INFO3067 Week 7 Class 1

Review

- google.maps.Geocoder()
- google.maps.Map
- google.maps.Marker

Today's Overview

We're ready to put the finishing touches on the application now. Today we will lock it down so it's more secure and then put it into a production environment.

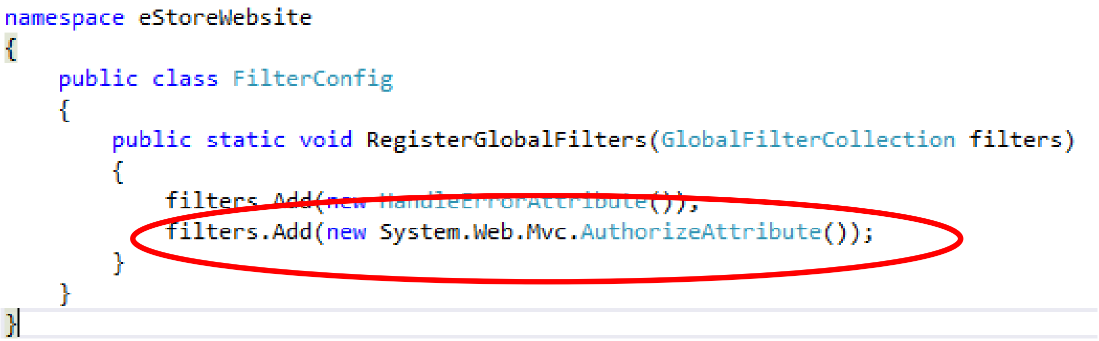
Locking Down the Application

There are 2 concepts to understand when talking about web site access, namely:

- **Authentication** – you are who you say you are
- **Authorization** – you are allowed to do something

For instance I can be authenticated (eg. logged on) but not authorized (may need certain extra privileges) to use a part of an application. Conversely, I can be authorized based on my account, but if I haven't logged on (authenticated) I am still not allowed to use the application. We'll keep our requirements simple for this course and work on the premise; if you are logged on you can access any page. If you are not logged on you can only access the **Home/Login** and **Home/Register** methods.

To lock all pages add a line to the **App_Start/FilterConfig.cs** class:

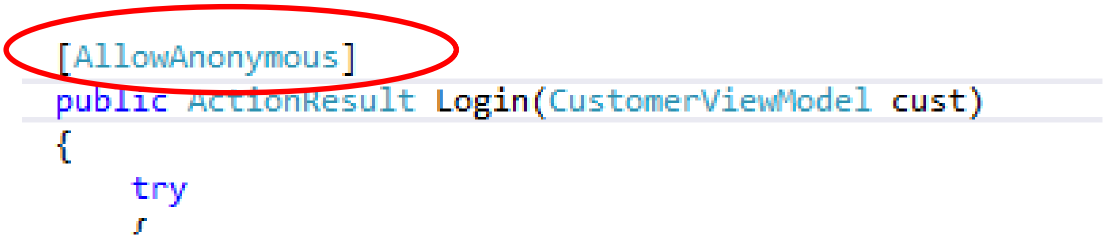


```
namespace eStoreWebsite
{
    public class FilterConfig
    {
        public static void RegisterGlobalFilters(GlobalFilterCollection filters)
        {
            filters.Add(new HandleErrorAttribute());
            filters.Add(new System.Web.Mvc.AuthorizeAttribute());
        }
    }
}
```

This will make every method in the system to require authorization before access is allowed.

Close your browser and then restart the application in Visual Studio and try and go to any of the pages and you should end up on your home page

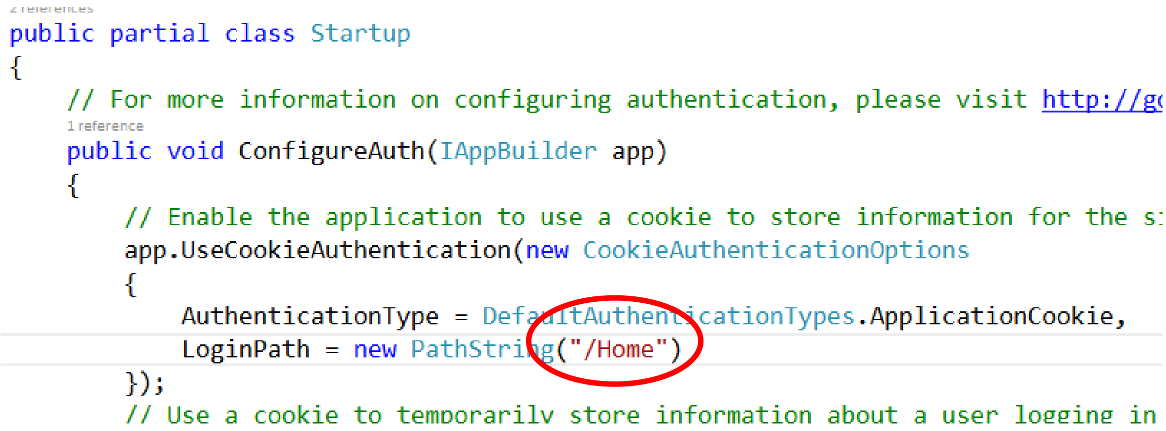
Now we need to get logged on, so we'll manually add the **[AllowAnonymous]** attribute to both the Login, Register and Index methods in the Home controller:



```
[AllowAnonymous]
public ActionResult Login(CustomerViewModel cust)
{
    try
    {
```

Changing Default Redirection

Right now the application is set to return to a controller called Account with a method called Login. We need to change this to redirect the user to our home page where they should be instructed to login, so change the **App_Start\Startup.Auth.cs** file to look like:

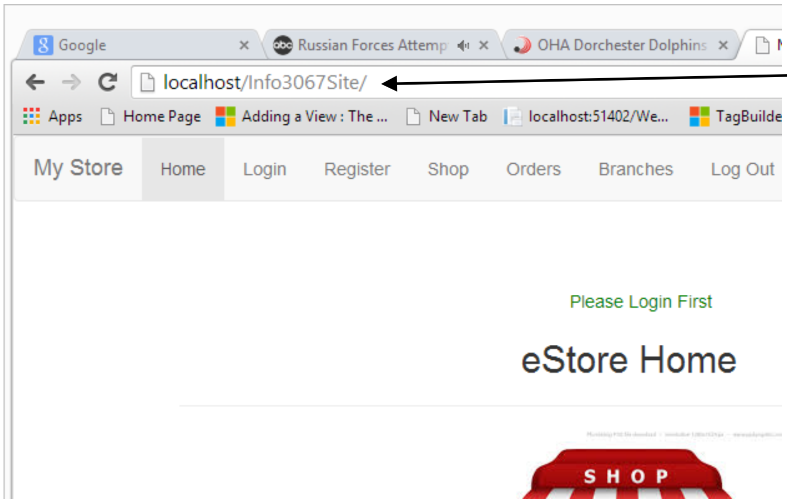


```
2 references
public partial class Startup
{
    // For more information on configuring authentication, please visit http://gc
    1 reference
    public void ConfigureAuth(IAppBuilder app)
    {
        // Enable the application to use a cookie to store information for the s:
        app.UseCookieAuthentication(new CookieAuthenticationOptions
        {
            AuthenticationType = DefaultAuthenticationTypes.ApplicationCookie,
            LoginPath = new PathString("/Home")
        });
        // Use a cookie to temporarily store information about a user logging in
```

Putting Your Web Application into Production

To do this we basically need to configure two pieces of software; IIS (web server) and SQL Server. Watch the archived video on FOL (**Final Installation Demonstration**) to see the steps involved. The video assumes you have IIS7+ installed, if you do not have IIS7 installed, you can obtain it here:

http://www.microsoft.com/web/downloads/platform.aspx

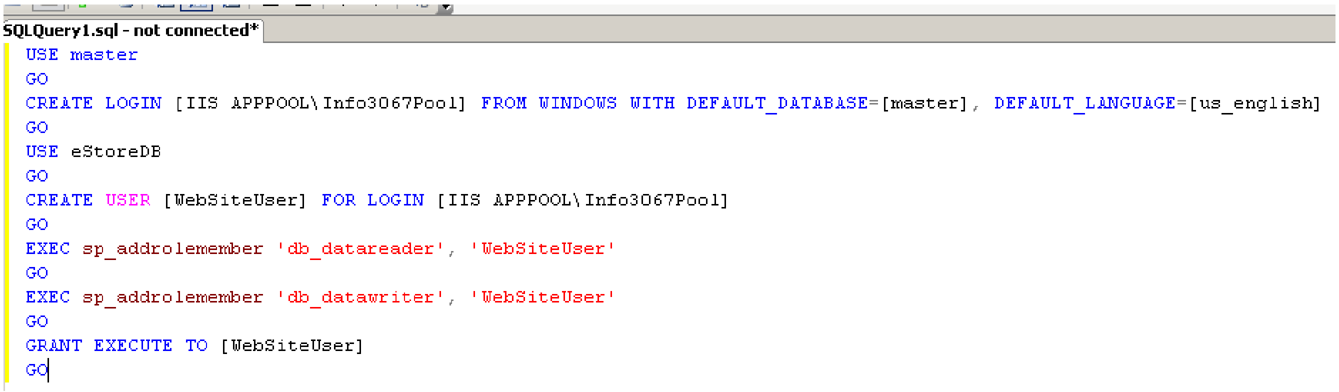


Please Login First

eStore Home

Notice the URL here is using a default pattern, not Visual Studio’s built in server

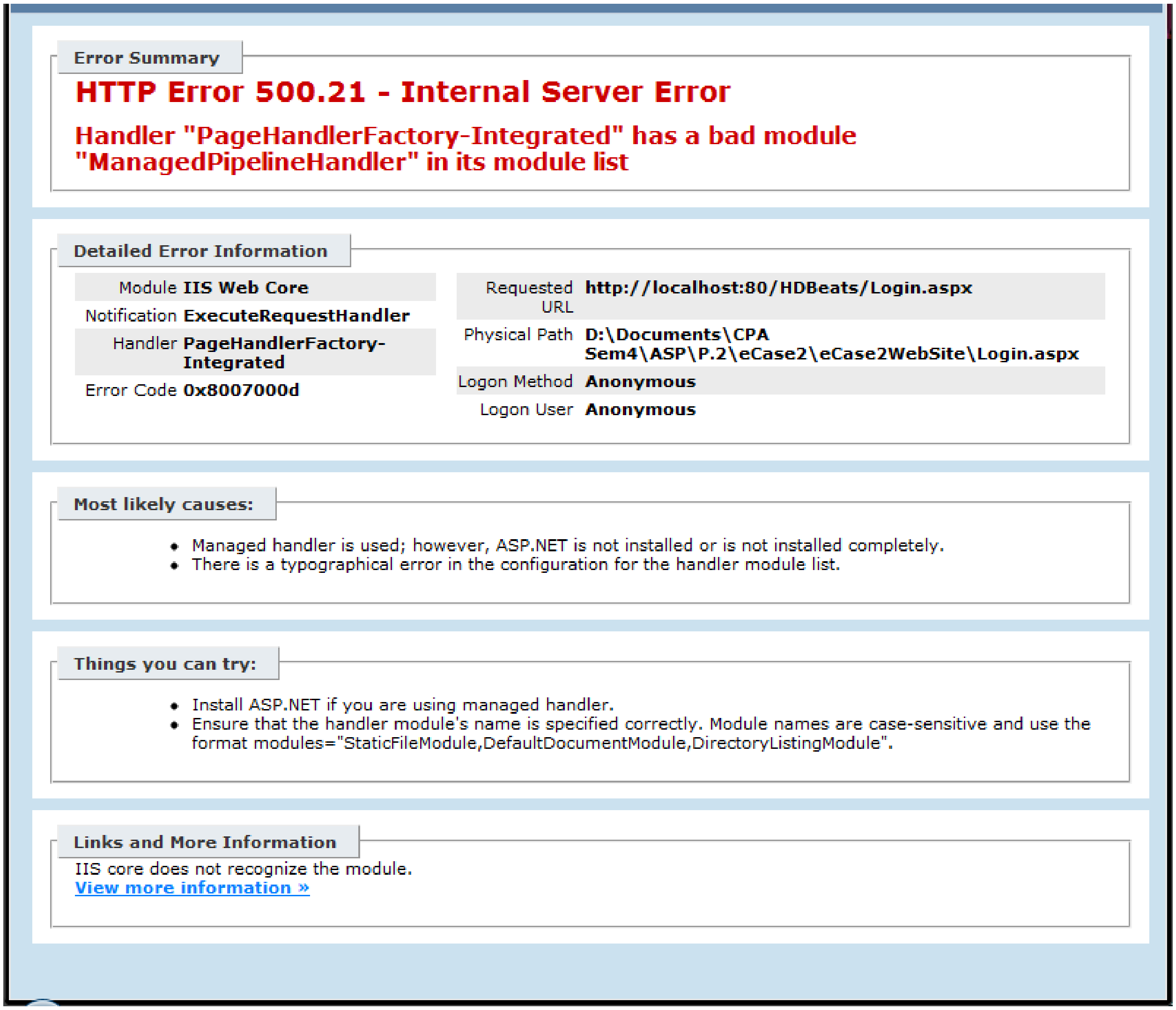
The video uses the following SQL script (hard to see but it is an = sign in the CREATE LOGIN line below).



```
USE master
GO
CREATE LOGIN [IIS APPPOOL\Info3067Pool] FROM WINDOWS WITH DEFAULT_DATABASE=[master], DEFAULT_LANGUAGE=[us_english]
GO
USE eStoreDB
GO
CREATE USER [WebSiteUser] FOR LOGIN [IIS APPPOOL\Info3067Pool]
GO
EXEC sp_addrolemember 'db_datareader', 'WebSiteUser'
GO
EXEC sp_addrolemember 'db_datawriter', 'WebSiteUser'
GO
GRANT EXECUTE TO [WebSiteUser]
GO
```

The **LOGIN** is used to get access to the server, and the **USER** is used to get access to the database.

If you end up getting the following error when viewing your website you will need to run a program from the framework to refresh the asp.net engine in IIS (see below)



Error Summary

HTTP Error 500.21 - Internal Server Error

Handler "PageHandlerFactory-Integrated" has a bad module "ManagedPipelineHandler" in its module list

Detailed Error Information

Module	**IIS Web Core**	Requested URL	**http://localhost:80/HDBeats/Login.aspx**
Notification	**ExecuteRequestHandler**	Physical Path	**D:\Documents\CPA Sem4\ASP\P.2\eCase2\eCase2WebSite\Login.aspx**
Handler	**PageHandlerFactory-Integrated**	Logon Method	**Anonymous**
Error Code	**0x8007000d**	Logon User	**Anonymous**

Most likely causes:

- Managed handler is used; however, ASP.NET is not installed or is not installed completely.
- There is a typographical error in the configuration for the handler module list.

Things you can try:

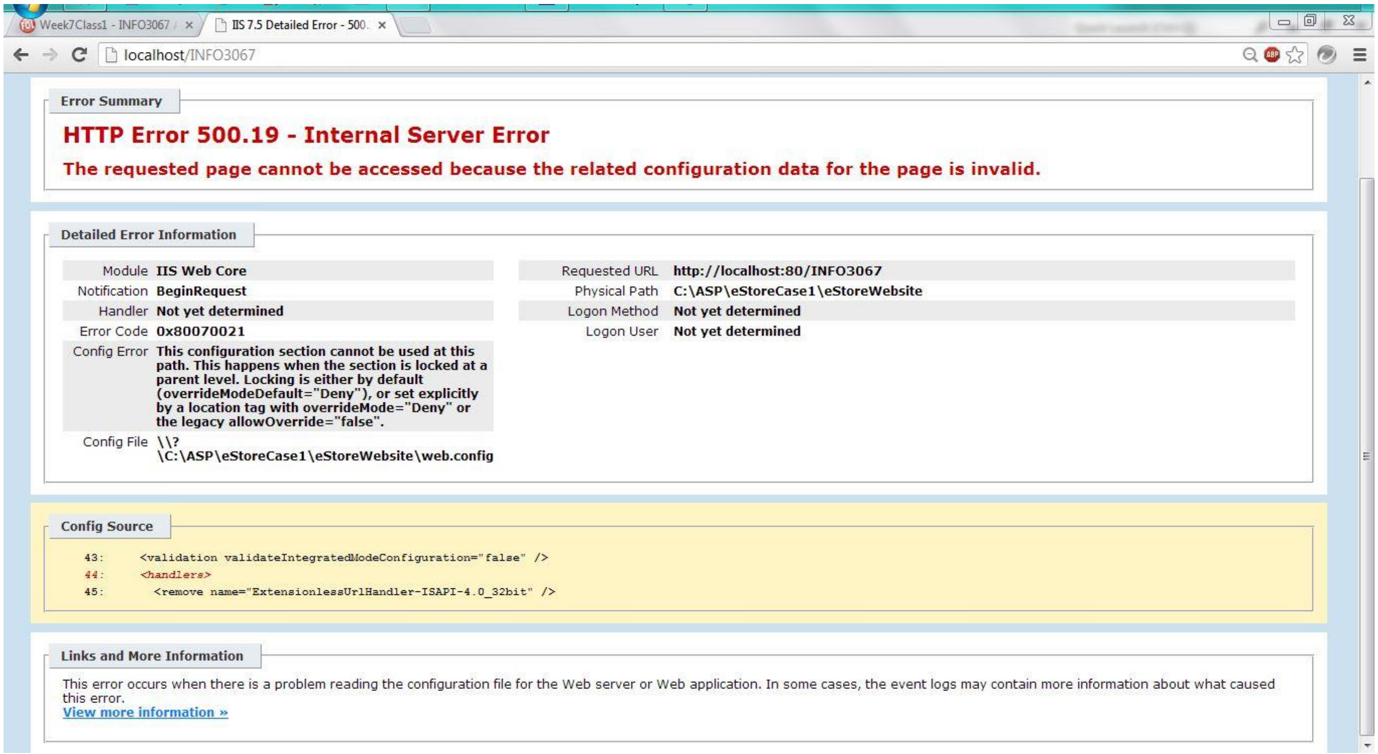
- Install ASP.NET if you are using managed handler.
- Ensure that the handler module's name is specified correctly. Module names are case-sensitive and use the format modules="StaticFileModule,DefaultDocumentModule,DirectoryListingModule".

Links and More Information

IIS core does not recognize the module.
View more information »

To fix this open a cmd window up as “administrator” and change directories to the path shown below, then run **aspnet_regiis –I** from a command prompt

```
cd\
cd C:\Windows\Microsoft.NET\Framework64\v4.0.30319
aspnet_regiis.exe -i
```



Error Summary

HTTP Error 500.19 - Internal Server Error

The requested page cannot be accessed because the related configuration data for the page is invalid.

Detailed Error Information

Module	**IIS Web Core**
Notification	**BeginRequest**
Handler	**Not yet determined**
Error Code	**0x80070021**
Config Error	**This configuration section cannot be used at this path. This happens when the section is locked at a parent level. Locking is either by default (overrideModeDefault="Deny"), or set explicitly by a location tag with overrideMode="Deny" or the legacy allowOverride="false".**
Config File	**\\? \C:\ASP\eStoreCase1\eStoreWebsite\web.config**

Requested URL	**http://localhost:80/INFO3067**
Physical Path	**C:\ASP\eStoreCase1\eStoreWebsite**
Logon Method	**Not yet determined**
Logon User	**Not yet determined**

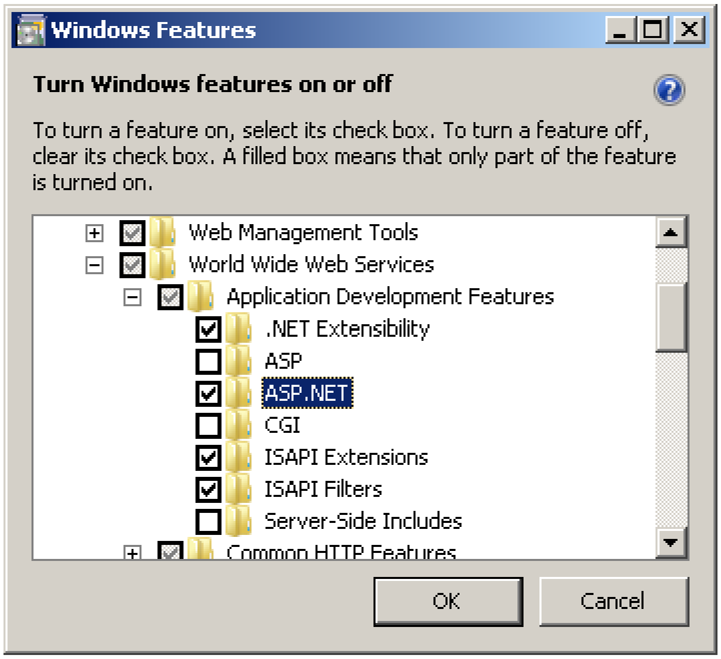
Config Source

```
43:     <validation validateIntegratedModeConfiguration="false" />
44:     <handlers>
45:       <remove name="ExtensionlessUrlHandler-ISAPI-4.0_32bit" />
```

Links and More Information

This error occurs when there is a problem reading the configuration file for the Web server or Web application. In some cases, the event logs may contain more information about what caused this error.
View more information »

500.19 is caused by Asp.net not being installed correctly, return to the control panel and choose asp.net: as follows:



Windows Features

Turn Windows features on or off

To turn a feature on, select its check box. To turn a feature off, clear its check box. A filled box means that only part of the feature is turned on.

- ☑ Web Management Tools
- ☑ World Wide Web Services
  - ☑ Application Development Features
    - ☑ .NET Extensibility
    - ☐ ASP
    - ☑ ASP.NET
    - ☐ CGI
    - ☑ ISAPI Extensions
    - ☑ ISAPI Filters
    - ☐ Server-Side Includes
  - ☑ Common HTTP Features

Then restart the web server

One other problem has cropped up:

- If your solutiion is **in c:\users**…. You will need to move it to somewhere else as the default privileges with that folder will cause IIS 7 some problems. I suggest copy the entire solution to c:\temp for demonstration purposes. Then install the application as in the video.

Other Things to Check

- Make sure your main menu is using the ~ character on all menu items:
  - `<li> <a href="~/Shop">Shop</a></li>`
- Make sure your image urls start with Content/… not /Content….
- Make sure images referenced in helpers and java script just point to the folder: img/somegraphic.png

Final Note

I've posted the 2nd case study formal requirements in the content area as well as a Final Exam review .pdf. We will do the exam in the first hour next class. Then we'll do case study demonstrations in hours 2 and 3.