

# Get the best out of Live Sessions HOW?

# e!



## Check your Internet Connection

Log in 10 mins before, and check your internet connection to avoid any network issues during the LIVE session.

## Speak with the Instructor

By default, you will be on mute to avoid any background noise. However, if required you will be **unmuted by instructor**.



## Clear Your Doubts

Feel free to clear your doubts. Use the “Questions” tab on your webinar tool to interact with the instructor at any point during the class.

## Let us know if you liked our content

Please share feedback after each class. It will help us to enhance your learning experience.



edureka!



# Microsoft Azure Developer Associate (AZ-204)

# COURSE OUTLINE

## MODULE 05

Introduction to Azure IaaS Compute Solutions

Implementing Azure Batch Service and Disk Encryption

Designing and Developing Applications That Use Containers

Implementing Azure App Service Web Apps and Mobile Apps

**Implementing Azure App Service API Apps and Azure Functions**

Developing Solutions That Use Azure Table Storage and Cosmos DB

Developing Solutions That Use Relational Database And Azure Blob Storage

Implementing Authentication and Access Control In Azure

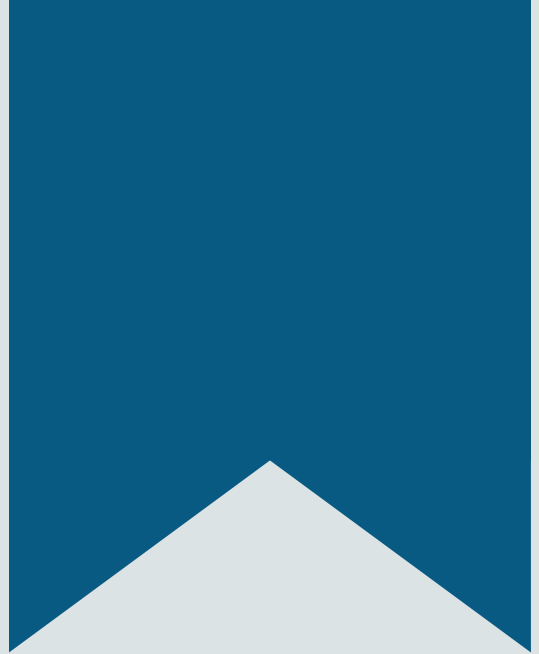
Implementing Secure Data Solutions and Integrate Caching & CDN

Instrument Monitoring, Logging and Scalability Of Apps & Services

Connecting to and Consuming Azure and Third-party Services

Developing Event-based and Message-based Solutions in Azure





# Module 5 – Implementing Azure App Service API Apps and Azure Functions

# Topics

---

- Azure App Service API Apps
- Shared features of API and WebApp
- Authentication in Azure API Apps
- API Management
- API Documentation using Swagger
- Azure Functions
- Languages Supported by Azure Functions
- Uses of Azure Functions
- Triggers and Bindings in Functions
- Azure Functions Cost
- Durable Functions and its benefits
- Durable Functions – Application Patterns
- Orchestrator Example – Human Interaction

# Objectives

---

After completing this module, you should be able to:

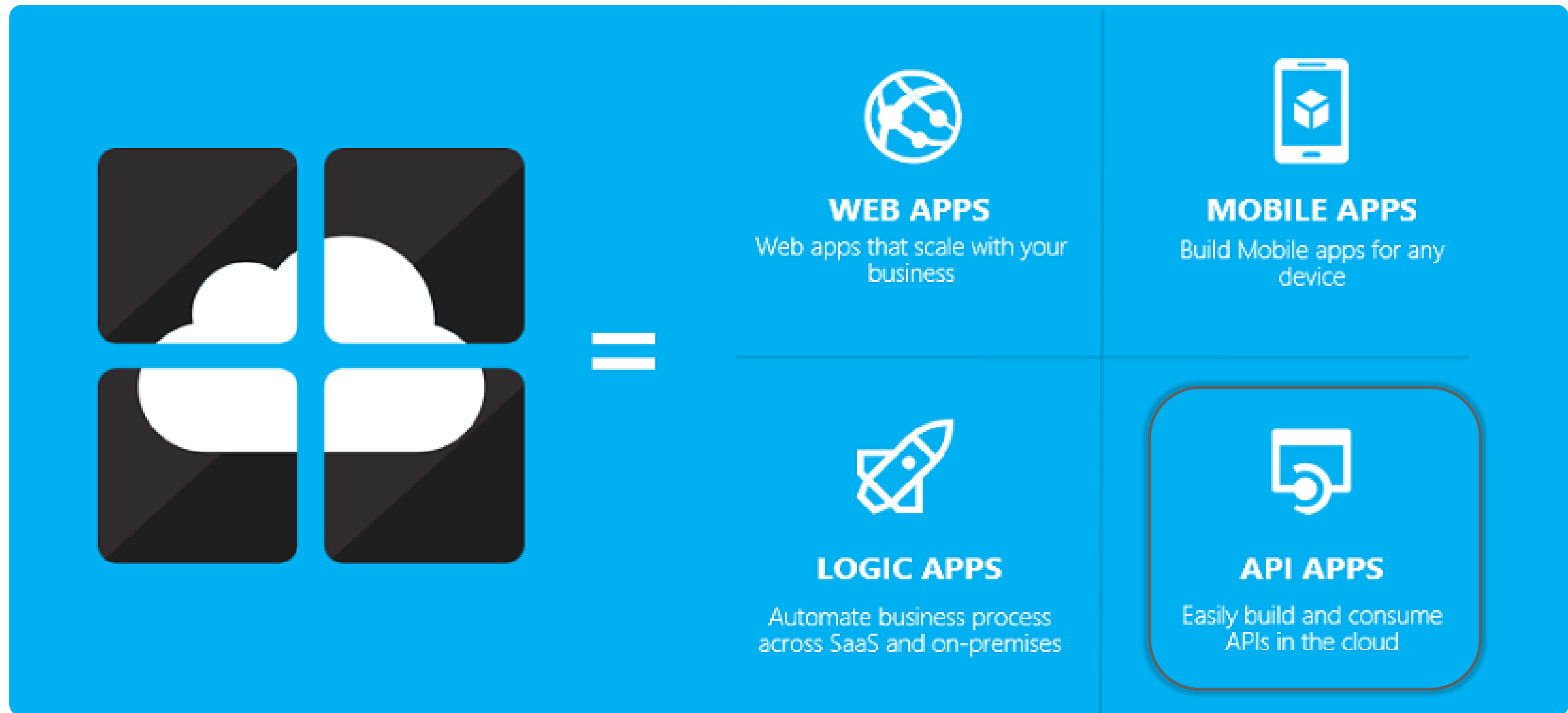
- Create APIs using Azure App Service
- Use Swagger to document an API
- Understand the core features and functionality of Azure Functions
- Develop Azure Functions using Visual Studio
- Implement durable functions



# Azure App Services

# Azure App Service Types

---







# Azure App Service API Apps

# Azure App Service API Apps

---

- API apps in Azure App Service offer features that make it easier to **develop, host, and consume APIs** in the cloud and on-premises
- With API apps you get
  - Enterprise grade security
  - Simple access control
  - Hybrid connectivity
  - Automatic SDK generation
  - and Seamless integration with Logic Apps



# Why Use Azure API Apps?

---



When you need a scalable, secure platform to build your API's then Azure API Apps is the best solution



You can develop your API's using multiple languages such as Python, C# or even PHP



You can select from a variety of web frameworks such as Node JS, ASP.net Web API and more



When you develop using Visual Studio you get integration through project templates and deployment wizards

# Shared Core Features With WebApps

Azure API Apps are hosted using Azure App Services offering meaning they share common features that are offered for Web Apps:

## Auto-scaling

The ability to Scale out instances according to changes in CPU and other metrics.

## Authentication

Harness Azure AD as an Identity Provider to federate user authentication using popular SSO protocols such as OAuth.

Minimise production application downtime by deploying to staging slots.

## Deployment slots

Host your API Apps in an App Hosting plan (Free, Shared, Basic, Standard, Premium) to suit your workload.

## App Service Plan



# API Apps for Continuous Integration and Deployment

---

01 Azure API Apps also supports continuous deployment from your source control system

02 You can then use Azure App Service deployment slots to deploy your changes to a staging area

03 This allows developers to pre-qualify changes before they are promoted to production

# Authentication in Azure API Apps

---

- If you wish to federate your authentication, then API Apps can be configured to use the following as Identity Providers:
  - Azure AD
  - Google
  - Facebook
  - Twitter
  - Microsoft

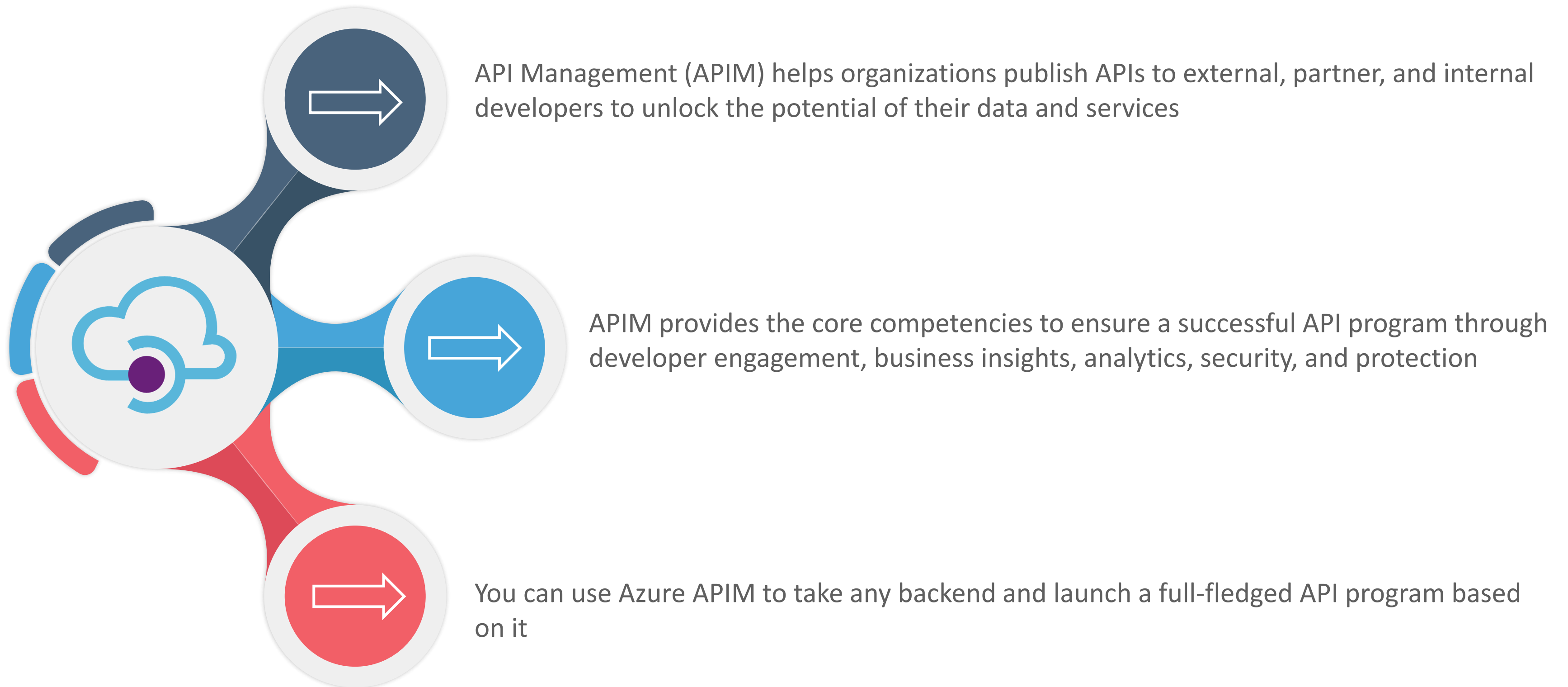
**NOTE:** By default, your API app will be configured with ***no*** authentication method. Azure Active Directory allows you to use **OAuth** to protect your API.



# Leveraging API Management in Azure

# What is API Management?

---





# API Management – Working

To use API Management, **Administrators** *create* APIs



Each API consists of *one or more* **operations**, and each API can be added to *one or more* **products**



To use an API, **Developers** *subscribe* to a **product** that contains that API



And then they can **call** the API's *operation*, subject to any usage policies that may be in effect



# Demo 1 – Creating an APIM Instance and a New API



# API Documentation

# Documenting an API Using Swagger

---

Swagger is an API documentation framework

With swagger API developers can build documentation using *metadata*

If you are building an Azure API App in *Visual Studio*, then the API App project template will add the *Swagger Nuget references* to the project

Once you deploy your API app to Azure you can access the *Swagger UI* using the URL: *https://yourwebdomainname/swagger*

# Using Swashbuckle to Create Swagger Objects

---

- **Swashbuckle** is an open source project for generating Swagger documents for Web APIs that are built with *ASP.NET Core MVC*
- There are two core components to Swashbuckle:
  - ***Swashbuckle.SwaggerGen***: provides the functionality to generate JSON Swagger documents that describe the objects, methods, return types, etc.
  - ***Swashbuckle.SwaggerUI***: an embedded version of the Swagger UI tool which uses the above documents for a rich customizable experience for describing the Web API functionality and includes built in test harness capabilities for the public methods.



# Demo 2 – Using Swashbuckle to Create Swagger Objects in ASP.NET Core

# Azure Functions

# Azure Functions Overview

---

Azure Functions is a solution for easily running small pieces of code, or "functions," in the cloud





# Languages Supported By Azure Functions

---

Functions can make development even more productive, and you can use your development language of choice, such as C#, F#, Node.js, Java, or PHP



# What Can You Do With Functions?

---

- ❖ Functions is a great solution for processing data
- ❖ Best fit for Integrating systems
- ❖ Working with the internet-of-things (IoT)
- ❖ Building simple APIs
- ❖ Serverless Microservices for event-driven scenarios



Consider Functions for tasks like image or order processing, file maintenance, or for any tasks that you want to run on a schedule.

# Triggers in Functions

---

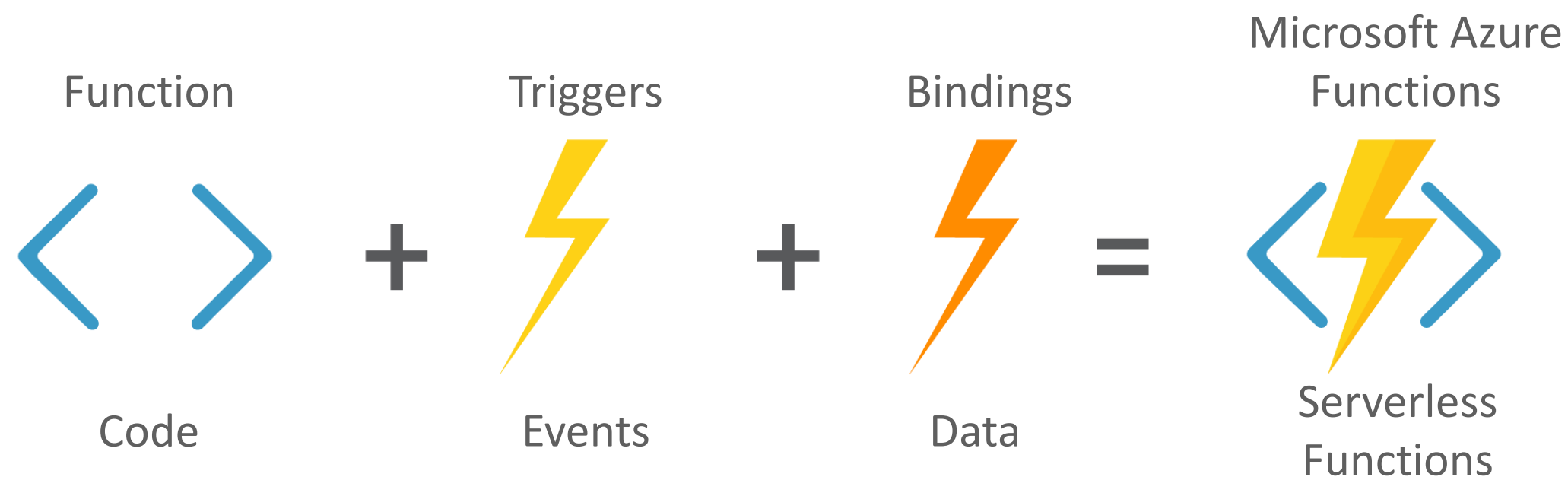
- Triggers are what cause a function to run
- A trigger defines how a function is invoked and a function must have exactly one trigger
- Triggers have associated data, which is often provided as the payload of the function



# Bindings in Functions

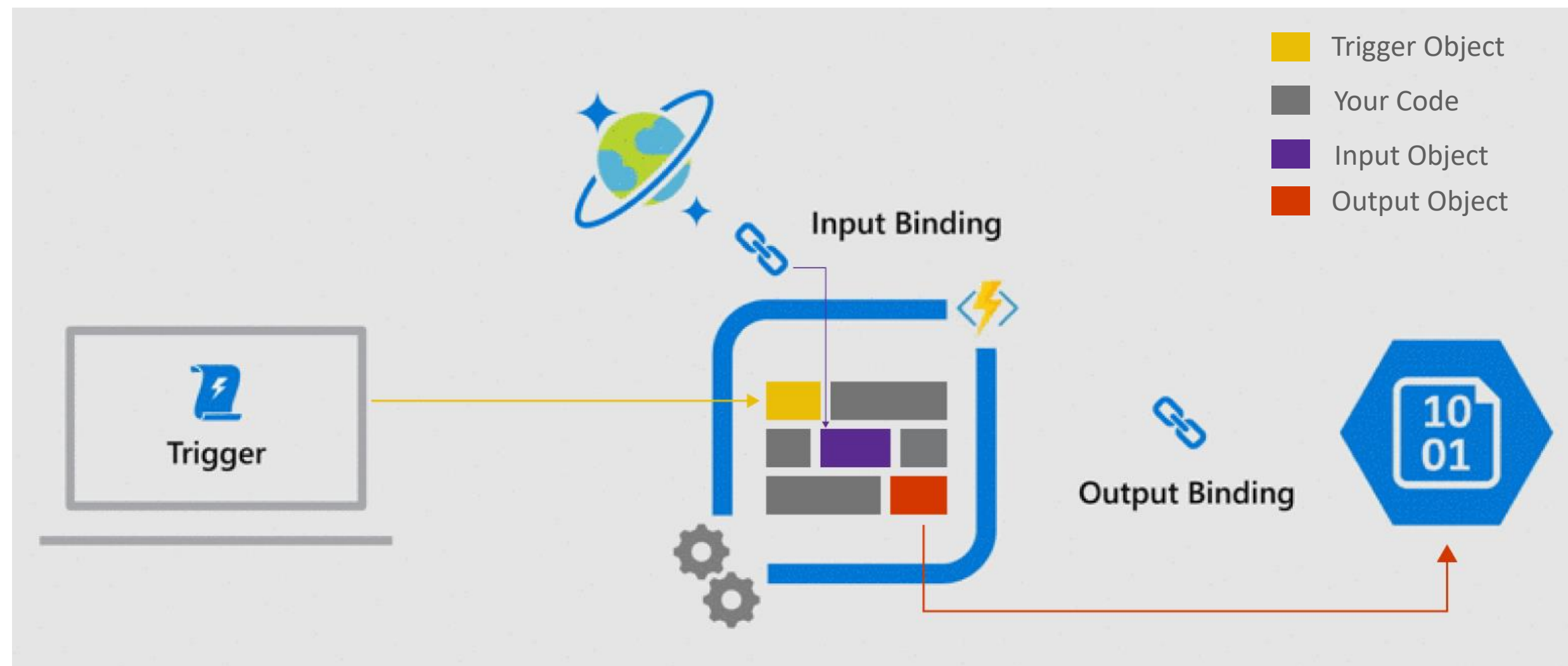
---

- Binding to a function is a way of declaratively connecting another resource to the function
- Bindings may be connected as input bindings, output bindings, or both
- Data from bindings is provided to the function as parameters
- You can mix and match different bindings to suit your needs
- Bindings are optional and a function might have one or multiple input and/or output bindings



# Triggers and Bindings Implemented in Functions

- Triggers and bindings let you avoid hardcoding access to other services
- Your function receives data (for example, the content of a queue message) in function parameters
- You send data (for example, to create a queue message) by using the return value of the function



# Different Triggers and Bindings Which Can Be Used

---

Triggers	Bindings
HTTPTrigger	HTTP & Webhooks
TimerTrigger	Timer
CosmosDBTrigger	Cosmos DB
BlobTrigger	Blob Storage
QueueTrigger	Queue storage
EventGridTrigger	Event Grid
EventHubTrigger	Event Hubs
ServiceBusQueueTrigger	Service Bus
ServiceBusTopicTrigger	Notification Hubs
	Mobile Apps
	Table storage
	Microsoft Graph Excel tables

# How Much Do Functions Cost?

---

Azure Functions has two kinds of pricing plans. Choose the one that best fits your needs:

## Consumption plan

- When your function runs, Azure provides all of the necessary computational resources
- You don't have to worry about resource management, and you only pay for the time that your code runs

## App Service plan

- Run your functions just like your web apps
- When you are already using App Service for your other applications, you can run your functions on the same plan at no additional cost



# Demo 3 – Creating Functions, Bindings, and Triggers

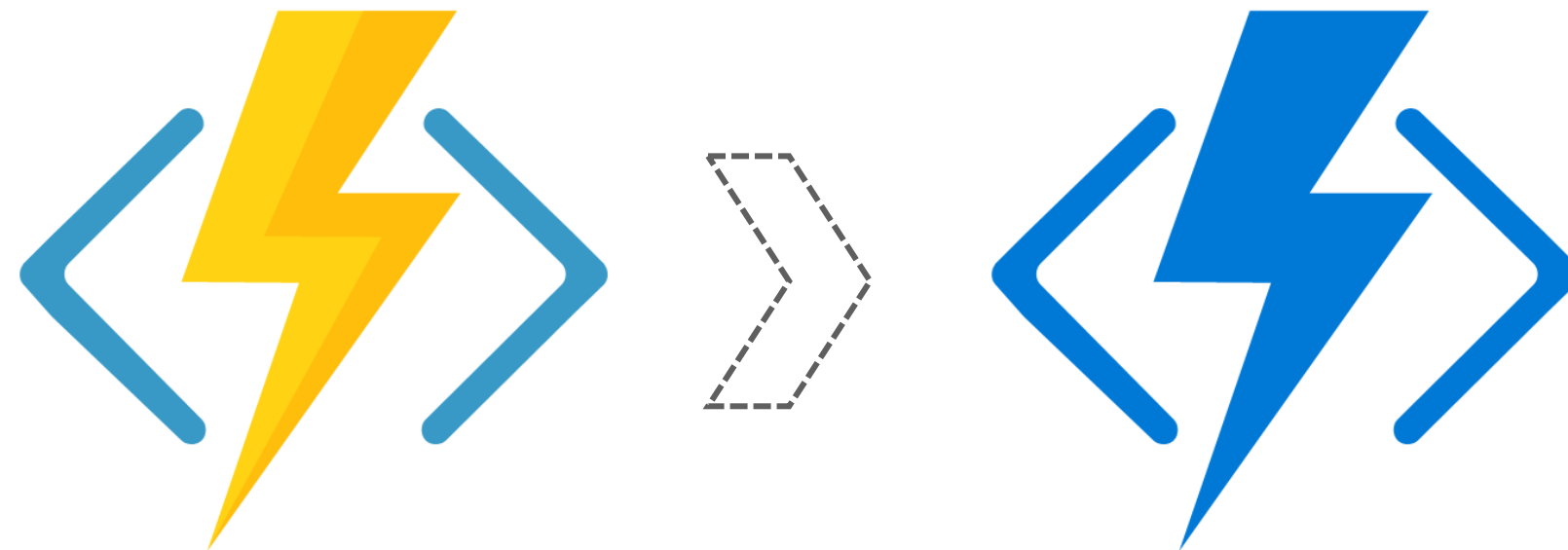


# Durable Functions

# What are Durable Functions?

---

- ***Durable Functions*** are an *extension* of Azure Functions that lets you write stateful functions in a serverless environment
- The extension manages state, checkpoints, and restarts for you
- The extension lets you define stateful workflows using an ***orchestrator function***



# Benefits of Durable Functions

---

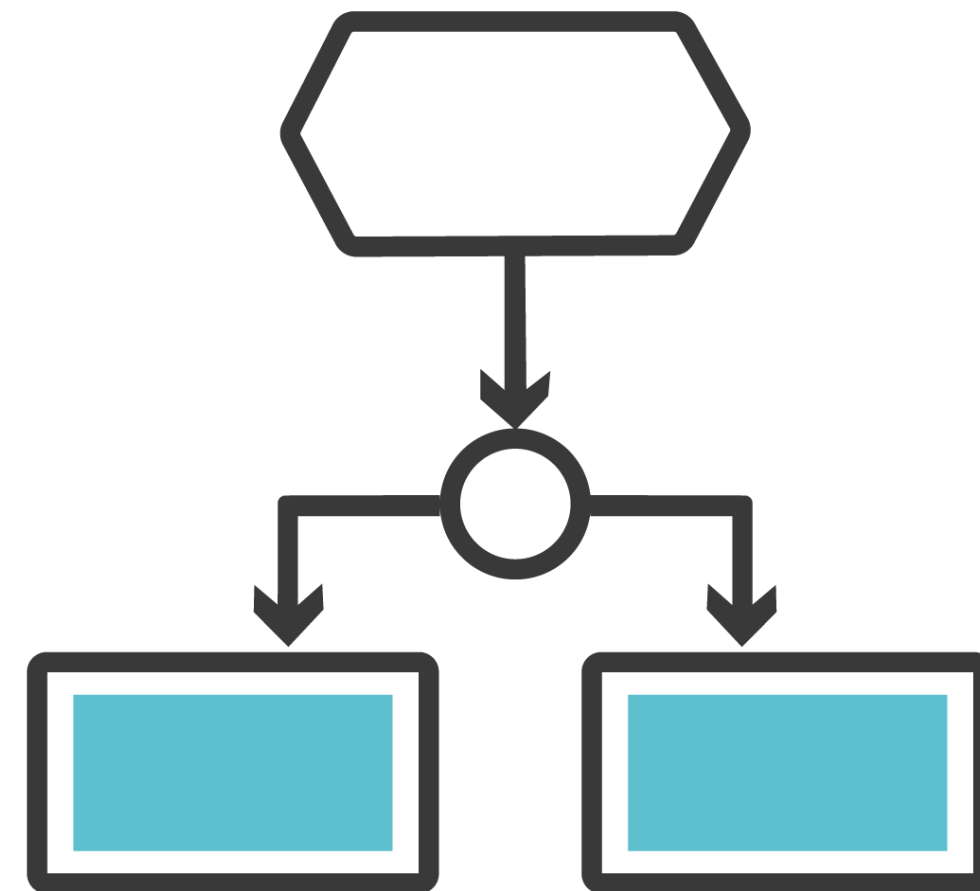
- The orchestrator function provides the following benefits:
  - ✓ You can define your workflows in code
  - ✓ No JSON schemas or designers are needed
  - ✓ Other functions can be called both synchronously and asynchronously
  - ✓ Outputs from called functions can be saved to local variables
  - ✓ Progress is automatically checkpointed when the function awaits
  - ✓ Local state is never lost when the process recycles or the VM reboots



# Durable Functions – Application Patterns

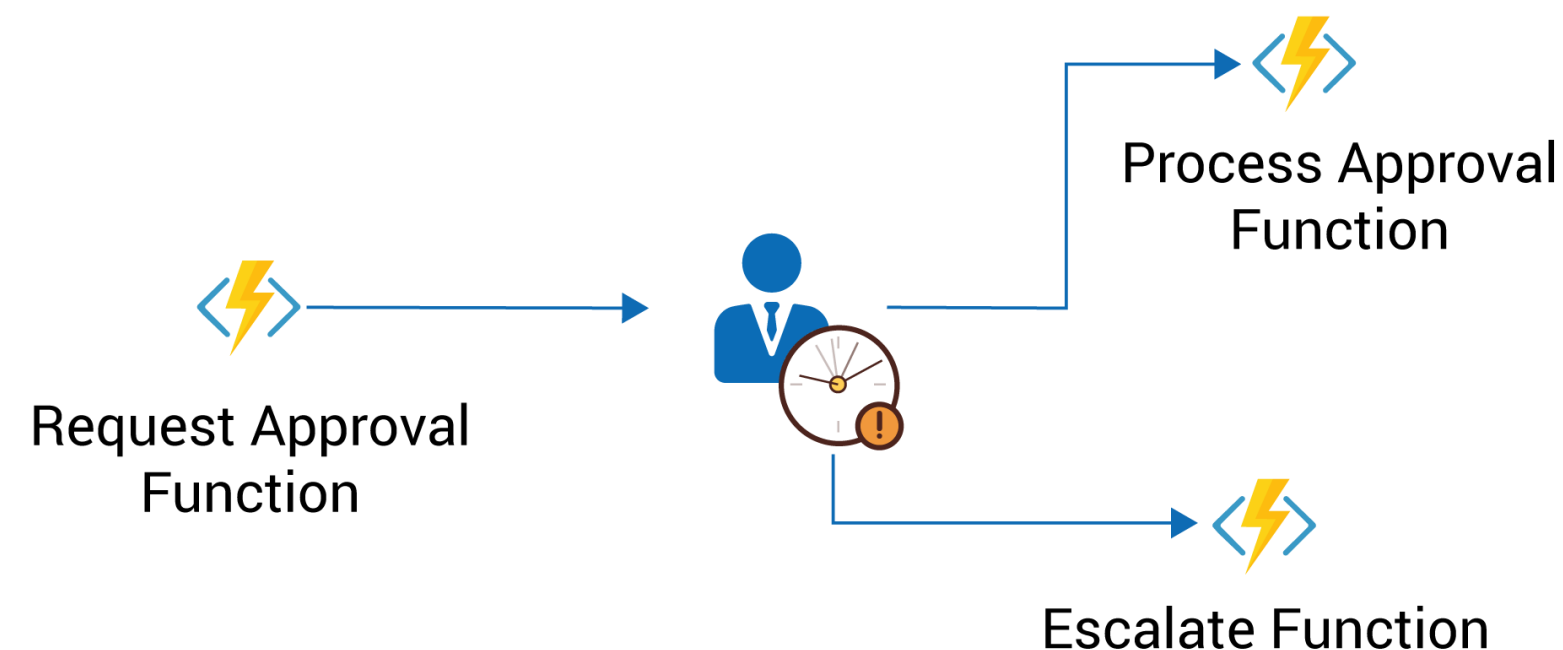
---

- The primary use case for Durable Functions is simplifying complex, stateful coordination requirements in serverless applications
- The following are some typical application patterns that can benefit from Durable Functions:
  - Chaining
  - Fan-out/fan-in
  - Async HTTP APIs
  - Monitoring
  - Human interaction



# Orchestrator Example – Human Interaction

- You can implement the Human Interaction pattern in the below example by using an orchestrator function
- The orchestrator uses a **durable timer** to request approval
- The orchestrator *escalates* if **timeout** occurs
- The orchestrator waits for an *external event*, such as a **notification** that's generated by a *human interaction*



Human Interaction w/Timeout Pattern

# Durable Functions – Supported Languages

---

- Durable Functions currently supports the following languages:
  - **C#** - Both precompiled class libraries and C# script
  - **F#** - Precompiled class libraries and F# script
    - F# script is only supported for version 1.x of the Azure Functions runtime
  - **JavaScript** - Supported only for version 2.x of the Azure Functions runtime
    - Requires version 1.7.0 of the Durable Functions extension, or a later version

**NOTE:** Durable Functions are billed the *same* as Azure Functions.



# Demo 4 – Creating Durable Functions Using C#

# Summary

## Azure App Service API Apps

- API apps in Azure App Service offer features that make it easier to **develop, host, and consume APIs** in the cloud and on-premises
- With API apps you get
  - Enterprise grade security
  - Simple access control
  - Hybrid connectivity
  - Automatic SDK generation
  - and Seamless integration with Logic Apps



edureka!

Copyright © edureka and/or its affiliates. All rights reserved.

## Shared Core Features With WebApps

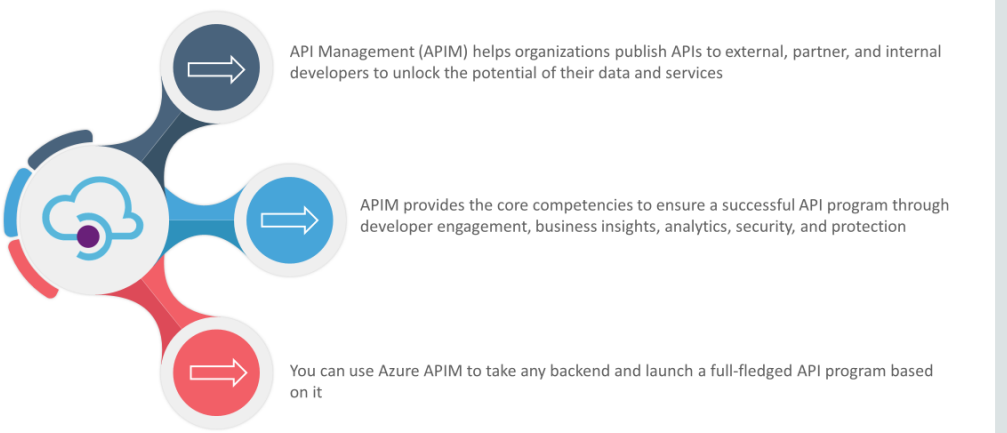
Azure API Apps are hosted using Azure App Services offering meaning they share common features that are offered for Web Apps:



edureka!

Copyright © edureka and/or its affiliates. All rights reserved.

## What Is API Management?



edureka!

Copyright © edureka and/or its affiliates. All rights reserved.

## Azure Functions Overview

Azure Functions is a solution for easily running small pieces of code, or "functions," in the cloud



Write just the code you need for the problem at hand, without worrying about a whole application or the infrastructure to run it

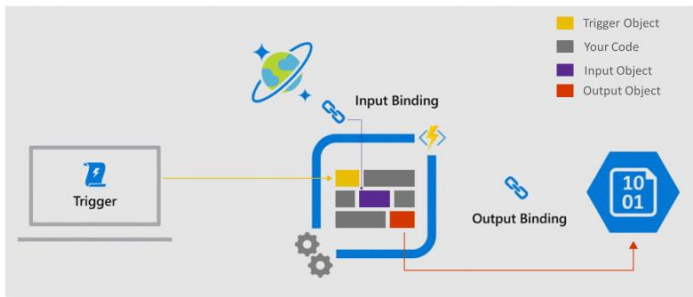
It is an event-driven serverless compute platform that can also solve complex orchestration problems

edureka!

Copyright © edureka and/or its affiliates. All rights reserved.

## Triggers And Bindings Implemented In Functions

- Triggers and bindings let you avoid hardcoding access to other services
- Your function receives data (for example, the content of a queue message) in function parameters
- You send data (for example, to create a queue message) by using the return value of the function

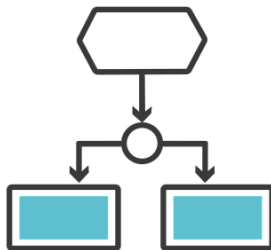


edureka!

Copyright © edureka and/or its affiliates. All rights reserved.

## Durable Functions – Application Patterns

- The primary use case for Durable Functions is simplifying complex, stateful coordination requirements in serverless applications
- The following are some typical application patterns that can benefit from Durable Functions:
  - Chaining
  - Fan-out/fan-in
  - Async HTTP APIs
  - Monitoring
  - Human interaction



edureka!

Copyright © edureka and/or its affiliates. All rights reserved.



# Questions



# FEEDBACK





# Thank You

---

For more information please visit our website  
[www.edureka.co](http://www.edureka.co)