Microsoft Azure for .NET Developers

HELLO!

I am Trevoir Williams

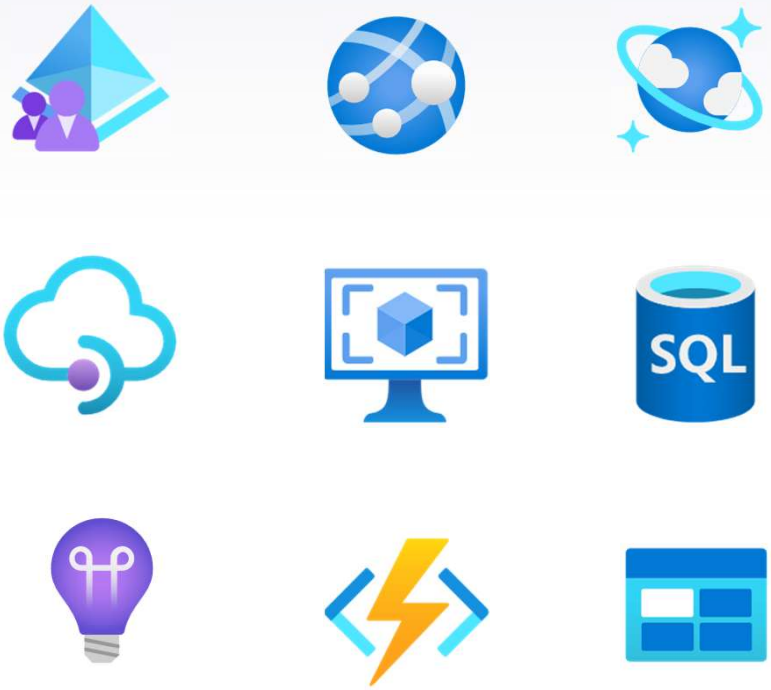
Software Engineer | Lecturer

About This Course

- This guide is designed for developers and architects starting their journey into Microsoft Azure.
- Explores the ins and outs of Microsoft Azure and teaches how to use the best services for different application scenarios.
- Reviews website, database, and desktop and mobile application integrations with various Azure services.
- Demonstrates how Microsoft Azure does the heavy infrastructural lifting for you.

Learning Outcomes

- Learn Microsoft Azure
- Different Cloud Hosting Models
  - IaaS | PaaS | SaaS
- Azure Resource Manager
  - Azure Portal | Powershell | CLI
- Integrating Azure Services
  - Azure App Services (Web, Functions, etc.)
  - Storage (SQL, CosmosDB, BLOB, etc.)
  - Events and Messaging (Service Bus, etc.)
  - Security (Azure AD, Key Vault)



Assumptions

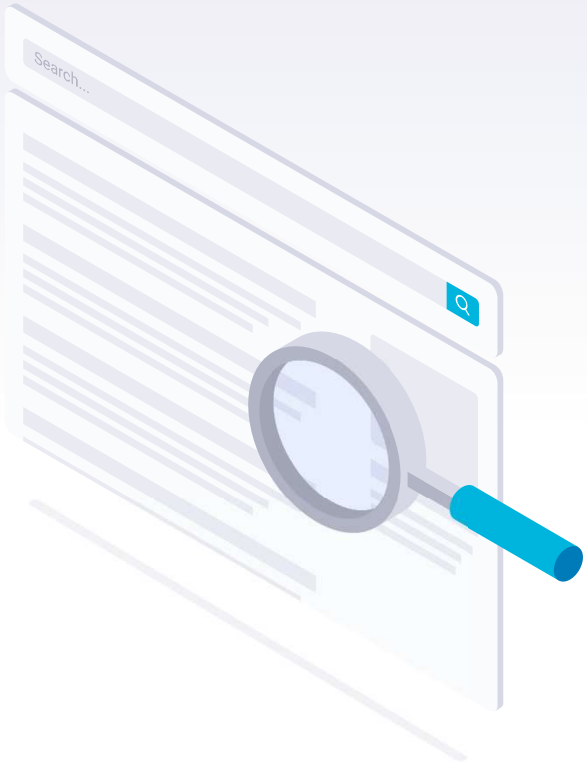
- You are already a Developer and have a working knowledge of development techniques.
- You have a working knowledge of databases and virtualization
- You have a working knowledge of application security

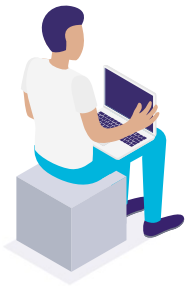
Development Environment

- Visual Studio 2022 (Windows / macOS)
  - Visual Studio Code (All)
  - .NET 6 / 7 SDK (Future Proof)
- Azure PowerShell and CLI (All)
- Docker (All)
- Azure Data Studio (All)

Welcome!

Let’s get Started





What is Microsoft Azure?

What is Microsoft Azure?

- Microsoft Azure is Microsoft's cloud computing platform, providing several services for development and hosting.
- Azure enables the rapid development of solutions and is a cloud hosting alternative to on-prem environments.
- Offers compute, storage, network, and application services, allowing you to focus on building great solutions

Azure Services

- Azure includes many services in its cloud computing platform.
- **Compute services** - Virtual Machines—both Linux and Windows, Cloud Services, App Services (Web Apps, Mobile Apps, Logic Apps, API Apps, and Function Apps), Container Service.

Azure Services

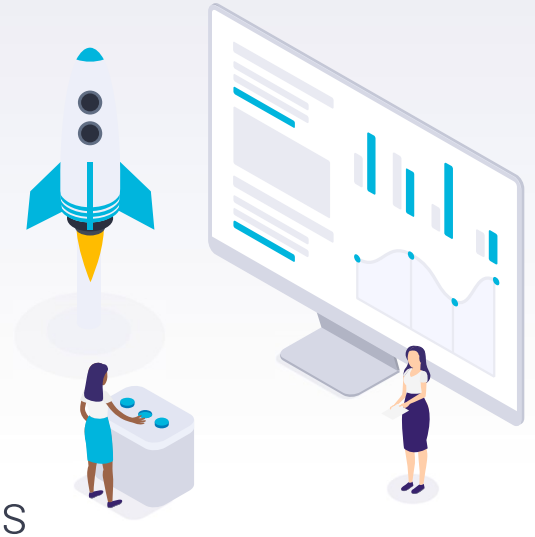
- **Data services** – Microsoft Azure Storage (BLOB, Queue, Table, and Azure Files services), Azure SQL Database, DocumentDB, and the Redis Cache.
- **Application services** – Azure Active Directory (Azure AD), Service Bus for connecting distributed systems, HDInsight for big data, Azure Scheduler, and Azure Media Services.

Azure Services

- **Network services** –Virtual Networks, ExpressRoute, Azure DNS, Azure Traffic Manager, and the Azure Content Delivery Network.
- It is worthwhile to have some understanding of the different services available, and how you might use them to improve your application.

Getting Started with Azure

Getting Started with Azure



- Sign up for an **Azure free account** at https://azure.microsoft.com/en-us/free/
  - This includes credits to explore paid Azure services and over 25 services you can use for free forever.
  - Start with $200 Azure credit
  - 30 Day trial
  - Flexible **Pay-As-You-Go** subscription model

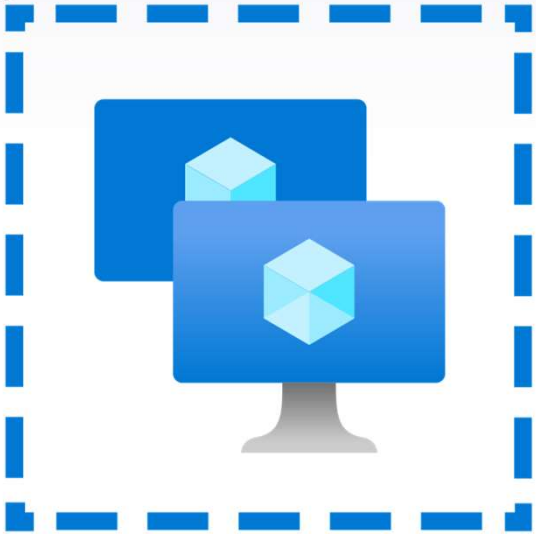
Different Cloud Hosting Options

Benefits of Microsoft Azure?

- Azure provides the flexibility to set up development and test configurations quickly.
- Azure allows you to start with low upfront costs and scale rapidly as needed.
- Offers an alternative to physical infrastructure needs.
  - IaaS – Infrastructure as a Service
  - PaaS – Platform as a Service
  - SaaS – Software as a Service

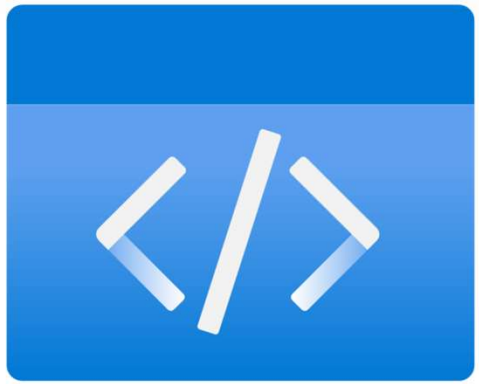
IaaS: Infrastructure as a service

- With IaaS, Azure runs and manages server farms running virtualization software, enabling you to create VMs that run on the vendor's infrastructure.
- Support for Windows or Linux VMs on which you can install anything. want on it.
- Azure provides the ability to set up virtual networks, load balancers, and storage and to use many other services that run on its infrastructure.
- No control over the hardware or virtualization software, but you do have control over almost everything else.



PaaS: Platform as a service

- Simply deploy your application into an optimized application-hosting environment provided by Azure.
- This frees developers from infrastructure management, so we can focus on development.
- Azure provides several PaaS compute offerings:
  - Web Apps feature in Azure App Service and
  - Azure Cloud Services (web and worker roles).
- Developer friendly alternative to IaaS



SaaS: Software as a service

- SaaS is software that is already hosted and managed for the end customer.
- It is based on a multitenant architecture, where a single version of the application is used for all customers.
- SaaS software typically is licensed through a monthly or annual subscription.
- Examples of SaaS:
  - Office 365
  - WordPress
  - Your next cloud hosted solution!

Section Review

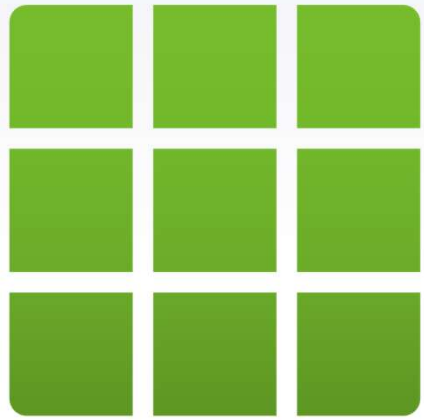
Azure in Review

- Review of Microsoft Azure
- Benefits of using Azure
- Getting Started With Azure
  - Create free account
- Cloud Hosting Models
  - IaaS
  - PaaS
  - SaaS

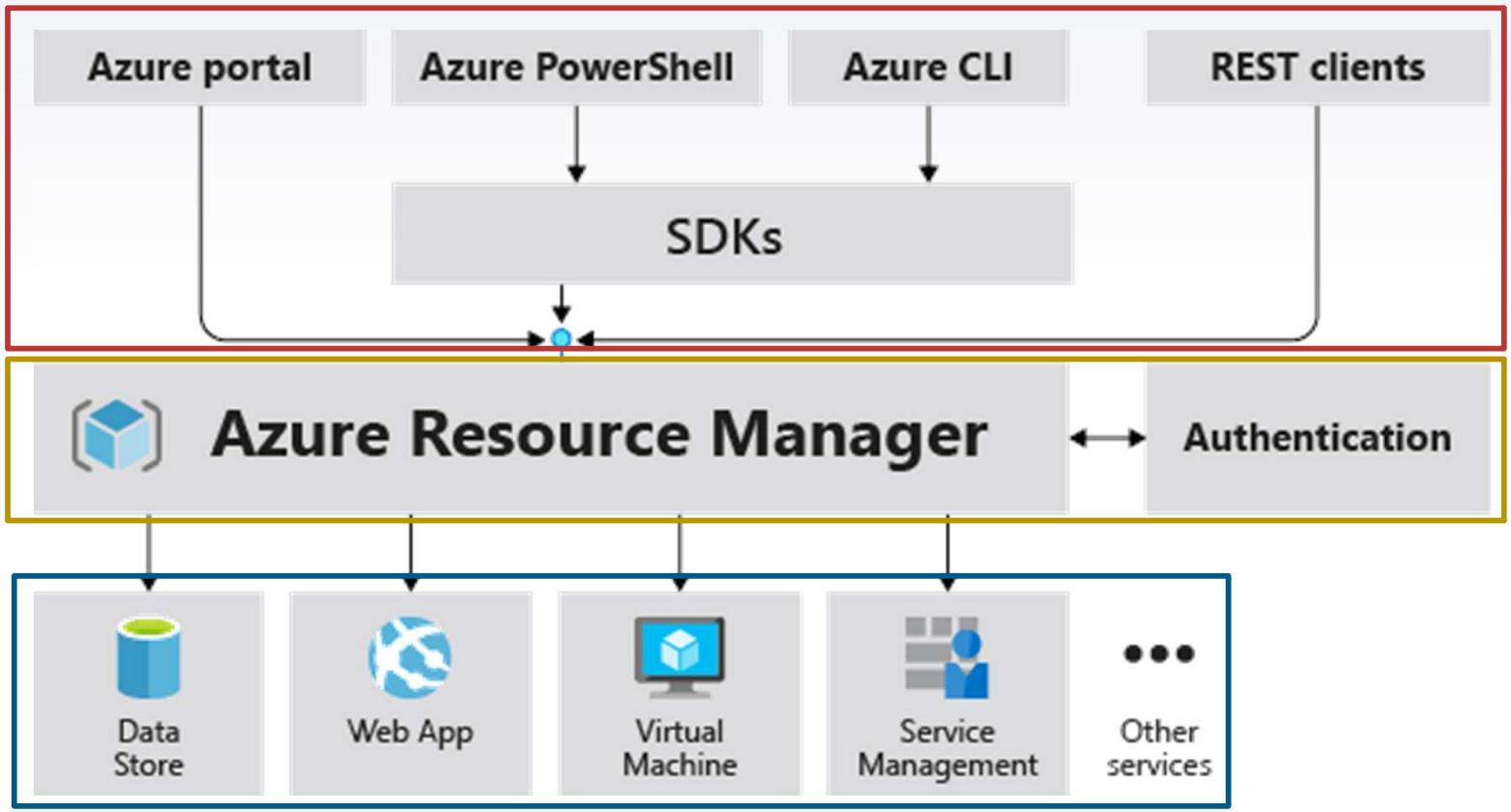
What is Azure Resource Manager?

Resource Manager

- Azure Resource Manager is the deployment and management service for Azure.
- It provides a management layer that enables you to create, update, and delete resources in your Azure account.
- Features management provides access control, locks, and tags, to secure and organize your resources after deployment.
- Underlying API for managing resources

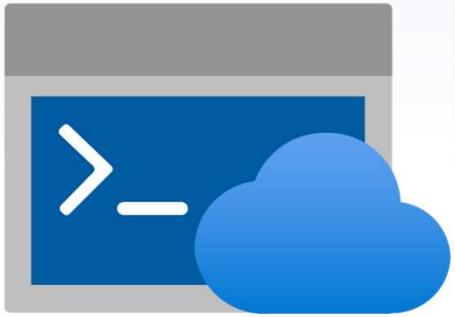


Resource Manager



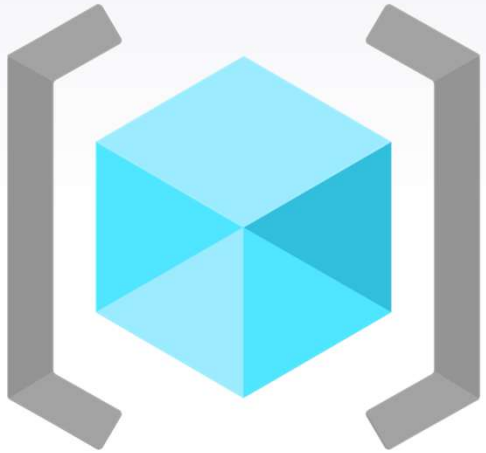
ARM Clients and Interfaces

- Commands through any of the Azure APIs, tools, or SDKs, are handled through the same API. This ensures consistency.
- **Azure Portal** – Web-based user interface that allows us to manage our resources and services.
- **Azure PowerShell and CLI** – A client that allows us to author scripts to automate resource management tasks
- **REST Clients** – Consume the underlying API and allow for custom resource management solutions



Key Terms

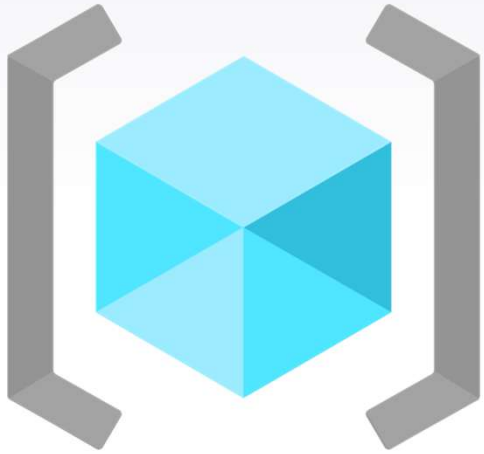
- **resource** - An item or service that can be provisioned in Azure. Virtual machines, storage accounts, web apps, databases, and virtual networks are examples of resources.



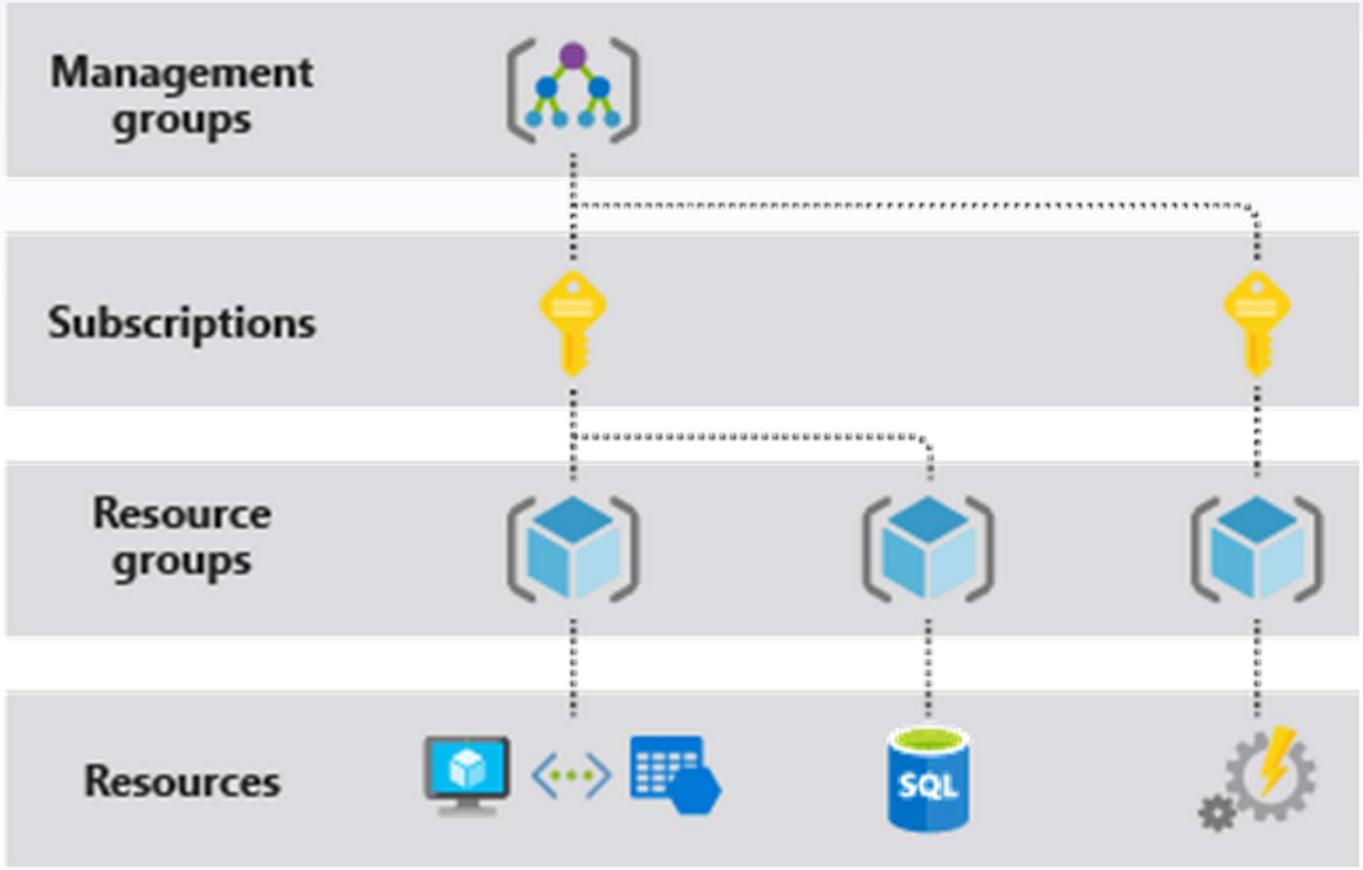
- **resource group** - A container with related resources for an Azure solution. A resource group includes those resources that you want to manage as a group.

Key Terms

- **declarative syntax** - Syntax that lets you state what you want to provision, without having to write the sequence of programming commands to create it.
  - ARM templates - A JSON file that defines one or more resources to deploy to a resource group, subscription, management group, or tenant.
  - Bicep - A file for declaratively deploying Azure resources. Optimized for Infrastructure as code delivery.



Resource Group Scope



Create Resources via Azure Portal

Understanding Azure PowerShell

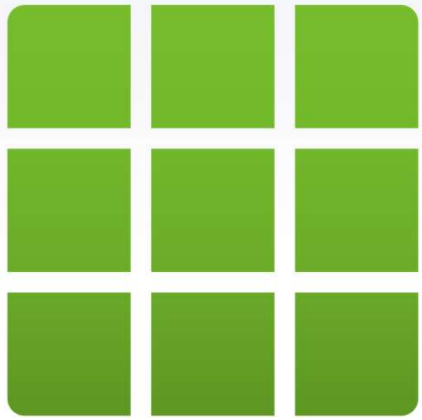
Understanding Azure CLI

Create Resources via Azure Portal

Section Review

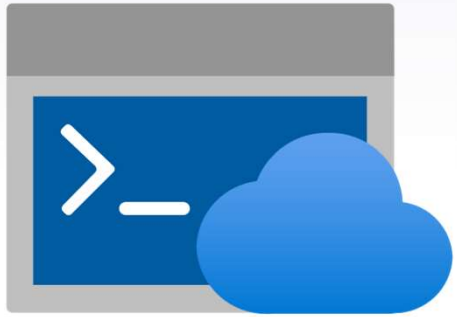
Azure Portal

- Easy
- User friendly
- Fastest way to manage resources
- Difficult for automation



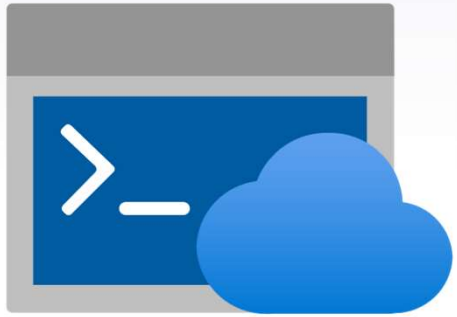
Azure PowerShell

- Extends PowerShell Commands
- Easy to access in Portal
- Easy to setup on local PC
- Easy to automate resource deployments
- Big learning curve



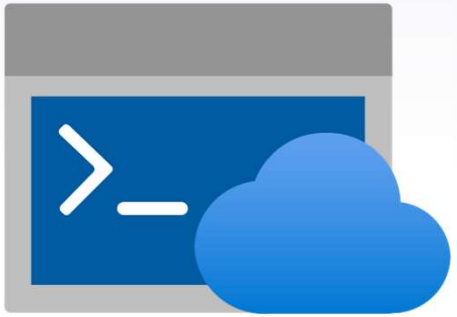
Azure CLI

- Native Azure Bash syntax
- Available in all command line interfaces once installed
- Easier to understand than PowerShell (debatable)
- Return is in JSON, so easy to integrate into existing dashboards and tools
- Big Learning Curve



ARM Clients and Interfaces

- **Azure Portal** – Web-based user interface that allows us to manage our resources and services.
- **Azure PowerShell and CLI** – A client that allows us to author scripts to automate resource management tasks
- Can be used to provision any Azure resource, with the same degree of accuracy.



Section Review

Azure App Service

Explore Azure App Service

- Learn Azure App Service key components
- Understand different hosting models
- Learn Web App deployment techniques
- Explore authentication and authorization
- Managing Azure App Service using CLI

Examining Azure App Service

- HTTP Based service for hosting web applications
  - REST APIs, back-ends, Web Apps, etc.
  - .NET/.NET Core, NodeJS, Java, PHP, Python
- Supports autoscaling to adjust to request loads
- Supports Windows and Linux runtimes
- Supports automated and manual deployment models
- PaaS offering

Reviewing Azure App Service

- Deployments
  - Manual - Azure CLI, Publish with UI Tools
  - Automated – CI/CD (GitHub Actions)
- App Configuration
- Monitoring and Logging
  - Log Stream and Application Insights (Open Telemetry)
  - Auto-Healing and Health Checks

Reviewing Azure App Service

- App Service Plans
  - Deployment Slots
  - Auto-Scaling
- Authentication and Authorization
  - Little to no code required

Azure SQL

Explore Azure SQL

- Learn Azure SQL Offerings
- Learn to create an Azure SQL Instance
- Learn how to connect an ASP.NET Core Web App
- Learn how to connect to an Azure App Service
- Understand key administration tasks

Environment and Tools

- SQL Server Management Studio
- Azure Data Studio
- Visual Studio
  - ASP.NET Core
  - Entity Framework
- Azure App Service

Azure SQL

Examining Azure SQL

- Hosted relational database-as-a-service (DBaaS). Falls into the *Platform-as-a-Service (PaaS)* category.
- Best for applications that need the latest stable SQL Server features.
- Requires little to no administration.
- Fully managed SQL Server database engine and based on the latest stable Enterprise Edition of SQL Server.

Azure SQL Deployment Models

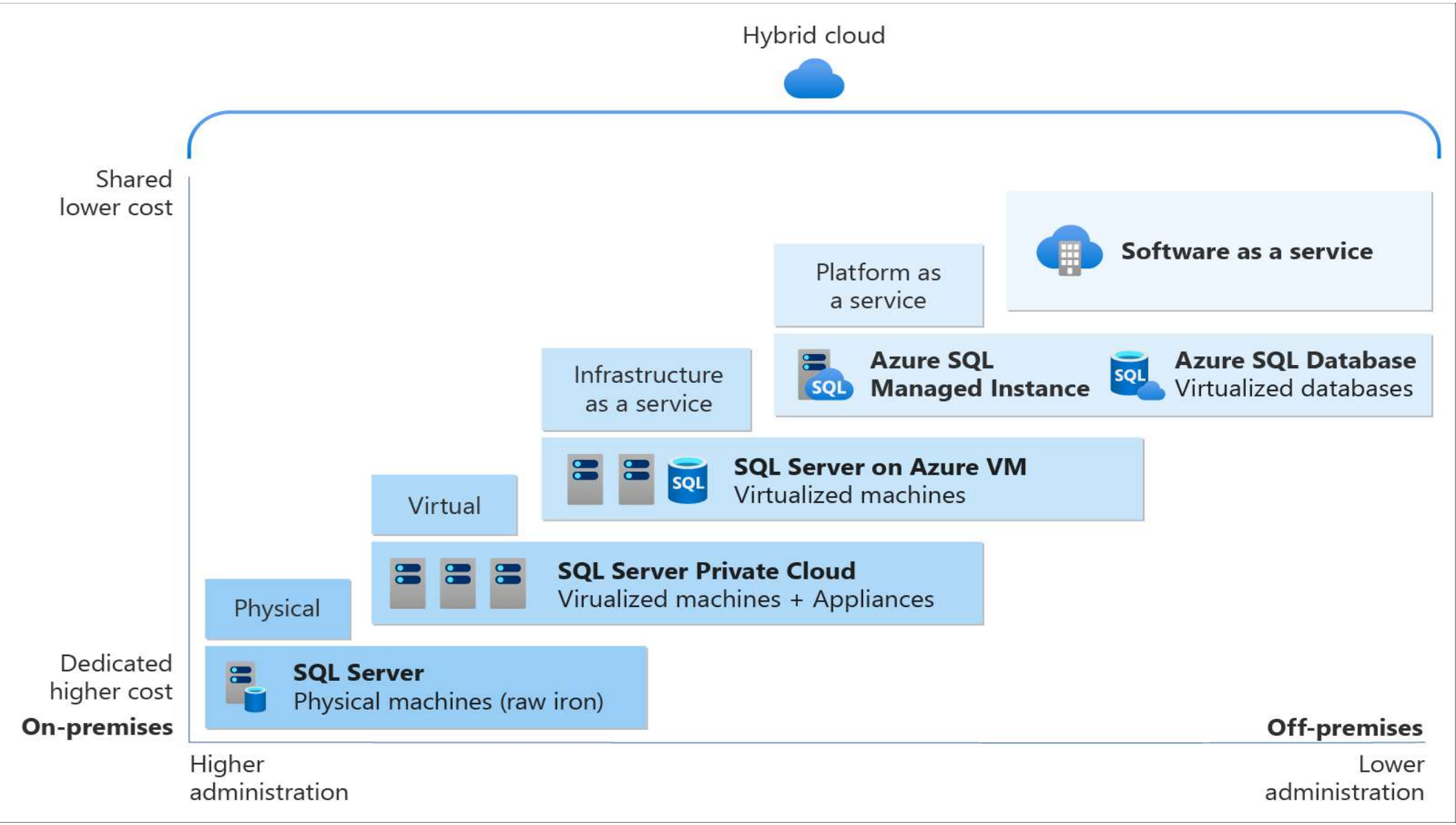
- Single Database
  - Resources are isolated and managed by a logical SQL Server
  - Optimized for cloud-applications
  - Auto-scaling, auto-pause, and high availability built-in
- Elastic pool
  - Collection of database-sharing resources
  - Single databases can be imported
  - Optimized for multi-tenant SaaS applications
  - Cost-effective solution for managing multiple databases with different usage patterns.

Benefits of Azure SQL

- Compatible with your favorite tools, including SQL Server Management Studio and Entity Framework.
- Databases in Azure SQL Database are reliable and robust, offering 99.99 percent uptime.
- Supports Geo-replication, replicating in real-time
- Features built-in encryption at rest and in transit
- Automatic auditing
- Automatic database tuning

Other Azure SQL Offerings

- Azure SQL Managed Instance
  - *Platform-as-a-Service (PaaS),*
  - Best for most migrations to the cloud.
  - SQL Managed Instance is a collection of system and user databases
  - lift-and-shift ready.
- SQL Server on Azure VM
  - *Infrastructure-as-a-Service (IaaS)*
  - Run SQL Server inside a virtual machine (VM) in Azure.


Hybrid cloud
Shared
lower cost
Software as a service
Platform as
a service
Azure SQL
Managed Instance
Azure SQL Database
Virtualized databases
Infrastructure
as a service
SQL Server on Azure VM
Virtualized machines
Virtual
SQL Server Private Cloud
Virualized machines + Appliances
Physical
SQL Server
Physical machines (raw iron)
Dedicated
higher cost
On-premises
Off-premises
Higher
administration
Lower
administration

MySQL, PostgreSQL, and MariaDB

- Azure provides **MySQL**, **PostgreSQL**, and **MariaDB** managed databases
- Just spin them up and don't have to worry about any of the underlying infrastructure.
- Just like Azure SQL Database and Azure Cosmos DB, these databases are universally available, scalable, highly secure, and fully managed.
- Easily replace your on-prem MySQL, PostgreSQL, and MariaDB databases with the cloud versions
- Enjoy the advantage of having it run fully managed in the cloud.

Azure SQL Review

Azure SQL Review

- How to publish Azure SQL
- Different Hosting options
- Learn to connect to Azure SQL with development tools
- Learn how to connect via Entity Framework Core
- Learn how to add Connection Strings in Azure Web App

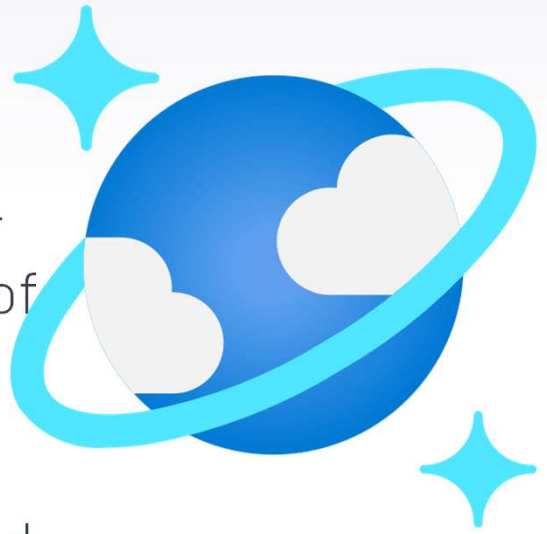
Azure Cosmos DB

Explore Azure Cosmos DB

- Learn Azure Cosmos DB
- Understand Azure Cosmos DB APIs
- Understand NoSQL Databases
- Use Azure Cosmos DB Local Emulator
- Perform CRUD operations with Cosmos Client with Blazor App

Explore Azure Cosmos DB

- Azure Cosmos DB is a fully managed NoSQL and relational database for modern app development.
- Perfect for applications that need to respond in real-time to significant changes and increasing volumes of data.
- Azure Cosmos DB takes database administration off your hands with automatic management, updates and patching



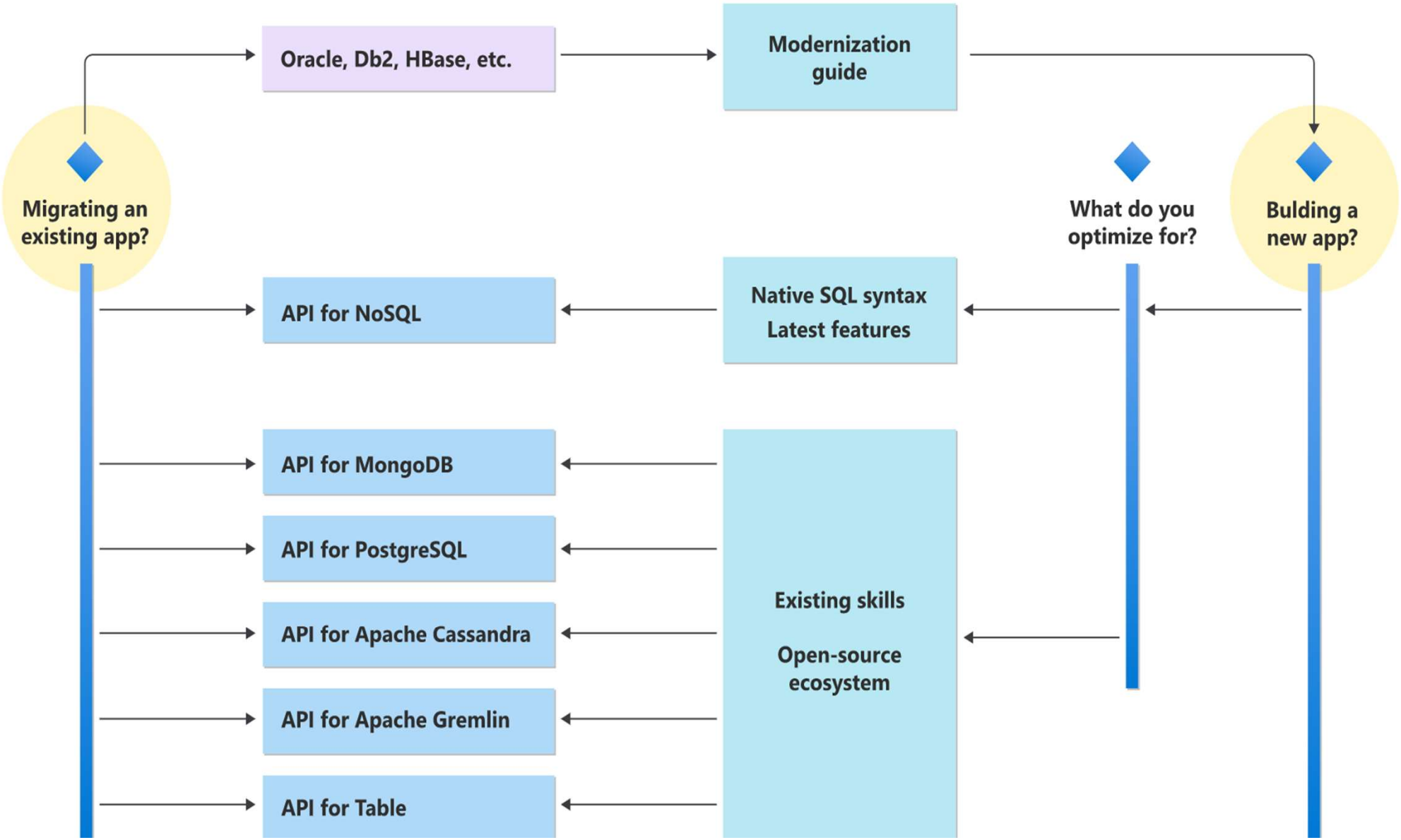
Azure Cosmos DB

Azure Cosmos DB APIs

- Azure Cosmos DB offers multiple database APIs.
- Offers storage options using documents, key-value, graph, and column-family data models.
- Azure Cosmos DB can emulate various other database technologies without the overhead of management and scaling approaches.
- All APIs offer automatic scaling of storage and throughput, flexibility, and performance guarantees.

Azure Cosmos DB APIs

- Offers multiple database APIs,
  - NoSQL - Recommended for new applications
  - MongoDB - Best for migrations
  - PostgreSQL - Best for migrations
  - Cassandra - Best for migrations
  - Gremlin - Best for migrations
  - Table – Key/value storage. Best for Migrations


Oracle, Db2, HBase, etc.
Modernization guide
Migrating an existing app?
What do you optimize for?
Bulding a new app?
API for NoSQL
Native SQL syntax
Latest features
API for MongoDB
API for PostgreSQL
API for Apache Cassandra
API for Apache Gremlin
API for Table
Existing skills
Open-source ecosystem

Azure Cosmos DB Parts

- Account
  - Databases
  - Containers
  - Items
- Unique endpoint
- Supports Stored Procedures and Triggers
- Supports SQL dialect

Azure Cosmos DB

Uses for Azure Cosmos DB

- Web, mobile, gaming, and IoT application that handle massive amounts of data reads and writes globally.
- Azure Cosmos DB's guarantees high availability, throughput, low latency, and tunable consistency.
- Automatic Azure region data replication with zero downtime with Strong consistency.

Azure Cosmos DB APIs

- Web, mobile, gaming, and IoT application that handle massive amounts of data reads and writes globally.
- Azure Cosmos DB's guarantees high availability, throughput, low latency, and tunable consistency.
- Automatic Azure region data replication with zero downtime with Strong consistency.

Design Considerations

- Uses JSON Objects to store data
- Uses primary key and partition key
- Partition key is used to categorize data and create logical partitions.
- Each logical partition can store up to 20 GB of data.
- Physical containers are created to host one or more logical containers

Choosing a partition key

- Must be a value that doesn't change, string and have high cardinality
- If your container has a property with a wide range of possible values, it is a good partition key candidate.
  - For instance, the record ID. It has several possible values, but only one value per record.
- For large read-heavy containers, choose a partition key that frequently appears as a filter in your queries, like the key value

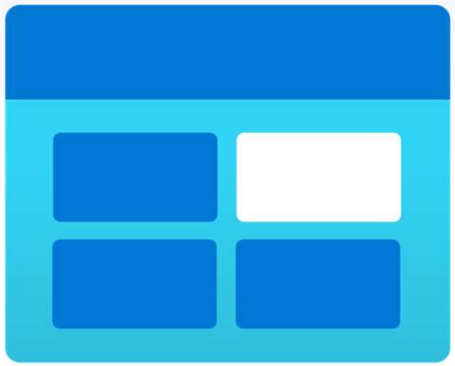
Azure Cosmos DB Review

- How to create Azure Cosmos DB Account
- How to create a database using NoSQL API
- How to manipulate records using .NET Core
- How to use Data Explorer
- How to setup local emulation
- NoSQL data management
- How to design for partition keys

Azure Storage Account

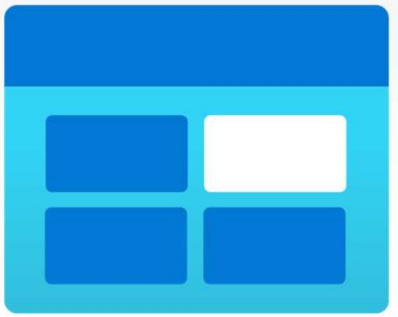
Objectives

- Learn about Azure Storage Accounts
- Explore Azure Blob Storage
- Explore Azure Table Storage
- Explore Azure Queue Storage
- Work with Azure Storage and .NET Core



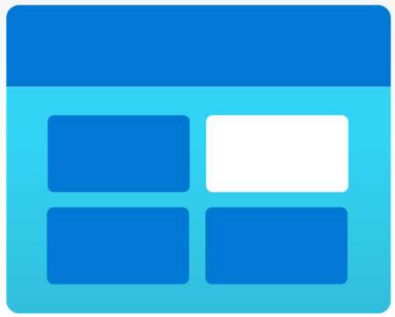
Explore Azure Blob Storage

- Azure Blob Storage is a cloud-based storage service for unstructured like text files, images, and videos.
- It is a scalable, secure, and cost-effective solution for storing large amounts of data in the cloud.
- Supports security features such as encryption, authentication, and authorization to protect your data.
- Data can be accessed from anywhere worldwide via HTTP or HTTPS.
- Client libraries are available for different languages, including: .NET, Ruby, Java, among others



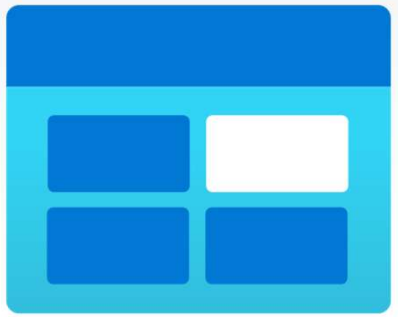
Commonly Used For

- Serving images or documents directly to a browser.
- Storing files for distributed access.
- Streaming video and audio.
- Writing to log files.
- Storing data for backup and restore, disaster recovery, and archiving.
- Storing data for analysis by an on-premises or Azure-hosted service.



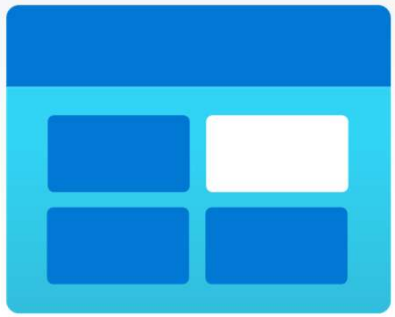
Commonly Used For

- Azure Data Lake Storage Gen2, Microsoft's enterprise big data analytics solution for the cloud.
- Azure Data Lake Storage Gen2 offers a hierarchical file system as well as the advantages of Blob Storage, including:
  - Low-cost, tiered storage
  - High availability
  - Strong consistency
  - Disaster recovery capabilities



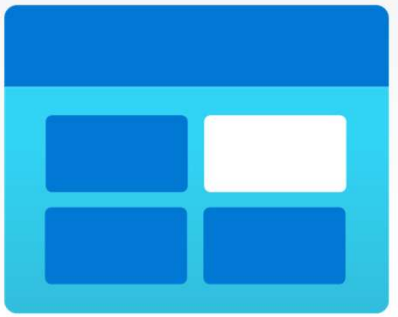
Storage Tiers

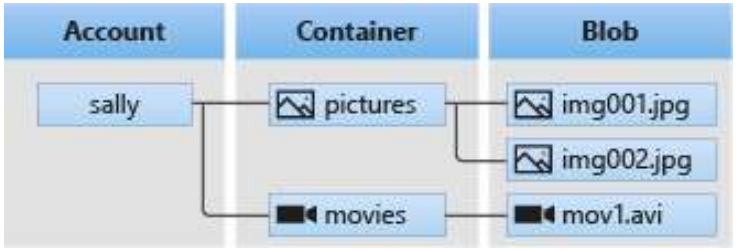
- Hot
  - Frequently accessed data
  - Most Costly
- Cool
  - Infrequently accessed data
  - Less Costly
- Archive
  - Rarely accessed data
  - For extended file backups and least costly



Components

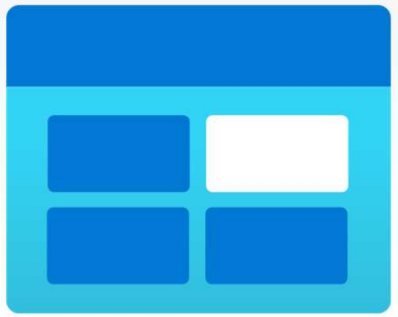
- The **storage account**
  - provides a unique namespace in Azure for your data.
  - Every object that you store in Azure Storage has an address that includes your unique account name.
- A **container** in the storage account
  - A container organizes a set of blobs, similar to a directory in a file system.
- A **blob** in a container





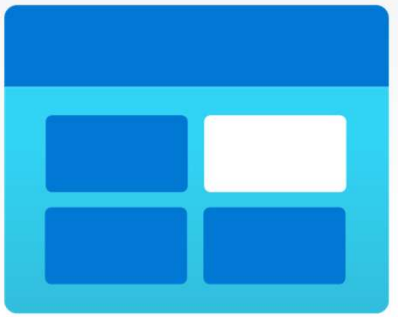
Types of Blobs

- Block blobs
  - Store text and binary data.
  - Made up of data blocks storing up to about 190.7 TiB.
- Append blobs
  - Has blocks like block blobs, but are optimized for append operations.
  - Ideal for logging data from virtual machines or similar operations



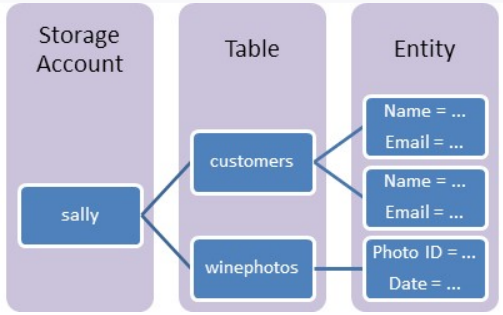
Types of Blobs

- Page blobs
  - Store random access files up to 8 TiB in size.
  - Page blobs store virtual hard drive (VHD) files for Azure virtual machines.



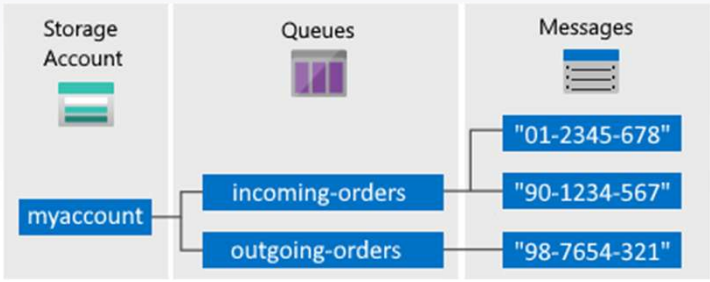
Azure Table Storage

- A service that stores non-relational structured data (also known as structured NoSQL data) in the cloud, providing a key/attribute store with a schemaless design.
- Easily to adapts to your data as the needs of your application evolve.
- Access is fast and cost-effective and is typically lower in cost than traditional SQL for similar volumes of data.



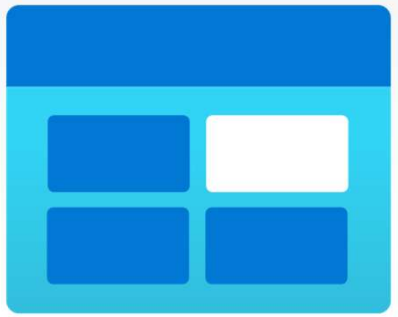
Azure Queue Storage

- Service for storing large numbers of messages.
- Messages can be accessed from anywhere via authenticated calls using HTTP or HTTPS.
- A queue message can be up to 64 KB in size.
- May contain millions of messages up to the total capacity limit of a storage account.
- Commonly used to create a backlog of work to process asynchronously.



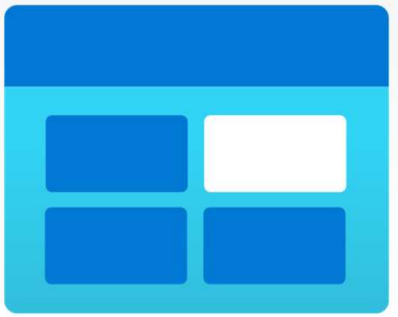
Web Project

- Registration form for conference attendees
- ASP.NET MVC
  - Azure Table Storage
  - Azure Blob Storage
  - Azure Queue Storage
    - Console App



Azure Storage Review

- How to create and manage a Storage Account
  - Blob, Table, and Queue Storage Services
- How to use Azure Portal and local storage browser clients to interact with services.
- Develop .NET Core Solution to use Storage Account
- Asynchronous communication using Queues (Pub-Sub pattern)



Azure Service Bus

Objectives

- Learn about Azure Service Bus
- Learn the difference between the Service Bus and Storage Queues
- Develop Publisher and Subscriber (Pub-Sub) code
- Understand how message brokers work

Why use Messaging?

- Transfer business data between applications.
- Decouple applications
- Improve reliability and scalability of applications and services.
- Less data loss since the message broker will store the message if the consumer is not online.
- Allows for multiple consumers to process the same data simultaneously, without interfering with each other.

Explore Azure Service Bus

- Fully managed enterprise message broker (PaaS)
- Supports message queues and publish-subscribe topics.
- Best for enterprise applications that need transactions, ordering, duplicate detection, strong consistency, and reliability.
- Supports features like first-in and first-out (FIFO), batching, transactions, dead-lettering, temporal control, routing and filtering, and duplicate detection

Explore Azure Service Bus

- Supports Topics and subscriptions, enabling 1:*n* relationships between publishers and subscribers.
- Allows you to do several operations in the scope of an atomic transaction. Results become visible to downstream consumers only upon success.
- Allows us to implement high-scale workflows and multiplexed transfers requiring strict message ordering or deferral.

Explore Azure Service Bus

- Similar to RabbitMQ, Apache ActiveMQ.
- PaaS advantage:
  - No hardware failure
  - No OS and infrastructure maintenance required
  - Automated Backups
  - Failover Mechanisms

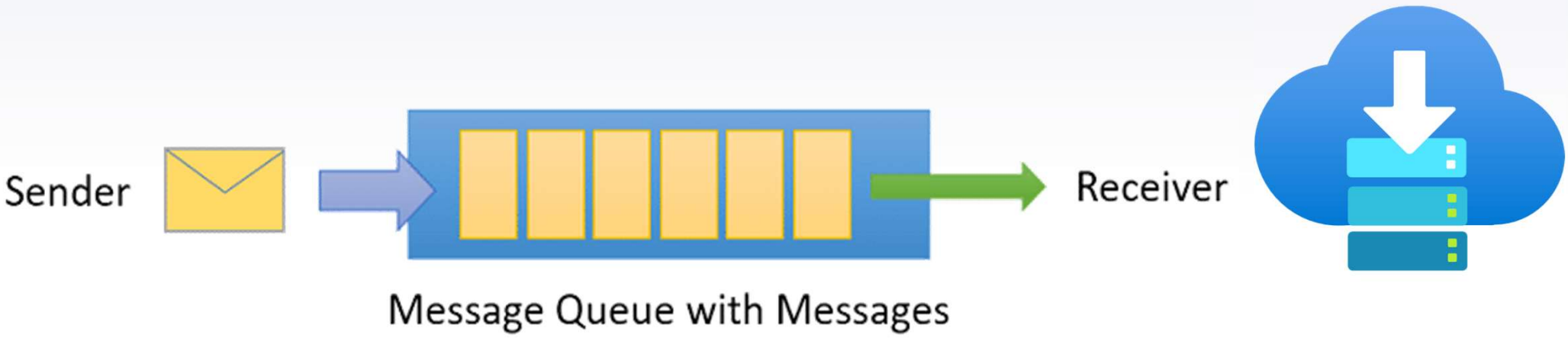
Queue Storage

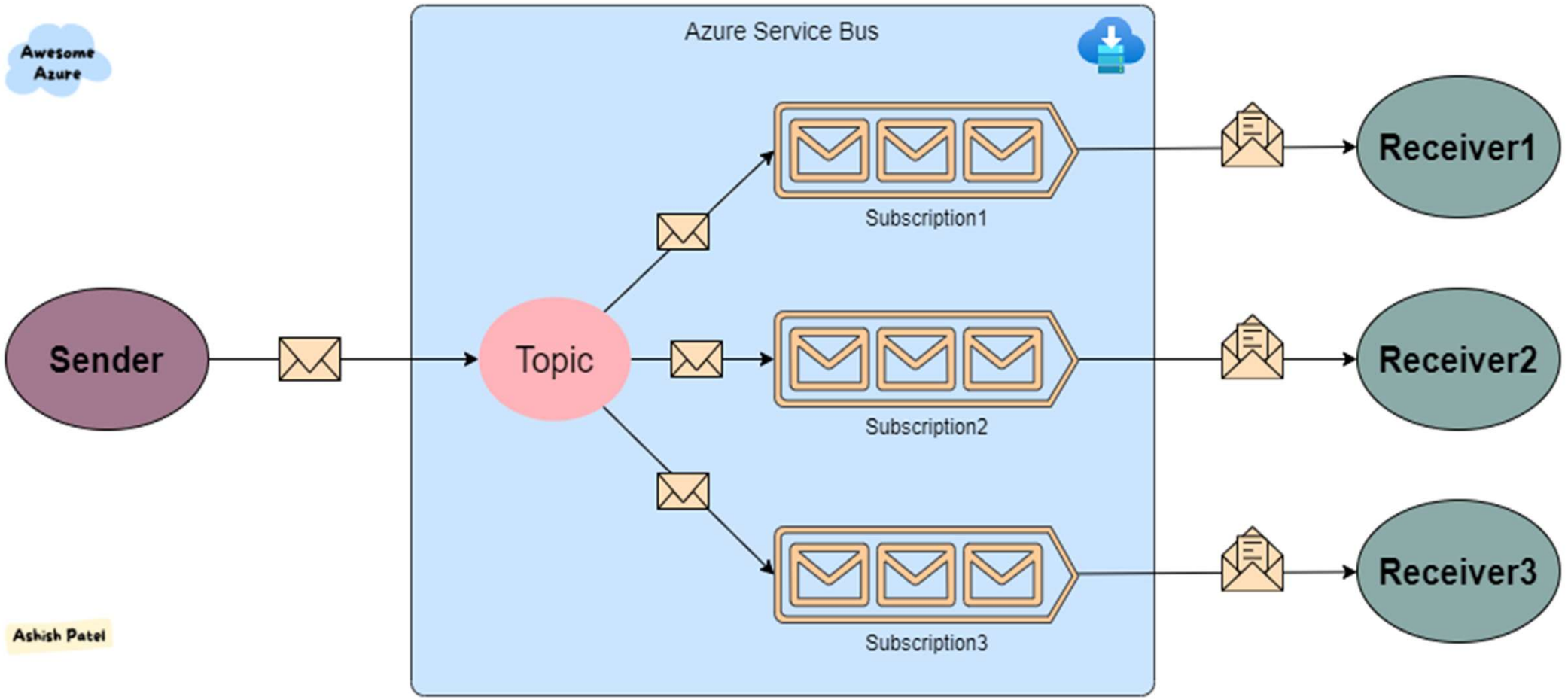
- Simple and flexible
- Queue Storage allows over 80GB of data in a queue
- Tracks progress for message processing, allowing several workers to process information in the event of failures
- Provides server-side logs of transactions in queues
- More costly than Service Bus
- Maximum Message live time is 7 days

Azure Service Bus

- Handles messages up to 1MB (as opposed to the 64KB limit on Queue Storage)
- Supports several enterprise-level features that ensure accurate message delivery
- Supports FIFO for ordered delivery
- Supports topics, a collection of queues
- Supports message transactions to ensure message consistency

Queue




Awesome Azure
Azure Service Bus
Sender
Topic
Subscription1
Subscription2
Subscription3
Receiver1
Receiver2
Receiver3
Ashish Patel

Service bus Review

- How to create a Service Bus Namespace
- How to create a Queue
- How to create a topic
- How to Send and Receive Messages from Queues
- How to Topics and Subscriptions (Pub/Sub Pattern)

Azure Functions

Objectives

- Learn about Azure Functions
- Understand serverless architecture
- Develop Azure Functions with Azure Portal, Visual Studio, and Visual Studio Code.
- Learn how to build event-driven applications
- Use Durable Functions for workflows

Exploring Azure Functions

- A serverless solution allowing you to
  - write less code,
  - maintain less infrastructure,
  - save on costs.
- Allows you to focus on writing code and not hardware and maintenance
- Allows us to develop event driven applications. Code runs when needed, based on some external factor
- Scalable and reliable solution

Exploring Azure Functions

- Scenarios:
  - RESTful API
  - Process BLOB storage uploads and activities
  - Real-time data processing (Azure SQL or Cosmos DB)
  - Process messages in queues and service bus
  - Scheduled tasks

Exploring Azure Functions

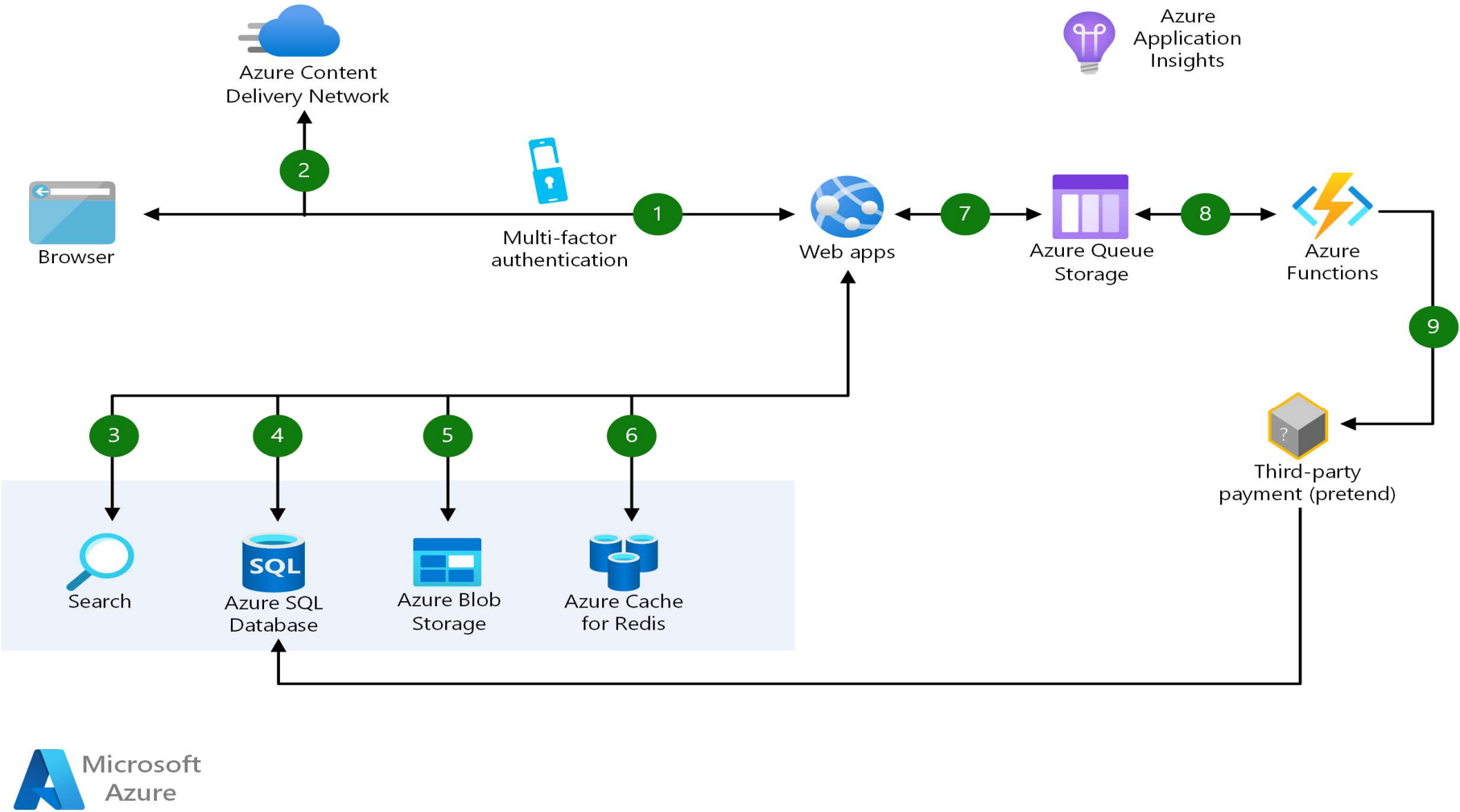
- Supports several languages
  - C#
  - Java
  - JavaScript
  - PowerShell
  - Python
  - Rust/Go
  - TypeScript

Exploring Azure Functions

- Pricing options:
  - Consumption – Pay as You Go
  - Premium – Have preset resources allocated
  - App Service Plan – Host like an app service web app

Serverless Architecture

- Minimizes time to delivery
- Fully managed services to boost developer productivity
- Optimizes resources and innovation
- Popular Azure Serverless Services:
  - Containers, Kubernetes, Functions, App Service, Service Bus, Event Grid, Cognitive Services (Bots, ML), Azure SQL, Cosmos DB, Blob Storage, Redis Cache, Azure AD


Azure Content Delivery Network
Azure Application Insights
Browser
2
Multi-factor authentication
1
Web apps
7
Azure Queue Storage
8
Azure Functions
9
Third-party payment (pretend)
3
4
5
6
Search
Azure SQL Database
Azure Blob Storage
Azure Cache for Redis
Microsoft Azure

Durable Functions

- *Durable Functions* allow you to write stateful functions in a serverless computing environment.
- Defines stateful workflows using *orchestrator functions* and *entity functions*
- Manages state, checkpoints, and restarts for you, allowing you to focus on your business logic.
- Simplifies **complex**, stateful coordination requirements in serverless applications.

Durable Functions

- Pattern 1:
  - Function Chaining.
  - Executes functions in a sequence
  - One output can be used for the following function
- Pattern 2
  - Fan-in/Fan-out
  - Parallel execution of functions
  - Waits on all to finish before moving on

Durable Functions

- *Other patterns*
  - *Aggregator – collect data over a period to be used in one operation*
  - *Human interaction – allows us to pause an operation pending human input (like an approval). Can be made to account for timeouts*
  - *Monitor – Polls until conditions are met*
  - *Async HTTP APIs – Can accept work and process in the background. The client can check on the status over time. Good for long-running operations*

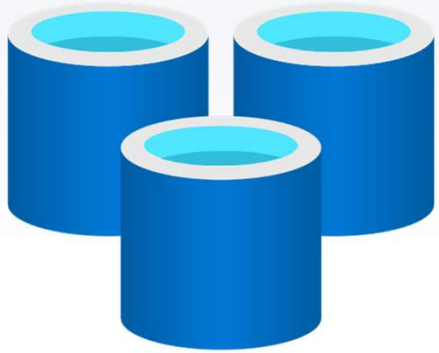
Section Review

- Learned to create Azure Functions
- Learned how to develop serverless solutions with Azure Services
- Used Visual Studio and Visual Studio Code for development
- Created a durable function
- Deployed Azure Functions App with several functions and bindings to different services

Azure Cache for Redis

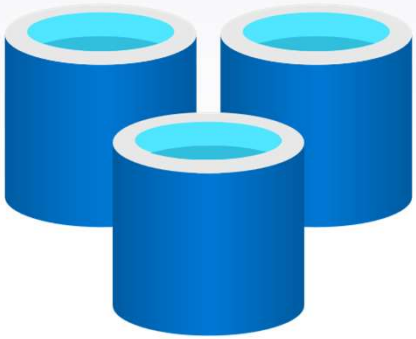
Objectives

- Learn about Azure Redis Cache
- Understand why caching is important
- Provision Azure Redis Cache service
- Develop a .NET Core solution with caching features.



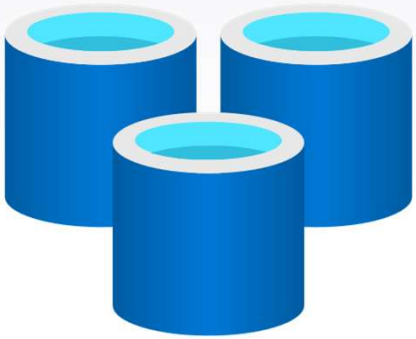
What is caching?

- A cache is a high-speed data storage layer that stores a subset of typically transient data.
- Future requests for that data are served up faster than is possible by accessing the data's primary storage location.
- A cache typically stores a subset of data that is typically stored in databases
- Caching allows you to reuse previously retrieved or computed data efficiently.



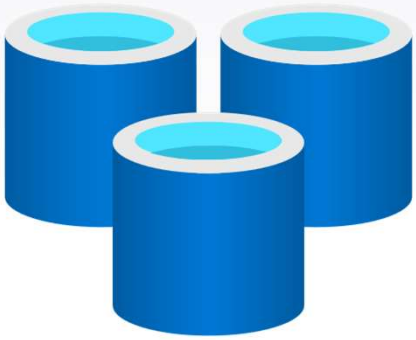
Redis Cache

- An open-source in-memory data store used as a:
  - database,
  - cache,
  - streaming engine,
  - message broker.
- Redis has a large community of developers, architects, and contributors.
- Low latency solution for decreasing the load on the main data store.



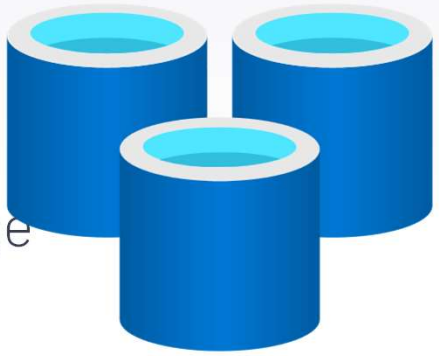
Redis Cache

- Redis supports 5 data types:
  - Strings – store strings up to 512 megabytes in size
  - Hashes – maps between string fields and values or key-value pairs.
  - Lists – list of strings sorted by insertion order. Add values to the head or tail
  - Sets – unordered collection strings that can be managed
  - Ordered Sets – like sets, but entries are scored and ordered.



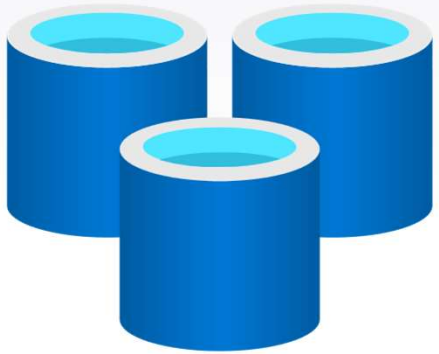
Exploring Azure Cache for Redis

- Azure Cache for Redis provides a managed cloud solution based on the Redis software.
- Azure Cache for Redis provides:
  - critical low-latency and high-throughput data storage
  - secure and dedicated Redis server instances
  - full Redis API compatibility.
- Can be used as a distributed data or content cache, a session store, a message broker, and more.



API Project

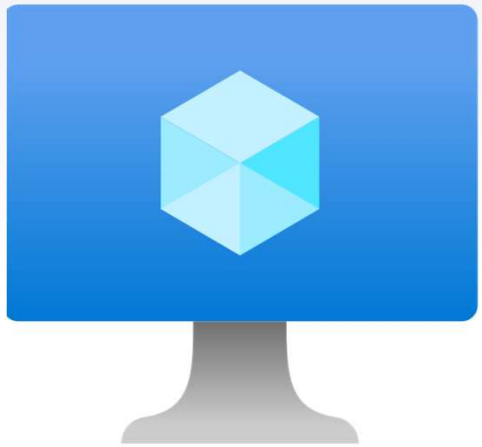
- Simple CRUD using Database
- Setup Local Redis Cache
  - Integrate with Redis for Read/Write operations
  - Provision Azure Cache for Redis
  - Use Azure Cache in code



Azure Virtual Machines

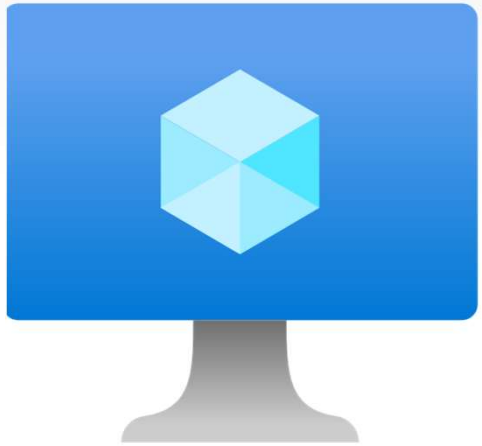
Section Overview

- Learn virtual machines
- Understand IaaS
- Provision Virtual Machines
  - Azure Portal
  - Azure PowerShell
  - Azure CLI
  - ARM Templates



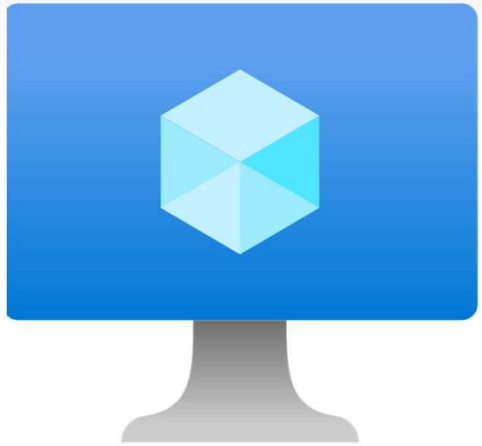
Infrastructure as a Service

- A cloud service model that offers infrastructure resources, such as:
  - compute, storage,
  - networking,
  - machine virtualization
- Helps you to reduce maintenance of on-premises infrastructure and save money on hardware costs.
- Gives you the flexibility to scale your IT resources based on demand.

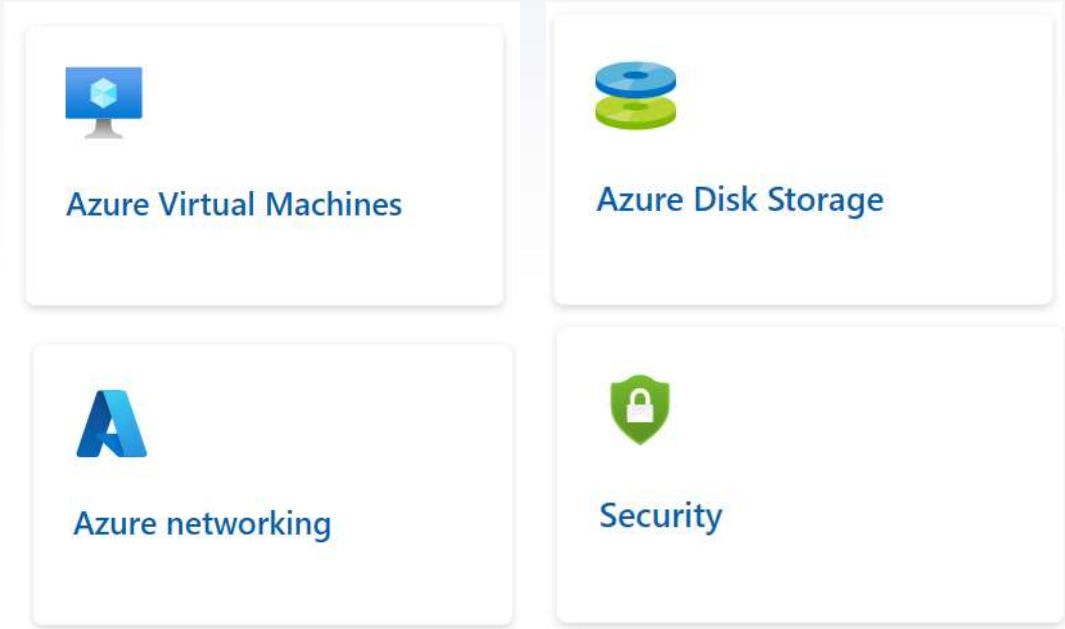


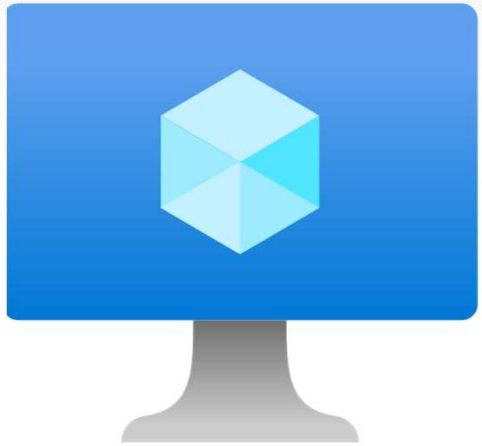
Infrastructure as a Service

- All PaaS and SaaS services rely on IaaS resources
- IaaS resources can be provisioned and managed by customer
- Easy for lift and shift migration without completely reconfiguring architecture
- Reduces CapEx and improves SLAs
- Improves business continuity and disaster recovery
- Increased security



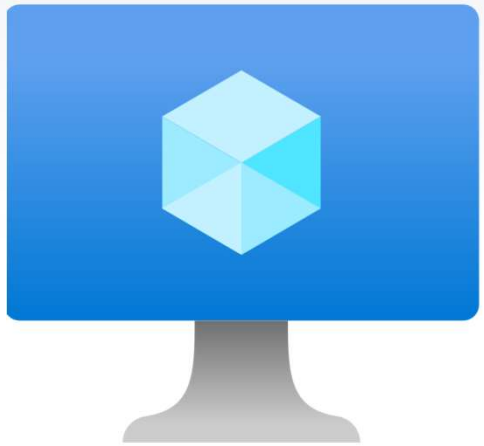
Infrastructure as a Service





Section Review

- Learned about IaaS and the benefits
- Provisioned Virtual Machine and supporting resources using:
  - Azure Portal
  - CLI and PowerShell
  - ARM Templates
- Connected to Virtual Machine via RDP



Azure Containers

Section Overview

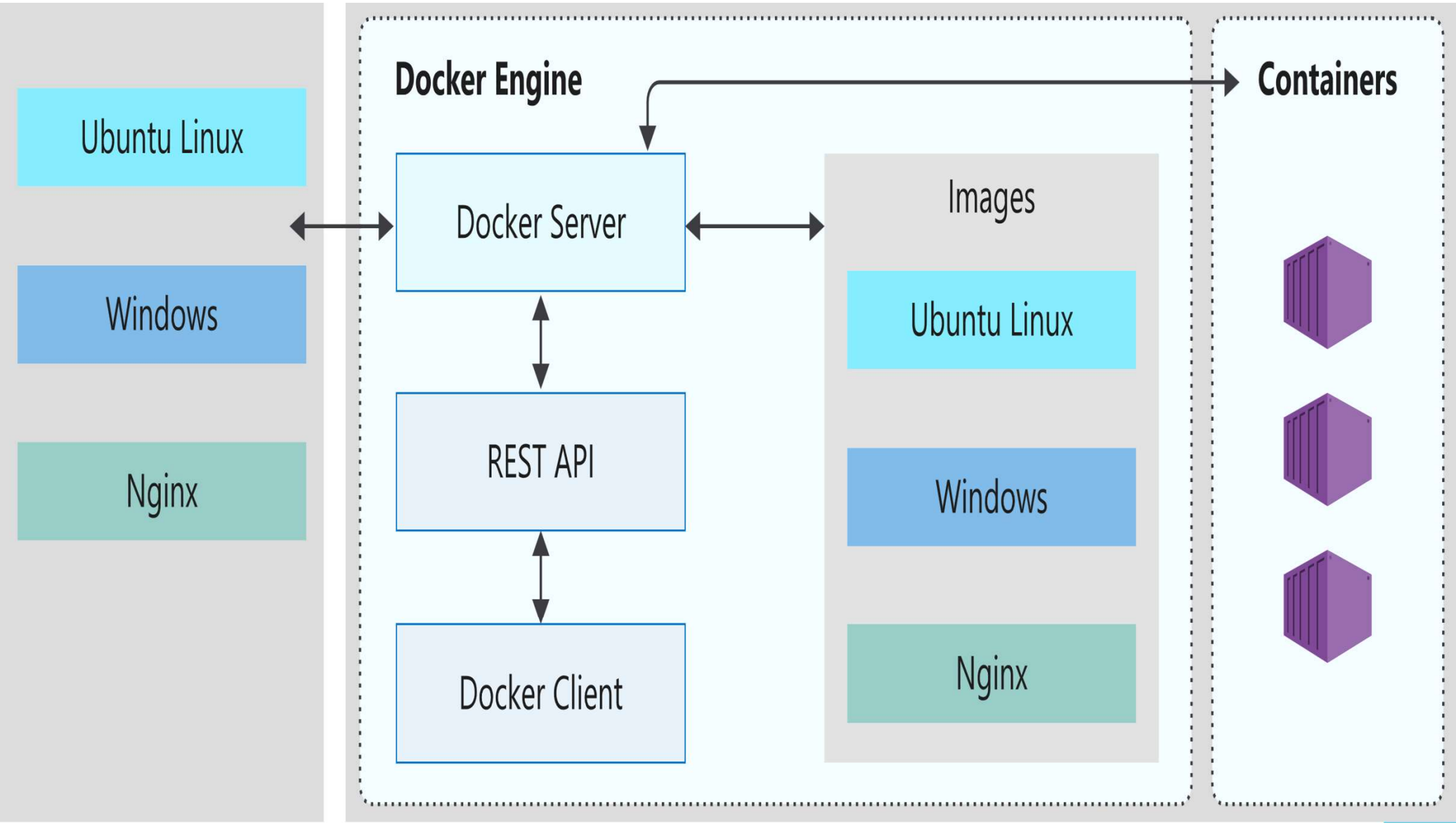
- Learn about containerization
- Setup and use Docker locally
- Containerize .NET Core Project
- Understand Azure Container Instances
- Understand Azure Container Registry
- Review Kubernetes

Understanding Containers

- A container is a standard unit of software that packages up code and all its dependencies, so the application runs quickly and reliably from one computing environment to another. *(source: www.docker.com)*
- The problem?
  - Cost of virtualization
  - Differences in environments (dev, QA, prod)
  - Portability (changing hosting, OS, etc.)

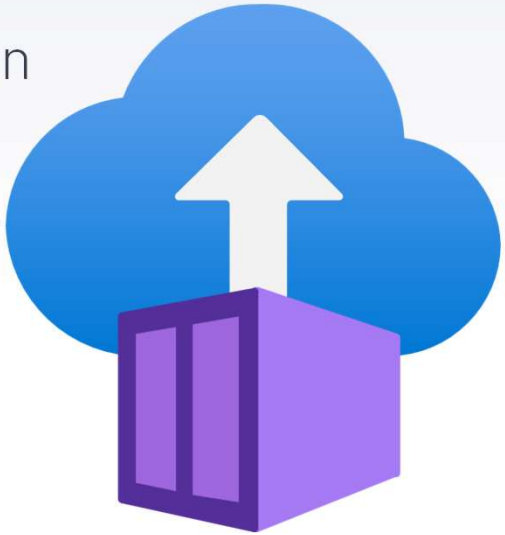
Understanding Containers

- Benefits of containers:
  - **Portability: Docker** created the industry standard for containers, so they could be portable anywhere
  - **Lightweight:** Containers share the machine's OS system kernel and therefore do not require an OS per application.
  - **Secure:** Applications are safer in containers.
  - **Immutable**: An image will always be the same when it is created.


Ubuntu Linux
Windows
Nginx
Docker Engine
Docker Server
REST API
Docker Client
Images
Ubuntu Linux
Windows
Nginx
Containers

Using Docker

- A containerization platform used to develop, ship, and run containers.
- Doesn't use a hypervisor, and you can run Docker on your desktop or laptop if you're developing and testing applications.
- Supports Linux, Windows, and macOS.
- Supports production workloads for many variants of Linux and Microsoft Windows Server 2016 and above.
- Supported by many cloud providers, including Microsoft Azure



Using DockerHub

- Docker Hub is a Software as a Service (SaaS) Docker container registry.
- Docker registries store and distribute the container images we create.
- Docker Hub is the default public registry Docker uses for image management.
- Supports public and private repositories

Azure Containers

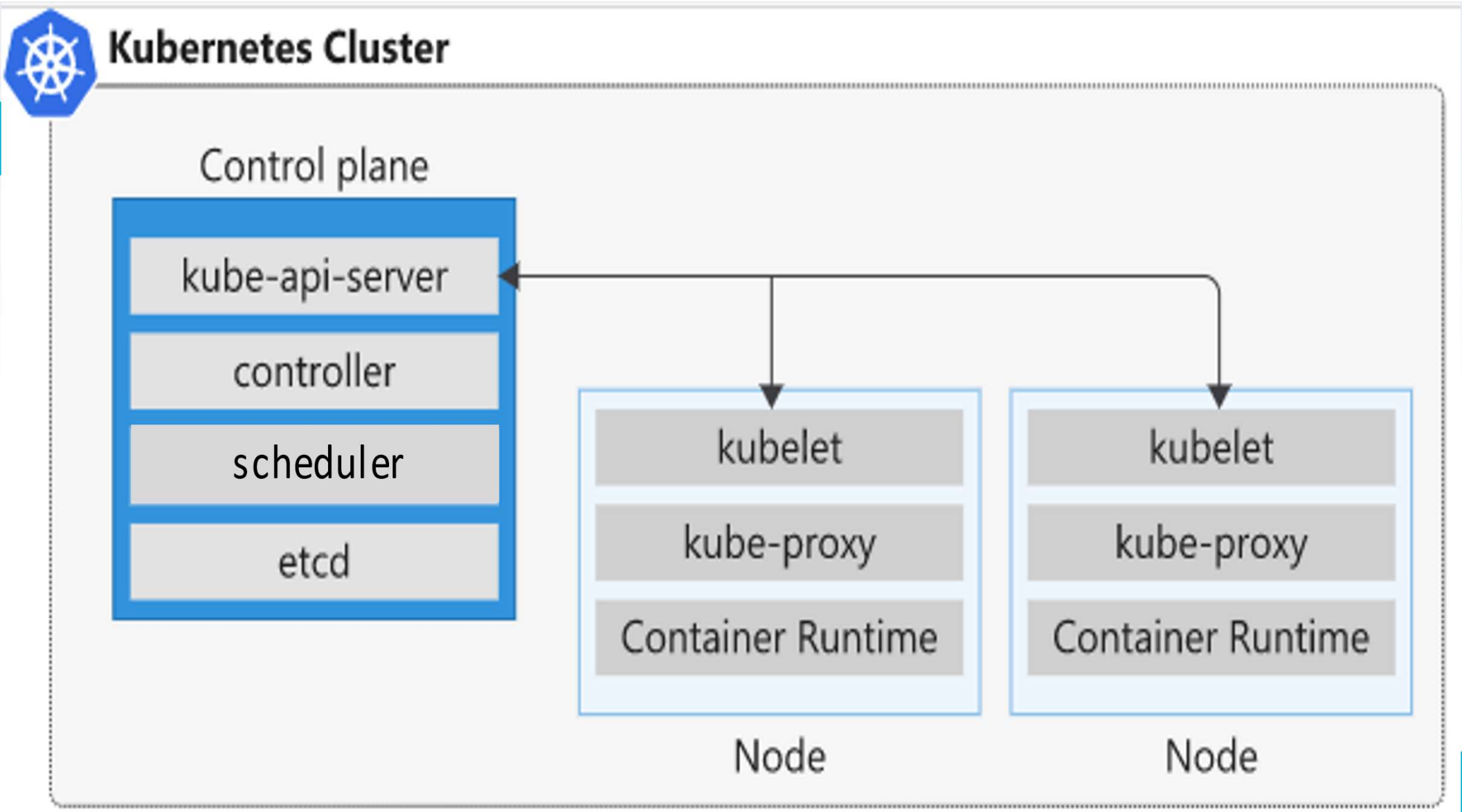
- Azure Container Instances
  - Loads and runs Docker images on demand.
  - Can retrieve an image from a registry, such as Docker Hub or Azure Container Registry.
- Azure Container Registry
  - A managed Docker registry service based on the open-source Docker Registry 2.0.
  - Private and hosted in Azure,
  - Allows you to build, store, and manage images for all types of container deployments.

Using Kubernetes

- Offers reliable scheduling and orchestration for fault-tolerant application workloads.
- Provides a declarative approach to deployments, backed by a robust set of APIs for management operations.
- Provides container management for organizing, adding, removing, or updating several containers at a time.

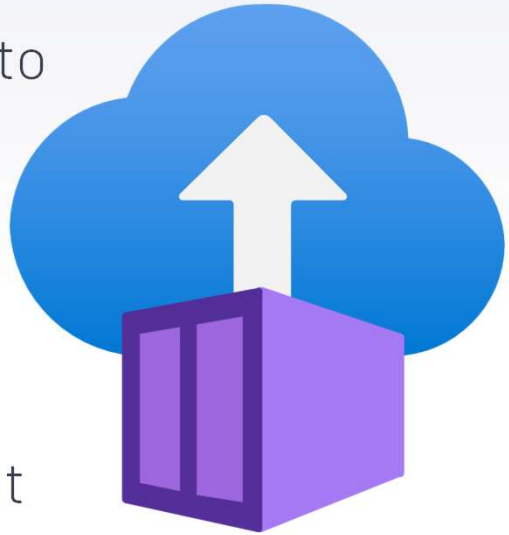
Using Kubernetes

- Abstracts useful tasks such as:
  - Self-Healing
  - Scaling
  - Network management
  - Storage
  - Container updates
  - Secret management


Kubernetes Cluster
Control plane
kube-api-server
controller
scheduler
etcd
kubelet
kube-proxy
Container Runtime
Node
kubelet
kube-proxy
Container Runtime
Node

Azure Kubernetes Service

- Azure Kubernetes Service (AKS) offers the quickest way to develop and deploy containerized apps in Azure.
- Full power of Kubernetes and orchestration, backed by Microsoft Azure
- Offers more orchestration and management than Azure Container Instances. Can be thought of as a management service for ACI.



Section Review

- Learned about containerization and Docker
- Containerized .NET Core Project
  - Dockerfile
  - Additional resources
- Deployed app to Azure Container Instances
- Created Container image for Azure Container Registry
- Reviewed Kubernetes and Container Orchestration

Azure AD

Section Overview

- Learn Azure AD
  - B2B – Business to Business (Internal facing)
  - B2C – Business to Customer (External facing)
- Learn to create an Azure Tennant
- Learn to develop a Single-Sign-On solution
  - Secure API with token
  - Understand OAuth2.0 Code Flow
  - Learn 'On Behalf Of' Flow

Version Check

- Visual Studio 2022
- .NET 5.0 (Out-Of-Support)
  - Startup.cs
  - Program.cs
    - services.AddAuthentication(...);
- .NET 6/7/8
  - Program.cs
    - builder.Services.AddAuthentication(...);

Understanding Azure AD

- Managed offering of Active Directory
- Enables your employees to access external resources like Microsoft 365, the Azure portal, and thousands of other SaaS applications.
- An OpenId Connect and OAuth2.0 identity provider
- Has two offerings:
  - Azure Active Directory (Internal)
  - Azure AD B2C (External)

Section Review

- Create an Azure Tennant
- Learned Azure AD
  - Register an application
  - Setup Users, roles, claims
- Created and configured ASP.NET Core applications
  - OAuth2.0 Code Flow
  - On Behalf of Flow

Conclusion

Course Review

- Getting started with Azure
  - Account creation
  - Cloud Hosting options
- Azure Management Tools
  - Portal, CLI, and PowerShell
- Azure Service configurations and provisioning
  - Web, SQL, Cosmos DB, Functions, Storage, Messaging, Active Directory

Course Review

- Serverless Architecture
- Infrastructure as a Service
  - Virtual machines
  - VNET
  - Storage Disks
- Docker, Containers and Registries
- Azure Active Directory

Course Review

- .NET Core Development
  - Visual Studio and VS Code (w/ dotnet CLI)
- Azure SDKs and packages
  - Cosmos DB Clients
  - Queue, Table and BLOB Storage
  - Service Bus senders and receivers
  - Azure Functions

THANK YOU

Trevoir Williams