

(.NET) Aspire aspira a crescere



Tommaso Stocchi
Aspire MammaMia Division

Aspire



Thanks to our sponsors!

improve



Modern Apps are a **Giant Puzzle**



Editor/IDE

- > Operating systems
- > Shells + terminals
- > Coding agents
- > Extensions



Frontends

- > Endless languages + frameworks



Services

- > Databases + storage
- > Auth
- > Caches + queues
- > Messaging



Agents + AI

- > Models, deployments, frameworks



Env

- > Secrets/tokens/keys
- > Provisioning parameters
- > Non-source-controllable assets
- > Changing ports and dependencies



Backends

- > Microservices, monoliths, containers, VMs



Observability

- > Different data pipelines for different parts of the app



Manifests + Scripts

- > Manifest languages (YAML, JSON, Dockerfiles)
- > Hard-to-read scripts (Bash, pwsh)
- > README.md, internal wikis, "tribal knowledge"



Deployment handoff

- > Manual deployments
- > Automated CI/CD
- > Enterprise devops workflow

Complex dev, complex deploy

Dev Loop



- > Impossible to "clone and run"
- > No easy way to add new services
- > Local dev does not reflect actual running system

Deployments



- > Fragile, always-changing assets
- > Complicated handoffs for architecture changes
- > "Only in prod" observability



**Aspire is the tool for
code-first, extensible,
observable dev and
deploy.**

What's in an **Aspirified app**



Aspire AppHost + CLI

Define your stack in **code** — type-safe, readable, and **debuggable**. Aspire runs it all locally, then shifts to **real cloud services** without changing your architecture. Dev to deploy, fully wired.



Aspire Integrations

Plug in **any code or service**, whether it's local or remote. Aspire brings **common patterns** – but can model **any executable, container, or cloud resource**, or **deployment pipeline**.



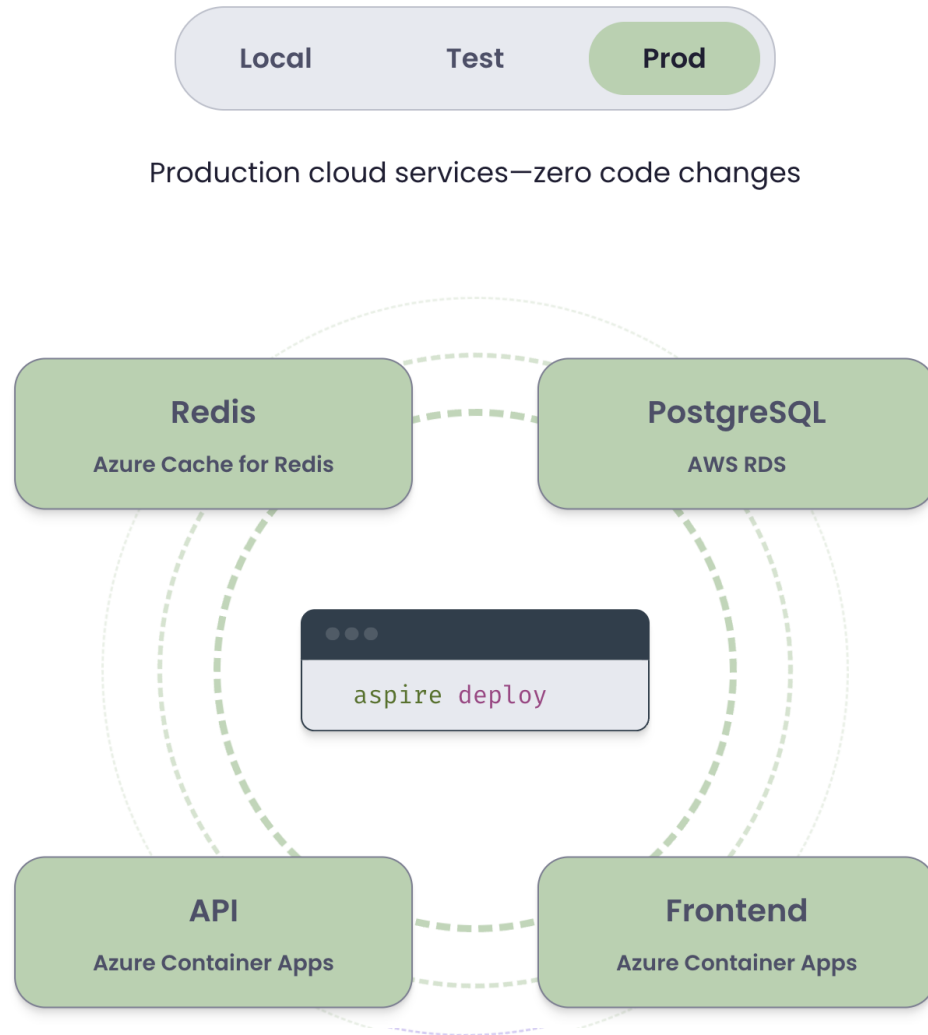
Aspire Dashboard

See your **entire stack** running in one place – all your **logs**, **endpoints**, **environment** variables, even full **OpenTelemetry**. Run it **locally** while you **dev**, then **deploy** it for a familiar overview in your prod environment.

AppHost +
CLI

Dev-first, ops friendly

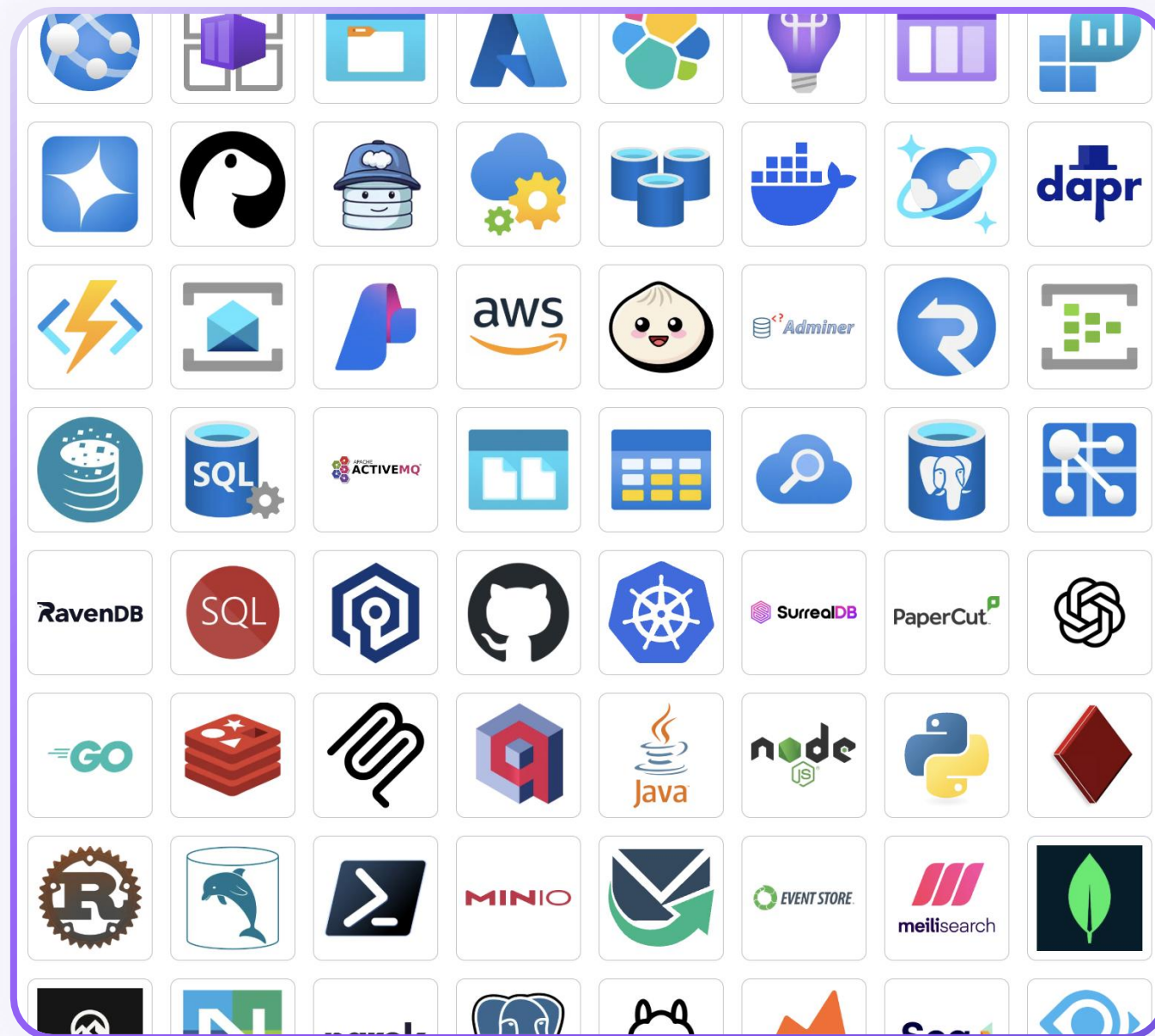
Clone and run any repo, add new services in one command, and deploy – or hand off to your deployment pipeline – without code changes, from one CLI.



Integrations

Building blocks, not black boxes

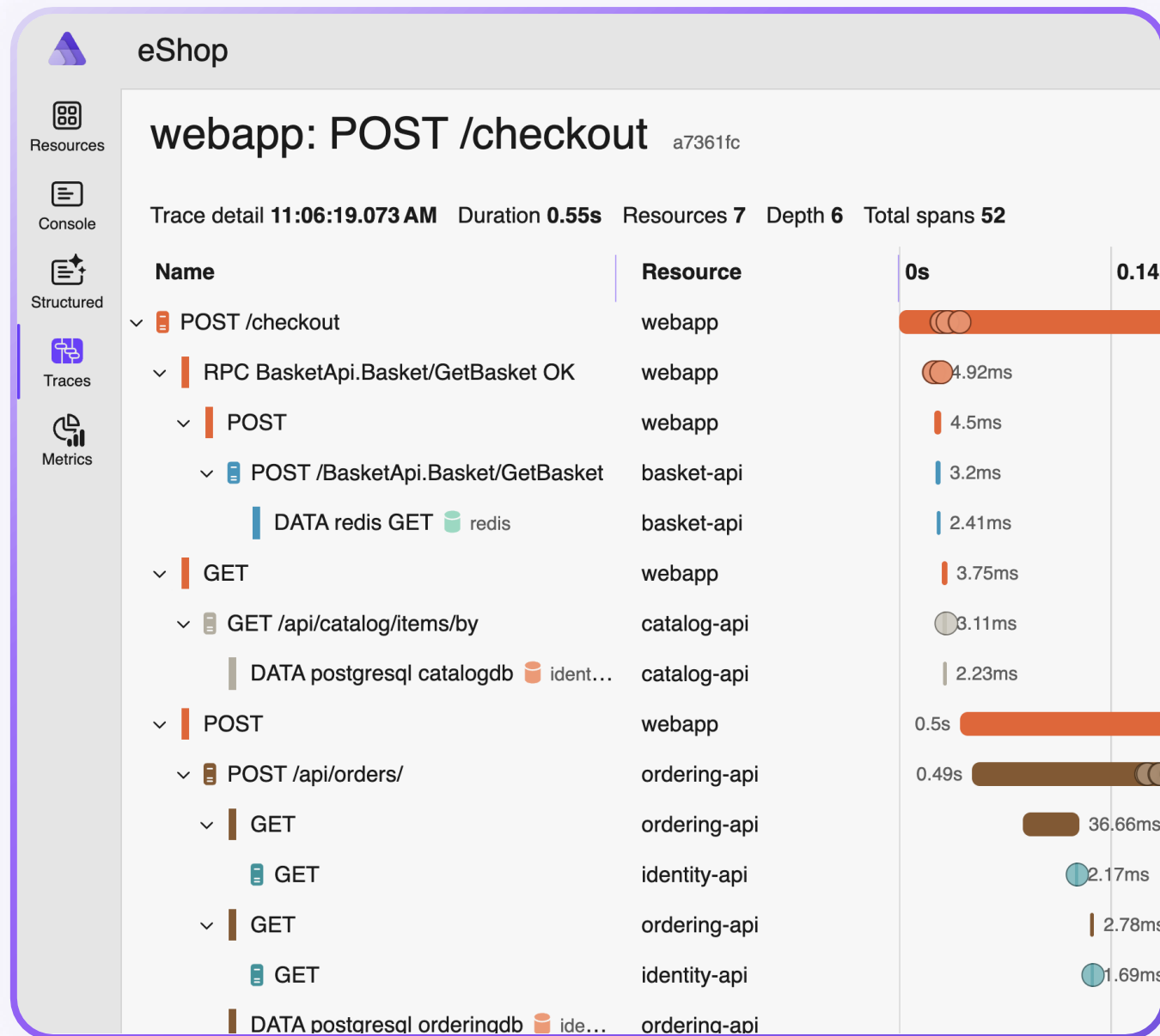
Use our prebuilt packages or bring your own.
Bootstrap and run any type of container, CLI,
or AI agent with a few lines of code.



Dashboard

Observability from the start

Logs, traces, health checks—all local, all wired up automatically. OpenTelemetry included with zero setup.





Demo(s)

What's coming next – UI Playground

[← Back to aspire.dev](#)

Resource Palette

Drag or click to add resources to your Aspire app

Search resources...

All

C#

Python

JavaScript

Go

Java

Apache Kafka
MESSAGING

Apache Kafka event streaming

C#

Python

JavaScript

Go

Java

NATS
MESSAGING

NATS messaging system

C#

Python

JavaScript

Go

OpenAI
AI

OpenAI API integration

C#

Python

JavaScript

Ollama
AI

Local LLM with Ollama

C#

Python

JavaScript

Aspire Playground
Build your distributed app visually

Redis
CACHE
"redis5"

PostgreSQL
DATABASE
"postgres12"
DB: "postgres12db"

OpenAI
AI
"openai85"

Vite App
PROJECT
"vite-app34"

Python App
PROJECT
"python-app23"

View Code

Clear

What's coming next – New MCP server

The screenshot shows the Visual Studio Code interface with the following components:

- EXPLORER:** A project named 'HELLOASPIRE' is open, containing files like `.aspire`, `.vscode`, `app`, `frontend`, `.mcp.json`, `apphost.cs`, `apphost.run.json`, `NuGet.config`, and `opencode.jsonc`.
- TERMINAL:** The terminal shows the execution of the `aspire` CLI commands:
 - `> aspire mcp init`: No agent environments were detected.
 - `> aspire config delete -g firstrun`: Configuration 'firstrun' deleted globally.
 - `> aspire mcp init`: Agent environment configuration complete.
 - `> aspire run`: Finding apphosts... `apphost.cs`. Created settings file at `'.aspire/settings.json'`.
- Summary:** A summary of the setup is provided:
 - cache** - Redis container (v8.2) with health checks passing
 - app** - Python backend (uvicorn) depending on cache
 - frontend** - Node.js frontend (npm/vite) depending on app
- CHAT:** A table showing the status of the components:

Component	Type	Status	URL
cache	container	(Healthy)	localhost:6379
app	Executable	Running (Healthy)	https://localhost:37609
frontend	Executable	Running	http://localhost:43587
app	Executable	Finished	-
frontend	Executable	Finished	-

At the bottom, the terminal prompts the user to "Press CTRL+C to stop the apphost and exit."

Try it **today**

Get Aspirified



aka.ms/aspire-discord



[@aspire.dev](https://twitter.com/aspiredev)



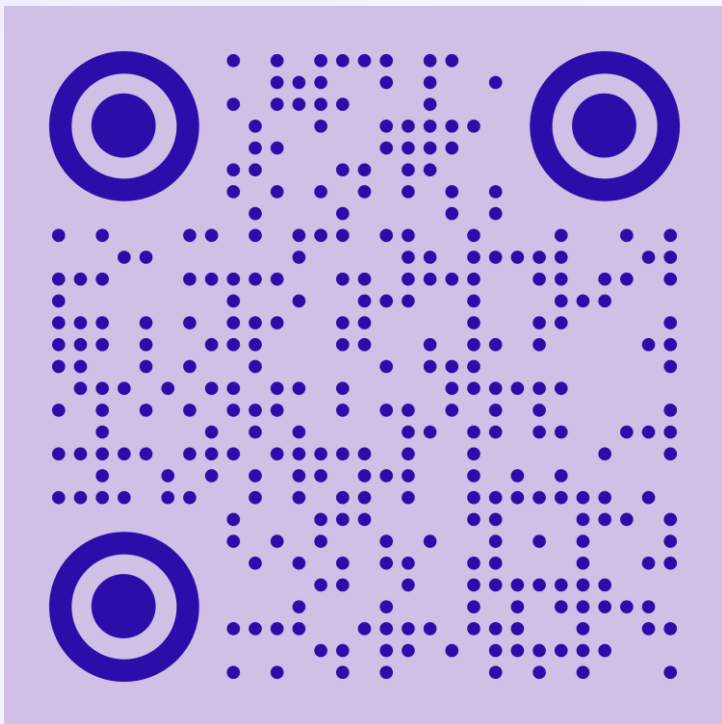
[@aspiredotdev](https://twitter.com/aspiredotdev)



[/aspiredotdev](https://www.youtube.com/channel/UCaspiredotdev)



[dotnet/aspire](https://github.com/dotnet/aspire)



I video saranno scaricabili
presto sul sito previa login
<https://dotnetconference.it>

Thank you!

Tommaso Stocchi

Cloud Solution Architect, Microsoft

@bsky.tommasostocchi.dev

