

**Azure ISV:** Application Migration (IaaS) Hack

@ Reykjavik

**Workbook**

Contents

[Labs and Activities 3](#_Toc476756907)

[Lab 1.1: Login to the Azure Portal 3](#_Toc476756908)

[Lab 1.2: Create a Resource Group 4](#_Toc476756909)

[Lab 2: Deploy a Virtual Network 5](#_Toc476756910)

[Lab 3.1.1: Deploy a Standard Storage Account 6](#_Toc476756911)

[Lab 3.1.2: Deploy a Premium Storage Account 6](#_Toc476756912)

[Lab 3.2.1 Azure Blob Storage 7](#_Toc476756913)

[Lab 3.3 Azure File Storage 8](#_Toc476756914)

[Lab 4.1 Deploy a DS1 / DS1v2 Virtual Machine with a Public IP Address 9](#_Toc476756915)

[Lab 4.2 Deploy a VM Scale Set of 2x DS1 / S1v2 Virtual Machines behind a load balancer with a Public IP Address. 10](#_Toc476756916)

[Lab 4.3 Deploy a public facing RDS Farm with a Public IP Address from the github repo 11](#_Toc476756917)

[Lab 5.1 Create your own ARM Template from scratch in Visual Studio 12](#_Toc476756918)

[Lab 5.2 Deploy an ARM Template from Visual Studio 13](#_Toc476756919)

[Lab 6 Azure Scripted Deployments – DSC Scripts: Deploy a webserver using the VM DSC extension 14](#_Toc476756920)

[These pages are intentionally left blank for your notes 15](#_Toc476756921)

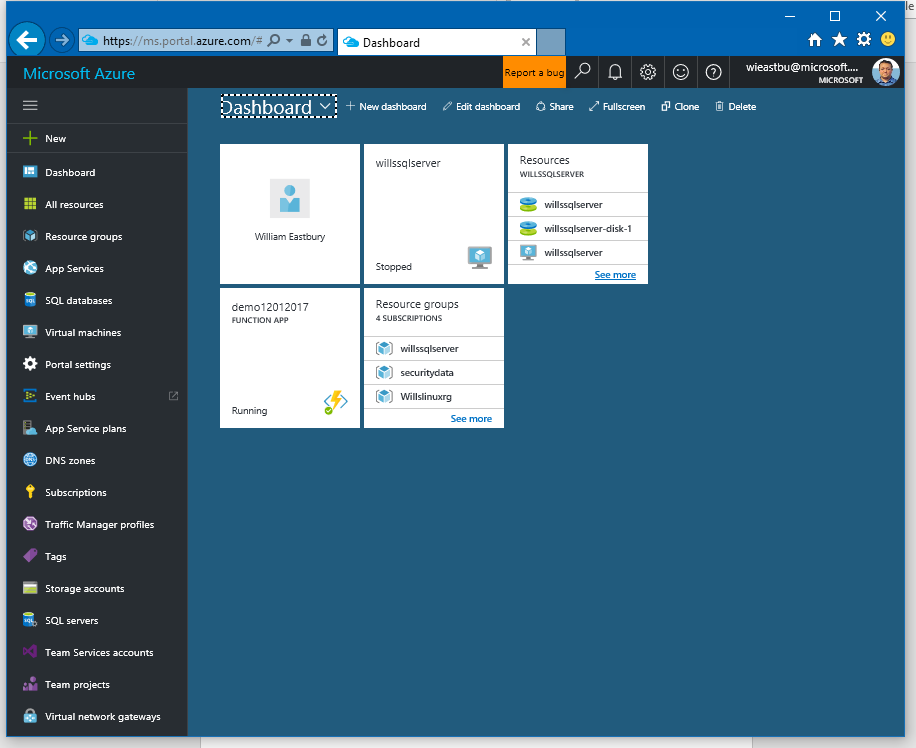
|  |  |
| --- | --- |
| Resource | Name |
| Resource Group |  |
| Standard Storage Account / Premium | / |
| Blob Container / File Share | / |
| Blob Shared Access Signature (SAS) URI |  |
| Load Balancer Name / LB IP Name | / |
| VM username and Password | User:  Password: |

# **Labs and Activities**

## Lab 1.1: Login to the Azure Portal

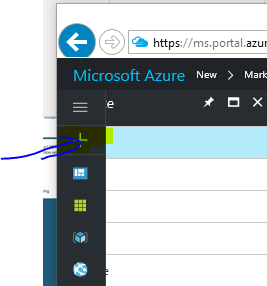
Navigate to <http://portal.azure.com>

Enter your username and password.

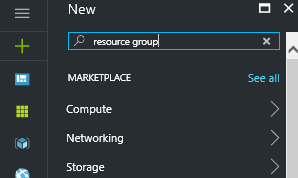
You should see a dashboard screen looking something like this.

## Lab 1.2: Create a Resource Group

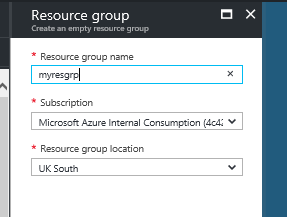
1. Open the Azure Portal
2. Click on the Green + in the top left corner to create a new resource group



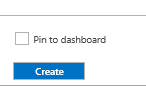
1. Type Resource Group and press Return



1. Select the top entry of “Resource Group” from the list and then click ‘create’
2. Give your resource group a name that does not include spaces or special characters apart from \_

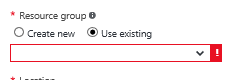


1. Click ‘Create’ and wait for the operation to complete.



## Lab 2: Deploy a Virtual Network

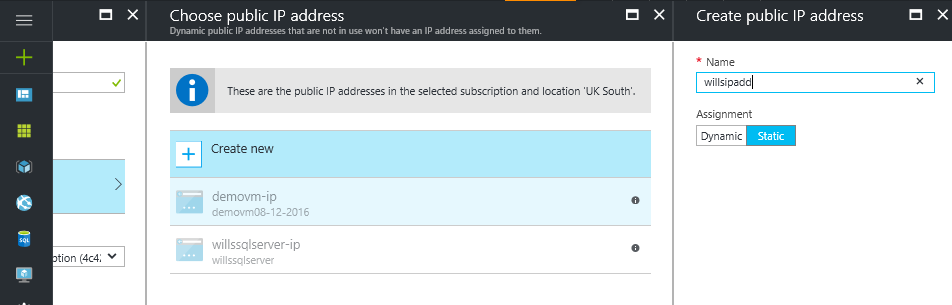
1. Open the portal
2. Click the green + in the top left
3. Type “Virtual Network” and Press return
4. Select the top entry of “Virtual Network” from the list and then click ‘create’
5. Give your virtual network a name that does not include spaces or special characters apart from \_
6. Select “Use Existing” resource group and enter the name of the resource group you created in lab 1.2



1. Click ‘Create’ and wait for the operation to complete.

Lab 2.1 Deploy a Load Balancer with a Public IP

1. Open the portal
2. Click the green + in the top left
3. Type “Load Balancer” and Press return
4. Select the top entry of “Load Balancer” from the list and then click ‘create’
5. Give your Load Balancer a name that does not include spaces or special characters apart from \_ and record it on page 2.
6. Select “Public” as the type (this assigns a Public External IP, or PIP)
7. Click “Choose a public IP address” then “Create New”
8. Give the IP a globally unique name then select ‘Static’ and record the name on page 2



1. Click OK at the bottom right of the blade.
2. Select “Use Existing” resource group and enter the name of the resource group you created in lab 1.2
3. Click ‘Create’ and wait for the operation to complete.

**This will create you an externally facing load balancer with an external static IP address, but no NAT rules hence no traffic will be passed.**

## Lab 3.1.1: Deploy a Standard Storage Account

1. Open the portal
2. Click the green + in the top left
3. Type “Storage Account” and Press return
4. Select the top entry of “Storage Account” from the list and then click ‘create’
5. Give your storage account a name that does not include spaces or special characters apart from \_ - keep the name short, but globally unique (this ID is used as a DNS prefix for your account).
6. Select “Resource Manager” as the deployment model
7. Select the “Standard” Performance tier
8. Select “Read-access geo-redundant storage” for the replication option
9. Keep all other settings at their defaults.
10. Select “Use Existing” resource group and enter the name of the resource group you created in lab 1.2
11. Click ‘Create’ and wait for the operation to complete.
12. Record the account name on page 2

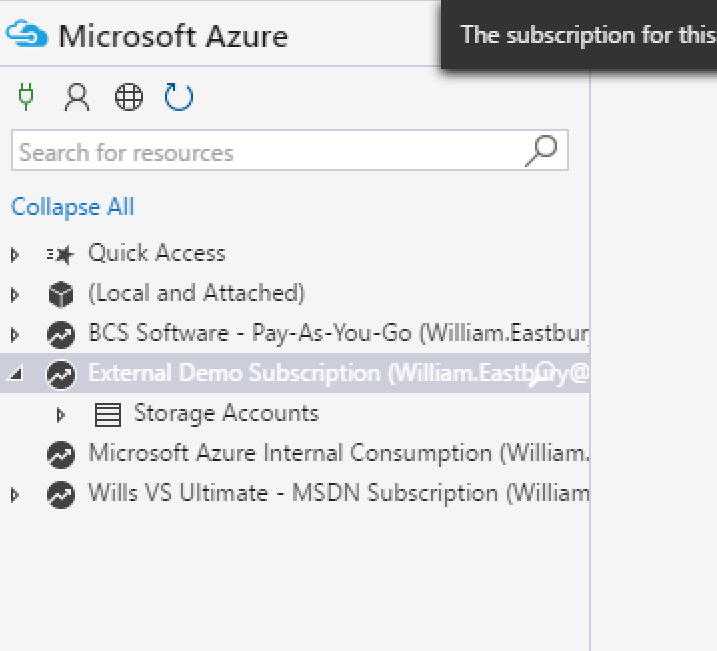
## Lab 3.1.2: Deploy a Premium Storage Account

1. Open the portal
2. Click the green + in the top left
3. Type “Storage Account” and Press return
4. Select the top entry of “Storage Account” from the list and then click ‘create’
5. Give your storage account a name that does not include spaces or special characters apart from \_ - keep the name short, but globally unique (this ID is used as a DNS prefix for your account).
6. Select “Resource Manager” as the deployment model
7. Select the Premium” Performance tier
8. “locally redundant storage” will be auto-selected for the replication option
9. Keep all other settings at their defaults.
10. Select “Use Existing” resource group and enter the name of the resource group you created in lab 1.2
11. Click ‘Create’ and wait for the operation to complete.
12. Record the account name on page 2

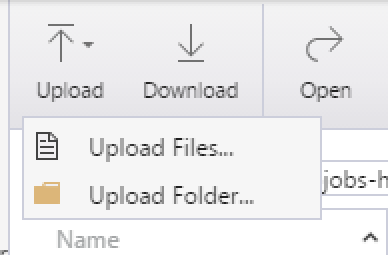
## Lab 3.2.1 Azure Blob Storage

**3.2.1.1 Upload a blob to a container**

1. Open Microsoft Azure Storage Explorer
2. Click on the user icon in the top left and enter your credentials to connect



1. Expand into your subscription and the storage accounts sub item
2. Expand into the standard storage account we created earlier, in lab 3.1.1 (the name will be on page 2)
3. Right Click on ‘Blob Containers’ and select “Create Blob Container” and enter a short alphanumeric name
4. Record the container name on page 2 for later use
5. Select the container you just created in the tree and click “Upload” on the menu bar, then select “Upload Files”



1. Select a file to upload (your application’s installer) and click ‘upload’.

**3.2.1.2 Issue a SAS for the blob and hit the URI.**

1. Right click on the uploaded file in the grid and select ”Get Shared Access Signature” from the menu.
2. Set a sensible expiry time and ensure that only “Read” is checked, then click create.
3. On the next screen, copy the URL and paste it into internet explorer to download the blob, then record it on page 2. (double check the writing or copy it to a text file on your machine !)

## Lab 3.3 Azure File Storage

1. Open Microsoft Azure Storage Explorer
2. Authenticate if necessary
3. Expand into your subscription and the storage accounts sub item
4. Expand into the standard storage account we created earlier, in lab 3.1.1 (the name will be on page 2)
5. Right Click on ‘File Shares’ and enter the name of a new SMB file share
6. Click Upload, Upload File, and select a file to upload.
7. Click on “Connect VM” and copy the command given
8. Bring up the run window and enter cmd <return>
9. Paste the command into the run window (substituting a valid drive letter) and execute it.
10. Note that the drive appears in explorer with read/write access
11. Marvel at the wonders of the internet connected storage you just accessed.

## Lab 4.1 Deploy a DS1 / DS1v2 Virtual Machine with a Public IP Address

1. Open the Azure Portal and click the green + then enter “Windows Server 2016 Datacenter” to open the ‘everything’ blade.
2. Select Windows Server 2016 Datacenter and click create
3. Basics:
   * Give your VM a globally unique name
   * Choose SSD (Premium) storage for the VM disk type
   * Enter a user name and >12 chars / complex password (+ confirm the password) for the admin account and optionally note them down on page 2.
   * Select “Use Existing” resource group and enter the name of the resource group you created in lab 1.2, then click OK.
4. Size:
   * Choose a DS1v2 Standard VM (If not available in your region, choose a DS1).
   * Click “Select”
5. Settings:
   * If managed disks is available then you can select to use that, otherwise select the premium storage account we created earlier (name on page 2).
   * Select the virtual network we created earlier
   * Allow the creation of a new PIP and NSG by accepting the defaults
   * Do not select any extensions
   * This is a single instance VM, so does not need to be in an availability set.
   * Click ‘OK’
6. Click ‘OK’ again.

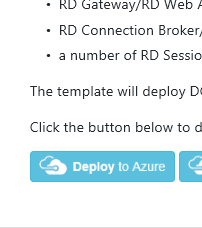
Your Virtual Machine will now provision automatically.

## Lab 4.2 Deploy a VM Scale Set of 2x DS1 / S1v2 Virtual Machines behind a load balancer with a Public IP Address.

1. Open the Azure Portal and click the green + then enter “Virtual Machine Scale Set” to open the ‘everything’ blade.
2. Select Virtual Machine Scale Set and click create
3. Basics:
   * Give your VMSS a globally unique name
   * Select ‘Windows’ as the OS type
   * Enter a user name and >12 chars / complex password (+ confirm the password) for the admin account and optionally note them down on page 2.
   * Choose ‘True’ for Limit to a single placement group
   * Enter the name of a **new** resource group to create and click OK.
4. Virtual machine scale set service settings:
   * Allow the creation of a new PIP by accepting the default
   * Add a unique dns prefix in the domain name label field
   * Enter ‘2’ for the number of instances
   * Select a D1v2 as the instance type
   * Select the disk type as ‘Managed’
   * Set autoscale to ‘enabled’
   * Click “OK”
5. Click ‘OK’ again.

**Your Virtual Machine Scale Set will now provision automatically.**

## Lab 4.3 Deploy a public facing RDS Farm with a Public IP Address from the github repo

1. Open github and find the template at (<https://github.com/Azure/azure-quickstart-templates/>)
2. Search on the page for “rds-deployment” and select that folder
3. Click the  button on the readme.md page
4. Either choose an existing resource group, or create a new one (note that for this template the RG location defines the VM region, which defines the availability of VM types).
5. Enter the following settings :-

* **Gwdns Label Prefix** a globally unique DNS prefix for your RD gateway
* **Gwpublic IP Address Name** the azure name of the gateway public ip address
* **Ad Domain Name** this template will build a windows active directory for you, what do you want to call the domain. *(For example contoso.com)*
* **Admin Username** The name of the administrator of the new VM and the domain. Exclusion list: 'administrator'. *For example johnadmin*
* **Admin Password** The password for the administrator account of the new VM and the domain
* **Image SKU** what OS do you want the farm to run *2016 datacenter would make sense.*
* **Number Of Rdsh Instances** How many session hosts do you want to scale to ? *2?*
* **RSDH vm size** what base Vm size do you want ?

*Note: if we chose UK South as the region here, this template needs to be edited as D1 and A1 sizes are not available in the UK, only the v2 types are available in the newer DCs.*

*Hence, click Edit*

*Then find the json for "rdshVmSize”*

*And add ,"Standard\_D1v2" to the end of the array*

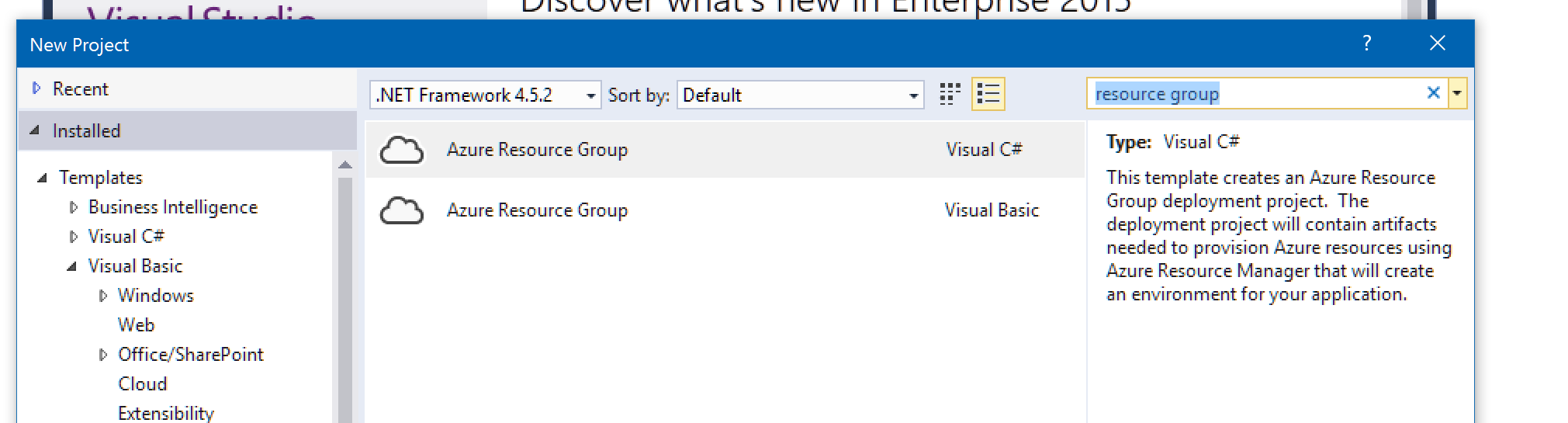
*And click save to go back.*

*Now you can select Standard\_D1v2 for your vm hosts.*

1. Check ‘I agree to the terms and conditions stated above’, then click “Purchase” to deploy your RDSH farm.

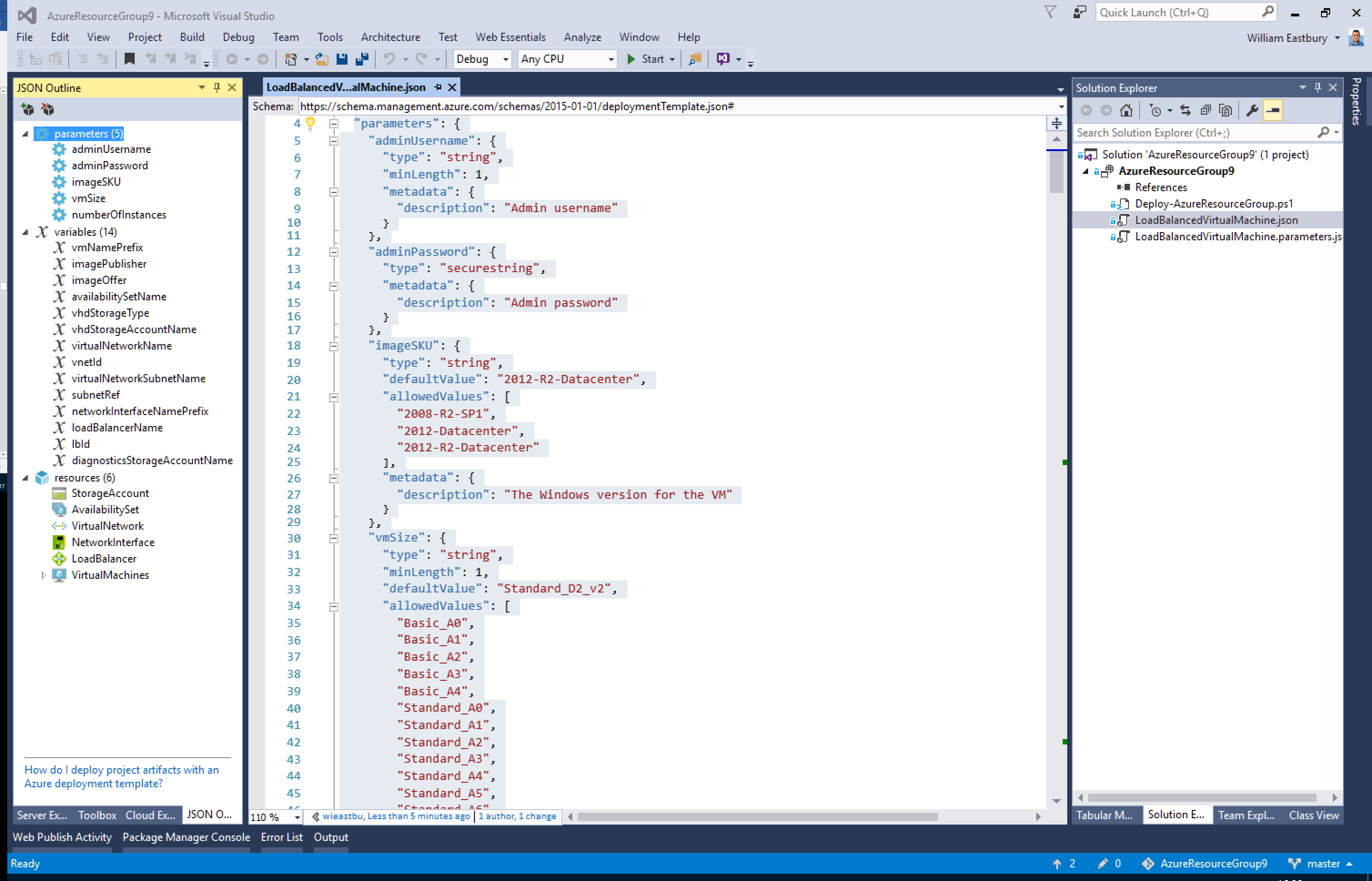
## Lab 5.1 Create your own ARM Template from scratch in Visual Studio

1. Open Visual Studio and Select File > New > Project
2. Enter “Resource Group” into the “Search installed templates” search box



1. Select the “Azure Resource Group” project type and click OK.
2. \*\* Bonus points if you check the code into a git repo in VSTS / github at this point \*\*
3. From the popup ensure that location is set to ‘visual studio templates’, **but note that ALL of the github quickstart templates are available for use here as a starting point.**
4. Select ‘Windows Server Virtual Machines with Load Balancer’ and click OK (Note: This template by default creates a configurable number of load-balanced static vms in an availability set, but without autoscaling in a VMSS).
5. Ensure that both the Solution Explorer (Ctrl-alt-L) and JSON Outline View (View > Other Windows > JSON Outline) panels are visible and open the file “LoadBalancedVirtualMachine.json” by double clicking it in Solution Explorer.
6. Expand the entire tree in the JSON outline view.

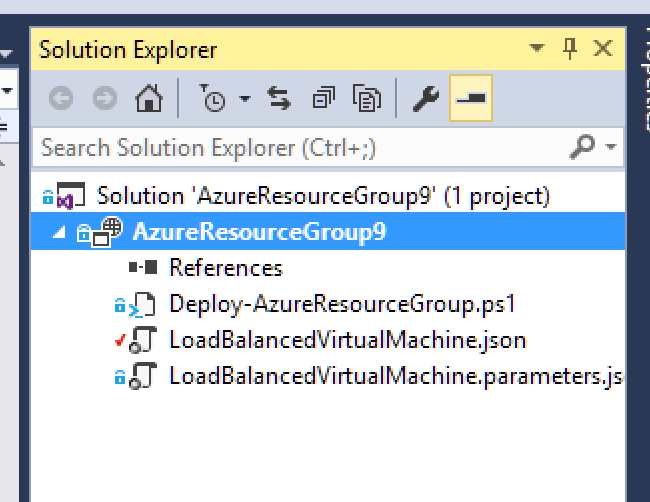
**Your screen should look something like the below.**



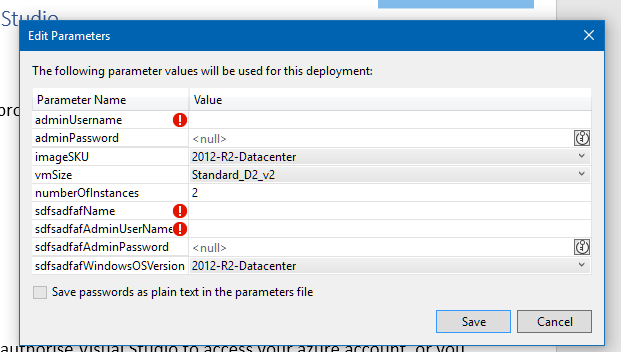
1. Right-click on the “Resources” tree node in the JSON outline view, and select ‘add resource’.
2. Select “Windows Virtual Machine” and give the VM a name, leave the defaults in place to add to the existing virtual network, but create a new storage account.
3. Click ‘Add’
4. Observe that new parameters, variables, and resources have been added.
5. Repeat steps 9-11 but add a “Public IP Address” type resource to the NIC created for you in step 11 (This will make your new virtual machine publicly accessible.
6. **Do not close visual studio until you have completed the NEXT lab.**

## Lab 5.2 Deploy an ARM Template from Visual Studio

1. Return to the prior visual studio project.
2. Right click on the Azure Resource Group project (not the solution, the level below) in Solution Explorer



1. Select “Deploy > New”
2. You may need to add an account here to authorise Visual Studio to access your azure account, or you might have to refresh your credentials.
3. Select the subscription to deploy to and select the resource group to deploy to (or create a new one from the dropdown list).
4. Click “Edit parameters” to show the parameter dialog (this contains the same parameters you would need to deploy from the azure portal.



1. Populate the 2x admin username and password entries with sensible values, and give the second vm instance we created in step 10 of the previous lab a sensible globally unique name.
2. At the bottom of the dialog (you might need to scroll the grid, enter a name for the public IP of the second vm (make it lowercase and unique).
3. Then hit ‘Save’.
4. Then hit ‘OK’ and the output window should appear, you may get prompted for the admin passwords again here, type them and hit “OK”.
5. If the output window has disappeared at this point, hit Ctrl-alt-O to display it, and watch your general DevOps awesomeness.
6. If you are ahead of the game here and have time, then bonus points if you can figure out how to use the script Deploy-azureresourcegroup.ps1 to run the deployment 😉
7. Double bonus points if you can check in the template to source control and get VSTS to deploy your template in CI mode to check it’s valid 😊😊

## Lab 6 Azure Scripted Deployments – DSC Scripts: Deploy a webserver using the VM DSC extension

1. Back to our visual studio project, and this time add a “PowerShell DSC Extension” task to your second vm instance we created in step 10 of the previous lab.
2. Give it a sensible name and click OK.
3. Notice that a “DSC” folder has appeared in the project, with a powershell script inside it (a file with a .ps1 extension).
4. Open this file and uncomment the contents (remove the “<# This commented section represents an example configuration that can be updated as required.”
5. And “#>” tags
6. You now have a script to install a working IIS deployment with ASP.NET and Web deploy enabled
7. Save the script and deploy the resource group again as you did in the last lab, but this time right click on the project and select Deploy > AzureResourceGroup1 (or whatever project name you have).
8. The PowerShell DSC extensions should simply deploy IIS onto your existing machine in the resource group as the infrastructure already exists.
9. Go view the resource group you created in the Azure Portal and connect to the VM’s public IP on port 80.
10. **You should see the IIS holding page.**

-End-

## These pages are intentionally left blank for your notes