

CLOUD-AZURE

DEVOPS:

DevOps is a **combination of**

- **tools,**
- **practices, and**
- **cultural philosophy**

that helps in delivering high-quality software **faster** and **more reliably** through **automation** and **collaboration**.

Gate Management in DevOps:

- Acts as a **control checkpoint**
- Used to decide when to allow a process or **deployment to move forward**
- Helps in **improving quality** by catching issues early and reducing technical debt

DevOps is a combination of 3 things:

1 Tools

Used for automation, integration, and tracking:

- **Ansible** – Configuration management
 - **Jenkins** – CI/CD automation
 - **Jira** – Project and issue tracking
-

2 Practices (8)

Key DevOps lifecycle stages:

Plan → Code → Build → Test → Release → Deploy → Operate → Monitor

Other Important Practices:



- **CI/CD** – Continuous Integration & Delivery
- **IaC (Infrastructure as Code)** – Automate infrastructure setup (e.g., using Terraform, ARM templates)
- **Certificate Management** – Secure communication
- **Security Checks** – Integration of tools to catch vulnerabilities early

3 Cultural Philosophy

- **Collaboration** between Development, Operations, QA, and Security
- Encourages **shared responsibility**, **faster feedback**, and **continuous improvement**

Why "Shift Left" Is Important

Moving things to the **left side of the pipeline (earlier)** helps reduce cost and risk.

 Where Bug is Found	 Cost to Fix
After release (prod)	Very High
In testing stage	Moderate
During coding	Low

✓ **Goal:** Catch defects early → Less rework → Lower cost → Faster delivery

DevSecOps:

- DevSecOps = Development + Security + Operations
- It integrates security into every step of the DevOps pipeline
- **SAST** – Static Application Security Testing
 - Scans source code for vulnerabilities
 - **Should be done during the integration phase** (not after deployment)
 - Helps catch security issues early in the build process

DNS & Cloud-Based Product :

1. DNS (Domain Name System) – Cost:

- DNS is typically a **monthly-paid service**.
- You pay as long as you use the domain, usually **low cost (e.g., ₹4/month)**.
- Essential for making your cloud product accessible over the internet (e.g., www.myproduct.com).

Q. Product in Cloud – Why It's a Success?

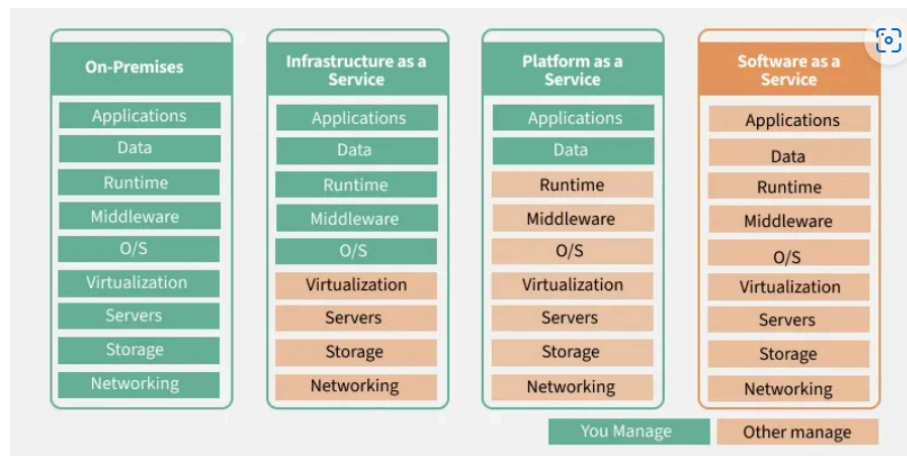
Ans:

- Cloud allows you to **host and scale** your product globally.
- No need to maintain **physical servers**.
- Helps in faster go-to-market and **availability**.

💰 Upfront Cost Comparison

Type	Upfront Cost?	Notes
Traditional Hosting	✗ No upfront cost	But may have limitations & hidden costs
Cloud	✓ Yes, minimal upfront	Small initial cost, but scalable and flexible







CLOUD SERVICE MODELS:



IaaS, PaaS, SaaS

1. **IaaS:** Cloud model where you **rent complete infrastructure** (like virtual machines, storage, networks) from a provider (e.g., Azure), and you manage everything on top of it.
 - eg: Working from home with **no laptop and no salary** = You have **no infrastructure** to work.
 - If **UST provides you a laptop, network, and setup**, now you have complete infra = That's IaaS.

Advantages of IaaS:

-  **Complete control** of resources
-  **Access to any type of cloud resource** (VM, Storage, Networking, etc.)
-  **Pay-as-you-go** model – pay only for what you use
-  **Scalability** – Scale from 1 VM to 100+ (ask for quota increase if needed)
-  **Elastic IP** – Assign fixed public IP to a resource (quota-restricted)
-  **Self-Service Provisioning** – Create resources yourself without contacting support

Tools for Self-Service Provisioning:

Tool/Method	Description
Azure ARM Template	Earlier method for provisioning (JSON-based)
Azure Bicep	Latest from Microsoft; faster, secure, standard-based
Terraform	3rd-party tool by HashiCorp, works with multiple clouds
Azure Portal	UI-based, easy and self-explanatory





Self service provisioning:

- create almost any kind of services(instance,cluster,acr,roles,acs,azure files,storages,devops pipeline,network,subnet)
- portal - self explanatory(dont need trainer)
- dis: time,consistency






Q. why so many options like Azure ARM template,Azure Bicep,Terraform?


Ans:

1. ARM Templates (Azure Resource Manager)






-  Written in JSON
-  Oldest Azure-native tool
-  Works well, but harder to read and maintain
-  Forms the base for newer tools like Bicep

2. Bicep





-  New and improved version of ARM Templates
-  More secure, easier to write, and modular
-  Designed and maintained by Microsoft
-  Supports all new Azure features
-  If you're fully using Azure – Bicep is the best choice

 Think of Bicep as a simpler, modern language that internally converts to ARM.

3. Terraform

-  Created by HashiCorp (a third-party company)
- **Hashicorp** - Other company, not from azure and microsoft
- - terraform ,vagrant,vault
- deploy an app -> artifact is docker images -> deploy docker images
- Use terraform and automatically create **provision infrastructure**.
- U dont have to switch to diff system , be in azure platform
- Terraform can **integrate** with any cloud.
- not tightly bound to azure .
-  Not just for Azure – supports multi-cloud (AWS, GCP, Azure, etc.)
-  Best if you are working across different cloud providers
-  More flexible – can connect to other tools like Vault, Consul
-  Great for companies who don't want to be locked into one cloud

4. Azure Portal

-  Graphical interface (UI) – no coding
-  Easy for beginners or one-time tasks
-  Not great for repeatable or large-scale automation
-  May lack consistency and take more time

Tool	Best For	Pros	Cons
ARM	Legacy support	Native, structured	Hard to manage manually
Bicep	Azure-only, modern setups	Simple, secure, all new Azure features	Azure-specific
Terraform	Multi-cloud environments	Cloud-agnostic, powerful integrations	Third-party, not Azure-native
Portal	Quick, manual setups	Visual, user-friendly	Not scalable or reusable

Q. Virtualization?

- ★ Multiple VMs run on a single physical machine.
- ★ Each vm have its own OS,CPU,RAM,Disk,Network.
- ★ Each VM is **isolated** – one tenant cannot access another's data
- ★ Helps in **cost reduction** and efficient **resource utilization**

Q. Multi-Tenancy?

- ★ One physical server is shared by many customers, but securely isolated.
- ★ 1 vm can be used by multiple users
- ★ u should not be able to see data and any kind with other tenants,by the same tym u r sharing all the resources
- ★ cost reduces
- ★ Can lead to issues in industry-specific high-security needs
- ★ Some company needs higher security, Solution-> companies coming up together -> still fails.



Q. How does IaaS work?

- ★ IaaS (Infrastructure as a Service) provides virtual infrastructure (like VMs, storage, network) but you manage everything above it — like the OS, middleware, and applications.

What You Need to Maintain in IaaS:

Layer	Responsibility	Example/Explanation
 OS	✓ Your Responsibility	You install, update, patch Linux/Windows
 Middleware	✓ Your Responsibility	e.g., Apache Tomcat – you install and upgrade
 Runtime	✓ Your Responsibility	e.g., Java, .NET – setup and manage
 Application & Data	✓ Your Responsibility	You write code, deploy app, and handle data


- ★ **Data** - data inside services (like VMs) is not automatically persisted.
- ★ Services are mostly **stateless** (won't persist data). Why?
- ★ To **enable scalability** — easily add/remove instances without losing data.
- ★ So you must **externalize the data**
- ★ external database store in Redis, Memcached (no SQL) - key-value pair DB
- ★ if you're storing by yourself -> you have to take care of that

External Storage	Purpose
 Azure Blob, S3, Disks	Persistent storage
 Redis, Memcached	Fast, in-memory storage (key-value)
 External Databases	MySQL, PostgreSQL, Cosmos DB, etc.

Q. Why Services Don't Store Data by Themselves?

- To **scale easily** and avoid **data loss**.
- Example: During an Amazon flash sale, 10,000+ people may add laptops to their cart, and backend servers need to scale instantly. If services store data inside themselves, new servers start fresh and old data is lost, but if they're stateless and use external databases, all instances share the same data, enabling seamless scaling and consistent user experience.

Q. Benefits of IaaS?

-  **Cost Reduction**
No need to invest in physical infrastructure. Pay-as-you-go model helps cut upfront costs.

- ★ ⚡ **Faster Deployment (Automated)**
Automation tools enable faster provisioning, testing, and releasing — reduces time-to-market.
- ★ 🔄 **High Availability & Reliability**
Use of AMIs (Amazon Machine Images) ensures reliable replication and fast recovery in case of failures.
- ★ 🌐 **Global Reach**
Easily deploy applications in any region around the world to serve users with low latency.
- ★ 🗝️ **Data Security and Compliance**
Data can be encrypted and moved to specific regions to comply with local regulations and standards.

Q. Challenges of IaaS?

- **Security & Vulnerabilities**
Cloud platforms may face vulnerabilities — providers send alerts, but you must apply patches yourself (especially in IaaS).
- **Compliance Requirements**
Industries like banking and healthcare (e.g., HIPAA in the US) require strict data confidentiality.
 - In SaaS, data is with the provider.
 - In IaaS, you are responsible for compliance and data protection.
 - compliance -> banking, medical
 - in US and Europe - you need to give your critical info to docs, so data is confidential..can't be shared
 - HIPAA complaints - HIPAA guidelines - followed by us
 - if using SaaS, they have data
 - IaaS - your responsibility
- **Integration with Existing Systems**
Old systems (legacy apps) may not work smoothly with new cloud or automation tools — so extra effort or changes may be needed to make them work together.
- **Dependency on Cloud Provider**
Service interruptions, pricing changes, or policy shifts by the provider can impact your business.

Q. Categories of cloud?

- ★ **Public** - Shared among multiple users
- ★ **Private** - Used by single organization
- ★ **Hybrid** - Mix of public and private clouds

Q. How can you trust the VM provided is secure and no one has ownership?

- **Limited Access:** Less than 2% of employees, even from outside the company, have access to the data centers. Only authorized people can access the VMs.
- **Data Protection:** When data is stored, it's split and stored securely. No one outside the organization has the private encryption keys (PEM files).
- **Secure despite Attacks:** While attacks can happen (like software vulnerabilities), the **cloud ensures data security**. Most attacks are common software-related ones, not specific to the cloud.

2. PaaS (Platform as a Service): web based

- **Scalability:**
PaaS allows you to scale your applications without worrying about the underlying infrastructure. Example: Azure App Service — It automatically knows where to deploy your code based on the language you select (e.g., Java, Python).
- **Deployment & Control:**
With PaaS, you don't need to manage the server or hardware — just deploy your application (artifact).
Scaling is easy:
 - If your app receives 1000 requests tomorrow, you pay for the resources used.
 - The next day, if the demand reduces, you don't pay as much.
- **Ownership and Control:**
You have full control over the application, but the cloud provider manages the infrastructure (servers, network, etc.).
- **Data & Application Responsibility:**
While the platform manages scaling, you are still responsible for your app and data.
- **Lift & Shift:**
You can move your existing app (without major changes) to PaaS for better scalability and management.
- **AKS (Azure Kubernetes Service):**
AKS is a PaaS platform for managing and scaling containerized applications using Kubernetes.
- **Real-World Example:**
MakeMyTrip — A PaaS product that allows businesses to build and customize their services while the platform manages the infrastructure.

Q. Key characteristics of PaaS?

- **Integrated DB and Backend Services:** All parts of the system (e.g., database, APIs, and services) work together seamlessly.

Q. Benefits of PaaS?

→ Non-Functional Requirements:

- ◆ Solutions span across all modules, ensuring uniform performance, security, and scalability.
- ◆ Means: "solutions span across all modules," it means that the design of the system ensures these concerns (performance, security, scalability) are applied equally to all parts of the system, not just isolated to one area. This leads to a more reliable and effective system overall.

Functional requirements:

- **Authentication:** Users can log in securely across all parts of the system.
- **Example (Amazon):** When buying a product, backend services handle actions like verifying login, updating the database, and processing payment.

In service-based companies, after winning a project through an RFP and proposal, you gather both **functional** and **non-functional requirements**. While customers mostly focus on **features** (functional requirements), they also expect things like **performance, reliability, and quality** (non-functional requirements). Customers rarely mention these non-functional needs, but they expect them to be met.

RFP - document to submit proposals for a specific project

A **proposal** is a detailed document submitted by a vendor in response to an RFP.

Benefits of PaaS:

1. **Accelerated Dev Cycles:**
PaaS provides pre-configured tools, speeding up development and deployment.
2. **Reduced Complexity:**
No need to manage infrastructure, letting developers focus on the application.
3. **Lower Cost:**
You pay only for what you use, reducing the need for expensive hardware.
4. **Scalability and High Availability:**
Easily scale your app with built-in redundancy and uptime.
5. **Support Diverse Technologies:**
PaaS supports various programming languages and tools, offering flexibility.
6. **Improved Collaboration:**
Teams can work together in real-time with shared resources.

7. **Faster Time to Market:**

Streamlined processes help deliver products to market more quickly.

Challenges of PaaS:

1. **Vendor Lock-In:**

Migrating from one PaaS to another can be difficult and costly.

2. **Security:**

You have less control over security, relying on the provider's measures.

3. **Integration with Existing Systems:**

It can be complex to connect PaaS with existing on-premise or other cloud systems.

4. **Customization Limitations:**

Some PaaS solutions have restrictions on how much you can customize the environment.

5. **Performance Issues:**

Performance may vary, and you lack control over the infrastructure.

6. **Cost Management:**

Costs can increase unexpectedly if usage is not carefully monitored.

22/04/25

Strategy vs Tactics

- **Strategy** → Long-term plan (vision-level)
- **Tactics** → Short-term steps to solve specific problems (especially NFRs)
- NFR(Non Functional Requirements) -
 - ◆ Example: Incoming event → system processes → outgoing response
- large number of events(like req) - cause performance issues
- solving NFR issues have tactics
- tactics for solving performance and tactics for solving Scalability is diff
- **Tactic for Performance:** Use Redis caching, enable GZIP compression.
- **Tactic for Scalability:** Use Kubernetes HPA + Azure Load Balancer.

Scalability

- Ability of the system to handle increasing load gracefully
- **Load increases** → more users, requests, events
- **Resources can become constrained** → CPU, memory, DB connections

- Can cause **latency**, **timeouts**, or **crashes** if not handled
- **Solution** : horizontal scaling(more machines),**Auto-scaling**(based on CPU, memory, request count),distribute load equally by load balancer.

Q. why iaas is more vulnerable than paas and saas?

You **manage** the entire **OS**, **patches**, **security groups**, firewall rules, etc. A small misconfiguration (like leaving **SSH open** to the world) can expose your system. Attackers exploit **weak IAM policies**, open buckets, default credentials, etc.

Buildtime vs Runtime

- **Build time** : ok to take more time to build
- **Build Time** →
 - ◆ Time taken to **create Docker image**
 - ◆ Okay if it takes more time (once per deployment)
 - ◆ **Image Caching** helps here (layers reused)
 - ◆ image caching helps in build time not run time
- **Solution:**
- Use **Multi-stage Builds**
 - ◆ Only copy the required final binaries into the production image
 - ◆ Means - After building your app, just **take the output file** (like a **.jar**, **.pyc**, or built folder) and put **only that** into the final Docker image.
 - ◆ Don't include: Source code,Build tools,Temporary files
 - ◆ **Final binaries** - These are the files that are ready to **run**.
- Use **small base image**
 - ◆ (alpine -they have networking)
 - ◆ So that attack surfaces reduces
- **Run time:**Time when the **app is running in production**
 - ◆ Affects **performance** and **user experience**
 - ◆ Image caching doesn't help here

3. SAAS(software as a service)

- You access applications via a web browser
- You **don't need to install or manage** the app
- Eg: gmail,googledocs
- **Key features:**
 - ◆ **Web-based access** (via browser)
 - ◆ **Multi-tenant architecture** (one app, many users/companies)
 - ◆ **Accessible from anywhere** (just need internet)
- **Benefits of SaaS:**
 - ◆ **Lower upfront cost** - No need to buy or install software

- ◆ **Reduced IT overhead** - No need to manage infrastructure or updates
- ◆ **Scalability** - Easily increase/decrease usage as needed
- ◆ **Flexibility (Pay-as-you-go)**- Pay only for what you use(**only for what u use**)
- ◆ **Accessibility**- Use it anytime, anywhere on any device
- ◆ **Usability**- learn how to use it easily and continue to remember it ,then the product is usable..

scaling in kubernetes:

- hpa - adding more pods
- vpa - increase resource of pods (memory,cpu)
- ca - adding more nodes in a cluster

Some imp terms

- **Bastion Host:** 1 machine in cloud ..from there u connect to other machines,ensure security from public cloud
 - A **simple VM** in the cloud used to **safely access other machines** (like app servers, DBs)
 - **Main purpose:** Secure admin access to VMs in **private networks**
 - Prevents exposing all machines to the public internet

Q. why use bastion host?

- No direct public access to critical servers
- Login only through Bastion Host = extra layer of security
- Think of it like:
 - 🔒 *"Gold in a locker — unless you know the full path and unlock the locker, you can't get it."*
- Notes: Port 443 → Used for HTTPS (secure web traffic)
 - ◆ Need to open specific ports when doing updates/upgrades (like SSH 22, HTTPS 443, etc.)
- **Vnet:**
 - VNet is like your own private network in the cloud
 - Used to **securely connect** your cloud resources like VMs, databases, and apps
 - Create subnet within the vnet
 - Subnet can be public and private
 - Inside subnet there will be vm
 - A **VNet spans the entire region, Ex: Central India region**
 - Includes **multiple Availability Zones (AZs)** like DataCenters in **Trivandrum (TVM) and Kochi**

Subnet:

A range of IP addresses in a VNet. Subnets help organize your network and can be used for isolating or segmenting different workloads.

→ **Availability Zones:**

- ◆ Each AZ = separate datacenter with independent power, cooling, backup, and infra
- ◆ VM gets deployed in one of the AZs within a region


→ **Subnet within the vnet:**

- You can create **subnets** for specific purposes (App, DB, Bastion, etc.)
- You can also **assign subnets to specific AZs** ➤ So **only allowed AZs can communicate** (controlled)

Q. VNet is Account Specific:

Each VNet is tied to your Azure account, You **cannot access someone else's VNet** unless it's **shared** with you

Q. Bastion Host with VNet:

- You can attach Bastion Host to a VNet
- You need:
 - ◆ Access to correct IP
 - ◆  .pem file or credentials
- If user tries to connect from unauthorized IP → Blocked

Q. VNet – Public and Private

Public VNet:

- Can be accessed from the internet
- Public IPs assigned to resources (VMs, Load Balancers, etc.)

Private VNet:

Cannot be accessed from the internet

Used for internal resources only

Bastion Host can be used within a private VNet to access internal resources securely

External users can't directly access the private VNet (must go through secure methods like Bastion Host)

Q. Region in Azure

- ❖ **its a geographical region where the azure has the datacenter**
- ❖ azure having datacenter in central india(bombay), south and west
- ❖ within region there will be availability zones- independent data centers(1 dc crashes other used..what should azure do?)
- ❖ **1 dc crashes other used..what should azure do?**

- entire infrastructure(backup,power,network) system independent
- u can have malware,terrorist attack
- keeping dc soo away (1 is attacked then use other eg: earthquake also)
- restart,os upgrade - group of machines they do it together

Q.vnet peering?

- Consider there are 2 vnets
- 1 from san francisco and 1 from india
- 2 vnets having diff range of ip where subnetting principle comes
- Default vnet ip range -same ,so diff range needed for both vnet to possible
- Soo if we need to connect the vnet ,s without using internet we can do vnet peering b/w the vnet
- After vnet peering the machines from the both vnet can talk using private ip

Q. bastion host helps to communicate using private ip within the vnet.how?

- Vm inside the Vnet can communicate without the internet.
- Bastion Host sits inside your VNet.
- You connect to Bastion through the Azure portal.
- Bastion then connects to your VM **using its private IP, without needing internet** or a public IP.
- So, Bastion acts like a **secure door** to reach your VMs inside the private network.

Note:

You connect to the **Bastion Host** through **Azure Portal** using HTTPS (port 443).From the Bastion Host, it initiates **RDP/SSH connection** to the target VM using its private IP address.

Q. private cloud:

- protected
- u have better control for dc(people with right access)
- more security
- still vulnerable to various kinds of attacks
- install vpn

Q. scaleset - It is a service

- automatically scale no of vm
- create load balancer in front of scale set
- If load increases,vm increases

Q. fault domain - distribute vm across multiple fault domains

- 5 vm created from 3 phy machines
- if 3 of the phy machines undergoes (restart,upgrade)-maintenance affected
- solution : fault domain

23/04/25

Q.while creating vm what are things we are doing in azure?

- mention **resource grp**-(like a container to manage multiple resources)-(VM, disk, VNet, etc.)
- Disk (OS disk is attached by default.)
- Network(**VNet** (Virtual Network) and **Subnet** must be attached to allow network communication.)
- **CPU & Image Size**(choose vm size - based on CPU,Ram,Disk),select **os image(ubuntu,windows)**
- VM Scale Set
- Username & Authentication(**password or pem file for ssh login**)
- Tags(key value pairs)
- Inbound & Outbound Traffic
 - ◆ Define **inbound rules** (e.g., open port 22 for SSH or 3389 for RDP).
 - ◆ **Outbound** is usually allowed by default unless restricted.

Q. What is subnetting?

- Let's consider 3 diff network - deepa,hr,azure
- when the req come = need CIDR (10.0.0.0/24)
 - ◆ What is this 24?
 - ◆ this means $2^{32-24} = 2^8 = 256$ machines
 - ◆ 10.0.0.0 to 10.0.0.255
- for 3 diff network we need table
- | | | | | | | | | | |
|-------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| network | 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | 256 |
| host | 256 | 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
| subnet mask | /24 | /25 | /26 | /27 | /28 | /29 | /30 | /31 | /32 |
- - ◆ 24 =256 machines
- now i want to create a separate network for hr (divide network to 3(hr,deepa,azure))
- no way u can create 3 network = only option = dividing into 4
- each network i get 64 machines
- hr ip range = 10.0.0.1 - 10.0.0.62
- prblm :
 - ◆ network ip(10.0.0.0),broadcast ip(10.0.0.63)
- 10.0.0.0/26
- deepa = range =10.0.0.65 - 10.0.0.126
- network =10.0.0.64 broadcast =10.0.0.127
- 10.0.0.64/26
- azure = range =10.0.0.129 - 10.0.0.190
- network =10.0.0.128 broadcast =10.0.0.191
- 10.0.0.128/26
- Network ip: 1st ip is reserved for network
- Broadcast ip : the last ip is reserved for broadcast

Q. Azure IP Reservation Rule?

- Azure reserves 5 ip (first 4 and last1)
- **First 4 IPs:** Reserved for **protocols and Azure services** (e.g., .0 - .3)
- **Last IP:** Reserved for **network broadcast** (e.g., .255)

Q. permission denied concepts in our vm?

- Creating a new resource group
- So need to use the existing one
- Cant move vnet to other regions
- Tried to create users -failed

Q. NAT Gateway?

- You created a private subnet
- VMs inside this cannot be accessed from the internet.
- To allow these VMs to go out to internet, you need a NAT Gateway

PRACTICAL:

- created virtual network with ip range 10.1.0.0/16 for **front-end**
- **Back-end** = 10.1.1.0
- **Private** = 10.1.2.0

Microsoft Azure portal interface showing the "Edit subnet" configuration page. The left sidebar displays the "Create virtual network" wizard with tabs for Basics, Security, IP addresses, Tags, and Review + create. The main content area is titled "Edit subnet" and includes instructions: "Select an address space and configure your subnet. You can customize a default subnet or select from subnet templates if you plan to add select services later. [Learn more](#)".

The configuration fields are as follows:

- Subnet purpose: Default
- Name: abhi-front-end
- IPv4 section:
 - Include an IPv4 address space: ☒
 - IPv4 address range: 10.1.0.0/16 (10.1.0.0 - 10.1.255.255)
 - Starting address: 10.1.0.0
 - Size: /24 (256 addresses)
 - Subnet address range: 10.1.0.0 - 10.1.0.255
- IPv6 section: (empty)

At the bottom, there are "Save" and "Cancel" buttons, and a "Give feedback" link.

Microsoft Azure portal interface showing the "Edit subnet" configuration page, specifically the "Security" and "Service Endpoints" sections. The left sidebar is identical to the previous screenshot.

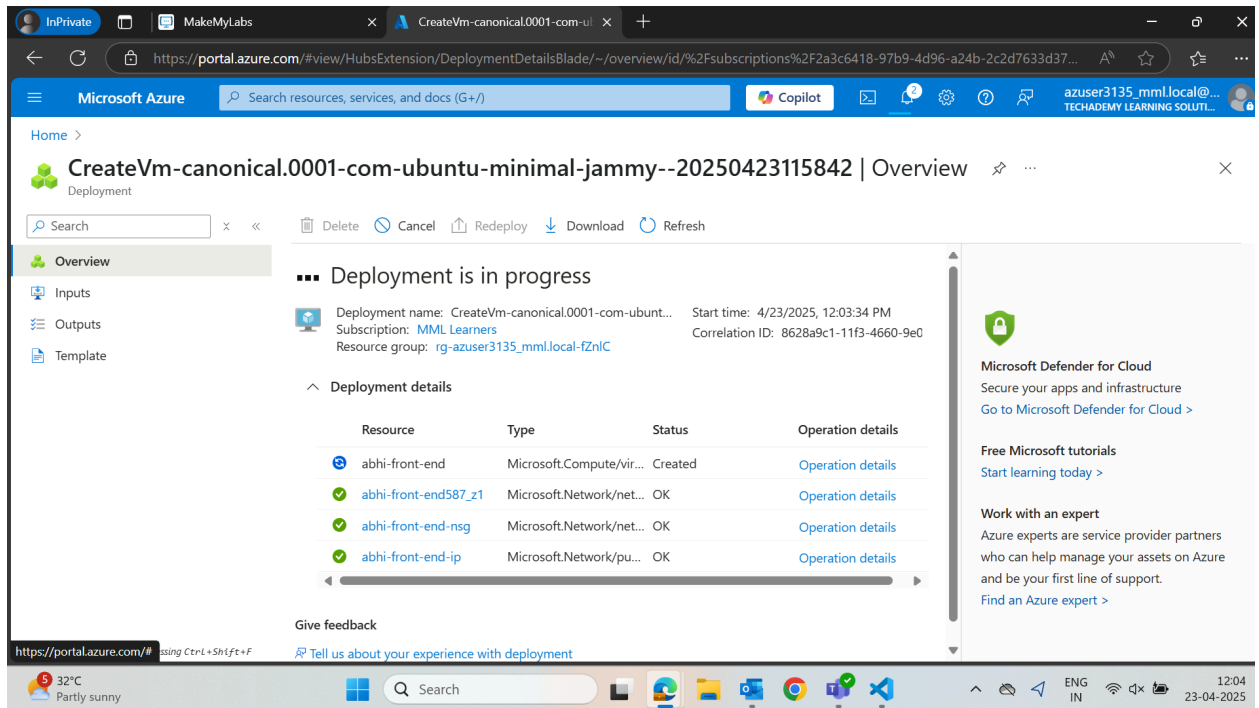
The "Security" section includes the following configuration:

- NAT gateway: None (with a "Create new" link)
- Network security group: None (with a "Create new" link)
- Route table: None

The "Service Endpoints" section includes:

- Services: (empty list)
- Remove service endpoint: (button)

At the bottom, there are "Save" and "Cancel" buttons, and a "Give feedback" link.



1. Created a **VNet** with 3 **subnets**:
 - **Frontend Subnet** (Public)
 - **Backend Subnet** (Public — but no Public IP assigned)
 - **Private Subnet**
2. **Virtual Machines**
 - **Frontend VM**
 - **Subnet: Frontend**
 - **Public IP assigned** ✓
 - **Can SSH from internet**
 - **Backend VM**
 - **Subnet: Backend**
 - **No Public IP** ✗
 - **Used for internal communication**
 - **Private VM**
 - **Subnet: Private**
 - **No Public IP** ✗

- Intended to be completely private (internal access only)
-

◆ 3. Network Security Groups (NSGs): Defining who can access what

◆ Created 3 Separate NSGs:

- Frontend NSG
- Backend NSG
- Private NSG

◆ Associations:

- Frontend NSG → Associated with Frontend VM
 - Backend NSG → Associated with Backend VM
 - Private NSG → Associated with Private VM
-



◆ 4. NetworkSecurityGroup Rules

✓ Frontend + Backend (Public) Subnets:

- Inbound Rule:
 - Source: *
 - Destination Port Ranges: *
 - Action: Allow
 - Priority: Default or custom (lower number)

Allows general access from any source (for demo/testing).

✓ Private Subnet Rules (Important):

-  **Allow from Backend:**
 - **Priority: 100** (higher priority means evaluated first)
 - **Source: Backend subnet or IP**
 - **Action: Allow**
-  **Deny All Rule:**
 - **Priority: 200** (lower than allow rule, so checked after)
 - **Source: ***
 - **Destination Port Ranges: ***
 - **Action: Deny**

 **This setup only allows traffic from Backend to Private VM, blocks everything else.**

Note:

- Copied the pemfile into frontend, backend and private.
- For copy : `scp -i pemfile pemfile username@ipaddress:/home/username`
- Then try to connect using ssh to frontend
- After connection, connect to backend from frontend using privateip of backend
- After that try to connect to private

Q. communication happen in public domain encrypted or not?

- Yes ., communication in the **public domain is encrypted**, if you're using secure protocols.
- Ssh - encrypted using pem file
- When going **over the internet** (e.g., Cloud Shell → VM), **encryption happens via SSH**.
- Communication **between VMs inside a VNet** is **private**, not exposed to the public internet.'
- So yes, **communication is encrypted in the public domain if you're using the right protocols** like HTTPS or SSH.

Q. BGP?- Border Gateway Protocol

- BGP is a **routing protocol** used to exchange routing info between **different networks**
- Finds best path
- Azure uses BGP for **hybrid networking**, like when connecting your **on-premise network to Azure** using **VPN Gateway or ExpressRoute**.

- Protocol runs the internet

24/4/25

Q. How did you create a user in our azure platform?

- tried to create user -> **entraid** -> **manage** in side -> **users** -> add user =failed
- because of no **p1 and p2 licenses**
- Sir assigned roles -> sir done
 - ◆ entraid-manage-users
 - ◆ users
 - ◆ selected user(user2)
 - ◆ assigned roles -add assignment - user administrator- adding
- Then we created a user =worked
- Only group administrator have access(yes)

Q. what u have done to give permissions to the user?

- The user can be added to the Azuregroup(created by sir)
- Sir created an Azure group and added all of us into it
- The Azuregroup contains all the subscriptions we need
- Users in the group can **access resources under that subscription.**

Q. After creating user what u have done ?

- After creating the user you can login to the user
- Here in Entra id ->group ->azure group -> members = u can see the user(you created)
- but cant add members - because no permission for this user(newly created user)
- The **new user cannot add more members** to the group unless explicitly given **Group Owner** or **Admin roles**.

Q. How to attach a user to a subscription in Azure?

- in subscription u r associating user to role
- Search subscriptions
- Select the subscription(eg:mml)
- Click on IAM
- + Add > Add role assignment
- **Role** → Select a role like **Reader**, **Contributor**, or **Owner**(based on what permission the user needs)
- **Assign access to** → **User**, **group**, or **service principal**
- **Select** → **Type and select the user you want to give access**
- The user now has access to that **subscription** based on the role.

Q. How do you create Resource Group in Azure?

- Search for Resource group

- Create
- Select subscription
- Select region
- create

Q. how do you create vm in the resource group you created?

- Select the resource group that u created

Q. How do you request quota for the region you are working with?

- You can request the quota
- Subscription - mml - settings -usage+quota -select region and request

Q. Why NFR is mostly handled by saas?

- SaaS is like **renting a fully serviced apartment** — you just move in and live, while the owner handles the electricity, plumbing, security, and maintenance.
- The provider is responsible for NFR
- Scalability,Availability,Security,Performance,Disaster Recovery,Compliance

Q. Challenges of public cloud?

Compliance	Following rules/laws for data storage (e.g., GDPR, HIPAA).
Vendor Lock-In	Getting stuck with one cloud provider because switching is hard.
Cost Management	Costs can increase quickly if you don't monitor or control usage.
Integration Issues	Difficult to connect old systems (on-premises) with cloud systems.
Security Concerns	Data is stored outside your system, so it needs strong protection (like IAM, firewalls).
Limited Control	You don't get full control over the hardware or some advanced configurations.
Performance & Latency	Might be slow for real-time tasks if data centers are far (without edge computing).

Q. uses of public cloud?

Storage	Save data like files or backups on the cloud.
Website Hosting	Host your website without worrying about servers.
Data Analytics	Analyze large amounts of data with cloud tools.
Disaster Recovery	Backup your data on the cloud, so you can recover if things go wrong.
SaaS (Software as a Service)	Use software online, like Google Docs, without installing it.
Development & Testing	Build and test apps on the cloud, so you don't need a lot of hardware.
Big Data & AI	Do complex calculations or train AI models without owning big computers.

Q. Azure Global Infrastructure?

Azure Global Infrastructure refers to the complete network of

- data centers,
- regions,
- availability zones, and connectivity components






Microsoft uses to deliver Azure cloud services across the world.

- ★ **Regions:** Specific geographic areas (e.g., Central India, East US) with Azure data centers.
- ★ **Availability zones:** Separate data centers **within** a region for **high availability**. If one zone fails, others stay up.
- ★ **Regional Pairs:** Two regions paired (e.g., **Mumbai-Hyderabad**) for **disaster recovery**. One fails, traffic moves to the other.
- ★ **More regions:** Azure has **more regions globally** than AWS (including **Mumbai and Hyderabad** in India).
- ★ **Edge location :** Small data centers close to users (in different parts of the world) to deliver content quickly.
 - CDN(content delivery Network): A network of servers that deliver content (images, videos, etc.) faster to users.
 - Reduces **traffic** to main servers and **loads content faster** for the user.
 - You visit a website → it wants to load an image
 - Instead of sending your request to the **main server in US**,
 - It checks the **nearest Edge Location** (e.g., in India)
 - If image is already stored (cached) there → it downloads it instantly
 - **Result:** Faster page load, less delay, less internet traffic

Q. Key aspects of cloud region?






- **Geographic Location:**
 - ◆ The physical location where the cloud data centers are placed.
- **Multiple Data Centers (DC):**
 - ◆ A region has more than one data center to ensure high availability and backup.
- **Isolation:**
 - ◆ One region doesn't affect the other if something fails — safe and independent.
- **Low Latency:**
 - ◆ Closer region = faster response time (less delay in getting your data).

Q. Benefits of Cloud Regions?

 Benefit	 What it Means
 Disaster Recovery	Run data across regions (active-active or active-passive) to avoid data loss.
 Scalability	Can easily scale resources within a region based on your needs.
 Data Sovereignty & Compliance	Store data in specific regions to meet legal and compliance rules.

Q. Characteristics of Availability Zones?

Characteristics of Availability Zones

 Characteristic	 What it Means
 Low Latency	Faster access to services and applications because they're closer to users.
 Redundancy	If one zone fails, another will take over — ensuring uptime.
 Fault Domain	Each zone is isolated, so failures in one won't affect others.

Q. Key Characteristics of Azure Cloud Data Centers?

- **Security:** Strict physical & logical security controls to protect your data.
- **Connectivity:**
 - ◆ Private IP-based for secure VM-to-VM communication within Azure.
 - ◆ VNet Peering to connect VNets across regions securely.
 - ◆ NAT Gateways to access the internet without exposing private IPs.
 - ◆ VNet Endpoints to access Azure services like Azure Files privately with low latency.
- **Note on Public IPs:** 2 Azure VMs communicating using public IPs won't use Azure's internal network = slower.
- **Resource Groups:**
 - ◆ - Logical container to manage resources like VMs, disks, IPs, etc.
 - ◆ - RG can be in one region, but resources inside can be in any region.
 - ◆ - Azure allows nesting groups (group inside group).
 - ◆ - Stores metadata about resources

Q. Benefits of using Resource Groups in Azure:

- **Manage the cost :**
 - ◆ cost of various services used in rg
 - ◆ Helps to manage billing efficiently.
 - ◆ set up budgets, alerts
- **Access Control**
 - ◆ assign permissions or roles to users at the resource group level using RBAC
- **Simplified Management**
 - ◆ It's easier to handle and organize related resources (like VMs, disks, IPs) together in one group. You can also use **tags** for better tracking and organizing.

Q. how do u manage resource grps?

- azure portal, cli, arm template, sdk

Q. Azure Resource Manager (ARM)?

- It's the **management layer** in Azure.
- Handles **deployment, management, and organization** of Azure resources.
- You can define resources using **ARM templates** (JSON).
- Supports **RBAC, policies, and tagging** for governance.
- You can group related resources into **Resource Groups**.

Q. Resource Locking?

- Prevents accidental deletion or modification.
- Two types:
 - **ReadOnly** – can't update or delete.
 - **CanNotDelete** – can read and update, but cannot delete.
- Lock can be applied to resource, resource group, or subscription.

Q. Azure subscriptions?

A container for resources and billing.

- Every subscription is associated with a payment account.
- Different subscriptions are independent — resources in one don't affect the other.
- Helpful for organizing departments, environments, or projects.

Q. azure account:

- account associated to **email id** and internally associated to **entra id** (database of users)-these all associated to tenant
- Entra id =database of users

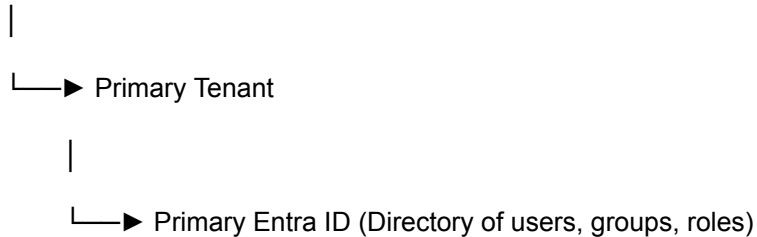
Q. What You Get When You Create an Azure Account?

- **Account** – Linked to your email.
- **Tenant** – Your own isolated space in Azure.
- **Azure Entra ID** – Where your user is registered.
- **Default Subscription** – Helps define policy and billing boundaries.
- *(Optional)* **Resource Group** – You can create this later; **not automatically related**.

Q. primary entra id - associated with tenant ?

- When you create an **Azure account**, a **primary Entra ID (directory)** is automatically created and **linked to a specific tenant**.
- The **Primary Entra ID** is the **identity store** for that tenant.
- All users, groups, and RBAC permissions are managed **inside this Entra ID**.

Azure Account (your email)



Q. Azure subscription types?

- Free Trial
- Pay-As-You-Go (PAYG)
- Azure AD Premium P1
- Azure AD Premium P2

Q. when you create the group, the membership type becomes assigned. Then what happens?

- When you create a group in **Azure Entra ID (formerly Azure AD)**, the **membership type** depends on your license level:
 - ◆ **Assigned Membership (Free / Default)**
 - You manually add or remove users to the group.
 - Available in **Free tier**.
 - ◆ **Dynamic Membership (P1 or P2 required)**
 - Users are added/removed automatically based on rules (like department, location, etc.).
 - Available in **Azure AD Premium P1 and P2**.
 - Example: `user.department == "IT"` → Auto-added to "IT Team" group.
- it will change in p1 and p2-(dynamic)
- **dynamic membership** becomes available only **if your tenant has P1 or P2 license**.
- Otherwise, it stays as **Assigned**.

Q. forecast and actual in azure budgets?

- **Forecast::**
 - ◆ The forecast is an estimate of how much you will spend on Azure services based on historical data and trends.
 - ◆ **Example:** Let's say in the last month, you used Azure services worth \$100. Based on that usage, Azure predicts that you will spend around \$120 in the next month based on your usage patterns. This prediction is the **forecast**.
- **Actual:**
 - ◆ The actual amount is the real cost you've incurred so far for Azure services.

- ◆ **Example:** After 15 days into the next month, you check your actual usage and find that you have already spent \$60 on Azure services. This is your **actual** spending up to that point.

Q. Vnet pairing practical?

- create vnet using azure client
- **Login to Azure CLI:**
 - ◆ az login
- **Create Virtual Networks (VNETs):**
 - ◆ Creating 2 vnets of diff range of ip to do vnet pairing
 - ◆ First vnet:
 - az network vnet create --resource-group abhi-resource-group --name abhi-vnet --address-prefixes 10.3.0.0/16
 - ◆ 2nd vnet:
 - az network vnet create --resource-group abhi-resource-group --name abhi-vnet2 --address-prefixes 10.4.0.0/16
- **Create Subnets:**
 - ◆ For each VNet, create subnets to define the network structure.
 - ◆ **Create subnet for abhi-vnet:**
 - az network vnet subnet create --resource-group abhi-resource-group --vnet-name abhi-vnet --name abhi-subnet --address-prefix 10.3.0.0/16
 - ◆ **Create subnet for abhi-vnet2:**
 - az network vnet subnet create --resource-group abhi-resource-group --vnet-name abhi-vnet2 --name abhi-subnet --address-prefix 10.4.0.0/16
- **Create Virtual Machines (VMs):**
 - ◆ create VMs in each of the VNETs to test connectivity between them.
 - ◆ **Create VM in abhi-vnet:**
 - az vm create --resource-group abhi-resource-group --name abhi-vm --location centralindia --image Ubuntu2204 --size Standard_B1s --vnet-name abhi-vnet --subnet abhi-subnet --admin-username abhi --admin-password abhi@12345678
 - ◆ **Create VM in abhi-vnet2:**
 - az vm create --resource-group abhi-resource-group --name abhi-vm2 --location centralindia --image Ubuntu2204 --size Standard_B1s --vnet-name abhi-vnet2 --subnet abhi-subnet --admin-username abhi --admin-password abhi@12345678
 - If you face a quota error (insufficient resources), request a quota increase for the Central India region.
- **Connect to VMs Using SSH:**
 - ◆ ssh abhi@<public-ip-of-abhi-vm>
 - ◆ ssh abhi@<public-ip-of-abhi-vm2>
 - ◆ Test if you can **ping** one VM from the other using their public IPs.
- **Ping from VM1 to VM2:**
 - ◆ ping <public-ip-of-abhi-vm2>
- **Set Up VNet Peering:**
 - ◆ You need to configure VNet peering in both VNETs so they can communicate with each other.
 - ◆ **Peer abhi-vnet with abhi-vnet2:**

- az network vnet peering create --name abhi-peering --resource-group abhi-resource-group --vnet-name abhi-vnet --remote-vnet abhi-vnet2 --allow-vnet-access
- ◆ **Peer **abhi-vnet2** with **abhi-vnet**** (you need to create peering in both directions):
 - az network vnet peering create --name abhi-peering --resource-group abhi-resource-group --vnet-name abhi-vnet2 --remote-vnet abhi-vnet --allow-vnet-access
- **Test Private IP Connectivity:**
 - ◆ After VNet peering is established, try to ping the VMs using their private IPs.
 - ◆ **Ping from VM1 to VM2 using Private IP:**
 - ping <private-ip-of-abhi-vm2>
 - ◆ **Ping from VM2 to VM1 using Private IP:**
 - ping <private-ip-of-abhi-vm>

25/4/25

Q. 2 things we are doing to give permission?

- Add user to **Azure group**
- **Mml subscription** -> contributor/reader
- Contributor cannot **add/modify roles**
- Add role to **resource group** -> the user will get privileges in the resource group only.

Q. Roles?

- **Roles** define what **actions** an **identity** can perform (e.g., read, write, delete).
- **Identity**- users, group, service principle-access to service
- when u attach a role to identity give access based on **scope**
- **Scope**- management grp, subscription, resource group, resource
- A role only applies **within the scope** it is assigned (and scopes below it).

Q. . Why Use Service Principals Instead of Users?

- If access is given directly to a **user**, and that user leaves or is removed, the **service breaks**.
- **Service Principals** are **non-human identities** created specifically for apps or services.
 - ◆ They provide a **stable identity for services** to authenticate and access resources.
 - ◆ Managed by Azure AD.

Q. Management Group?

- A container for **subscriptions**.
- You can assign **roles** at this level to control access across multiple subscriptions.
- **Subscription:**
- Contains **resources** and is tied to **billing**.
- subscription will be in tenant group
- Can have different subscription for diff management groups

- subscription within tenant associated to **payment**
- You can have **multiple subscriptions in a tenant**, useful for separating environments (prod, dev, etc.).

Q. Microsoft Entra ID ?

- **Identity and Access Management (IAM)** service from Microsoft.
- Manages **users, groups, service principals, roles, devices**, etc.
- **active directory** to maintain identity
- It helps with **authentication** (verifying users) and **authorization** (giving users the right access to resources).
- user management, access to management users -not treated as resources
- They are more like security objects that help manage access and roles across Azure resources.
- You do **not need** subscription-level permissions to **create users** or **groups**. These are controlled at the **Azure AD level** (not subscription level).
- Permissions for **user management** are controlled by **roles** assigned to users within **Azure AD**.

Q. user administrator vs global administrator?

1. User Administrator:

- **Primary Focus:** Manage **users** and **groups** within Azure AD.
- **Permissions:**
 - ◆ **Create, Update, Delete** users.
 - ◆ **Reset passwords** for users.
 - ◆ **Create, Update, Delete** groups.
- Assign users to roles but with restricted access to global settings.

2. Global Administrator:

- Can manage all aspects of Microsoft Entra ID and Microsoft services that use Microsoft Entra identities.
- **Primary Focus:** Full administrative control over Azure AD and all Microsoft services tied to Entra identities.
- **Complete access** to all aspects of **Azure AD**: manage users, roles, security policies, and configurations.
- Can configure **enterprise-level settings**, such as **conditional access, multi-factor authentication**, and directory settings.

Q. Eventual Consistency:

- This is a model where, over time, a system will become consistent, but there may be temporary states where data is inconsistent. In other words, the system doesn't guarantee immediate consistency after each update, but eventually, all nodes (or replicas) will converge to the same state.

- eg: **Amazon Payments**: If you purchase an item, the payment status might not be immediately reflected across all systems. For example, while one system says “payment pending,” another system may say “payment processed.” Over time, these systems will sync, and the data will become consistent, but there's no guarantee of immediate consistency.
- Eg: nosql database

Q. ACID Properties (for Relational Databases):?

Atomicity:

- Either all parts of the transaction are completed, or none of them are.
- If you're transferring money from **Bank A to Bank B**, the transaction will either:
 - Fully complete (money deducted from Bank A, added to Bank B), or
 - **Rollback** if any part fails (like a system crash).
 - If the transfer fails midway (e.g., money withdrawn from A but not deposited in B), the system ensures that the transaction is rolled back to its initial state.

Consistency:

- A database must move from **one valid state to another valid state**. The data must always meet the rules or constraints defined for the database (e.g., foreign keys, data types).
- **Example**: If you write data to the database, the data should be **consistent** with the database's rules. For instance, if you try to store a negative value in a column that only allows positive numbers, the database will reject it.
- Whatever written = whatever reading.
- **NoSQL** (e.g., **Cassandra, DynamoDB**) focuses on **eventual consistency**, where data may be inconsistent temporarily but will eventually become consistent across all nodes.

Isolation:

- **Definition**: Transactions should be **isolated** from each other. Even if multiple transactions are happening at the same time, they should not interfere with each other's operations.
- **Isolation levels**:
 - ◆ **Read Uncommitted**: Transactions can read uncommitted changes.
 - ◆ **Read Committed**: Only committed changes are visible.
 - ◆ **Repeatable Read**: The data read by a transaction cannot be changed by other transactions.
 - ◆ **Serializable**: The highest level, ensuring that transactions are executed serially (one after the other).
 - ◆ **Example**: If two users are trying to withdraw money at the same time, **isolation** ensures that one transaction doesn't interfere with the other.

Durability:

- **Definition**: Once a transaction is committed, it is permanent, even in the case of a system crash.
- **Example**: If a transaction to update a bank balance is committed, the change is saved, and even if the database crashes immediately afterward, the transaction will not be lost.

Q. Primary, Secondary, Territory -related to databases?

→ **Primary:**

The main database instance where all write operations happen first.

→ **Secondary:**

A copy of the primary; it syncs data from the primary and can be used for reading (depending on the setup).

→ **Tertiary:**

Another replica — basically a third copy to add even more redundancy.

→ **Read Replica:**

A replica used *only* for read queries to reduce load on the primary. Updates from the primary are asynchronously synced.

Q. . NoSQL Database – Sharding?

→ Breaking a large database into smaller parts (shards) to distribute across multiple machines.

Example:

→ Shard 1 → keys a - e

→ Shard 2 → keys f - j

→ And so on.

→ **Replication Rules:**

→ Replica stored on at least two other machines:

Every piece of data is stored not just on the primary node, but also on at least 2 other nodes (for fault tolerance).

→ Transaction commits - $n/2 + 1$ instances written:

For a transaction to be considered successful:

◆ It must be written to more than half the nodes.

◆ If there are n total replicas, at least $(n/2) + 1$ must acknowledge.

Eventually Consistent Sync

- Eventual Consistency:

The replicas might not be immediately consistent after a write, but over time, all copies will have the same data.

Q. Runbook?

Every day evening, a runbook will trigger automatically.

Purpose:

→ Check if any VM is active (running).

→ Based on our wish: terminate or stop the VM.

Q. azure client and powershell:

- az login # Login to Azure account
- az --version # Check Azure CLI version
- az account list # List all subscriptions
- az account list --output table # Show subscriptions in table format
- az config set defaults.location="centralindia" # Set default location as centralindia
- az config set defaults.group="abhi-resource-group" # Set default resource group
- az account set --subscription "your-subscription-id" # Set subscription

- az group list # list all resources
- az group list --query "[?location='centralindia']" # List resource groups in centralindia
- az group list --query "[?name=='abhi-resource-group']" # List specific resource group
- az group list --query "[].name" -o tsv # List all resource group names in TSV format

- az vm list # List all virtual machines in the active subscription
- az vm list -g Admin-Azure --query "[].name" --output tsv # List VMs in specific resource group
- az vm list -g admin-azure --query "[].{Name:name}" --output tsv
- az vm list -g Admin-Azure --query "[].{Name:name,ResourceGroup:resourceGroup}" --output tsv

- az storage account list # List all storage accounts in the active subscription
- az storage account list -g abhi-resource-group # List storage accounts in specific resource group
- az storage account list -g abhi-resource-group --query "[?location=='centralindia']" --output table # List storage accounts in centralindia

- az storage account create --name abhi-storage --resource-group abhi-resource-group --location centralindia --sku Standard_LRS --kind StorageV2 # Create a new storage account
- az storage account create --name abhistorage2025 --resource-group abhi-resource-group --location centralindia --sku Standard_LRS --kind StorageV2 # Create another storage account
- az storage account list -g abhi-resource-group --query "[?location=='centralindia']" --output table # List storage accounts in centralindia for specific resource group
- az vm list-skus --location "centralindia" --query "[?starts_with(name, 'Standard_DS')]" --output table # List VM SKUs available in centralindia region starting with 'Standard_DS'
- env | grep -i storage # List all environment variables related to storage
- history # Show the history of commands run in the shell
- az storage account list # List all storage accounts in the active subscription
- az vm list-skus --location "centralindia" --query "[?starts_with(name, 'Standard_DS')]" --output table # List available VM SKUs in 'centralindia' that start with 'Standard_DS', displayed in table format

Q. storage account and Resource group?

- A **Storage Account** in Azure is a resource that provides storage services for your data in the cloud. It serves as a container for storing different types of data in Azure, such as files, blobs(manged using containers), tables(no sql database), and queues.
- When you create a **storage account**, it is often created within a **resource group**.
- If you are using the **Azure UI**, you will be prompted to choose an existing resource group or create a new one.
- If you don't specify a resource group, Azure will create the storage account in the **default resource group**.
- `env | grep -i storage` # used to display the environment variables related to storage, which includes things like your storage account settings.
- **My doubt ans:** If created the storage group in my default resource group, and if i need to get that storage acc by executing the command `env | grep -i storage`, i need to connect it with the cloud shell.(reset user settings)
- added by clicking the storage class and created **files abhi-files**
- `az storage account list` # can see everyones storage acc

Q. powershell commands:

- `Connect-AzAccount` # Logs into your Azure account (interactive login)
- `Connect-AzAccount -UseDeviceAuthentication` # Logs into Azure using device authentication (usually for secure login on new devices)
- `get-AzContext` # Gets the context of your current Azure session (subscription, account, and other settings)
- `set-AzContext -Subscription "<subscription-id>"` # Sets the current Azure subscription context to the specified subscription ID
- `get-AzSubscription` # Lists all Azure subscriptions available for the current account
- `get-AzResourceGroup` # Lists all Azure resource groups under the current subscription
- `get-AzResourceGroup --help` # Shows help information for the 'get-AzResourceGroup' cmdlet
- `get-AzResourceGroup -Location "centralindia"` # Lists all resource groups in the 'centralindia' location
- `get-AzResourceGroup -Name "abhi-resource-group"` # Gets details of the resource group named 'abhi-resource-group'
- `get-AzVM` # Lists all virtual machines in the current subscription
- `remove-AzVM` # Removes a virtual machine (it needs a VM name to specify which VM to delete)
- `remove-AzVM -Name "PRADHISHA-RG"` # Removes the virtual machine named 'PRADHISHA-RG'
- `new-AzVM -Name "abhi-vm"` # Creates a new virtual machine named 'abhi-vm' in the current resource group
- `new-AzVM -Name "abhi-vm"` # Creates a new virtual machine named 'abhi-vm' (repeated)
- `new-AzVM -Name "abhi-vm"` # Creates a new virtual machine named 'abhi-vm' (repeated)
- `new-AzVM -Name "abhi-vm"` # Creates a new virtual machine named 'abhi-vm' (repeated)

- `remove-AzVM -Name "abhi-vm" -ResourceGroupName "abhi-resource-group"` # Removes the 'abhi-vm' VM from the 'abhi-resource-group' resource group
- `get-AzVM` # Lists all virtual machines again after deletion

Q. Saving plan and Reservations in Azure?

- In Azure, **Savings Plans** let you save money by committing to a certain amount of usage over time, while **Reservations** offer discounts for reserving specific resources like VMs for one or three years.

Q. How do I automate the termination of Spot Instances and the stopping of Normal Instances based on a scheduled time using a script in Azure?

- Create an automation account
- Give permission as contributor
- Create a runbook(powershell)
- Edit runbook
- Paste the code
- Publish
- Create a schedule in automation acc
- Link the schedule to runbook

Q. what is automation account?

- An **Automation Account** in Azure is a cloud service that provides a way **to automate repetitive tasks and processes** in your Azure environment, such as resource management, configuration, monitoring, and maintenance tasks.
- It enables the **automation of actions** like starting or stopping virtual machines (VMs), scaling resources, deploying updates, managing configurations, and handling system events—**without** needing to **do so manually each time**.
 - ◆ **Key features:**
 - **Runbooks:** automate tasks like stopping vm
 - **Update Management:** update vm to make it secure
 - **Scheduling:** set tasks to run at specific time

Q. How to create automation account in azure?

- Search for **automation account**
- create

Q. What are Managed Identities in Azure?

- **Managed Identities** are identities provided by Azure Active Directory (azure AD)
- allow Azure services to authenticate to each other **without the need for credentials**
- **managed identities belongs to groups**
- **managed identities work within scope of azure**
- **Credentials:** password and secret(stored in code)
- Credentials are automatically managed by azure

- We don't want to handle
- Need to give permissions to managed identities
- **Two types of managed identities:**
 - ◆ **System-Assigned Managed Identity:**
 - Created automatically **when you enable managed identity for a resource** like a virtual machine (VM), app service, or function.
 - If the **resource** is deleted, the **identity is also deleted**.
 - The identity is used only by **that specific resource**.
 - ◆ **User-Assigned Managed Identity:**
 - We need to create it
 - Created separately from any resource and can be assigned to one or more resources.
 - It stays around until you manually delete it.
 - Used across multiple resources

Q. To see all resources/services who use managed identities?

- entraid - enterprise application - application-type=managed identities -apply

Q. How can I give access to managed identities?

- ResourceGroup - IAM - Add role assignment - Reader/contributor
- In reader - member - Managed identity -select member -managed identity(automation account)
- U can edit the role later
- Once access granted, update the automation account
- Go to **Automation Account > Modules > Set the runtime version to 7.2**.
- Apply any necessary modules to automate tasks.

Q. Shared Resources and Modules in Azure Automation:?

- **Shared Resources:**
 - ◆ Reusable components like **scripts, credentials, variables**, etc.
 - ◆ Helps avoid duplication across automation tasks.(once defined, we can refer from there)
- **Modules:**
 - ◆ Instead of writing long scripts you can use cmdlets(commands)
 - ◆ **Modules** are collections of ready-made commands that help automate tasks in Azure.
 - ◆ Pre-built sets of **PowerShell cmdlets** to interact with Azure services.(creating vm)
 - ◆ Example: **AzureRM.Compute** module to manage VMs.

Q. What are all the other ways to achieve automation in Azure?

- Powershell
- Python
- Function as a service
- App functions(azure app services)

Q. How to create a runbook?

- Automation account - side - process automation -runbook -create
- Code done in class:

\$PSVersionTable

Ensures you do not inherit an AzContext in your runbook

Disable-AzContextAutosave -Scope Process

Connect to Azure with system-assigned managed identity

\$AzureContext = (Connect-AzAccount -Identity).context

Set and store context

\$AzureContext = Set-AzContext -SubscriptionName \$AzureContext.Subscription -DefaultProfile
\$AzureContext

- click testpane
- start
- Stopped -**not working**

Q. Function app in azure?

- A **Function App** is a container for running your **Azure Functions**.
- You can use this instead of azure automation
- Attached to storage account
- No limits
- No stop issues
- High scalability
- Azure Functions is a serverless compute service that allows you to execute small pieces of code (functions) in the cloud without managing infrastructure.
- These functions can be triggered by events, such as HTTP requests, timer-based schedules, or changes in Azure storage.
 - ◆ Function app - choose hosting plan(flex consumption) -create
 - ◆ create azure function using azure portal:
 - ◆ create resource -search function -app -create -pin to dashboard
 - ◆ function app -create new -choose webhook+api = httptrigger1

Q. azure functions:

Azure Functions is the code that you write and run in response to events or triggers.

- **Pay Only for What You Use:**
- **Integrated Security:**
 - ◆ **Azure Functions** can be secured with built-in authentication and authorization options.
 - ◆ You can use **HTTP triggers** to accept **GET**, **POST**, or other types of HTTP requests, ensuring that your functions can be securely accessed over the web.
- **Create Microservices and Simple APIs:**

- ◆ You can use **Azure Functions** to create **microservices** or **lightweight APIs** that can handle specific tasks, such as processing events or serving HTTP requests.
- **Run Azure Functions on Various Events:**
 - ◆ Time based triggers
 - ◆ Http req

TERRAFORM

laac - when u move from physical world to virtual stuff, there is possible to write code and create infrastructure

- **adv** : reusability
- same tym
- multi cloud
- consistency
- speed
- faster
- automation
- scalability
- version control
- Collaboration
- security
- Reusable code
- Enforce company standards:
 - ◆ Eg: vnet,roles
 - ◆ You should not open the code that u are not allowed to open
 - ◆ there will be a core team for this to enforce this standards
 - ◆ those team create vnet,subnet
 - ◆ they understand
 - Security,
 - Impact,
 - they only have access
 - ◆ they have terraform module ->used to **create vnet**
 - ◆ **Compliance** and all become easier if u do with code
 - The code acts as a clear definition of how resources should be provisioned and maintained.
 - ◆ Mistakes can be tracked
 - Since all changes are made through **code** and tracked in **version control** (e.g., Git), any mistakes can be identified and traced back to the person who made them.
 - ◆ we dont want anybody to directly to production and make changes
 - ◆ when u change -push to git- there is a man approves-once approves - transaction to give it to production
 - ◆ adv - **complete traceability**
 - ◆ **There are some tools like**
 - **Fluxcd**
 - **Argocd**
 - ◆ if u change also within 1 or 2 mins..it will overridden whateven in git
 - ◆ things are now version control

Q. iaac have 3 tools:

1. Infrastructure provisioning tools
2. Configuration management tools:
3. Server templating tools:

Infrastructure provisioning tools:

- Terraform – multicloud
- Boto in aws
- Cloud formation in aws
- Azure biceps
- Azure ARM template(json format)
- Adv: if its is supported in azure its possible
- maynot be 100 % for biceps and terraform
- It may **delete and recreate**, some times it **shutdown**
- In Azure -> delete and recreate-> because resizing isn't always supported dynamically
- No guarantee that Changes Will Just Modify because Cloud providers handle resources differently.(aws and azure), Provisioning tools follow cloud provider rules. so they sometimes destroy and recreate resources.
- Most of time when u provision it bring up a new instance

Configuration management tools:

- **Used for upgrading and modifying infrastructure configurations**
- Manual: people do in diff ways, cannot guarantee the consistency and when error occurs they fix it by their own ways-> delta grows
- At some point of time some code will not work on a particular machine. Need to fix without accessing that machine. Give one command and execute it (before they validate and execute). One by one doing without actually having access to production is not good idea. So I can use same script and create new machine. Here high chance of **configuration dripped/snow flakes**.
- **Ansible**
 - ◆ **Git went from 0 to 95 when devops comes**
 - ◆ install ansible on **centralized server** and connect to the production server.
 - ◆ Connection done using ssh ,internally uses scp(**Send configuration files to servers** using scp.),
 - ◆ **Ansible works on push pull model**: gather **the facts** and push(configuration changes always push based)
 - ◆ **No agent required on push model**
 - ◆ **Push**: Ansible collects facts from a central server and pushes configurations to target machines via SSH.,**No agent**
 - ◆ **Pull model**: Each **server (agent)** pulls its configuration from a central server and applies it.
 - **Agents should be installed**
 - **if agent is there -possibility of security attack**
 - **problem: push things to production server**
 - **in puppet and chef - unless u trigger it wont push**
 - ○ Works even if the central server is down temporarily.

- ○ Requires **agents** to be installed on every server.
- Fact collection is pull model(agent will come and pull and install)

Pull models:

- **Chef**
- **Puppet**
- **in puppet and chef - unless u trigger it wont push**

Server templating tools: Server templating tools help create pre-configured environments so you can quickly set up and manage servers without manual setup.

- ○ Vagrant (help to create vm)
- ○ Docker (create containers)

Q. What does compliance mean?

- **Compliance** means following the **rules and regulations** set by a country, region, or governing body, like
 - ◆ **USFDA** certification or
 - ◆ **HIPAA** for healthcare.

Q. Arm template advantage?

- Arm template - adv :if its is supported in azure its possible
- maynot be 100 % for biceps and terraform
- ARM templates fully support all Azure resources as soon as they are available, but Bicep and Terraform might not immediately support 100%.

Q. Practical - Bash.

- **Goal:** Installing azure CLI inside the vm
- **Create a vm:**

```
az vm create \

--resource-group abhi-resource-group \

--name abhi-backend-vm \

--image Ubuntu2204 \

--size Standard_B1s \

--admin-username abhiuser \

--generate-ssh-keys \
```

--vnet-name abhi-vnet \

--subnet abhi-backend-subnet \

--location centralindia

→ **connected using private ip:**

◆ ssh -i id_rsa abhiuser@4.224.248.48(inside.ssh folder thats y gave id_rsa)

◆ sudo su

→ Cd

→ apt update -y (all packages are up to date)

→ search the site : **install azure cli in ubuntu**

→ Site: Install the Azure CLI on Linux | Microsoft Learn

→ do 1 st command

→ then

→ **az login**

→ **install terraform:**

→ install terraform in azure ubuntu

◆ Linux

→ do all commands

→ create folder tf-test

→ cd tf-test

→ nano file(name.tf)

◆ write according to hashicrp configuration linux(hcl)

→ **nano.tf**

```
provider "azurerm" {  
  
    features{}  
  
}  
  
terraform{  
  
    required_providers{  
  
        azurerm={  
  
            source="hashicorp/azurerm"  
  
            version="2.40.0"  
  
        }  
  
    }  
  
}
```

```

resource "azurerm_resource_group" "abhi-rg1" {

    name="abhi-rg1"

    location="centralindia"

}

```

- Should define the **provider(cloud)**
- **ls -la** -after doing terraform init u see a folder ,file
- **terraform init** - u will see a file and a folder
 - ◆ .terraform - download all modules
- **Internal working:**
- **terraform apply** (imp command)
 - ◆ When u execute -work with **cloud provider**,
 - ◆ **how does it knw which provider has to work with**
 - (provider block)
- **is it possible to work with 2 provider(yes)**-generally we dont do
- **providers** means provision something
 - ◆ eg:mamaths,kaffehouse provide us food
- **terraform init** - understand code and dwld the modules associated with that provider
- **Terraform apply** - work with module and module work with actual destinations(cloud)

Q. some imp commands:

- **cd .terraform/tab/tab**
 - ◆ Where the latest provider plugins are downloaded
- **.terraform.lock.hcl file**
 - ◆ terraform is never be backward compatible
- because it has to work with cloud provider
- **Cloud providers keep updating** their services, so **Terraform must update too.**
- If you **already wrote some code** and Terraform updates modules, **your old code might break** (because of new changes).
- To **protect** you:
 - ◆ Terraform **creates a lock file** (**.terraform.lock.hcl**) to **freeze the versions** you are using.
 - ◆ Unless you **manually force an update**, Terraform **keeps using the old (locked) versions.**
 - ◆ Even if you **remove version** from your **.tf** code, Terraform **still knows** the version from the **lock file.**
 - ◆ Only if you **delete lock file** or **force an update**, it will pick new versions.

- ◆ **Lock file = Safety net** to prevent breaking your working Terraform code when providers change.
- When you do `cat .terraform.lock.hcl` (to view it):
- You will **see the version** number of the provider/module you are using
- The rest of the content is in **hashed format** (for security and integrity checking).
- Even if you **remove the provider version** from your `.tf` code file,
- Terraform will **still remember** the version **from the lock file**.
- **It won't automatically upgrade** unless you **delete or update** the lock file.

- **Terraform plan**
- Its like database execution plan
- whenever u r executing any query - do execution plan
- analyze the best way to execute - improve performance
- analyze code and do whatever need to go
 - ◆ modify -golden
 - ◆ remove -red
 - ◆ added -green
- Id will be know only after we apply the terraform
- so review this and apply
- After reviewing the plan → run `terraform apply` to **actually create/update/delete** resources.
- If you want to **save the plan** and **reuse it** without re-planning:
- output this as a file and pass this as parameter to apply
- You can **skip manual approval** by using:
 - ◆ `terraform apply -auto-approve`

Q. when all it compares with the 3 things(.tf file,cloud,.tf state file)?

- Plan
- Apply
- destroy

Q. why do we need to create a plan file?

- **Plan file = freeze the changes you reviewed, then safely apply them later.**
- **Terraform always compares three things** before applying:
 - **Your current code** (what you wrote in `.tf` files).
 - **Terraform state file** (`terraform.tfstate`) → (what Terraform believes already exists).
 - What we made changes will be in state
 - **Real cloud resources** → (what actually exists in Azure, AWS, etc.).

When you run **terraform plan**:

→ It checks all three and **shows the differences** (what will be created, changed, or destroyed).

- **Saving the plan to a file** (`terraform plan -out=planfile`) is helpful because:
 - It **locks** the exact changes you reviewed.
 - Later, you can **apply exactly that** without regenerating a new plan.
 - **Avoids surprises** if the cloud or code changes in between plan and apply.
 - when u **create a rg for new** ..this is compared with **cloud** - is it there or not
 - if u **apply** create a state file(terraform.tfstate)

Q. why do u want state file?

- Terraform can understand what u have done last time and compare with 3 things

Q. formatting the file and validate ?

- **formatting the file : terraform fmt**
- **validate: terraform validate**

Q. terraform destroy?

- destroy : terraform destroy (a backup file created internally)
- analyse and shows - i going to destroy this
- alternative : delete from main.tf file

Q.Sonarqube?

- ★ **Vulnerabilities:** SonarQube can detect security vulnerabilities in your code, such as common attacks (e.g., SQL injection).
- ★ **Code Smells:** Refers to bad code practices (not necessarily errors, but things that might cause problems later).
- ★ **Your Decision:** You can accept or fix these issues based on your code quality standards.

Q. Availability Sets in Azure:

- ★ **Fault Domain:** Group of VMs that share a **physical rack**. If a hardware failure occurs in that rack, the VMs in that fault domain are affected.
- ★ **Update Domain:** Group of VMs that are updated **simultaneously**. If there's an update to the system, VMs in the same update domain are updated at the same time to avoid downtime.

Q. Practical: goal: creating vm in terraform -doin

→ <https://github.com/vilasvarghese/terraform-tutorial/tree/master/azure>

- **dns** - within the address anyone try to talk with any domain name this will be checked by **dns server**
- **how do u find the default dns server?**
 - ◆ etc/resolv.conf file - name server configured
- **resource block** -the real infrastructure u want to create
 - ◆ It specifies the type of resource, its parameters, and the name used to reference it within the configuration.
- A typical resource block has **3 key components**:
 - ◆ Type of Resource(vm,vnet)
 - ◆ Name(**unique name** given to the resource)
 - ◆ **Parameters**:(specific **settings or attributes** that define the resource)
 - It can involve size,os to vm
- in Azure (and most cloud platforms), **resource names** need to be unique within their respective scope.

- create a directory
- get inside
- create a file

Code:

```
provider "azurerm" {

  features {}

}
```

```
terraform {

  required_providers {

    azurerm = {

      source = "hashicorp/azurerm"

      version = "2.40.0"

    }

  }

}

resource "azurerm_virtual_network" "vnet" {
```

```

name    = "abhi-vnet-1"

location = "centralindia"

resource_group_name = "abhi-resource-group"

address_space    = ["10.0.0.0/16"]
}

```

- can see on vs code
- after code ..init and apply
- trying to create resource -network resource(imp)
- destroy - commenting the vnet part and apply
- or destroy

30/4/25

Q. init container in kubernetes?

- Runs **before** the main container(s).
- If any init container **fails**, the pod restarts until it succeeds.
- **Main container starts only after all init containers complete.**
- when the pod got crashed/killed its sandbox changed and kubelet will watch the pods
- Kubelet will fix the containers

Pods creation:

Scheduler - event driven architecture, Assign the node(based on data in etcd=memory,cpu), find best machine to schedule

Controller- desired state != actual, api server forward req to kubelet(master to worker(direct=they know which machine to talk), worker to master(load balancer)->kubelet work with container run time, every few sec worker node is sending info about cpu, memory to master node through load balancer, kubelet changes it to running and controller is watching.

Who helps in load balancing? What do you have to create?=kubernetes services

Replica set = you can horizontally scale

What does deployment help us to achieve?=rolling update(db manipulation)

Does Kubernetes support 2 updates? Complete replacement and rolling update(whenever u update, one more entry of replica set created and previous one deleted)

Q. creating vm in terraform.

- Written code inside a folder tf-test
- Check out the documentation
- [azurerm_linux_virtual_machine | Resources | hashicorp/azurerm | Terraform | Terraform Registry](#)
- Search virtual machine(azurerm_linux_virtual_machine)
- Copy paste the code and edit
- Then plan and apply
- Search publicip
- Public ip is a separate service which u can configure as -static and dynamic(ensure ip is private or public -doubt): In both private and public we can do that
- You need to create and attach it(vm,bastian host)
- When machine goes down and comes back -same ip - associate with elastic ip
- In azure possible to change the type of ip
- Vm-network interface-ur nic -settings-ip configuration-internal-click-edit ip conf-associate public ip(create new public ip)
- Create the nsg,settings,inbound rules,icmp,allow,100,add
- So can able to ping
- Also allow ssh
- Try to connect using public ip
- Ssh -i id_rsa azureuser@publicip
- Path need to converted into reference of the file

Q. policy of code?

- Can create policies in terraform which dont allow us to do certain things.
- Can see logs who did the things(monitor part)
- Lock the rg
- U need to do code acc to policies

Q. adv of ci/cd in terraform?

- Continuous,automated
- Version control
- Improves security
- Only problem of terraform : need to track the code

Q. file -Terraform advanced in [github.com/vilasvarghese/terraform/Terraformadvanced.txt](#).

Q. require_provider:Minor version,Major version and patch?

- ~> = minor version that can be upgraded
- > inside “ ”
- 2.0 means it will go to latest version of 2
- 2.40.0 = major version,minor version and patch
- Patch =fixing bugs
 - ◆ Schedule for releasing patch
- Minor versions= within the generics

- ◆ they improves performance,
- ◆ adding to existing significant changes
- Major versions= java version
 - ◆ (1.7 =they added generics,1.8=lambda (pass around functions),
 - ◆ so these are **big changes**.
 - ◆ Introduced a lot of features and improved speed.
- Hot fixes - within 24 hrs
 - ◆ Immediately release
- Provider = cloud provider in that code
- [terraform-tutorial/azure/01_1ResourceGroup/main.tf at master · vilasvarghese/terraform-tutorial](https://github.com/terraform-tutorial/terraform-tutorial/blob/master/01_1ResourceGroup/main.tf)
- Terraform =work with modules
- In aws= no terraform block,azure=terraform block

Q. problem with managing state file?

- .tf file = monitoring the changes
- Whenever u change = in ur local machine,friend does not have that .tf file,in local machine there is no .tf file,u check in and that person access...but when 2 person need to access simultaneously,what will we do?
- Configure the Backend = on the cloud(storage account)
- State file maintained in cloud
- Using cloud feature u can put lock.(when 2 persons doing until a lock is released other person blocked)
- When i apply the .tf file is locked for me
- I can do the changes that other person cant access until i finish

Practical:

```
terraform {

  required_providers {

    azurerm = {

      source  = "hashicorp/azurerm"

      version = "~> 2.0"

    }

  }

  required_version=">=1.0"

}
```

Version is 2.0 heree,~> upgrades the minor version to latest version, 2=major,0=minor,no patch applied

Required version is terraform version= checks is it >=1.0 worked,otherwise <=1.0 error

Goal: Code it to get an existing rg and create vnet

→ Search resource group, data sources

▼ Data Sources

azurerm_resource_group


```
data "azurerm_resource_group" "example" {

  name = "abhi-resource-group"

}

output "id" {

  value = data.azurerm_resource_group.example.id

}
```

The reason y we are giving data = "we use existing data dont want to create new"

- Do terraform apply
- Add vnet into it
- Search virtual network
- Destroy = clean state file

Diff b/w resource and data?

- Data - try to query not modify
- Resource - create
- To use existing vm created by someone and need to manage in terraform. Bring to .tf file and updated in state file but in main.tf file not updated(manually add)
- If not manually add,the contents only in state file and terraform compare with .tf and state file and the contents not present in .tf file
- Terraform thinks u created previously now u want delete so while apply it will delete the things in state file and do the things in .tf file
- To bring cloud into terraform .tf file = **terraform import**

Full code:

```
provider "azurerm" {
```

```
features {}

}

terraform {

  required_providers {

    azurearm = {

      source = "hashicorp/azurearm"

      version = "~> 3.0"

    }

  }

  required_version=">=1.0"

}

data "azurearm_resource_group" "example" {
```

```

name = "abhi-resource-group"

}

output "id" {

value = data.azure_rm_resource_group.example.id

}

```

Q. create 2 diff vm.terraform understands what need to execute serial and parallel(execution)

→ code:

```

provider "azurerm" {

  features {}

}

resource "azurerm_resource_group" "example" {

  name      = "abhi-resource-group1"

  location = "Centralindia"

}

resource "azurerm_virtual_network" "example" {

```

```
name                = "abhi-vnet-2"

address_space       = ["10.0.0.0/16"]

location            = azurerm_resource_group.example.location

resource_group_name = azurerm_resource_group.example.name
}
```

```
resource "azurerm_subnet" "example" {

  name                = "abhi-subnet"

  resource_group_name = azurerm_resource_group.example.name

  virtual_network_name = azurerm_virtual_network.example.name

  address_prefixes     = ["10.0.2.0/24"]
}
```

```
resource "azurerm_network_interface" "nic1" {

  name                = "abhi-nic1"

  location            = azurerm_resource_group.example.location

  resource_group_name = azurerm_resource_group.example.name

  ip_configuration {

    name                = "internal"

    subnet_id           = azurerm_subnet.example.id

    private_ip_address_allocation = "Dynamic"

  }
}
```

```
}

resource "azurerm_network_interface" "nic2" {

  name                = "abhi-nic2"

  location            = azurerm_resource_group.example.location

  resource_group_name = azurerm_resource_group.example.name

  ip_configuration {

    name                = "internal"

    subnet_id           = azurerm_subnet.example.id

    private_ip_address_allocation = "Dynamic"

  }

}

#define vm

resource "azurerm_linux_virtual_machine" "vm1" {

  name                = "abhi-virtualmachine1"

  resource_group_name = azurerm_resource_group.example.name

  location            = azurerm_resource_group.example.location

  size                = "Standard_F2"

  admin_username      = "adminuser"

  network_interface_ids = [

    azurerm_network_interface.nic1.id,

  ]

}
```

```
admin_ssh_key {  
  
    username    = "adminuser"  
  
    public_key = file("~/ssh/id_rsa.pub")  
  
}  
  
os_disk {  
  
    caching          = "ReadWrite"  
  
    storage_account_type = "Standard_LRS"  
  
}  
  
source_image_reference {  
  
    publisher = "Canonical"  
  
    offer      = "0001-com-ubuntu-server-jammy"  
  
    sku        = "22_04-lts"  
  
    version    = "latest"  
  
}  
}  
  
#define vm2  
  
resource "azurerm_linux_virtual_machine" "vm2" {  
  
    name          = "abhi-virtualmachine2"  
  
    resource_group_name = azurerm_resource_group.example.name
```

```
location          = azurerm_resource_group.example.location

size              = "Standard_F2"

admin_username    = "adminuser"

network_interface_ids = [
    azurerm_network_interface.nic2.id,
]

admin_ssh_key {
    username  = "adminuser"

    public_key = file("~/ssh/id_rsa.pub")
}

os_disk {
    caching          = "ReadWrite"

    storage_account_type = "Standard_LRS"
}

source_image_reference {
    publisher = "Canonical"

    offer      = "0001-com-ubuntu-server-jammy"

    sku        = "22_04-lts"

    version    = "latest"
}
```



```
}

# # # valid for terraform version >= 0.14

terraform {

  required_providers {

    azurerm = {

      source  = "hashicorp/azurerm"

      version = "~> 2.0"

    }

  }

  required_version=">=1.0"

}

data "azurerm_resource_group" "example" {

  name = "abhi-resource-group"

}

output "id" {

  value = data.azurerm_resource_group.example.id

}
```

Destroy it

Errors got while executing code:

- Resource name should be unique(while creating vm "vnet1" "vnet2")
- Resource is using same nic(so created diff nic for diff vm)
- Should give diff resource group if using resource block otherwise use datablock and edit everywhere by data.
- In my code vm's are not dependent to eachother so create parallelly
- Unless u give dependency,it will create parallelly and it may fail(if rg is not created still and other things execute)
- Dependency means=rg.example.name
 - ◆ Like referring
 - ◆ Vnet belongs to rg=rg.name
 - ◆ Subnet belongs to vnet=vnet.name
- Terraform doesnt knw vnet is belongs to that rg

Explicitly give dependency:

- Give field like depends on
- Front end is depend on backend
- Backend come first
- Then front end
 - ◆ Field:depends_on
 - ◆ This resource depend on vm1
- This will create backend then frontend

2-5-25

Q. Multiple vm's creation in terraform?

Q. storage account?

- **Blobs**- Binary large Objects(managed by containers)- to store large files
 - ◆ Binary storage is block storage,possible some block is underutilized,access-fast
- **File share**
- **Tables(NOSQL Date)**- used for storing huge data
 - ◆ **Partition key**-you have multiple machines,based on which key ur data is partitioned,
 - stored on diff partitions or nodes
 - ◆ Row key - Within the same machine
 - ◆ If u r searching is based on partition key and row key -> very fast
 - ◆ Because the data is index spaced on these 2
 - ◆ Index: like an index of the book
 - ◆ Which tells u where u can find the data'
 - ◆ Data stored as pages in db

- ◆ Data stored based on rows in pages
- ◆ When u r querying the db go to index
- ◆ Very fast
- ◆ **Wide Column based db?**
 - Store data based on columns and not rows
 - If we only want to retrieve the things of 2 specific column(phn number,aadhaar number)
 - So we can store many phone number and aadhar in 1 page
 - So time reduced
 - Read time reduced

→ **Queues:**

- Here synchronous and asynchronous communication comes
- In synchronous- load balancer - helps to route ur req to one of the instance
 - ◆ Sent a msg
 - ◆ Wait for response
- Asynchronous - no load balancer
 - ◆ Always a broker in it
 - ◆ broker(kafka,rabbit and queue)
 - ◆ Senter is not waiting
- 2 types of asynchronous - queue mode and publish subscribe model
- Queue -client will come and pick the req
 - ◆ No need a load balancer
 - ◆ One of the instance of client pick the msg
- Publish subscribe-
- Broker will notify immediately
- All services can intimate using broker so no load balancer needed
- **Microservices:**
 - ◆ Load balancer -2 types
 - Path based routing
 - Distribute load
 - ◆ Authentication(valid user/not) and authorization(can you do the thing/not)
 - ◆ How do u manage authorization-RBAC
 - ◆ When u create user
 - ◆ Front end application - data maintained in cache -y - scalability -cant scale-externalize the data(redis,memcache)

Q. Practical : create a storage account,create a container inside it,create a directory and upload a file and access on browser.

- Local redundant storage - to reduce the cost
- Storage acc - accessed by - keys and entra id

private endpoints

Network access *

- ☒ Enable public access from all networks
- ☐ Enable public access from selected virtual networks and IP addresses
- ☐ Disable public access and use private access

- Created storage account
- Create containers
- Search storage acc
- Click what u created
- Side ->data storage ->containers>create
- Side ->storage browser->blob containers -.click ur container->add directory
- Upload a file
- download
- Open the file
- Copy the link and paste on another tab
- No access
- Side ->security+networking ->access keys - wont work
- Shared access signature->give ip address (search myipaddress)
- Go back to containers ->ur file ->generate sas token and url->and search with that url

Q. do the same thing using terraform?

- Search storage blob(azurerm_storage_blob)
- Copy the code and edit and do init and apply
- If smthg happens do rm -rf .terraform

```
terraform {  
  
  required_providers {  
  
    azurerm = {  
  
      source  = "hashicorp/azurerm"  
  
      version = "~> 3.0"  
  
    }  
  
  }  
  
  required_version=">=1.0"  
  
}
```

```

provider "azurerm" {

  features {}

}

resource "azurerm_resource_group" "example" {

  name      = "abhi-rg"

  location = "centralindia"

}

resource "azurerm_storage_account" "example" {

  name                                = "abhistore2"

  resource_group_name                 = azurerm_resource_group.example.name

  location                            = azurerm_resource_group.example.location

  account_tier                        = "Standard"

  account_replication_type            = "LRS"

}

resource "azurerm_storage_container" "example" {

  name                                = "abhi-container1"

  storage_account_name                = azurerm_storage_account.example.name

  container_access_type               = "private"

}

```

- Uploaded a file from azure
- Generate SAS and access using url in azure

- Go to storage acc -> side file share->create
- Connect -linux -showscript -copy
- Go to cloudshell
- Create a vm
- Create vm with public ip
- Also associate ip with vm
- Search publicip in terraform registry and copy the required code
- Added these 2 things on ur code for creating publicip

```
resource "azurerm_public_ip" "example" {

  name                = "abhi-public-ip"

  location            = azurerm_resource_group.example.location

  resource_group_name = azurerm_resource_group.example.name

  allocation_method   = "Dynamic"

}

resource "azurerm_network_interface" "nic1" {

  name                = "abhi-nic1"

  location            = azurerm_resource_group.example.location

  resource_group_name = azurerm_resource_group.example.name

  ip_configuration {

    name                = "internal"

    subnet_id           = azurerm_subnet.example.id

    private_ip_address_allocation = "Dynamic"

    public_ip_address_id =
azurerm_public_ip.example.id

  }

}
```

- Connect the vm using public ip
- `ssh -i .ssh/id_rsa adminuser@4.213.120.28`
- Copy paste the linux script copied file.sh
- Created folder is getting mounted to that storage..when i execute the script
- And uploaded a file from azure in fileshare
- And when i cat the script it have a folder
- When i enter into that folder i can see the file i uploaded from azure.

Q. scale something in azure which is deployed - how will u scale it -**scaleset**,to route req-load balancer ,backend of scaleset -load balancer -deploy ths in public subnet then in priv subnet,create an vm,image and install apache

Deploy it as microservice(this will then talk to private load balancer) distribute traffic to

1.create an image

Capture option

2.my images,publish images

3.Create a new machine out of that

4. Confirm apache is there
5. Create vnet and public and priv subnet
6. Create vm scale set - option attach load b - pubic ip -routing traffic to vm scalest -backend configured as vm scale set
7. Attach load balancer attach backend pol
- 8.

Ans:

- Created a vm
- Connected to that vm using public ip
- Got an error
- Pem file permission only given as read
- `Chmod 400 pemfile`
- Connected to vm
- Installed apache
- `Curl localhost:80` =worked
- Change inbound rule : 80 and any
- Search publicip:80 in google
- Go to vm ,capture,create image
- Create scaleset ->search vmss->create
- Vmss -use private subnet -in nic also same subnet
- loadbalancer=azure load balancer -create-public
- Create vmss =nat rules are attached when u create load balancer from vmss
- Use the frontend ip and search on browser [**scaleset-networking-loadbalancing-frontend ip address**]

5/5/2025

Q. do the above thing using terraform.

```
provider "azurerm" {

  features {}

}

terraform {

  required_providers {

    azurerm = {

      source  = "hashicorp/azurerm"

      version = "~> 3.0.0"

    }

  }

}

resource "azurerm_resource_group" "rg" {

  name      = "abhi-vmss-rg"

  location = "Central India "

}

# VNET and Subnets

resource "azurerm_virtual_network" "vnet" {

  name            = "abhi-vmss-vnet"

  address_space   = ["10.0.0.0/16"]

  location        = azurerm_resource_group.rg.location

  resource_group_name = azurerm_resource_group.rg.name

}
```



```
}

resource "azurerm_subnet" "public" {

  name                = "abhi-pub-subnet"

  resource_group_name = azurerm_resource_group.rg.name

  virtual_network_name = azurerm_virtual_network.vnet.name

  address_prefixes     = ["10.0.1.0/24"]

}

resource "azurerm_subnet" "private" {

  name                = "abhi-private-subnet"

  resource_group_name = azurerm_resource_group.rg.name

  virtual_network_name = azurerm_virtual_network.vnet.name

  address_prefixes     = ["10.0.2.0/24"]

}

# Public IP for Load Balancer

resource "azurerm_public_ip" "lb_ip" {

  name                = "abhi-lb-ip"

  location            = azurerm_resource_group.rg.location

  resource_group_name = azurerm_resource_group.rg.name

  allocation_method   = "Static"

  sku                  = "Standard"

}

# Load Balancer

resource "azurerm_lb" "lb" {
```

```
name          = "abhi-lb"

location      = azurerm_resource_group.rg.location

resource_group_name = azurerm_resource_group.rg.name

sku           = "Standard"

frontend_ip_configuration {

    name          = "abhiLoadBalancer"

    public_ip_address_id      = azurerm_public_ip.lb_ip.id

    private_ip_address_allocation = "Dynamic"

}

}

# Backend Pool

resource "azurerm_lb_backend_address_pool" "backend_pool" {

    name          = "abhi-backend-pool"

    loadbalancer_id = azurerm_lb.lb.id

}

# Health Probe

resource "azurerm_lb_probe" "http_probe" {

    name          = "abhi-http-probe"

    loadbalancer_id = azurerm_lb.lb.id

    protocol      = "Tcp"

    port          = 80

    interval_in_seconds = 5

    # number_of_probes      = 2

}
```

```

}

# Load Balancer Rule

resource "azurerm_lb_rule" "http" {

  name                        = "abhi-http-rule"

  loadbalancer_id            = azurerm_lb.lb.id

  frontend_ip_configuration_name = "abhiLoadBalancer"

  backend_address_pool_ids    =
[azurerm_lb_backend_address_pool.backend_pool.id]

  protocol                    = "Tcp"

  frontend_port               = 80

  backend_port                = 80

  probe_id                   = azurerm_lb_probe.http_probe.id
}

resource "azurerm_network_security_group" "web_nsg" {

  name                        = "abhi-nsg"

  location                   = resource.azurerm_resource_group.rg.location

  resource_group_name        = resource.azurerm_resource_group.rg.name

  security_rule {

    name                      = "AllowHTTP"

    priority                  = 100

    direction                 = "Inbound"

    access                    = "Allow"

    protocol                  = "Tcp"

    source_port_range         = "*"
  }
}

```

```

        destination_port_range      = "80"

        source_address_prefix      = "*"

        destination_address_prefix = "*"
    }
}

resource "azurerm_subnet_network_security_group_association" "nsg_assoc" {

    subnet_id          = azurerm_subnet.private.id

    network_security_group_id = azurerm_network_security_group.web_nsg.id
}

# Image Version (adjust as per your gallery setup)

data "azurerm_shared_image_version" "abhi-image" {

    name          = "0.0.1"

    image_name     = "abhi-image"

    gallery_name   = "abhi_images"

    resource_group_name = "abhi-resource-group"
}

# VMSS

resource "azurerm_linux_virtual_machine_scale_set" "vmss" {

    name          = "abhi-apache-vmss"

    location      = azurerm_resource_group.rg.location

    resource_group_name = azurerm_resource_group.rg.name

    sku           = "Standard_B1s"

    instances     = 2

```

```
admin_username      = "azureuser"

source_image_id     = data.azurearm_shared_image_version.abhi-image.id

  secure_boot_enabled = true
os_disk {

  caching          = "ReadWrite"

  storage_account_type = "Standard_LRS"
}

admin_ssh_key {

  username  = "azureuser"

  public_key = file("~/ssh/id_rsa.pub")
}

upgrade_mode = "Manual"

network_interface {

  name      = "abhi-vmss-nic"

  primary = true

  ip_configuration {

    name      = "internal"

    subnet_id = azurerm_subnet.private.id

    load_balancer_backend_address_pool_ids = [

      azurerm_lb_backend_address_pool.backend_pool.id

    ]

  }

}
```

```
health_probe_id = azurerm_lb_probe.http_probe.id
}
```

Q. what are the steps i done here?

- creating a Virtual Machine Scale Set (VMSS) with a load balancer, health probe, and network security group in Azure.
- Creates two subnets within the VNet:
 - ◆ A **public subnet** (10.0.1.0/24)
 - ◆ A **private subnet** (10.0.2.0/24), where the VMSS will be placed.
- Created a **public ip for load balancer**: Creates a public IP resource (abhi-lb-ip) with a static allocation method. This will be **used by the load balancer to provide public access**.
- **Creates a load balancer** named abhi-lb in the same resource group and location. The **load balancer has a frontend configuration** that binds to the public IP (abhi-lb-ip).
- **Backend Pool**: Defines a backend address pool (abhi-backend-pool) for the load balancer, which will hold the virtual machines (VMs) that the load balancer will distribute traffic to.
- **Health Probe**: Defines a TCP health probe (abhi-http-probe) for checking the health of the backend VMs. The probe checks port 80 (HTTP) every 5 seconds.
- **Load Balancer Rule**: Defines a rule (abhi-http-rule) that binds the frontend IP (port 80) to the backend pool's VMs. It uses the health probe to ensure traffic is only sent to healthy VMs.
- **Network Security Group (NSG)**: Defines an NSG (abhi-nsg) that allows inbound TCP traffic on port 80 (HTTP). This NSG is associated with the private subnet.
- **NSG Association**: Associates the abhi-nsg NSG with the private subnet, ensuring the security rules are applied to the subnet's traffic.
- **VMSS (Virtual Machine Scale Set)**: Creates a VM scale set (abhi-apache-vmss) with two instances of a Linux VM using an image from a shared image gallery. The VMs will be deployed in the private subnet (abhi-private-subnet).
 - ◆ **Load Balancer Integration**: Each VM in the scale set is associated with the backend pool of the load balancer (abhi-backend-pool).
 - ◆ **Health Probe**: The health probe checks the status of the VMs and ensures that traffic is only routed to healthy VMs.

Q. What the Backend Pool connects to:


- The **backend pool** is connected to the **Load Balancer**.

- Inside the backend pool, you define which resources (in your case, **VMSS**) will receive the incoming traffic.
- The load balancer uses this backend pool to forward traffic to the VMSS instances.

Q. vmss connects to what?

- The **VMSS (Virtual Machine Scale Set)** connects its VM instances to the **backend pool** of the **Load Balancer**.
- So yes, **VMSS is connected to the backend pool**, and that's how traffic reaches the Apache servers.

Q. Azure Storage Account.

- ★ When you create a **Storage Account** (`azurerm_storage_account`), it supports different services:
 - Blob storage
 - Queue storage
 - Table storage
 - **File shares**  ← this is what we use for Azure File Share.

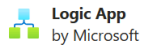
Q. create a table and a queue and create a logical app to talk to it.

- Storage acc - data storage - table
- Created : cant access the table using the URL
- Storage browser - table -add entity (partition key(id),Rowkey(name))
- No sql is schemaless =diff type of data coming,
- Now we need to add data = go to **logic apps - add**
- Multitenant

Create Logic App (Multi-tenant) ...

Basics Tags Review + create

Summary



Details

Subscription	bf7e75db-e819-49ca-b6d2-69c32a2353fe
Resource Group	abhi-rg
Name	abhi-logicapp
Region	centralindia
Log analytics	Disabled
Tags	abhi: 1

- **Logical app** - logic app designer - add trigger (right side) - click schedule -click recurrence - (freq-day),
- Add an action - search table -see more -insert or merge entity -> add connection
- Give access key (storage acc - security - access key)copy and paste
- Add partition key and json format

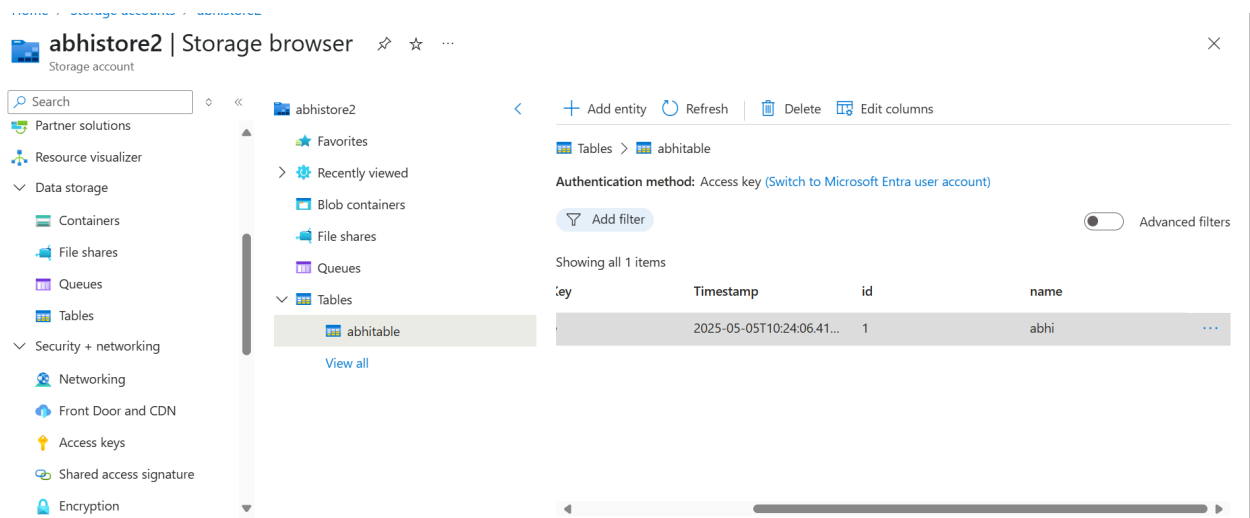
Run Save Discard Parameters Code view Errors Info File a bug

The screenshot shows the Logic App Designer interface. On the left, a workflow diagram is visible with a 'Recurrence' trigger connected to an 'Insert or Merge Entity (V2)' action. The right-hand pane is titled 'Insert or Merge Entity (V2)' and contains the following configuration:

- Parameters:** name
- Storage Account Name Or Table Endpoint *:** Use connection settings(abhistore2)
- Table *:** abhitable
- Entity *:**

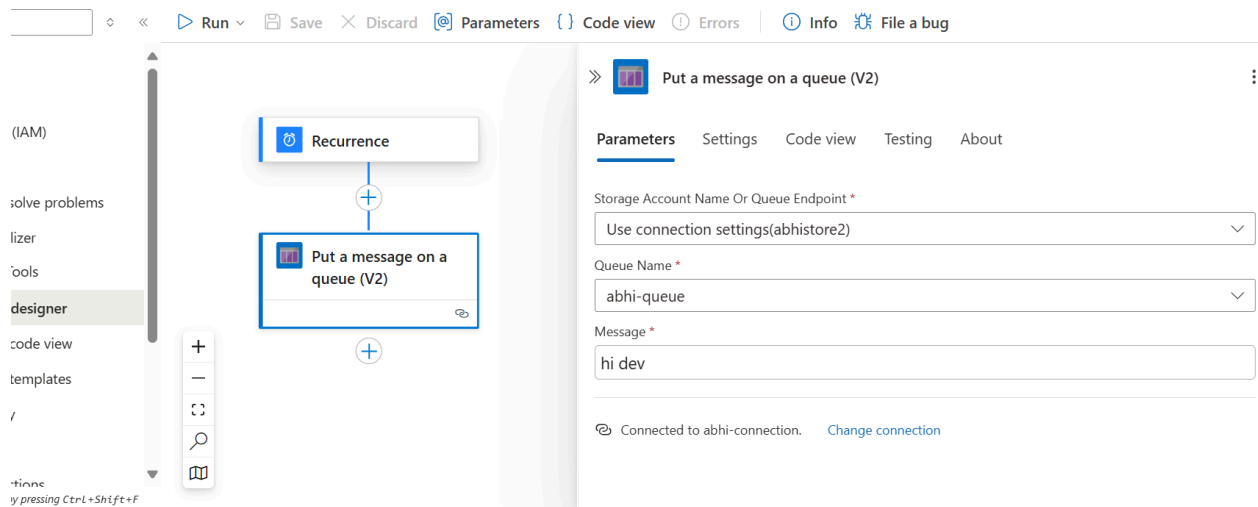
```
{
  "id": "1",
  "name": "abhi"
}
```

- Save and run
- Come to storage acc- storage browser -tables -abhitable -u can see the data u have given



Queue:

- Created queue and add msg
- Create logic app
- Logic app scheduler -add recurrence
- Add actions -queue
- Put a msg
- And type, save and run



ibhstore2 | Storage browser

✂ ☆ ...

Storage account

ch

Diagnose and solve problems

Access Control (IAM)

Account migration

Containers

Storage browser

Storage Mover

Network solutions

Source visualizer

Storage storage

Containers

abhistore2

Favorites

Recently viewed

Blob containers

File shares

Queues

abhi-queue

View all

Tables

+

 Add message

🗑

 Dequeue message

✕

 Clear queue

🔄

 Refresh

🔧

 Edit columns

Queues > abhi-queue

Authentication method: Access key [\(Switch to Microsoft Entra user account\)](#)

Showing all 2 items

Message text	Id	Insertion time	Expiration time
hi	3c876752-61c5-4e63-9a...	5/5/2025, 5:01:27 pm	12/5/2025, 5:01:27 pm
hi dev	5bfc39bd-24d0-40b1-b...	5/5/2025, 5:09:42 pm	12/5/2025, 5:09:42 pm

Q. meta-agrs?

- In **Terraform**, **meta-arguments** (also called **meta-args**) are **special arguments** that you can apply to **any resource or module**. They're not specific to Azure or AWS — they're built into Terraform itself.
- Within the resource block many parameters u can pass
- **Depends on** (meaning same across all providers)
- **count**

```

terraform {

  required_providers {

    azurerm = {

      source = "hashicorp/azurerm"

      version = "~> 3.0"

    }

  }

  required_version=">=1.0"

}

provider "azurerm" {

```

```

features {}

}

resource "azure_rm_resource_group" "rg" {

  name      = "abhi-rg1${count.index+1}"

  location = "centralindia"

  count=2

}

```

Q. for-each is a meta -args?

- **for_each** is a **meta-argument** that lets you **create multiple instances of a resource** using a **map** or **set** of values.
- "Make this thing for each item in my **list or map** — and I want to control the **name** or **location** using those items."
- Use **count** when you just need **many copies**.
- Use **for_each** when you want to use a **specific name or value** for each one.
- You have a **map** or **unique strings**
- You want to use **each.key** and **each.value**

Q.practical -tenant?

- Search entra id - manage tenants - create
- Created user
- Invited a user
- Created group
- Add members to that group

Q. practical - terraform - variables

- mkdir variables
- cd variables
- mkdir 01
- People can modify the variables and execute
- touch main.tf

→ **Terraform 3 types of variables**

- ◆ Variables,
- ◆ locals(scope to that file),
- ◆ output

→ Search variables in terraform

→ main.tf

```
terraform {  
  
  required_providers {  
  
    azurerm = {  
  
      source  = "hashicorp/azurerm"  
  
      version = "~> 3.0"  
  
    }  
  
  }  
  
  required_version=">=1.0"  
  
}  
  
provider "azurerm" {  
  
  features {}  
  
}  
  
resource "azurerm_resource_group" "rg" {  
  
  name      = var.name  
  
  #name      = "myrg${var.name}"  
  
  
  location = "centralus"  
  
}
```

→ variable.tf

```
variable "name"{

    type =string

    #default ="abhi-rg1"

    description="enter resource name"

}
```

→ touch terraform.tfvars = maintainability

name="abhi-tfvars"

→ Replaces the existing rg

→ Create an output file: **add in main.tf**

```
◆ output "name"{
◆     value=azurerm_resource_group.rg.name
◆ }
```

Output:

```
Terraform will perform the following actions:

# azurerm_resource_group.rg must be replaced
-/+ resource "azurerm_resource_group" "rg" {
  ~ id          = "/subscriptions/bf7e75db-e819-49ca-b6d2-69c32a2353fe/resourceGroups/myrgabhi-rg3" -> (known after apply)
  ~ name        = "myrgabhi-rg3" -> "myrgabhi-tfvars" # forces replacement
  - tags        = {} -> null
    # (2 unchanged attributes hidden)
}
```

Afternoon practicals:

- Mkdir 02
- Cd 02
- Touch main.tf
- Touch variables.tf
- Created a variables.tf file and gave the reference in main.tf file and created a resource group if it starts with ust-

main.tf:

```
terraform {
```

```
required_providers {

  azurerm = {

    source = "hashicorp/azurerm"

    version = "~> 3.0"

  }

}

required_version=">=1.0"

}

provider "azurerm" {

  features {}

}

resource "azurerm_resource_group" "rg" {

  name      = "${var.rgname}myrg${var.number_type}"

  location = "centralus"

  tags=var.tags

}

output "bool_type"{

  value=var.boolean_type

}

output "list_type"{

  value=var.list_type

}
```

```
output "map"{  
  
    value=var.map_type  
  
}  
  
output "object"{  
  
    value=var.object_type  
  
}  
  
output "tuple_type"{  
  
    value=var.tuple_type  
  
}  
  
output "set_example"{  
  
    value=var.set_example  
  
}  
  
output "map_of_objects"{  
  
    value=var.map_of_objects  
  
}  
  
output "list_of_objects"{  
  
    value=var.list_of_objects  
  
}
```

variables.tf:

```
variable "number_type" {  
  
description = "This is a variable of type number"
```

```
type          = number

default      = 42

}

variable "boolean_type" {

description = "This is a variable of type bool"

type        = bool

default     = true

}

variable "list_type" {

description = "This is a variable of type list"

type        = list(string)

default     = ["string1", "string2", "string3"]

}

variable "map_type" {

description = "This is a variable of type map"

type        = map(string)

default     = {

    key1 = "value1"

    key2 = "value2"

}

}
```



```
}

variable "object_type" {

description = "This is a variable of type object"

type      = object({

    name      = string

    age       = number

    enabled   = bool

}))

default = {

    name      = "John Doe"

    age       = 30

    enabled   = true

}

}

variable "tuple_type" {

description = "This is a variable of type tuple"

type      = tuple([string, number, bool])

default    = ["item1", 42, true]

}

variable "set_example" {
```

```
description = "This is a variable of type set"

type        = set(string)

default     = ["item1", "item2", "item3"]
}

variable "map_of_objects" {

    description = "This is a variable of type Map of objects"

    type = map(object({

        name = string,

        cidr = string

    })))

    default = {

        "subnet_a" = {

            name = "Subnet A",

            cidr = "10.10.1.0/24"

        },

        "subnet_b" = {

            name = "Subnet B",

            cidr = "10.10.2.0/24"

        },

        "subnet_c" = {

            name = "Subnet C",

            cidr = "10.10.3.0/24"
```

```
}  
  
}  
  
}  
  
variable "list_of_objects" {  
  
    description = "This is a variable of type List of objects"  
  
    type = list(object({  
  
        name = string,  
  
        cidr = string  
  
    }))  
  
    default = [  
  
        {  
  
            name = "Subnet A",  
  
            cidr = "10.10.1.0/24"  
  
        },  
  
        {  
  
            name = "Subnet B",  
  
            cidr = "10.10.2.0/24"  
  
        },  
  
        {  
  
            name = "Subnet C",  
  
            cidr = "10.10.3.0/24"  
  
        }  
    ]  
}
```

```

]

}

variable "rgname" {
  type          = string

  description = "should start with ust-"

  #default      = "ami-0d26eb3972b7f8c96"

  validation {
    condition     = length(var.rgname) > 4 && substr(var.rgname, 0, 4) ==
"ust-"

    error_message = "Please provide a valid value for resourcegroup name."
  }
}

variable "tags" {
  type = object({
    name = string
    env  = string
  })

  description = "Tags for the azure vm instance"

  default = {
    name = "My Virtual Machine"
    env  = "Dev"
  }
}

```

Q. backend in terraform is?

→ State file

Q. practical: created vm using local in terraform?

- Locals: scope is local
- Create it and access from other file but cant modify
- Can create and access within the same module, but can't modify.
- Cant pass variables
- Mkdir variables
- Cd variables
- Mkdir 03
- Cd 03
- Touch main.tf
- Touch vm.tf

main.tf:

```
terraform {  
  
  required_providers {  
  
    azurerm = {  
  
      source  = "hashicorp/azurerm"  
  
      version = "~> 3.0"  
  
    }  
  
  }  
  
  required_version=">=1.0"  
}  
  
provider "azurerm" {  
  
  features {}  
  
}
```

vm.tf:

```
locals {  
  
    vm_size = "Standard_B1s"  
  
    tags = {  
  
        Name = "My Virtual Machine"  
  
        Env  = "Dev"  
  
    }  
}  
  
resource "azurerm_resource_group" "main" {  
  
    name      = "abhi-rg4"  
  
    location = "centralindia"  
}  
  
resource "azurerm_virtual_network" "vnet" {  
  
    name                = "abhi-Vnet"  
  
    address_space       = ["10.0.0.0/16"]  
  
    location             = azurerm_resource_group.main.location  
    resource_group_name = azurerm_resource_group.main.name  
}  
  
resource "azurerm_subnet" "subnet" {
```

```
name                = "abhi-Subnet"

resource_group_name = azurerm_resource_group.main.name

virtual_network_name = azurerm_virtual_network.vnet.name

address_prefixes     = ["10.0.1.0/24"]
}

resource "azurerm_network_interface" "nic" {

  name                = "abhi-NIC"

  location            = azurerm_resource_group.main.location

  resource_group_name = azurerm_resource_group.main.name

  ip_configuration {

    name                = "internal"

    subnet_id           = azurerm_subnet.subnet.id

    private_ip_address_allocation = "Dynamic"

  }

  tags = {

    Name = "My NIC"

  }
}

resource "azurerm_linux_virtual_machine" "myvm" {
```

```
name                = "abhi-VM"

resource_group_name = azurerm_resource_group.main.name

location            = azurerm_resource_group.main.location

size                = local.vm_size

admin_username      = "azureuser"

disable_password_authentication = false

network_interface_ids = [

    azurerm_network_interface.nic.id

]

admin_password = "P@ssword1234!"

os_disk {

    caching            = "ReadWrite"

    storage_account_type = "Standard_LRS"

    name                = "myOsDisk"

}

source_image_reference {

    publisher = "Canonical"

    offer     = "0001-com-ubuntu-server-jammy"
```



```

sku      = "22_04-lts"

version  = "latest"

}

tags = local.tags
}

```

Q. map:

Q.unused providers

Q.terraform plan -destroy -provider=which provider = impact of destroy of 1 provider(if i remove the provider)

Q.cleanup

Q.how to deploy a spring boot application with azure db in terraform

maven

The screenshot shows a configuration window for a Spring Boot project. At the top, there are radio buttons for selecting a version: 3.5.0 (SNAPSHOT), 3.5.0 (RC1), 3.4.6 (SNAPSHOT), 3.4.5 (selected), 3.3.12 (SNAPSHOT), and 3.3.11. Below this is the 'Project Metadata' section with the following fields:

- Group: com.ust
- Artifact: employee
- Name: employee
- Description: no project for connecting fromSpring Boot to azure db
- Package name: com.ust.employee
- Packaging: Jar (selected), War
- Java: 24, 21, 17 (selected)

At the bottom of the window, there are three buttons: 'GENERATE' (with a keyboard shortcut CTRL + G), 'EXPLORE' (with a keyboard shortcut CTRL + SPACE), and an ellipsis button '...'.

-

Q. for_each

```
provider "azurerm" {

  features {}

}

terraform {

  required_providers {

    azurerm = {
```

```
    source = "hashicorp/azurerm"

    version = "4.27.0"

  }

}

}

variable "server_config" {

  description = "Configuration for the Azure Virtual Machines"

  type = map(object({

    os_type    = string

    publisher  = string

    offer      = string

    sku        = string

    vm_size    = string

  }))

  default = {

    "web-server-a" = {

      os_type    = "Linux"

      publisher  = "Canonical"

      offer      = "0001-com-ubuntu-server-jammy"

      sku        = "22_04-lts"

      vm_size    = "Standard_B1s"

    },

  },
```

```
"app-server-b" = {  
  
    os_type      = "Windows"  
  
    publisher    = "MicrosoftWindowsServer"  
  
    offer        = "WindowsServer"  
  
    sku          = "2019-Datacenter"  
  
    vm_size      = "Standard_B2ms"  
  
}  
  
}  
  
}  
  
variable "resource_group_name" {  
  
    description = "Name of the resource group"  
  
    type        = string  
  
    default     = "abhiResourceGroup"  
  
}  
  
variable "location" {  
  
    description = "Location for the resources"  
  
    type        = string  
  
    default     = "WestUS"  
  
}  
  
# Resource Group
```

```
resource "azurerm_resource_group" "rg" {  
  
  name      = var.resource_group_name  
  
  location = var.location  
  
}  
  
# Virtual Network and Subnet  
  
resource "azurerm_virtual_network" "vnet" {  
  
  name            = "abhivyVNet"  
  
  address_space   = ["10.0.0.0/16"]  
  
  location        = var.location  
  
  resource_group_name = azurerm_resource_group.rg.name  
  
}  
  
resource "azurerm_subnet" "subnet" {  
  
  name            = "abhivySubnet"  
  
  resource_group_name = azurerm_resource_group.rg.name  
  
  virtual_network_name = azurerm_virtual_network.vnet.name  
  
  address_prefixes     = ["10.0.0.0/24"]  
  
}  
  
# Public IP  
  
resource "azurerm_public_ip" "public_ip" {  
  
  for_each        = var.server_config
```

```

name                = "${each.key}-public-ip"

location            = var.location

resource_group_name = azurerm_resource_group.rg.name

allocation_method   = "Static"

sku                 = "Standard"
}

# Network Interface

resource "azurerm_network_interface" "nic" {

  for_each          = var.server_config

  name              = "${each.key}-nic"

  location          = var.location

  resource_group_name = azurerm_resource_group.rg.name

  ip_configuration {

    name                = "internal"

    subnet_id           = azurerm_subnet.subnet.id

    private_ip_address_allocation = "Dynamic"

    public_ip_address_id =
azurerm_public_ip.public_ip[each.key].id

  }
}

# Linux VMs

```

```
resource "azurerm_linux_virtual_machine" "linux_vm" {

  for_each = { for k, v in var.server_config : k => v if v.os_type ==
"Linux" }

  name                = each.key

  resource_group_name = azurerm_resource_group.rg.name

  location            = var.location

  size                = each.value.vm_size

  admin_username      = "azureuser"

  disable_password_authentication = true

  network_interface_ids = [azurerm_network_interface.nic[each.key].id]

  os_disk {

    name                = "${each.key}-osdisk"

    caching              = "ReadWrite"

    storage_account_type = "Standard_LRS"

  }

  source_image_reference {

    publisher = each.value.publisher

    offer      = each.value.offer

    sku        = each.value.sku

    version    = "latest"

  }

}
```

```

}

admin_ssh_key {

    username    = "azureuser"

    public_key = file("~/ssh/id_rsa.pub")

}

}

# Windows VMs

resource "azurerm_windows_virtual_machine" "windows_vm" {

    for_each = { for k, v in var.server_config : k => v if v.os_type ==
"Windows" }

    name                = each.key

    resource_group_name = azurerm_resource_group.rg.name

    location            = var.location

    size                = each.value.vm_size

    admin_username      = "myadmin"

    admin_password      = "Gowri1234!" # Do not hardcode in production

    network_interface_ids = [azurerm_network_interface.nic[each.key].id]

    os_disk {

        name                = "${each.key}-osdisk"
    }

```



```

    caching          = "ReadWrite"

    storage_account_type = "Standard_LRS"
}

source_image_reference {

    publisher = each.value.publisher

    offer      = each.value.offer

    sku        = each.value.sku

    version    = "latest"
}
}

```

Q. inplace update - without destroying,update

- Create vm in terraform
- If we modify tag -it wont goes down
- And updated the tag and again plan and apply
- An **in-place update** with Terraform means applying a change **without destroying and recreating the resource**—Terraform will just update the existing infrastructure (e.g., changing tags, VM size, or password).

```

terraform {

    required_providers {

        azurerm = {

            source = "hashicorp/azurerm"

            version = "~> 3.0"

        }

    }

}

```

```
required_version=">=1.0"

}

provider "azurerm" {

  features {}

}

locals {

  vm_size = "Standard_B1s"

  location = "centralindia"

  tags = {

    Name = "abhi-vm-updated" #first it was abhi-vm then i updated

    Env  = "Dev"

  }

}

resource "azurerm_resource_group" "main" {

  name      = "abhi-rg7"

  location = "centralindia"

}

resource "azurerm_virtual_network" "vnet" {

  name          = "abhi-Vnet1"

  address_space = ["10.0.0.0/16"]

}
```

```
location          = azurerm_resource_group.main.location

resource_group_name = azurerm_resource_group.main.name
}

resource "azurerm_subnet" "subnet" {

  name          = "abhi-Subnet1"

  resource_group_name = azurerm_resource_group.main.name

  virtual_network_name = azurerm_virtual_network.vnet.name

  address_prefixes      = ["10.0.1.0/24"]
}

resource "azurerm_public_ip" "public_ip" {

  name          = "abhi-public-ip"

  location      = local.location

  resource_group_name = azurerm_resource_group.main.name

  allocation_method = "Dynamic"

  sku           = "Basic"
}

resource "azurerm_network_interface" "nic" {

  name          = "abhi-NIC1"

  location      = azurerm_resource_group.main.location

  resource_group_name = azurerm_resource_group.main.name
```

```
ip_configuration {  
  
    name = "internal"  
  
    subnet_id = azurerm_subnet.subnet.id  
  
    private_ip_address_allocation = "Dynamic"  
  
    public_ip_address_id = azurerm_public_ip.public_ip.id  
  
}  
  
tags = {  
  
    Name = "My NIC"  
  
}  
}  
  
resource "azurerm_linux_virtual_machine" "myvm" {  
  
    name = "abhi-VM1"  
  
    resource_group_name = azurerm_resource_group.main.name  
  
    location = azurerm_resource_group.main.location  
  
    size = local.vm_size  
  
    admin_username = "azureuser"  
  
    disable_password_authentication = false  
  
    network_interface_ids = [  

```

```

    azurerm_network_interface.nic.id

  ]

  admin_password = "P@ssword1234!"

  os_disk {

    caching          = "ReadWrite"

    storage_account_type = "Standard_LRS"

    name              = "myOsDisk"

  }

  source_image_reference {

    publisher = "Canonical"

    offer     = "0001-com-ubuntu-server-jammy"

    sku       = "22_04-lts"

    version   = "latest"

  }

  tags = local.tags

}

```

Q. install java in a shell script and pass in the code of vm creation in terraform?

- Created a sh file with java installation commands
- Created a vm and passed the sh file in the code

Q. dns and firewall required within azure .y?

Q.security?

- ★ Defence at depth
- ★ Layered security : implementing security at multiple layers
- ★ Monitoring:
 - 1.
 - 2.
 - 3.Hardening machines
 - 4.Securing docker images
 - 5. Surface reduces - area of attack reduces
 - 6. Checking image security
 - 7.communication b/w pods (network policies)
 - 8.network watchers
 - 9.services which can watch traffic and network based communication
- ★ Monolithic:
 - Middle layer(business logic layer),first layer(presentation layer),3rd layer(database)
- ★ No sql database - 4 types
 - Document based
 - Key value
 - Graph : instagram
 - Wide column database
 - Req comes fall on these and forwarded to load balancer - capable of doing 2 things
 - Path based routing
 - Distribute traffic to diff instances
 - Req will come to microservice before it will do authentication and authorization-conditional access policy(are u accessing it from india or abroad) -ask u to do mfa - from somewhere else u cant do that
 - Here multiple load balancers and one of the instance picked up
 - That one of the instance cant persist data
 - Because services are stateless
 - Externalise the data - db or nosql or combinational
 - Store the same data in db but cache
 - Communication -synchronous(need load balancer(hardware or software)) and asynchronous
 - Problem:multiple instances
 - So load balancers - paying money for that so create eureka
 - 1 db per service design - because 3 axis of scalability
 - Horizontal scalability - make ur service stateless
 - Functional decomposition -done by developers
 - Data partitioning: if using single db b/w multiple services,become bottleneck,cant scale
 - no -sql database : talking to broker(kafka,rabbit mq)
 - asynchronous:Queue and publish subscribe model -no loadbalancer needed
 - Also have api gateways : cloud has api gateways u can configure it
 - Api gateway: will work with authentication and authorization

- Req will forwarded to microservice -can be synchronous call or asynchronous -again same story
- Actuators:
- How to deploy
- Lb deployed in 1 public subnet all other in private subnet
- When u have n no of microservice -loadbalancer sit there -ms1 -ms2 -ms3 - db
- Multiple instances for ms1,ms2....(multiple subnet)
- b/w ms1 and ms2 -need load balancing
- Many monitoring services-
 - any communication in vnet
 - Extracting info of machines
 - Enforce security at web traffic
 - Aks -solution for security issue
- Delay the attack - prevent security attack
- Circuit breaker:

Q. functions in terraform:

- <https://spacelift.io/blog/terraform-functions-expressions-loops>
- Random string: create before destroy
 - ◆ Created vm and added create before destroy and applied -created vm
 - ◆ So again changed the image and seeing created before destroy
 - ◆ Vm -change image-Default behaviour -destroy and create
 - ◆ Create new vm with new image then destroy
- Ignore changes:
- Prevent from destroy

- Kubernetes service in portal -cluster

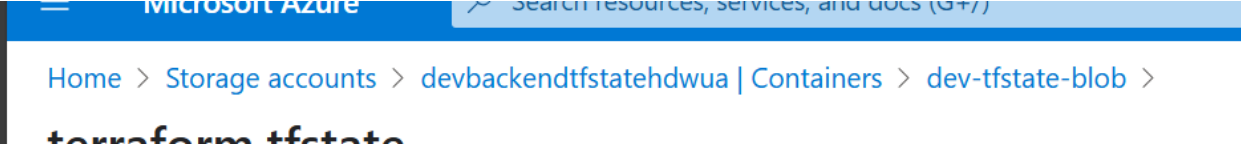
9/5/2025

-var:

- [How to Use Terraform Variables: Examples & Best Practices](#)
- The **-var** flag is used in **Terraform CLI** to **pass values to variables** at runtime when executing commands like **terraform plan** or **terraform apply**.
- When u code anything related to provider then only u should define terraform block
- Precedence: -var have highest precedence

- Aim:
- Backend configuration: where it store tf state file
- Refer to a storage account -not yet created yet - apply -created
- Solution: azure cli and terraform attach, terraform block
- Init -looks - refer to remote in storage acc
- Apply - already there
- [Store Terraform state in Azure Storage | Microsoft Learn](#)
- Use azure cli better
- Path to check the state file is locked or not

→



```

user02 [ ~/azure-repo/terraformstate ]$ terraform apply -auto-approve

Error: Error acquiring the state lock

Error message: state blob is already locked
Lock Info:
  ID:          0ed10f89-dc59-b38e-38b1-0dc9333d6aa0
  Path:        dev-tfstate-blob/terraform.tfstate
  Operation:   OperationTypeApply
  Who:         user03@SandboxHost-638823626926775016
  Version:     1.11.3
  Created:     2025-05-09 06:42:08.091515018 +0000 UTC
  Info:

Terraform acquires a state lock to protect the state from being written
by multiple users at the same time. Please resolve the issue above and try
again. For most commands, you can disable locking with the "-lock=false"
flag, but this is not recommended.
  
```

→

Q. some important terms:

1. Staging environment:

- a. **Staging** is used to test before production.
- b. It should be **same as production** setup.
- c. Create **2 machines** (VMs) with **bigger size** to simulate production load.
- d. **Why do we create 2 machines in staging?**

- i. You're building a website like Amazon. In **production** (the live version users see), there are always **multiple machines (servers)** working together:
- ii. 1 handles user login, 1 handles payments, 1 stores product data
- iii. We test everything in **staging** (a testing area). If you test with **only 1 machine** in staging, it won't show problems that happen **when machines work together**.
- iv. We check
 1. Machines can **talk to each other**
 2. **Load balancing** works (sharing traffic)
 3. **Deployments** happen on both properly

2. Modules in terraform

- A **module** is a folder that contains Terraform code.
- You can **reuse** this folder in other Terraform files.
- Helps avoid writing the same code again and again.
- Example:
- One folder has code to create a network.
- You can call this folder in other Terraform files using **module** block.

3. Workspaces

- a. Workspaces allow us to create multiple environments (like dev, staging, prod) using the same Terraform code.
- b. Each workspace has its own state file.
- c. **Problem:**
 - i. Sometimes dev and staging share the same state file.
 - ii. This is wrong because changes in staging can affect dev.
- d. **Solution:**
 - i. Use workspaces to separate them:
 - ii. terraform workspace new staging
 - iii. terraform workspace select staging
 - iv. Terraform will keep separate state files for each workspace.

4. Var Files:

- a. **Each environment should have a different variable file:**
 - i. **dev.tfvars** for dev
 - ii. **staging.tfvars** for staging
 - iii. **prod.tfvars** for production
 - iv. terraform apply -var-file=dev.tfvars

5. Functions in Terraform

a. Input variables:

i. These are the values you pass into the Terraform code.

1. variable "location" {
2. default = "eastus"
3. }

b. Local Values:

i. You use them when you want to store a value once and use it many times, just within that file.

1. locals {
2. rg_name = "abhi-resourcegroup"
3. location = "eastus"
4. }
5. Can use like this inside the block ...

```
resource "azurerm_resource_group" "example" {
  name      = local.rg_name
  location  = local.location
}

resource "azurerm_storage_account" "example" {
  name                  = "abhi123storage"
  resource_group_name  = local.rg_name
  location              = local.location
  account_tier         = "Standard"
  account_replication_type = "LRS"
}
```

c. Output Variables

i. Values that you want to **use in other files** or show after **apply**.

```
output "vm_name" {
  value = azurerm_linux_virtual_machine.vm.name
}
```

ii.

Example:

You created a resource group:

```
hcl                                                                    Copy Edit

resource "azurerm_resource_group" "example" {
  name      = "abhi-rg"
  location  = "eastus"
}
```

iii.

Now you want to **display the name** of the resource group after applying.

👉 Add this output variable:

```
hcl

output "rg_name" {
  value = azurerm_resource_group.example.name
}
```

When you run:

```
bash

terraform apply
```

iv.

When you run:

```
bash

terraform apply
```

➡ You'll see:

```
makefile

Outputs:

rg_name = "abhi-rg"
```

v.

6. Important Terraform Commands:

a. Refresh-Only Apply

- i. Use when you want to update the state without making changes.
- ii. “Don't make any changes to the actual infrastructure — just check what's changed and update the **state file**.”
- iii. Someone **changes something directly** in the cloud (like Azure/AWS portal).
- iv. But your Terraform **state file** doesn't know about it yet.
- v. You don't want to apply anything new — just want to **refresh your state** to match the real cloud environment.
- vi. Eg: u created a resource group in terraform,u updated the name of rg in azure portal manually, but ur tf.state file still have old name so..use command:
 1. terraform apply -refresh-only
 2. Update the **Terraform state file**
 3. Without this -refresh-only..when u do terraform apply...it thinks something new ..so try to delete and recreate it

7. Terraform State File Changes:

- a. If you make changes manually or outside Terraform, you need to update the Terraform configuration.

- b. Then use these commands to manage the state:
- c. `terraform state list` → shows all resources in the state
- d. `terraform state show <resource>` → details of one resource

8. Bastion Host Setup (Azure Portal Example)

- a. **Suppose you create a VM with only private IP (not reachable from outside).**

Use Azure Bastion to connect:

1. Create a Bastion host in the same VNet.
2. Bastion gets a public IP.
3. You can then connect securely to the private VM through Bastion.

