Simplify the development of distributed applications with .NET Aspire

Thang Chung

November 23, 2024

DEVCafe
CONNECT AND SHARE

Nash Tech.

Thang Chung

Technical Manager at NashTech VN
Microsoft Azure MVP

Creator of Vietnam Microservices Group on Facebook (>20k members): https://www.facebook.com/groups/645391349250568

Experience: >18 years in software consult, design, development, and deployment software for outsourcing, product, and startup companies.

Expertise in cloud computing, cloud-native platform, serverless, and WebAssembly/WASI.

Blog: https://dev.to/thangchung

GitHub: https://github.com/thangchung

LinkedIn: https://www.linkedin.com/in/thangchungatwork

X (former Twitter): @thangchung

Agenda

01

.NET Aspire

02

What’s new in .NET Aspire 9?

03

Intelligent Apps with .NET Aspire

DEVCafe
CONNECT AND SHARE

01
.NET Aspire

9

Build anything with a unified platform

.NET

Cloud | Web | Desktop | Mobile | Gaming | IoT | AI

Tools


Visual Studio


Visual Studio Code


CLI


GitHub Copilot

+

Operating systems


Windows


Linux

macOS

+

Ecosystem


NuGet


GitHub


.NET Aspire

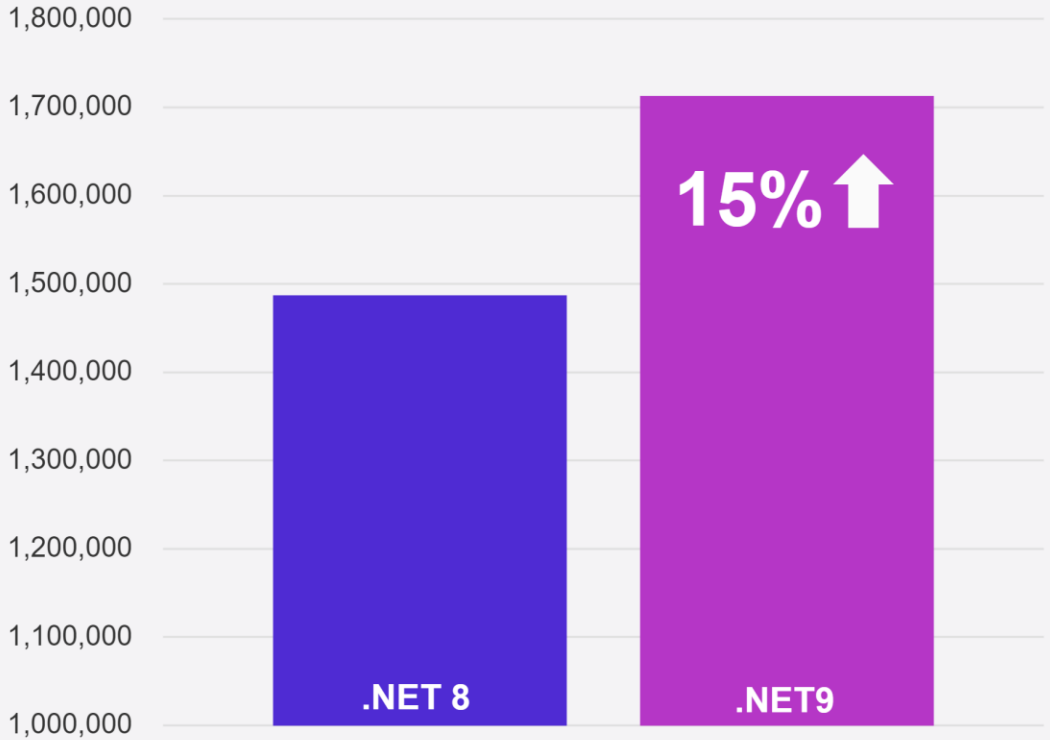

Components, tools, library vendors

.NET 9 Minimal API Performance

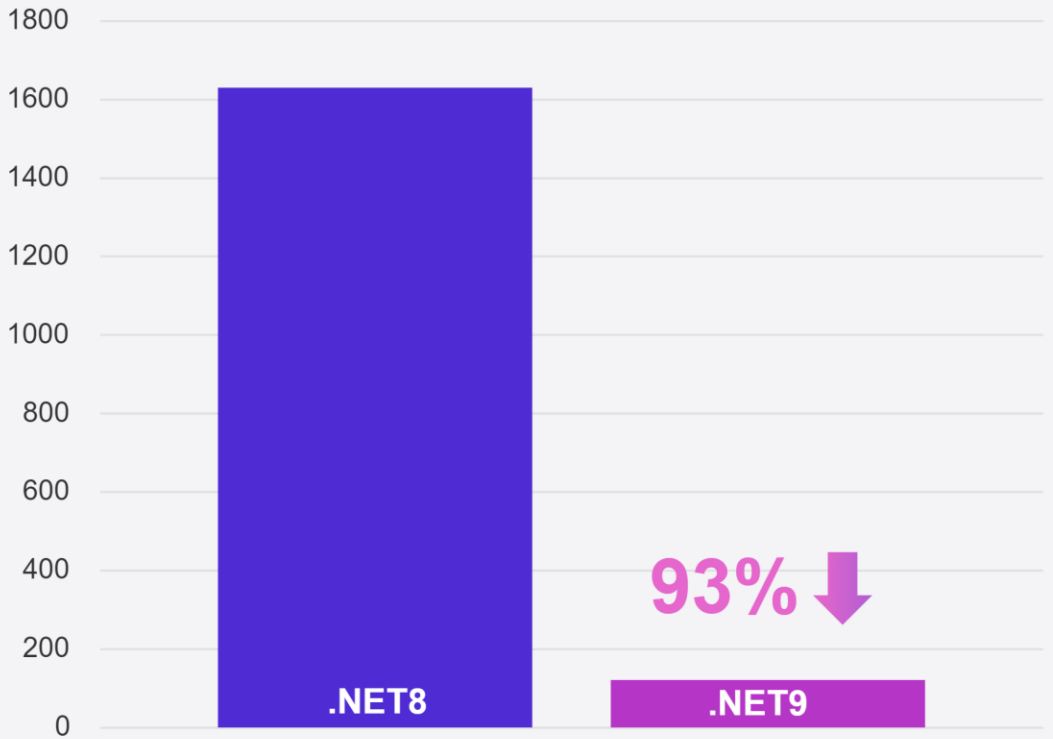
JSON Benchmarks

TechEmpower

Intel Gold 56 cores (logical) Linux
Source: aka.ms/aspnet/benchmarks



Requests per Second (higher is better)



Memory (MB; lower is better)

Every App Needs

Observability

Resiliency

Scalability

Manageability

.NET gives you a lot in the box

Observability

- **Built in metrics with dimensions**
- **DI integration for metrics**
- **Better Logging support** (faster, can object serialization)
- **Enrichment**
- **Redaction**
- **Testing fakes for Logging & Metrics**

Resiliency

- **New Polly based resiliency packages**
- **SignalR Stateful Reconnect**

Scalability

- **AOT** (increased density)
- **Performance**
- **Chiseled Ubuntu**

Manageability

- **Certificate auto-rotation support in Kestrel**

It’s Still Not Easy 😟

Complex

Getting Started

Choices

Paved Path

.NET Aspire

A cloud ready stack for building observable, production ready, distributed applications

.NET Aspire

A cloud ready stack for building observable, production ready, distributed applications

- Smart Defaults
- Developer Dashboard
- Orchestration
- Service Discovery
- Integrations
- Deployment

.NET Aspire

A cloud ready stack for building observable, production ready, distributed applications

- Smart Defaults
- Developer Dashboard
- Orchestration
- Service Discovery
- Integrations
- Deployment

.NET Aspire Service Defaults

Observability

Resiliency

Health Checks

DEMO

.NET Aspire - Smart Defaults

Clone: https://github.com/thangchung/practical-dotnet-aspire

.NET Aspire

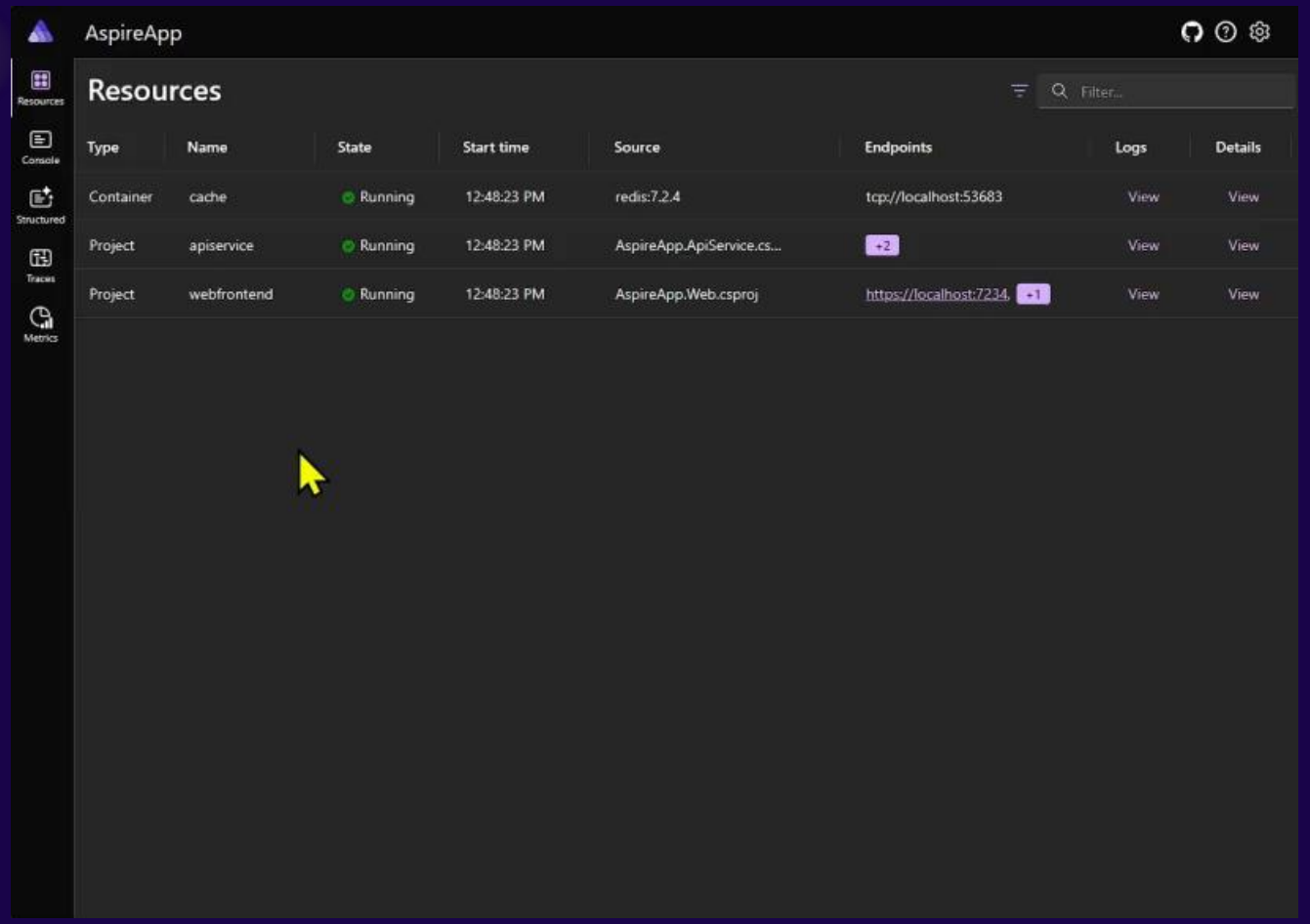
A cloud ready stack for building observable, production ready, distributed applications

- Smart Defaults
- Developer Dashboard
- Orchestration
- Service Discovery
- Integrations
- Deployment

.NET Aspire

Developer Dashboard

- Structured Logs
- Metrics
- Distributed Traces
- Dependencies

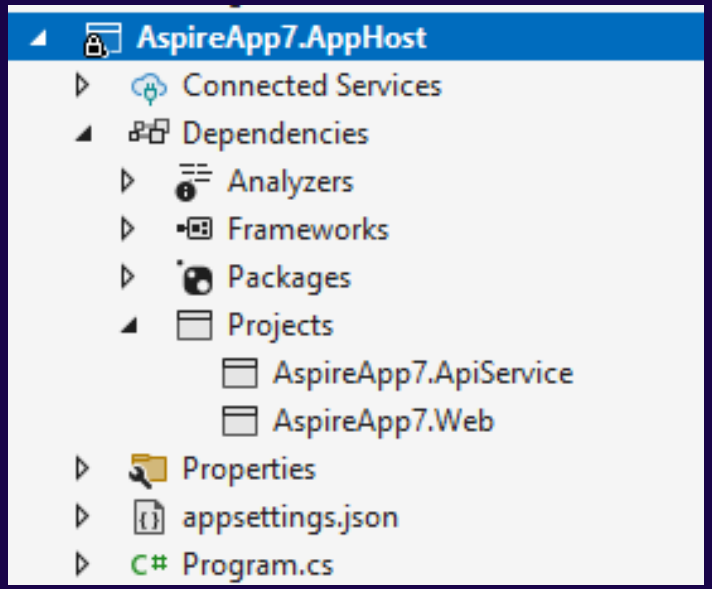


Integrating the Developer Dashboard

Standalone

```
docker run --rm -it `
    -p 18888:18888 `
    -p 4317:18889 -d `
    --name aspire-dashboard `
    mcr.microsoft.com/dotnet/aspire-dashboard:8.0.0
```

.NET Aspire App Host



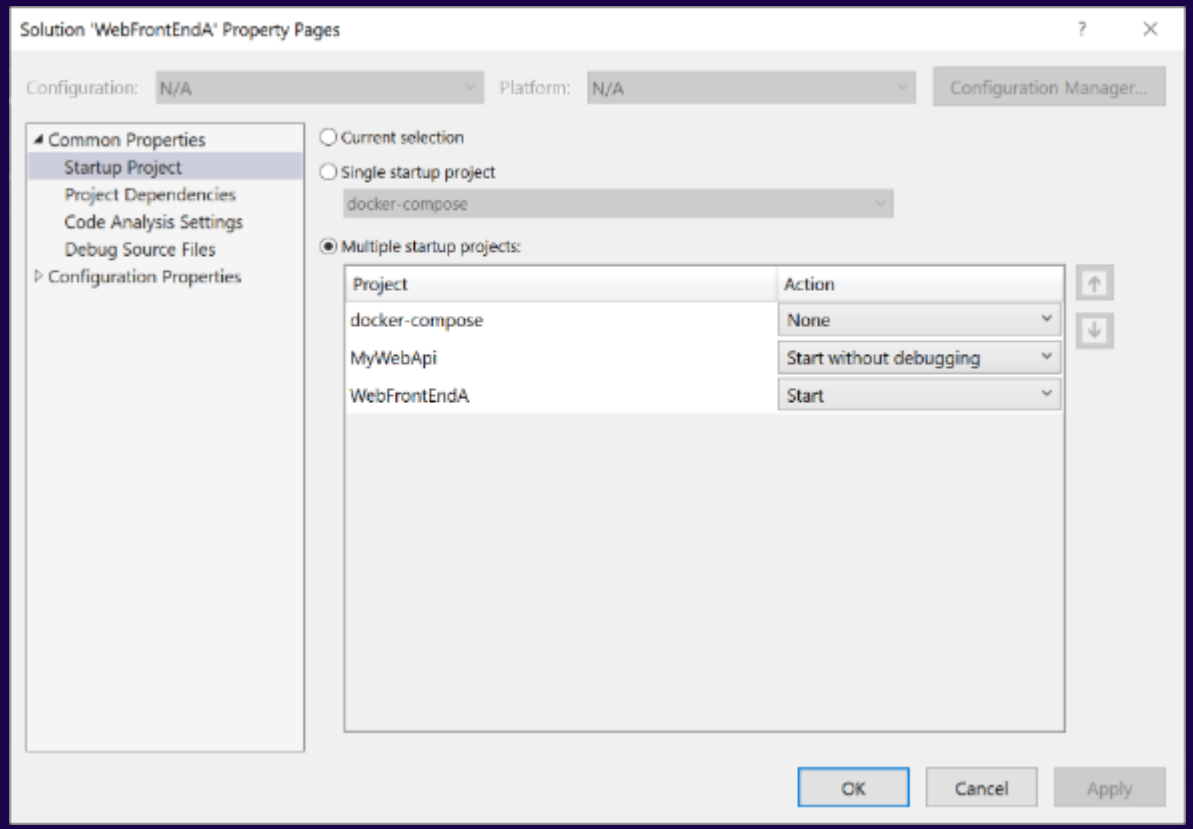
.NET Aspire

A cloud ready stack for building observable, production ready, distributed applications

- Smart Defaults
- Developer Dashboard
- Orchestration
- Service Discovery
- Integrations
- Deployment

Orchestration Before

```
{
    "version": "0.2.0",
    "compounds": [
        {
            "name": "Run all",
            "configurations": [
                "Run products",
                "Run store",
            ]
        }
    ],
    "configurations": [
        {
            "name": "Run products",
            "type": "dotnet",
            "request": "launch",
            "projectPath": "${workspaceFolder}\\Products\\Products.csproj"
        },
        {
            "name": "Run store",
            "type": "dotnet",
            "request": "launch",
            "projectPath": "${workspaceFolder}\\Store\\Store.csproj"
        }
    ]
}
```

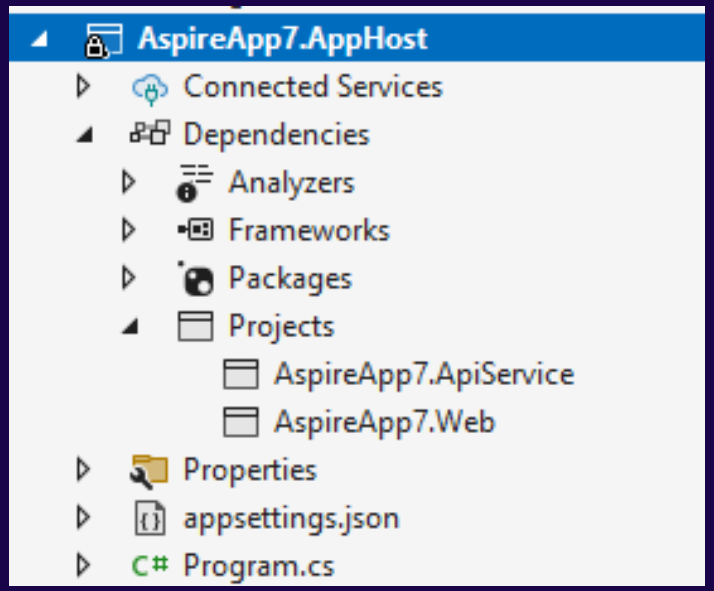


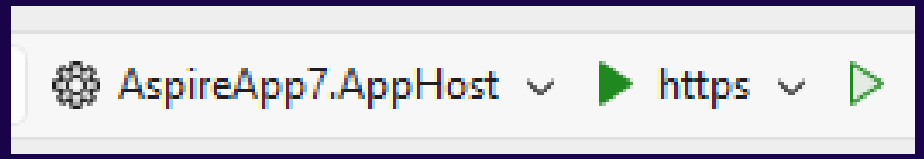
.NET Aspire

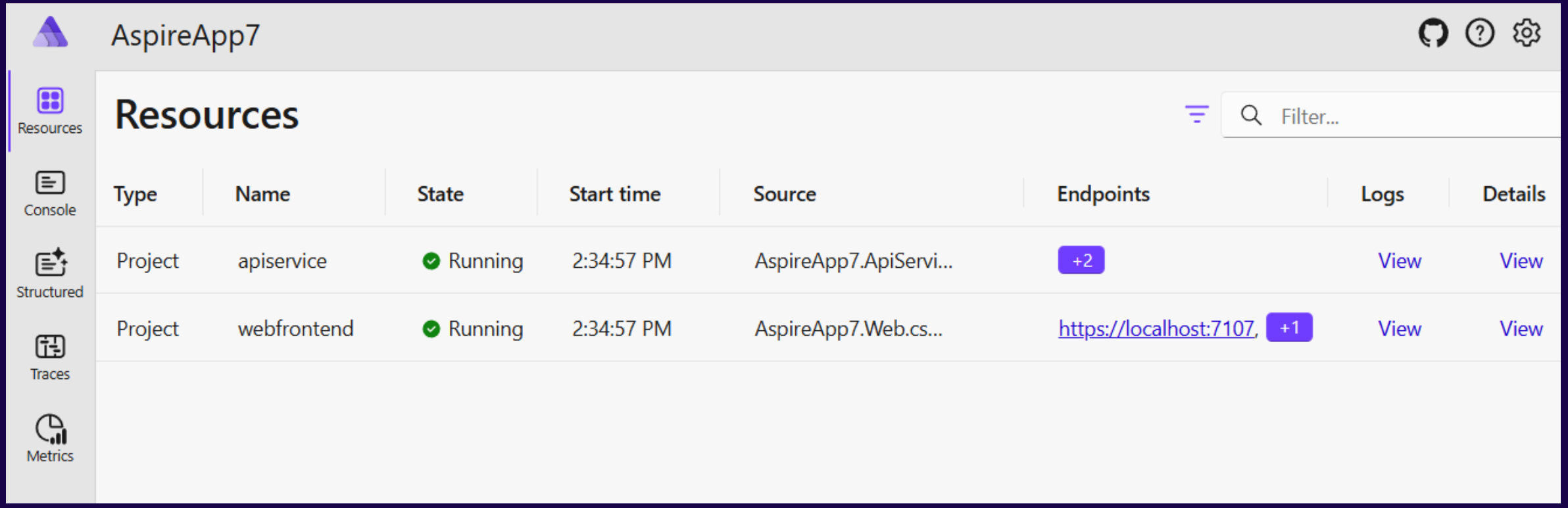
Orchestration

```
var builder = DistributedApplication.CreateBuilder(args);

builder.AddProject<Projects.ApiService>("apiservice");

builder.AddProject<Projects.Web>("webfrontend");

builder.Build().Run();
```

After with .NET Aspire







DEMO

.NET Aspire - Orchestration

Clone: https://github.com/thangchung/practical-dotnet-aspire

.NET Aspire

A cloud ready stack for building observable, production ready, distributed applications

- Smart Defaults
- Developer Dashboard
- Orchestration
- Service Discovery
- Integrations
- Deployment

Service Discovery Before

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "AllowedHosts": "*",
  "ProductEndpoint": "http://localhost:5228",
  "ProductEndpointHttps": "https://localhost:7130"
}
```

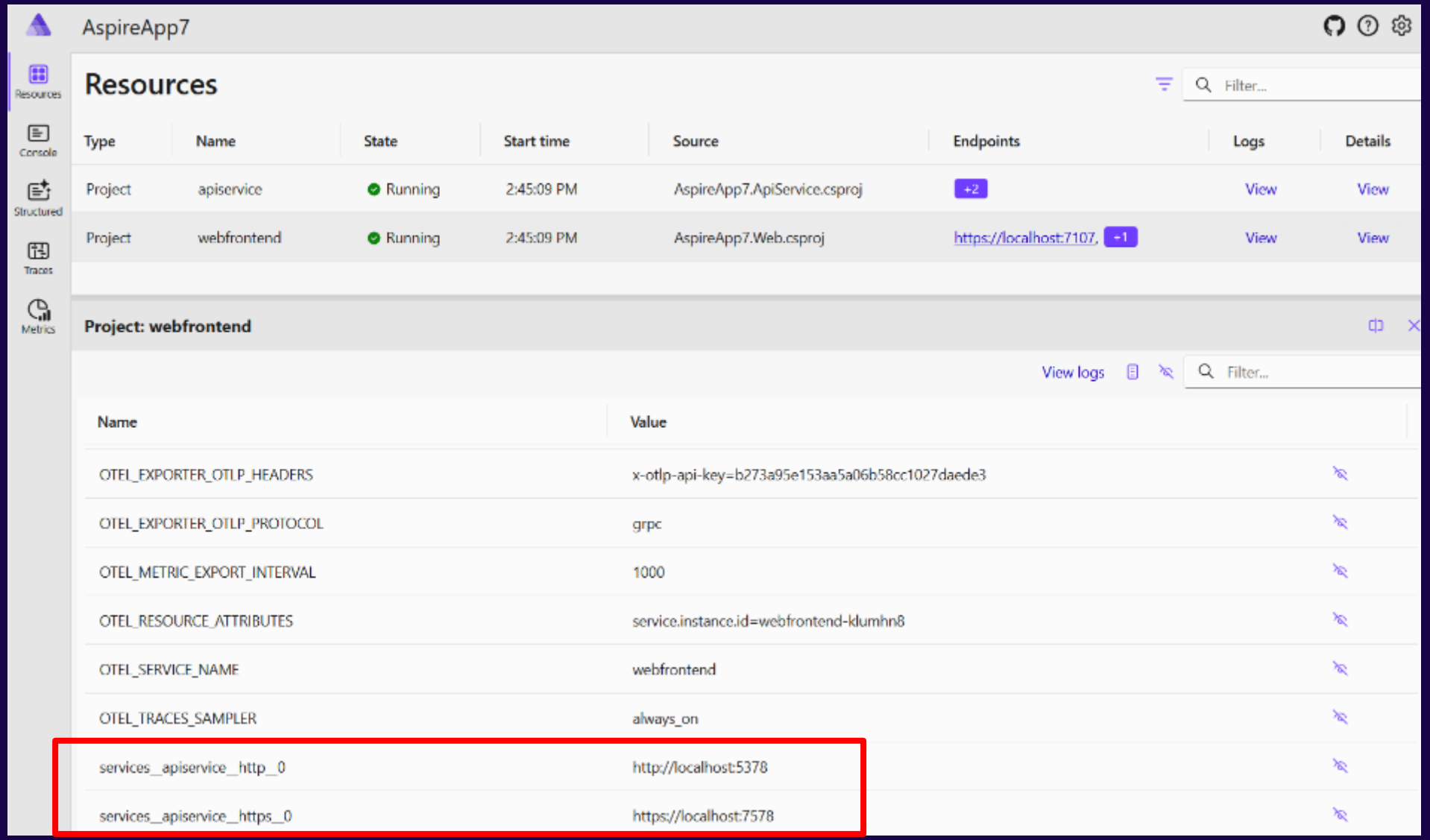
```
builder.Services.AddHttpClient<ProductService>(c =>
{
    var url = builder.Configuration["ProductEndpoint"] ?? throw new
        InvalidOperationException("ProductEndpoint is not set");

    c.BaseAddress = new(url);
});
```

Service Discovery

```
var builder = DistributedApplication.CreateBuilder(args);


var apiService = builder.AddProject<Projects.ApiService>("apiservice");

builder.AddProject<Projects.Web>("webfrontend")
       .WithReference(apiService);

builder.Build().Run();
```

```
builder.Services.AddHttpClient<WeatherApiClient>(client =>
    {
        client.BaseAddress = new("https+http://apiservice");
    });
```

There are still connection strings 😀



DEMO

.NET Aspire - Service Discovery

Clone: https://github.com/thangchung/practical-dotnet-aspire

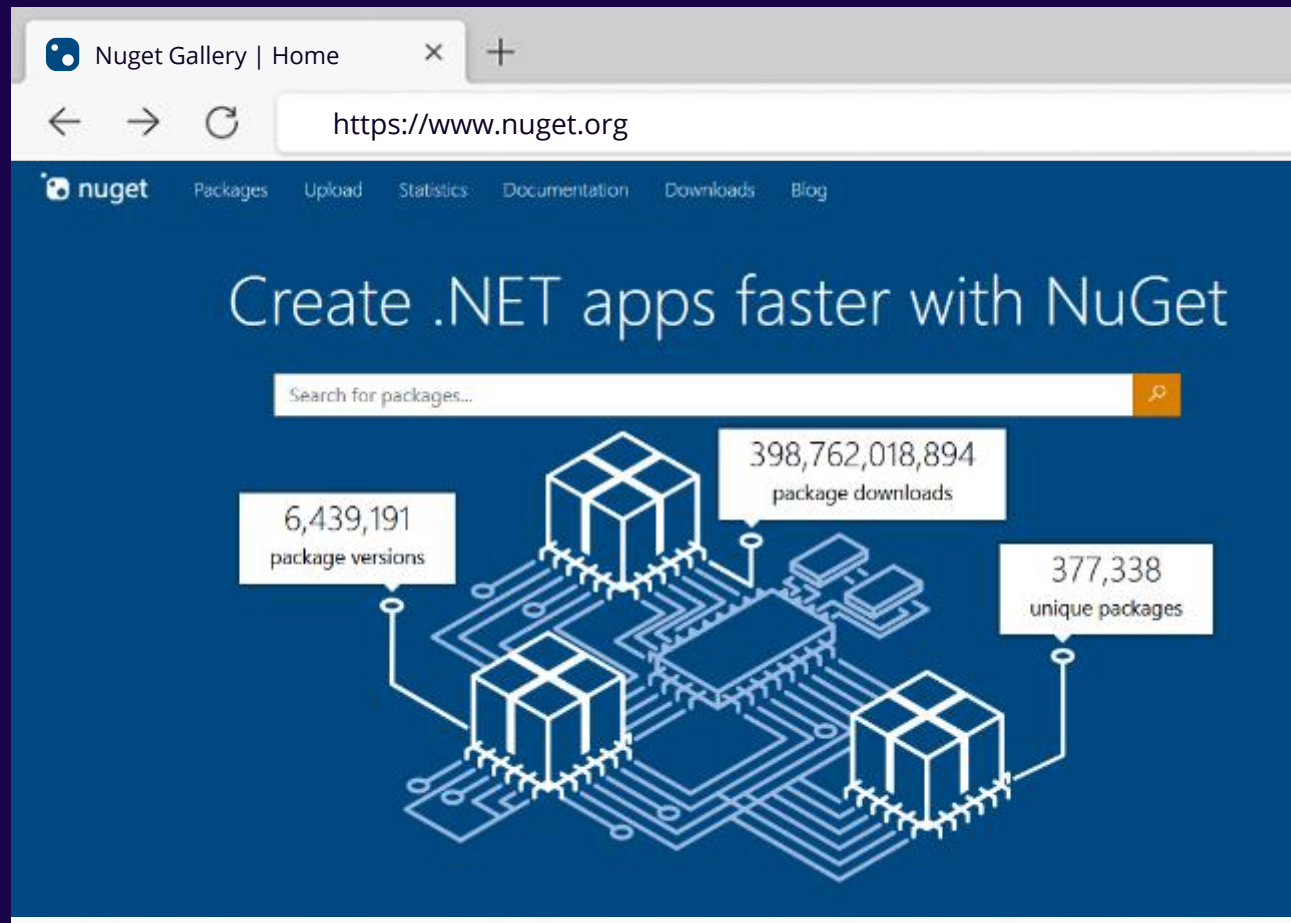
.NET Aspire

A cloud ready stack for building observable, production ready, distributed applications

- Smart Defaults
- Developer Dashboard
- Orchestration
- Service Discovery
- Integrations
- Deployment

.NET Aspire

Integrations



.NET Aspire

Automatically will pull down container image and start it in Docker or Podman!

Hosting Integration

```
var builder = DistributedApplication.CreateBuilder(args);

var postgres = builder.AddPostgres("postgres").WithPgAdmin();
var db = postgres.AddDatabase("db");

var cache = builder.AddRedis("cache");

var apiService = builder.AddProject<Projects.ApiService>("apiservice")
                        .WithReference(db);

builder.AddProject<Projects.Web>("webfrontend")
       .WithReference(apiService)
       .WithReference(cache);

builder.Build().Run();
```

.NET Aspire

Client Integration

```
var builder = WebApplication.CreateBuilder(args);

builder.AddServiceDefaults();
builder.AddRedisOutputCache("cache");

builder.Services.AddRazorComponents()
    .AddInteractiveServerComponents();

var app = builder.Build();

app.UseOutputCache();

app.MapRazorComponents<App>()
    .AddInteractiveServerRenderMode();

app.MapDefaultEndpoints();

app.Run();
```

Configures service with retries, corresponding health checks, logging, and telemetry!!!!

DEMO

.NET Aspire - Integrations

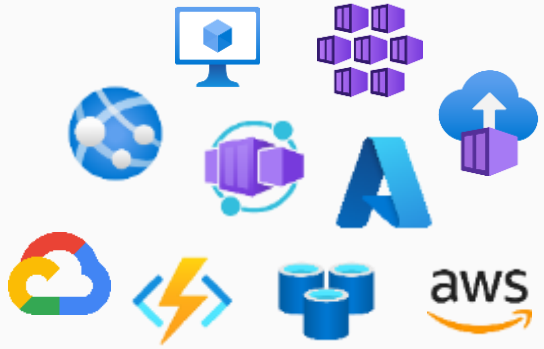
Clone: https://github.com/thangchung/practical-dotnet-aspire

.NET Aspire

A cloud ready stack for building observable, production ready, distributed applications

- Smart Defaults
- Developer Dashboard
- Orchestration
- Service Discovery
- Integrations
- Deployment

Flexible Integrations & Deployment

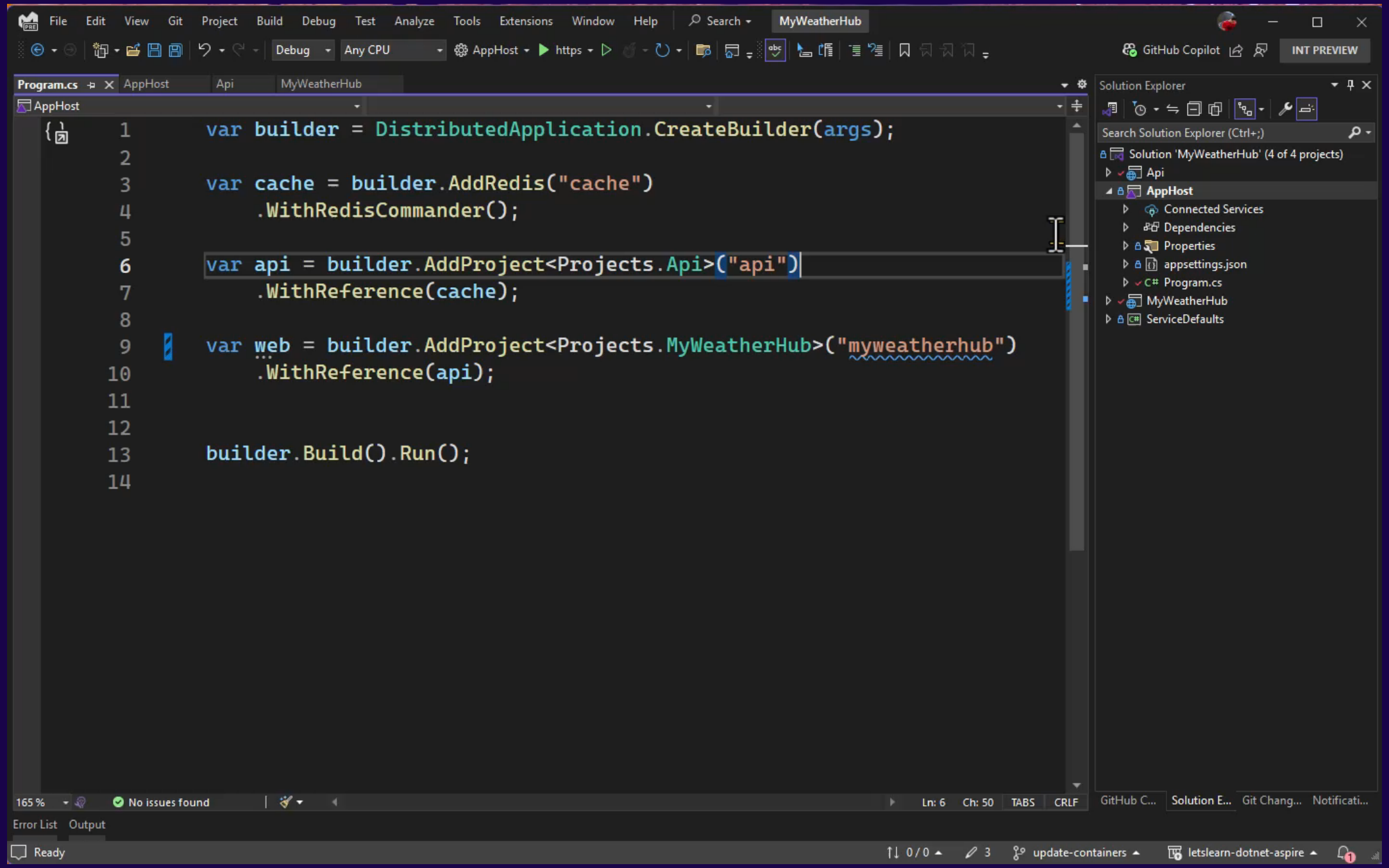


How You Do It Today!

AWS CDK

Azure Dev CLI

Visual Studio


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search
MyWeatherHub
Debug
Any CPU
AppHost
https
GitHub Copilot
INT PREVIEW
Program.cs
AppHost
Api
MyWeatherHub
AppHost
var builder = DistributedApplication.CreateBuilder(args);
var cache = builder.AddRedis("cache")
.WithRedisCommander();
var api = builder.AddProject<Projects.Api>("api")
.WithReference(cache);
var web = builder.AddProject<Projects.MyWeatherHub>("myweatherhub")
.WithReference(api);
builder.Build().Run();
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'MyWeatherHub' (4 of 4 projects)
Api
AppHost
Connected Services
Dependencies
Properties
appsettings.json
Program.cs
MyWeatherHub
ServiceDefaults
165 %
No issues found
Ln: 6
Ch: 50
TABS
CRLF
GitHub C...
Solution E...
Git Chang...
Notificati...
Error List
Output
Ready
0 / 0
3
update-containers
letslearn-dotnet-aspire

DEVCafe
CONNECT AND SHARE

02

.NET Aspire – What’s new in .NET Aspire 9?

.NET Aspire 9.0

Azure Functions preview support

More secure defaults

"WaitFor" resources to spin up

Start/Stop Resources

Improved Azure Configuration

Simplified acquisition

Improved Azure Container Apps integration

Persistent Containers

Support for .NET 8 & .NET 9

OpenAI Integration

AWS Stable Integrations

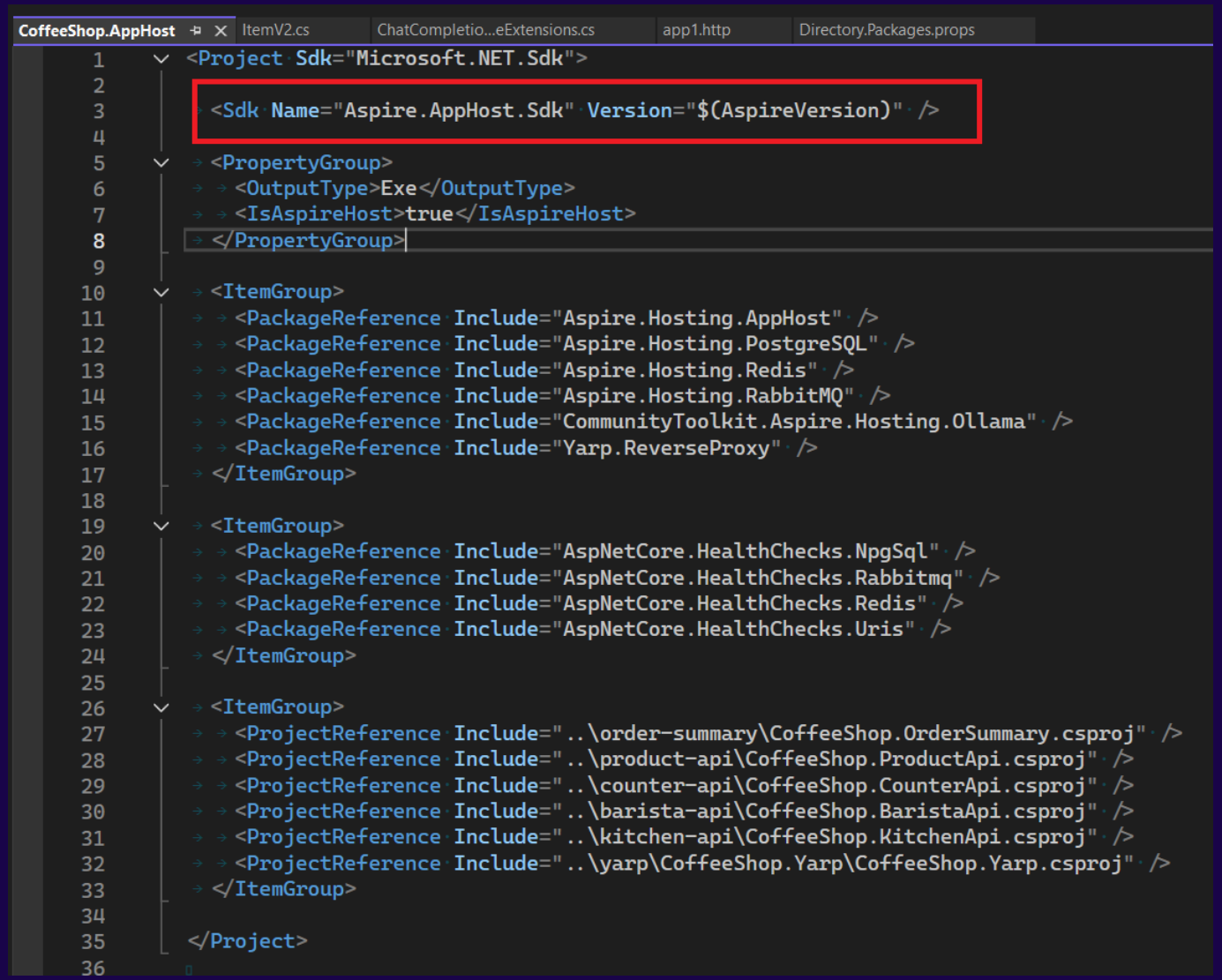
Visual Studio & C# Dev Kit enhancements

.NET Aspire Community Toolkit

Some important features I applied for practical-dotnet-aspire

Check it out...

No longer need a .NET workload



Waiting for dependencies

```
using CoffeeShop.AppHost;

var builder = DistributedApplication.CreateBuilder(args);

var postgresQL = builder.AddPostgres("postgresQL")
                        .WithPgWeb();
var postgres = postgresQL.AddDatabase("postgres");

var redis = builder.AddRedis("redis")
                   .WithRedisCommander();

var rabbitmq = builder.AddRabbitMQ("rabbitmq")
                      .WithManagementPlugin();

var productApi = builder.AddProject<Projects.CoffeeShop_ProductApi>("product-
api")                 .WithReference(postgres).WaitFor(postgres);

var counterApi = builder.AddProject<Projects.CoffeeShop_CounterApi>("counter-
api")                 .WithReference(productApi).WaitFor(productApi)
                      .WithReference(rabbitmq).WaitFor(rabbitmq);

builder.AddProject<Projects.CoffeeShop_BaristaApi>("barista-api")
       .WithReference(rabbitmq).WaitFor(rabbitmq);

builder.AddProject<Projects.CoffeeShop_KitchenApi>("kitchen-api")
       .WithReference(rabbitmq).WaitFor(rabbitmq);

builder.AddProject<Projects.CoffeeShop_Yarp>("yarp")
    .WithReference(productApi).WaitFor(productApi)
    .WithReference(counterApi).WaitFor(counterApi);

builder.Build().Run();
```

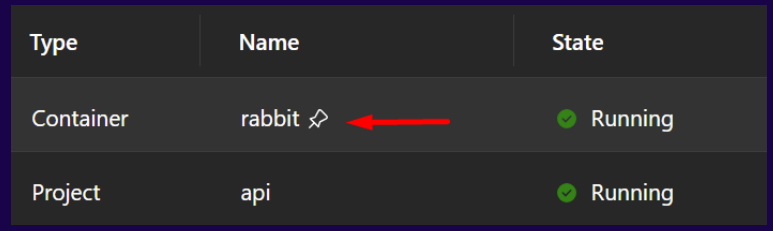
Resource health checks

```
var builder = DistributedApplication.CreateBuilder(args);

var catalogApi = builder.AddContainer("catalog-api", "catalog-
api")                    .WithHttpEndpoint(targetPort: 8080)
                         .WithHttpHealthCheck("/health");

builder.AddProject<Projects.WebApplication1>("store")
       .WithReference(catalogApi.GetEndpoint("http"))
       .WaitFor(catalogApi);

builder.Build().Run();
```

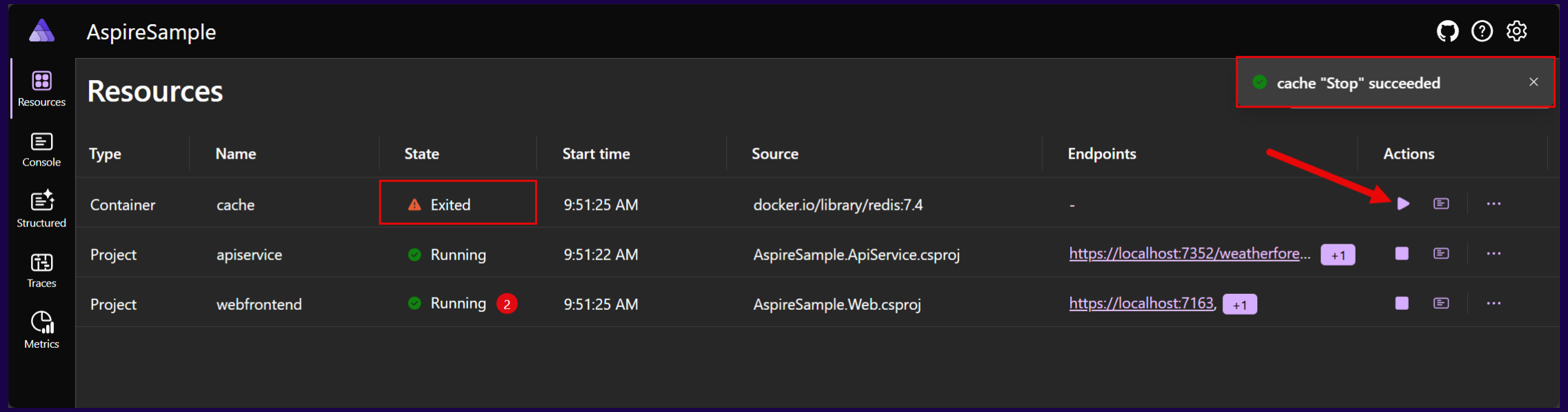
Type	Name	State
Project	catalog-api	Running (Unhealthy)
Project	store	Waiting

Persistent containers

```
using CoffeeShop.AppHost;

var builder = DistributedApplication.CreateBuilder(args);

var postgresQL = builder.AddPostgres("postgresQL")
                        .WithLifetime(ContainerLifetime.Persistent)
                        .WithPgWeb();
var postgres = postgresQL.AddDatabase("postgres");

var redis = builder.AddRedis("redis")
                   .WithLifetime(ContainerLifetime.Persistent)
                   .WithRedisCommander();

var rabbitmq = builder.AddRabbitMQ("rabbitmq")
                      .WithLifetime(ContainerLifetime.Persistent)
                      .WithManagementPlugin();
```

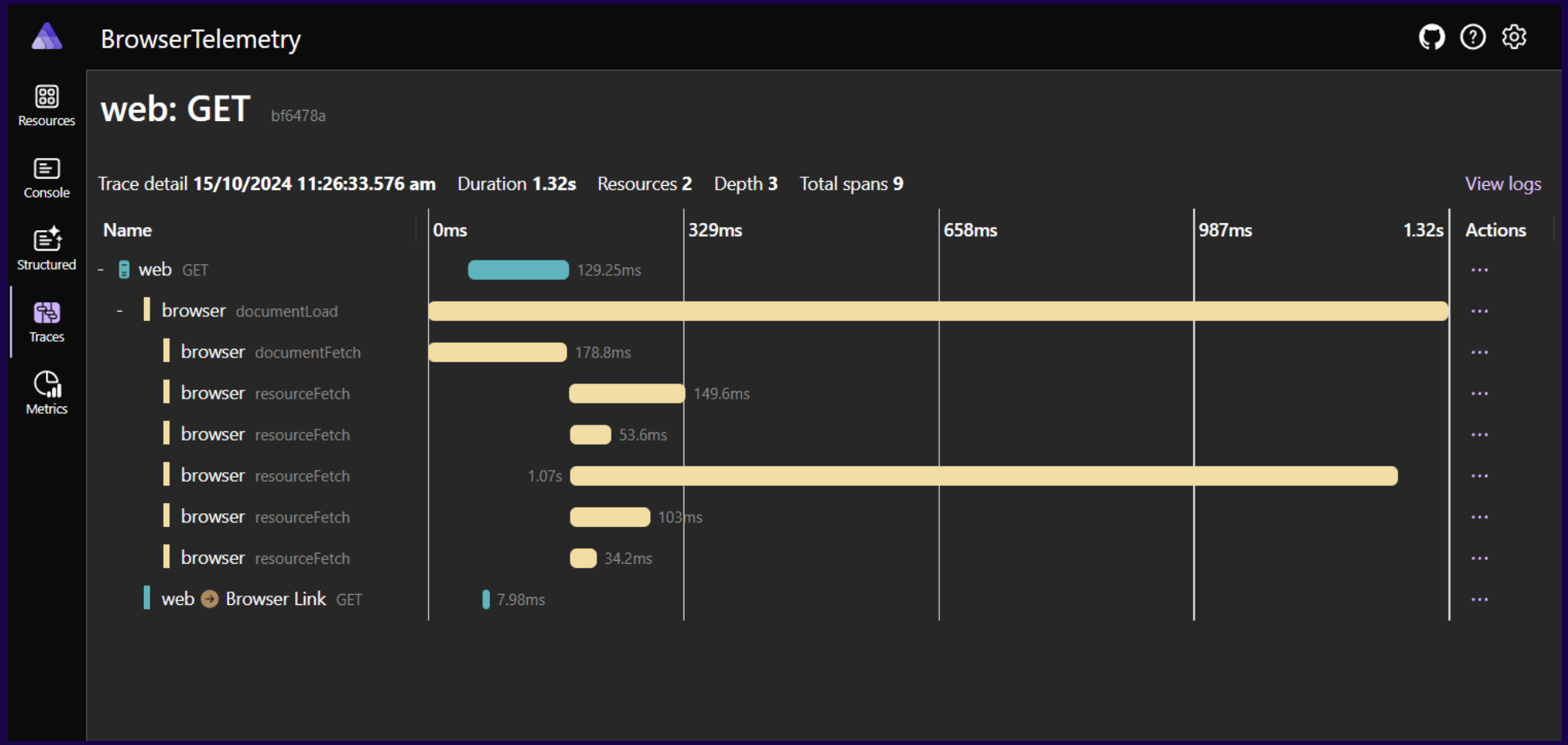


Manage resource lifecycle



Browser telemetry support

The dashboard supports OpenTelemetry Protocol (OTLP) over HTTP and cross-origin resource sharing (CORS). These features unlock the ability to send OpenTelemetry from browser apps to the .NET Aspire dashboard.



OpenAI (Preview)

```
// https://www.nuget.org/packages/Aspire.OpenAI/9.0.0

builder.AddOpenAIClientFromConfiguration("openai");
```

DEVCafe
CONNECT AND SHARE

03

.NET Aspire – Intelligent Apps with .NET Aspire

9

.NET Application

Leveraging AI

Microsoft.Extensions.AI (Middleware)

AI, Data Tools, & Services

UI Components
(Smart Components)

AI SDKS
(OllamaSharp, OpenAI, Azure AI Inference)

Libraries
(Semantic Kernel,AutoGen)

Vector Store SDKs

Developer Tools (Prompty)

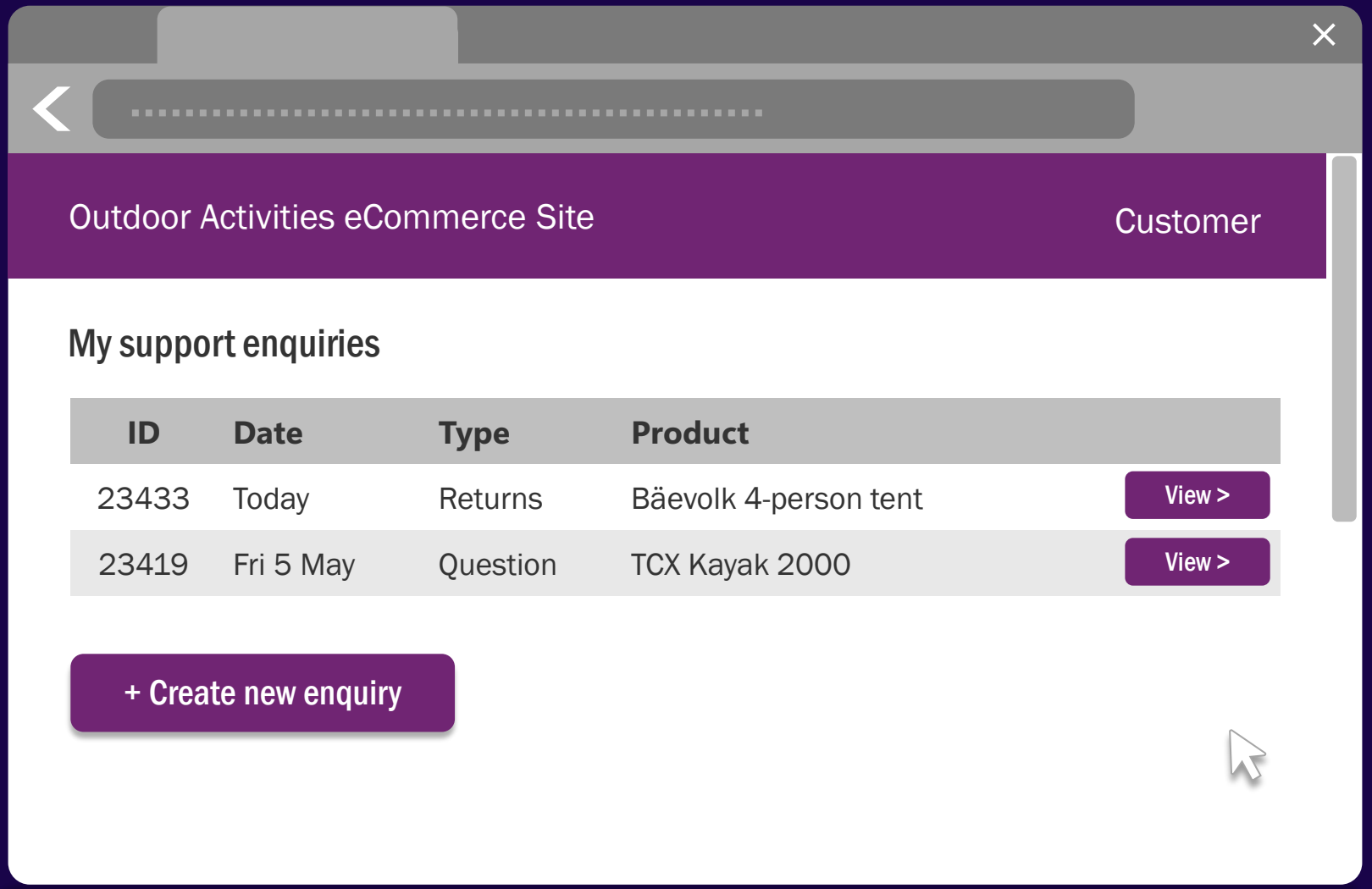
Microsoft.Extensions.AI.Abstractions

Microsoft.Extensions.VectorData

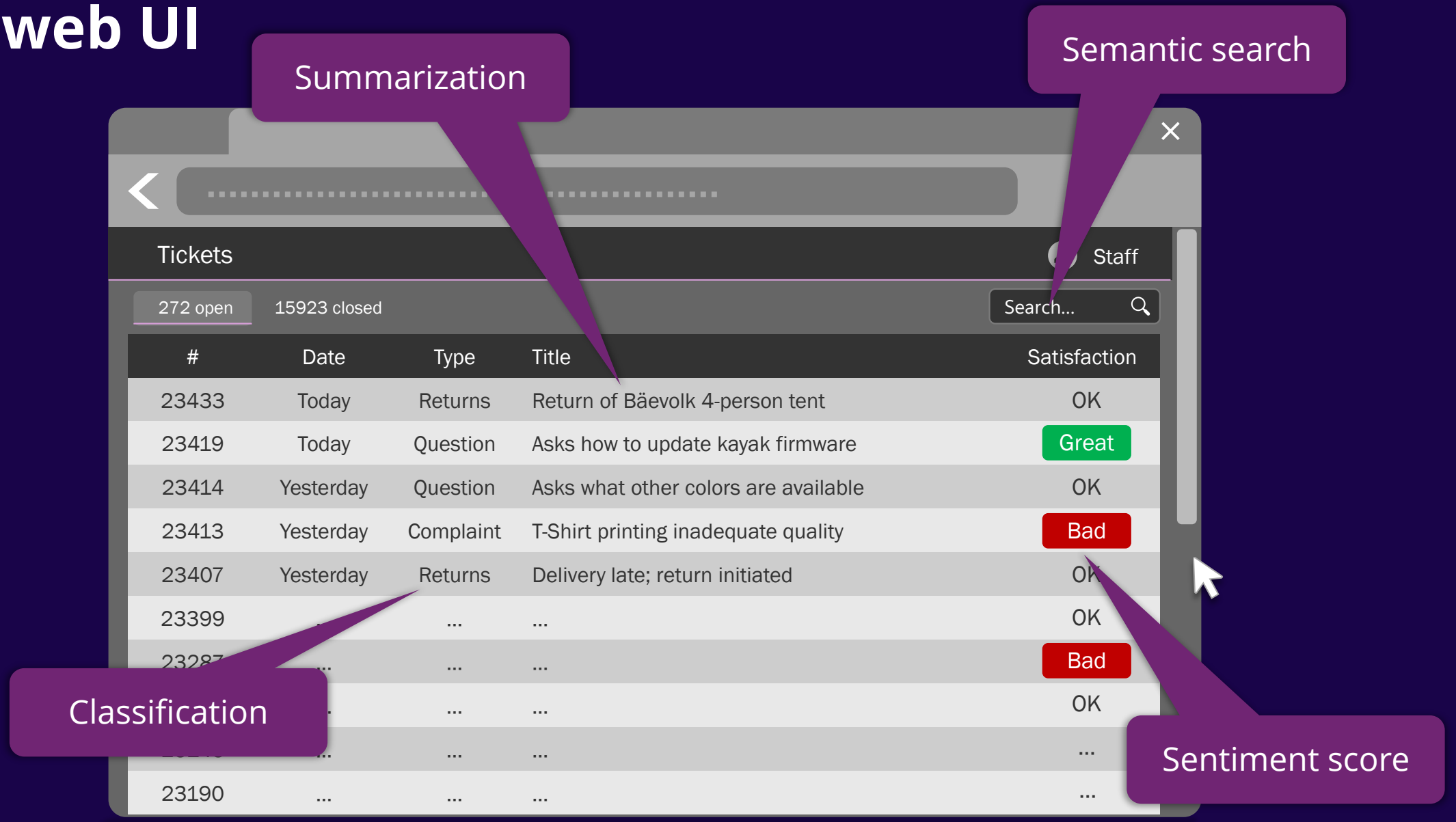
Sample: eShopSupport

https://github.com/dotnet/eShopSupport

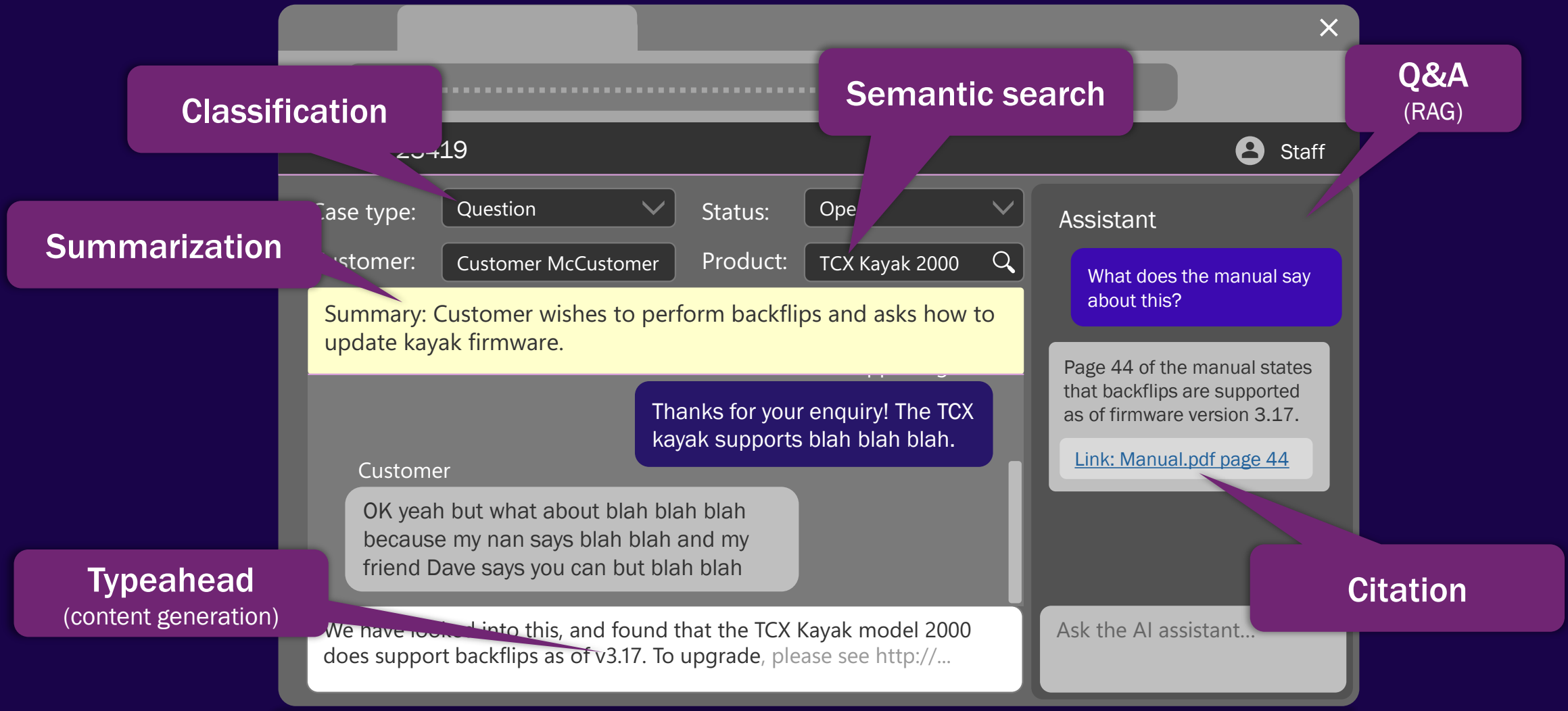
Public site


Outdoor Activities eCommerce Site
Customer
My support enquiries
ID
Date
Type
Product
23433
Today
Returns
Bäevolk 4-person tent
View >
23419
Fri 5 May
Question
TCX Kayak 2000
View >
+ Create new enquiry

Staff web UI


Summarization
Semantic search
Tickets
Staff
272 open
15923 closed
Search...
Date Type Title Satisfaction
23433 Today Returns Return of Bäevolk 4-person tent OK
23419 Today Question Asks how to update kayak firmware Great
23414 Yesterday Question Asks what other colors are available OK
23413 Yesterday Complaint T-Shirt printing inadequate quality Bad
23407 Yesterday Returns Delivery late; return initiated OK
23399 ... ... OK
... ... Bad
... ... OK
... ...
23190 ... ...
Classification
Sentiment score

Staff web UI


Classification
Semantic search
Q&A
(RAG)
Summarization
Staff
Case type:
Question
Status:
Ope
Customer:
Customer McCustomer
Product:
TCX Kayak 2000
Assistant
What does the manual say about this?
Summary: Customer wishes to perform backflips and asks how to update kayak firmware.
Thanks for your enquiry! The TCX kayak supports blah blah blah.
Page 44 of the manual states that backflips are supported as of firmware version 3.17.
Link: Manual.pdf page 44
Customer
OK yeah but what about blah blah blah because my nan says blah blah and my friend Dave says you can but blah blah
Typeahead
(content generation)
Citation
into this, and found that the TCX Kayak model 2000 does support backflips as of v3.17. To upgrade, please see http://...
Ask the AI assistant...

DEMO

.NET Aspire – Microsoft.Extensions.AI (Seed data and Sematic Search)

Clone: https://github.com/thangchung/practical-dotnet-aspire

That’s enough for today…

Get .NET 9

Download .NET 9
aka.ms/get-dotnet-9

More on

https://learn.microsoft.com/en-us/dotnet/aspire/whats-new/dotnet-aspire-9

.NET Aspire

Learning Resources

aka.ms/letslearn/dotnet/aspire

.NET Aspire Videos

aka.ms/aspire/videos

Documentation

aka.ms/dotnet-aspire

Engage with team on GitHub

github.com/dotnet/aspire

Thang Chung
NashTech

@thangchung
https://www.linkedin.com/in/thangchungatwork

Thank you