**PRACTICAL NO:-7**

**a. Create a web application to demonstrate the use of different types of Cookies.**

Here there are two buttons which creates two cookies as follow.

1) create persistent cookie for certain interval.

2) create non-persistent cookie.

**First page**

**CODE:-**

protected void Button1\_Click(object sender, EventArgs e)

{

HttpCookie ck = new HttpCookie("cookie1");

ck["fav\_color"] = "blue";

ck.Expires = DateTime.Now.AddSeconds(10);

Response.Cookies.Add(ck);

}

protected void Button2\_Click(object sender, EventArgs e)

{

HttpCookie ck2 = new HttpCookie("cookie2");

ck2["fav\_color"] = "red";

Response.Cookies.Add(ck2);

}

protected void Button3\_Click(object sender, EventArgs e)

{

Response.Redirect("cookie validate.aspx");

}

**Second page:-**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

namespace db1

{

public partial class cookie\_validate : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

HttpCookie ck1 = Request.Cookies["cookie1"];

HttpCookie ck2 = Request.Cookies["cookie2"];

if (ck1!=null)

{

Label1.Text = "cookie one has value that is favourite color: "+ck1["fav\_color"] ;

}

else{

Label1.Text = "cookie 1 is expired";

}

if (ck2 != null)

{

Label2.Text = "cookie one has value that is favourite color: " + ck2["fav\_color"];

}

else

{

Label2.Text = "cookie 2 is expired";

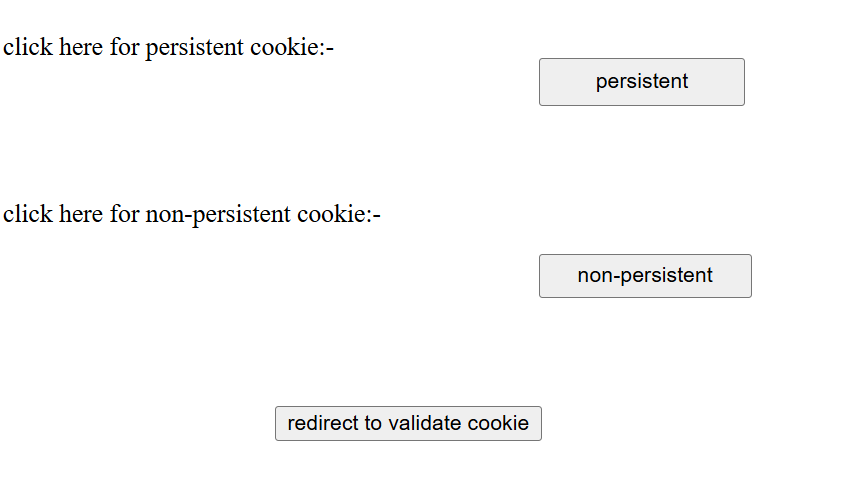
}

}

}

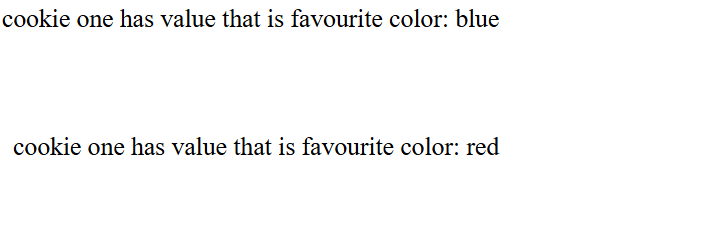
}

Page 1:

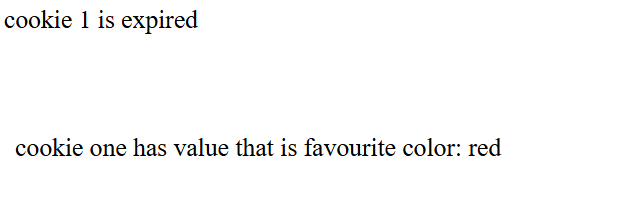
****

After creating both cookies.

**Page 2:-**

****

After certain interval here 10 sec:-



**b. Create a web application to demonstrate Form Security and Windows Security with proper Authentication and Authorization properties.**

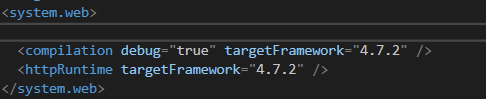
First we have to defined two pages both pages should be normal forms pages.

Here my two pages are: login\_page.aspx and home.aspx

Here we will first defining logic in **web.config** page;

Here you have to defined your all logic in the below tag:-

**<System.web >**tag



Now added below tags to define security logic.

<system.web>

**<authentication mode="Forms"> <!-- by default is also same..... -->**

**<forms defaultUrl="home\_page.aspx" loginUrl="login\_page.aspx" timeout="1" slidingExpiration="true">**

**<!--slidingExpiration="true" Controls whether the timeout resets on activity.-->**

**</forms>**

**</authentication>**

<compilation debug="true" targetFramework="4.7.2" />

<httpRuntime targetFramework="4.7.2" />

</system.web>

**Now for login\_page.aspx:-**

CODE:-

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Web.Security; // imp required here

namespace security

{

public partial class login\_page : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

}

protected void Button1\_Click(object sender, EventArgs e)

{

String username = TextBox1.Text, password = TextBox2.Text;

if (username == "Admin" && password == "password")

{

FormsAuthentication.RedirectFromLoginPage(username, true);

}

else {

Response.Write("login failed...");

}

}

}

}

On home page:-

**CODE:-**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Web.Security; ///IMP required

namespace security

{

public partial class home\_page : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

if (User.Identity.IsAuthenticated)

{

Response.Write("Welcome " + User.Identity.Name);

}

else {

Response.Redirect("login\_page.aspx");

}

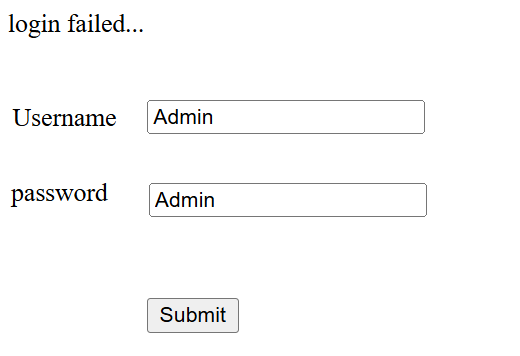
}

}

}

**OUTPUT:-**

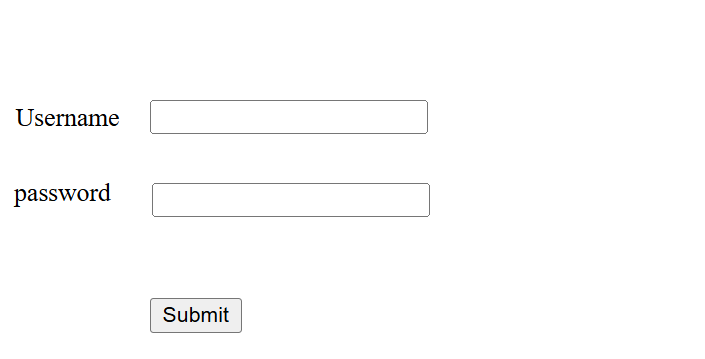
**On incorrect password**

****

**On correct password:-**

****

**After 1 minutes after not interacting with anything on page..**

****

If you do not react with server till one minute then the authorization will end you can communicate through server through button etc..