Azure:

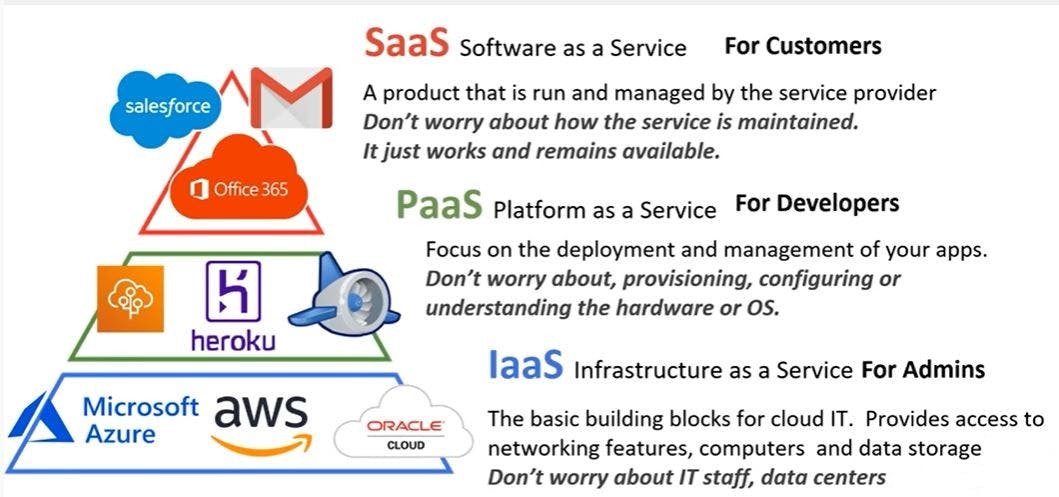
Azure is Microsoft's cloud computing platform. It provides a wide range of services, like storage, databases, and virtual machines, over the internet. Instead of owning and maintaining physical servers, businesses can use Azure to run applications, store data, and more, all in a flexible, scalable way. It’s like renting a powerful computer in the cloud that you can access from anywhere.

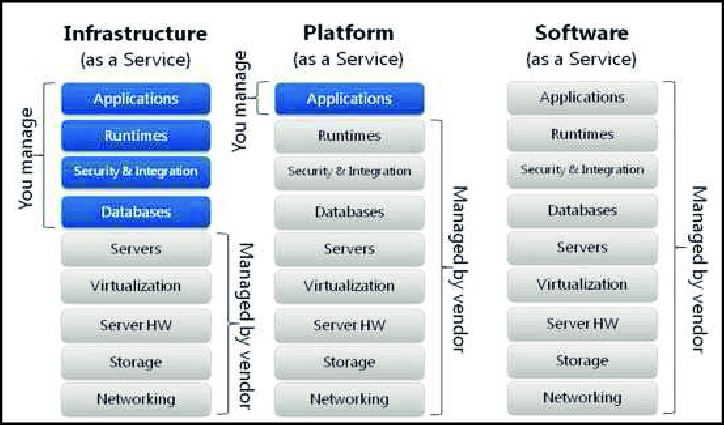
Decrease the capital expenditure and operational expenditure we use cloud service

Capex=baying your own infrastructure,big intial investment

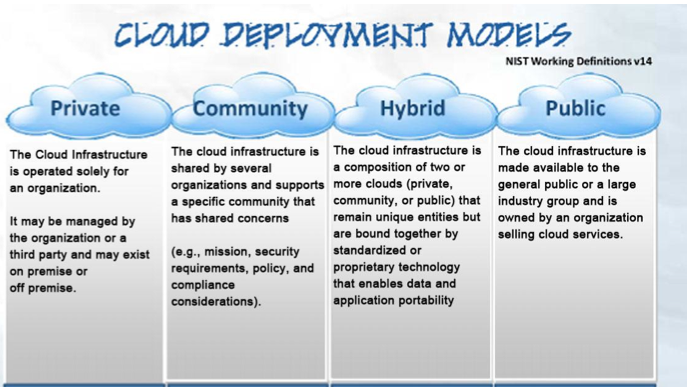
Opex= rent the infrastuture,no intial investment

Service models:





Deployment models:



Azure offers several deployment models to accommodate different business needs and preferences. These models define how and where your cloud resources are deployed and managed. Here’s a breakdown:

**1. Public Cloud**

**What it is:**

* Resources are owned and operated by a third-party cloud service provider (like Microsoft Azure) and shared with multiple organizations over the internet.

**Key Points:**

* **Cost-Effective:** You only pay for what you use, with no need to invest in hardware.
* **Scalability:** Easily scale resources up or down based on demand.
* **Maintenance-Free:** The cloud provider manages hardware and infrastructure.

**Real-Time Example:**

* A startup uses Azure’s public cloud to host its website, leveraging Azure’s global data centers without owning any physical infrastructure.

**Why Use Public Cloud:**

* Ideal for businesses looking for flexibility, cost savings, and scalability without the need to manage infrastructure.

**2. Private Cloud**

**What it is:**

* Cloud resources are used exclusively by a single organization. The infrastructure can be hosted on-site or in a data center, but it is not shared with others.

**Key Points:**

* **Enhanced Security:** Greater control over security, compliance, and data privacy.
* **Customization:** Infrastructure can be tailored to meet specific business needs.
* **Dedicated Resources:** No sharing of resources with other organizations.

**Real-Time Example:**

* A bank uses a private cloud to store sensitive financial data, ensuring compliance with strict regulatory requirements.
* **Why Use** Best for organizations with strict security, compliance, or performance needs, like financial institutions or government agencies.

**3. Hybrid Cloud**

**Private Cloud:**

**What it is:**

* A combination of public and private clouds, allowing data and applications to be shared between them. This model offers the best of both worlds.

**Key Points:**

* **Flexibility:** Businesses can keep sensitive data in a private cloud while using the public cloud for less sensitive operations.
* **Cost-Effective:** Utilize the public cloud for high-volume tasks, reducing the load on the private cloud.
* **Seamless Integration:** Allows for easy movement of workloads between public and private clouds.

**Real-Time Example:**

* A retail company runs its e-commerce site on the public cloud but stores customer payment information on a private cloud for enhanced security.

**Why Use Hybrid Cloud:**

* Suitable for organizations that want to balance security with cost-efficiency and scalability.
* Ideal for businesses with fluctuating workloads or sensitive data that must remain on-premises.

Virtual machines:

Virtually creating a machines.

No physical wires involved in connecting by virtual network.

Collection of subnets is a virtual network

Security group is nothing but a fire walls.

1. **On-Demand Computing Power:** Azure VMs give you flexible, cloud-based computers that you can start and stop as needed, paying only for the time you use.
2. **Choose Your Operating System:** You can run either Windows or Linux on your Azure VM, just like a physical computer.
3. **Scalable Resources:** Easily increase or decrease your VM's power and storage based on your business needs, ensuring you only pay for what you need.
4. **Customization and Control:** You have full control over your VM, allowing you to install and run any software or application just like on a physical machine.
5. **Global Reach:** Azure VMs can be deployed in data centers around the world, helping you improve performance and accessibility for users no matter where they are.

**Use Cases:**

1. **Development and Testing:**
   * Developers can quickly spin up VMs to test new applications or features in an isolated environment, then shut them down when no longer needed.
2. **Running Applications:**
   * Businesses can run applications that require specific operating systems or software configurations. For example, running a legacy business application that requires an older version of Windows.
3. **Disaster Recovery:**
   * VMs can be used for backup and disaster recovery solutions, ensuring that critical applications and data are protected and can be quickly restored.
4. **Hosting Websites:**
   * Azure VMs can host websites, especially when you need full control over the server environment or need to run custom server-side applications.

**Real-Time Example:**

* **E-Commerce Platform:** An online store may use Azure VMs to host its website and database. During peak shopping seasons, the store can easily increase the number of VMs to handle more traffic, and then scale back down afterward to save costs.

**Why Use Azure Virtual Machines?**

* **Flexibility and Control:** You have complete control over the computing environment, making it suitable for a wide range of applications and workloads.
* **Global Reach:** Azure’s global data centers ensure that your VMs can be deployed close to your customers, improving performance and reliability.
* **Security:** Azure provides built-in security features, like encryption and compliance certifications, to protect your data and applications.

Azure Virtual Machines offer a flexible, scalable, and cost-effective solution for businesses needing computing resources without investing in physical hardware. They can be tailored to a wide range of needs, from simple testing environments to hosting large-scale, mission-critical applications.

Storage:

Azure offers a variety of storage services that cater to different data storage needs, from simple file storage to large-scale data archiving. Here’s a breakdown:

**Azure Storage Services Overview**

1. **Azure Blob Storage:**
   * **What It Is:** A service for storing unstructured data, like images, videos, and backups.
   * **Key Features:**
     + **Scalable:** Handle massive amounts of data.
     + **Accessible:** Store any type of text or binary data, accessible over the internet.
   * **Use Case:** Storing website images, backup files, or video content for streaming.
2. **Azure File Storage:**
   * **What It Is:** A managed file share service that can be accessed via SMB (Server Message Block) protocol.
   * **Key Features:**
     + **File Sharing:** Share files across multiple virtual machines or users.
     + **Accessible:** Mount it just like a network drive.
   * **Use Case:** Centralized file storage for applications or shared files between teams.
3. **Azure Disk Storage:**
   * **What It Is:** Provides high-performance, durable block storage for use with Azure VMs.
   * **Key Features:**
     + **Persistent Storage:** Keep your data even if the VM is shut down.
     + **High Performance:** Ideal for applications that require fast data access.
   * **Use Case:** Attaching disks to virtual machines for databases, applications, or other high-performance workloads.
4. **Azure Queue Storage:**
   * **What It Is:** A service for storing large numbers of messages that can be accessed from anywhere in the world.
   * **Key Features:**
     + **Message Queuing:** Decouple and scale components of your applications.
     + **Reliability:** Ensure messages are delivered even if the system fails temporarily.
   * **Use Case:** Managing tasks or processing orders in an e-commerce application.
5. **Azure Table Storage:**

**Child of nosql,structure will change dynamically**

* + **What It Is:** A NoSQL key-value store for storing large amounts of structured, non-relational data.
  + **Key Features:**
    - **Simple Structure:** Store data in a simple key-value format.
    - **Scalable:** Easily handle large amounts of structured data.
  + **Use Case:** Storing user profiles, device information, or any other large-scale, structured data.

**Why Use Azure Storage Services?**

* **Scalability:** Easily scale your storage needs up or down depending on your usage, ensuring you only pay for what you need.
* **Security:** Built-in encryption and compliance certifications to protect your data.
* **Global Accessibility:** Store and access your data from anywhere in the world through Azure’s global data centers.
* **Integration:** Seamlessly integrates with other Azure services, making it easier to build and manage your applications.

Azure Storage Services provide versatile, secure, and scalable storage solutions that cater to a wide range of business needs, from simple file storage to complex data management systems.

Virtual networks:

Communication channel between resource lanctued in the cloud

Azure Virtual Networks (VNets) allow you to create a private, isolated network in the Azure cloud. This network can be used to securely connect and manage your Azure resources, such as Virtual Machines (VMs), databases, and other services.

**What is an Azure Virtual Network (VNet)?**

* **Private Networking:** Azure VNets provide a private IP space that is isolated from the internet and other Azure VNets, giving you complete control over your network environment.
* **Subnets:** Within a VNet, you can create subnets to organize and segment your resources for better management and security.
* **Connectivity Options:** VNets can connect to on-premises networks, other VNets, or the internet using VPNs, ExpressRoute, or other services.
* **Security:** Use Network Security Groups (NSGs) and Azure Firewall to control inbound and outbound traffic to resources within the VNet.

**Steps to Create a Virtual Network in Azure**

Here’s a step-by-step guide on how to create an Azure VNet using the Azure portal:

**1. Sign in to the Azure Portal**

* Go to <https://portal.azure.com> and sign in with your Azure account.

**2. Create a New Virtual Network**

* In the Azure portal, select **"Create a resource"** from the left-hand menu.
* Search for **"Virtual Network"** and select it from the search results.
* Click on **"Create"** to start the creation process.

**3. Configure Basic Settings**

* **Subscription:** Choose the Azure subscription you want to use.
* **Resource Group:** Either create a new resource group or select an existing one. Resource groups help organize related resources.
* **Name:** Give your VNet a name that reflects its purpose or the environment it supports.
* **Region:** Choose the Azure region where you want to create the VNet. The region determines where the VNet and its resources will be hosted.

**4. Configure IP Address Space**

* **IP Address Space:** Define the address space for your VNet using CIDR notation (e.g., 10.0.0.0/16). This range will be used for assigning private IP addresses to resources within the VNet.
* **Subnets:** Create one or more subnets within the VNet by defining the subnet name and address range (e.g., 10.0.1.0/24). Subnets help organize and secure resources.

**5. Configure Security Settings**

* **DDoS Protection:** Choose whether to enable DDoS protection for your VNet (optional).
* **Firewall:** Decide if you want to deploy an Azure Firewall within the VNet to control and monitor traffic.

**6. Review and Create**

* **Tags:** Optionally, add tags to categorize your VNet (e.g., environment, department).
* **Review and Create:** Review all the settings you configured. If everything looks good, click **"Create"** to deploy the VNet.

**7. Deployment**

* Azure will begin deploying your VNet. This usually takes a few seconds. Once done, you’ll get a notification that your VNet has been successfully created.

**8. Connect Resources to the VNet**

* After the VNet is created, you can start connecting Azure resources like Virtual Machines (VMs), Azure SQL Database, and more to your VNet.
* When creating a VM, for example, you can select this VNet and its subnets to define where the VM will reside.

**Summary**

* **Azure Virtual Networks (VNets)** are private networks that provide isolation, security, and connectivity for your Azure resources.
* **Creating a VNet** involves configuring the basic settings, IP address space, subnets, and security options in the Azure portal.
* After creating a VNet, you can connect and manage various Azure resources within it.

This setup allows you to build secure, scalable, and well-organized network environments within the Azure cloud.

Here are Azure Virtual Networks (VNets) explained in 5 simple points:

1. **Private Networking:** VNets create a private network in Azure, isolating your resources from the public internet.
2. **Subnets:** Divide your VNet into smaller networks (subnets) to organize and secure resources like Virtual Machines.
3. **Secure Connectivity:** Connect your VNet to on-premises networks or other VNets using VPNs or ExpressRoute.
4. **Traffic Control:** Use tools like Network Security Groups (NSGs) and Azure Firewall to manage and secure network traffic.
5. **Global Reach:** Deploy your VNet in any Azure region, providing flexibility and scalability for your global infrastructure needs.

**Azure active directoty:**

**1. Why We Use It:**

* **Centralized Identity Management:** Azure AD manages user identities and access across multiple apps and services.
* **Single Sign-On (SSO):** Users can sign in once to access all their apps, reducing the need for multiple passwords.
* **Security:** Provides multi-factor authentication (MFA) to enhance security.
* **Seamless Integration:** Works with Microsoft 365, Azure services, and many third-party applications.
* **Compliance:** Helps organizations meet security and compliance requirements.

**2. What It Is Used For:**

* **User Authentication:** Verify user identities when they log into apps or services.
* **Access Management:** Control who has access to what within your organization.
* **Device Management:** Manage and secure devices that access company resources.
* **Application Management:** Integrate and manage third-party and custom apps.
* **Collaboration:** Enable secure collaboration with external users (e.g., partners, clients).

**3. Key Features:**

* **User Groups:** Organize users into groups for easier access management.
* **Conditional Access:** Set policies to control access based on conditions like location or device.
* **Identity Protection:** Detect and respond to potential identity threats.
* **Self-Service Password Reset:** Allow users to reset their passwords without IT support.
* **Reporting and Monitoring:** Get insights into sign-in activities and potential security risks.

**4. How to Use It:**

* **Set Up Users and Groups:** Add users, create groups, and assign them to roles or apps.
* **Configure SSO:** Enable Single Sign-On for your apps so users can access everything with one login.
* **Implement MFA:** Set up multi-factor authentication to secure user logins.
* **Create Access Policies:** Define conditional access policies to control how and when users can access resources.
* **Monitor Activity:** Use Azure AD’s reporting tools to monitor sign-ins and detect unusual activities.

**5. Summary:**

* **Centralized Control:** Azure AD simplifies and secures the management of user identities and access.
* **Enhanced Security:** Provides tools like MFA and Conditional Access to protect against unauthorized access.
* **Seamless Experience:** Users get a seamless sign-in experience across multiple apps and services.
* **Flexibility:** Integrates with a wide range of Microsoft and third-party apps.
* **Compliance and Reporting:** Helps organizations meet security standards and monitor access activities.

Azure AD is a powerful tool for managing and securing identities in the cloud, ensuring that the right people have access to the right resources, with the right level of security.

Kubernates:

1)what is devops?

- devops is not a tool, devops is culture (or) process. - it is a process of continuous development, continuous build, continuous test, continuous release the software faster & reliable with automation tools. - it is a combination of development & operations. - this automation process is hapend between the development & IT Teams

2) why we need devops? - main culture of devops is collaboration between teams (development & operations). - we can achieve the our goal fast & reliably. - to understand devops culture try to understand the previous models. –

every software company to develop an application (or) software it has 2 main teams. 1)development 2)operations (testing, monitoring,infrastructure)

Development:

development team develop the application with some tools. eg: java based website - front end by UI team (html, CSS, JAVASCRIPT, JQUARY, ANGULARJS, REACTJS) - back end by JAVA developers (server side) - DB admins give access of DATABASE to the developers. - all people together develop an website. - integration the code. - unit test the code (developer self test). - for another process need to move the code into another environment (Testing, QA, etc..) (developers not doing this)

IT Team (operatons): ------------------- -

IT Team is setup the required environment to the Development, Testing, QA, Pre-production, Production. eg: set up infrastructure for java website application. - setup the hardware - setup the platform (WINDOWS, LINUX, etc..) - setup the required applications for devlopers such JAVA, JAVA compiler, Tomcat, Databases, etc... - they move the code from one environment to another. - they monitor the infrastructure, etc.. - they test the code.

Tradational development life cycle: ----------------------------------- -

tradationally to develop the software, waterfall model is their. Water Fall Model: ----------------- - It is a tradational SDLC(software development life cycle). - to develop an application some linear steps are their.

requirment -- gather information, what is purpose, documentation & planning |

design -- setup hardware & software |

implementation -- actual coding |

verification -- Integration |

maintaince -- deploy to customer - if any update is required you need to design this cycle again. - no return here. - bugs identify late. –

it takes months to complete a phase, for whole cycle takes min 2 years.

- for any changes start from 1st stage.

Agile Method: ------------- - It is a SDLC(software development life cycle) into multiple iterations. ---- 1)planning -- | | |

2)analysis 5)Testing | | | 4)build --3)design –

1)plan for what you need

2)analysis your planning

3)design (actual coding)

4)build (unit testing & building)

5)testing (integration and testing) - here we break the entire application into small pieces. - we can develop each piece in one iteration. - we can complete each iteration with in less time (min. 2 weeks)

These models are only at development stage only. - if the code is moving into the another environment the code maybe fail. - due to infrastructure, versions of software, coding issues, testing problem ,etc... - here which team raise the issue is not found. - teams are pointing each other for that issues. - to deliver the product to the customers with prevent the mistakes Quality assurance is one way. - but still we have problem to overcome this DEVOPS culture is very useful.

Advantages of DEVOPS: -------------------- -

software building & deployment with fast rate.

- application maintainence & bug fixing easy.

- release new features and versions of the applications in less time.

- higher product quality, faster productivity rate, increased customer satisfaction, more revenue and thus more profits.

- every tool is input of another tool. - we can get the feedback in this process.

TOOLS involved in Devops Life-Cycle: ------------------------------------

**Containers:**

1. **Lightweight:**
   * **Resource Usage:** Containers share the host OS kernel and resources, making them more lightweight and faster to start.
   * **Efficiency:** They have a smaller footprint and use fewer resources compared to VMs.
2. **Portability:**
   * **Consistent Environment:** Containers package an application and its dependencies together, ensuring it runs the same way across different environments.
   * **Cross-Platform:** Easily move containers between different environments, such as development, testing, and production.
3. **Isolation:**
   * **Process-Level Isolation:** Containers provide process-level isolation within the same OS, which is less isolated than VMs but sufficient for many use cases.
   * **Shared Kernel:** They share the same OS kernel but run in isolated user spaces.
4. **Deployment Speed:i**
   * **Fast Start-Up:** Containers start almost instantly because they don’t need to boot up an entire OS.
   * **Rapid Scaling:** Easily scale applications up or down with minimal overhead.
5. **Use Cases:**
   * **Microservices:** Ideal for deploying microservices due to their lightweight and portable nature.
   * **DevOps and CI/CD:** Frequently used in DevOps pipelines for consistent development and testing environments.

**Virtual Machines (VMs):**

1. **Heavyweight:**
   * **Resource Usage:** VMs run a full OS with its own kernel, which requires more resources and takes longer to start.
   * **Overhead:** VMs have more overhead due to the need to virtualize the entire operating system.
2. **Full OS Environment:**
   * **Complete Isolation:** Each VM runs a full OS, providing strong isolation from other VMs and the host system.
   * **Versatility:** You can run different OSes on the same physical hardware, like Windows VMs alongside Linux VMs.
3. **Deployment Speed:**
   * **Slower Start-Up:** VMs take longer to start because they need to boot up an entire OS.
   * **Resource Intensive:** Scaling requires creating and booting new VMs, which can be slower compared to containers.
4. **Security:**
   * **Stronger Isolation:** VMs offer stronger isolation as each VM has its own OS, making them suitable for running untrusted applications or different OS environments.
5. **Use Cases:**
   * **Legacy Applications:** Suitable for running applications that require a full OS or specific OS configurations.
   * **Resource-Intensive Applications:** Good for applications needing dedicated resources or those that can’t be easily containerized.

**Summary:**

* **Containers** are lightweight, fast, and portable, making them ideal for microservices and modern application deployments.
* **VMs** offer full OS isolation and are better for applications requiring a complete OS or that need strong security isolation.

Choosing between containers and VMs depends on your specific needs, such as resource requirements, isolation levels, and deployment speed.

What is Kubernetes:

 **Automates Container Management:** Kubernetes automatically deploys, scales, and manages containerized applications, ensuring they run smoothly across a cluster of machines.

 **Self-Healing and Load Balancing:** It automatically restarts failed containers, distributes network traffic, and scales applications up or down based on demand, providing high availability and reliability.

**Kubernetes Architecture in 5 Simple Points:**

1. **Master Node:**
   * **Control Plane:** Manages the cluster and makes global decisions about scheduling, scaling, and deployment.
   * **Components:** Includes the API server, scheduler, controller manager, and etcd (a key-value store for configuration data).
2. **API Server:**
   * **Central Interface:** The entry point for all API requests, managing communication between the control plane and worker nodes.
   * **Function:** Handles API requests from users and internal components, ensuring they are processed and stored.
3. **Scheduler:**
   * **Resource Assignment:** Assigns containers to worker nodes based on resource availability and workload requirements.
   * **Function:** Ensures containers are placed on nodes that have sufficient resources.
4. **Worker Nodes:**
   * **Execution Hosts:** Run the actual containers and manage application workloads.
   * **Components:** Include Kubelet (agent that communicates with the master node), container runtime (e.g., Docker), and Kube-Proxy (manages network routing).
5. **Pods:**
   * **Basic Unit of Deployment:** The smallest deployable unit in Kubernetes, containing one or more containers that share the same network namespace and storage.
   * **Function:** Encapsulates and runs the application containers and ensures they are scheduled and managed properly.

**Summary:**

* **Master Node:** Manages the overall cluster and makes high-level decisions.
* **Worker Nodes:** Run containers and execute workloads.
* **API Server, Scheduler, and Controller Manager:** Handle communication, scheduling, and state management.
* **Pods:** Basic units that run containerized applications.

Kubernetes architecture ensures efficient, scalable, and reliable management of containerized applications by coordinating the efforts of the master and worker nodes.

 **Kubelet** manages containerized applications on each node, ensuring they run correctly and report their status to the Kubernetes control plane.

 **Kube-Proxy** handles networking tasks, routing service requests to the appropriate Pods and balancing network traffic.

**\*\*1. Pay-As-You-Go Pricing:**

* **Usage-Based:** You pay for what you use, such as compute hours, storage space, and network bandwidth.
* **No Upfront Costs:** No long-term commitments or upfront payments are required; you are billed based on your actual usage.

**\*\*2. Reserved Instances:**

* **Discounted Rates:** Purchase reserved capacity for services like Virtual Machines (VMs) for a one- or three-year term to receive significant discounts compared to pay-as-you-go rates.
* **Commitment:** Requires an upfront commitment for the reserved term, which can result in cost savings.

**\*\*3. Spot Instances:**

* **Bargain Pricing:** Access unused Azure capacity at reduced rates. Ideal for workloads that are flexible and can tolerate interruptions.
* **Variable Availability:** Spot instances are not guaranteed and can be evicted with little notice if Azure needs the capacity.

**\*\*4. Free Tier:**

* **Limited Usage:** Azure offers a free tier with limited amounts of certain services, such as a small VM, limited storage, and a specific number of database transactions.
* **No Cost:** Suitable for testing and development purposes or small-scale applications.

**\*\*5. Pricing Calculator:**

* **Cost Estimator:** Use the [Azure Pricing Calculator](https://azure.microsoft.com/en-us/pricing/calculator/) to estimate the cost of Azure services based on your specific requirements and configurations.
* **Custom Estimates:** Input details such as resource types, quantities, and regions to get a tailored cost estimate.

**Summary:**

1. **Pay-As-You-Go:** Pay based on actual usage without upfront costs.
2. **Reserved Instances:** Get discounts by committing to use specific resources for a longer term.
3. **Spot Instances:** Save costs with reduced pricing for flexible, interruptible workloads.
4. **Free Tier:** Use limited free resources for testing and small projects.
5. **Pricing Calculator:** Estimate costs with the Azure Pricing Calculator based on your needs.

Understanding these pricing options can help you choose the most cost-effective approach for using Azure services based on your specific needs and usage patterns.

