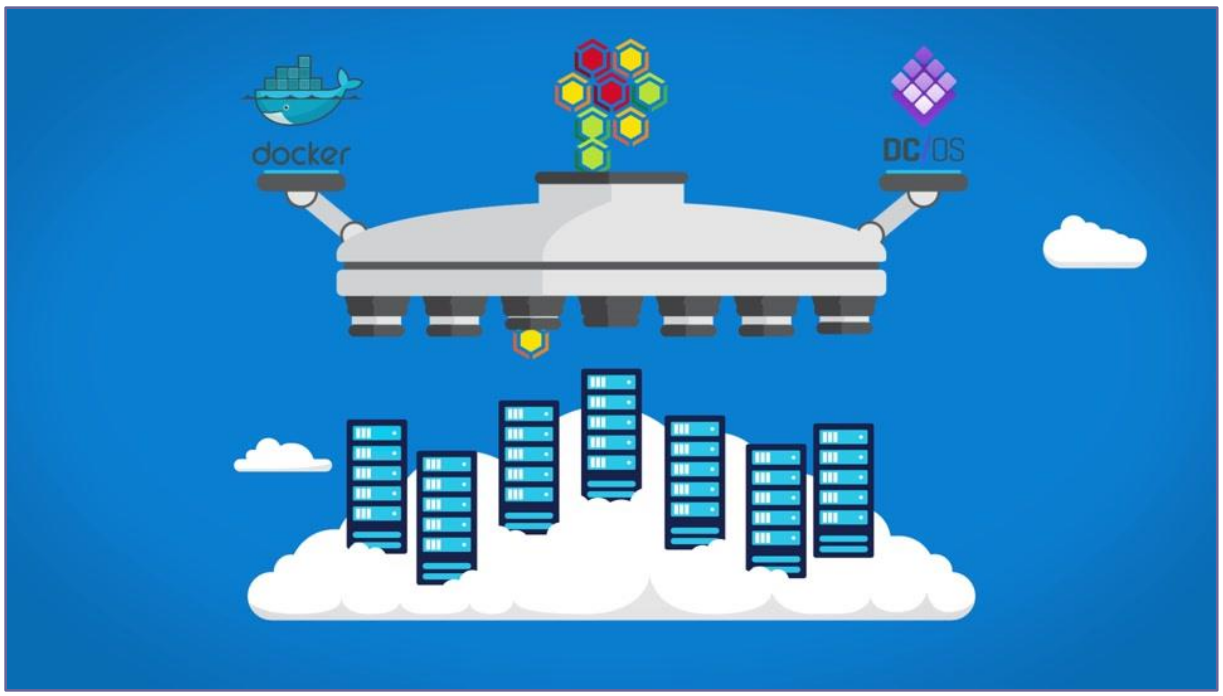



# Azure Developer Series

## Migrating a legacy ASP.NET 2-tier application to Azure using Container Services

Hands-On-Labs step-by-step guides

Prepared by:

Peter De Tender

CEO and Lead Technical Trainer
PDTIT and 007FFFLearning.com

@pdtit @007FFFLearning

Version: April 2019 – 2.0

Information in this document, including URL and other Internet Web site references, is subject to change without notice. Unless otherwise noted, the example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, e-mail address, logo, person, place or event is intended or should be inferred. 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 names of manufacturers, products, or URLs are provided for informational purposes only and Microsoft makes no representations and warranties, either expressed, implied, or statutory, regarding these manufacturers or the use of the products with any Microsoft technologies. The inclusion of a manufacturer or product does not imply endorsement of Microsoft of the manufacturer or product. Links may be provided to third party sites. Such sites are not under the control of Microsoft and Microsoft is not responsible for the contents of any linked site or any link contained in a linked site, or any changes or updates to such sites. Microsoft is not responsible for webcasting or any other form of transmission received from any linked site. Microsoft is providing these links to you only as a convenience, and the inclusion of any link does not imply endorsement of Microsoft of the site or the products contained therein.

© 2018 Microsoft Corporation. All rights reserved.

Microsoft and the trademarks listed at https://www.microsoft.com/en-us/legal/intellectualproperty/Trademarks/Usage/General.aspx are trademarks of the Microsoft group of companies. All other trademarks are property of their respective owners.

# Contents

Abstract and Learning Objectives ..... 4
Hands-On-Lab Scenario ..... 5
Requirements ..... 5
Naming Conventions: ..... 5
Azure Subscription: ..... 5
Other requirements: ..... 5
Alternative Approach: ..... 6
Final Remarks: ..... 6
Lab 1: Preparing your Hands-On-Lab environment ..... 7
What you will learn ..... 7
Time estimate ..... 7
Task 1: Deploying the lab-jumpVM Virtual Machine using Azure Portal Template deployment ..... 7
Task 2: Deploying the baseline Virtual Machine environment using an ARM-template from within Visual Studio 2017 ..... 15
Summary ..... 26

# Migrating a legacy ASP.NET 2-tiered application to Azure using Container Services - Hands-On-Labs step-by-step

## Abstract and Learning Objectives

This workshop enables anyone to learn, understand and build a Proof of Concept, in performing a multi-tiered legacy ASP.NET web application using Microsoft SQL Server database, platform migration to Azure public cloud, leveraging on different Azure Platform Azure A Service (PaaS) and Azure Container Services.

After an introductory module on cloud app migration strategies and patterns, students get introduced to the basics of automating Azure resources deployments using Visual Studio and Azure Resource Manager (ARM) templates. Next, attendees will learn about Microsoft SQL database migration to SQL Azure PaaS, as well as deploying and migrating Azure Web Apps.

After these foundational platform components, the workshop will totally focus on the core concepts and advantages of using containers for running web apps, based on Docker, Azure Container Registry (ACR), Azure Container Instance (ACI), as well as how to enable container cloud-scale using Azure Container Services (ACS) with Kubernetes and Azure Kubernetes Service (AKS).

The focus of the workshop is having a Hands-On-Labs experience, by going through the following exercises and tasks:

- Deploying a 2-tier Azure Virtual Machine (Webserver and SQL database Server) using ARM-template automation with Visual Studio 2017;
- Migrating a legacy SQL 2012 database to Azure SQL PaaS (Lift & Shift);
- Migrating a legacy ASP.NET web application to Azure Web Apps (Lift & Shift);
- Containerizing a legacy ASP.NET web application using Docker;
- Running Azure Container Instance (ACI) from an Azure Container Registry (ACR) image;
- Deploy and run Azure Container Services (ACS) with Kubernetes;
- Deploy and run Azure Kubernetes Services (AKS);
- Managing and Monitoring Azure Container Services (ACS) and Azure Kubernetes Services (AKS);

## Hands-On-Lab Scenario

The baseline of the hands-on-lab scenario is starting from an 8-year-old legacy ASP.NET application, developed around 2010, currently offering a product catalog browsing web page, pulling the product catalog list from a legacy Microsoft SQL Server 2012 database, running on dedicated Windows Server 2012 R2 Virtual Machines. (This could be seen as a subset of a larger application landscape within your organization, think of manufacturing database information, HR solutions, e-commerce platforms,… and alike). Imagine this application is running in our on-premises datacenter, where you want to perform an "application digital migration" to Azure Public cloud. You will use several Azure cloud-native services and solutions, ranging from Virtual Machines, Platform Services and different Container Solutions on Azure.

## Requirements

### Naming Conventions:

IMPORTANT: Most Azure resources require unique names. Throughout these steps you will see the word **"[SUFFIX]"** as part of resource names. You should replace this with your initials, guaranteeing those resources get uniquely named.

### Azure Subscription:

Participants need a "pay-as-you-go", MSDN or other paid Azure subscription

a) In one of the Azure Container Services tasks, you are required to create an Azure AD Service Principal, wich typically requires an Azure subscription owner to log in to create this object. If you don't have the owner right in your Azure subscription, you could ask another person to execute this step for you.
b) The Azure subscription must allow you to run enough cores, used by the baseline Virtual Machines, but also later on in the tasks when deploying the Azure Container Services, where ACS agent and master machines are getting set up. If you follow the instructions as written out in the lab guide, you need 12 cores.
c) If you run this lab setup in your personal or corporate Azure payable subscription, using the configuration as described in the lab guide, the estimated Azure consumption costs for running the setups during the 2 days of the workshop is $20.

### Other requirements:

Participants need a local client machine, running a recent Operating System, allowing them to:

- browse to https://portal.azure.com from a most-recent browser;
- establish a secured Remote Desktop (RDP) session to a lab-jumpVM running Windows Server 2016;

## Alternative Approach:

Where the lab scenario assumes all exercises will be performed from within the lab-jumpVM, (since several tools will be installed on the lab-jumpVM or are already installed by default), participants could also execute (most, if not all…) steps from their local client machine.

The following tools are being used throughout the lab exercises:

- Visual Studio 2017 community edition (updated to latest version)
- Docker for Windows (updated to latest version)
- Azure CLI 2.0 (updated to latest version)
- Kubernetes CLI (updated to latest version)

Make sure you have these tools installed prior to the workshop, if you are not using the lab-jumpVM. You should also have full administrator rights on your machine to execute certain steps within using these tools.

## Final Remarks:

VERY IMPORTANT: You should be typing all of the commands as they appear in the guide, except where explicitly stated in this document. Do not try to copy and paste from Word to your command windows or other documents where you are instructed to enter information shown in this document. There can be issues with Copy and Paste from Word or PDF that result in errors, execution of instructions, or creation of file content.

IMPORTANT: Most Azure resources require unique names. Throughout these steps you will see the word "[SUFFIX]" as part of resource names. You should replace this with your initials, guaranteeing those resources get uniquely named.

# Lab 1: Preparing your Hands-On-Lab environment

## What you will learn

In this first lab, you prepare the baseline for executing all hands-on-labs exercises:

- Log on to your Azure subscription;
- Deploy the lab-jumpVM within your Azure subscription;
- Verify and install the required tools to run the lab exercises;
- Deploy the 2-tiered Azure Virtual Machine infrastructure (WebVM and SQLVM);

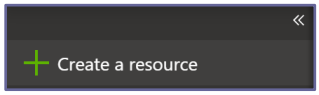
## Time estimate

This lab is estimated to take **60min**, assuming your Azure subscription is already available.

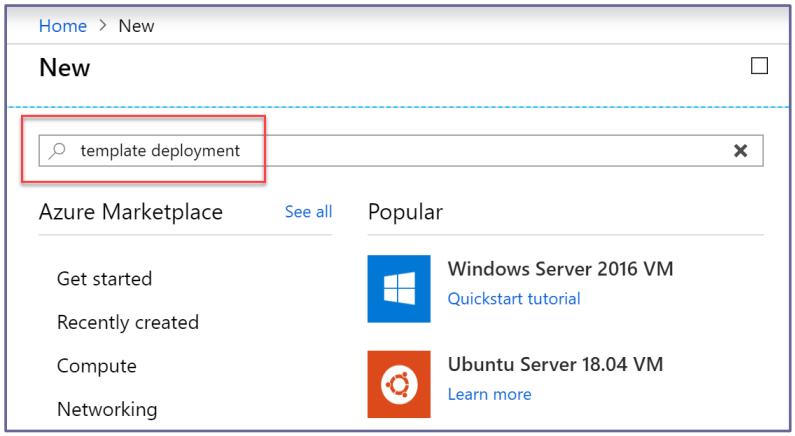
## Task 1: Deploying the lab-jumpVM Virtual Machine using Azure Portal Template deployment

In this task, you start from deploying the "lab-jumpVM" Virtual Machine in your Azure environment. This machine becomes the starting point for all future exercises, as it has most required tools already installed.

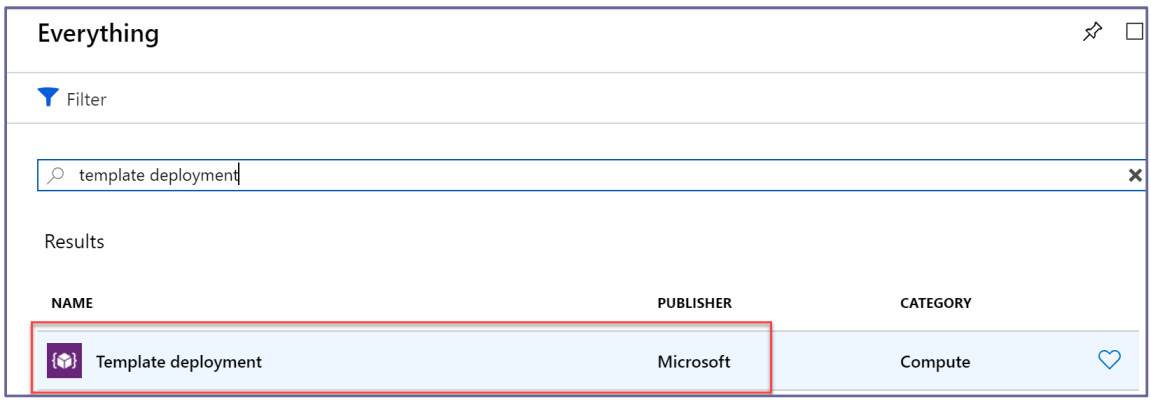
1. Once you are logged on to your Azure subscription, select **Create a Resource**



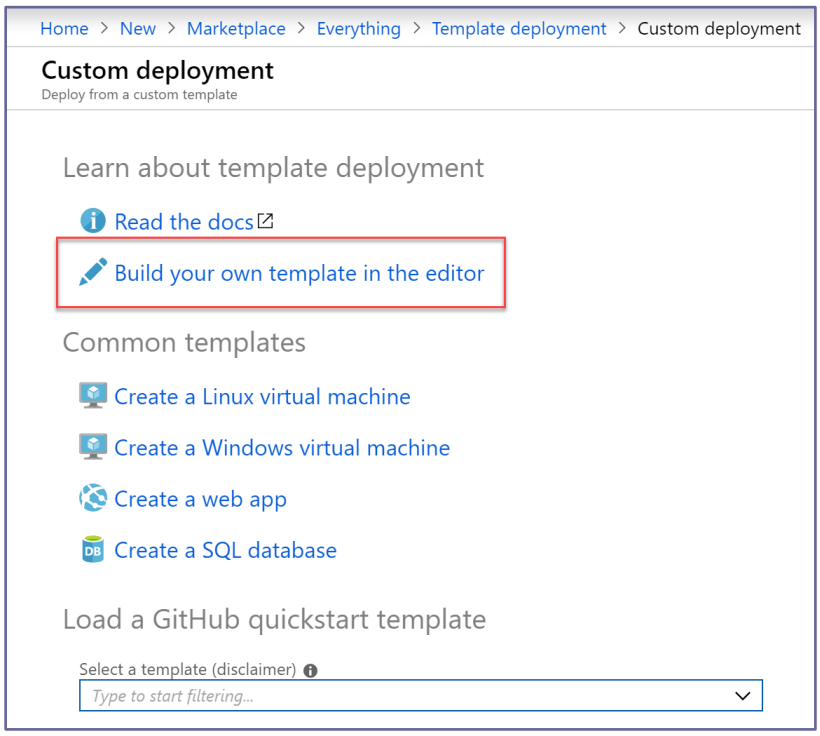

2. In the Search Azure Marketplace field, type "template deployment"



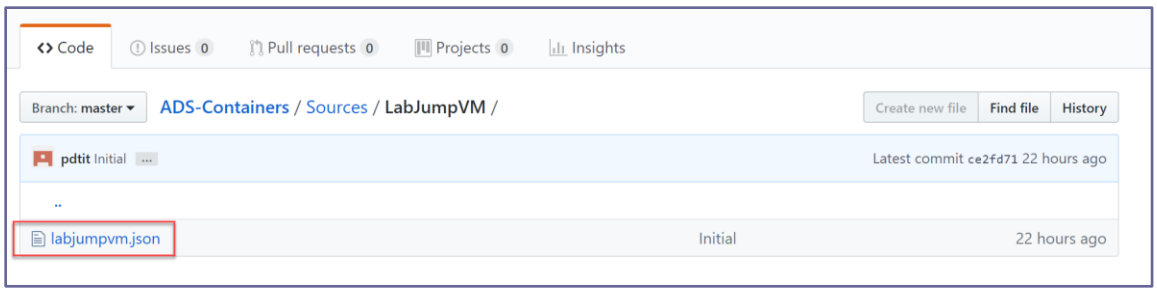

3. And select **template deployment** from the list of MarketPlace results. Followed by clicking the **Create** button down at the bottom.



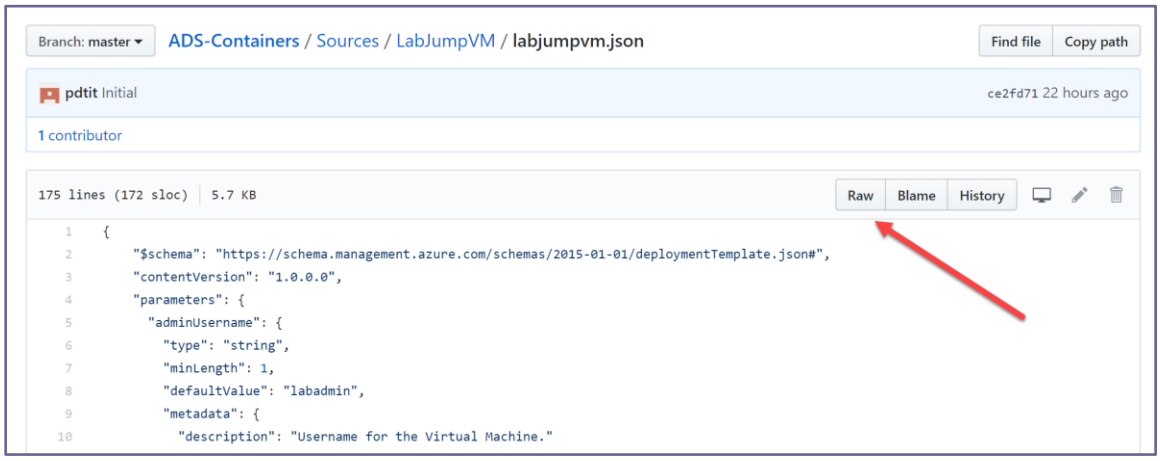

4. This opens the **Custom Deployment** blade. Here, **select "build your own template in the editor"**




5. First, from a **second tab** in your browser window, go to the following URL on GitHub, browsing the source files repository for this lab, specifically the LabjumpVM folder: https://github.com/007FFFLearning/ADS-Containers/tree/master/Sources/LabJumpVM




6. Select the labjumpvm.json object in there. This exposes the details of the actual JSON deployment file. Click the Raw button, to open the actual file in your browser.

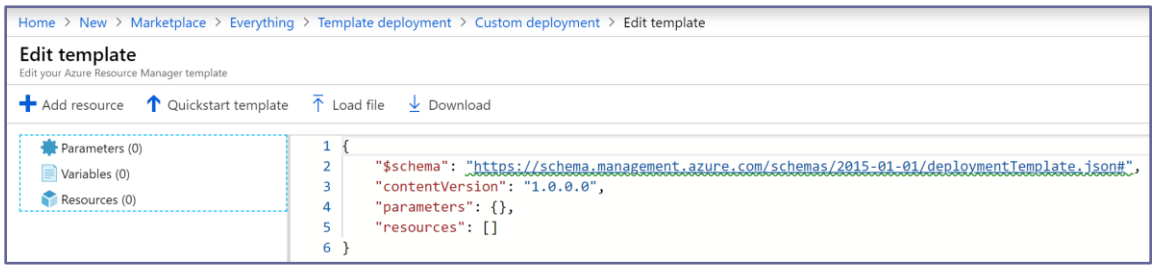



7. Your browser should show the content as follows:

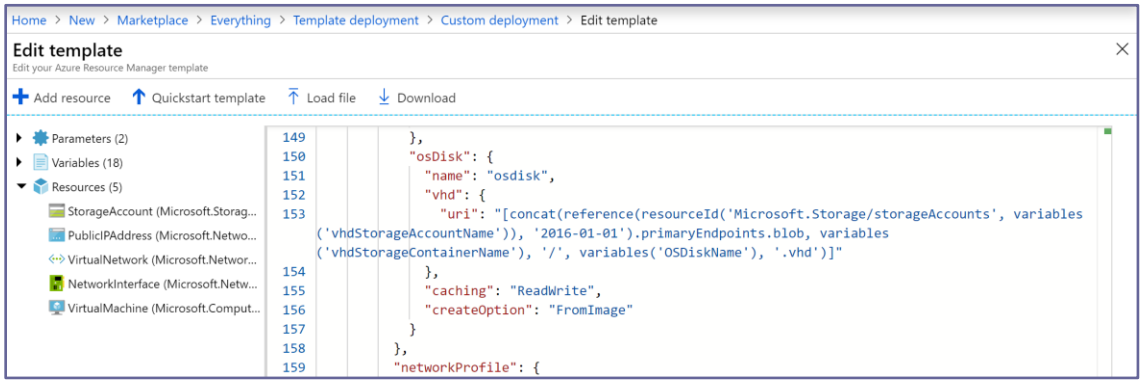
```
{
    "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
    "contentVersion": "1.0.0.0",
    "parameters": {
      "adminUsername": {
        "type": "string",
        "minLength": 1,
        "defaultValue": "labadmin",
        "metadata": {
          "description": "Username for the Virtual Machine."
        }
      },
      "adminPassword": {
        "type": "securestring",
        "defaultValue": "L@BadminPa55w.rd",
        "metadata": {
          "description": "Password for the Virtual Machine."
        }
      }
```

8. Here, use Ctrl+A to select all lines in the JSON file, and use Ctrl+C to copy it to the clipboard.

9. Go back to the Azure Portal; From the Edit Template blade, remove the first 6 lines of code you see in there, and paste in the JSON content from the clipboard.

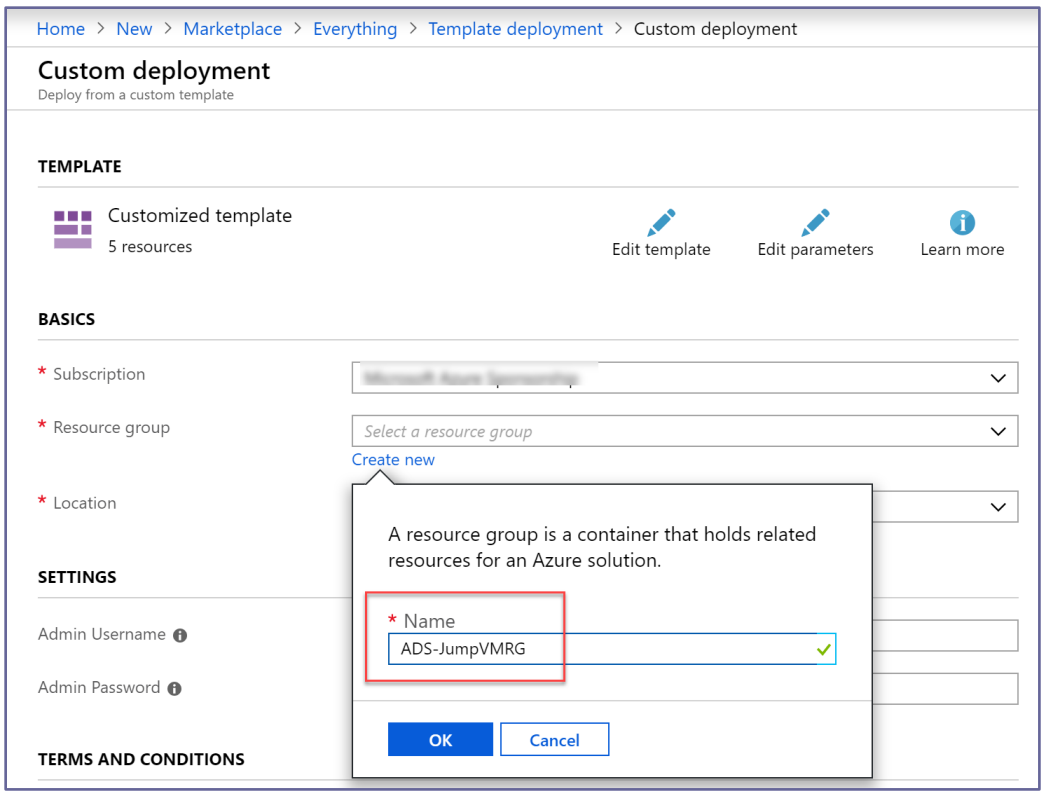



10. The Edit template blade should recognize the content of the JSON file, and showing the details in the JSON Outline on the left

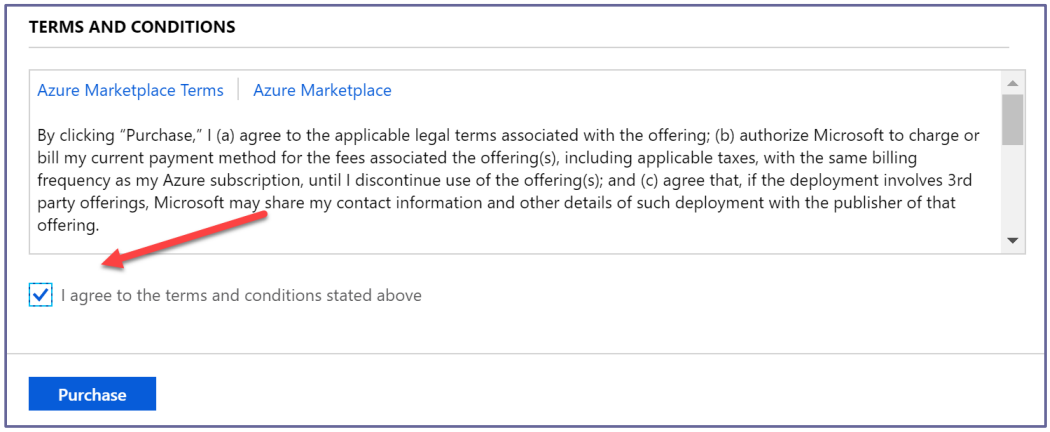



11. Press the Save button.

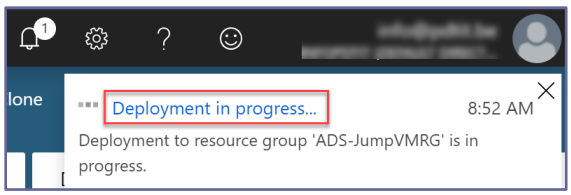
12. This redirects you back to the Custom deployment blade, from where you will execute the actual template deployment, filling in the required fields as follows:
    - Subscription: your Azure subscription
    - Resource Group: Create New / [SUFFIX]-JumpVMRG
    - Location: your closest by Azure Region
    - Admin Username: labadmin (this information is picked up from the ARM-template; although you could change this, we recommend you to not do so for consistency with the lab guide instructions)
    - Admin Password: L@BadminPa55w.rd (this information is picked up from the ARM-template; although you could change this, we recommend you to not do so for consistency with the lab guide instructions)



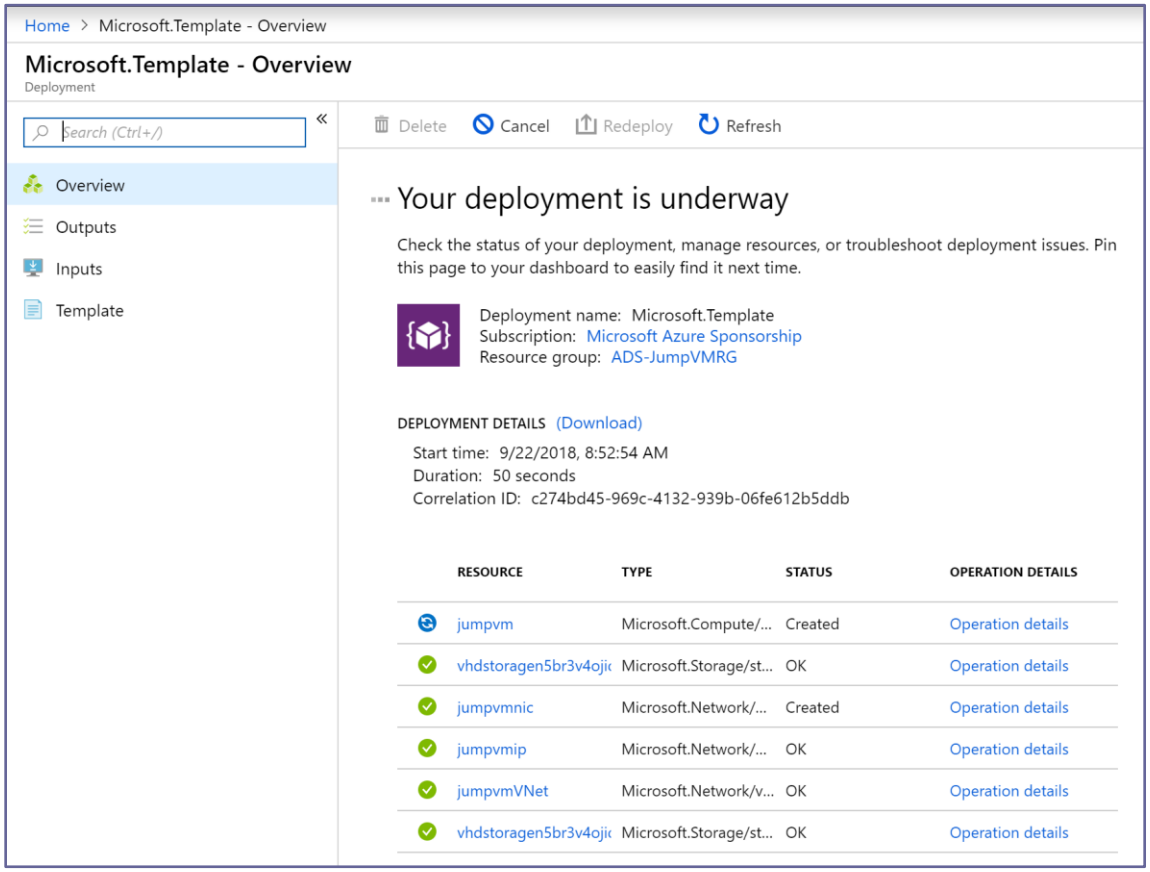

13. When all fields have been completed, scroll down in the blade. Under the terms and conditions section, **Check "I agree to the terms and conditions state above"**, and **press** the **Purchase** button.



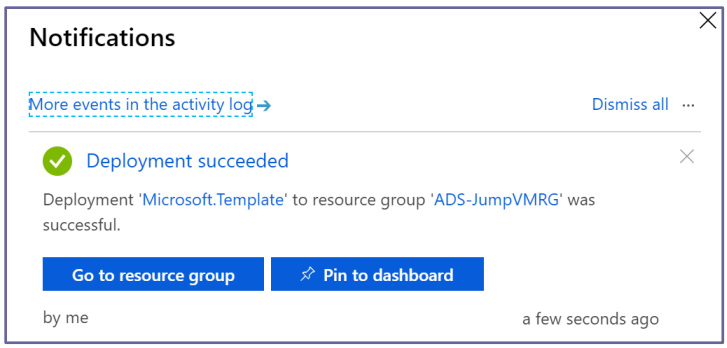

14. This sets off the actual Azure Resource deployment process. From the **notification area**, you can get update information about the deployment.



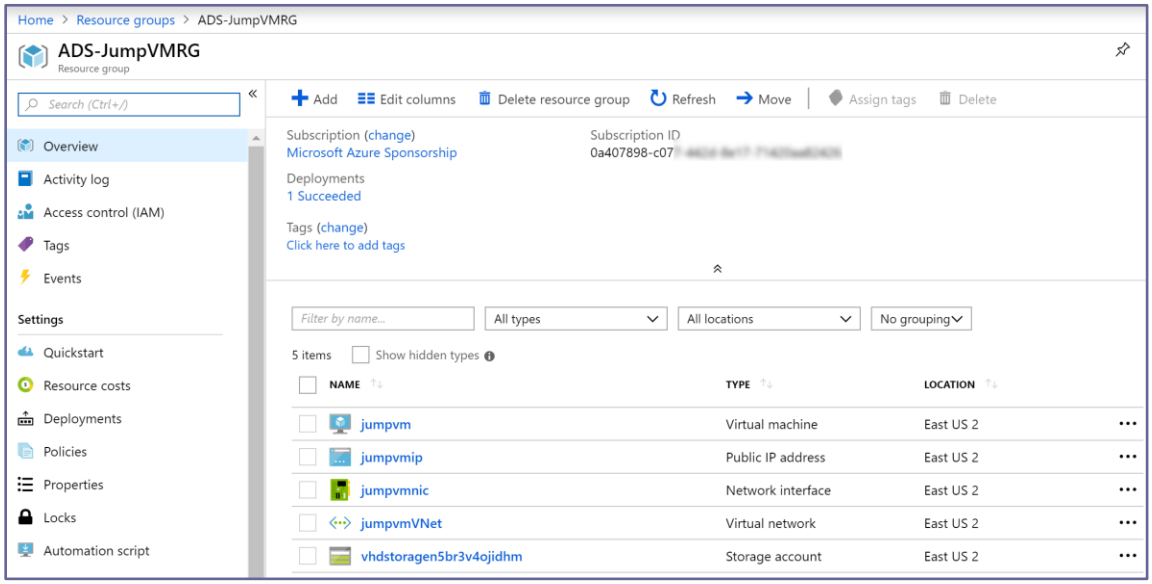

15. If you click the "…Deployment in progress…", you will get redirected to the Microsoft template Overview blade, showing you the details of each Azure Resource getting deployed.



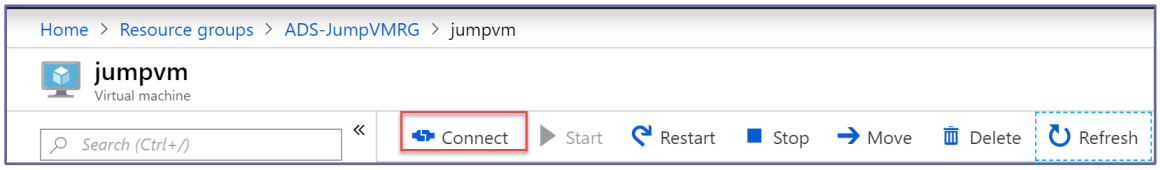

16. Wait for the deployment to complete successfully, which you can see from this detailed view, or from the notification area.



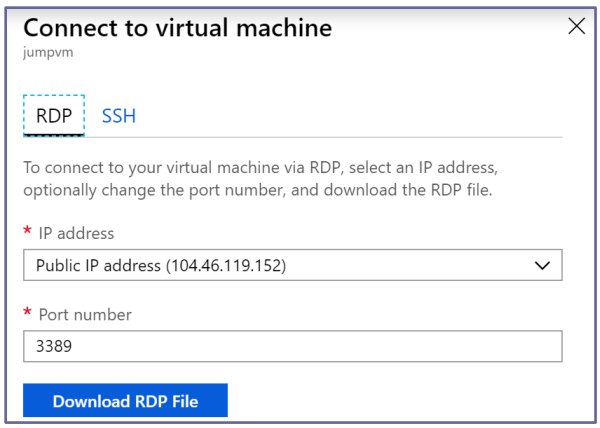

17. From the notification message, **click** "Go to resource group". (If you already closed the notification message, from the Azure Portal navigation menu to the left, select Resource Groups).



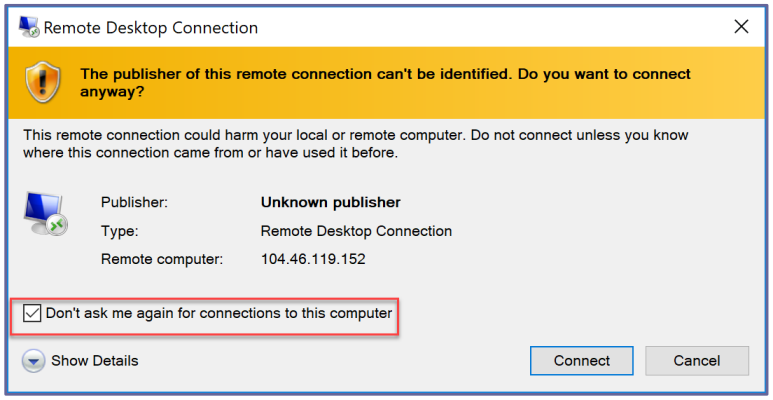

18. **Click** on the **jumpvm Azure Virtual Machine** resource. This redirects you to the detailed blade for the jumpvm resource. Here, **press** the **Connect** button.




19. From the **Connect to Virtual machine** blade, notice the **public IP-address and port 3389**. This allows you to establish an RDP session to the Azure VM. Do this by **clicking** the **Download RDP file** button.




20. After the RDP connection file has been **downloaded**, **open** it up, which will launch the **Remote Desktop Connection** to that VM. In the appearing popup window, set the flag to "Don't ask me again for connections to this computer.




21. Press the **Connect** button; when it is asking for your **VM machine admin credentials** in the next step, provide the **VM administrator name (labadmin) as well as its password**.

22. Your Remote Desktop session to this Azure VM gets established.

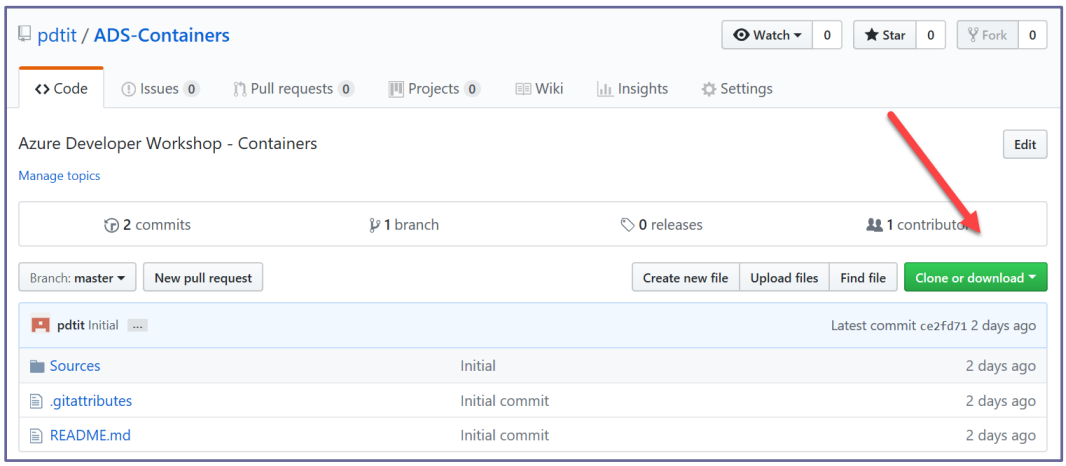
23. **Close** the appearing "**Server Manager**", you will access it again later.

This completes this task, in which you deployed a Windows 2016 lab-VM, by using Azure Resource Manager template-based deployment.

## Task 2: Deploying the baseline Virtual Machine environment using an ARM-template from within Visual Studio 2017

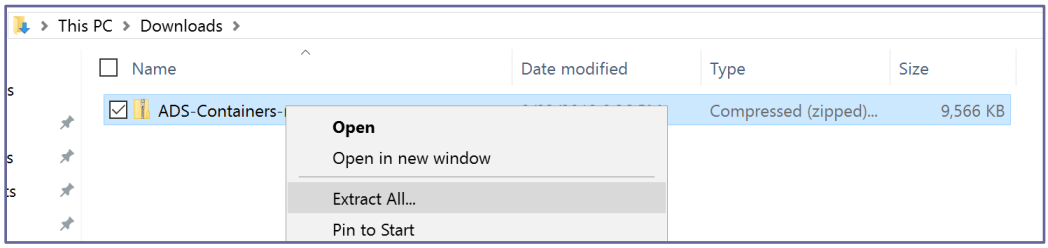
In this task, you run the ARM-template which deploys the baseline Virtual Machine environment you need in the next lab. Deployment will be performed from within Visual Studio 2017.

1. From within the lab-JumpVM Virtual Machine RDP-session, open a browser session to https://github.com/007FFFLearning/ADS-Containers/, and **click** the **green Clone or Download button**.

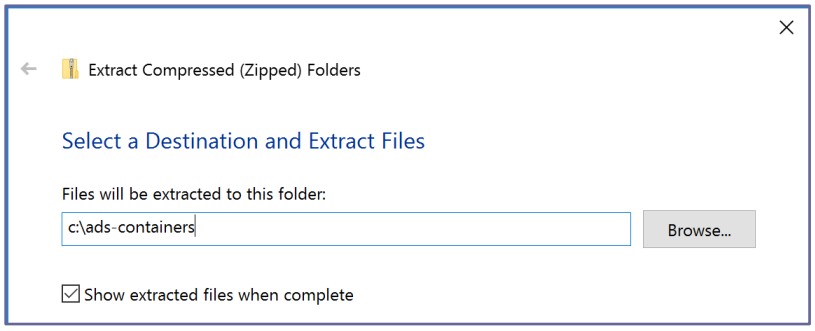



2. From the appearing popup, select **Download Zip.** This downloads the sources directory to the downloads folder on the lab-jumpVM.
**(note: if the Internet Explorer browser doesn't allow downloads, go into its settings / internet options / and enable File Downloads and Font Downloads), and restart the browser.**

3. Once downloaded, **open** the downloads folder from within **File Explorer**, **right-click** the ADS-Containers.zip downloaded file, **choose Extract all...**

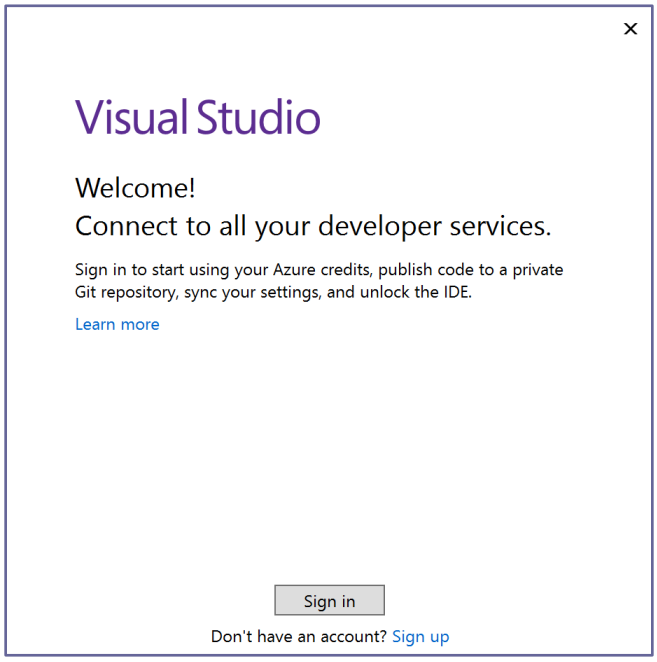


4. Extract the files to c:\ADS-Containers, or any other folder location of your choice.

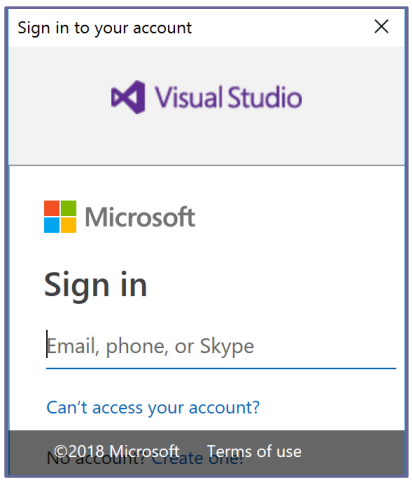



5. Once the extraction is completed successfully, **browse to the folder**. Within the folder, browse to **\ads-containers\ADS-Containers-master\Sources**. **Select** the compressed file "**ProductCat-VM-ARM-Deploy**". **Right-click** this file, **choose Extract all**, and extract its content in a folder of choice, for example **c:\ProductCat-VM-ARM-Deploy**.

6. From the **lab-JumpVM Start Menu** or the shortcut on the desktop, **open Visual Studio 2017.** Since this is the first time you launch Visual Studio after a fresh install, you are greeted with the Visual Studio welcome message.

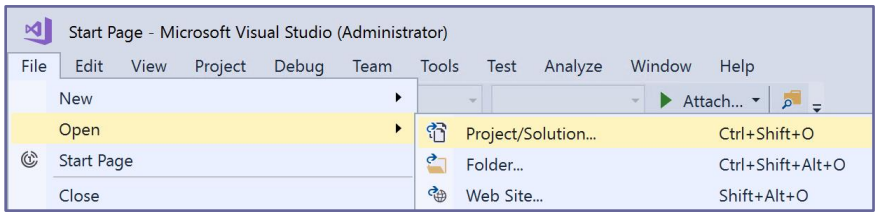



7. It asks you to sign in, so press that button. Here, authenticate with your **Azure subscription credentials.**

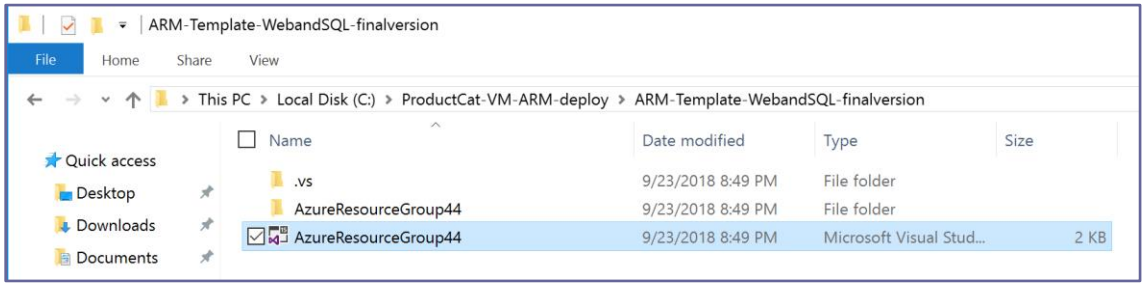



8. After successful authentication, you can choose a layout theme. Select a theme of choice, and wait until the Visual Studio environment completed loading.

9. From the Visual Studio menu, click File / Open / Project/Solution…

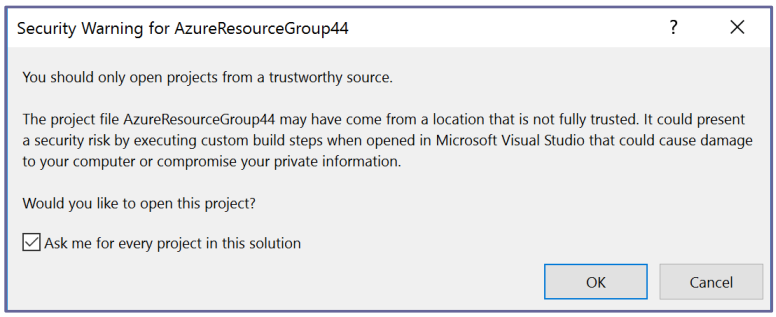



10. Browse to the folder where you extracted the "productCat-VM-ARM-Deploy" files. Click through the subfolders until you are at the folder showing **AzureResourceGroup44 Microsoft Visual Studio Project file type** (in our setup, this is c:\ProductCat-VM-ARM-deploy\ARM-Template-WebandSQL-finalversion)

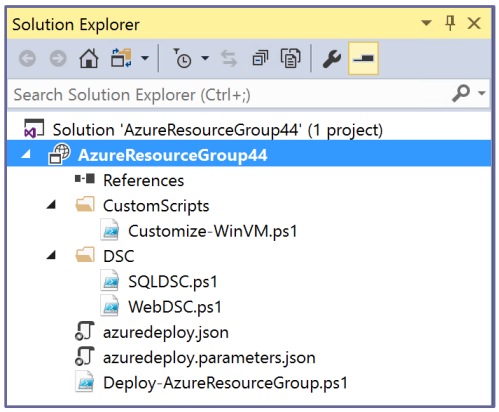



11. **Confirm** to open this project by pressing the **Open** button. This loads the project in Visual Studio.

12. Visual Studio will throw a security warning popup message; this is to warn you to only open Projects from trusted locations. Press the Ok button to continue.



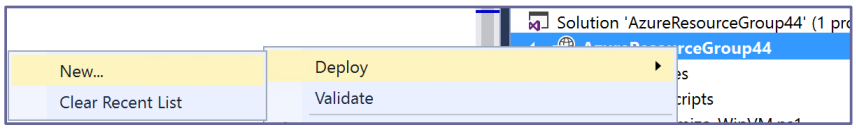

13. Once the project is opened in Visual Studio, you should have the Solution Explorer to the right of the screen, showing the actual deployment project folder and file structure.




14. In short, these files are doing the following:

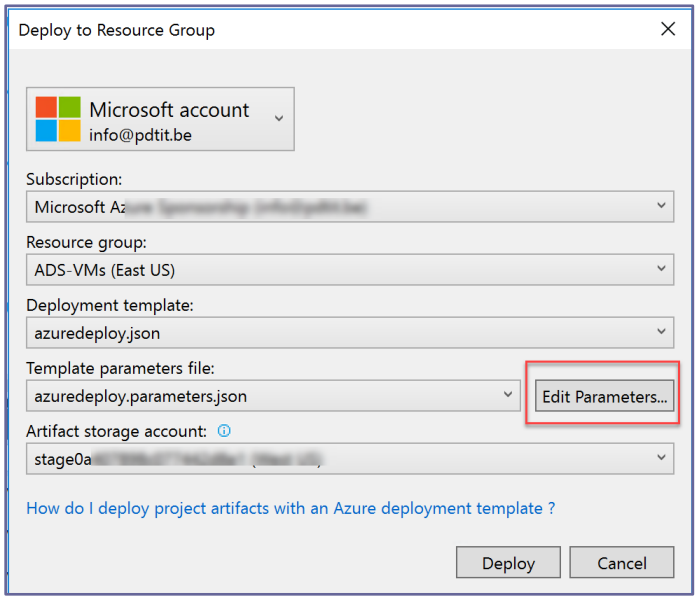
| File | Purpose |
|---|---|
| Azuredeploy.json | The actual ARM-template deployment file, which creates the different Azure Resources for both WebVM and SQLVM infrastructure. |
| Azuredeploy.parameters.json | The ARM-template parameters file |
| \CustomScripts\Customize-WinVM.ps1 | A PowerShell script, containing specific settings that get applied to both VMs using PowerShell |
| DSC\SQLDSC.ps1 | A PowerShell script that is used to customize the installation and configuration of SQL Server on the SQLVM |
| DSC\WebDSC.ps1 | A PowerShell script that is used to customize the installation and configuration of IIS Web Server on the WebVM |
| Deploy-AzureResourceGroup.ps1 | A PowerShell script that is used by VS2017 to run the actual deployment of the ARM-template |
| | |

15. From within the Solution Explorer window, select the **AzureResourceGroup44** project, **right-click** it and from the context menu, select **Deployment / New...**

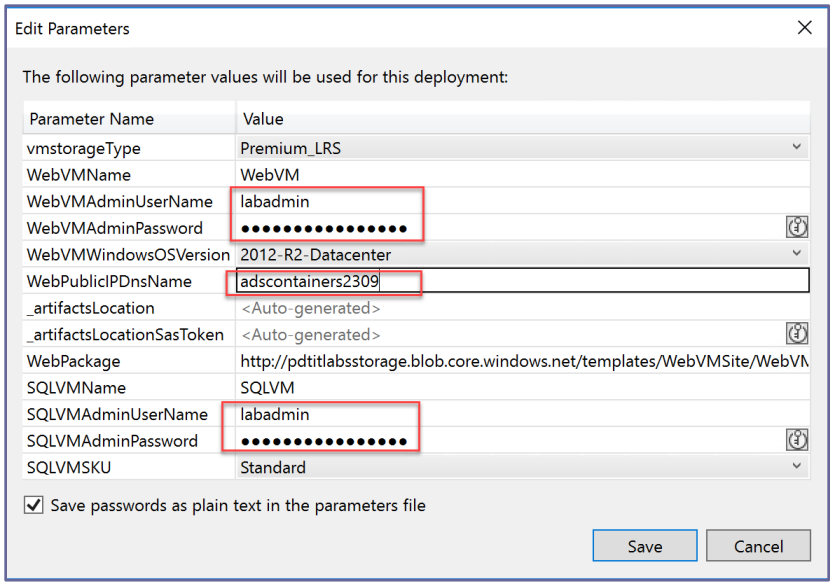



16. In the appearing "Deploy to Resource Group" popup, complete the following settings:

- Subscription: **Your Azure Subscription**
- Resource Group: **Create New / [SUFFIX]-VMs** – location = **closest by to your location**
- Deployment template: **azuredeploy.json**
- Template Parameters File: **azuredeploy.parameters.json**



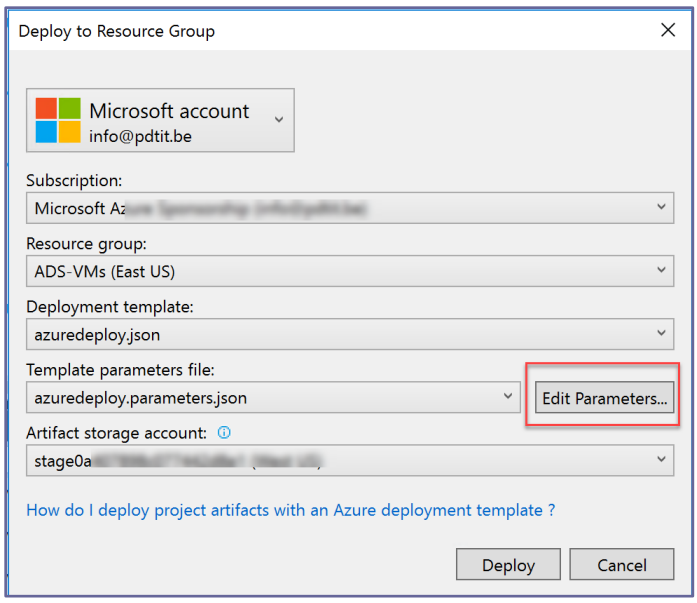

17. **Before pressing the Deploy** button, complete some additional deployment settings by **pressing the Edit Parameters button**:




- WebVMName: WebVM
- WebVMAdminUserName: labadmin
- WebVMAdminPassword: L@BadminPa55w.rd (do not alter this password, as otherwise the customization script later on won't work)
- WebVMWindowsOSVersion: 2012-R2-Datacenter
- WebPublicIPDnsName: [SUFFIX]containersDATE
- SQLVMName: SQLVM
- SQLVMAdminUsername: labadmin
- SQLVM WebVMAdminPassword: L@BadminPa55w.rd (do not alter this password, as otherwise the customization script later on won't work)

18. **Check** the "Save passwords as plain text in the parameters file". (Note: This is ok in this lab environment, but not recommended in production deployments. If this option is not checked, you will get a PowerShell window appearing, asking you for this administrator password there).

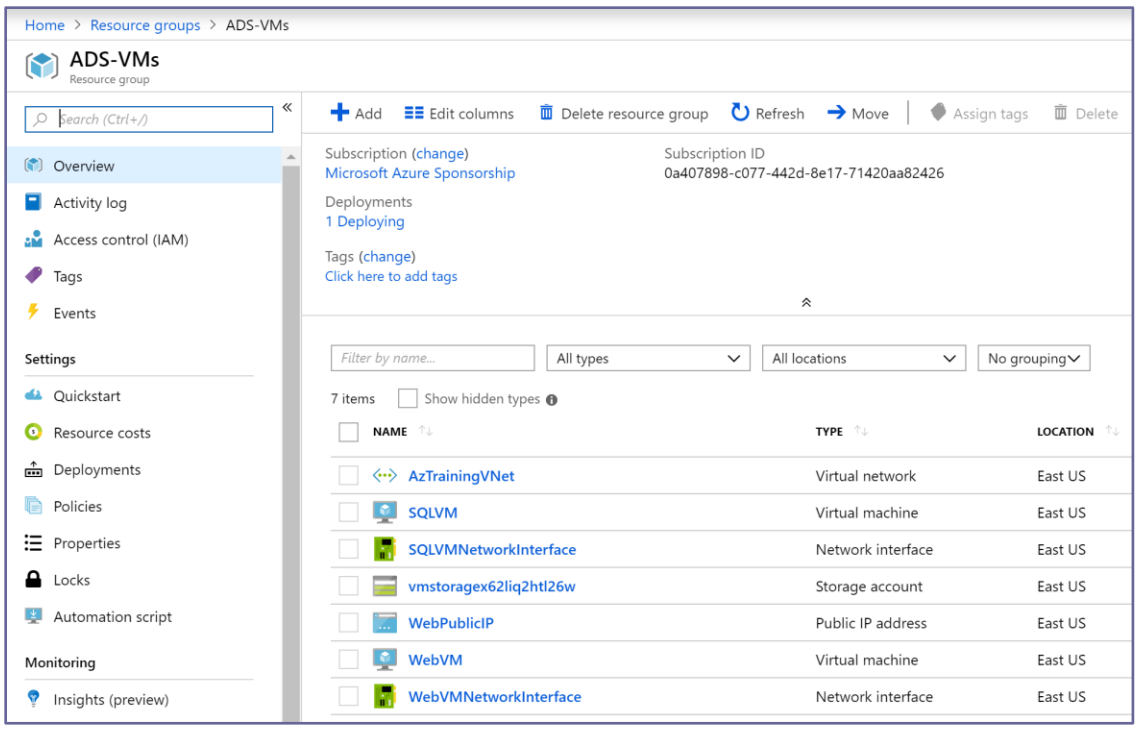
19. Once all settings have been completed in the Parameters popup window, click **Save**. You are redirected to the "Deploy to Resource Group" window. Start the actual deployment by pressing the **Deploy** button.




20. The Azure Resources deployment kicks off, which can be followed from the Visual Studio **Output** window. (**For your info, this deployment takes about 15-20min might be a good time for a break** 😊).

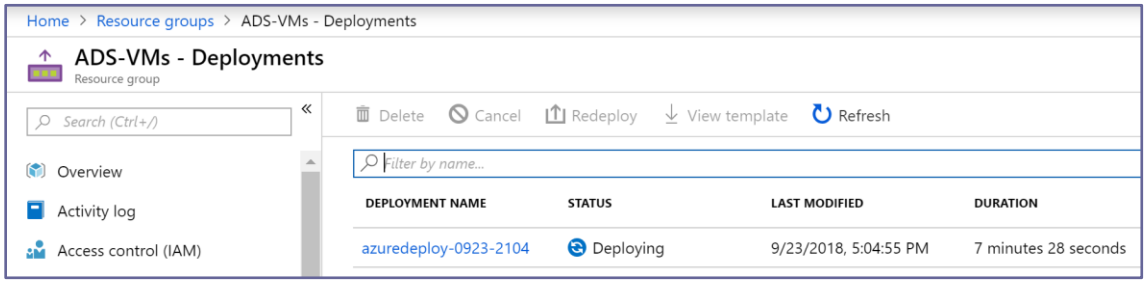
```
21:04:07 - LastModified      : 9/23/2018 9:04:07 PM +00:00
21:04:07 - SnapshotTime      :
21:04:07 - ContinuationToken :
21:04:07 - Context           : Microsoft.WindowsAzure.Commands.Common.Storage.LazyAzureStorageContext
21:04:07 - Name              : DSC/WebDSC.zip
21:04:07 -
21:04:08 - VERBOSE: Performing the operation "Replacing resource group ..." on target "".
21:04:17 - VERBOSE: 9:04:09 PM - Created resource group 'ADS-VMs' in location 'eastus'
21:04:17 -
21:04:17 - ResourceGroupName : ADS-VMs
21:04:17 - Location          : eastus
21:04:17 - ProvisioningState : Succeeded
21:04:17 - Tags              :
21:04:17 - TagsTable         :
21:04:17 - ResourceId        : /subscriptions/0a407898-c077-442d-8e17-71420aa82426/resourceGroups/ADS-VMs
21:04:17 -
21:04:17 - VERBOSE: Performing the operation "Creating Deployment" on target "ADS-VMs".
21:04:17 - VERBOSE: 9:04:13 PM - Template is valid.
21:04:17 - VERBOSE: 9:04:15 PM - Create template deployment 'azuredeploy-0923-2104'
21:04:17 - VERBOSE: 9:04:15 PM - Checking deployment status in 5 seconds
21:04:20 - VERBOSE: 9:04:20 PM - Checking deployment status in 10 seconds
21:04:30 - VERBOSE: 9:04:30 PM - Resource Microsoft.Network/virtualNetworks 'AzTrainingVNet' provisioning status i
21:04:30 - VERBOSE: 9:04:30 PM - Resource Microsoft.Network/publicIPAddresses 'WebPublicIP' provisioning status is
21:04:30 - VERBOSE: 9:04:30 PM - Resource Microsoft.Storage/storageAccounts 'vmstoragex62liq2htl26w' provisioning
```

21. While the deployment from Visual Studio is still running, open your **internet browser**, **connect** to http://portal.azure.com, **authenticate** with your Azure subscription credentials. **Go to Resource Groups, open the [SUFFIX]containersDATE Resource Group**. Here, you can see the different resources getting created.

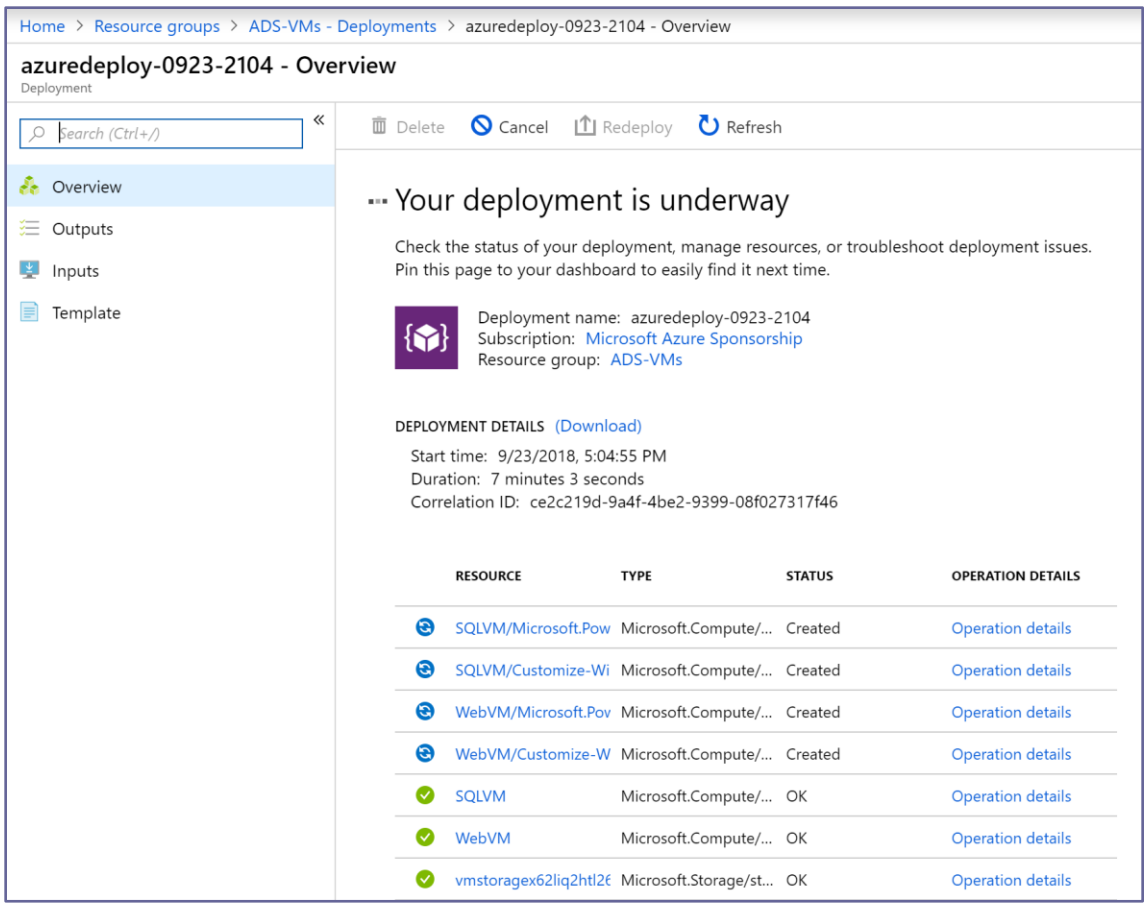



22. From the **Resource Group** blade, **Settings** section, click **Deployments.**

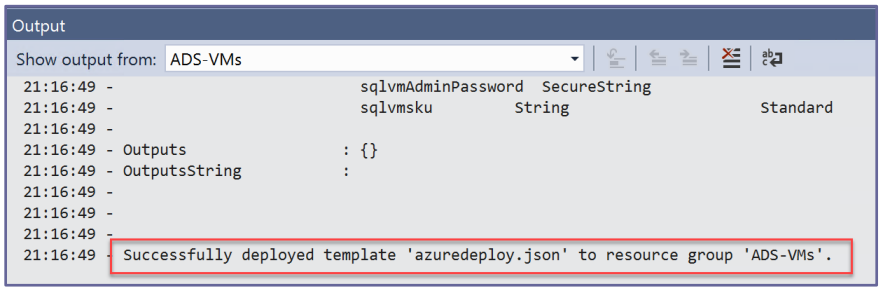
23. This **shows** the actual running deployment task.

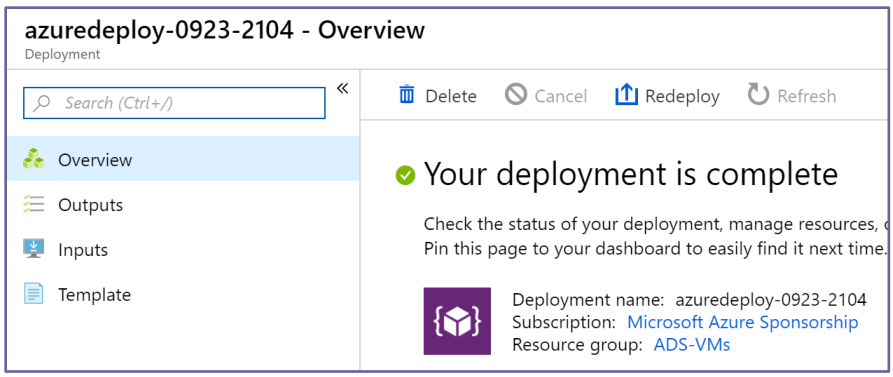



24. **Click the deployment name <e.g. azuredeploy-0923-2104>,** which shows you more details about the actual deployment process, including the already deployed resources.




25. Wait for the deployment to complete successful. This is noticeable from within the Visual Studio Output window, or from within the Azure Portal deployment blade you were in before.

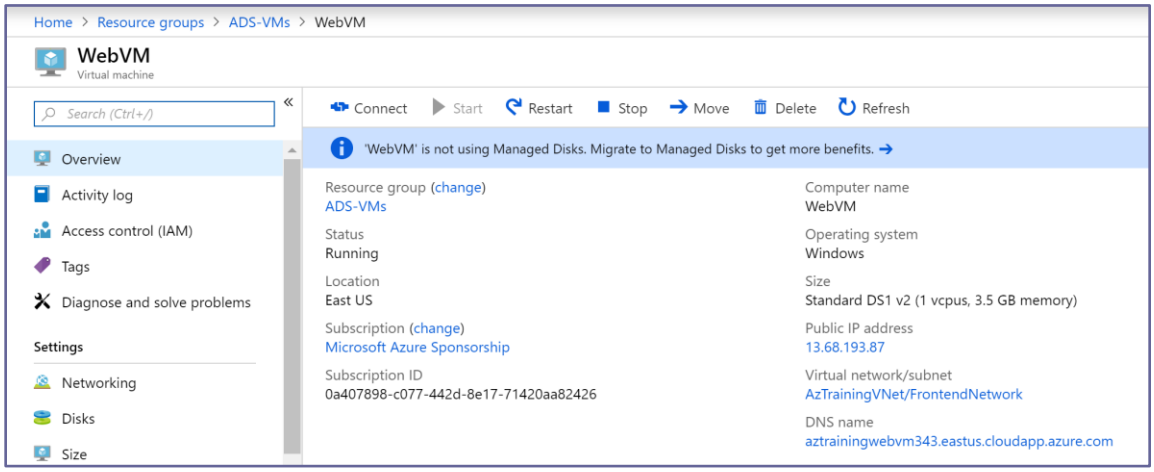




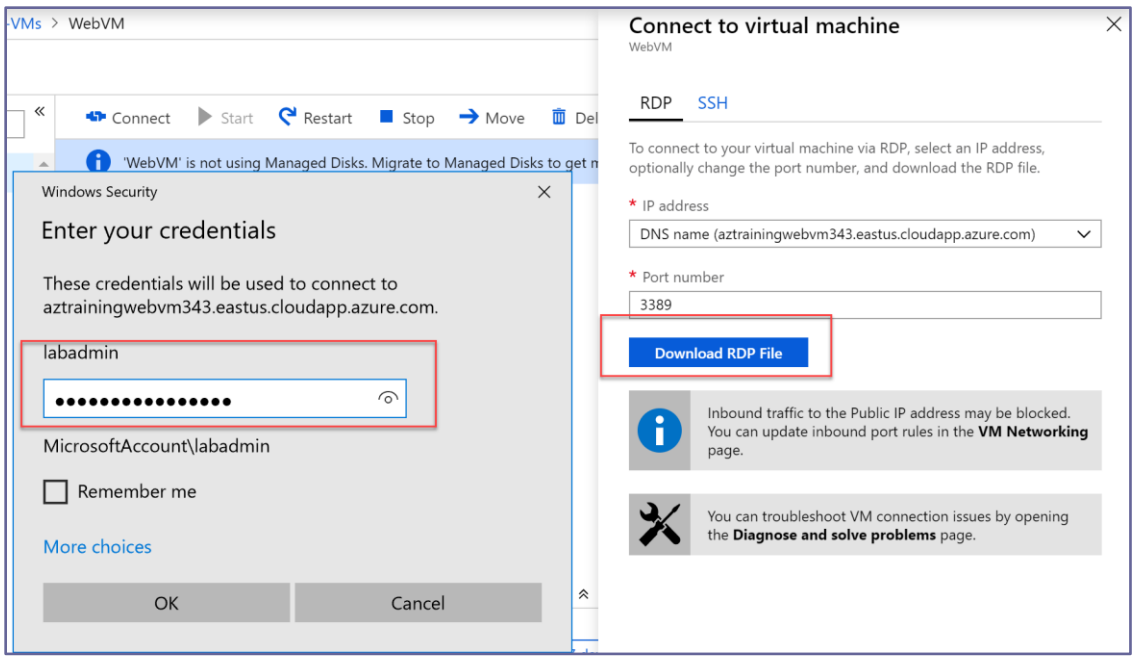

26. Close Visual Studio without saving changes to the project.

To verify all went fine during the deployment of the Azure Resources, as well as the customization and configuration using PowerShell Desired State Configuration, log on to the WebVM to validate the web application is running as expected.

27. From within the Azure Portal, go to Resource Groups, and select the Resource Group where you deployed the VMs. In here, select the WebVM Virtual Machine by clicking on it. This opens the WebVM detailed blade.

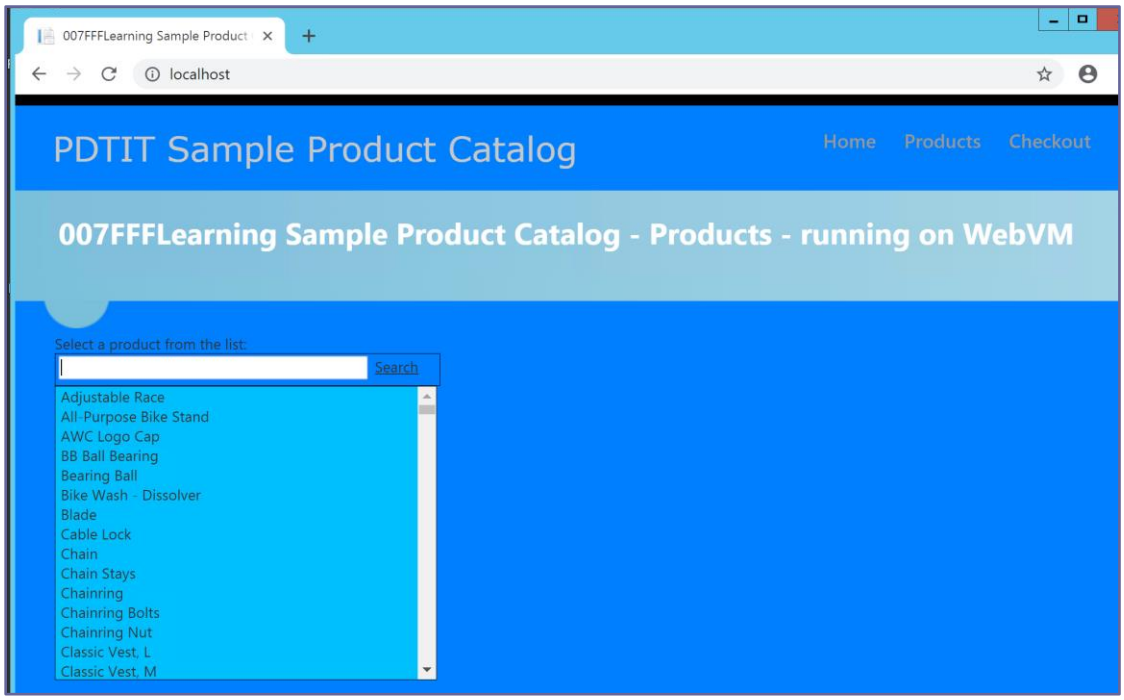



28. Similar to what you did with the lab-JumpVM, press the Connect button, to open the Remote Dekstop session to this WebVM Virtual Machine.




29. Here, log on with the credentials from the ARM template (labadmin / L@BadminPa55w.rd) unless you changed those before the deployment.

30. From within the WebVM RDP-session, open an internet browser, and browse to http://localhost. This opens the Product Catalog web application, establishing a connection to the SQLVM to connect to the underlying SQL database.



31. Close the browser session on the WebVM.

32. Close the RDP session for the WebVM Virtual Machine.

This completes the task in which you deployed Azure Resources using Visual Studio 2017 ARM-template with customizations, and validated the good functioning of the web application.

## Summary

In this lab, you started with deploying an ARM-template from within the Azure Portal, deploying a lab-JumpVM Virtual Machine in Azure. In the next task, you learned how to deploy a more complex Azure environment, again using an ARM-template, where deployment was executed from within Visual Studio 2017.