پیاده‌سازی sso بر اساس OAuth۲

در فایل web.config مقادیر زیر اضافه شود:

```
  <appSettings>
    <add key="ssoHost" value="https://sso.niocexp.ir"/>
    <add key="ssoClientId" value="pargarOAuthID"/>
    <add key="ssoClientSecret" value="pargarOAuthSecret"/>
    <add key="ssoRedirectUrl" value="https://pargar.niocexp.ir/login.aspx"/>
    <add key="ssoFirstUrl" value="https://sso.niocexp.ir/oauth2.0/authorize"/>
    <add key="ssoTokenUrl" value="https://sso.niocexp.ir/oauth2.0/accessToken"/>
    <add key="ssoUserProfileUrl" value="https://sso.niocexp.ir/oauth2.0/profile"/>
    <add key="ssoLogoutUrl" value="https://sso.niocexp.ir/logout"/>
  </appSettings>
```

```
<authentication mode="Forms">
      <forms loginUrl="~/login.aspx" timeout="30" defaultUrl="~/"
cookieless="UseCookies" slidingExpiration="true" enableCrossAppRedirects="true"
path="~/" />
</authentication>
<authorization>
   <deny users="?" />
</authorization>
```

در فایل Global.asax.cs متودهای SSOLogin و AuthenticateRequest اضافه شود:

```
using Newtonsoft.Json;
using System.Security.Principal;
using System.Web.Security;
```

```
protected void Application_AuthenticateRequest(object sender, EventArgs e)
        {
            if (HttpContext.Current.User==null)
            {
                if (HttpContext.Current.Request.QueryString["code"] != null)
                {
                    string userId =
SSOLogin(HttpContext.Current.Request.QueryString["code"]);

                    if (userId != null)
                    {
                                   _userId=userId;
                        //GetUser(userId);
                    }
                }
                else
                {
                    string firstSSOUrl =
ConfigurationSettings.AppSettings["ssoFirstUrl"].ToString().Trim();
                    firstSSOUrl += "?client_id=" +
ConfigurationSettings.AppSettings["ssoClientId"].ToString().Trim();
```

```
                    firstSSOUrl += "&redirect_uri=" +
ConfigurationSettings.AppSettings["ssoRedirectUrl"].ToString().Trim();
                    firstSSOUrl += "&response_type=code";

                    HttpContext.Current.Response.Redirect(firstSSOUrl, false);
                    return;
                }
            }
            else
            {
                //Page.Response.Redirect("~/frmMainRep.aspx");
            }
}



private string SSOLogin(string code)
{
if (HttpContext.Current.User!=null)
{
if (HttpContext.Current.User.Identity.IsAuthenticated)
{
return HttpContext.Current.User.Identity.Name;
}
}
string result = "";
string accessTokenUrl =
ConfigurationSettings.AppSettings["ssoTokenUrl"].ToString().Trim();
accessTokenUrl += "?code=" + code;
accessTokenUrl += "&client_id=" +
ConfigurationSettings.AppSettings["ssoClientId"].ToString().Trim();
accessTokenUrl += "&grant_type=authorization_code";
accessTokenUrl += "&redirect_uri=" +
ConfigurationSettings.AppSettings["ssoRedirectUrl"].ToString().Trim();

string credentials =
ConfigurationSettings.AppSettings["ssoClientId"].ToString().Trim() + ":" +
ConfigurationSettings.AppSettings["ssoClientSecret"].ToString().Trim();
string encodedCredentials =
System.Convert.ToBase64String(System.Text.Encoding.UTF8.GetBytes(credentials));


using (WebClient client = new WebClient())
{
client.Headers.Add("Authorization", "Basic " + encodedCredentials);
client.Headers.Add("Accept", "application/hal+json, application/json; q=0.5");
try
{
//HttpContext.Current.Response(accessTokenUrl);
byte[] response = client.UploadValues(accessTokenUrl, new
System.Collections.Specialized.NameValueCollection() { });
result = System.Text.Encoding.UTF8.GetString(response);
var payload = JsonConvert.DeserializeObject<Dictionary<String, Object>>(result);

/////////////////////////////////////////
// Get user profile
/////////////////////////////////////////
string userProfileUrl =
ConfigurationSettings.AppSettings["ssoUserProfileUrl"].ToString().Trim();
```

```
object token = null;
payload.TryGetValue("access_token", out token);
if (token == null)
{
throw new Exception("Error in authentication: Cannot get access_token from SSO
server.");
}
userProfileUrl += "?access_token=" + token.ToString();
string userProfile = client.DownloadString(userProfileUrl);
var payloadUserProfile = JsonConvert.DeserializeObject<Dictionary<String,
Object>>(userProfile);
object userIdObj = null;
payloadUserProfile.TryGetValue("id", out userIdObj);
string userId = "" + userIdObj.ToString();
//Session["CurrentUser"] = userId;
_userId = userId;
FormsAuthentication.SetAuthCookie(userId, true);
HttpContext.Current.User = new GenericPrincipal(new GenericIdentity(userId), new
string[] { });

return userId;

}
catch (WebException ex)
{
using (WebResponse resp = ex.Response)
{
if (((HttpWebResponse)resp).StatusCode != HttpStatusCode.OK)
{
throw new Exception("Error in authentication: " + ex.ToString()
+"\n accessTokenUrl:"+accessTokenUrl);
}

}
}

return null;

}

}
```

در هر جایی که Login انجام می شود، مقدار HttpContext.Current.User.Identity.Name
فراخوانی شود.