



Advanced Serverless Workshop

Azure

Who is Chad Green

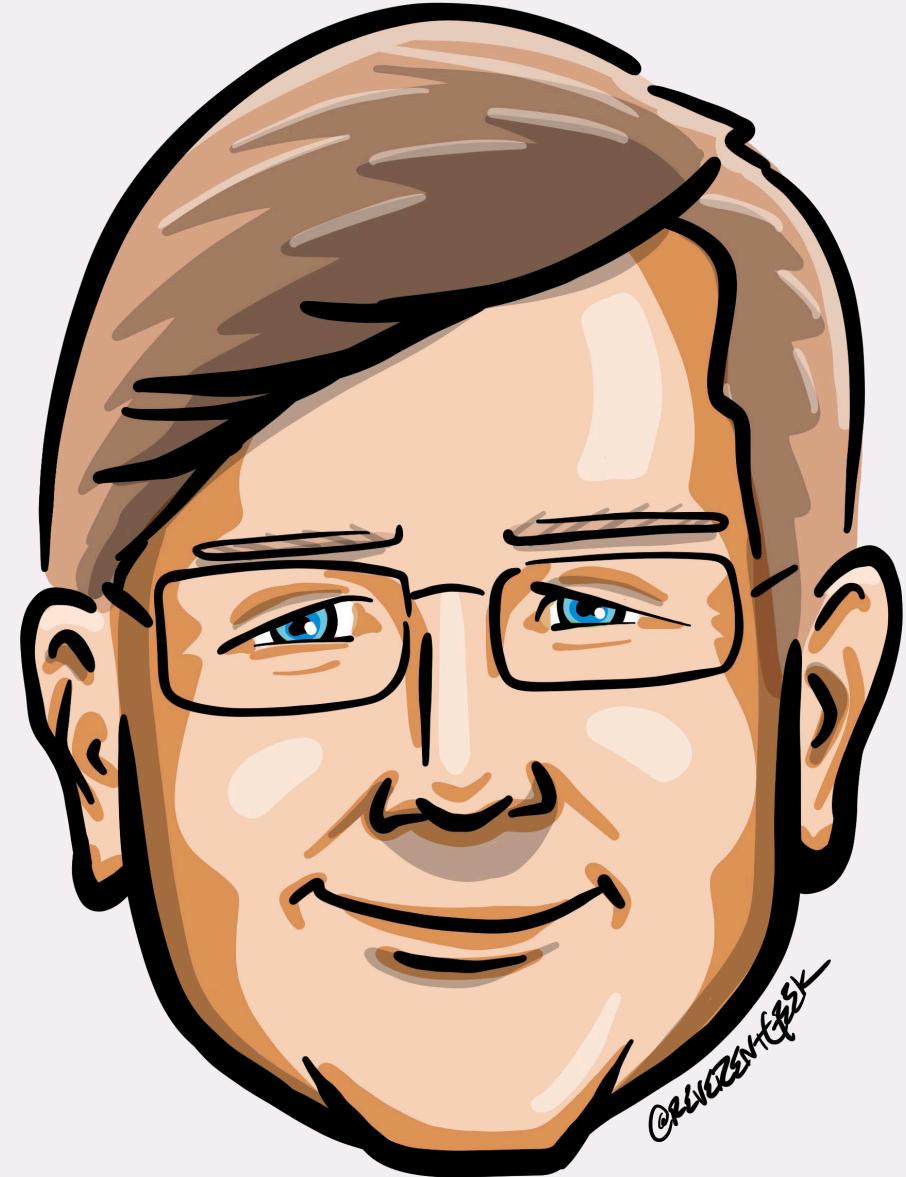
✉ chadgreen@chadgreen.com

.twitch TaleLearnCode

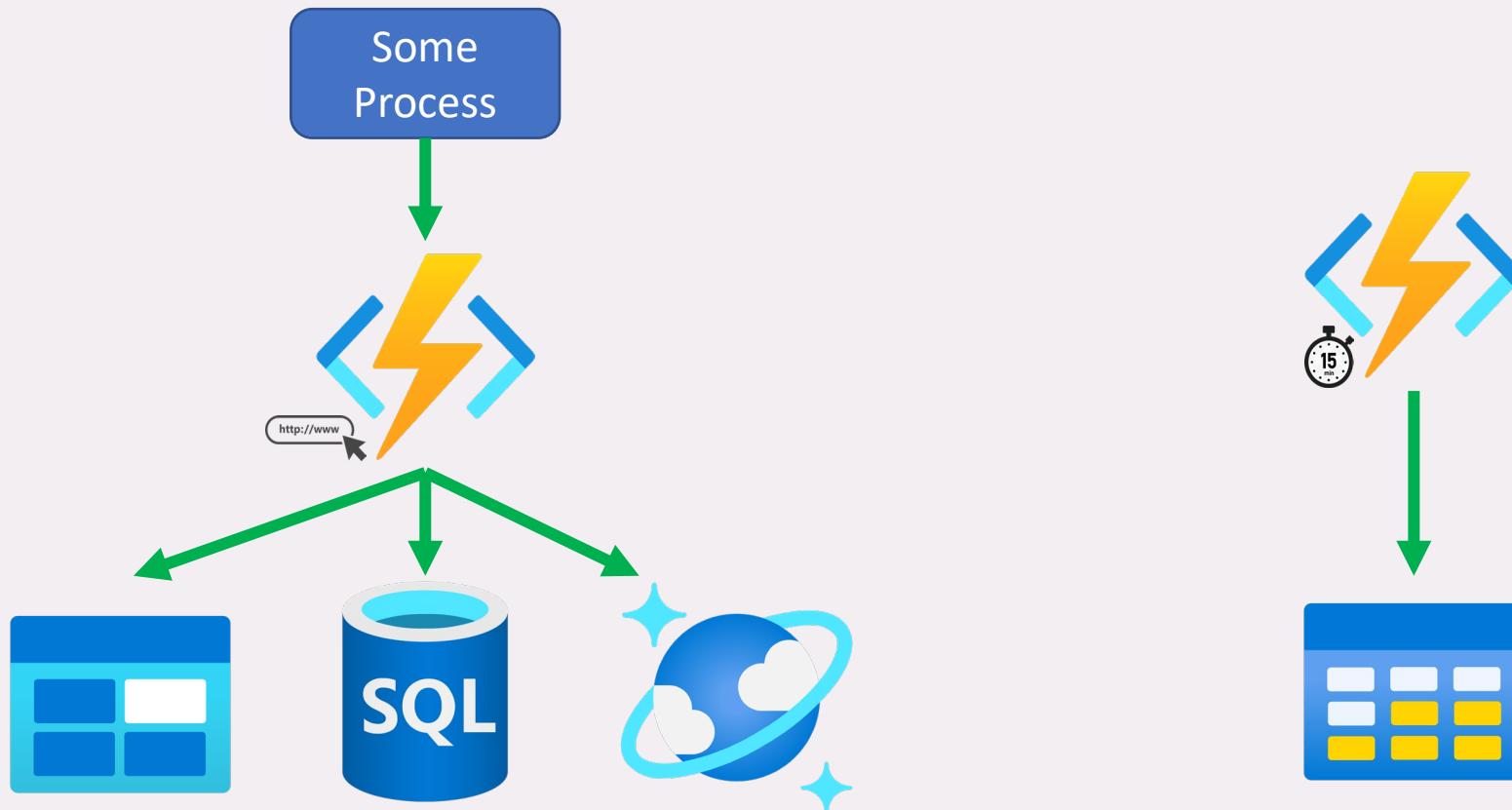
🌐 ChadGreen.com

🐦 ChadGreen & TaleLearnCode

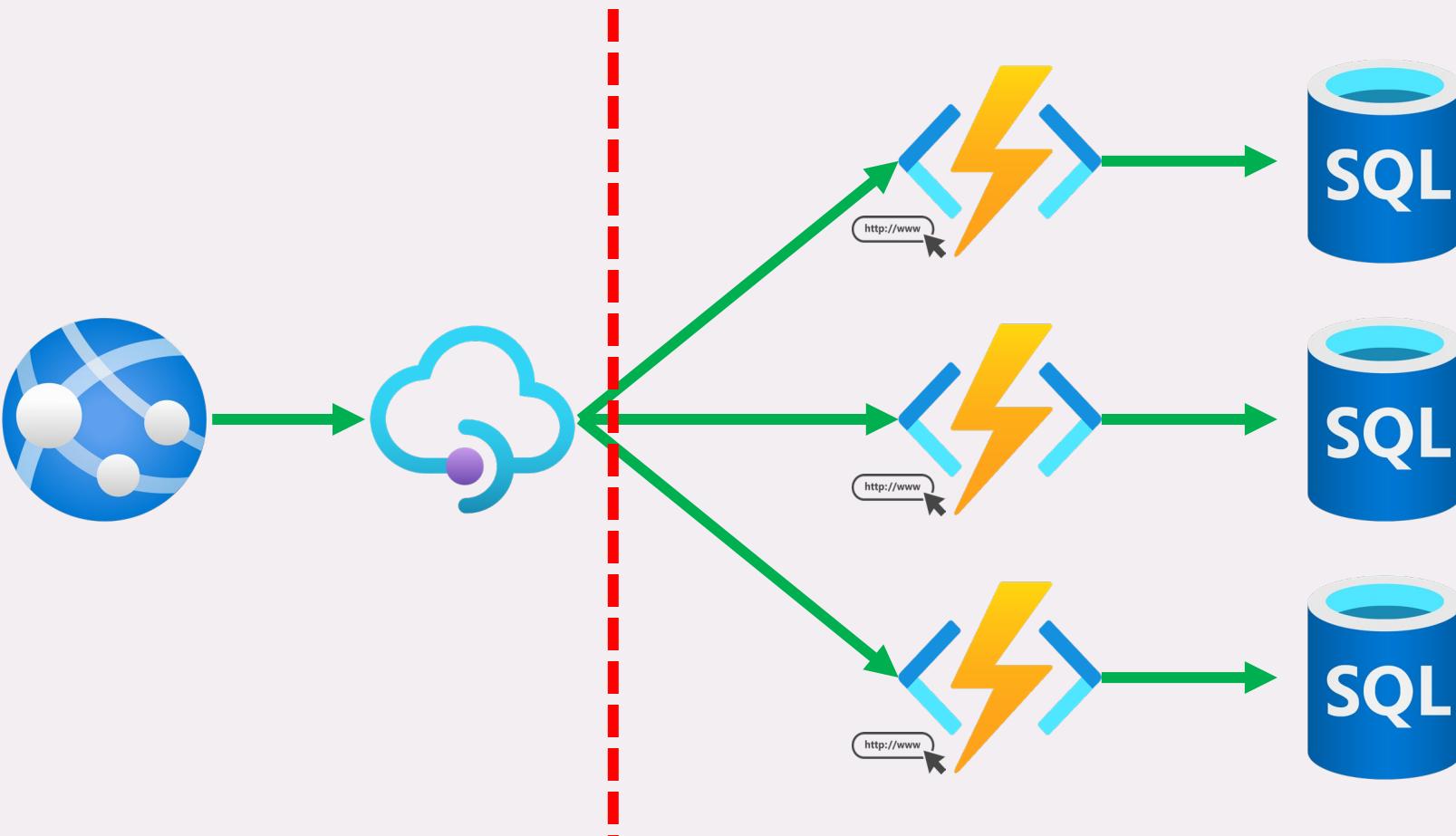
linkedin ChadwickEGreen



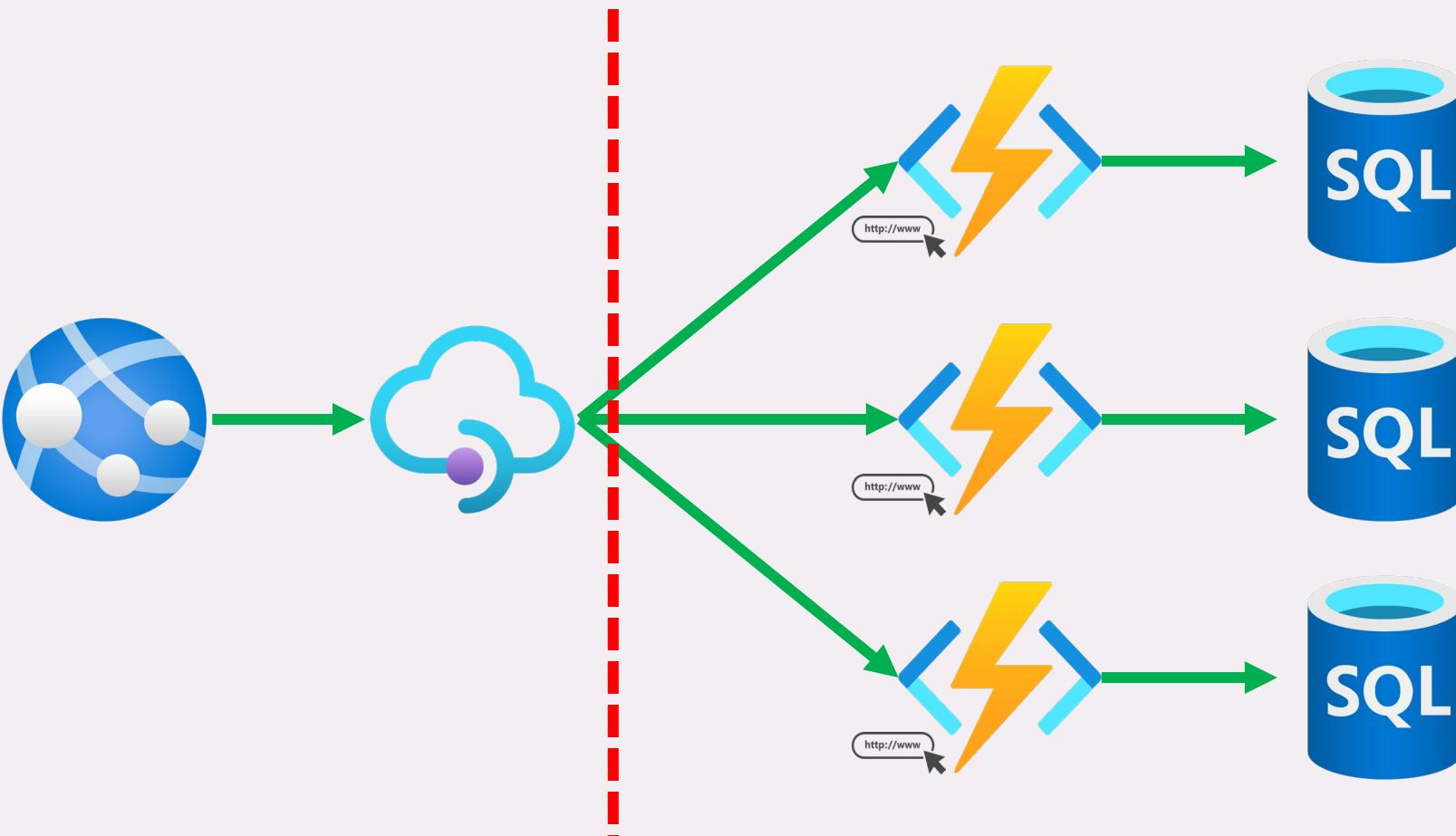
Simple Serverless Functionality



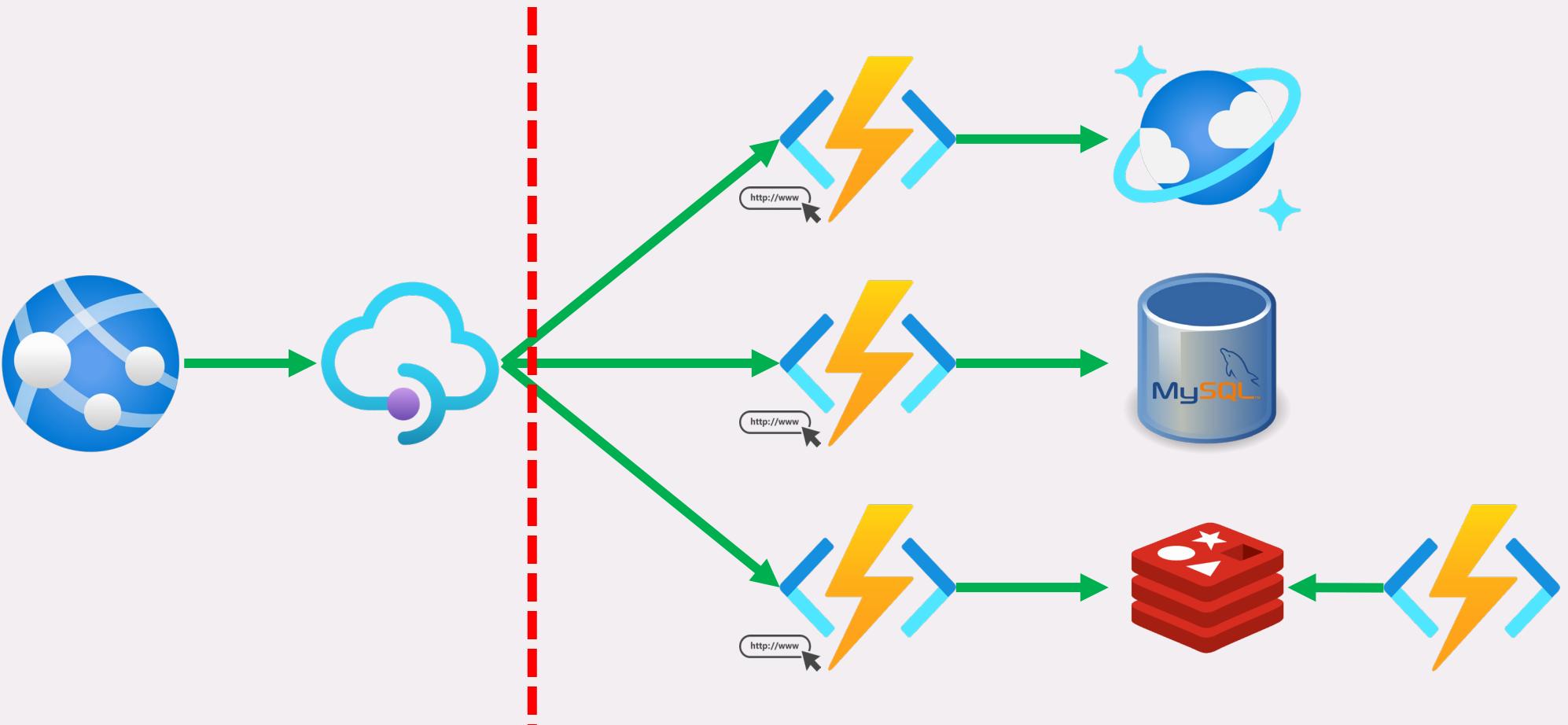
CRUD Microservice



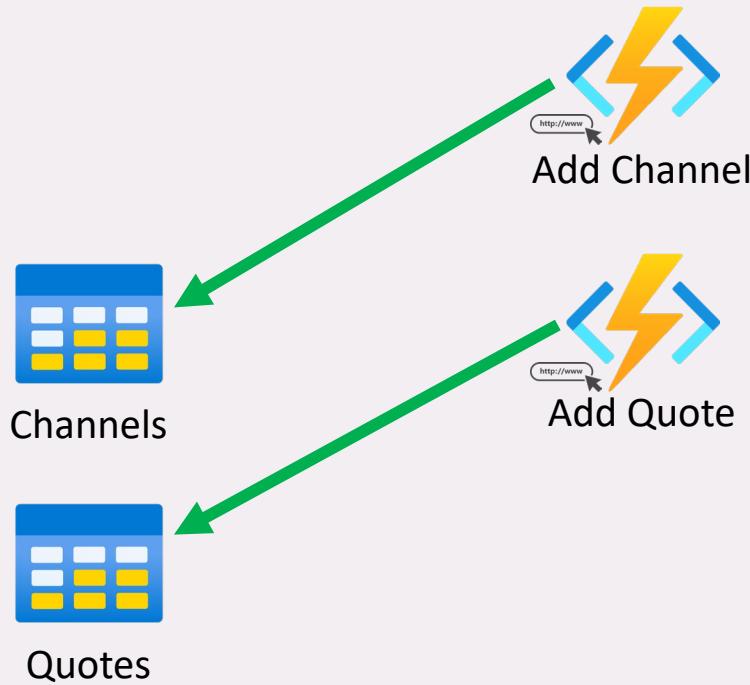
CRUD Microservice



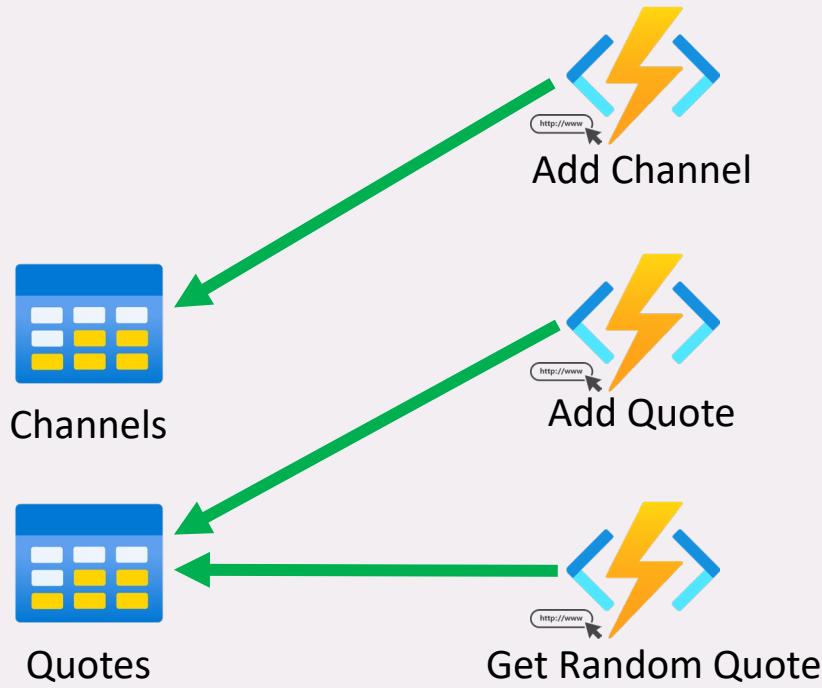
CRUD Microservice



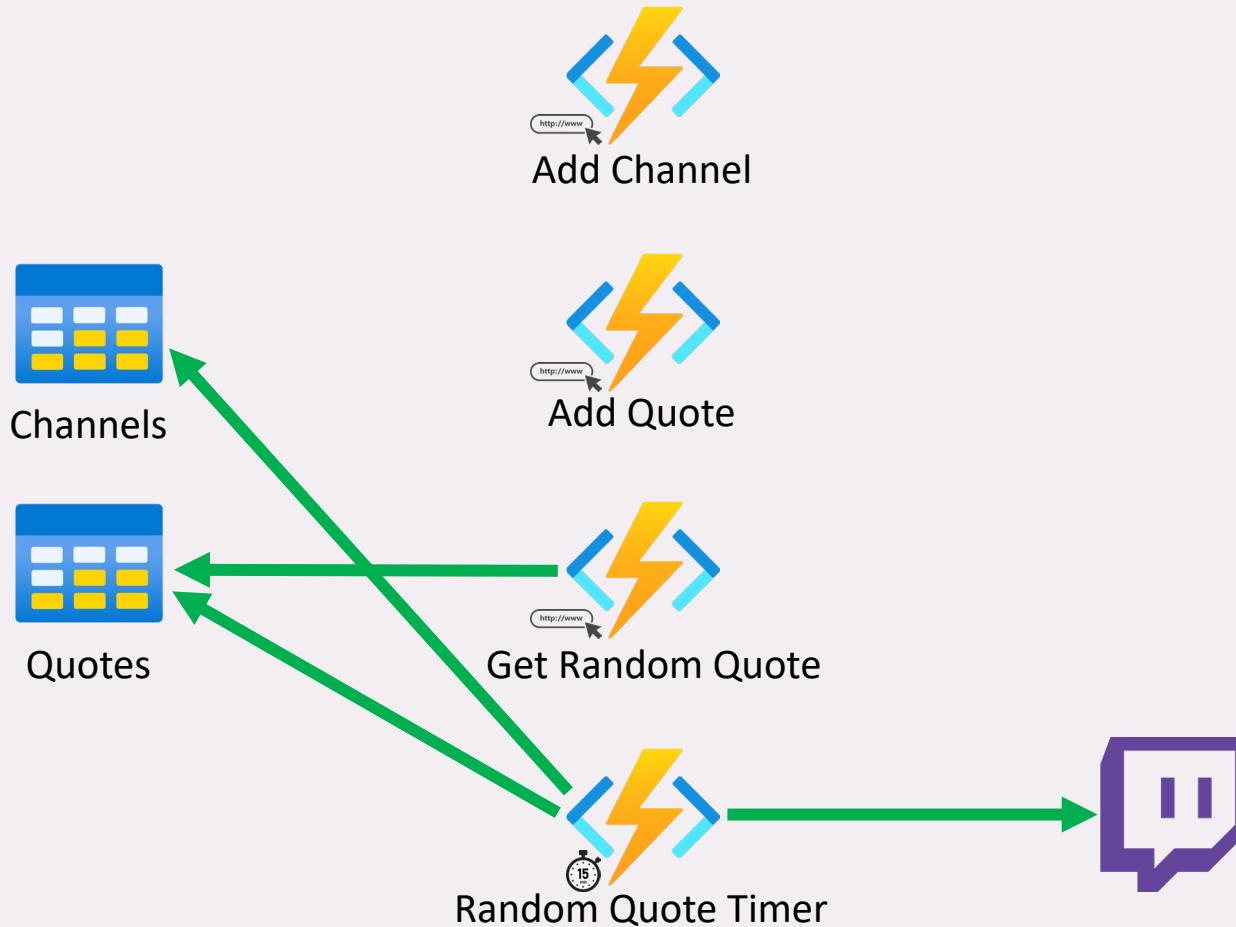
Random Quote Generator



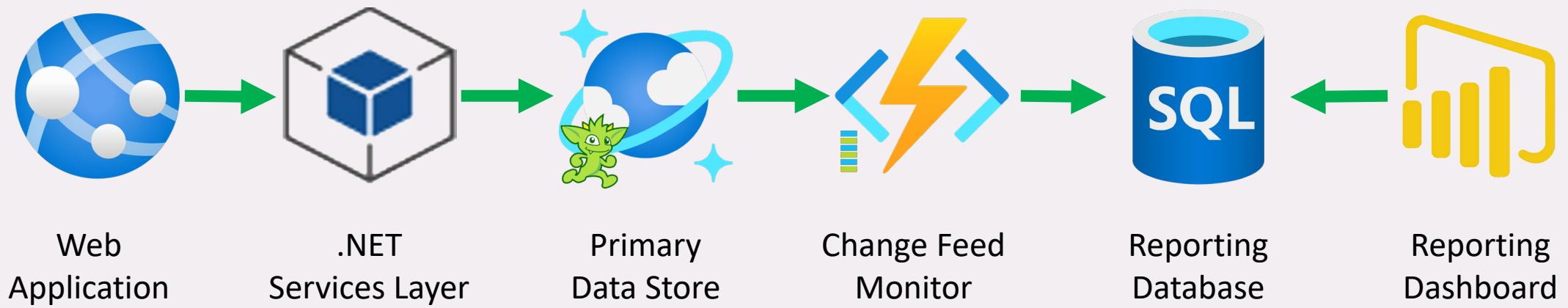
Random Quote Generator



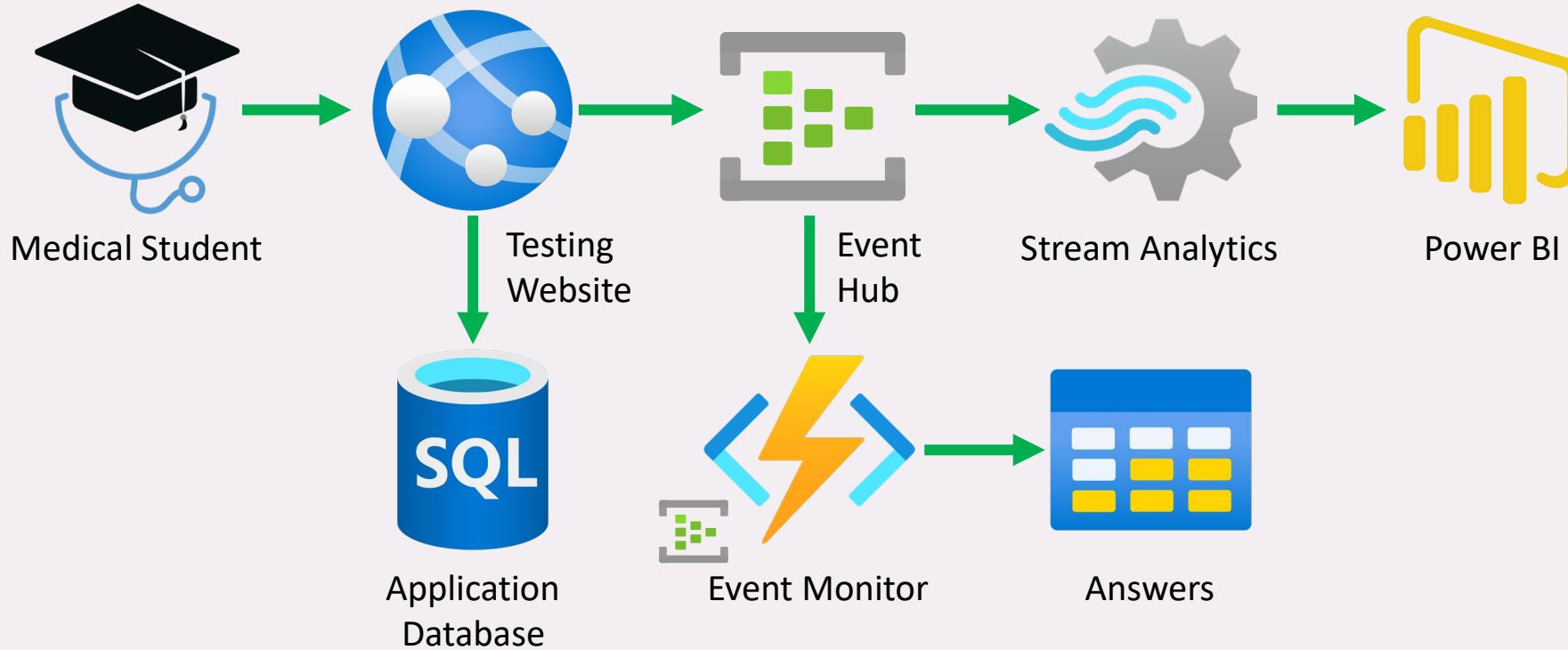
Random Quote Generator



Replicating Data



Real-Time Reporting



Glennis Platform



Portfolio



Sales



Leasing &
Billing



Resident
Care



eMAR



Resident
Experience



Quality



Financials



Insights



Glennis Platform



Portfolio



Sales



Leasing &
Billing



Quality



Financials



Insights

Glennis Platform

Glennis Portfolio

- Portfolio Management
- Care and Services Management
- Pricing Management
- Marketing Data Management
- Marketing APIs

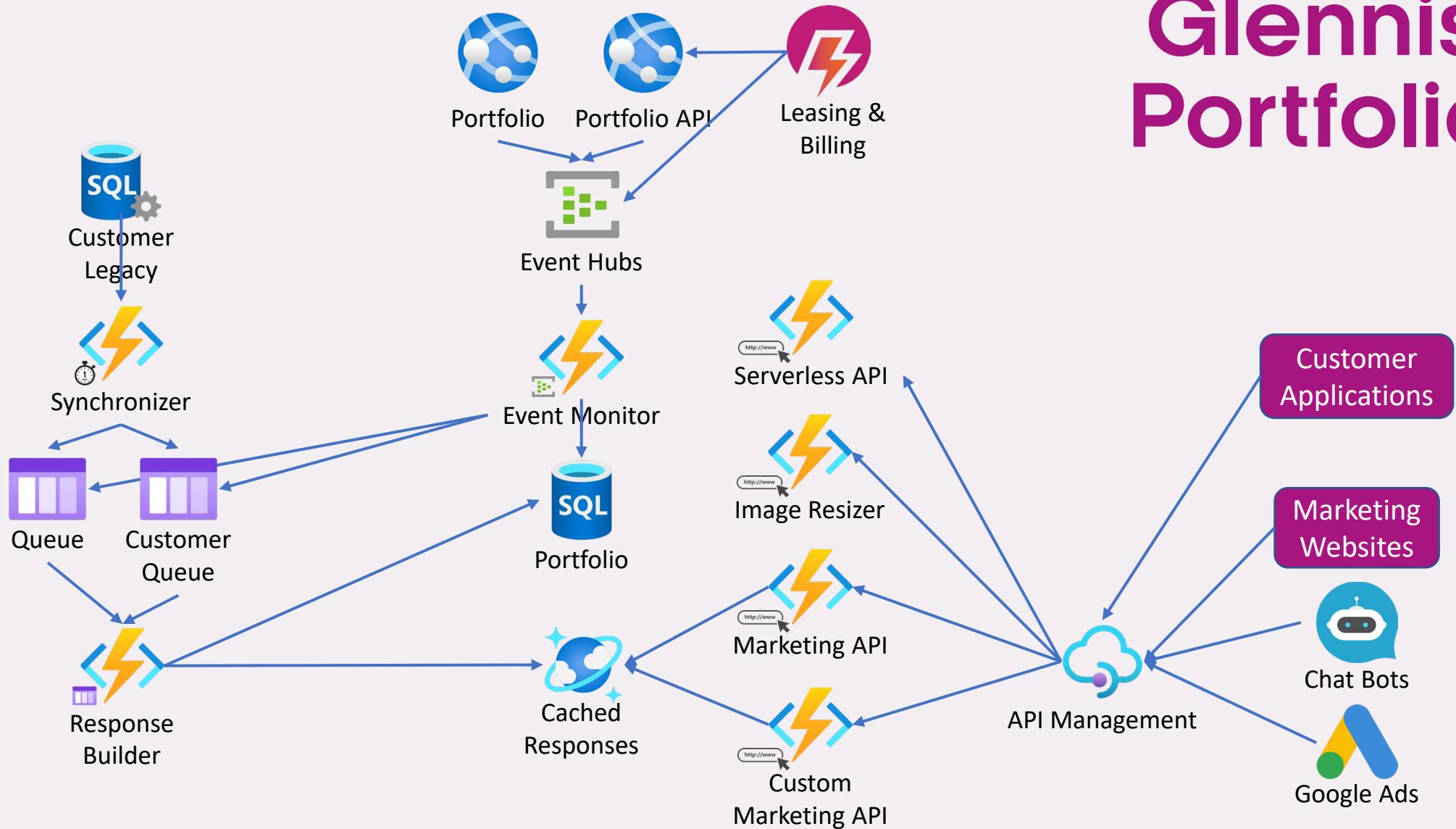
Glennis Platform



Glennis Portfolio

- Portfolio Management
- Care and Services Management
- Pricing Management
- Marketing Data Management
- Marketing APIs

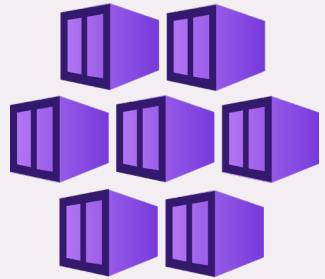
Glennis Portfolio



Microsoft Azure Serverless Offerings & Tools

Advanced Serverless Workshop: Azure

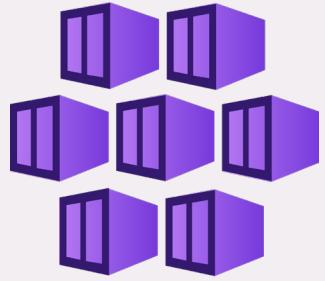
Azure Serverless Offerings & Tools



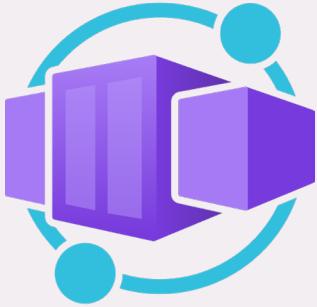
Azure Kubernetes Service

Compute

Azure Serverless Offerings & Tools



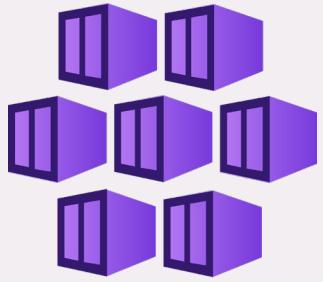
Azure Kubernetes Service



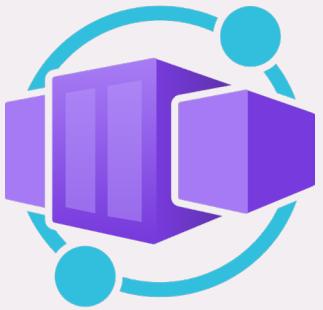
Azure Container Apps

Compute

Azure Serverless Offerings & Tools



Azure Kubernetes Service



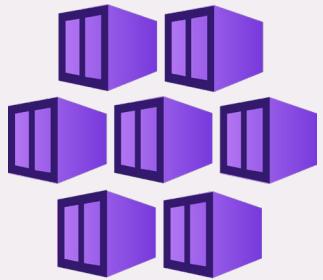
Azure Container Apps



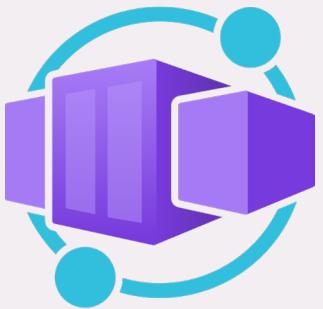
Azure App Service

Compute

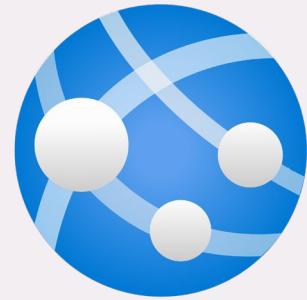
Azure Serverless Offerings & Tools



Azure Kubernetes Service



Azure Container Apps



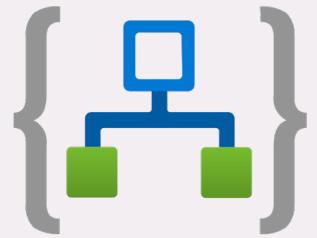
Azure App Service



Azure Functions

Compute

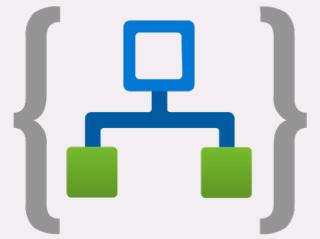
Azure Serverless Offerings & Tools



Azure Logic Apps

Workflows and Integration

Azure Serverless Offerings & Tools



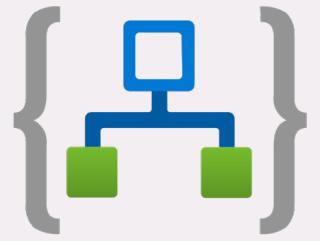
Azure Logic Apps



API Management

Workflows and Integration

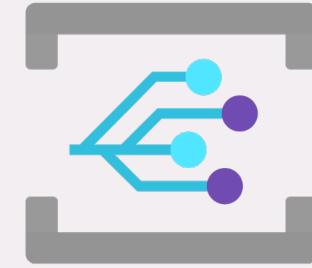
Azure Serverless Offerings & Tools



Azure Logic Apps



API Management



Azure Event Grid

Workflows and Integration

Azure Serverless Offerings & Tools



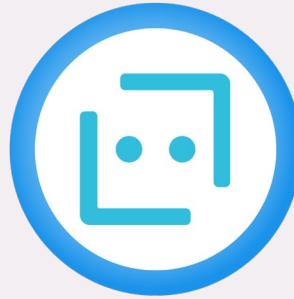
Azure Cognitive Services

AI and Machine Learning

Azure Serverless Offerings & Tools



Azure Cognitive Services



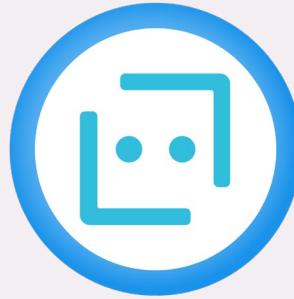
Azure Bot Services

AI and Machine Learning

Azure Serverless Offerings & Tools



Azure Cognitive Services



Azure Bot Services



Azure Machine Learning

AI and Machine Learning

Azure Serverless Offerings & Tools



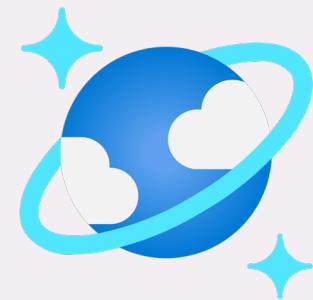
Azure SQL Database

Database

Azure Serverless Offerings & Tools



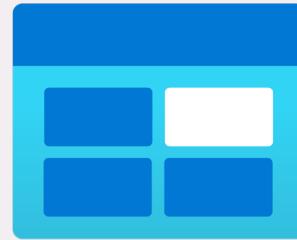
Azure SQL Database



Azure Cosmos DB

Database

Azure Serverless Offerings & Tools



Azure Blob Storage

Storage

Azure Serverless Offerings & Tools



Azure Monitor

Monitoring

Azure Serverless Offerings & Tools

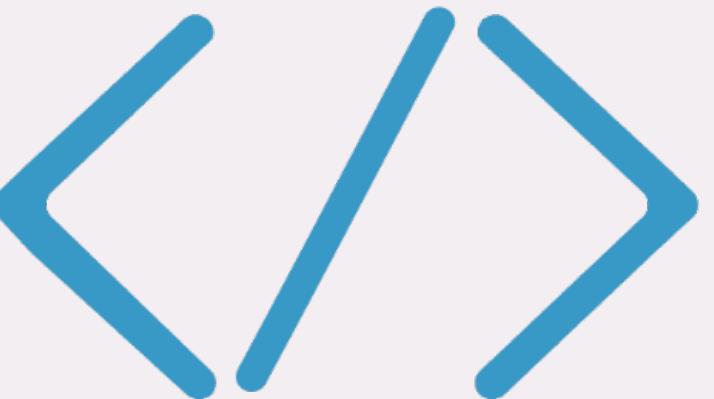


Azure Stream Analytics

Analytics

Azure Functions

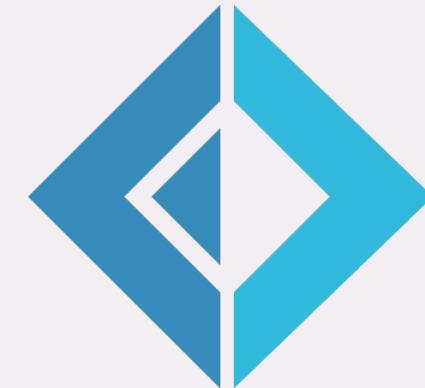
Code



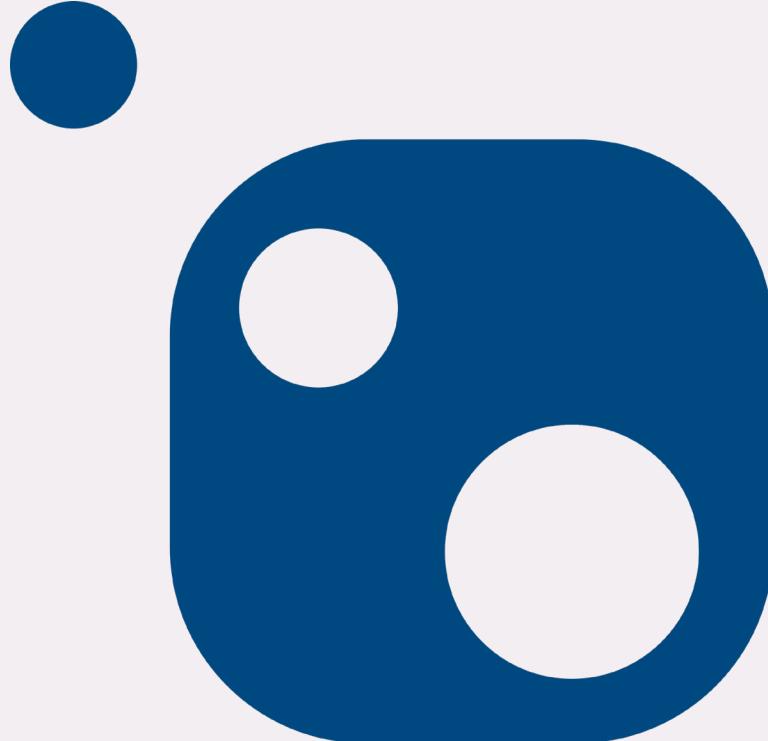
Events + data



Choice of Language



Bring your own dependencies

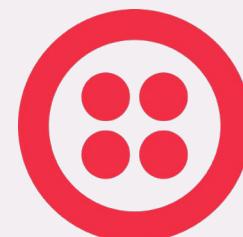


Simplified Integration

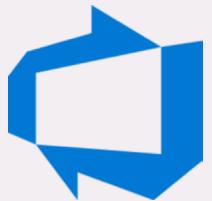
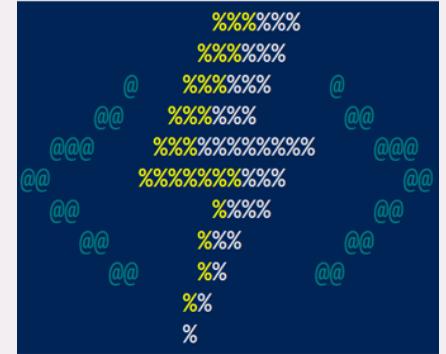
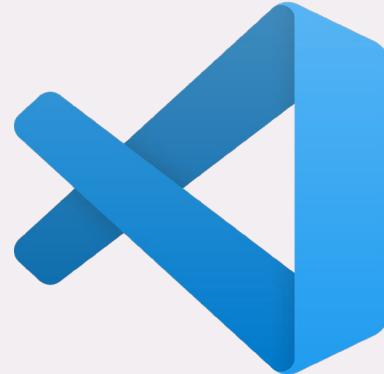
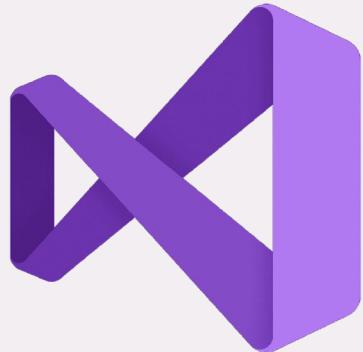


 RabbitMQ



 twilio

Flexible Development



Maven™



Many Deployment Options

Consumption



Serverless

Many Deployment Options

Consumption



Serverless

App Service Plan



Free, Basic,
Standard,
Premium

Many Deployment Options

Consumption



Serverless

App Service Plan



Free, Basic,
Standard,
Premium

App Service Environment



Network
Isolation

Many Deployment Options

Consumption



Serverless

App Service Plan



Free, Basic,
Standard,
Premium

App Service Environment



Network
Isolation

Azure Stack



On Premises

Many Deployment Options

Consumption



Serverless

App Service Plan



Free, Basic,
Standard,
Premium

App Service Environment



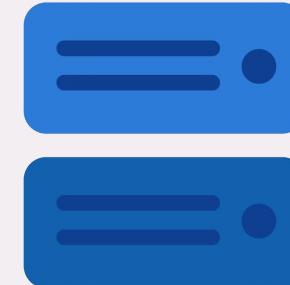
Network
Isolation

Azure Stack



On Premises

Azure Functions Runtime



Functions on
Your Server

Many Deployment Options

Consumption



Serverless

App Service Plan



Free, Basic,
Standard,
Premium

App Service Environment



Network
Isolation

Azure Stack



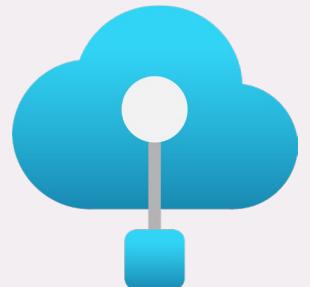
On Premises

Azure Functions Runtime



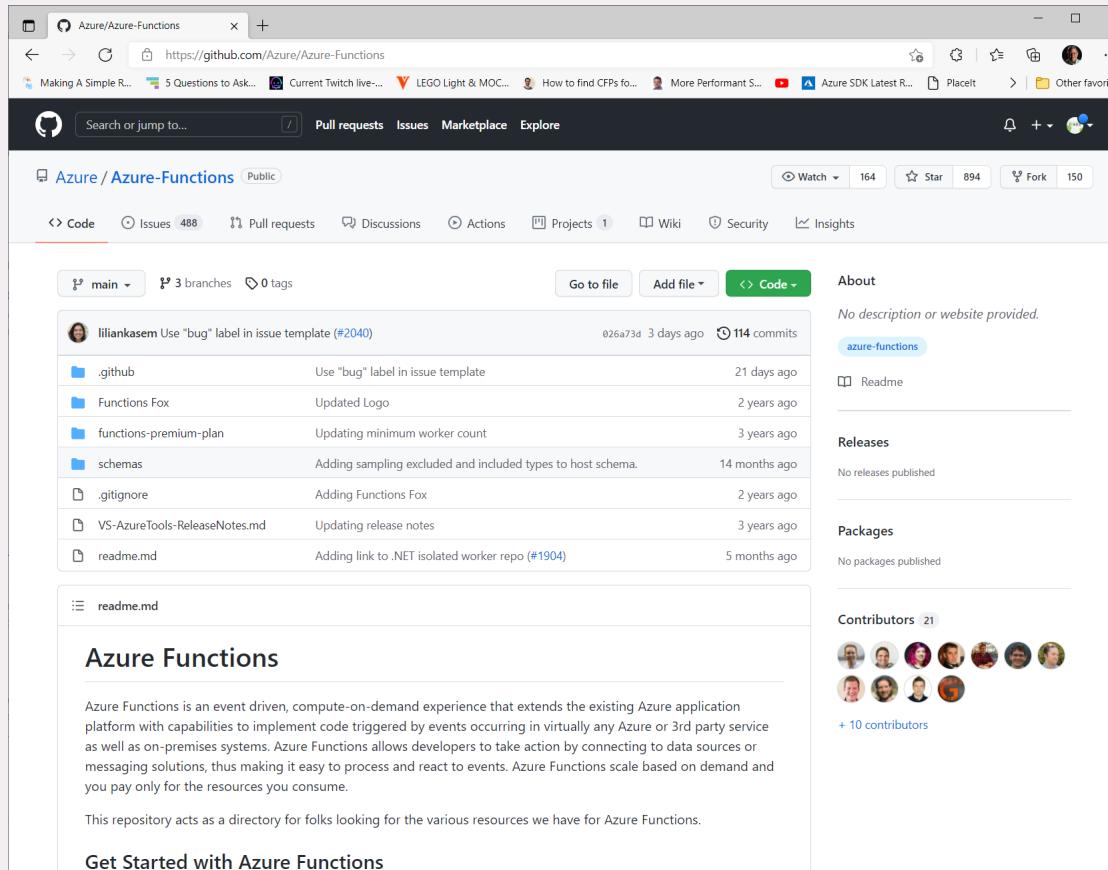
Functions on
Your Server

Azure IoT Edge



On Devices

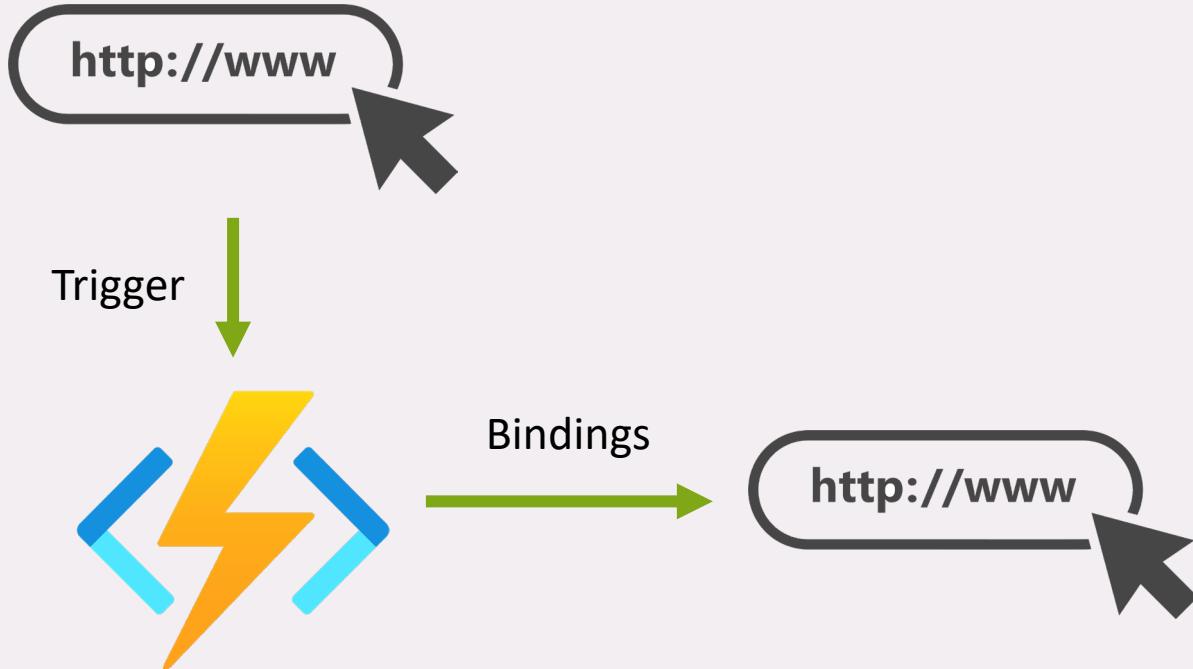
Open-Source



Supported Triggers and Bindings

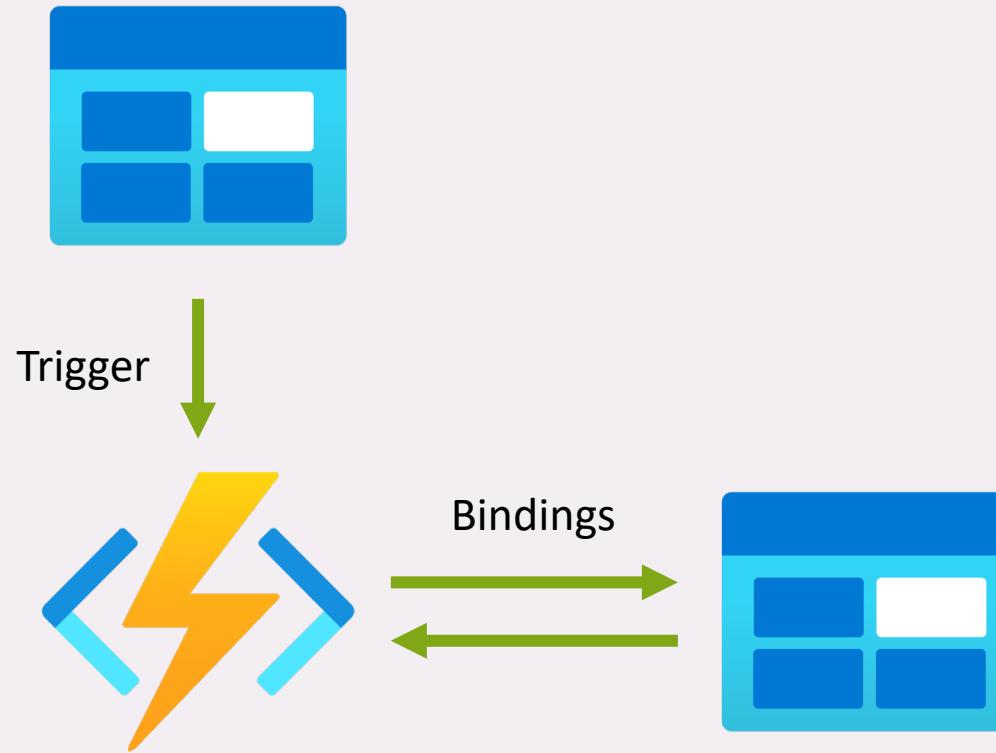
HTTP

Trigger	✓
Input	✗
Output	✓
Consumption	✓
Premium	✓
Dedicated	✓
C# In-Process	✓
C# Isolated	✓
Java	✓
JavaScript	✓
PowerShell	✓
Python	✓



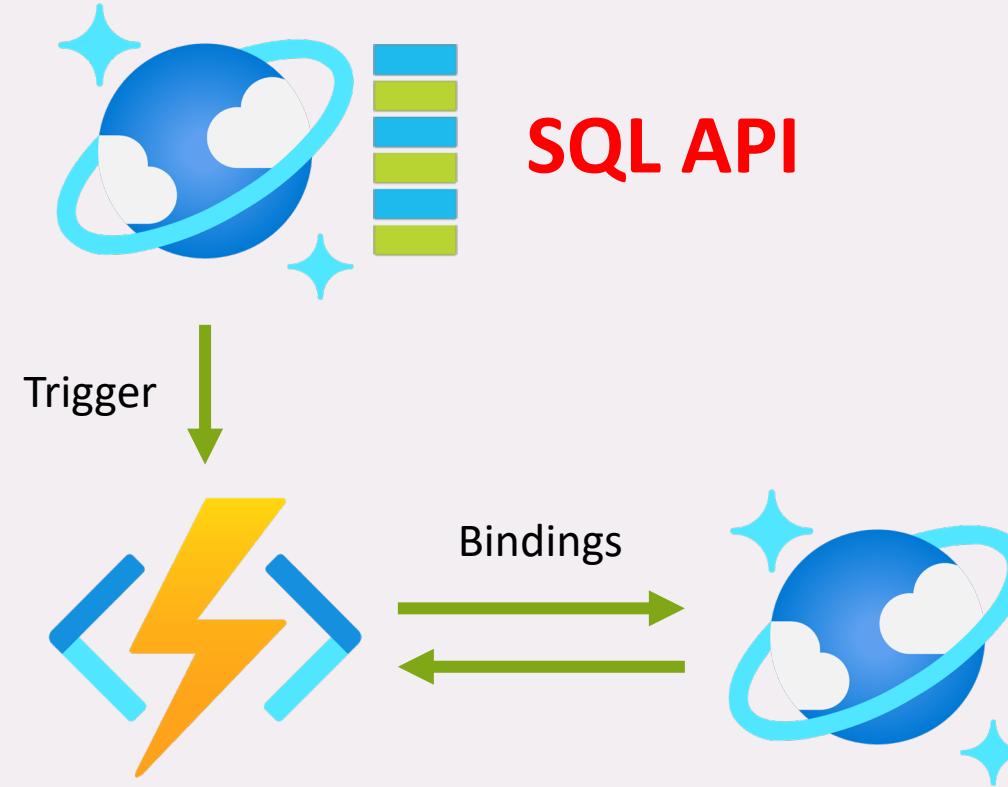
Blob

Trigger	✓
Input	✓
Output	✓
Consumption	✓
Premium	✓
Dedicated	✓
C# In-Process	✓
C# Isolated	✓
Java	✓
JavaScript	✓
PowerShell	✓
Python	✓



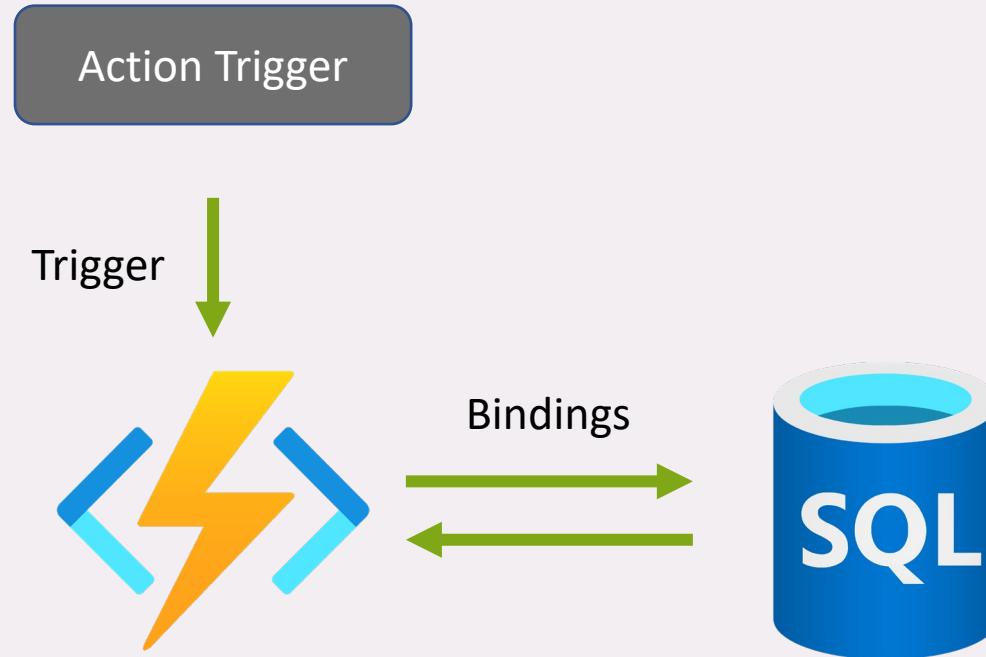
Cosmos DB

Trigger	✓
Input	✓
Output	✓
Consumption	✓
Premium	✓
Dedicated	✓
C# In-Process	✓
C# Isolated	✓
Java	✓
JavaScript	✓
PowerShell	✓
Python	✓



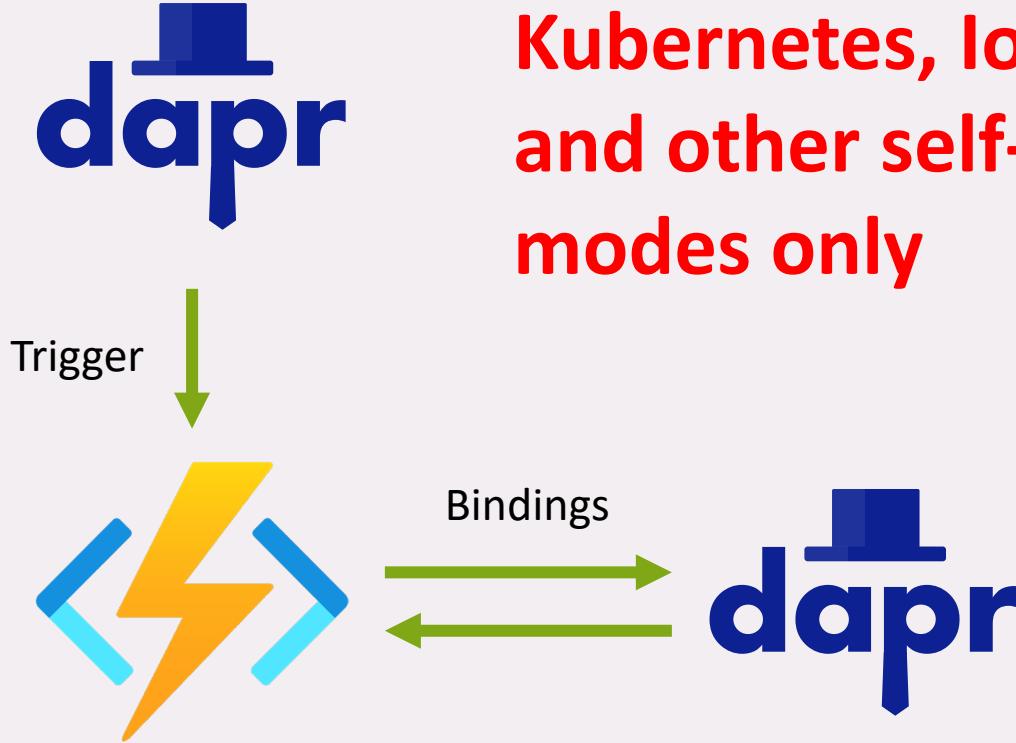
Azure SQL (Preview)

Trigger	✗
Input	✓
Output	✓
Consumption	✓
Premium	✓
Dedicated	✓
C# In-Process	✓
C# Isolated	✗
Java	✗
JavaScript	✓
PowerShell	✓
Python	✓



Dapr

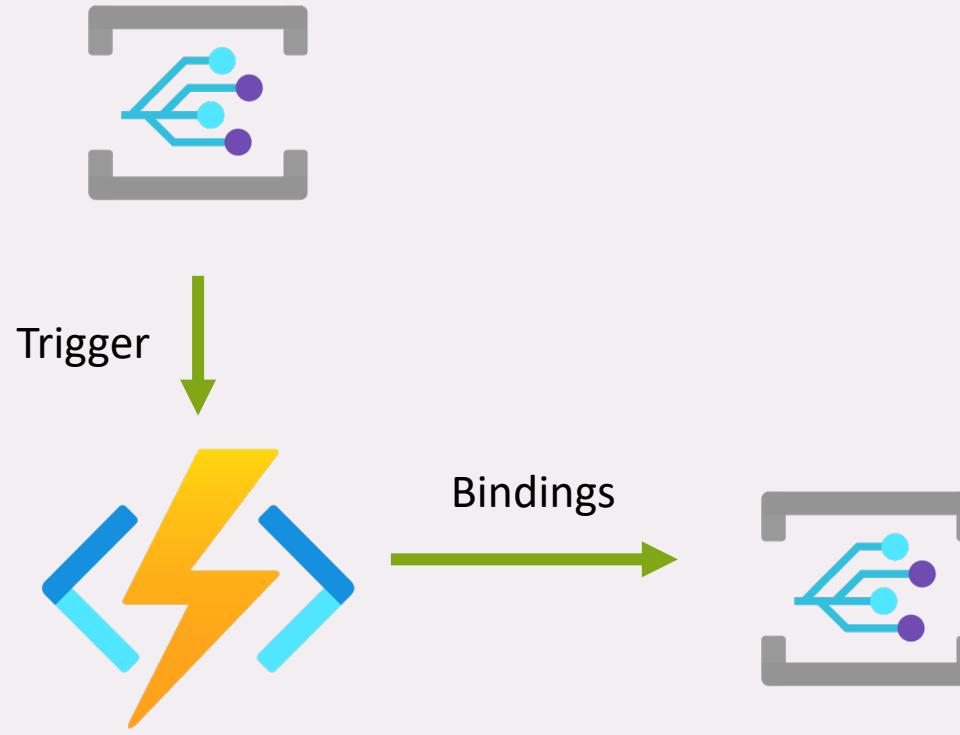
Tigger	✓
Input	✓
Output	✓
Consumption	✗
Premium	✗
Dedicated	✓
C# In-Process	✓
C# Isolated	✗
Java	✗
JavaScript	✓
PowerShell	✗
Python	✓



**Supported only in
Kubernetes, IoT Edge,
and other self-hosted
modes only**

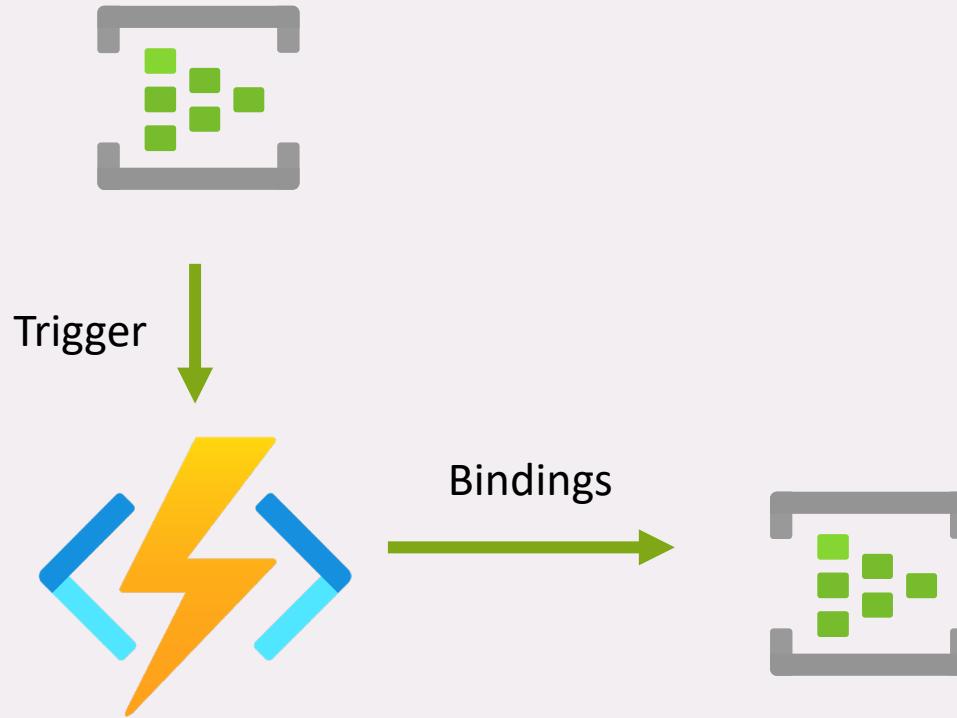
Event Grid

Trigger	✓
Input	✗
Output	✓
Consumption	✓
Premium	✓
Dedicated	✓
C# In-Process	✓
C# Isolated	✓
Java	✓
JavaScript	✓
PowerShell	✓
Python	✓



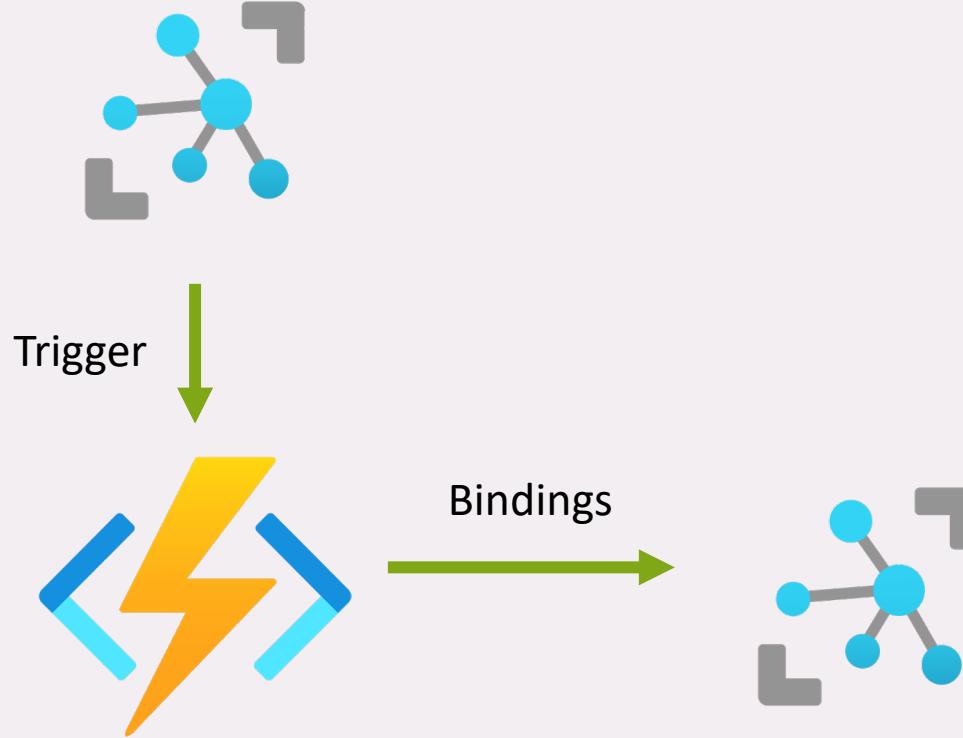
Event Hub

Trigger	✓
Input	✗
Output	✓
Consumption	✓
Premium	✓
Dedicated	✓
C# In-Process	✓
C# Isolated	✓
Java	✓
JavaScript	✓
PowerShell	✓
Python	✓



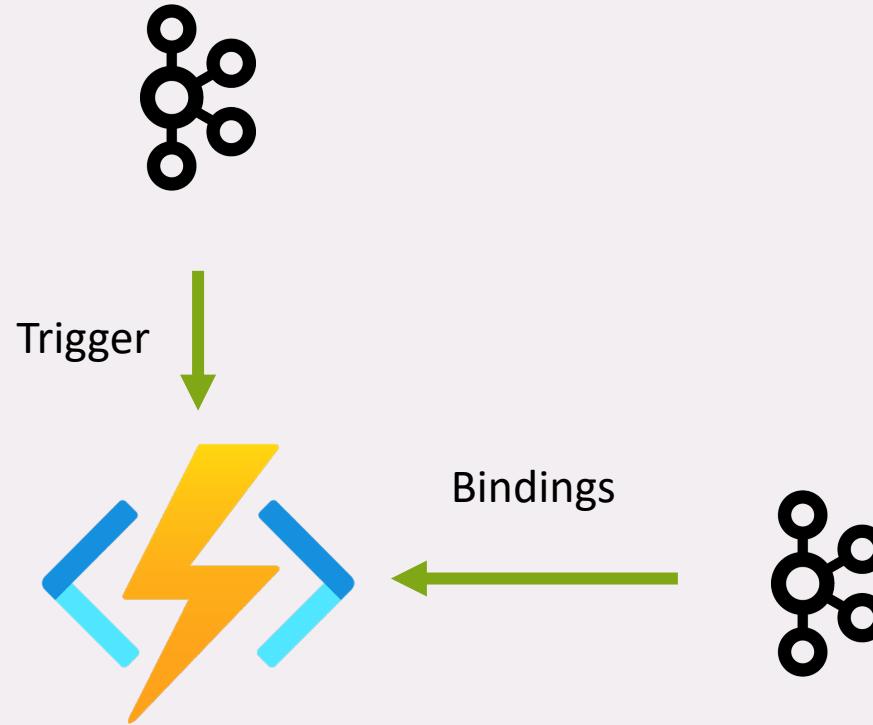
IoT Hub

Trigger	✓
Input	✗
Output	✓
Consumption	✓
Premium	✓
Dedicated	✓
C# In-Process	✓
C# Isolated	✓
Java	✓
JavaScript	✓
PowerShell	✓
Python	✓



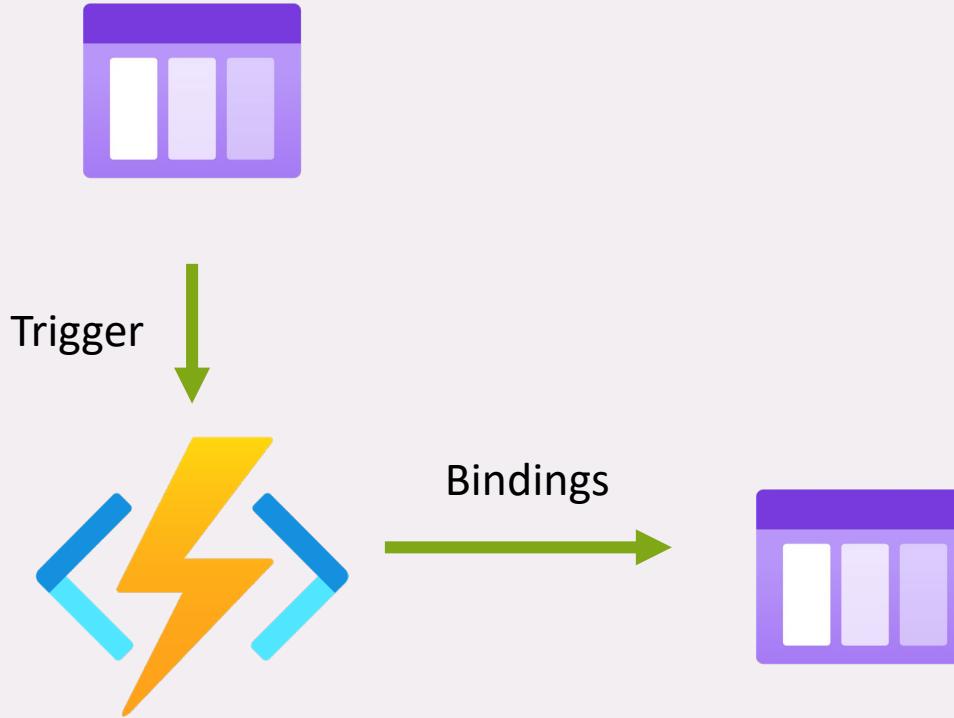
Kafka

Trigger	✓
Input	✗
Output	✓
Consumption	✗
Premium	✓
Dedicated	✓
C# In-Process	✓
C# Isolated	✓
Java	✓
JavaScript	✓
PowerShell	✓
Python	✓



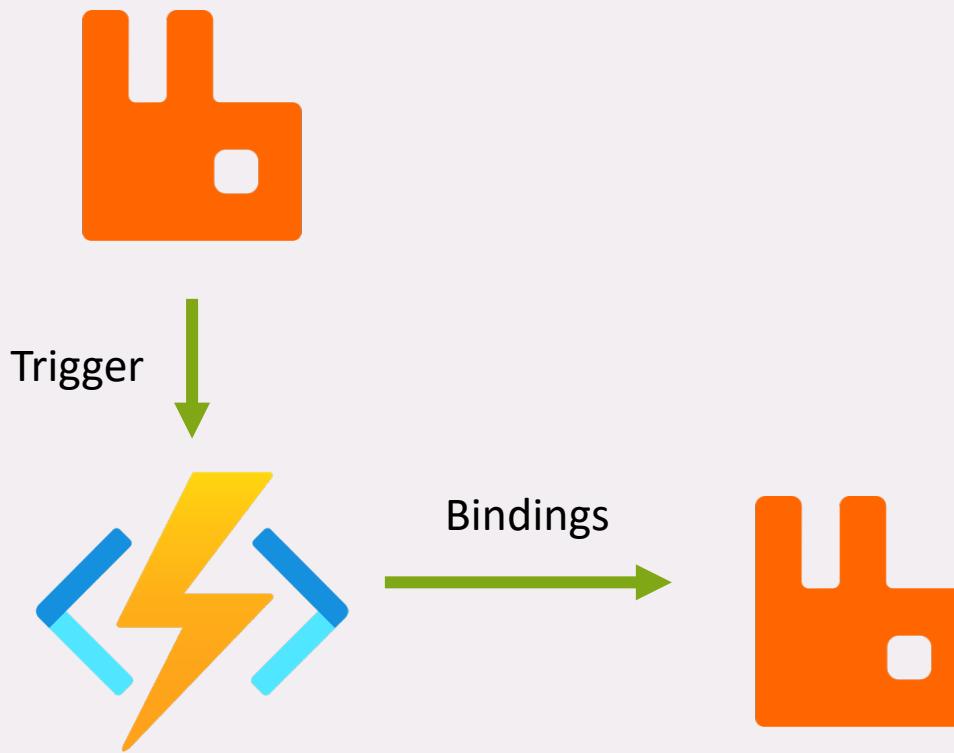
Queue Storage

Trigger	✓
Input	✗
Output	✓
Consumption	✓
Premium	✓
Dedicated	✓
C# In-Process	✓
C# Isolated	✓
Java	✓
JavaScript	✓
PowerShell	✓
Python	✓



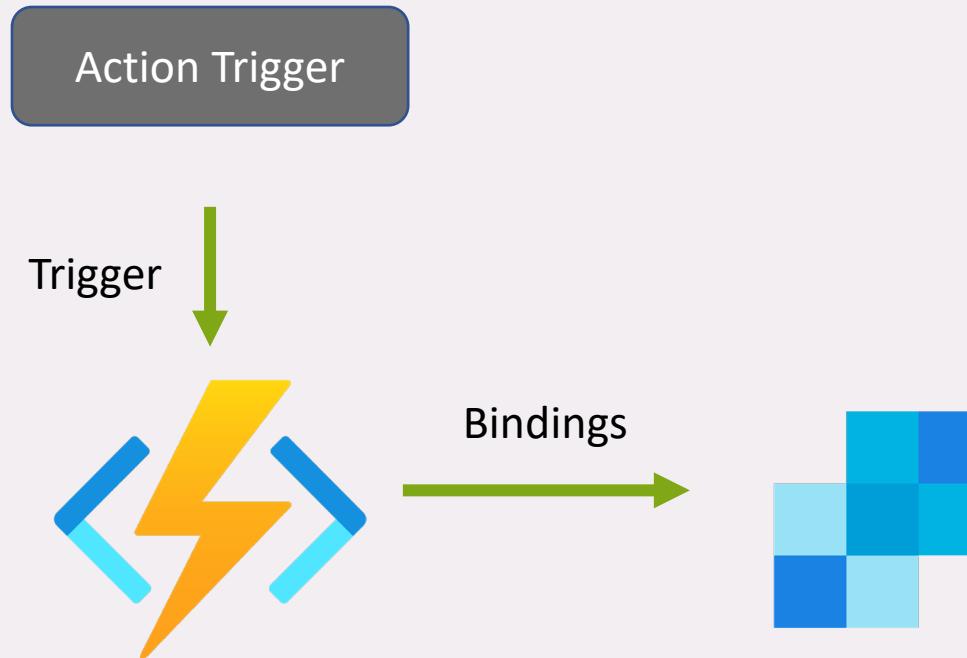
RabbitMQ

Trigger	✓
Input	✗
Output	✓
Consumption	✗
Premium	✓
Dedicated	✓
C# In-Process	✓
C# Isolated	✓
Java	✓
JavaScript	✓
PowerShell	✓
Python	✓



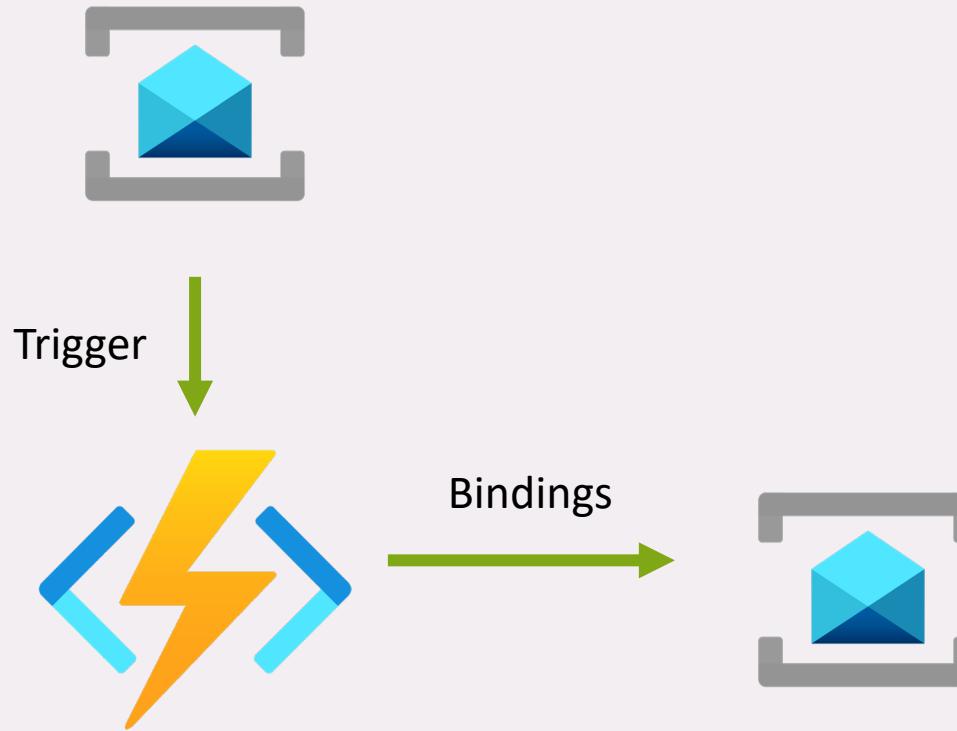
SendGrid

Trigger	
Input	
Output	
Consumption	
Premium	
Dedicated	
C# In-Process	
C# Isolated	
Java	
JavaScript	
PowerShell	
Python	



Service Bus

Trigger	✓
Input	✗
Output	✓
Consumption	✓
Premium	✓
Dedicated	✓
C# In-Process	✓
C# Isolated	✓
Java	✓
JavaScript	✓
PowerShell	✓
Python	✓



SignalR

Trigger	✓
Input	✓
Output	✓
Consumption	✓
Premium	✓
Dedicated	✓
C# In-Process	✓
C# Isolated	✓
Java	✓
JavaScript	✓
PowerShell	✓
Python	✓

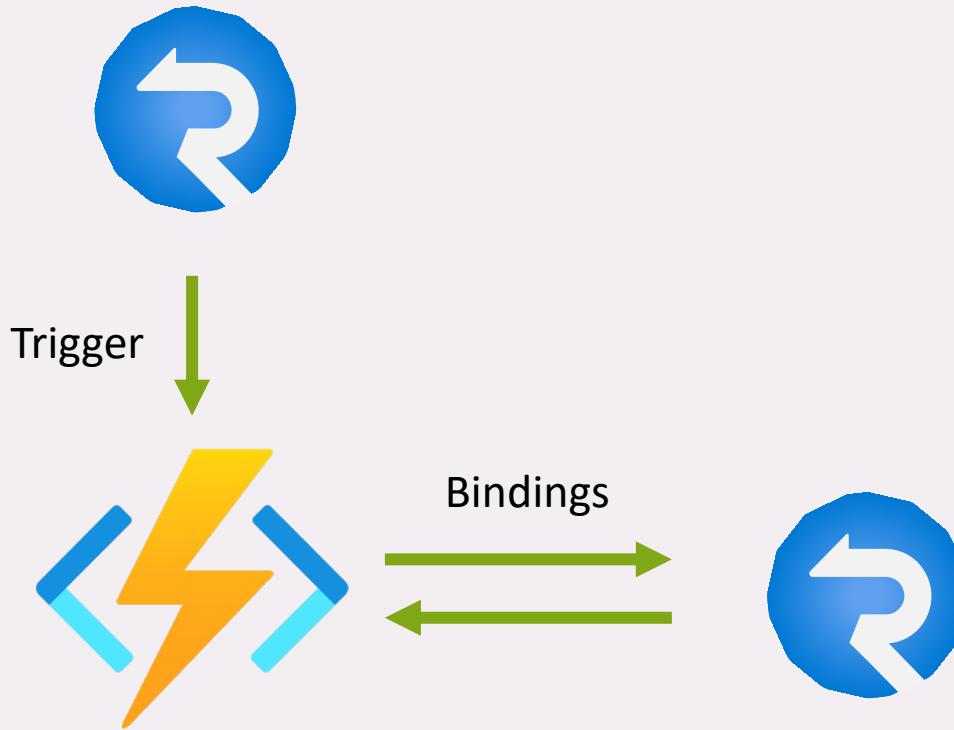
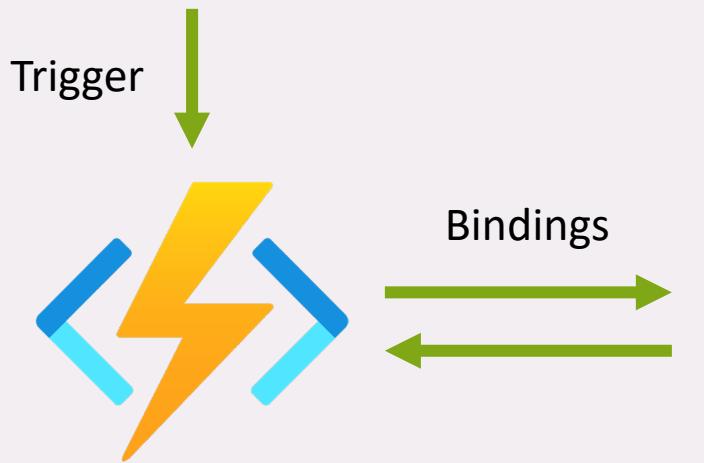


Table Storage

Trigger	
Input	
Output	
Consumption	
Premium	
Dedicated	
C# In-Process	
C# Isolated	
Java	
JavaScript	
PowerShell	
Python	

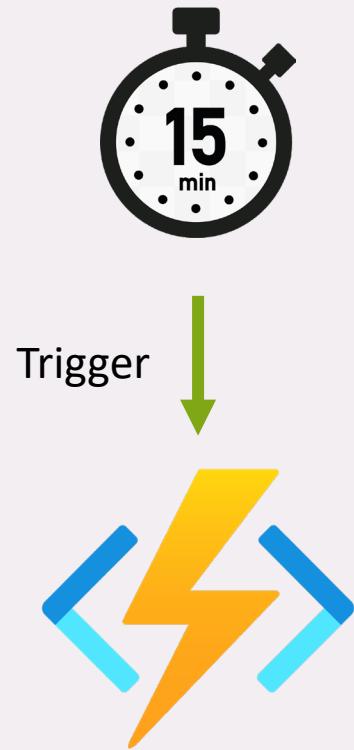
Action Trigger



**Cosmos DB support
in preview**

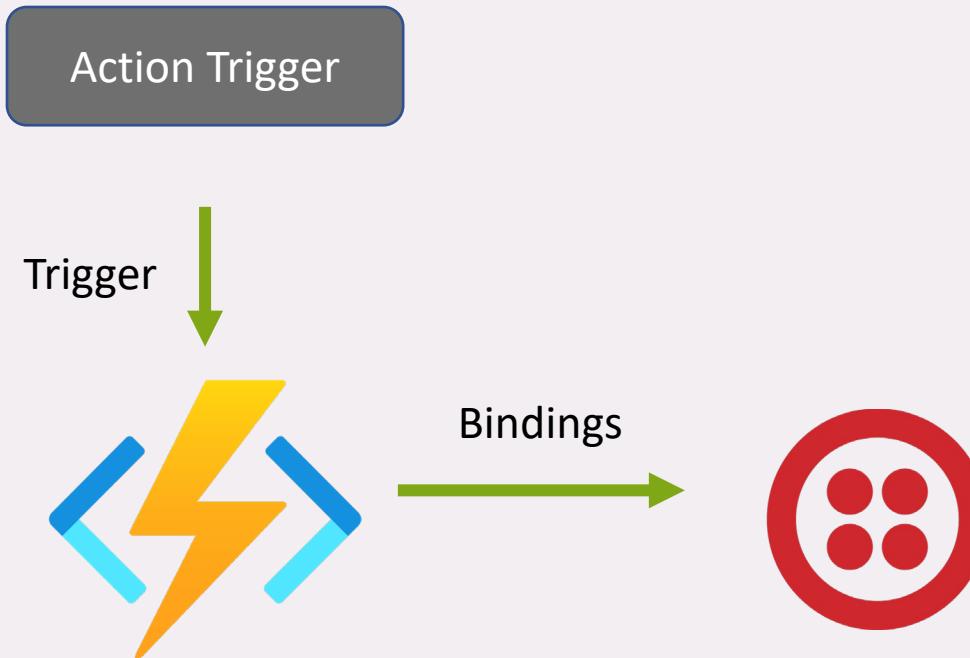
Timer

Trigger	✓
Input	✗
Output	✗
Consumption	✓
Premium	✓
Dedicated	✓
C# In-Process	✓
C# Isolated	✓
Java	✓
JavaScript	✓
PowerShell	✓
Python	✓



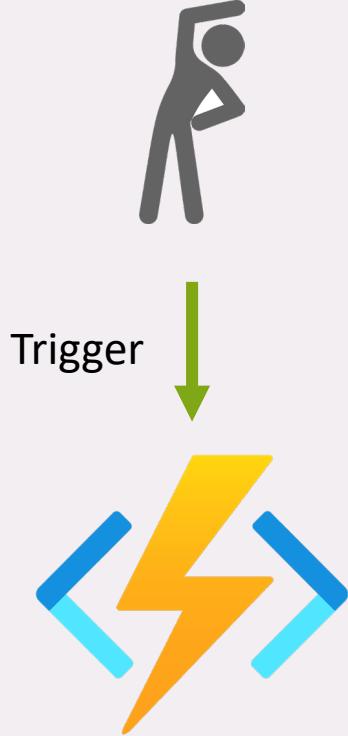
Twilio

Trigger	
Input	
Output	
Consumption	
Premium	
Dedicated	
C# In-Process	
C# Isolated	
Java	
JavaScript	
PowerShell	
Python	

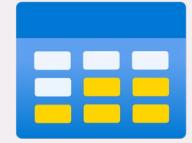
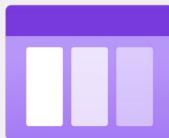
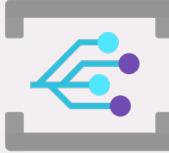


Warmup

Trigger	✓
Input	✗
Output	✗
Consumption	✗
Premium	✓
Dedicated	✓
C# In-Process	✓
C# Isolated	✓
Java	✓
JavaScript	✓
PowerShell	✓
Python	✓



Supported Bindings



Third Party Bindings

Custom Bindings

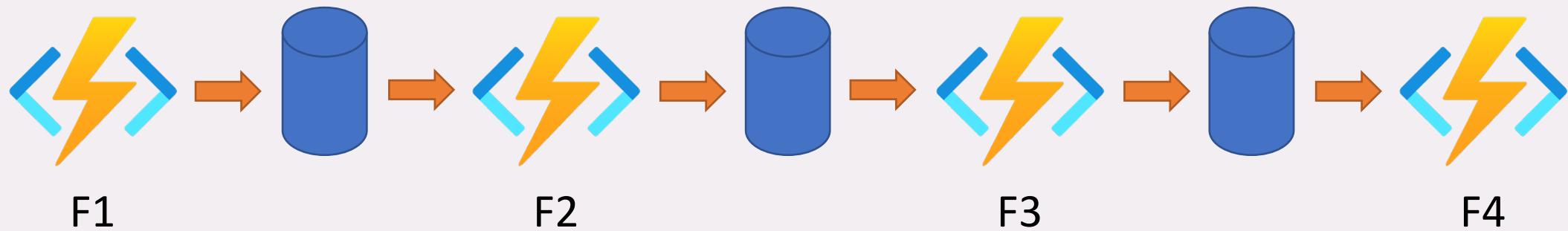
Durable Functions

Durable Functions

Stateful Federated Functions

Durable Functions

Patterns – Function Chaining



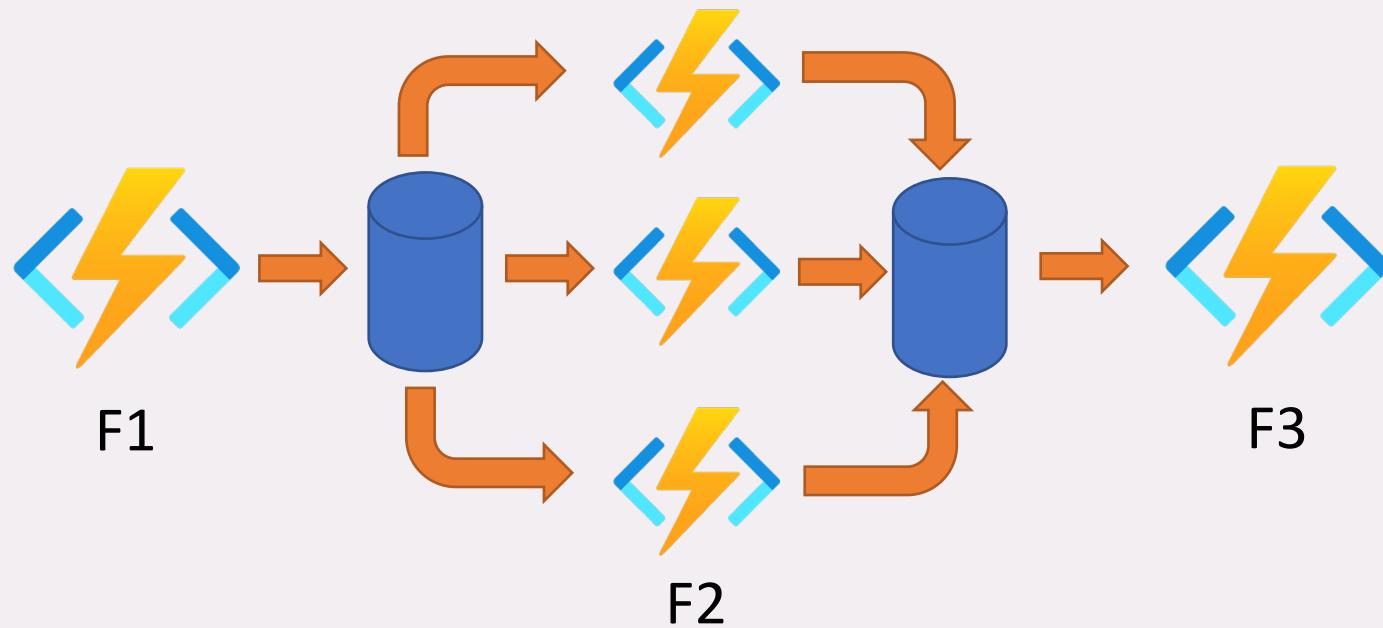
Durable Functions

Patterns – Function Chaining

```
[FunctionName("Chaining")]
public static async Task<object> Run(
    [OrchestrationTrigger] IDurableOrchestrationContext context)
{
    try
    {
        var x = await context.CallActivityAsync<object>("F1", null);
        var y = await context.CallActivityAsync<object>("F2", x);
        var z = await context.CallActivityAsync<object>("F3", y);
        return await context.CallActivityAsync<object>("F4", z);
    }
    catch (Exception)
    {
        // Error handling or compensation goes here.
    }
}
```

Durable Functions

Patterns – Fan out/fan in



Durable Functions

Patterns – Fan out/fan in

```
[FunctionName("FanOutFanIn")]
public static async Task Run(
    [OrchestrationTrigger] IDurableOrchestrationContext context)
{
    var parallelTasks = new List<Task<int>>();

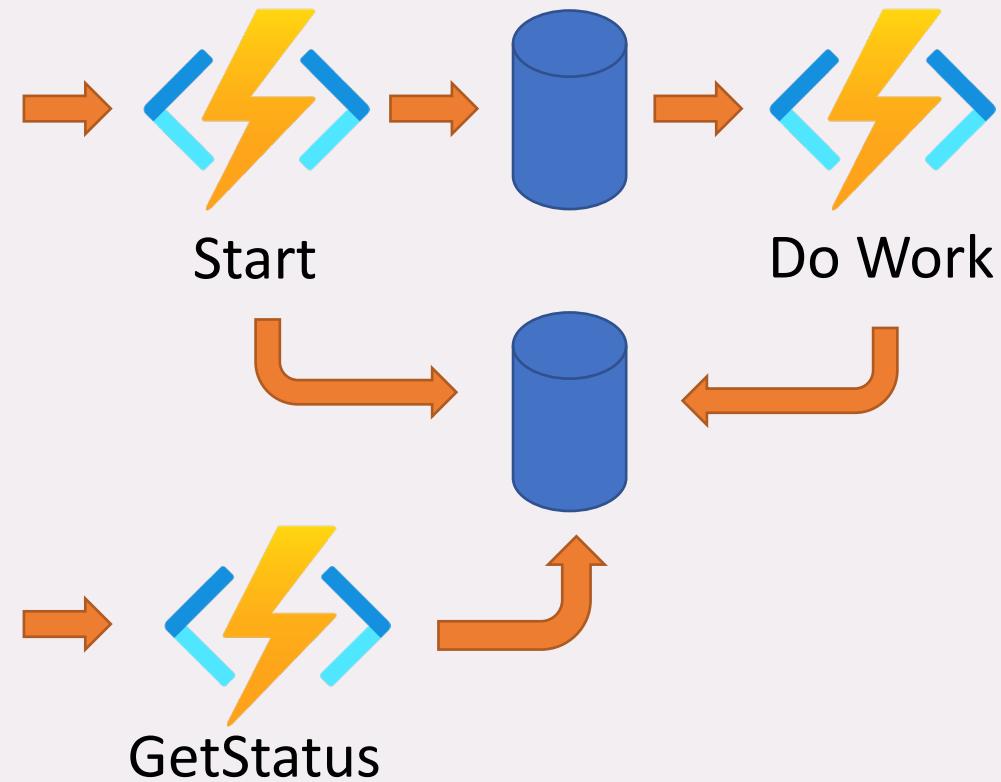
    // Get a list of N work items to process in parallel.
    object[] workBatch = await context.CallActivityAsync<object[]>("F1", null);
    for (int i = 0; i < workBatch.Length; i++)
    {
        Task<int> task = context.CallActivityAsync<int>("F2", workBatch[i]);
        parallelTasks.Add(task);
    }

    await Task.WhenAll(parallelTasks);

    // Aggregate all N outputs and send the result to F3.
    int sum = parallelTasks.Sum(t => t.Result);
    await context.CallActivityAsync("F3", sum);
}
```

Durable Functions

Patterns – Async HTTP APIs



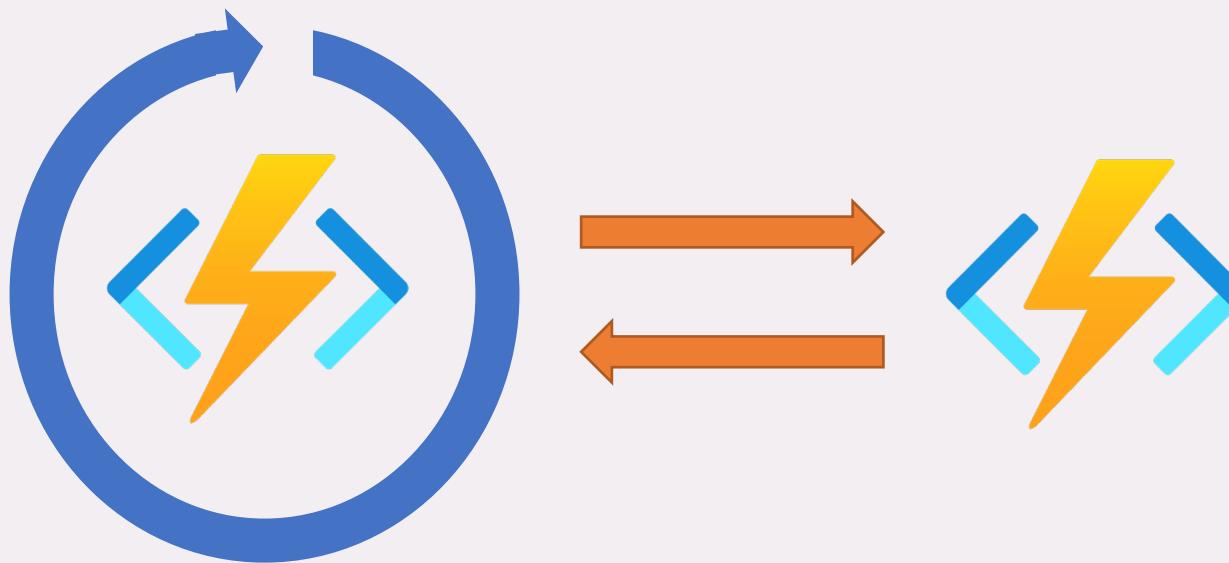
Durable Functions

Patterns – Async HTTP APIs

```
> curl -X POST https://myfunc.azurewebsites.net/api/orchestrators/DoWork -H "Content-Length: 0" -i  
HTTP/1.1 202 Accepted  
Content-Type: application/json  
Location: https://myfunc.azurewebsites.net/runtime/webhooks/durabletask/instances/b79baf67f717453ca9e86c5da21e03ec  
{"id":"b79baf67f717453ca9e86c5da21e03ec", ...}  
  
> curl https://myfunc.azurewebsites.net/runtime/webhooks/durabletask/instances/b79baf67f717453ca9e86c5da21e03ec  
HTTP/1.1 202 Accepted  
Content-Type: application/json  
Location: https://myfunc.azurewebsites.net/runtime/webhooks/durabletask/instances/b79baf67f717453ca9e86c5da21e03ec  
{"runtimeStatus":"Running","lastUpdatedTime":"2019-03-16T21:20:47Z", ...}  
  
> curl https://myfunc.azurewebsites.net/runtime/webhooks/durabletask/instances/b79baf67f717453ca9e86c5da21e03ec  
HTTP/1.1 200 OK  
Content-Length: 175  
Content-Type: application/json  
  
{"runtimeStatus":"Completed","lastUpdatedTime":"2019-03-16T21:20:57Z", ...}
```

Durable Functions

Patterns – Monitor



Durable Functions

Patterns – Monitor

```
[FunctionName("MonitorJobStatus")]
public static async Task Run(
    [OrchestrationTrigger] IDurableOrchestrationContext context)
{
    int jobId = context.GetInput<int>();
    int pollingInterval = GetPollingInterval();
    DateTime expiryTime = GetExpiryTime();

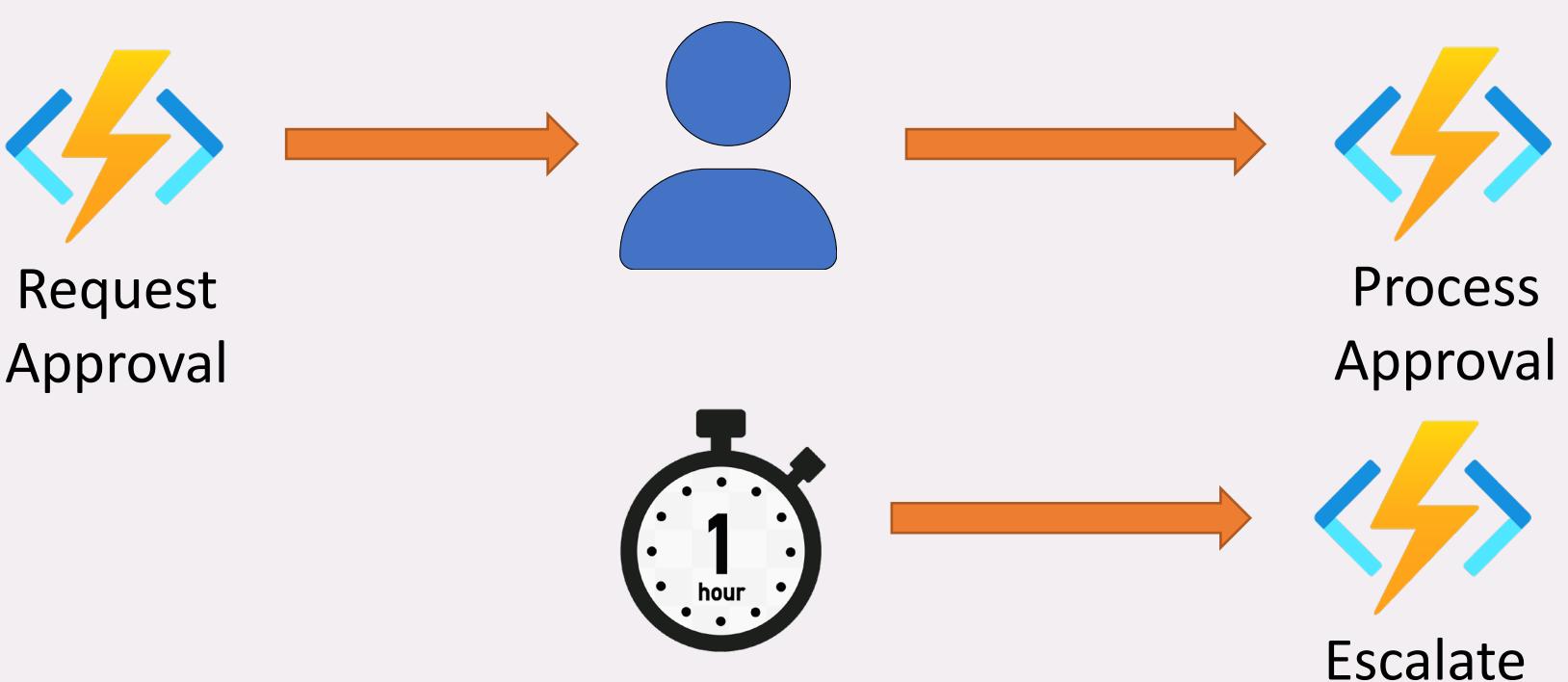
    while (context.CurrentUtcDateTime < expiryTime)
    {
        var jobStatus = await context.CallActivityAsync<string>("GetJobStatus", jobId);
        if (jobStatus == "Completed")
        {
            // Perform an action when a condition is met.
            await context.CallActivityAsync("SendAlert", machineId);
            break;
        }

        // Orchestration sleeps until this time.
        var nextCheck = context.CurrentUtcDateTime.AddSeconds(pollingInterval);
        await context.CreateTimer(nextCheck, CancellationToken.None);
    }

    // Perform more work here, or let the orchestration end.
}
```

Durable Functions

Patterns – Human Interaction



Durable Functions

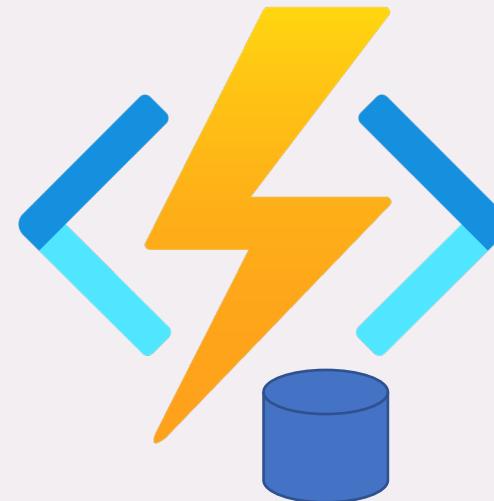
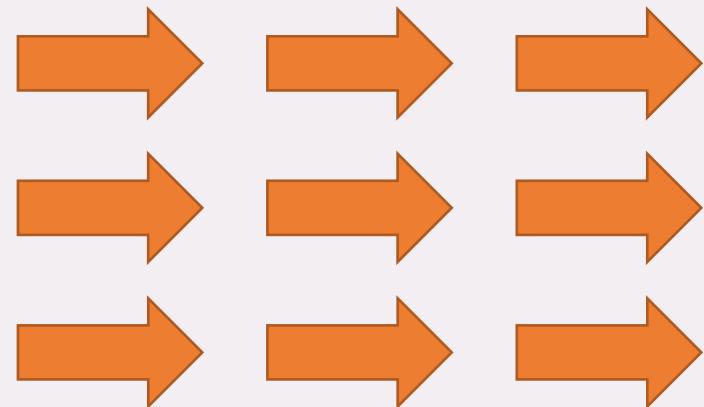
Patterns – Human Interaction

```
[FunctionName("ApprovalWorkflow")]
public static async Task Run(
    [OrchestrationTrigger] IDurableOrchestrationContext context)
{
    await context.CallActivityAsync("RequestApproval", null);
    using (var timeoutCts = new CancellationTokenSource())
    {
        DateTime dueTime = context.CurrentUtcDateTime.AddHours(72);
        Task durableTimeout = context.CreateTimer(dueTime, timeoutCts.Token);

        Task<bool> approvalEvent = context.WaitForExternalEvent<bool>("ApprovalEvent");
        if (approvalEvent == await Task.WhenAny(approvalEvent, durableTimeout))
        {
            timeoutCts.Cancel();
            await context.CallActivityAsync("ProcessApproval", approvalEvent.Result);
        }
        else
        {
            await context.CallActivityAsync("Escalate", null);
        }
    }
}
```

Durable Functions

Patterns – Aggregator (stateful entities)



Durable Functions

Patterns – Aggregator (stateful entities)

```
[FunctionName("Counter")]
public static void Counter([EntityTrigger] IDurableEntityContext ctx)
{
    int currentValue = ctx.GetState<int>();
    switch (ctx.OperationName.ToLowerInvariant())
    {
        case "add":
            int amount = ctx.GetInput<int>();
            ctx.SetState(currentValue + amount);
            break;
        case "reset":
            ctx.SetState(0);
            break;
        case "get":
            ctx.Return(currentValue);
            break;
    }
}
```

Durable Functions

Patterns – Aggregator (stateful entities)

```
public class Counter
{
    [JsonProperty("value")]
    public int CurrentValue { get; set; }

    public void Add(int amount) => this.CurrentValue += amount;

    public void Reset() => this.CurrentValue = 0;

    public int Get() => this.CurrentValue;

    [FunctionName(nameof(Counter))]
    public static Task Run([EntityTrigger] IDurableEntityContext ctx)
        => ctx.DispatchAsync<Counter>();
}
```

Durable Functions

Patterns – Aggregator (stateful entities)

```
[FunctionName("EventHubTriggerCSharp")]
public static async Task Run(
    [EventHubTrigger("device-sensor-events")] EventData eventData,
    [DurableClient] IDurableEntityClient entityClient)
{
    var metricType = (string)eventData.Properties["metric"];
    var delta = BitConverter.ToInt32(eventData.Body, eventData.Body.Offset);

    // The "Counter/{metricType}" entity is created on-demand.
    var entityId = new EntityId("Counter", metricType);
    await entityClient.SignalEntityAsync(entityId, "add", delta);
}
```

In-Process vs Isolated Process

Advanced Serverless Workshop - Azure

Benefits of Isolated Process

Fewer Conflicts

Full Control

Dependency
Injection

Startup and Configuration

```
var host = new HostBuilder()
    .ConfigureFunctionsWorkerDefaults(builder =>
{
    builder
        .AddApplicationInsights()
        .AddApplicationInsightsLogger();
})
.ConfigureServices(s =>
{
    s.AddSingleton<IHttpResponderService, DefaultHttpResponderService>();
})
.Build();
```

Middleware

```
var host = new HostBuilder()
    .ConfigureFunctionsWorkerDefaults(workerApplication =>
{
    // Register our custom middlewares with the worker

    workerApplication.UseMiddleware<ExceptionHandlingMiddleware>();

    workerApplication.UseMiddleware<MyCustomMiddleware>();

    workerApplication.UseWhen<StampHttpHeaderMiddleware>((context) =>
{
    // We want to use this middleware only for http trigger invocations.
    return context.FunctionDefinition.InputBindings.Values
        .First(a => a.Type.EndsWith("Trigger")).Type == "httpTrigger";
})
    .Build();
}
```

ReadyToRun

```
<PropertyGroup>
  <TargetFramework>net6.0</TargetFramework>
  <AzureFunctionsVersion>v4</AzureFunctionsVersion>
  <RuntimeIdentifier>win-x86</RuntimeIdentifier>
  <PublishReadyToRun>true</PublishReadyToRun>
</PropertyGroup>
```

Not Everything Supported

Azure Functions C# Azure Cloud

Functions worker [\(i\)](#)

.NET 6.0 (Long-term support)

Function [\(i\)](#)

- Http trigger
- Cosmos DB Trigger
- Durable Functions Orchestration
- Event Grid trigger
- Event Hub trigger
- Http trigger
- Http trigger with OpenAPI
- IoT Hub trigger
- Kafka output
- Kafka trigger
- Queue trigger
- RabbitMQ trigger
- SendGrid
- Service Bus Queue trigger
- Service Bus Topic trigger
- SignalR
- SQL input binding
- SQL output binding
- Timer trigger

Azure Functions C# Azure Cloud

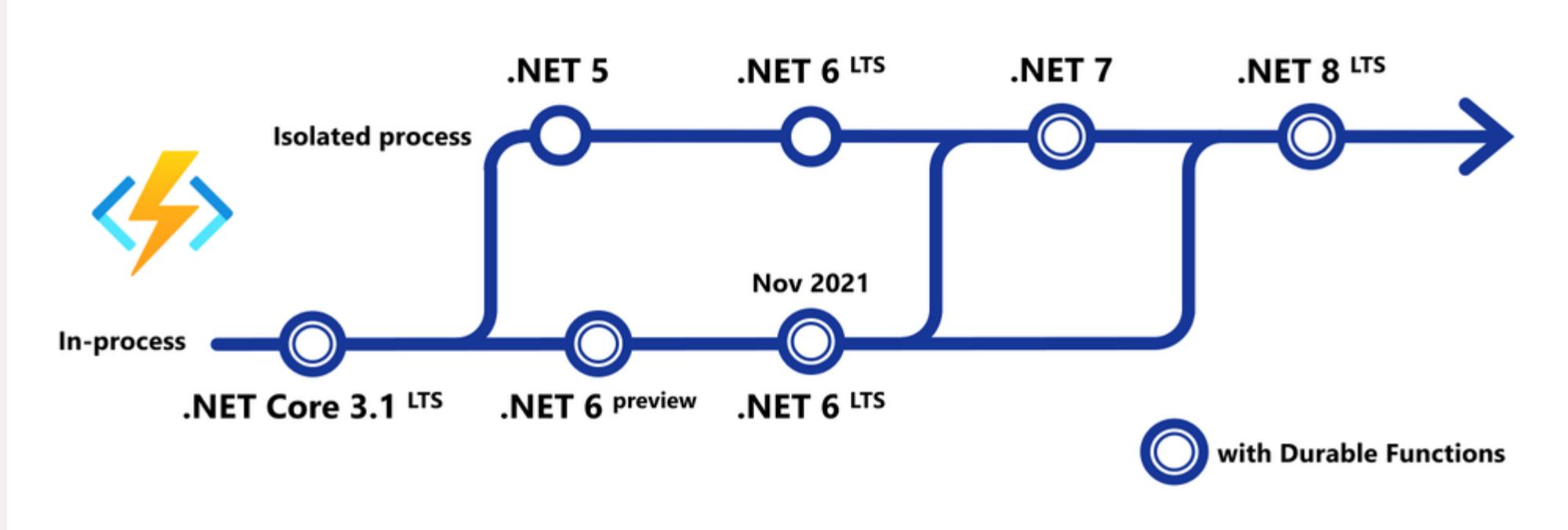
Functions worker [\(i\)](#)

.NET 6.0 Isolated (Long-term support)

Function [\(i\)](#)

- Http trigger
- Empty
- Blob trigger
- Cosmos DB Trigger
- Event Grid trigger
- Event Hub trigger
- Http trigger
- Queue trigger
- RabbitMQ trigger
- Service Bus Queue trigger
- Service Bus Topic trigger
- SignalR
- Timer trigger

Timelines



Best Practices

Advanced Serverless Workshop - Azure

Configure Storage Correctly

Create storage account
in same region

Configure Storage Correctly

Create storage account
in same region

Use a separate storage
account

Configure Storage Correctly

Create storage account
in same region

Use a separate storage
account

Do not enable Data
Lake Storage

Organize your functions

Performance and
Scaling

Organize your functions

Performance and
Scaling

Configurations and
Deployment

Organize your functions

Performance and
Scaling

Configurations and
Deployment

Privilege

Monitor effectively

AzureWebJobs
Dashboard

Monitor effectively

AzureWebJobs
Dashboard

Adjust Sampling
Settings

Build for scalability

Share and mange
connections

Build for scalability

Share and mange
connections

Avoid blocking calls

Build for scalability

Share and mange
connections

Avoid blocking calls

Receive messages in
batch

Build for scalability

Share and mange
connections

Avoid blocking calls

Receive messages in
batch

Write defensive
functions

Performance and reliability

Avoid long running
functions

Performance and reliability

Avoid long running
functions

Cross function
communication

Performance and reliability

Avoid long running
functions

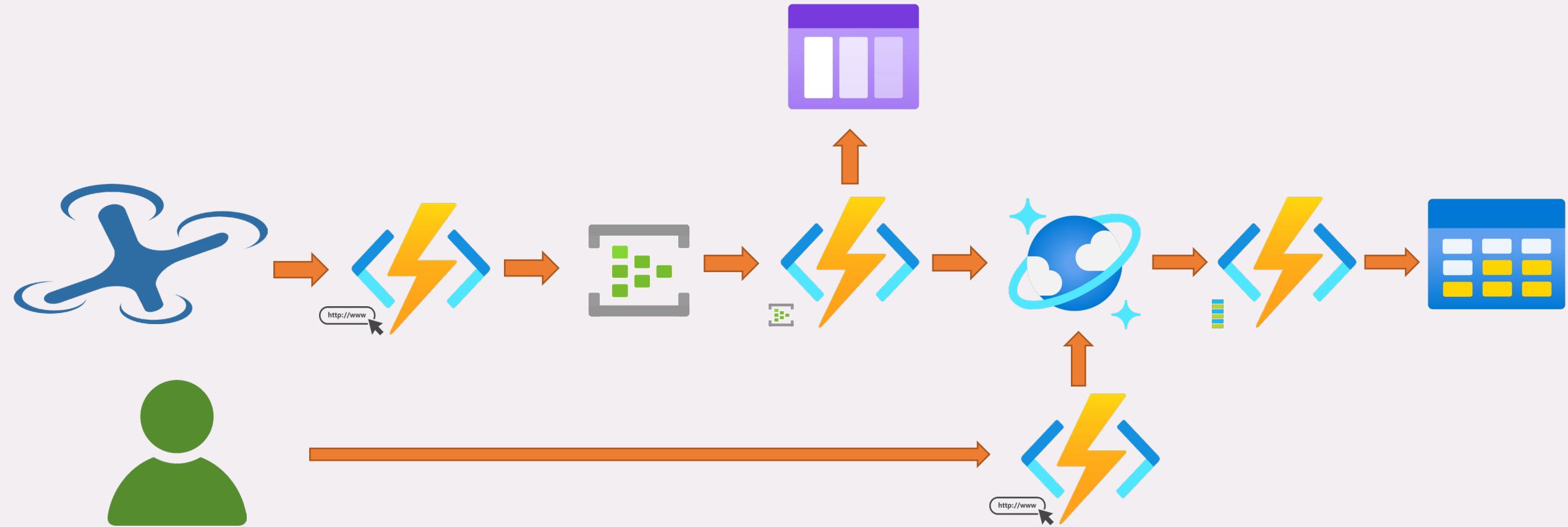
Cross function
communication

Write functions to be
stateless

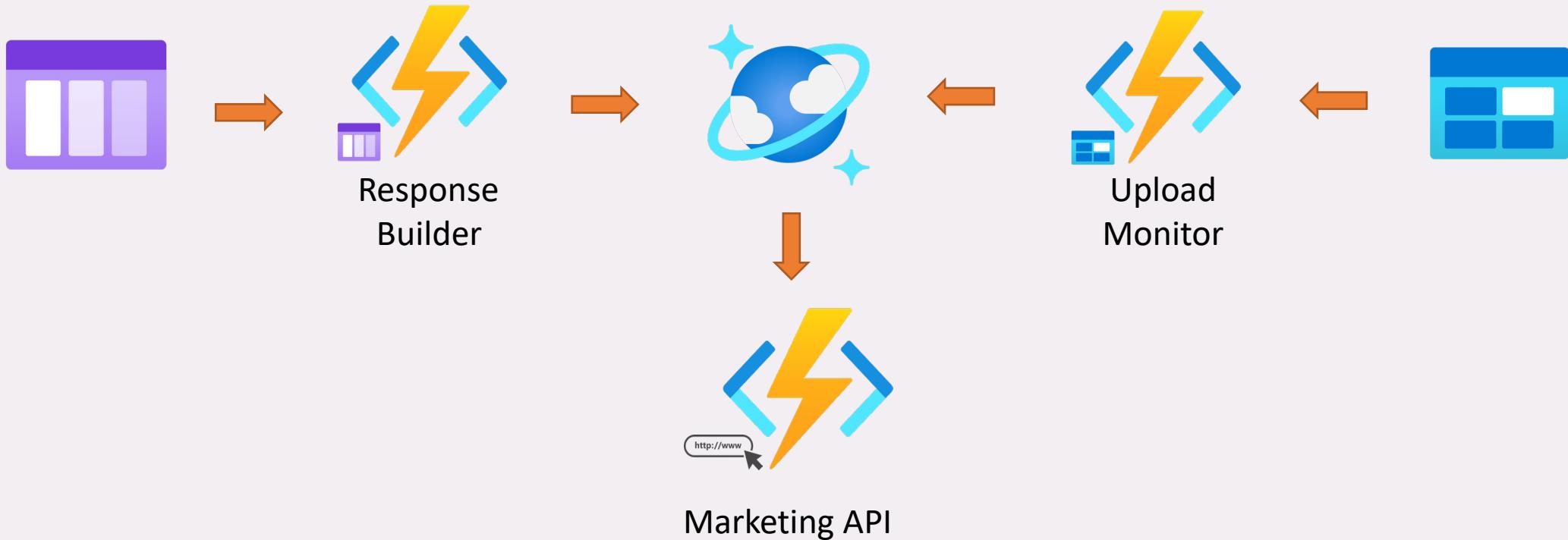
Demos

Advanced Serverless Workshop - Azure

Drone Telemetry



Portfolio Management



Thank You

- ✉ chadgreen@chadgreen.com
- .twitch TaleLearnCode
- 🌐 ChadGreen.com
- 🐦 ChadGreen & TaleLearnCode
- linkedin ChadwickEGreen

