فصل نهم :

مبحث Authentication در ASP.NET

مقدمه :

ASP.NET سرویس های تصدیق هویت و اعتبار سنجی را که جزئی از IIS می باشد ، فراهم نموده است. در اغلب اوقات مبحث امنیت آخرین موضوعی است که برنامه نویسان به آن فکر می کنند و این تفکر با اینچنین سؤالاتی آغاز می شود : چگونه می توان اطمینان حاصل کرد که عده ای نتوانند به یک قسمت از برنامه دسترسی پیدا کنند؟ چگونه می توان اعتبار سنجی نمود که آیا درخواست فرستاده شده به برنامه از طرف شخصی است که باید اینکار را انجام دهد یا خیر؟ و غیره. یکی از مباحث مهم در برنامه نویسی وب ، مبحث تعیین اعتبار و تصدیق هویت کاربران می باشد. البته این موضوع به محتویات یک سایت هم بستگی پیدا می کند. سایت هایی که کار اطلاع رسانی از هر نوعی را انجام می دهند عموما کاربران ناشناس را به سادگی می پذیرند و برایشان اهمیت ندارد که چه کسانی از سایت بازدید می کنند اما برنامه های حساس تر مانند فروشگاه های آنلاین و غیره اینچنین نیستند.

توضیحاتی در مورد دو واژه ی متداول:

Authentication : عملیاتی که در طی آن مشخص می گردد شخص مراجعه کننده به سایت کیست. این عملیات با ارائه ی نام کاربری و پسورد صورت می گیرد.

Authorization : عملیات اعطای مجوز برای دسترسی به یک سری از منابع سایت و یا عکس آن.

سه حالت Authentication در ASP.NET :

ASP.NET از سه حالت forms ، Windows و Passport Authentication استفاده می نماید. بیشترین استفاده ، حالت Forms Authentication می باشد که در این فصل به صورت مشروحی مورد بررسی قرار خواهد گرفت.

Windows Authentication : بر روی اینترانت ها مفید است تا اینترنت. در این حالت هرکاربر دسترسی پیدا کننده به سایت داخلی ، یک یوزر تعریف شده در ویندوز سرور می باشد.

Passport Authentication : استفاده از سرویس پاسپورت مایکرسافت ؛ که احتمالا نمونه ی آنرا در سایت وب ماتریکس هاستینگ مشاهده نموده اید.

Forms Authentication : در ادامه بحث خواهد گردید.

مراحل یک Forms Authentication به صورت زیر است:

۱- کلاینت درخواستی را برای مشاهده ی یک صفحه ی محافظت شده می فرستد.

۲- IIS در خواست را دریافت می کند ، اگر کلاینت توسط IIS تعیین هویت و اعتبار شود به برنامه ی ASP.NET هدایت می شود. در اکثر موارد دسترسی Anonymous داده شده است. بنابراین هر بازدید کننده ای می تواند برنامه را مشاهده نماید. باید خاطرنشان کرد چون از حالت Forms Authentication می خواهیم استفاده کنیم ، از تصدیق هویت و اعتبار IIS نمی توان استفاده کرد.

۳- اگر کلاینت دارای بلیط (کوکی) معتبر نباشد به صفحه ی مشخص شده در LoginURL قسمت Authentication در فایل web.config برنامه ، هدایت می شود. این صفحه باید صفحه ی لاگین باشد (در قسمت بعد توضیح داده خواهد شد) .

۴- تعیین اعتبار و ایجاد کوکی ، در صورت موفقیت لاگین ، انجام خواهد شد.

۵- سپس کلاینت به صفحه ای که در ابتدا درخواست کرده بود هدایت می شود. بعد از این با توجه به حضور کوکی و تا زمان منقضی نشدن آن ، کلاینت به صورت اتوماتیک مجوز مشاهده ی تمام صفحات را خواهد داشت. بعد از منقضی شدن این کوکی ، کاربر دوباره باید لاگین نماید.

تنظیمات مربوط به Forms Authentication :

۱- فعال کردن دسترسی Anonymouse در IIS . (به صورت پیش فرض فعال است)

۲- تنظیم وب کانفیگ به صورت زیر :

```
<configuration>
  <system.web>
    <authentication mode="Forms">
      <forms name=".COOKIEDEMO"
             loginUrl="login.aspx"
             protection="All"
             timeout="30"
             path="/"/>
    </authentication>
    <authorization>
      <deny users="?" />
    </authorization>
  </system.web>
</configuration>
```

جدول ۱- توضیح ویژگی های مربوط به Forms Authentication

ویژگی	توضیح
name	نام کوکی خواهد بود که اطلاعات در آن ذخیره می شود.
protection	نحوه ی محافظت از کوکی است. یکی از مقادیر زیر را می پذیرد: All : در این حالت از الگوریتم triple DES برای رمزگذاری اطلاعات درون آن کمک گرفته می شود (حالت پیش فرض و توصیه شده). None : از رمزگذاری استفاده نمی کند. Encryption : از الگوریتم DES و یا 3DES استفاده می نماید اما تعیین اعتباری را روی

	کوکی انجام نخواهد داد. یعنی کوکی رمزگذاری می شود اما در برابر تغییرات مهاجمان عکس العملی نشان داده نخواهد شد. Validation : کوکی تعیین اعتبار می شود (در برابر تغییرات احتمالی خارج از برنامه) اما رمزگذاری نمی شود.
Timeout	زمان منقضی شدن کوکی (پیش فرض آن ۳۰ دقیقه است).
Path	مسیر ذخیره سازی کوکی

نکته :

۱- در اینجا اگر deny users="?" قرار داده شود تمام کاربران Anonymouse به صفحه ی لاگین هدایت می شوند.

۲- برای استفاده از این روش باید پذیرش کوکی ها در مرورگر وب کامپیوتر کلاینت فعال باشد.

مثال یک :

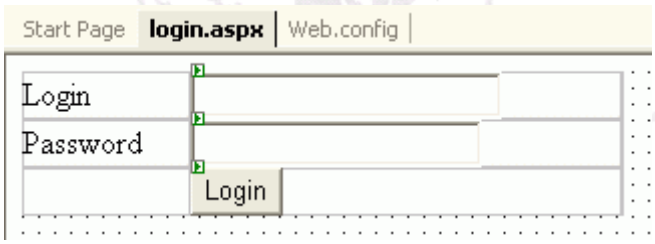
بعنوان یک برنامه نویس ASP.NET بیشتر سروکار ما با Forms Authentication خواهد بود تا مبحث Windows Authentication (فرض کنید سایت شما هزار کاربر دارد. در این حالت باید هزار کاربر در ویندوز تعریف شوند!) . در این مثال برنامه ای را ایجاد خواهیم نمود که با استفاده از Forms Authenticatoin امنیت خود را تامین می کند.

قدم اول:

یک برنامه ی ASP.NET را ایجاد نمایید. نام وب فرم پیش فرض آنرا به login.aspx تغییر دهید . سپس فایل Web.Config آنرا به صورت زیر ویرایش نمایید.

```
    <!--  AUTHENTICATION
          This section sets the authentication policies of the application.
          "Passport" and "None"
    -->
<authentication mode="Forms">
<forms loginUrl="login.aspx" name=".ASPXCOOKIEAUTH" path="/">
<credentials passwordFormat="SHA1">
<user name="User1" password="4A60935FE851C99148EFD66122CDD0F43D5A3059" />
<user name="User2" password="114B3C899D75F5A3FE3FDF83531C42E33214555B" />
<user name="Admin" password="1F95EC61B6EF02B5D2B138654DA138BFDBBC7F3C" />

</credentials>
</forms>
</authentication>
<authorization>
<deny users="?" />
</authorization>
```

همانطور که ملاحظه می نمایید Authentication mode=Forms قرار گرفته است و سه کاربر در آن تعریف شده اند. در ادامه مطابق شکل زیر یک صفحه ی لاگین را طراحی نمایید:



شکل ۱- طراحی ظاهر صفحه ی لاگین

فضای نام آشنای زیر را به برنامه اضافه نمایید :

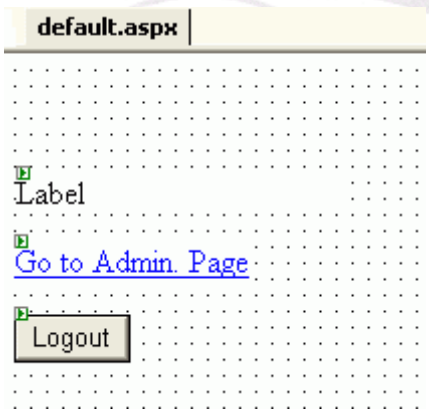
using System.Web.Security;

عمل لاگین نمودن کاربران توسط کد زیر انجام می شود:

```
private void btnSubmit_Click(object sender, System.EventArgs e)
{
      if (FormsAuthentication.Authenticate(txtLogin.Text,
            txtPassword.Text))
      {
            FormsAuthentication.SetAuthCookie(txtLogin.Text, true);
            FormsAuthentication.RedirectFromLoginPage(
                                                txtLogin.Text, true);
      }

}
```

در اینجا می توان صفحات دیگری را به برنامه اضافه نمود (default.aspx) و باید خاطر نشان کرد که تنظیمات امنیتی شامل ساب فولدرهای برنامه نیز می شوند.

این برنامه برای سادگی ، از یک صفحه ی ساده که پیغام خوش آمد گویی را نمایش می دهد تشکیل شده است و با یک لینک به صفحه ی مدیریت همراه است (شکل زیر) :



شکل ۲- طراحی صفحه ی خوش آمد گویی پس از موفقیت کاربر در عملیات لاگین نمودن.

برای نمایش خوش آمد گویی از کد زیر استفاده می شود:

```
private void Page_Load(object sender, System.EventArgs e)
{
      Label1.Text = "Hello " + Context.User.Identity.Name;

}
```

و برای logout از کد زیر کمک می گیریم:

```
private void Button1_Click(object sender, System.EventArgs e)
{
      FormsAuthentication.SignOut();
      Response.Redirect("login.aspx");

}
```

اکنون می توان برنامه را تست کرد. همانطور که از فصل رمزنگاری در ASP.NET بخاطر دارید ، پسوردهای ذخیره شده در اینجا Hash شده بوده و اصل آنها به صورت زیر است:

User Name	Password
User1	1resU
User2	2resU
Admin	AdminAdmin

برای ایجاد قسمت مدیریت برنامه می توان صفحه ای را طراحی نمود که بوسیله ی آن بتوان کاربر جدیدی را اضافه نمود و یا پسوردها را تغییر داد. برای بالاتر بردن امنیت این قسمت یک فولدر مخصوص به نام Admin ایجاد خواهیم کرد به همراه یک web.config مخصوص این فولدر.

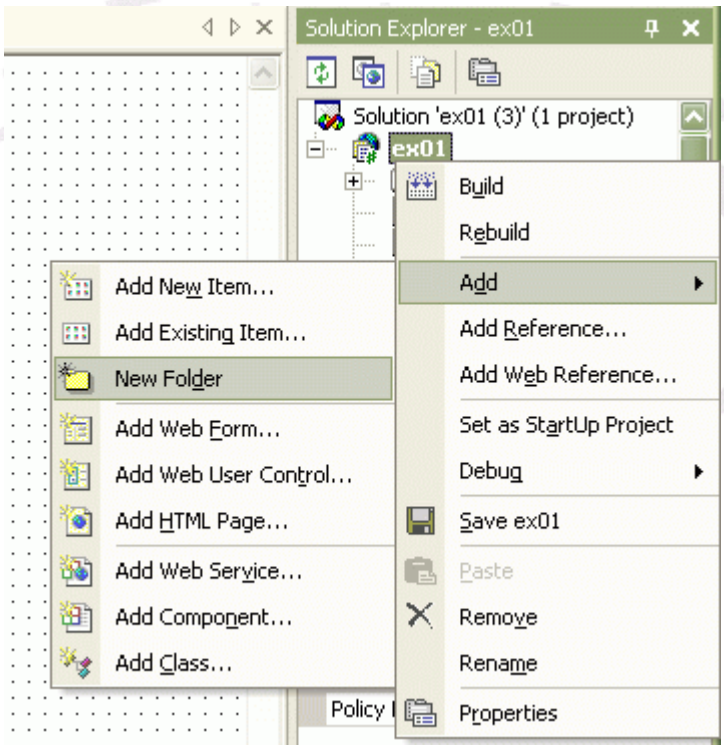
برای اضافه کردن یک فولدر جدید به پروژه ، مطابق تصویر ۳ عمل می شود. فولدر admin را ایجاد نموده و سپس فایل admin.aspx را به آن از منوی پروژه اضافه نمایید.

در ادامه همانند اضافه کردن فولدر جدید ، یک فایل جدید web.config را به این فولدراضافه نمایید (شکل ۴).

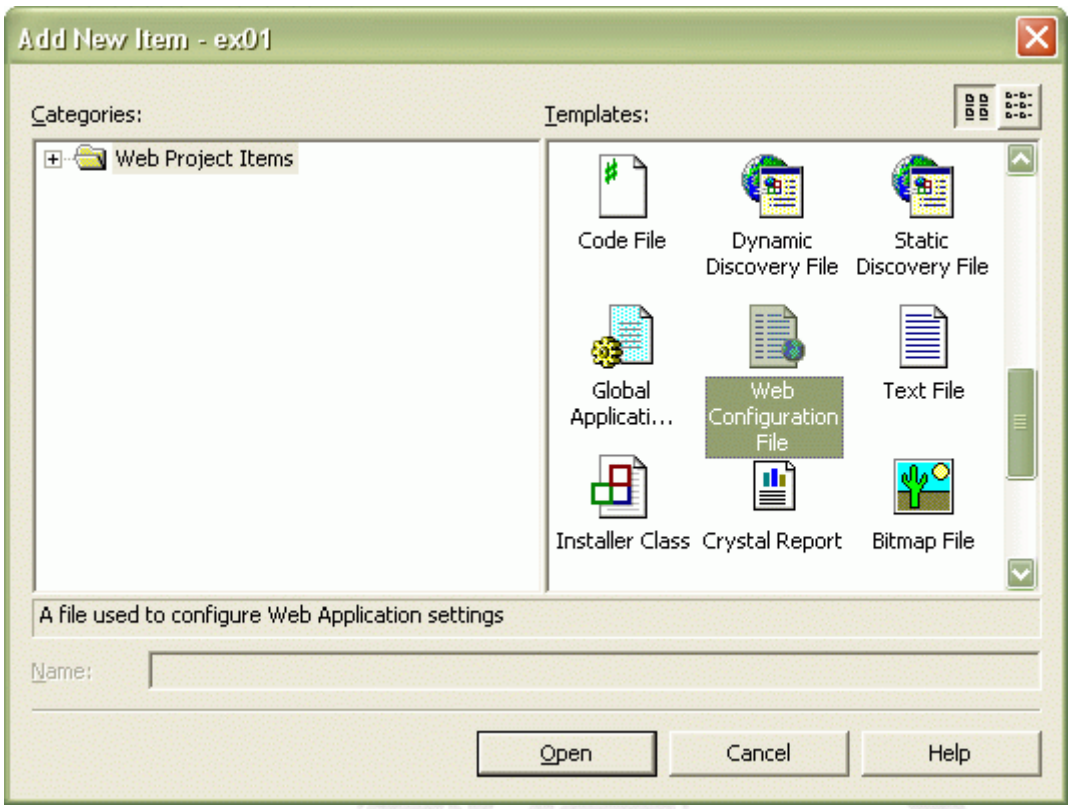
کل محتویات آنرا پاک کرده و سپس آنرا به صورت زیر تغییر دهید:

Admin\Web.config

```
<?xml version="1.0" encoding="utf-8" ?>

<configuration>
<system.web>
<authorization>
<allow users="Admin"/>
<deny users="*" />
</authorization>
</system.web>
</configuration>
```



شکل ۳- اضافه کردن فولدر Admin به پروژه.



شکل ۴- اضافه کردن یک فایل وب کانفیگ جدید به فولدر Admin .

اکنون در صفحه ی default.aspx که لینکی به این صفحه ی مدیریتی وجود دارد مسیر لینک را به Admin/Admin.aspx تغییر دهید. روی صفحه ی admin.aspx یک دیتاگرید قرار دهید.

با توجه به محتویات فوق ، تنها کاربر Admin به این فولدر دسترسی داشته و بلا استثناء بقیه محروم هستند.

در اینجا از دیتاگرید برای نمایش فایل xml وب کانفیگ شامل نام کاربران و پسوردها استفاده خواهیم کرد.

سپس یک دیتاست به صورت زیر تعریف می شود:

```
private DataSet ds;
```

در Page_Load ، این فایل xml را خوانده و به گرید اضافه می کنیم:

```
private void Page_Load(object sender, System.EventArgs e)
{
      ds = new DataSet();
      ds.ReadXml(Server.MapPath("../web.config"));
      DataGrid1.DataSource = ds.Tables["user"];
      if (!IsPostBack)
            DataGrid1.DataBind();

}
```

اکنون با استفاده از Property builder دیتاگرید ، دکمه های Edit ، Update و Cancel را به آن اضافه نمایید.

به رخداد EditCommand دیتاگرید کد زیر را ضافه نمایید:

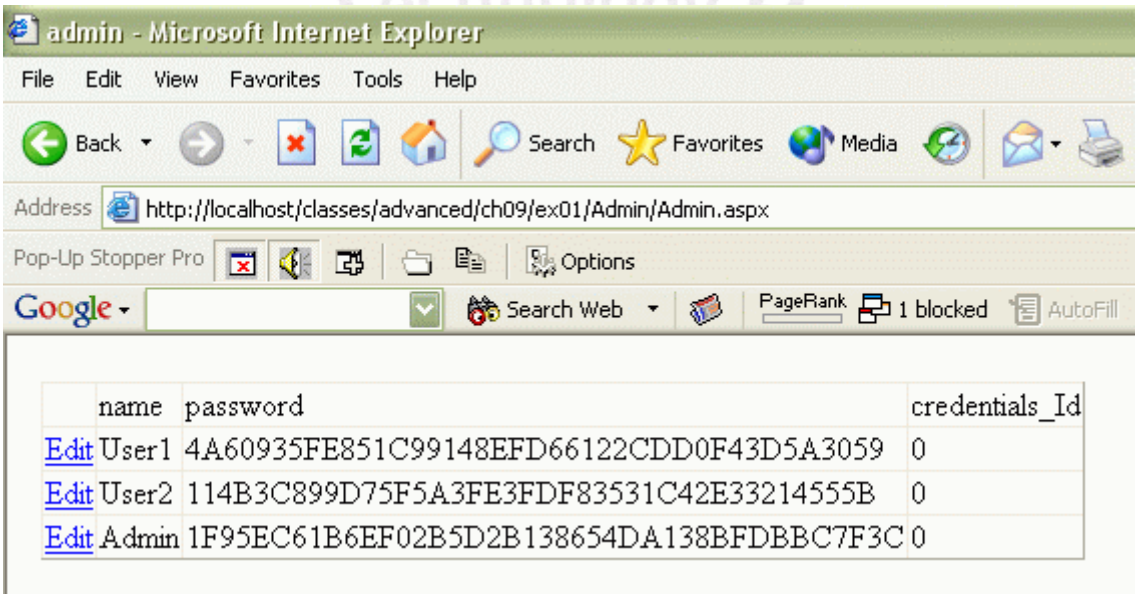
```
private void DataGrid1_EditCommand(object source,
             System.Web.UI.WebControls.DataGridCommandEventArgs e)
{
      DataGrid1.EditItemIndex = e.Item.ItemIndex;
      DataGrid1.DataBind();
}
```

به رخداد CancelCommand کد زیر را اضافه نمایید:

```
private void DataGrid1_CancelCommand(object source,
            System.Web.UI.WebControls.DataGridCommandEventArgs e)
{

      DataGrid1.EditItemIndex = -1;
      DataGrid1.DataBind();
}
```

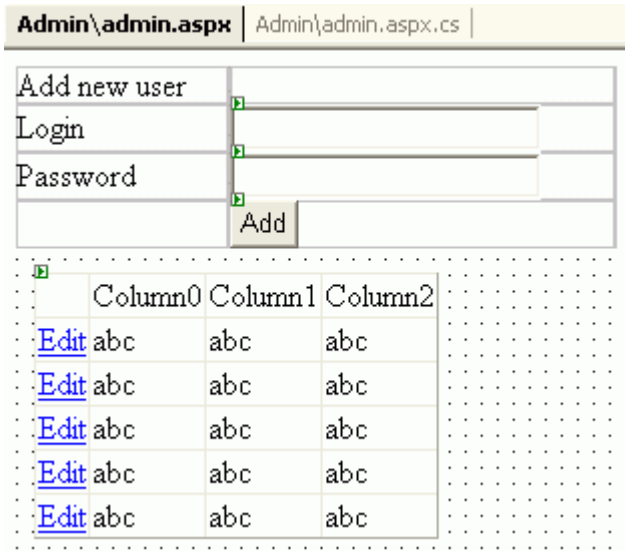
و برای ذخیره کردن تغییرات به رخداد UpdateCommand کد زیر اضافه می شود. لازم به ذکر است که در اینجا پسوردها به صورت hash شده به ادمین نمایش داده می شوند و تنها او می تواند پسورد قبلی را پاک کرده و پسورد جدیدی را وارد نماید. سپس این پسورد دوباره hash خواهد شد.

```
private void DataGrid1_UpdateCommand(object source,
        System.Web.UI.WebControls.DataGridCommandEventArgs e)
{
      int i = e.Item.ItemIndex;
      DataRow oRow = ds.Tables["user"].Rows[i];
      oRow["password"] =
      FormsAuthentication.HashPasswordForStoringInConfigFile(
      ((TextBox)e.Item.Cells[2].Controls[0]).Text, "sha1");

      ds.AcceptChanges();
      ds.WriteXml(Server.MapPath("web.config"));

      DataGrid1.EditItemIndex = -1;
      DataGrid1.DataBind();

}
```



شکل ۵- نمایی از اجرای قسمت تغییر پسوردهای موجود.

برای اضافه کردن کاربر جدید به فرم مدیریتی دو تکست باکس و یک دکمه را مطابق شکل زیر بـه فـرم اضافه نمایید :



شکل ۶- طراحی ظاهر فرم admin.aspx برای ایجاد امکان اضافه نمودن کاربران جدید

```
private void Button1_Click(object sender, System.EventArgs e)
{
      DataRow oRow = ds.Tables["user"].NewRow();
      oRow["name"] = TextBox1.Text;
      oRow["password"] =
        FormsAuthentication.HashPasswordForStoringInConfigFile
        (TextBox2.Text, "sha1");
      oRow["credentials_Id"]=0;
      ds.Tables["user"].Rows.Add(oRow);
      ds.Tables["user"].AcceptChanges();

      ds.WriteXml(Server.MapPath("../web.config"));
      DataGrid1.DataBind();

}
```

Start Page | Admin\admin.aspx | Admin\admin.aspx.cs | Admin\Web.config | **Web.config**

```
<?xml version="1.0" standalone="yes"?>
<configuration>
  <system.web>
    <compilation defaultLanguage="c#" debug="true" />
    <customErrors mode="RemoteOnly" />
    <authentication mode="Forms">
      <forms loginUrl="login.aspx" name=".ASPXCOOKIEAUTH" path="/">
        <credentials passwordFormat="SHA1">
          <user name="User1" password="4A60935FE851C99148EFD66122CDD0F43D5A3059" />
          <user name="User2" password="114B3C899D75F5A3FE3FDF83531C42E33214555B" />
          <user name="Admin" password="1F95EC61B6EF02B5D2B138654DA138BFDBBC7F3C" />
          <user name="a" password="86F7E437FAA5A7FCE15D1DDCB9EAEAEA377667B8" />
          <user name="x" password="E9D71F5EE7C92D6DC9E92FFDAD17B8BD49418F98" />
        </credentials>
```

شکل ۷- کاربران جدیدی که با استفاده از صفحه ی ادمین اضافه شده اند .

تمرین :

۱- مثال یک را طوری تغییر دهید که تنها قسمت های مربوط به کاربر جدید به فایل اضافه شوند و کل فایل هر بار OverWrite نشود.

۲- در مثال یک ، یک صفحه ی مخصوص برای کاربر ایجاد نمایید تا بتواند پسورد خود را عوض نماید.