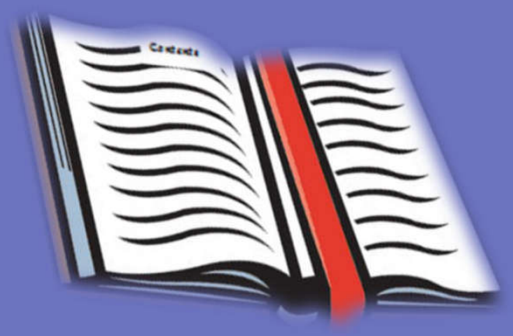
ASP.NET MVC Advanced Topics

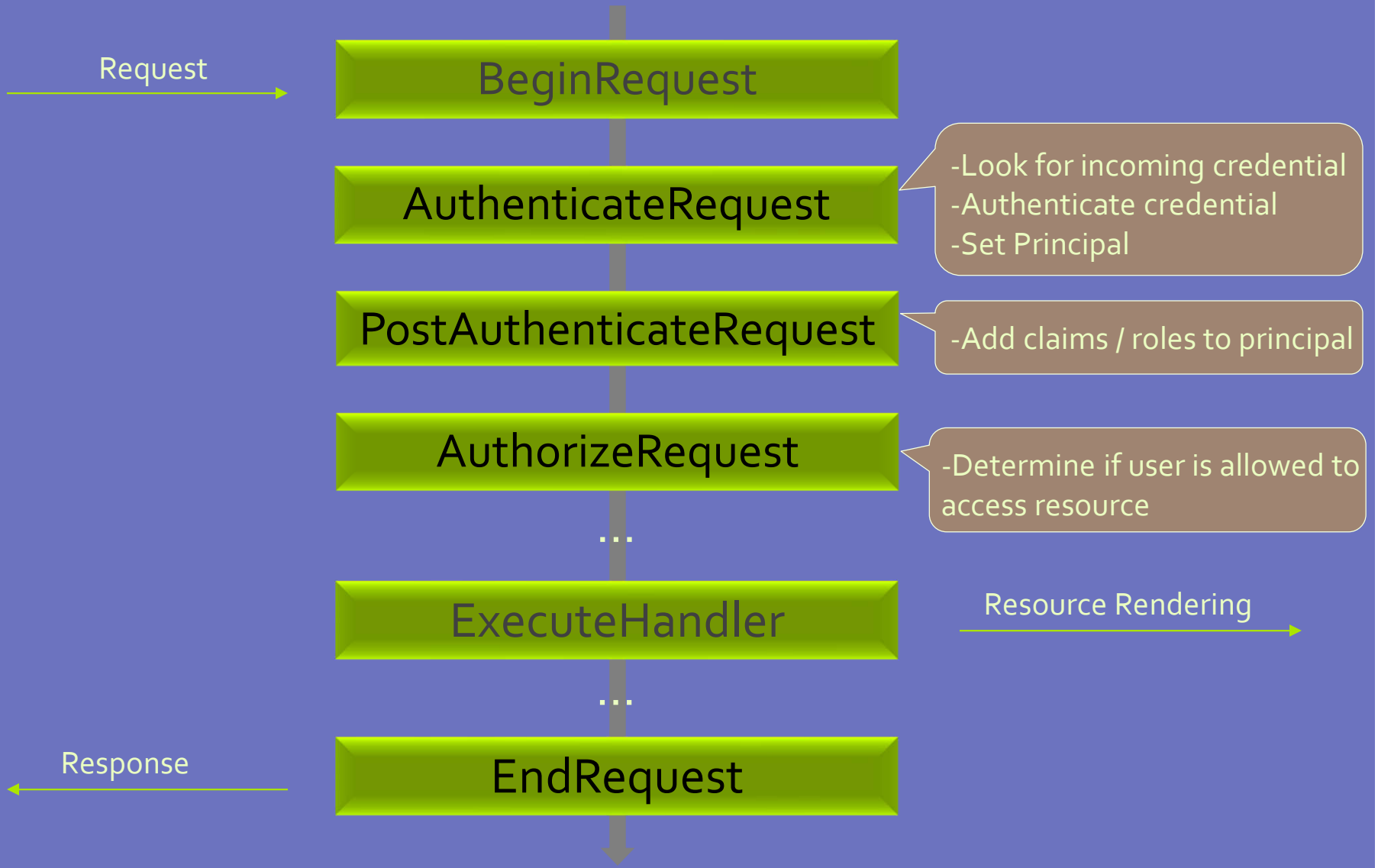
Authentication, Security, Configuration, Performance, Best Practices

Table of Contents

- Authentication and Authorization
- Security (CSRF and XSS)
- SimpleMembership
- Performance and Caching
- Localization and Resources
- Unit Testing
- Deployment and Configuration
- Good Practices



ASP.NET Pipeline

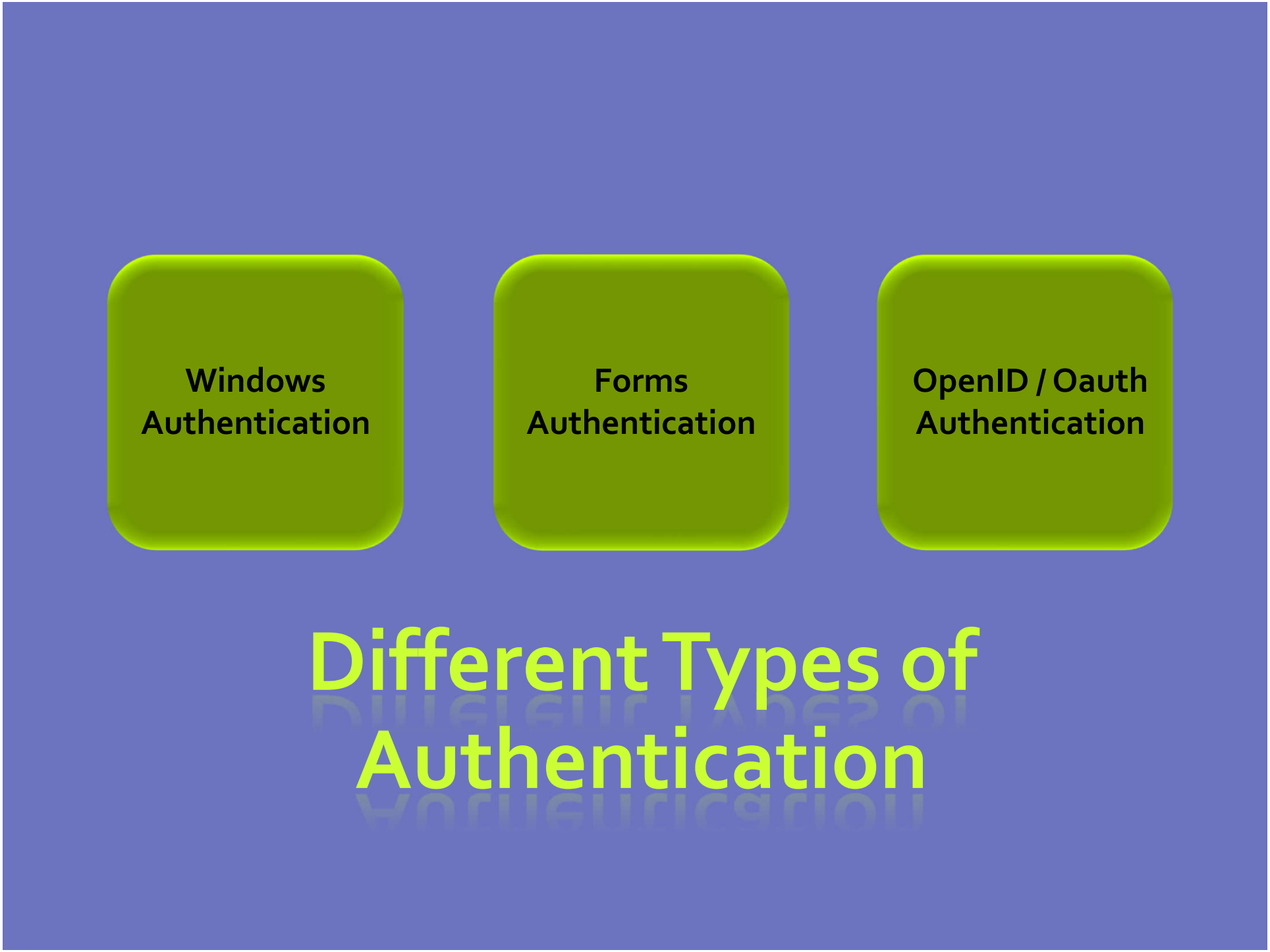


Authentication

What is Authentication?

Authentication

- Why we verify the identity of a user?
  - Bank account
  - Picture collection
  - Shows information specific to a user and track information that we want.
- The authentication type is set in the configuration file
- User.Identity


Windows Authentication
Forms Authentication
OpenID / Oauth Authentication
Different Types of Authentication

Windows authentication

- Typically used for Intranet Applications
  - Uses components and services from the OS
  - "Integrated authentication" – single sign on through Active Directory server
  - Works on variety of browsers
  - It is not recommended for Internet applications
    - Users from different domains
    - Users using different operating systems

Forms Authentication

- GET -> POST -> Redirect
  - Get a login or registration form
  - POST back the input to a controller action
  - If credentials are correct, redirect to another controller action (members area)
- Cookies – (.ASPXAUTH=…)
- Session – (.ASP.NET_SessionId=…)
- Secure socket layer - SSL

Forms Authentication

- Return the login form via GET request
- By default every Action method in ASP.NET MVC will handle requests via GET

Restricts action method so that it handles only HTTP GET requests

```
[HttpGet]
[AllowAnonymous]
public ActionResult Login(string returnUrl)
{
        ViewBag.ReturnUrl = returnUrl;
        return View();
}
```

Forms Authentication

- Process the POST request of the login form

```
[HttpPost]
[AllowAnonymous]
[RequireSSL]
public ActionRe
returnUrl)
{
        if (ModelS
        WebSecurit
        persistCoo
        {
                return RedirectToLocal(returnUrl);
        }
        ModelState.AddModelError("", "The user name or
        password provided is incorrect.");
        return View(model);
}
```

Restricts action method so that it handles only HTTP POST requests

This request must be executed through a secure socket layer

Redirect the user if the login was successful

Configure OpenID / OAuth

- Configuration takes place during application start

```
public static class AuthConfig
{
        public static void RegisterAuth()
        {
                //OAuthWebSecurity.RegisterMicrosoftClient(
                //    clientId: "",
                //    clientSecret: "");

                //OAuthWebSecurity.RegisterFacebookClient(
                //    appId: "",
                //    appSecret: "");

                //OAuthWebSecurity.RegisterGoogleClient();
        }
}
```

OpenID / OAuth

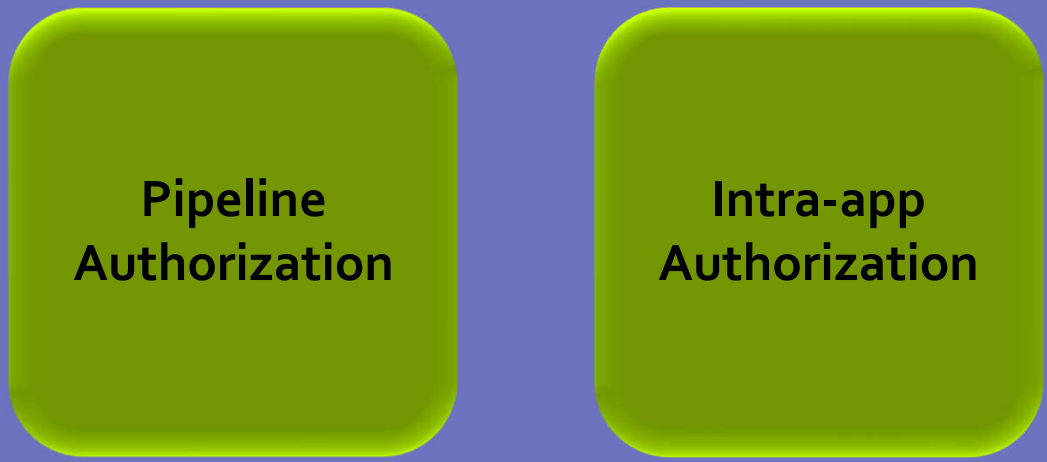
- DotNetOpenAuth library
- Authentication through external services
- Don't need to manage passwords
- Easier registration and authentication process
- Similar to Forms authentication
  - Cookies
  - Redirects

Authorization

Authorization management in ASP.NET MVC

Authorization and Roles

- Authorization is giving permissions
  - Give permission to see a specific page
  - Restrict someone to delete something
- Authorization can be done against
  - Anonymous users
  - Already registered user or group of users
  - Roles
- Authorization on a controller or an action
- Sets a cookie (.ASPXROLES=…)



Different approaches for Authorization

Pipeline Authorization

- URL authorization module
  - It is not recommended because it depends on a hardcoded path. MVC has powerful routing mechanism that can change the route and open security holes.

```
<location path="customers">
        <system.web>
                <authorization>
                        <allow roles="Technical Support" />
                        <deny users="*" />
                </authorization>
        </system.web>
</location>
```

Roles Authorization

- Roles in ASP.NET MVC

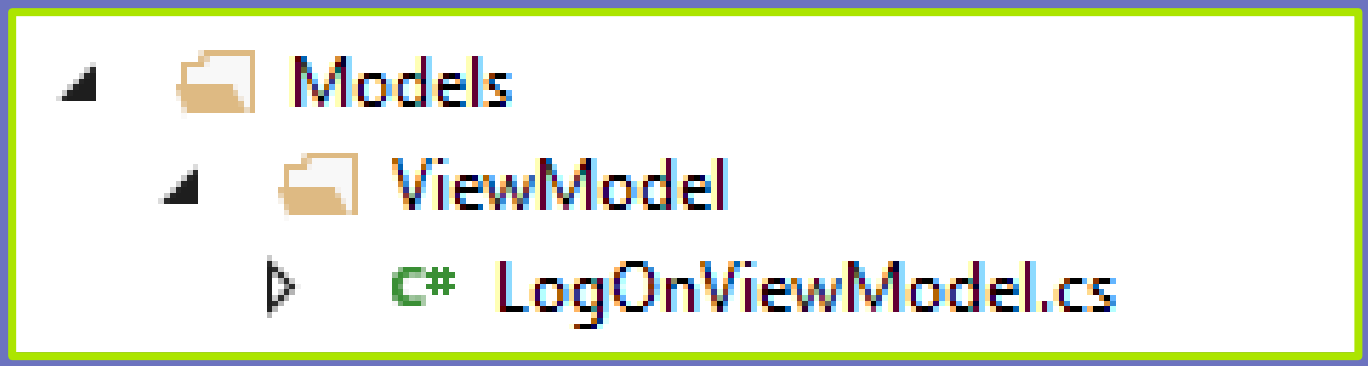
```
[AllowAnonymous]
public ActionResult Register()
{
      return View();
}

[Authorize(User="Niki")]
public ActionResult Register()
{
      return View();
}

[Authorize(Role="Administrator")]
public ActionResult Register()
{
      return View();
}
```

Form Authentication Example

- Web Shop:
  - Create ViewModel Folder in Model Folder
  - Create class LogOnViewModel.cs for Login



Form Authentication Example

- **Web Shop:**

```
namespace webBanHang.Models.ViewModel
{
    public class LogOnViewModel
    {
        [Display(Name="User name:")]
        [Required(ErrorMessage="Bạn chưa nhập username")]
        public string Username { get; set; }
        [DataType (DataType.Password)]
        [Display(Name="Password:")]
        [Required(ErrorMessage="Bạn chưa nhập password")]
        public string  Password { get; set; }
    }
}
```

Form Authentication Example

- Web Shop:
  - Create AccountController with actions:

```
public class AccountController : Controller
{
    private NorthwindCopyEntities db = new NorthwindCopyEntities();
    [HttpGet]
    public ActionResult LogOn()
    {
        return View();
    }
```

Form Authentication Example

- Web Shop:
  - Create AccountController with actions:

```
[HttpPost]
public ActionResult LogOn(LogOnViewModel m)
{
    if (ModelState.IsValid)
    {
        if (CheckUser(m.Username, m.Password))
        {
            FormsAuthentication.SetAuthCookie(m.Username, true);//lưu thông tin cookies
            return RedirectToAction("Index", "Account");
        }
        else ModelState.AddModelError("", "Thông tin chưa hợp lệ!");
    }
    return View(m);
}
public ActionResult Logout()
{
    FormsAuthentication.SignOut();
    return RedirectToAction("Index", "Account");
}
```

Form Authentication Example

- Web Shop:
  - Create View with LogOn action:

```
@model webBanHang.Models.ViewModel.LogOnViewModel
@{     ViewBag.Title = "LogOn";
    Layout = "~/Views/Shared/_LayoutAdmin.cshtml";}
<h2>FORM ĐĂNG NHẬP</h2>
@using (@Html.BeginForm("LogOn", "Account", FormMethod.Post))
{    @Html.ValidationSummary(true,"Đăng nhập không thành công, kiểm tra lại các thông tin!")
    <div><fieldset> <legend>Thông tin đăng nhập</legend>
            <div> @Html.LabelFor(m=>m.Username)<br />
                @Html.TextBoxFor(m=>m.Username)
                @Html.ValidationMessageFor(m=>m.Username)</div>
            <div> @Html.LabelFor(m => m.Password)<br />
                @Html.PasswordFor(m => m.Password)
                @Html.ValidationMessageFor(m => m.Password) </div>
            <div><input type="submit" value="Log On" /></div>
        </fieldset>
    </div>
}
```

Form Authentication Example

- Web Shop:
  - web.config file :

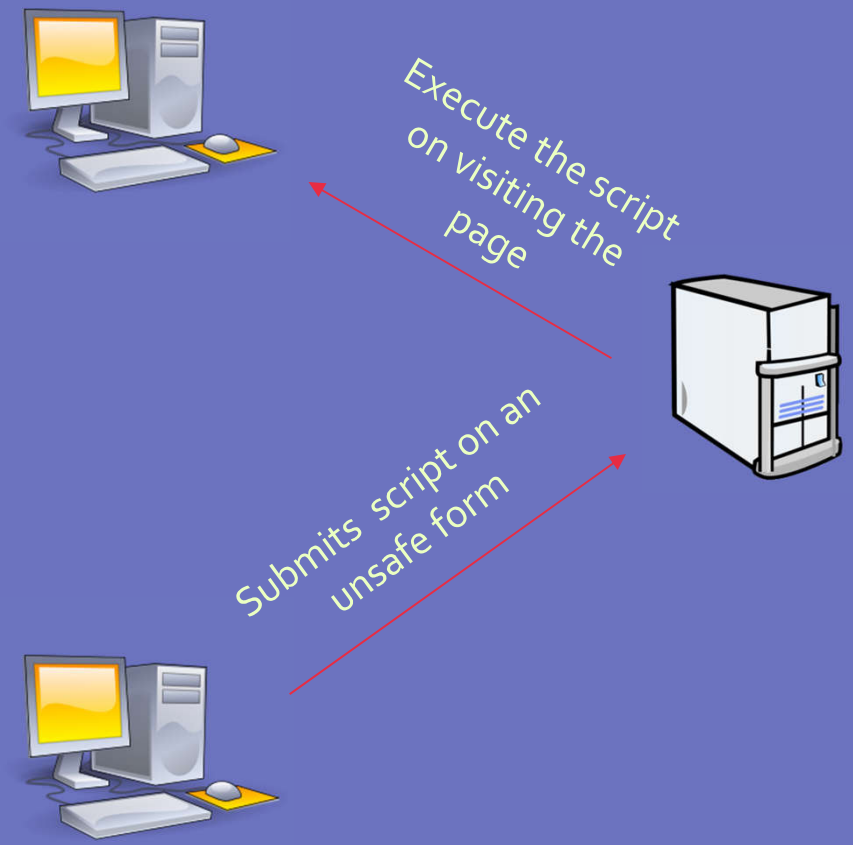
```
<system.web>
  <compilation debug="true" targetFramework="4.5" />
  <httpRuntime targetFramework="4.5" />
  <authentication mode="Forms">
    <forms loginUrl="~/Account/LogOn" timeout="60"></forms>
  </authentication>
</system.web>
```

Security

Cross-site scripting, cross-site request forgery and sql injection

XSS

- Cross-site scripting attack
  - Cookie theft
  - Account hijacking
  - Modify content
  - Modify user settings
  - Download malware
  - Submit CRSF attack
  - Password prompt

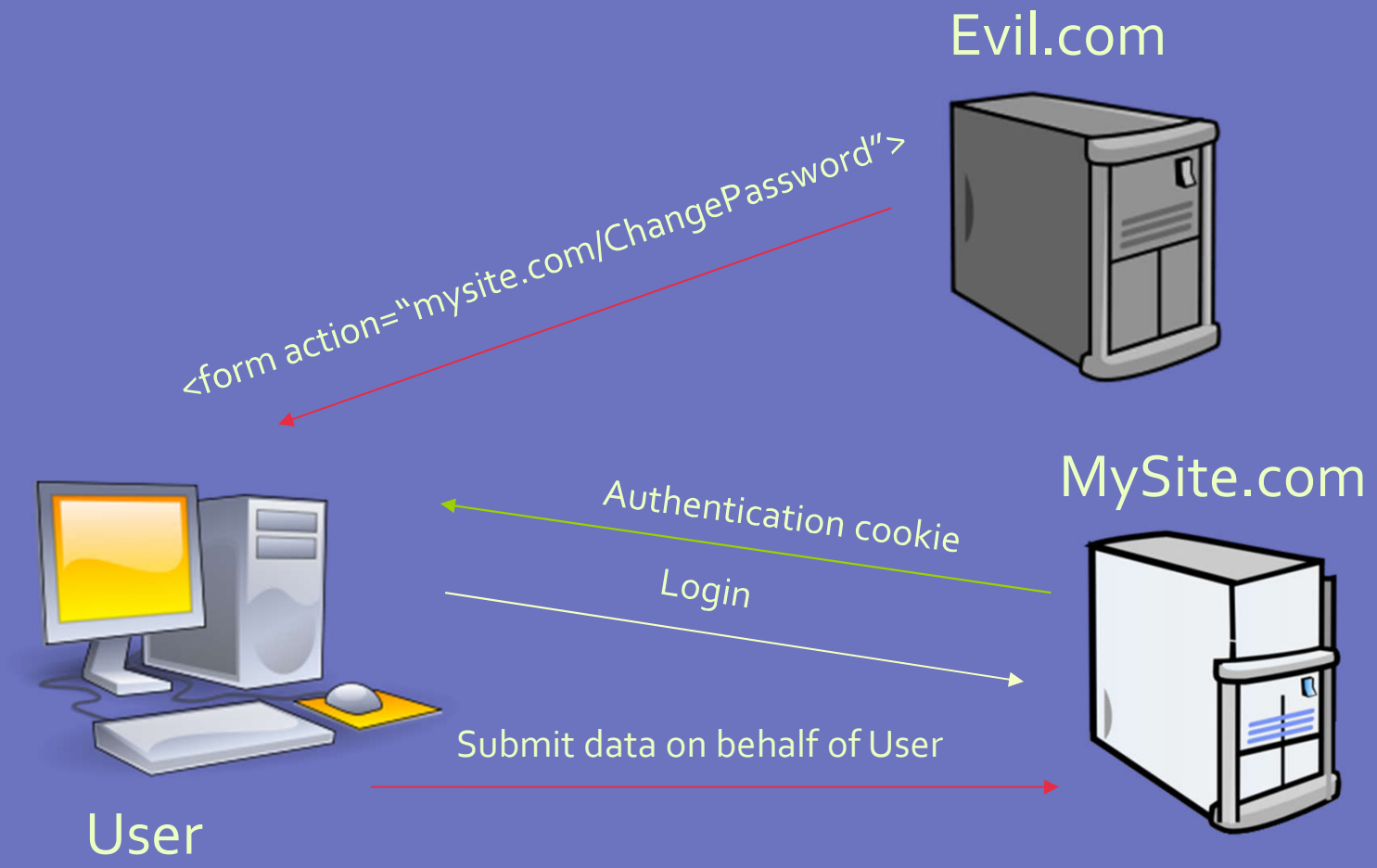


Protecting from XSS

- ASP.NET has automatic protection from submitting html content or scripts.
  - It can be disabled with [ValidateInput(false)]
  - [AllowHtml] on model property disables it.
- Razor view engine automatically html encode
  - Html.Raw() helper is used to show html content
  - Html.Encode() and Html.Decode()
- Some of the modern browsers may detect it
- Use approved libraries to submit Html-AntiXSS

CSRF

- **Cross-site request forgery attack**



Protect from CSRF

- Check if the submitted form came from our server

```
[HttpPost]
[ValidateAntiForgeryToken]
[Authorize]
public ActionResult ChangePassword()
{
        ChangePassword...
}

@using (Html.BeginForm("ChangePassword", "Account")) {
        @Html.AntiForgeryToken()
        @Html.ValidationSummary()
        <li>
                @Html.
                @Html.
        </li>
}
```

Prevents forgery of a request

Generates a hidden field(anti-forgery token) that is validated on form submission

SQL Injection

- Commands inserted into SQL where only data was expected

Expected user input

```
Select * from users where username = ’Niki’
;Delete from Users where username = ‘Niki’;. . .
```

Added as addition to the input

- Entity framework helps to prevent SQL injection

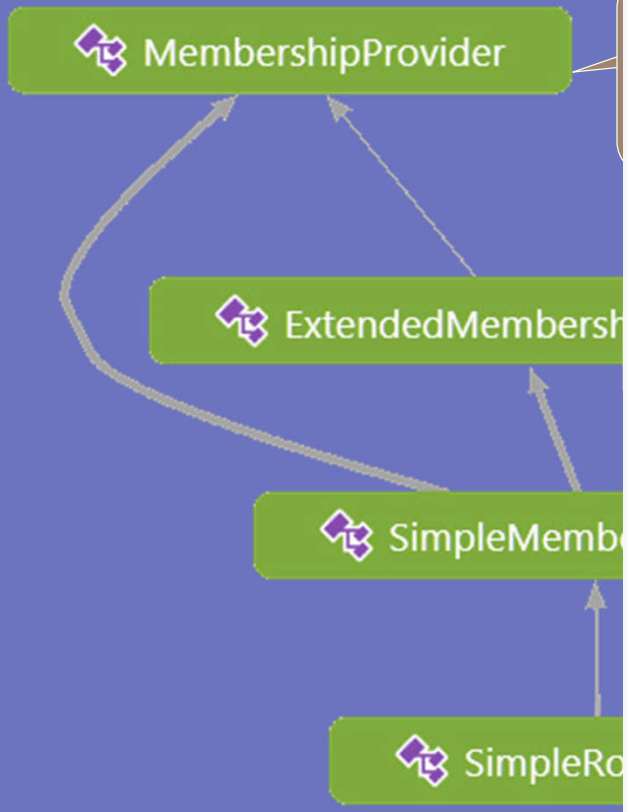
Security useful links

- http://haacked.com/archive/2009/06/25/json-hijacking.aspx
- http://en.wikipedia.org/wiki/Cross-site_request_forgery

Simple Membership

Membership system

- Membership classe



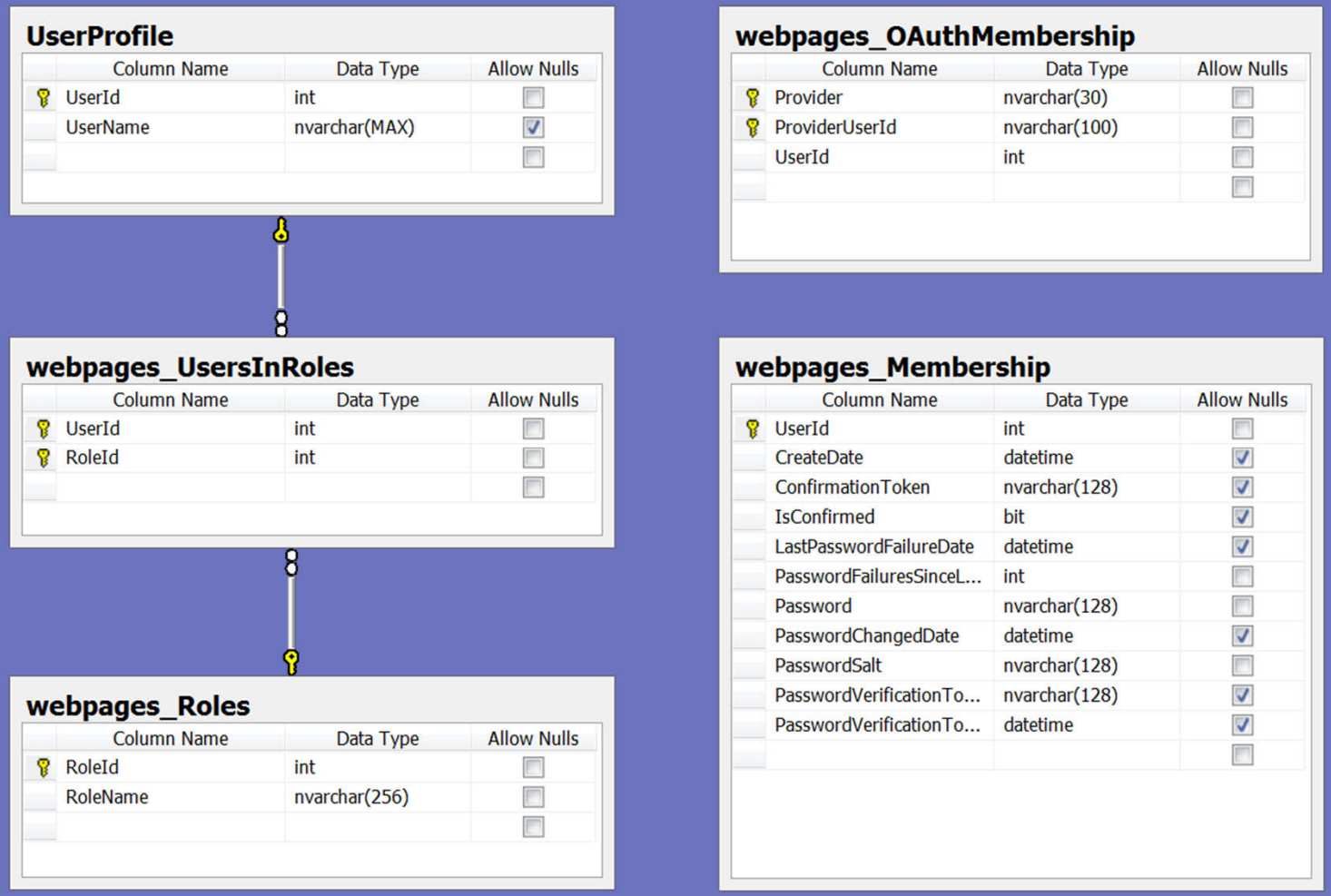
public abstract class **ExtendedMembershipProvider** : System.Web.Security.MembershipProvider
Member of WebMatrix.WebData

Summary:
Represents an abstract class that is used to extend the membership system that is provided by the System.Web.Security.MembershipProvider class.

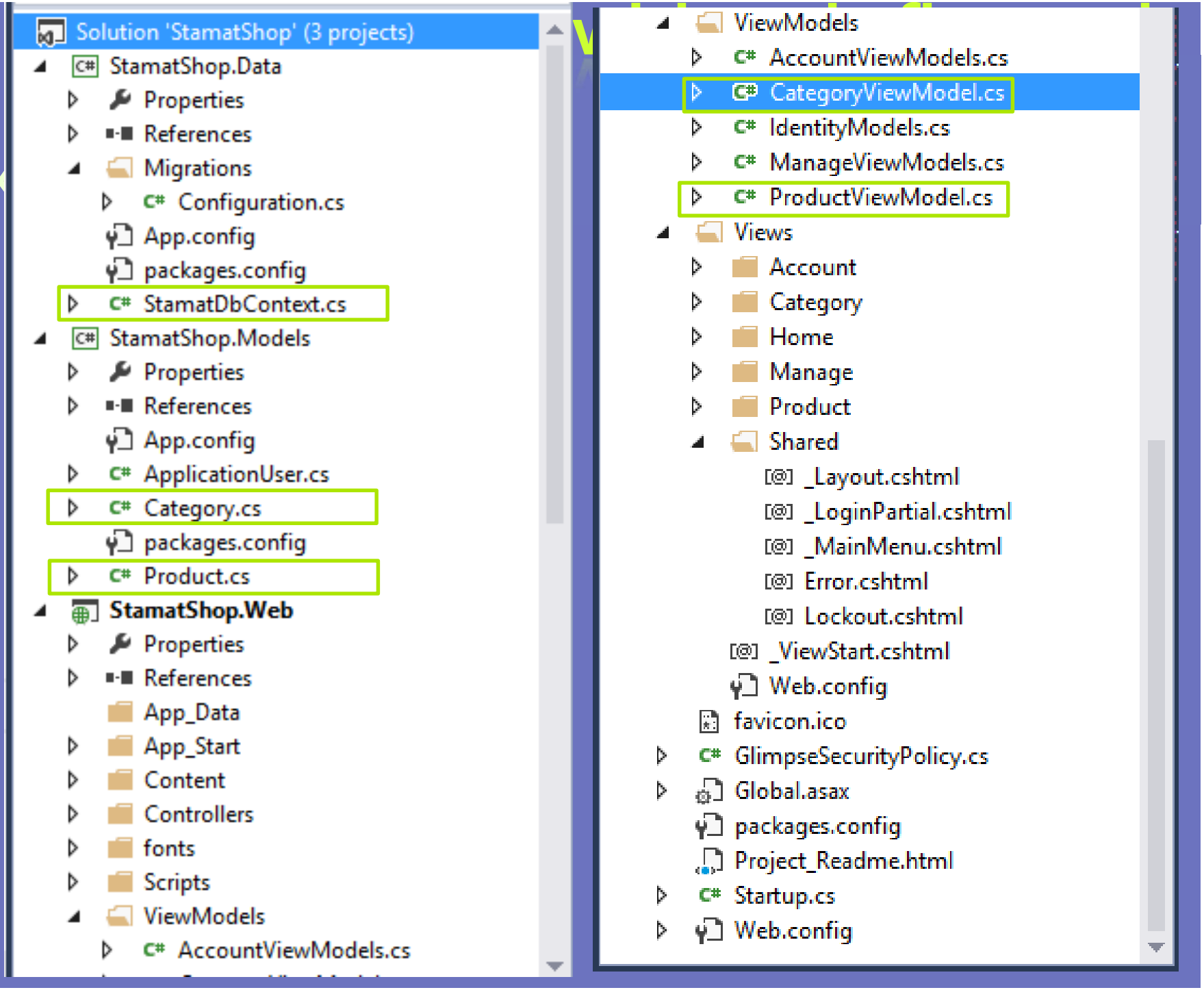
- { } Microsoft.Internal.Web.Utils
- { } WebMatrix.WebData
  - ExtendedMembershipProvider
    - ConfirmAccount(string)
    - ConfirmAccount(string, string)
    - CreateAccount(string, string, bool)
    - CreateAccount(string, string)
    - CreateOrUpdateOAuthAccount(string, string, string)
    - CreateUserAndAccount(string, string, bool, System.Collections.Generic.IDictionary<string,object>)
    - CreateUserAndAccount(string, string, System.Collections.Generic.IDictionary<string,object>)
    - CreateUserAndAccount(string, string, bool)
    - CreateUserAndAccount(string, string)
    - DeleteAccount(string)
    - DeleteOAuthAccount(string, string)
    - DeleteOAuthToken(string)
    - ExtendedMembershipProvider()
    - GeneratePasswordResetToken(string, int)
    - GeneratePasswordResetToken(string)
    - GetAccountsForUser(string)
    - GetCreateDate(string)
    - GetLastPasswordFailureDate(string)
    - GetOAuthTokenSecret(string)
    - GetPasswordChangedDate(string)
    - GetPasswordFailuresSinceLastSuccess(string)
    - GetUserIdFromOAuth(string, string)
    - GetUserIdFromPasswordResetToken(string)
    - GetUserNameFromId(int)
    - HasLocalAccount(int)
    - IsConfirmed(string)
    - ReplaceOAuthRequestTokenWithAccessToken(string, string, string)
    - ResetPasswordWithToken(string, string)
    - StoreOAuthRequestToken(string, string)

Simple membership schema

- Works with existing schema
  - It's easy to integrate it with existing Entity Model



- Ex


Solution 'StamatShop' (3 projects)
StamatShop.Data
Properties
References
Migrations
Configuration.cs
App.config
packages.config
StamatDbContext.cs
StamatShop.Models
Properties
References
App.config
ApplicationUser.cs
Category.cs
packages.config
Product.cs
StamatShop.Web
Properties
References
App_Data
App_Start
Content
Controllers
fonts
Scripts
ViewModels
AccountViewModels.cs
ViewModels
AccountViewModels.cs
CategoryViewModel.cs
IdentityModels.cs
ManageViewModels.cs
ProductViewModel.cs
Views
Account
Category
Home
Manage
Product
Shared
_Layout.cshtml
_LoginPartial.cshtml
_MainMenu.cshtml
Error.cshtml
Lockout.cshtml
_ViewStart.cshtml
Web.config
favicon.ico
GlimpseSecurityPolicy.cs
Global.asax
packages.config
Project_Readme.html
Startup.cs
Web.config

Project MVC with code first and multi project

- Example StamatShop:

```
namespace StamatShop.Models
{
    using System;
    using System.Collections.Generic;
    using System.Linq;

    public class Category
    {
        public Category()
        {
            this.Products = new HashSet<Product>();
        }

        public int Id { get; set; }
        public string Name { get; set; }
        public ICollection<Product> Products { get; set; }
    }
}
```

Project MVC with code first and multi project

- Example StamatShop:

```
namespace StamatShop.Models
{
    using System;
    using System.Collections.Generic;

    public class Product
    {
        public int Id { get; set; }
        public string Name { get; set; }
        public string Description { get; set; }
        public int Rating { get; set; }
        public decimal Price { get; set; }
        public virtual Category Category { get; set; }
    }
}
```

Project MVC with code first and multi project

- Example StamatShop:

```
namespace StamatShop.Data
{
    public class StamatDbContext : IdentityDbContext<ApplicationUser>
    {
        public StamatDbContext()
            : base("DefaultConnection", throwIfV1Schema: false)
        {
            Database.SetInitializer(
                new MigrateDatabaseToLatestVersion<StamatDbContext, Configuration>());
        }
        public static StamatDbContext Create()
        {
            return new StamatDbContext();
        }

        public IDbSet<Product> Products { get; set; }
        public IDbSet<Category> Categories { get; set; }
    }
}
```

Project MVC with code first and multi project

- Example StamatShop:

```
namespace StamatShop.Web.ViewModels
{
    using System;
    using System.Collections.Generic;
    using System.Linq;

    public class CategoryViewModel
    {
        public int Id { get; set; }
        public string Name { get; set; }
    }
}
```

Project MVC with code first and multi project

- Example StamatShop:

```
namespace StamatShop.Web.ViewModels
        public static Expression<Func<Product, ProductViewModel>> FromModel()
        {
            return x => new ProductViewModel()
            {
                Id = x.Id,
                Name = x.Name,
                Description = x.Description,
                Price = x.Price,
                Rating = x.Rating,
                Category = new CategoryViewModel
                {
                    Id = x.Category.Id,
                    Name = x.Category.Name
                }
            };
        }
    }
}
        public CategoryViewModel Category { get; set; }
```

Performance

Optimizing ASP.NET MVC application

Disable unused view engines

- Disable unused view engines
  - Global.asax
    - ViewEngines.Engines.Clear();
    - ViewEngines.Engines.Add(new RazorViewEngine());
- When accessing data via LINQ rely on IQueryable
- Use caching

Bundles and Minification

- Bundling – concatenating multiple files into a single download
- Minification – making the download file as small as possible
- Decrease page load times
- System.Web.Optimization
  - WebGrease.dll and Antlr3.Runtime.dll
- Measure time for getting all resources for a certain page with browser Dev. tools or Fiddler

Bundles

- Introduced in ASP.NET MVC 4
- Concatenating files into a single file – browsers supports limited concurrent requests ~ 6
- Minifies files
- Validating the code in the JavaScript files
- Sprites any background images in CSS files
- Manually through the console application:

[Full Path..]\WebGrease.1.3.0\tools>WG.exe -b -in:.\scripts -out:.\bscripts.js – Create a bundle

Bundles in ASP.NET MVC

- Configure bundles
  - Add bundles to the global bundle table
  - Specify a global virtual path
    - Be careful with relative images paths
  - Include the files in the bundle.
    - Use wildcards (*) to avoid issues with file versions
- Register bundle table during application startup
  - <compilation debug="true" />
  - BundleTable.EnableOptimization = true;

Scripts and styles bundles

- Adding bundles in the Bundle Table

```
bundles.Add(new ScriptBundle("~/bundle/jquery").Include(
        "~/Scripts/jquery-{version}.js"));
));
bu                                                    o").Include(
        "~/Scripts/kendo/kendo.ui.                    ",
        "~/Scripts/kendo/kendo.                       ure.bg.js",
));
 bundles.Add(new StyleBundle("~/content/kendo").Include(
        "~/Content/kendo/kendo.common                 ",
        "~/Content/kendo/kendo.                       d.css",
        "~/Content/kendo/kendo.                       ked.css"
 ));
 BundleTable.EnableOptimization = true;
```

Bundle table

Use wildcards and {version}

Virtual path for the bundle

Virtual path for the bundle

Enable / Disable optimization

Rendering Bundles

- Rendering bundles in ASP.NET MVC

```
@Scripts.Render("~/bundle/jquery");
@Scripts.Render("~/bundle/kendo")
<link
href="@Bundle.B
rel="stylesheet
@Styles.Render(
@Scripts.Render

<script
        src="/bundles/modernizr?v=jmdBhqkI3eMaPZJduAyIYBj
        7MpXrGd2ZqmHAOSNeYcg1">
</script>
```

Lives inside System.Web.Optimization so we need to include it in web.config

Magic string value helps to check changes in the bundle to avoid chaching

http://channel9.msdn.com/Series/mvcConf/mvcConf-2-Steven-Smith-Improving-ASPNET-MVC-Application-Performance

Caching

Output cache

- OutputCache action filter
  - Use as attribute on action or controller
  - Specify Duration and VaryByParam
  - Configurable with cache profiles in web.config
- Don't use OutputCache on views in APS.NET MVC

```
Public class CachedController : Controller
{
        [OutputCache(Duration=60, VaryByParam="none")]
        public ActionResult Index()
        {
                Return View();
        }
}
```

OutputCache properties

Attribute	Description
CacheProfile	Associates a response with a group of output-caching settings specified in the *web.config* file.
Duration	The time, in seconds, that the response is cached.
Location	Specifies the location (browser, proxy, or server) to store the response of the method call. The attribute takes its value from the *OutputCacheLocation* enumeration.
NoStore	Indicates whether to send a *Cache-Control:no-store* header to prevent browser-side storage of the response.
SqlDependency	Indicates a dependency on the specified table on a given Microsoft SQL Server database. Whenever the contents of the table changes, the response is removed from the cache.

OutputCache properties (2)

Attribute	Description
VaryByContentEncoding	Content encoding by which you intend to differentiate cached responses.
VaryByCustom	A semicolon-separated list of strings that lets you maintain distinct cached copies of the response based on the browser type or user-defined strings.
VaryByHeader	A semicolon-separated list of HTTP headers.
VaryByParam	A semicolon-separated list of strings representing query string values sent with GET method attributes, or parameters sent using the POST method.

- OutputCache action filter

Localization and Resources

Localization and Culture

- **Internationalization** involves **Globalization** and **Localization**.
- **Globalization** is the process of designing applications that support different cultures.
- **Localization** is the process of customizing an application for a given culture.

One Code Base

Types of Culture

- Invariant
- Neutral
- Specific

Invariant Culture

- Should be used for storing data in a culture independent way
- Should not be used for User Interface elements
- Has no country/region

Eg:

```
CultureInfo ci = CultureInfo.InvariantCulture
```

Neutral Culture

- Associated with a language not with country/region
- Can be used for UI related options
- Cannot be used for retrieving information such as date/time formatting
- Specified using <languagecode2> format:
  - Arabic – "ar"
  - Exceptions – zh-CHT, zh-CH

Eg.

```
using System.Globalization;
…
CultureInfo ci = new CultureInfo("fr");
```

Specific Culture

- Associated with Language *and* a Country/Region
  - fr → Neutral Culture
  - fr-FR → Specific Culture
- Provides additional information about the date, time, currency and number formatting options
- Can be used wherever a Neutral culture is used, the opposite is not true

Eg.

```
using System.Globalization;
…
CultureInfo ci = new CultureInfo("fr-FR");
```

Numeric Formats

Culture	Format	Result
en-US	N	123,456.78
fr-FR	N	123 456,78
hi-IN	N	1,23,456.78
"" (invariant)	R	123456.78

- Store as binary data type if possible
  - Integer, decimal, floating-point
- Invariant storage as text
  - Format using CultureInfo.InvariantCulture
  - Use standard format character "R" (reversible) for floating point numbers

Currencies

```
decimal dec = decimal.Parse("$1000000.23",
NumberStyles.Currency,CultureInfo.CurrentCulture);
System.Console.WriteLine("{0:C}", dec);
```

```
Output: $1,000,000.23
```

- Preferably store as decimal with meta data
  - Culture
  - DateTime
- Use 3rd party service for conversion
- When storing as text use invariant culture
  - Reversible text floating-point format
  - Currency text format: ¤1,000,000.23

CultureInfo class

- provides culture specific information
- Controls date comparison string comparison number comparison, etc

CultureInfo
userculture=Thread.CurrentThread.CurrentCulture

(Used for calculation and Internal Manipulation)

CultureInfo
userculture=Thread.CurrentThread.CurrentUICulture

(Used for DisplayPurpose)

CurrentCulture Thread

- Date and number formatting
- String comparison and casing
- It determines the results of culture dependant functions.
- You can define the CurrentCulture object with specific cultures and not with neutral cultures.

CurrentUICulture Thread

- It determines which resources are loaded by the Resource Manager if you have provided resources in multiple languages.
- Because this controls only which language is used you can define CurrentUICulture with either neutral or specific cultures.

Changing the Culture programmatically

```
CultureInfo ci = new CultureInfo(culture);
 // set culture for formatting
Thread.CurrentThread.CurrentCulture = ci;
// set culture for resources
Thread.CurrentThread.CurrentUICulture = ci;
```

Implementing localization

- When the Localizable property is set to true, the resource file resX is generated form the form.

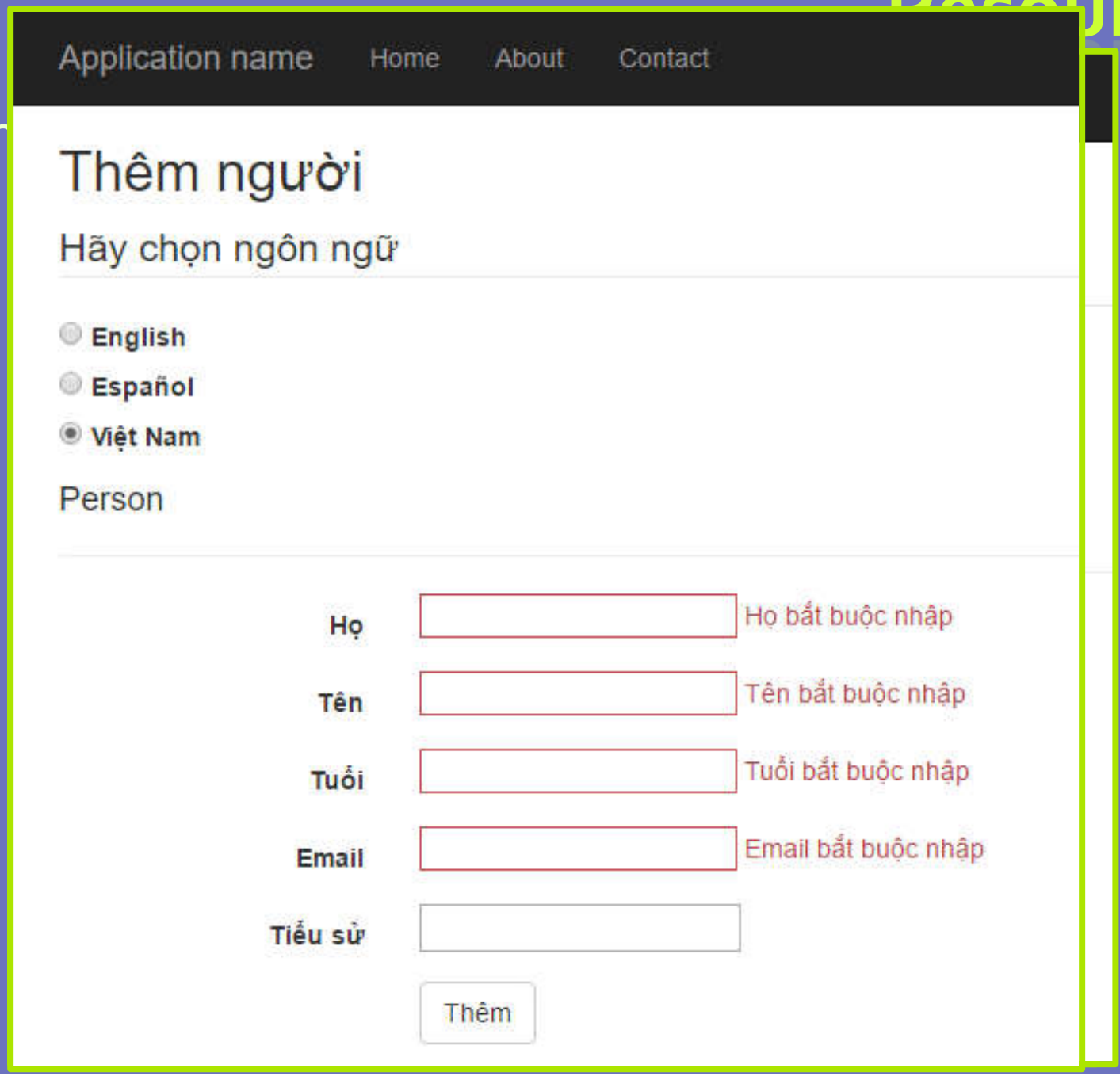
  BookOfTheDay.resX

- change the Language property of the form and the properties of some form elements, a new resource file is generated for the specified language.

  BookOfTheDayForm.de.resX

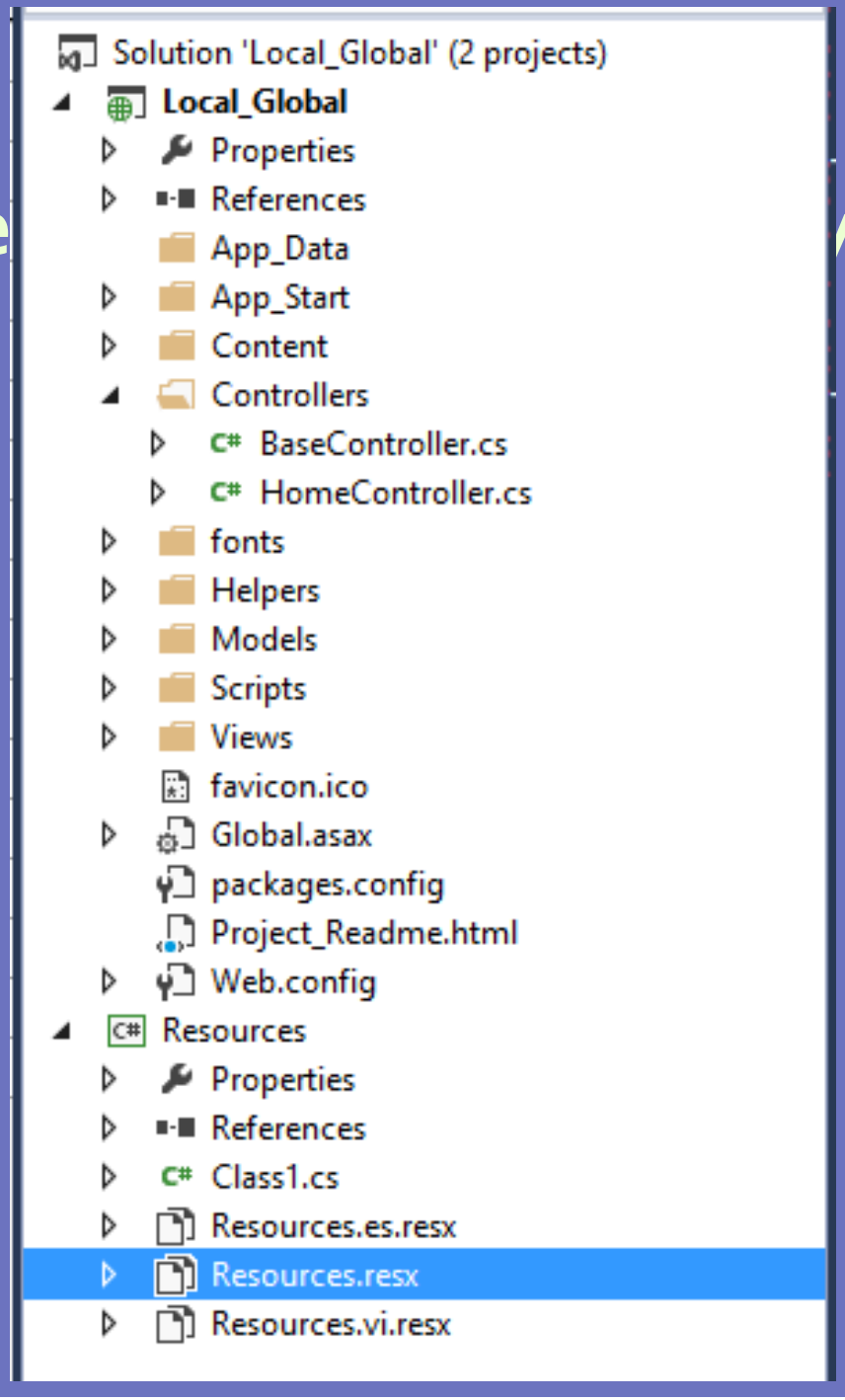
Resources

- Exar
  appl



Resources

Create ne): Resources



Resources

Create

Resou

Resources.resx

Strings | Add Resource | Remove Resource | Access Modifier: Public

	Name	Value
	AddPerson	Add Person
	Age	Age
	AgeRange	Must be between 10 to 130
	AgeRequired	Age is required
	Biography	Biography
	ChooseYourLanguage	Choose Your Language
	Create	Create
	Email	Email
	EmailInvalid	Email is not valid
	EmailRequired	Email is required
	FirstName	First name
	FirstNameLong	Must be less than 50 characters
	FirstNameRequired	First Name is Required
	LastName	Last name
	LastNameLong	Must be less than 50 characters
	LastNameRequired	Last name is required
*		

Resources

Create

Resour

Resources.es.resx

Strings | Add Resource | Remove Resource | Access Modifier: Public

Name	Value
AddPerson	Agregar persona
Age	Años
AgeRange	Debe estar entre 10-130
AgeRequired	Se requiere Edad
Biography	Biografía
ChooseYourLanguage	Elige tu idioma
Create	Crear
Email	Email
EmailInvalid	El correo no es válido
EmailRequired	correo electrónico es requerido
FirstName	Nombre de pila
FirstNameLong	Debe ser inferior a 50 caracteres
FirstNameRequired	Se requiere el primer nombre
LastName	Apellido
LastNameLong	Debe ser inferior a 50 caracteres
LastNameRequired	Se requiere el apellido

Resources

Create

Resou

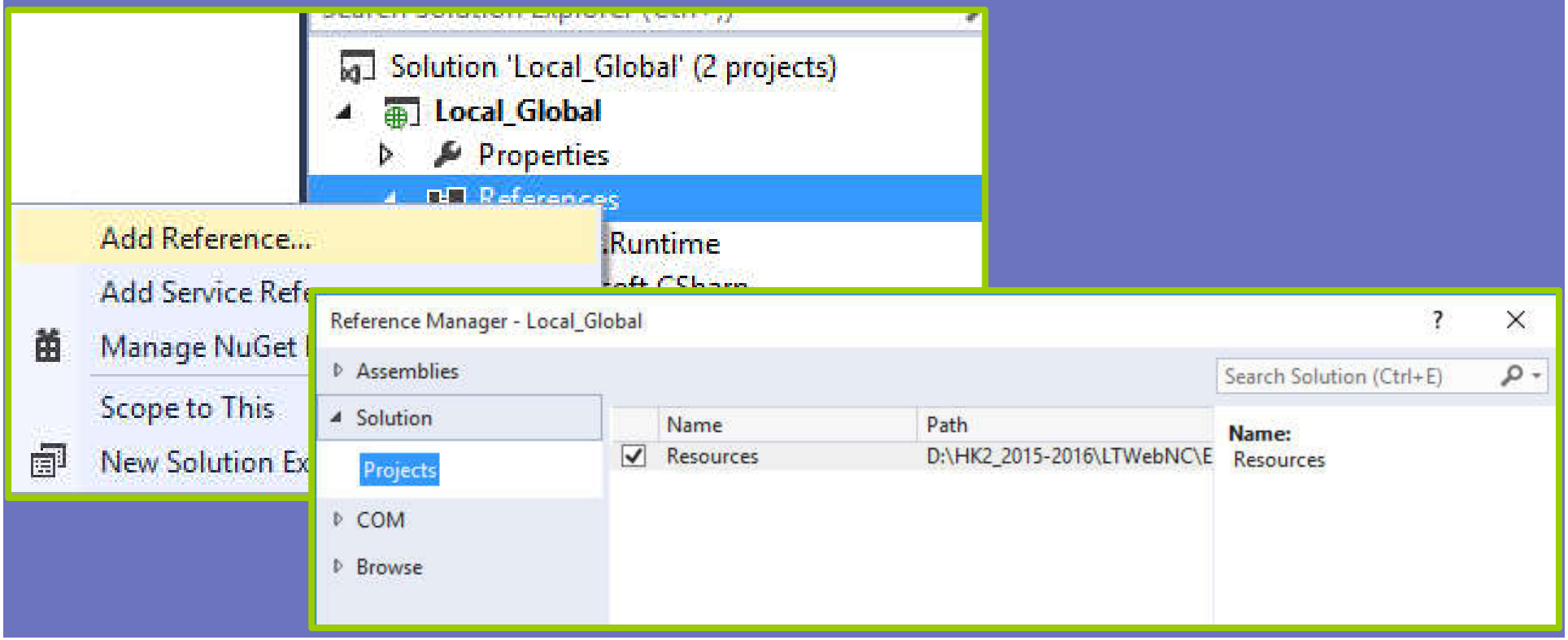
Resources.vi.resx

Strings | Add Resource | Remove Resource | Access Modifier: Public

Name	Value
AddPerson	Thêm người
Age	Tuổi
AgeRange	Tuổi phải từ 10 đến 130
AgeRequired	Tuổi bắt buộc nhập
Biography	Tiểu sử
ChooseYourLanguage	Hãy chọn ngôn ngữ
Create	Thêm
Email	Email
EmailInvalid	Email chưa hợp lệ
EmailRequired	Email bắt buộc nhập
FirstName	Họ
FirstNameLong	Phải ít hơn 50 ký tự
FirstNameRequired	Họ bắt buộc nhập
LastName	Tên
LastNameLong	Phải ít hơn 50 ký tự
LastNameRequired	Tên bắt buộc nhập

Resources

Add References for website: Right click on "**References**" under our web project "**Local_Global**", and choose the "**Resources**" project from Projects tab



Resources

Create Class Person in Model Folder:

```
public class Person
{
        public string FirstName { get; set; }
        public string LastName { get; set; }
        public int Age { get; set; }
        public string Email { get; set; }
        public string Biography { get; set; }
}
```

```
public class Person
{   [Display(Name = "FirstName", ResourceType = typeof(Resources.Resources))]
    [Required(ErrorMessageResourceType = typeof(Resources.Resources),
              ErrorMessageResourceName = "FirstNameRequired")]
    [StringLength(50, ErrorMessageResourceType = typeof(Resources.Resources),
                      ErrorMessageResourceName = "FirstNameLong")]
    public string FirstName { get; set; }
    [Display(Name = "LastName", ResourceType = typeof(Resources.Resources))]
    [Required(ErrorMessageResourceType = typeof(Resources.Resources),
              ErrorMessageResourceName = "LastNameRequired")]
    [StringLength(50, ErrorMessageResourceType = typeof(Resources.Resources),
                      ErrorMessageResourceName = "LastNameLong")]
    public string LastName { get; set; }
    [Display(Name = "Age", ResourceType = typeof(Resources.Resources))]
    [Required(ErrorMessageResourceType = typeof(Resources.Resources),
              ErrorMessageResourceName = "AgeRequired")]
    [Range(10, 130, ErrorMessageResourceType = typeof(Resources.Resources),
                    ErrorMessageResourceName = "AgeRange")]
    public int Age { get; set; }
    [Display(Name = "Email", ResourceType = typeof(Resources.Resources))]
    [Required(ErrorMessageResourceType = typeof(Resources.Resources),
              ErrorMessageResourceName = "EmailRequired")]
    [RegularExpression(".+@.+\\..+", ErrorMessageResourceType = typeof(Resources.Resources),
                                     ErrorMessageResourceName = "EmailInvalid")]
    public string Email { get; set; }
    [Display(Name = "Biography", ResourceType = typeof(Resources.Resources))]
    public string Biography { get; set; }
}
```

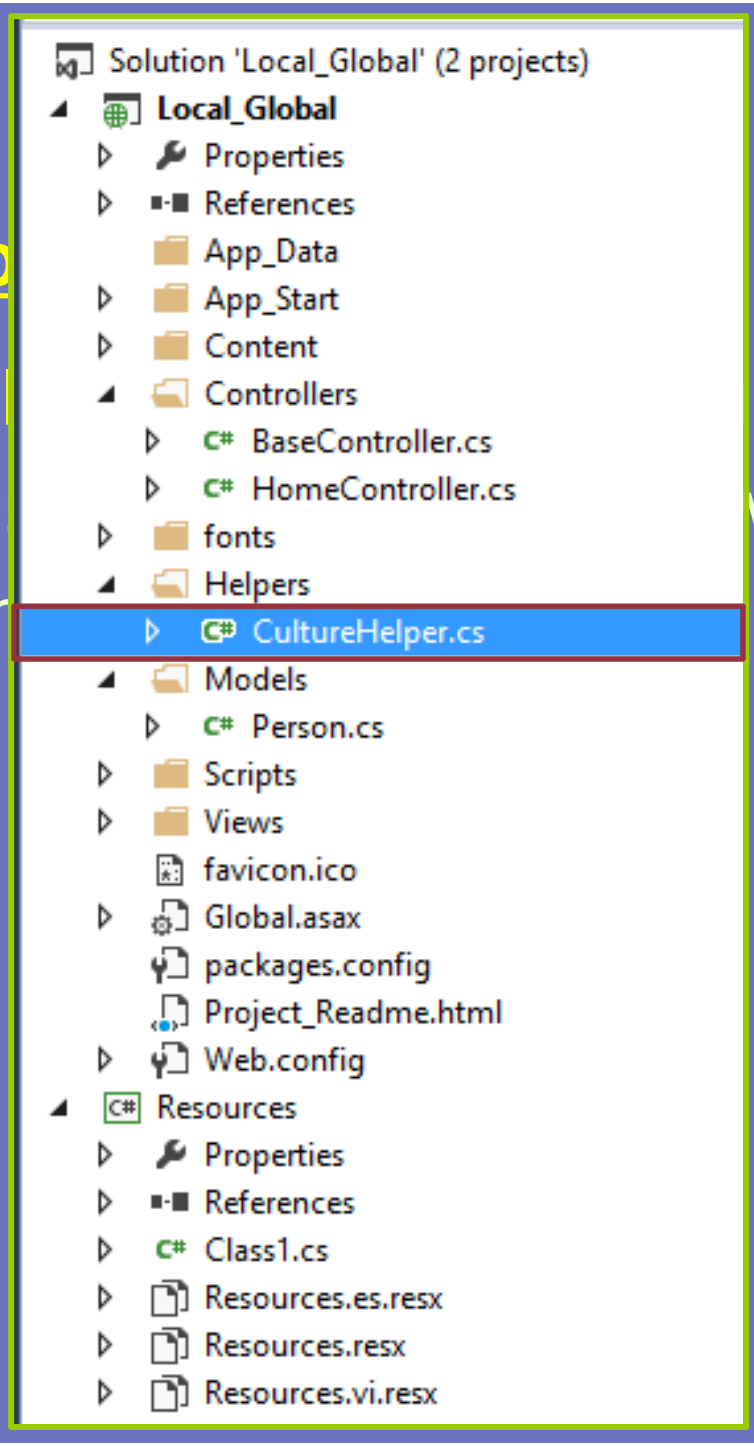
Resources

Create BaseController: Controller and override BeginExecuteCore method

```
protected override IAsyncResult BeginExecuteCore(AsyncCallback callback, object state)
{
    string cultureName = null;
    // Đọc dữ liệu từ cookies
    HttpCookie cultureCookie = Request.Cookies["_culture"];
    if (cultureCookie != null)
        cultureName = cultureCookie.Value;
    else
        cultureName = null;
    // kiểm tra hợp lệ tên culture
    cultureName = CultureHelper.GetImplementedCulture(cultureName); // This is safe
    // thay đổi culture
    Thread.CurrentThread.CurrentCulture = new System.Globalization.CultureInfo(cultureName);
    Thread.CurrentThread.CurrentUICulture = Thread.CurrentThread.CurrentCulture;
    return base.BeginExecuteCore(callback, state);
}
```

Resources

- **CultureHelp**
- The **Culture**[illegible]tility that allows us to [illegible]ve are implementir[illegible]



Resources

- **CultureHelper** Class
- The **CultureHelper** is basically a utility that allows us to store culture names we are

```
public static class CultureHelper
{
    // các culture
    private static readonly List<string> _cultures = new List<string> {
    "en-US",  // first culture is the DEFAULT
    "es", // Spanish NEUTRAL culture
    "vi" }; // Viet Nam NEUTRAL culture
    /// phương thức trả về tên culture nếu hợp lệ, nếu không trả về culture mặc định (en-US)
    /// <param name="name" />Culture's name (VD: en-US)</param>
    public static string GetImplementedCulture(string name)
    {
        // kiểm tra name khác null
        if (string.IsNullOrEmpty(name))
            return GetDefaultCulture(); // trả về culture mặc định (en-US)
        // nếu có trong danh sách culture trả về tên culture
        if (_cultures.Where(c => c.Equals(name,
            StringComparison.InvariantCultureIgnoreCase)).Count() > 0)
            return name; // accept it
```

Resources

- CultureHelper Class

```
        //trả về NeutralCulture có 2 ký tự đầu VD: en-GB, en-US thì trả về en-US
        var n = GetNeutralCulture(name);
        foreach (var c in _cultures)
            if (c.StartsWith(n))
                return c;
        // else It is not implemented
        return GetDefaultCulture(); // trả về culture mặc định (en-US)
    }
    // phương thức trả về culture mặc định (en-US)
    public static string GetDefaultCulture()
    {
        return _cultures[0]; // return Default culture
    }
    //phương thức trả về NeutralCulture
    public static string GetNeutralCulture(string name)
    {
        if (!name.Contains("-")) return name;
        return name.Split('-')[0]; // Read first part only. E.g. "en", "es"
    }
}
```

Resources

- CultureHelper Class

```
        //trả về NeutralCulture có 2 ký tự đầu VD: en-GB, en-US thì trả về en-US
        var n = GetNeutralCulture(name);
        foreach (var c in _cultures)
            if (c.StartsWith(n))
                return c;
        // else It is not implemented
        return GetDefaultCulture(); // trả về culture mặc định (en-US)
    }
    // phương thức trả về culture mặc định (en-US)
    public static string GetDefaultCulture()
    {
        return _cultures[0]; // return Default culture
    }
    //phương thức trả về NeutralCulture
    public static string GetNeutralCulture(string name)
    {
        if (!name.Contains("-")) return name;
        return name.Split('-')[0]; // Read first part only. E.g. "en", "es"
    }
}
```

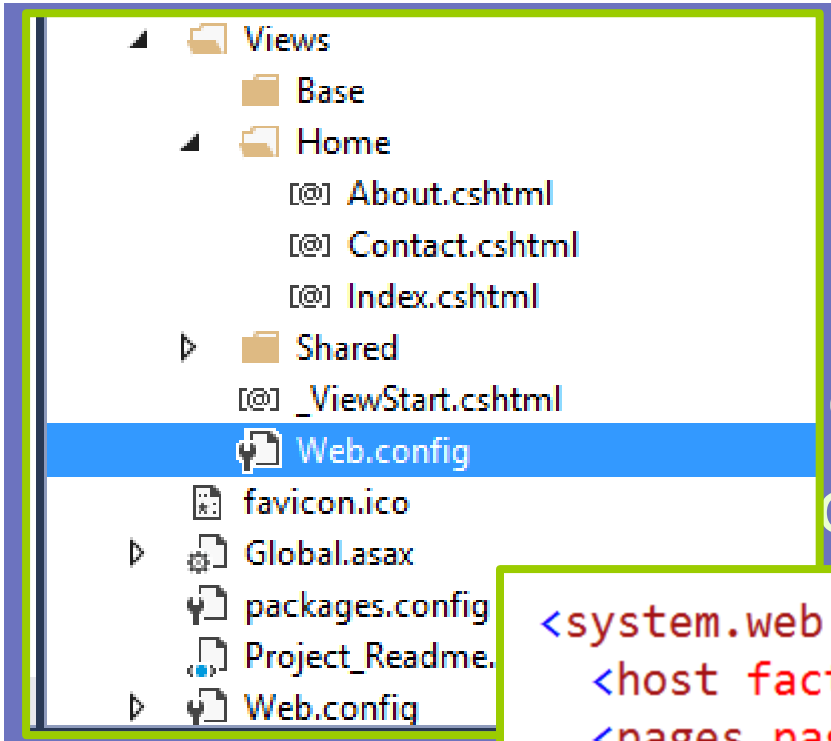
Create

urces

```
public class HomeController : BaseController
{
    [HttpGet]
    public ActionResult Index()
    {
        return View();
    }
    [HttpPost]
    public ActionResult Index(Person per)
    {
        //thao tác lưu thông tin vào database
        return View();
    }
```

```
public ActionResult SetCulture(string culture)
{
    // kiểm tra culture có hợp lệ không
    culture = CultureHelper.GetImplementedCulture(culture);
    // Lưu culture in a cookie với tên: "_culture"
    HttpCookie cookie = Request.Cookies["_culture"];
    if (cookie != null)
        cookie.Value = culture;   // update cookie value
    else
    {
        cookie = new HttpCookie("_culture");
        cookie.Value = culture;
        cookie.Expires = DateTime.Now.AddYears(1);
    }
    Response.Cookies.Add(cookie);
    return RedirectToAction("Index");
}
```

Resources

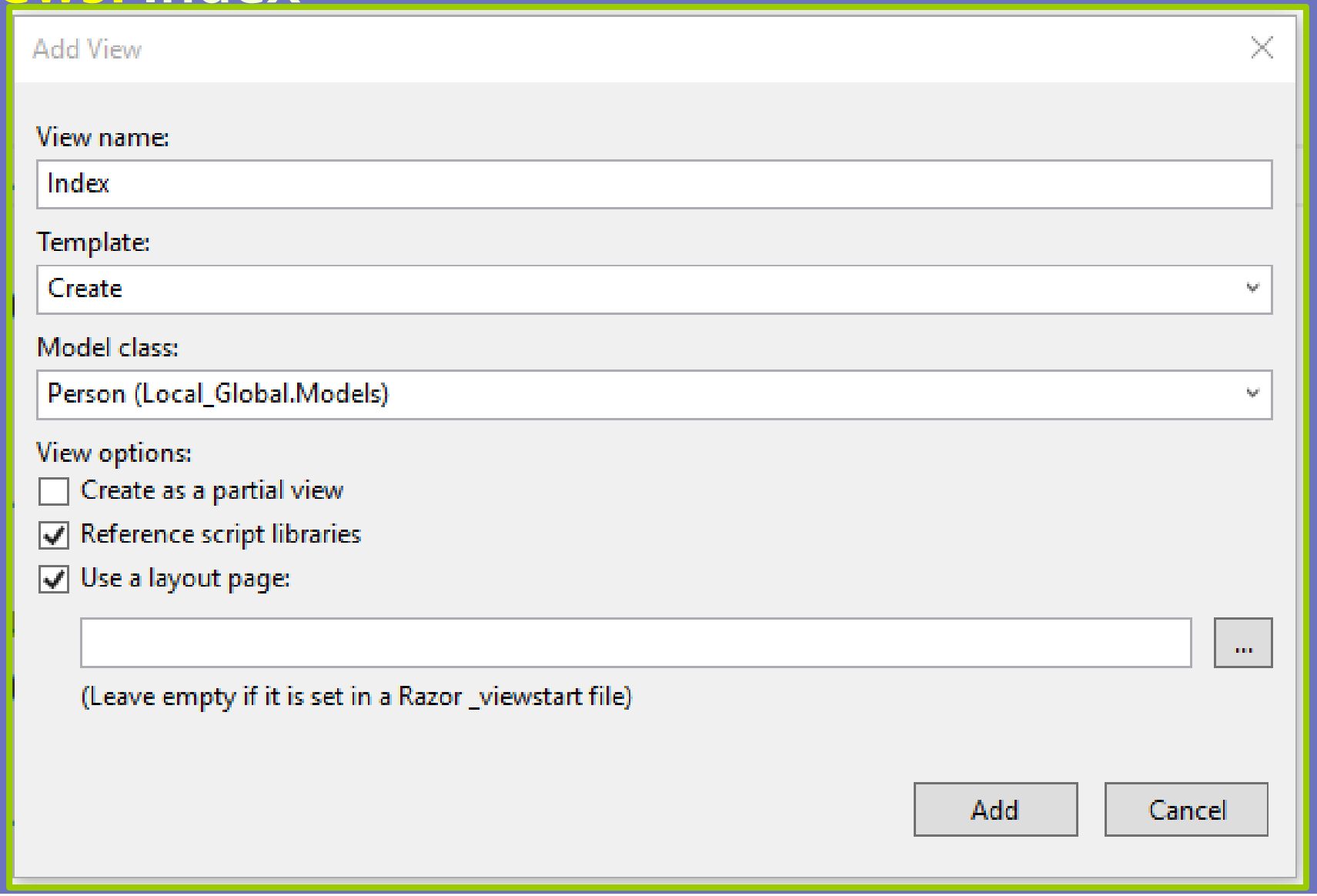


ew: Open the **Web.config** older

```
<system.web.webPages.razor>
  <host factoryType="System.Web.Mvc.MvcWebRazorHostF
  <pages pageBaseType="System.Web.Mvc.WebViewPage">
    <namespaces>
      <add namespace="System.Web.Mvc" />
      <add namespace="System.Web.Mvc.Ajax" />
      <add namespace="System.Web.Mvc.Html" />
      <add namespace="System.Web.Optimization"/>
      <add namespace="System.Web.Routing" />
      <add namespace="Local_Global" />
      <add namespace="Resources"/>
    </namespaces>
  </pages>
</system.web.webPages.razor>
```

Resources

Views: Index



Resources

Views: modify Index

```
@model Local_Global.Models.Person

@{
    ViewBag.Title = @Resources.AddPerson;
    var culture = System.Threading.Thread.CurrentThread.CurrentUICulture
        .Name.ToLowerInvariant();
}
@helper selected(string c, string culture)
{
    if (c == culture)
    {
        @:checked="checked"
    }
}
```

Continue …

Resources

Views: modify Index

```
<h2>@Resources.AddPerson</h2>
@using (Html.BeginForm("SetCulture", "Home"))
{ // Form chọn ngôn ngữ
    <fieldset>
        <legend>@Resources.ChooseYourLanguage</legend>
        <div class="control-group">
            <div class="controls">
                <label for="en-us">
                    <input name="culture" id="en-us" value="en-us" type="radio"
                           @selected("en-us", culture) /> English
                </label>
            </div>
        </div>
        <div class="control-group">
            <div class="controls">
```

Continue …

Resources

Views: modify Index

```
                <label for="es">
                    <input name="culture" id="es" value="es" type="radio"
                           @selected("es", culture) /> Español
                </label>
            </div>
        </div>
        <div class="control-group">
            <div class="controls">
                <label for="vi">
                    <input name="culture" id="vi" value="vi" type="radio"
                           @selected("vi", culture) /> Việt Nam
                </label>
            </div>
        </div>

    </fieldset>
}
```

Continue …

Resources

Views: modify Index

```
@using (Html.BeginForm())
{
    @Html.AntiForgeryToken()
    <div class="form-horizontal">
        <hr />
        @Html.ValidationSummary(true)
        <div class="form-group">
            @Html.Label(@Resources.FirstName, new { @class = "control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.FirstName)
                @Html.ValidationMessageFor(model => model.FirstName)
            </div>
        </div>
        <div class="form-group">
            @Html.Label(@Resources.LastName, new { @class = "control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.LastName)
                @Html.ValidationMessageFor(model => model.LastName)
            </div>
```

Continue …

Resources

Views: modify Index

```
        </div>
        <div class="form-group">
            @Html.Label(@Resources.Age, new { @class = "control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.Age)
                @Html.ValidationMessageFor(model => model.Age)
            </div>
        </div>
        <div class="form-group">
            @Html.Label(@Resources.Email, new { @class = "control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.Email)
                @Html.ValidationMessageFor(model => model.Email)
            </div>
        </div>
```

Continue …

Views: modify Index

```
        <div class="form-group">
            @Html.Label(@Resources.Biography, new { @class = "control-label col-md-2" })
            <div class="col-md-10">
                @Html.EditorFor(model => model.Biography)
                @Html.ValidationMessageFor(model => model.Biography)
            </div>
        </div>

        <div class="form-group">
            <div class="col-md-offset-2 col-md-10">
                <input type="submit" value="@Resources.Create" class="btn btn-default" />
            </div>
        </div>
    </div>
}
```

Continue …

Resources

Views: modify Index (add jquery)

```
@section Scripts {
    @Scripts.Render("~/bundles/jqueryval")
    <script type="text/javascript">
        $(function(){
            $("input[type = 'radio']").click(function () {
                $(this).parents("form").submit(); // post form
            });
        })
    </script>
}
```

Diagnostics and Health Monitoring

Health Monitoring, Elmah and log4net

Diagnostic and Monitoring

- When application started and shutdown
- Unhandled exceptions – stack traces
- Security related diagnostics
  - Malicious user tries to access unauthorized area
  - When a user logged in
- Tracing is a great feature for monitoring ASP.NET Web Forms projects (Lifecycles events)
- Errors can be send on email, log in a file or save in a database

Health Monitoring

- The built in system in ASP.NET for creating, monitoring and publishing diagnostic info
- The settings for this monitoring system are set in the machine level web.config file C:\Windows\Microsoft.NET\Framework*{.NET version}*\Config\web.config
- <eventMappings>- Map specific types of errors to an event
- <rules>Routed events to a provider
- <providers>Set where to store diagnostic info

Unit testing and TDD

TDD

- Test Driven Development is:
  - Software executable specification
  - Interactive design
  - Like using a white board with real code
  - Removing the fear of changing something
- Test Driven Development is not:
  - Just writing unit test
  - 100 % code coverage
  - A replacement for QA and integration tests

Unit testing

- What to test
  - Did the controller return the proper ActionResult?
  - Did the controller build the proper model?
  - Did the controller produce the right side-effects?

Unit testing

- Check if conventional view is rendered up on executing an action in a specific controller.

```
[TestClass]
Public class IsMovieControllerIndexActionExecutes
{
  [TestMethod]
  public void IsItRenders
    var re          ontroller.Index();

    Assert.AreEqual("",
  }
}
```

Arranging something

Asserting some characterics of the performed action

Performing some action

Deployment and Configuration

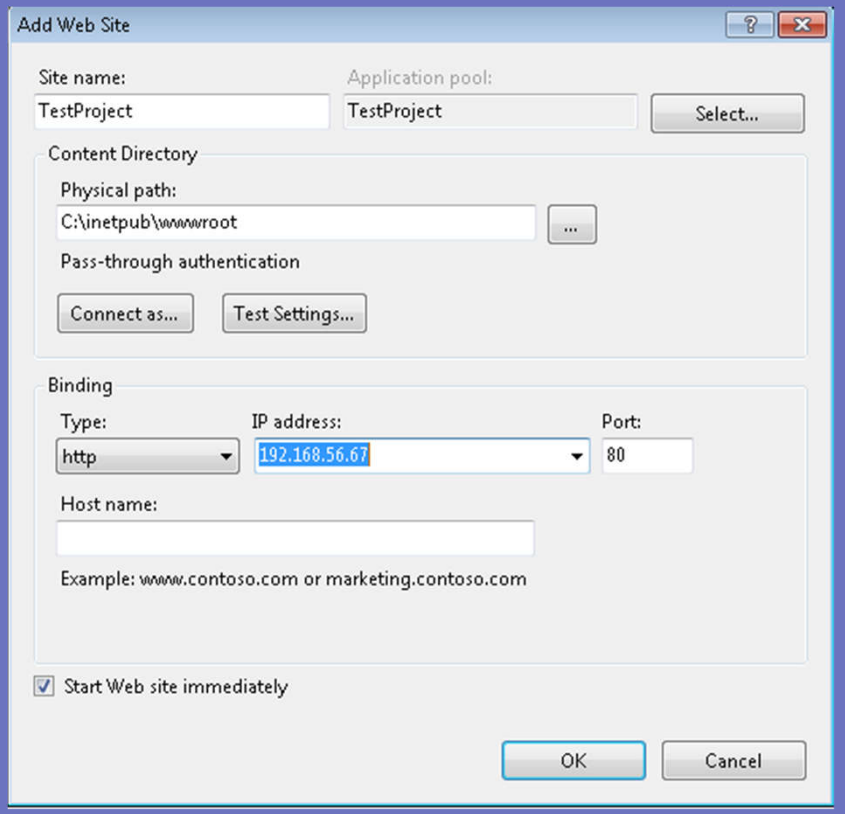
Configuration files

- XML files
  - Authentication and Authorization
  - Compilation
  - Connections
  - Custom errors
  - Page settings
  - Trace and Debug settings

- Hierarchy of the configuration files
- Extensibility of the configuration files

Deployment in IIS

- Install IIS through "Turn windows on/off" in Control panel
- Add site in the IIS configuration manager
  - Set site name
  - Physical path(inetpub)
  - Add local IP and port
  - Check .NET version in the application pool



Deploy the application

- Open the publish window, right click on project
- Different publish methods
  - Web deploy
  - Build deployment package and add it manually
- Configure service URL – IP address of the server
- Site/Application name as it was added in the IIS
- Credentials and destination URL
- Different types of deploy configurations – release, debug, deploy. Different web.config

ASP.NET MVC
Good Practices

Good Practices

- Use ViewModels and Model Validation
- Remove unused ViewEngines
- Add namespaces to Views
- Speed things up with output caching
- Explore the ASP.NET MVC source code
  - http://aspnetwebstack.codeplex.com/
- Use strongly typed views
  - Avoid the ViewBag

Isolate your layers properly

- ViewModel for transmitting data to the view
  - simple POCO de-normalized objects
- Use Controllers for selecting the view to be shown and not for business logic
- Use the view for displaying Html which will be rendered by the browser
  - Not for business logic!
- Use Services/Repositories for manipulating business objects

Use the PRG (PostRedirectGet)

- Prevent reposts to the form
- Issues an HTTP302 with temporary redirect
- Use proper verbs [HttpPost], [HttpGet] on you controllers
- Saving Temporary Data Across Redirects – TempData Dictionary

Productivity Tips

- Use "NuGet" packages that help with productivity
  - RouteDebugger
- ELMAH
- MvcScafolding
- JustCode (ReSharper)

Other tips

- You can extend using HttpModules, HttpHandlers
- You can use HttpCaching
- HTML5 support
- Easier deployment + minification (Including cloud deployment)
- Asynchronous / Await
- Tooling (Page Inspector)
- Web Sockets

Think about globalization

- Make you application support globalisation if its going to be on the internet
- Don't forget to make accessibility
  - http://plugins.jquery.com/project/KeyTips
- Mobile phone support improvements

Summary

- Model–view–controller (MVC) is a software architecture pattern that runs on top of ASP.NET
- It has great separation of concerns and the code is testable, reusable and very extensible
- It produces clean HTML5 and SEO URLs
- Supports code first and database migrations
- Services Web API


Questions?
http://schoolacademy.telerik.com

TODO

- What is in the next version (ASP.NET MVC 5)
- Async?