



BUILDING MICROSERVICE

REST APIS

USING AZURE FUNCTIONS

Who is Chad Green

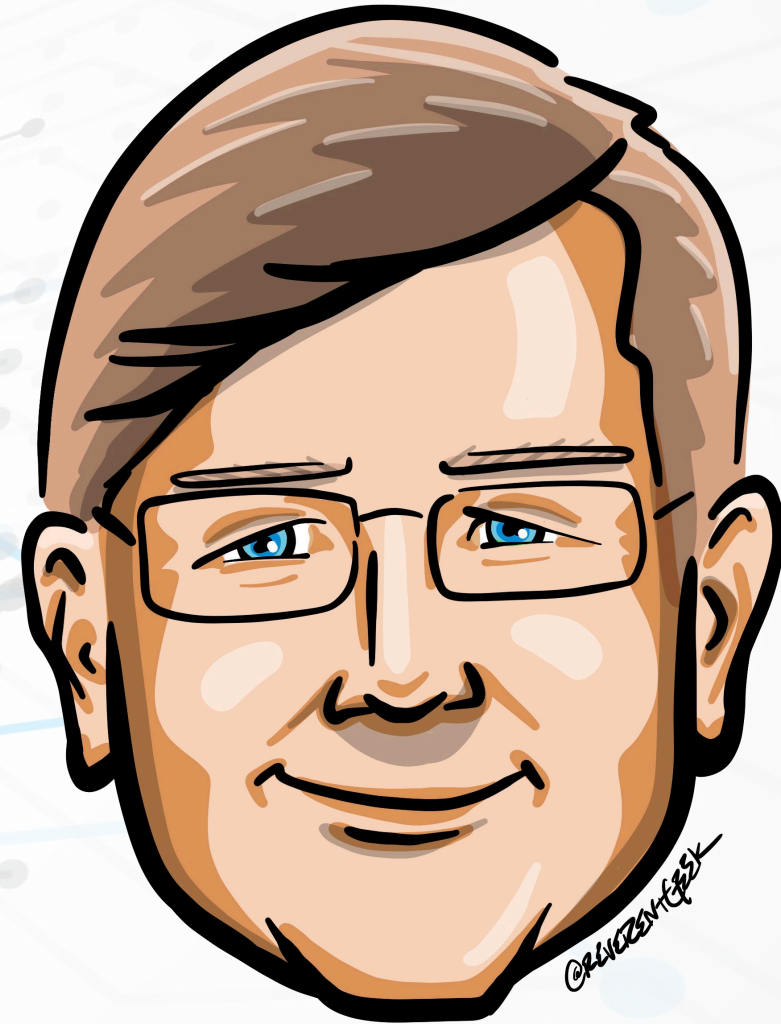
✉ chadgreen@chadgreen.com

💬 TaleLearnCode

🌐 ChadGreen.com

🐦 ChadGreen & TaleLearnCode

📘 ChadwickEGreen



Building Microservice REST APIs Using Azure Functions

Building Microservice REST APIs Using Azure Functions

Building Microservice REST APIs Using Azure Functions

Microservice

REST

APIs
Microservice

Serverless
REST

Azure Functions
APIs

Serverless



Microservice

REST

APIs

Serverless

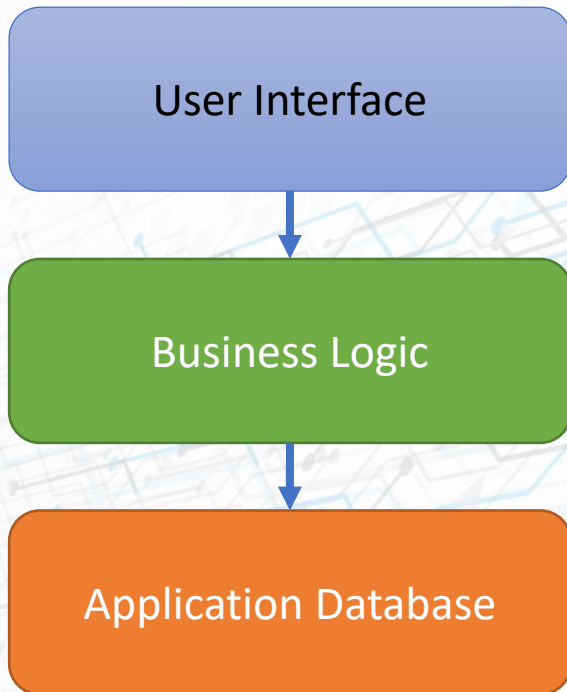
Azure Functions

What are microservices?

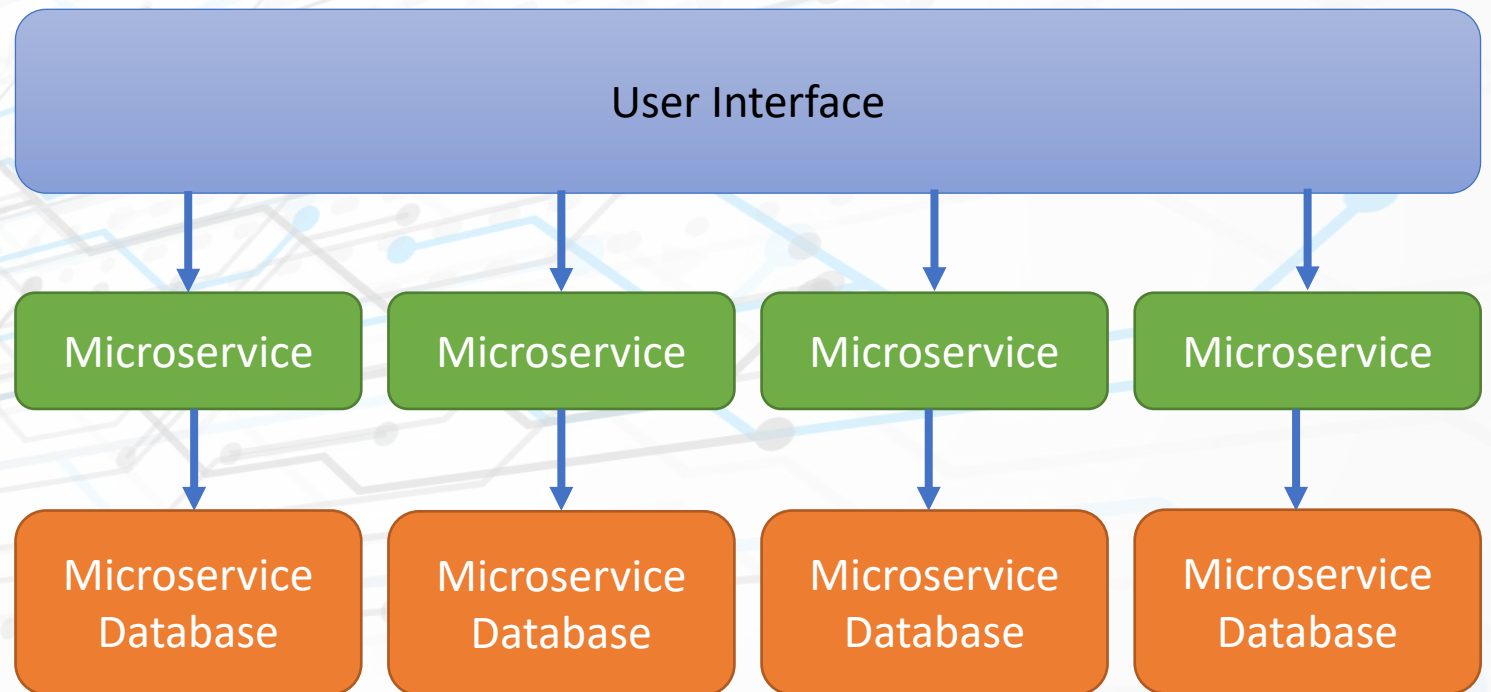
Building Microservice REST APIs Using Azure Functions

What are microservices?

Monolithic Architecture

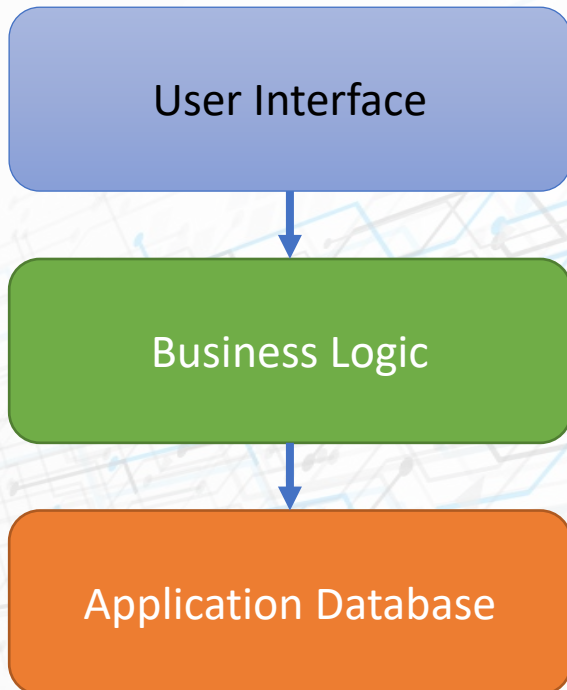


Microservices Architecture

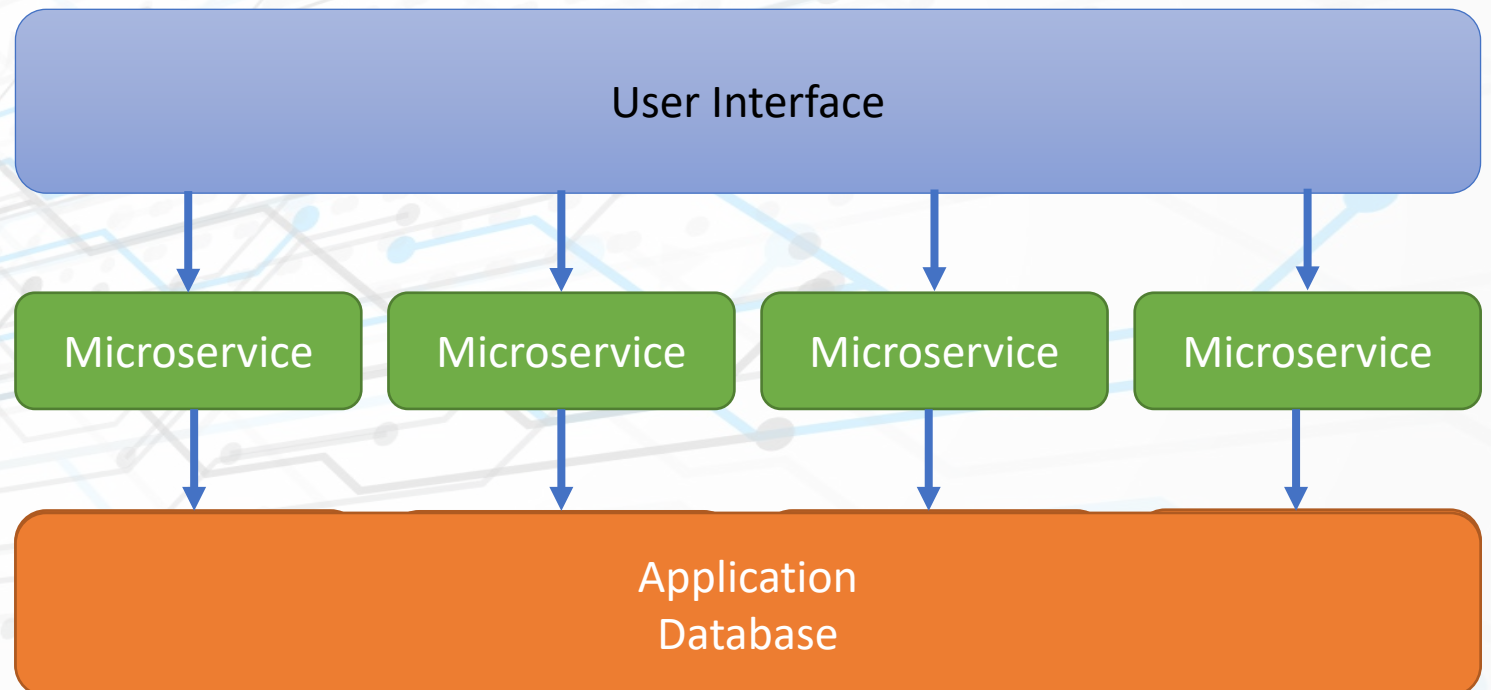


What are microservices?

Monolithic Architecture



Microservices Architecture



Advantages of Microservices

Quick

Advantages of Microservices

Quick

Quick

Advantages of Microservices

Quick

Resilience

Advantages of Microservices

Quick

Resilience

Advantages of Microservices

Quick

Resilience

Scalability

Advantages of Microservices

Quick

Resilience

Scalability

Advantages of Microservices

Quick

Resilience

Scalability

Maintainability

Advantages of Microservices

Quick

Resilience

Scalability

Flexibility

Advantages of Microservices

Quick

Resilience

Scalability

Flexibility

Maintainable

Advantages of Microservices

Quick

Resilience

Scalability

Flexibility

Maintainable

Advantages of Microservices

Quick

Resilience

Scalability

Maintainability

Flexibility

REST

Building Microservice REST APIs Using Azure Functions

Introduction to REST

REST

REpresentational **S**tate **T**ransfer

Introduction to REST



REST

Representational **S**tate **T**ransfer

Introduction to REST



REST

• **RE**presentational **S**tate **T**ransfer

Introduction to REST



Client



Server

REST

REpresentational State Transfer

Introduction to REST

Client

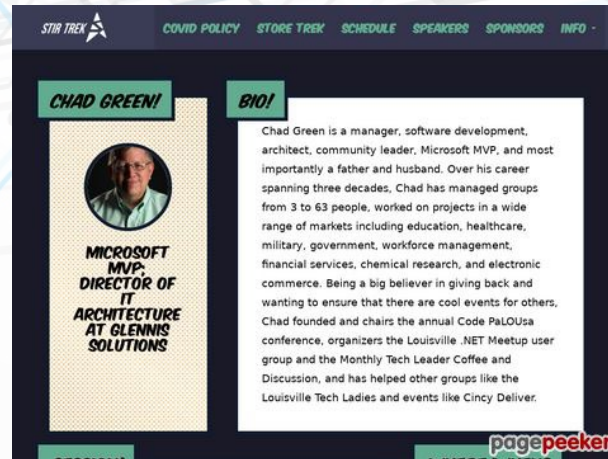
Send Request

stirtrek.com/speakers/2022/Chad-Green

Send Response

JSON, XML, HTML, SOAP, Image, etc.

Server



REST Architectural Constraints

**Client-Server
Architecture**

REST Architectural Constraints

**Client-Server
Architecture**

Statelessness

REST Architectural Constraints

Client-Server
Architecture

Statelessness

Cacheability

REST Architectural Constraints

**Client-Server
Architecture**

Statelessness

Cacheability

**Layered
System**

REST Architectural Constraints

Client-Server
Architecture

Statelessness

Cacheability

Layered
System

Code on
Demand
(optional)

REST Architectural Constraints

**Client-Server
Architecture**

Statelessness

Cacheability

**Layered
System**

**Code on
Demand**
(optional)

**Uniform
Interface**

REST Verbs

POST

Create

POST <https://exampleapi.com/employees>

| | HTTP Status Code | Description |
|---------|------------------|-------------|
| Success | 201 | Created |
| Failure | 400 | Bad Request |
| | 409 | Conflict |

REST Verbs

POST

Create

GET

Read

GET `https://exampleapi.com/employees/{id}`

| | HTTP Status Code | Description |
|---------|------------------|-------------|
| Success | 200 | OK |
| Failure | 400 | Bad Request |
| | 404 | Failure |

REST Verbs

POST

Create

GET

Read

PUT

Update

PUT `https://exampleapi.com/employees/{id}`

| | HTTP Status Code | Description |
|---------|------------------|-------------|
| Success | 204 | No Content |
| | 201 | Created |
| Failure | 400 | Bad Request |
| | 404 | Failure |

REST Verbs

POST

Create

GET

Read

PUT

Update

PATCH

~Update

PATCH <https://exampleapi.com/employees/{id}>

| | HTTP Status Code | Description |
|---------|------------------|-------------|
| Success | 204 | No Content |
| Failure | 400 | Bad Request |
| | 404 | Failure |

REST Verbs

POST

Create

GET

Read

PUT

Update

PATCH

~Update

DELETE

Delete

DELETE <https://exampleapi.com/employees/{id}>

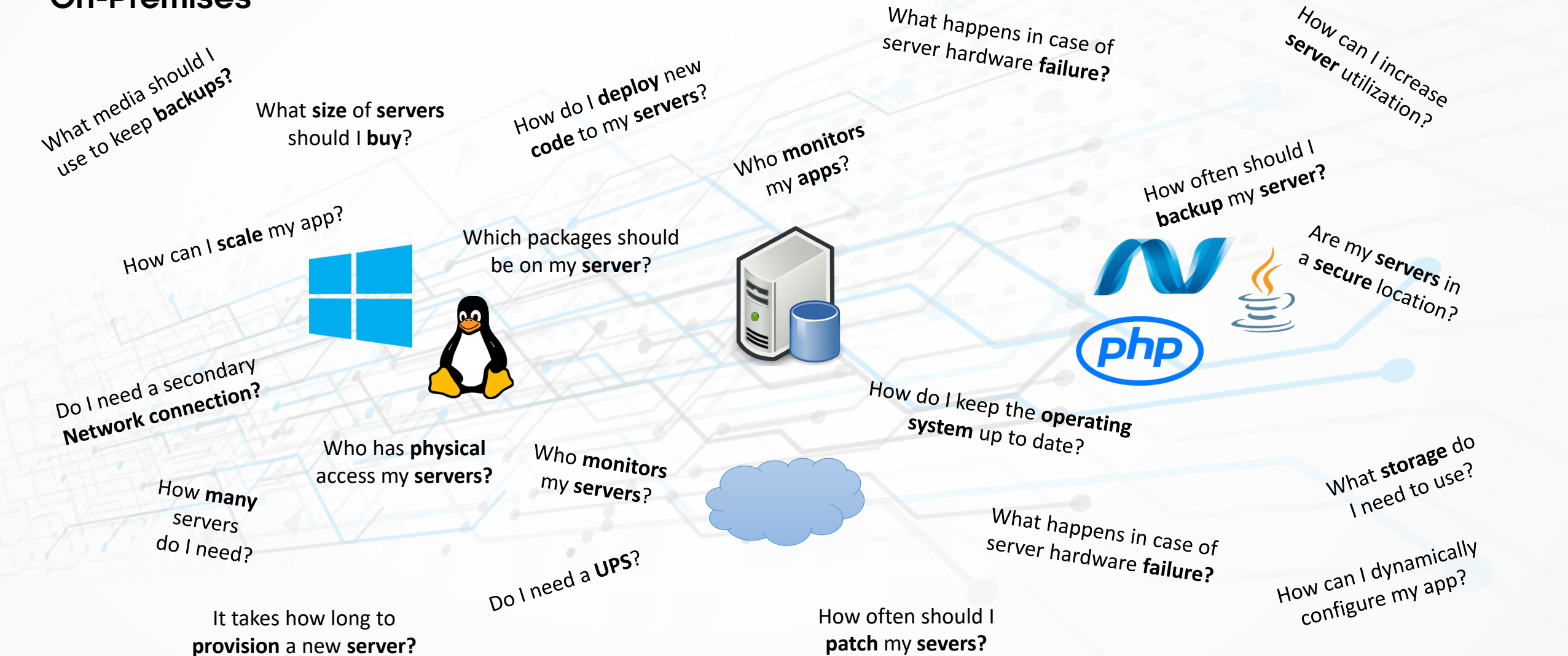
| | HTTP Status Code | Description |
|---------|------------------|-------------|
| Success | 200 | OK |
| Failure | 400 | Bad Request |
| | 404 | Failure |

What is Serverless?

Building Microservice REST APIs using Azure Functions

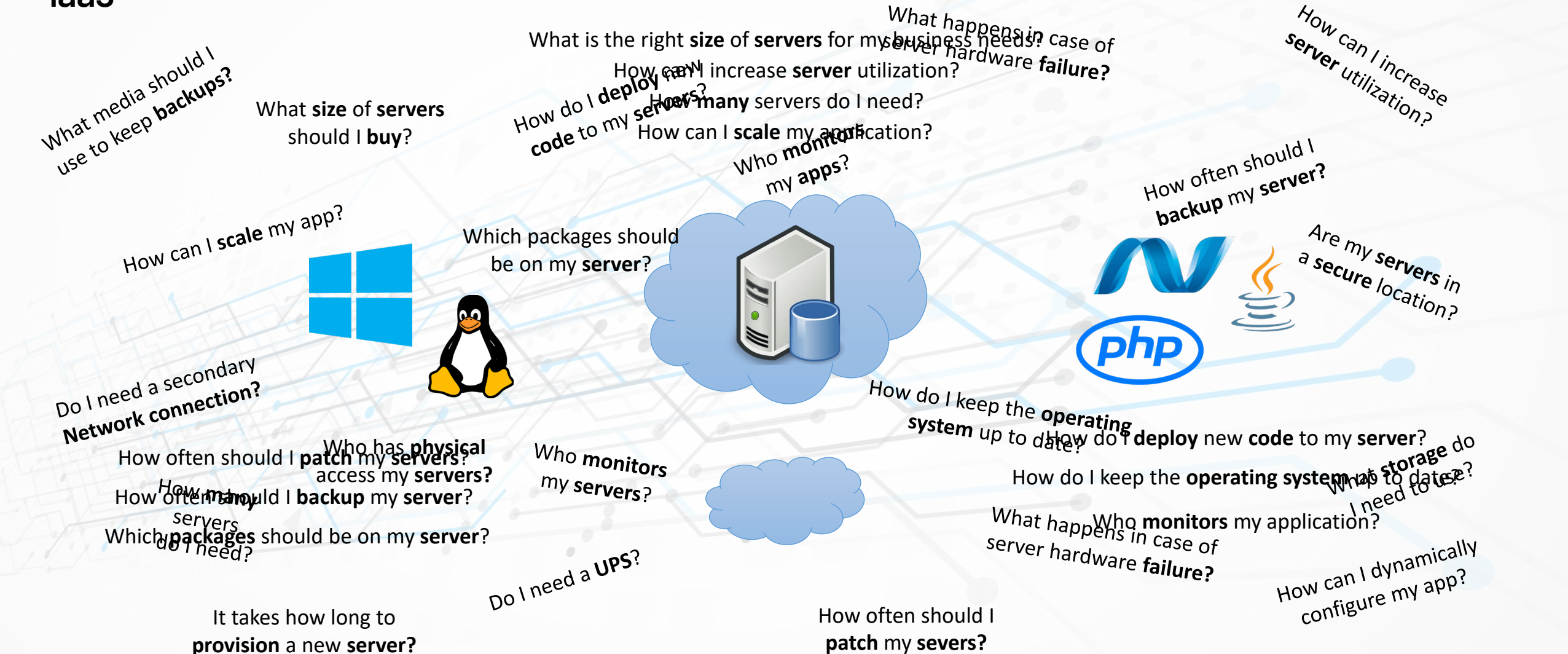
The evolution of application platforms

On-Premises



The evolution of application platforms

IaaS



The evolution of application platforms

PaaS

What is the right **size** of **servers** for my business needs?

How can I increase **server** utilization?

How **many** servers do I need?

How can I **scale** my application?



How do I **deploy** new **code** to my **server**?

How do I keep the **operating system** up to date?

Who **monitors** my application?

How often should I **patch** my **servers**?

How often should I **backup** my **server**?

Which **packages** should be on my **server**?

The evolution of application platforms

Serverless

What is the right **size** of **servers** for my business needs?

How can I increase **server** utilization?

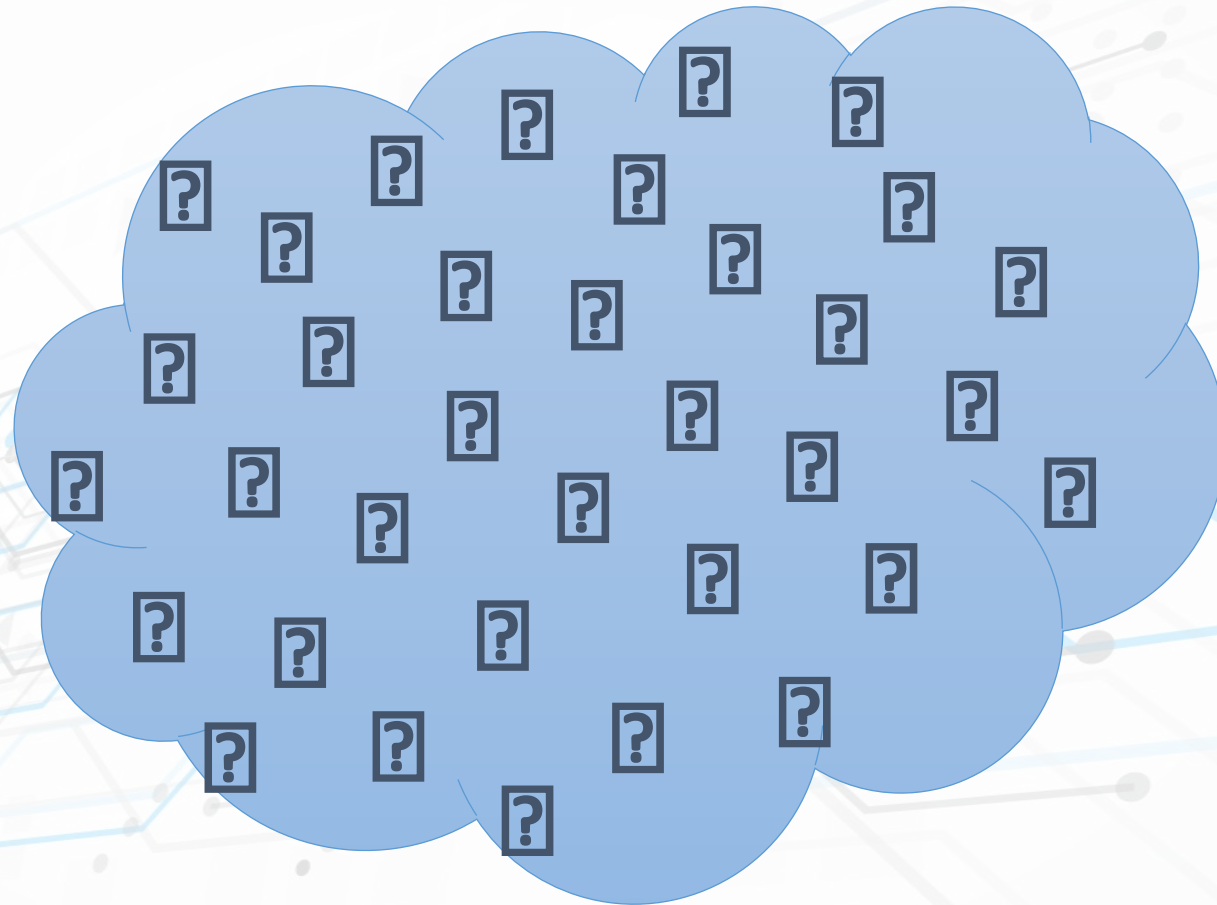
How **many** servers do I need?

How can I **scale** my application?



The evolution of application platforms

Serverless



The background of the slide features a complex, abstract circuit pattern. It consists of numerous thin, light blue and grey lines that meander and intersect across the frame, resembling a printed circuit board or a network diagram. Some lines terminate in small circular nodes, while others form loops or zig-zag patterns. The overall effect is a technical, digital aesthetic.

Not there isn't servers

Just, you can think about the servers less

~~**Server Configuration**~~

~~**Server Scaling**~~

Serverless Benefits

**No Server
Management**

Serverless Benefits

**No Server
Management**

Serverless Benefits

**No Server
Management**

**Simplified
Scalability**

Serverless Benefits

**No Server
Management**

**Simplified
Scalability**

Serverless Benefits

**No Server
Management**

**Simplified
Scalability**

Lower Costs

Serverless Benefits

**No Server
Management**

**Simplified
Scalability**

Lower Costs

Serverless Benefits

**No Server
Management**

**Simplified
Scalability**

Lower Costs

**Quicker
Turnaround**

Serverless Benefits

**No Server
Management**

**Simplified
Scalability**

Lower Costs

**Quicker
Turnaround**

Serverless Benefits

No Server
Management

Simplified
Scalability

Lower Costs

Quick
Turnaround

Simplified
Code

Serverless Benefits

No Server
Management

Simplified
Scalability

Lower Costs

Quick
Turnaround

Simplified
Code

Serverless Benefits

**No Server
Management**

**Simplified
Scalability**

Lower Costs

**Quicker
Turnaround**

**Simplified
Code**

Disadvantages of Serverless

**Testing
Challenges**

Disadvantages of Serverless

**Testing
Challenges**

Disadvantages of Serverless

**Testing
Challenges**

**Security
Concerns**

Disadvantages of Serverless

**Testing
Challenges**

**Security
Concerns**

Disadvantages of Serverless

**Testing
Challenges**

**Security
Concerns**

**Short –Running
Processes**

Disadvantages of Serverless

**No Server
Management**

**Simplified
Scalability**

**Short-Running
Processes**

Disadvantages of Serverless

**Testing
Challenges**

**Security
Concerns**

**Short-Running
Processes**

Cold Starts

Disadvantages of Serverless

**Testing
Challenges**

**Security
Concerns**

**Short-Running
Processes**

Cold Starts

Disadvantages of Serverless

Testing
Challenges

Security
Concerns

Short-Running
Processes

Cold Start

Vendor
Lock-In

Disadvantages of Serverless

Testing
Challenges

Security
Concerns

Short-Running
Processes

Cold Starts

Vendor
Lock-In

Disadvantages of Serverless

**Testing
Challenges**

**Security
Concerns**

**Short-Running
Processes**

Cold Starts

**Vendor
Lock-In**

Azure Functions

Building Microservice REST APIs Using Azure Functions

Features of Azure Functions

Choice of
Language

C#



Features of Azure Functions

Choice of
Language

Pay-Per-Use
Pricing

Consumption Plan

Features of Azure Functions

Choice of
Language

Pay-Per-Use
Pricing

Consumption Plan
Premium Plan

Features of Azure Functions

Choice of
Language

Pay-Per-Use
Pricing

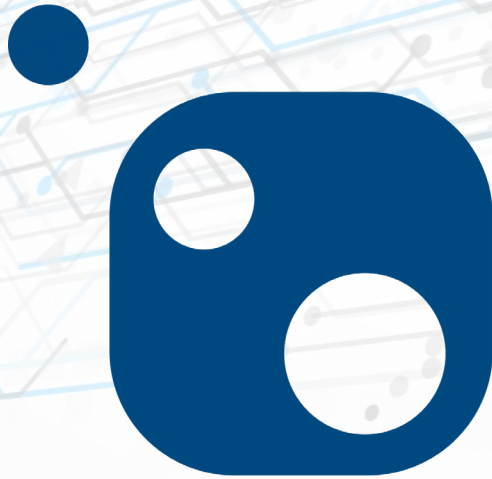
Consumption Plan
Premium Plan
Azure App Service Plan

Features of Azure Functions

Choice of
Language

Pay-Per-Use
Pricing

Bring Your Own
Dependencies



Features of Azure Functions

Choice of
Language

Pay-Per-Use
Pricing

Bring Your Own
Dependencies

Integrated Security

Features of Azure Functions

Choice of
Language

Pay-Per-Use
Pricing

Bring Your Own
Dependencies

Integrated Security

Simplified
Integration



Features of Azure Functions

Choice of
Language

Pay-Per-Use
Pricing

Bring Your Own
Dependencies

Integrated Security

Simplified
Integration

Flexible
Development

Features of Azure Functions

Choice of
Language

Pay-Per-Use
Pricing

Bring Your Own
Dependencies

Integrated Security

Simplified
Integration

Flexible
Development

Open Source

<https://github.com/Azure/Azure-Functions>

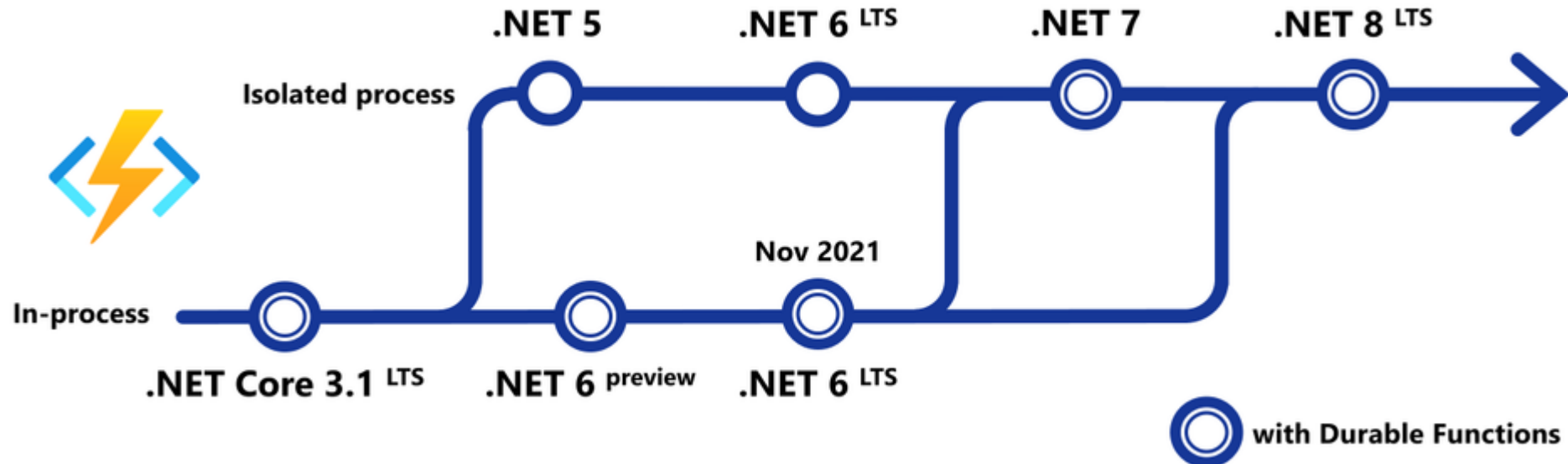
Triggers and Bindings

- Blob Storage
- Cosmos DB
- Event Grid
- Event Hubs
- External Table
- HTTP
- Microsoft Graph Excel tables
- Microsoft Graph OneDrive Files
- Microsoft Graph Outlook email
- Microsoft Graph Events
- Microsoft Graph Auth tokens
- Mobile Apps
- Notification Hubs
- Queue Storage
- SendGrid
- Sever Bus
- Table Storage
- Timer
- Twilio

C# Process Models

In-Process

Isolated Process

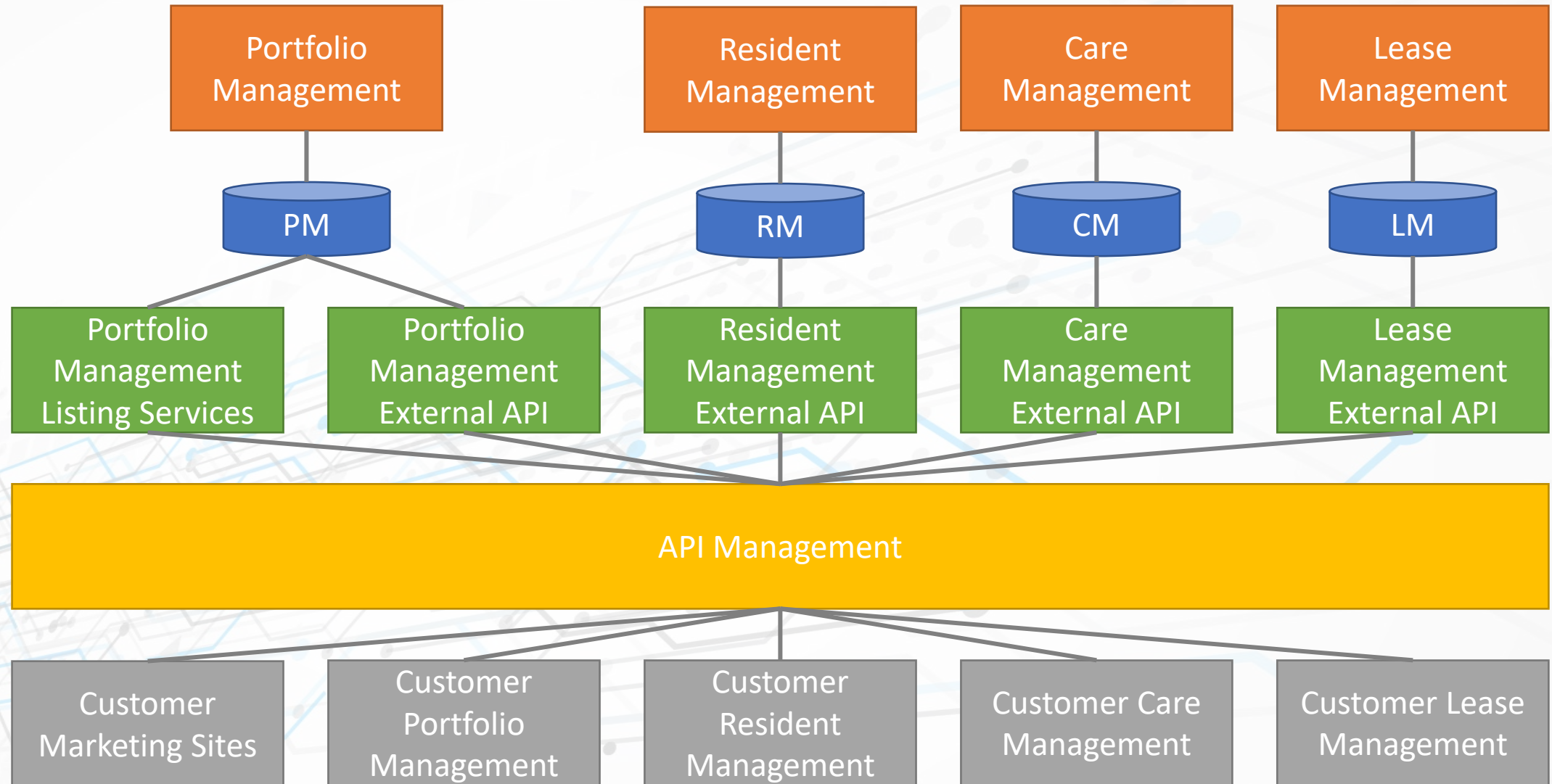


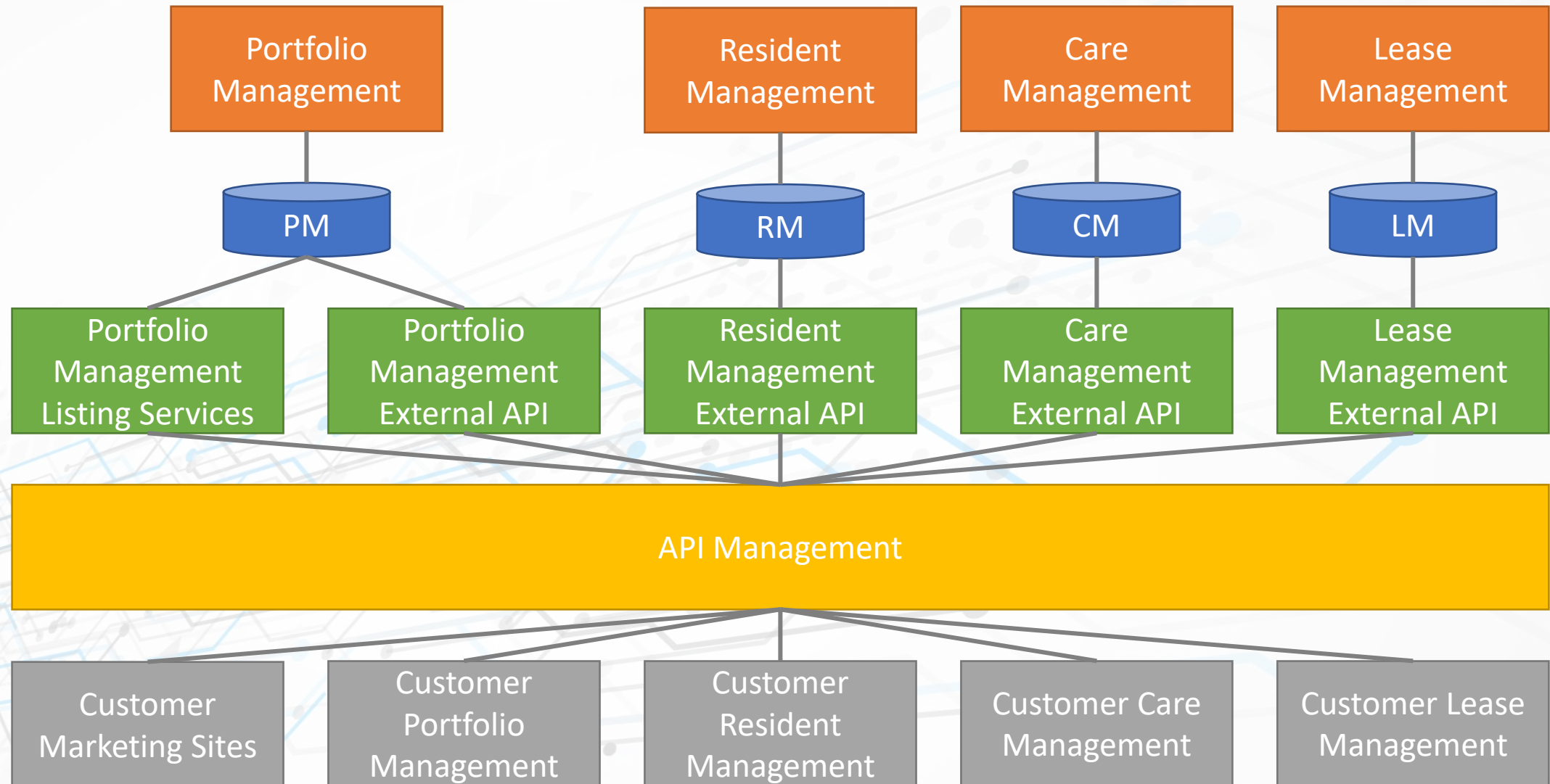
Code Walkthrough

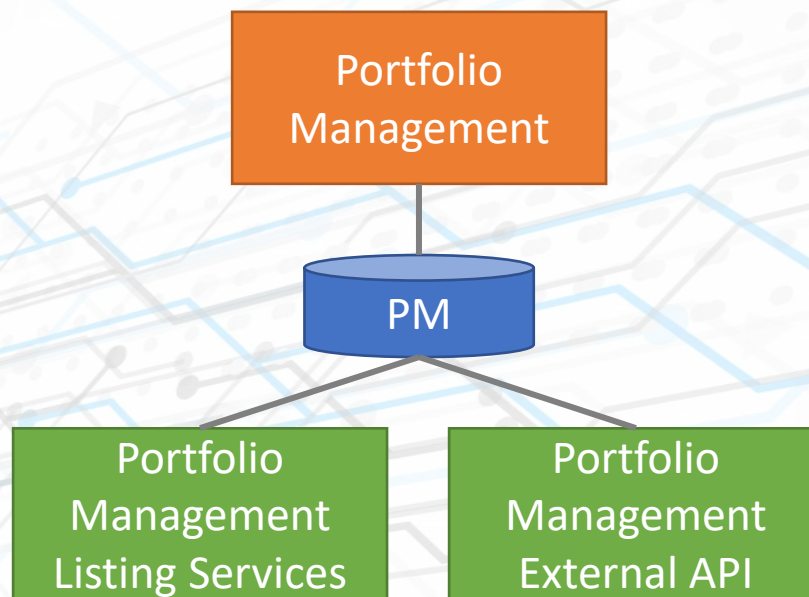
Building Microservice REST APIs Using Azure Functions











Serverless Microservice REST API Demo

CRUD Operations using HTTP Trigger



Serverless Microservice REST API Demo

Generating Open API Details for Endpoints



Serverless Microservice REST API Demo

Using the Azure SQL Bindings



Best Practices

Building Microservice REST APIs Using Azure Functions

Best Practices

Avoid long running functions

Best Practices

**Avoid cross functional
communication**

Best Practices

**Write functions to be
stateless**

Best Practices

Organize functions for performance and scaling

Best Practices

Share and manage connections

Best Practices

Avoid sharing storage accounts

Best Practices

**Use async code but avoid
blocking calls**

Thank You

✉ chadgreen@chadgreen.com

💬 TaleLearnCode

🌐 ChadGreen.com

🐦 ChadGreen & TaleLearnCode

🌐 ChadwickEGreen

