Student Name: **Weight: 5%**

Student ID: **Marks: /15**

Lab: JSON and Bicep

Lab Objectives

In this lab, you'll explore how to create Azure resources using the JSON and Bicep languages, PowerShell, and the CLI. You will:

1. View the JSON syntax
2. Deploy a virtual machine with an ARM template
3. Use the Azure CLI from a local system
4. Use Powershell from a local system
5. Deploy a Bicep file

Lab Requirements

- Up to date browser
- Azure account
- Github free account

Instructions

1. Working individually, follow the procedure below.
2. Take screenshots, as described in the *Marking Criteria* section.
3. Create a document that includes all screenshots appropriately titled and described, and then upload it to Brightspace, as indicated by your instructor.
4. Be sure to include your name and student ID in the document.

Marking Criteria

Screenshots	Marks
Edited JSON template	/2
All steps of creating a virtual machine with the CLI	/4
All steps of creating a storage account with PowerShell	/4
Bicep file and screenshots of deployment	/5
Total	/15

Note: This icon indicates when a screenshot is required.

Source: Flatiron.com, Freepik, Image: screenshot_983871

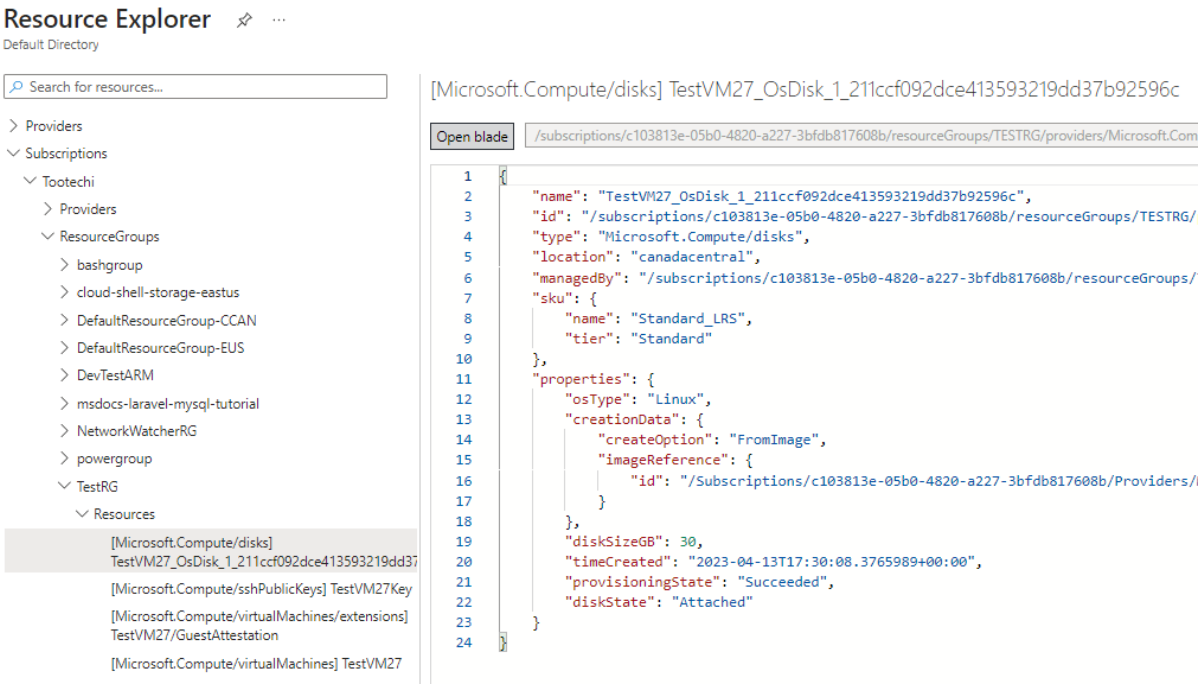
Procedure

Section 1: View the JSON Syntax

In previous labs, you've used PowerShell and CLI commands to create and manage resources. You've also briefly looked at the JSON code. JSON is a declarative language, meaning that it allows you to define resources without having to use explicit commands or steps like "create" or "add." You describe *what* you want to be done (e.g., a virtual machine of a specific size) not *how* to do it (e.g., create a processor, memory, disk). These types of languages are also called nonprocedural languages.

- ☐ Create an inexpensive virtual machine in your account.
- ☐ Navigate to the **Resource Explorer** in the Azure portal, and then navigate to your virtual machine.

When you select any resource, you see the JSON code for that resource in the blade menu. In the example below, you can see the JSON code for the operating system disk for the virtual machine and all its properties.



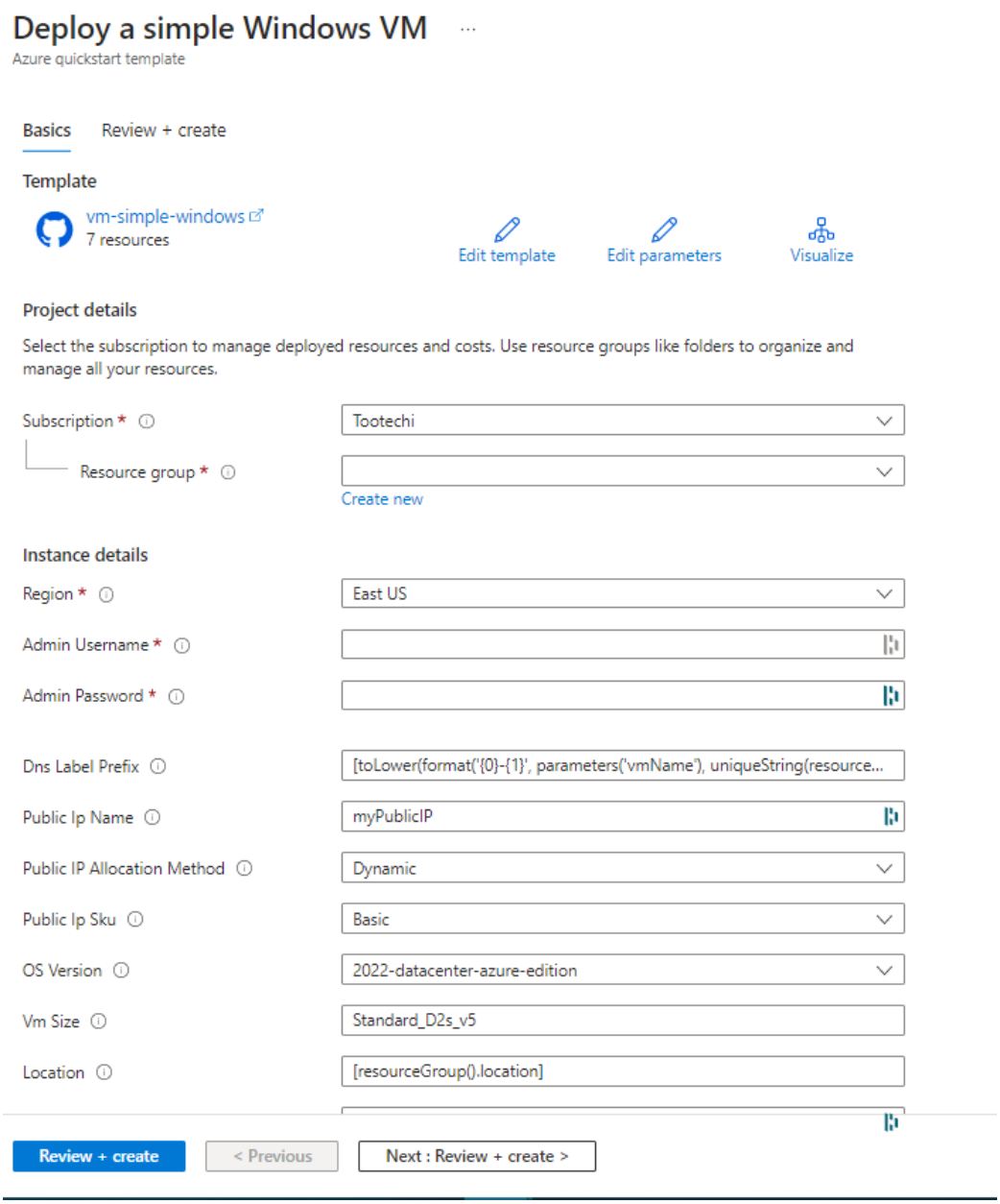
- ☐ Look through the components of the virtual machine and note how the JSON syntax is used to describe each of the pieces.
- ☐ Read: Understand the structure and syntax of ARM templatesv (https://learn.microsoft.com/en-us/azure/azure-resource-manager/templates/syntax).

Part 2: Deploy a Virtual Machine with an ARM Template

Whether you are using the Azure portal, PowerShell or the CLI, you are interfacing with the Azure Resource Manager (ARM). By using ARM templates, you can keep repositories for specially configured virtual machines, as well as deploy many virtual machines quickly and easily. ARM templates are stored in the JSON (JavaScript Object Notation) language and you can see the existing template of anything you have created. These templates are also known as IAC (infrastructure as code).

- ☐ Read the following resources:
  - What is Azure Resource Manager? (https://learn.microsoft.com/en-us/azure/azure-resource-manager/management/overview)
  - Understand the structure and syntax of ARM templates (https://learn.microsoft.com/en-us/azure/azure-resource-manager/templates/syntax)
- ☐ If you don't have a virtual machine, create one.
- ☐ Go to the main page for the virtual machine and from the **Overview** window select **JSON View**.
- ☐ You see the template for the virtual machine you created. Note the profile sections for the disks, operating system, network, security and all other configuration sections for the virtual machine.
- ☐ Azure has some simple ARM templates anyone can use, so you don't have to create one from scratch. Go to: Deploy a simple Windows VM (https://learn.microsoft.com/en-us/samples/azure/azure-quickstart-templates/vm-simple-windows/) to connect to a simple Windows VM template.
- ☐ Click the **Visualize** button to see the components that will be created. Click each one to see the JSON code for that section.
- ☐ Go back to the original link and click the **Deploy to Azure** button.

  This brings you back to your portal window with most of the virtual machine information filled out. The name, resource group and user information is blank.



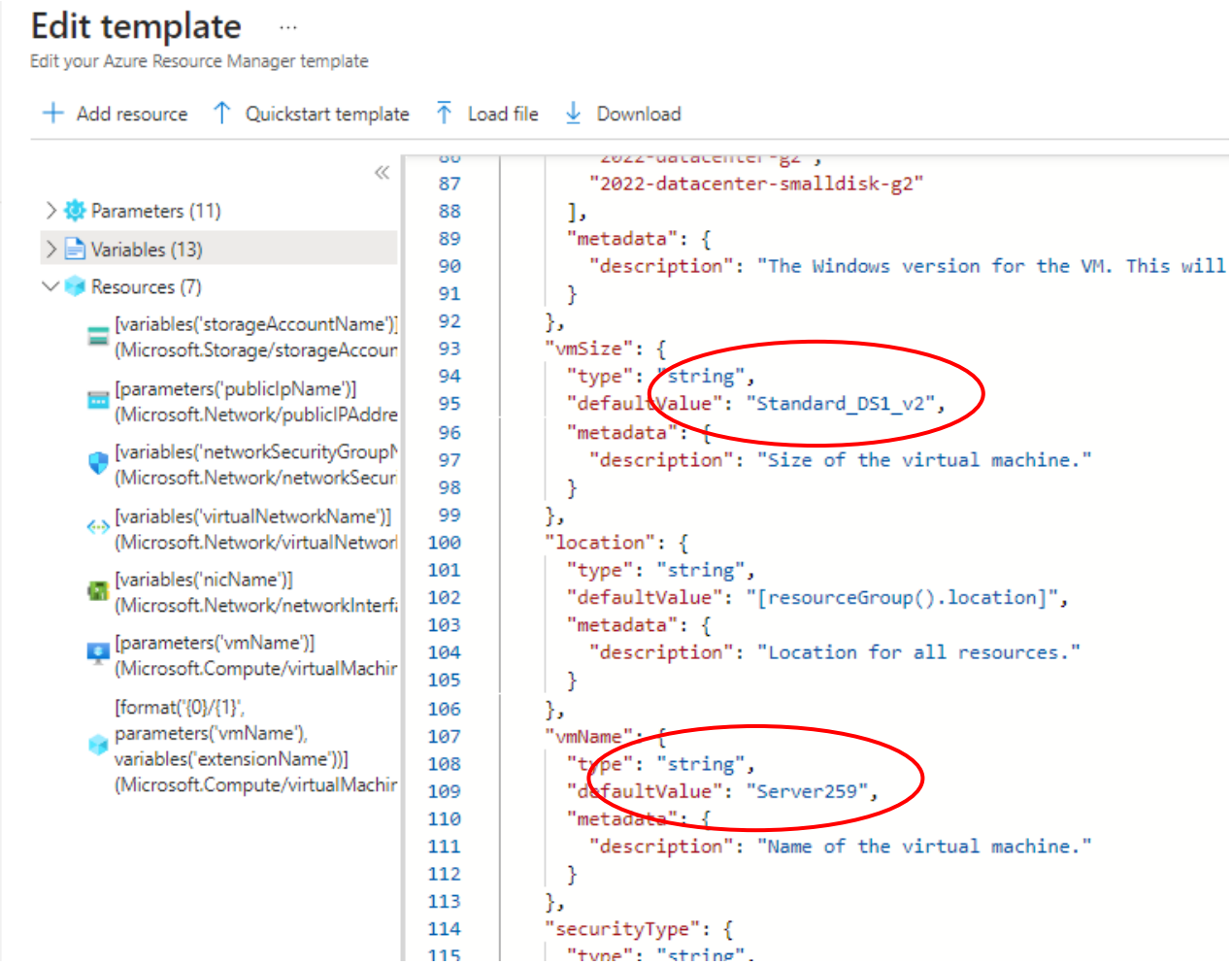
☐ Click **Edit Template** at the top of the screen.

There are three options in the blade menu:

- Parameters – input values you can create in the deployment
- Variables – values that can be reused in the template
- Resources – create your own functions in the template

☐ Click **Variables** and find the *vmName parameter* in the main window.

- ☐ Change the default name from simple-vm. In the example below, the virtual machine is named Server 259.
- ☐ Find the *vmSize* parameter and change it to **Standard_DS1_v2**.

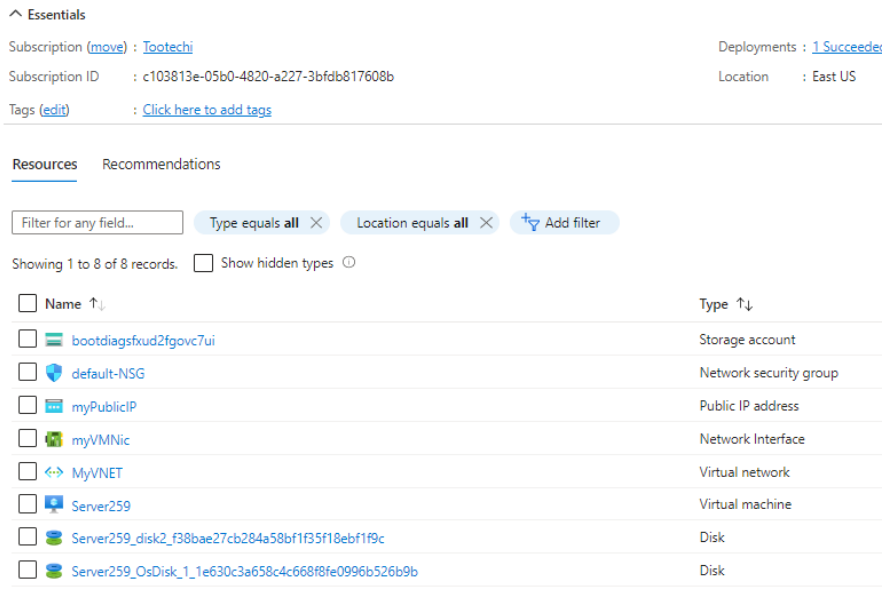


- ☐ Examine the variables to see the different types of information.
- ☐ Save the template. The Deployment page summarizes the changes you made.
- ☐ Put your template in the security group you created earlier, add the admin user and password, and then review and create your VM from the template.

As you've already learned, when you create a virtual machine, you also need to create several other components if you do not already have them.

- IP addresses
- Network components
- NIC cards
- Operating system and temporary disks
- Data disks
- Storage accounts

☐ Go to the **Resource Group** main page. You'll see each of the components.



☐ Once you have all the components, download the ARM template by selecting the **Export Template** option from the blade menu.

Note: GitHub also has several ARM templates you can use. Go to Azure quickstarts (https://github.com/Azure/azure-quickstart-templates/tree/master/quickstarts) to see the full list of available templates.

Part 3: Use the Azure CLI from a Local System

Until now, you have been executing CLI and PowerShell commands from the portal. However, you can also connect to Azure and execute commands from a command line or the PowerShell on your local computer. You can install the appropriate software on your operating system or use a virtual machine for installation.

- ☐ Install the latest version of the Azure CLI by following the instructions at: Install Azure CLI on Windows (https://learn.microsoft.com/en-us/cli/azure/install-azure-cli-windows?tabs=azure-cli).

  Note: You can execute the commands from the PowerShell interface or from the command line.

- ☐ Use the `az login` command to log into your Azure account.
- ☐ Type `az group` to access helpful information about how to use the az group command and where to find more information.
- ☐ Type the `az group` list command to see a list of all your resource groups.
- ☐ Complete all sections of the Azure tutorial, beginning with: Overview and Prerequisites (https://learn.microsoft.com/en-us/cli/azure/azure-cli-vm-tutorial-1).

  Notes:

  - Use the CLI from your local machine, not in the portal.
  - Take screenshots of all the sections.
    - Overview and prerequisites
    - Create a virtual network
    - Create a virtual machine
    - Get VM information with queries
    - Set shell variables
    - Cleanup
    - Summary

Part 4: Use Azure PowerShell from a Local System

- ☐ To update your version of PowerShell, download the PowerShell .msi from: Installing the MSI package (https://learn.microsoft.com/en-us/powershell/scripting/install/installing-powershell-on-windows?view=powershell-7.3#msi)
- ☐ Install the .msi to the local system or virtual machine you are using.
- ☐ Click the Start menu (Windows icon). You should see a PowerShell selection under *Recently Added*.
- ☐ Use the following commands to install the Az modules:

```
Install-Module -Name Az -AllowClobber
Import-Module Az.Accounts
Install-Module AzureAD
Import-Module AzureAD
```

- ☐ Open PowerShell on your local system and use the: `Connect-AzAccount` command to log in to Azure.

  If you have MFA enable, you must use the command:

  `Connect-AzAccount -TenantID yourtenantidhere`

- ☐ Download the PowerShell Reference Guide (https://slstudentpublic.blob.core.windows.net/operations/PowerShell%20Guide_Skylines%20Academy_AZ.pdf).
- ☐ Follow the commands at: Create a storage account (https://learn.microsoft.com/en-us/azure/storage/common/storage-account-create?tabs=azure-powershell)
- ☐ Take screenshots.

Part 5: Deploy a Bicep File

While JSON is very useful and templates save enormous amounts of time in redeployments, JSON is more complex. Azure has created a new declarative language called Bicep just for working with Azure resources, which is a little less complex than JSON.

- ☐ From the command line or from the PowerShell interface, use the following command to install Bicep:

```
az bicep install
```

- ☐ Once installed, use the following command to check the version:

```
az bicep version
```

- ☐ Use the following command to upgrade Bicep, if necessary:

```
az bicep upgrade
```

- ☐ Log in to Azure via the CLI.
- ☐ You will need an editor. We'll use Visual Studio.
  - Install Visio Studio (https://code.visualstudio.com/download).
  - Install Bicep for Visual Studio (https://marketplace.visualstudio.com/items?itemName=ms-azuretools.visualstudiobicep).
- ☐ While things are installing, read the article: Understand the structure and syntax of Bicep files (https://learn.microsoft.com/en-us/azure/azure-resource-manager/bicep/file).
- ☐ Open Visual Studio from the Windows menu, and then create a new file.
- ☐ Put the following code in the file:

```
targetScope='subscription'

param resourceGroupName string
param resourceGroupLocation string

resource newRG 'Microsoft.Resources/resourceGroups@2021-01-01' =
{
  name: resourceGroupName
  location: resourceGroupLocation
}
```

This is the basic code to create a new, empty resource group in Azure. When you deploy it, it asks you for the resource group name and location.

- ☐ Save the file to your Documents directory with the file extension: **.bicep**.

- ☐ From the command line, navigate to your Documents directory.
- ☐ Deploy the file with the `az deploy` command. You must give the location and name of the template file. It will ask for the location and name of the resource group you wish to create.

```
az deployment sub create --location eastus --template-file filename.bicep
```

- ☐ When everything has run, go to your portal and check the Resource Group main page, to see that the resource group has been created.
- ☐ Create a Bicep template for a Linux virtual machine with the following specifications:
  - OS = Any Ubuntu
  - Size = Standart_B1s – 1vcpu, 1 GiB memory
  - Resource group name = LinuxTestRG
  - VM Name = LinuxTest
  - Network Security Group Name = TestLinSG
  - No Public IP

  Note: Use the following article as a guide: Create an Ubuntu Linux virtual machine using a Bicep file (https://learn.microsoft.com/en-us/azure/virtual-machines/linux/quick-create-bicep?tabs=CLI)
- ☐ Submit a copy of the template file and a screenshot of the deployment of the template from the command line.