ASP.NET Identity Framework

John F. Gnazzo

Introduction

Securing a web site, is important to maintaining security and privacy for both users and hosted information. ASP.NET Identity provides a membership system for building and securing ASP.NET web applications. ASP.NET Identity allows the developer to add login features to an application and makes it easy to customize data about the logged in user, by extending the database schema.

ASP.NET Identity can be used with all of the ASP.NET frameworks, such as ASP.NET MVC, Web Forms, Web Pages, Web API, and SignalR.

ASP.NET Identity can also be used for building web, phone, store, or hybrid applications.

ASP.NET Identity also supports using Microsoft, Google, Facebook, and Twitter as user login's.

ASP.NET Database Schema

The following database schema is used to persist user logon and application role information. The schema can be freely extended to add additional functionality.

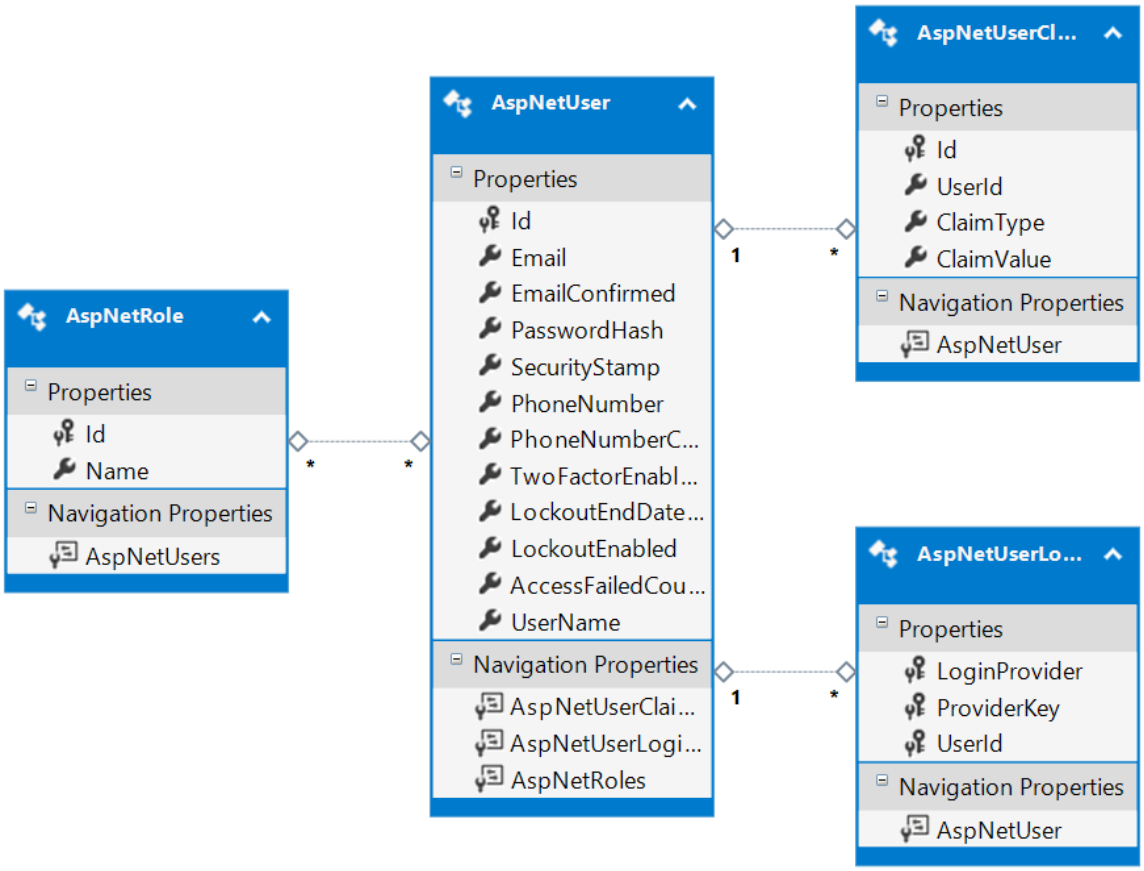

Figure 1 - ASP.NET Identity Database Schema

The **AspNetUser** table is used to persist information about the User including username and password as well as their demographics.

The **AspNetRole** table is used to store information regarding application roles. A user gets assigned to one or more roles through which the user gets access rights. Also, by assigning a user to a role, the user immediately gets all the access rights defined for that role.

The **AspNetUserClaim** table is used to store information regarding a user's or application claims. A claims-based identity is the set of claims. A claim is a statement that an entity (a user or another application) makes about itself. For example a claim list can have the user's name, user's e-mail, user's age, user's authorization for an action. Compared to Role-Based Security, a user presents the credentials directly to the application. In a claims-based model, the user presents the claims and not the credentials to the application. For a claim to have practical value, it must come from an entity the application trusts.

The **AspNetUserLogin** table is used to store login provider information. For instance Facebook, Google, Twitter and Microsoft identities can be used with ASP.NET Identity.

Using ASP.NET Identity in an MVC Controller

The following code snippet shows how a controller action method can be decorated to different levels of access.

Anonymous Access
The *AllowAnonymous* attribute gives all users access to the action method.

```
[AllowAnonymous]
public ActionResult Login(LoginViewModels model, string command)
{
    if (ModelState.IsValid)
    {
        // code eliminated for brevity
    }

    return View(model);
}
```

Authenticated Access
The *Authorize* attribute gives authenticated (login in) users access to the action method.

```
[Authorize]
public ActionResult GetPeople(string selectedRole = "All")
{
    return View((object)selectedRole);
}
```

Role Access
The *Authorize* attribute allows access to authenticated users having the *Admin* role access to the GetAjaxData method.

```
[Authorize(Roles = "Admin")]
public PartialViewResult GetAjaxData()
{
    var time = DateTime.Now;
    return PartialView(time);
}
```

Controller Class Access
Also all methods in a controller class can be set to a specific authentication profile by decorating the class accordingly. The *Authorize* attribute requires the user to be fully authenticated to use any of the methods in the AccountController

```
[Authorize]
public class AccountController : Controller
{
   // code eliminated for brevity
}
```

Conclusion

ASP.NET Identity provides a membership system to secure a web site, and is important to maintaining security and privacy for both users and hosted information.