Lab 02

Instructions:

1. Paste all screenshots (highlighted in red) in a single Word document in the correct order
2. Name the document as **YourName-lab02**
3. Submit the document as an attachment in Bb under Labs
4. Use a WSL terminal for all activities

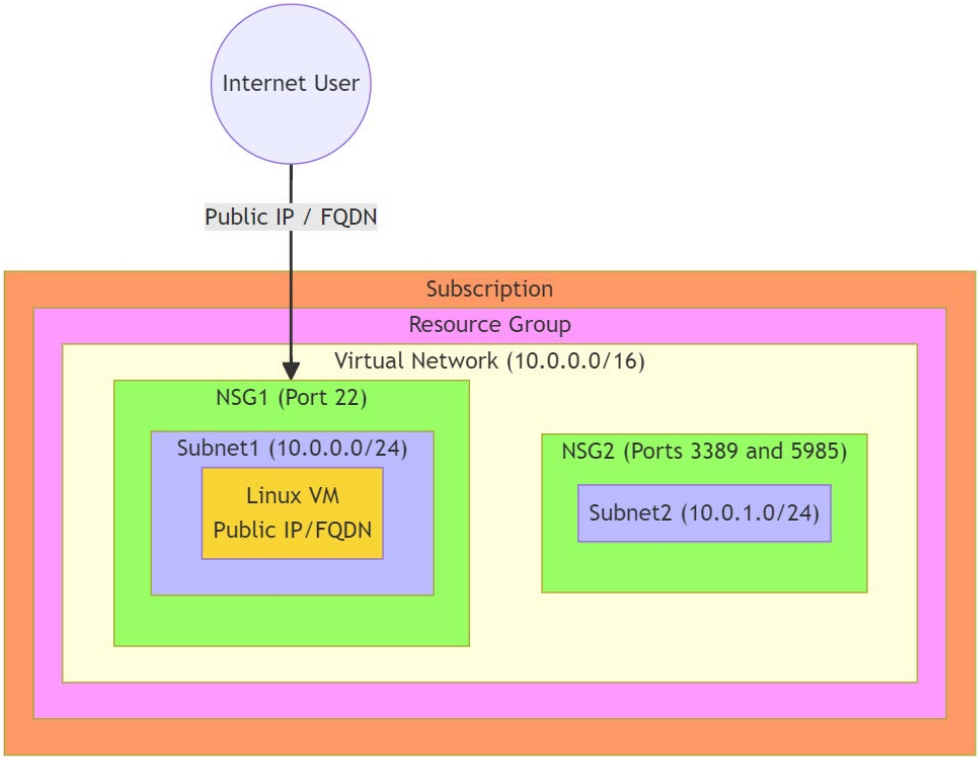
Lab submissions must be made by the due date (as indicated on the Critical Path). Each day thereafter will incur a **10%** deduction from the earned marks, up to a maximum of **3 days**. Submissions beyond this deadline will receive a grade of **Zero**.

Lab Objectives:

There are 3 sections in this lab. Each section has a different set of objectives. The sections are described below:

Section 1: Create, modify, and destroy resources
Section 2: Parametrize Section 1 configuration
Section 3: Expand the parametrized configuration from Section 2 and add a Linux virtual machine to the landscape

End-State Architecture Diagram:



WARNING

Code generated by ChatGPT or a similar generative AI tool, and copied and pasted without making the **right** modifications will result in a **ZERO** for that **entire section**.

Section 1

Objectives:

- Create a single Terraform script called network-main.tf containing the following:
  - Provider (azurerm version 3.104.0 or newer) and Terraform blocks
  - Code to build the required infrastructure
- Validate, deploy, expand, analyze, and destroy infrastructure

Important Note: Deploy all the resources in an Azure region (location) of your choice such as Australia Central, Australia East, UK West, France Central, etc.

Part 1: Prepare for the lab:

1. Create a folder called **lab02s1** under **~/automation/terraform** and change into it
2. Create an empty file called **network-main.tf**

Part 2: Codify the following in a single Terraform script:

Make sure to enclose values within double quotation marks.

3. Open **network-main.tf** file and define resource blocks as follows:
   a. Define a resource group called **network-rg** using azurerm_resource_group
   b. Define a virtual network called **network-vnet** with address space 10.0.0.0/16 using azurerm_virtual_network
   c. Define a subnet called **network-subnet1** with address space 10.0.0.0/24 in the **network-vnet** virtual network using azurerm_subnet
   d. Define a network security group called **network-nsg1** with two inbound allow TCP rules for port 22 called rule1 with priority 100 and for port 80 called rule2 with priority 200 using azurerm_network_security_group
   e. Associate the network security group **network-nsg1** to **network-subnet1** using azurerm_subnet_network_security_group_association

SCREENSHOT of the entire Terraform script

Part 3: Initialize Terraform:

4. Log in to Azure using the **az login** command
5. Initialize Terraform and download plug-ins as required (**terraform init**)

Part 4: Format and validate configuration:

6. Format the configuration to look neat and clean (**terraform fmt**)
7. Validate the configuration to ensure there are no errors or typos in the file (**terraform validate**)
8. Fix any issues in the **network-main.tf** file if reported
9. Re-run the validation until no errors are reported (**terraform validate**) **SCREENSHOT**

Part 5: Run simulation:

10. Perform a dry run (**terraform plan**)
11. Review output and ensure all configuration is as per requirements. Observe the resources with +, -, or -/+ signs.
12. Fix any issues in the **network-main.tf** file if reported
13. Redo the dry run until no errors are reported (**terraform plan**)

Part 6: Deploy infrastructure:

14. Deploy the infrastructure and monitor progress (**terraform apply**)
15. View the content of terraform.tfstate file (**tail -20 terraform.tfstate**) **SCREENSHOT**

Part 7: Get information from Terraform state:

16. View and analyze state information (**terraform state list | nl** and **terraform show**) **SCREENSHOT**

PS: The **terraform state list | nl** command should show **5** rows in the output.

Part 8: Confirm resource creation in Azure:

17. Log in to the Azure Portal. Navigate to the resource group and confirm all resources exist as per the specifications. **SCREENSHOT**

Part 9: Expand the network-main.tf script and add code to perform the following:

18. Add another subnet to the **network-vnet** virtual network called **network-subnet2** with address space 10.0.1.0/24 using azurerm_subnet
19. Define a network security group called **network-nsg2** with two inbound allow TCP rules for port 3389 called rule1 with priority 100 and for port 5985 called rule2 with priority 200 using azurerm_network_security_group

20. Associate the network security group **network-nsg2** to **network-subnet2** using [azurerm_subnet_network_security_group_association]()

SCREENSHOT of the added code

Part 10: Validate configuration:

21. Validate configuration to ensure there are no errors or typos in the file (**terraform validate**)
22. Fix any issues in the **network-main.tf** file if reported
23. Re-run the validation until no errors are reported (**terraform validate**) **SCREENSHOT**

Part 11: Run simulation:

24. Perform a dry run (**terraform plan**)
25. Observe output closely. Observe the resources with +, -, or -/+ signs.
26. Fix any issues in the **network-main.tf** file if reported
27. Redo the dry run until no errors are reported (**terraform plan**) **SCREENSHOT**

Part 12: Deploy infrastructure if no errors were reported or found:

28. Deploy the infrastructure and monitor progress (**terraform apply**)
29. View Terraform log file and review the details
30. View the content of terraform.tfstate file (**tail -20 terraform.tfstate**) **SCREENSHOT**

Part 13: Get information from Terraform state:

31. View and analyze state information. Observe differences. (**terraform state list | nl**) **SCREENSHOT**

PS: The **terraform state list | nl** command should show **8** rows in the output.

Part 14: Confirm resource creation in Azure:

32. Log in to the Azure Portal. Navigate to the resource group and confirm all resources created as per the specifications. **SCREENSHOT**

Part 15: Destroy all resources and verify:

33. Destroy all the resources (**terraform destroy**)
34. Verify deletion (**terraform state list | nl** and **terraform show**) **SCREENSHOT**
35. View the content of terraform.tfstate file (**cat terraform.tfstate**) **SCREENSHOT**

========================== End of Section 1 ====================================

Section 2

Objectives:

- Move resource values to a separate file as variable blocks
- Update main Terraform file to use variables
- Validate, deploy, expand, analyze, and destroy infrastructure

Part 1: Prepare for the lab:

1. Create a folder called **lab02s2** under **automation/terraform**
2. Copy **network-main.tf** file from **lab02s1** to **lab02s2** directory
3. Change into **lab02s2**
4. Create an empty file called **network-vars.tf**
5. Create an empty file called **providers.tf**

Part 2: Update network-vars.tf file:

6. Copy provider and Terraform blocks from **network-main.tf** to **providers.tf**
7. Open **network-vars.tf** file and define variable blocks as follows. Make sure to enclose values within double quotation marks.
    a. One block for resource group name
    b. One block for location
    c. One block for virtual network name
    d. One block for virtual network address space
    e. Two blocks for subnet names (one per subnet)
    f. Two blocks for subnet address spaces (one per address space)
    g. Two blocks for network security groups (one per network security group)

SCREENSHOT of network-vars.tf

Part 3: Update network-main.tf file:

8. Open **network-main.tf** file and update as follows:
    a. Remove provider and Terraform blocks
    b. Update all the resource blocks to source values from **network-vars.tf**

SCREENSHOT of network-main.tf

Part 4: Initialize Terraform:

9. Initialize Terraform to download plug-ins as required (**terraform init**)

Part 5: Validate configuration:

10. Validate the configuration to ensure there are no errors or typos in the file (**terraform validate**)
11. Fix any issues in the **network-main.tf** and/or **network-vars.tf** files if reported
12. Re-run the validation until no errors are reported (**terraform validate**) **SCREENSHOT**

Part 6: Run simulation:

13. Perform a dry run (**terraform plan**)

14. Review output and ensure all configuration is as per requirements. Observe the resources with +, -, or -/+ signs.
15. Fix any issues in the **network-main.tf** and/or **network-vars.tf** files if reported
16. Redo the dry run until no errors are reported (**terraform plan**)

Part 7: Deploy infrastructure:

17. Deploy the infrastructure and monitor progress (**terraform apply**)
18. View Terraform log file and review details
19. View the content of terraform.tfstate file (**tail -20 terraform.tfstate**) **SCREENSHOT**

Part 8: Get information from Terraform state:

20. View and analyze state information (**terraform state list | nl** and **terraform show**) **SCREENSHOT**

PS: The **terraform state list | nl** command should show **8** rows in the output.

Part 9: Confirm resource creation in Azure:

21. Log in to the Azure Portal. Navigate to the resource group and confirm all resources created as per the specifications. **SCREENSHOT**

Part 10: Destroy all resources and verify:

22. Destroy all the resources (**terraform destroy**)
23. Verify deletion (**terraform state list | nl** and **terraform show**) **SCREENSHOT**
24. View the content of terraform.tfstate file (**tail terraform.tfstate**) **SCREENSHOT**

========================== End of Section 2 ==================================

Section 3

Objectives:

- Use configuration from Section 2
- Define resource and variable blocks for virtual machine resources
- Validate, deploy, expand, analyze, and destroy infrastructure

Part 1: Prepare for the lab:

1. Create a folder called **lab02s3** under **automation/terraform**
2. Copy **network-main.tf**, **network-vars.tf**, and **providers.tf** files from **lab02s2** to **lab02s3** directory (note: do not make any modifications to these files)
3. Change into **lab02s3**
4. Create two empty files called **vmlinux-main.tf** and **vmlinux-vars.tf**

Part 2: Update vmlinux-vars.tf file:

5. Open **vmlinux-vars.tf** file and define variable blocks for Linux virtual machine as follows. Make sure to enclose values within double quotation marks.
   a. Name (eg: linux_name): <**HumberID>-u-vm1** # Must use lowercase letters
   b. Size: **Standard_B1s**
   c. Admin username: <**HumberID**> [from Lab01]
   d. Public key: **/home/**<**HumberID**>**/.ssh/id_rsa.pub** [from Lab01]
   e. OS disk attributes:
      i. Storage account type: **Premium_LRS**
      ii. Disk size: **32**
      iii. Caching: **ReadWrite**
   f. Ubuntu Linux OS information
      i. Publisher: **Canonical**
      ii. Offer: **UbuntuServer**
      iii. Sku: **18.04-DAILY-LTS**
      iv. Version: **latest**

SCREENSHOT of vmlinux-vars.tf

Part 3: Update vmlinux-main.tf file:

6. Open **vmlinux-main.tf** file and define resource blocks as follows:
   a. Define public IP address called **${var.linux_name}-pip** using azurerm_public_ip. Use Static IP address allocation method. Also add a DNS label that matches the VM name.
   b. Define network interface called **${var.linux_name}-nic** with IP configuration name **${var.linux_name}-ipconfig** using azurerm_network_interface. Use Dynamic IP address allocation and attach the public IP to this interface.
   c. Define virtual machine using azurerm_linux_virtual_machine. Use **${var.linux_name}-os-disk** as the OS disk name.

SCREENSHOT of vmlinux-main.tf

Note: At this point, you should have 5 Terraform files—**providers.tf**, **network-main.tf, network-vars.tf, vmlinux-main.tf, and vmlinux-vars.tf**—under **lab02s3** directory.

Part 4: Initialize Terraform:

7. Initialize Terraform to download plug-ins as required (**terraform init**)

Part 5: Validate configuration:

8. Validate the configuration to ensure there are no errors or typos in the file (**terraform validate**)
9. Fix any issues in the Terraform files if reported
10. Re-run the validation until no errors are reported (**terraform validate**) **SCREENSHOT**

Part 6: Run simulation:

11. Perform a dry run (**terraform plan**)
12. Review output and ensure all configuration is as per requirements. Observe the resources with +, -, or -/+ signs.
13. Fix any issues in the Terraform files if reported
14. Redo the dry run until no errors are reported (**terraform plan**)

Part 7: Deploy infrastructure:

15. Deploy the infrastructure and monitor progress (**terraform apply**)
16. View Terraform log file and review details

Part 8: Get information from Terraform state:

17. View and analyze state information (**terraform state list | nl** and **terraform show**) **SCREENSHOT**

PS: The **terraform state list | nl** command should show **11** rows in the output.

Part 9: Confirm resource creation in Azure:

18. Log in to the Azure Portal. Navigate to the resource group and confirm all resources created as per the specifications. **SCREENSHOT**

Part 10: Destroy all resources and verify:

19. Destroy all the resources (**terraform destroy**)
20. Verify deletion (**terraform state list | nl** and **terraform show**) **SCREENSHOT**

========================= End of Section 3 ================================