Microsoft Azure - Starter Kits for Partners

Hands on Lab

Dev & Test Scenario

Last Update: September 2015





**MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.**

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

The descriptions of other companies’ products in this document, if any, are provided only as a convenience to you. Any such references should not be considered an endorsement or support by Microsoft. Microsoft cannot guarantee their accuracy, and the products may change over time. Also, the descriptions are intended as brief highlights to aid understanding, rather than as thorough coverage. For authoritative descriptions of these products, please consult their respective manufacturers.

© 2014 Microsoft Corporation. All rights reserved. Any use or distribution of these materials without express authorization of Microsoft Corp. is strictly prohibited.

Microsoft and Windows are either registered trademarks of Microsoft Corporation in the United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

Contents

[Overview 4](#_Toc397013869)

[Objectives 4](#_Toc397013870)

[Prerequisites 5](#_Toc397013871)

[Exercises 5](#_Toc397013872)

[Infrastructure Provisioning 6](#_Toc397013873)

[Exercise 1: Creating Virtual Machines for IIS 6](#_Toc397013874)

[Exercise 2: Creating a SQL Server Virtual Machine 24](#_Toc397013875)

[Exercise 3 –Provisioning Automation with Custom Images and PowerShell 48](#_Toc397013876)

[Application Deployment and Provisioning 53](#_Toc397013877)

[Exercise 4 - Deploying a Simple MVC4 Application - Manually 53](#_Toc397013878)

[Exercise 5 – Automate Application Deployment to Virtual Machines with WebDeploy, PowerShell and Visual Studio 62](#_Toc397013879)

[Summary 78](#_Toc397013880)

[Additional References: 78](#_Toc397013881)

## Overview

Using Microsoft Azure, as your Infrastructure as a Service (IaaS) platform, will enable you to create and manage your infrastructure quickly, provisioning and accessing any host ubiquitously. Grow your business through the cloud-based infrastructure, reducing the costs of licensing, provisioning and backup.

In this hands-on Lab, you will learn how to deploy a simple ASP.NET MVC 4 Web application to a Web server hosted in Microsoft Azure, using SQL Server and configuring load balancing. Additionally, you will learn how to automate your application deployment to Virtual Machines through PowerShell and WebDeploy.

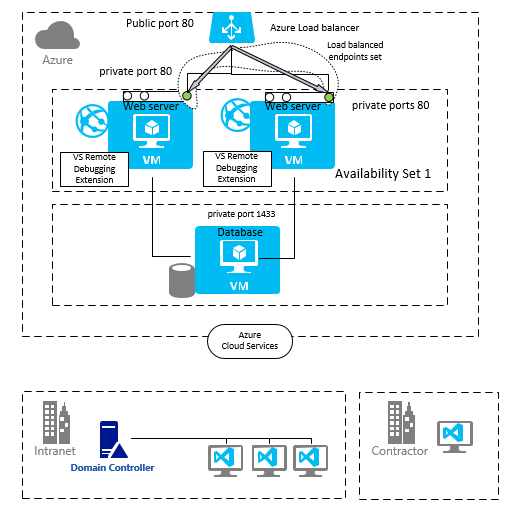
**Audience**: IT Pro, Architect, Application Owners and Developers

### Objectives

In this hands-on lab, you will learn how to:

* Create a Web Farm using Microsoft Azure Management Portal
* Configure Load Balancing in IIS
* Deploy a Simple MVC4 Application that consumes SQL Server Features
* Create a Virtual Machine with SQL Server Full-Text Search feature to be consumed by the MVC Application
* Automate provisioning with PowerShell
* Use Visual Studio to connect and debug application on Azure IaaS.

The Hands On covers the following Architecture:



### Prerequisites

The following is required to complete this hands-on lab:

* A Microsoft Azure subscription - [sign up for a free trial](http://aka.ms/WATK-FreeTrial)
* [Azure PowerShell 0.7.4](http://go.microsoft.com/fwlink/p/?linkid=320376)
* [Windows PowerShell 3.0](http://go.microsoft.com/fwlink/p/?LinkId=393708)

## Exercises

This hands-on lab includes the following exercises:

1. Creating Virtual Machines for IIS
2. Creating a SQL Server Virtual Machine
3. Deploying a Simple MVC4 Application
4. Deployment Imaging and Automation with PowerShell
5. DEVELOPER: Using Visual Studio with Virtual Machines

Estimated time to complete this lab: **60 minutes**.

## Infrastructure Provisioning

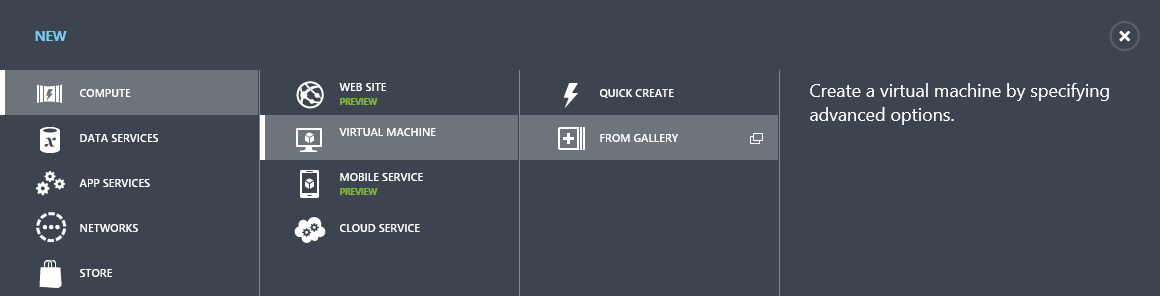
### Exercise 1: Creating Virtual Machines for IIS

In this exercise, you will learn how to create a Virtual Machine in Microsoft Azure. Then, you will configure an Internet Information Server adding roles to use later on in this lab.

#### Task 1 - Creating IIS Virtual Machines

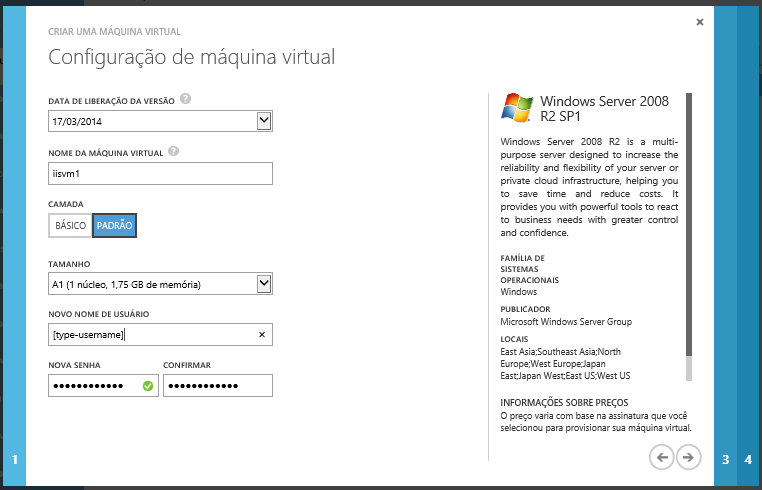
In this task, you will provision a Virtual Machine and configure the Load Balancing to host an MVC4 application.

1. Open Internet Explorer and browse to <https://manage.windowsazure.com/> to enter the Microsoft Azure portal. Then, log in with your credentials.
2. In the menu located at the bottom, select **New | Compute | Virtual Machine | From Gallery** to start creating a new virtual machine.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/creating-a-new-virtual-machine.png?raw=true)

Creating a new Virtual Machine

1. In the **Virtual Machine OS Selection** page, click **Platform Images** on the left menu and select the **Windows Server 2008 R2 SP1** OS image from the list. Click the arrow to continue.
2. In the **Virtual Machine Configuration** page, leave the version release date by default (latest). Enter the Virtual Machine Name (i.e. "iisvm1"), provide a user name for the **New User Name** field and a password for the **New Password** and **Confirm Password** fields. This password needs to contain three of these – lower case characters, uppercase characters, numbers and special characters. Make sure you remember your choice. Lastly, set the Virtual Machine **Size** to Small and click **Next** to continue.

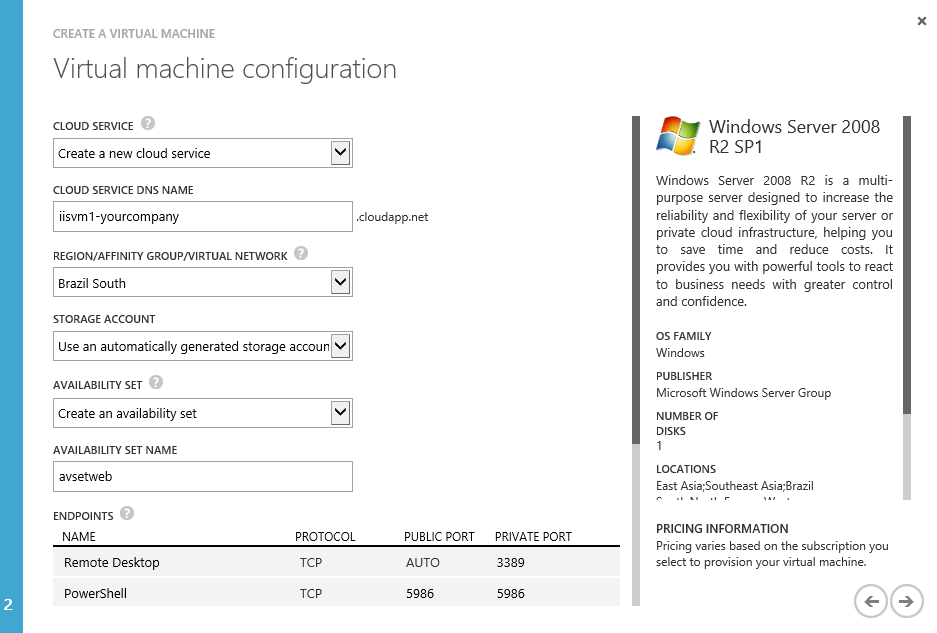


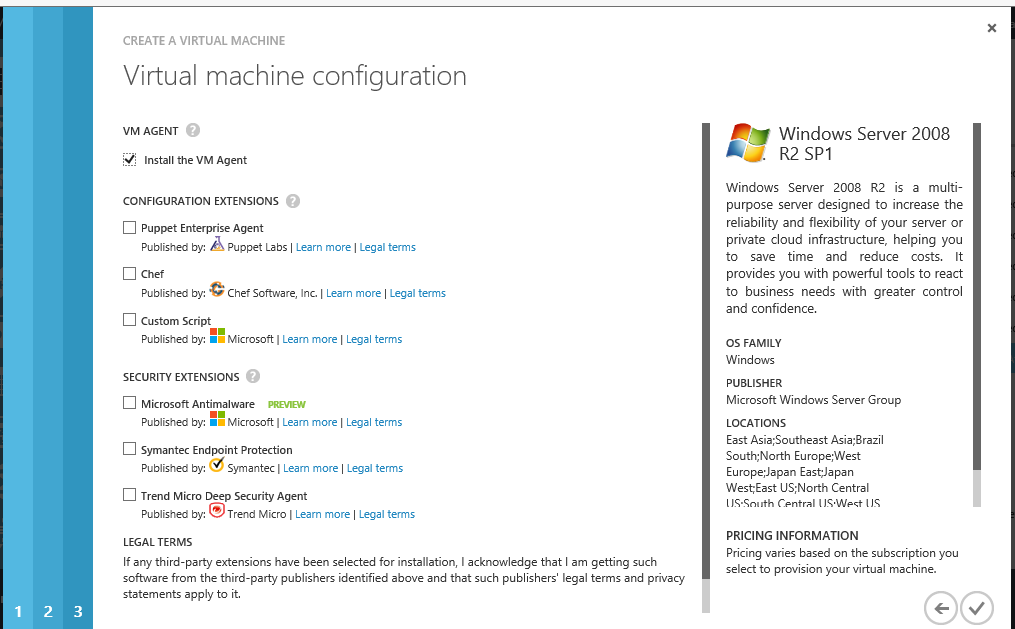
Creating a Virtual Machine - Configuration

**Note:** It is suggested to use secure passwords for admin users, as Microsoft Azure virtual machines could be accessible from the Internet knowing just their DNS.

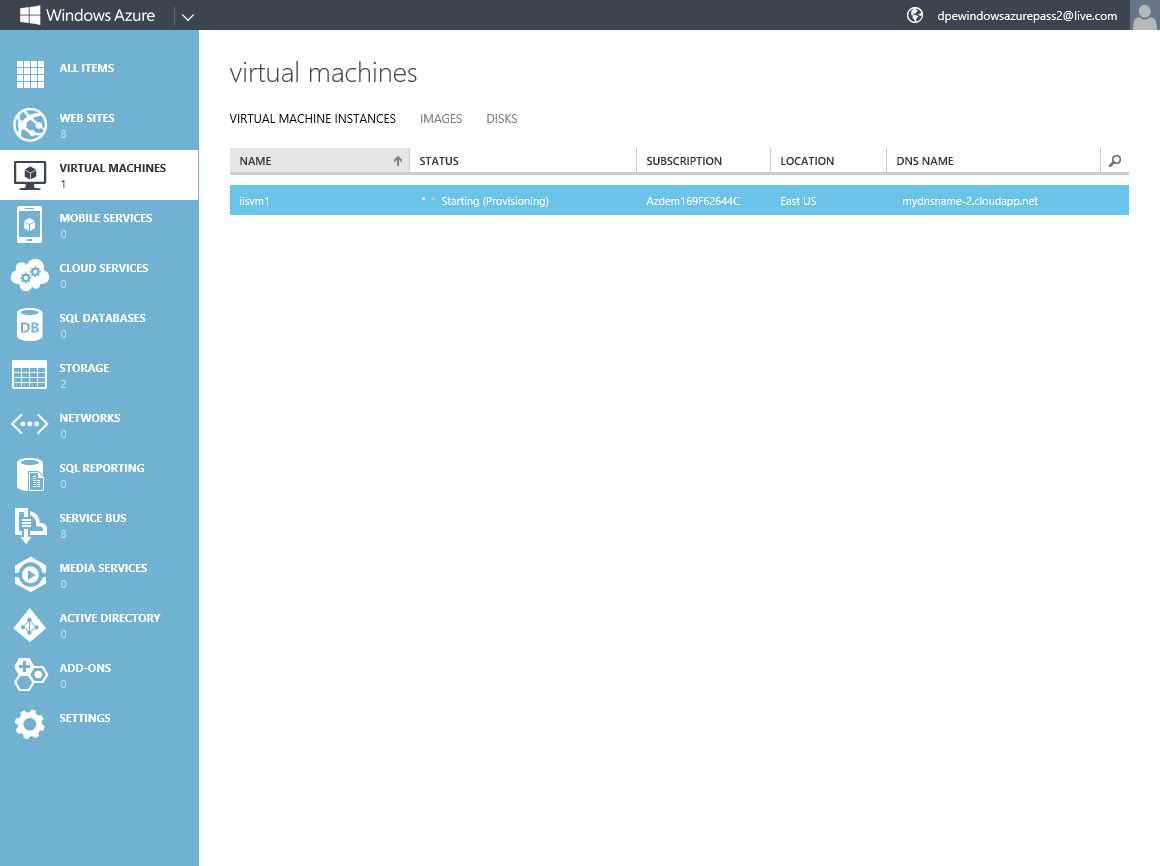
You can also read this document on the Microsoft Security website that will help you select a secure password: <http://www.microsoft.com/security/online-privacy/passwords-create.aspx>

1. In the **CREATE A** **Virtual Machine Mode** page, select **Crate a New Cloud Service**, enter the **Cloud Service** **DNS Name**, select a **Storage Account** or leave the default value Use Automatically Generated Storage Account, and select a **Region/Affinity Group/Virtual Network**. Click the **right arrow** to continue.
   1. Select a **Storage Account** or leave the default value Use Automatically Generated Storage Account, and select a **Region/Affinity Group/Virtual Network**. Click the **right arrow** to continue.





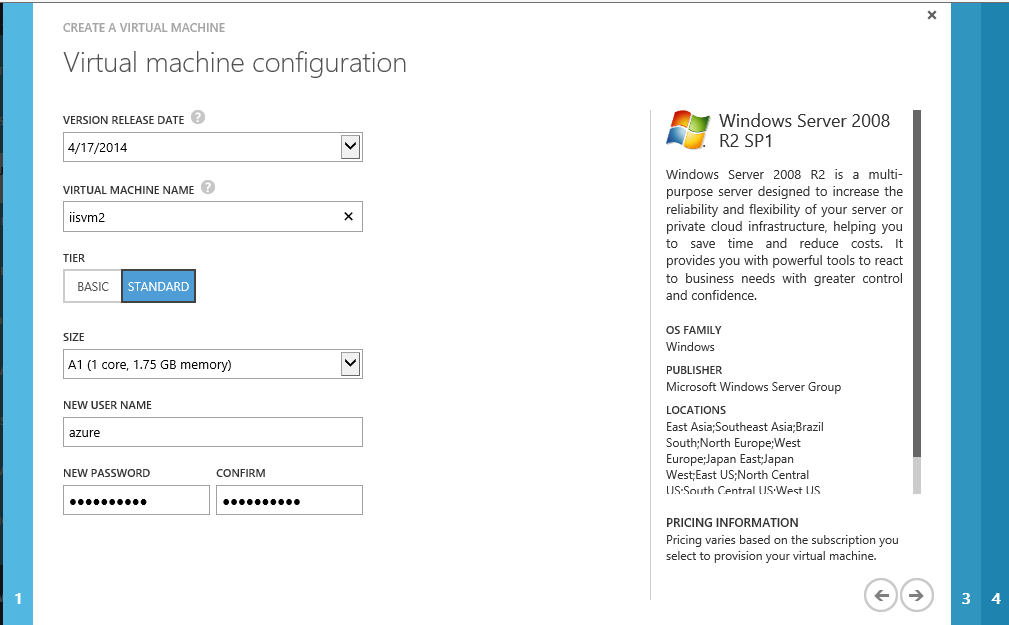
1. In the **Virtual Machines** section, you will see the Virtual Machine you created with a Starting (provisioning) status. Wait until it changes to Running in order to continue with the following step as you will need a provisioned Virtual Machine on the following steps.

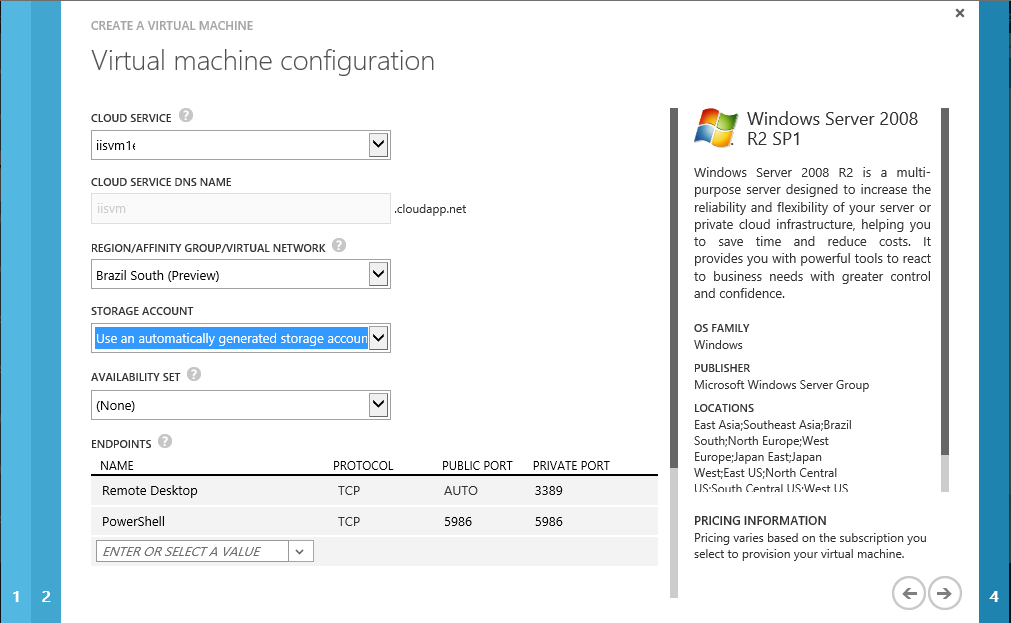
[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/creating-vm-for-iis-web-farm.png?raw=true)

Creating Virtual Machine for IIS Web Farm

**Note:** It will take from 8 to 10 minutes for the Virtual Machine to complete the provisioning process.

1. You will now add the second Virtual Machine for the IIS Load Balancing. In the portal, select **New | Compute | Virtual Machine | From Gallery**.
2. In the **Virtual Machine OS Selection** page, click **Platform Images** on the left menu and select the **Windows Server 2008 R2 SP1, February** OS image from the list. Click the **arrow** to continue.
3. In the **Virtual Machine Configuration** page, set the version release date to **February 12, 2013**. Enter the **Virtual Machine Name** (i.e. "iisvm2"), a **User Name**, a **Password** and the **Size**. Click the **right arrow** to continue.
4. In the **Create a Virtual Machine** page, select **Cloud Service** and choose the first **cloud service** you created from the drop down list. Select a **Storage Account** or leave the default value Use Automatically Generated Storage Account and click the **right arrow** to continue. This step adds the new virtual machine to the cloud service created in the previous step. This allows the virtual machines to be on the same network.





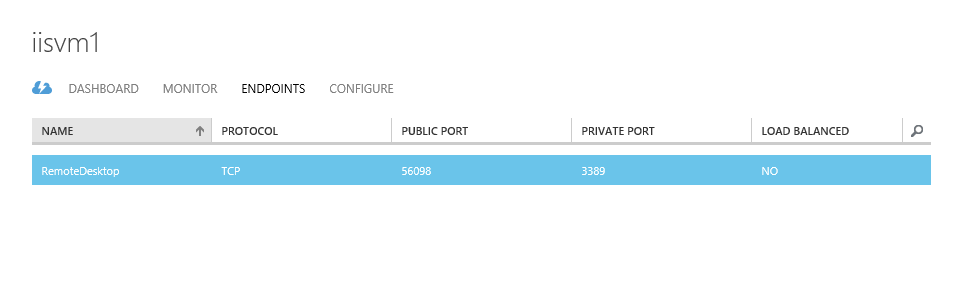
Select an AVSetweb

Creating a Virtual Machine - Virtual Machine Mode

1. In the **Virtual Machine Options** page, leave the default values and click the button to create a new Virtual Machine.
2. Wait until the second Virtual Machine is created. You can check the Virtual Machine status from the Virtual Machines section within the portal.

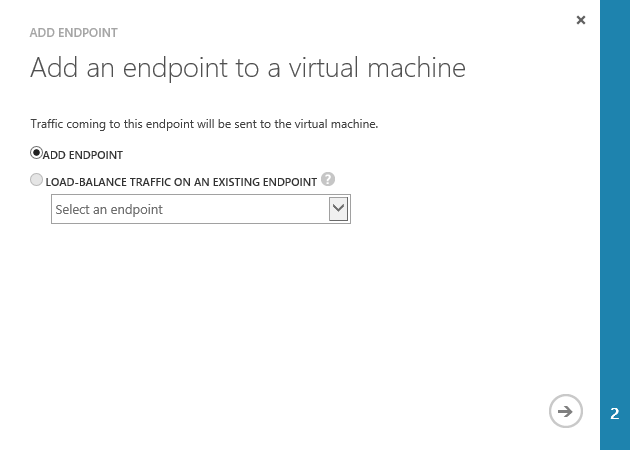
**Note:** It will take from 8 to 10 minutes for the virtual machine to complete the provisioning process.

1. After creating the second virtual machine, you will create an endpoint in the port 80 in the Virtual Machine you created first. To do this, click on the first Virtual Machine Name (iisvm1) to go to the **Dashboard** page and then click **Endpoints**. Click **Add Endpoint** on the bottom pane.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/adding-a-new-endpoint-dashboard.png?raw=true)

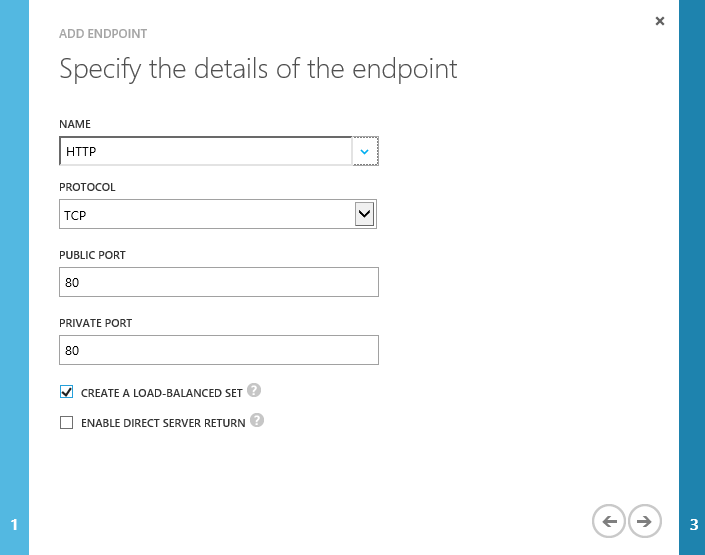
Selecting Add Endpoint in the dashboard

1. Make sure that **Add Endpoint** option is selected and then click the **right arrow** button to continue.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/adding-a-new-endpoint.png?raw=true)

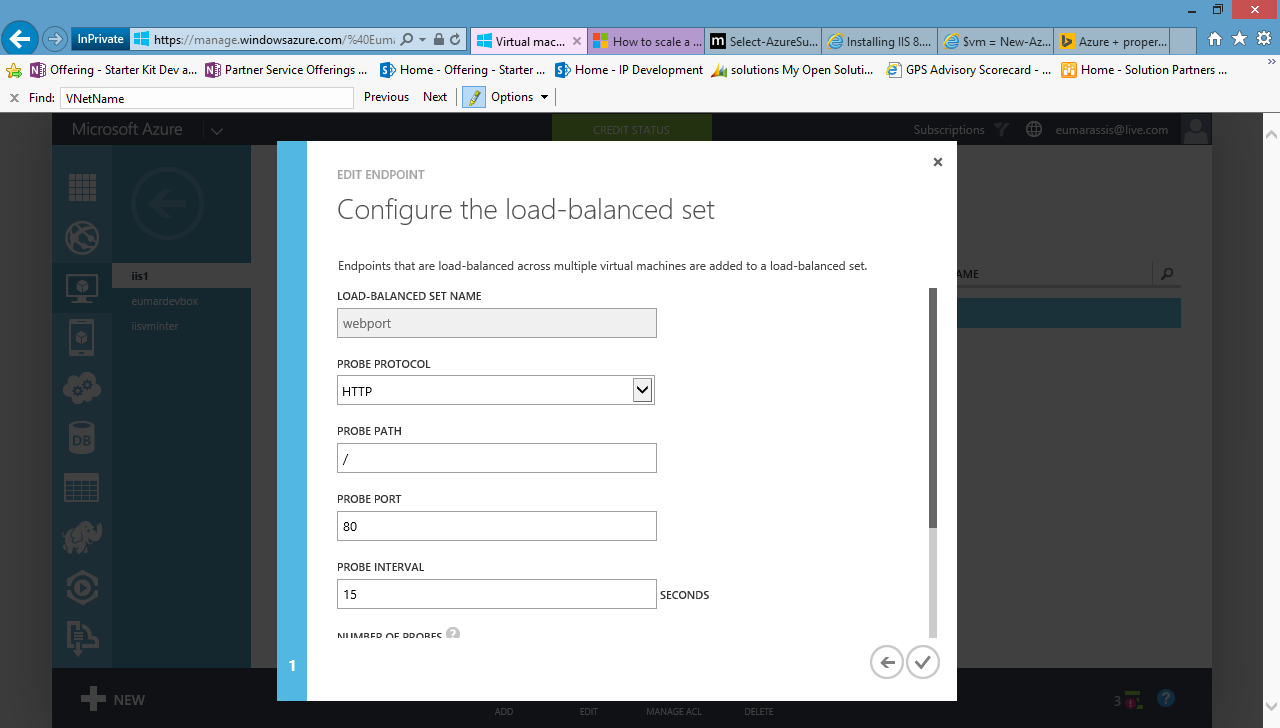
Adding a new Endpoint

1. In the **Specify endpoint details** page, set the **Name** to webport, the **Protocol** to TCP and the **Public Port** and **Private Port** to 80. Click the button to create the endpoint. Wait until the Endpoint is created before continue to the following steps.



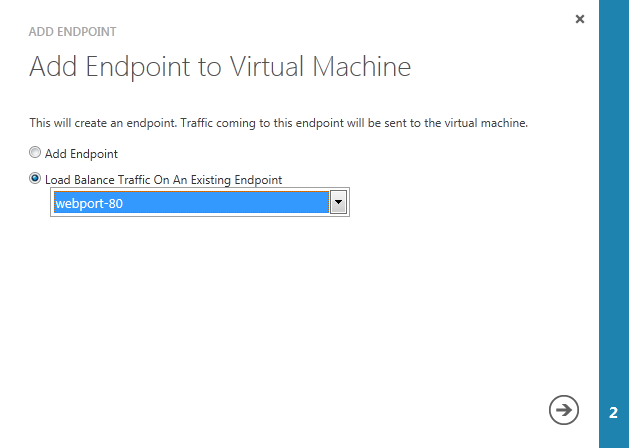
New Endpoint Details

1. In the Configure the Configure the **load-balanced** set, input the following and click on



**Note:** It will take some minutes to create a new endpoint. **Wait to finish before going to the next step**

1. Now, create a new Endpoint in the second Virtual Machine in order to enable Load Balancing between both Virtual Machines. To do this, click **Virtual Machines** and then select the second Virtual Machine you created. Then, click **Endpoints**.
2. Click **Add Endpoint**, select **Load Balance Traffic On An Existing Endpoint** option. Select the endpoint you created for the first Virtual Machine from the drop down list and then click the **right arrow** to continue.

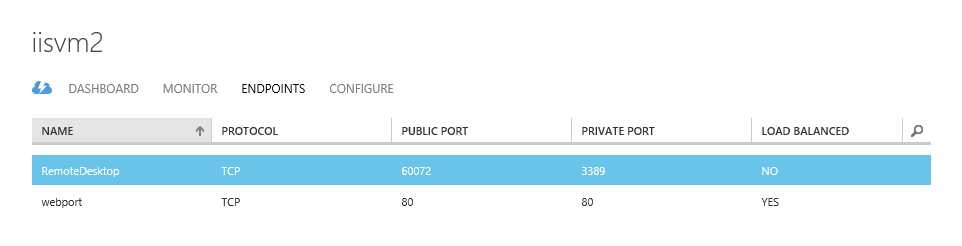
[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/load-balance-traffic-an-an-existing-endpoint.png?raw=true)

Load Balance Traffic On An Existing Endpoint

1. In the **New Endpoint Details** page, set the **Name** to webport and the **Private Port** to 80. Click the button to create the endpoint.

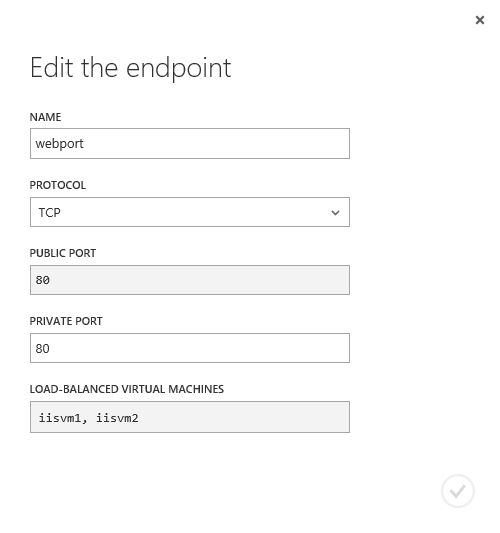
**Note:** It will take some minutes to create a new endpoint.

1. In the **Virtual Machines** section, click on the first Virtual Machine Name (iisvm1) and then click **Endpoints**.
2. Select the **webport** endpoint you have created. Make sure the **Load Balancer** column value is **Yes**.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/creating-load-balancing-endpoint-1.png?raw=true)

Verification: enabling IIS Load Balancing

1. Click the **Edit Endpoint** button in the bottom bar to enter the endpoint details and verify the load balancing is enabled. Repeat this step in the second Virtual Machine.

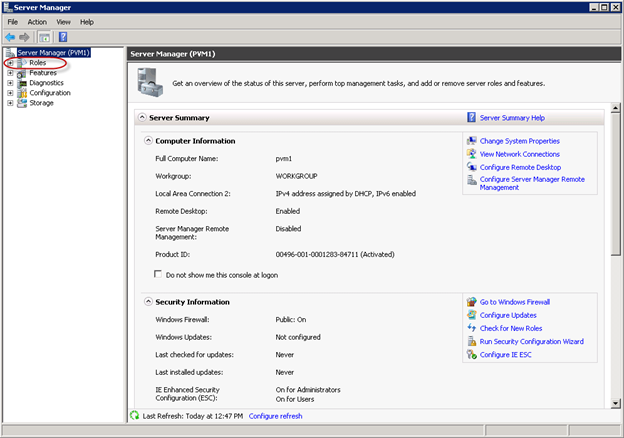
[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/creating-load-balancing-endpoint-2.png?raw=true)

Verification: enabling IIS Load Balancing, details

#### Task 2 - Configuring IIS Virtual Machines

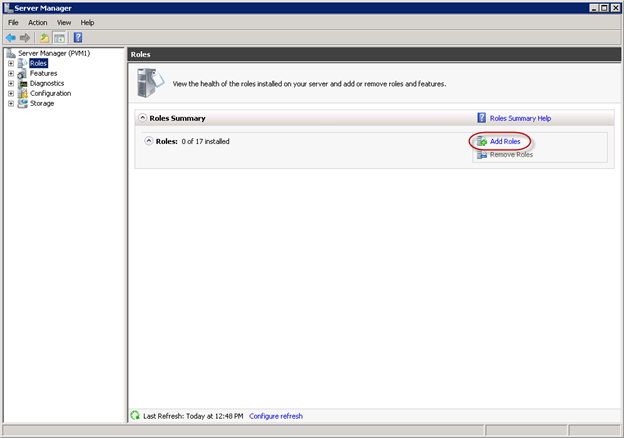
In this task, you will configure the IIS Virtual Machines by adding the necessary roles to deploy the MVC application.

1. In the Portal, click **Virtual Machines** on the left menu.
2. You will see a list with your existing Virtual Machines. Select the first one you created in Task 1 and click the **Connect** button in the bottom bar. If you used the proposed name, this Virtual Machine should be named **iisvm1**.
3. You will be asked to download the remote desktop settings file. Click **Open** and log on using the credentials you defined when creating the Virtual Machine.
4. In the Azure Virtual Machine, open **Server Manager** from **Start | Administrative Tools**.
5. In the **Server Manager** window, select **Roles** node.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/Server-Manager.png?raw=true)

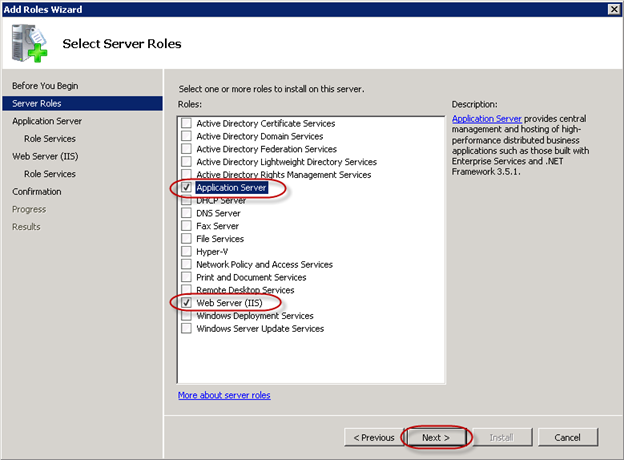
Server Manager

1. Click **Add Roles** link.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/Adding-Server-Roles.png?raw=true)

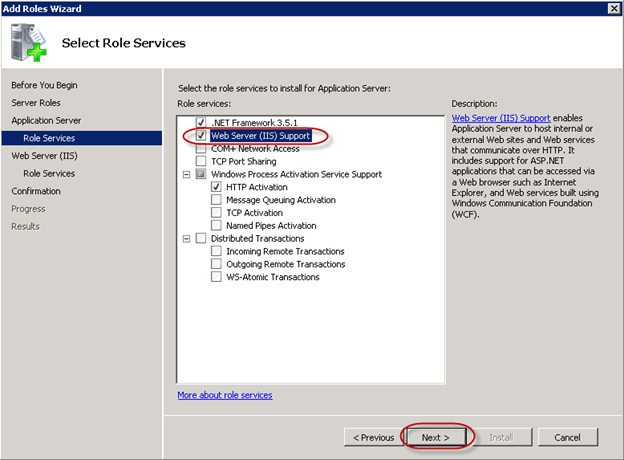
Adding Server Roles

1. The **Add Roles Wizard** will appear.
2. In the **Before You Begin** page, read the content and click **Next**.
3. In the **Select Server Roles** page, check the **Application Server** and **Web Server (IIS)**. A warning will show, informing the Required Role Services that are missing. Click **Add Required Features** to install them and then click **Next**.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/add-roles-wizard2.png?raw=true)

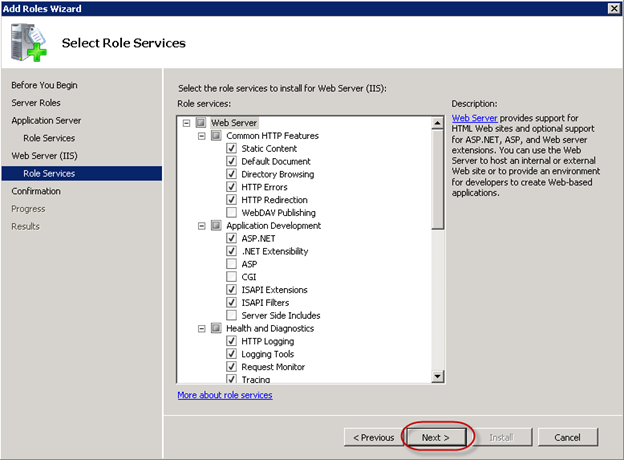
Add Roles Wizard

1. The **Application Server** page provides a brief introduction about Application Server's capabilities. Click **Next** when you complete reading it.
2. In the **Select Role Services** page for **Application Server**, select **Web Server (IIS)** **Support** and make sure **.NET Framework 3.5.1** is selected. It will prompt a dialog warning about missing Required Role Services. Click **Add Required Role Services** to install them and then click **Next**

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/add-roles-wizard3.png?raw=true)

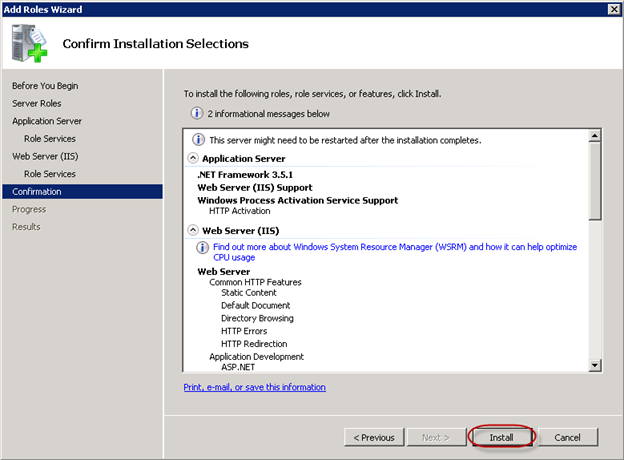
Add Roles Wizard

1. The **Web Server (IIS)** page provides a brief introduction about Web Server (IIS) capabilities. Click **Next** when you complete reading.
2. The **Select Role Services** page for **Web Server (IIS)** page will display the selected role services that will be installed. Click **Next**.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/add-roles-wizard4.png?raw=true)

Add Roles Wizard

1. In the **Confirm Installation Selections** page, make sure the displayed services that will be installed are the ones you have selected (.NET Framework 3.5.1 support and IIS), and then click **Install**.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/Add-Roles-Wizard.png?raw=true)

Add Roles Wizard

**Note:** It will take some minutes to complete the installation.

1. Close the **Remote Desktop Connection**.

Repeat this task on the second Virtual Machine to install IIS, starting from step 4. If you used the proposed name, the second Virtual Machine should be named **iisvm2**.

### Exercise 2: Creating a SQL Server Virtual Machine

In this exercise, you will create a new Virtual Machine and learn how to install SQL Server. You will add disk images to the existing Virtual Machine in order to split the data from the logs generated by SQL Server.

#### Task 1 - Creating a SQL Server Virtual Machine

In this task, you will create a new Virtual Machine using the Microsoft Azure portal in the same Cloud App you deployed the IIS Virtual Machines.

1. In the menu located at the bottom, select **New | Compute | Virtual Machine | From Gallery** to start creating a new virtual machine.
2. In the **Virtual Machine OS Selection** page, click **Platform Images** on the left menu and select the **Microsoft SQL Server 2012** image from the list. Click the arrow to continue.
3. In the **Virtual Machine Configuration** page, enter the **Virtual Machine Name** (i.e. "sqlvm1"), a **User Name**, a **Password** and the **Size**. Click the **right arrow** to continue.
4. In the **Virtual Machine Mode** page, select the **Cloud Service you choose** for the first IIS Virtual Machine you created from the drop down list (iisvm1). Click the **right arrow** to continue.
5. In the **Virtual Machine Options** page, leave the default values and click the button to create a new Virtual Machine.
6. In the **Virtual Machines** section, you will see the Virtual Machine you created with a provisioning status. Wait until it changes to Running in order to continue with the following step.

**Note:** It will take from 8 to 10 minutes for the Virtual Machine to complete the provisioning process.

#### Task 2 - Attaching Empty Disk Images

In this task, you will create two empty data disks and attach them to an existing Virtual Machine using the Microsoft Azure Management Portal. You will use these data disks to split SQL Server Data and Logs.

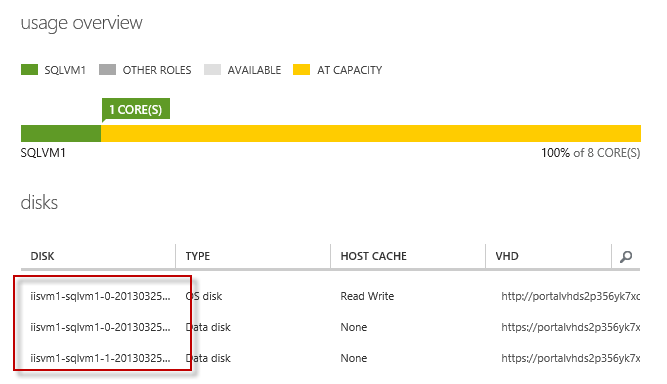
1. Now, you will create and attach empty data disks to store the SQL Server logs and data files, and you will also add an endpoint. To do this, in the **Virtual Machines** section, select the SQL Server Virtual Machine you created in the previous task.
2. In the Virtual Machine's **Dashboard**, click **Attach** in the menu at the bottom of the page and select **Attach Empty Disk**.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/attach-empty-disk.png?raw=true)

Attach Empty Disk

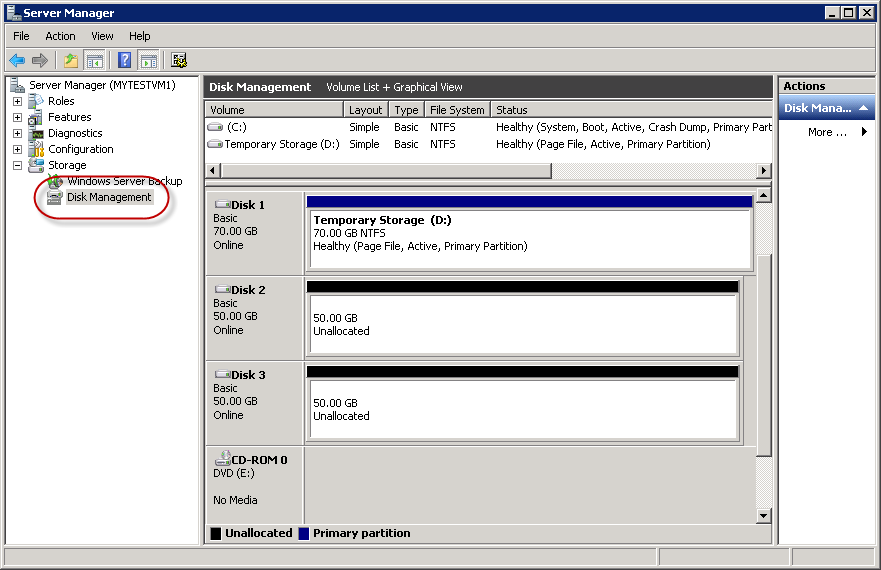
1. In the **Attach Empty Disk** page, set the **Size** to 50 GB and create the Disk.
2. Wait until the process to attach the disk finishes. Repeat the steps 1 to 3 to create a second disk.
3. Open the Virtual Machine's **Dashboard**. You will see three disks: one for the **OS** and other two for **Data** and **Logs**.

**Note:** It might take a few minutes until the data disks appear in the Virtual Machine's dashboard within the Azure Portal.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/Attached-Data-Disks.png?raw=true)

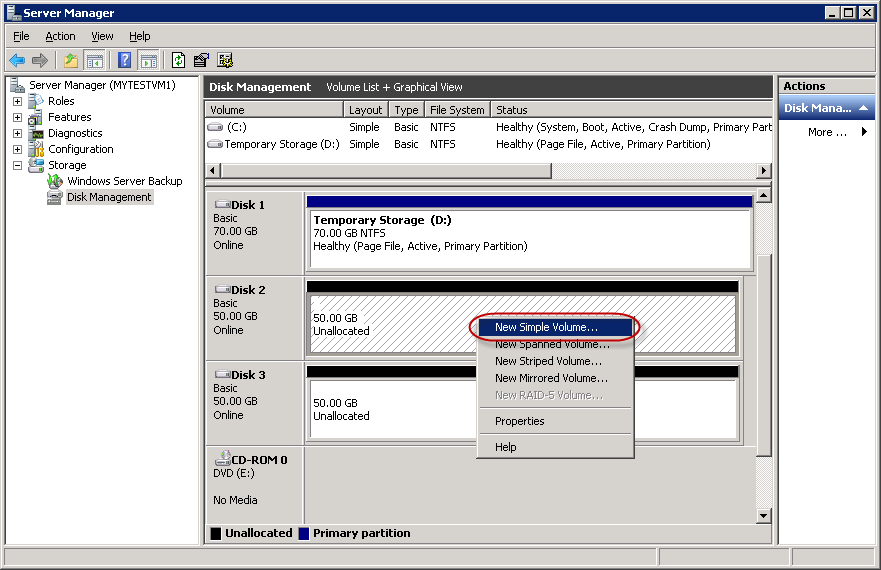
Attached Data Disks

1. Finally, you need to format the disks in order to access them from the Virtual Machine. To do this, click **Connect** to connect to the Virtual Machine using **Remote Desktop connection**.
2. It will ask you to download the remote desktop settings file. Click **Open** and log on using the credentials you defined when creating the Virtual Machine.
3. In the virtual machine, open **Server Manager** from **Start | Administrative Tools**.
4. Expand **Storage** node and select **Disk Management** option.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/disk-management2.png?raw=true)

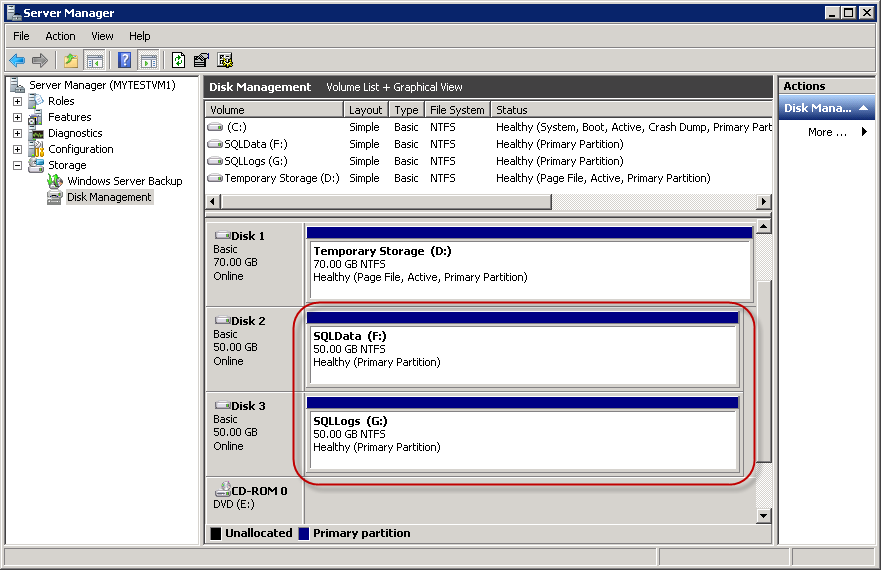
Disks Management

1. The **Initialize Disk** dialog will appear. Leave the default values and click **OK**.
2. Right-click the first disk unallocated space and select **New Simple Volume**.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/disk-management.png?raw=true)

Disks Management

1. Follow the **New Simple Volume Wizard**. When asked for the **Volume Label** use SQLData.
2. Wait until the process for the first disk is completed. Repeat the steps 11 to 12 but this time using the second disk. Set the **Volume Label** to SQLLogs.
3. The **Disk Management** list of available disks should now show the **SQLData** and **SQLLogs** disks like in the following figure:

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/Disks-Management.png?raw=true)

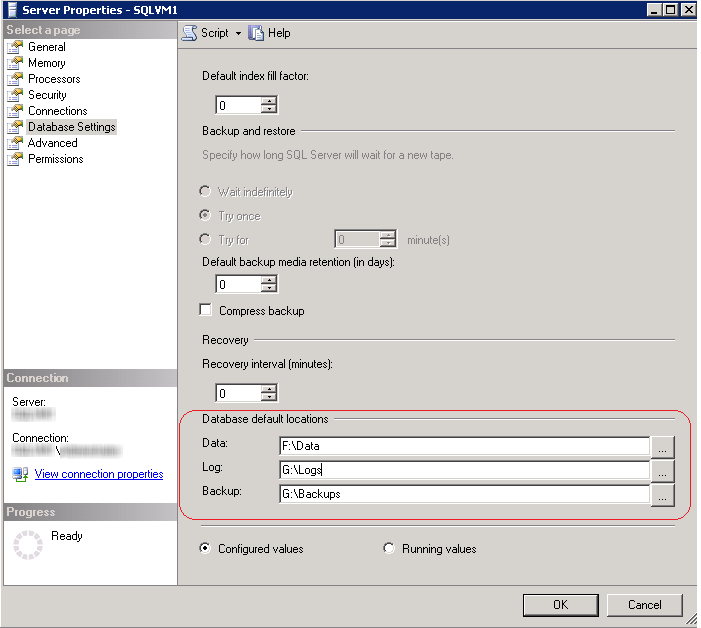
Disks Management

**Note:** Do not close the **Remote Desktop Connection**. You will use it in the following task.

#### Task 3 - Configuring SQL Server in the Virtual Machine

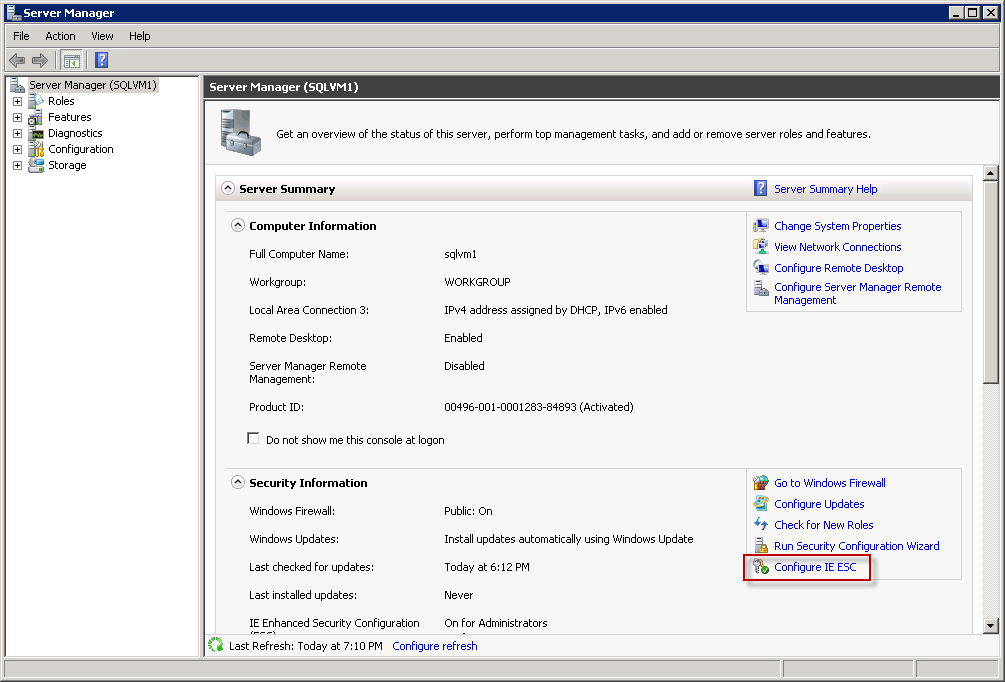
In this task, you will configure SQL Server 2012. You will create the database that will be used by the MVC4 application and add Full-Text Search capabilities to it. Additionally, you will create a SQL Server user for the MVC4 website.

1. Open Windows Explorer and create the following folders: **F:\Data, G:\Logs** and **G:\Backups**.
2. Open the SQL Server Management Studio from **Start | All Programs | Microsoft SQL Server 2012 | SQL Server Management Studio**.
3. Connect to the SQL Server 2012 default instance using your Windows Account.
4. Now, you will update the database's default locations in order to split the DATA from the LOGS. To do this, right click on your SQL Server instance and select **Properties**.
5. Select **Database Settings** from the left side pane.
6. Locate the **Database default locations** section and update the default values to point to the disks you attached in the previous task and then click "Ok".

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/Setting-Database-Default-Locations.png?raw=true)

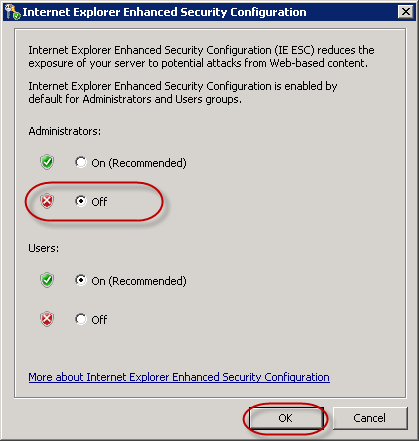
Setting Database Default Locations

1. Restart SQL Server. In the **Object Explorer**, right-click on the server node and select **Restart**.
2. In order to enable downloads from Internet Explorer you will need to update **Internet Explorer Enhanced Security Configuration**. In the Azure Virtual Machine, open **Server Manager** from **Start | Administrative Tools | Server Manager**.
3. In the **Server Manager**, click **Configure IE ESC** within **Security Information** section.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/configuring-internet-explorer-enhanced-security-configuration.png?raw=true)

Configuring IE ESC

1. In the **Internet explorer Enhanced Security** configuration, turn **off** the enhanced security for **Administrators** and click **OK**.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/internet-explorer-enhanced-security2.png?raw=true)

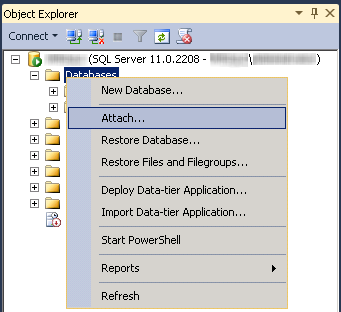
Internet Explorer Enhanced Security

**Note:** Modifying **Internet Explorer Enhanced Security** configurations is not good practice and is only for the purpose of this particular lab. The correct approach should be to download the files locally and then copy them to a shared folder or directly to the Virtual Machine.

1. This lab uses the **AdventureWorks2012** database. Open an **Internet Explorer** browser and go to <http://msftdbprodsamples.codeplex.com/> to download the **SQL Server 2012** sample databases. Once on the page click on **AdventureWorks Databases – 2008, 2008R2 and 2012** and then download Adventure Works 2012 Data File. Download the file to F:\Data.

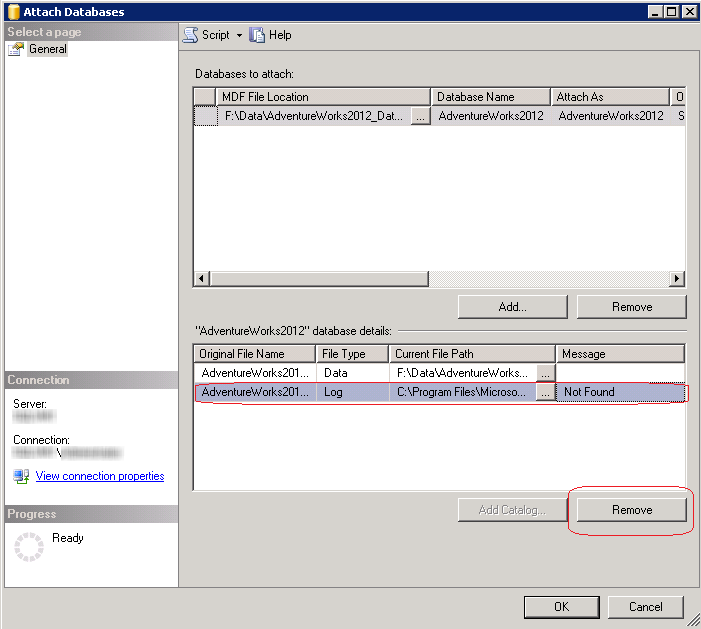
**Note:** The **AdventureWorks2012** database can also be downloaded as a .zip file. If you choose this format, right-click the file to open its properties window and then click **Unblock**. Then, extract the database to F:\Data.

1. Add the **AdventureWorks2012** sample database to your SQL Server. To do this, in the **SQL Server Management Studio**, locate your SQL Server instance node and expand it. Right click the **Databases** folder and select **Attach**.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/attaching-adventureworks-database-menu.png?raw=true)

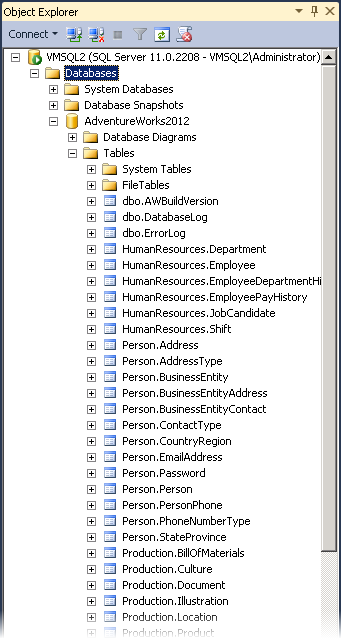
Attaching the database

1. In the **Attach Databases** dialog, press **Add**. Browse to the data disk and select the Adventure Works 2012 data file.
2. Select the **AdventureWorks2012** Log entry and click **Remove**.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/removing-adventureworks-log-entry.png?raw=true)

Removing AdventureWorks2012 Log entry

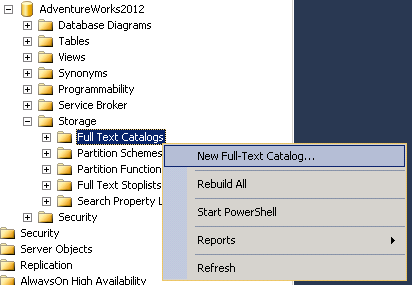
1. Press **OK** to add the database.
2. In the **Databases** folder, locate the new **AdventureWorks2012** database and explore its tables.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/adventureworks-sample-database.png?raw=true)

AdventureWorks Sample Database

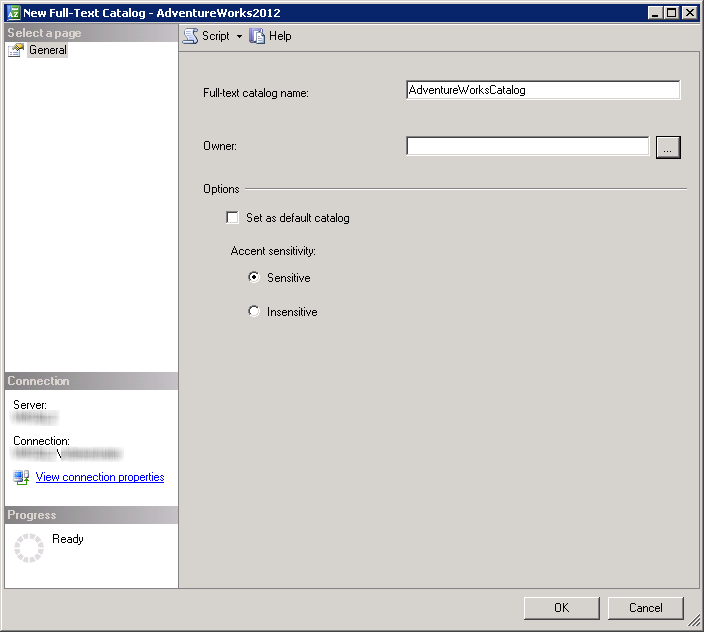
1. Expand **Storage** node within **AdventureWorks2012** database, right-click **Full Text Catalogs** folder and select **New Full-Text Catalog**.

**Note:** You are creating a Full Text Catalog for the database that will be used later by the MVC application.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/create-new-full-text-catalog2.png?raw=true)

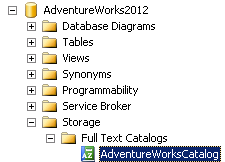
Create New Full-Text Catalog

1. In the New Full-Text Catalog dialog, set the **Name** value to AdventureWorksCatalog and press **OK**.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/create-new-full-text-catalog3.png?raw=true)

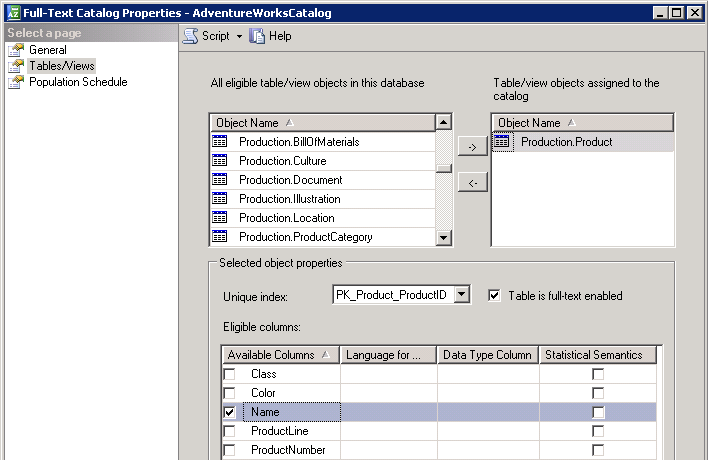
Create New Full-Text Catalog

1. Check that the Full-Text Catalog you created appears in the **Full-Text Catalogs** folder.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/create-new-full-text-catalog5.png?raw=true)

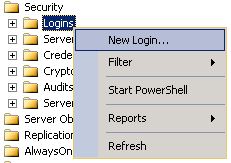
Create New Full-Text Catalog

1. Right-click **AdventureWorksCatalog** and select **Properties**. In the **Full-Text Catalog Properties** dialog, switch to **Tables/Views** page.
2. Add the **Production.Product** table to the **Table/View objects assigned to the Catalog** list. Then, check the Name column and click **OK**.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/create-new-full-text-catalog4.png?raw=true)

Create New Full-Text Catalog

1. Add a new user for the MVC4 application you will deploy in the following exercise. To do this, expand **Security** folder within the SQL Server instance. Right-click **Logins** folder and select **New Login**.

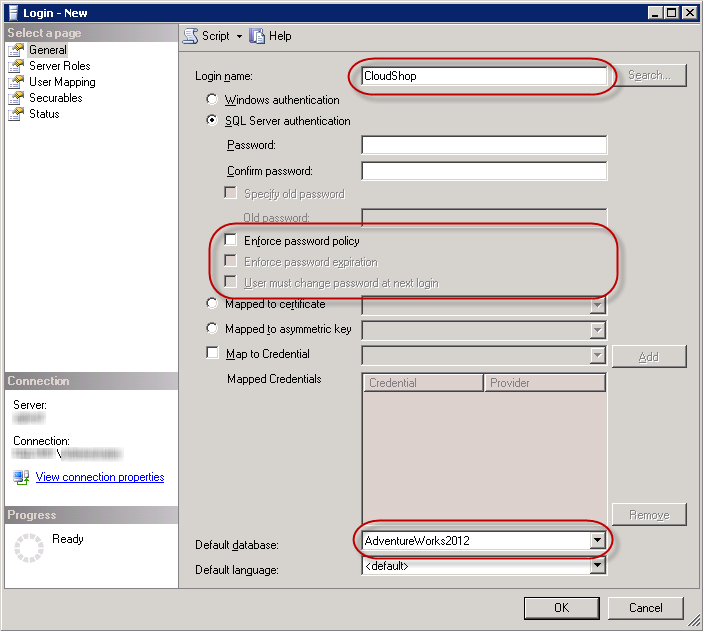
[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/creating-a-new-login2.png?raw=true)

Creating a New Login

1. In the **General** section, set the **Login name** to CloudShop. Select **SQL Server authentication** option and set the **Password** to Azure$123.

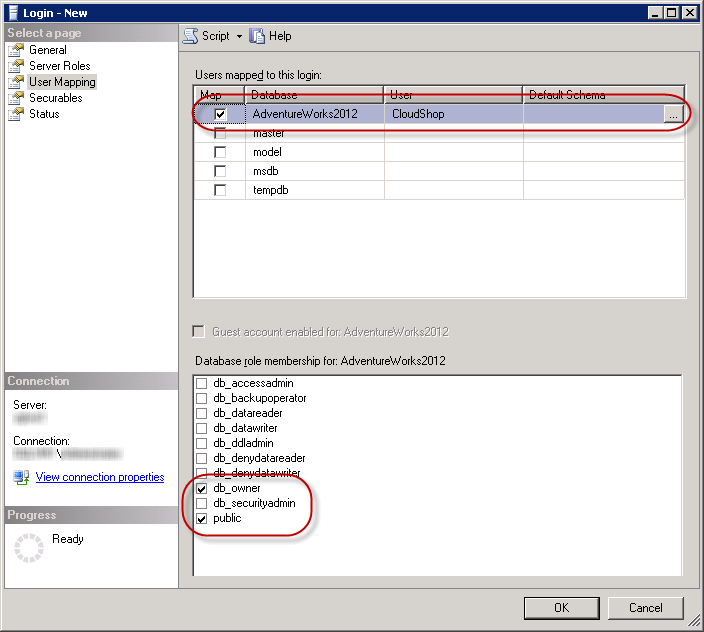
**Note:** If you enter a different username or password than those suggested in this step, do not forget in the next exercise to update the web.config file of the MVC4 application to match those values.

1. Unselect **Enforce password policy** checkbox to avoid having to change the password the first time you log on, and set the **Default database** to AdventureWorks2012.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/Creating-a-New-Login.png?raw=true)

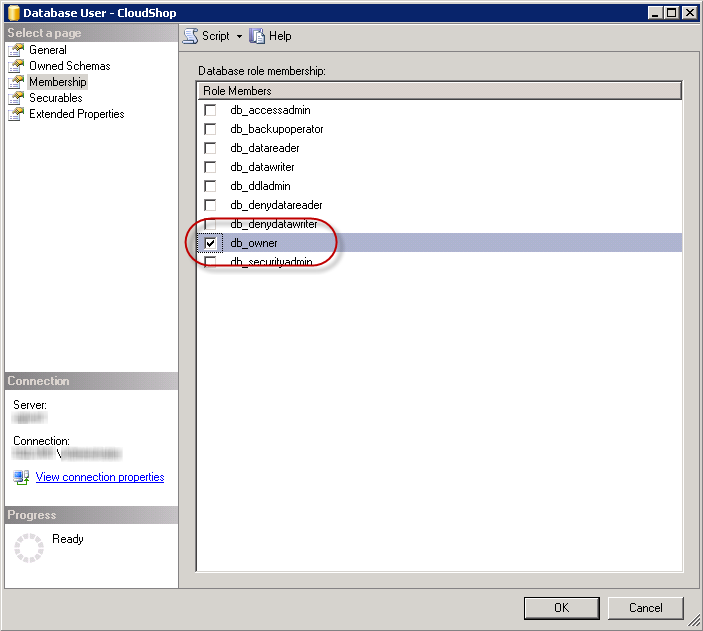
Creating a New Login

1. Click **User Mapping** on the left pane. Select the map checkbox in the AdventureWorks2012 database row and click **OK**.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/mapping-new-user-database-2.png?raw=true)

Mapping the new User to the AdventureWorks Database

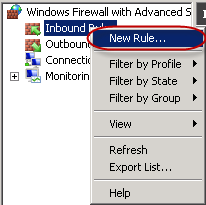
1. Expand **AdventureWorks2012** database within **Databases** folder. In the **Users** folder under **Security**, double-click **CloudShop** user.
2. Select the **Membership** page, and select the db\_owner role checkbox for the **CloudShop** user and click **OK**.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/Adding-Database-role-membership-to-CloudShop-user.png?raw=true)

Adding Database role membership to CloudShop user

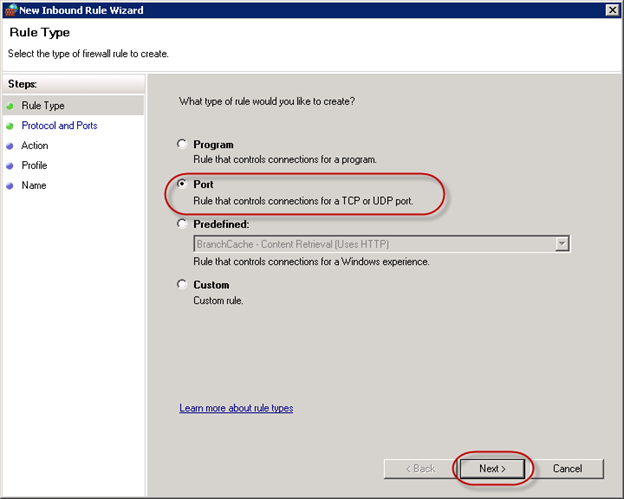
**Note:** The application you will deploy in the next exercise uses Universal Providers to manage sessions. The first time the application runs, the provider will create the Sessions table within the database. For that reason, you are assigning a db\_owner role to the CloudShop user. Once you run the application for the first time, you can remove this role as these permissions will not be needed.

1. Now, enable **Mixed Mode Authentication** to the SQL Server instance. To do this, in the **SQL Server Management Studio**, right-click the server instance and click **Properties**.
2. Click **Security** in the left side pane and then select **SQL Server and Windows Authentication mode** under **Server Authentication** section. Click **OK** to save changes.
3. Restart the SQL Server instance. To do this, right-click the SQL Server instance and click **Restart**.
4. Close the **SQL Server Management Studio**.
5. In order to allow the MVC4 application access the SQL Server database you will need to add an **Inbound Rule** for the SQL Server requests in the **Windows Firewall**. To do this, open **Windows Firewall with Advanced Security** from **Start | Administrative Tools**.
6. Select **Inbound Rules** node, right-click it and select **New Rule** to open the **New Inbound Rule Wizard**.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/Creating-an-Inbound-Rule.png?raw=true)

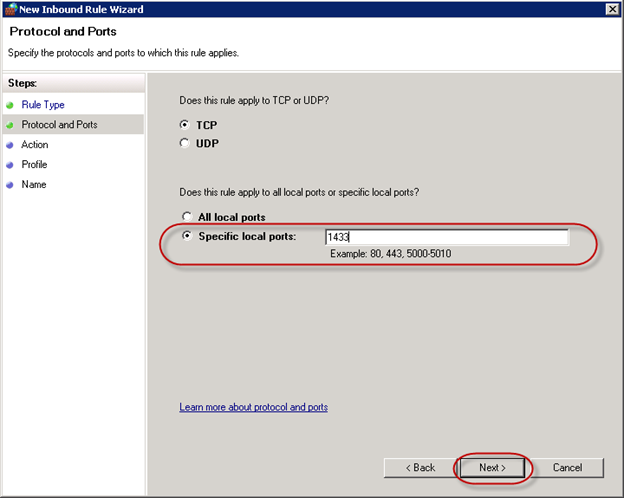
Creating an Inbound Rule

1. In the **Rule Type** page, select **Port** and click **Next**.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/new-inbound-rule-wizard2.png?raw=true)

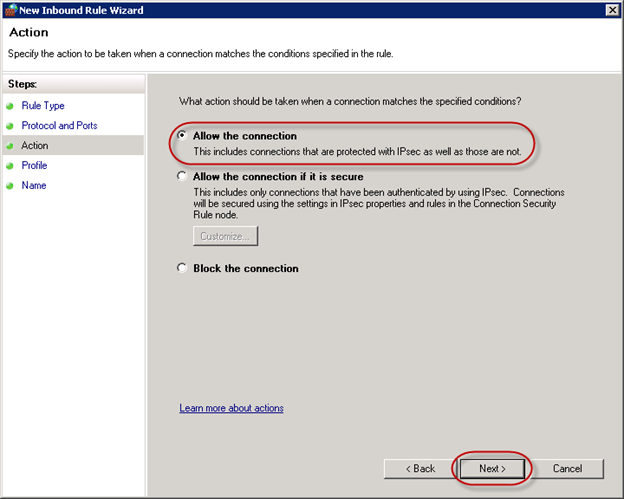
New Inbound Rule Wizard

1. In **Protocol and Ports** page, leave TCP selected, select **Specific local ports,** and set its value to 1433. Click **Next** to continue.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/new-inbound-rule-wizard.png?raw=true)

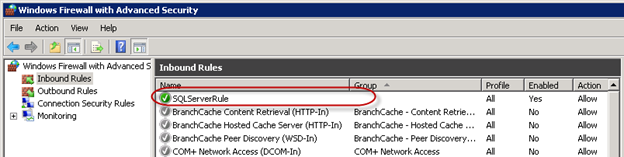
New Inbound Rule Wizard

1. In the **Action** page, make sure that **Allow the connection** is selected and click **Next**.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/new-inbound-rule-wizard3.png?raw=true)

Protocol and Ports

1. In the **Profile** page, leave the default values and click **Next**.
2. In the **Name** page, set the Inbound Rule's **Name** to SQLServerRule and click **Finish**

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/new-inbound-rule-wizard4.png?raw=true)

New Inbound Rule Wizard

1. Close **Windows Firewall with Advanced Security** window.

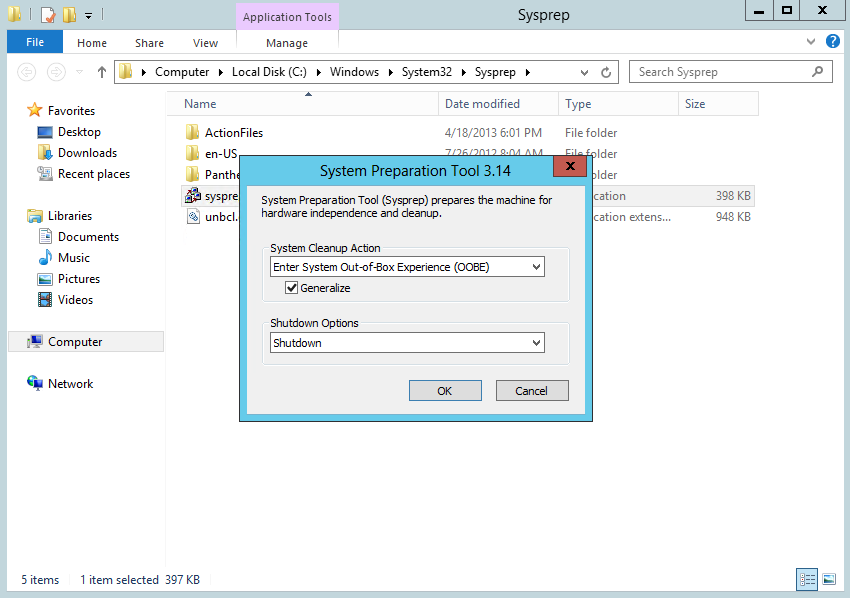
**Note:** Make sure the Named Pipes and TCP/IP protocols are enabled for the Server Instance. You can verify this by going to the SQL Server Configuration Manager and within SQL Server Network Configuration node check that these protocols' status are set to enable.

Remember to restart the SQL Server instance after enabling a protocol.

1. Close the **Remote Desktop Connection**.

### Exercise 3 –Provisioning Automation with Custom Images and PowerShell

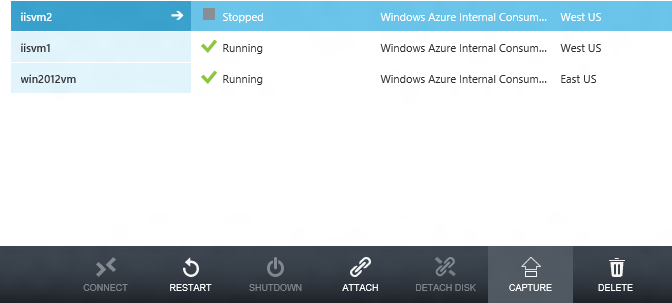
1. Microsoft Azure Virtual Machines allow you to customize a virtual machine by installing software or making configuration changes and then saving the result as a generalized image. In this part you will generalize a customized Windows virtual machine and capture it as an image. You will then create multiple identical virtual machines from the same image.
2. RDP into **iisvm2** and click **Start -> Run**. Type in sysprep and press **ENTER**.
3. In the opened window, right-click on sysprep.exe and select **Run as administrator**.



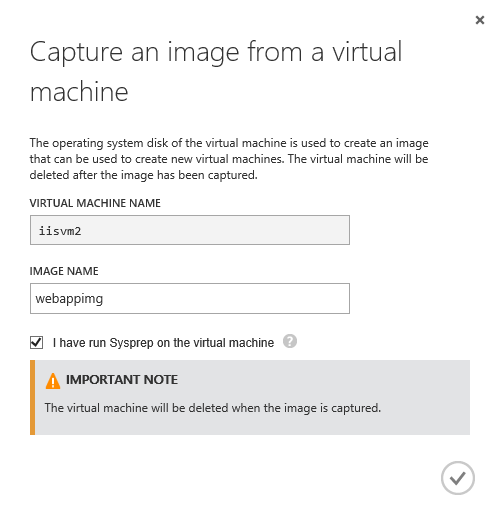
1. Select **Enter System Out-of-Box Experience (OOBE)**, check **Generalize** and select **Shutdown**.
2. Press **OK** and wait for the virtual machine to show the **Stopped** status in the Windows Azure Management portal.

**Note**: It takes from 8 to 10 minutes to stop the virtual machine.

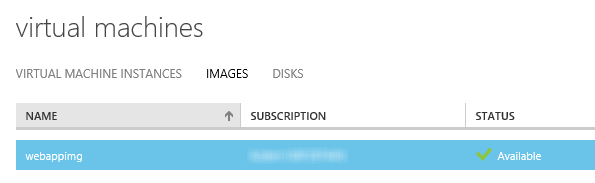
1. Highlight the virtual machine and press the **Capture** button in the toolbar.



1. When the capture dialog appears, name the new image **webappimg** and check **I have run sysprep in the virtual machine**.



1. Once the virtual machine has completed capturing, note that iisvm2 is no longer listed and show the available images by clicking **Images** on the virtual machine's listing page.



1. The final step of the demo is to show how to create multiple virtual machines from the image. Paste in and run the following powershell script to create three identical virtual machines using the image name.

PowerShell

Add-AzureAccount

Select-AzureSubscription -SubscriptionName '[SUBSCRIPTION-NAME]'

$adminUser = "[ADMIN-USER]"

$adminPassword = "[ADMIN-PWD]"

$cloudService = '[SERVICE-NAME]'

$iisvm2 = New-AzureVMConfig -Name 'iisvm2' -InstanceSize Small -ImageName 'WebAppImg' |

Add-AzureEndpoint -Name web -LocalPort 80 -PublicPort 80 -Protocol tcp -LBSetName webport -ProbePath '/' -ProbeProtocol http -ProbePort 80 |

Add-AzureProvisioningConfig -Windows -AdminUserName $adminUser -Password $adminPassword

$iisvm3 = New-AzureVMConfig -Name 'iisvm3' -InstanceSize Small -ImageName 'WebAppImg' |

Add-AzureEndpoint -Name web -LocalPort 80 -PublicPort 80 -Protocol tcp -LBSetName webport -ProbePath '/' -ProbeProtocol http -ProbePort 80 |

Add-AzureProvisioningConfig -Windows -AdminUserName $adminUser -Password $adminPassword

$iisvm4 = New-AzureVMConfig -Name 'iisvm4' -InstanceSize Small -ImageName 'WebAppImg' |

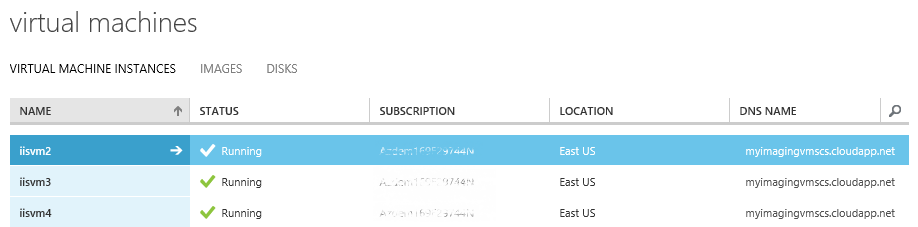
Add-AzureEndpoint -Name web -LocalPort 80 -PublicPort 80 -Protocol tcp -LBSetName webport -ProbePath '/' -ProbeProtocol http -ProbePort 80 |

Add-AzureProvisioningConfig -Windows -AdminUserName $adminUser -Password $adminPassword

New-AzureVM -ServiceName $cloudService -VMs $iisvm2, $iisvm3, $iisvm4

**Note:** Ensure to replace the subscription name and storage account name with the ones where the image was stored. Additionally, replace the service name you used in the **Migrating a Web Farm** demo and provide an admin username and password. If you created the virtual machine from scratch use a new service name of your preference, but in this case you may need to specify the **Location** parameter in the **New-AzureVM** command to the same location where the image resides.

1. The three virtual machines based on the image will be created. You can see them in the virtual machines view of the management portal.



## Application Deployment and Provisioning

### Exercise 4 - Deploying a Simple MVC4 Application - Manually

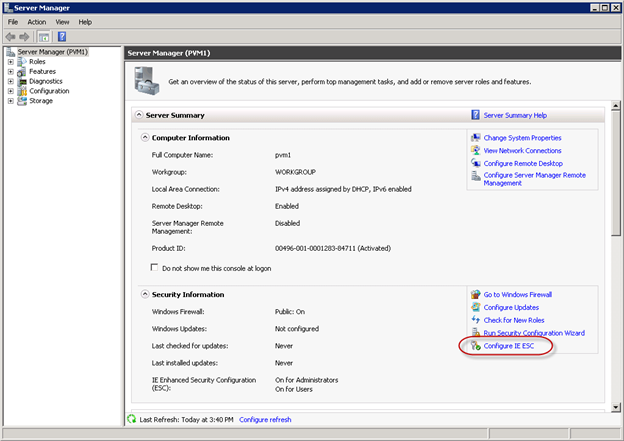
In this exercise, you will learn how to deploy a simple ASP.NET MVC4 application in the IIS of the Azure Virtual Machine you have previously configured.

**Note:** To make this solution highly available, you need to configure the SQL Servers in an availability set and set up SQL Server Mirroring between the instances.

#### Task 1 - Deploying a Simple MVC4 Application

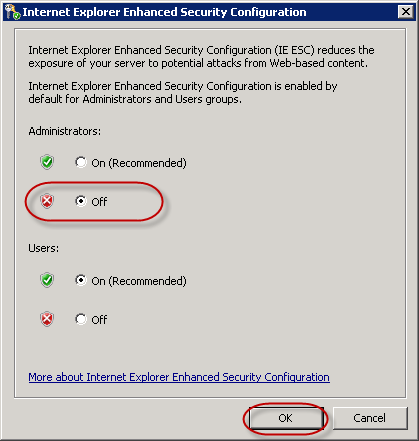
In this task, you will deploy the MVC4 application to the IIS Virtual Machines.

1. In the Azure Portal, Click **Virtual Machines** on the left menu.
2. You will see a list with your existing Virtual Machines. Select the first one you created in Exercise 1 and click **Connect**. If you used the proposed name, this Virtual Machine's should be named **iisvm1**.
3. You will be prompted to download the remote desktop client. Click **Open** and log on using the Admin credentials you defined when creating the Virtual Machine.
4. You need to install **.NET Framework 4.0** before deploying the MVC4 application. In order to do that, you will enable downloads from IE update **Internet Explorer Enhanced Security Configuration**.
   1. In the Azure Virtual Machine, open Server Manager from **Start | Administrative Tools**.
   2. In the **Server Manager**, click **Configure IE ESC** **within Security Information** **section**.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/internet-explorer-enhanced-security3.png?raw=true)

Internet Explorer Enhanced Security

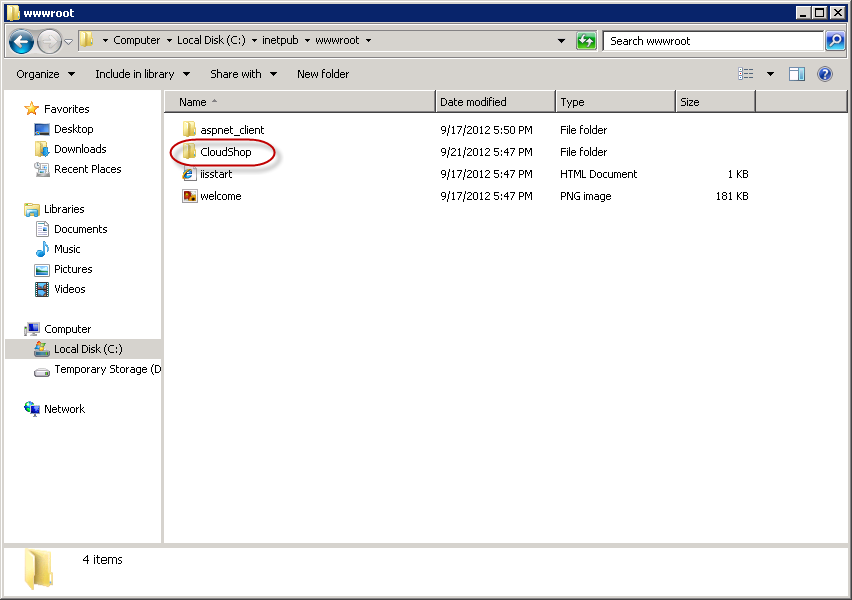
* 1. In the **Internet explorer Enhanced Security** dialog, turn **off** enhanced security for **Administrators** and click **OK**.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/Internet-Explorer-Enhanced-Security.png?raw=true)

Internet Explorer Enhanced Security

**Note:** Modifying Internet Explorer Enhanced Security configurations is not a good practice and it only for the purpose of this particular lab. The correct approach would be to download the files locally and then copy them to a shared folder or directly to the Virtual Machine.

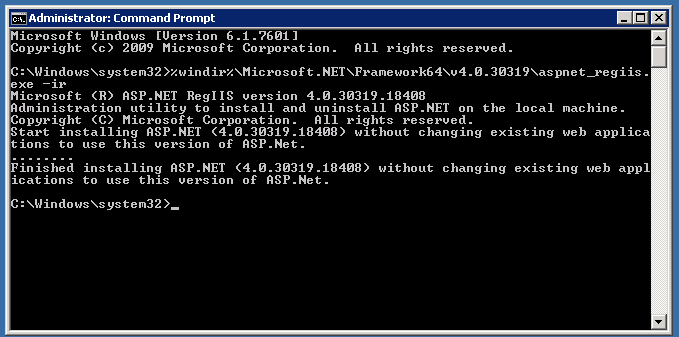
1. Now that you have permissions to download files, open an **Internet Explorer** browser session and navigate to <http://www.microsoft.com/en-us/download/details.aspx?id=30653>. Download and install **.NET Framework 4.5**.
2. Once **.Net Framework 4.5** installation finishes, open **wwwroot** folder located at **C:\inetpub\** and copy the file **7 - Hand On Lab - Dev Test StarterKit.zip** located in the **current** **folder** of this lab. To do this, copy **7 - Hand On Lab - Dev Test StarterKit.zip** (**Ctrl + C**) and paste it (**Ctrl + V**) in the Virtual Machine's **wwwroot** folder. Extract all files to **C:\inetpub\wwwroot\CloudShop** folder.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/wwwroot-folder.png?raw=true)

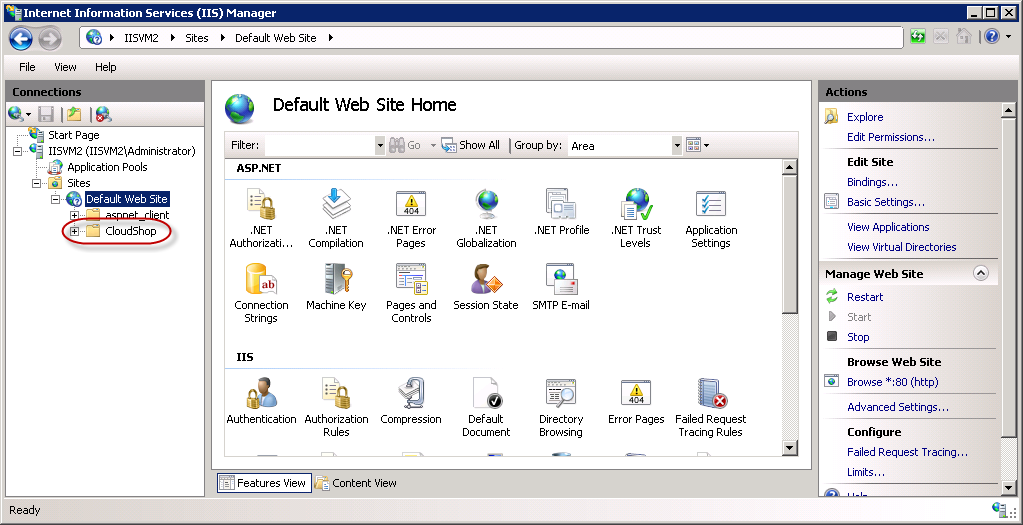
wwwroot folder

1. Open with **Notepad** the **Web.config** file located in **C:\inetpub\wwwroot\CloudShop**. Replace the connection strings placeholder with the name of your SQL Server (by default, is the Virtual Machine's name).
2. <connectionStrings>
3. <add name="AdventureWorksEntities" connectionString="metadata=res://\*/Models.AdventureWorks.csdl|res://\*/Models.AdventureWorks.ssdl|res://\*/Models.AdventureWorks.msl;provider=System.Data.SqlClient;provider connection string=&quot;data source=[ENTER YOUR SQL SERVER NAME];initial catalog=AdventureWorks2012;Uid=CloudShop;Password=Azure$123;multipleactiveresultsets=True;App=EntityFramework&quot;" providerName="System.Data.EntityClient" />
4. <add name="DefaultConnection" connectionString="Data Source=[ENTER YOUR SQL SERVER NAME];initial catalog=AdventureWorks2012;Uid=CloudShop;Password=Azure$123;MultipleActiveResultSets=True" providerName="System.Data.SqlClient" />
5. </connectionStrings>
6. Open the **Internet Information Services (IIS) Manager** from **Start | Administrative Tools**.
7. Run the following Command As Administrator

%windir%\Microsoft.NET\Framework64\v4.0.30319\aspnet\_regiis.exe –ir

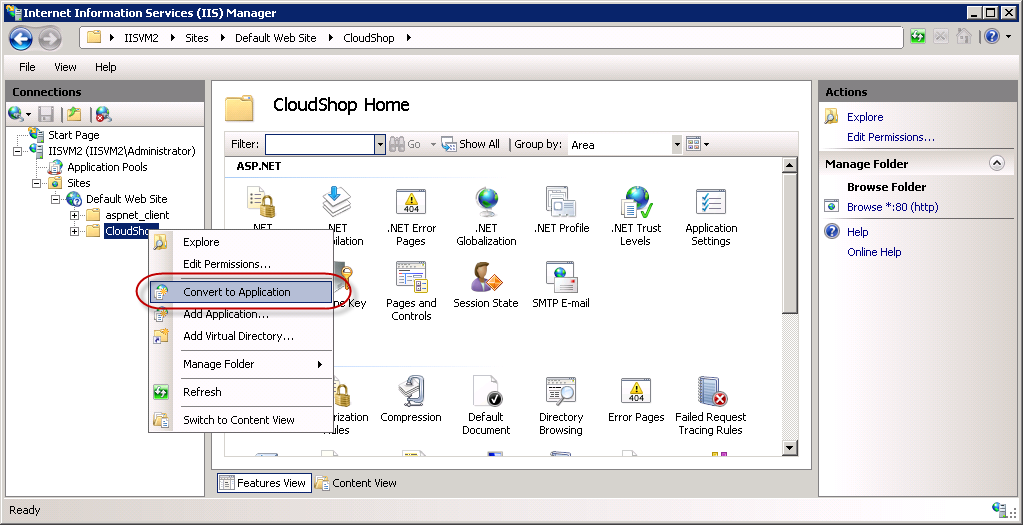


1. In the **Connections** pane, expand **Default Web Site** within your IIS Server's node. You will see the **CloudShop** folder you copied in the **wwwroot** folder.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/IIS-Manager.png?raw=true)

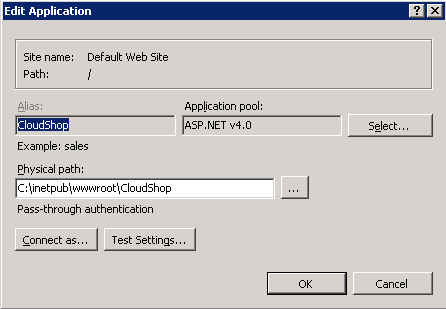
IIS Manager

1. Right-click **CloudShop** folder and select **Convert to Application**.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/iis-manager-convert-to-application.png?raw=true)

IIS Manager - Convert to Application

1. In the **Add Application** dialog, Change the Application Pool to **ASP.Net v4.0** **OK**.



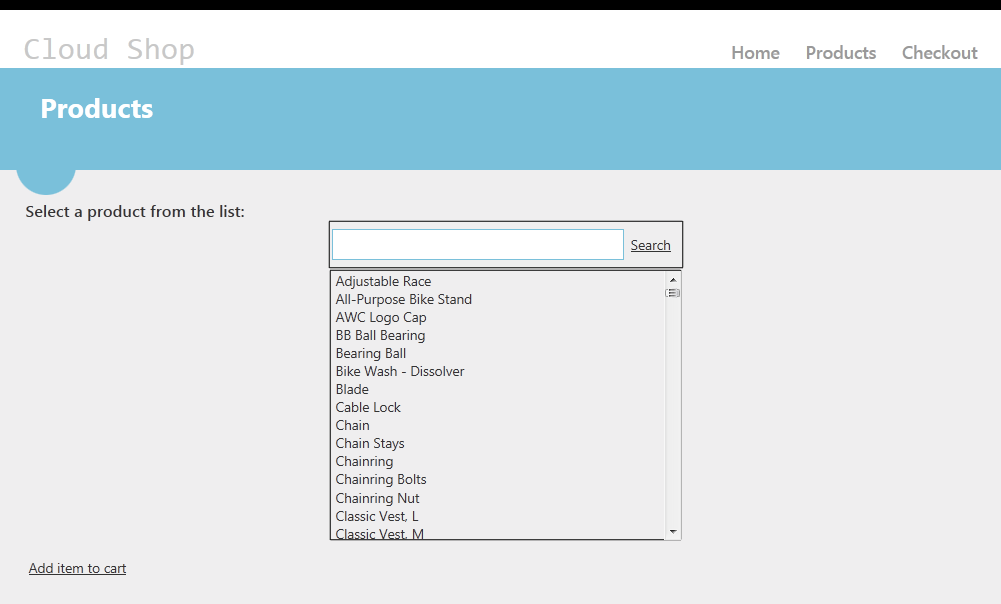
Add Application dialog

1. Close the **Internet Information Server (IIS) Manager** window.
2. Close the **Remote Desktop Connection**.
3. Repeat this task in the second Virtual Machine you created in **Exercise 1 -Task 1**. If you used the proposed name, this Virtual Machine should be named **iisvm2**.

#### Verification

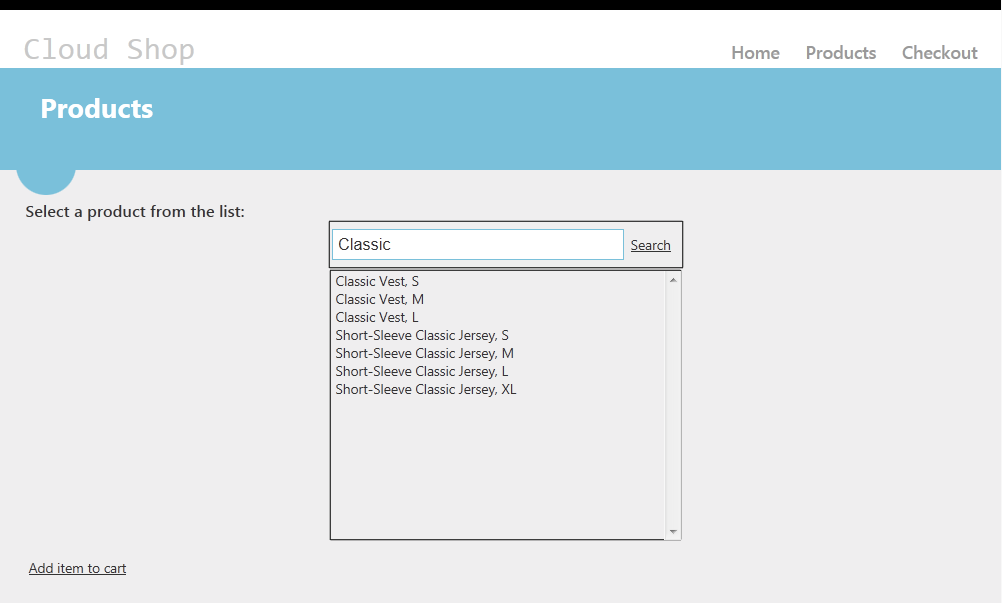
In this task, you will test the Cloud Shop MVC4 application you deployed in the previous task.

1. In your local machine, open **Internet Explorer**.
2. Go to http://[**YOUR-SERVICE-NAME**].cloudapp.net/CloudShop. The Service Name is the one you used when creating the IIS Virtual Machines (you can also check it in the Azure Portal, within Virtual Machine's dashboard).

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/MVC4-Application-running-in-the-Web-Farm.png?raw=true)

MVC4 Application running in the Web Farm

1. In the **Search** box, write Classic and click **Search**. It will show all the products that have a product name that match the search criteria.

[](https://github.com/WindowsAzure-TrainingKit/HOL-IntroToWindowsAzureVirtualMachines/blob/master/Images/searching-products2.png?raw=true)

Searching Products

### Exercise 5 – Automate Application Deployment to Virtual Machines with WebDeploy, PowerShell and Visual Studio

1. Requirements for this Exercise: [Visual Studio 2013 Update 2](http://go.microsoft.com/fwlink/p/?LinkId=393707) or higher.

##### *Web Deploy Overview*

Web Deploy (msdeploy) simplifies deployment of Web applications and Web sites to IIS servers. Administrators can use Web Deploy to synchronize IIS servers or to migrate to newer versions of IIS. Web Deploy Tool also enables administrators and delegated users to use IIS Manager to deploy ASP.NET and PHP applications to an IIS server.

Web Deploy empowers Visual Studio and WebMatrix to help developers streamline the deployment of Web applications to Microsoft IIS Web servers or to Microsoft Azure Websites. In addition, integration with Web Platform Installer allows developers to simply and easily install community web applications.

Web Deploy enables you to package your Web application content, configuration, databases and any other artifacts like registry, GAC assemblies etc., which can be used for storage or redeployment. If the package needs to be redeployed to a different environment, configuration values within the package can be parameterized during deployment without requiring modifications to the packages themselves. Once created, these packages can be deployed using Web Deploy cmdline or IIS Manager without requiring administrative privileges.

#### New Applications - Automate Deployment to a new Virtual Machines with WebDeploy and Visual Studio

When you create a web application in Visual Studio, you can generate a Windows PowerShell script that you can use later to automate the publishing of your web site to Azure as an Azure Web Site or a virtual machine. You can edit and extend the Windows PowerShell script in the Visual Studio editor to suit your requirements, or integrate the script with existing build, test, and publishing scripts.

Using these scripts, you can provision customized versions (also known as dev and test environments) of your site for temporary use. For example, you might set up a particular version of your web site on an Azure virtual machine or on the staging slot on an Azure web site to run a test suite, reproduce a bug, test a bug fix, trial a proposed change, or set up a custom environment for a demo or presentation. After you've created a script that publishes your project, you can recreate identical environments by re-running the script as needed, or run the script with your own build of your web application to create a custom environment for testing.

[**Scripts that Visual Studio generates**](javascript:void(0))

Visual Studio generates a solution-level folder called **PublishScripts** that contains two Windows PowerShell files, a publish script for your virtual machine or web site, and a module that contains functions that you can use in the scripts. Visual Studio also generates a file in the JSON format that specifies the details of the project you are deploying.

**JSON Configuration File**

The JSON file is created in the **Configurations** folder and contains configuration data that specifies exactly which resources to deploy to Azure. The name of the file that Visual Studio generates is project-name-WAWS-dev.json if you created an Azure Web Site, or project name-VM-dev.json if you created a virtual machine. Here's an example of a JSON configuration file that's generated when you create a web site. Most of the values are self-explanatory. The web site name is generated by Azure, so it might not match your project name.

{

    "environmentSettings": {

        "webSite": {

            "name": "WebApplication26632",

            "location": "West US"

        },

        "databases": [

            {

                "connectionStringName": "DefaultConnection",

                "databaseName": "WebApplication26632\_db",

                "serverName": "YourDatabaseServerName",

                "user": "sqluser2",

                "password": "",

                "edition": "",

                "size": "",

                "collation": "",

                "location": "West US"

            }

        ]

    }

}

When you create a virtual machine, the JSON configuration file looks like the following. Note that a cloud service is created as a container for the virtual machine. The virtual machine contains the usual endpoints for web access through HTTP and HTTPS, as well as endpoints for Web Deploy, which lets you publish to the web site from your local machine, Remote Desktop, and Windows PowerShell.

{

    "environmentSettings": {

        "cloudService": {

            "name": "myusernamevm1",

            "affinityGroup": "",

            "location": "West US",

            "virtualNetwork": "",

            "subnet": "",

            "availabilitySet": "",

            "virtualMachine": {

                "name": "myusernamevm1",

                "vhdImage": "a699494373c04fc0bc8f2bb1389d6106\_\_Win2K8R2SP1-Datacenter-201403.01-en.us-127GB.vhd",

                "size": "Small",

                "user": "vmuser1",

                "password": "",

                "enableWebDeployExtension": true,

                "endpoints": [

                    {

                        "name": "Http",

                        "protocol": "TCP",

                        "publicPort": "80",

                        "privatePort": "80"

                    },

                    {

                        "name": "Https",

                        "protocol": "TCP",

                        "publicPort": "443",

                        "privatePort": "443"

                    },

                    {

                        "name": "WebDeploy",

                        "protocol": "TCP",

                        "publicPort": "8172",

                        "privatePort": "8172"

                    },

                    {

                        "name": "Remote Desktop",

                        "protocol": "TCP",

                        "publicPort": "3389",

                        "privatePort": "3389"

                    },

                    {

                        "name": "Powershell",

                        "protocol": "TCP",

                        "publicPort": "5986",

                        "privatePort": "5986"

                    }

                ]

            }

        },

        "databases": [

            {

                "connectionStringName": "",

                "databaseName": "",

                "serverName": "",

                "user": "",

                "password": ""

            }

        ],

        "webDeployParameters": {

            "iisWebApplicationName": "Default Web Site"

        }

    }

}

You can edit the JSON configuration to change what happens when you run the publish scripts. The cloudService and virtualMachinesections are required, but you can delete the databases section if you don't need it. The properties that are empty in the default configuration file that Visual Studio generates are optional; those that have values in the default configuration file are required.

If you have a website that has multiple deployment environments (known as slots) instead of a single production site in Azure, you can include the slot name in the name of the web site in the JSON configuration file. For example, if you have a web site that's named **mysite** and a slot for it named**test** then the URI is mysite-test.cloudapp.net, but the correct name to use in the configuration file is mysite(test). You can only do this if the website and slots already exist in your subscription. If they don't exist, create the web site by running the script without specifying the slot, then create the slot in the Azure portal, and thereafter run the script with the modified web site name. For more information about deployment slots for web sites, see [Web site deployment slots](http://go.microsoft.com/fwlink/?LinkId=397853).

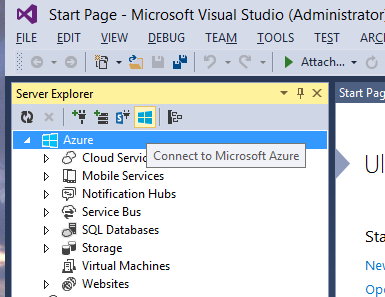
* 1. [More information](http://msdn.microsoft.com/en-us/library/azure/dn642480.aspx)

#### Task 1 - Existing Applications - Automate Deployment to an existing Virtual Machines with WebDeploy and Visual Studio

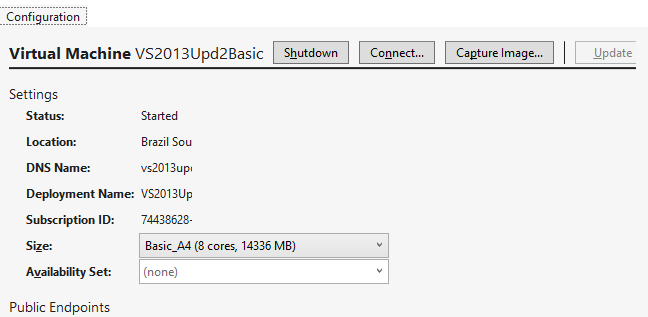
To Enable WebDeploy on an existing Virtual Machine, you can use the out-of-the-box Extension provided by Azure. Follow the steps below on Visual Studio 2013 Update 3 or higher to enable Web Deploy on existing Virtual Machines.

**Pre-Requisites:** The existing VM should have IIS Internet Information Server 7 or higher installed.

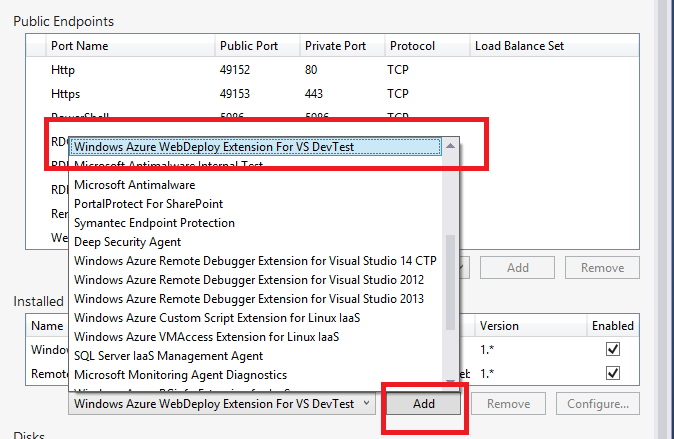
1. Open Visual Studio Server Explorer going to Top Menu | Server Explorer (Ctrl +W, L)
2. Connect to your Visual Studio Account



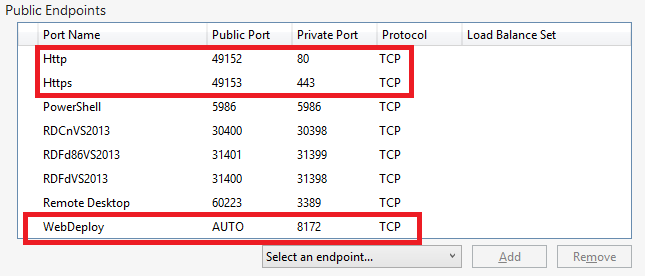
1. Expand the Virtual Machines option, select the virtual machine you want to enable Web Deploy
2. Then, right click on the VM | **Configure**
   1. You should see the screen below



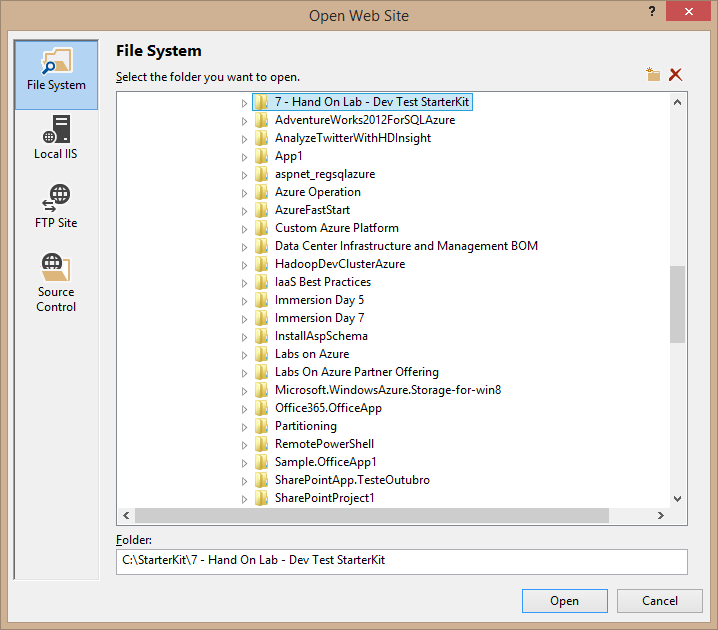
1. Scroll down to the section “Installed Extensions”, click on the “Add” Button, then select



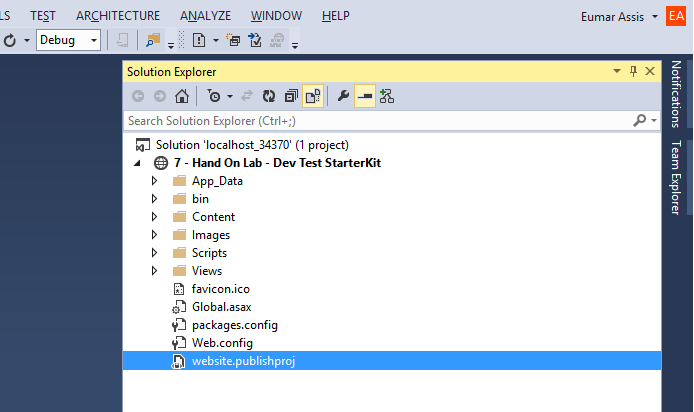
1. Next, in the Public Endpoints sections add “Http”, “Https” and “WebDeploy”



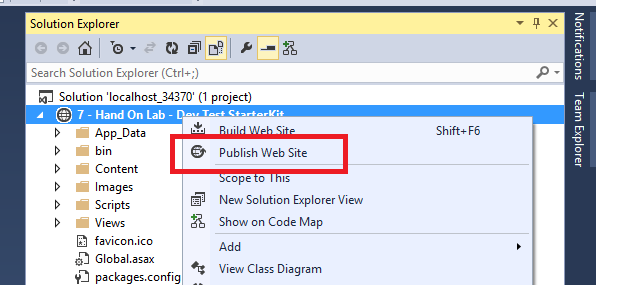
1. Unzip the file **7 - Hand On Lab - Dev Test StarterKit.zip** located in the **current** folder to C:/StarterKit/
2. On Visual Studio, go to **File | Open | Web Site**
3. Select **File System** and Point to Current Folder you created above C:/StarterKit/



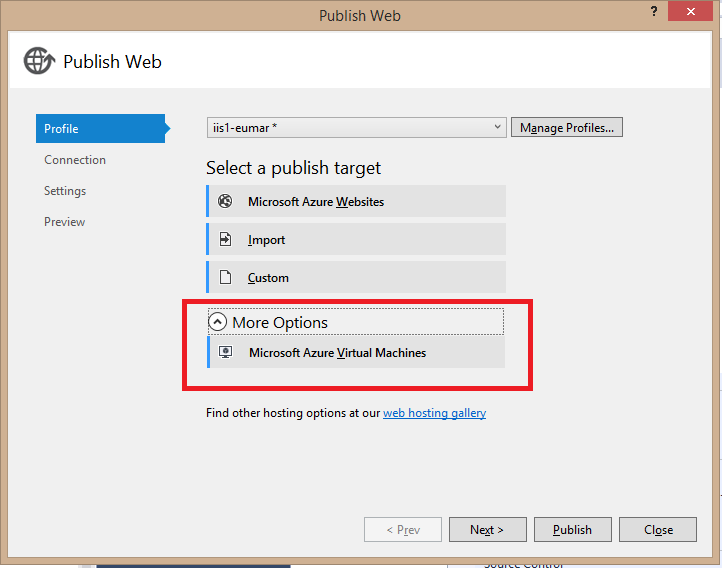
1. You should see the following screen



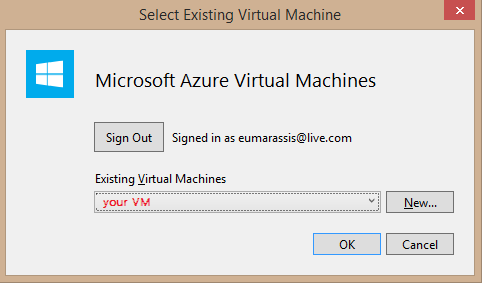
1. Right Click on the Project Name | Publish Web



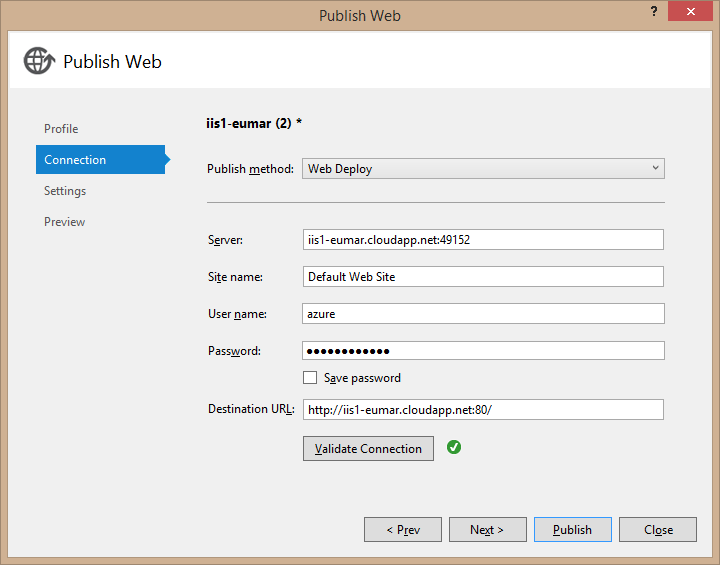
1. Select More Options | Microsoft Azure Virtual Machines



1. Sig in to Azure, then you should see the list of Virtual Machines available on your account, then select the Virtual Machine you have enabled Web Deploy Extension on step 4.



1. Fill out User Name and Password, and then click on **Validate Connection**.



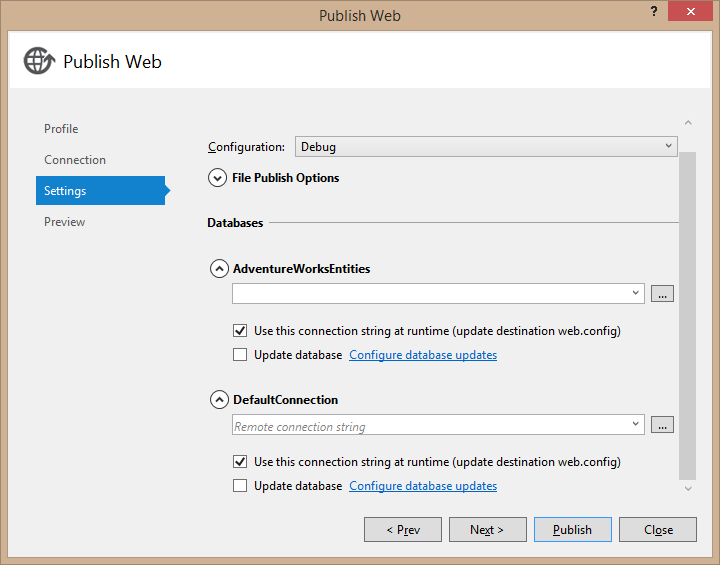
1. On the **Settings Page**, edit the ConnectionString of the **AdventureWorksEntities** and **AdventureWorks** databases to point to the SQL Server Database you have created previously on this demo.

<connectionStrings>

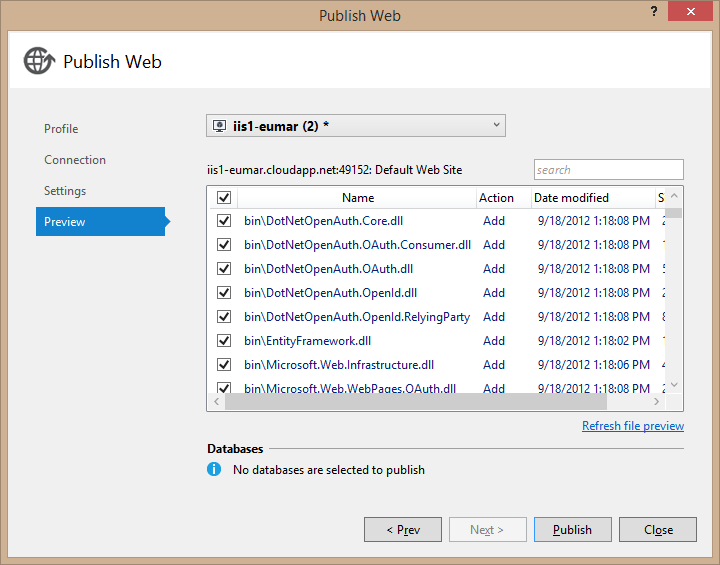
<add name="AdventureWorksEntities" connectionString="metadata=res://\*/Models.AdventureWorks.csdl|res://\*/Models.AdventureWorks.ssdl|res://\*/Models.AdventureWorks.msl;provider=System.Data.SqlClient;provider connection string=&quot;data source=[ENTER YOUR SQL SERVER NAME];initial catalog=AdventureWorks2012;Uid=CloudShop;Password=Azure$123;multipleactiveresultsets=True;App=EntityFramework&quot;" providerName="System.Data.EntityClient" />

<add name="DefaultConnection" connectionString="Data Source=[ENTER YOUR SQL SERVER NAME];initial catalog=AdventureWorks2012;Uid=CloudShop;Password=Azure$123;MultipleActiveResultSets=True" providerName="System.Data.SqlClient" />

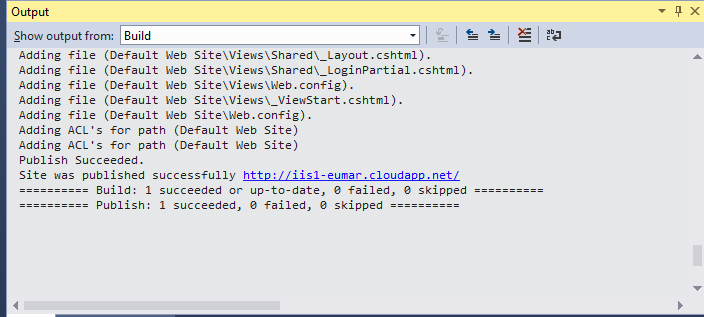
</connectionStrings>

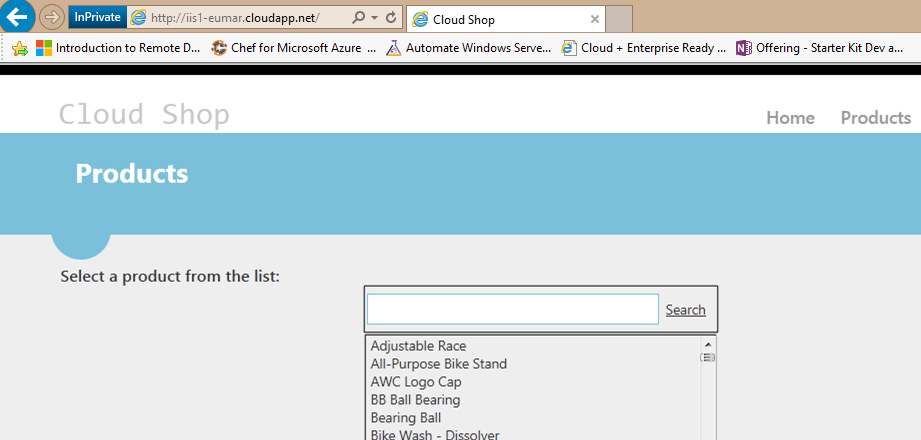


1. Finally, click publish to deploy your application to the Azure Virtual Machine and generate the

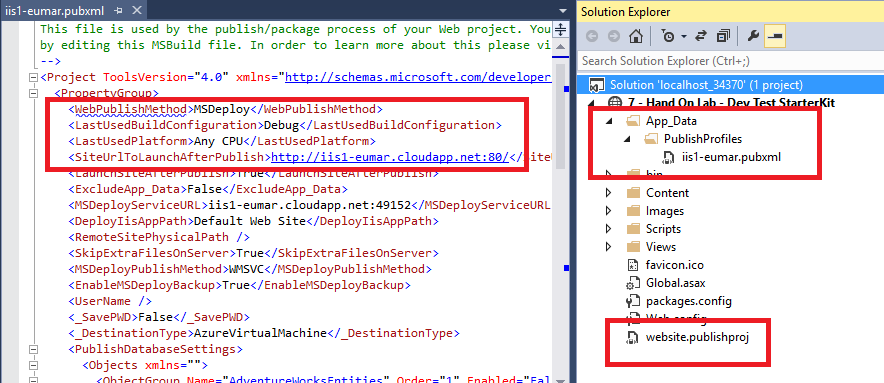


1. Once successfully published, you should see the result below on Visual Studio and a browser window would be opened loaded with your Web Application.



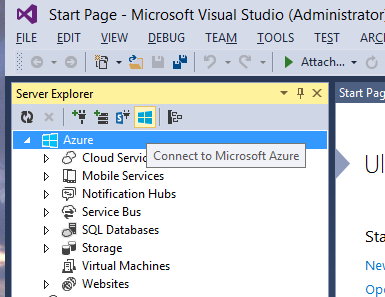


1. The Publish wizard will generate the \***.publishproj** and **App\_Date/\*.pubxml** files that you can you later use to automate your deployment with Visual Studio Team Foundation Server Build or Visual Studio Online Build

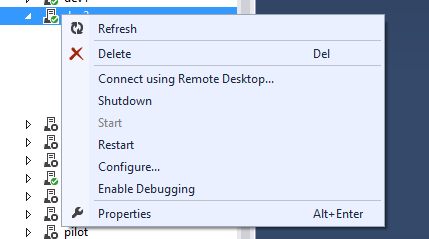


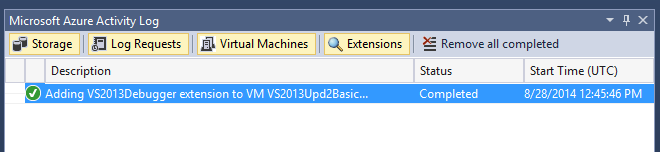
#### Task 2 - Enabling Remote Debugging

1. By using Visual Studio Server Explorer you can easily enable Visual Studio Remote Tools on Azure Virtual Machines:
2. Open Visual Studio Server Explorer going to Top Menu | Server Explorer (Ctrl +W, L)
3. Connect to your Visual Studio Account



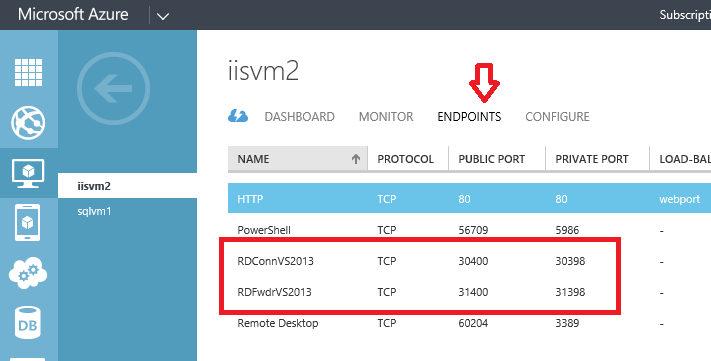
1. Expand the Virtual Machines option, select the virtual machine you want to enable Remote Debug
2. Then, right click on the VM | **Enable Debugging**



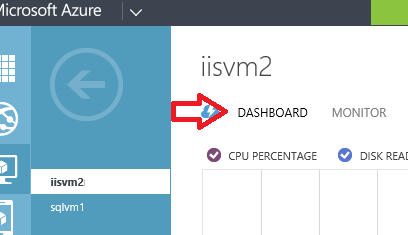


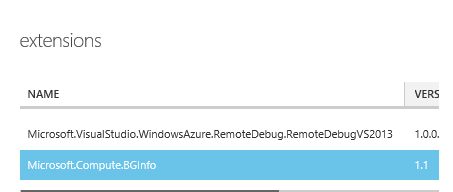
1. Go to the Azure Management Portal | Virtual Machines, then select the Virtual Machine you have just enabled Remote Debugging.
   1. As result, Visual Studio has installed the following Extensions and EndPoint on the Virtual Machine

**End Points:**

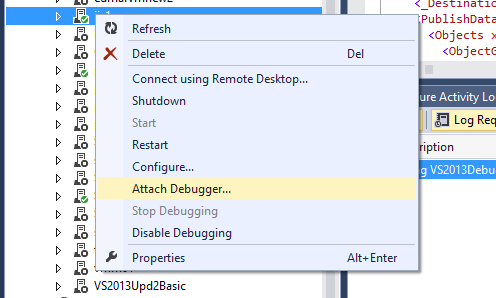


Turn the Virtual Machine on to see the extensions installed, as below

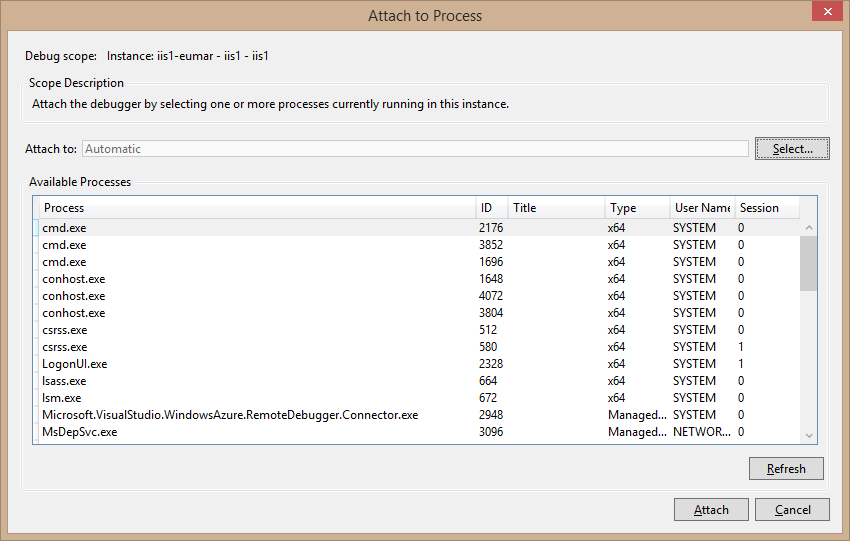
.



1. Go back to the Virtual Machines list on Visual Studio; right click on **Attach Debugger…**



1. You should see the next screen with the list of processes on the target Virtual Machine that you can attach and start debugging your application



## Summary

In this hands-on Lab, you have learnt how to deploy a simple ASP.NET MVC 4 Web application to a Web server hosted in Microsoft Azure, using SQL Server and configuring load balancing. Finally, you automated the deployment with PowerShell, WebDeploy and Visual Studio.

## Additional References:

* [Using Windows PowerShell Scripts to Publish to Dev and Test Environments](http://msdn.microsoft.com/en-us/library/azure/dn642480.aspx#BK_CustomBuild)
* [Debug programs that run on Azure virtual machines](http://msdn.microsoft.com/en-us/library/azure/ff683670.aspx)
* [Imaging Virtual Machines](https://github.com/Azure-Readiness/MicrosoftAzureTrainingKit/blob/master/Demos/Demo-ImagingVMs/Demo.md)
* [Web Deploy - Overview](http://www.iis.net/downloads/microsoft/web-deploy)
* [Install Web Deploy on Windows Manually](http://www.iis.net/learn/publish/using-web-deploy/configure-the-web-deployment-handler)