CAPSTONE PROJECT

You work as an Azure professional for a corporation. You are assigned the task of implementing the below architecture for the company's website. There are three web pages to be deployed:

1. The home page is the default page (VM2)

2. The upload page is where you can upload the files to your Azure Blob Storage (VM1)

3. The error page for 403 and 502 errors

Application Gateway has to be configured in the following manner:

1. Example.com should be pointed to the home page

2. Example.com/upload should be pointed to the upload page

3. Application Gateway's error pages should be pointed to error.html which should be hosted as a static website in Azure Containers. The error.html file is present in the GitHub repository

The term 'Example' here refers to the Traffic Manager's domain name. The client wants you to deploy them in the Central US and the West US regions such that the traffic is distributed optimally between both regions.

Storage Account has to be configured in the following manner:

1. You need to host your error.html as a static website here, and then point the application gateway's 403 and 502 errors to it.

2. Create a container named upload, this will be used by your code to upload the files.

Technical specifications for the deployments are as follows:

1. Deployments in both regions should have VMs inside VNets.

2. Clone the GitHub repo https://github.com/azcloudberg/azproject to all the VMs.

3. On VM1, please run vm1.sh this will deploy the upload page, on VM2 please run VM2.sh, this will install the home page.

4. For running the scripts, please run the following command inside the GitHub directory from the terminal. **VM1: ./vm1.sh** **VM2: ./vm2.sh**

5. After running the scripts, please edit the config.py file on VM1, and enter the details related to your storage account where the files will be uploaded.

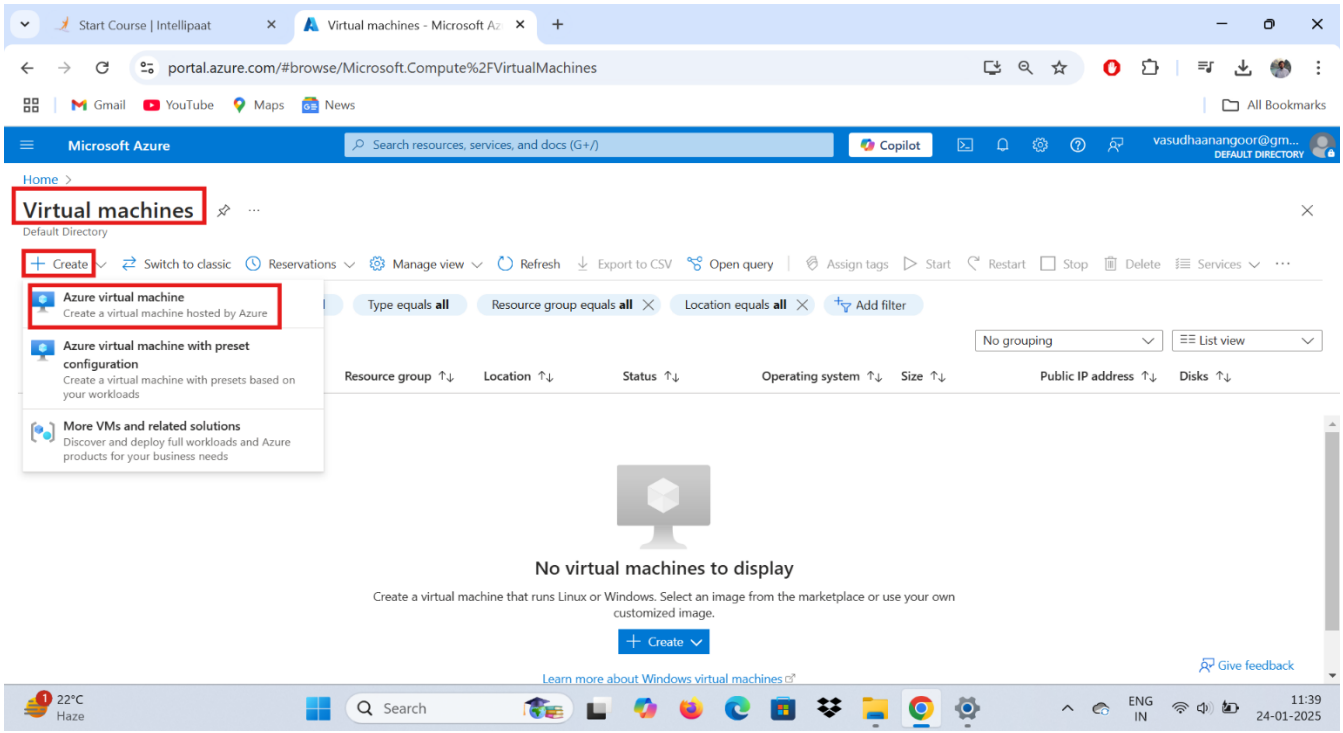
6. Once done, please run the following command: sudo python3 app.py

7. Both regions should be connected to each other using VNet-VNet Peering.

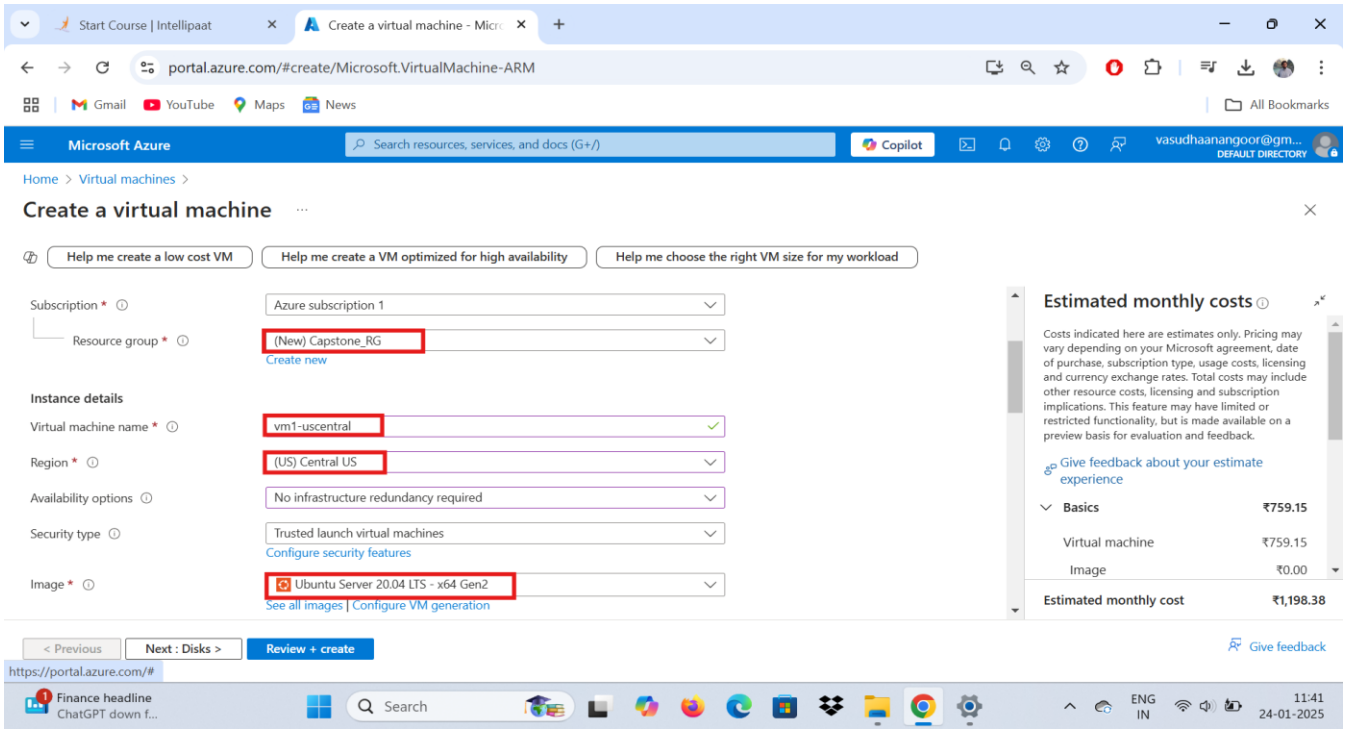
8. Finally, your Traffic Manager should be pointing to the application gateway of both the regions.

SOLUTION:

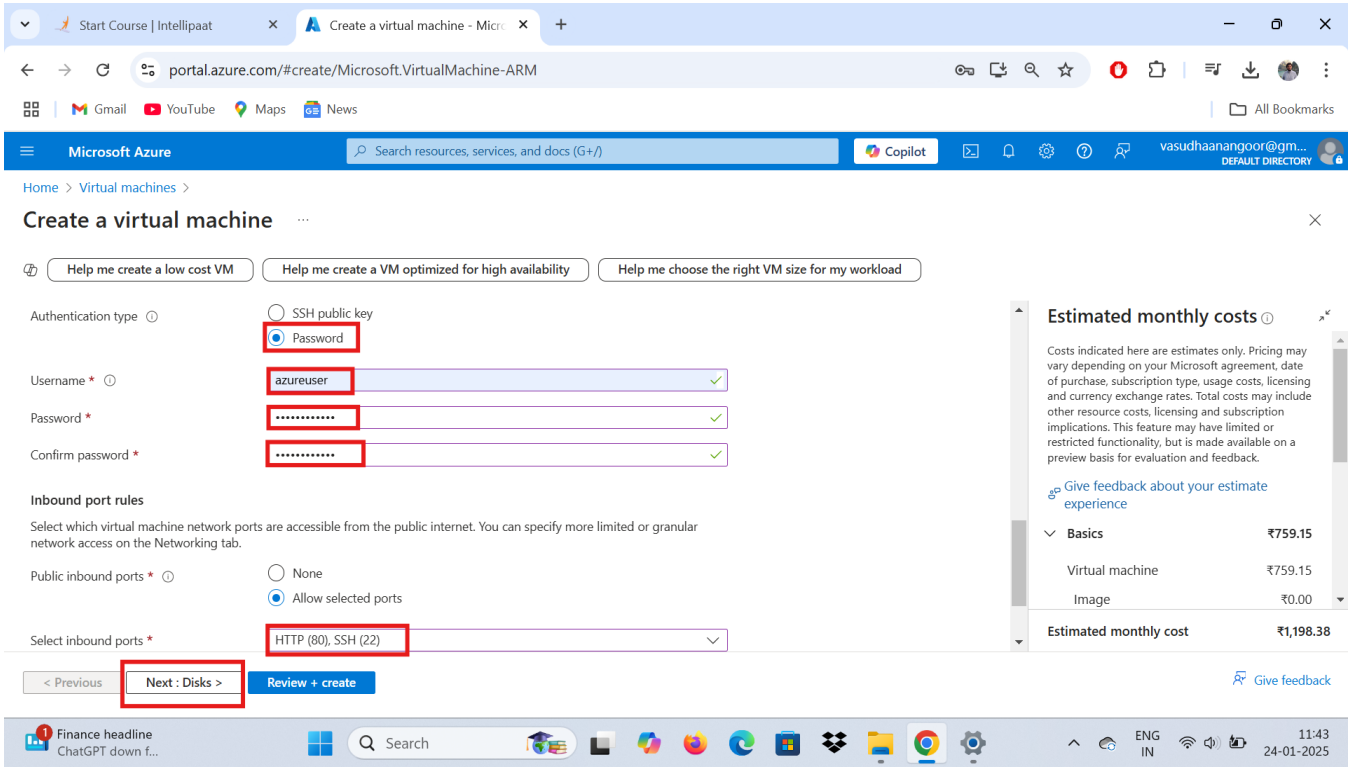
1. CREATING VIRTUAL MACHINES: Firstly, we need to create 4 virtual machines (VMs) for this project. Two VMs will be deployed in Central US and two VMs in West US. For this, type virtual machine in the search bar at the top and navigate to its console. Then click on Create Azure virtual machine.



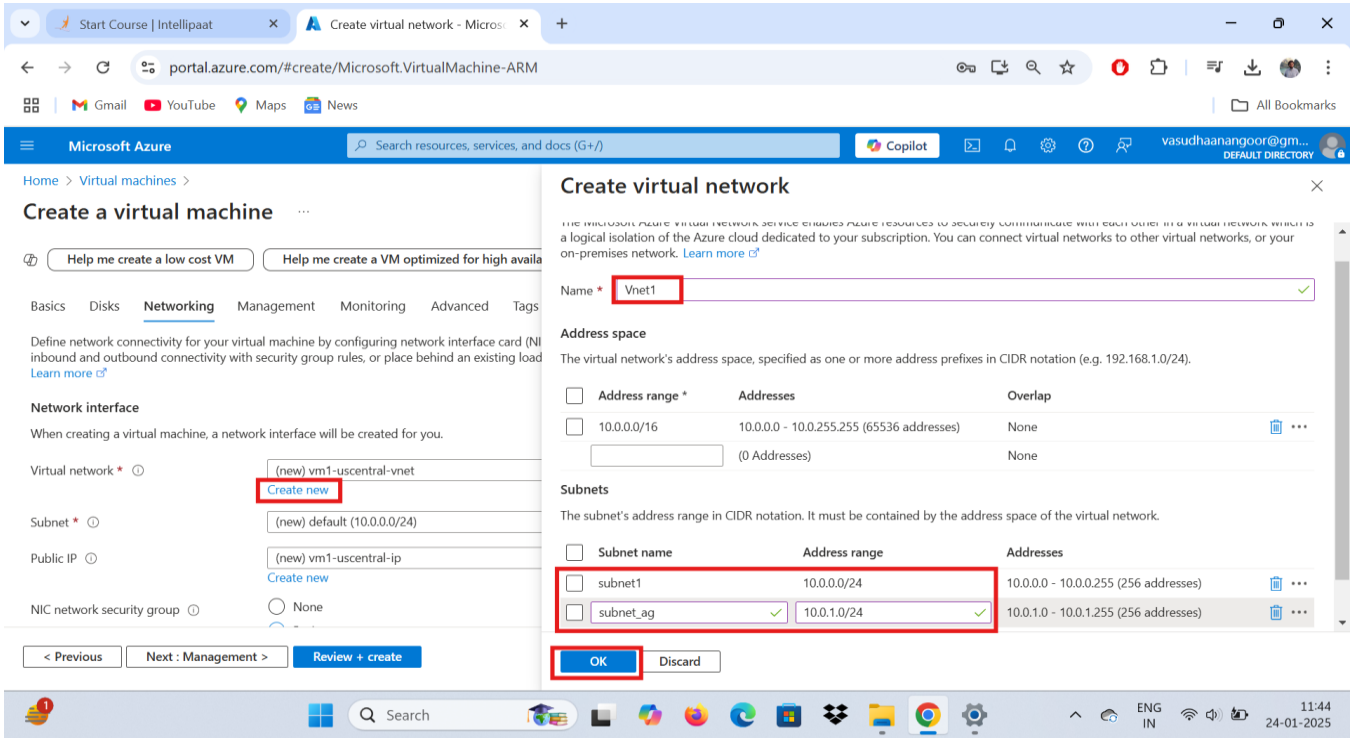
We will create a resource group named "**Capstone_RG**" . We will create a VM named "**vm1-uscentral**" in the Central US region and select the image as Ubuntu 20.04.



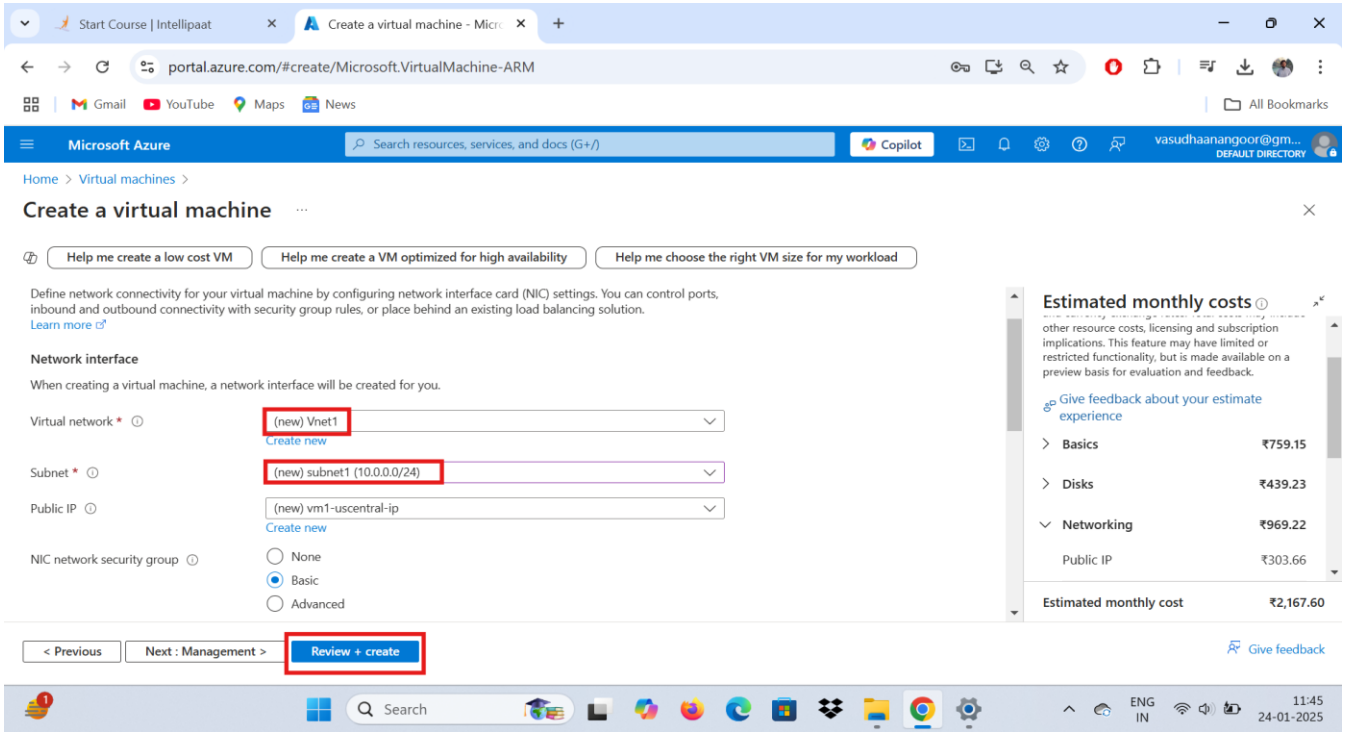
We will select the authentication type as password and provide the username along with the password. Then we will select SSH & HTTP port. After that we will click on next until we reach the networking section.



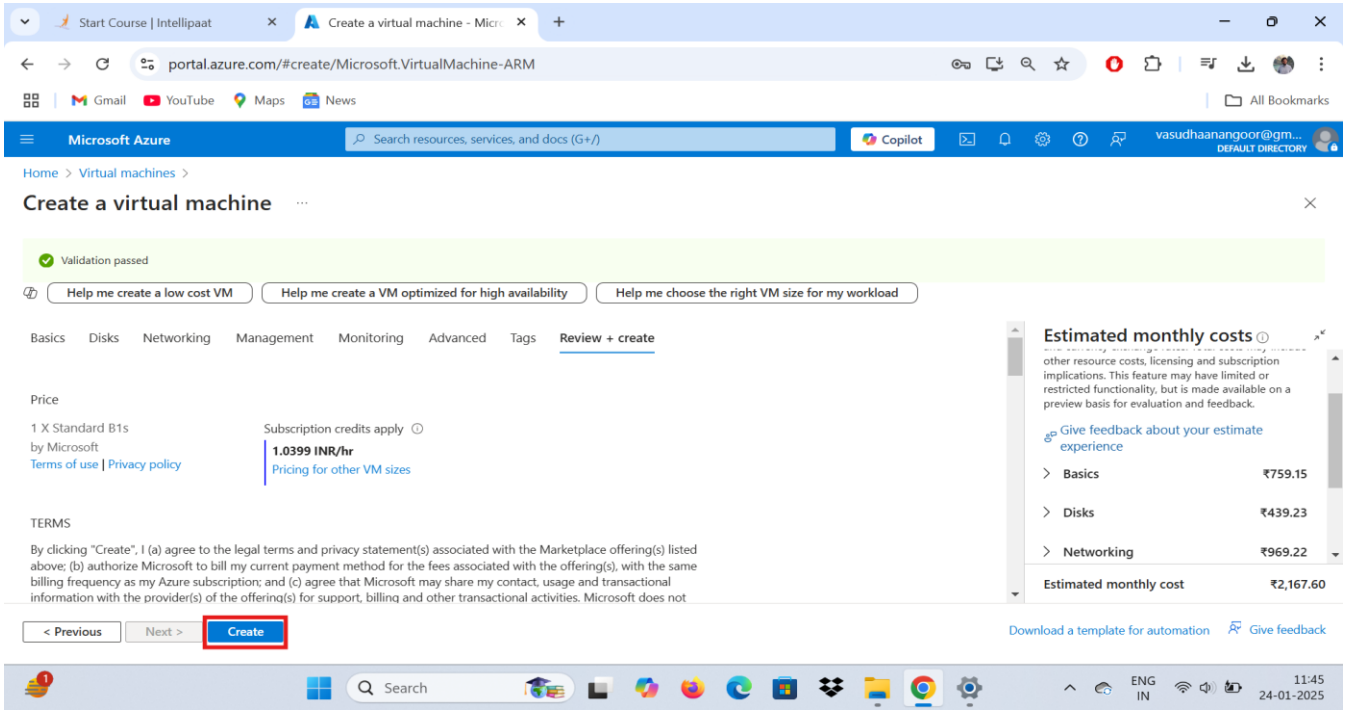
In the networking, we will create a new virtual network named “**Vnet1**” . Under subnets, we will change the default name of subnet and name it” **subnet1**”. Also, we will create a new subnet required for application gateway and name it as “**subnet_ag**” and provide the address range. After that we will click on Ok.



Then in the networking section, select the newly created virtual network and the default subnet. After that we will click on review+create.

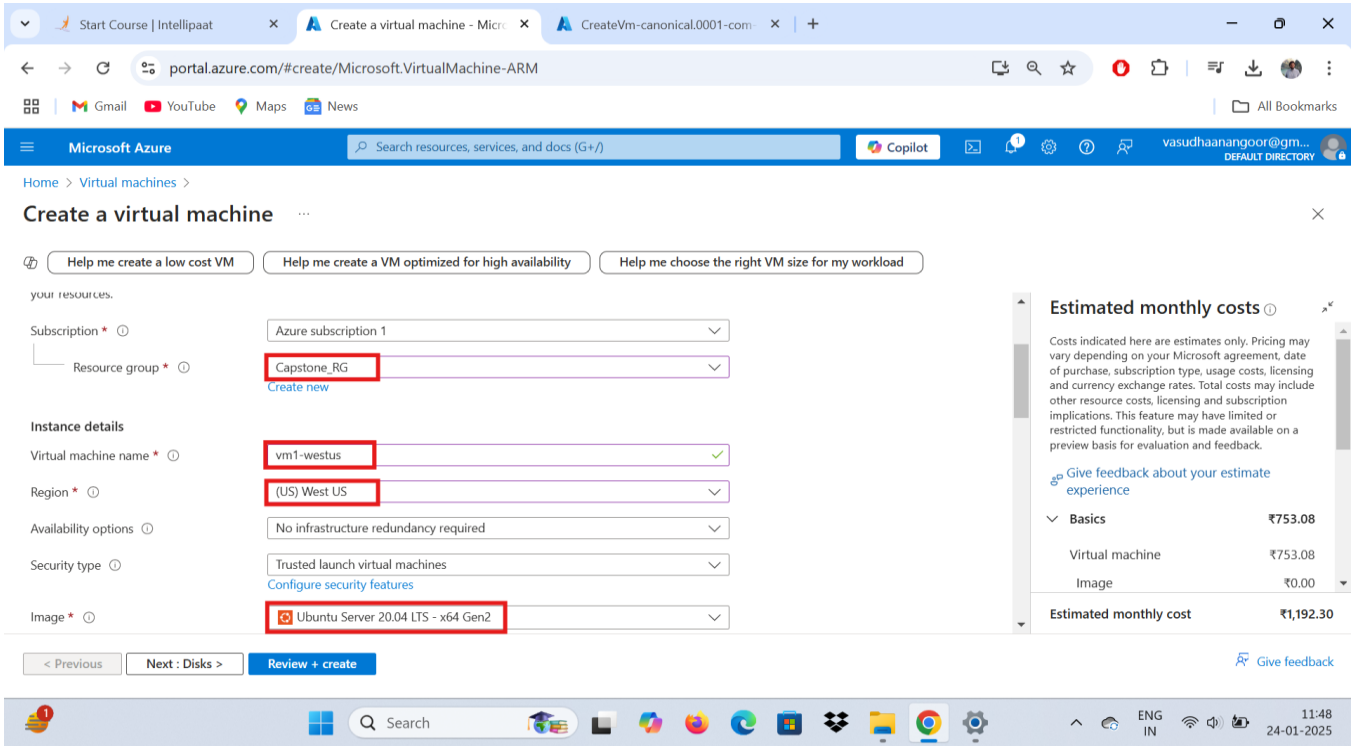


The validation will be successfully passed and we will click on Create to finally create the VM.

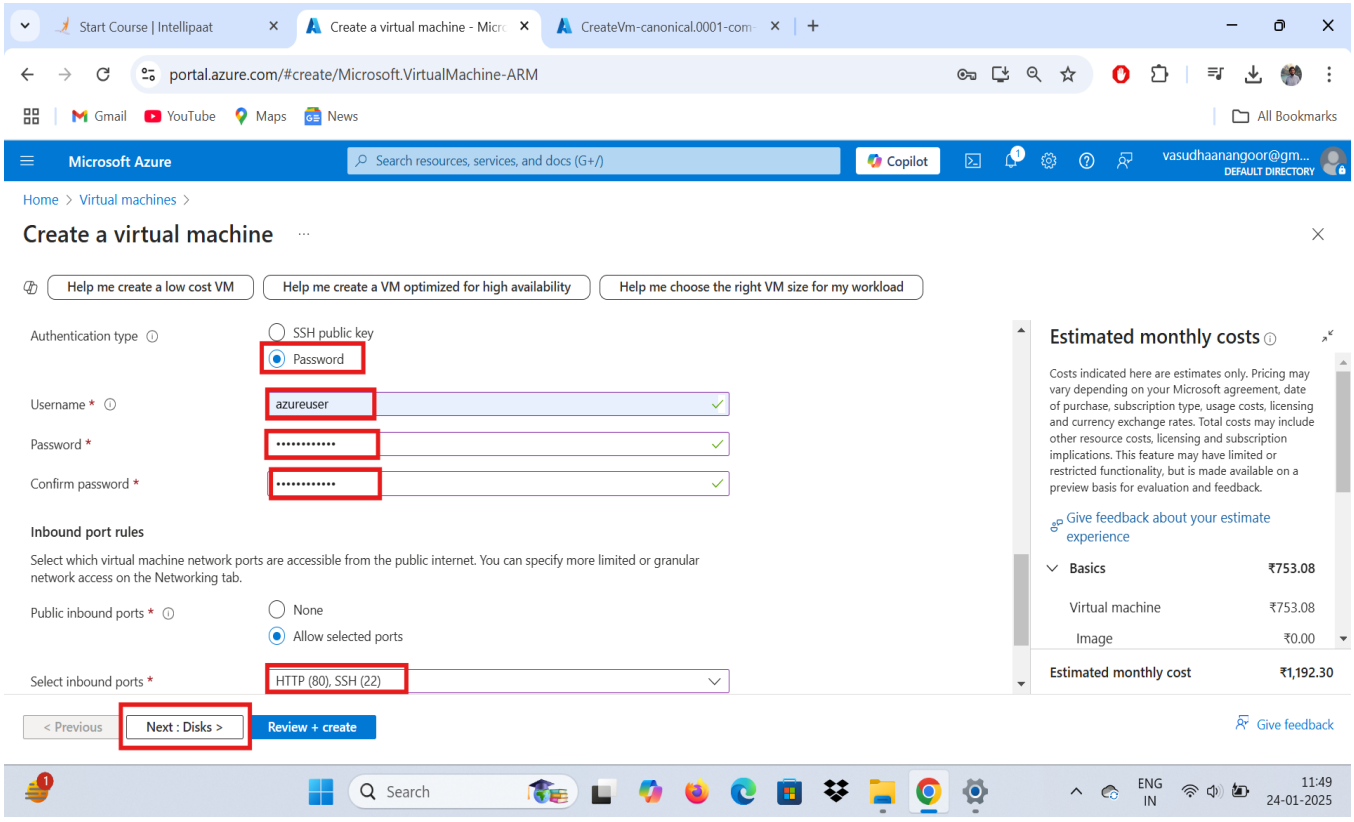


Similarly, we will create one more VM in Central Us with the name "**vm2-uscentral**", image as Ubuntu 20.04 and with all the same configurations as the first VM including the same virtual network and subnet.

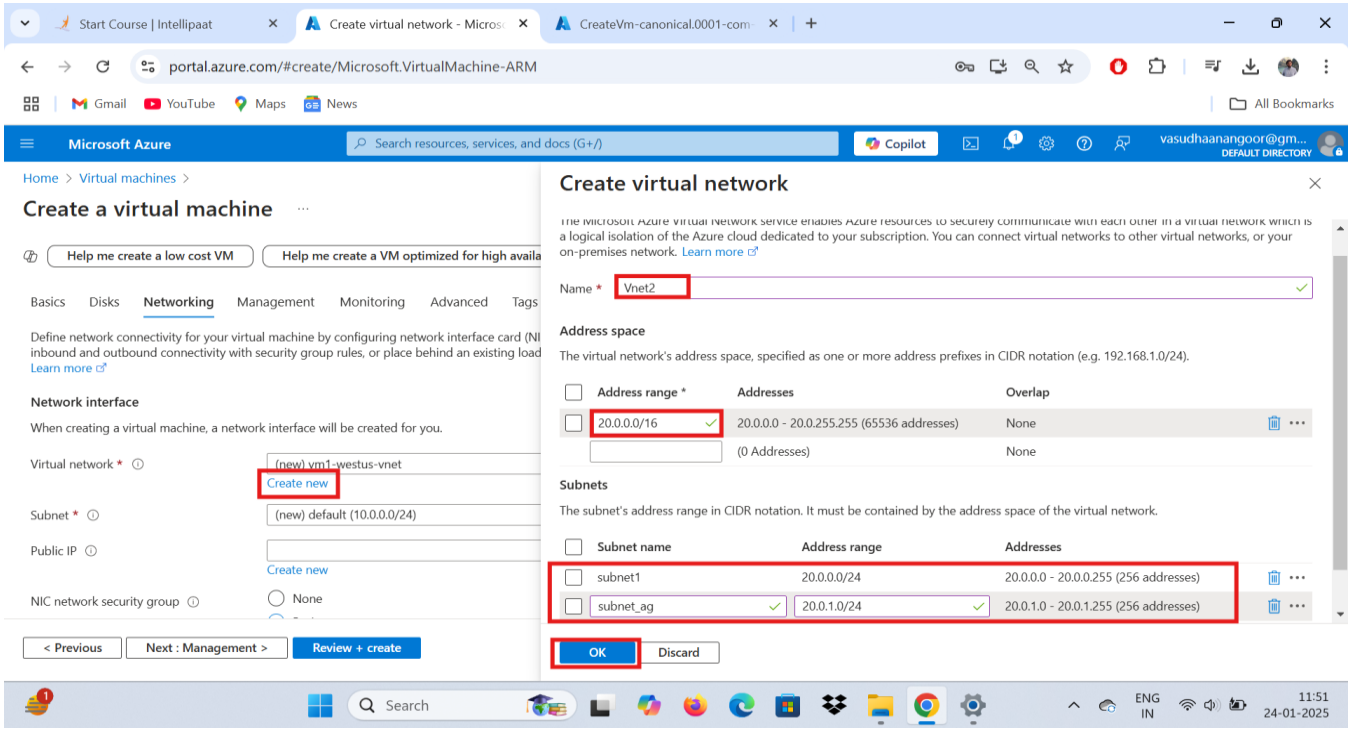
Next, we will create the third VM in West US region with the name "**vm1-westus**" ,same resource group as above and image as Ubuntu 20.04.



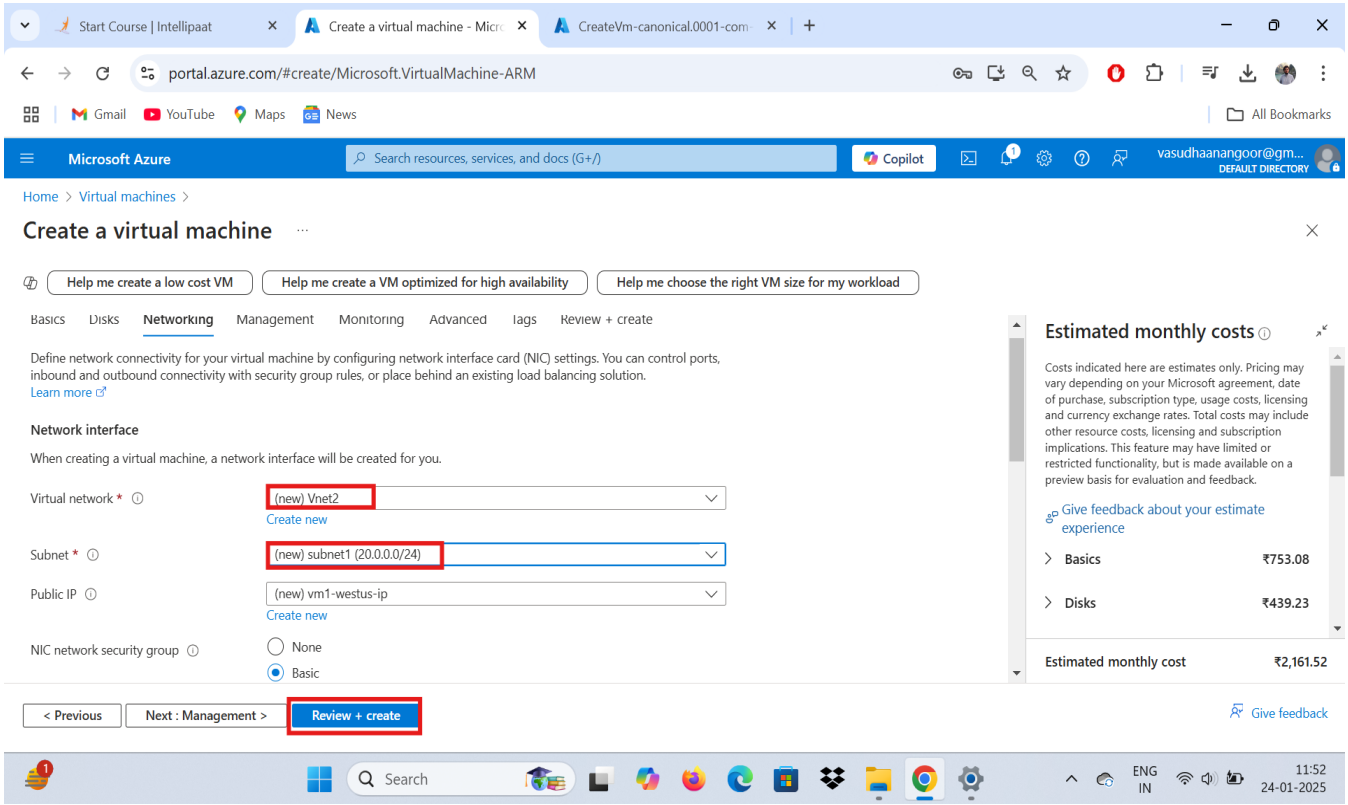
We will select the authentication type as password and provide the username along with the password. Then we will select SSH & HTTP port. After that we will click on next until we reach the networking section.



In the networking, we will create a new virtual network named “**Vnet2**” . Under subnets, we will change the default name of subnet and name it” **subnet1**”. Also, we will create a new subnet required for application gateway and name it as “**subnet_ag**” and provide the address range. After that we will click on Ok.



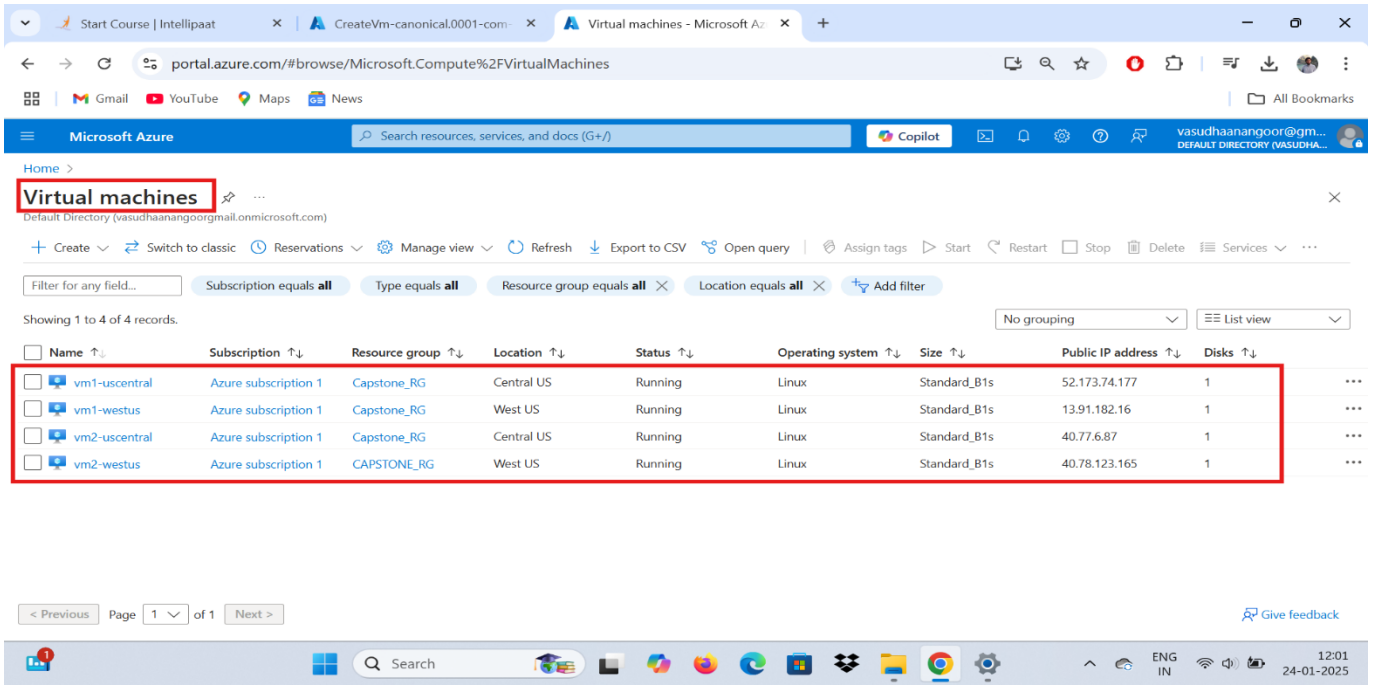
Then in the networking section, select the newly created virtual network and the default subnet. After that we will click on review+create.



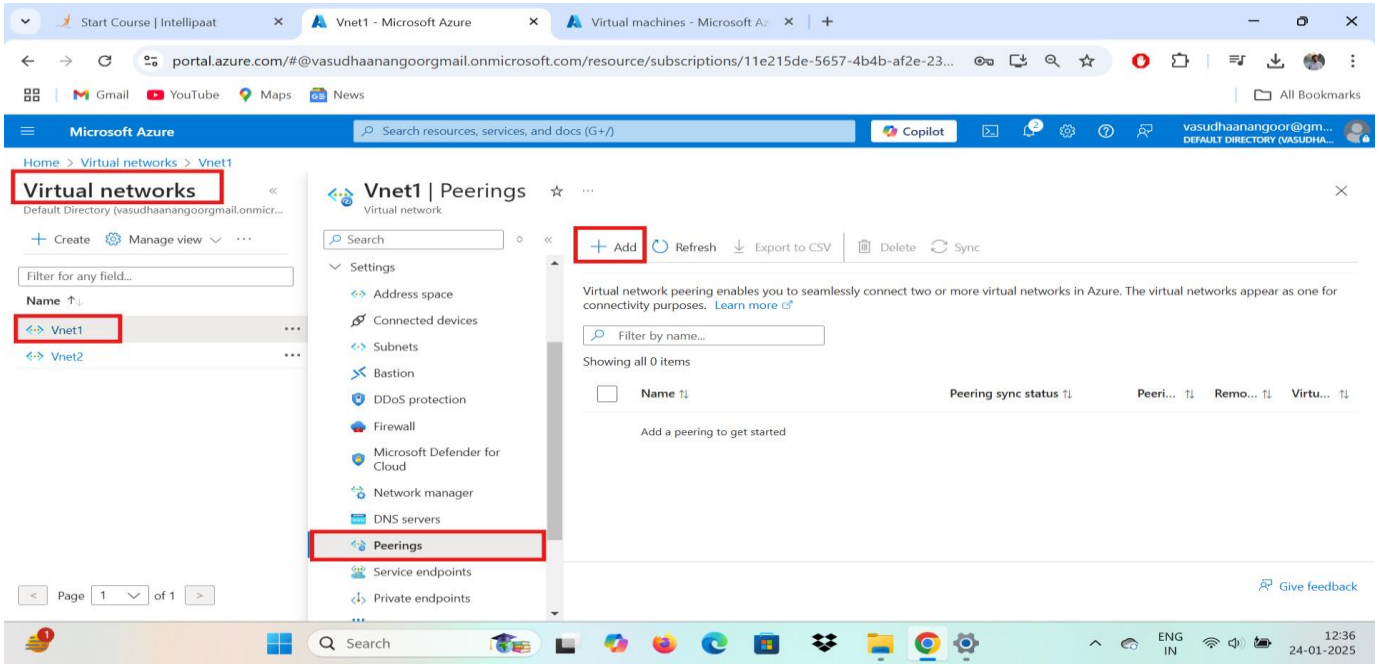
The validation will be successfully passed and we will click on Create to finally create the VM.

Similarly, we will create the fourth VM in West Us with the name "**vm2-westus**", image as Ubuntu 20.04 and with all the same configurations as the third VM including the same virtual network and subnet.

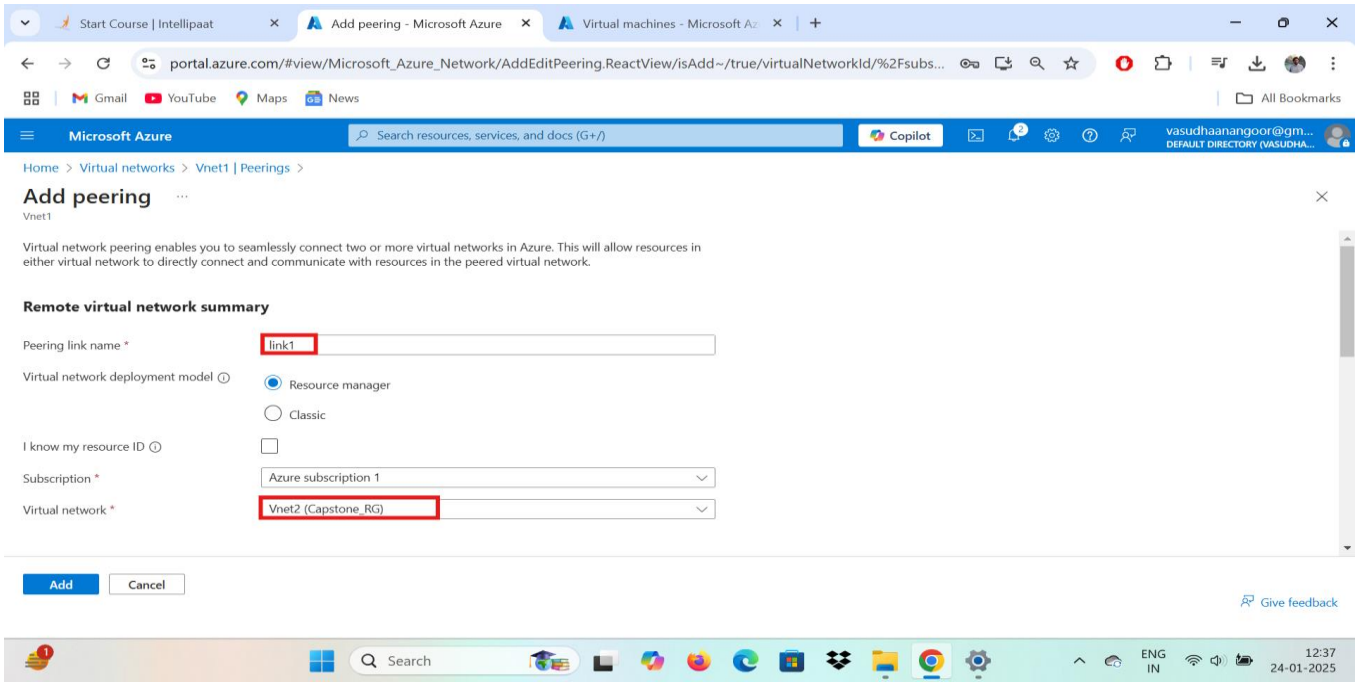
Finally, all the four VMs will be successfully listed as shown below.



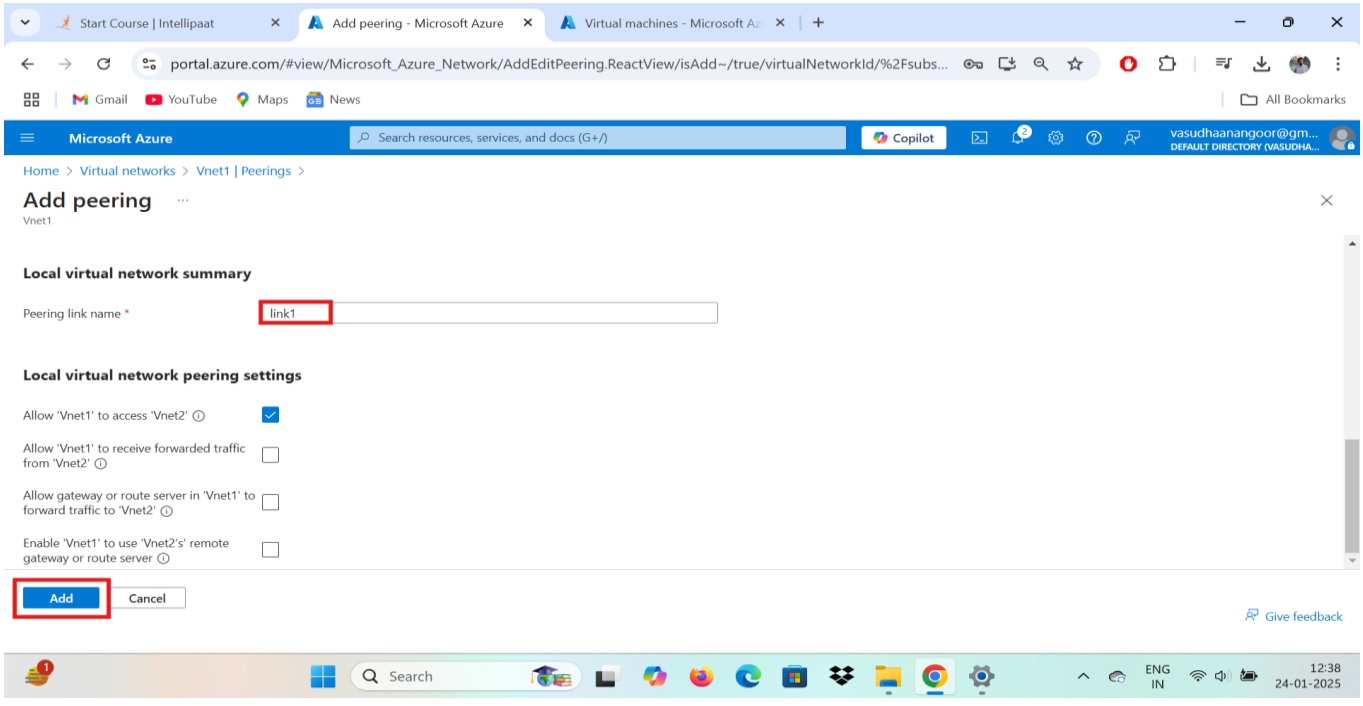
2.CREATING VNET-VNET PEERING: Next step is to create VNet-VNet peering. For this, go to Virtual network console and open Vnet1 (we can choose any one of the VMs). Under settings, click on Peerings and Add.



We will name the peering as “**link1**” and under Virtual network, select Vnet 2 .

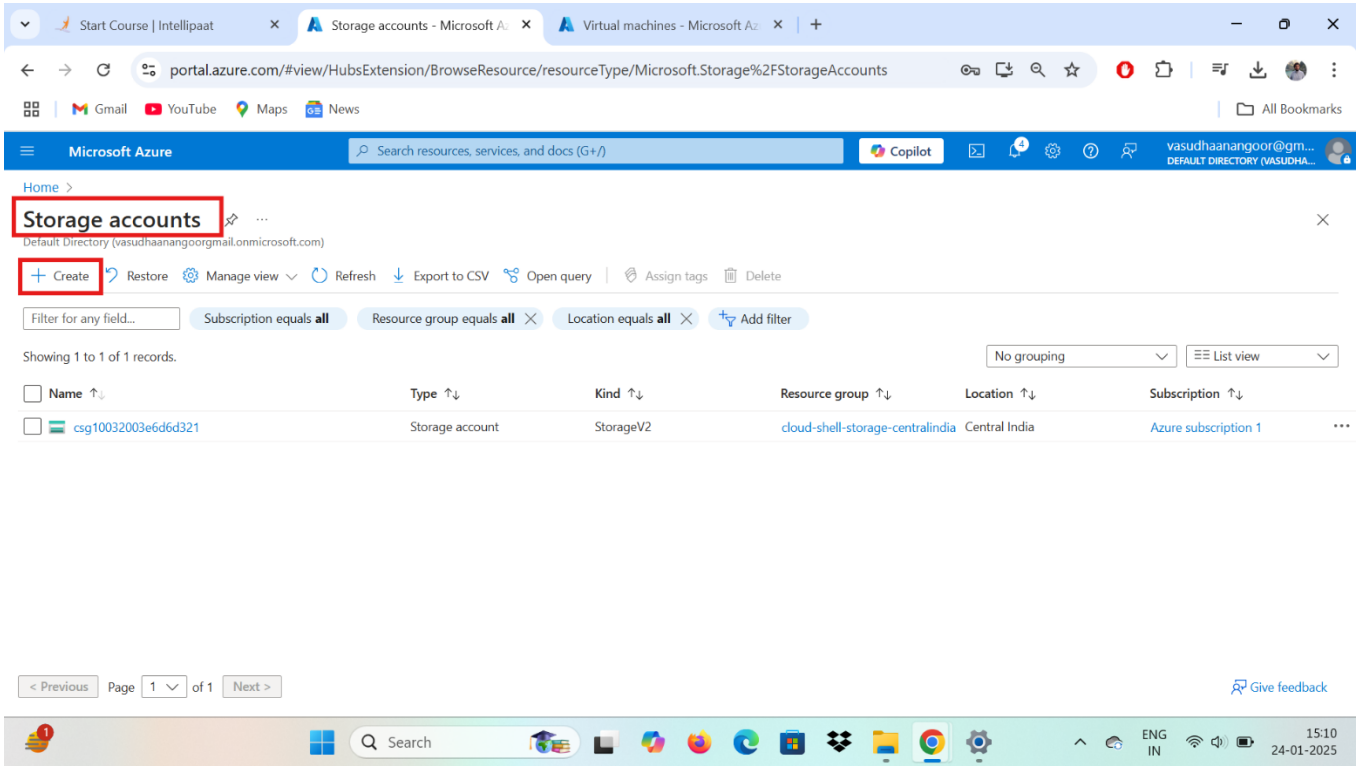


The Peering link name will also be link1. Then click on Add to create the peering.

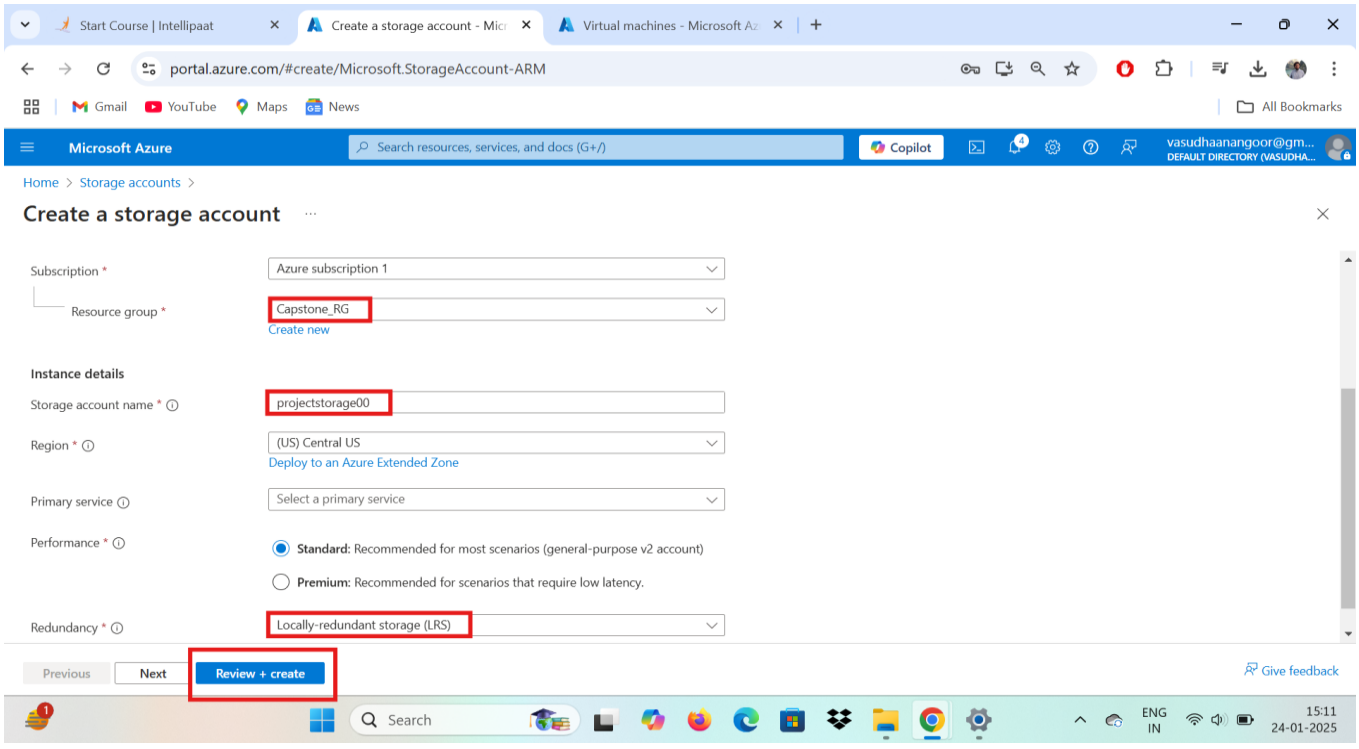


The same peering will be added by default in Vnet1 also.

3.CREATING STORAGE ACCOUNT: To create a storage account, ,we will type storage account in the search bar at the top and navigate to its console. Then click on Create.

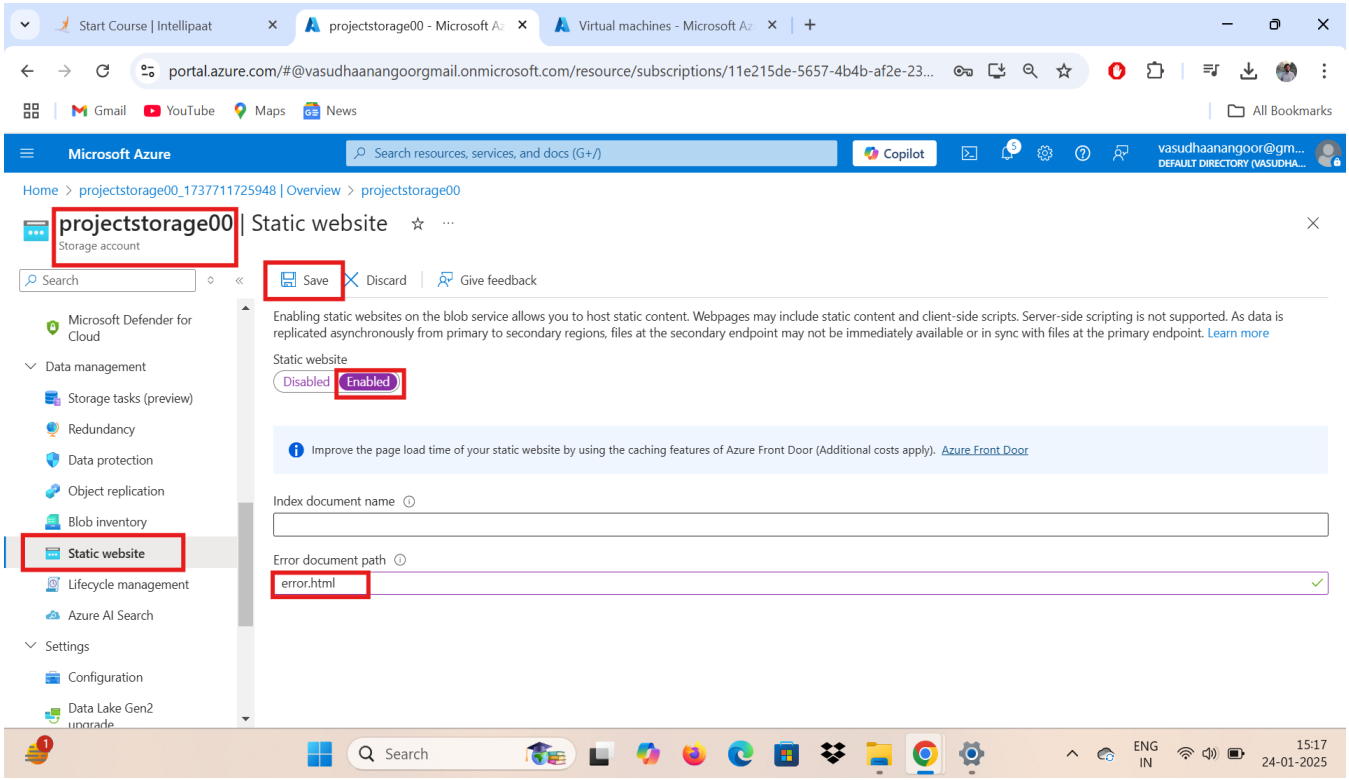


We will select the resource group and name the storage account as "**projectstorage00**", region as Central US and Redundancy as LRS. Then click on Review+create.

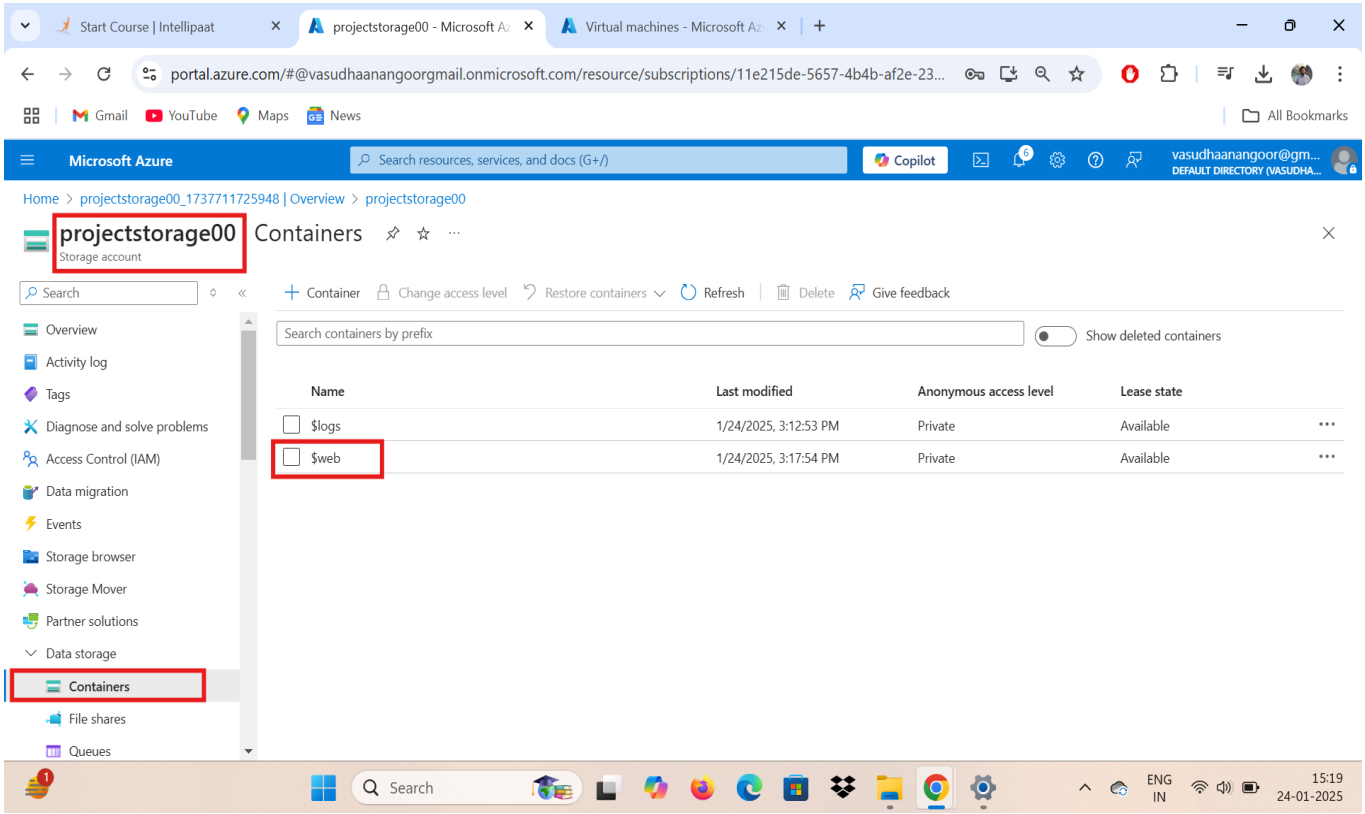


Once the validation is passed, click on create. The storage account will be successfully created.

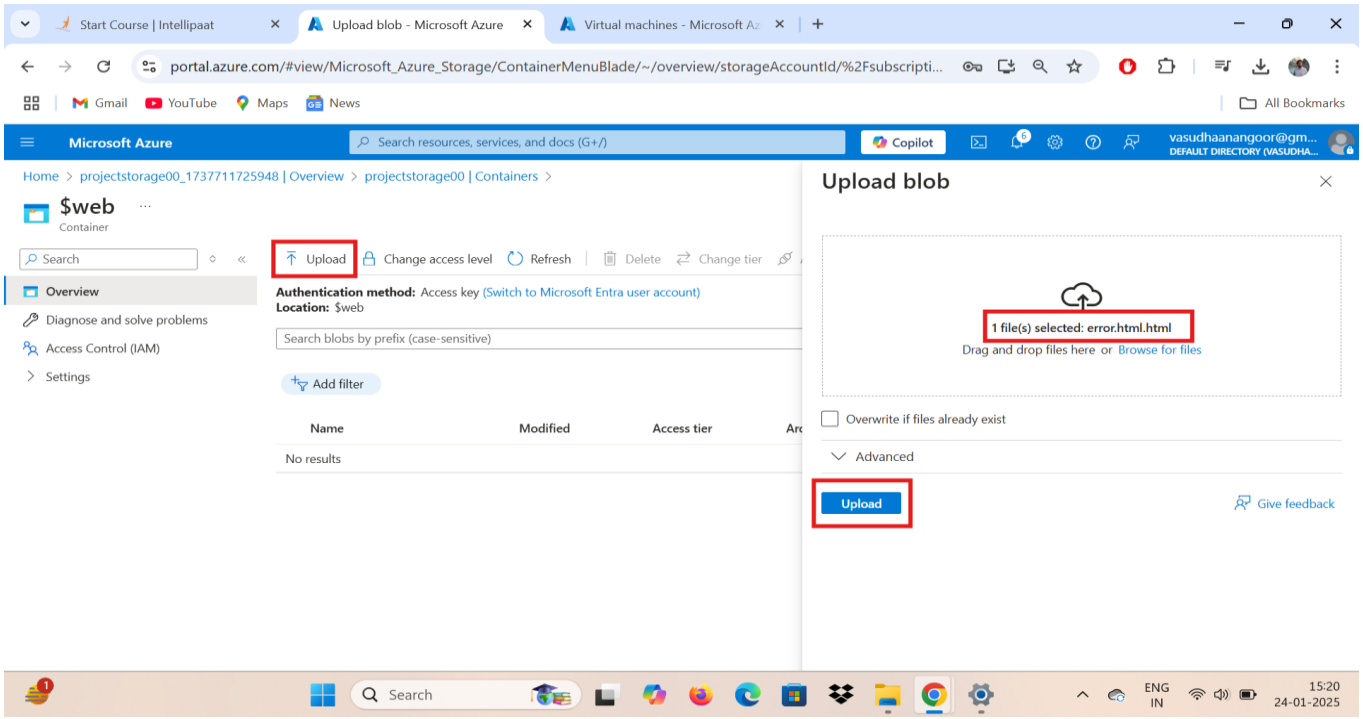
Next part is to enable the static website for the storage account. To do this, go to the storage account, click on Static website under Data management in the left-hand side. Click on Enabled and give error.html as the Error document path. Once done, click on Save.



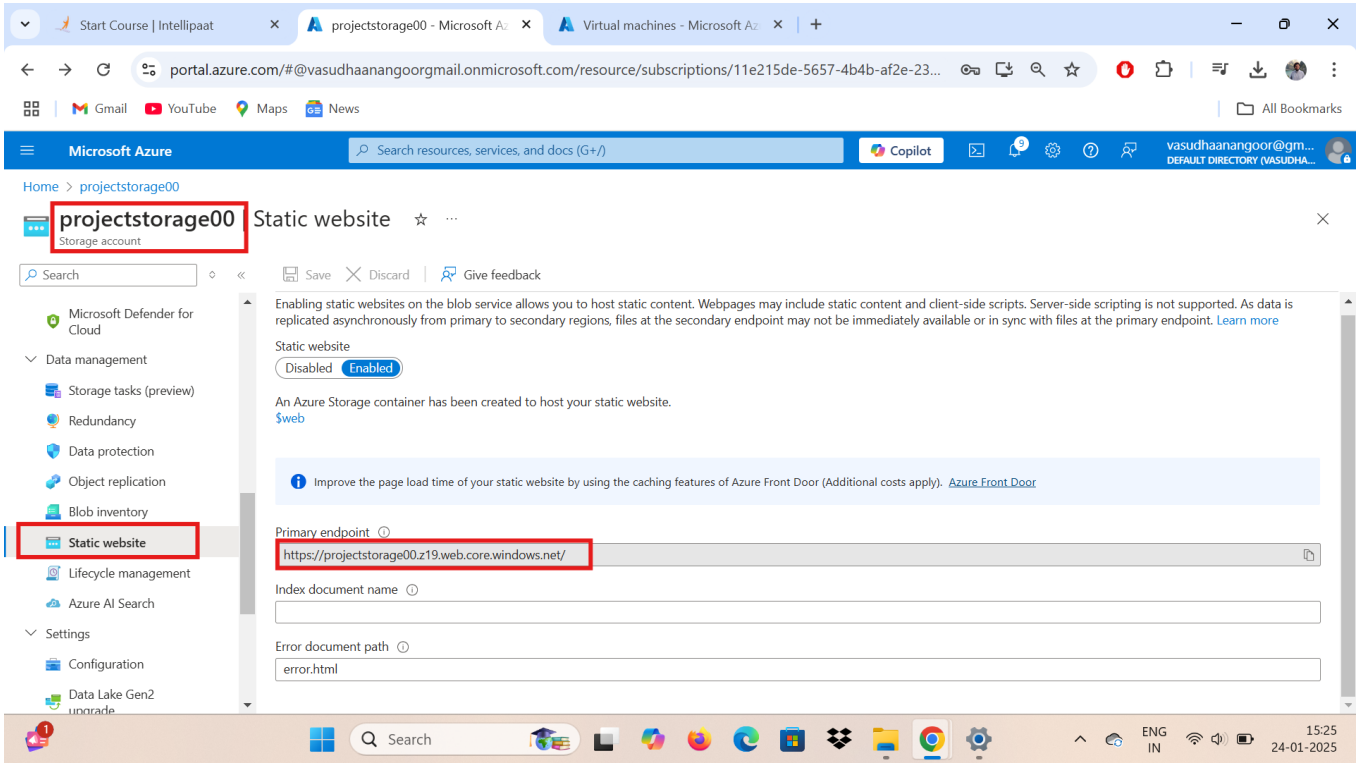
Once enabled, the default container $web will be created automatically.



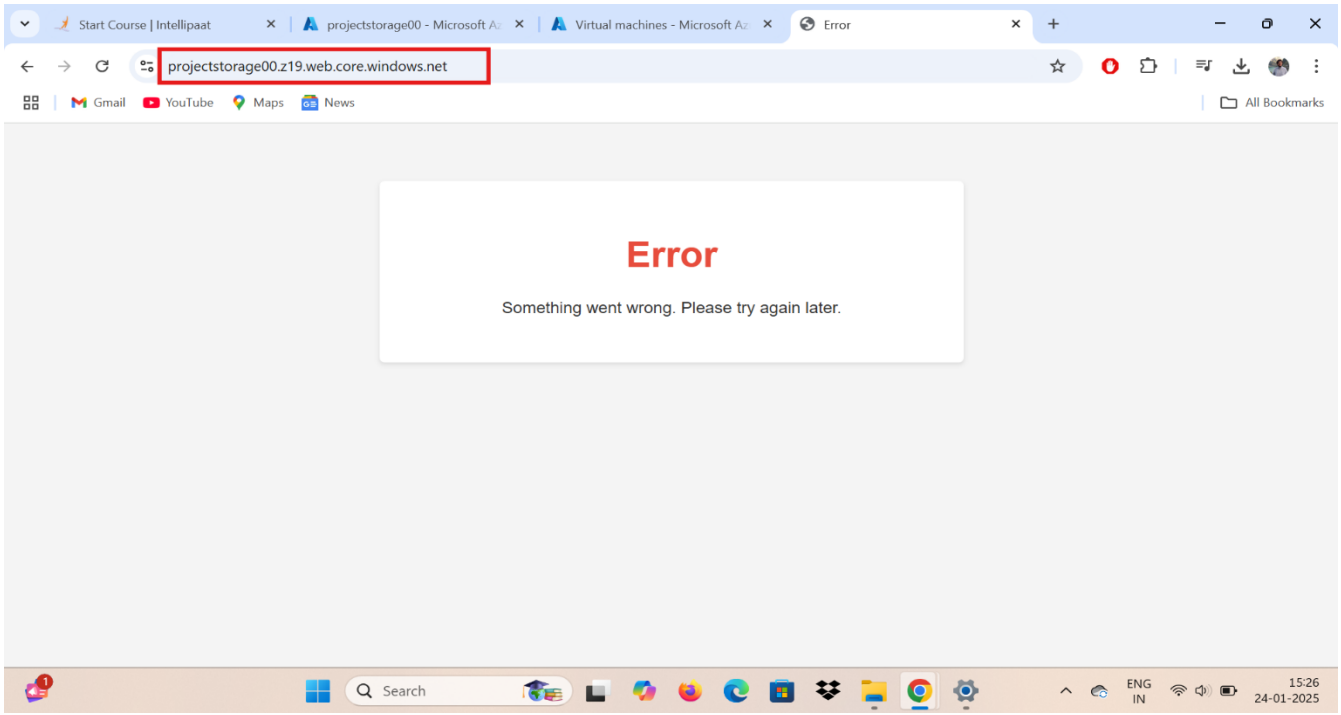
We will open the container and upload the error.html file as shown below:



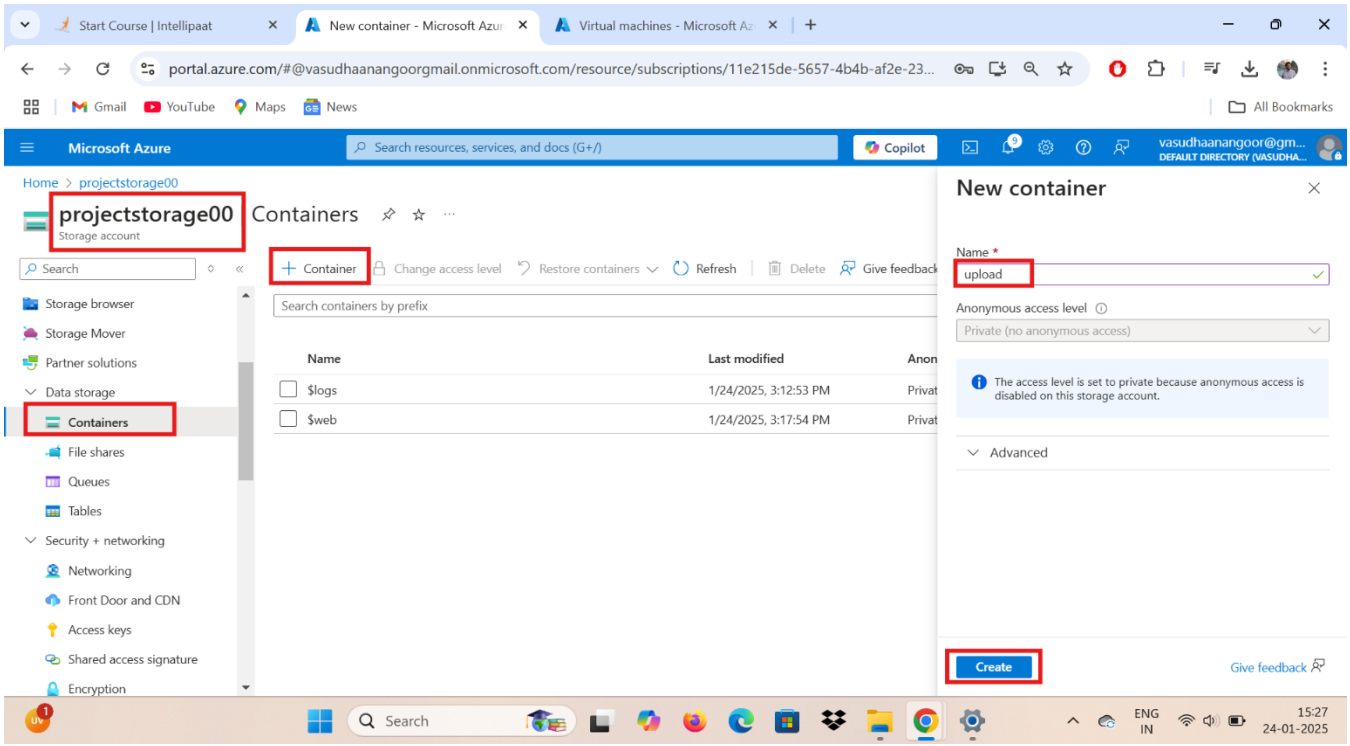
To verify it, go to the Static website and copy the Primary endpoint and paste it in the browser.



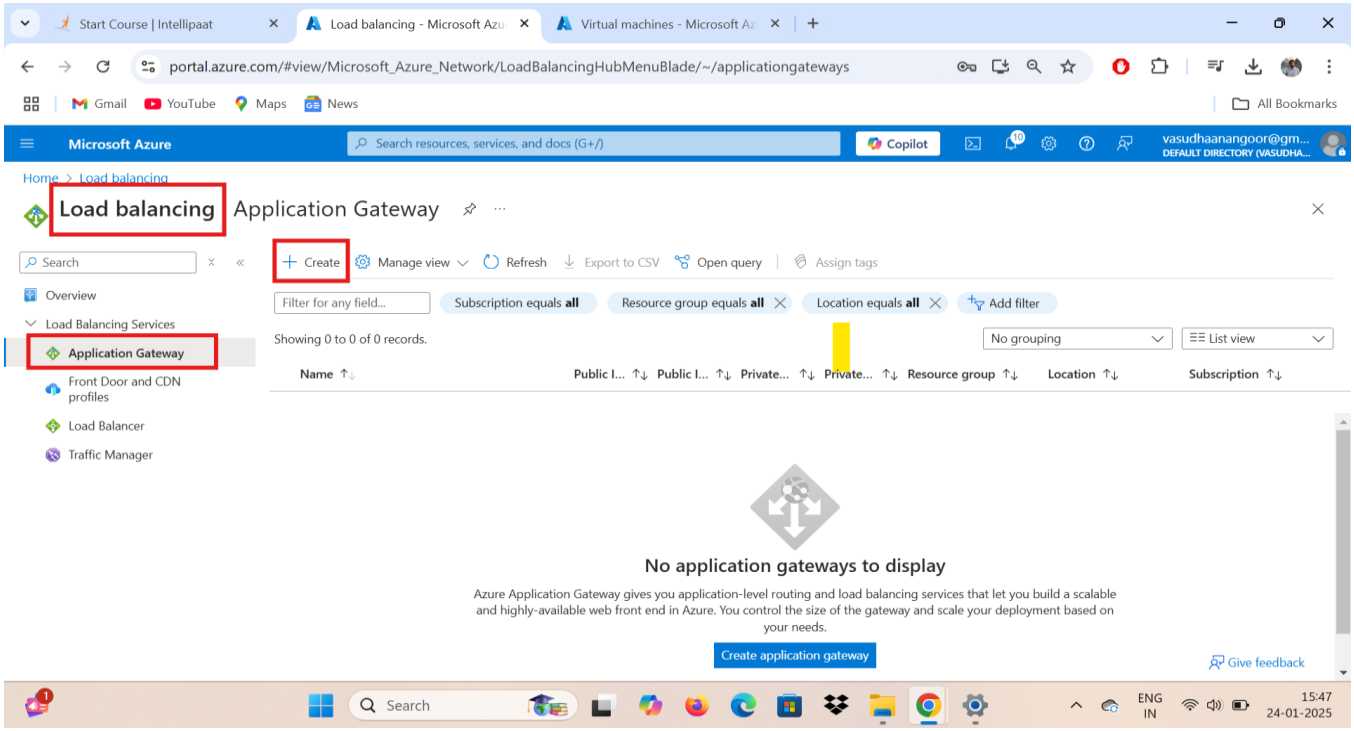
The error page will be successfully loaded as shown below:



4.CREATING UPLOAD CONTAINER: To create a container, go to the storage account and click on Containers under Data Storage. Then click on the + symbol. Give the container name as "**upload**" and click on Create.

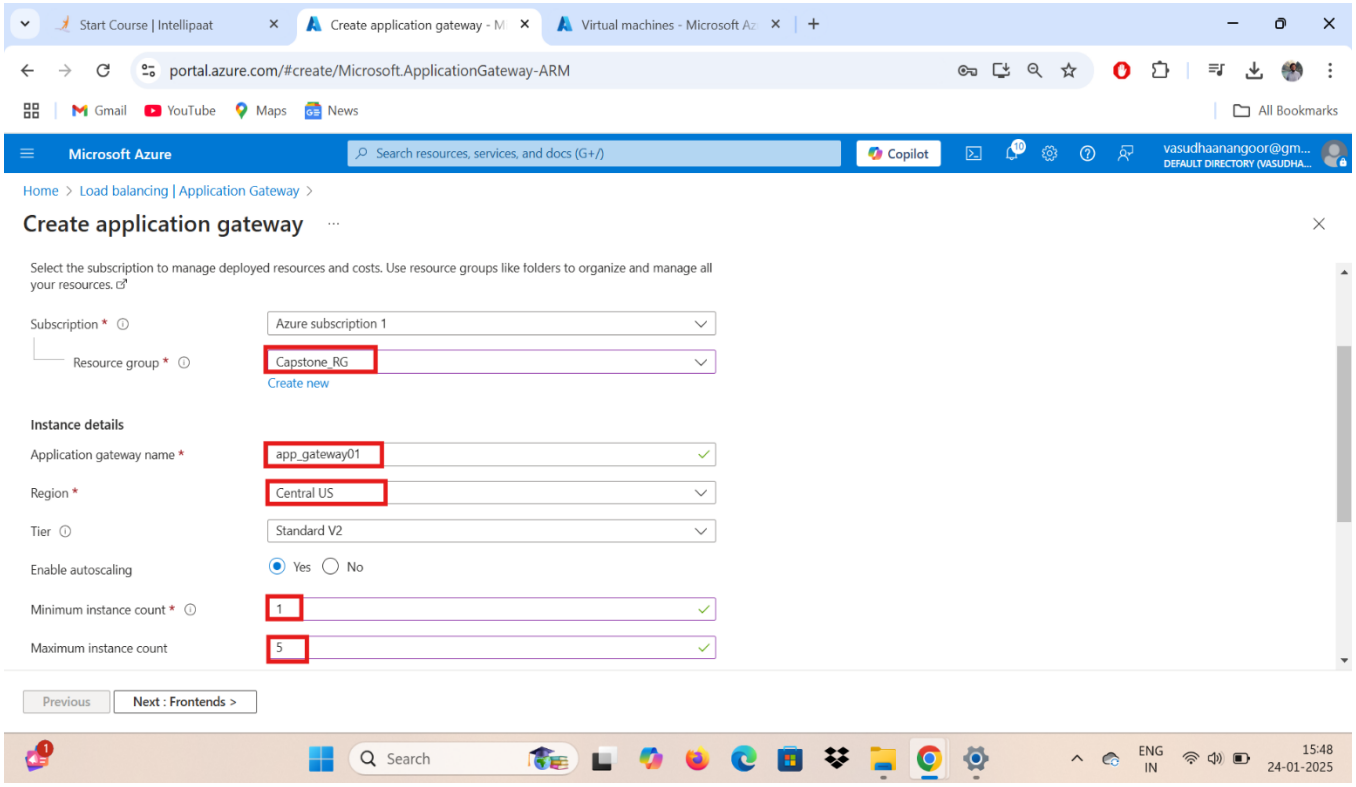


5.CREATING APPLICATION GATEWAY: We need to create Application Gateways in both the regions. For this, type Application Gateway in the search bar at the top and navigate to its console. Then click on create.

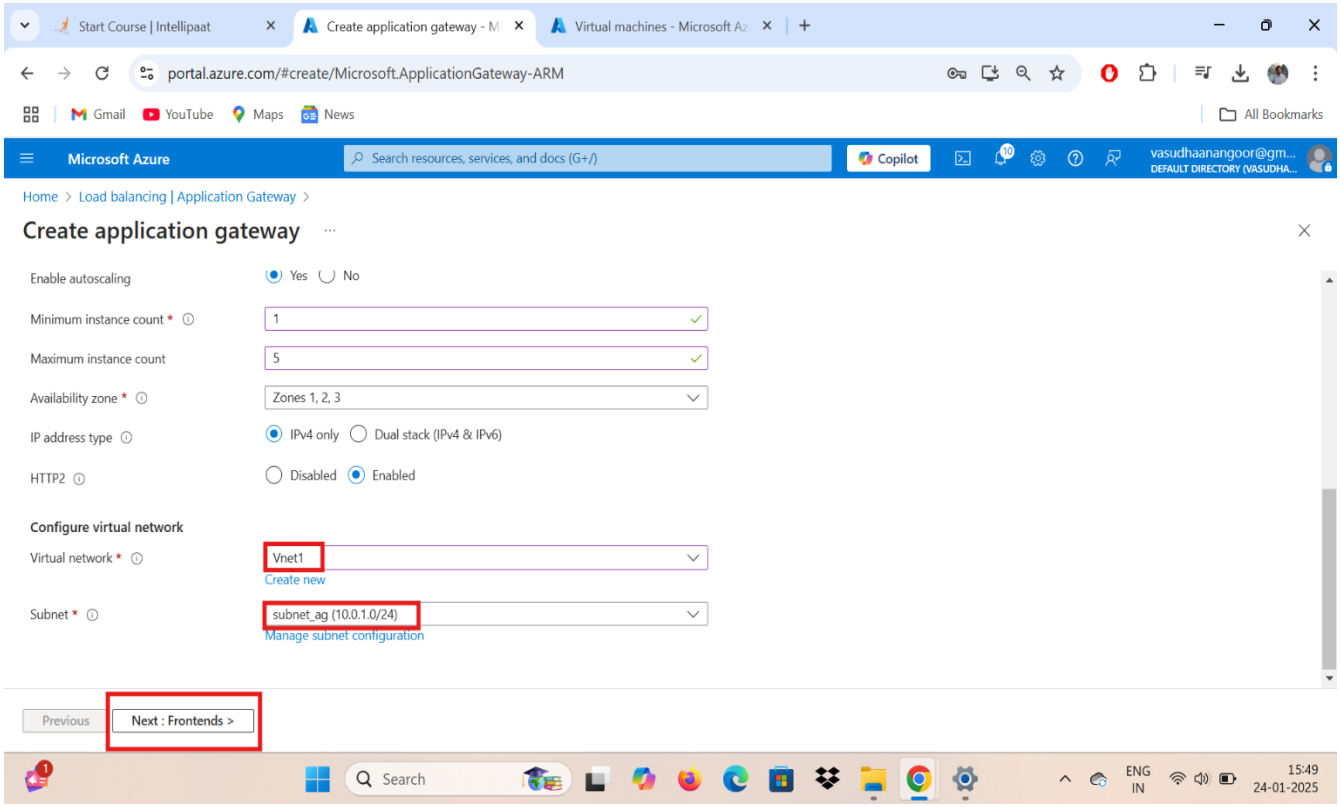


Firstly, we will create an Application Gateway for the Central Us region.

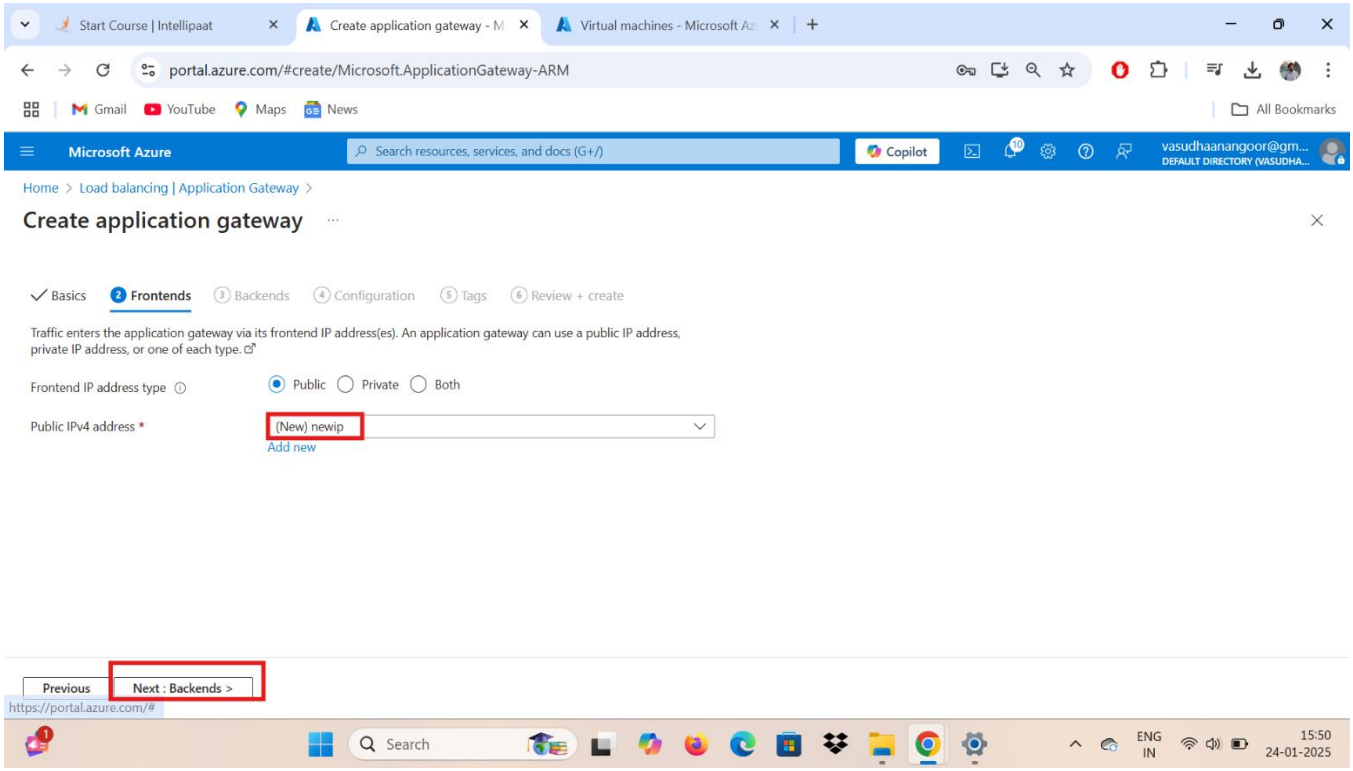
Select the resource group, then we will name the gateway as "**app_gateway01**" and the region as Central US and instance counts as shown below:



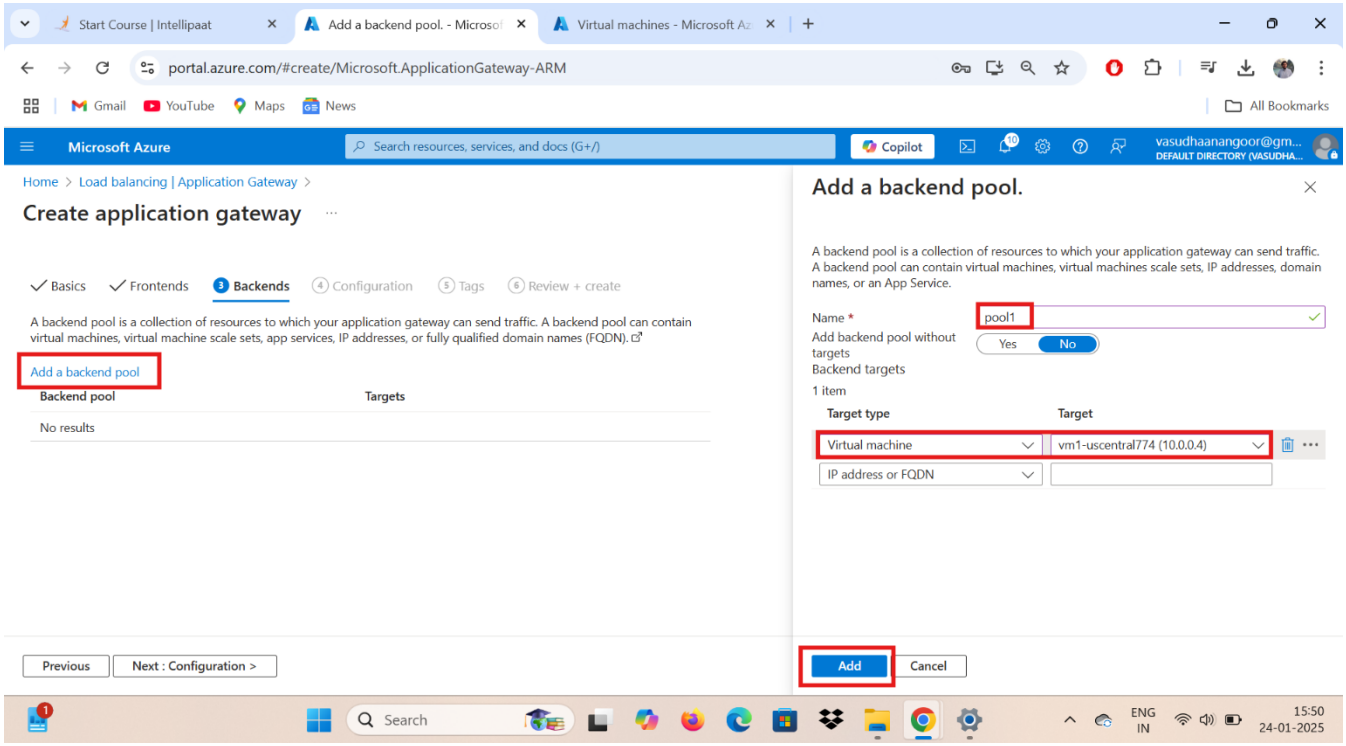
Next, we will give the same virtual network as Vnet1 and under Subnet, select the subnet we created for application gateway purpose (subnet_ag). Then click on Next.



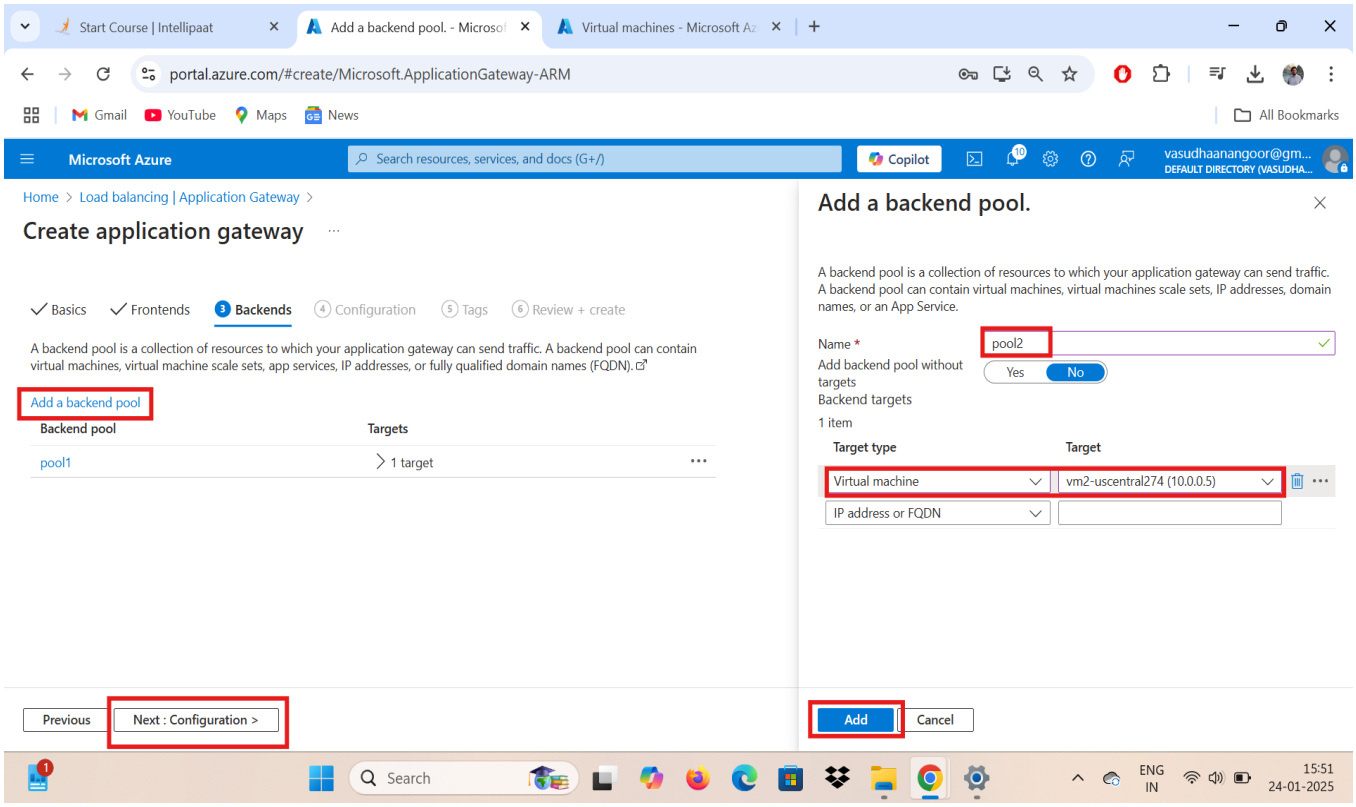
Under Frontends, create a new Public Address named "**newip**" and click on Next.



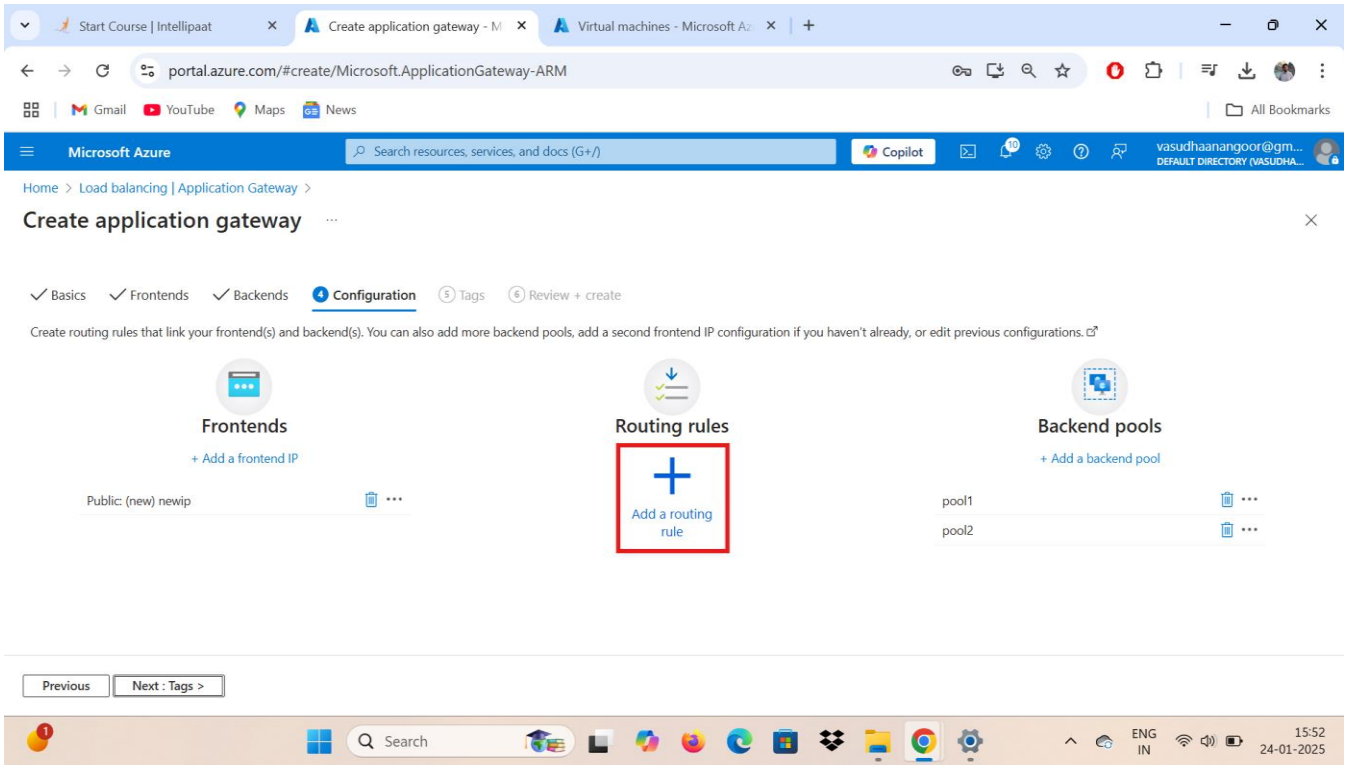
Under Backends, Click on Add a backend pool, give the name as "**pool1**" , then select the target as virtual machine and select vm1-uscentral. Finally click on Add.



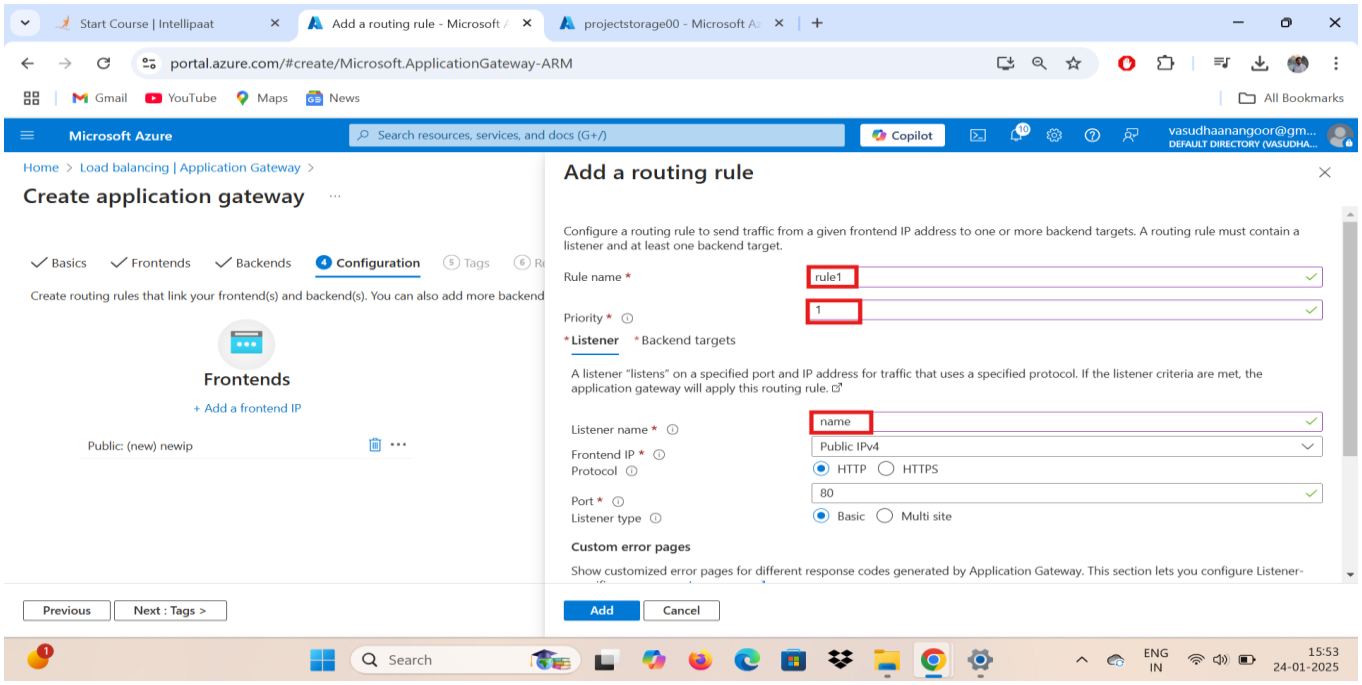
Then again click on Add a backend pool and create a "**pool2**" to add vm2-uscentral as shown below. Once done click on Next.



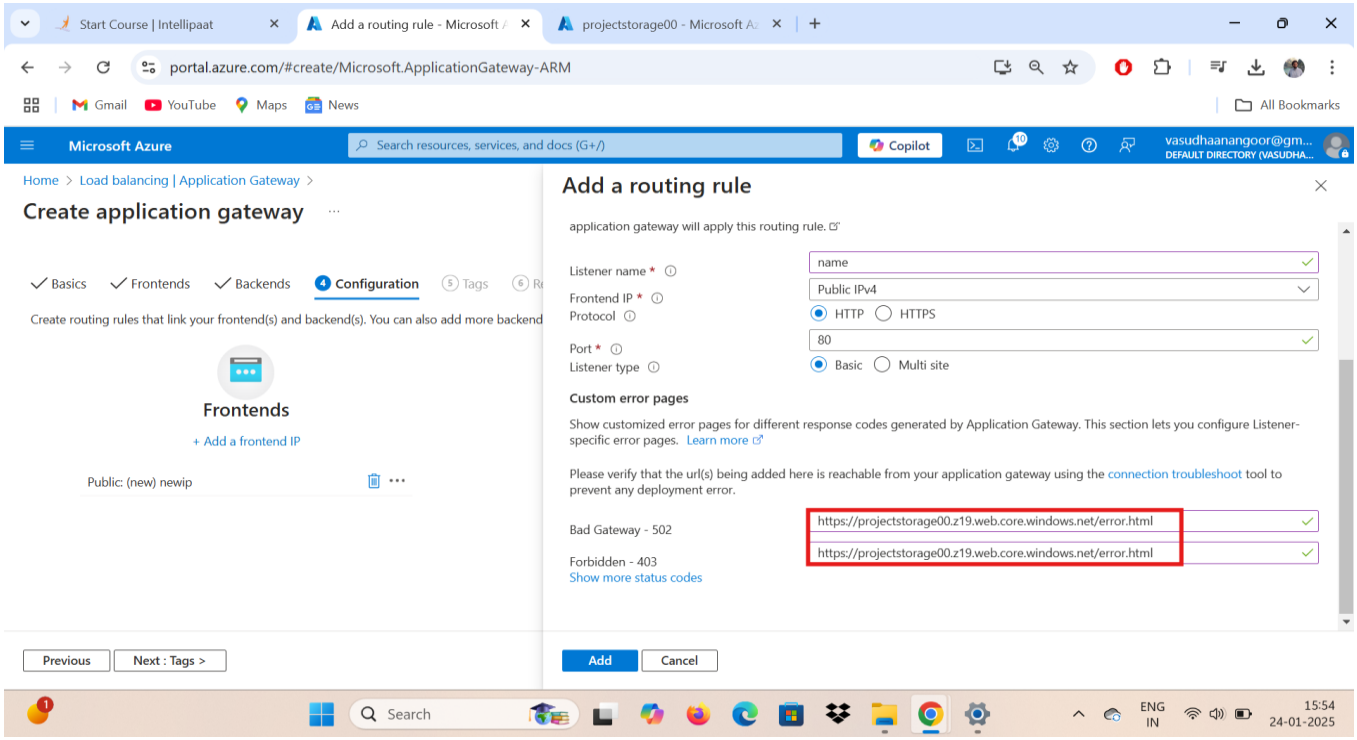
Under configuration, click on Add a routing rule and we will add the rule.



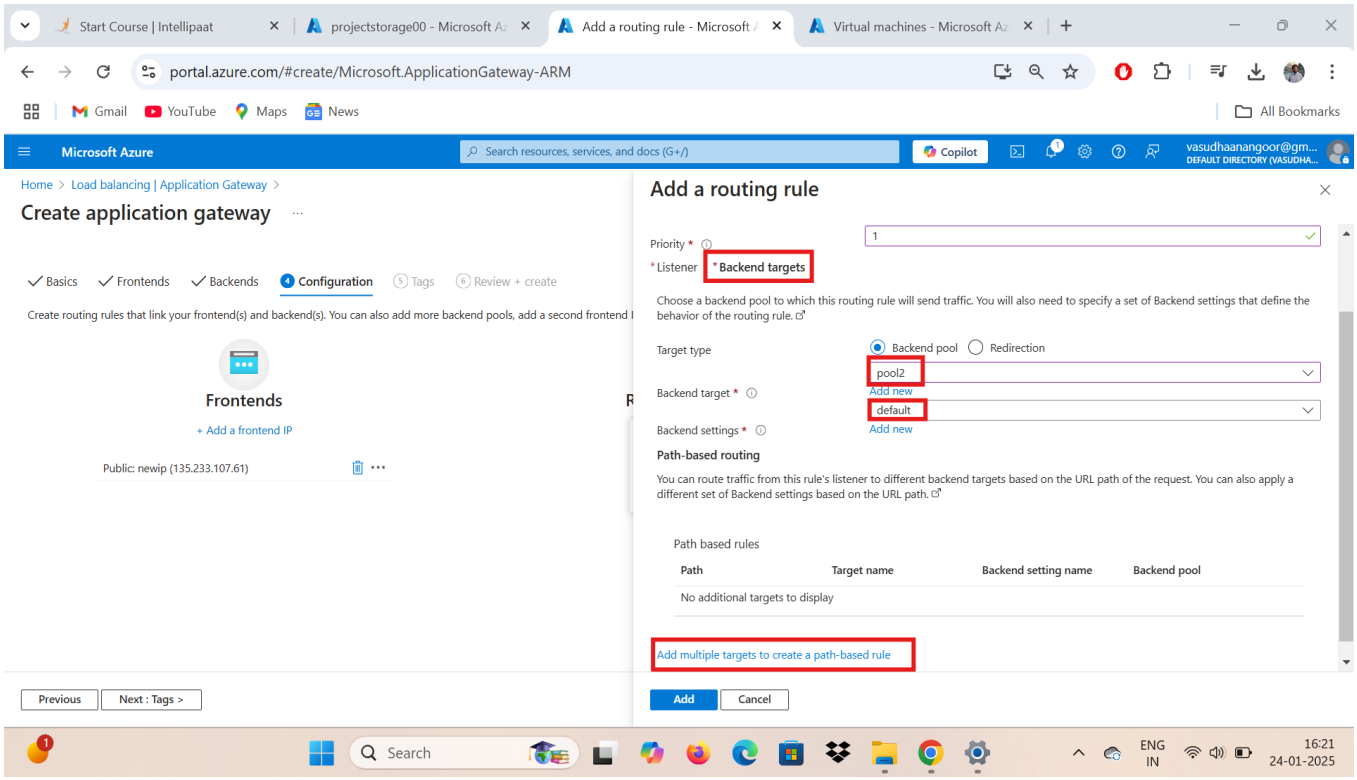
Give the rule name as "**rule1**" , priority as 1 and Listener name as "**name**".



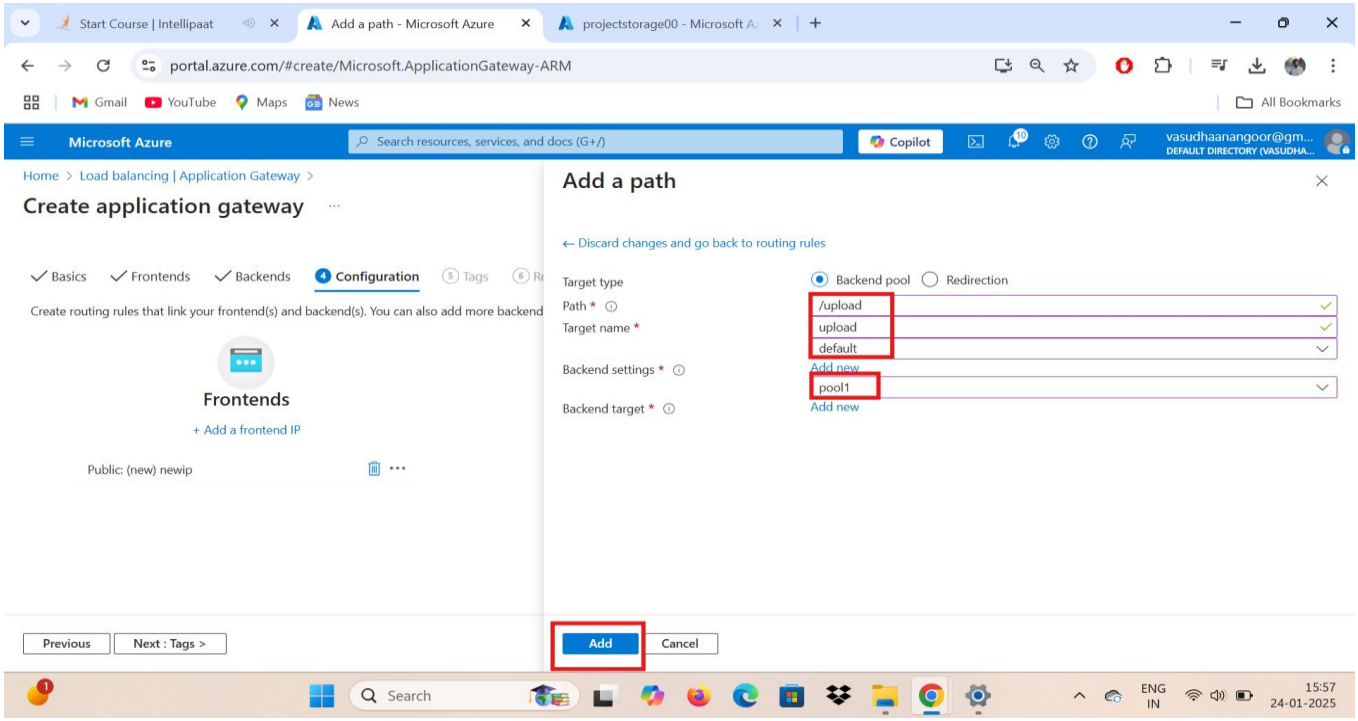
Then under Custom error pages, we will give the primary endpoint of the static website followed by error.html as the Bad Gateway and Forbidden page. Then click on Backend targets to add the pool.



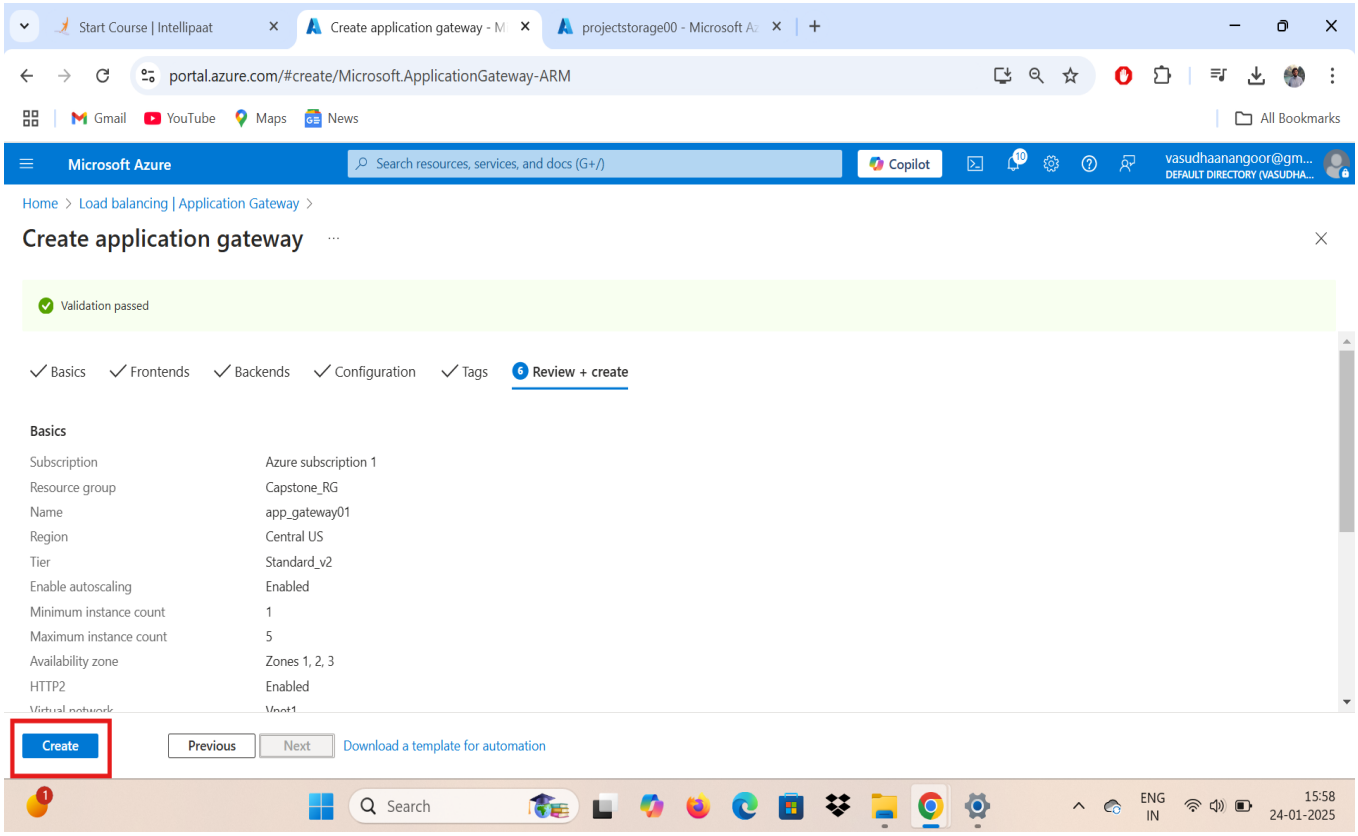
Select the target type as pool2 and settings as default. Once done, click on Add multiple targets to add the VMs.



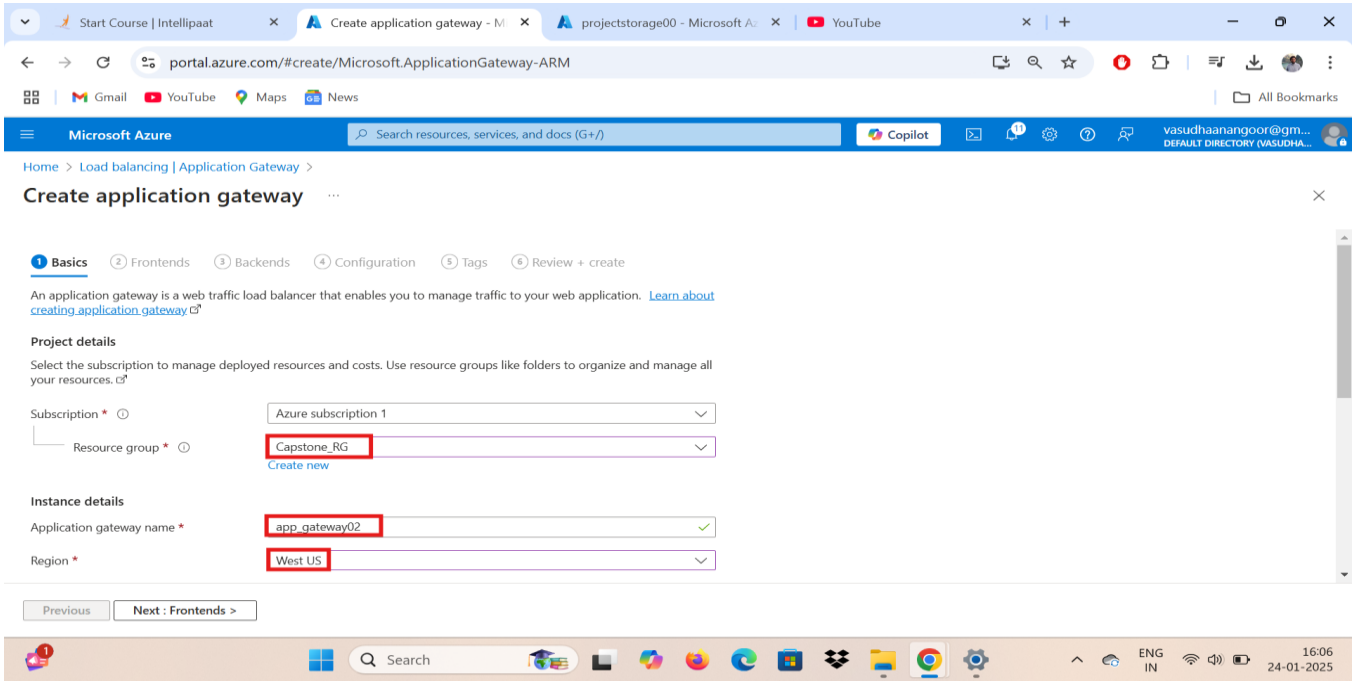
First, we will give the Path as /upload with target name as upload and default backend settings. Then select the pool1 as the Backend target. Finally click on Add.



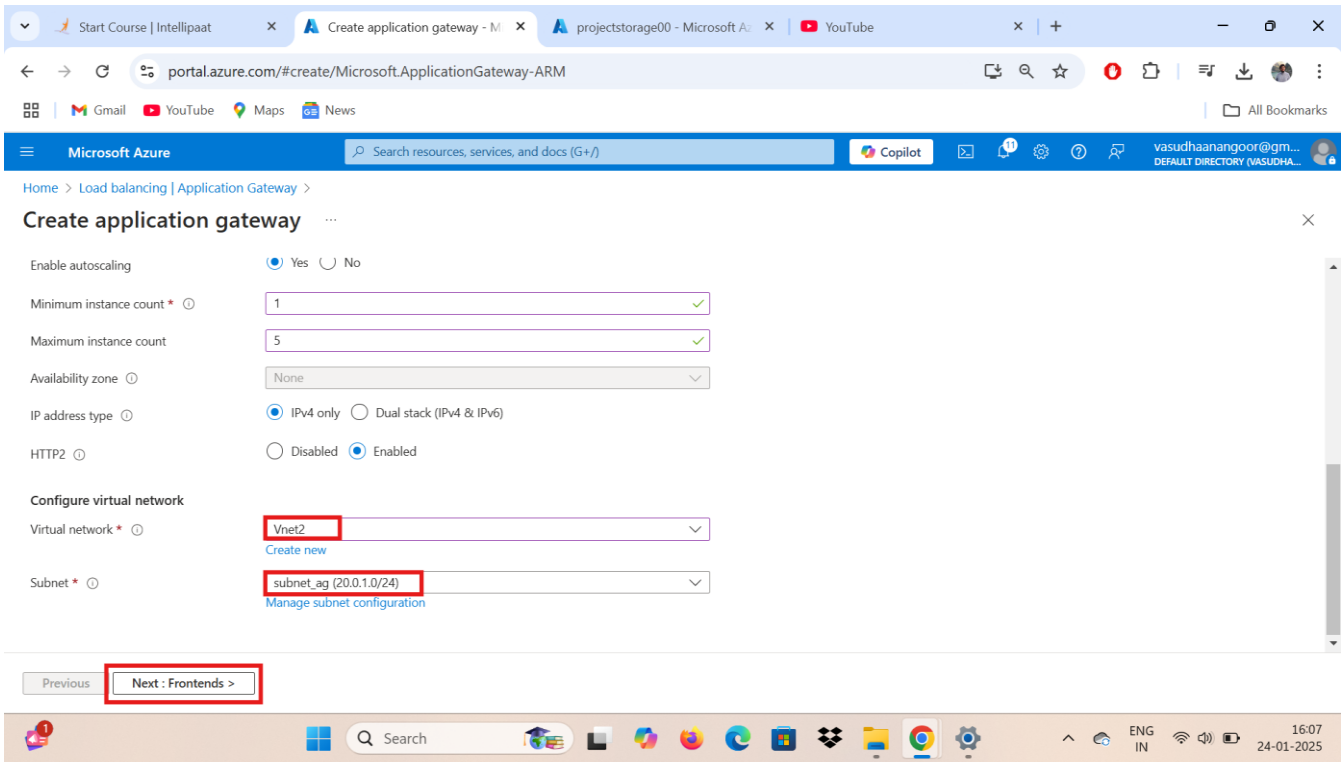
After adding the Configurations, click on Next and click on Review+create. The validation will be successfully passed and we will click on create to finally create the application gateway.



Similarly, we will create an Application Gateway in West US region. We will name it as "**app_gateway02**" with the same resource group as before and select the region as West US.

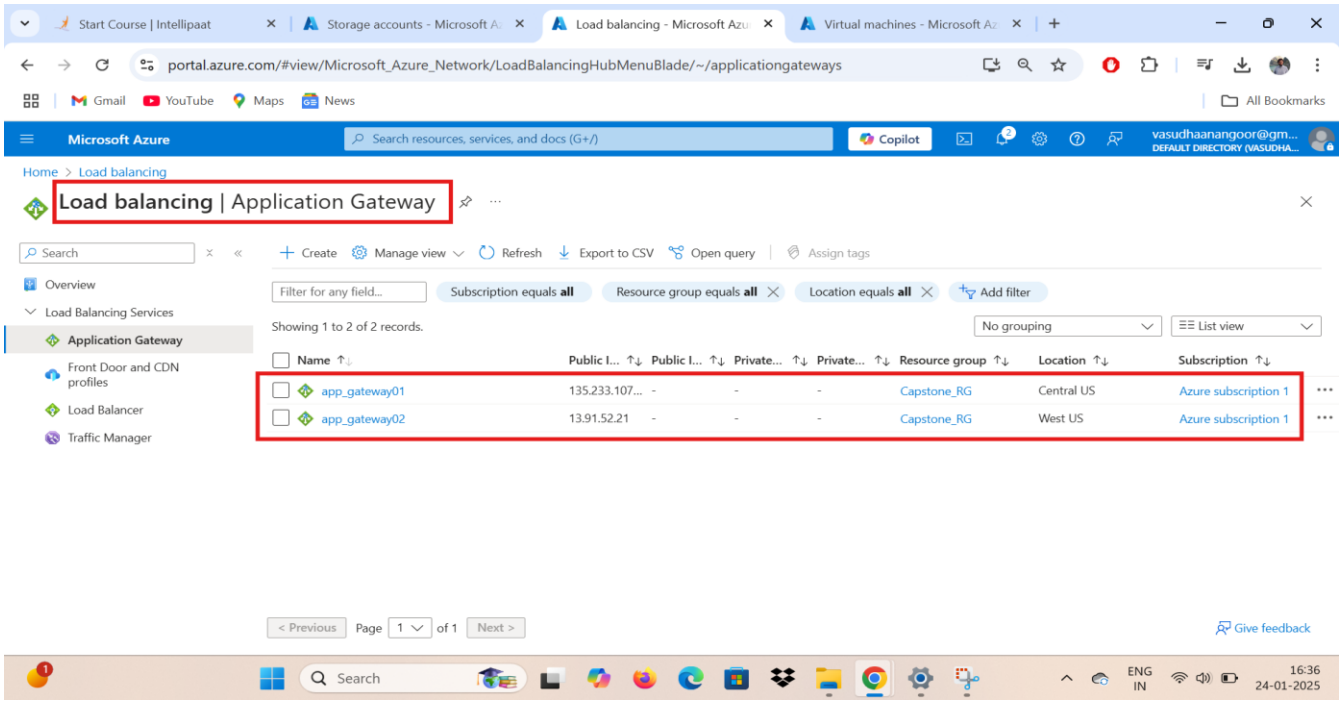


Next, we will give the same virtual network as Vnet2 and under Subnet, select the subnet we created for application gateway purpose (subnet_ag). Then click on Next.

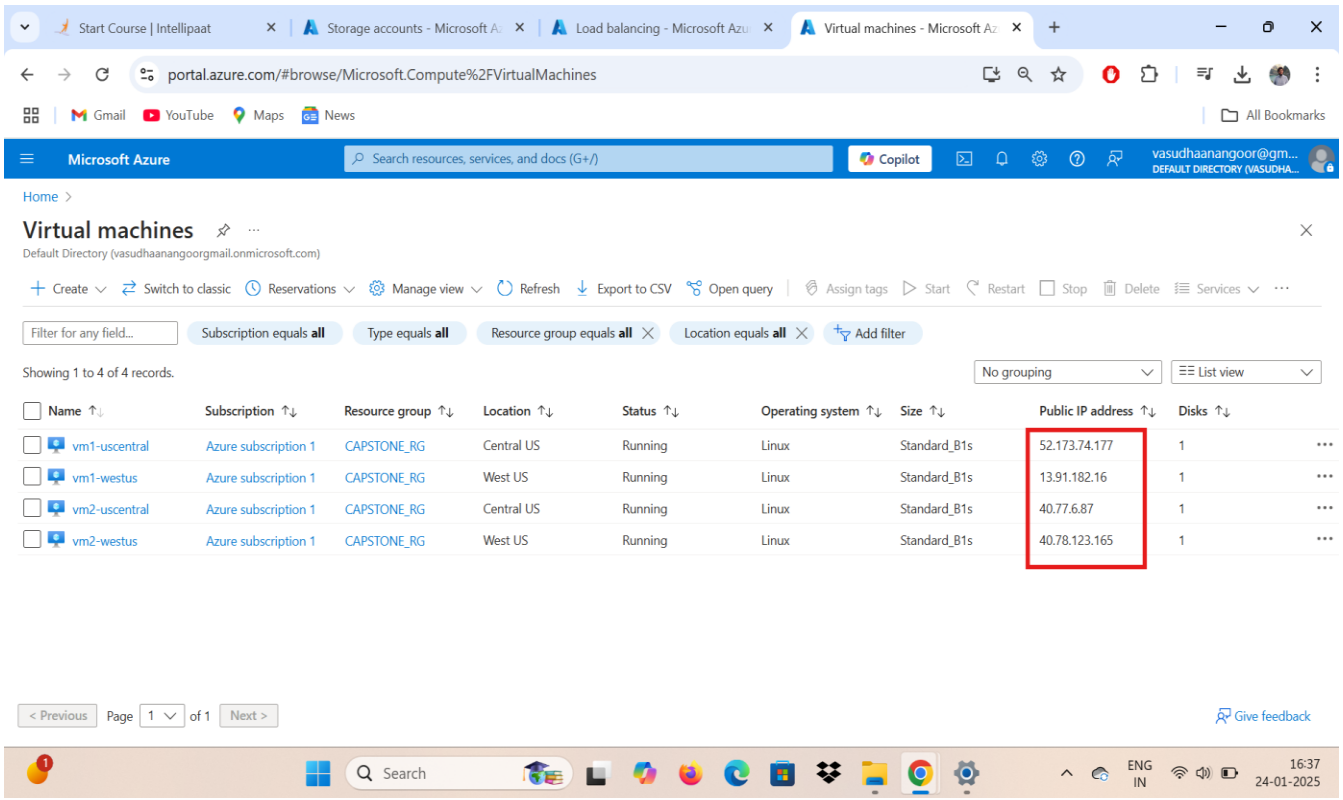


The rest of the configurations will be the same as the previous gateway . We only need to change the routing rules and select the resources from West US region.

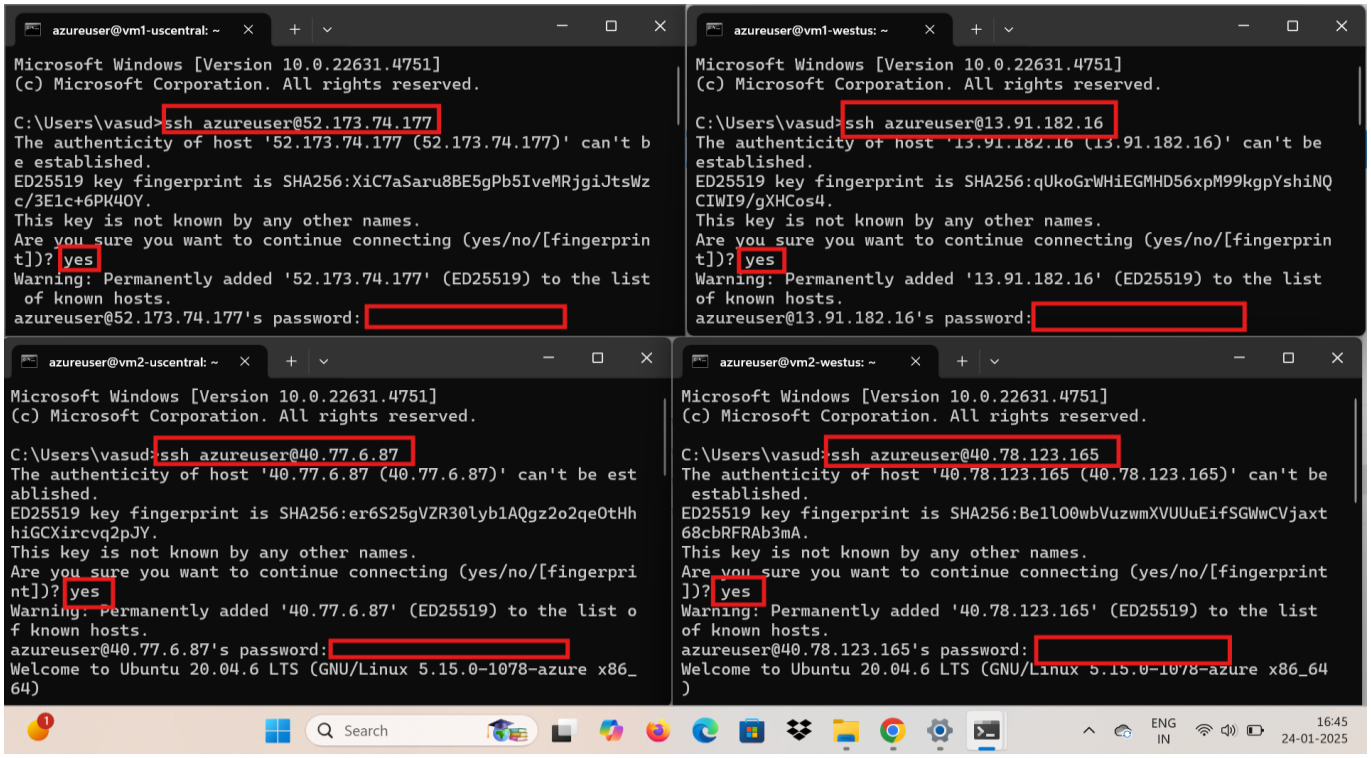
Finally, both the Application Gateways will be listed as below:



6:RUNNING SCRIPTS IN THE VM: We need to connect to the VM through command prompt. To do this we will copy the public IP addresses of all the four VMs .

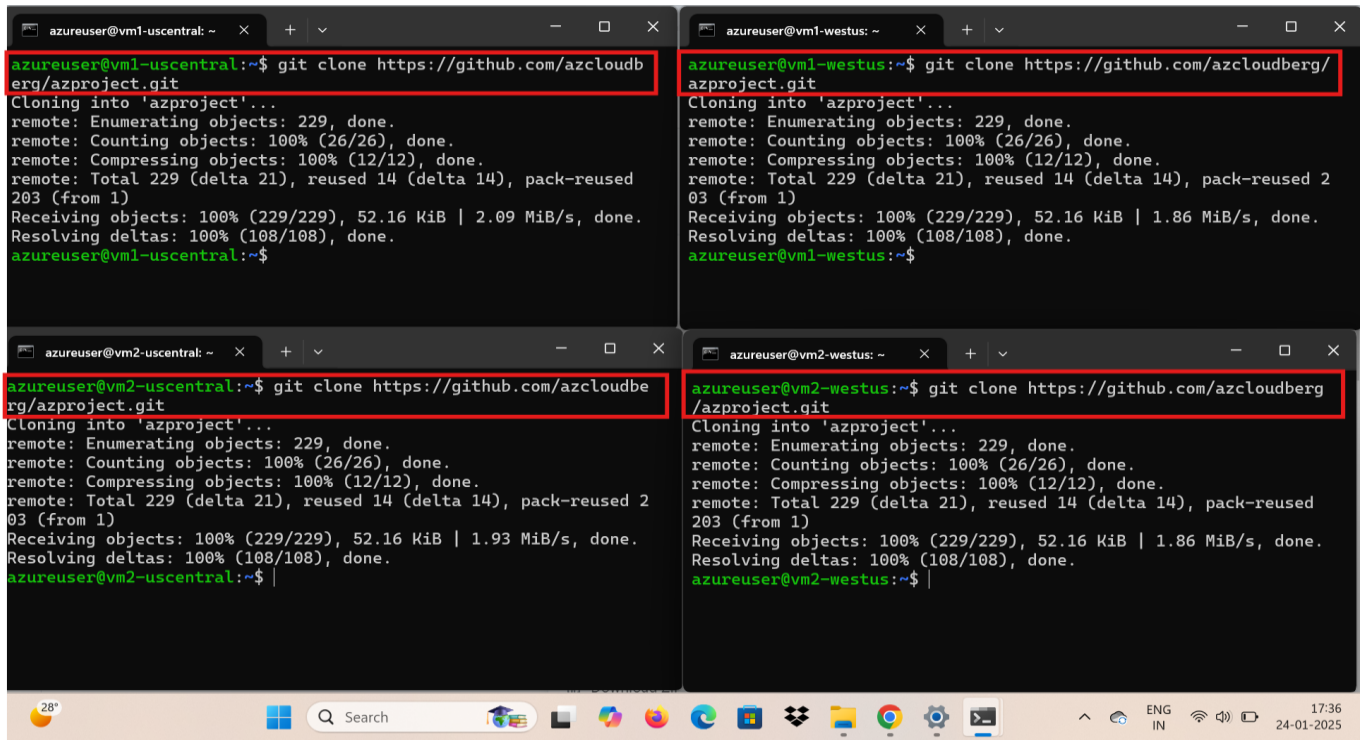


We will use the command "**ssh user@publicipaddress**" to connect. Also, we will provide the password.

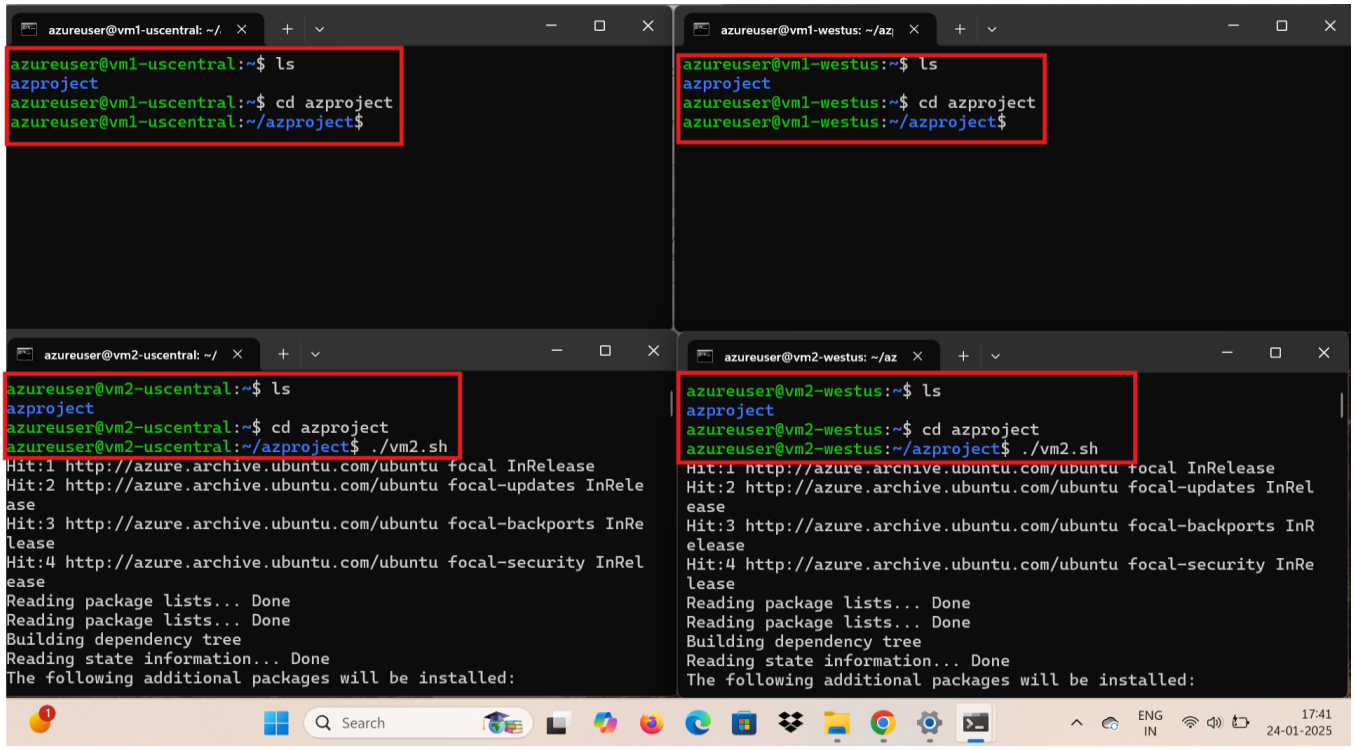


Once the connection is success, we will be logged in to the VMs. We will update the machines using the command "**sudo apt-get update**".

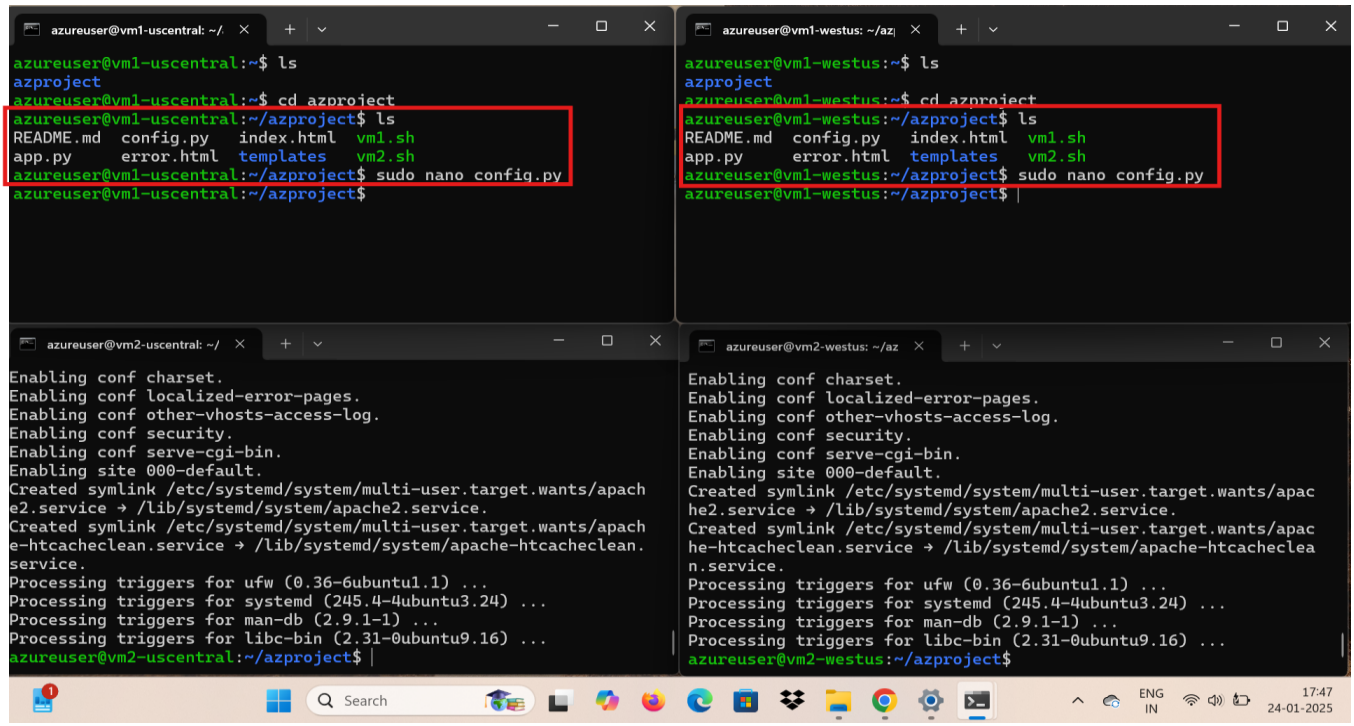
Next, we will clone the GitHub repo mentioned in the problem statement using "**git clone"** command in all the VMs.

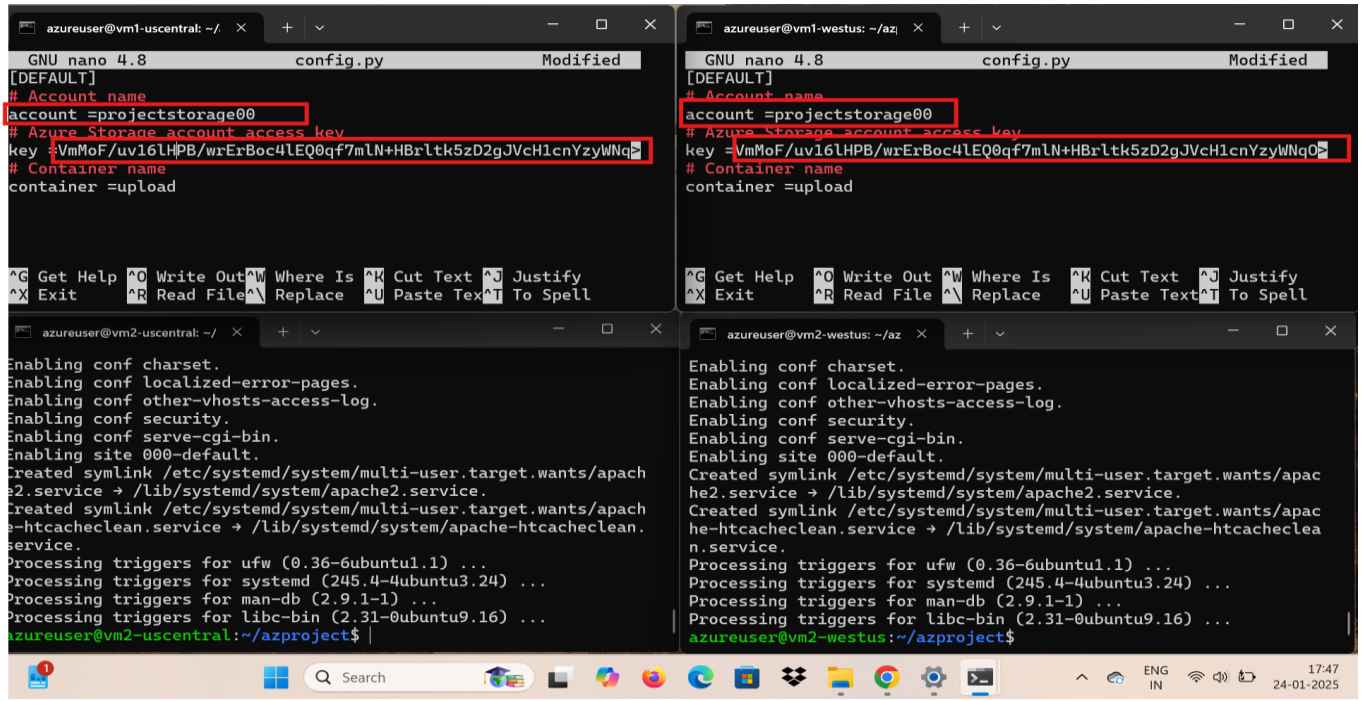


Once done, the “azproject” folder will be available. We will get inside this folder and execute the vm2.sh file containing the scripts only in the VM2 of both the regions .

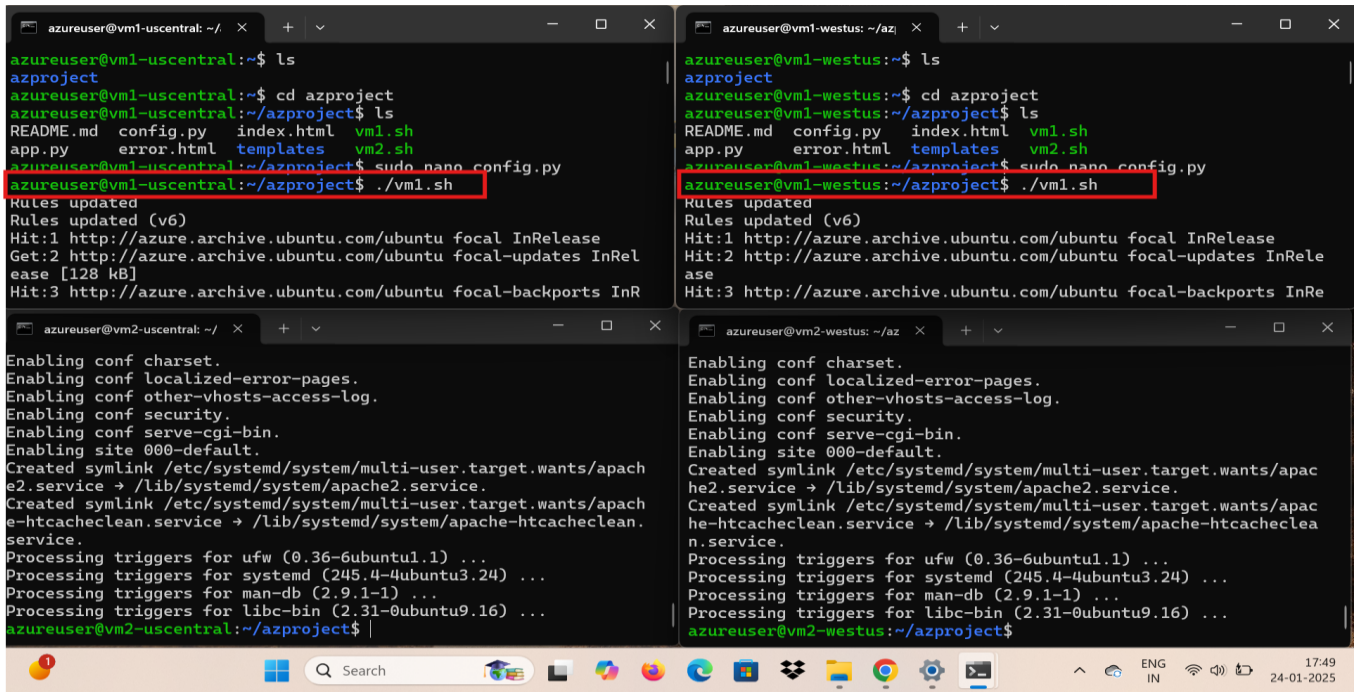


Next, we will go to the config.py file in VM1 of both the regions and add the storage account name and access key. (The access key will be available in the storage account under Keys).



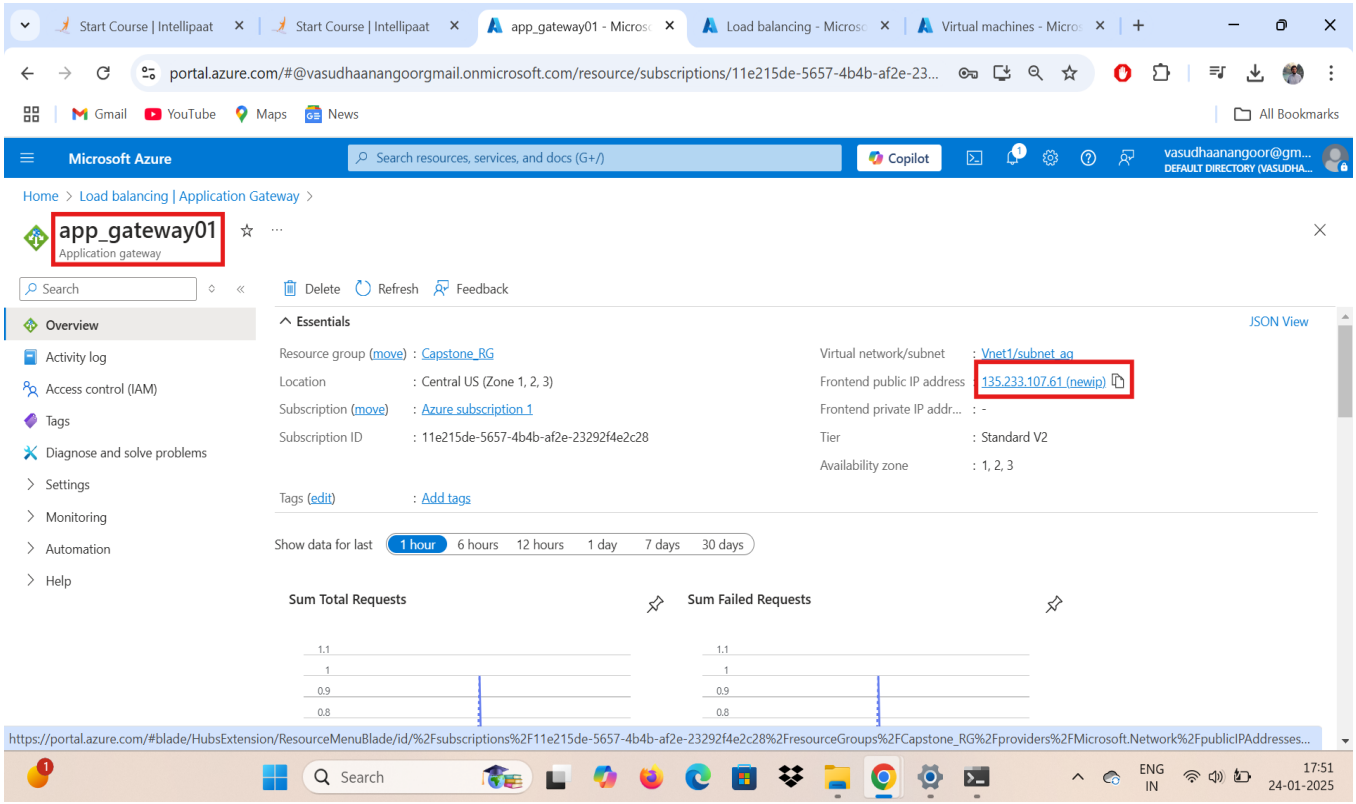


Once done , we will execute the vm2.sh file containing the scripts in the VM1s of both the regions.

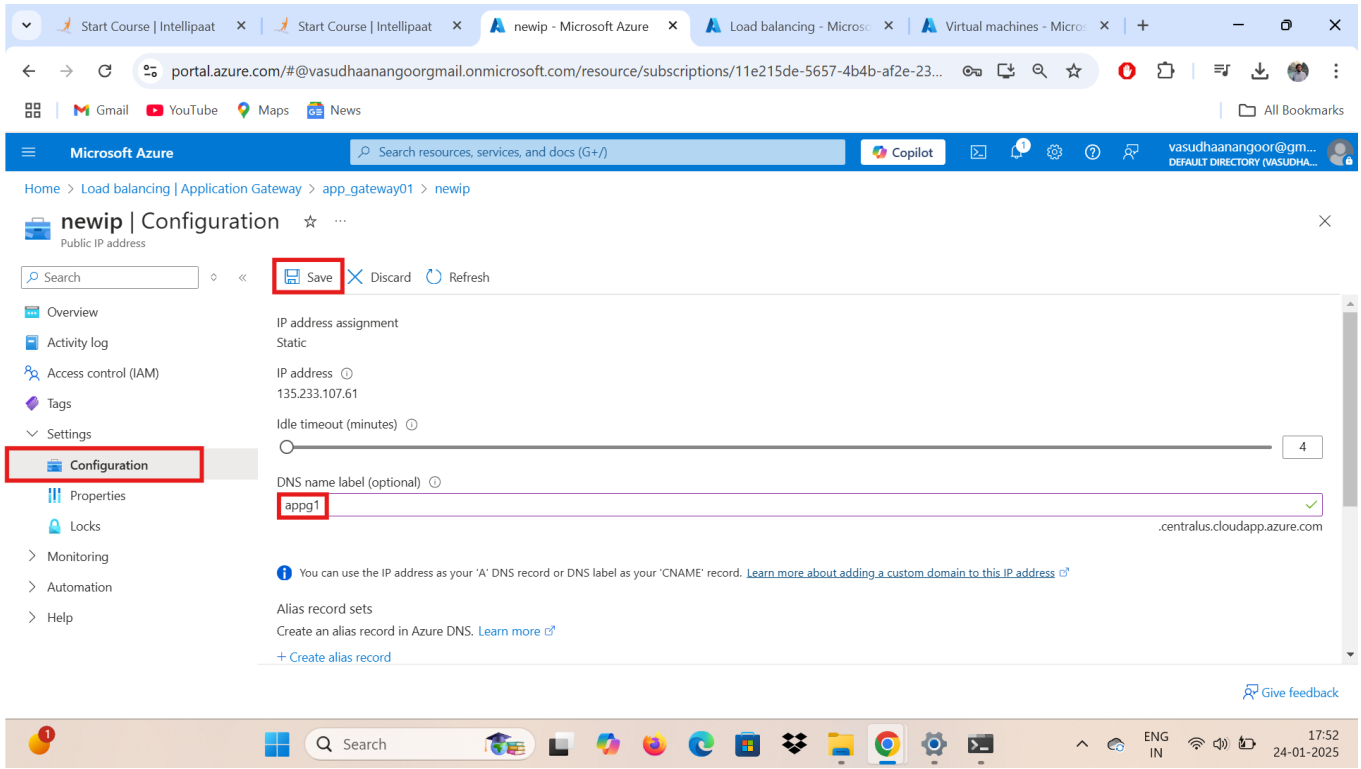


7.CREATING TRAFFIC MANAGER: To balance the load, we need to create traffic managers in both the regions. For this, firstly we need to configure the DNS names of both the Application Gateways.

We will go to the Application Gateway and click on the Frontend public IP address to configure it.

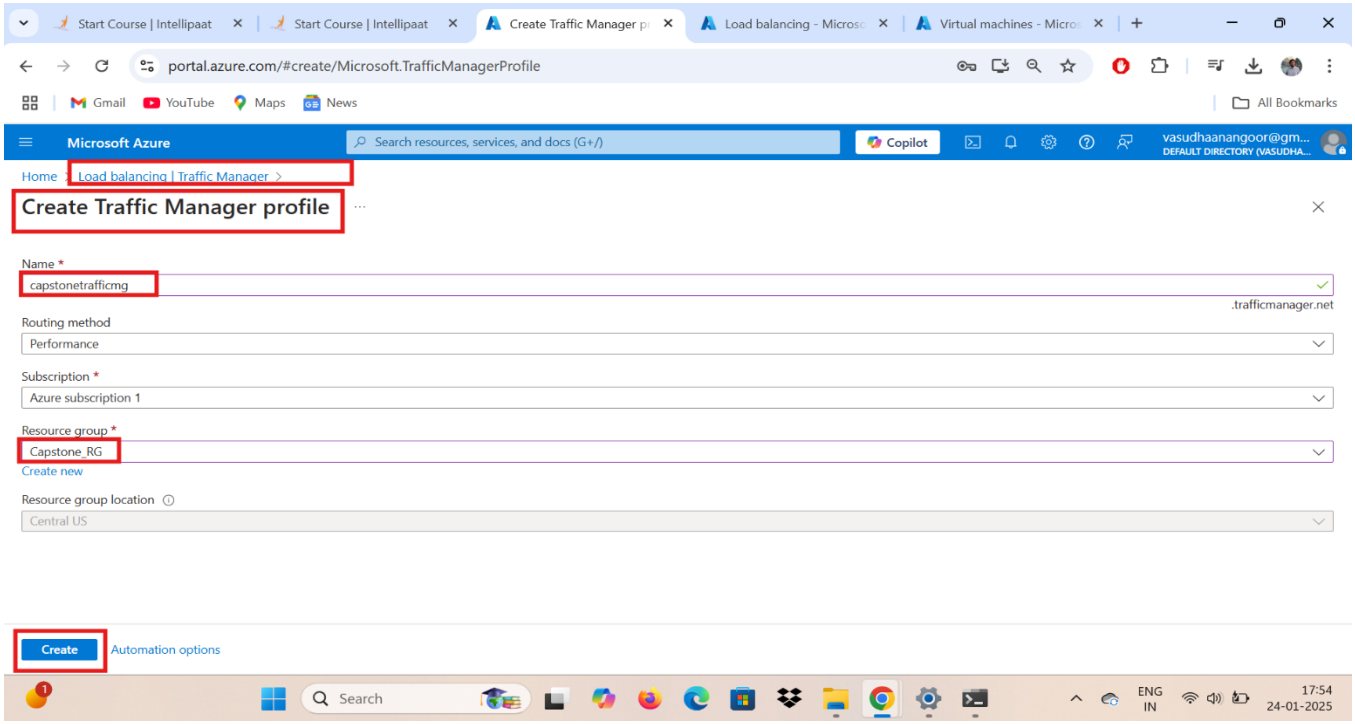


Then we will click on Configurations under Settings. We will give the DNS name as "**appg1**" and click on Save.

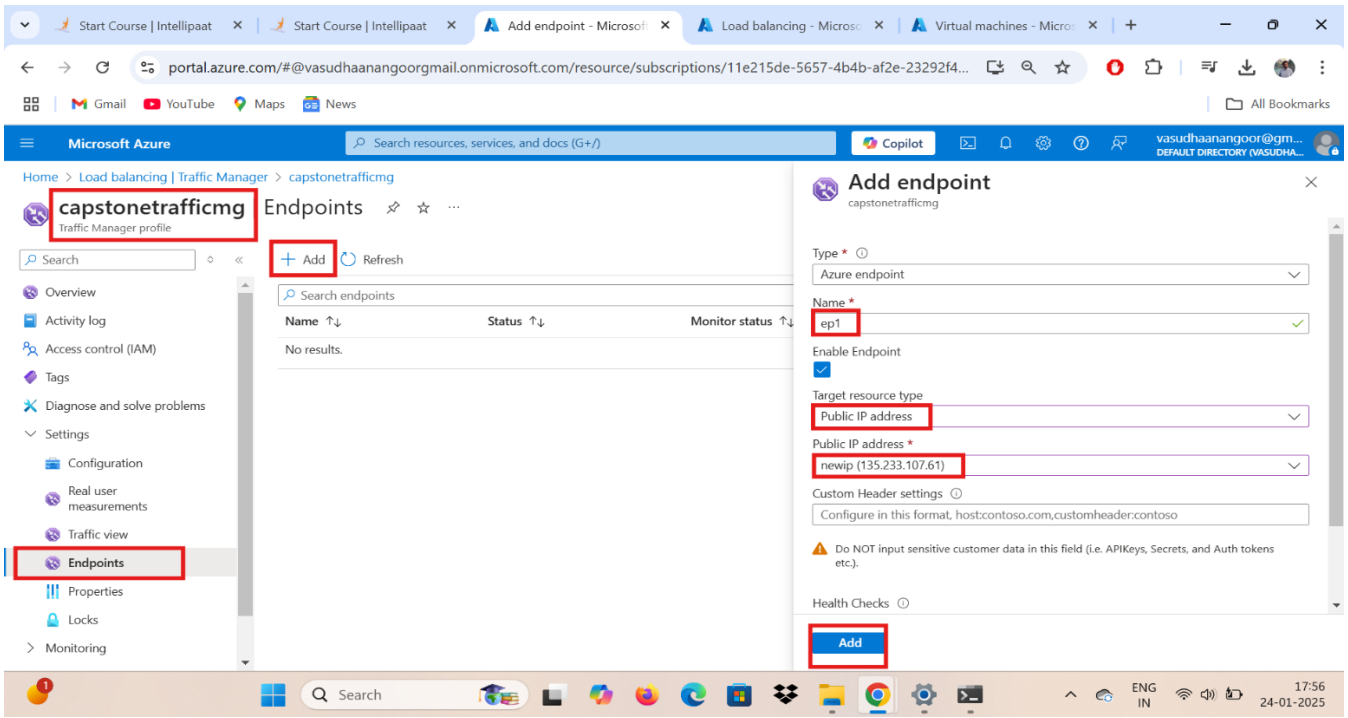


We will follow the same procedure in the second Application Gateway and give the DNS Name as “**appg2**”.

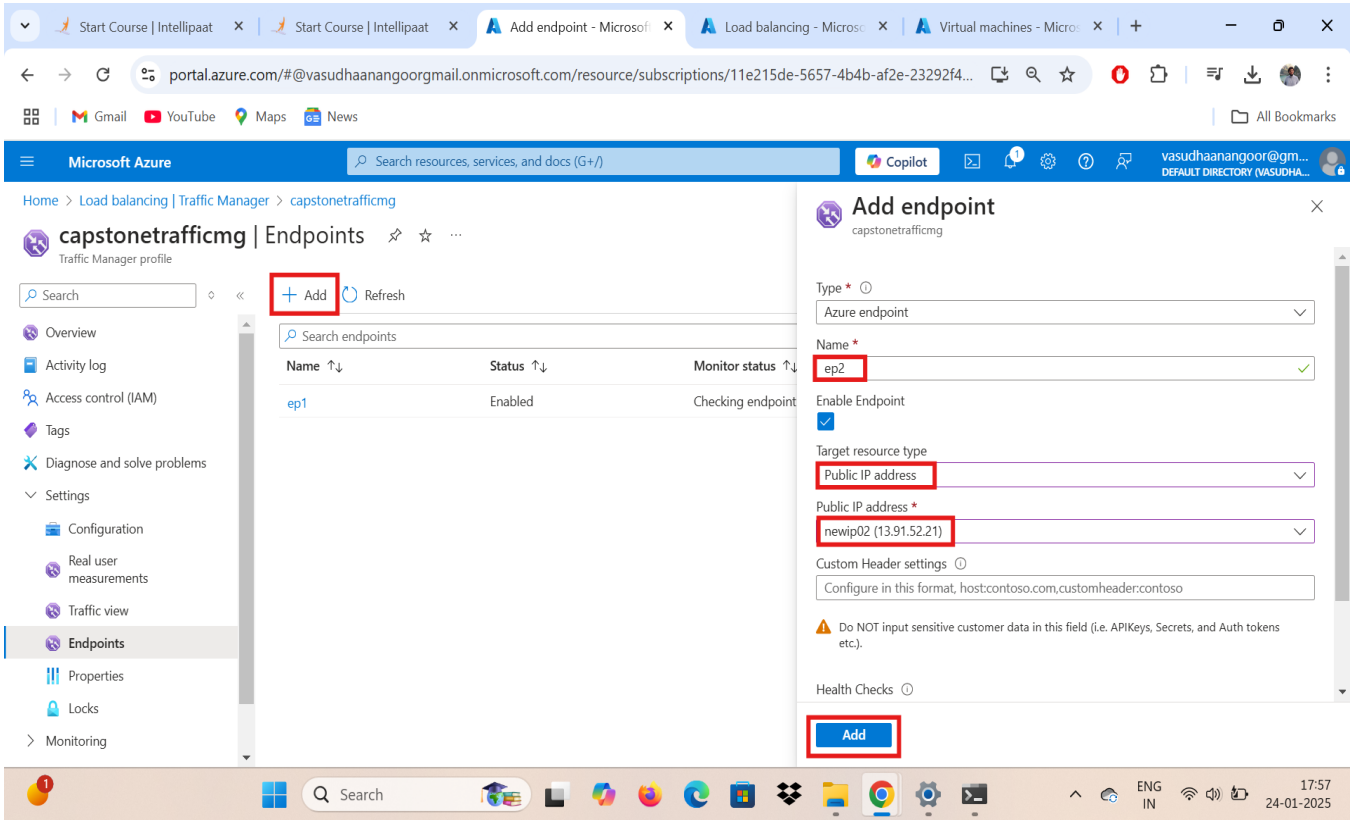
Next, we will create a Traffic Manager profile for load balancing. For this, type traffic manager in the search bar at the top and navigate to its console. Then click on create.We will name it as “**capstonetrafficmg**” and select the resource group. Then click on create.



After creating the traffic manager profile, go to Endpoints under settings and click on Add. We will give the name as “**ep1**” and select Public IP address as the target and select the IP of the first Application Gateway. Finally click on Add.

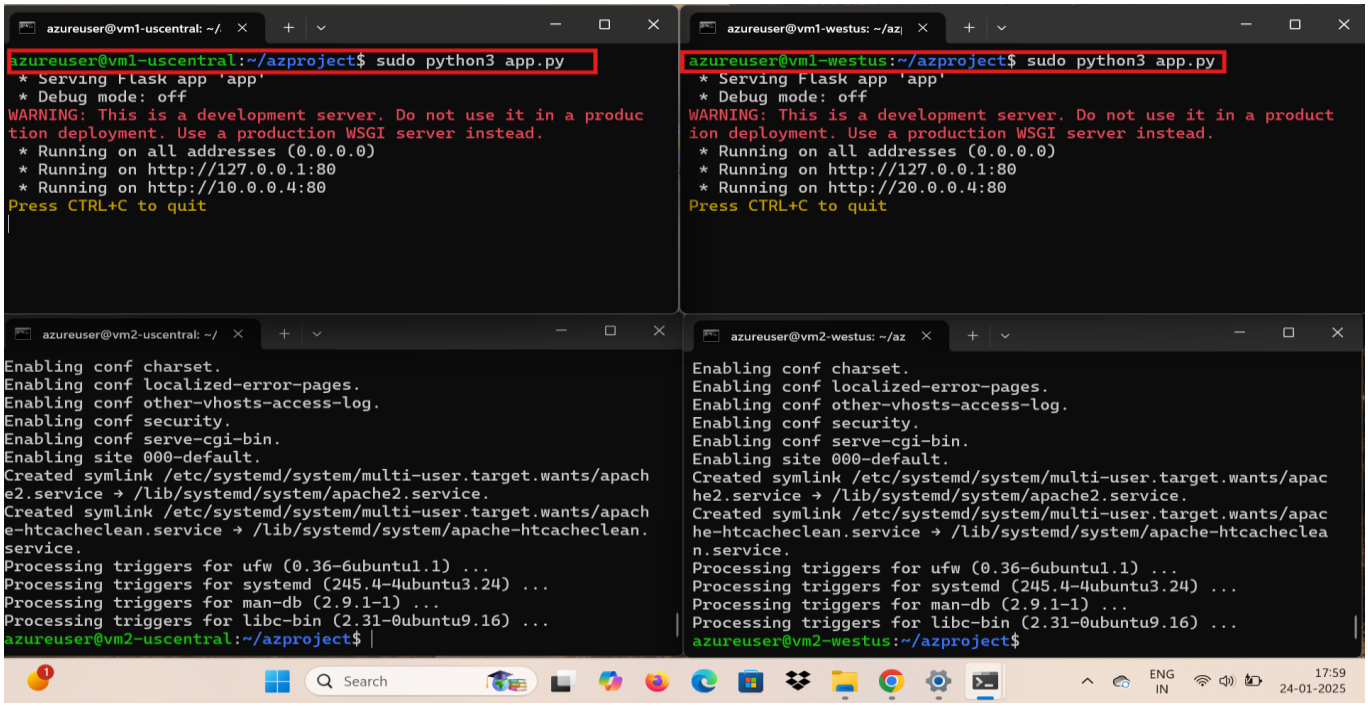


Again, click on Add to add the second IP. We will give the name as "ep2" and select Public IP address as the target and select the IP address of the second Application Gateway. Finally click on Add.

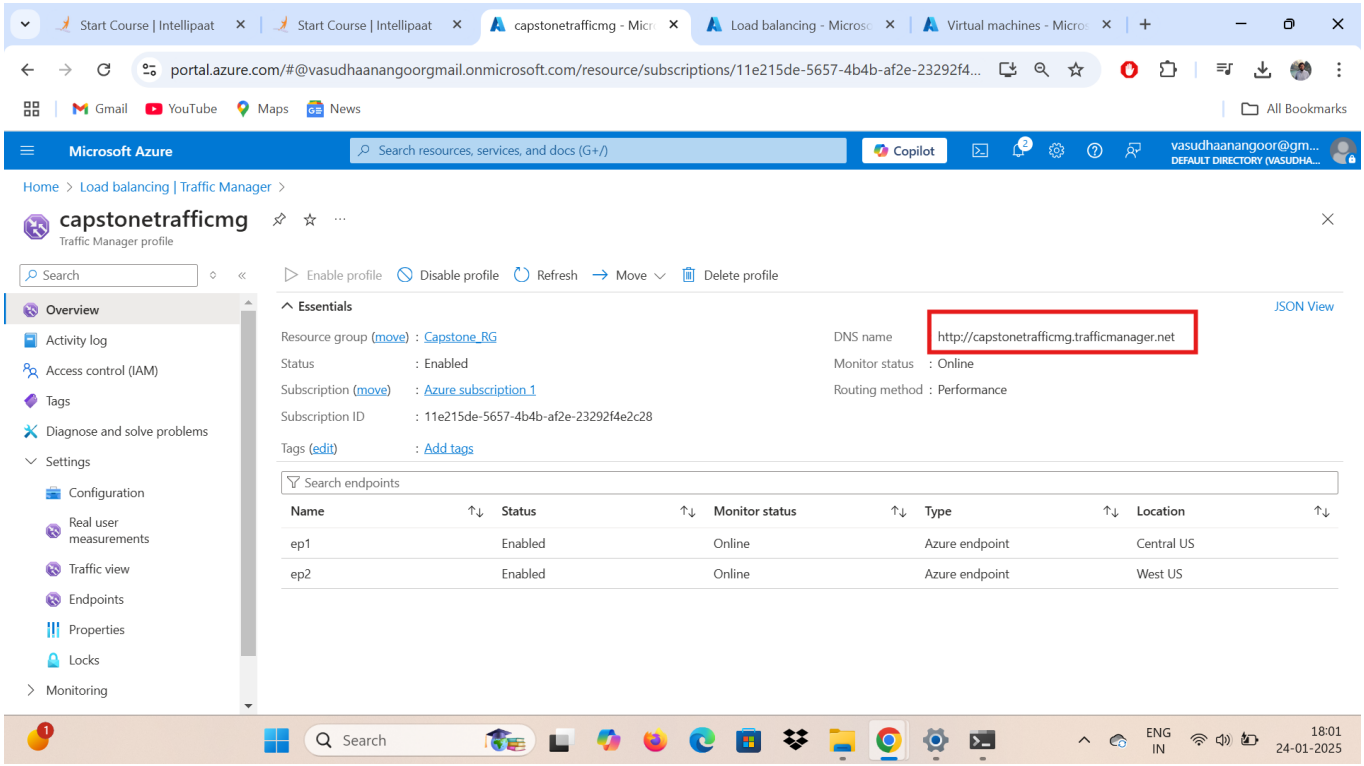


Once added, the Traffic Manager profile will be successfully created.

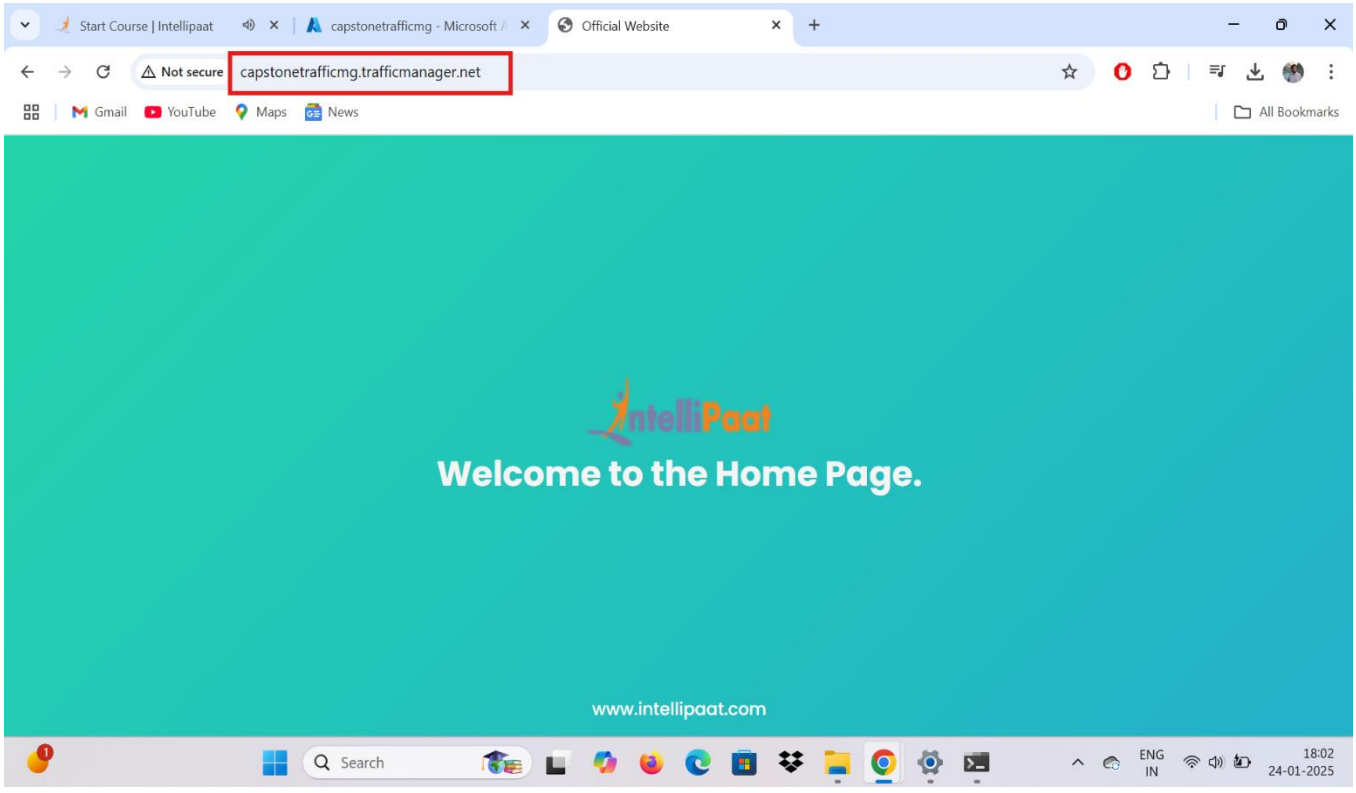
Next, we will go to the command prompt and run the command **sudo python3 app.py** in the VM1s of both the regions.



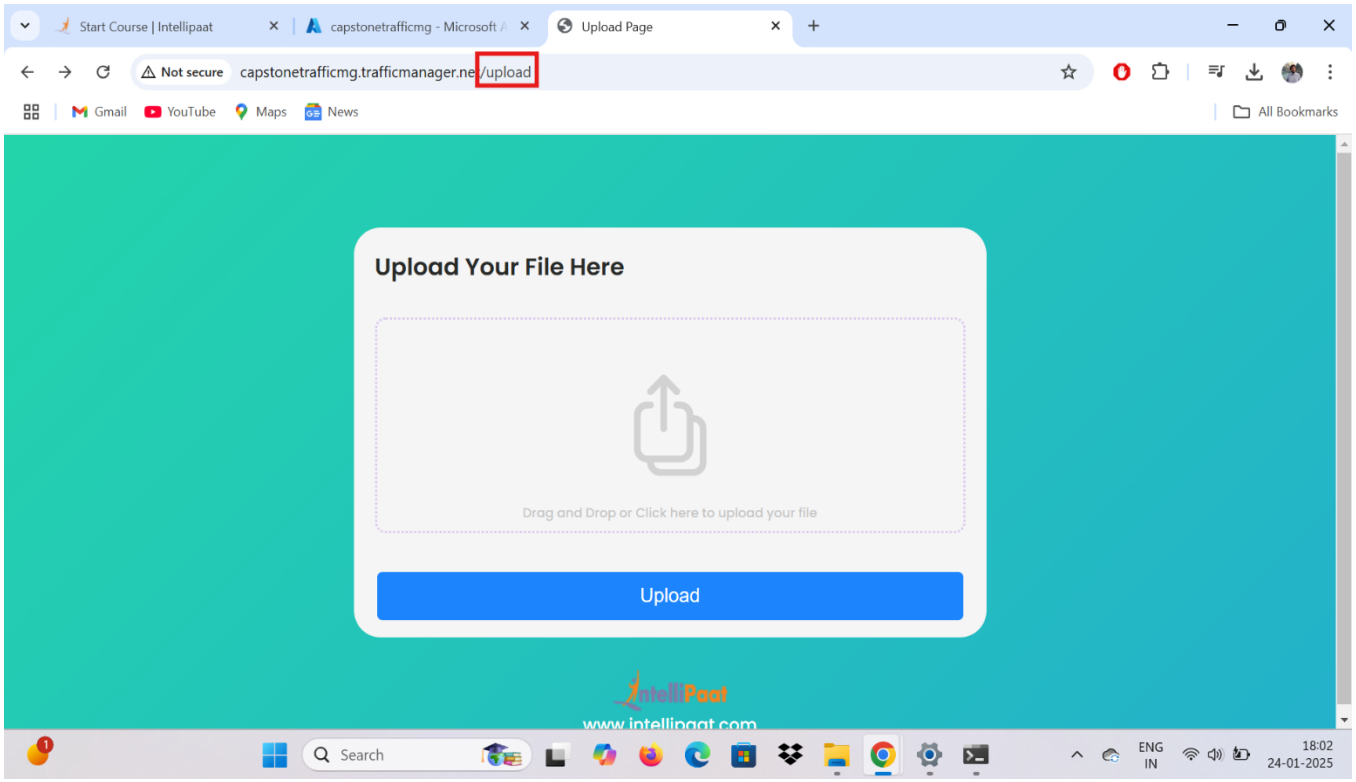
8.VERIFICATION: To verify the implementation of this project, we will copy the DNS name of the traffic manager and paste it in the browser.



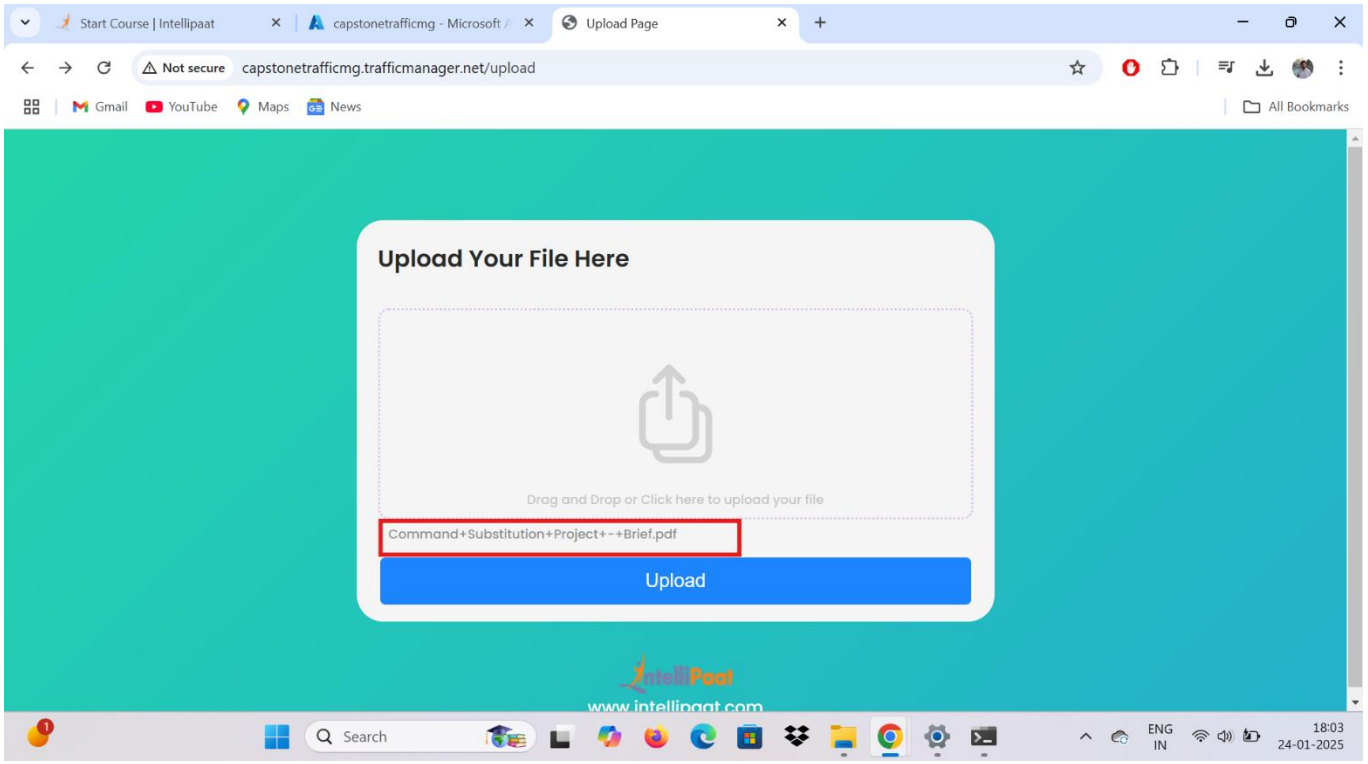
We will be navigated to the home page.



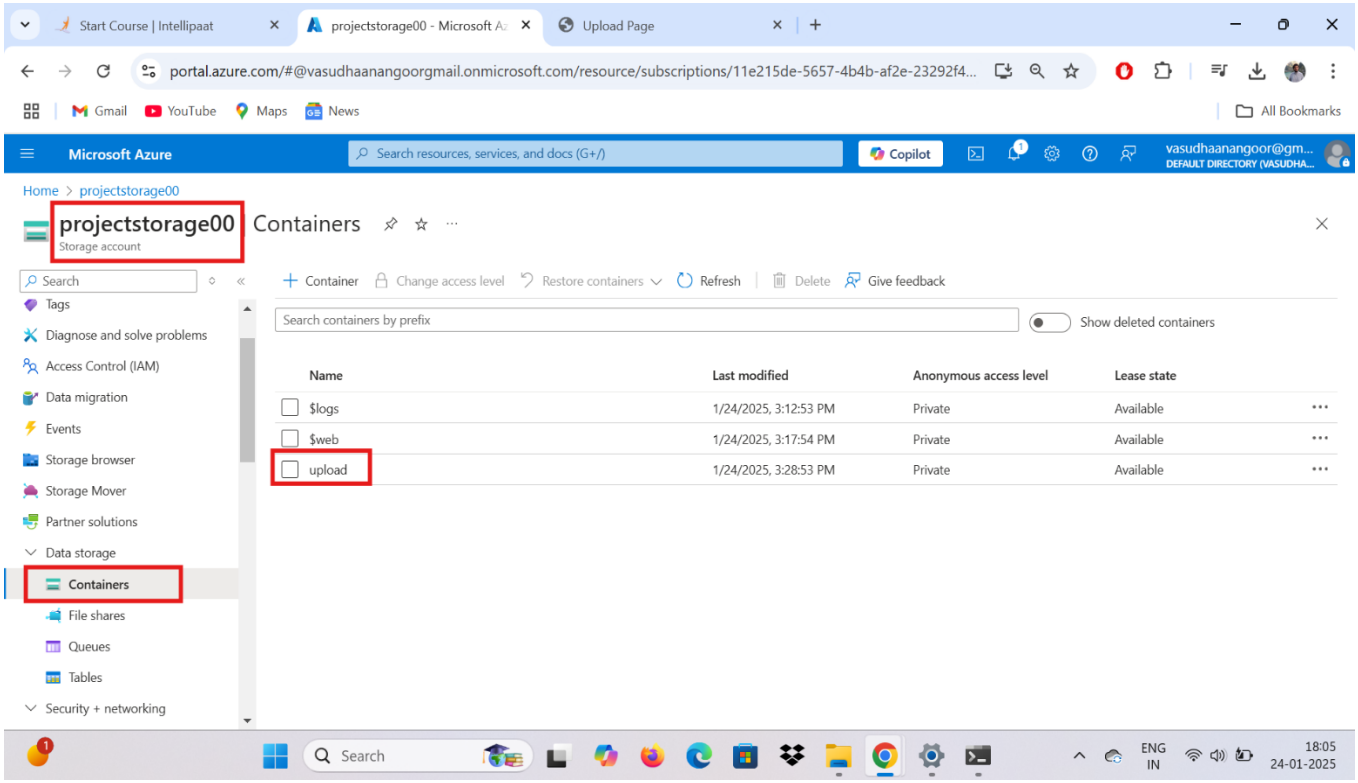
Next we will navigate to the upload page by tying /upload along with the DNS Name.



Here we will upload a file.



Once uploaded, the same file will be uploaded to the container “**upload**” by default. To check this, we will go to the Storage Account and click on Containers under Data storage. Then click on the upload container.



The file will be successfully uploaded in the container as shown below: