        }\n\n    }\n    <div>\n        @if (loginUser != null && loginUser != String.Empty)\n        {\n            <p>Hello @loginUser!</p>\n        }\n        else
```

```
    {  
      <p>Nobody is logged in :(</p>  
    }  
  
  </div>  
</body>  
</html>
```

Application

Application objekat je na nivou aplikacije (ista vrednost je za različite korisnike aplikacije i u različitim veb čitačima). Prekopirati sledeći kod u Home.cshtml fajl.

```
In [1]: @{
    Layout = null;
}

<!DOCTYPE html>

<html>
<head>
    <meta name="viewport" content="width=device-width" />
    <title></title>
</head>
<body>

    @{
        List<string> users = (List<string>)HttpContext.Current.Application["USERS"];

        if (users == null)
        {
            users = new List<string>();
        }

        var loginUser = (string)Session["LOGGEDIN"];

        if (loginUser == null)
        {
            Session["LOGGEDIN"] = String.Empty;
            loginUser = String.Empty;
        }

        var user = Request["username"] != null && Request["username"] != "" ? Request["username"] : String.Empty;

        /*
        TODO: Dopuniti logiku za proveru prilikom logovanja
        */
        if (loginUser == String.Empty && user != String.Empty)
        {
            loginUser = user;
            Session["LOGGEDIN"] = loginUser;
            users.Add(loginUser);
            HttpContext.Current.Application["USERS"] = users;
        }
    }
}
```

```

    }
    <div>
        @if (loginUser != null && loginUser != String.Empty)
        {
            <p>Hello @loginUser!</p>
        }
        else
        {
            <p>Nobody is logged in :(</p>
        }

        <ol>
            Logged in users!
            @foreach(var u in users)
            {
                <li>@u</li>
            }
        </ol>

    </div>
</body>
</html>

```

Napomena: Primer radi bez logout-a. Neće dozvoliti da se još jedan korisnik u istoj sesiji uloguje i pregazi ulogovanog korisnika. Potrebno je implementirati logout opciju.

Pokušajte da primer pokrenete u drugom veb browser-u ili incognito prozoru. Šta će se desiti ako pokušate da pristupite:

localhost:port/Views/Home.cshtml stranici?

Ostale predefinisane promenljive pogledati još jednom na sledećem [linku \(https://canvas.ftn.uns.ac.rs/courses/2541/files/folder/Predavanja/4-ASP.NET?preview=27572\)](https://canvas.ftn.uns.ac.rs/courses/2541/files/folder/Predavanja/4-ASP.NET?preview=27572)

Zadatak

Treba da se kreiraju .cshtml stranice koje će inicijalno da služe za prikaz podataka i biznis logiku. Kasnije razdvojićemo ove dve uloge na .cshtml (View, prikaz) i .cs (Controller, biznis logika).

Potrebno je kreirati Models folder unutar kojeg treba da se nalaze potrebne klase za predstavljanje entiteta ove aplikacije. Jedna klasa je User tj. Korisnik. Klasa korisnik treba da ima neophodna polja koja predstavljaju korisnika vaše aplikacije.