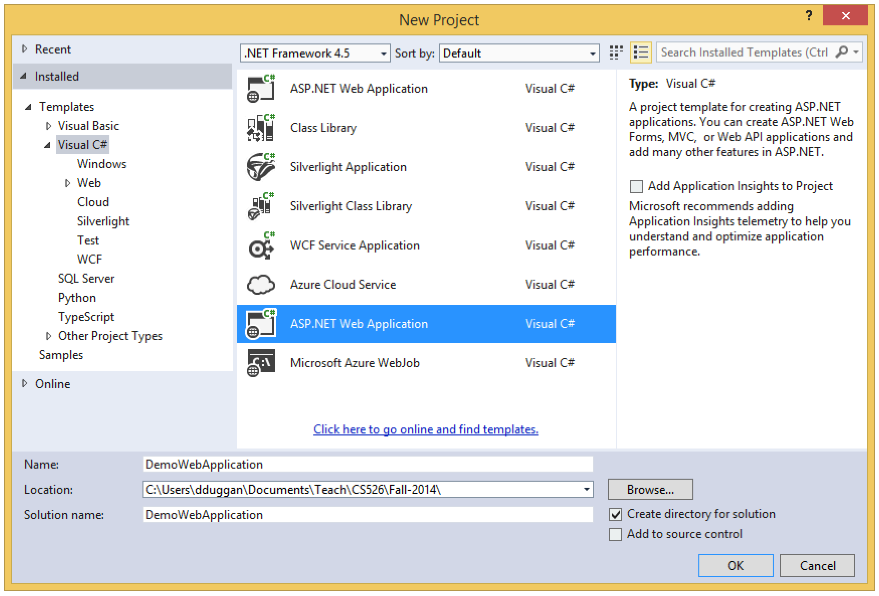
Web Server Security

Dominic Duggan
Stevens Institute of Technology

Based in part on materials by
D. Boneh, J. Mitchell, S. Mitchell

1



.NET Framework 4.5 Sort by: Default

ASP.NET Web Application	Visual C#
Class Library	Visual C#
Silverlight Application	Visual C#
Silverlight Class Library	Visual C#
WCF Service Application	Visual C#
Azure Cloud Service	Visual C#
ASP.NET Web Application	Visual C#
Microsoft Azure WebJob	Visual C#

Type: Visual C#

A project template for creating ASP.NET applications. You can create ASP.NET Web Forms, MVC, or Web API applications and add many other features in ASP.NET.

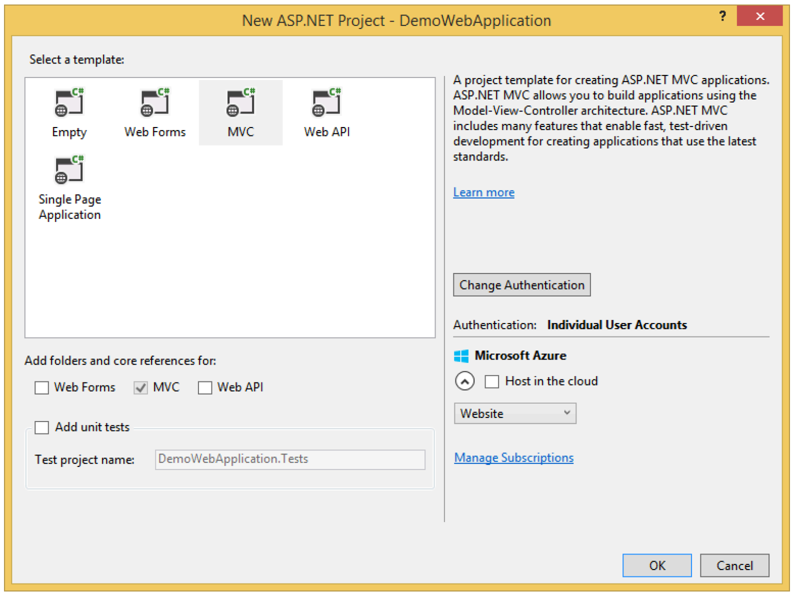
☐ Add Application Insights to Project

Microsoft recommends adding Application Insights telemetry to help you understand and optimize application performance.

Name: DemoWebApplication
Location: C:\Users\dduggan\Documents\Teach\CS526\Fall-2014\
Solution name: DemoWebApplication

☑ Create directory for solution
☐ Add to source control

2


New ASP.NET Project - DemoWebApplication

Select a template:

Empty | Web Forms | MVC | Web API | Single Page Application

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.

Learn more

Change Authentication

Authentication: **Individual User Accounts**

Add folders and core references for:

☐ Web Forms ☑ MVC ☐ Web API

☐ Add unit tests

Test project name: DemoWebApplication.Tests

Microsoft Azure

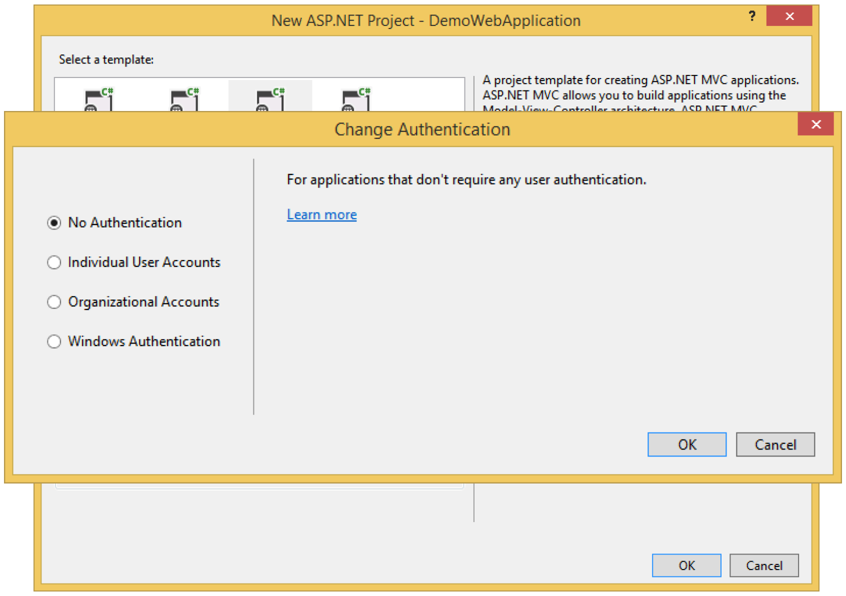
☐ Host in the cloud

Website

Manage Subscriptions

OK Cancel

3


New ASP.NET Project - DemoWebApplication

Select a template:

A project template for creating ASP.NET MVC applications. ASP.NET MVC allows you to build applications using the

Change Authentication

● No Authentication

○ Individual User Accounts

○ Organizational Accounts

○ Windows Authentication

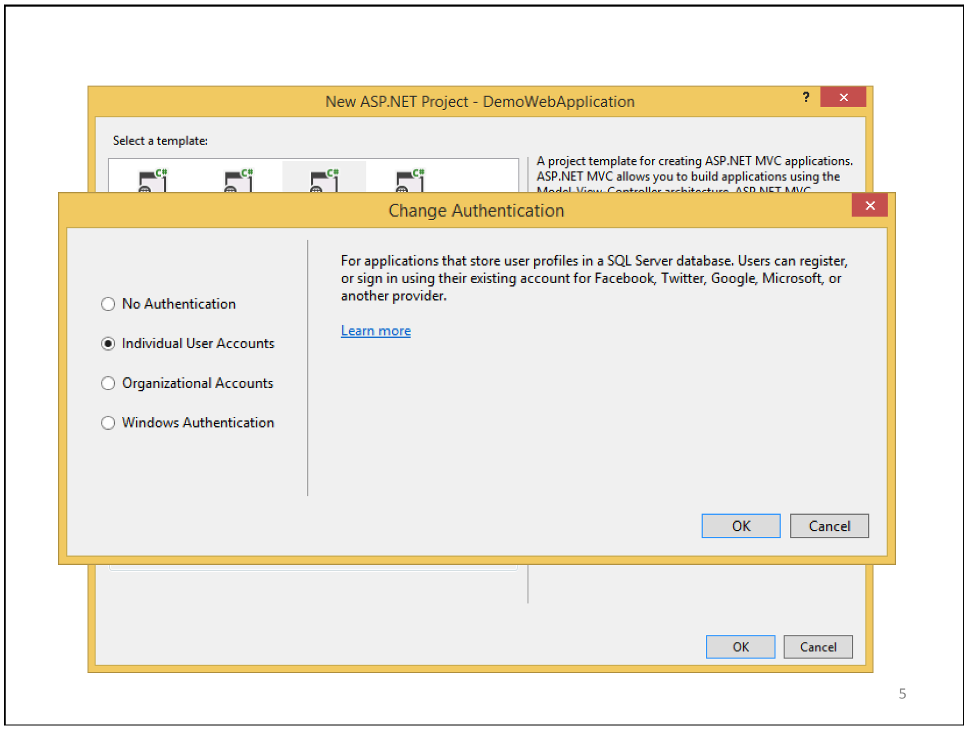
For applications that don't require any user authentication.

Learn more

OK Cancel

OK Cancel

4



New ASP.NET Project - DemoWebApplication

Select a template:

A project template for creating ASP.NET MVC applications. ASP.NET MVC allows you to build applications using the

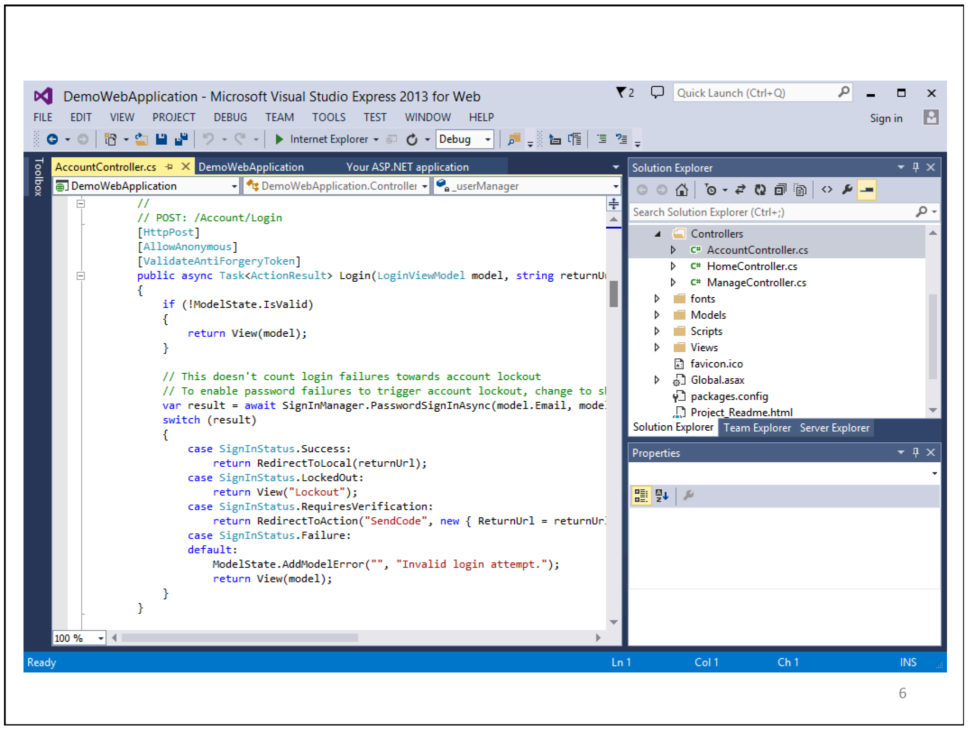
Change Authentication

- No Authentication
- [✓ Individual User Accounts]
- Organizational Accounts
- Windows Authentication

For applications that store user profiles in a SQL Server database. Users can register, or sign in using their existing account for Facebook, Twitter, Google, Microsoft, or another provider.

Learn more

5



DemoWebApplication - Microsoft Visual Studio Express 2013 for Web

```
//
// POST: /Account/Login
[HttpPost]
[AllowAnonymous]
[ValidateAntiForgeryToken]
public async Task<ActionResult> Login(LoginViewModel model, string returnU
{
    if (!ModelState.IsValid)
    {
        return View(model);
    }

    // This doesn't count login failures towards account lockout
    // To enable password failures to trigger account lockout, change to s
    var result = await SignInManager.PasswordSignInAsync(model.Email, mode
    switch (result)
    {
        case SignInStatus.Success:
            return RedirectToLocal(returnUrl);
        case SignInStatus.LockedOut:
            return View("Lockout");
        case SignInStatus.RequiresVerification:
            return RedirectToAction("SendCode", new { ReturnUrl = returnUr
        case SignInStatus.Failure:
        default:
            ModelState.AddModelError("", "Invalid login attempt.");
            return View(model);
    }
}
```

6

AUTHENTICATION: INTERNAL LOGIN

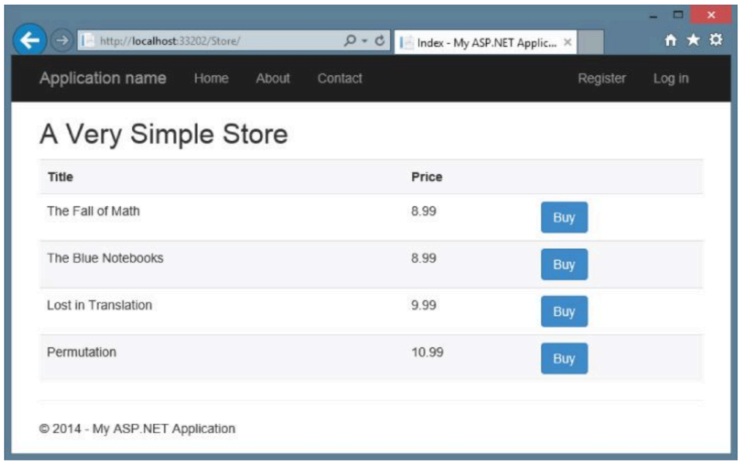
Restricting Access

- Restrict access to registered users
- Restrict administrative access

- Web Forms: directory-based
  - `/Admin/...`

- MVC: action-based
  - Rich routing structure
  - `[Authorize]` for filtering

Browsing


http://localhost:33202/Store/

Index - My ASP.NET Applic...

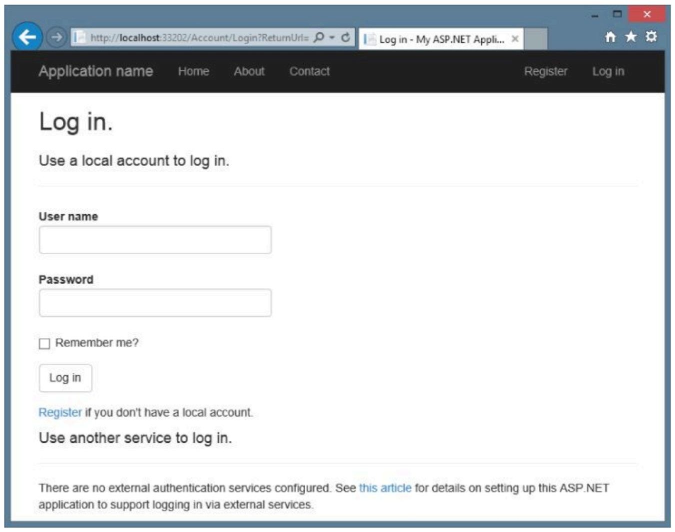
A Very Simple Store

Title	Price	
The Fall of Math	8.99	Buy
The Blue Notebooks	8.99	Buy
Lost in Translation	9.99	Buy
Permutation	10.99	Buy

© 2014 - My ASP.NET Application

9

Browsing


http://localhost:33202/Account/Login?ReturnUrl=

Log in - My ASP.NET Appli...

Log in.

Use a local account to log in.

User name

Password

☐ Remember me?

Register if you don't have a local account.

Use another service to log in.

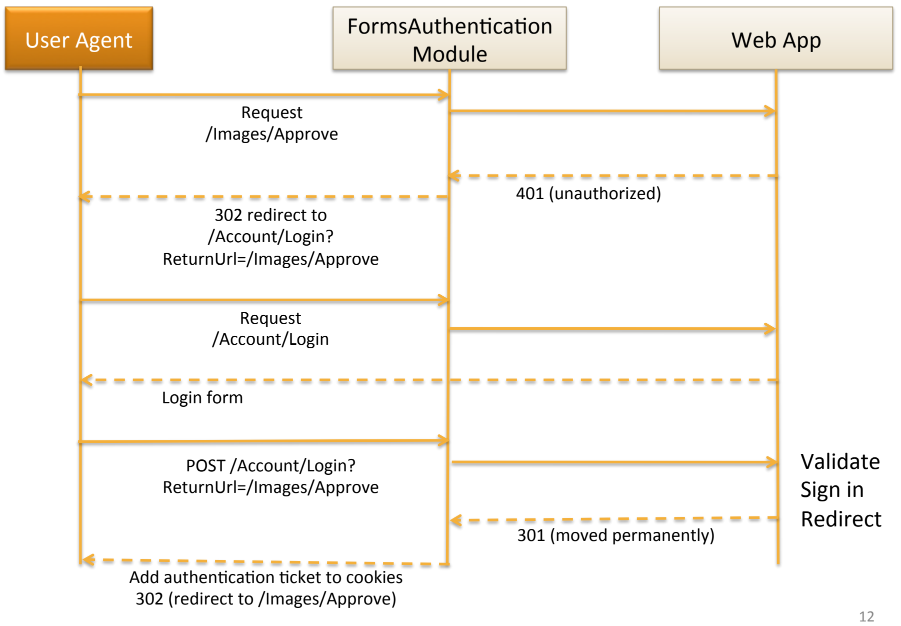
There are no external authentication services configured. See this article for details on setting up this ASP.NET application to support logging in via external services.

10

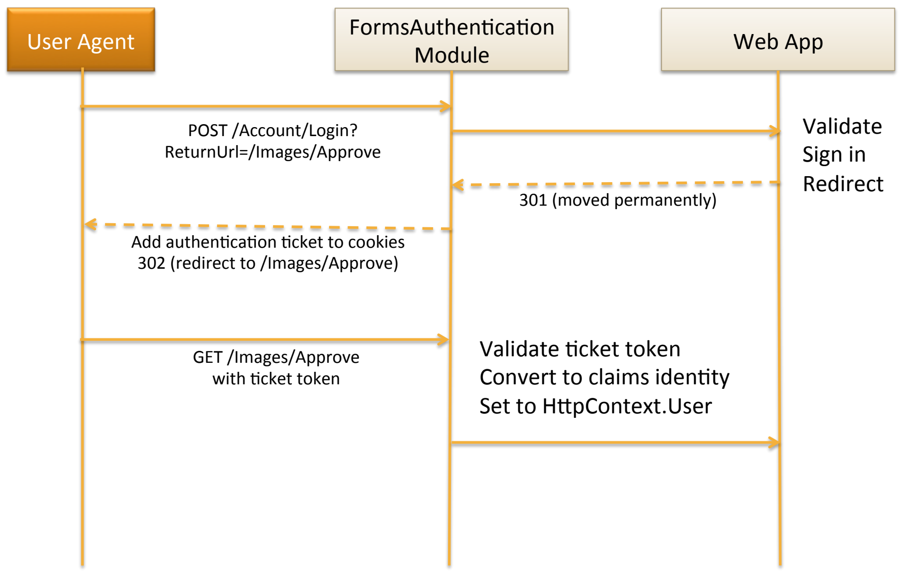
Authorized Actions

- Example:
```
[Authorize]
public class ImagesController {
   public ActionResult Approve(int ImageId)
   {
      ...
   }
}
```
- In App_Start/Startup.Auth.cs:
```
public void ConfigureAuth(IAppBuilder app) {
   app.UseCookieAuthentication(new CookieAuthenticationOption {
      AuthenticationType =
            DefaultAuthenticationTypes.ApplicationCookie,
      LoginPath = new PathString("/Account/Login")
   });
}
```

11


User Agent
FormsAuthentication Module
Web App
Request /Images/Approve
401 (unauthorized)
302 redirect to /Account/Login? ReturnUrl=/Images/Approve
Request /Account/Login
Login form
POST /Account/Login? ReturnUrl=/Images/Approve
Validate
Sign in
Redirect
301 (moved permanently)
Add authentication ticket to cookies
302 (redirect to /Images/Approve)

12


User Agent
FormsAuthentication Module
Web App
POST /Account/Login?ReturnUrl=/Images/Approve
Validate
Sign in
Redirect
301 (moved permanently)
Add authentication ticket to cookies
302 (redirect to /Images/Approve)
GET /Images/Approve with ticket token
Validate ticket token
Convert to claims identity
Set to HttpContext.User

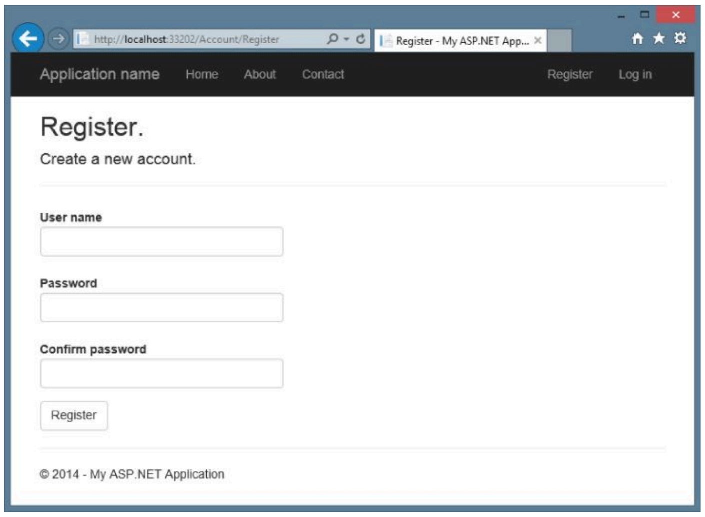
13

Example Login Action

```
[HttpPost]
public ActionResult Login(LoginViewModel model, string returnUrl) {
   if (!ModelState.IsValid) {
      return new View(model);
   }

   var result =
      await SignInManager.PasswordSignInAsync(model.Email,
                                              model.Password, ...);

   switch (result) {

      case SignInStatus.Success:
         return RedirectToLocal(returnUrl);

      case SignInStatus.RequiresVerification:
         return RedirectToAction("SendCode", ... returnUrl ...);
   }
```

14

Registration


http://localhost:33202/Account/Register
Register - My ASP.NET App...
Application name
Home
About
Contact
Register
Log in

Register.

Create a new account.

User name

Password

Confirm password

Register

© 2014 - My ASP.NET Application

15

Example Registration Action

```
[HttpPost]
public async Task<ActionResult> Register(RegisterViewModel model) {
   if (ModelState.IsValid) {
      var user = new ApplicationUser { UserName = model.Email,
                                       Email = model.Email };
      var result = await UserManager.CreateAsync(user,
                                                 model.Password);
      if (result.Succeeded) {
         await SignInManager.SignInAsync(user, ...);
         return RedirectToAction("Index", "Home");
      }
      AddErrors(result);
   }

   // If we got this far, something failed, redisplay form
   return View(model);
}
```

16

Example Login Action

```
[HttpPost]
public ActionResult Login(LoginViewModel model, string returnUrl) {
   if (!ModelState.IsValid) {
      return new View(model);
   }

   var result =
      await SignInManager.PasswordSignInAsync(model.Email,
                                              model.Password, ...);

   switch (result) {

      case SignInStatus.Success:
         return RedirectToLocal(returnUrl);

      case SignInStatus.RequiresVerification:
         return RedirectToAction("SendCode", ... returnUrl ...);
}
```

17

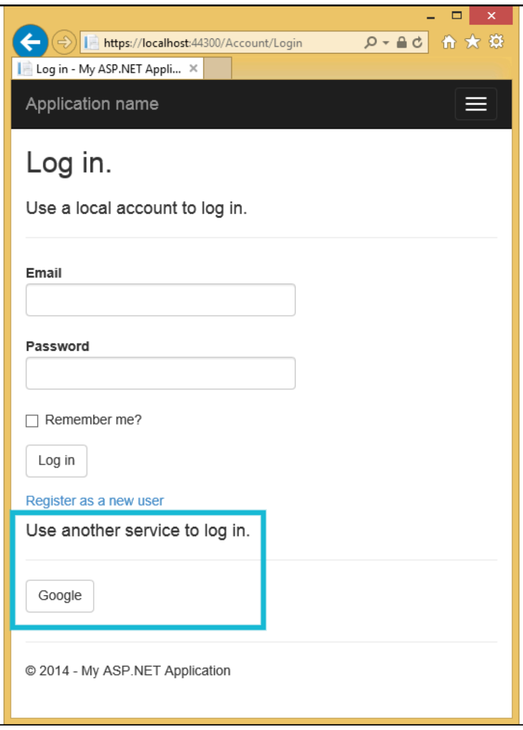
AUTHENTICATION: EXTERNAL LOGIN

18

Internal vs External

- Internal:
  - User database part of app
  - **Active** middleware: redirects unauthorized requests
- External
  - External authority provider (e.g. Facebook)
  - **Passive** middleware: responds to app requests

19


https://localhost:44300/Account/Login
Log in - My ASP.NET Appli...
Application name

Log in.

Use a local account to log in.

Email

Password

Remember me?

Log in

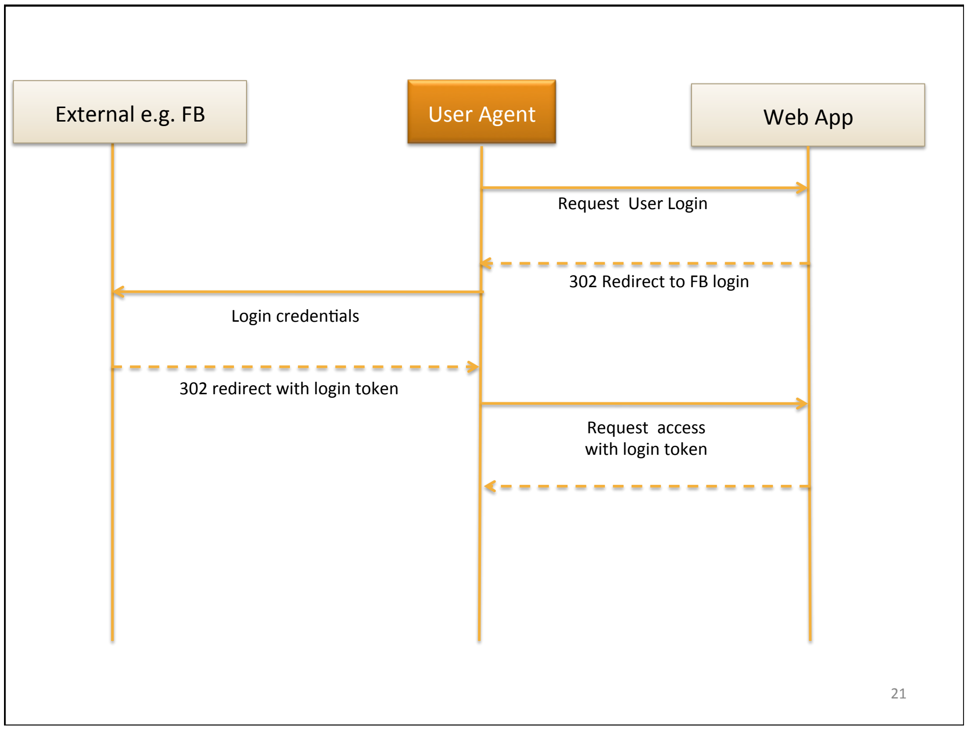
Register as a new user

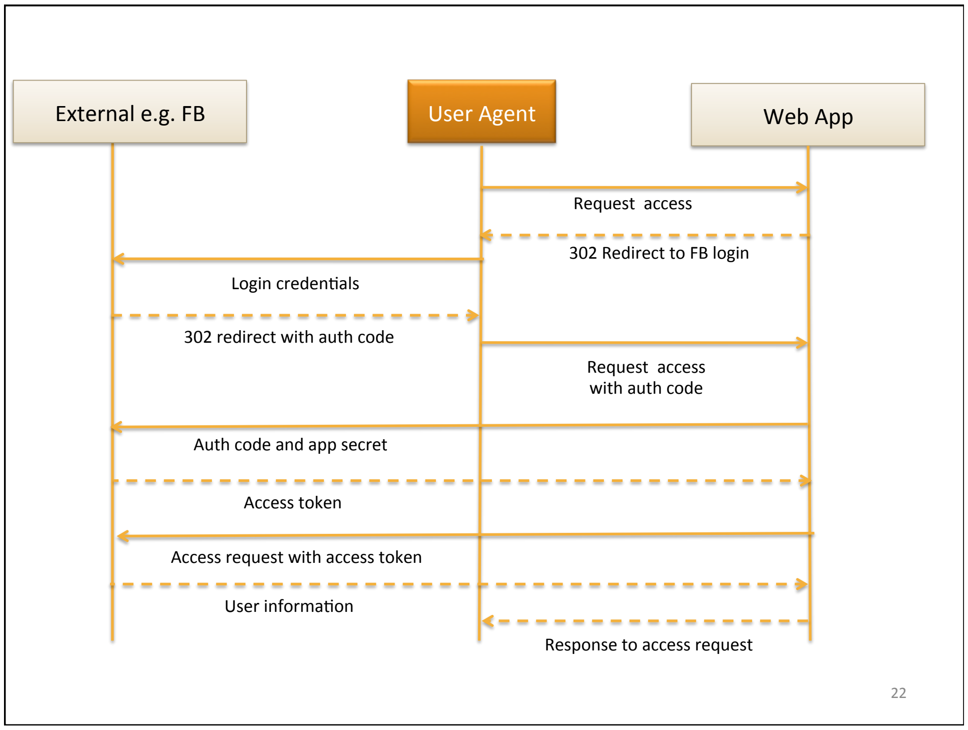
Use another service to log in.

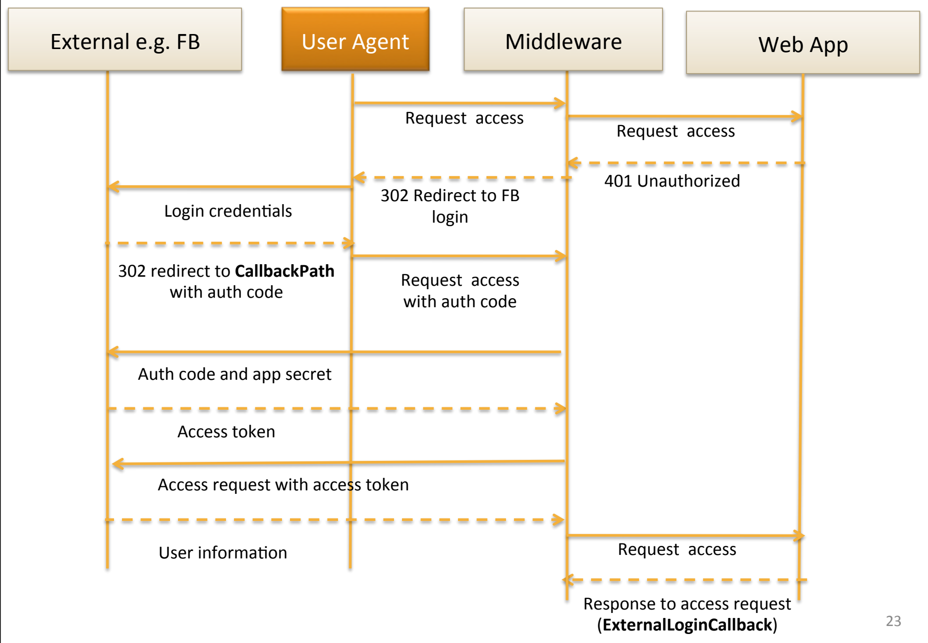
Google

© 2014 - My ASP.NET Application

20


External e.g. FB
User Agent
Web App
Request User Login
302 Redirect to FB login
Login credentials
302 redirect with login token
Request access with login token
21


External e.g. FB
User Agent
Web App
Request access
302 Redirect to FB login
Login credentials
302 redirect with auth code
Request access with auth code
Auth code and app secret
Access token
Access request with access token
User information
Response to access request
22



23

Authorized Actions

- Example:

```
[Authorize]
public class ImagesController {
   public ActionResult Approve(int ImageId)
   {
      ...
   }
}
```

- In App_Start/Startup.Auth.cs:

```
public void ConfigureAuth(IAppBuilder app) {
   app.UseExternalSignInCookie(
      DefaultAuthenticationTypes.ExternalCookie);
   app.UseFacebookAuthentication(appId: "",
                                 appSecret: "");
}
```

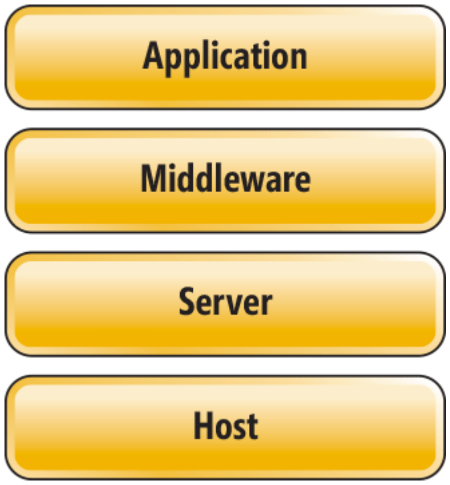
24

Middleware: OWIN

- Open Web Interface for .NET
- Connecting apps to Web server
  - Decouple .NET from IIS
- Components register app delegates
  - Delegates chained in pipeline
  - Server pushes operations
  - Asynchronous for performance
- Katana: OWIN-based framework

25

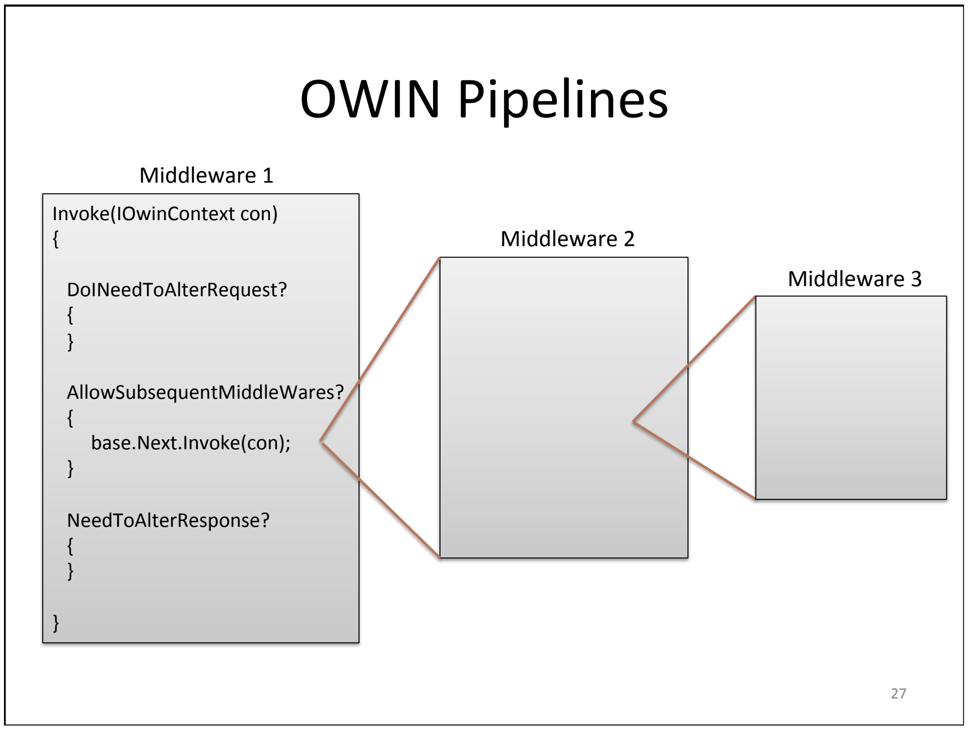
Framework Architecture



Ex: MVC5, Web API

Ex: IIS

26

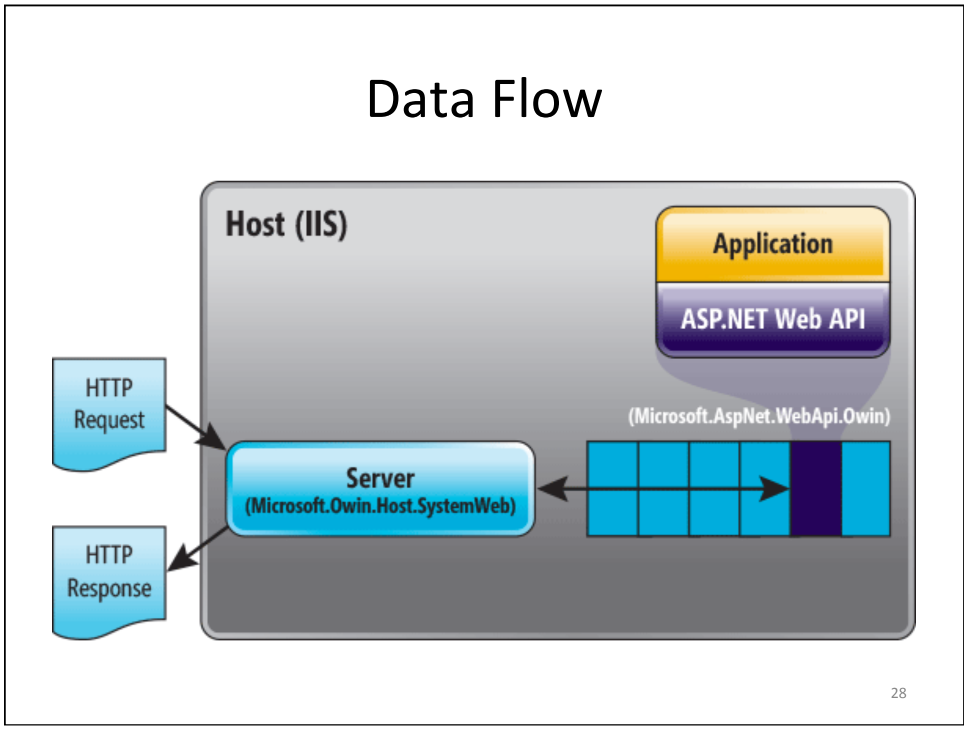


OWIN Pipelines

Middleware 1

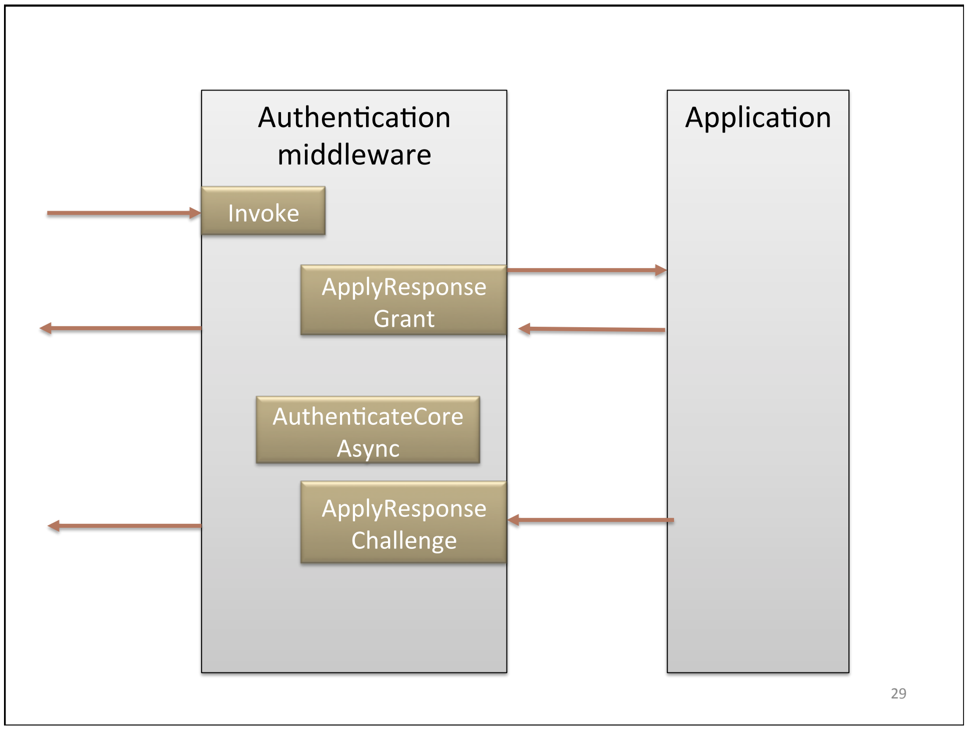
```
Invoke(IOwinContext con)
{

  DoINeedToAlterRequest?
  {
  }

  AllowSubsequentMiddleWares?
  {
      base.Next.Invoke(con);
  }

  NeedToAlterResponse?
  {
  }

}
```

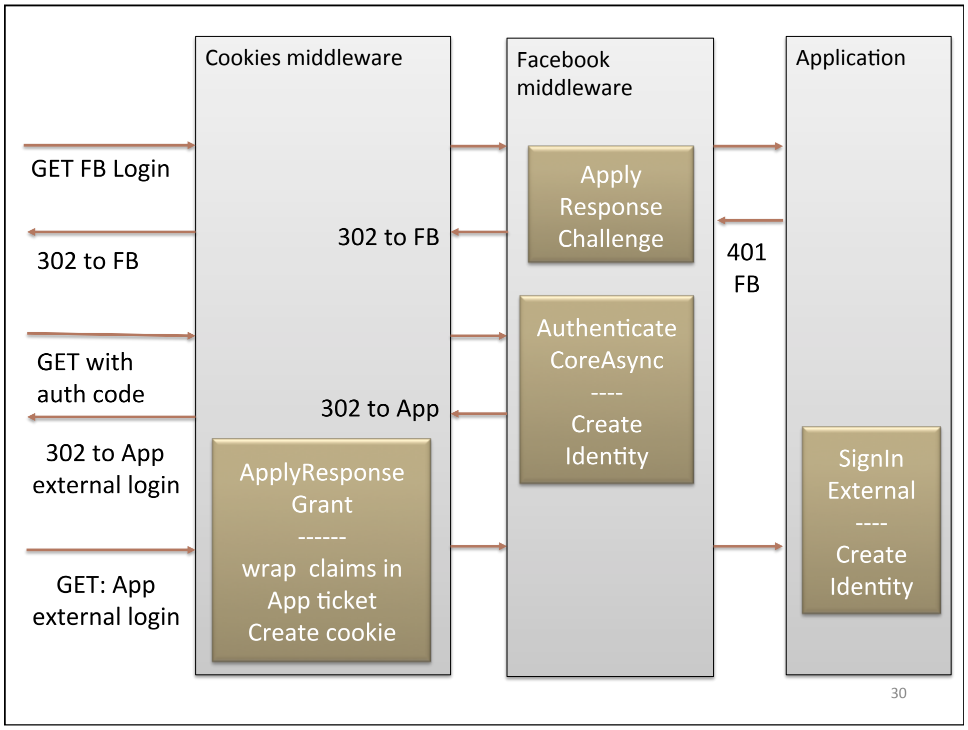
27



Data Flow

28


Authentication middleware
Application
Invoke
ApplyResponse Grant
AuthenticateCore Async
ApplyResponse Challenge
29


Cookies middleware
Facebook middleware
Application
GET FB Login
302 to FB
302 to FB
Apply Response Challenge
401 FB
GET with auth code
302 to App
Authenticate CoreAsync ---- Create Identity
302 to App external login
ApplyResponse Grant ------ wrap claims in App ticket Create cookie
GET: App external login
SignIn External ---- Create Identity
30

Configuring forms-based authentication

- Configure in `App_Start/Startup.Auth.cs`
  - Specify URL of login page
- Create login action
  - Use `AccountController` methods to login user
- Specify authorization settings
  - Authorization database
  - What users can access what URLs

31

Application Sign-In

```
public static IAppBuilder
   UseApplicationSignInCookie(thisIAppBuilder app)
{
   return UseFormsAuthentication(app,
      new FormsAuthenticationOptions {
        AuthenticationType = FormsAuthenticationDefaults
                              .ApplicationAuthenticationType,
        AuthenticationMode = AuthenticationMode.Active,
        CookieName = FormsAuthenticationDefaults.CookiePrefix +
                     FormsAuthenticationDefaults
                         .ApplicationAuthenticationType,
        LoginPath = FormsAuthenticationDefaults.LoginPath,
        LogoutPath = FormsAuthenticationDefaults.LogoutPath,
    });
}
```

32

External Sign-In

```
public static IAppBuilder
   UseExternalSignInCookie(thisIAppBuilder app)
{
   app.SetDefaultSignInAsAuthenticationType(
           FormsAuthenticationDefaults.ExternalAuthenticationType);

   return UseFormsAuthentication(app,
      new FormsAuthenticationOptions {
        AuthenticationType = FormsAuthenticationDefaults
                                  .ExternalAuthenticationType,
        AuthenticationMode = AuthenticationMode.Passive,
        CookieName = FormsAuthenticationDefaults.CookiePrefix +
                     FormsAuthenticationDefaults
                              .ExternalAuthenticationType,
        ExpireTimeSpan = TimeSpan.FromMinutes(5),
    });
}
```

33

Forms Authentication Defaults

```
app.UseFormsAuthentication(new FormsAuthenticationOptions() {
    AuthenticationMode =
         Microsoft.Owin.Security.AuthenticationMode.Active,
    AuthenticationType = "MyApplication",
    CookieDomain = ".myapp.com",
    CookieHttpOnly = true,
    CookieName = ".AspNet.MyApplication",
    CookiePath = "/Account",
    CookieSecure = CookieSecureOption.Always,
    ExpireTimeSpan = TimeSpan.FromDays(1),
    LoginPath = "/Account/Login",
    ReturnUrlParameter = "return_url",
    SlidingExpiration = true,
    Provider = new FormsAuthenticationProvider() {
        OnResponseSignin = ...,
        OnValidateIdentity = ...
    }
});
```

34

Authorize as Global Filter

- Register in App_Start/FilterConfig.cs:

```
public static void RegisterGlobalFilters
                (GlobalFilterCollection filters) {
   filters.Add(
      new System.Web.Mvc.AuthorizeAttribute());
   filters.Add(new HandleErrorAttribute());
}
```

- Allow anonymous where required:

```
[AllowAnonymous]
public ActionResult Login(String returnUrl) {
   …
}
```

ACCESS CONTROL

Access Control

- What are authenticated users allowed to do?
- General abstraction: access matrix

	Password File	Joe's Calendar	Payroll File
Administrator	RW		
Joe		RW	
Jane		R	RW

37

Access Control

- Access control lists

	Password File
Administrator	RW
Joe	
Jane	

	Joe's Calendar
Administrator	
Joe	RW
Jane	R

	Payroll File
Administrator	
Joe	
Jane	RW

38

Access Control

- Role-based access control (RBAC)

	Roles
Sam	Administrator
Joe	Administrator, Accounts
Jane	Accounts

	Password File
Administrator	RW

	Joe's Calendar
Joe	RW
Accounts	R

	Payroll File
Accounts	RW

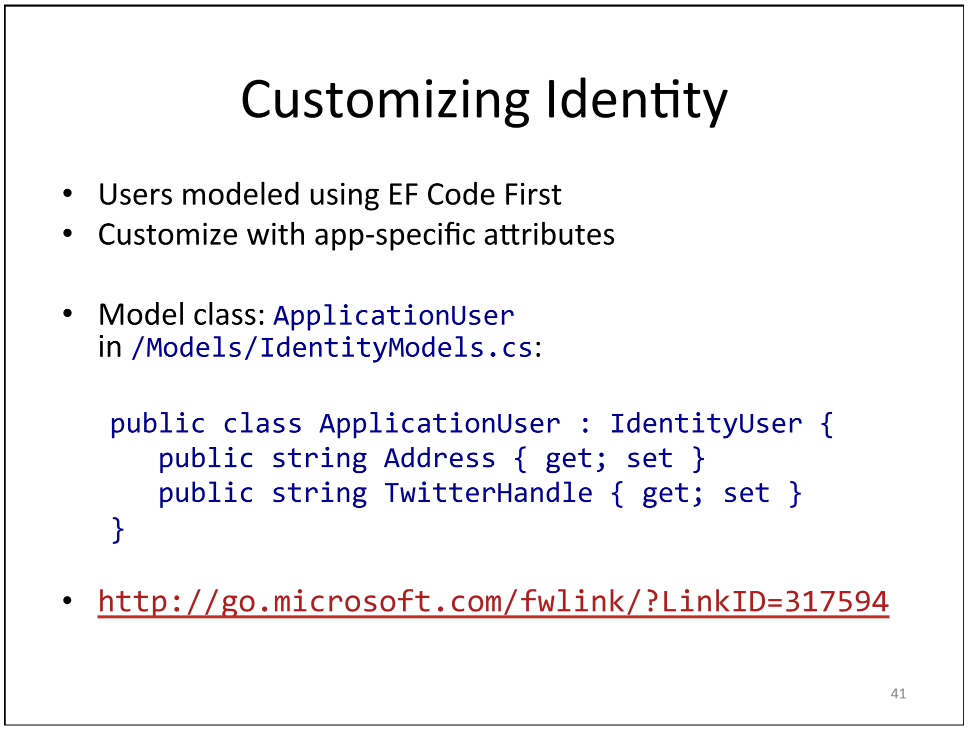
39

Access Control in MVC

- Roles for RBAC

```
public class ImagesController {
   [Authorize(Roles="Approver")]
   public ActionResult Approve(int ImageId) {...}
}

public class AccountController {
   [Authorize(Roles="Administrator")]
   public ActionResult EditUser(UserView user) {...}
}

public class ImagesController {
   [Authorize(Roles="User")]
   public ActionResult Upload(ImageView img) {...}
}
```

40



Customizing Identity

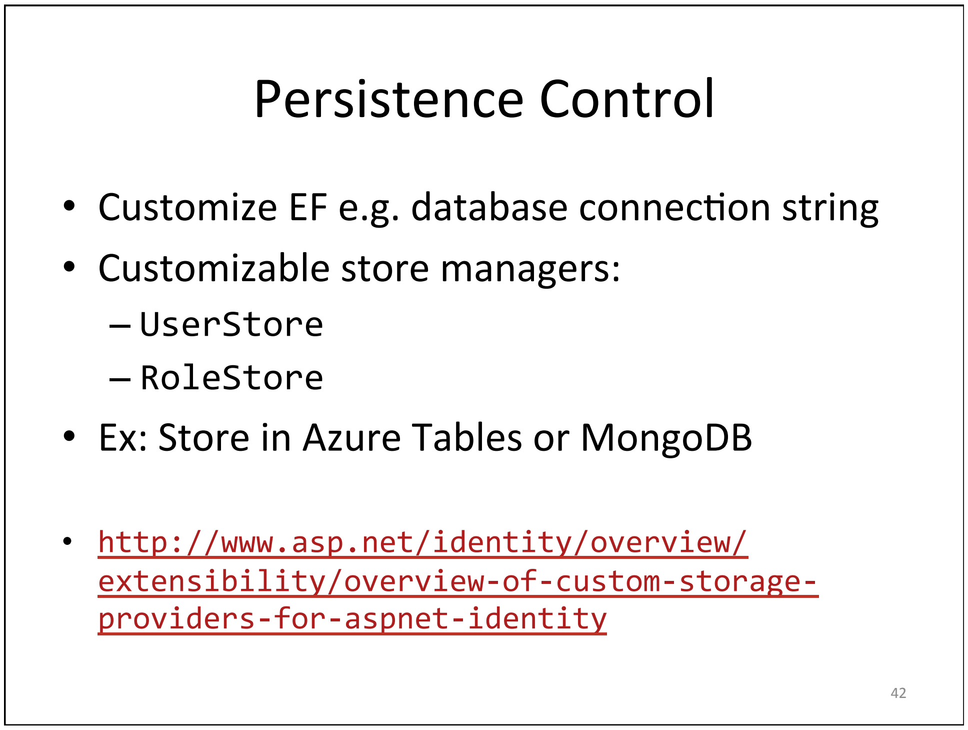
- Users modeled using EF Code First
- Customize with app-specific attributes

- Model class: `ApplicationUser`
  in `/Models/IdentityModels.cs`:

```
public class ApplicationUser : IdentityUser {
   public string Address { get; set }
   public string TwitterHandle { get; set }
}
```

- http://go.microsoft.com/fwlink/?LinkID=317594

41



Persistence Control

- Customize EF e.g. database connection string
- Customizable store managers:
  - `UserStore`
  - `RoleStore`
- Ex: Store in Azure Tables or MongoDB

- http://www.asp.net/identity/overview/extensibility/overview-of-custom-storage-providers-for-aspnet-identity

42

Managing Users and Roles

- Customizable user and role managers:
  - `UserManager`
  - `RoleManager`

- http://azure.microsoft.com/en-us/documentation/articles/web-sites-dotnet-deploy-aspnet-mvc-app-membership-oauth-sql-database/