BUILDING MICROSERVICE
REST APIS
USING AZURE FUNCTIONS

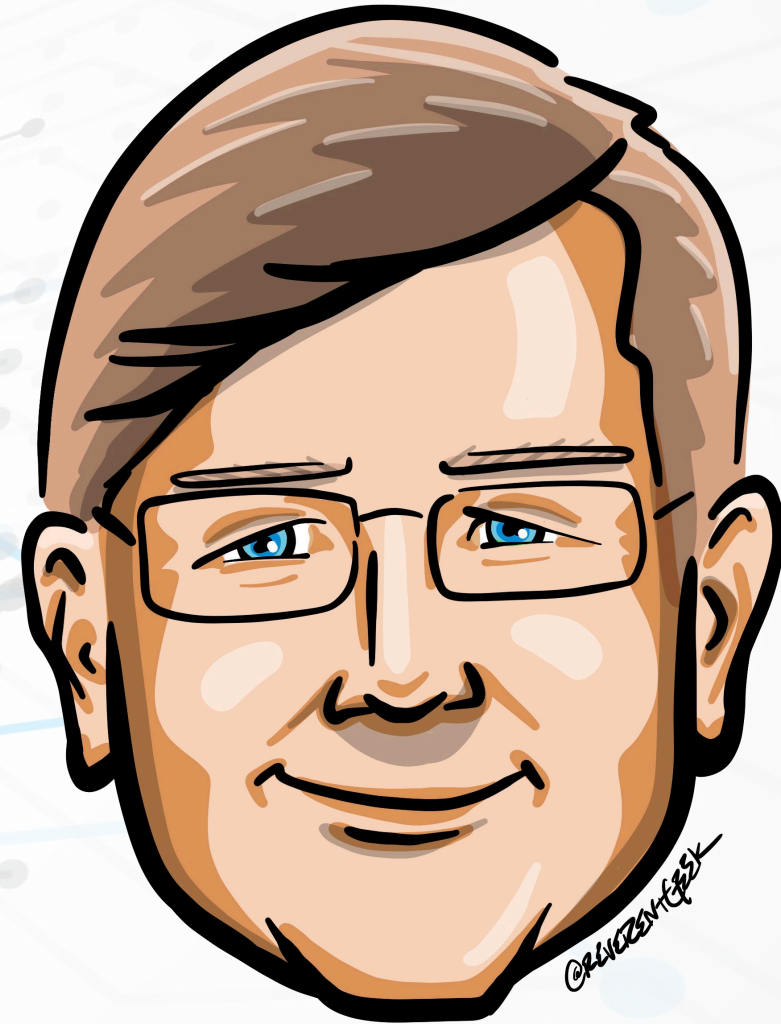# Who is Chad Green

✉ chadgreen@chadgreen.com

🖵 TaleLearnCode

🌐 ChadGreen.com

🐦 ChadGreen & TaleLearnCode

in ChadwickEGreen

MVP Microsoft® Most Valuable Professional

{Code PaLOUsa}

Building Microservice REST APIs Using Azure Functions

Building Microservice REST APIs Using Azure Functions

Building Microservice REST APIs Using Azure Serverless Functions
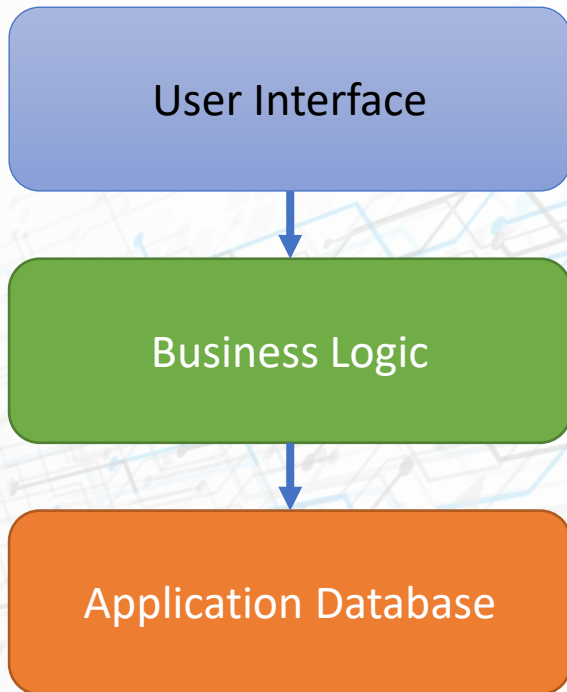
Microservice REST APIs Serverless Azure Functions

What are microservices?

Building Microservice REST APIs Using Azure Functions

What are microservices?

Monolithic Architecture

Microservices Architecture

What are microservices?

Monolithic Architecture

Microservices Architecture

User Interface

Business Logic

Application Database

User Interface

Microservice	Microservice	Microservice	Microservice

Application Database
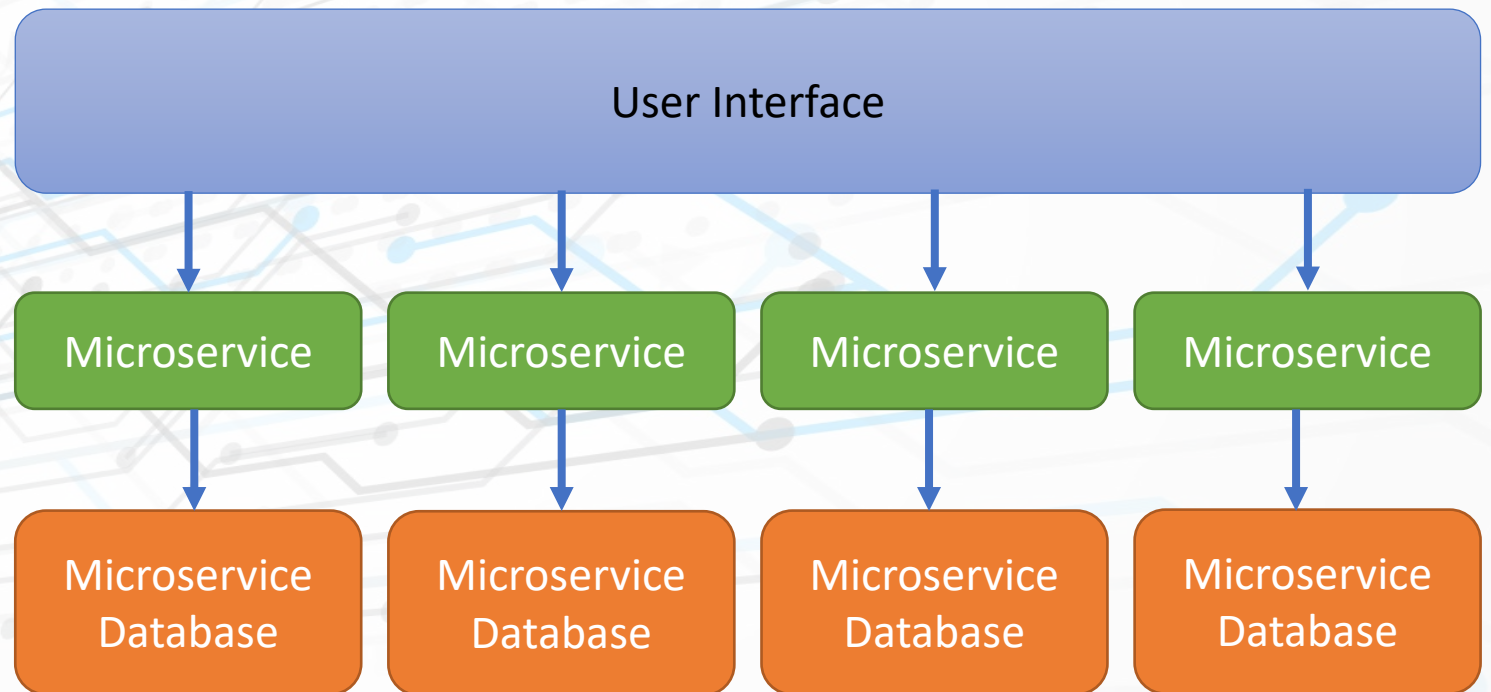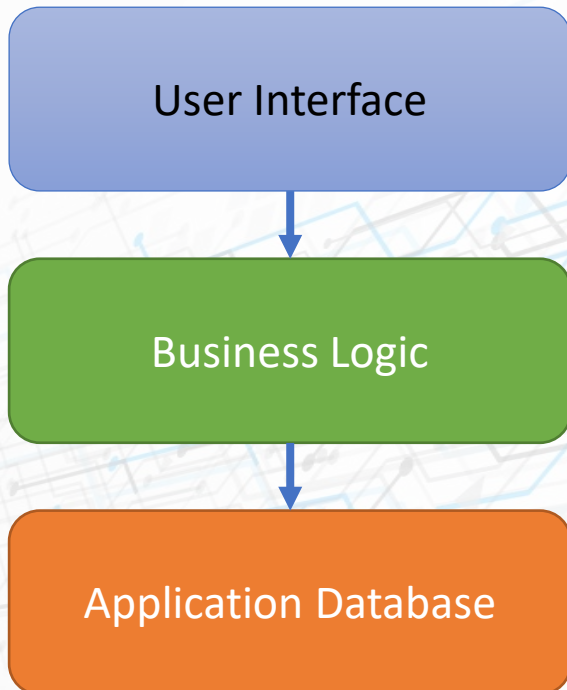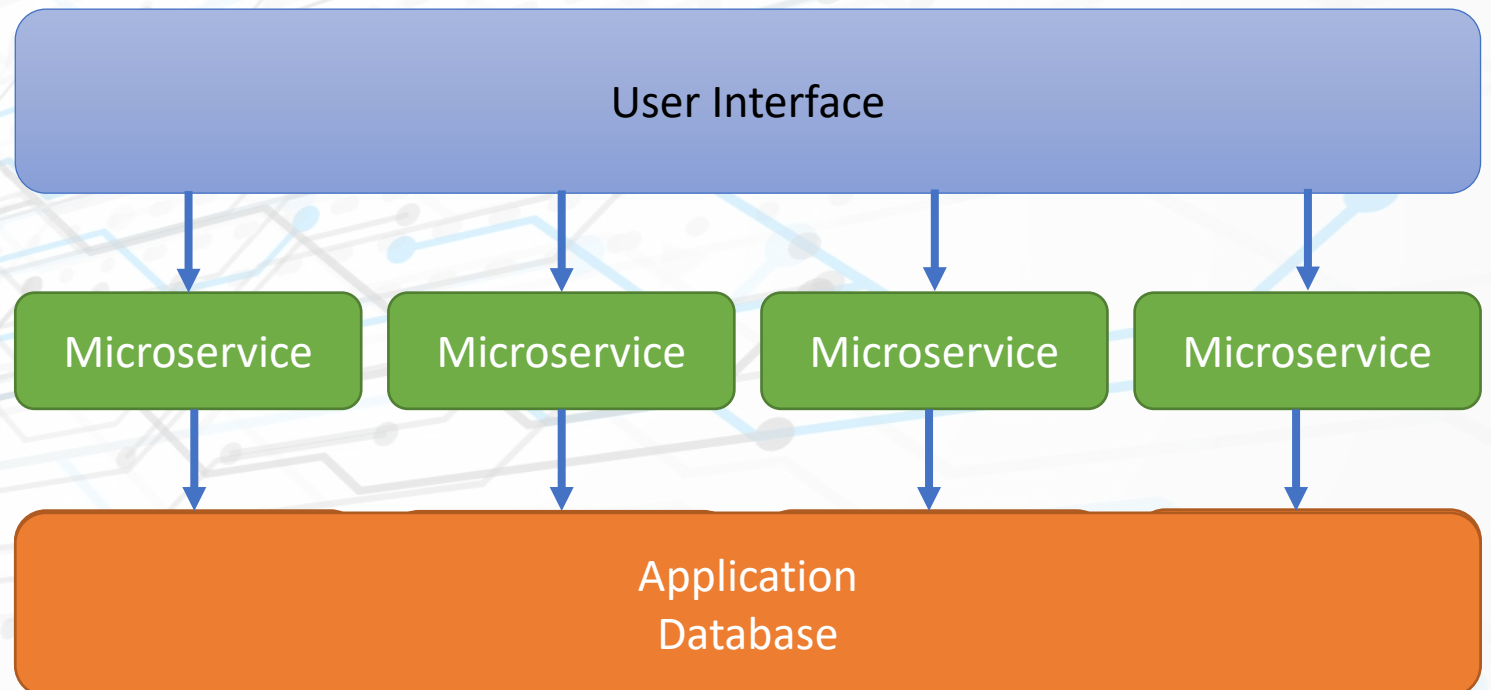
Advantages of Microservices

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

Maintainability

Flexibility

Advantages of Microservices

Quick

Resilience

Scalability

Maintainability

Flexibility

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

sentational **S**tate **T**ransfer

Introduction to REST

REST

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

Introduction to REST

Client

Server

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

Introduction to REST

Client

Send Request

stirtrek.com/speakers/2022/Chad-Green

Send Response

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

Server
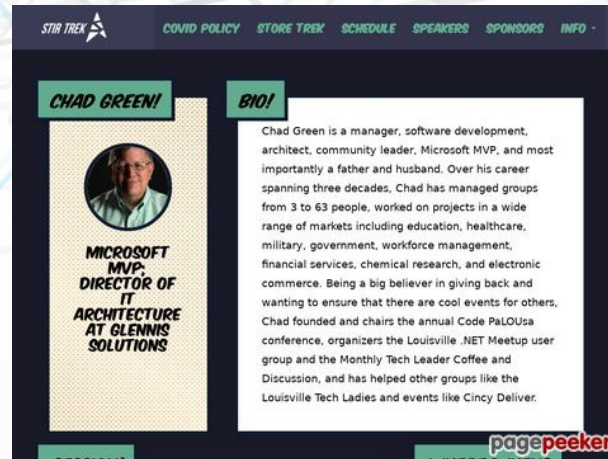
REST Architectural Constraints

Client-Server Architecture

REST Architectural Constraints

Client-Server Architecture

Statelessness

REST Architectural Constraints

Client-Server Architecture

Statelessness

Cacheability

Building Microservice REST APIs Using Azure Functions

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	GET	PUT
Create	Read	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	GET	PUT	PATCH
Create	Read	Update	~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

Why Serverless?

Building Microservice REST APIs using Azure Functions

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 Severless

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 St

Vendor Lock-In

Disadvantages of Serverless

Testing Challenges

Security Concerns

Short-Running Processes

Cold St...

Vendor Lock-In

Disadvantages of Serverless

Testing Challenges

Security Concerns

Short-Running Processes

Cold Starts

Vendor Lock-In

Function-as-a-Service (FaaS)

Building Microservice REST APIs Using Azure Functions

Function-as-a-Service (FaaS)

Small Bits of Code

Responds to Triggers

FaaS Market Landscape

AWS Lambda

Azure Functions

Google Cloud Functions

FaaS Market Landscape

AWS Lambda

Azure Functions

Google Cloud Functions

Azure Functions

Building Microservice REST APIs Using Azure Functions

Features of Azure Functions

Choice of Language

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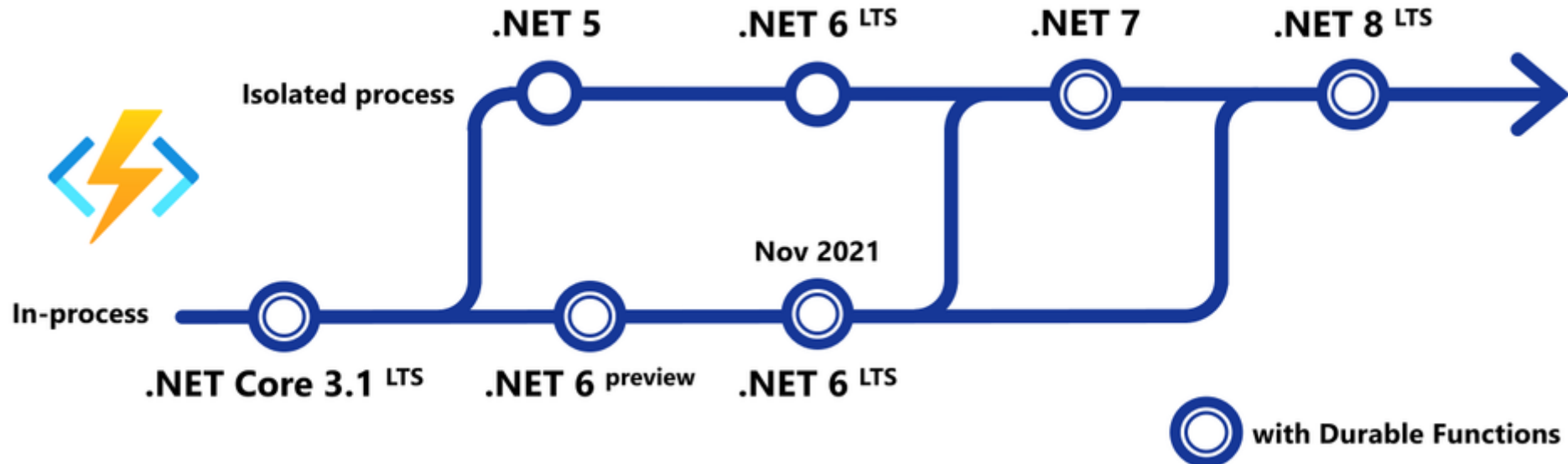
https://github.com/Azure/Azure-Functions

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
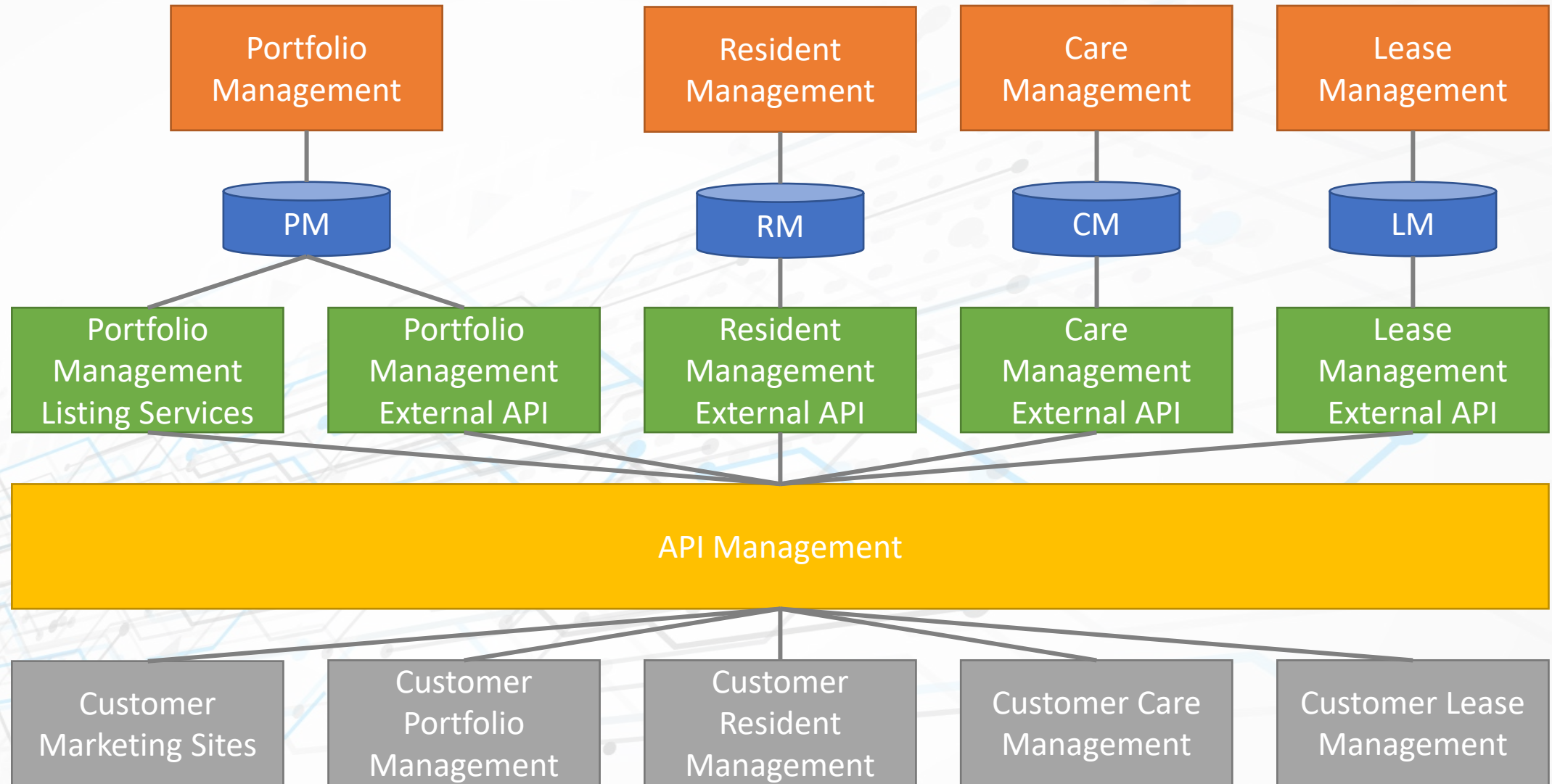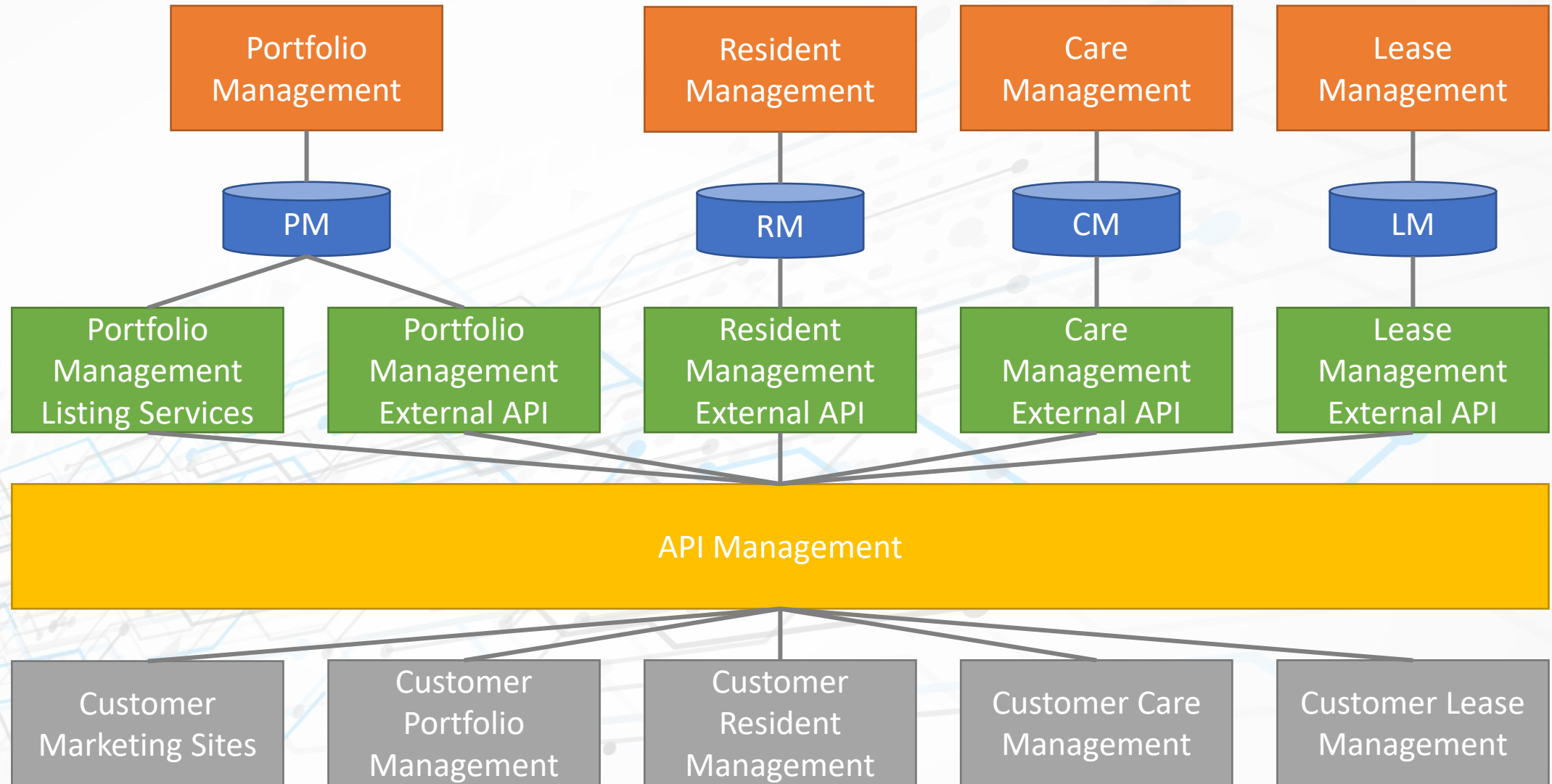- Twilio

C# Process Models

In-Process

Isolated Process

Isolated process

.NET 5 — **.NET 6** LTS — **.NET 7** — **.NET 8** LTS

Nov 2021

In-process

.NET Core 3.1 LTS — **.NET 6** preview — **.NET 6** LTS

⊚ with Durable Functions

Code Walkthrough

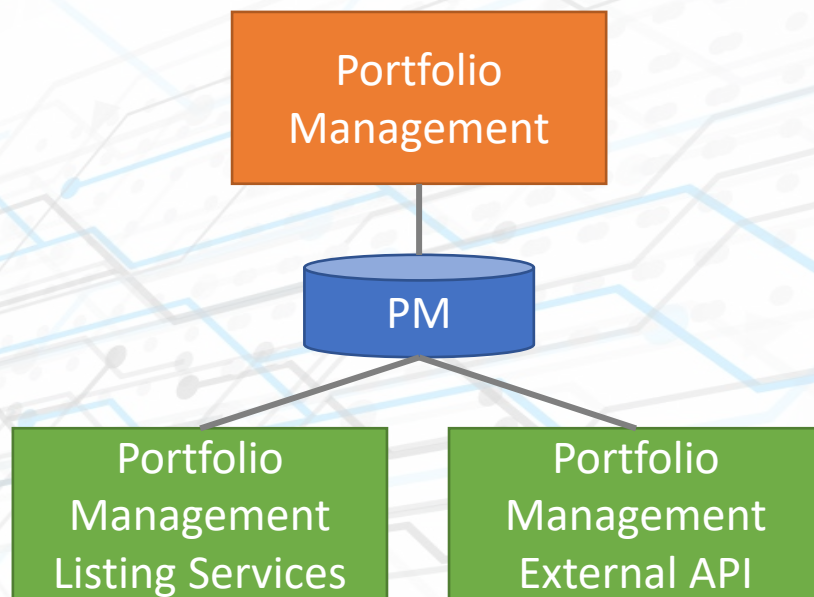Building Microservice REST APIs Using Azure Functions

Building Microservice REST APIs Using Azure Functions

Serverless Microservice REST API Demo

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

in ChadwickEGreen