Frontend Optimization.
JS and CSS Bundling and Minification

R.JS (require.js)

```
> npm install -g requirejs
> r.js -o app.build.js

({
    baseUrl: "../../Scripts",
    paths: {
        "d3": "d3",
        "jquery": "jquery-2.0.3",
        "bootstrap": "bootstrap",
        "select2": "select2/select2",
    },
    name: "../App/EntryPoint",
    exclude: ["JsAction", "signalr", "Config"],
    optimize: "uglify2",
    generateSourceMaps: true,
    preserveLicenseComments: false,
    out: "app.js"
})

<script type="text/javascript" src="app.js"></script>
```

Microsoft ASP.NET Web Optimization Framework

```
> Install-Package Microsoft.AspNet.Web.Optimization

 public class BundleConfig
   {
      public static void RegisterBundles(BundleCollection bundles)
      {
         ScriptBundle thirdPartyScripts = new ScriptBundle("~/Scripts/ThirdParty");
         thirdPartyScripts.Include("~/Scripts/jquery-2.0.3.js",
                        "~/Scripts/bootstrap.min.js");
         bundles.Add(thirdPartyScripts);
         BundleTable.EnableOptimizations = true;
      }
   }

BundleConfig.RegisterBundles(BundleTable.Bundles);

@Scripts.Render("~/Scripts/ThirdParty")
```

System.Web.Optimization

```
var lessBundle = new Bundle("~/st-style-common-2")
    .IncludeDirectory("~/Admin/Resources/styles/", "*.less");
lessBundle.Transforms.Add(new LessTransform());
lessBundle.Transforms.Add(new CssMinify());
bundles.Add(lessBundle);

@Scripts.Render("~/st-style-common-2")
```

HTTP GZIP compression

1. Open IIS server by using inetmgr from windows Run command box.
2. Click on sites. It shows you all the sites hosted on this server. If you want to to enable http compression for all sites you can configure settings by clicking on server name.
3. Select the site for which you want to configure http compression under Sites node.
4. From right pane of IIS manager click on Compression.
5. Select Enable dynamic content compression for dynamic contents.
6. Select Enable static Content compression for static contents.

```
<system.webServer>
<httpCompression directory="%SystemDrive%\inetpub\temp\IIS Temporary Compressed Files">
   <scheme name="gzip" dll="%Windir%\system32\inetsrv\gzip.dll" staticCompressionLevel="9" />
   <dynamicTypes>
   <add mimeType="text/*" enabled="true" />
   <add mimeType="message/*" enabled="true" />
   <add mimeType="application/x-javascript" enabled="true" />
   <add mimeType="application/json" enabled="true" />
   <add mimeType="*/*" enabled="false" />
   </dynamicTypes>
   <staticTypes>
   <add mimeType="text/*" enabled="true" />
   <add mimeType="message/*" enabled="true" />
   <add mimeType="application/x-javascript" enabled="true" />
   <add mimeType="application/atom+xml" enabled="true" />
   <add mimeType="application/xaml+xml" enabled="true" />
   <add mimeType="*/*" enabled="false" />
   </staticTypes>
</httpCompression>
<urlCompression doStaticCompression="true" doDynamicCompression="true" />
</system.webServer >
```

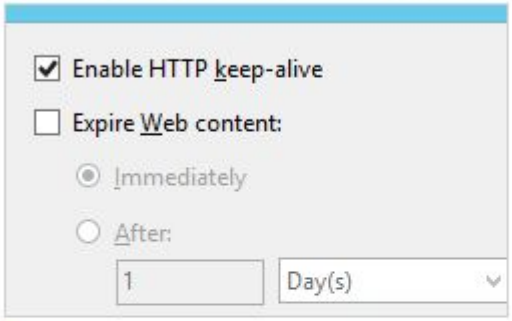
HTTP GZIP compression

```
public class CompressAttribute : ActionFilterAttribute
  {
     public override void OnActionExecuting(ActionExecutingContext filterContext)
     {
        var request = filterContext.HttpContext.Request;
        var response = filterContext.HttpContext.Response;

        CompressResponse(request, response);
     }

     public static void CompressResponse(HttpRequestBase request, HttpResponseBase response)
     {
        var acceptEncoding = request.Headers["Accept-Encoding"];
        if (string.IsNullOrEmpty(acceptEncoding)) return;
        acceptEncoding = acceptEncoding.ToUpperInvariant();
        if (acceptEncoding.Contains("GZIP"))
        {
           response.AppendHeader("Content-encoding", "gzip");
           response.Filter = new GZipStream(response.Filter, CompressionMode.Compress);
        }
        else if (acceptEncoding.Contains("DEFLATE"))
        {
           response.AppendHeader("Content-encoding", "deflate");
           response.Filter = new DeflateStream(response.Filter, CompressionMode.Compress);
        }
     }
  }
```

HTTP expire header

```
<configuration>
<system.webServer>
   <staticContent>
     <clientCache cacheControlCustom="public"
      cacheControlMaxAge="30.00:00:00" cacheControlMode="UseMaxAge" />
   </staticContent>
  </system.webServer>
</configuration>
```

5. In the **Set Common HTTP Response Headers** dialog box, check the box to expire Web content, select the option to expire after a specific interval or at a specific time, and then click **OK**.



HTTP expire header

```
public class NoCache : ActionFilterAttribute
  {
     public bool Disable { get; set; }

     public override void OnResultExecuting(ResultExecutingContext filterContext)
     {
        if (!Disable)
        {
           filterContext.HttpContext.Response.Cache.SetExpires(DateTime.UtcNow.AddDays(-1));
           filterContext.HttpContext.Response.Cache.SetValidUntilExpires(false);
           filterContext.HttpContext.Response.Cache.SetRevalidation(HttpCacheRevalidation.AllCaches);
           filterContext.HttpContext.Response.Cache.SetCacheability(HttpCacheability.NoCache);
           filterContext.HttpContext.Response.Cache.SetNoStore();
        }
        base.OnResultExecuting(filterContext);
     }
  }
```