

# **DEVOPS with MULTI-CLOUD**

## **Practice Tasks**

**Institute Name** : V Cube software solutions  
**Course** : DevOps with Multi-Cloud  
**Batch** : 30  
**Trainer** : Krishna reddy sir

**Prepared by** : G.Bhavish  
(MCD-AZ30-024)

# TASK-17 :- Image Creation.

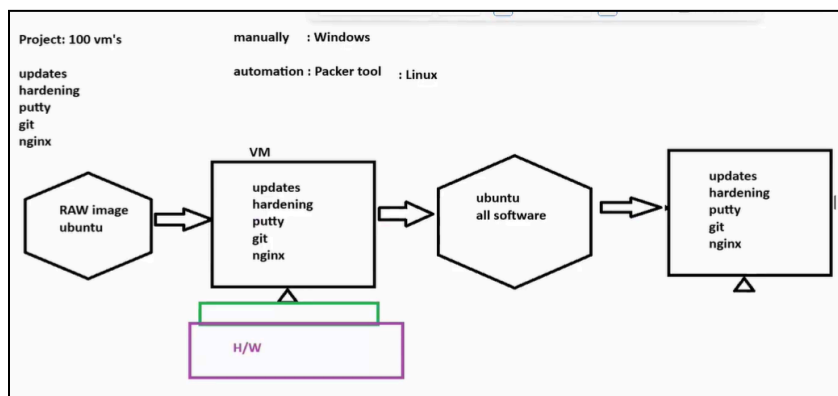
Date : 11/02/26

## Objective :-

To create a reusable custom virtual machine image in Microsoft Azure that enables fast, consistent, and scalable deployment of multiple identical VMs.

## Image Creation :-

- Generally till now when we created a VM we selected images from the market place, those are raw images.
- But, in companies and organizations we use our own created custom images rather than raw images.
- An Azure VM Image is a template that contains the operating system, installed software, configurations, and settings required to create new virtual machines.
- It captures the Operating System, applications, and configurations of an existing VM.
- And helps in creating multiple VMs with the same setup and settings.

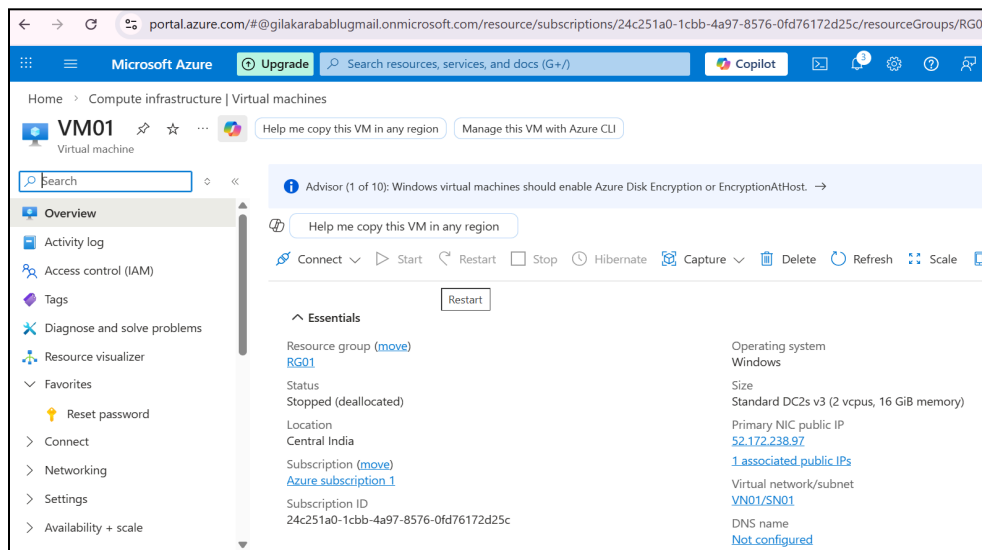


## Creating Image :-

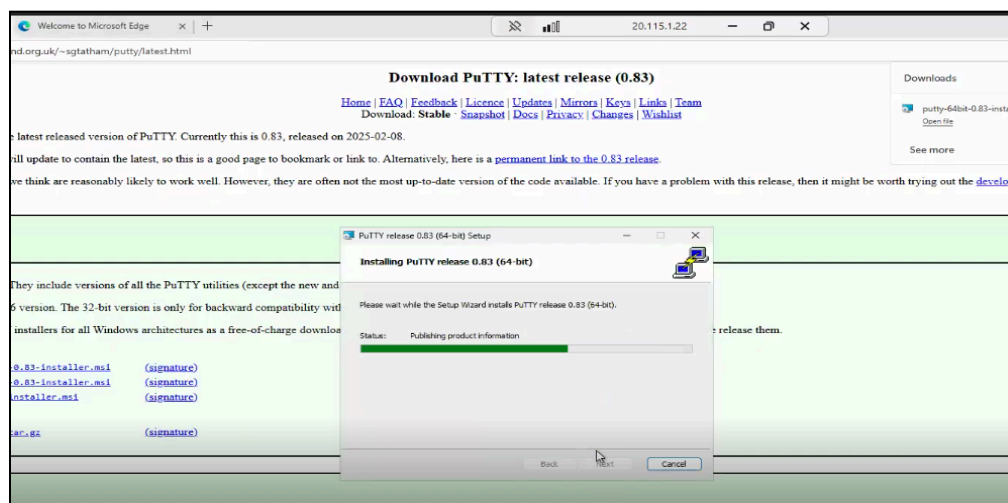
We can create an image in two ways, manually and automation using packer.

### ❑ Creating image manually :-

→ create a windows machine, login to the machine and download putty(64-bit 0.83).



fig(1) created a vm successfully.

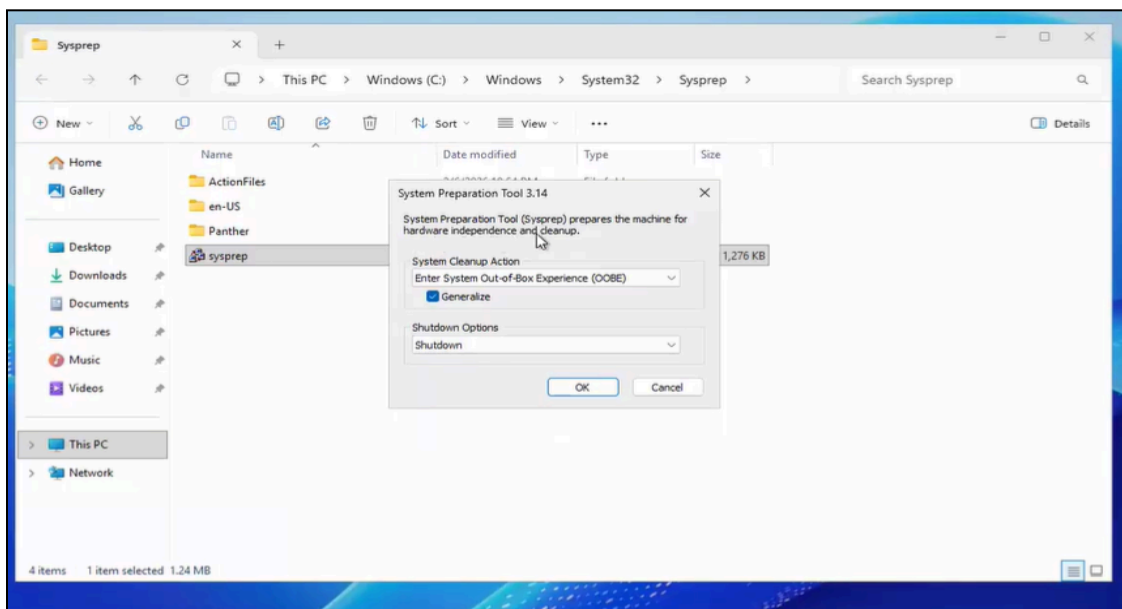
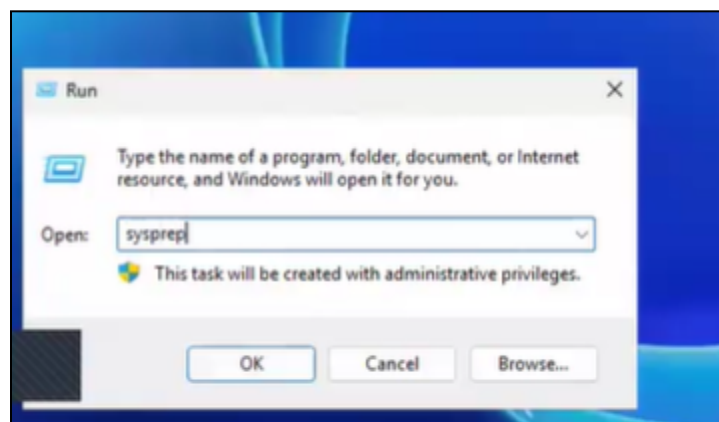


fig(2) successfully installed putty in the machine.

→ Now we need to deprovision/generalise the machine i.e remove the hardware dependencies.

→ we generalise because The cpu attracts the hardware files, which may cause issues in drives, reboot and activation.

- Run sysprep in the machine.
  - Search run>open>type sysprep.



fig(3&4) successfully generalised the machine using sysprep.

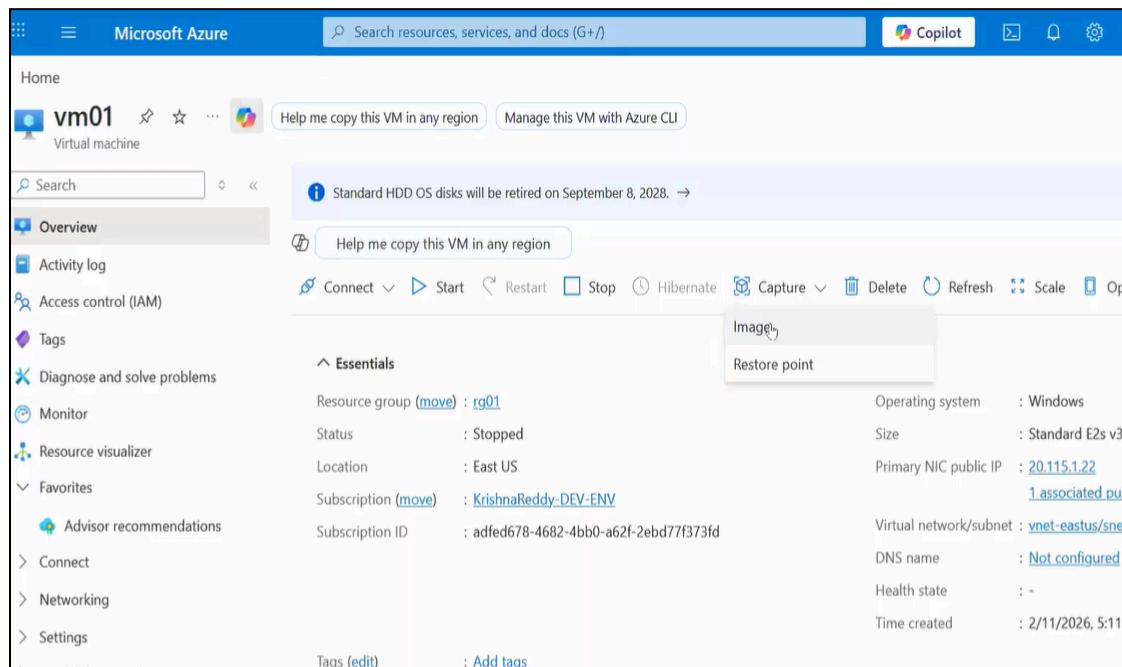
→ Now the vm will be in stop state

Now convert the machine to image :-

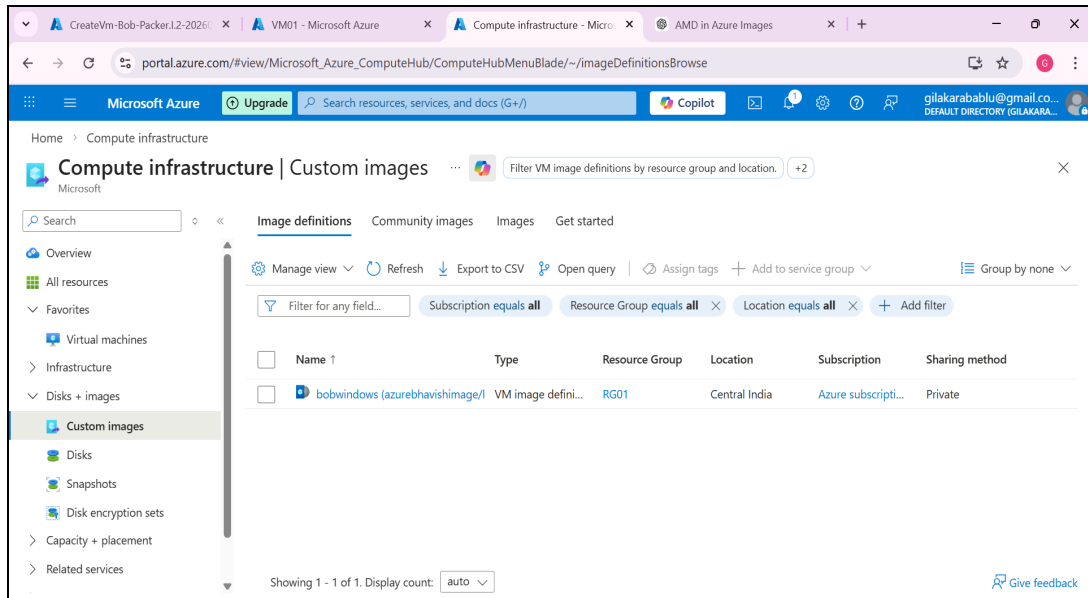
- Goto vm>capture> Image.
- Select the option to automatically delete this vm after creating the image.
- Now after filling the gallery details,create the image.

→ Now we have created the image using the machine, here we installed putty s/w, this s/w will be installed in all the machines which are created using this image.

→ we can use our required s/w and all the machines will have installed it, and they are similar machines too.



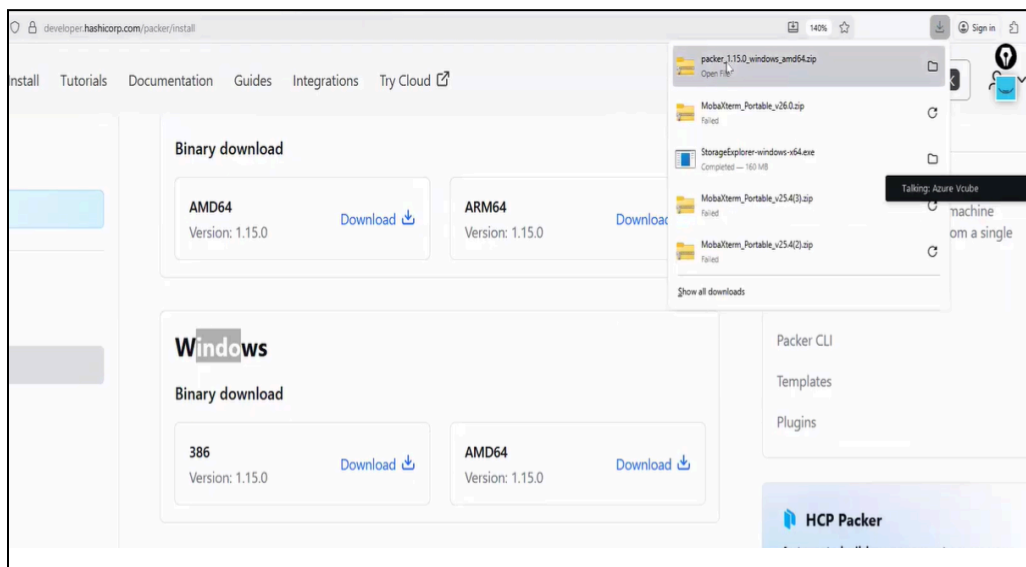
fig(5) machine is in stop mode.



fig(6) successfully created an image {Manually}.

## ❑ Creating Image with Automation using Packer :-

- Download packer for windows in our laptop
- Extract all the files
- Now copy the main packer file and paste it in the C drive with a new folder.



fig(7) downloaded the packer in our system.

→ Now right click and open the terminal.  
→ Now search linux image creation using packer. And copy the json code.  
→ open vs code>C drive>open the packer folder and now create a new json file in vscode under the packer.exe folder.  
→ Now in this file paste the json code. And fill the details in the code.  
→ To fill in details we need to have a service account. Since packer is a 3rd party tool so it should connect to the azure account and execute the code, for this we use authentication method i.e will give service account name.  
Crate service account :-

- Microsoft entra id>manage>app registration>new registration.
  - name = azurebob30 and register.
  - Copy application id for client id.
- For client secret :-
  - azurebob30>manage>certificates&secret
  - Add client secret
  - Copy the value for client secret.
- Tenet id :-
  - Copy the tenet id(application id).
- Subscription id :-
  - Subscription id

→ The new user (azurebob30) should have the right to execute any operation on the subscription, so we need to assign a role .

```
home>subscription>access control(iam)> +add> +add role assignment.>role>privileged administrator roles - owner.
```

- Members :-
  - Assign access to - user,group or service principle
  - Members - select azurebob30
- Condition :-
  - Allow user to assign all roles
- Resource group :-
  - Use the existing one or create a new one.
- Managed image name:-
  - The image will be created on this name.

→ goto terminal and execute the commands

- Code . (if it directs to vs code - json file, then path is correct.)
- .\packer.exe – version (to get available commands)
- .\packer.exe inspect .\packer.json (variables <no variables>)
- .\packer.exe build .\packer.json (the building process will be started.)

```
Windows PowerShell
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\packer> .\packer.exe -- version
Usage: packer [--version] [--help] <command> [<args>]

Available commands are:
  build      build image(s) from template
  console    creates a console for testing variable interpolation
  fix        fixes templates from old versions of packer
  fmt        Rewrites HCL2 config files to canonical format
  hcl2_upgrade transform a JSON template into an HCL2 configuration
  init       Install missing plugins or upgrade plugins
  inspect    see components of a template
  plugins    Interact with Packer plugins and catalog
  validate   check that a template is valid
  version    Prints the Packer version

PS C:\packer>
```

fig(8) executed version command.

```
learn.microsoft.com/en-us/azure/virtual-machines/linux/build-image-with-packer

JSON
Copy

{
  "builders": [{
    "type": "azure-arm",

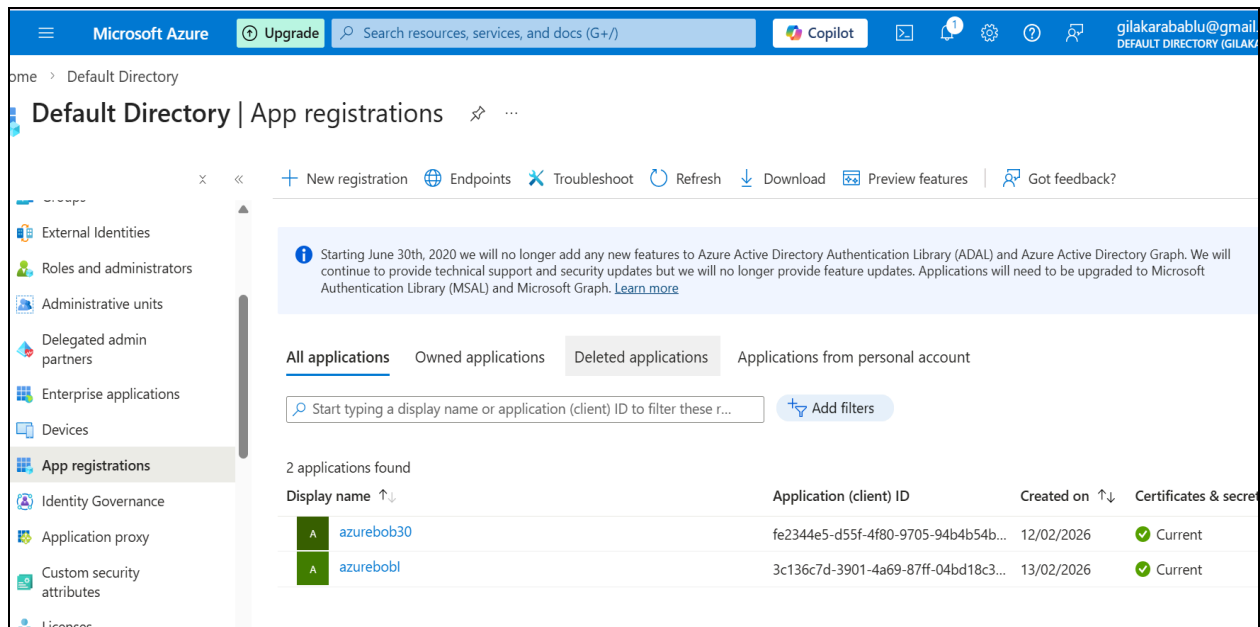
    "client_id": "00001111-aaaa-2222-bbbb-3333cccc4444",
    "client_secret": "0e760437-bf34-4aad-9f8d-870be799c55d",
    "tenant_id": "aaaabbbb-0000-cccc-1111-dddd2222eeee",
    "subscription_id": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",

    "managed_image_resource_group_name": "myResourceGroup",
    "managed_image_name": "myPackerImage",

    "os_type": "Linux",
    "image_publisher": "canonical",
    "image_offer": "0001-com-ubuntu-server-jammy",
    "image_sku": "22_04-lts",

    "azure_tags": {
      "dept": "Engineering",
      "task": "Image deployment"
    }
  }]
}
```

fig(9) code for linux image creation in json format.



fig(10) app registrations.

```

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\packer> code .
PS C:\packer> .\packer.exe inspect .\packer.json
Packer Inspect: JSON mode
Variables:

<No variables>

Builders:

azure-arm

Provisioners:

shell

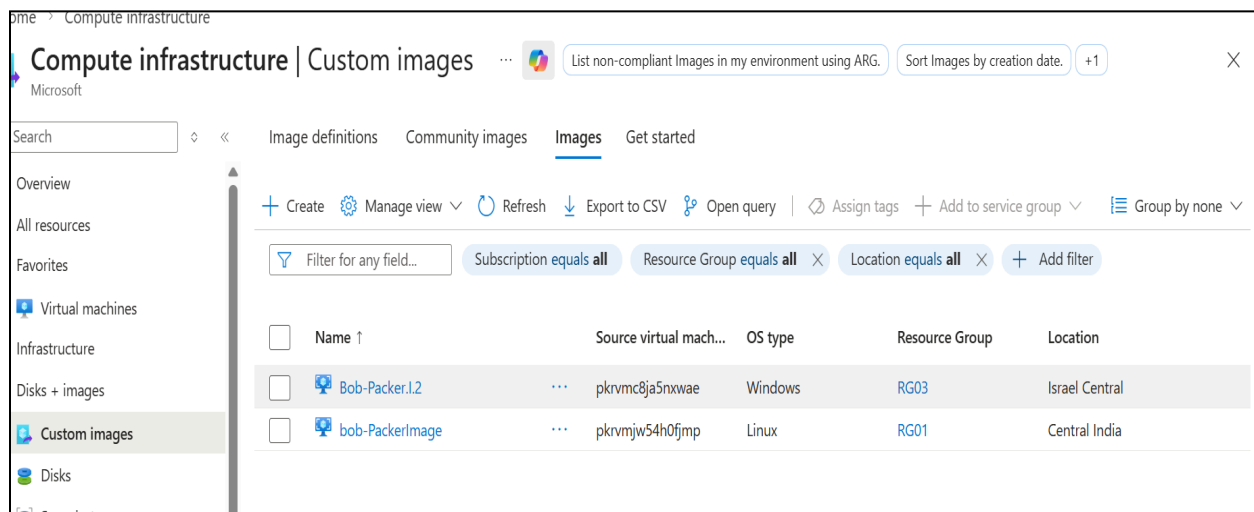
```

fig(11) executed inspect command.

```
Windows PowerShell
PS C:\packer> .\packer.exe build .\packer.json
azure-arm: output will be in this color.

==> azure-arm: Running builder ...
==> azure-arm: Creating Azure Resource Manager (ARM) client ...
==> azure-arm: ARM Client successfully created
==> azure-arm: Getting source image id for the deployment ...
==> azure-arm: -> SourceImageName: '/subscriptions/24c251a0-1cbb-4a97-8576-0fd76172d25c/providers/Microsoft.Compute/lindia/publishers/canonical/ArtifactTypes/vmimage/offers/0001-com-ubuntu-server-jammy/skus/22_04-lts/ver
==> azure-arm: Creating resource group ...
==> azure-arm: -> ResourceGroupName: 'pkr-Resource-Group-jw54h0fjmp'
==> azure-arm: -> Location: 'central india'
==> azure-arm: -> Tags:
==> azure-arm: ->> dept: Engineering
==> azure-arm: ->> task: Image deployment
==> azure-arm: Validating deployment template ...
==> azure-arm: -> ResourceGroupName: 'pkr-Resource-Group-jw54h0fjmp'
==> azure-arm: -> DeploymentName: 'pkrdpjw54h0fjmp'
==> azure-arm: Deploying deployment template ...
==> azure-arm: -> ResourceGroupName: 'pkr-Resource-Group-jw54h0fjmp'
==> azure-arm: -> DeploymentName: 'pkrdpjw54h0fjmp'
==> azure-arm: Getting the VM's IP address ...
==> azure-arm: -> ResourceGroupName: 'pkr-Resource-Group-jw54h0fjmp'
==> azure-arm: -> PublicIPAddressName: 'pkripjw54h0fjmp'
==> azure-arm: -> NicName: 'pkrnijw54h0fjmp'
==> azure-arm: -> Network Connection: 'PublicEndpoint'
==> azure-arm: -> IP Address: '104.211.94.74'
==> azure-arm: Querying the machine's properties ...
==> azure-arm: -> ResourceGroupName: 'pkr-Resource-Group-jw54h0fjmp'
==> azure-arm: -> ComputeName: 'pkrmvjw54h0fjmp'
==> azure-arm: -> Managed OS Disk: '/subscriptions/24c251a0-1cbb-4a97-8576-0fd76172d25c/resourceGroup-jw54h0fjmp/providers/Microsoft.Compute/disks/pkrvmjw54h0fjmp'
```

fig(12) the building of image is started.



fig(13) the images which we created manual & automation.

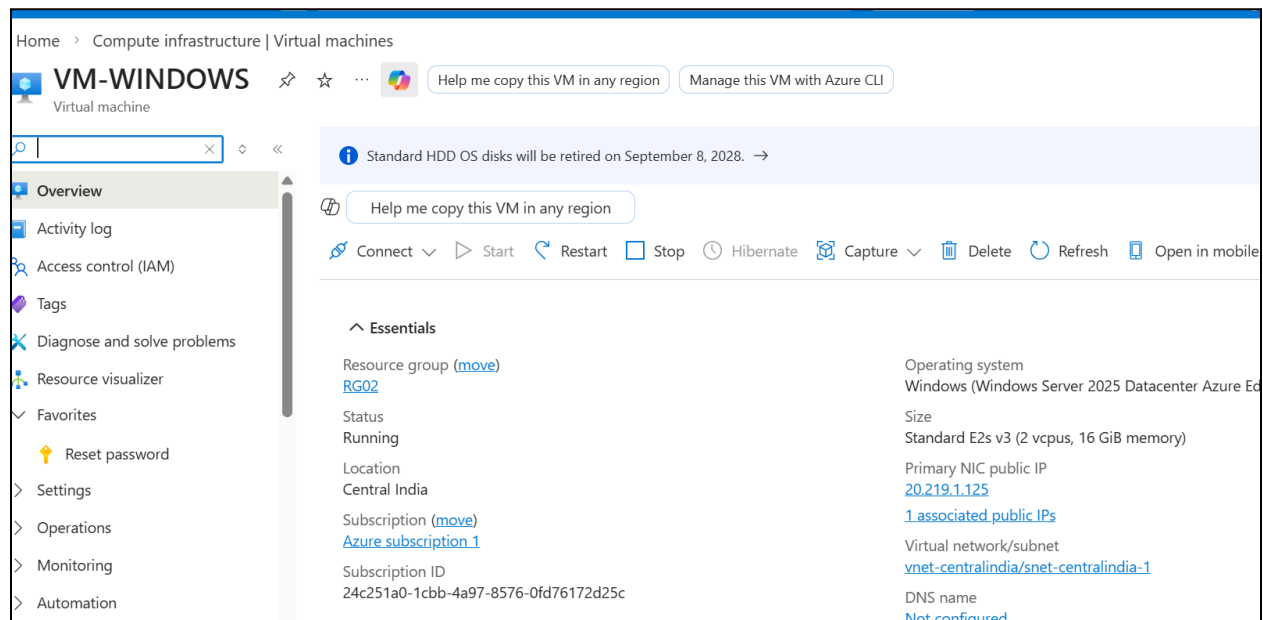
→ now lets create a new windows machine using our custom image.

For using the custom image :-

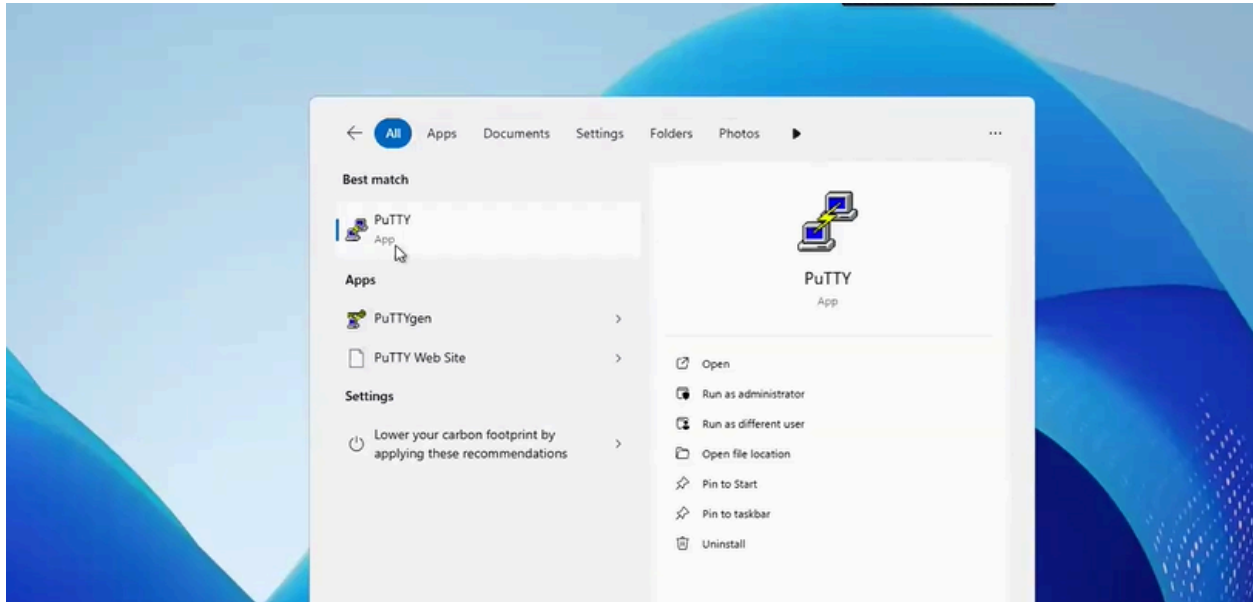
- image>see all images> shared images> select the image we created.

→ Now login to the new windows machine and search for the putty , we can find the putty already installed.

(because we have s/w in the machine which we created the image from.)



fig(14) windows machine.



fig(15) already putty installed.

→ like this we use packer and make the images and with that image we can make more similar machines.