

# Final Usecase Document

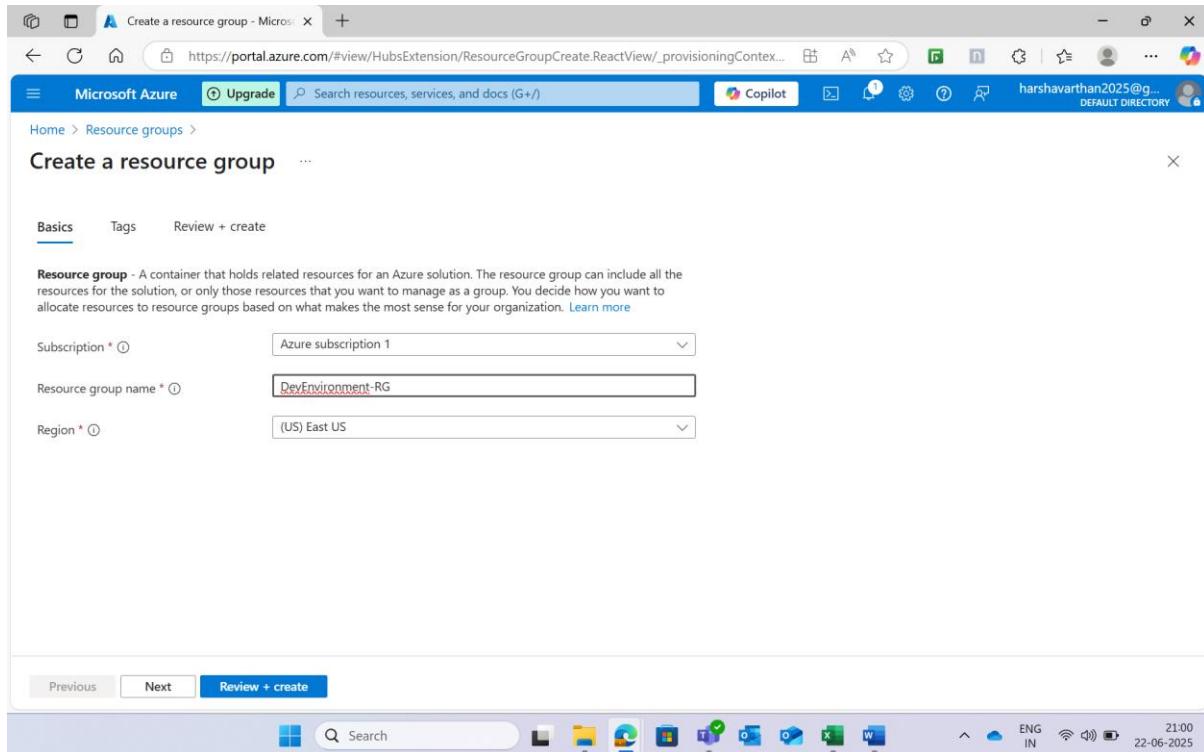
**Objective:** To familiarize the associate with cloud infrastructure, containerization, and automation. By deploying a "Hello World" web app using HTML and CSS on Azure Free Tier, participants will learn to set up resources like VMs, AKS, and ACR, containerize applications using Docker, and automate deployment with Azure DevOps pipelines. The goal is to provide hands-on experience in DevOps practices, CI/CD workflows, and resource optimization within free-tier limits.

## Phase 1 : Infrastructure Setup

### Step1 :

**Created a resource group named “DevEnvironment-RG”.**

- **Name:** DevEnvironment-RG
- **Region:** East US
- **Purpose:** Group all resources related to this project.



## Step2:

**Created a virtual network with the address space 10.0.0.0/16 and created a subnet with the address space 10.0.1.0/24.**

- Name:** Dev-VNet
- Address Space:** 10.0.0.0/16
- Purpose:** Enable communication between resources in the Azure environment.

The screenshot shows the 'Create virtual network' wizard. The 'Project details' step is selected. It shows the subscription is set to 'Azure subscription 1' and the resource group is 'DevEnvironment-RG'. The 'Instance details' step is shown below, where the virtual network name is 'Dev-VNet' and the region is '(US) East US'. At the bottom, there are 'Previous', 'Next', and 'Review + create' buttons.

The screenshot shows the 'Add a subnet' dialog for the 'Dev-VNet' virtual network. Under 'IPv4', the subnet purpose is 'Default', the name is 'Dev-Subnet', and the IPv4 address range is '10.0.0.0/16'. The starting address is '10.0.1.0' and the size is '/24 (256 addresses)'. The 'Add' button is at the bottom. On the left, the 'Subnets' section of the Dev-VNet settings is visible.

### Step 3:

**Created a Storage account named devstorage7777 in the same resource group.**

- **Name:** devstorage7777
- **Location:** East US
- **SKU:** Standard\_LRS
- **Purpose:** Store logs and application-related data.

The screenshot shows the 'Create a storage account' wizard in the Microsoft Azure portal. The 'Project details' step is selected. Under 'Subscription', 'Azure subscription 1' is chosen. Under 'Resource group', 'DevEnvironment-RG' is selected. In the 'Instance details' section, the 'Storage account name' is set to 'devstorage7777'. The 'Region' is '(US) East US'. Under 'Performance', the 'Standard' option is selected. At the bottom, there are 'Previous', 'Next', and 'Review + create' buttons, along with a 'Give feedback' link.

The screenshot shows the 'Overview' page for the storage account 'devstorage7777\_1750606656104'. A green checkmark indicates 'Deployment succeeded'. Deployment details show the name 'devstorage7777\_17506...', start time '6/22/2025, 9:08:53 PM', subscription 'Azure subscription 1', and resource group 'DevEnvironment-RG'. There are sections for 'Deployment details', 'Next steps', 'Give feedback', and 'Tell us about your experience with deployment'. On the right side, there are links for 'Cost Management', 'Microsoft Defender for Cloud', 'Free Microsoft tutorials', 'Work with an expert', and other Azure services.

## Step 4:

**Created a Virtual machine of ubuntu image with the size Standard\_B1s.**

- **Name:** Dev-VM
- **Operating System:** Ubuntu 20.04 LTS
- **Size:** Standard\_B1s (1 vCPU, 1 GB RAM, Free Tier eligible)
- **Admin Username:** azure-dev-user
- **Tags:** Project=DevEnvironment, Owner=azure-dev-user

⚠️ Changing Basic options may reset selections you have made. Review all options prior to creating the virtual machine.

Help me create a low cost VM | Help me create a VM optimized for high availability | Help me choose the right VM size for my workload

**Project details**  
Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription \* ○ Azure subscription 1  
Resource group \* ○ DevEnvironment-RG  
Create new

**Instance details**  
Virtual machine name \* ○ Dev-VM  
Region \* ○ (US) East US  
Availability options ○ Availability zone  
Zone options ○ Self-selected zone

< Previous | Next : Disks > | Review + create | Give feedback

Deployment succeeded  
Deployment 'CreateVm-canonical.ubuntu-24\_04-lts-server-20250622211042' to resource group 'DevEnvironment-RG' was successful.

Deployment name: CreateVm-canonical.ubuntu-24\_04-lts-server-20250622211042 | Start time: 6/22/2025, 9:15:21 PM | Correlation ID: e4eb4cd5-f915-4736-9283-1 | Subscription: Azure subscription 1 | Resource group: DevEnvironment-RG

Your deployment is complete

Deployment details  
Next steps  
Setup auto-shutdown Recommended  
Monitor VM health, performance and network dependencies Recommended  
Run a script inside the virtual machine Recommended

Go to resource | Create another VM

Give feedback | Tell us about your experience with deployment

Cost Management  
Get notified to stay within your budget and prevent unexpected charges on your bill.  
Set up cost alerts >

Microsoft Defender for Cloud  
Secure your apps and infrastructure  
Go to Microsoft Defender for Cloud >

Free Microsoft tutorials  
Start learning today >

Work with an expert  
Azure experts are service provider partners who can help manage your assets on Azure and be your first line of support.

## Step 5:

**Create a network security group in the virtual machine .**

- Name:** Dev-NSG
- Rules:** Allow HTTP traffic on port 80 and SSH traffic on port 22.

The screenshot shows the 'Create network security group' wizard in the Microsoft Azure portal. The 'Basics' tab is selected. Under 'Project details', 'Subscription' is set to 'Azure subscription 1' and 'Resource group' is set to 'DevEnvironment-RG'. Under 'Instance details', 'Name' is 'Dev-NSG' and 'Region' is 'East US'. At the bottom, there are 'Review + create' and 'Next : Tags >' buttons.

## Step 6:

**Add the inbound rules in the Nsg with the ports 80 and 22.**

The screenshot shows the 'Add inbound security rule' dialog for the 'Dev-NSG' network security group. The 'Source' dropdown is set to 'Any'. The 'Source port ranges' dropdown is set to '\*'. The 'Destination' dropdown is set to 'Any'. The 'Service' dropdown is set to 'Custom'. The 'Destination port ranges' dropdown is set to '80'. The 'Protocol' section has 'TCP' selected. At the bottom, there are 'Add' and 'Cancel' buttons.

The screenshot shows the Microsoft Azure portal interface. On the left, there's a sidebar with various options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Resource visualizer, Settings, Inbound security rules (which is selected), Outbound security rules, Network interfaces, Subnets, Properties, Locks, Monitoring, and Automation. The main area shows 'Dev-NSG | Inbound security rules' with a table of rules:

Priority	Name	Port
100	Allow-HTTP	80
65000	AllowVnetInBound	Any
65001	AllowAzureLoadBalanc...	Any
65500	DenyAllInBound	Any

To the right, a modal window titled 'Add inbound security rule' is open. It has sections for Protocol (TCP is selected), Action (Allow is selected), Priority (set to 110), Name (set to 'Allow-SSH'), and Description. At the bottom are 'Add' and 'Cancel' buttons.

## Step 7:

Connected the virtual machine.

```
harshavarthan [ ~ ]$ ssh azure-dev-user@74.235.101.92
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.11.0-1015-azure x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of Sun Jun 22 16:28:27 UTC 2025

System load:  0.1          Processes:           112
Usage of /:   5.6% of 28.02GB  Users logged in:   0
Memory usage: 28%          IPv4 address for eth0: 10.0.1.4
Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

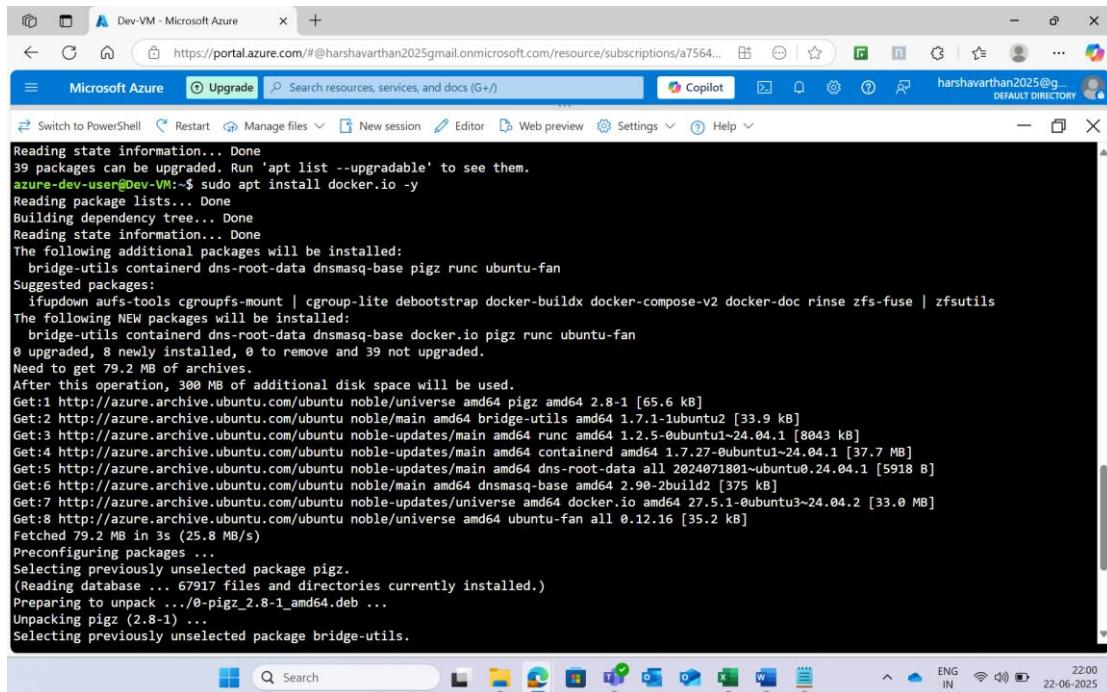
The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Sun Jun 22 16:26:37 2025 from 134.33.229.8
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```

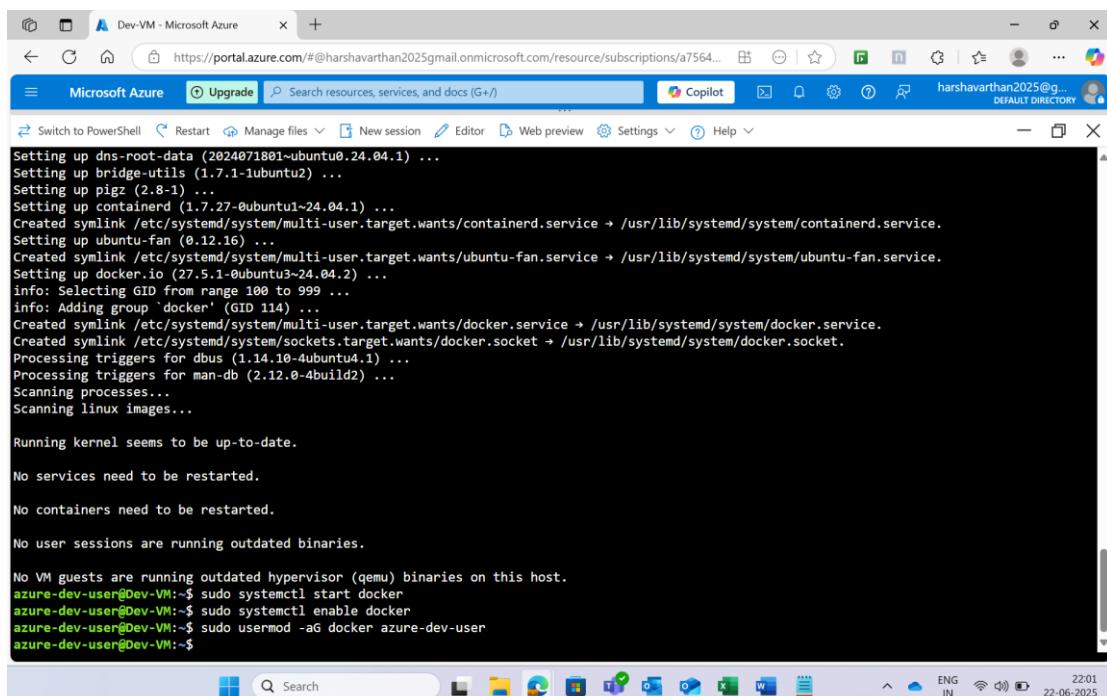
## Step 8:

### Install the docker in the VM.

- sudo systemctl start docker
- sudo systemctl enable docker
- sudo usermod -aG docker azure-dev-user



```
Reading state information... Done
39 packages can be upgraded. Run 'apt list --upgradable' to see them.
azure-dev-user@Dev-VM:~$ sudo apt install docker.io -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base pigz runc ubuntu-fan
Suggested packages:
  ifupdown aufs-tools cgroups-mount | cgroup-lite debootstrap docker-buildx docker-compose-v2 docker-doc rinse zfs-fuse | zfsutils
The following NEW packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base docker.io pigz runc ubuntu-fan
0 upgraded, 8 newly installed, 0 to remove and 39 not upgraded.
Need to get 79.2 MB of archives.
After this operation, 300 MB of additional disk space will be used.
Get:1 http://azure.archive.ubuntu.com/ubuntu noble/universe amd64 pigz amd64 2.8-1 [65.6 kB]
Get:2 http://azure.archive.ubuntu.com/ubuntu/noble/main amd64 bridge-utils amd64 1.7.1-1ubuntu2 [33.9 kB]
Get:3 http://azure.archive.ubuntu.com/ubuntu/noble-updates/main amd64 runc amd64 1.2.5-0ubuntu1~24.04.1 [8043 kB]
Get:4 http://azure.archive.ubuntu.com/ubuntu/noble-updates/main amd64 contained amd64 1.7.27-0ubuntu1~24.04.1 [37.7 MB]
Get:5 http://azure.archive.ubuntu.com/ubuntu/noble-updates/main amd64 dns-root-data all 2024071801~ubuntu0.24.04.1 [5918 B]
Get:6 http://azure.archive.ubuntu.com/ubuntu/noble/main amd64 dnsmasq-base amd64 2.90-2build2 [375 kB]
Get:7 http://azure.archive.ubuntu.com/ubuntu/noble-updates/universe amd64 docker.io amd64 27.5.1-0ubuntu3~24.04.2 [33.0 MB]
Get:8 http://azure.archive.ubuntu.com/ubuntu/noble/universe amd64 ubuntu-fan all 0.12.16 [35.2 kB]
Fetched 79.2 MB in 3s (25.8 MB/s)
Preconfiguring packages ...
Selecting previously unselected package pigz.
(Reading database ... 67917 files and directories currently installed.)
Preparing to unpack .../0-pigz_2.8-1_amd64.deb ...
Unpacking pigz (2.8-1) ...
Selecting previously unselected package bridge-utils.
```



```
Setting up dns-root-data (2024071801~ubuntu0.24.04.1) ...
Setting up bridge-utils (1.7.1-1ubuntu2) ...
Setting up pigz (2.8-1) ...
Setting up contained (1.7.27-0ubuntu1~24.04.1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/containerd.service → /usr/lib/systemd/system/containerd.service.
Setting up ubuntu-fan (0.12.16) ...
Created symlink /etc/systemd/system/multi-user.target.wants/ubuntu-fan.service → /usr/lib/systemd/system/ubuntu-fan.service.
Setting up docker.io (27.5.1-0ubuntu3~24.04.2) ...
info: Selecting GID from range 100 to 999 ...
info: Adding group 'docken' (GID 114) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.
Processing triggers for dbus (1.14.10-4ubuntu4.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

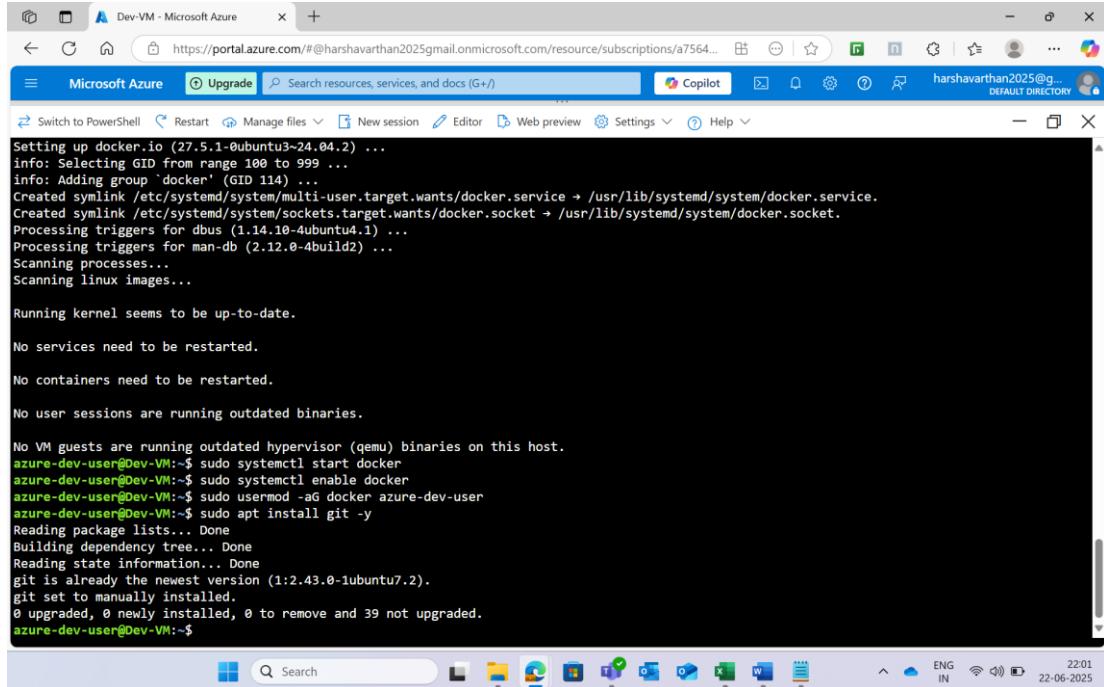
No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
azure-dev-user@Dev-VM:~$ sudo systemctl start docker
azure-dev-user@Dev-VM:~$ sudo systemctl enable docker
azure-dev-user@Dev-VM:~$ sudo usermod -aG docker azure-dev-user
azure-dev-user@Dev-VM:~$
```

## Step 9:

**Install the git in the virtual machine.**

- sudo apt install -y git



```
Setting up docker.io (27.5.1-0ubuntu3~24.04.2) ...
info: Selecting GID from range 100 to 999 ...
info: Adding group 'docker' (GID 114) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.
Processing triggers for dbus (1.14.10-4ubuntu4.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

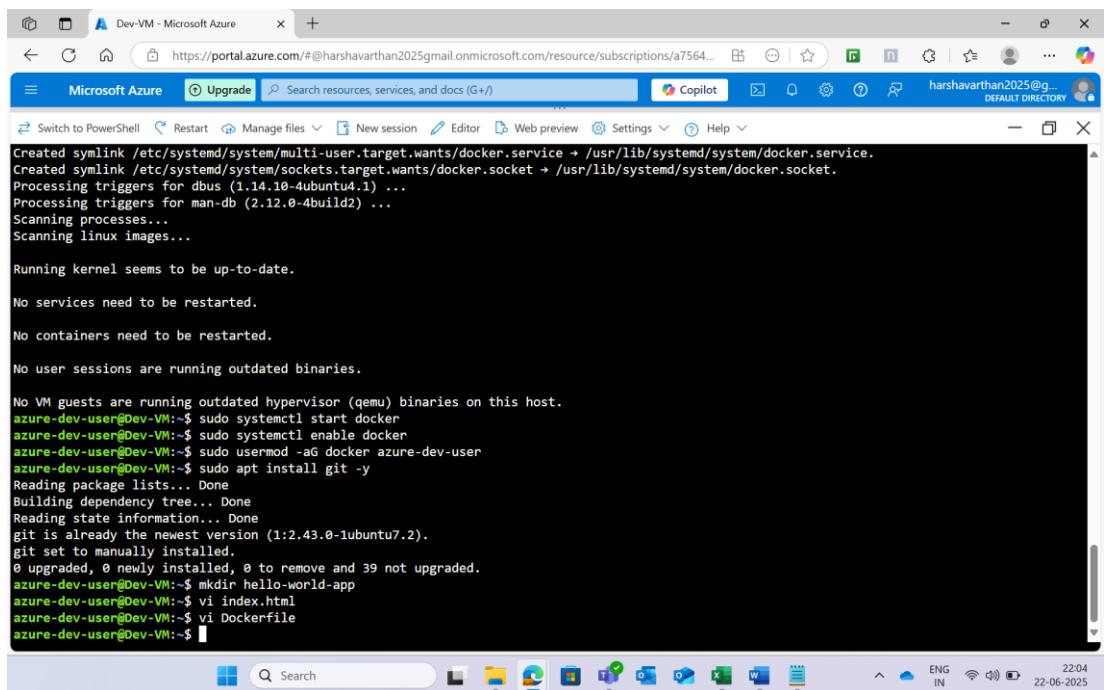
No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
azure-dev-user@Dev-VM:~$ sudo systemctl start docker
azure-dev-user@Dev-VM:~$ sudo systemctl enable docker
azure-dev-user@Dev-VM:~$ sudo usermod -aG docker azure-dev-user
azure-dev-user@Dev-VM:~$ sudo apt install git -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
git is already the newest version (1:2.43.0-1ubuntu7.2).
git set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 39 not upgraded.
azure-dev-user@Dev-VM:~$
```

## Step 10:

**Create a Directory named “Hello-world-app” and make a file named “index.html” and “Dockerfile”.**



```
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.
Processing triggers for dbus (1.14.10-4ubuntu4.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

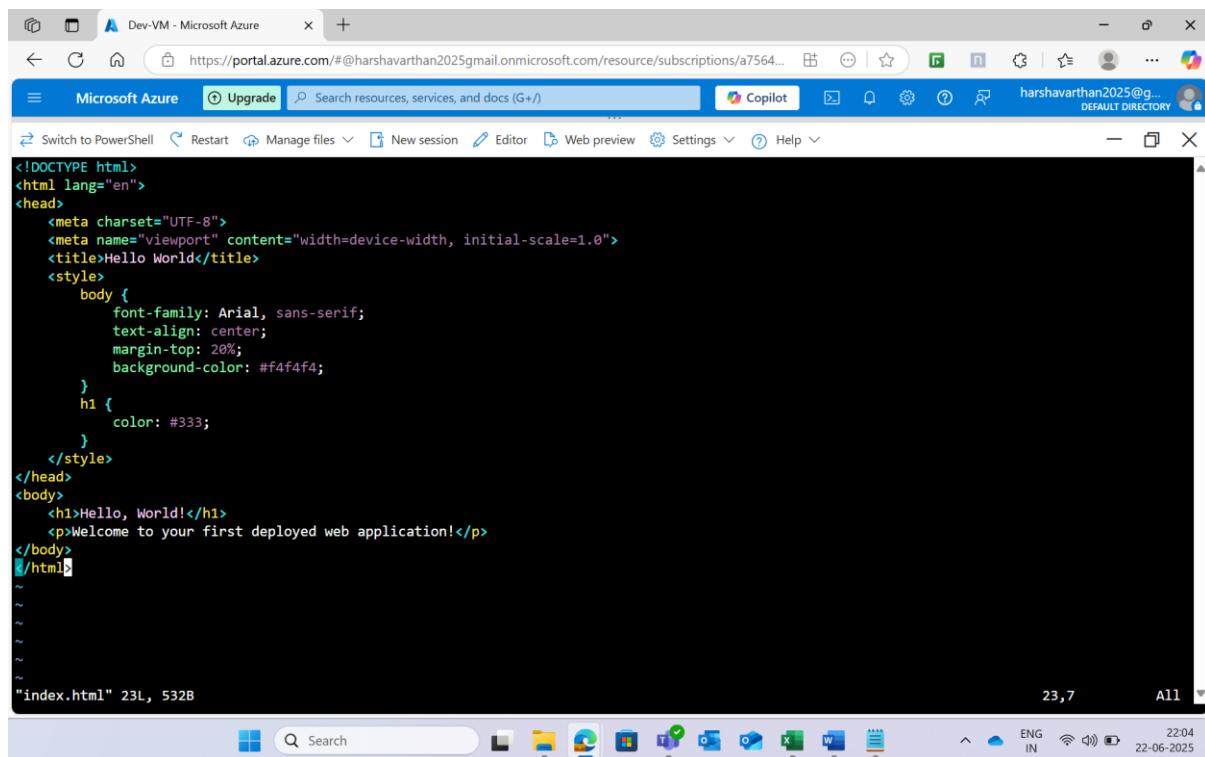
No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
azure-dev-user@Dev-VM:~$ sudo systemctl start docker
azure-dev-user@Dev-VM:~$ sudo systemctl enable docker
azure-dev-user@Dev-VM:~$ sudo usermod -aG docker azure-dev-user
azure-dev-user@Dev-VM:~$ sudo apt install git -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
git is already the newest version (1:2.43.0-1ubuntu7.2).
git set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 39 not upgraded.
azure-dev-user@Dev-VM:~$ mkdir hello-world-app
azure-dev-user@Dev-VM:~$ vi index.html
azure-dev-user@Dev-VM:~$ vi Dockerfile
azure-dev-user@Dev-VM:~$
```

- Insert the html file in this index.html.

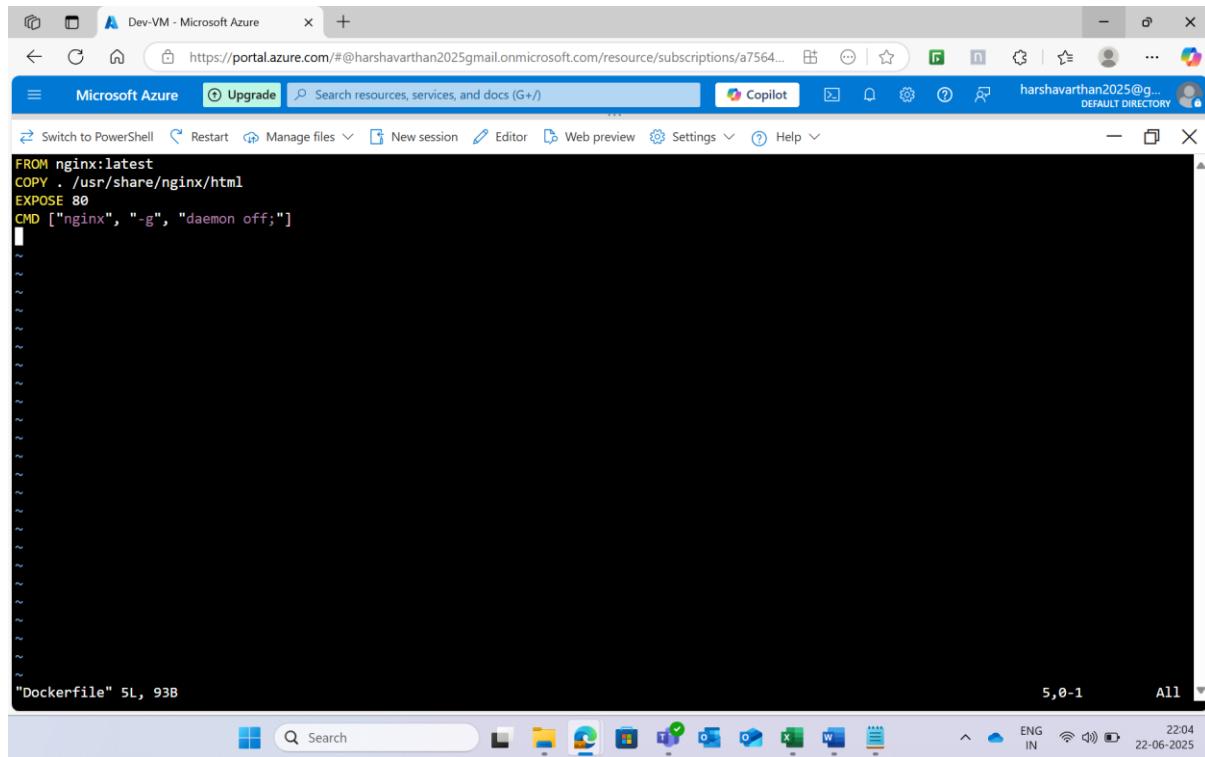


The screenshot shows a Microsoft Azure Dev-VM session titled "Dev-VM - Microsoft Azure". The browser window displays the URL <https://portal.azure.com/#@harshavarthan2025@gmail.onmicrosoft.com/resource/subscriptions/a7564...>. The page content is a code editor showing the "index.html" file. The code is as follows:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Hello World</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            text-align: center;
            margin-top: 20%;
            background-color: #f4f4f4;
        }
        h1 {
            color: #333;
        }
    </style>
</head>
<body>
    <h1>Hello, World!</h1>
    <p>Welcome to your first deployed web application!</p>
</body>
</html>
```

The status bar at the bottom indicates "index.html" has 23L and 532B. The system tray shows the date and time as 22-06-2025 22:04.

- Insert the dockerfile.



The screenshot shows a Microsoft Azure Dev-VM session titled "Dev-VM - Microsoft Azure". The browser window displays the URL <https://portal.azure.com/#@harshavarthan2025@gmail.onmicrosoft.com/resource/subscriptions/a7564...>. The page content is a code editor showing the "Dockerfile" file. The code is as follows:

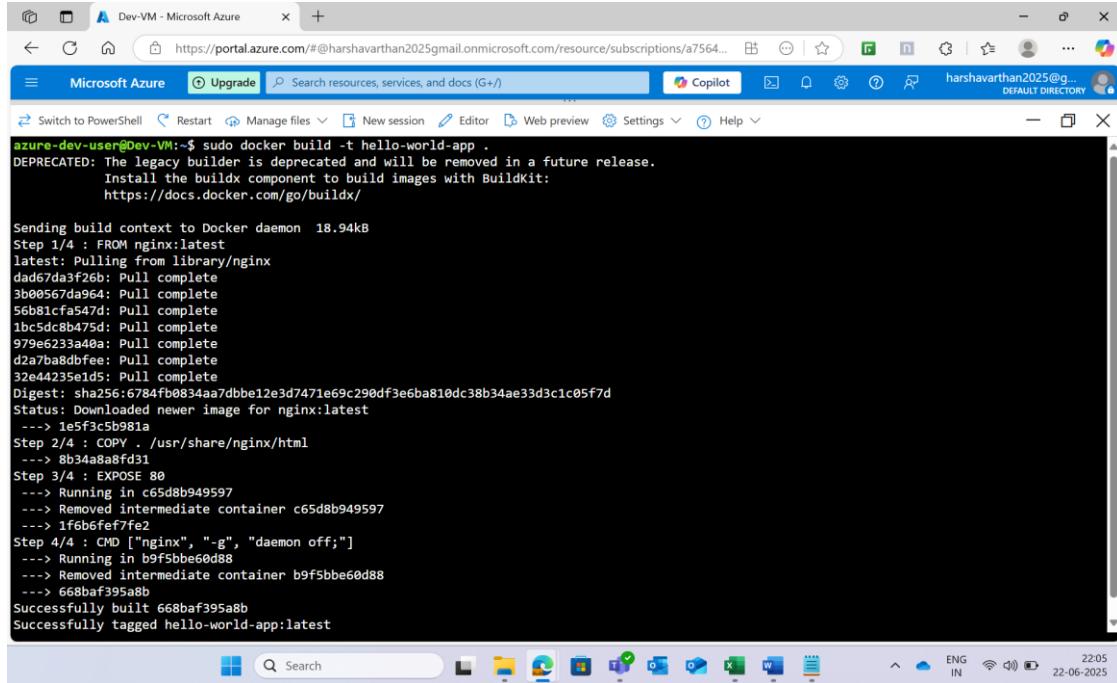
```
FROM nginx:latest
COPY . /usr/share/nginx/html
EXPOSE 80
CMD ["nginx", "-g", "daemon off;"]
```

The status bar at the bottom indicates "Dockerfile" has 5L and 93B. The system tray shows the date and time as 22-06-2025 22:04.

## Step 11:

### Build the image

- docker build -t hello-world-app .



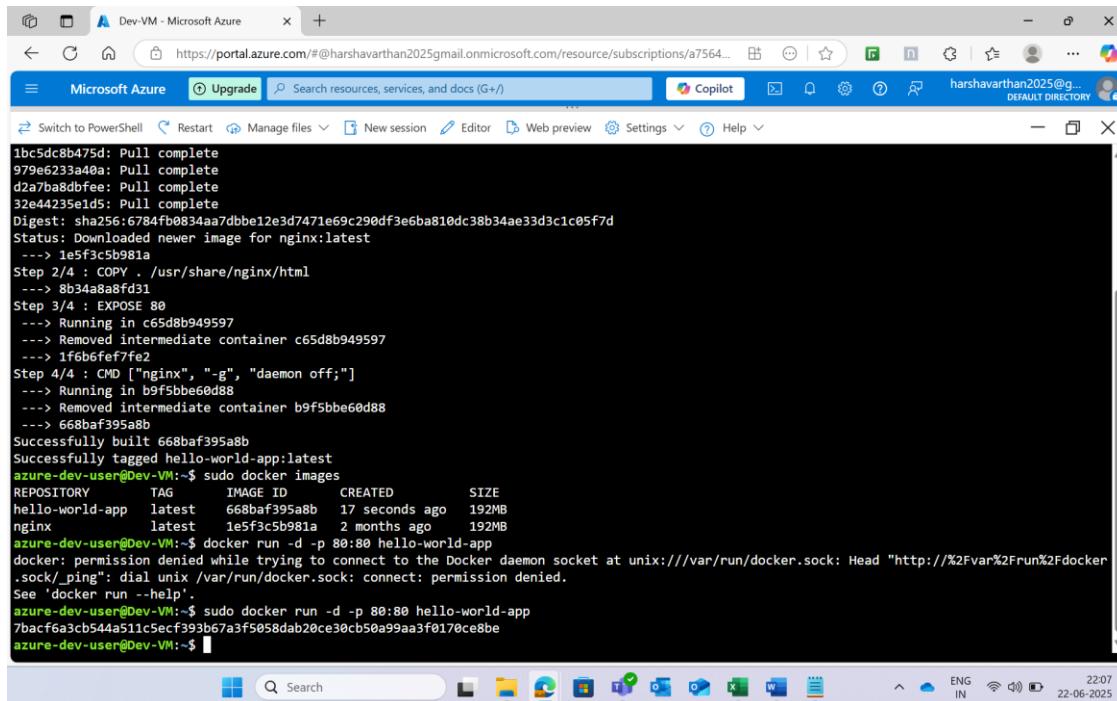
```
azure-dev-user@Dev-VM:~$ sudo docker build -t hello-world-app .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
Install the buildx component to build images with BuildKit:
https://docs.docker.com/go/buildx/

Sending build context to Docker daemon 18.94kB
Step 1/4 : FROM nginx:latest
latest: Pulling from library/nginx
dade67da3f26b: Pull complete
3b00567da964: Pull complete
56b81cfa547d: Pull complete
1bc5dc8b475d: Pull complete
979e6233a40a: Pull complete
d2a7ba8dbfee: Pull complete
32e44235e1d5: Pull complete
Digest: sha256:6784fb0834aa7dbbe12e3d7471e69c290df3e6ba810dc38b34ae33d3c1c05f7d
Status: Downloaded newer image for nginx:latest
--> 1e5fc3c5b981a
Step 2/4 : COPY . /usr/share/nginx/html
--> 8b34a8a8fd31
Step 3/4 : EXPOSE 80
--> Running in c65d8b949597
--> Removed intermediate container c65d8b949597
--> 1f6bbeff7fe2
Step 4/4 : CMD ["nginx", "-g", "daemon off;"]
--> Running in b9f5bbe60d88
--> Removed intermediate container b9f5bbe60d88
--> 668baf395a8b
Successfully built 668baf395a8b
Successfully tagged hello-world-app:latest
```

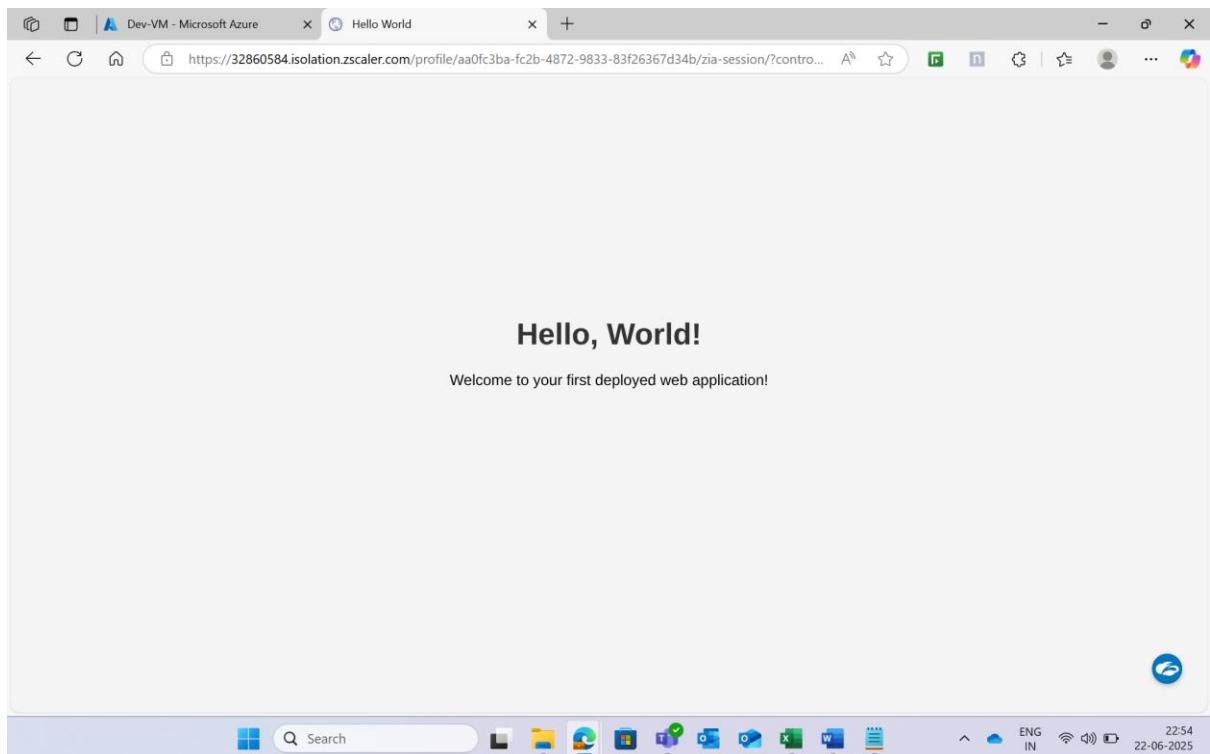
## Step 12:

### Run the image.

- docker run -p 80:80 hello-world-app

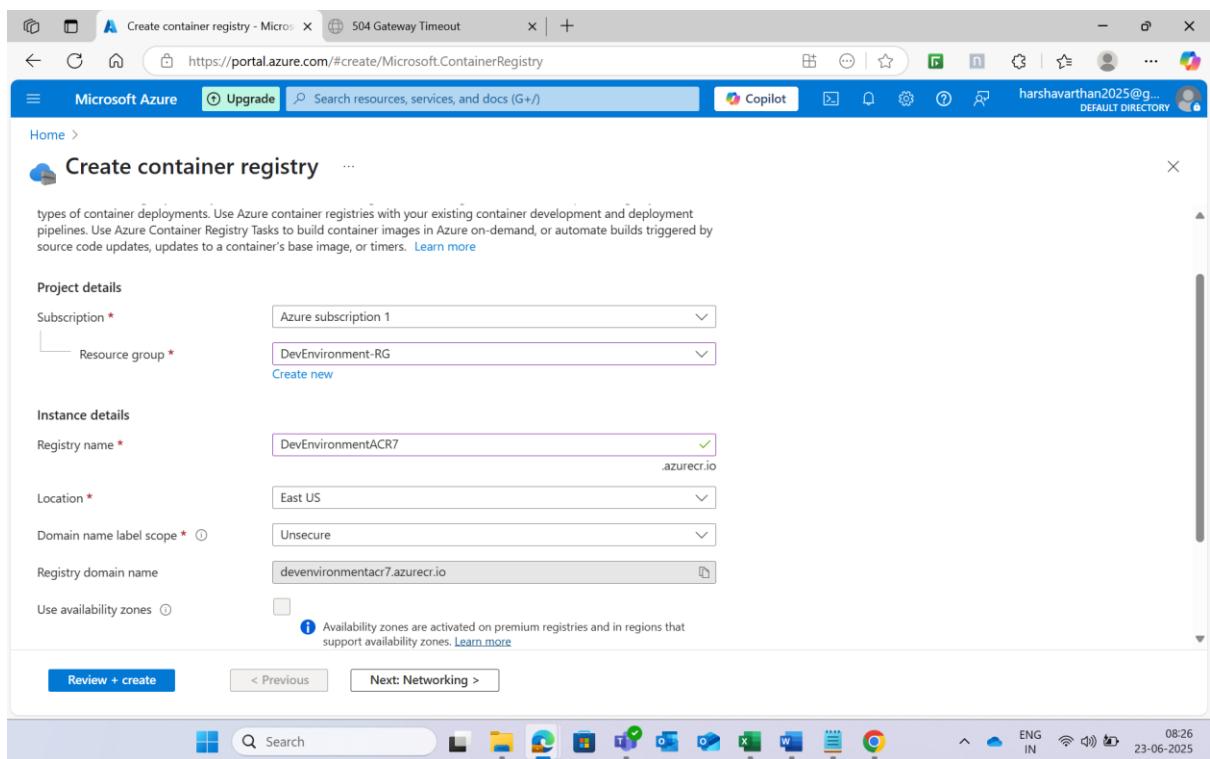


```
1bc5dc8b475d: Pull complete
979e6233a40a: Pull complete
d2a7ba8dbfee: Pull complete
32e44235e1d5: Pull complete
Digest: sha256:6784fb0834aa7dbbe12e3d7471e69c290df3e6ba810dc38b34ae33d3c1c05f7d
Status: Downloaded newer image for nginx:latest
--> 1e5fc3c5b981a
Step 2/4 : COPY . /usr/share/nginx/html
--> 8b34a8a8fd31
Step 3/4 : EXPOSE 80
--> Running in c65d8b949597
--> Removed intermediate container c65d8b949597
--> 1f6bbeff7fe2
Step 4/4 : CMD ["nginx", "-g", "daemon off;"]
--> Running in b9f5bbe60d88
--> Removed intermediate container b9f5bbe60d88
--> 668baf395a8b
Successfully built 668baf395a8b
Successfully tagged hello-world-app:latest
azure-dev-user@Dev-VM:~$ sudo docker images
REPOSITORY      TAG      IMAGE ID      CREATED      SIZE
hello-world-app latest   668baf395a8b  17 seconds ago  192MB
nginx           latest   1e5fc3c5b981a  2 months ago   192MB
azure-dev-user@Dev-VM:~$ docker run -d -p 80:80 hello-world-app
docker: permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Head "http://%2Fvar%2Frun%2Fdocker.sock/_ping": dial unix /var/run/docker.sock: connect: permission denied.
See 'docker run --help'.
azure-dev-user@Dev-VM:~$ sudo docker run -d -p 80:80 hello-world-app
7bacf6a3cb544a511c5ecf393b67a3f5058dab20ce30cb50a99aa3f0170ce8be
azure-dev-user@Dev-VM:~$
```



## Step 13:

Create a Azure Container registry to push the image.



```

azurite-dev-user@Dev-VM:~$ curl -sI https://aka.ms/InstallAzureCLIDeb | sudo bash
Hit:1 http://azure.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://azure.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://azure.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://azure.archive.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://azure.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [161 kB]
Get:6 http://azure.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [376 kB]
Get:7 http://azure.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [212 kB]
Get:8 http://azure.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packages [940 kB]
Get:9 http://azure.archive.ubuntu.com/ubuntu noble-backports/main amd64 Packages [7084 kB]
Get:10 http://azure.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Packages [16.4 kB]
Get:11 http://azure.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Packages [216 kB]
Get:12 http://azure.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Packages [212 kB]
Get:13 http://azure.archive.ubuntu.com/ubuntu noble-security/main amd64 Packages [21.5 kB]
Get:14 http://azure.archive.ubuntu.com/ubuntu noble-security/universe amd64 Packages [52.3 kB]
Get:15 http://azure.archive.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [212 kB]
Get:16 http://azure.archive.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [212 kB]
Fetched 1015 kB in 1s (1263 kB/s)

```

- az acr login --name DevEnvironmentACR
- docker tag hello-world-app DevEnvironmentACR.azurecr.io/hello-world-app:latest
- docker push DevEnvironmentACR.azurecr.io/hello-world-app:latest

```

azurite-dev-user@Dev-VM:~$ curl -sI https://packages.microsoft.com/repos/azure-cli/noble/main/amd64/azure-cli_amd64_2.74.0-1-noble.deb
Fetched 54.0 MB in 1s (95.2 MB/s)
Selecting previously unselected package azure-cli.
Reading database ... 68286 files and directories currently installed.
Preparing to unpack .../azure-cli_2.74.0-1-noble_amd64.deb ...
Unpacking azure-cli (2.74.0-1-noble) ...
Setting up azure-cli (2.74.0-1-noble) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

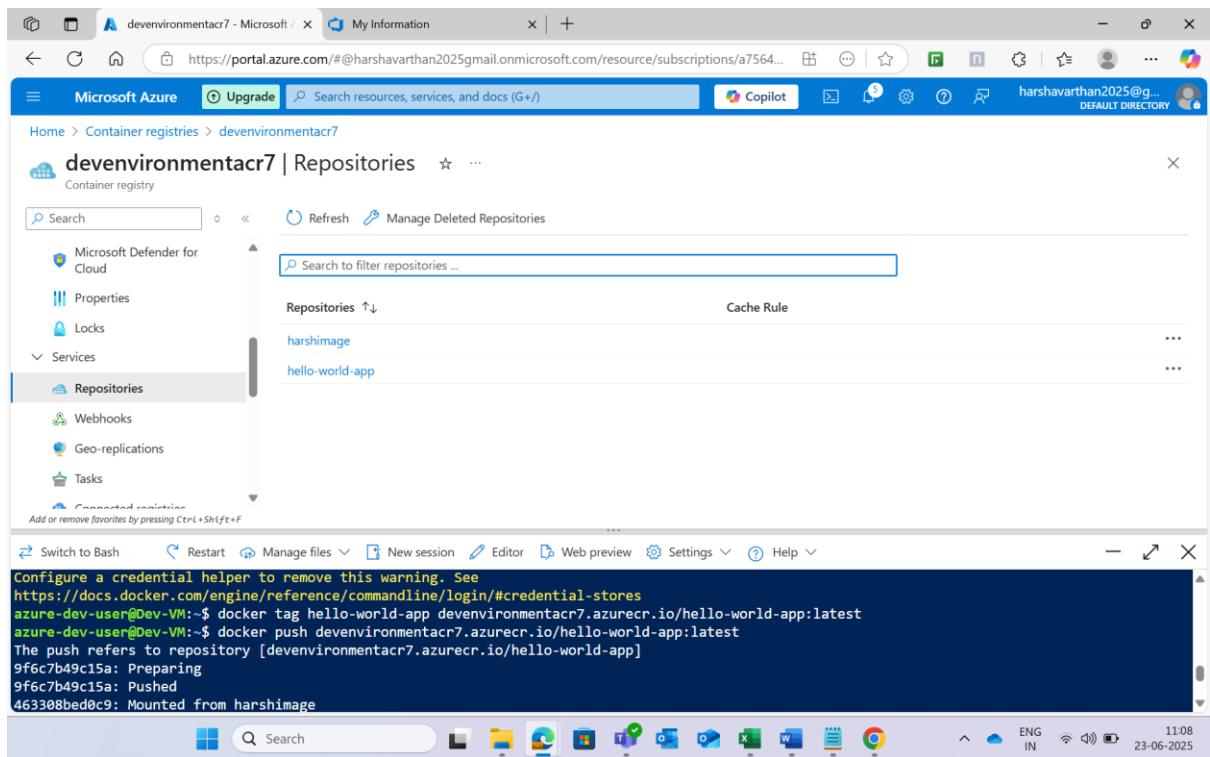
No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.

azurite-dev-user@Dev-VM:~$ az acr login --name DevEnvironmentACR7
Unable to get AAD authorization tokens with message: Please run 'az login' to setup account.
Unable to get admin user credentials with message: Please run 'az login' to setup account.
Username: DevEnvironmentACR7
Password:
Uppercase characters are detected in the registry name. When using its server url in docker commands, to avoid authentication errors, use all lowercase.
Login Succeeded
WARNING! Your password will be stored unencrypted in /home/azurite-dev-user/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credential-stores

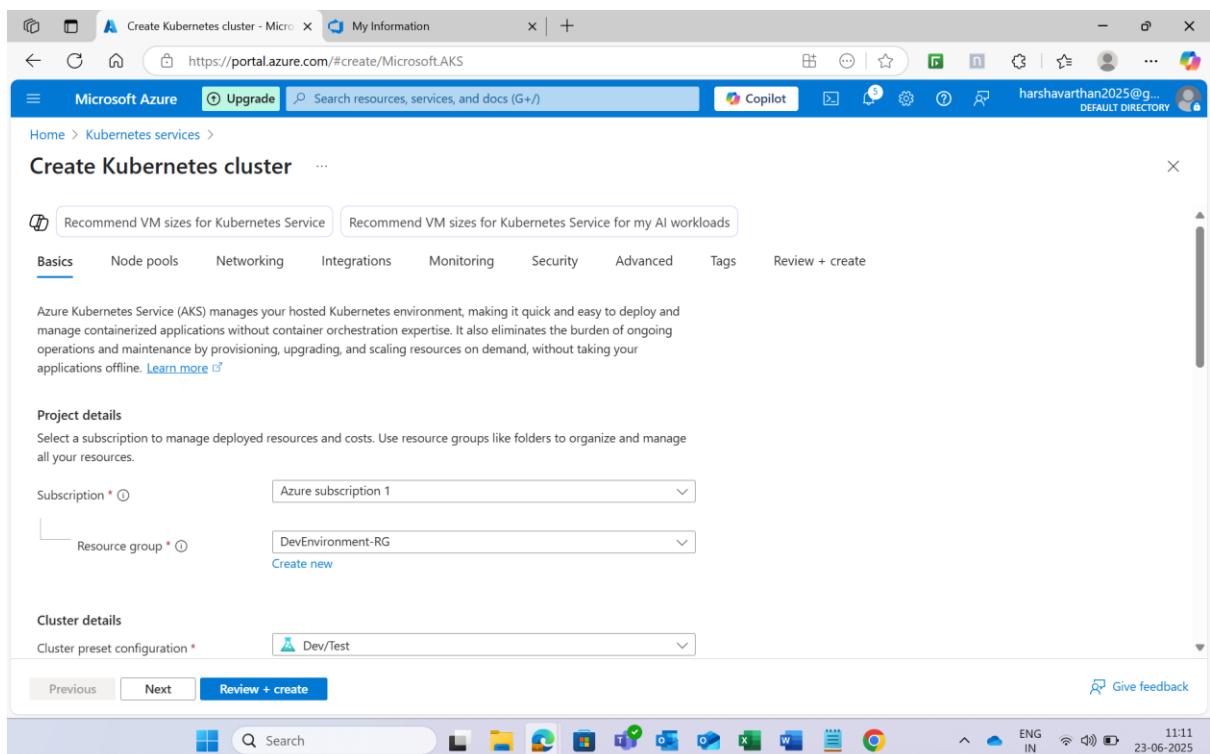
```



## Step 14:

### Create a Azure Kubernetes Cluster to Deploy the images.

- Cluster Name:** Dev-AKS
- Node Count:** 1
- Node Size:** Standard B2s (Free Tier eligible)



The screenshot shows the Microsoft Azure portal with a deployment overview page. The deployment name is 'microsoft.aks-1750657576137'. The status message says 'Your deployment is complete'. Deployment details include a start time of 6/23/2025, 11:16:41 AM, and a Correlation ID: be30fc64-f1d4-4e42-9ed2-67b4eea977a. The resource group is 'DevEnvironment-RG'. There are sections for 'Deployment details' and 'Next steps', with a 'Go to resource' button. On the right side, there are promotional cards for 'Cost Management', 'Microsoft Defender for Cloud', 'Free Microsoft tutorials', and 'Work with an expert'. The taskbar at the bottom shows various application icons.

- Create a Deployment.yaml file and deploy it.
- Create a Service.yaml.

The screenshot shows the Microsoft Azure portal with the 'Kubernetes services' section selected. Under 'Dev-AKS', the 'Workloads' tab is active. The table lists several Kubernetes workloads, including 'coredns', 'konnektivity-agent', 'metrics-server', 'eraser-controller-manager', 'azure-wi-webhook-control', and 'nginx-deployment'. Each entry includes columns for Name, Namespace, Ready status, Age, CPU usage, and Memory usage. The taskbar at the bottom shows various application icons.

Name	Namespace	Ready	Age	CPU	Memory
coredns	kube-system	2/2	16 minutes	Enable metrics	
coredns-autoscaler	kube-system	1/1	16 minutes		
konnektivity-agent	kube-system	2/2	16 minutes		
konnektivity-agent-autoscale	kube-system	1/1	16 minutes		
metrics-server	kube-system	2/2	16 minutes		
eraser-controller-manager	kube-system	1/1	14 minutes		
azure-wi-webhook-control	kube-system	2/2	14 minutes		
nginx-deployment	default	3/3	2 minutes		

Dev-AKS | Services and ingresses

Services

Name	Namespace	Status	Type	Cluster IP	External IP	Ports
kubernetes	default	Ok	ClusterIP	10.0.0.1		443/TCP
kube-dns	kube-system	Ok	ClusterIP	10.0.0.10		53/UDP,53/TCP
metrics-server	kube-system	Ok	ClusterIP	10.0.211.118		443/TCP
azure-wi-webhook-web...	kube-system	Ok	ClusterIP	10.0.93.14		443/TCP
nginx-service	default	Ok	LoadBalancer	10.0.175.180	128.203.140.236	80:31835/TCP

## Step 15:

After Deploying open the ip address of the service.

Hello, World!

Welcome to your first deployed web application!

## PHASE2: AZURE DEVOPS CI/CD PIPELINES

Using Azure Devops we would use build , push an image to the azure container registry and deploy it to the Azure Kubernetes service.

- **CI (Build Pipeline):**

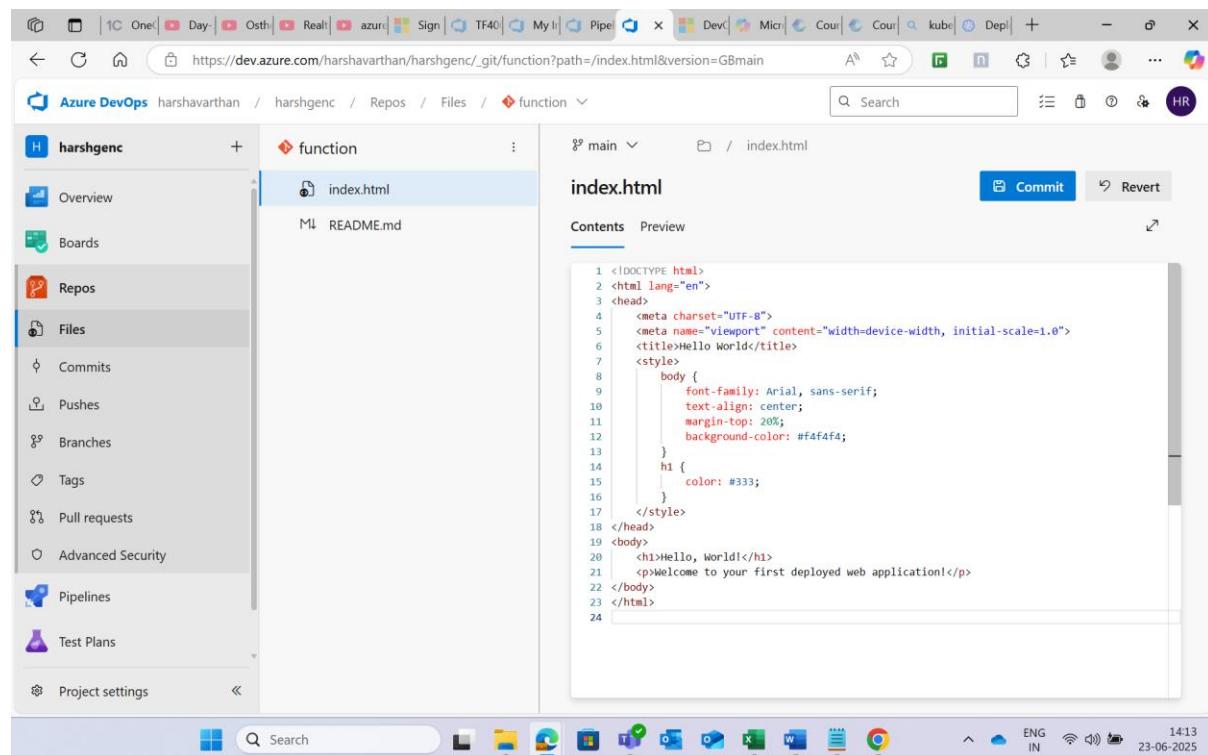
- Build the Docker image from the application code.
- Push the image to Azure Container Registry (ACR).

- **CD (Release Pipeline):**

- Pull the Docker image from ACR.
- Deploy the application to Azure Kubernetes Service (AKS).

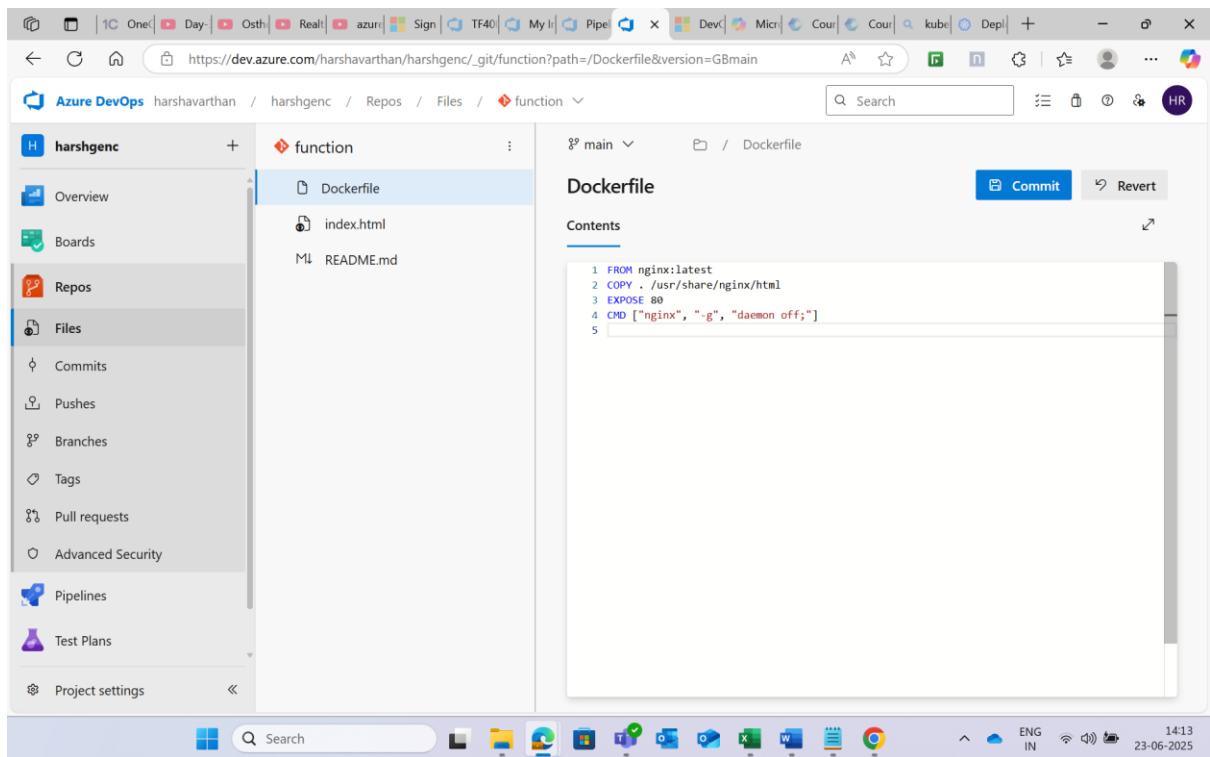
### Step 1:

In azure repos create files named as “index.html” and “Dockerfile”.



The screenshot shows a Microsoft Edge browser window displaying an Azure DevOps repository. The URL is https://dev.azure.com/harshavarthan/harshgenc/\_git/function?path=/index.html&version=GBmain. The left sidebar shows project navigation with 'Files' selected. The main area displays the 'index.html' file content:

```
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title>Hello World</title>
7     <style>
8       body {
9         font-family: Arial, sans-serif;
10        text-align: center;
11        margin-top: 20%;
12        background-color: #f4f4f4;
13      }
14      h1 {
15        color: #333;
16      }
17    </style>
18  </head>
19  <body>
20    <h1>Hello, World!</h1>
21    <p>Welcome to your first deployed web application!</p>
22  </body>
23</html>
```

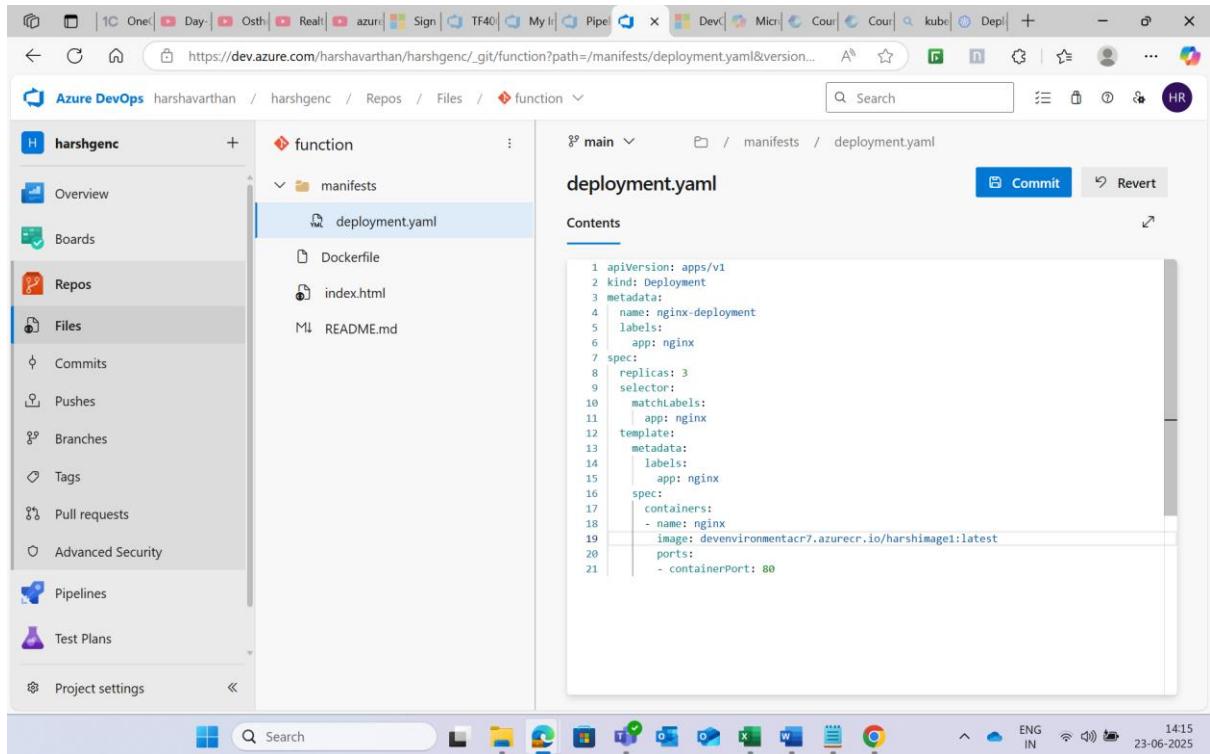


A screenshot of the Azure DevOps interface showing a repository named 'harshgenc'. The 'Files' tab is selected. Inside the 'function' folder, there is a 'Dockerfile' which contains the following code:

```
1 FROM nginx:latest
2 COPY . /usr/share/nginx/html
3 EXPOSE 80
4 CMD ["nginx", "-g", "daemon off;"]
```

## Step 2: Create a folder and add a “deployment.yaml” file

Create another file “Service.yaml”



A screenshot of the Azure DevOps interface showing a repository named 'harshgenc'. The 'Files' tab is selected. Inside the 'function' folder, there is a 'manifests' folder containing a 'deployment.yaml' file. The 'deployment.yaml' file contains the following YAML configuration:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: devenvironmentacr7.azurecr.io/harshimage1:latest
          ports:
            - containerPort: 80
```

The screenshot shows the Azure DevOps interface for a repository named 'harshgenc'. The left sidebar is open, showing options like Overview, Boards, Repos, Files, Pipelines, Test Plans, and Project settings. Under 'Files', 'services.yaml' is selected. The main pane displays the YAML file content:

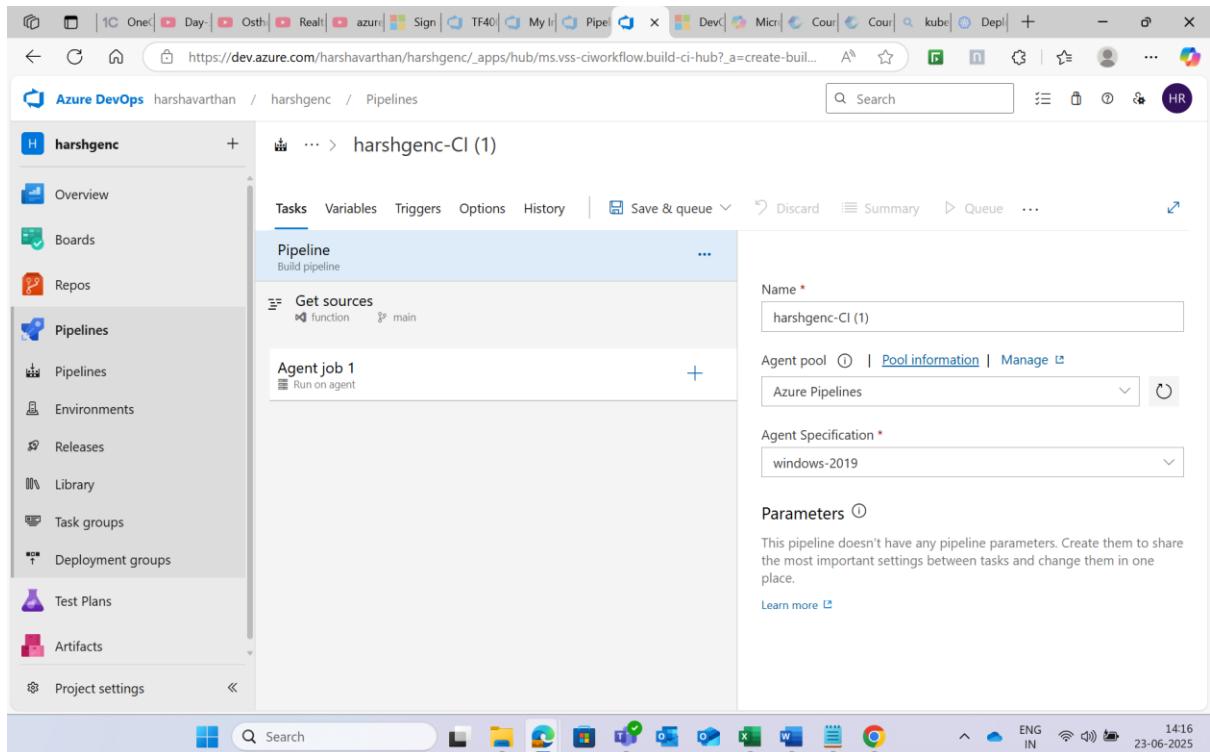
```

1 apiVersion: v1
2 kind: Service
3 metadata:
4   name: nginx-service
5 spec:
6   type: LoadBalancer
7   selector:
8     app: nginx
9   ports:
10    - port: 80
11      targetPort: 80

```

**Step 3:** After the repos setup , I was not able to run the pipeline because of the **parallelism error**. So I filled the form to get access but I didn't get the access.

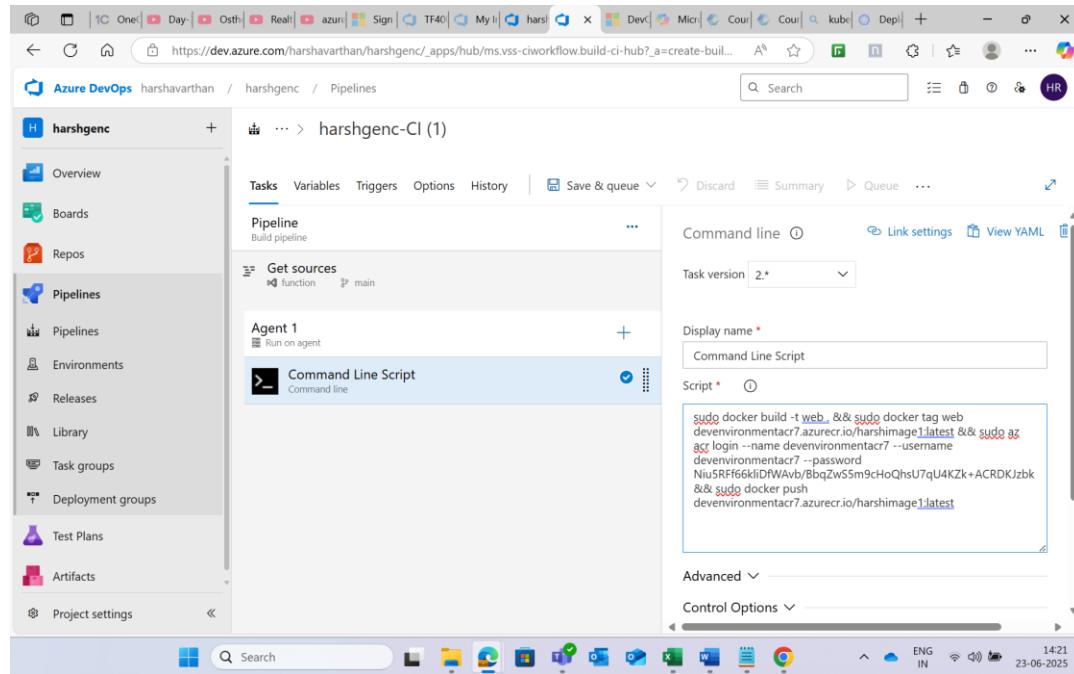
- So I used self hosted agent ( created a virtual machine in azure and used that as agent to run pipeline)



## Step 4:

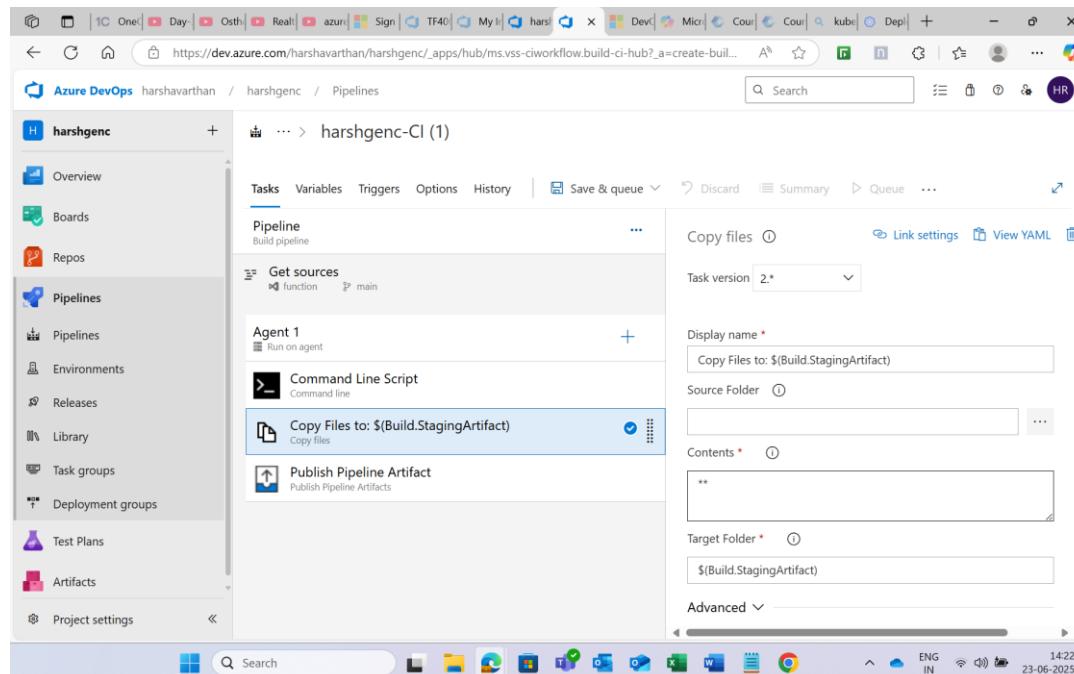
The first task was command line where I used the commands to :

- Build a docker image
- Login to the azure container registry
- Push the image to Azure Container registry



## Step 5:

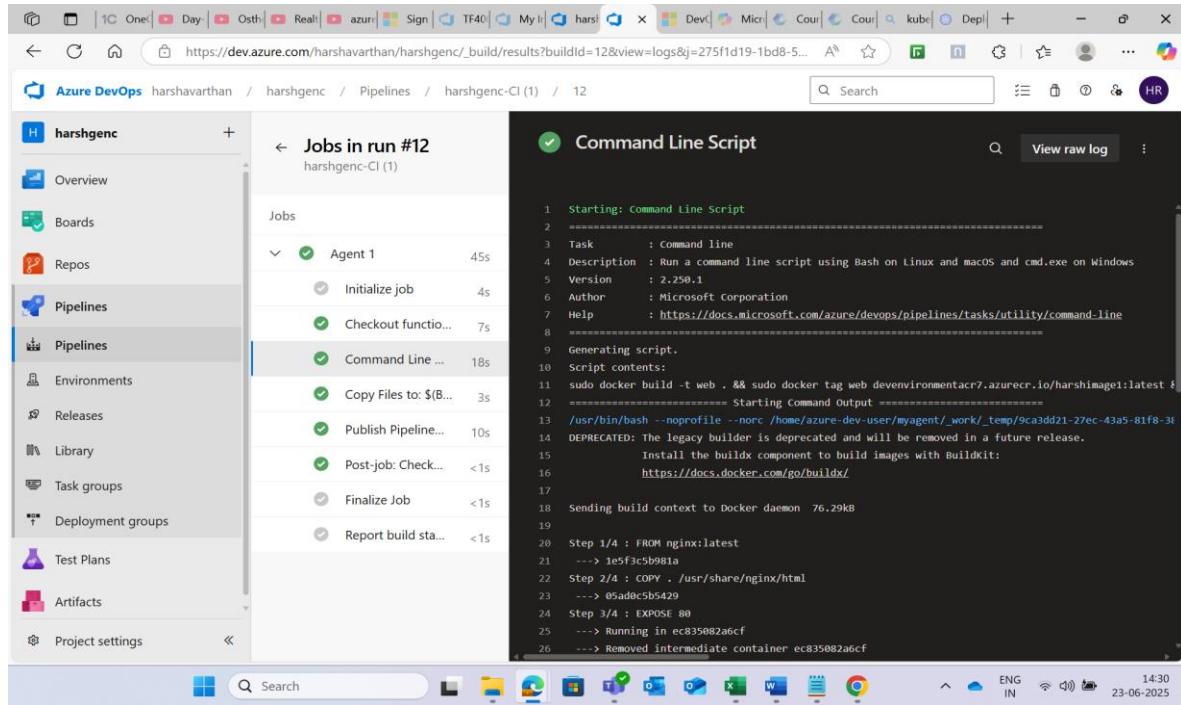
The next task is to add the Copy files task where I mentioned the target folder path to store the build.



**Step 6:** Added publish artifact task.

**Step 7:** Saved and Created run the pipeline.

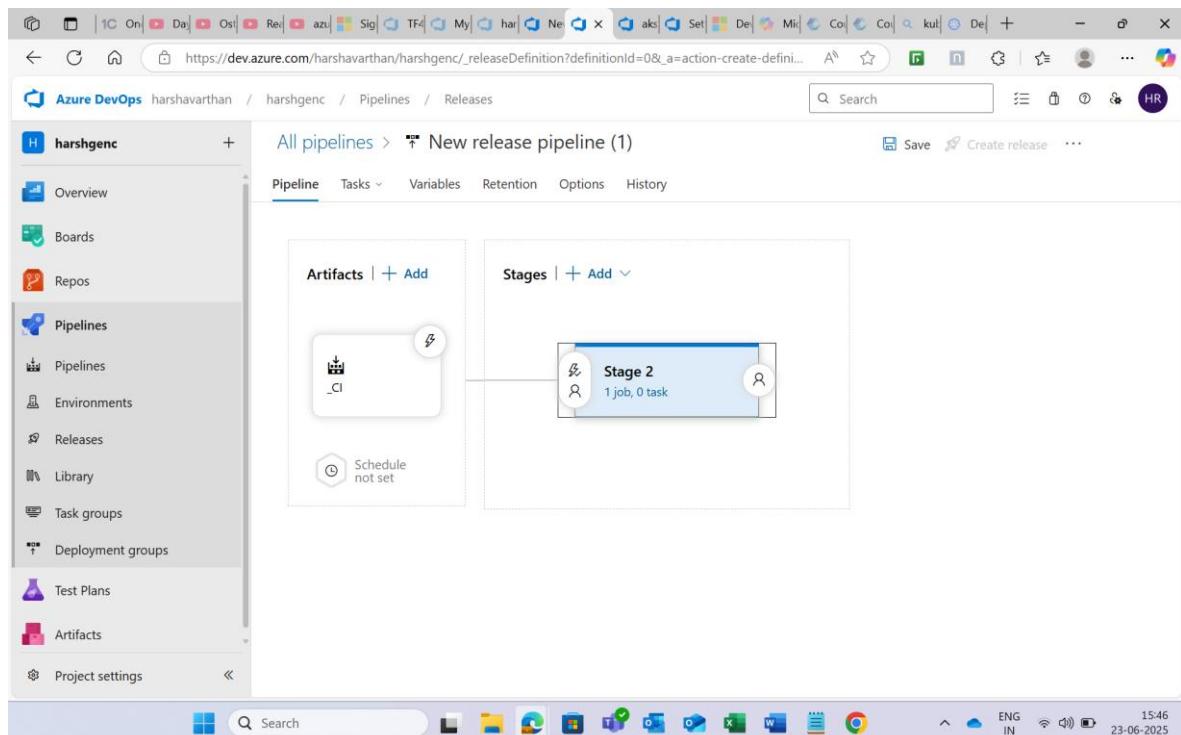
The pipeline executed correctly and image has been pushed to the azure container registry.



A screenshot of a web browser displaying the Azure DevOps interface. The URL is https://dev.azure.com/harshavarthan/harshgenc/\_build/results?buildId=12&view=logs&j=275f1d19-1bd8-5... The page shows the 'Jobs in run #12' for the 'harshgenc-CI (1)' pipeline. The 'Command Line Script' step is highlighted with a green checkmark. The log output for this step is as follows:

```
1 Starting: Command Line Script
2 -----
3 Task      : Command line
4 Description : Run a command line script using Bash on Linux and macOS and cmd.exe on Windows
5 Version   : 2.250.1
6 Author    : Microsoft Corporation
7 Help      : https://docs.microsoft.com/azure/devops/pipelines/tasks/utility/command-line
8 -----
9 Generating script.
10 Script contents:
11 sudo docker build -t web . && sudo docker tag web devenvironmentacr7.azurecr.io/harshimage1:latest &
12 ===== Starting Command Output =====
13 /usr/bin/bash --norc /home/azure-dev-user/.agent/_work/_temp/9ca3dd21-27ec-43a5-81f8-38
14 DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
15     Install the buildx component to build images with Buildkit:
16         https://docs.docker.com/go/buildx/
17
18 Sending build context to docker daemon 76.29kB
19
20 Step 1/4 : FROM nginx:latest
21 ---> 1ef3f3c5981a
22 Step 2/4 : COPY ./ /usr/share/nginx/html
23 ---> 05ad0c5b5429
24 Step 3/4 : EXPOSE 80
25 ---> Running in ec835082a6cf
26 ---> Removed intermediate container ec835082a6cf
```

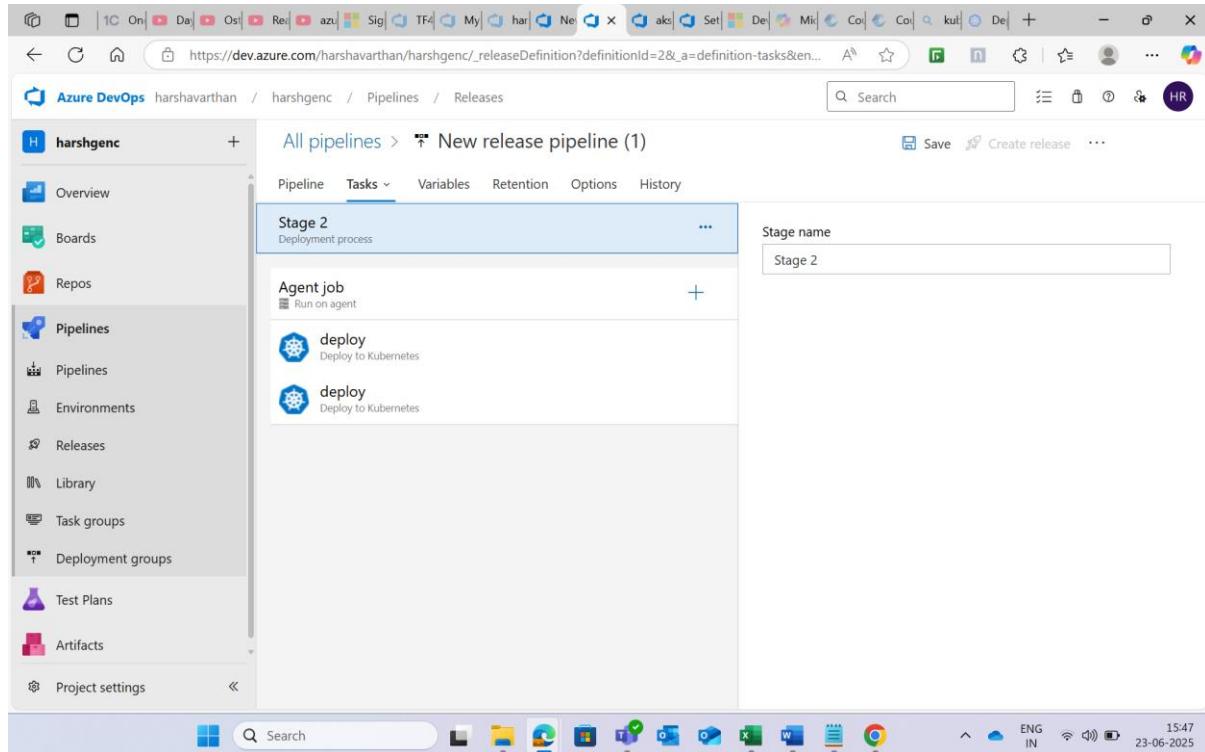
**The next phase is to create the release pipeline to deploy the image in azure Kubernetes.**



## Step 8:

**create a new release pipeline with empty job and add agent pool as “Default”**

- Add task “ Deploy to Kubernetes” two times for separate deployment and service
- Give the deployment.yaml path to the first task.
- Give the service.yaml path to the next task.
- Save and Run the Pipeline.



## Step 9:

**The release pipeline executed successfully and the image has been deployed in the azure Kubernetes services.**

The screenshot shows the Azure DevOps interface for a release pipeline. The left sidebar is titled 'harshgenc' and includes options like Overview, Boards, Repos, Pipelines, Environments, Releases, Library, Task groups, Deployment groups, Test Plans, Artifacts, and Project settings. The main area displays a 'Deployment process' with a status of 'Succeeded'. Below it, under 'Agent job', is another 'Agent job' also marked as 'Succeeded'. The log table shows five tasks: 'Initialize job' (6s), 'Download artifact - \_CI - artifact1' (12s), 'deploy' (6s), 'deploy' (24s), and 'Finalize Job' (<1s). The log table has columns for task name, status, duration, and a 'More' button.

## Step 10:

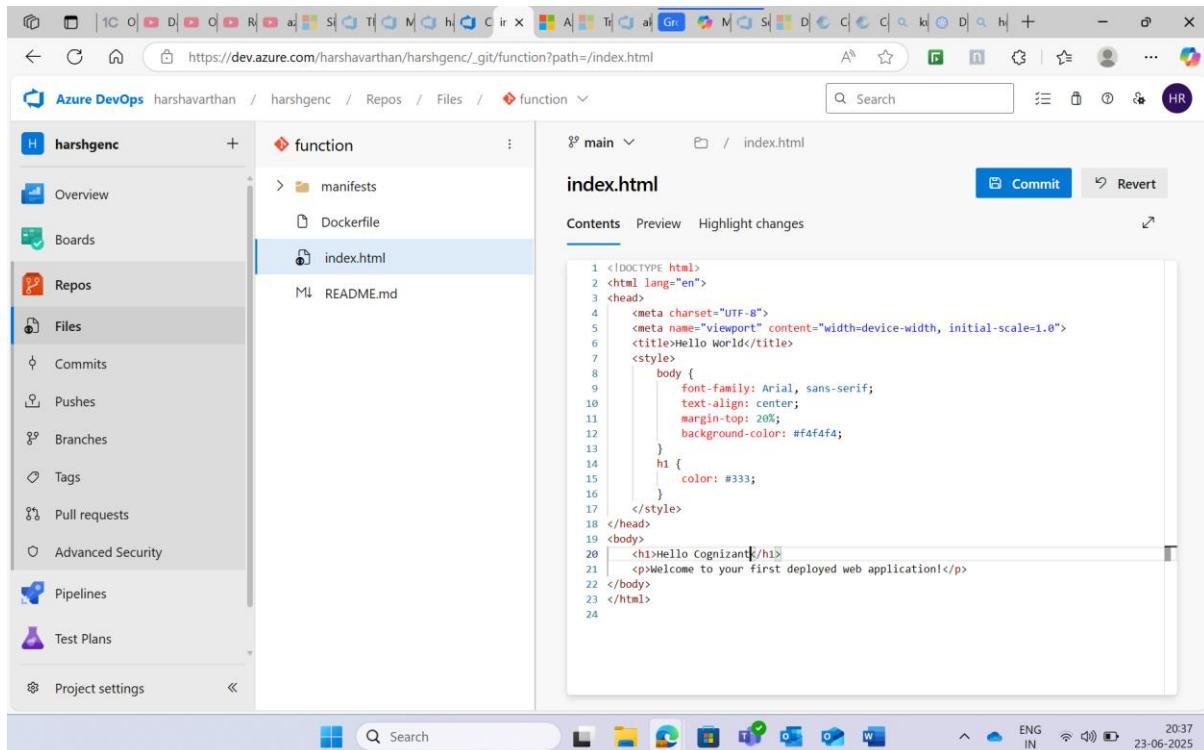
Opened the external ip address in the services to open the deployed image.

The screenshot shows a Microsoft Edge browser window with multiple tabs open. The active tab displays a 'Hello, World!' application at the URL <https://32860584.isolation.zscaler.com/profile/aa0fc3ba-fc2b-4872-9833-83f26367d34b/zia-session/?control>. The page content reads 'Hello, World!' and 'Welcome to your first deployed web application!'. The browser toolbar includes icons for search, refresh, and various Microsoft services like OneDrive, SharePoint, and Teams. The system tray at the bottom shows the date and time as 23-06-2025 and 20:37.

## PHASE 3

The next phase is to make changes in the source file and check whether it triggered automatically and got deployed again.

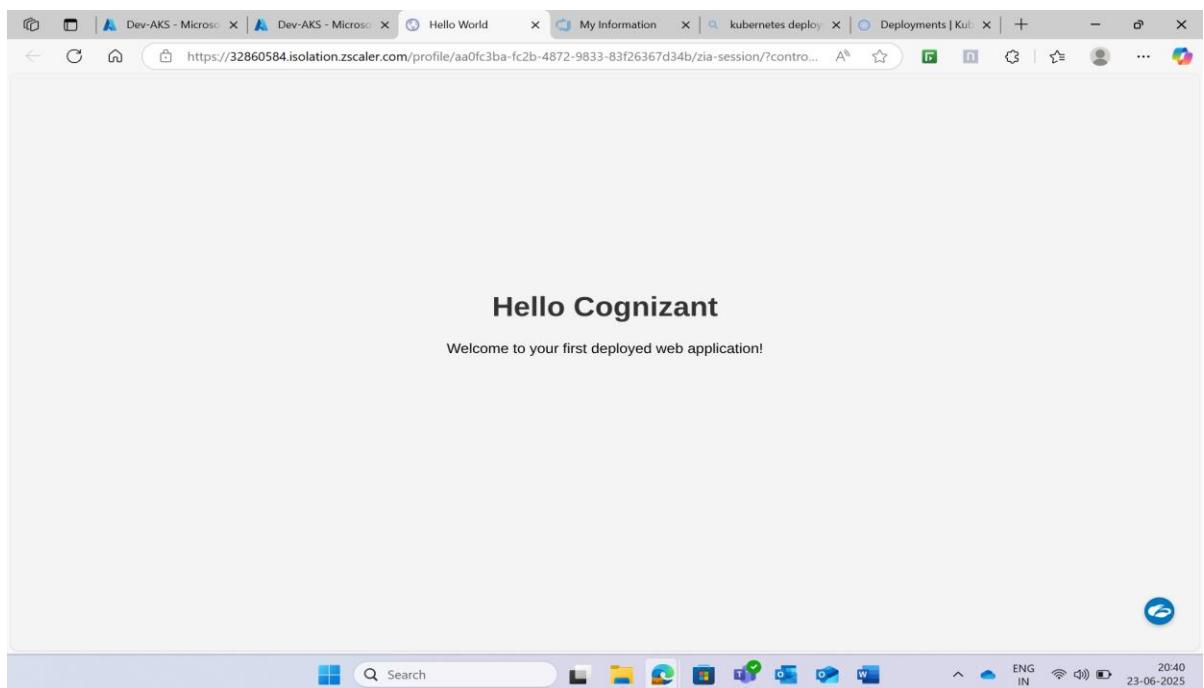
- I made a change in “index.html” from “Hello world” to “hello Cognizant”.
- It automatically triggered the pipeline and release pipeline.



A screenshot of a web browser displaying the Azure DevOps interface. The URL is https://dev.azure.com/harshavarthan/harshgenc/\_git/function?path=/index.html. The left sidebar shows the project structure under 'function': 'manifests', 'Dockerfile', and 'index.html' (which is selected). Below these are 'README.md', 'Commits', 'Pushes', 'Branches', 'Tags', 'Pull requests', 'Advanced Security', 'Pipelines', and 'Test Plans'. The main content area shows the 'index.html' file's code:

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Hello World</title>
7   <style>
8     body {
9       font-family: Arial, sans-serif;
10      text-align: center;
11      margin-top: 20px;
12      background-color: #f4f4f4;
13    }
14    h1 {
15      color: #333;
16    }
17  </style>
18 </head>
19 <body>
20   <h1>Hello Cognizant</h1>
21   <p>Welcome to your first deployed web application!</p>
22 </body>
23 </html>
```

- When I checked the deployed imaged it got changed.



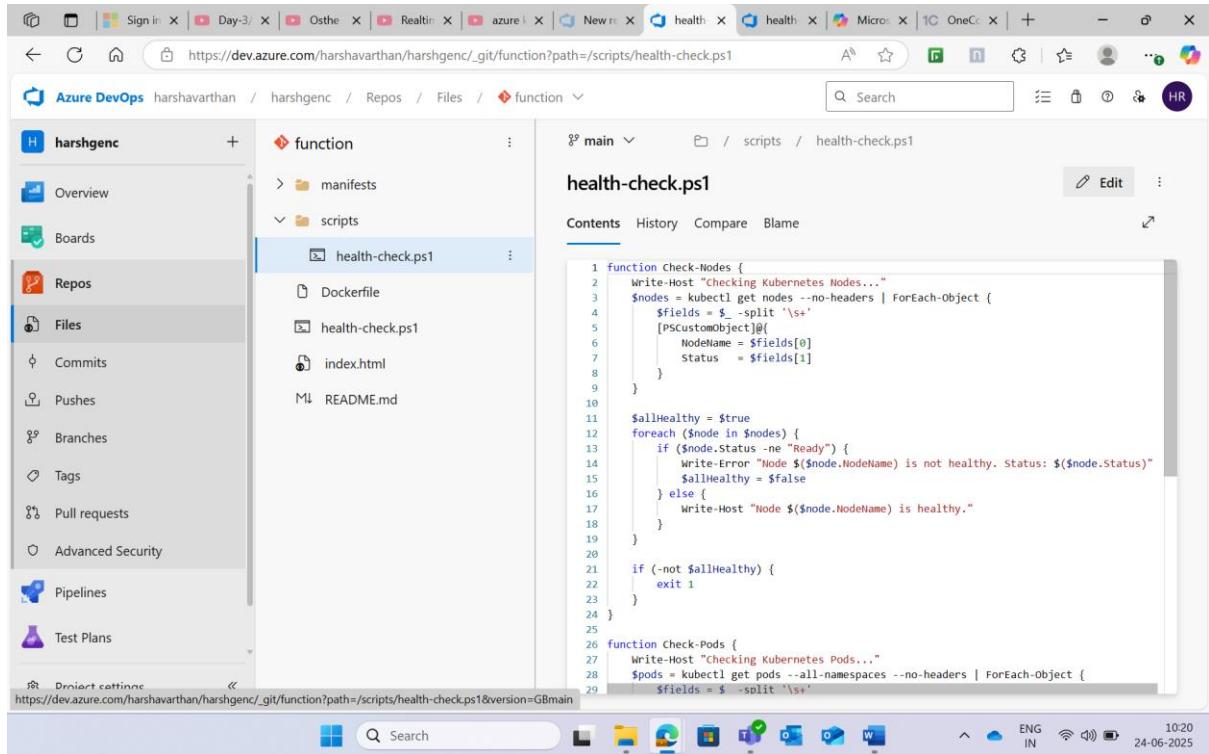
## PHASE 4

This phase has the health check using powershell script to check the podes and nodes in the azure container service.

- If the pods and nodes are all online and running then the script will run successfully.
- Else the script will stop with errors.

### Step 1:

Add the health-check.ps1 script in the azure repos.



The screenshot shows the Azure DevOps interface for a repository named 'harshgenc'. The 'Files' tab is selected. Inside the 'function' folder, there is a 'scripts' folder which contains the 'health-check.ps1' file. The code for 'health-check.ps1' is displayed in the editor:

```
1 Function Check-Nodes {
2     Write-Host "Checking Kubernetes Nodes..."
3     $nodes = kubectl get nodes --no-headers | ForEach-Object {
4         $fields = $_ -split '\s+'
5         [PSCustomObject]@{
6             NodeName = $fields[0]
7             Status   = $fields[1]
8         }
9     }
10
11    $allHealthy = $true
12    foreach ($node in $nodes) {
13        if ($node.Status -ne "Ready") {
14            Write-Error "Node $($node.NodeName) is not healthy. Status: $($node.Status)"
15            $allHealthy = $false
16        } else {
17            Write-Host "Node $($node.NodeName) is healthy."
18        }
19    }
20
21    if (-not $allHealthy) {
22        exit 1
23    }
24 }
25
26 Function Check-Pods {
27     Write-Host "Checking Kubernetes Pods..."
28     $pods = kubectl get pods --all-namespaces --no-headers | ForEach-Object {
29         $fields = $_ -split '\s+'
```

### Step 2:

In the release pipeline add a task as “ powershell Script” and Give the path of the health-check.ps1 file the the script path.

The screenshot shows the Azure DevOps interface for creating a new release pipeline. On the left, the 'Pipelines' section is selected. In the main area, a 'Stage 1' deployment process is shown with an 'Agent job' containing three tasks: 'deploy' (Deploy to Kubernetes), 'deploy' (Deploy to Kubernetes), and 'PowerShell Script'. The 'PowerShell Script' task is currently selected. The task configuration pane on the right shows the following details:

- Task version:** 2.\*
- Display name:** PowerShell Script
- Type:** File Path (selected)
- Script Path:** \$(System.DefaultWorkingDirectory)/\_CI/artifact1/s/scripts/health-check.ps1
- Arguments:** (empty)
- Preference Variables:** (empty)
- Advanced:** (empty)

- The script exited with the error as pods not healthy.
- But the pod was online and running.
- Then I figured out that the script given in the usecase has some mistakes .

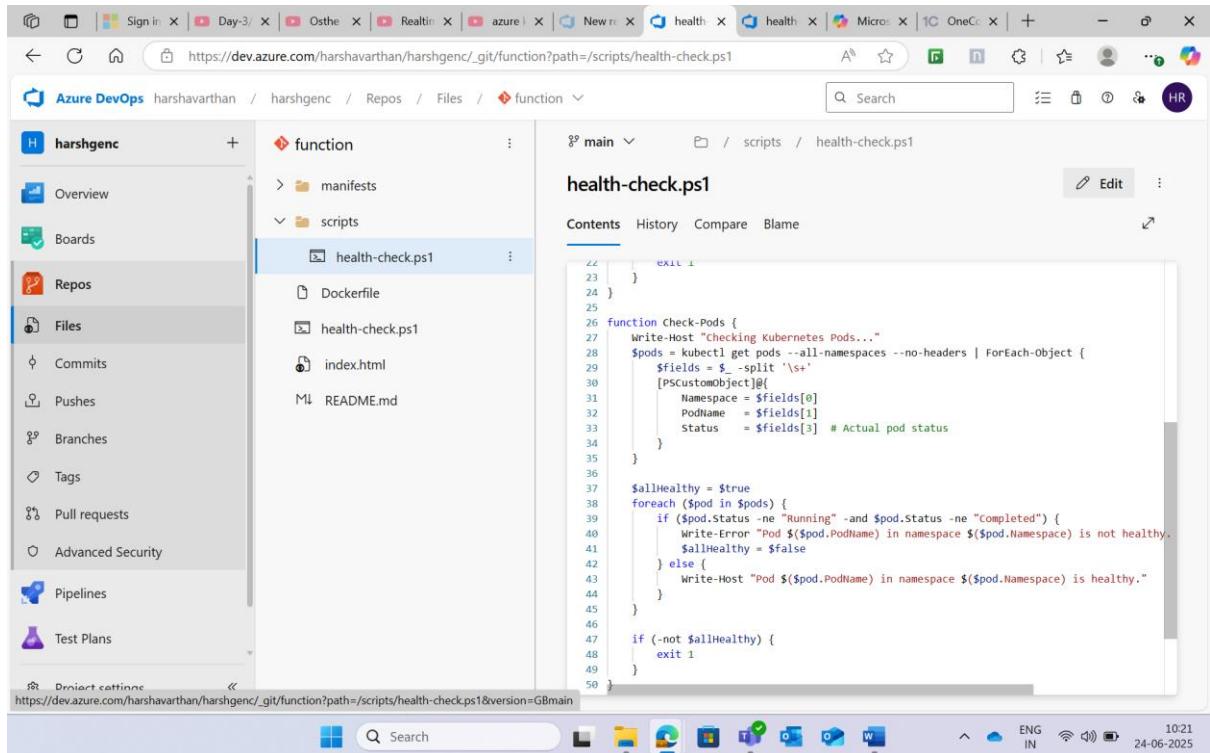
The screenshot shows the logs for the PowerShell task. The command output is as follows:

```

1 2025-06-24T04:10:47.321918Z ##[section]Starting: PowerShell Script
2 2025-06-24T04:10:47.3281614Z =====
3 2025-06-24T04:10:47.3281778Z Task : PowerShell
4 2025-06-24T04:10:47.3281871Z Description : Run a PowerShell script on Linux, macOS, or Windows
5 2025-06-24T04:10:47.3281979Z Version : 2.247.1
6 2025-06-24T04:10:47.3282278Z Author : Microsoft Corporation
7 2025-06-24T04:10:47.3282367Z Help : https://docs.microsoft.com/azure/devops/pipelines/tasks/utility/powershell
8 2025-06-24T04:10:47.3282498Z =====
9 2025-06-24T04:10:50.5624380Z Generating script.
10 2025-06-24T04:10:50.5695202Z ===== Starting Command Output =====
11 2025-06-24T04:10:50.5706291Z [command]/usr/bin/pwsh -NoLogo -NoProfile -NonInteractive -Command . /home/azure-dev-user/myagent/_.
12 2025-06-24T04:10:50.9379085Z Checking Kubernetes Nodes...
13 2025-06-24T04:10:51.6159091Z Node aks-agentpool011-33035357-vmss000002 is healthy.
14 2025-06-24T04:10:51.6179397Z Checking Kubernetes Pods...
15 2025-06-24T04:10:52.1465132Z Check_Pods: homeazure-dev-user\myagent\_work\3a_cArtifact\scripts\health-check.ps1:54
16 2025-06-24T04:10:52.1465908Z Line |
17 2025-06-24T04:10:52.1466233Z | 54 | Check_Pods
18 2025-06-24T04:10:52.1466586Z | ~~~~~
19 2025-06-24T04:10:52.1467316Z | Pod nginx-deployment-6697b9ffc-69rn5 is not healthy. Status: 11
20 2025-06-24T04:10:52.1467482Z
21 2025-06-24T04:10:52.1539104Z ##[error]PowerShell exited with code '1'.
22 2025-06-24T04:10:52.1571768Z ##[section]Finishing: PowerShell Script
23

```

- Then I changed the script with correct script and gave run pipeline.

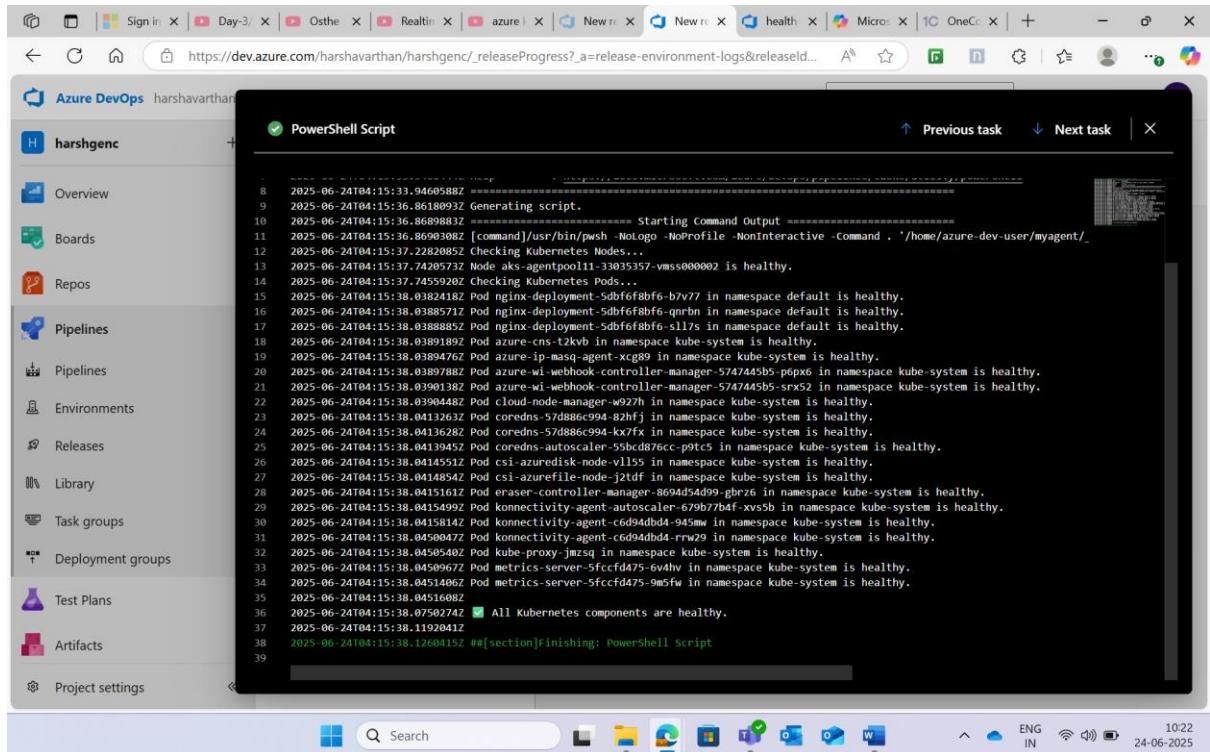


```

24 } exit 1
25
26 function Check_Pods {
27     Write-Host "Checking Kubernetes Pods..."
28     $pods = kubectl get pods -all-namespaces --no-headers | ForEach-Object {
29         $fields = $_ -split ' +'
30         [PSCustomObject]@{
31             Namespace = $fields[0]
32             PodName   = $fields[1]
33             Status    = $fields[3] # Actual pod status
34         }
35     }
36
37 $allHealthy = $true
38 foreach ($pod in $pods) {
39     if ($pod.Status -ne "Running" -and $pod.Status -ne "Completed") {
40         Write-Error "Pod $($pod.PodName) in namespace $($pod.Namespace) is not healthy."
41         $allHealthy = $false
42     } else {
43         Write-Host "Pod $($pod.PodName) in namespace $($pod.Namespace) is healthy."
44     }
45 }
46
47 if (-not $allHealthy) {
48     exit 1
49 }
50 }

```

- It executed correctly and gave the message “All Kubernetes components are healthy”



```

8 2025-06-24T04:15:33.9460588Z =====
9 2025-06-24T04:15:36.8618092Z Generating script.
10 2025-06-24T04:15:36.8689883Z ===== Starting Command Output =====
11 2025-06-24T04:15:36.8690308Z [command]/usr/bin/pwsh -NoLogo -NoProfile -NonInteractive -Command . '/home/azure-dev-user/myagent/_powershell.ps1'
12 2025-06-24T04:15:37.2282085Z Checking Kubernetes Nodes...
13 2025-06-24T04:15:37.7420573Z Node aks-agentpool011-33035357-vms000002 is healthy.
14 2025-06-24T04:15:37.7455926Z Checking Kubernetes Pods...
15 2025-06-24T04:15:38.0388218Z Pod nginx-deployment-5dbf6f8bf6-b7v77 in namespace default is healthy.
16 2025-06-24T04:15:38.0388571Z Pod nginx-deployment-5dbf6f8bf6-qnrnb in namespace default is healthy.
17 2025-06-24T04:15:38.0388885Z Pod azure-cns-t2kvb in namespace kube-system is healthy.
18 2025-06-24T04:15:38.0389189Z Pod azure-ip-masq-agent-xcg89 in namespace kube-system is healthy.
19 2025-06-24T04:15:38.0389476Z Pod azure-wi-webhook-controller-manager-5747445b5-poxpx in namespace kube-system is healthy.
20 2025-06-24T04:15:38.0389782Z Pod azure-wi-webhook-controller-manager-5747445b5-srx52 in namespace kube-system is healthy.
21 2025-06-24T04:15:38.0390138Z Pod cloud-node-manager-w927h in namespace kube-system is healthy.
22 2025-06-24T04:15:38.0390448Z Pod coredns-5d88e094_82hfj in namespace kube-system is healthy.
23 2025-06-24T04:15:38.0413263Z Pod coredns-5d88e094_kx7fx in namespace kube-system is healthy.
24 2025-06-24T04:15:38.0413628Z Pod coredns-autoscaler-55bcd876cc-ptc5 in namespace kube-system is healthy.
25 2025-06-24T04:15:38.0413945Z Pod csi-azuredisk-node-vll15 in namespace kube-system is healthy.
26 2025-06-24T04:15:38.0414551Z Pod csi-azurefile-node-jztd in namespace kube-system is healthy.
27 2025-06-24T04:15:38.0414854Z Pod eraser-controller-manager-8694d54dd9-ghrzr in namespace kube-system is healthy.
28 2025-06-24T04:15:38.0415161Z Pod connectivity-agent-autoscaler-679b774af-xvssb in namespace kube-system is healthy.
29 2025-06-24T04:15:38.0415499Z Pod connectivity-agent-er94d4bd4-945me in namespace kube-system is healthy.
30 2025-06-24T04:15:38.0415814Z Pod connectivity-agent-er94d4bd4-rwx29 in namespace kube-system is healthy.
31 2025-06-24T04:15:38.0450477Z Pod connectivity-agent-er94d4bd4-945me in namespace kube-system is healthy.
32 2025-06-24T04:15:38.0458540Z Pod kube-proxy-jmzsq in namespace kube-system is healthy.
33 2025-06-24T04:15:38.0459067Z Pod metrics-server-5fcffad75-ov4hv in namespace kube-system is healthy.
34 2025-06-24T04:15:38.0451466Z Pod metrics-server-5fcffad75-9m5fw in namespace kube-system is healthy.
35 2025-06-24T04:15:38.0451682Z
36 2025-06-24T04:15:38.0750274Z All Kubernetes components are healthy.
37 2025-06-24T04:15:38.1192041Z
38 2025-06-24T04:15:38.1260015Z #[section]Finishing: PowerShell Script

```

