# Objective:

The purpose of this document is to demonstrate how to **use Azure Active Directory for authentication from ASP.NET web sites.**

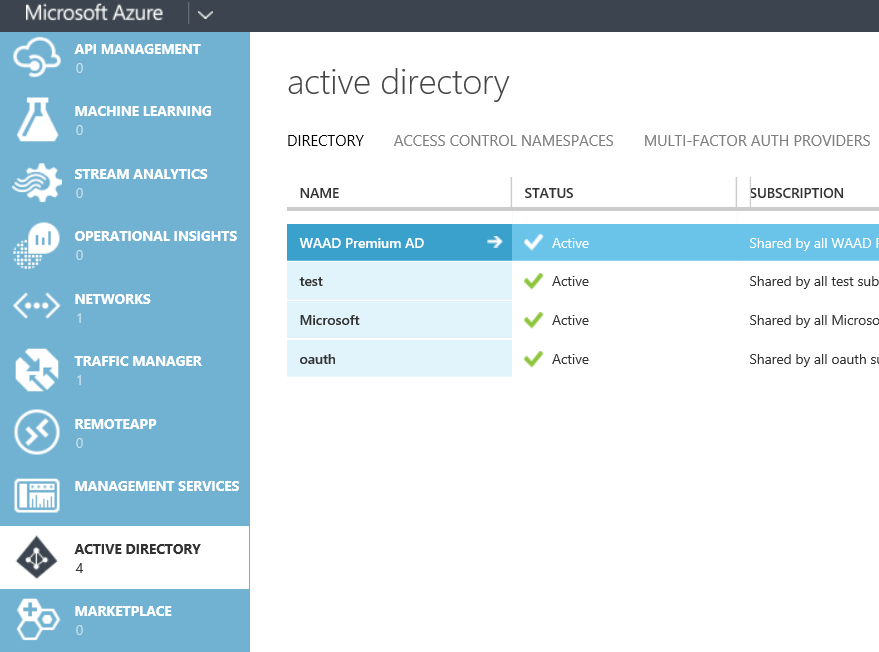
# Prerequisites:

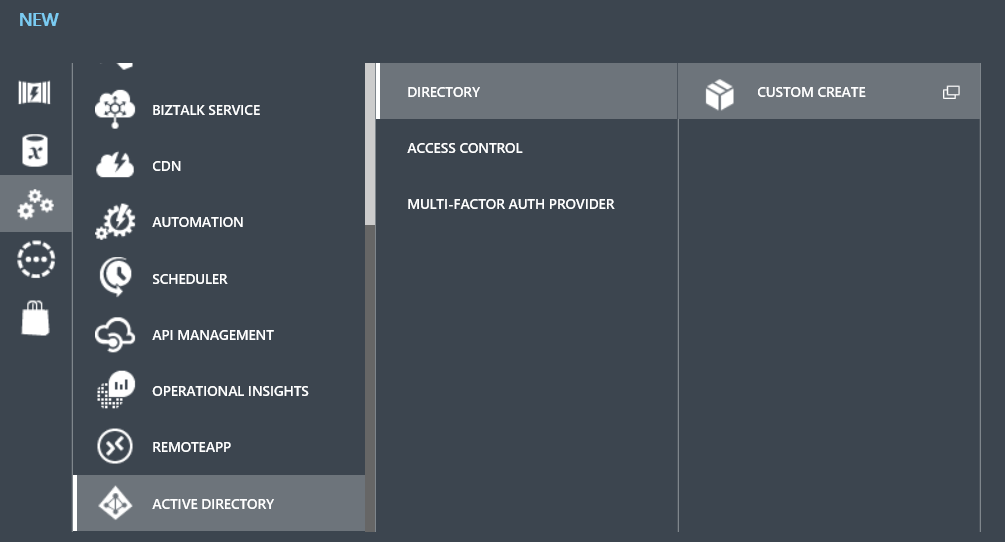
* Visual Studio 2015 with IIS
* Valid Azure Subscription

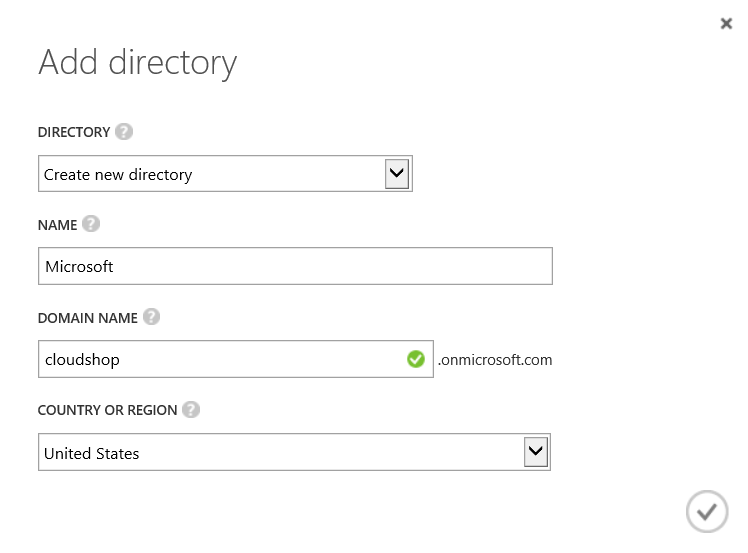
# Steps Overview:

* Setting up Azure Active Directory
* Create an ASP.NET app that uses Azure AD for authentication from VS
* Configure Azure Active Directory for the demo CloudShop App

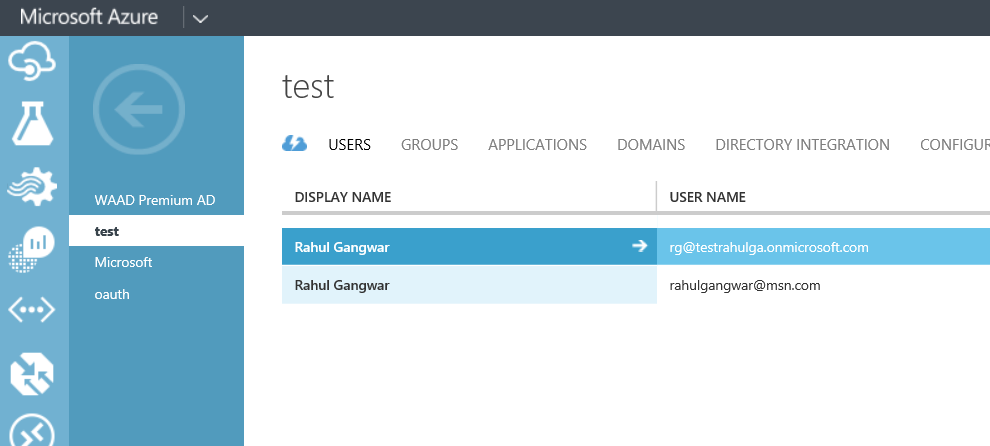
# Setting up Azure Active Directory

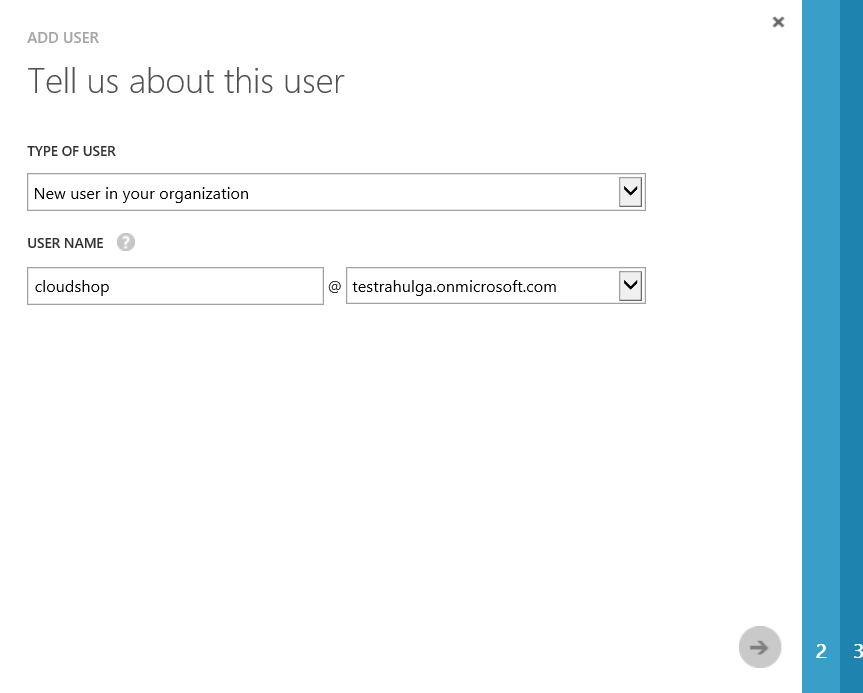
* From browser, navigate to manage.windowsazure.com and click “Active Directory” as shown below. You must see some default directories created. 
* If you did not see any directories, please create one using below steps.

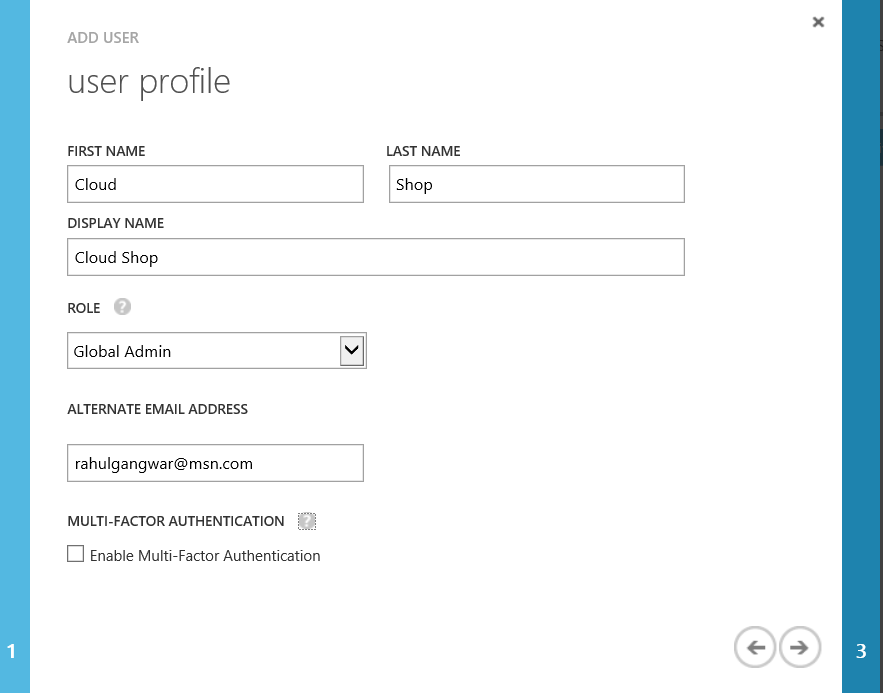


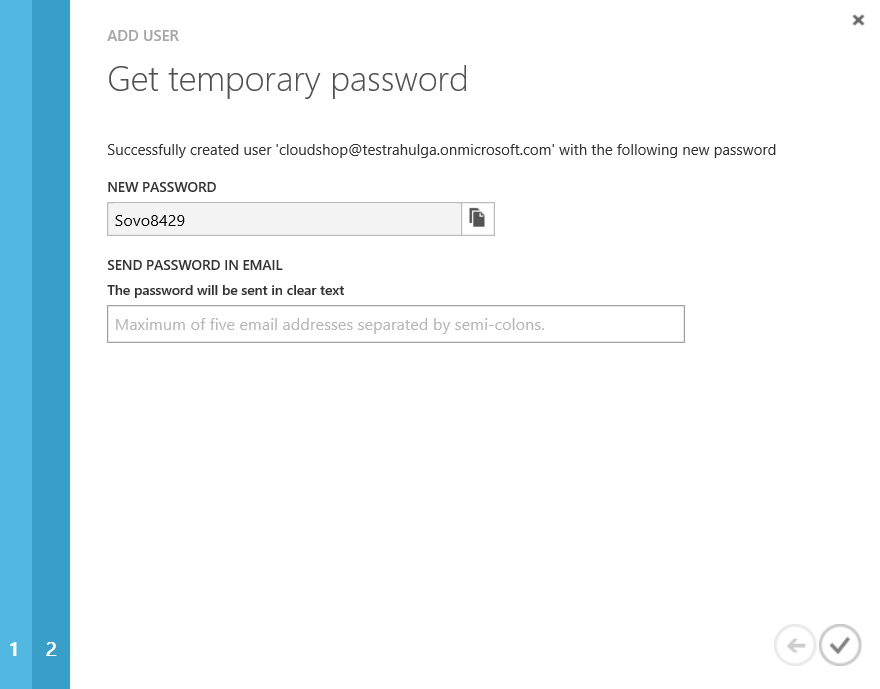


* Navigate to any one of the directories where you wish to provision users. If you do not see any users, create a new user using “Ad User” link at bottom. Make sure this user is a Global Admin as shown in below sequence of steps.



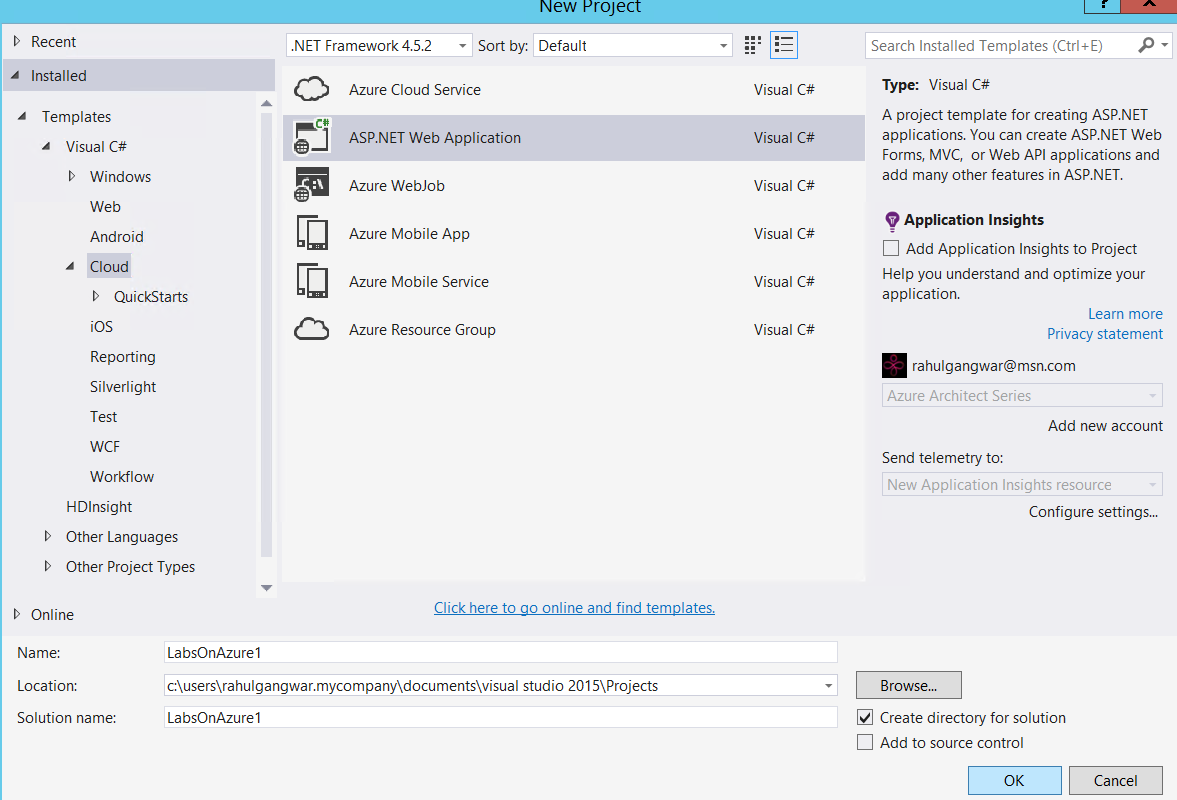


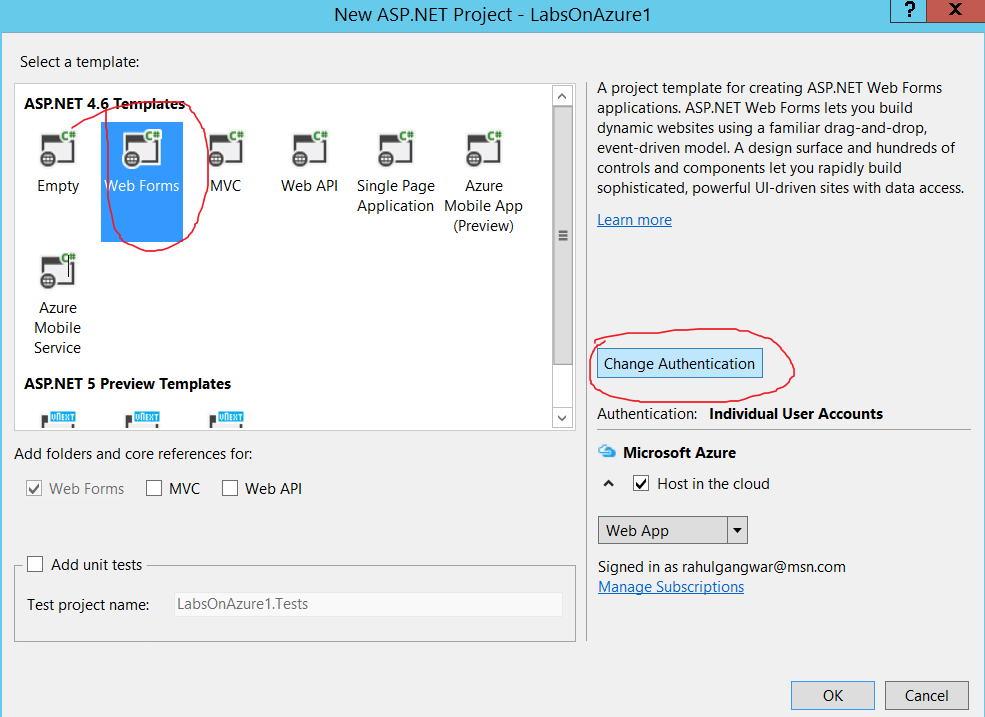




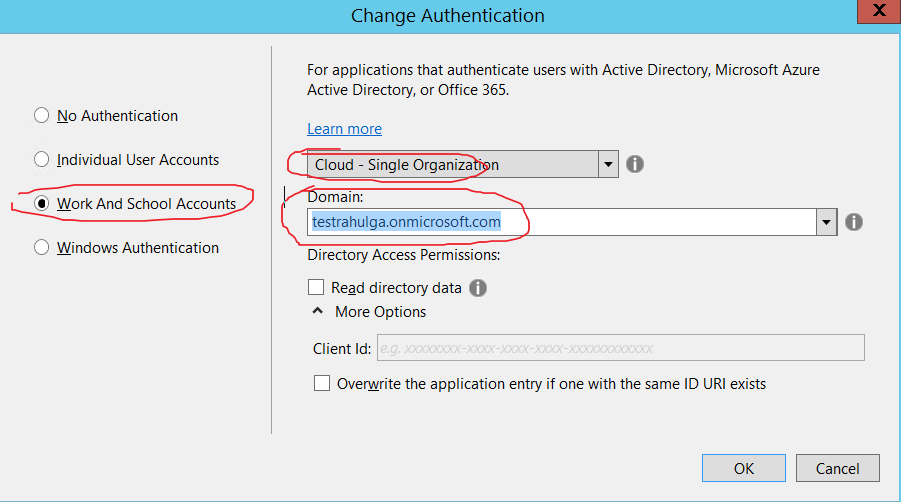
# Create an ASP.NET app that uses Azure AD for authentication from VS

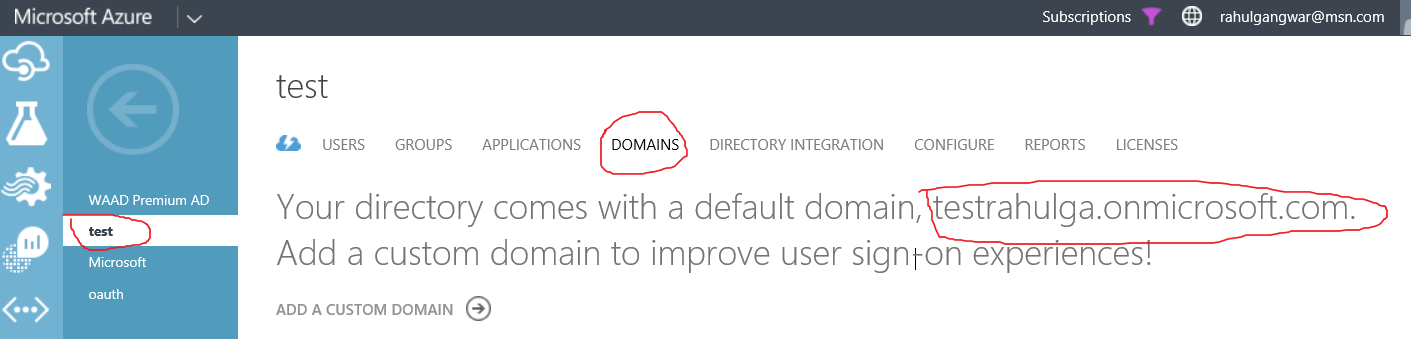
* Open VS 2015 in administrative mode.
* Create a new ASP.NET web application as shown in below screens.
* File🡪 New Project🡪 ASP.NET Web Application



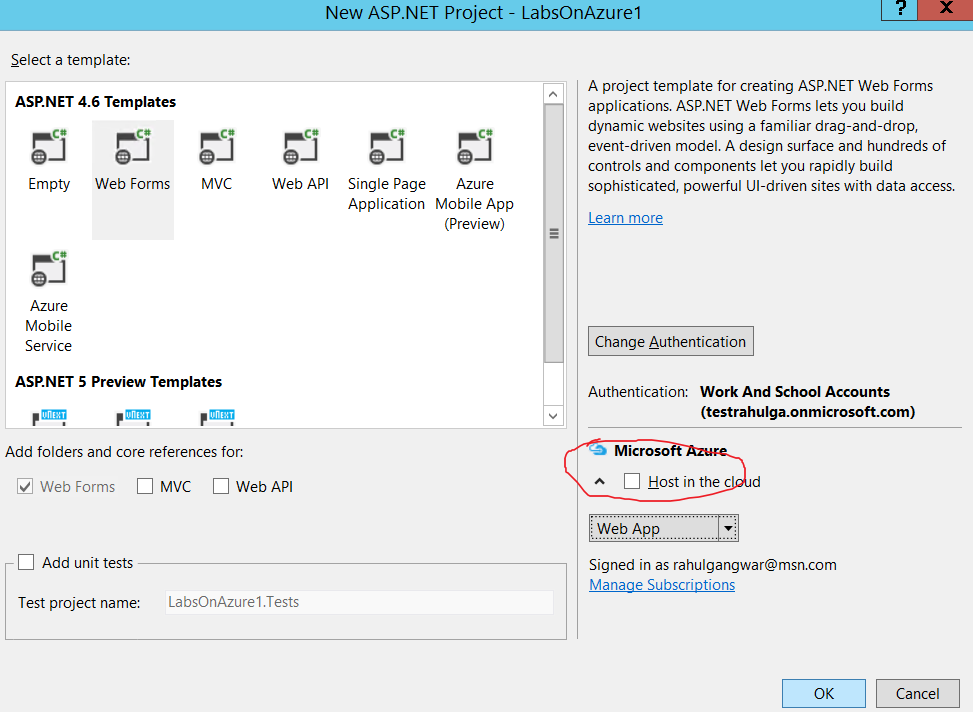


* At this point you will be shown option to select authentication mechanisms. Select “work And School Accounts” and “Cloud Single Organization” as shown below.
* You’d also notice that you are presented with option to select an Azure AD tenant. Make sure to select the tenant where you have provisioned user. In this case it is “testrahulga.onmicrosoft.com”.
* On Azure Portal you can verify the name of your domain by navigating to domain tab and looking at associated domain. Both screen shots of VS as well as Azure Portal having domain name are shown below.

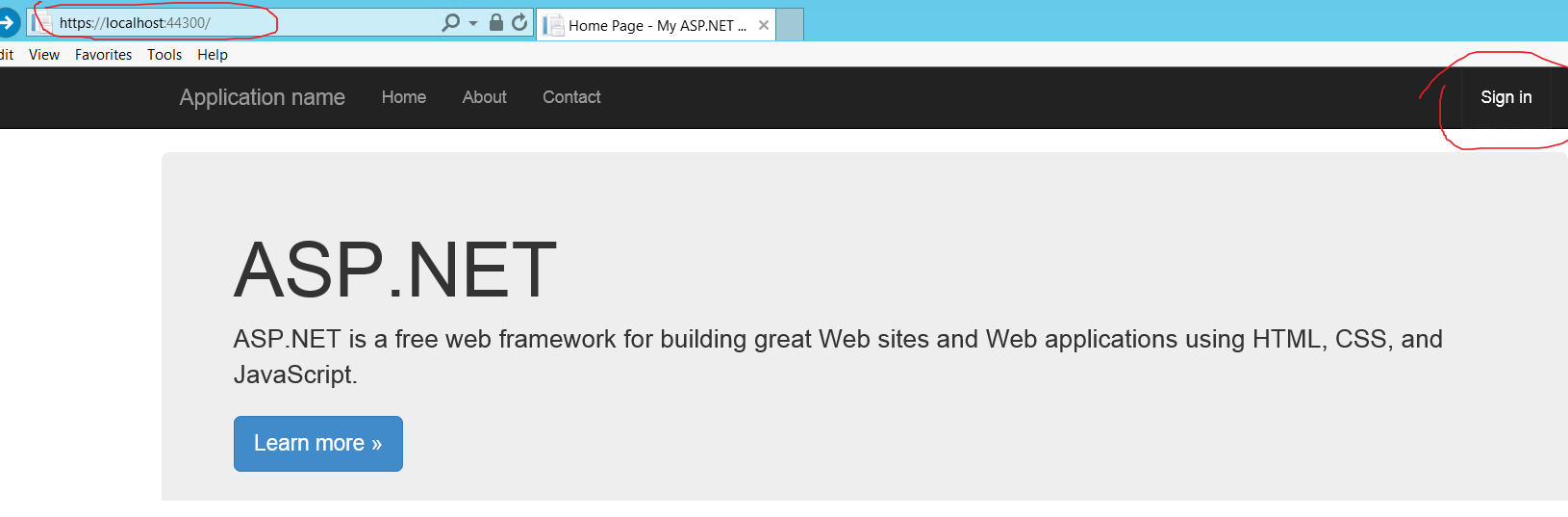


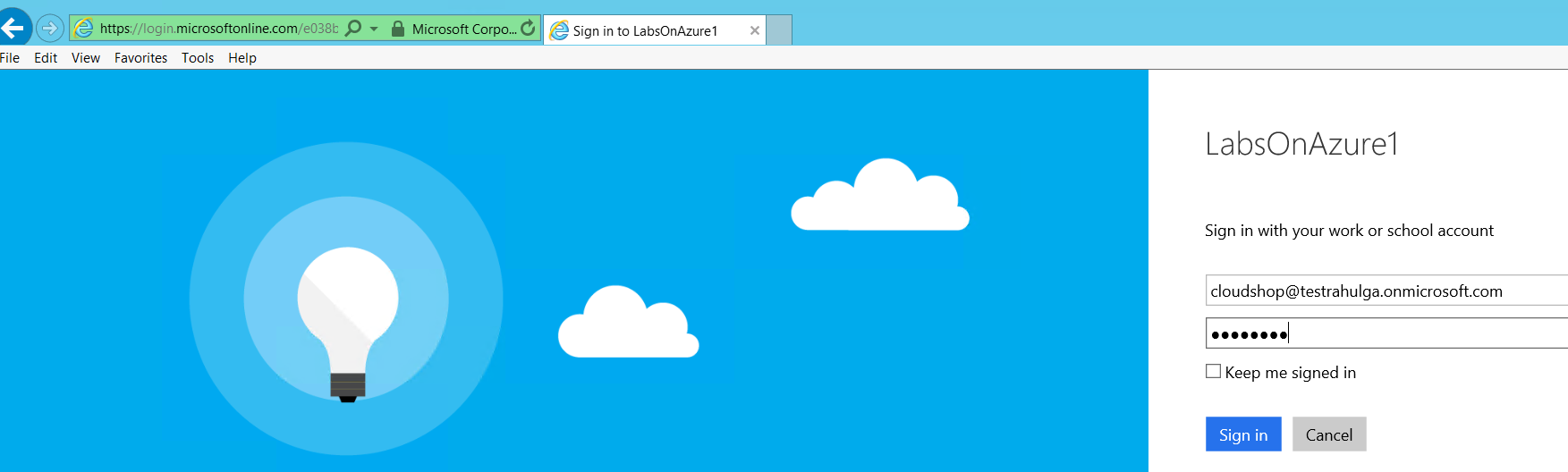


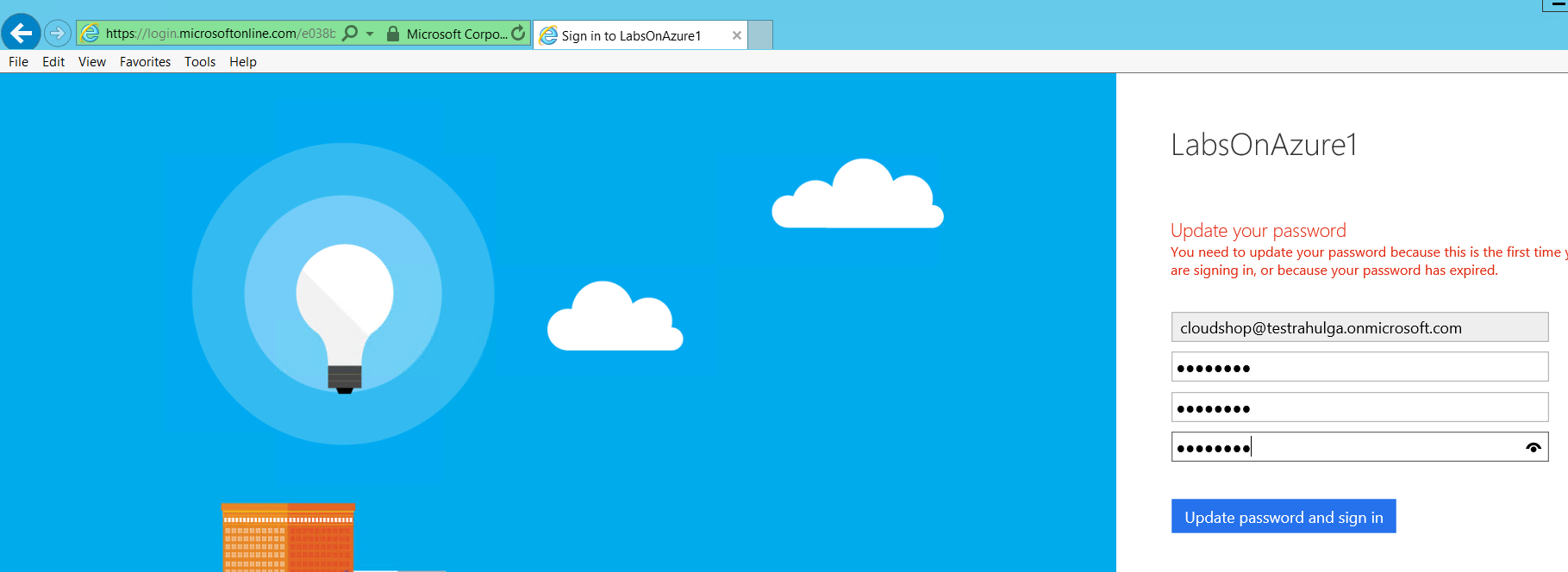
* Make sure to deselect “Host on Azure” option as shown below.

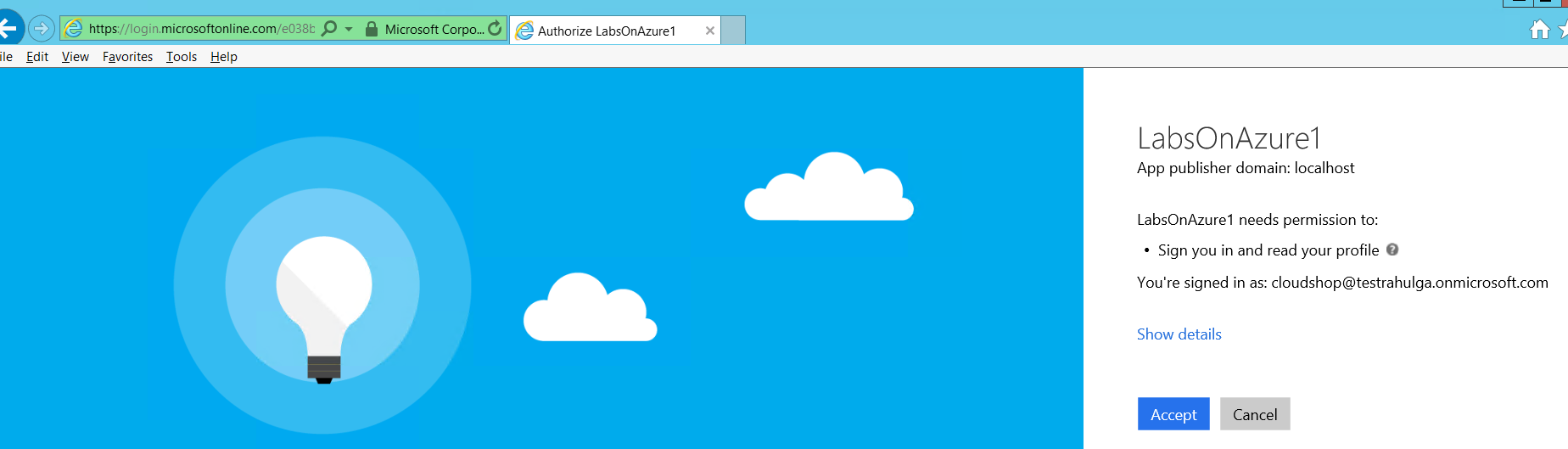


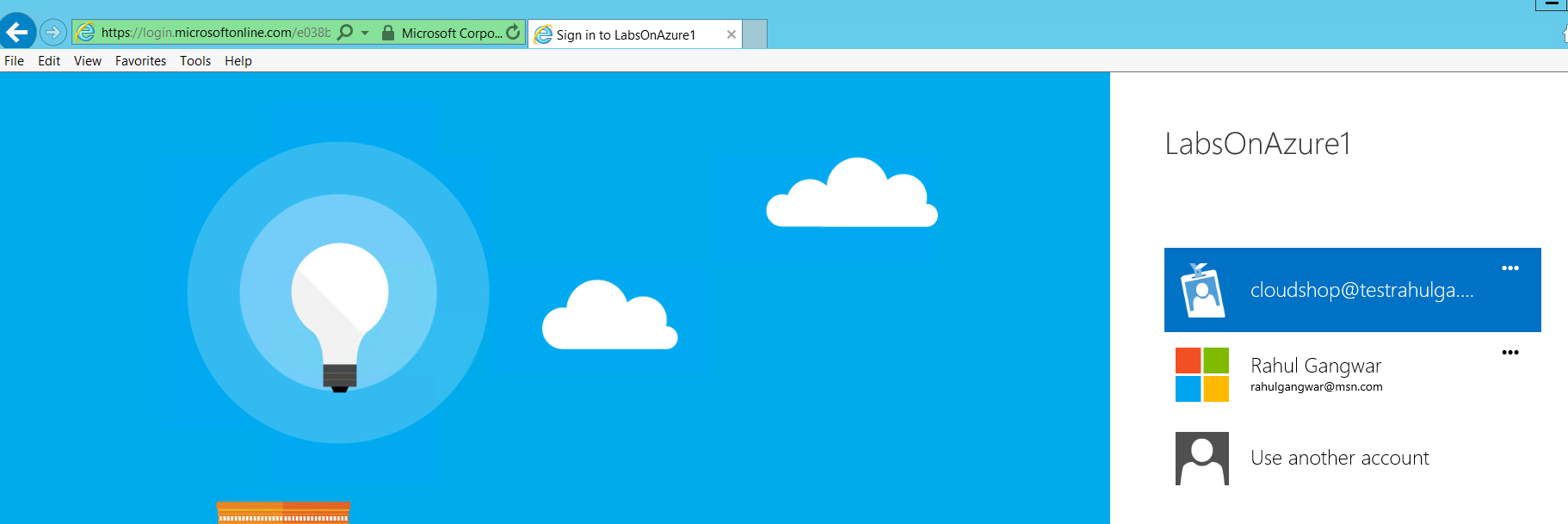
* Congratulations! You have successfully create an ASP.NET website that uses Azure AD as authentication provider. At this point, you can launch your application from VS and see that it takes to Azure AD for authentication as shown by below screen shots.

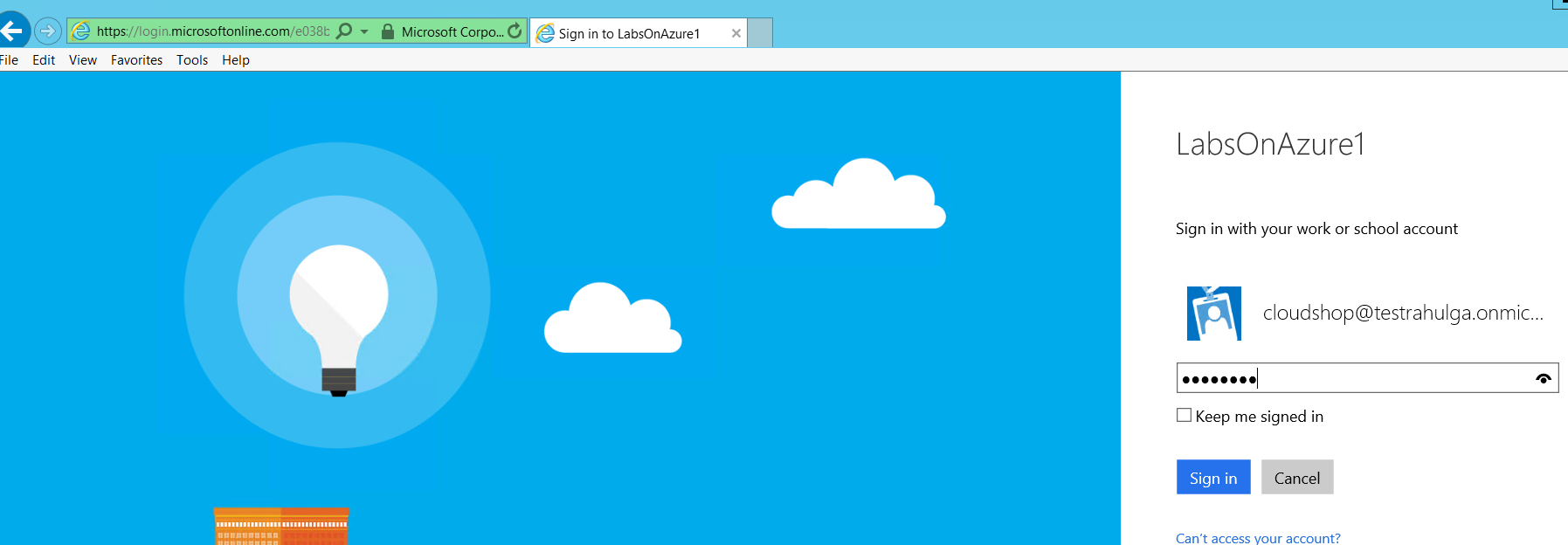


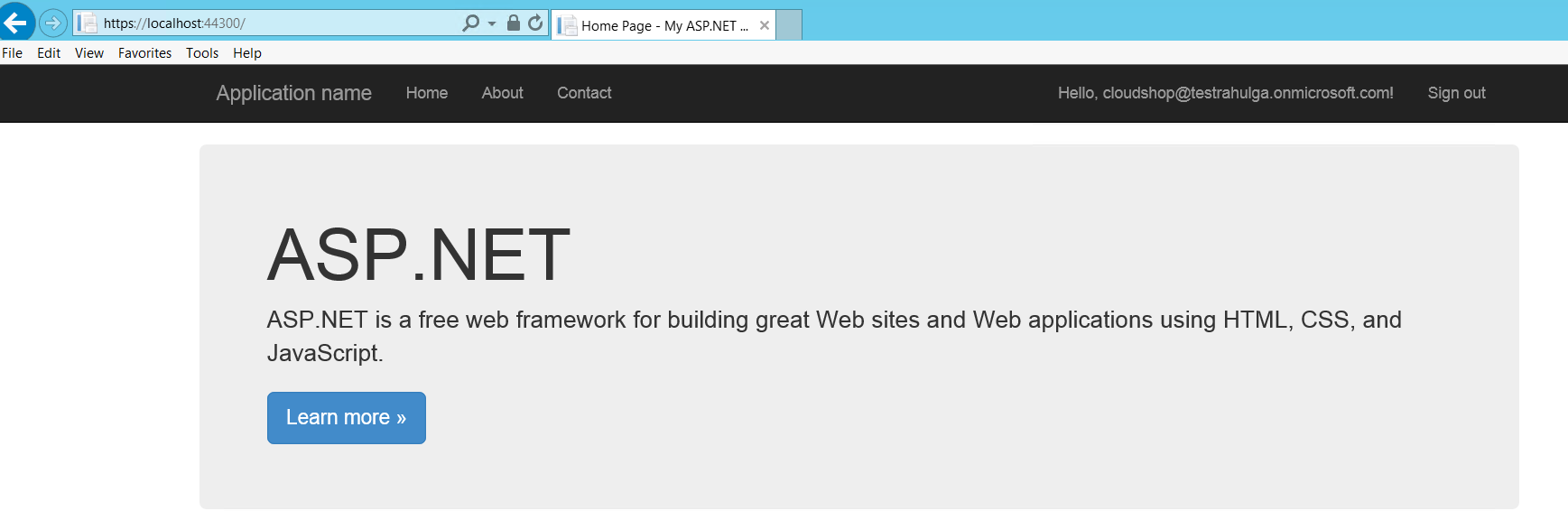






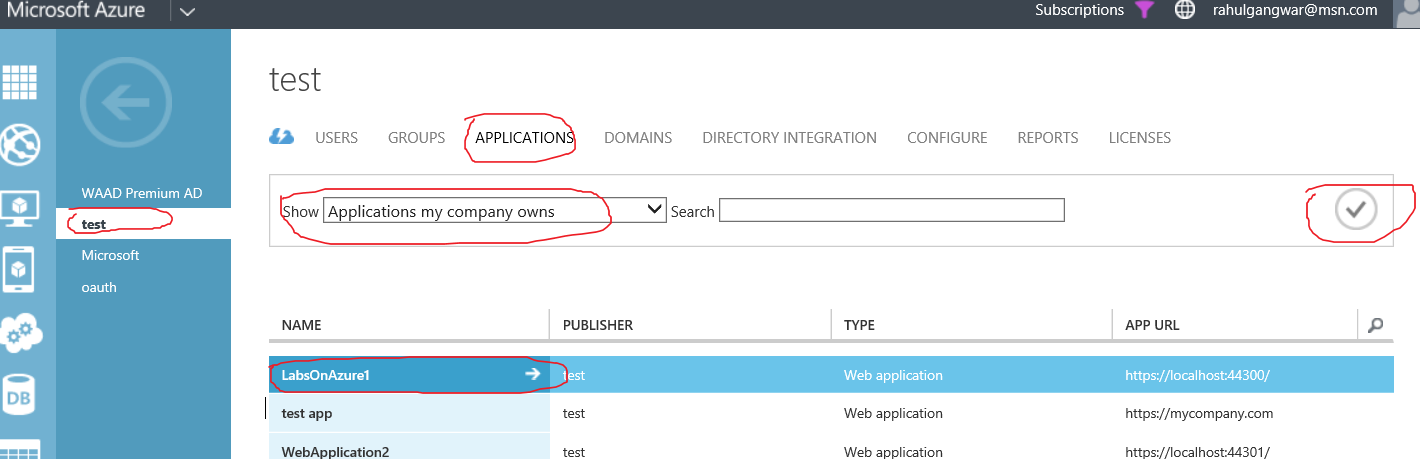




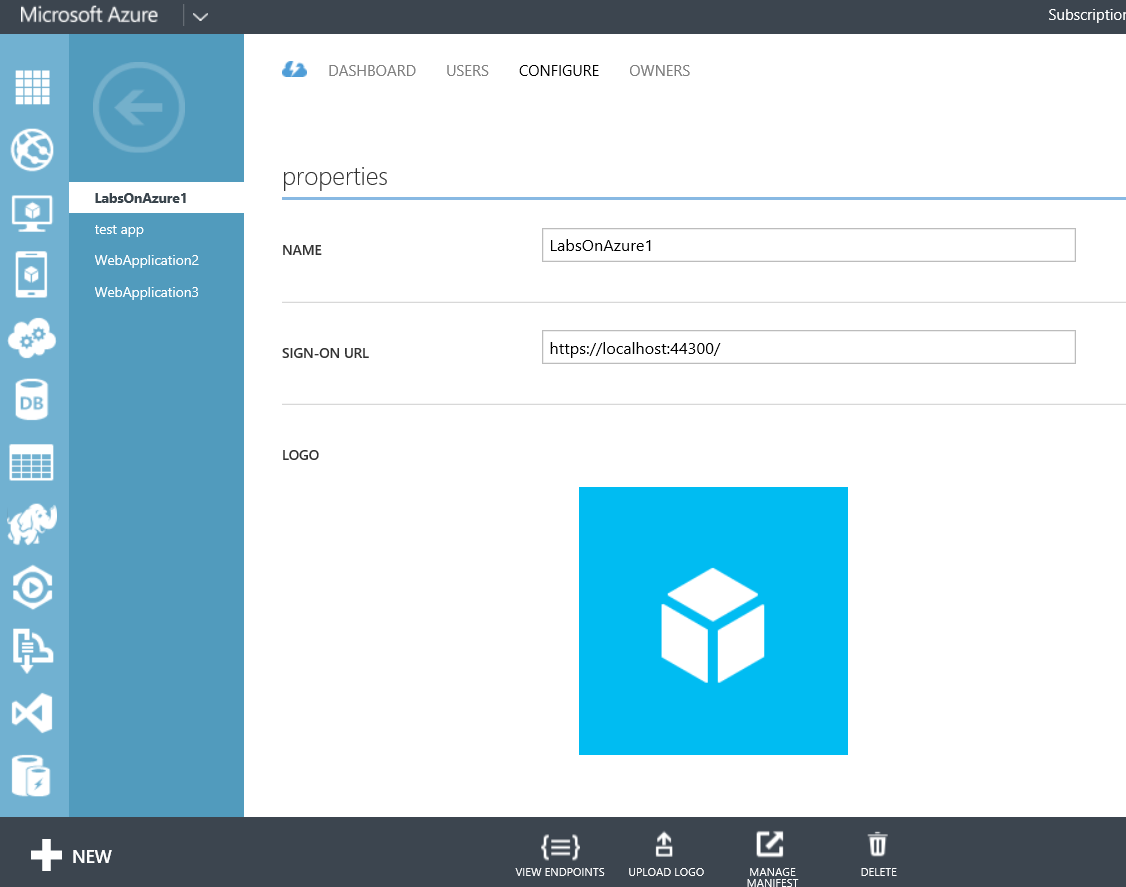


# Note to Presenter:

* Please walk through the attendees to show what happened in backend when we configured the application to use Azure AD as authentication provider. Basically navigate to your AAD tenant and open Applications tab and then select Show Applications my company owns as shown below:

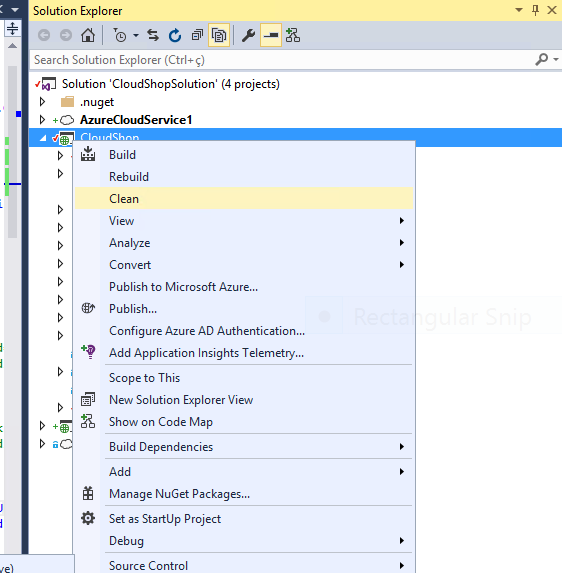


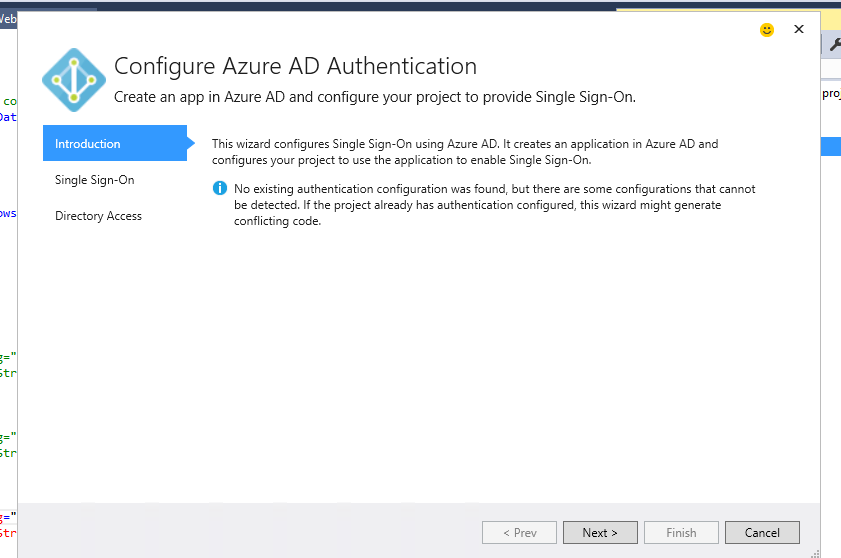
* Also select the registered application and walk through the registration details (like sign in URL, reply url etc.) as shown below.



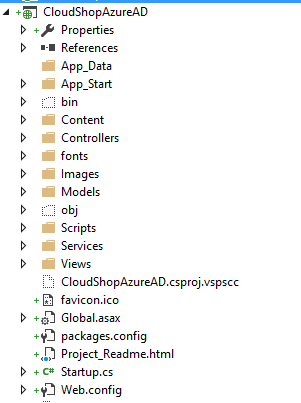
# Configuring Azure Active Directory in an existing Application

* Note that Visual Studio 2015 RC has added a feature for enabling Azure AAD in an existing Application.

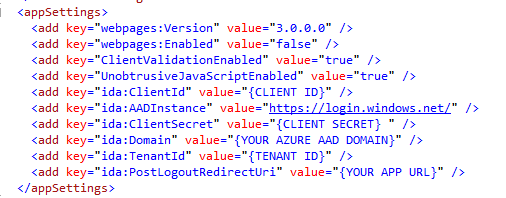




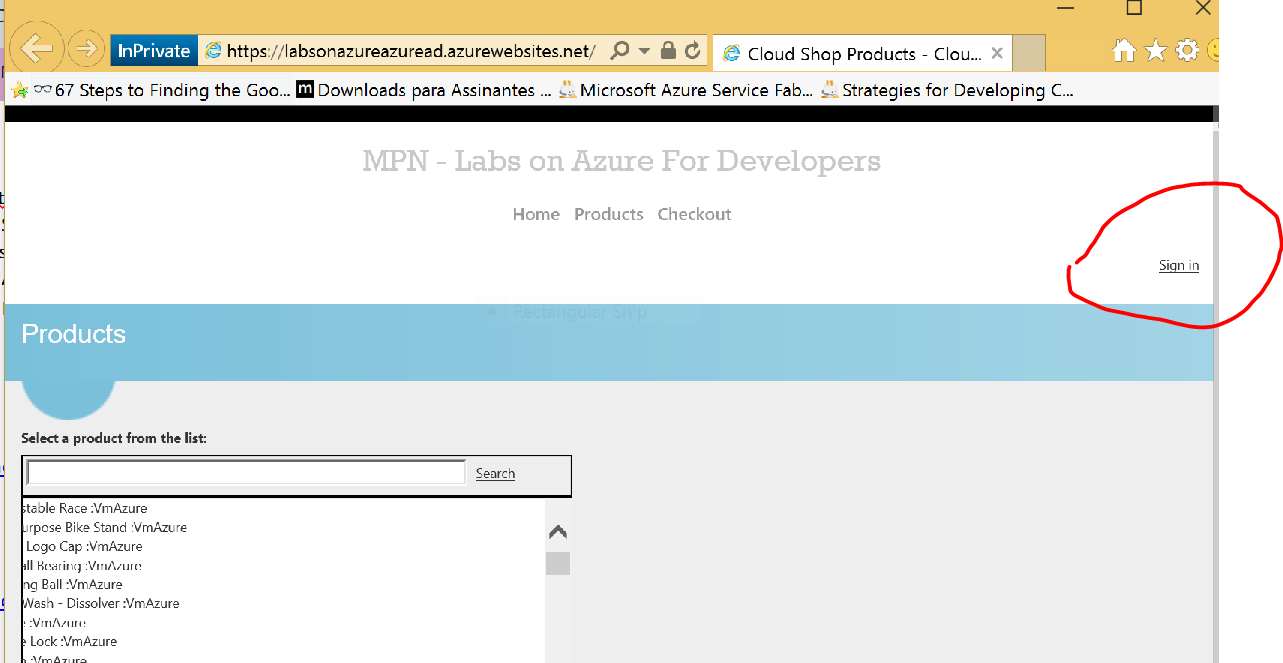
* For the CloudShop sample app, we were not able to use this option due to some dlls references. However, we have included a version of the App that already includes the integration with Azure AD.



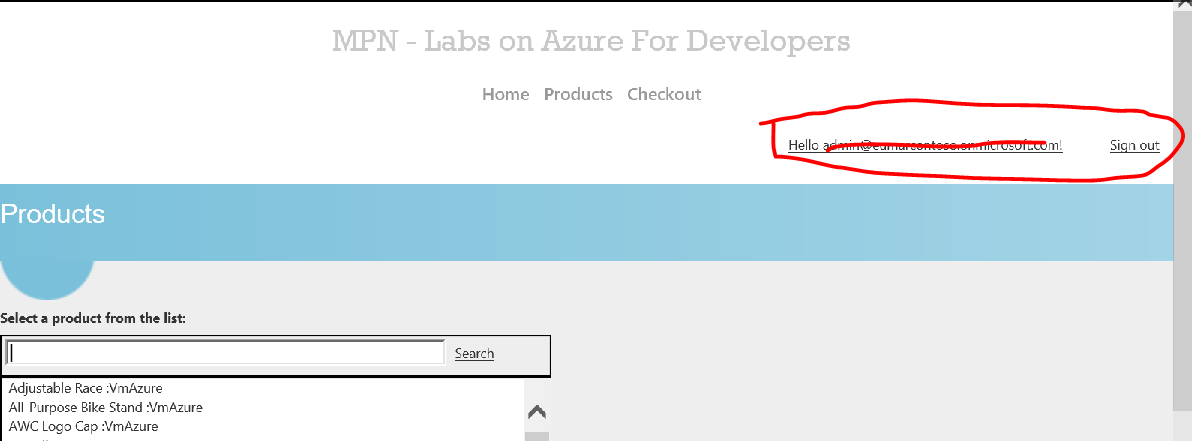
* You will need to change the following variables in the CloudShopAzureAD App’s Web.Config, you can copy and paste it from the App created earlier in this demo Web.Config



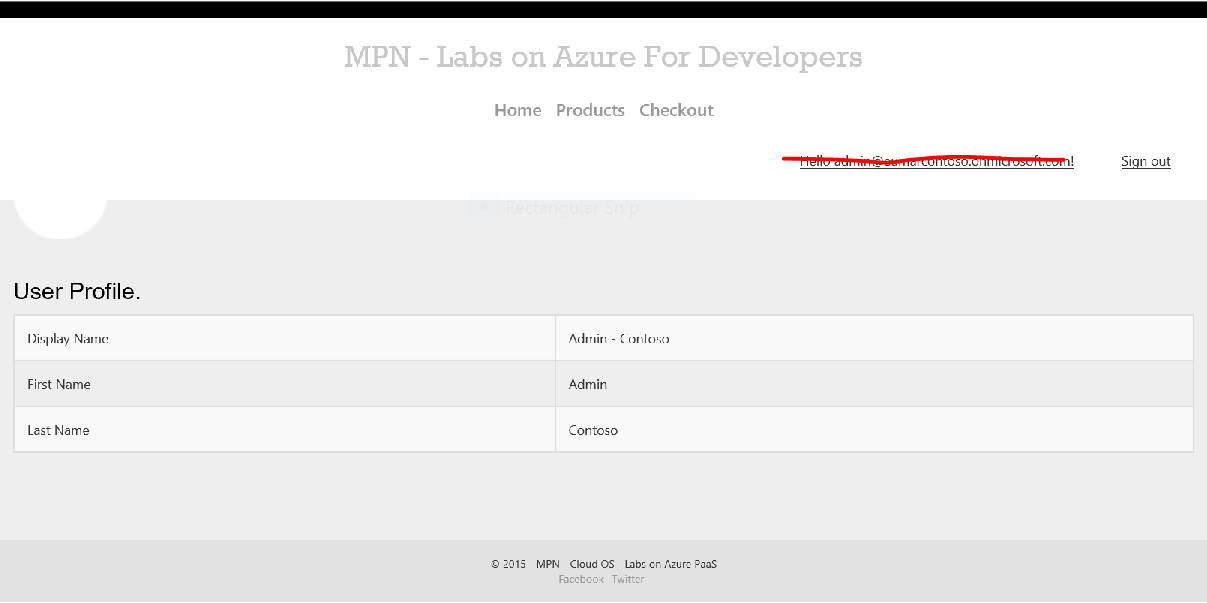
* After configuring these variables, you should be able to view the CloudShopAzureAD in your browser, the app should look like this.



(After sign in – Initial Page)



(After sign in – profile page)



* If you are in the doubt on how to set up this variables, read the section below

## Configuring the Client ID, Key, and Tenant ID

When an application is registered with AAD, you’re provided with a Client ID, Key (which is your password), and Tenant ID. These values need to be added into your custom application to “hook” it to AAD during the authentication process. To get these values, login to the Azure portal, click on the Active Directory option and locate the directory where your custom application was registered.

Click on the target directory and then locate the Applications item in the menu. After clicking on Applications you can select the application that was previously registered with AAD. Next, click the Configure item in the menu to get to the Client ID and Key. An example of what these values looks like once you’re in the application’s configuration screen is shown in Figure 4.

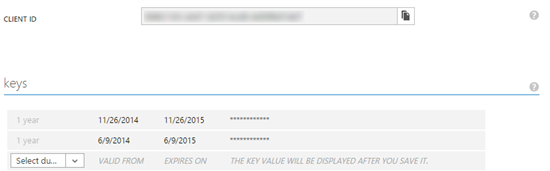
[](http://www.itunity.com/content/content/770/wahlin_fig4.png)

Figure 4. Accessing the Client ID and Key in AAD for a registered application.

If you didn’t store the key value when you initially registered the application, it won’t be available to you now. In that case, you can delete the old key if it’s not already being used (be careful here!) and create a new one by selecting a value from the dropdown. Keys can have a duration of 1 or 2 years. After you save the application configuration changes, the key will be displayed. Make a note of the Client ID and Key before continuing.

Look up at the URL in your browser and find the GUID value that follows /Directory in the URL (it’ll be in the TENANTID\_GOES\_HERE section of the URL below):

https://manage.windowsazure.com/tenant.onmicrosoft.com#Workspaces/ActiveDirectoryExtension/Directory/TENANTID\_GOES\_HERE/RegisteredApp/APP\_GUID/registeredAppConfigure

This GUID value is your Tenant ID and is required as well, so make a note of it.

Now that the Client ID, Key, and Tenant ID are available, open your application’s web.config file in Visual Studio and add the following code into the <appSettings> section to configure the AAD values:

<!-- Update your Tenant, TenantID, ClientID and Key/Password values from Azure Active Directory -->

<add key="ida:Tenant" value="YOUR TENANT NAME" />

<add key="ida:TenantID" value="YOUR TENANT ID" />

<add key="ida:ClientID" value="YOUR CLIENT ID" />

<add key="ida:Password" value="YOUR KEY/PASSWORD" />

<add key="ida:AuthorizationUri" value="https://login.windows.net" />

The ida:Tenant value will be the tenant value you signed up with while the ida:TenantID, ida:ClientID, and ida:Password values are the ones you noted previously while in the AAD application configuration screen.

See more at:

* <http://www.itunity.com/article/integrating-aad-services-angularjs-office-365-part-3-770#sthash.D5gixiHp.dpuf>
* <http://www.itunity.com/article/integrating-aad-services-angularjs-office-365-part-3-770>

Lab Completed