УДК 004.4

А.С. Філенко, І.В. Пономарьов

СИСТЕМА АВТОРИЗАЦІЇ ТА АВТЕНТИФІКАЦІЇ ASP.NET IDENTITY

Анотація. При розробці веб-сайтів, особливо комерційних, часто виникає необхідність у впроваджені системи авторизації та автентифікації користувачів сайту, для надання їм різних рівнів доступу до інформації чи сервісів. Наразі технологія ASP.NET пропонує розробникам систему Identity, яка є потужним та зручним інструментом, що дозволяє реалізувати необхідні в сучасних умовах механізми роботи з користувачами.

Ключові слова: ASP.NET, Identity, MVC, авторизація, автентифікація, OWIN, користувач, cookies, Entity Framework.

Вступ. З розвитком web технологій змінювалися й способи та підходи до організації автентифікації та авторизації. Відповідно до цього розвивалася й система членства (membership) в ASP.NET. Вперше ASP.NET Membership була представлена в 2005. Ця система дозволяла проводити автентифікацію на основі форм даних та використовувала виключно SQL Server для зберігання необхідних даних (імена, паролі та ін..). Схема даних була фіксованою і розробники не мали можливості змінити її під потреби свого проекту. Крім того, Membership не підтримує OWIN (Open Web Interface for .NET), що унеможливлює використання даних для ідентифікації з інших сторонніх сервісів (Google, Twitter, Facebook). З появою ASP.NET Identity розробникам стали доступними можливості з налаштування системи автентифікації під свої потреби; зміна схеми даних; інтерфейс OWIN; підтримка різноманітних сховищ даних, що підтримують Entity Framework; ролі користувачів та ствердження (Claims) [1].

Постановка проблеми. ASP.NET Identity має широкий спектр інструментів для побудови та налаштування механізму автентифікації та авторизації користувачів сайту, слід розглянути

 ISSN 1562-9945

основні підходи до використання та окреслити можливості даної системи.

Основна частина.

Архітектура Identity. Архітектура системи складається з декількох шарів (рис. 1), які взаємодіють між собою для провадження процесів автентифікації.

Модуль ASP.NET Identity керує захистом паролів користувача, зберіганням його даних та керує ролями. Цей модуль «не знає» нічого про HTTP запити, що надходять до застосунку. Account Controller – це контролер, який пов'язує всі складові частини системи та забезпечує взаємодію між ними. Модулі OWIN відповідають за обробку автентичності за допомогою зовнішніх постачальників та встановлення сесії з застосунком через файли cookie. OWIN визначає стандартний інтерфейс між веб-серверами .NET та веб-додатками. Метою інтерфейсу є відокремлення сервера та програми. Фактично OWIN отримує HTTP запити та виконує роботу з формування cookie [2].

Рисунок 1 – Складові шари системи ASP.NET Identity

Використаня API-інтерфейсу Identity в проекті ASP.NET MVC.

Налаштування Identity є досить складним процесом, який повністю описується за посиланням [3]. Аби розглянути способи використання Identity можна створити проект ASP.NET MVC з готовою налаштованою системою Identity. Для цього при створенні нового проекту слід обрати шаблон MVC з індивідуальними обліковими записами (рис.2). Крім того, до проекту потрібно підключити необхідні для роботи пакети:

- Microsoft.AspNet.Identity.EntityFramework - пакет має реалізацію Entity Framework ASP.NET Identity, яка зберігатиме дані ідентифікації ASP.NET та схему БД;

• Microsoft.AspNet.Identity.Core - пакет має базові інтерфейси для ASP.NET Identity;

• Microsoft.AspNet.Identity.OWIN - пакет містить функціональність, яка використовується для підключення автентифікації OWIN в програмах ASP.NET.

Це робиться за допомогою менеджера пакетів NuGet.

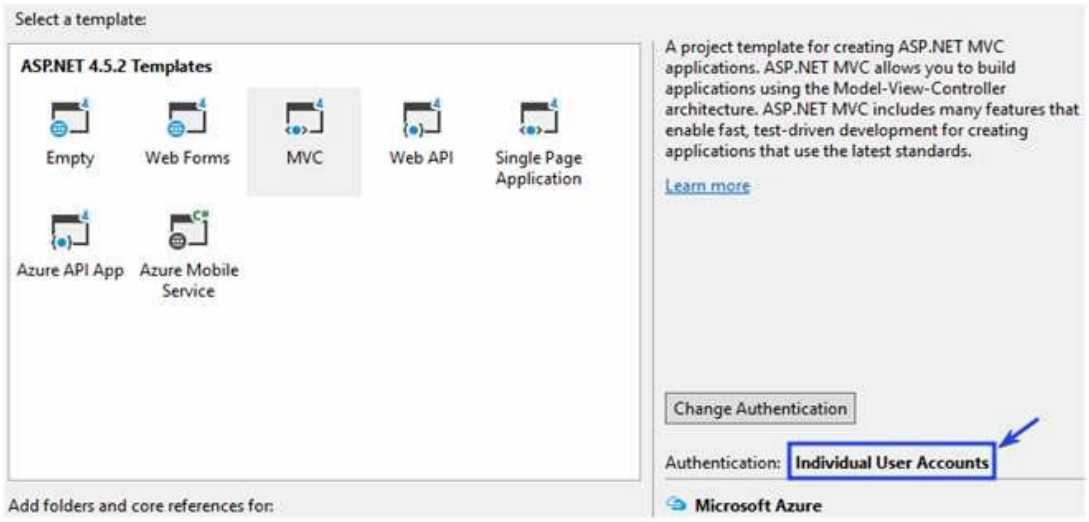

Рисунок 2 – Створення нового проекту за шаблоном MVC з Identity

Для доступу до класу менеджера користувачів AppUserManager в контролерах зручно використовувати властивість, яка повертає об'єкт менеджера з контексту OWIN, який в свою чергу береться з контексту запиту.

```
public ApplicationUserManager UserManager
    {
      get
      {
        return _userManager ??
HttpContext.GetOwinContext().GetUserManager<ApplicationUserManager>();
      }
      private set { _userManager = value; }
    }
```

Менеджер надає доступ до методів роботи з користувачами:

ChangePasswordAsync(id, old, new) – зміна паролю зазначеного користувача;

CreateAsync(user) – створення нового користувача без паролю;

CreateAsync(user, pass) – з паролем;

DeleteAsync(user) - видалення користувача;

FindAsync(user, pass) - пошук об'єкта користувача та перевірка правильності паролю;

FindByIdAsync(id) - пошук користувача за id;

FindByNameAsync(name) - пошук за ім'ям;

UpdateAsync(user) - оновлення об'єкта користувача (зміна даних);

Users - властивість, яка повертає список всіх користувачів;

Для обмеження доступу до дій неавторизованих користувачів, або визначення ролей, що мають доступ до дії, використовується атрибут:

```
[Authorize]
public ActionResult Index()
{…}
```

Аби визначити ролі, які мають доступ до даної дії використовується параметр атрибута:

```
[Authorize(Roles = "Users")]
public ActionResult Index()
{…}
```

Аналогічно до менеджера користувачів відбувається доступ до менеджера авторизації, за допомогою якого відбувається авторизація користувачів:

```
private IAuthenticationManager AuthManager
{
   get
   {
       return HttpContext.GetOwinContext().Authentication;
    }
}
```

AuthManager надає доступ до двох методів SignIn() – вхід, SignOut() – вихід.

Як приклад, розглянемо метод входу користувача в систему, в контролері AccountController, який відповідає за всі дії над користувачами.

```
[HttpPost]
[ValidateAntiForgeryToken]
public async Task<ActionResult> Login(UserLoginModel details, string returnURL)
{
   if(ModelState.IsValid)
```

```
    {
        AppUser user = await UserManager.FindAsync(details.Name,
details.Password);
        if (user == null)
            ModelState.AddModelError("", "Некоректне ім'я або пароль");
        else
        {
            ClaimsIdentity ident = await UserManager.CreateIdentityAsync(user,
                DefaultAuthenticationTypes.ApplicationCookie);
            AuthManager.SignOut();
    AuthManager.SignIn(new AuthenticationProperties {
IsPersistent = true}, ident);
            return Redirect(returnURL);
        }
    }
        ViewBag.returnURL = returnURL;
        return View(details); }
```

Метод оброблює POST запит, параметрами якого є інформація про користувача (пароль та ім'я) та адреса, з якої неавторизований користувач був перенаправлений на сторінку входу. За допомогою методу менеджера FindAsync відбувається пошук користувача за заданим ім'ям та паролем.

Якщо користувач не знайшовся, то на сторінку повертається повідомлення. Інакше, метод менеджера користувачів CreateIdentityAsync на основі об'єкта користувача формує об'єкт ClaimsIdentity – ідентифікуюча інформація. Потім, метод менеджера автентифікації SignIn виконує вхід. Цей метод приймає першим параметром властивості автентифікації AuthenticationProperties, в даному випадку IsPersistent = true означає, що cookie користувача після першого входу будуть збережені в браузері і система запам'ятає користувача, вхід відбуватиметься автоматично. Другий параметр – інформація ідентифікації, що була отримана раніше.

Висновки. ASP.NET Identity API є потужним та гнучким інструментом, який дозволяє реалізовувати різноманітні механізми авторизації користувачів. OWIN спрощує роботу з cookies, генеруючи їх автоматично. Робота з БД заснована на Entity Framework є ще однією перевагою Identity. Система автентифікації надає також можливість реалізувати двохфакторну автентифікацію з використан-

ням SMS повідомлень чи Email, або авторизувати користувачів за допомогою облікових записів Facebook, Google та ін..

ЛІТЕРАТУРА

1. J. Galloway. Introduction to ASP.NET Identity / J. Galloway, P. Rastogi, R. Anderson, T. Dykstra. – 2013 [Електронний ресурс]. – Режим доступу: https://docs.microsoft.com/en-us/aspnet/identity/overview/getting-started/introduction-to-aspnet-identity.

2. A. Abel. ASP.NET Identity and Owin Overview / A. Abel. – 2014 [Електронний ресурс]. – Режим доступу: https://coding.abel.nu/2014/06/asp-net-identity-and-owin-overview/.

3. ASP.NET Identity и системы аутентификации / [Електронний ресурс]. – Режим доступу: https://professorweb.ru/my/ASP_NET/identity/level1/.