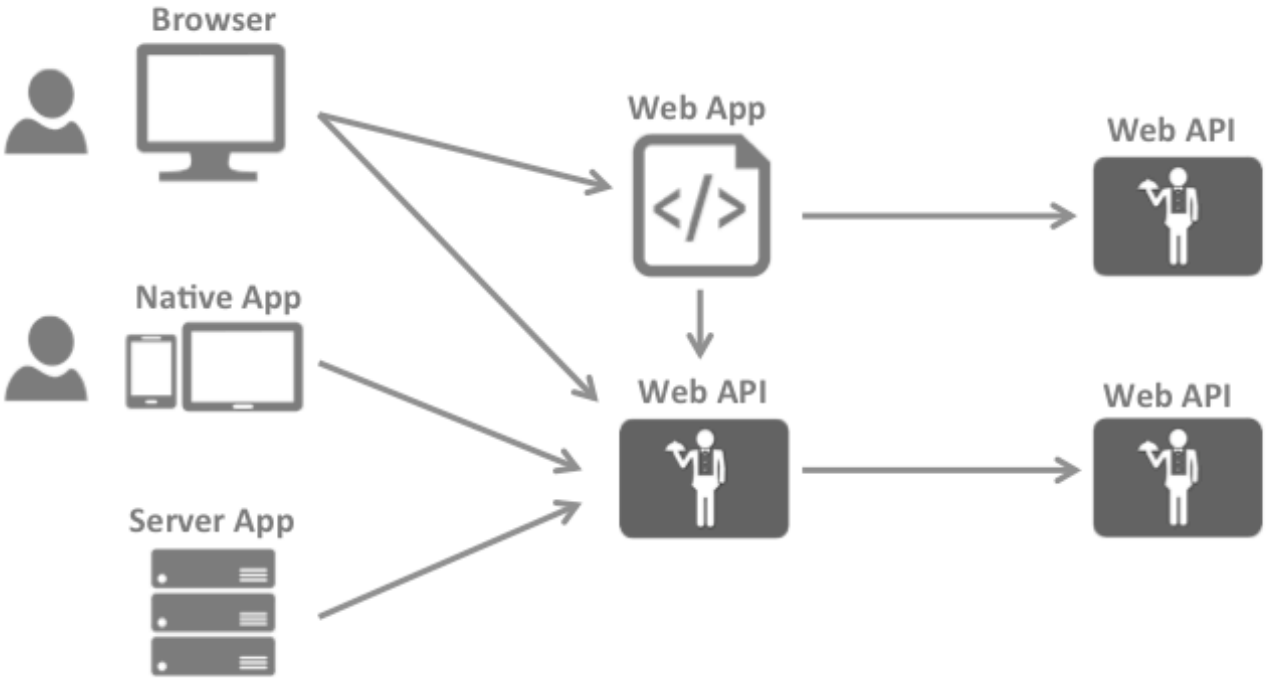
عنوان: **IdentityServer قسمت اول**
نویسنده: ناصرنیازی
تاریخ: ۱۳۹۴/۰۵/۲۸ ۱۸:۰
آدرس: www.dotnettips.info
گروه‌ها: Security, OpenID, ASP.NET Identity, IdentityServer, OAuth2

استفاده از سرویس‌های متنوع گوگل همگی با یک آکانت واحد، ایده‌ی جالبی است که پایه‌ی ایجاد پروژه‌ای به نام IdentityServer بوده است. IdentityServer یک پروژه‌ی متن باز است که قرار بود و شاید هنوز هم هست که بخشی از ویژوال استودیو باشد.

این پروژه یک سرور واحد برای مدیریت هویت ایجاد می‌کند که تمام کلاینت‌ها از این سرور اهراز هویت شده و سپس از سرویس‌ها استفاده می‌کنند. یعنی بخش مدیریت هویت تمام کاربران در پروژه برعهده‌ی IdentityServer گذاشته می‌شود و همه‌ی برنامه‌ها هویت کاربران را از IdentityServer می پرسند.
تصویری که توسعه دهندگان این پروژه برای معماری پروژه خود ارائه داده‌اند:



برای استفاده از آی‌دنتیتی‌سرور، ابتدا آن را از مخزن گیت‌هاب، بارگذاری می‌کنیم و سپس برای پروژه، یک Application جدید در IIS ایجاد می‌کنیم.
دقت داشته باشید که IdentityServer از SSL و بستر امن استفاده می‌کند که تنظیمات ساخت Certificate را می‌توانید از اینجا فرا بگیرید.
با توجه به پشتیبانی گسترده‌ی این پروژه از OpenID و OAuth2 مطالعه‌ای مختصر در این موارد به درک فرایند اهراز هویت توسط آی دنتیتی سرور بسیار کمک خواهد کرد.
پس از راه اندازی SSL و تنظیمات Certificate مربوط به آن می‌توانید شروع به راه اندازی سرور خود کنید. راه اندازی اولیه سرور تنظیمات مربوط به بانک اطلاعاتی، نکات ریزی دارد که بخش کلی از آن اینجا آمده است.
برای شروع به استفاده از سرور و درک چگونگی عملکرد آن نیاز دارید تا مدیر سرور را نصب کنید؛ تا هم بتوانید کاربر تعریف کنید و هم نقش‌ها (Roles) و دسترسی‌ها را مدیریت کنید. نگارش مدیر آی‌دنتیتی‌سرور نیز از اینجا قابل دسترسی می‌باشد.
پس از نصب آی‌دنتیتی‌سرور باید تنظیمات مربوط به ذخیره سازی اطلاعات آن را انجام دهید که آی‌دنتیتی‌سرور پیاده سازی خوبی برای Entity Framework دارد که می‌توانید با نصب آن همه‌ی کارهای ذخیره سازی در بانک اطلاعاتی را به EF بسپارید. البته برای

ذخیره‌ی یوزر می‌توان از حالت InMemory هم استفاده کرد که در نسخه‌ی مثال یک کاربر با نام bob و رمز bob در داخل کد نویسی تعریف شده بود، که می‌توان برای پروژه‌های کوچک همان را توسعه داد.

در سطح بانک هم آی‌دنتیتی‌سرور از دو بانک اطلاعاتی، استفاده می‌کند؛ یکی برای ذخیره‌ی تنظیمات سرور و دیگری برای ذخیره‌ی اطلاعات هویتی.

مزیت بزرگ آی‌دنتیتی‌سرور در اعتبار سنجی جدای از پیاده سازی‌های فراوان انجام شده برای محیط‌های مختلف و در قابلیت اعتبار سنجی دو طرفه‌ی آن می‌باشد.

یعنی هم سمت سرور و هم سمت کلاینت و هم سرویس‌هایی که از آی‌دنتیتی‌سرور استفاده می‌کنند، باید اعتبار سنجی شوند و همه چیز به یک رمز و نام کاربری ساده خلاصه نمی‌شود.

در زمان نگارش این متن، نسخه‌ی 2 نسخه‌ی پایدار ارائه شده است و نسخه‌ی سه در مرحله تست می‌باشد. البته پیاده سازی‌هایی هم از نسخه‌ی 3 در محیط‌های مختلف مثل MVC و WEB API ارائه شده است؛ ولی هنوز به پایداری لازم نرسیده است.

نظرات خوانندگان

نویسنده: مسعود دانش پور
تاریخ: ۱۳۹۴/۰۵/۳۰ ۲۲:۳۳

من سه سالی هست دارم سعی می‌کنم این پروژه رو نهایی کنم.ولی هنوز به دلیل مشغله زیاد موفق نشدم.

عنوان: **IdentityServer قسمت دوم**
نویسنده: ناصرنیازی
تاریخ: ۱۳۹۴/۰۶/۰۳ ۱:۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, Security, ASP.NET Identity, IdentityServer

پس از تلاش‌های اولیه برای راه اندازی که نیاز به گوگل کردن موارد مختلف دارد از جمله راه اندازی ssl و certification در لوکال هاست و تنظیم IIS برای استفاده از آن، می‌توان به راه اندازی اولیه آی‌دن‌تی‌تی سرور رسید .
پیش فرض این آموزش این نسخه از آی دنتیتی سرور است :
https://github.com/IdentityServer/IdentityServer2
نگاهی اجمالی به سورس: IdentityServer2

```
Sampel----

AdfsIntegrationFullSample --------

AdfsIntegrationSampleClient --------

InMemoryHost --------

(MVC and WCF RP (SAML --------

(MVC and Web API (JWT --------

MembershipRebootUserRepository --------

OIDC --------

SelfHostConsoleHost --------

ServiceBus Integration --------

src----

Libraries --------

WebSite --------

Tests --------
```

از نام مثال‌ها کاملاً مشخص است که هر کدام چه بخشی را پوشش می‌دهند.
مثلاً پوشه‌ی MembershipRebootUserRepository، مربوط به مدیریت کاربران است و توابع غنی بسیار خوبی در آن استفاده شده‌است. خود MembershipReboot یک پروژه‌ی دیگر است برای مدیریت کاربران که توسط BrockAllen توسعه داده شده و مدیریت کاربران را بسیار آسان کرده‌است؛ برای مثال به راحتی می‌توان

```
BrockAllen.MembershipReboot
BrockAllen.MembershipReboot.Ef
BrockAllen.MembershipReboot  .Repository
```

از این آدرس را به سورس اصلی آی دنتیتی سرور اضافه کرد و از توابع مدیریت کاربران این پروژه استفاده کرد. چون کار را بسیار آسان کرده است.
برای مثال من برای ایجاد کاربر نیاز داشتم که علاوه بر درج کاربر در سرور آی دنتیتی، در بانک اطلاعاتی خودم هم کاربر درج شود. برای همین برای آی دنتیتی سرور صفحه‌ی ثبت نام نوشتم. برای اینکار فقط با یک شیء از membership به راحتی این تابع پیاده سازی شد:

```
Membership.CreateUser(userName, password, email);
          Roles.AddUserToRoles(userName, "IdentityServerUsers");
```

نمونه استفاده

این افزودن کاربر را می‌توان از هسته‌ی اصلی خود آی دنتیتی سرور استفاده کرد. ولی استفاده از آن به علت راحت بودن و توابع خیلی زیاد BrockAllen.MembershipReboot است و استفاده از آن بسیار کاربردی می‌تواند باشد.
برای استفاده از خود آی دنتیتی سرور برای ساخت یوزر، وارد قسمت سورس شوید و به کنترلر احراز هویت، یک تابع به نام رجیستر اضافه کنید؛ به این فایل:

https://github.com/IdentityServer/IdentityServer2/blob/master/src/OnPremise/WebSite/Controller/AccountController.cs

شکل تابع :

```
[HttpPost]
public ActionResult Register(RegisterModel model)
 {
         UserRepository.CreateUser(model.userName,model.password,model.email);
         Roles.AddUserToRoles(userName, "IdentityServerUsers");
         return  View(model);
}
```

خود کلاس UserRepository یک کلاس پیاده سازی شده از اینترفیس IUserManagementRepository است که می‌توان خود آن‌را نیز با تزریق وابستگی‌ها تغییر داد و از Membership و بازنویسی توابع قدرتمند MembershipReboot آن استفاده کرد. لیست توابع این اینترفیس که می‌توانید استفاده کنید:

```
void CreateUser(string userName, string password, string email = null);
        void DeleteUser(string userName);

        IEnumerable<string> GetUsers(int start, int count, out int totalCount);
        IEnumerable<string> GetUsers(string filter, int start, int count, out int totalCount);

        void SetPassword(string userName, string password);

        void SetRolesForUser(string userName, IEnumerable<string> roles);
        IEnumerable<string> GetRolesForUser(string userName);

        IEnumerable<string> GetRoles();
        void CreateRole(string roleName);
        void DeleteRole(string roleName);
```

آدرس اینترفیس