

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:

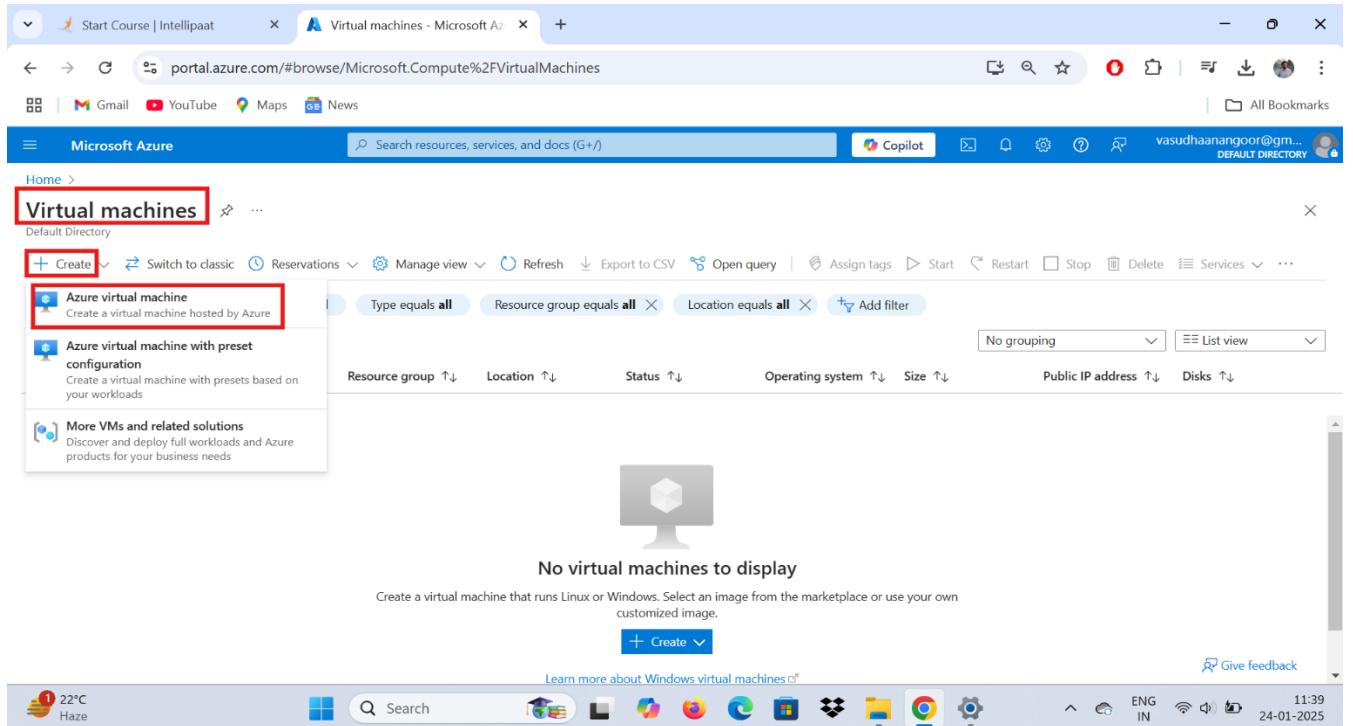
1. You need to host your error.html as a static website here, and then point the application gateway's 403 and 502 errors to it.
2. Create a container named upload, this will be used by your code to upload the files.

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.

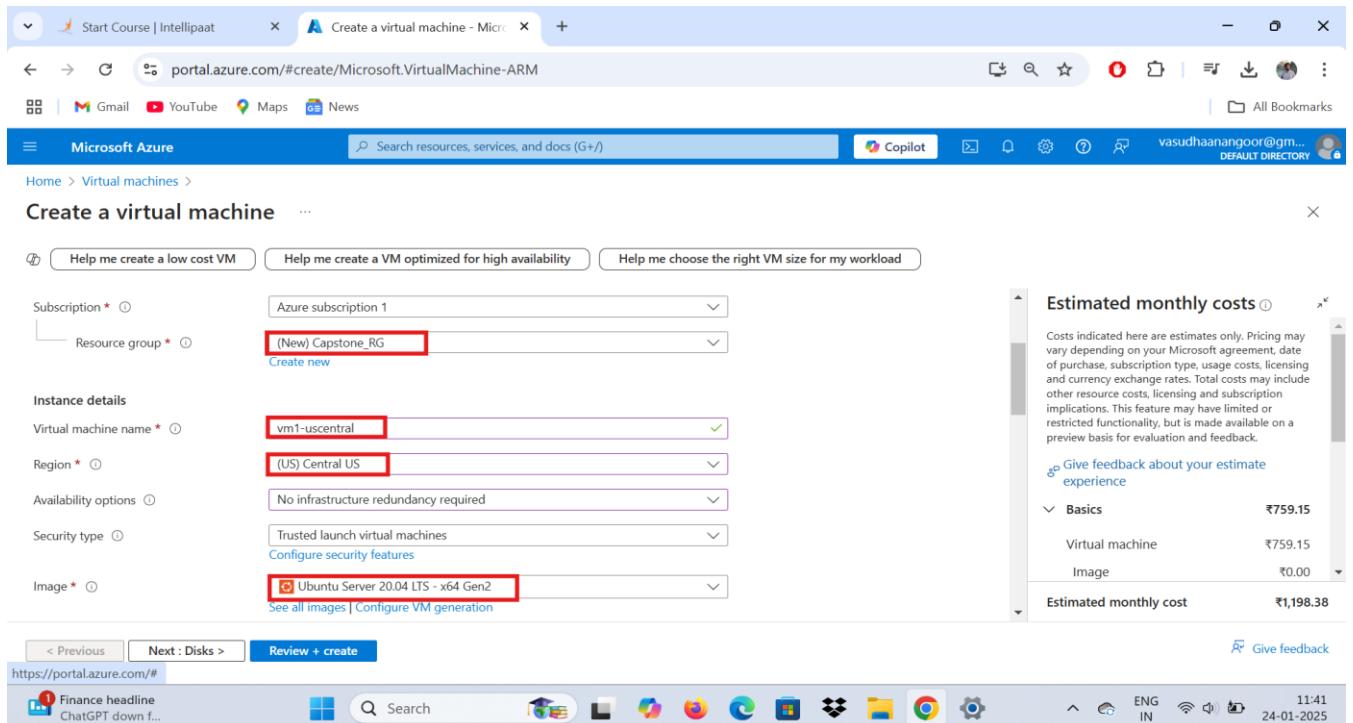
SOLUTION:

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.



The screenshot shows the Microsoft Azure portal's Virtual Machines page. A red box highlights the 'Create' button under the 'Azure virtual machine' section. The page displays a search bar, filter options, and a list of VM configurations. Below the list, a message says 'No virtual machines to display' with a link to learn more about Windows virtual machines. The browser toolbar and taskbar are visible at the bottom.

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.



The screenshot shows the 'Create a virtual machine' wizard. A red box highlights the 'Resource group' dropdown, which contains '(New) Capstone_RG'. Other fields include 'Virtual machine name' (vm1-uscentral), 'Region' (US Central US), and 'Image' (Ubuntu Server 20.04 LTS - x64 Gen2). On the right, an 'Estimated monthly costs' table is shown with one row for 'Basics' (Virtual machine: ₹759.15, Image: ₹0.00). The total estimated monthly cost is ₹1,198.38. The browser toolbar and taskbar are visible at the bottom.

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.

The screenshot shows the 'Create a virtual machine' wizard on the Microsoft Azure portal. The 'Networking' tab is selected. In the 'Inbound port rules' section, 'Allow selected ports' is chosen, and 'HTTP (80), SSH (22)' is selected from the dropdown. A red box highlights the 'Next : Disks >' button at the bottom left of the page. On the right, there's an 'Estimated monthly costs' summary table:

Category	Cost
Virtual machine	₹759.15
Image	₹0.00
Estimated monthly cost	₹1,198.38

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.

The screenshot shows the 'Create virtual network' wizard on the Microsoft Azure portal. The 'Networking' tab is selected. In the 'Subnets' section, two subnets are defined:

- Subnet1: Address range 10.0.0.0/24, Addresses 10.0.0.0 - 10.0.255.255 (55536 addresses)
- Subnet_ag: Address range 10.0.1.0/24, Addresses 10.0.1.0 - 10.0.1.255 (256 addresses)

A red box highlights the 'OK' button at the bottom right of the page.

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

The screenshot shows the 'Create a virtual machine' wizard on the Microsoft Azure portal. The current step is 'Networking'. The 'Virtual network' dropdown is set to '(new) Vnet1' and the 'Subnet' dropdown is set to '(new) subnet1 (10.0.0.0/24)'. On the right side, there is a sidebar titled 'Estimated monthly costs' which lists the cost for Basics, Disks, Networking, and Public IP. The total estimated monthly cost is ₹2,167.60. At the bottom of the page, there are 'Previous' and 'Next : Management >' buttons, and the 'Review + create' button is highlighted with a red box.

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

The screenshot shows the 'Create a virtual machine' wizard on the Microsoft Azure portal. The current step is 'Review + create'. A green bar at the top indicates 'Validation passed'. Below it, there are tabs for Basics, Disks, Networking, Management, Monitoring, Advanced, Tags, and Review + create. The 'Review + create' tab is selected. On the right, there is a sidebar titled 'Estimated monthly costs' showing the breakdown of costs. At the bottom, there are 'Previous' and 'Next >' buttons, and the 'Create' button is highlighted with a red box.

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.

The screenshot shows the Microsoft Azure portal's 'Create a virtual machine' wizard. The 'Instance details' step is selected. Key fields include:

- Subscription:** Azure subscription 1
- Resource group:** Capstone_RG (highlighted with a red box)
- Virtual machine name:** vm1-westus (highlighted with a red box)
- Region:** (US) West US (highlighted with a red box)
- Image:** Ubuntu Server 20.04 LTS - x64 Gen2 (highlighted with a red box)

A sidebar on the right displays 'Estimated monthly costs' for the selected configuration, showing a total of ₹1,192.30.

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.

The screenshot shows the 'Create a virtual machine' wizard at the 'Authentication type' step. The 'Password' option is selected (highlighted with a red box). The 'Inbound port rules' section is expanded, showing the selection of ports 80 and 22 (highlighted with a red box).

Below the 'Inbound port rules' section, the 'Public inbound ports' dropdown is set to 'Allow selected ports' (highlighted with a red box), and the 'Select inbound ports' dropdown also lists 'HTTP (80), SSH (22)' (highlighted with a red box).

The 'Next : Disks >' button is highlighted with a red box, indicating the next step in the wizard.

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.

A screenshot of the Microsoft Azure portal showing the Virtual machines list. The page title is "Virtual machines - Microsoft Azure". The search bar contains "Search resources, services, and docs (G+/-)". The filter bar includes "Subscription equals all", "Type equals all", "Resource group equals all", "Location equals all", and "Add filter". The table headers are Name, Subscription, Resource group, Location, Status, Operating system, Size, Public IP address, and Disks. There are four records displayed:

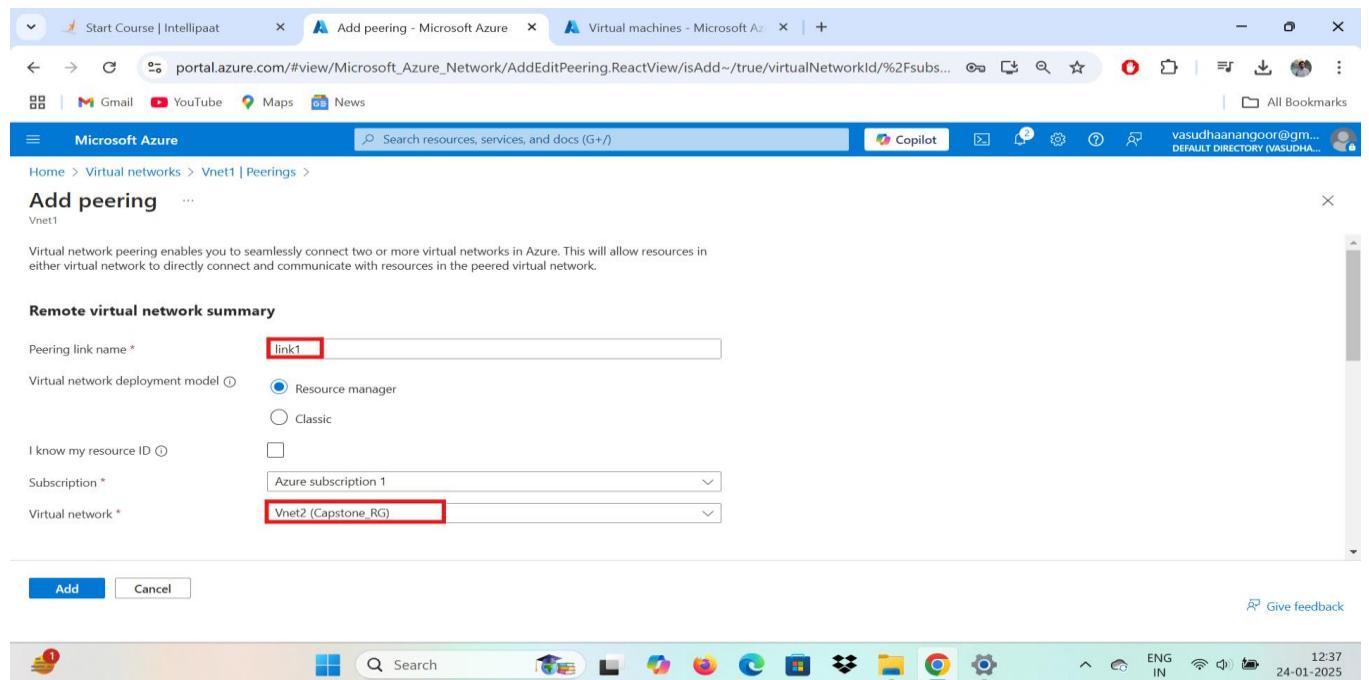
Name	Subscription	Resource group	Location	Status	Operating system	Size	Public IP address	Disks
vm1-uscentral	Azure subscription 1	Capstone_RG	Central US	Running	Linux	Standard_B1s	52.173.74.177	1
vm1-westus	Azure subscription 1	Capstone_RG	West US	Running	Linux	Standard_B1s	13.91.182.16	1
vm2-uscentral	Azure subscription 1	Capstone_RG	Central US	Running	Linux	Standard_B1s	40.77.6.87	1
vm2-westus	Azure subscription 1	CAPSTONE_RG	West US	Running	Linux	Standard_B1s	40.78.123.165	1

At the bottom, there are navigation links for < Previous, Page 1 of 1, Next >, and a "Give feedback" button. The taskbar at the bottom shows various application icons.

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.

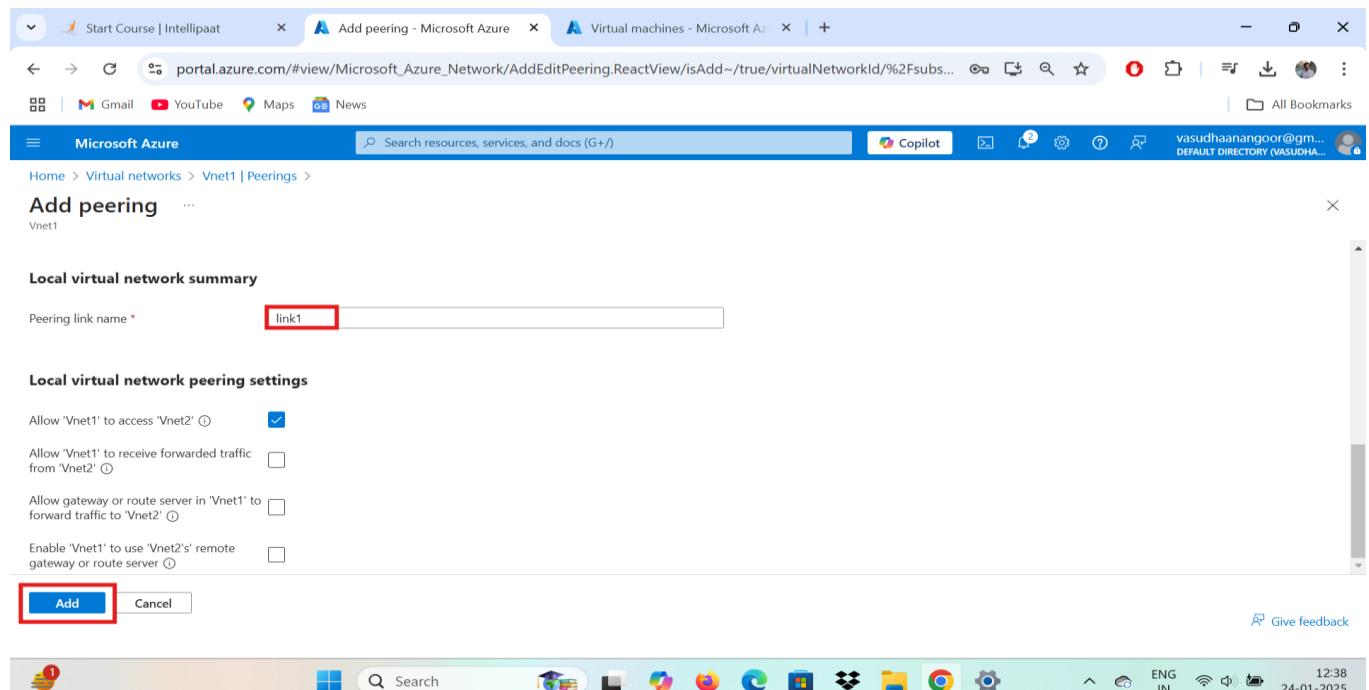
A screenshot of the Microsoft Azure portal showing the Vnet1 | Peerings page. The page title is "Vnet1 | Peerings" under "Virtual networks". The left sidebar shows "Virtual networks" with "Vnet1" selected. The main area has a heading "Virtual network peering enables you to seamlessly connect two or more virtual networks in Azure. The virtual networks appear as one for connectivity purposes. Learn more". Below it is a search bar and a table header with columns "Name", "Peering sync status", "Peerin...", "Remo...", and "Virtu...". A message "Showing all 0 items" is displayed. At the top right, there is a "+ Add" button. The table body is currently empty. The taskbar at the bottom shows various application icons.

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



The screenshot shows the 'Add peering' page in the Microsoft Azure portal. The 'Peering link name' field contains 'link1'. The 'Virtual network deployment model' section shows 'Resource manager' selected. Under 'Subscription', 'Azure subscription 1' is chosen. In the 'Virtual network' dropdown, 'Vnet2 (Capstone_RG)' is selected. At the bottom, there are 'Add' and 'Cancel' buttons, with 'Add' being highlighted by a red box.

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



The screenshot shows the 'Add peering' page in the Microsoft Azure portal. The 'Peering link name' field contains 'link1'. Under 'Local virtual network peering settings', 'Allow 'Vnet1' to access 'Vnet2'' is checked. The 'Add' button is highlighted with a red box.

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.

The screenshot shows the Microsoft Azure Storage accounts page. A red box highlights the 'Storage accounts' link in the navigation bar. Another red box highlights the '+ Create' button. The page displays one record:

Name	Type	Kind	Resource group	Location	Subscription
csg10032003e6d6d321	Storage account	StorageV2	cloud-shell-storage-centralindia	Central India	Azure subscription 1

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.

The screenshot shows the 'Create a storage account' wizard. In the 'Instance details' step, the following fields are filled:

- Subscription: Azure subscription 1
- Resource group: Capstone_RG (highlighted with a red box)
- Storage account name: projectstorage00 (highlighted with a red box)
- Region: (US) Central US
- Primary service: Select a primary service
- Performance: Standard (radio button selected)
- Redundancy: Locally-redundant storage (LRS) (highlighted with a red box)

The 'Review + create' button is highlighted with a red box at the bottom of the form.

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.

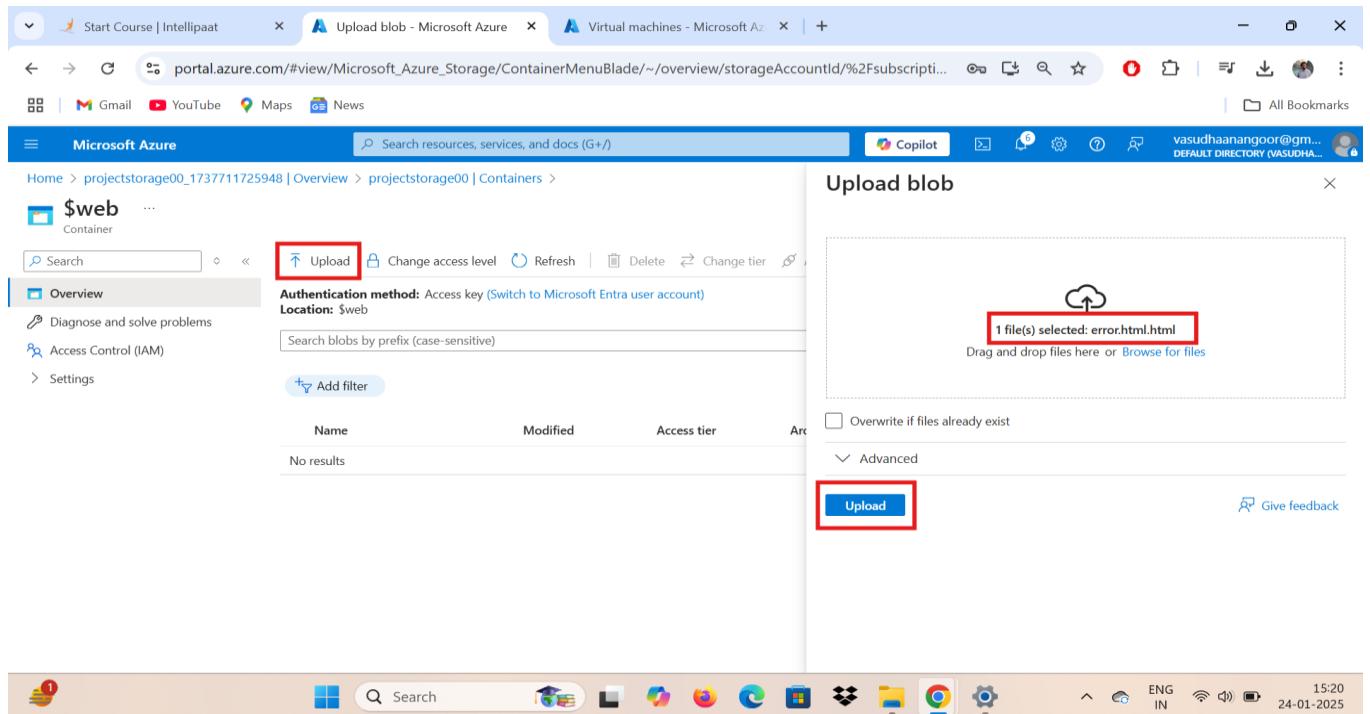
The screenshot shows the Microsoft Azure portal interface for a storage account named 'projectstorage00'. The 'Static website' section is highlighted with a red box. The 'Enabled' button is also highlighted with a red box. The 'Error document path' field contains 'error.html', which is also highlighted with a red box. The 'Save' button at the top right is visible. The left sidebar shows other management options like Data management, Settings, and Data storage.

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

The screenshot shows the 'Containers' section of the Azure Storage account 'projectstorage00'. A table lists two containers: '\$logs' and '\$web'. The '\$web' container is highlighted with a red box. The table columns are 'Name', 'Last modified', 'Anonymous access level', and 'Lease state'. Both containers are private and available. The left sidebar shows other storage-related options like Overview, Activity log, and Data migration.

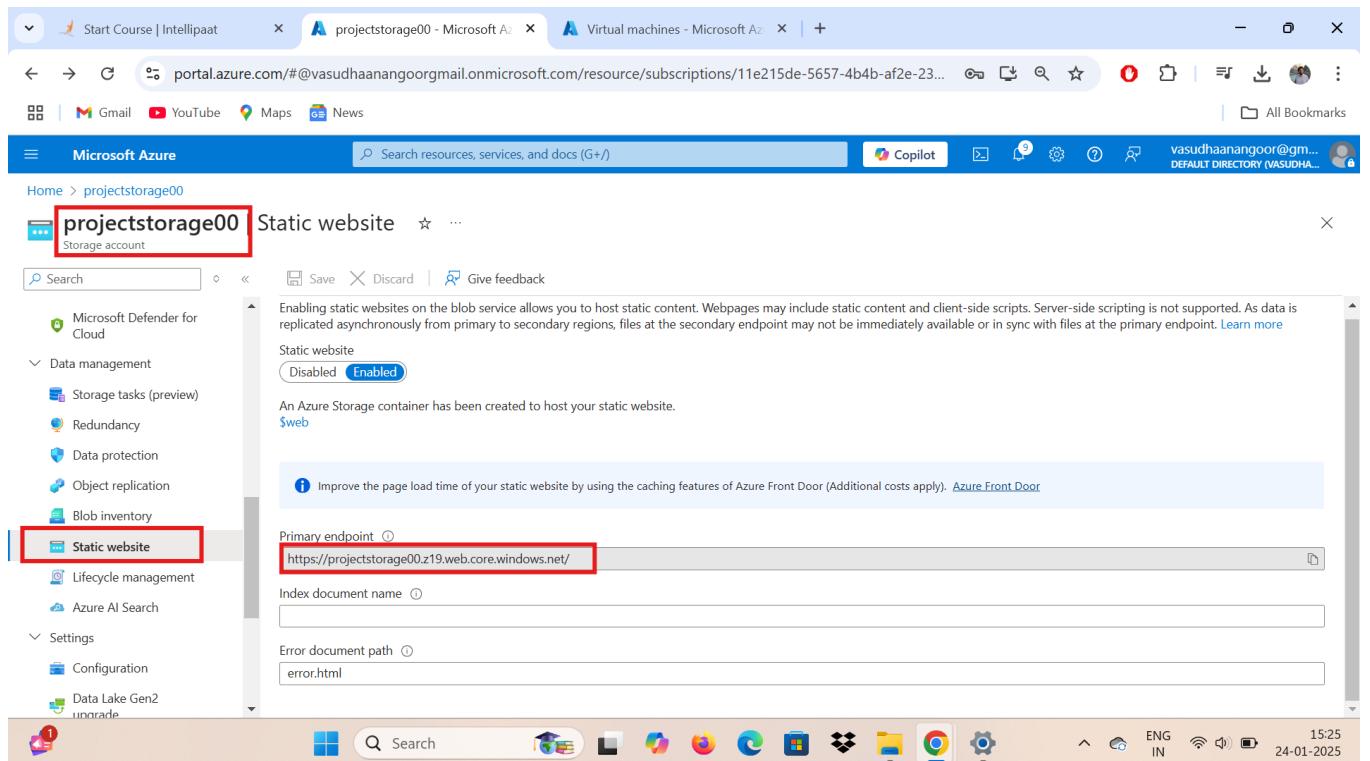
Name	Last modified	Anonymous access level	Lease state
\$logs	1/24/2025, 3:12:53 PM	Private	Available
\$web	1/24/2025, 3:17:54 PM	Private	Available

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



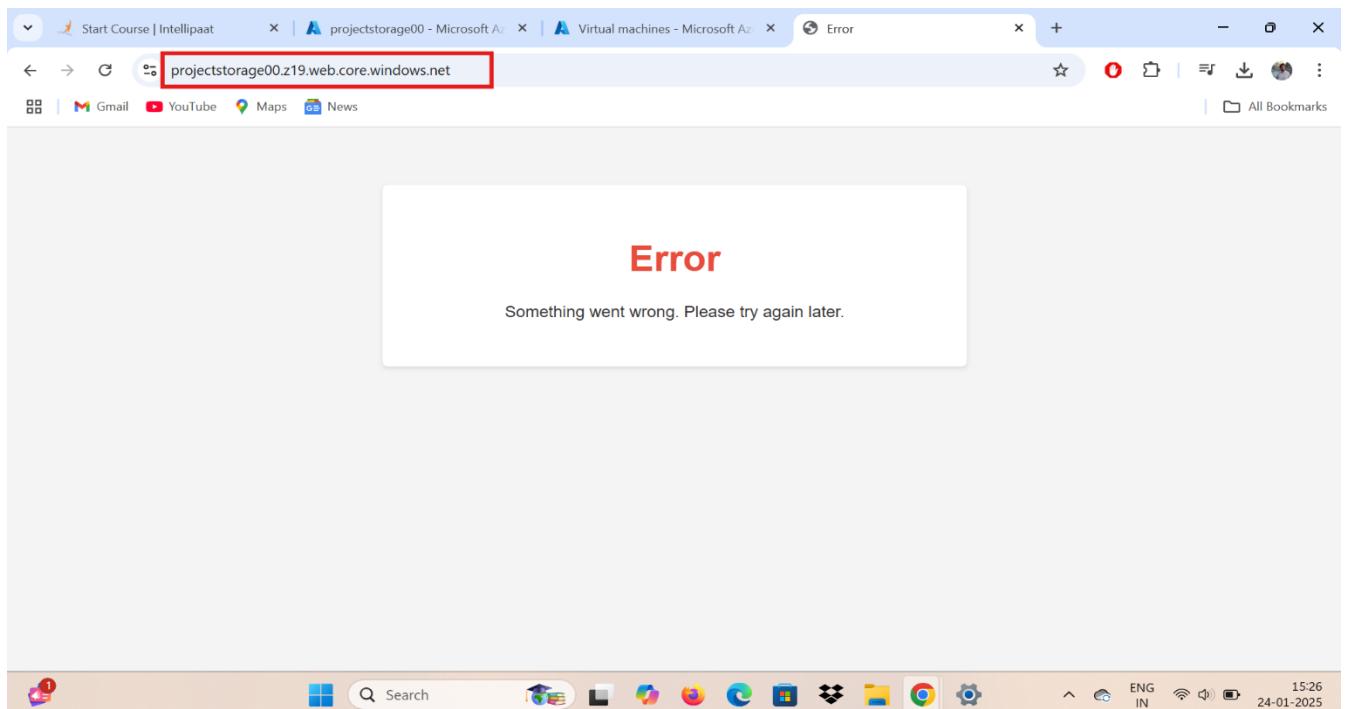
The screenshot shows the Microsoft Azure portal with the 'Upload blob' interface. On the left, there's a sidebar for a storage account named '\$web'. The main area shows a table with one row labeled 'No results'. To the right, a large modal window titled 'Upload blob' is open. It contains a file selection area with a cloud icon and the text '1 file(s) selected: error.html.html'. Below this is a checkbox for 'Overwrite if files already exist'. At the bottom of the modal is a prominent blue 'Upload' button, which is highlighted with a red rectangle. The status bar at the bottom of the browser window shows the date and time as 24-01-2025.

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



The screenshot shows the Microsoft Azure portal with the 'Static website' configuration page for a storage account named 'projectstorage00'. The left sidebar has a 'Static website' section highlighted with a red rectangle. The main pane displays the 'Primary endpoint' field, which contains the URL 'https://projectstorage00.z19.web.core.windows.net/'. Other fields include 'Index document name' (empty) and 'Error document path' ('error.html'). The status bar at the bottom of the browser window shows the date and time as 24-01-2025.

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.

A screenshot of the Microsoft Azure Storage account interface. On the left, there is a navigation sidebar with sections like "Storage browser", "Storage Mover", "Partner solutions", "Data storage", "Containers" (which is highlighted with a red box), "File shares", "Queues", "Tables", "Security + networking", "Networking", "Front Door and CDN", "Access keys", "Shared access signature", and "Encryption". The main content area shows a "Containers" list with two entries: "\$logs" and "\$web". Above this list, there is a "New container" form. The "Name" field is filled with "upload" (also highlighted with a red box). The "Anonymous access level" dropdown is set to "Private (no anonymous access)". At the bottom right of the form, there is a large blue "Create" button (also highlighted with a red box). The top of the screen shows the Azure search bar and some pinned icons. The bottom taskbar shows the date/time "24-01-2025 15:27".

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.

The screenshot shows the Microsoft Azure Load balancing console. The left sidebar has 'Load Balancing Services' expanded, with 'Application Gateway' selected. The main area displays a message: 'No application gateways to display'. Below it, a brief description of Azure Application Gateway is provided, followed by a 'Create application gateway' button. The browser's address bar shows 'portal.azure.com/#view/Microsoft_Azure_Network/LoadBalancingHubMenuBlade/~/applicationgateway'. The taskbar at the bottom shows various pinned icons and the date '24-01-2025'.

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:

The screenshot shows the 'Create application gateway' wizard. It asks to select a subscription and resource group. The 'Subscription' dropdown is set to 'Azure subscription 1' and the 'Resource group' dropdown is set to 'Capstone RG'. Under 'Instance details', the 'Application gateway name' is 'app_gateway01', 'Region' is 'Central US', 'Tier' is 'Standard V2', and 'Enable autoscaling' is set to 'Yes' with 'Minimum instance count' at 1 and 'Maximum instance count' at 5. At the bottom, there are 'Previous' and 'Next : Frontends >' buttons. The browser's address bar shows 'portal.azure.com/#create/Microsoft.ApplicationGateway-ARM'. The taskbar at the bottom shows various pinned icons and the date '24-01-2025'.

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.

The screenshot shows the 'Create application gateway' wizard in the Microsoft Azure portal. The current step is 'Basics'. The 'Virtual network' dropdown is set to 'Vnet1' and the 'Subnet' dropdown is set to 'subnet_ag (10.0.1.0/24)'. Both dropdowns are highlighted with red boxes. At the bottom, the 'Next : Frontends >' button is also highlighted with a red box. The browser's address bar shows 'portal.azure.com/#create/Microsoft.ApplicationGateway-ARM'.

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

The screenshot shows the 'Create application gateway' wizard in the Microsoft Azure portal, currently at Step 2: Frontends. The 'Public IPv4 address' dropdown is set to '(New) newip' and is highlighted with a red box. At the bottom, the 'Next : Backends >' button is highlighted with a red box. The browser's address bar shows 'https://portal.azure.com/#'.

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.

The screenshot shows the Microsoft Azure portal interface for creating an Application Gateway. The 'Backends' tab is selected. A red box highlights the 'Add a backend pool' button. The 'Targets' section shows one item: 'Virtual machine' with 'vm1-uscentral774 (10.0.0.4)' selected. The 'Add' button at the bottom right of the modal window is highlighted with a red box.

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

The screenshot shows the Microsoft Azure portal interface for creating an Application Gateway. The 'Backends' tab is selected. A red box highlights the 'Add a backend pool' button. The 'Targets' section shows one item: 'Virtual machine' with 'vm2-uscentral274 (10.0.0.5)' selected. The 'Next : Configuration >' button at the bottom left of the modal window is highlighted with a red box.

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

The screenshot shows the Microsoft Azure portal with the URL portal.azure.com/#create/Microsoft.ApplicationGateway-ARM. The 'Create application gateway' wizard is open, and the 'Configuration' tab is selected. On the right, under 'Routing rules', there is a red box around the '+ Add a routing rule' button. Other sections like 'Frontends', 'Backend pools', and 'Tags' are also visible.

Give the rule name as “rule1” , priority as 1 and Listener name as “name”.

The screenshot shows the 'Add a routing rule' dialog box. The 'Rule name' field is filled with 'rule1'. The 'Priority' field is set to '1'. The 'Listener name' field is filled with 'name'. Other fields like 'Frontend IP', 'Protocol', 'Port', and 'Listener type' are also visible. At the bottom, there are 'Add' and 'Cancel' buttons.

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.

The screenshot shows the 'Add a routing rule' configuration in the Azure portal. The 'Custom error pages' section is highlighted with a red box, showing two entries: 'Bad Gateway - 502' and 'Forbidden - 403', both pointing to the URL 'https://projectstorage00.z19.web.core.windows.net/error.html'. The 'Backend targets' section is also highlighted with a red box.

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

The screenshot shows the 'Add a routing rule' configuration in the Azure portal. The 'Backend targets' section is highlighted with a red box, showing 'pool2' selected as the target type and 'default' selected as the backend setting. A button 'Add multiple targets to create a path-based rule' is highlighted with a red box.

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.

The screenshot shows the Microsoft Azure portal interface for creating an Application Gateway. On the left, there's a navigation pane with 'Create application gateway' selected. The main area is titled 'Add a path' with sub-sections for 'Target type', 'Path', 'Target name', 'Backend settings', and 'Backend target'. Under 'Target type', 'Backend pool' is selected. In the 'Path' field, '/upload' is entered. In the 'Target name' field, 'upload' is selected. Under 'Backend settings', 'pool1' is chosen as the 'Backend target'. At the bottom right of the dialog, the 'Add' button is highlighted with a red box.

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.

The screenshot shows the 'Review + create' step of the application gateway creation wizard. A green banner at the top indicates 'Validation passed'. Below it, the configuration summary includes sections for 'Basics', 'Frontends', 'Backends', 'Configuration', 'Tags', and 'Review + create'. Under 'Basics', various parameters like 'Subscription', 'Resource group', 'Name', 'Region', 'Tier', etc., are listed. At the bottom, the 'Create' button is highlighted with a red box.

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.

The screenshot shows the 'Create application gateway' wizard in the Microsoft Azure portal. The current step is 'Basics'. The 'Subscription' dropdown is set to 'Azure subscription 1'. The 'Resource group' dropdown is set to 'Capstone_RG' and has a red box around it. The 'Application gateway name' input field contains 'app_gateway02' and has a red box around it. The 'Region' dropdown is set to 'West US' and has a red box around it. Below the form are 'Previous' and 'Next : Frontends >' buttons. At the bottom, there's a taskbar with various icons and a status bar showing 'ENG IN' and the date '24-01-2025'.

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 screenshot shows the 'Create application gateway' wizard in the Microsoft Azure portal. The current step is 'Configure virtual network'. Under 'Virtual network', 'Vnet2' is selected in the dropdown. Under 'Subnet', 'subnet_ag (20.0.1.0/24)' is selected in the dropdown and has a red box around it. Below the form are 'Previous' and 'Next : Frontends >' buttons. At the bottom, there's a taskbar with various icons and a status bar showing 'ENG IN' and the date '24-01-2025'.

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:

Name	Public IP	Resource group	Location	Subscription
app_gateway01	135.233.107...	Capstone_RG	Central US	Azure subscription 1
app_gateway02	13.91.52.21	Capstone_RG	West US	Azure subscription 1

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 .

Name	Subscription	Resource group	Location	Status	Operating system	Size	Public IP address	Disk
vm1-uscentral	Azure subscription 1	CAPSTONE_RG	Central US	Running	Linux	Standard_B1s	52.173.74.177	1
vm1-westus	Azure subscription 1	CAPSTONE_RG	West US	Running	Linux	Standard_B1s	13.91.182.16	1
vm2-uscentral	Azure subscription 1	CAPSTONE_RG	Central US	Running	Linux	Standard_B1s	40.77.6.87	1
vm2-westus	Azure subscription 1	CAPSTONE_RG	West US	Running	Linux	Standard_B1s	40.78.123.165	1

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

The image shows four separate terminal windows, each representing a different VM. Each window displays the command `ssh azureuser@[IP]` being run, followed by a series of prompts asking if the host is known and if the user wants to proceed. The user's responses ('yes') and the password are highlighted with red boxes. The terminals are arranged in a 2x2 grid.

- Top Left Terminal:** IP: 52.173.74.177
- Top Right Terminal:** IP: 13.91.182.16
- Bottom Left Terminal:** IP: 40.77.6.87
- Bottom Right Terminal:** IP: 40.78.123.165

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.

The image shows four separate terminal windows, each representing a different VM. Each window displays the command `git clone https://github.com/azcloudberg/azproject.git` being run. The output shows the progress of cloning the repository, including object enumeration, counting, compressing, and receiving objects, followed by resolving deltas. The terminals are arranged in a 2x2 grid.

- Top Left Terminal:** VM1-USCentral
- Top Right Terminal:** VM1-WestUS
- Bottom Left Terminal:** VM2-USCentral
- Bottom Right Terminal:** VM2-WestUS

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 .

The screenshot shows four terminal windows side-by-side:

- Top Left:** azuser@vm1-uscentral: ~\$ ls
azproject
- Top Right:** azuser@vm1-westus: ~\$ ls
azproject
- Bottom Left:** azuser@vm2-uscentral: ~\$ ls
azproject
azuser@vm2-uscentral:~/azproject\$./vm2.sh
Hit:1 http://archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:3 http://archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:4 http://archive.ubuntu.com/ubuntu focal-security InRelease
Reading package lists... Done
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
Reading package lists... Done
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
- Bottom Right:** azuser@vm2-westus: ~\$ ls
azproject
azuser@vm2-westus:~/azproject\$ cd azproject
azuser@vm2-westus:~/azproject\$./vm2.sh
Hit:1 http://archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:3 http://archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:4 http://archive.ubuntu.com/ubuntu focal-security InRelease
Reading package lists... Done
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:

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).

The screenshot shows four terminal windows side-by-side:

- Top Left:** azuser@vm1-uscentral: ~\$ ls
azproject
azuser@vm1-uscentral:~/azproject\$ ls
README.md config.py index.html vm1.sh
app.py error.html templates vm2.sh
azuser@vm1-uscentral:~/azproject\$ sudo nano config.py
- Top Right:** azuser@vm1-westus: ~\$ ls
azproject
azuser@vm1-westus:~/azproject\$ ls
README.md config.py index.html vm1.sh
app.py error.html templates vm2.sh
azuser@vm1-westus:~/azproject\$ sudo nano config.py
- Bottom Left:** azuser@vm2-uscentral: ~\$ Enabling conf charset.
Enabling conf localized-error-pages.
Enabling conf other-vhosts-access-log.
Enabling conf security.
Enabling conf serve-cgi-bin.
Enabling site 000-default.
Created symlink /etc/systemd/system/multi-user.target.wants/apache2.service → /lib/systemd/system/apache2.service.
Created symlink /etc/systemd/system/multi-user.target.wants/apache-htcacheclean.service → /lib/systemd/system/apache-htcacheclean.service.
Processing triggers for ufw (0.36-6ubuntu1.1) ...
Processing triggers for systemd (245.4-4ubuntu3.24) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for libc-bin (2.31-0ubuntu9.16) ...
azuser@vm2-uscentral:~/azproject\$ |
- Bottom Right:** azuser@vm2-westus: ~\$ Enabling conf charset.
Enabling conf localized-error-pages.
Enabling conf other-vhosts-access-log.
Enabling conf security.
Enabling conf serve-cgi-bin.
Enabling site 000-default.
Created symlink /etc/systemd/system/multi-user.target.wants/apache2.service → /lib/systemd/system/apache2.service.
Created symlink /etc/systemd/system/multi-user.target.wants/apache-htcacheclean.service → /lib/systemd/system/apache-htcacheclean.service.
Processing triggers for ufw (0.36-6ubuntu1.1) ...
Processing triggers for systemd (245.4-4ubuntu3.24) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for libc-bin (2.31-0ubuntu9.16) ...
azuser@vm2-westus:~/azproject\$ |

```

azuser@vm1-uscentral: ~/ + - x
GNU nano 4.8 config.py Modified
[DEFAULT]
# Account name
account =projectstorage00
# Azure Storage account access key
key =VmMoF/uv16lHPB/wrErBoc4lEQ0qf7mLN+HBrltk5zD2gJVcH1cnYzyWNqP
# Container name
container =upload

azuser@vm1-westus: ~/az + - x
GNU nano 4.8 config.py Modified
[DEFAULT]
# Account name
account =projectstorage00
# Azure Storage account access key
key =VmMoF/uv16lHPB/wrErBoc4lEQ0qf7mLN+HBrltk5zD2gJVcH1cnYzyWNqP
# Container name
container =upload

azuser@vm2-uscentral: ~/ + - x
Enabling conf charset.
Enabling conf localized-error-pages.
Enabling conf other-vhosts-access-log.
Enabling conf security.
Enabling conf serve-cgi-bin.
Enabling site 000-default.
Created symlink /etc/systemd/system/multi-user.target.wants/apache2.service → /lib/systemd/system/apache2.service.
Created symlink /etc/systemd/system/multi-user.target.wants/apache-htcacheclean.service → /lib/systemd/system/apache-htcacheclean.service.
Processing triggers for ufw (0.36-6ubuntu1.1) ...
Processing triggers for systemd (245.4-4ubuntu3.24) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for libc-bin (2.31-0ubuntu9.16) ...
azuser@vm2-uscentral:~/azproject$ |

azuser@vm2-westus: ~/az + - x
Enabling conf charset.
Enabling conf localized-error-pages.
Enabling conf other-vhosts-access-log.
Enabling conf security.
Enabling conf serve-cgi-bin.
Created symlink /etc/systemd/system/multi-user.target.wants/apache2.service → /lib/systemd/system/apache2.service.
Created symlink /etc/systemd/system/multi-user.target.wants/apache-htcacheclean.service → /lib/systemd/system/apache-htcacheclean.service.
Processing triggers for ufw (0.36-6ubuntu1.1) ...
Processing triggers for systemd (245.4-4ubuntu3.24) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for libc-bin (2.31-0ubuntu9.16) ...
azuser@vm2-westus:~/azproject$ |

```

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

```

azuser@vm1-uscentral: ~/ + - x
azuser@vm1-uscentral:~$ ls
azproject
azuser@vm1-uscentral:~$ cd azproject
azuser@vm1-uscentral:~/azproject$ ls
README.md config.py index.html vm1.sh
app.py error.html templates vm2.sh
azuser@vm1-uscentral:~/azproject$ sudo nano config.py
azuser@vm1-uscentral:~/azproject$ ./vm1.sh
Rules updated
Rules updated (v6)
Hit:1 http://azure.archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://azure.archive.ubuntu.com/ubuntu focal-updates InRelease [128 kB]
Hit:3 http://azure.archive.ubuntu.com/ubuntu focal-backports InR

azuser@vm1-westus: ~/az + - x
azuser@vm1-westus:~$ ls
azproject
azuser@vm1-westus:~$ cd azproject
azuser@vm1-westus:~/azproject$ ls
README.md config.py index.html vm1.sh
app.py error.html templates vm2.sh
azuser@vm1-westus:~/azproject$ sudo nano config.py
azuser@vm1-westus:~/azproject$ ./vm1.sh
Rules updated
Rules updated (v6)
Hit:1 http://azure.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://azure.archive.ubuntu.com/ubuntu focal-updates InRe
ease
Hit:3 http://azure.archive.ubuntu.com/ubuntu focal-backports InRe

azuser@vm2-uscentral: ~/ + - x
Enabling conf charset.
Enabling conf localized-error-pages.
Enabling conf other-vhosts-access-log.
Enabling conf security.
Enabling conf serve-cgi-bin.
Enabling site 000-default.
Created symlink /etc/systemd/system/multi-user.target.wants/apache2.service → /lib/systemd/system/apache2.service.
Created symlink /etc/systemd/system/multi-user.target.wants/apache-htcacheclean.service → /lib/systemd/system/apache-htcacheclean.service.
Processing triggers for ufw (0.36-6ubuntu1.1) ...
Processing triggers for systemd (245.4-4ubuntu3.24) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for libc-bin (2.31-0ubuntu9.16) ...
azuser@vm2-uscentral:~/azproject$ |

azuser@vm2-westus: ~/az + - x
Enabling conf charset.
Enabling conf localized-error-pages.
Enabling conf other-vhosts-access-log.
Enabling conf security.
Enabling conf serve-cgi-bin.
Enabling site 000-default.
Created symlink /etc/systemd/system/multi-user.target.wants/apache2.service → /lib/systemd/system/apache2.service.
Created symlink /etc/systemd/system/multi-user.target.wants/apache-htcacheclean.service → /lib/systemd/system/apache-htcacheclean.service.
Processing triggers for ufw (0.36-6ubuntu1.1) ...
Processing triggers for systemd (245.4-4ubuntu3.24) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for libc-bin (2.31-0ubuntu9.16) ...
azuser@vm2-westus:~/azproject$ |

```

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.

The screenshot shows the Microsoft Azure portal with the URL https://portal.azure.com/#@vasudhaaanangoorgmail.onmicrosoft.com/resource/subscriptions/11e215de-5657-4b4b-af2e-23292f4e2c28/resourceGroups/Capstone_RG/providers/Microsoft.Network/publicIPAddresses/app_gateway01. The page displays the 'app_gateway01' application gateway. The 'Frontend public IP address' field is highlighted with a red box, showing the value **135.233.107.61 (newip)**.

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

The screenshot shows the Microsoft Azure portal with the URL https://portal.azure.com/#@vasudhaaanangoorgmail.onmicrosoft.com/resource/subscriptions/11e215de-5657-4b4b-af2e-23292f4e2c28/resourceGroups/Capstone_RG/providers/Microsoft.Network/publicIPAddresses/app_gateway01/configurations/newip. The page displays the configuration for the 'newip' public IP address. The 'Configuration' tab is selected. The 'DNS name label (optional)' field is highlighted with a red box, showing the value **appg1**. The 'Save' button is also highlighted with a red box.

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.

The screenshot shows the Microsoft Azure portal interface. A new tab titled "Create Traffic Manager profile" is open. The "Name" field contains "capstonetrafficmg". The "Resource group" dropdown is set to "Capstone RG". At the bottom right of the form, there is a prominent blue "Create" button.

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.

The screenshot shows the Microsoft Azure portal interface. On the left, the navigation menu is expanded, and the "Endpoints" option under "Load balancing | Traffic Manager" is selected. In the main pane, a sub-menu titled "Endpoints" is open, and the "Add" button is highlighted with a red box. To the right, a modal window titled "Add endpoint" is displayed for the "capstonetrafficmg" profile. The "Name" field is filled with "ep1". Under "Target resource type", "Public IP address" is selected. Below it, a dropdown menu shows "newip (135.233.107.61)". At the bottom of the modal, there is a large blue "Add" button.

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.

The screenshot shows the Microsoft Azure portal interface. On the left, there's a sidebar with options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Settings (Configuration, Real user measurements, Traffic view), Endpoints (selected), Properties, Locks, and Monitoring. The main area shows a list of endpoints with one entry named 'ep1'. To the right, a modal window titled 'Add endpoint' is open. It has fields for Type (set to 'Azure endpoint'), Name (set to 'ep2'), Target resource type (set to 'Public IP address' with 'newip02 (13.91.52.21)' selected), and a 'Custom Header settings' section. At the bottom of the modal is a large blue 'Add' button.

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.

The screenshot displays four terminal windows side-by-side. Each window shows the command `sudo python3 app.py` being run. The output from each terminal indicates that the Flask application is serving on multiple addresses and ports. A warning message at the top of each terminal states: "WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead." The terminals are arranged in a 2x2 grid, with the top row showing VM1 instances and the bottom row showing Ubuntu hosts.

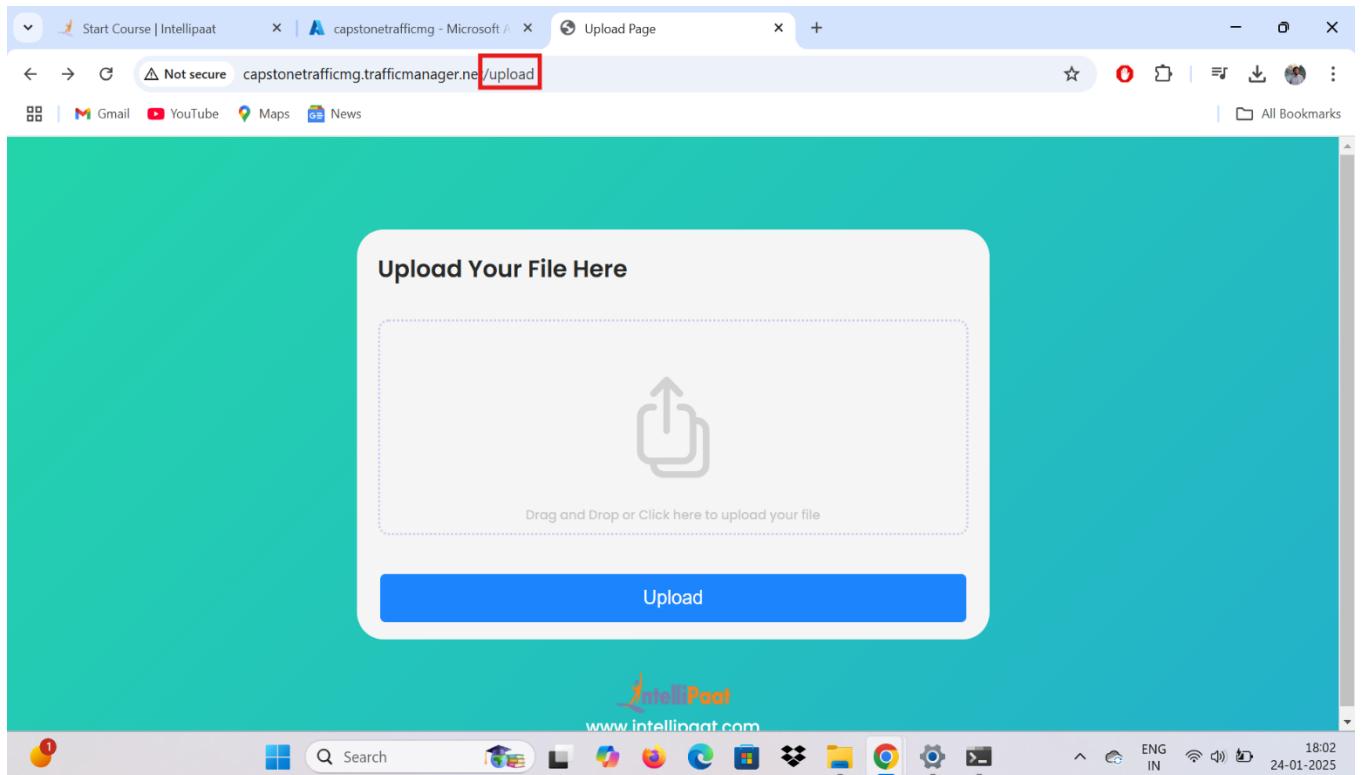
8.VERIFICATION: To verify the implementation of this project, we will copy the DNS name of the traffic manager and paste it in the browser.

The screenshot shows the Microsoft Azure portal interface. The user is viewing a Traffic Manager profile named 'capstonetrafficmg'. On the right side, under the 'Essentials' section, the 'DNS name' field is displayed as 'http://capstonetrafficmg.trafficmanager.net', which is highlighted with a red box. The 'Monitor status' is listed as 'Online'. Below this, there is a table titled 'Search endpoints' showing two entries: 'ep1' and 'ep2', both of which are 'Enabled' and 'Online'. The 'Type' for both is 'Azure endpoint'. The location for 'ep1' is 'Central US' and for 'ep2' is 'West US'. On the left, there is a sidebar with various navigation options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Settings (Configuration, Real user measurements, Traffic view, Endpoints, Properties, Locks), and Monitoring.

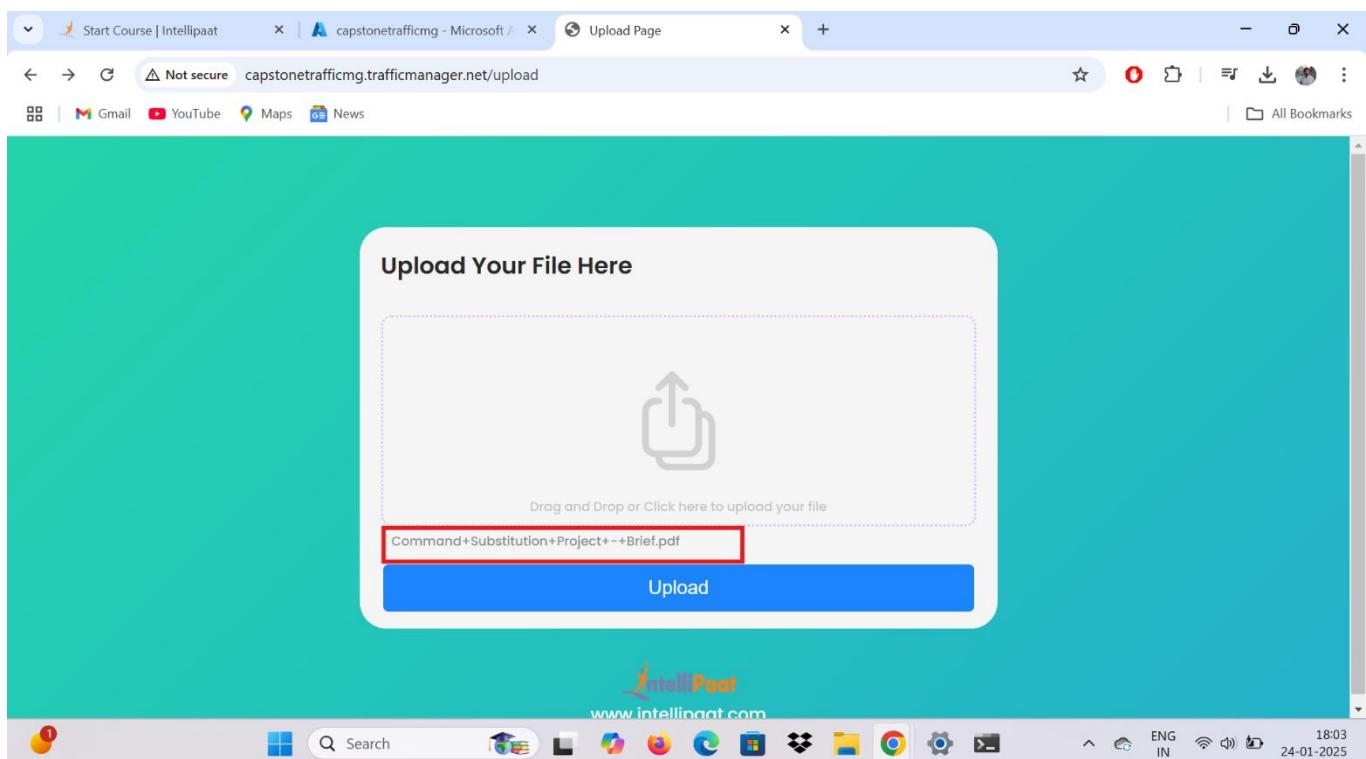
We will be navigated to the home page.

The screenshot shows a web browser window with the URL 'capstonetrafficmg.trafficmanager.net' entered in the address bar. The address bar is highlighted with a red box. The page content features the IntelliPaat logo at the top, followed by the text 'Welcome to the Home Page.' and the website URL 'www.intellipaat.com' at the bottom. The browser's taskbar at the bottom shows various pinned icons and the system tray on the right indicating the date and time as 24-01-2025.

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.

Name	Last modified	Anonymous access level	Lease state
\$logs	1/24/2025, 3:12:53 PM	Private	Available
\$web	1/24/2025, 3:17:54 PM	Private	Available
upload	1/24/2025, 3:28:53 PM	Private	Available

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

Name	Modified	Access tier	Archive status	Blob type	Size
CommandSubstitutionProject-Brief.pdf	1/24/2025, 6:04:50 PM	Hot (Inferred)		Block blob	60.82 KiB