1.Create VM (Windows)

1. **Log in to Azure Portal:**
   * Go to<https://portal.azure.com>.
2. **Create a Resource:**
   * Click on **"Create a resource"** > **"Compute"** > **"Windows Virtual Machine"**.
3. **Configure Basic Settings:**
   * **Subscription**: Free trial
   * **Resource Group**: rg-vm-shreya
   * **Virtual Machine Name**: vm-windows
   * **Region**: North Europe
   * **Image**:Windows Server 2019 Datacenter
   * **Size**: Standard\_B2s
   * **Administrator Account:** set username and password
4. **Inbound Port Rules:**
   * Open necessary ports (e.g., **RDP (3389)** for remote desktop access).
   * Review the summary and click **"Create"** to deploy the VM.
5. **Access Your VM:**
   * After deployment completes, go to the **VM overview** and copy the public IP address.
   * Open **Remote Desktop (RDP)** on your local machine

2.Creating Linux VM on Azure Portal

1. **Log in to Azure Portal**
   * Visit:<https://portal.azure.com>
2. **Create a Resource**
   * Click on **"Create a resource"** → **"Compute"** → **"Linux Virtual Machine"**
3. **Configure Basic Settings**
   * **Subscription**: Free trial
   * **Resource Group**: rg-vm-shreya
   * **Virtual Machine Name**: vm-linux
   * **Region**: North Europe
   * **Image**: Ubuntu Server 20.04 LTS
   * **Size**: Standard\_B2s
4. **Administrator Account**
   * Choose either **Password** or **SSH public key** authentication:
     + If using **Password**: Set a **Username** and **Password**.
     + If using **SSH key**: Provide your public SSH key.
5. **Inbound Port Rules**
   * Open **SSH (22)** to allow remote login.
6. **Review and Deploy**
   * Click **Review + Create** → **Create** to deploy the VM.
7. **Access Your VM**
   * After deployment, copy the **public IP address** from the **VM Overview**.
   * Use an SSH client to connect:  
       
     ssh -i <path to downloaded ssh public key> <username>@<public-ip>

3. Network Security Groups (NSGs) in Azure for VMs

A **Network Security Group (NSG)** is a virtual firewall used to control inbound and outbound network traffic to Azure resources, such as Virtual Machines (VMs). NSGs operate at both the **subnet** and **network interface** levels to enhance security.

#### **Key Features:**

1. **Inbound and Outbound Rules**:
   * Control traffic to and from VMs based on protocols, IP addresses, and ports.
   * Example: Allowing RDP (3389) or SSH (22) traffic to specific VMs.
2. **Rule Components**:
   * **Source and Destination**: IP ranges or services.
   * **Port**: Port number (e.g., 80 for HTTP).
   * **Protocol**: TCP, UDP, or Any.
   * **Priority**: Lower priority rules take precedence.

#### **Example Use Case:**

* Allow **RDP** (3389) to a Windows VM for remote access.
* Allow **SSH** (22) to a Linux VM.
* Block all other inbound traffic to ensure security.

By using NSGs effectively, you ensure that your Azure VMs are protected from unauthorized access and follow best practices for network security.

4.HOST A WEBSITE ON LINUX THROUGH ANY WEB SERVER LIKE NGINX OR APACHE

I created a node.js project and hosted it on a linux vm using NGINX web server

* Created a linux vm on azure portal
* Created a nodejs project on local machine
* Moved the files to vm from local machine using scp command

scp -i ~/Downloads/myKey.pem -r my-node-app <your-username>@<VM-IP-Address>:/home/<your-username>/

* Connected to vm through powershell:

Ssh -i <path to public key .pem file> <username>@<public ip address>

* Navigate to the app directory:  
  cd /home/<your-username>/my-node-app
* Install Node.js on the VM:  
  sudo apt update

sudo apt install nodejs npm -y

* Install dependencies for the Node.js app:  
  npm install
* Run the Node.js app:  
  node server.js

5. CONFIGURE INBOUND/OUTBOUND RULES

* Create a VM
* GO to network settings
* Add a new inbound port rule as follows:

Source:MyIPAddress

Destination: Any

Service: custom

Destination port range:80

Protocol: any

Action : allow

* Save changes

Now the website hosted on vm will only be able to be visible to devices connected to the same ip address.

6. Use Custom data / User data

Difference between custom and user data:

| **Feature** | **User Data** | **Custom Data** |
| --- | --- | --- |
| **Primary Use** | Runs initialization scripts (e.g., cloud-init) at first boot to set up the VM. | Passes additional data to the VM, such as configuration files or scripts, but doesn't always guarantee execution on its own. |
| **Execution** | Typically executed by the VM's **cloud-init** or similar bootstrapping service. | Needs to be explicitly integrated (e.g., via cloud-init). Used to **configure applications** or provide non-script data. |
| **Examples** | Install software, set up users, configure network settings at startup. | Send configuration files, metadata, or instructions for internal services or apps. |

I configured the vm such that nginx download is automated on creation of vm.

* Go to advanced tab while creating vm
* Add the following code in custom data:

#cloud-config

packages:

- nginx

runcmd:

- systemctl enable nginx

- systemctl start nginx

* Create the vm
* Connect to vm through powershell
* ssh -i <path to downloaded ssh public key> <username>@<public-ip>
* No need to download the nginx server, we can directly visit the default website.