Microsoft Azure Fundamentals

- Azure Virtual Machines
- Azure Web Apps
- Azure Container Instances
- Azure Virtual Networks
- Azure Blob Storage
- Azure SQL Database
- Azure IoT Hub
- Azure Functions
- Azure VM with a Template
- Azure VM with a PowerShell
- Azure VM with a CLI
- Azure Key Vault
- Azure Secure Network Traffic
- Azure Manage Access with RBAC
- Azure Manage Resource Locks
- Azure Resource Tagging
- Azure Create Policy
- Azure Tools

Azure Virtual Machines

Azure virtual machines (VMs) can be created through the Azure portal. This method provides a browser-based user interface to create VMs and their associated resources. This quickstart shows you how to use the Azure portal to deploy a Linux virtual machine (VM) running Ubuntu 18.04 LTS.

Why we use Azure Virtual Machines?

- **Flexibility:** Azure VMs allow you to choose from a wide variety of virtual machine sizes and configurations, including different CPU, memory, and storage options. This flexibility enables you to tailor VMs to your specific workload requirements.
- **Compatibility:** Azure VMs support multiple operating systems, including various versions of Windows Server, Linux distributions, and even specialized operating systems. This makes it easy to migrate or deploy existing applications to the cloud.
- **Customization:** You have full control over the virtual machine's configuration, allowing you to install and configure software, security settings, and networking options based on your needs.
- **Development and Testing:** Azure VMs are often used for development, testing, and QA environments. Developers can create VMs with specific configurations, replicate production environments, and test applications without affecting the production system.
- **Legacy Application Support:** Organizations can run legacy applications that might not be easily migrated to modern cloud-native services. Azure VMs provide a way to maintain and support these

applications in a cloud environment.

- **Infrastructure Migration:** Azure VMs facilitate the lift-and-shift migration of on-premises workloads to the cloud. You can replicate your existing infrastructure in Azure VMs and take advantage of cloud benefits.
- **High-Performance Computing:** Azure VMs with powerful hardware configurations are suitable for high-performance computing (HPC) workloads, such as simulations, rendering, and data analysis.
- **Scalability:** Azure VMs allow you to scale up or scale out your resources as needed. You can resize VMs to handle increased demand or deploy multiple VM instances behind a load balancer for improved performance.
- **Backup and Disaster Recovery:** Azure VMs can be backed up and replicated to provide reliable disaster recovery options. Azure Site Recovery helps automate the replication and recovery process.
- **Virtual Network Integration:** Azure VMs can be connected to Azure Virtual Networks, allowing you to create complex networking architectures, define routing, and establish secure communication between VMs and other Azure services.
- **Hybrid Scenarios:** Azure VMs can be part of hybrid scenarios, connecting on-premises environments to the cloud using technologies like Azure VPN Gateway or Azure ExpressRoute.
- **Custom Images:** You can create custom virtual machine images with preconfigured software and settings, making it easy to deploy consistent environments for your applications.
- **Compliance and Security:** Azure VMs offer security features such as encryption at rest and in transit, integration with Azure Active Directory for identity management, and network security groups for controlling traffic.
- **Cost Management:** Azure VMs provide a variety of pricing options, including pay-as-you-go and reserved instances, allowing you to optimize costs based on your usage patterns.

Create a VM

1. Sign in to the Azure portal at https://portal.azure.com.
2. Choose **Create a resource** in the upper left-hand corner of the Azure portal.
3. In the **Search the Marketplace** field, type **Ubuntu**. When **Ubuntu Server 18.04 LTS** appears in the search results, select it.
4. In the **Ubuntu Server 18.04 LTS** panel, select **Create**.
5. In the **Basics** tab, under **Project details**, make sure the correct subscription is selected and then choose to **Create new** resource group. Type **myResourceGroup** for the name.
6. Under **Instance details**, type **myVM** for the Virtual machine name and choose **East US** for your **Region**.
7. Under **Administrator account**, select **SSH public key**. For **Username**, type **azureuser**. For **Authentication type**, select **SSH public key**. For **SSH public key source**, select **Generate new key**

pair, then select **Create**.

8. In the **Size** tab, select **View all** and then select **D2s v3**. Select **Select**.
9. In the **Settings** tab, accept the default options and then select **Review + create**.
10. On the **Review + create** tab, select **Create**.

Connect to VM

1. After the deployment is complete, select **Go to resource**.
2. On the **Overview** page for your virtual machine, select **Connect**.
3. On the **Connect** to virtual machine page, keep the default options to connect by IP address over port 22, and then select **Download** to download the SSH connection file.
4. Use your SSH client to open the downloaded SSH connection file. For example, on Windows, you can use PuTTY. When prompted, enter the username **azureuser**.
5. When prompted, enter the password **you specified when creating the virtual machine**.
6. You should now be connected to your VM.

Clean up resources

When no longer needed, you can use the Azure portal to delete the resource group, virtual machine, and all related resources.

1. In the Azure portal, select **Resource groups** on the far left. Then select **myResourceGroup** in the resource group list.
2. On the **myResourceGroup** page, make sure that **myVM** is selected. Then select **Delete**.
3. Enter **myResourceGroup** for **TYPE THE RESOURCE GROUP NAME**. Then select **Delete**.

Azure Web Apps

Azure App Service enables you to build and host web apps, mobile back ends, and RESTful APIs in the programming language of your choice without managing infrastructure. It offers auto-scaling and high availability, supports both Windows and Linux, and enables automated deployments from GitHub, Azure DevOps, or any Git repo.

Why we use Azure Web Apps?

- **Ease of Deployment:** Azure Web Apps provides a simplified deployment process for web applications. You can deploy your code using various methods such as Git, GitHub, Azure DevOps, FTP, or Docker containers, without worrying about managing the underlying infrastructure.
- **Scalability:** Azure Web Apps offers easy scalability to handle changes in traffic. You can scale up or scale out your application as needed, both vertically (by upgrading the app service plan) and horizontally (by adding instances).

- **Managed Infrastructure:** With Azure Web Apps, you don't need to manage the underlying operating system, patching, or hardware. Microsoft handles the infrastructure management, allowing you to focus on your application code.
- **Multiple Programming Languages and Frameworks:** Azure Web Apps supports a variety of programming languages and frameworks, including .NET, Java, Python, Node.js, PHP, and more. This flexibility allows you to use the tools and technologies you are most comfortable with.
- **Integration and Extensibility:** Azure Web Apps seamlessly integrates with other Azure services, such as Azure SQL Database, Azure Blob Storage, Azure Cosmos DB, and Azure Active Directory. This makes it easy to build comprehensive and integrated cloud solutions.
- **Continuous Integration and Deployment (CI/CD):** Azure Web Apps can be easily integrated into your CI/CD pipelines, enabling you to automate the deployment process and ensure a consistent and reliable release cycle.
- **Automatic Scaling:** Azure Web Apps offers auto-scaling based on predefined rules or metrics, ensuring your application can handle fluctuations in user traffic without manual intervention.
- **Staging and Testing:** You can use deployment slots to create staging environments for testing new features or changes before deploying them to the production environment.
- **High Availability:** Azure Web Apps provide high availability with built-in load balancing and geographic distribution. You can also configure backup and disaster recovery options.
- **Security:** Azure Web Apps offers various security features, including HTTPS support, integration with Azure Active Directory for identity and access management, and the ability to restrict access using IP whitelisting.
- **Application Insights:** Azure Web Apps seamlessly integrates with Azure Application Insights, providing detailed monitoring, diagnostics, and performance insights for your applications.
- **Geo-Redundancy:** You can deploy your web app in multiple Azure regions for enhanced fault tolerance and geographic redundancy.
- **Serverless Capabilities:** Azure Web Apps can also run serverless functions using Azure Functions, allowing you to build event-driven, scalable, and cost-efficient solutions.

Create a web app

1. Sign in to the Azure portal at https://portal.azure.com.
2. Choose **Create a resource** in the upper left-hand corner of the Azure portal.
3. In the **Search the Marketplace** field, type **Web App**. When **Web App** appears in the search results, select it.
4. In the **Web App** panel, select **Create**.
5. In the **Basics** tab, under **Project details**, make sure the correct subscription is selected and then choose to **Create new** resource group. Type **myResourceGroup** for the name.

6. Under **Instance details**, type **mywebapp** for the Web app name and choose **East US** for your **Region**.
7. Under **Publish**, select **Code**.
8. Under **Runtime stack**, select **.NET Core 3.1 (LTS)**.
9. Under **Operating System**, select **Linux**.
10. Under **Linux Plan (East US)**, select **Create new**.
11. Under **App Service plan**, type **myAppServicePlan** for the name.
12. Under **Sku and size**, select **F1** for the **Size**.
13. Under **Application Insights**, select **Off**.
14. Under **Monitoring**, select **Off**.
15. Under **Backup**, select **Off**.
16. Under **Tags**, select **Next: Review + create**.
17. On the **Review + create** tab, select **Create**.

Connect to web app

1. After the deployment is complete, select **Go to resource**.
2. On the **Overview** page for your web app, select **Browse**.
3. You should now see the default web page for your web app.

Clean up resources

When no longer needed, you can use the Azure portal to delete the resource group, web app, and all related resources.

1. In the Azure portal, select **Resource groups** on the far left. Then select **myResourceGroup** in the resource group list.
2. On the **myResourceGroup** page, make sure that **mywebapp** is selected. Then select **Delete**.
3. Enter **myResourceGroup** for **TYPE THE RESOURCE GROUP NAME**. Then select **Delete**.

Azure Container Instances

Azure Container Instances offers the fastest and simplest way to run a container in Azure, without having to provision any virtual machines and without having to adopt a higher-level service.

Why we use Azure Container Instances?

- **Ease of Use:** Azure Container Instances abstracts away the complexities of managing virtual machines and orchestration platforms. It provides a simple way to deploy and manage containers without the

need for extensive container orchestration knowledge.

- **Rapid Deployment:** ACI enables you to quickly deploy containerized applications in seconds. This is especially useful for scenarios where you need to spin up containers for short-lived tasks or to handle bursty workloads.
- **Cost Efficiency:** With ACI, you pay only for the resources your containers consume during their execution. This granular billing model can be more cost-effective than running and managing full virtual machines for container workloads.
- **Isolation and Security:** Each container instance in ACI is isolated from others, providing a level of security and separation. This can be useful for scenarios where you want to run different containers in isolation.
- **Stateless Applications:** ACI is well-suited for stateless applications that don't require persistent storage. It's ideal for microservices and other types of applications that can be easily scaled out horizontally.
- **Event-Driven Workloads:** ACI can be triggered by events such as HTTP requests, timers, or messages from Azure Queue Storage, making it suitable for serverless-like scenarios where containers are spun up in response to specific events.
- **Batch Processing and Data Processing:** ACI is suitable for running batch processing jobs, data processing tasks, and other types of computation-intensive workloads that can be encapsulated in containers.
- **Testing and Development:** ACI provides a convenient environment for testing and development, allowing developers to quickly deploy and test containerized applications without the need to set up and manage a full development environment.
- **Integration with CI/CD Pipelines:** ACI can be easily integrated into continuous integration and continuous deployment (CI/CD) pipelines, enabling automated deployment and scaling of containers as part of your software delivery process.
- **Hybrid Scenarios:** ACI can be used in hybrid scenarios to extend on-premises applications to the cloud or to connect cloud-based microservices with on-premises data sources.
- **No Infrastructure Management:** With ACI, you don't need to worry about managing virtual machine instances, container orchestrators, or networking configurations. Azure takes care of the underlying infrastructure management.
- **Support for Multiple Operating Systems:** ACI supports both Linux and Windows containers, providing flexibility for running a wide range of containerized applications.

Create a container instance

1. Sign in to the Azure portal at https://portal.azure.com.
2. Choose **Create a resource** in the upper left-hand corner of the Azure portal.
3. In the **Search the Marketplace** field, type **Container Instances**. When **Container Instances** appears in the search results, select it.

4. In the **Container Instances** panel, select **Create**.
5. In the **Basics** tab, under **Project details**, make sure the correct subscription is selected and then choose to **Create new** resource group. Type **myResourceGroup** for the name.
6. Under **Instance details**, type **mycontainerinstance** for the Container name and choose **East US** for your **Region**.
7. Under **Image source**, select **Quickstart images**.
8. Under **Image**, select **microsoft/aci-helloworld**.
9. Under **Operating System**, select **Linux**.
10. Under **Size**, select **1 vCPU and 1 GiB memory**.
11. Under **Authentication type**, select **SSH public key**. For **Username**, type **azureuser**. For **SSH public key source**, select **Generate new key pair**, then select **Create**.
12. Under **Networking**, select **Public**.
13. Under **Tags**, select **Next: Review + create**.
14. On the **Review + create** tab, select **Create**.

Connect to container instance

1. After the deployment is complete, select **Go to resource**.
2. On the **Overview** page for your container instance, select **Containers**.
3. On the **Containers** page, select **mycontainerinstance**.
4. On the **mycontainerinstance** page, select **Connect**.
5. On the **Connect** page, select **SSH**.
6. On the **SSH** page, select **Click to copy to clipboard** to copy the SSH command to your clipboard.
7. Open a command prompt and paste the SSH command. Then press Enter.
8. When prompted, enter the password **you specified when creating the container instance**.
9. You should now be connected to your container instance.

Clean up resources

When no longer needed, you can use the Azure portal to delete the resource group, container instance, and all related resources.

1. In the Azure portal, select **Resource groups** on the far left. Then select **myResourceGroup** in the resource group list.
2. On the **myResourceGroup** page, make sure that **mycontainerinstance** is selected. Then select **Delete**.

3. Enter **myResourceGroup** for **TYPE THE RESOURCE GROUP NAME**. Then select **Delete**.

Azure Virtual Networks

Azure Virtual Network enables many types of Azure resources, such as Azure Virtual Machines (VM), to securely communicate with each other, the internet, and on-premises networks. Azure Virtual Network is foundational to Azure Networking and provides connectivity for Azure Virtual Machines, including Azure Kubernetes Service (AKS) and Azure Virtual Machine Scale Sets (VMSS), to communicate with internet, your on-premises environment, and other Azure services.

Why we use Azure Virtual Networks?

- **Network Isolation:** Azure Virtual Networks provide logical isolation for your resources in the cloud. You can create multiple VNETs to separate different environments, such as development, testing, and production, to prevent communication between them.
- **Security:** VNETs allow you to define network security rules using Network Security Groups (NSGs) and firewalls. You can control inbound and outbound traffic, ensuring that only authorized communication is allowed to and from your resources.
- **Hybrid Connectivity:** Azure Virtual Networks support hybrid scenarios, enabling you to establish secure connections between your on-premises data centers and Azure resources. This is achieved through technologies like Azure VPN Gateway or Azure ExpressRoute.
- **Subnetting:** Within a VNET, you can create subnets to further segment and organize your resources. Subnets can have different security requirements and access controls, helping you manage your network traffic effectively.
- **Private IP Addressing:** VNETs allow you to use private IP addresses for your resources, providing internal-only communication. This enhances security and helps you comply with regulatory requirements.
- **Application Architecture:** Azure Virtual Networks are essential for creating complex application architectures, where different components of your application can communicate securely while still maintaining isolation from other components.
- **Virtual Appliances:** You can deploy virtual appliances, such as network virtual appliances (NVAs) or firewall appliances, within your VNET to enhance security, monitoring, and routing capabilities.
- **Load Balancing:** Azure Virtual Networks support Load Balancers that distribute incoming network traffic across multiple instances of your application to ensure high availability and improved performance.
- **Multi-Region Deployments:** For applications requiring global reach, you can create VNETs in different Azure regions and connect them using Azure Global VNET Peering or Virtual Network Gateways.
- **Service Endpoints:** Azure services like Azure Storage and Azure SQL Database offer service endpoints within your VNET, enabling secure and optimized communication between your resources and those services.

- **DDoS Protection:** Azure Virtual Networks can be configured with DDoS protection plans to help safeguard your resources from distributed denial-of-service (DDoS) attacks.
- **Compliance and Governance:** By segmenting your resources into different VNETs and applying security controls, you can better meet compliance requirements and enforce network governance policies.
- **Resource Management:** VNETs provide a way to logically organize your Azure resources, making it easier to manage and maintain your network infrastructure.

Create a virtual network

1. Sign in to the Azure portal at https://portal.azure.com.
2. Choose **Create a resource** in the upper left-hand corner of the Azure portal.
3. In the **Search the Marketplace** field, type **Virtual Network**. When **Virtual Network** appears in the search results, select it.
4. In the **Virtual Network** panel, select **Create**.
5. In the **Basics** tab, under **Project details**, make sure the correct subscription is selected and then choose to **Create new** resource group. Type **myResourceGroup** for the name.
6. Under **Instance details**, type **myVirtualNetwork** for the Virtual network name and choose **East US** for your **Region**.
7. Under **Subnet**, select **Subnet**.
8. Under **Subnet name**, type **mySubnet**.
9. Under **Subnet address range**, type
10. Under **Tags**, select **Next: Review + create**.
11. On the **Review + create** tab, select **Create**.

Clean up resources

When no longer needed, you can use the Azure portal to delete the resource group, virtual network, and all related resources.

1. In the Azure portal, select **Resource groups** on the far left. Then select **myResourceGroup** in the resource group list.
2. On the **myResourceGroup** page, make sure that **myVirtualNetwork** is selected. Then select **Delete**.
3. Enter **myResourceGroup** for **TYPE THE RESOURCE GROUP NAME**. Then select **Delete**.

Azure Blob Storage

Azure Blob storage is Microsoft's object storage solution for the cloud. Blob storage is optimized for storing massive amounts of unstructured data. Unstructured data is data that does not adhere to a particular data model or definition, such as text or binary data.

Why we use Azure Blob Storage?

- **Scalability:** Azure Blob Storage offers virtually unlimited storage capacity, allowing you to scale your storage resources up or down based on your needs without worrying about managing physical hardware.
- **Cost-Effectiveness:** Blob Storage offers a pay-as-you-go pricing model, where you only pay for the storage you use. This can be more cost-effective than managing and maintaining on-premises storage infrastructure.
- **Durability and Availability:** Azure Blob Storage provides high durability and availability for your data. It replicates your data to multiple data centers and availability zones, ensuring that your data remains accessible even in the face of hardware failures or other disruptions.
- **Security:** Blob Storage offers multiple layers of security, including encryption at rest and in transit, role-based access control (RBAC), shared access signatures (SAS), and integration with Azure Active Directory for identity and access management.
- **Data Lifecycle Management:** Blob Storage allows you to set up data retention policies, data archival, and automatic tiering to optimize storage costs based on the access patterns of your data.
- **Data Analytics and Big Data:** Blob Storage integrates seamlessly with other Azure services like Azure Data Lake Storage, Azure Databricks, and Azure HDInsight, enabling you to perform data analytics, big data processing, and machine learning on your stored data.
- **Content Distribution:** Blob Storage supports Content Delivery Network (CDN) integration, making it easy to distribute and deliver content, such as images, videos, and web assets, to users around the world with low latency.
- **Backup and Disaster Recovery:** Organizations can use Blob Storage to store backups, snapshots, and replicas of critical data, providing a reliable solution for disaster recovery scenarios.
- **Media and Streaming:** Azure Blob Storage is commonly used to store media files for streaming services, such as videos and audio files, due to its high performance and support for streaming protocols.
- **IoT Data Storage:** Blob Storage is suitable for storing data generated by Internet of Things (IoT) devices, sensors, and other sources, making it a foundational component for IoT solutions.
- **Integration with Applications:** Azure Blob Storage provides REST APIs, SDKs for various programming languages, and integration with popular tools, making it easy for developers to integrate storage capabilities into their applications.

Create a storage account

1. Sign in to the Azure portal at https://portal.azure.com.

2. Choose **Create a resource** in the upper left-hand corner of the Azure portal.
3. In the **Search the Marketplace** field, type **Storage account**. When **Storage account** appears in the search results, select it.
4. In the **Storage account** panel, select **Create**.
5. In the **Basics** tab, under **Project details**, make sure the correct subscription is selected and then choose to **Create new** resource group. Type **myResourceGroup** for the name.
6. Under **Instance details**, type **myStorageAccount** for the Storage account name and choose **East US** for your **Region**.
7. Under **Performance**, select **Standard**.
8. Under **Account kind**, select **StorageV2 (general purpose v2)**.
9. Under **Replication**, select **Locally-redundant storage (LRS)**.
10. Under **Access tier (default)**, select **Hot**.
11. Under **Tags**, select **Next: Review + create**.
12. On the **Review + create** tab, select **Create**.

Clean up resources

When no longer needed, you can use the Azure portal to delete the resource group, storage account, and all related resources.

1. In the Azure portal, select **Resource groups** on the far left. Then select **myResourceGroup** in the resource group list.
2. On the **myResourceGroup** page, make sure that **myStorageAccount** is selected. Then select **Delete**.
3. Enter **myResourceGroup** for **TYPE THE RESOURCE GROUP NAME**. Then select **Delete**.

Azure SQL Database

Azure SQL Database is a fully managed relational database with built-in intelligence supporting self-driving features such as performance tuning and threat alerts. Microsoft handles all patching and updating of the code base, so you can focus on data and not on database management. SQL Database delivers dynamic scalability with no downtime, built-in intelligent optimization, global scalability and availability, and advanced security options that keep your data protected.

Why we use Azure SQL Database?

We use Azure SQL Database for the following reasons:

- **Scalability:** Azure SQL Server allows you to easily scale your database resources up or down based on your application's needs. You can increase performance during peak times and reduce resources during off-peak periods.

- **High Availability:** Azure SQL Server provides built-in high availability and disaster recovery mechanisms. It supports automatic backups, geo-replication, and failover groups, ensuring your data is safe and accessible even in the event of hardware or software failures.
- **Security:** Azure SQL Server offers robust security features, including data encryption at rest and in transit, advanced threat protection, firewall rules, and identity and access management controls. This helps you protect sensitive data and meet compliance requirements.
- **Managed Service:** Azure SQL Server is a fully managed service, meaning Microsoft handles routine database management tasks such as patching, backups, and updates. This allows your team to focus on application development rather than database maintenance.
- **Global Reach:** Azure has data centers distributed around the world. You can deploy Azure SQL Server instances in different regions for improved performance and reduced latency for users in various geographic locations.
- **Integration:** Azure SQL Server seamlessly integrates with other Azure services and tools, such as Azure App Service, Azure Functions, Azure Logic Apps, Power BI, and more, enabling you to build comprehensive and integrated cloud solutions.
- **Cost Efficiency:** Azure SQL Server offers different pricing tiers to suit various budgets and performance requirements. The pay-as-you-go model allows you to pay only for the resources you use.
- **Development and Deployment Tools:** Azure SQL Server supports various development tools and languages, including SQL Server Management Studio (SSMS), Azure Data Studio, Entity Framework, and various programming languages with SQL libraries.
- **Compatibility:** Azure SQL Server is built on the SQL Server database engine, which provides compatibility with existing SQL Server applications. This makes it easier to migrate your on-premises databases to the cloud.
- **Advanced Capabilities:** Azure SQL Server offers advanced features like in-memory processing, columnstore indexes, and support for machine learning services, enabling you to build high-performance and data-driven applications.

Create a SQL database

1. Sign in to the Azure portal at https://portal.azure.com.
2. Choose **Create a resource** in the upper left-hand corner of the Azure portal.
3. In the **Search the Marketplace** field, type **SQL database**. When **SQL database** appears in the search results, select it.
4. In the **SQL database** panel, select **Create**.
5. In the **Basics** tab, under **Project details**, make sure the correct subscription is selected and then choose to **Create new** resource group. Type **myResourceGroup** for the name.
6. Under **Database details**, type **mySampleDatabase** for the Database name.
7. Under **Server**, select **Create new**.

8. In the **New server** panel, under **Server name**, type **mySampleServer**.
9. Under **Server admin login**, type **azureuser**.
10. Under **Password**, type **Azure123456!**.
11. Under **Location**, select **East US**.
12. Under **Allow Azure services to access server**, select **Yes**.
13. Under **Use existing data**, select **Sample**.
14. Under **Compute + storage**, select **Configure database**.
15. Under **Configure database**, select **Basic**.
16. Under **Data max size**, select **2 GB**.
17. Under **Collation**, select **SQL_Latin1_General_CP1_CI_AS**.
18. Under **Tags**, select **Next: Review + create**.
19. On the **Review + create** tab, select **Create**.

Clean up resources

When no longer needed, you can use the Azure portal to delete the resource group, SQL database, and all related resources.

1. In the Azure portal, select **Resource groups** on the far left. Then select **myResourceGroup** in the resource group list.
2. On the **myResourceGroup** page, make sure that **mySampleDatabase** is selected. Then select **Delete**.
3. Enter **myResourceGroup** for **TYPE THE RESOURCE GROUP NAME**. Then select **Delete**.

Azure IoT Hub

Azure IoT Hub is a fully managed service that enables reliable and secure bidirectional communications between millions of IoT devices and a solution back end. Azure IoT Hub is a service that you can use to implement the IoT Hub device SDKs. The SDKs provide you with the APIs and libraries that you can use to write device and back-end applications to manage IoT devices and send and receive messages.

Why we use Azure IoT Hub?

- **Device Connectivity:** Azure IoT Hub provides a centralized platform for connecting, managing, and communicating with a wide range of IoT devices, sensors, and equipment.
- **Secure Communication:** IoT Hub ensures secure and bidirectional communication between devices and the cloud using protocols like MQTT, AMQP, and HTTPs. It supports device-to-cloud and cloud-to-device messaging, enabling real-time data exchange.

- **Device Management:** IoT Hub offers robust device management capabilities, including device provisioning, configuration, firmware updates, and remote monitoring. This simplifies the management and maintenance of a large fleet of devices.
- **Scalability:** IoT Hub is designed to handle massive amounts of device data. It can automatically scale to accommodate increased device traffic and data volumes.
- **Message Routing:** IoT Hub supports message routing and filtering based on device attributes, ensuring that data is directed to the appropriate services or endpoints for processing.
- **Data Ingestion:** Azure IoT Hub acts as a data ingestion point, collecting data from devices and transmitting it to other Azure services like Azure Stream Analytics, Azure Functions, and Azure Machine Learning for further analysis and processing.
- **Integration with Azure Services:** IoT Hub seamlessly integrates with other Azure services, such as Azure Storage, Azure Cosmos DB, and Power BI, allowing you to create end-to-end IoT solutions.
- **Edge Computing:** Azure IoT Hub integrates with Azure IoT Edge, which enables you to run data processing and analytics closer to the devices, reducing latency and conserving bandwidth.
- **Security and Compliance:** IoT Hub offers features like device authentication, role-based access control (RBAC), and integration with Azure Active Directory to ensure data security and compliance with regulatory requirements.
- **Telemetry and Monitoring:** IoT Hub provides insights into device telemetry and behavior, allowing you to monitor the health, performance, and status of your devices.
- **Custom Business Logic:** You can implement custom business logic by integrating IoT Hub with Azure Functions or Azure Logic Apps to trigger specific actions based on device events or data.
- **Global Reach:** Azure IoT Hub has a global presence, enabling you to connect devices and process data from different geographic regions with low latency.
- **Hybrid Scenarios:** Azure IoT Hub supports hybrid scenarios, allowing you to connect on-premises devices to the cloud for data aggregation and analysis.

Create an IoT Hub

1. Sign in to the Azure portal at https://portal.azure.com.
2. Choose **Create a resource** in the upper left-hand corner of the Azure portal.
3. In the **Search the Marketplace** field, type **IoT Hub**. When **IoT Hub** appears in the search results, select it.
4. In the **IoT Hub** pane, select **Create**.
5. In the **Basics** tab, complete the fields as follows:
    - **Subscription:** Select the subscription to use for your hub.
    - **Resource Group:** Select the resource group to use for your hub. You can create a new resource group or use an existing one. To create a new one, select **Create new** and fill in the name you

want to use. To use an existing resource group, select that resource group. For more information, see Manage Azure Resource Manager resource groups.

- **Region:** Select the region in which you want your hub to be located. Select the location closest to you.
- **IoT Hub Name:** Enter a name for your hub. This name must be globally unique. If the name you enter is available, a green check mark appears.
- **Important:** Because the IoT hub will be publicly discoverable as a DNS endpoint, be sure to avoid entering any sensitive or personally identifiable information when you name it.

6. Select **Next: Size and scale** to continue creating your hub.
7. In the **Size and scale** tab, you can accept the default settings, which create an IoT Hub with a free F1 tier. In a production environment, you may want to choose a different tier. For more information, see Choosing the right IoT Hub tier.
8. Select **Review + create** to review your choices.
9. Select **Create** to create your new hub. Creating the hub takes a few minutes.
10. After your hub is created, open your new hub by selecting **Go to resource**.

Register a new device

1. In your IoT hub navigation menu, open **IoT devices**.
2. Select **New** to add a new device.
3. In the **Create a device** dialog, provide a name for your device. Leave **Auto-generate keys** checked. Select **Save**.
4. After the device is created, open the device from the list in the **IoT devices** pane. Copy the **Primary Connection String**.

Send telemetry from a device

1. In the Azure portal, open your IoT hub.
2. In the IoT hub navigation menu, open **IoT devices**.
3. Select the device you created earlier.
4. In the **Device details** pane, select **Message to Device**.
5. In the **Message to Device** pane, enter a message in the **Message body** field. Select **Send message**.
6. In the **Device details** pane, select **Refresh**. You should see the message you sent in the **Messages** list.

Clean up resources

When no longer needed, you can use the Azure portal to delete the resource group, IoT hub, and all related resources.

1. In the Azure portal, select **Resource groups** on the far left. Then select **myResourceGroup** in the resource group list.
2. On the **myResourceGroup** page, make sure that **myIoTHub** is selected. Then select **Delete**.
3. Enter **myResourceGroup** for **TYPE THE RESOURCE GROUP NAME**. Then select **Delete**.

Azure Functions

Azure Functions is a serverless compute service that enables you to run code on-demand without having to explicitly provision or manage infrastructure. Use Azure Functions to run a script or piece of code in response to a variety of events. Azure Functions supports triggers, which are ways to start execution of your code, and bindings, which are ways to simplify coding for input and output data.

Why we use Azure Functions?

- **Event-Driven Programming:** Azure Functions are designed for event-driven scenarios. You can write code to respond to a wide range of events, such as HTTP requests, database changes, file uploads, timers, and messages from service queues.
- **Serverless Architecture:** Azure Functions follows a serverless model, which means you don't need to worry about provisioning or managing servers. You only pay for the compute resources used during the execution of your functions.
- **Cost Efficiency:** With Azure Functions, you are billed based on the execution time and resources consumed by your functions. This pay-as-you-go pricing model can be more cost-effective than traditional infrastructure models.
- **Auto-Scaling:** Azure Functions automatically scales out based on demand. If your function experiences high traffic, Azure will automatically allocate additional resources to handle the load.
- **Microservices:** Azure Functions are well-suited for building microservices and serverless APIs. You can compose functions to create complex workflows and APIs without the need to manage a monolithic application.
- **Integration:** Azure Functions can easily integrate with other Azure services, such as Azure Storage, Azure Service Bus, Azure Cosmos DB, and more, enabling you to build end-to-end solutions.
- **Rapid Development:** Azure Functions allow you to focus on writing code without worrying about infrastructure setup or management. This accelerates development and reduces time to market.
- **Stateless Execution:** Functions are stateless by default, which simplifies development and makes it easier to scale and manage your application.
- **Event Sourcing:** Azure Functions can be used as part of an event sourcing architecture, where events are captured and processed to update application state.
- **Real-Time Processing:** Azure Functions can be used for real-time data processing, such as stream processing and event-driven analytics.

- **Continuous Integration and Deployment (CI/CD):** Azure Functions can be integrated into CI/CD pipelines, allowing you to automate deployment and updates.
- **IoT and Edge Computing:** Azure Functions can run on IoT devices and Azure IoT Edge, enabling you to process data closer to the source and reduce latency.
- **Custom Triggers and Bindings:** Azure Functions offers a variety of built-in triggers and bindings for popular Azure services, and you can also create custom triggers and bindings to connect with other services.

Create an Azure Function

1. Sign in to the Azure portal at https://portal.azure.com.
2. Choose **Create a resource** in the upper left-hand corner of the Azure portal.
3. In the **Search the Marketplace** field, type **Function App**. When **Function App** appears in the search results, select it.
4. In the **Function App** pane, select **Create**.
5. In the **Basics** tab, complete the fields as follows:
    - **Subscription:** Select the subscription to use for your function app.
    - **Resource Group:** Select the resource group to use for your function app. You can create a new resource group or use an existing one. To create a new one, select **Create new** and fill in the name you want to use. To use an existing resource group, select that resource group. For more information, see Manage Azure Resource Manager resource groups.
    - **Function App name:** Enter a name for your function app. This name must be unique across all Azure Function apps. The name you enter is also used as the default domain name. For example, if you enter **myfunctionapp**, the URL of your function app is **https://myfunctionapp.azurewebsites.net**. The URL of your function app must be unique across all Azure websites, so you may need to choose a different name.
    - **Publish:** Select **Code**.
    - **Runtime stack:** Select **.NET Core**.
    - **Version:** Select **3.1**.
    - **Region:** Select the region in which you want your function app to be located. Select the location closest to you.
    - **Operating System:** Select **Windows**.
    - **Hosting Plan:** Select **Consumption (Serverless)**.
    - **Storage:** Select **Create new**. Enter a name for a new storage account. The name must be between 3 and 24 characters in length and may contain numbers and lowercase letters only.
    - **Application Insights:** Select **Off**.

6. Select **Review + create** to review your choices. The **Validation passed** message appears.
7. Select **Create** to create your new function app. Deployment may take a few minutes.
8. After your function app is created, select **Go to resource** to open the function app.

Create a function

1. In the Azure portal, open your function app.
2. In the left menu, select **Functions**.
3. Select **New function**.
4. Select **HTTP trigger**. Then select **Create**.
5. In the **New Function** dialog, enter a name for your function. Select **Create**.
6. In the **Code + Test** tab, select **Get function URL**. Copy the URL.
7. Open a new browser tab, paste the URL into the address bar, and press Enter. You should see the message **Please pass a name on the query string or in the request body**.
8. Add **?name=** followed by your name to the end of the URL. For example, if your name is **Burak**, the URL should look like this: **https://myfunctionapp.azurewebsites.net/api/HttpTrigger1?name=Burak**. Press Enter. You should see the message **Hello Burak**.

Clean up resources

When no longer needed, you can use the Azure portal to delete the resource group, function app, and all related resources.

1. In the Azure portal, select **Resource groups** on the far left. Then select **myResourceGroup** in the resource group list.
2. On the **myResourceGroup** page, make sure that **myFunctionApp** is selected. Then select **Delete**.
3. Enter **myResourceGroup** for **TYPE THE RESOURCE GROUP NAME**. Then select **Delete**.

Azure VM with a Template

- This command lists all resource groups in your subscription.

```
Get-AzResourceGroup | Format-Table
```

- This command creates a new virtual machine named **myVMPS** in the **myRGPS** resource group. The virtual machine is created in the **East US** region. The virtual machine is connected to the **myVnetPS** virtual network and the **mySubnetPS** subnet. The virtual machine is assigned the **myNSGPS** network security group and the **myPublicIpPS** public IP address.

```
New-AzVm `
-ResourceGroupName "myRGPS" `
-Name "myVMPS" `
-Location "East US" `
-VirtualNetworkName "myVnetPS" `
-SubnetName "mySubnetPS" `
-SecurityGroupName "myNSGPS" `
-PublicIpAddressName "myPublicIpPS"
```

- This command displays the status of the virtual machine named **myVMPS**.

```
Get-AzVM -name myVMPS -status | Format-Table -autosize
```

- This command stops the virtual machine named **myVMPS**.

```
Stop-AzVM -ResourceGroupName myRGPS -Name myVMPS
```

- Review Azure Advisor recommendations for the virtual machine.
    - Azure Advisor is a free Azure service that analyzes your Azure usage and configurations and provides personalized recommendations to help you optimize your resources for high availability, security, performance, and cost.
    - All Services > Advisor > Overview > All Recommendations

---

Azure VM with a PowerShell

Same with the previous one.

Azure VM with a CLI

- Verify the resource group you are using by entering the following command.

```
az group list --output table
```

- Create a virtual machine.

```
az vm create \
--name myVMCLI \
--resource-group myRGCLI-lod33166774 \
--image UbuntuLTS \
--location EastUS2 \
```

```
--admin-username azureuser \
--admin-password Pa$$w0rd1234
```

- Retrieve information about the virtual machine you provisioned, including name, resource group, location, and status.

```
az vm show --resource-group myRGCLI-lod33166774 --name myVMCLI --show-details --
output table
```

- Stop the virtual machine.

```
az vm stop --resource-group myRGCLI-lod33166774 --name myVMCLI
```

- Review Azure Advisor recommendations for the virtual machine.
    - Azure Advisor is a free Azure service that analyzes your Azure usage and configurations and provides personalized recommendations to help you optimize your resources for high availability, security, performance, and cost.
    - All Services > Advisor > Overview > All Recommendations

---

Azure Key Vault

Azure Key Vault is a cloud service that provides a secure store for secrets. You can securely store keys, passwords, certificates, and other secrets. You can also encrypt keys and secrets using keys you've stored in Key Vault.

Why we use Azure Key Vault?

- **Secure Key Management:** Key Vault provides a secure and isolated environment for storing cryptographic keys, which are essential for encryption, decryption, and other security operations. Storing keys in Key Vault helps protect them from unauthorized access and potential compromise.
- **Secret Management:** Key Vault allows you to securely store and manage secrets, such as connection strings, API keys, passwords, and other sensitive configuration data. These secrets can be easily accessed by authorized applications and services without exposing them in code or configuration files.
- **Centralized Management:** With Key Vault, you can centralize the management of keys and secrets across your organization. This makes it easier to enforce consistent security practices and policies across different applications and services.
- **Auditing and Logging:** Key Vault provides auditing and logging capabilities, allowing you to track who accessed the keys and secrets, when they were accessed, and what operations were performed. This helps with compliance requirements and security monitoring.

- **Integration with Azure Services:** Azure Key Vault seamlessly integrates with various Azure services, such as Virtual Machines, Azure Functions, Azure App Service, Azure Kubernetes Service (AKS), and more. This integration allows these services to securely retrieve the necessary keys and secrets they require to operate.
- **Encryption and Data Protection:** By using Key Vault to manage keys and secrets, you can enhance the security of your applications by enabling encryption of data at rest and in transit. This is crucial for protecting sensitive information and meeting regulatory compliance requirements.
- **High Availability and Redundancy:** Azure Key Vault offers high availability and redundancy features, ensuring that your keys and secrets are always accessible even in the case of datacenter outages or other disruptions.
- **Role-Based Access Control (RBAC):** Key Vault supports RBAC, which means you can control who has access to keys and secrets and define fine-grained permissions. This helps enforce the principle of least privilege and reduces the risk of unauthorized access.
- **Managed Certificates:** Azure Key Vault can also be used to manage SSL/TLS certificates, making it easier to provision, renew, and manage certificates for your applications and services.
- **Developer and DevOps Efficiency:** Using Key Vault simplifies the process of managing sensitive information for developers and DevOps teams. It reduces the need to hardcode or distribute sensitive data, making applications more maintainable and secure.

Create a Key Vault

1. In the Azure portal, select **Create a resource**.
2. In the **New** dialog, select **Key Vault**.
3. In the **Key Vault** dialog, select **Create**.
4. In the **Create key vault** dialog, enter the following values:
    - **Subscription**: Select your Azure subscription.
    - **Resource group**: Select **myResourceGroup**.
    - **Key vault name**: Enter **myKeyVault**.
    - **Region**: Select the region closest to you.
    - **Pricing tier**: Select **Standard**.
    - **Soft delete**: Select **Enable**.
    - **Purge protection**: Select **Enable**.
    - **Access policies**: Select **None selected**.
5. Select **Review + create**.
6. Select **Create**.

Create a secret

1. In the Azure portal, select **Resource groups** on the far left. Then select **myResourceGroup** in the resource group list.
2. On the **myResourceGroup** page, select **myKeyVault**.
3. On the **myKeyVault** page, select **Secrets** under **Settings**.
4. On the **Secrets** page, select **Generate/Import**.
5. On the **Create a secret** page, enter the following values:
    - **Upload options**: Select **Manual**.
    - **Name**: Enter **mySecret**.
    - **Value**: Enter **mySecretValue**.
    - **Content type**: Select **text/plain**.
    - **Enabled**: Select **Yes**.
6. Select **Create**.

Retrieve a secret

1. In the Azure portal, select **Resource groups** on the far left. Then select **myResourceGroup** in the resource group list.
2. On the **myResourceGroup** page, select **myKeyVault**.
3. On the **myKeyVault** page, select **Secrets** under **Settings**.
4. On the **Secrets** page, select **mySecret**.
5. On the **mySecret** page, select **Show secret value**.
6. Copy the secret value to the clipboard.

Delete a secret

1. In the Azure portal, select **Resource groups** on the far left. Then select **myResourceGroup** in the resource group list.
2. On the **myResourceGroup** page, select **myKeyVault**.
3. On the **myKeyVault** page, select **Secrets** under **Settings**.
4. On the **Secrets** page, select **mySecret**.
5. On the **mySecret** page, select **Delete**.
6. Select **Yes** to confirm the deletion.

Azure Secure Network Traffic

Azure Secure Network Traffic is a service that provides secure network connectivity between Azure resources. It enables you to connect virtual machines, Azure Kubernetes Service (AKS) clusters, Azure App Service, and other Azure services privately without exposing them to the public internet.

Why we use Azure Secure Network Traffic?

- **Data Protection:** Secure network traffic helps protect sensitive data from interception, eavesdropping, and unauthorized access as it traverses the network. This is crucial for maintaining the confidentiality of sensitive information.
- **Compliance Requirements:** Many industries and regulatory standards (such as GDPR, HIPAA, and PCI DSS) require organizations to implement security measures to protect data during transmission. Secure network traffic helps you meet these compliance requirements.
- **Mitigation of Threats:** Secure network traffic helps guard against various threats, including man-in-the-middle attacks, packet sniffing, and data tampering. By encrypting the traffic, you can ensure that even if a malicious actor intercepts the data, they cannot decipher its contents.
- **Authentication and Authorization:** Secure network traffic often involves mechanisms for mutual authentication and authorization, ensuring that only trusted parties can communicate with each other. This prevents unauthorized entities from accessing your resources.
- **Vulnerability Prevention:** Secure network traffic can help prevent vulnerabilities like session hijacking or replay attacks by ensuring that data cannot be easily intercepted, altered, or replayed by attackers.
- **Data Integrity:** Secure network traffic measures, such as transport layer security (TLS) or secure sockets layer (SSL) encryption, help ensure the integrity of data during transmission. This prevents tampering or modification of data in transit.
- **Public Network Protection:** In cloud environments like Azure, where resources may be distributed across different geographical regions and data centers, securing network traffic becomes even more critical to protect data as it traverses potentially untrusted public networks.
- **Securing Multi-Tier Architectures:** In complex multi-tier architectures, secure network traffic ensures that communication between different tiers, such as web servers, application servers, and databases, is encrypted and protected.
- **Secure Communication with APIs:** Many applications communicate with external services and APIs. Secure network traffic ensures that these communications are encrypted and secure, preventing potential exposure of sensitive data.
- **Zero Trust Security:** Implementing secure network traffic aligns with the principles of zero trust security, where every interaction and communication is treated as potentially untrusted. This approach enhances overall security posture by minimizing the attack surface.

Azure Manage Access with RBAC

Azure Role-Based Access Control (RBAC) is a service that provides fine-grained access management of Azure resources. It enables you to grant users the specific permissions they need to perform their jobs, while preventing unauthorized access to resources.

Why we use Azure Manage Access with RBAC?

- **Granular Access Control:** RBAC provides fine-grained control over permissions. You can define exactly what actions (read, write, delete, etc.) a user or group can perform on specific Azure resources. This ensures that users have the minimum required permissions to perform their tasks without granting unnecessary access.
- **Security and Compliance:** RBAC helps you enforce security and compliance policies by ensuring that only authorized users can access and manage resources. This reduces the risk of unauthorized access, data breaches, and configuration errors.
- **Reduced Attack Surface:** RBAC helps minimize the attack surface by limiting access to resources. Users and applications only have access to the resources they need, reducing the potential impact of security breaches.
- **Delegation of Responsibility:** RBAC allows you to delegate specific administrative tasks to different teams or individuals without granting them full administrative access. This enables separation of duties and helps prevent accidental or intentional misuse of privileges.
- **Dynamic Access Management:** RBAC enables dynamic access management by allowing you to assign and revoke permissions as needed. As roles change or employees leave the organization, you can easily adjust access without making extensive changes to resource configurations.
- **Simplified Access Management:** RBAC simplifies access management by providing predefined roles with well-defined permissions. Instead of manually configuring permissions for each user, you can assign users to appropriate roles, which saves time and reduces the risk of misconfiguration.
- **Audit and Monitoring:** RBAC provides an audit trail of who performed which actions on resources. This helps with compliance requirements, troubleshooting, and security monitoring.
- **Integration with Identity Providers:** RBAC integrates seamlessly with Azure Active Directory (Azure AD) and other identity providers, allowing you to use existing user and group definitions for access management.
- **Scalability:** As your organization's Azure environment grows, RBAC scales easily. You can continue to manage access efficiently without resorting to ad-hoc access controls.
- **Application-Specific Access:** RBAC is not limited to human users; it can also be used to manage access for applications and services that interact with Azure resources. This ensures that applications have the necessary permissions to function correctly without unnecessary access.

Create a Azure resource group

1. In the Azure portal, select **Create a resource** in the upper left-hand corner.
2. In the **Search the Marketplace** field, enter **Resource group**. Select **Resource group** from the results.

3. Select **Create**.
4. On the **Create a resource group** page, enter the following values:
    - **Subscription**: Select your subscription.
    - **Resource group**: Enter **myResourceGroup**.
    - **Region**: Select **East US**.
5. Select **Review + create**.
6. Select **Create**.

Create a Azure role assignment

1. In the Azure portal, select **Resource groups** on the far left. Then select **myResourceGroup** in the resource group list.
2. On the **myResourceGroup** page, select **Access control (IAM)** under **Settings**.
3. On the **Access control (IAM)** page, select **Add role assignment**.
4. On the **Add role assignment** page, enter the following values:
    - **Role**: Select **Contributor**.
    - **Assign access to**: Select **User, group, or service principal**.
    - **Select**: Select your user account.
5. Select **Save**.

Delete a Azure role assignment

1. In the Azure portal, select **Resource groups** on the far left. Then select **myResourceGroup** in the resource group list.
2. On the **myResourceGroup** page, select **Access control (IAM)** under **Settings**.
3. On the **Access control (IAM)** page, select **Contributor**.
4. On the **Contributor** page, select **Remove**.
5. Select **Yes** to confirm the deletion.

Azure Manage Resource Locks

Azure Resource Locks is a service that provides fine-grained access management of Azure resources. It enables you to grant users the specific permissions they need to perform their jobs, while preventing unauthorized access to resources.

Why we use Azure Manage Resource Locks?

- **Prevent Deletion:** By applying a delete lock on a resource, you can prevent it from being deleted. This is particularly useful for protecting important resources that should not be removed accidentally.
- **Prevent Modification:** Applying a read-only lock on a resource prevents any modifications or updates to the resource's properties. This ensures that the resource's configuration remains unchanged.
- **Data Protection:** Resource Locks help protect against data loss or corruption that could occur due to accidental or unauthorized actions. For example, if a critical virtual machine or database is accidentally deleted, it could lead to data loss and operational disruptions. Applying a lock prevents such actions.
- **Compliance and Governance:** Resource Locks support compliance requirements and governance policies. They can help enforce separation of duties by allowing certain teams or individuals to manage resources while preventing them from making irreversible changes.
- **Emergency Situations:** In emergency situations, where an immediate action might be required, Resource Locks can prevent hasty decisions that could lead to further issues. They provide a buffer period to assess the situation before making changes.
- **Collaboration:** In collaborative environments, multiple users might have access to Azure resources. Resource Locks help maintain consistency and prevent accidental changes that one user might make without knowing the actions of others.
- **Testing and Development:** In development and testing environments, you might want to prevent changes to resources that are being used for testing purposes, ensuring that the environment remains stable for all team members.
- **Resource Lock Hierarchy:** Azure Resource Locks can be applied at different levels of the resource hierarchy, including the subscription, resource group, and individual resource levels. This provides flexibility in how you protect your resources based on their criticality.

Create a Azure resource group

1. In the Azure portal, select **Create a resource** in the upper left-hand corner.
2. In the **Search the Marketplace** field, enter **Resource group**. Select **Resource group** from the results.
3. Select **Create**.
4. On the **Create a resource group** page, enter the following values:
    - **Subscription**: Select your subscription.
    - **Resource group**: Enter **myResourceGroup**.
    - **Region**: Select **East US**.
5. Select **Review + create**.
6. Select **Create**.

Create a Azure resource lock

1. In the Azure portal, select **Resource groups** on the far left. Then select **myResourceGroup** in the resource group list.
2. On the **myResourceGroup** page, select **Locks** under **Settings**.
3. On the **Locks** page, select **Add**.
4. On the **Add lock** page, enter the following values:
    - **Lock name**: Enter **myLock**.
    - **Lock type**: Select **Delete**.
    - **Notes**: Enter **Prevent accidental deletion**.
5. Select **OK**.

Delete a Azure resource lock

1. In the Azure portal, select **Resource groups** on the far left. Then select **myResourceGroup** in the resource group list.
2. On the **myResourceGroup** page, select **Locks** under **Settings**.
3. On the **Locks** page, select **myLock**.
4. On the **myLock** page, select **Delete**.
5. Select **Yes** to confirm the deletion.

Azure Resource Tagging

Azure Resource Tags is a service that provides fine-grained access management of Azure resources. It enables you to grant users the specific permissions they need to perform their jobs, while preventing unauthorized access to resources.

Why we use Azure Resource Tagging?

- **Resource Categorization:** Tags enable you to categorize resources based on attributes such as department, project, environment (e.g., production, development), owner, business unit, or any other custom classification that makes sense for your organization. This helps you organize and manage resources more effectively.
- **Cost Allocation and Tracking:** Tags play a crucial role in cost allocation and tracking. By tagging resources with relevant attributes, you can easily identify which resources are associated with specific projects or departments. This information is essential for accurately allocating costs and optimizing spending.
- **Budget Management:** Tags allow you to set up and manage budgets based on specific criteria, such as tags associated with cost centers or projects. This helps you monitor and control spending by providing insights into how resources are being used and by whom.

- **Resource Management:** With tagged resources, you can filter and search for specific sets of resources based on their tags. This makes it easier to manage, monitor, and take actions on groups of resources that share similar attributes.
- **Automation and Policy Enforcement:** Tags can be used in conjunction with Azure Policy and Azure Automation to enforce specific policies and automation tasks based on resource tags. For example, you can automatically enforce naming conventions or security policies based on tags.
- **Governance and Compliance:** Tagging helps enforce governance and compliance requirements by enabling you to classify resources according to specific guidelines or regulatory standards. This allows you to easily demonstrate compliance during audits.
- **Resource Lifecycle Management:** Tags can also be useful for managing the lifecycle of resources. For example, you can use tags to identify resources that are part of a temporary project and need to be cleaned up after the project's completion.
- **Reporting and Analysis:** Tags provide valuable insights into resource usage patterns, usage trends, and resource distribution across your organization. This information can be used for reporting, analysis, and strategic decision-making.
- **Cross-Platform Consistency:** Tags can be applied consistently across different Azure services, making it easier to track and manage resources regardless of the services they belong to.
- **Customization and Flexibility:** Azure Resource Tagging is highly customizable, allowing you to define tags that align with your organization's structure and needs. You can create and use tags that make sense for your specific use cases.

Azure Create Policy

Azure Policy is a service that provides fine-grained access management of Azure resources. It enables you to grant users the specific permissions they need to perform their jobs, while preventing unauthorized access to resources.

Why we use Azure Create Policy?

- **Enforce Compliance and Governance:** Azure Policy helps you ensure that your resources adhere to your organization's standards, industry regulations, and best practices. It helps maintain a consistent and compliant environment by preventing the creation of resources that do not meet defined criteria.
- **Security Enhancements:** Azure Policy can enforce security controls by preventing the deployment of resources with certain configurations that might introduce security vulnerabilities. This ensures that security best practices are followed, reducing the risk of data breaches and unauthorized access.
- **Resource Consistency:** Azure Policy helps maintain consistent configurations across resources. It ensures that resources are created and configured according to predefined guidelines, reducing the chances of misconfigurations and operational issues.
- **Customizable Policies:** You can create custom policies to meet the specific needs of your organization. These policies can enforce naming conventions, tag requirements, access controls, network configurations, and more.

- **Tagging and Organization:** Azure Policy can enforce the use of specific tags on resources, making it easier to categorize, track, and manage resources for cost allocation, reporting, and resource lifecycle management.
- **Cost Management:** By enforcing policies related to cost management, you can control spending by preventing the creation of expensive or unnecessary resources. This is particularly useful for controlling resource provisioning and usage in large organizations.
- **Adherence to Architectural Guidelines:** Azure Policy can enforce architectural best practices, ensuring that resources are deployed in alignment with your organization's preferred architectural patterns.
- **Centralized Management:** Azure Policy provides a centralized way to manage and enforce policies across multiple subscriptions and resource groups, simplifying policy management and ensuring consistency.
- **Automation and Remediation:** Azure Policy can automatically remediate non-compliant resources. This means that if a resource is created or modified in a way that violates a policy, Azure Policy can automatically correct the configuration to bring it into compliance.
- **Integration with DevOps:** Azure Policy can be integrated into your DevOps processes to enforce policy compliance during resource deployment and updates. This helps ensure that policies are applied throughout the resource lifecycle.
- **Auditing and Reporting:** Azure Policy provides auditing and reporting capabilities, allowing you to track policy compliance, violations, and remediation activities. This information is valuable for internal audits and regulatory compliance.

Azure Tools

Azure Tools is a service that provides fine-grained access management of Azure resources. It enables you to grant users the specific permissions they need to perform their jobs, while preventing unauthorized access to resources.