7.2.0 – Securing Web Forms

Introduction

In previous lessons we setup security for our web site. When we did we modified the **`SecurityAdmin.aspx.cs`** file to have the following code:

```
protected void Page_Load(object sender, EventArgs e)
{
    if (Request.IsAuthenticated)
    {
        if (User.IsInRole("Administrators"))
        {
            MessageUserControl.ShowInfo("Success", "You may continue");
        }
        else
        {
            //redirect to a page that states no authorization for the requested action
            Response.Redirect("~/Default.aspx");

        }
    }
    else
    {
        //redirect to login page
        Response.Redirect("~/Account/Login.aspx");
    }
}//eom
```

The Page_Load event method runs when the web page is requested from the web server, thus any code in here will run each time we navigate to this page. In this code we see several things happening:

1. Is the user logged in? (`if (Request.IsAuthenticated)`)
2. If logged in, is the user in the **Administrators** role? (`if (User.IsInRole("Administrators"))`)
3. If logged in, and not an administrator, redirect to **Default.aspx**. (`Response.Redirect("~/Default.aspx")`)
4. If not logged in, redirect to **Login.aspx** (`Response.Redirect("~/Account/Login.aspx")`)

We can apply this same pattern to other pages on our web site.

PurchaseOrders.aspx

In our eStore_2018 database, we have the following `EmployeeTypes`:

	EmployeeTypeID	TypeName	TypeDescription
1	1	MA	Managerial level employee
2	2	SP	Sales staff involved with creating a Sale
3	3	PD	Staff involved with creating a Purchase Order
4	4	EM	General employee (includes janitorial and maintenance staff)

Figure 1: Employee Types

To create, or view Purchase Orders, an Employee needs to be a **MA** or **PD** employee type. Using the **`SecurityAdmin.aspx`** web form, we will add two new Roles, **Managers** and **Purchasers**. We will remove the **bwaters** account from the **Administrators** role, and add the **bwaters** account to the two new roles. We will need to log in as the **Webmaster** to accomplish these tasks. (Watch the video on how this was done.)

Now that the database is configured for us, we can modify the **`PurchaseOrders.aspx.cs`** code file to secure this web form by coding the Page_Load event method to be:

Listing 1: Page_Load for Purchase Orders

```
protected void Page_Load(object sender, EventArgs e)
{
    if (Request.IsAuthenticated)
    {
        if (User.IsInRole("Managers") || User.IsInRole("Purchasers"))
        {
            MessageUserControl.ShowInfo("Success", "You may continue");
        }
        else
        {
            //redirect to a page that states no authorization for the requested action
            Response.Redirect("~/Default.aspx");
        }
    }
    else
    {
        //redirect to login page
        Response.Redirect("~/Account/Login.aspx");
    }
}//eom
```

Now when we attempt to access this page, without logging in, we should get redirected to the login page. Once **bwaters** has logged in, the user will now have access to the **Purchase Orders** web form.

ProductMaintenance.aspx

This web form must only be accessible to the same Roles as the **`PurchaseOrders.aspx`** web form. Therefore, modify the Page_Load of the **`ProductMaintenance.aspx.cs`** with the same code as that from the **`PurchaseOrders.aspx.cs`**.

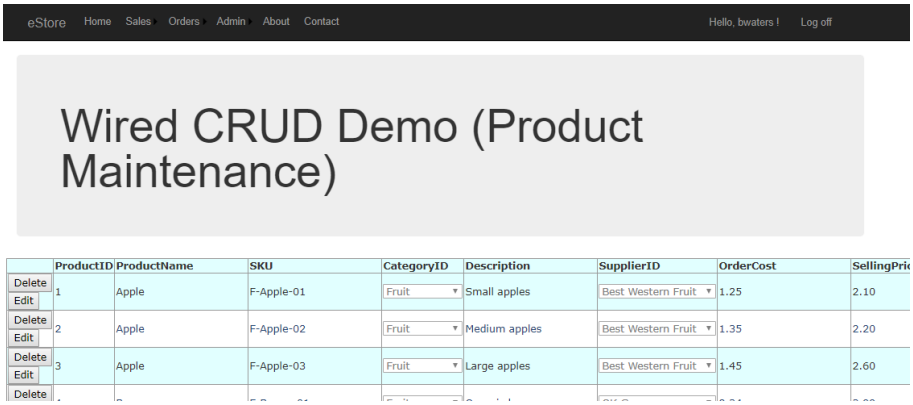

Figure 2: Bob Waters Accessing Product Maintenance

Exercise

TBD