ASP.NET MVC Boilerplate

From OWASP and the Template Readme File

Introduction to MVC Boilerplate

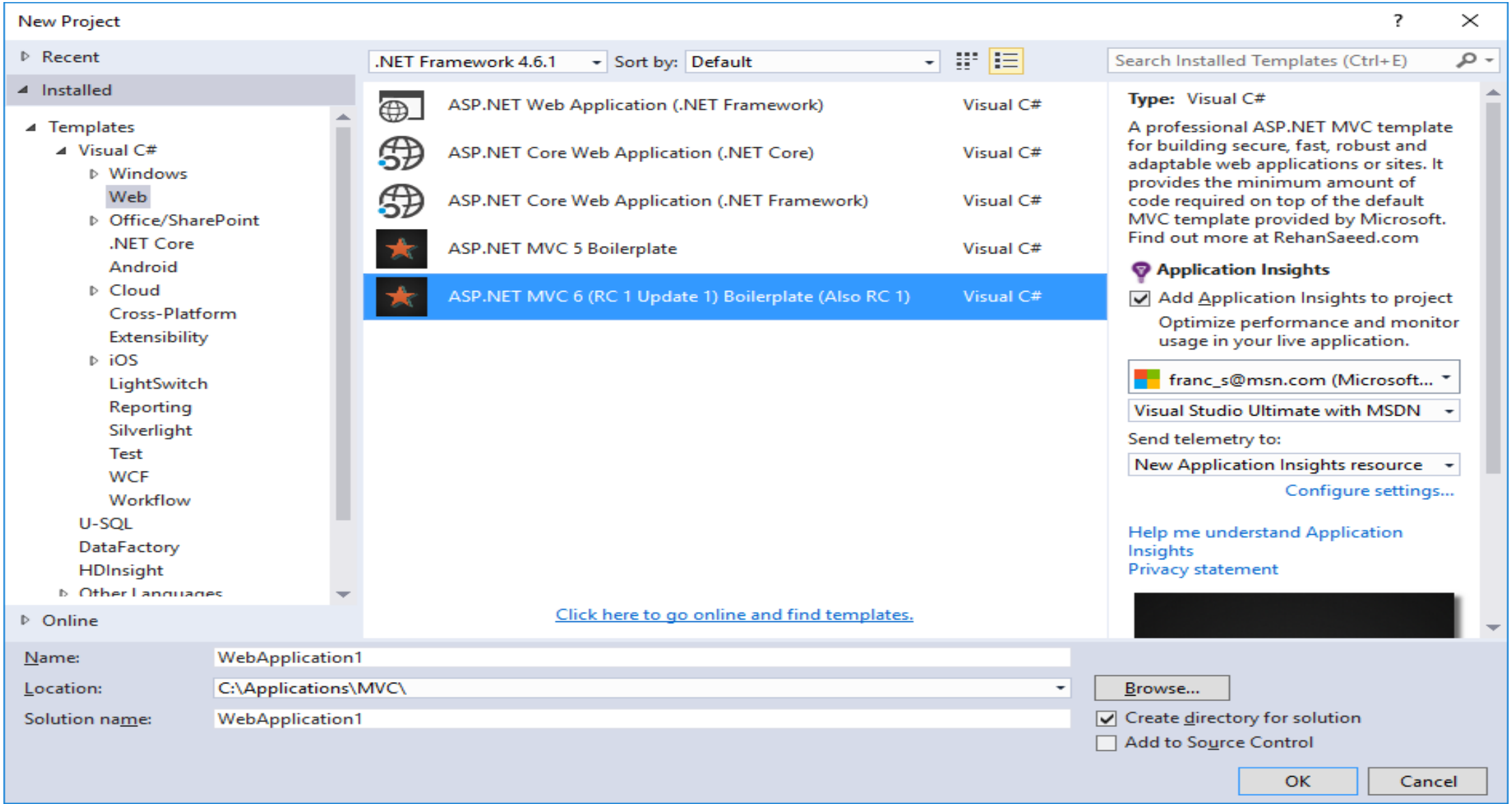
- The default ASP.NET MVC project template uses insecure defaults and omits many security features altogether.
- ASP.NET MVC Boilerplate is a Visual Studio project template that enables security features by default and adds liberal comments and links to further resources to help developers get started.
- Go to link for OWASP Presentation: https://www.owasp.org/index.php/OWASP_ASP.NET_MVC_Boilerplate_Project
- Also: http://rehansaeed.com/asp-net-mvc-boilerplate/

Overview: MVC Boilerplate (OWASP)

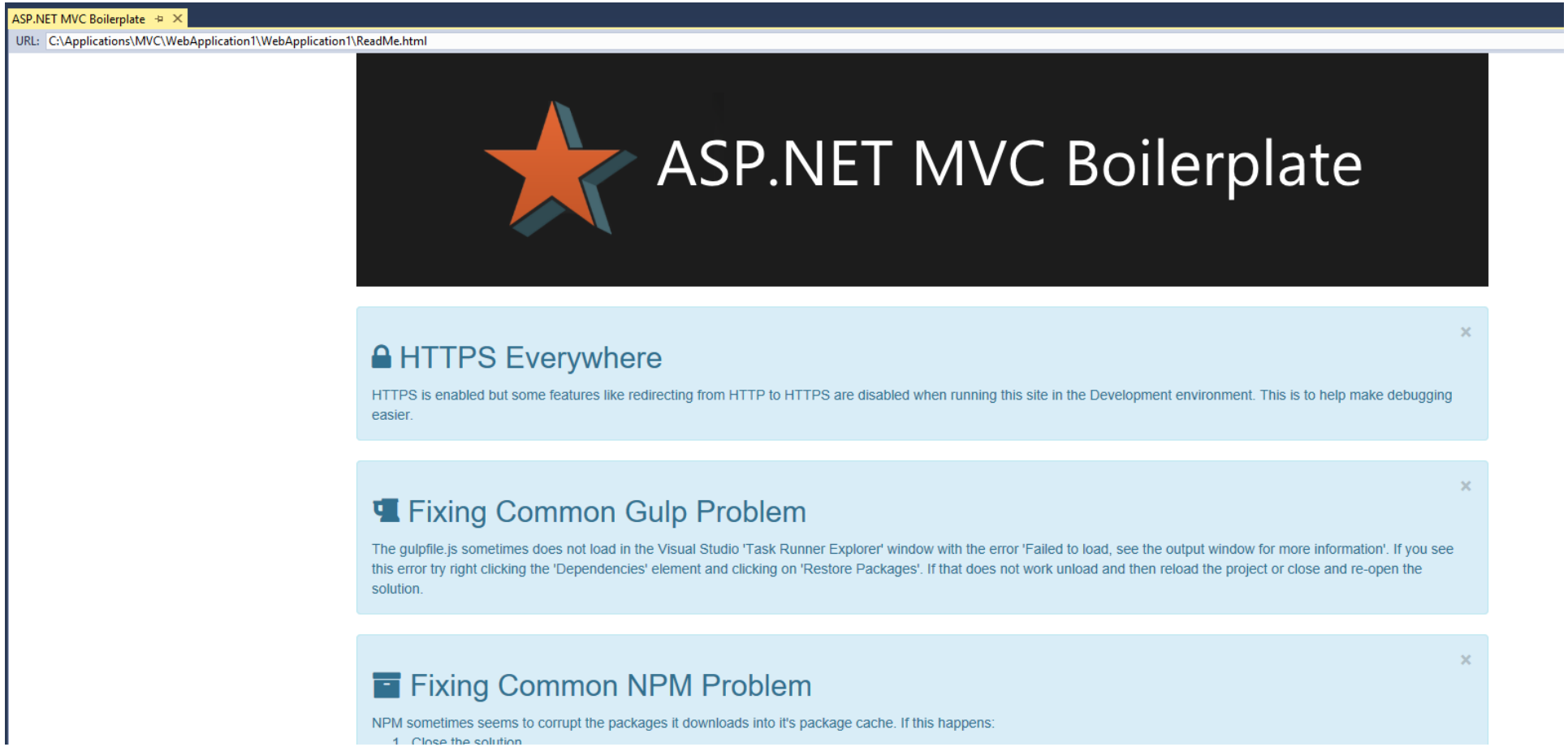
The main benefits of using this template are:

- Security
- Performance
- Search Engine Optimization (SEO)
- Accessibility
- Browser Compatibility
- Resilience and Error Handling
- Easier Debugging and Performance Testing Tools
- Patterns and Practices
- Atom Feed
- Search
- Social Media Support

Creating an MVC Boilerplate Project in VS 2015



The MVC Boilerplate Template



The Task Check List for Pre-Requisites

Task Check List

This is a check-list of tasks you must perform to get your site up and running faster. Open this file in the browser of your choice and it will remember what you have checked, just don't clear your cache.

Pre-Requisites

- ☐ **Update Visual Studio** - Update your version of Visual Studio with all patches and updates.
- ☐ **Update NuGet** - Update the NuGet Visual Studio extension from the Tools -> Extensions and Updates menu.
- ☐ **Update Visual Studio ASP.NET Core Tools and DNVM** - Update the Visual Studio ASP.NET Core tools and the DNVM to RC1 Update 2. Find out more here.
- ☐ **Update TypeScript** - Update the TypeScript extension from the Tools -> Extensions and Updates menu.
- ☐ **Update NuGet Package Source** -
  1. Go to Visual Studio -> Tools -> Options -> NuGet Package Manager -> Package Sources
  2. Enter a new package source named 'ASP.NET Core Master' and source as the following (Without quotes):

     ```
     https://www.myget.org/F/aspnetmaster/api/v2
     ```

- ☐ **IIS HttpPlatformHandler** - If you are using IIS and configuring it yourself, install the HttpPlatformHandler from here.

Server Security Checklist

- **Security**
- **TLS over HTTPS**
- **Configure Site-Wide HTTPS** - Please note that SSL has been superseded by TLS. SSL is vulnerable to the POODLE security vulnerability and should not be used. These steps outline how to secure your site so that all requests and responses are made over HTTPS using TLS, you should consider using it across your whole site for best security, rather than having a mix of HTTP and HTTPS pages. TLS certificates can be obtained for free at:
  - LetsEncrypt.org
  - StartSSL.com
  - Digicert (Digicert is only for Microsoft employees or MVP's)
- You can follow this guide to setting up TLS using StartSSL.
  - Open the Startup.cs file and uncomment the application.UseHpkp(...); line to turn on Public Key Pinning.
  - If using Strict-Transport-Security, submit your domain to the HSTS Preload site so that your domain can be preloaded using HTTPS rather than HTTP See this for more information about preloading.
  - Use the SSLLabs.com site to check that you have implemented TLS over HTTPS correctly.

Content Security Policy

Content Security Policy (CSP)

- [ ] **Configure Content Security Policy** - Content Security Policy (CSP) is a great new HTTP header that controls where a web browser is allowed to load content from and the type of content it is allowed to load. It uses a white-list of allowed content and blocks anything not in the allowed list. It gives us very fine grained control and allows us to run our site in a sandbox in the users browser. Learn more about it here, then Configure your CSP policy globally in Startup.Filters.cs or on your individual MVC controllers and actions if they have special access requirements.

- [ ] **Configure Report URI** - Register with Report URI to get a URL where you can send your CSP violation reports. Then add this URL to the CspReportUriAttribute in Startup.ContentSecurityPolicy.cs.

- [ ] **Check Modernizr CSP Support** - Modernizr does not currently support Content Security Policy (CSP) because it uses in-line CSS styles rather than separate files. To get around this problem, in-line Styles have been set to allowed in the CSP policy (See Startup.Filters.cs). This is not a good practice and reduces some of the security benefits of using it. Keep an eye on this StackOverflow post and this issue raised on the Modernizr development site for when Modernizr finally adds support for CSP.

Other Security

Other Security

- ☐ **Use Secret Store** - If you want to store connection strings or a machine key, use the secret store. See this and this for more information.
- ☐ **Adjust Request Limits** - There are settings in the Web.config file under the requestLimits element that limit maximum size of the requests clients can make to your site. You can limit the maximum content size, maximum URL length and maximum query string length. You should lower these as much as possible, while still having a working site.
- ☐ **Enable Retail Mode** - Enable retail mode in the machine.config file on the production server you are hosting your site on. You can find the file in the locations below:
  - 32-bit - %windir%\Microsoft.NET\Framework\[.NET Version]\config\machine.config
  - 64-bit - %windir%\Microsoft.NET\Framework64\[.NET Version]\config\machine.config
- ☐ **Keep NuGet Up To Date** - Keep your NuGet packages up to date to patch security vulnerabilities. See the section about updating below.
- ☐ **Understand OWASP Top 10** - Understand the top ten security vulnerabilities and how you can protect yourself from them at OWASP Top 10. Also take a look at the OWASP Top Ten Cheat Sheet.
- ☐ **Understand Cross Site Scripting (XSS)** - Understand Cross Site Scripting (XSS) security vulnerabilities and how you can protect yourself from them using the OWASP XSS Cheat Sheet and OWASP DOM Based XSS Cheat Sheet.
- ☐ **Understand Cross Site Request Forgery (CSRF)** - Understand Cross Site Request Forgery security vulnerabilities and how you can protect yourself from them using the OWASP CSRF Cheat Sheet.
- ☐ **Check Site On ASafaWeb.com** - Scan your site for security vulnerabilities at ASafaWeb.com.
- ☐ **Upgrade to .NET 4.6** - Upgrade to .NET 4.6 to enable randomized hashes.
- ☐ **Turn on Azure SQL Database Threat Detection** - If you are using SQL Database on Azure, then turn on Threat Detection.

Performance

Performance

Caching

- ☐ **Configure Caching** - The sitemap.xml, robots.txt and error pages are generated programatically and then cached for a day. If these resources won't change much on your site, change the length of time they are cached. Open the config.json file and take a look at the cache profiles section.
- ☐ **Add More Caching** - Use the ResponseCache attribute in conjunction with the cache profiles section in the config.json file to cache pages that don't change often and do not contain sensitive information.

Content Delivery Networks (CDN)

- ☐ **Upload Static Files to CDN** - Upload the static files to a CDN for vastly better performance. examples of static content include your images, minified CSS bundle content, minified JavaScript bundle content, etc. examples on how to do this are shown here.

Images

- ☐ **Save Images For Web** - Reduce the size of your images by saving them so that they are as small as possible while still looking good. The quality of JPG's can be reduced for example. Some image applications like PhotoShop have a save for the web feature.
- ☐ **Compress Images** - Run the 'optimize-images' Gulp task from the 'Task Runner Explorer' in Visual Studio. This will compress the images in the wwwroot/img folder.

Performance (Cont’d)

Benchmarking

- ☐ **Run Google Page Speed** - Use Google Page Speed to benchmark your sites performance and to get suggestions on how to further improve performance. Note that this template is already very quick to begin with.
- ☐ **Run Yahoo's YSlow** - Use Yahoo's YSlow to benchmark your sites performance and to get suggestions on how to further improve performance.

Other Performance

- ☐ **Disable ETags** - If running IIS 7.5 or below, disable ETags. Open the web.config file and search for 'setEtag' for further instructions.
- ☐ **Pre-fetch/Pre-render Pages** - If there are subsequent pages that a user is likely to visit, such as the next link on a gallery page, consider adding pre-fetch or pre-render link tags to the head of the page. See here and here for more details. Also consider implementing IE11 flip ahead, if you have next and previous buttons.
- ☐ **Enable Windows Server Pre-fetcher** - This is only relevant if your site is hosted on Windows Server. You can enable the pre-fetcher to get a performance and reduce the disk-read cost of application start-up. Click here for more information on how to do this.

Code Quality

Code Quality

- [ ] **HTML Validation** - Validate your HTML using the W3C's HTML Validator and/or using Dr. Watson.
- [ ] **CSS Validation** - Validate your CSS using the W3C's CSS Validator.
- [ ] **Run CSSLint** - Building this project will run the 'lint-css' Gulp task which will print warnings in the Visual Studio 'Task Runner Explorer' window and give you pointers in how you can improve your CSS & LESS code.
- [ ] **Run JSHint** - Building this project will run the 'lint-js' Gulp task which will print warnings in the Visual Studio 'Task Runner Explorer' window and give you pointers in how you can improve your JavaScript & TypeScript code.
- [ ] **Internationalization Validation** - If you are supporting multiple languages, validate your CSS using the W3C's Internationalization Validator.
- [ ] **Fix Broken Links** - Use the W3C's Link Checker or IWebTool.com's Link Checker to fix any broken links in your site.
- [ ] **Run Spell Check** - Check the spelling on your site for any spelling errors using an Online Spell-Checker.

Compatibility

Compatibility

- [ ] **IIS 7.5/8** - Some Web.config settings do not exist in older versions of IIS (7.5 and 8) so you must edit the Web.config file and remove the dynamicIpSecurity settings.
- [ ] **Test Browser Compatibility** - Test the compatibility of your site with various browsers at BrowserShots.org, Browserling.com and Spoon.net.
- [ ] **Write Forms Properly** - Forms are hard to write, ensure you read this tutorial on how to do them properly and don't forget to set the input types.
- [ ] **Test Site On Mobile** - Test your site on real mobile devices here. Ensure that the site is still fully functional. Bootstrap helps a lot in this regard.
- [ ] **Run Modern.IE Scan** - Scan your site for issues with at Modern.IE.
- [ ] **Printer Friendly (Save Some Trees)** - Ensure that you use the Bootstrap Print Classes to hide content from printers. Use the print preview (Please don't actually print) feature to optimize your site for printing.

Authentication and Authorization

Authentication & Authorization

☐ **Choose Authentication Provider** - If you want no authentication, then you don't need to do anything. **Do not** build your own authentication provider unless you are a security expert, it is much harder than it looks and even professionals get it wrong. Choose from one of the options below:

1. ASP.NET Identity - This is basically forms authentication with Open Auth. it is what you get when you select 'Individual Account' from Microsoft's 'Change Authentication' dialogue, when you create a new Web Project. Follow this tutorial. Don't forget to add the NoLowercaseQueryStringAttribute to the AccountController to enable the RedirectToCanonicalUrlAttribute to work.
2. Windows Authentication - Follow this tutorial.

Error Handling

⚠ Resilience and Error Handling

☐ **Configure Dynamic IP Security** - Update the Dynamic IP Security limits in the dynamicIpSecurity section of the web.config file. This feature of IIS prevents Denial of Service (DoS) attacks from taking down your site. Dynamic IP Security is currently set to logging only mode, where IIS will log requests from the client that would be rejected without actually rejecting them. You should run your site in this mode for some time and slowly raise the limits until you get no more 403.501 or 403.502 errors. Normal legitimate requests that can look like a DoS attack are:

- Google and Bing often make large numbers of requests that can look like a DoS attack and you do not want to block them by accident.
- Some organizations put themselves behind a proxy and their requests can look like they are from a single IP address.
- Pages on your site that cause a large number of requests to be made your site may trigger the Dynamic IP Security limits to be triggered and the page loading will be treated like a Denial of Service (DoS) attack.

You can test this feature by recording a request to your site using Fiddler and then replaying it many times.

Accessibility & Debugging

Accessibility

- ☐ **Understand Accessibility Problems** - 4% of the world population is estimated to be visually impaired, while 0.55% are blind. Get more statistics here. Read a list of common accessibility problems here.
- ☐ **Add ARIA Markup** - Ensure that your site is accessible by adding aria attributes to your HTML markup. Read a practical guide from the W3C here and documentation about ARIA from Mozilla here.
- ☐ **Check Site Accessibility** - Use WebAim, the Accessibility Inspector Firefox extension or the Web Accessibility Checker to test the accessibility of your site.
- ☐ **Check Site Colour Contrast** - Use CheckMyColours.com to check your site for problems with colour contrast. This will help the colour blind.
- ☐ **Check Site With Screen Reader** - Try the free NVDA Screen Reader to test the accessibility of your site.

</> Debugging

- ☐ **Learn Glimpse** - Navigate to {your site}/glimpse and also learn how to use Glimpse here. Glimpse can also be used to performance tune your application. It records the length of time taken to perform various actions.

Securing Cookies

- By default JavaScript from external sites can access the cookies from the default ASP.NET MVC template.
- They can also be sent unencrypted over the wire, because they don’t use SSL.
- The httpCookies section can be added to secure your cookies

```
<httpCookies httpOnlyCookies="true" requireSSL="false" />
```

Prevent ASP.NET HTTP headers

- By default ASP.NET sends HTTP headers with each response telling what version of ASP.NET your site is hosted on and even what version of MVC you are using.
- To fix this problem you need to first is to set the enableVersionHeader setting on the httpRuntime section to false.

```
<httpRuntime targetFramework="4.6" enableVersionHeader="false" />
```

- Second, you need to clear the custom headers as shown below.

```
<httpProtocol>
  <customHeaders>
      <clear />
  </customHeaders>
</httpProtocol>
```

ASP.NET Session Cookie

- Rename the ASP.NET session cookie from its default name of ‘ASP.NET_SessionId’ to ‘AnythingElse’.

<sessionState cookieName="AnythingElse " />

Removing Tracing

- Enabling tracing while debugging is done with a single line of config:

```
<system.web>
  <trace enabled="true"/>      Should be set to false in release build
</system.web>
```

- Tracing information can be easily viewed by navigating to http://YourSite/trace.axd
- remove the tracing HTTP handlers altogether to stop site responding to this URL

```
<handlers>       Should only be done in release build
    <remove xdt:Transform="Insert" name="TraceHandler-Integrated" />
    <remove xdt:Transform="Insert" name="TraceHandler-Integrated-4.0" />
  </handlers>
```

403.14 Forbidden Responses to Directories

- Navigating to a directory using IIS and ASP.NET MVC can cause a 403 Forbidden response to be returned
- IIS reveals that directory browsing is disabled (As it should be, directory browsing is a severe security risk that allow attackers to see the Web.config file with connection strings in it)
- The way to fix this is to handle 403.14 errors and replace the response with a standard 404 Not Found

```
<system.webServer>
  <defaultDocument enabled="false"/>
</system.webServer>
```

- Now navigating to ‘/Content’ will return 404 Not Found

NWebSec

- The NWebSec NuGet packages written by André N. Klingsheim are a great way to add additional security to a ASP.NET MVC site.
- The ASP.NET MVC Boilerplate project template includes them by default.
- Everything is preconfigured and commented as much as possible out of the box but remember this is a project template to get you started.
- You still need to put the effort in to customize the site security to your own requirements and put in some time learning about what each of the security features that follow does and how best to use it.

X-Frame-Options

- The X-Frame-Options HTTP header stops click-jacking by stopping the page from opening in an iframe or only allowing it from the same origin (your domain). There are three options to choose from:
- **Deny** – Specifies that the X-Frame-Options header should be set in the HTTP response, instructing the browser to display the page when it is loaded in an iframe – but only if the iframe is from the same origin as the page.
- **SameOrigin** – Specifies that the X-Frame-Options header should be set in the HTTP response, instructing the browser to not display the page when it is loaded in an iframe.
- **Disabled** – Specifies that the X-Frame-Options header should not be set in the HTTP response.

```
// Filters is the GlobalFilterCollection from GlobalFilters.Filters
filters.Add(
    new XFrameOptionsAttribute()
    {
        Policy = XFrameOptionsPolicy.Deny
    });
```

Strict-Transport-Security

- This HTTP header is only relevant for TLS.
- Ensures that content is loaded over HTTPS and refuses to connect in case of certificate errors and warnings.
- Use the Owin (Using the added NWebSec.Owin NuGet package) extension to apply it.

```
app.UseHsts(options =>
options.MaxAge(days:30).IncludeSubdomains());
```

- Using HTTPS throughout the site is shown below:

```
filters.Add(new RequireHttpsAttribute());
```

X-Content Options

- This HTTP header stops IE9 and below from sniffing files and overriding the Content-Type header (MIME type) of a HTTP response.

filters.Add(new XContentTypeOptionsAttribute());

- This HTTP header stops the automatic downloading and opening of your HTML pages by browsers which then go on to run the page as if it were part of a site. It forces the user to save the page and manually open the HTML document.

filters.Add(new XDownloadOptionsAttribute());

- These filters are added by default in ASP.NET MVC Boilerplate.

Summary

- Went over the new OWASP .NET MVC Security Boilerplate template