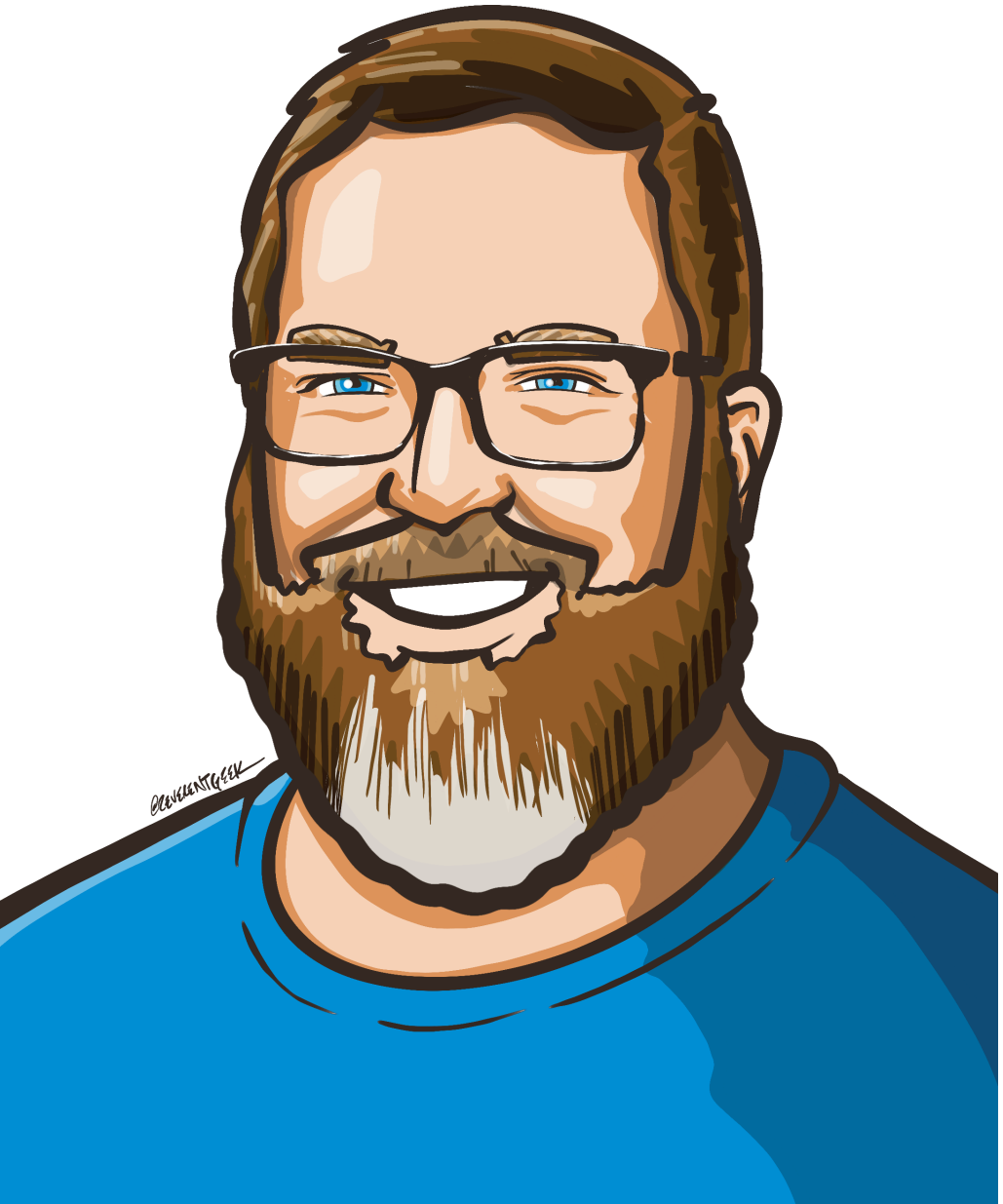
ASPIRING .NET WITH OPENAI AND OLLAMA

CHRIS AYERS

@chris-ayers.com - chris-ayers.com



Chris Ayers

Principal Software Engineer
Azure EngOps AzRel
Microsoft

BlueSky: @chris-ayers.com

LinkedIn: - chris-l-ayers

Blog: https://chris-ayers.com/

GitHub: Codebytes

Mastodon: @Chrisayers@hachyderm.io

~~Twitter: @Chris_L_Ayers~~

Aspire is designed to improve the experience of building cloud-native apps

Why Aspire

Orchestration

- Service Management
- Configuration
- Service Discovery
- Health & Telemetry

Integrations

- Azure Services
- Local Development
- Multi-Platform
- Community Ecosystem

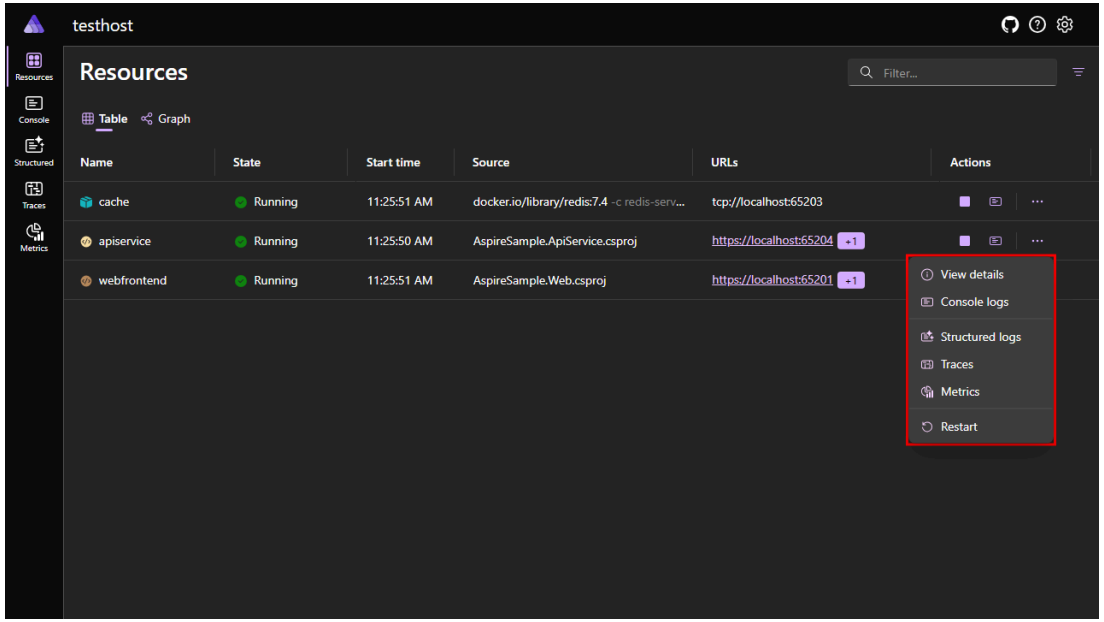
Tooling

- IDE Integration
- Dashboard
- AppHost & CLI

Service Defaults & Features

- **Security Defaults**
  - Azure AD authentication
  - Secure secrets management
- **Resilience & Scalability**
  - Automatic retry policies
  - Circuit breakers for failure protection
- **Custom Commands**
  - Workflow automation
  - Database migrations, data seeding
  - Development environment reset
- **Launch Profiles**
  - Multiple environments (local, cloud, test)
  - Environment variable configuration

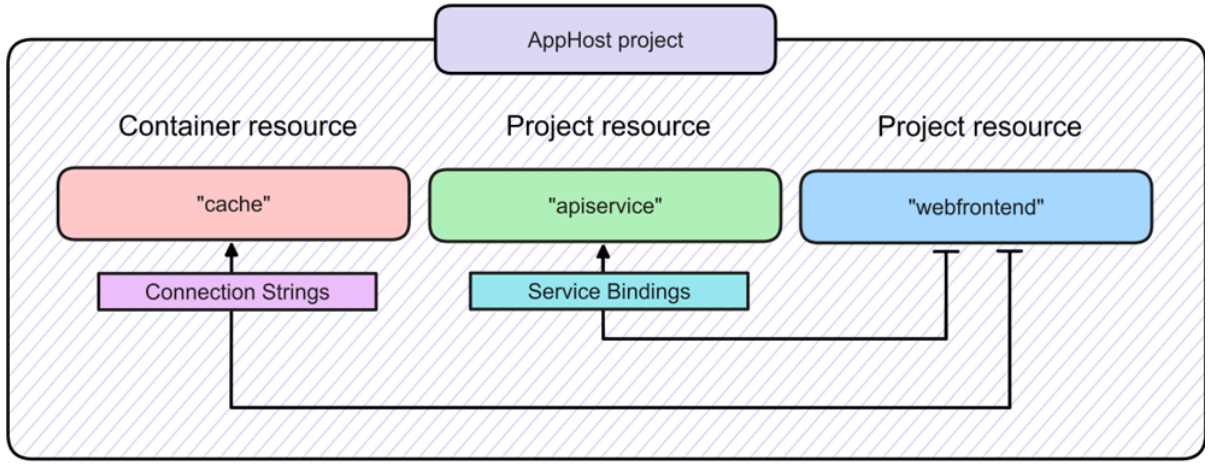
Aspire Dashboard

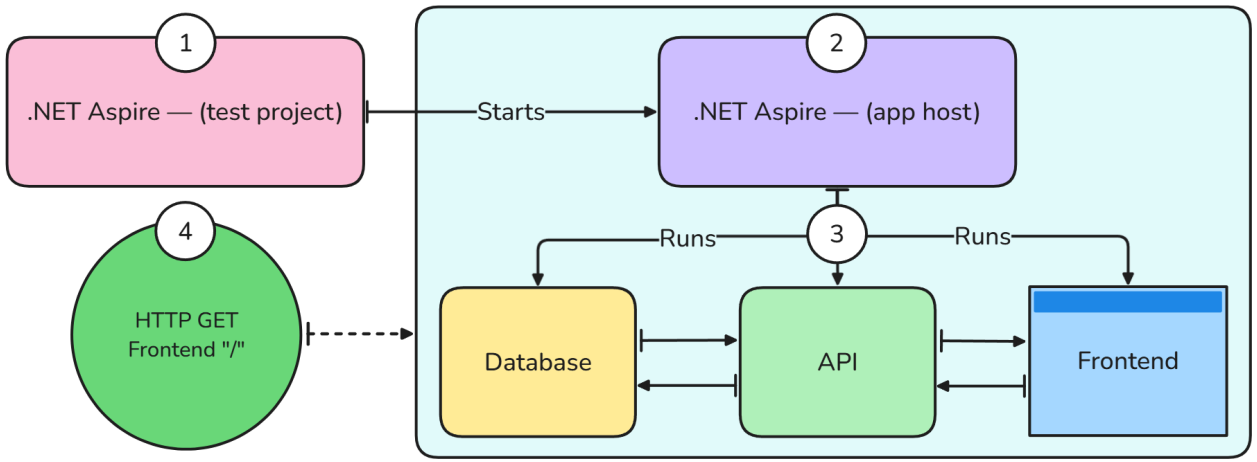


- Real-time Monitoring
- Interactive Debugging
- Resource Management
- Secret Management
- Copilot AI Debugging
- GenAI Visualizer

Service Discovery and Configuration

- **Automatic Configuration**: AppHost passes settings to services
- **Implicit Discovery**: Services reference only what they need
- **Named Endpoints**: Multiple endpoints per service
- **Environment Variables**: Structured configuration strings

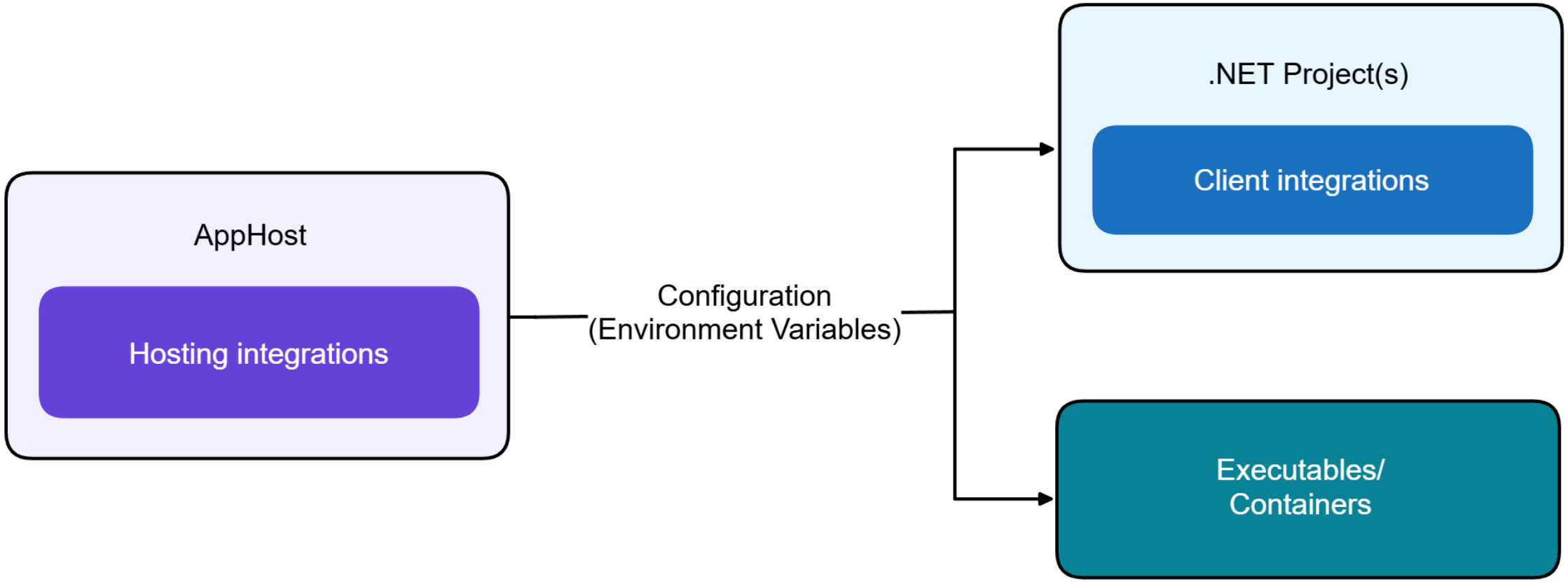




Testing in Aspire

- **Integration Testing**: Test multiple components together
- **Container Testing**: Test with containerized services
- **Emulator Support**: Use local emulators for cloud services
- **End-to-End Testing**: Test full application workflows

Aspire Integrations



Aspire Integrations

Hosting Integrations

- Provision and manage resources
- Automate infrastructure setup
- Cloud-native functionality
- Integrated logging and monitoring
- Authentication and authorization
- Configuration management

Client Integrations

- Configure service connections
- Consume hosted services
- Service discovery
- Service defaults
- Integrated logging and monitoring

Official Integrations

Cloud-Agnostic

- **Databases**: PostgreSQL, MySQL, MongoDB, SQL Server
- **Messaging**: Kafka, RabbitMQ, NATS
- **AI & Observability**: Ollama, Semantic Kernel, OpenTelemetry

Cloud-Specific

- **Azure Services**: OpenAI, Cosmos DB, SQL, Redis, Key Vault, App Service, ACR, App Config
- **AWS Support**: Via Hosting.AWS package

Community Toolkit Integrations

Multi-Language Support

- **JavaScript**: Bun, Deno
- **Systems Programming**: Go, Rust
- **Enterprise**: Java/Spring
- **Extensions**: Node.js, SQL, MongoDB, Redis

Additional Services

- **AI & Search**: Ollama, Meilisearch
- **Data**: SQLite, Data API Builder
- **Web**: Azure Static Web Apps emulator
- **Customization**: Advanced configurations

What's New in .NET Aspire 9.5

⚙️ CLI & Tooling

- **aspire update** - Auto-update packages
- **SSH Remote port forwarding** in VS Code

🎨 Dashboard Enhancements

- **GenAI Visualizer** - Explore AI interactions
- **Multi-resource console** logs view

What's New in .NET Aspire 9.5 (continued)

🤖 New Integrations

- **OpenAI hosting** integration
- **GitHub Models** & **Azure AI Foundry** catalogs
- **Dev Tunnels** hosting support

🚀 Deployment

- **Azure Container App Jobs**
- **Built-in Azure deployment** via `aspire deploy`

Compute Environments & Aspire CLI

Multiple Environments

```
var k8s = builder.AddKubernetesEnvironment("k8s");
var compose = builder.AddDockerComposeEnvironment("docker");

builder.AddProject<Projects.Api>("api")
    .WithComputeEnvironment(compose);

builder.AddProject<Projects.Frontend>("frontend")
    .WithComputeEnvironment(k8s);
```

Aspire CLI

```
# Install
curl -sSL https://aspire.dev/install.sh | bash
dotnet tool install -g Aspire.Cli

# Commands
aspire new | run | add | update
aspire config | publish | deploy
aspire exec
```

Azure Deployment

Targets

- Azure App Service
- Azure Container Apps
- Kubernetes

Azure Developer CLI

```
azd up          # Deploy everything
azd deploy      # App only
azd provision   # Infrastructure only
azd init        # Initialize
```

Features

- Auto-detects app structure
- Environment variable mapping
- Native Aspire support
- Credential providers
- Key Vault integration
- Secure access

Interactive Parameter Prompting

```
// Parameters without defaults trigger dashboard prompts
var apiKey = builder.AddParameter("api-key", secret: true);
var dbUrl = builder.AddParameter("database-url");

var api = builder.AddProject<Projects.Api>("api")
    .WithEnvironment("API_KEY", apiKey)
    .WithEnvironment("DATABASE_URL", dbUrl);
```

Interactive Parameter Prompting (continued)

Features

- **Automatic prompting** for missing parameters
- **Rich form inputs** in dashboard
- **Secret masking** for sensitive data
- **Validation support** with custom rules
- **Save to user secrets** for persistence

Input Types: Text, Password, Choice, Boolean, Number

Local to Cloud Integrations

Component	Local	Cloud
AI Models	Ollama, Foundry local, LM Studio	Azure OpenAI, AI Foundry
Models	Llama 3, Phi-4, Qwen	GPT-4.1, GPT-5-mini, Claude 4
Vector DB	Qdrant, Chroma, pgvector	Azure AI Search, Cosmos DB

Features: Zero-friction transitions • Emulator support • Hybrid development • Auto configuration • Secret management

Microsoft.Extensions.AI

API & Architecture	Description	Development	Description
Unified API	Common interface for AI providers	**Local Development**	Connect to Ollama, LM Studio
Pipeline Architecture	Chain components efficiently	**Cloud Ready**	Same code for Azure OpenAI
DI Integration	Works with .NET service container	**Transport Options**	HTTP, gRPC, direct calls
Provider-Agnostic	Single interface, multiple backends	**Performance**	Auto-batching and throttling

AI Local Development

```
// Local development with Ollama
builder.AddOllama("ollama")
  .WithModel("llama3");

// Add client to use the model
builder.AddOllamaApiClient()
  .AddChatClient();
```

```
public class ExampleService(
    IChatClient chatClient)
{
    // Use chat client...
}
```

- **Zero Cost Development**: No API charges for local models
- **Privacy & Security**: All data stays on your machine
- **Offline Capability**: Work without internet connectivity
- **Fast Iteration**: No network latency for testing
- **Model Experimentation**: Try different open-source models

AI Cloud Development

```
// Single line change for production
builder.AddAzureOpenAI("ai");

// Same client code works unchanged
builder.AddAzureOpenAIClient()
  .AddChatClient();
```

```
public class ExampleService(
    IChatClient chatClient)
{
    // Use chat client...
}
```

- **Latest Models**: Access to GPT-4, Claude, and cutting-edge AI
- **Global Availability**: 99.9% SLA with worldwide deployment
- **Managed Infrastructure**: No server maintenance required
- **Advanced Features**: Function calling, vision, audio processing
- **Compliance Ready**: SOC 2, HIPAA, and enterprise certifications

The Evolution of .NET AI Frameworks

Previous Landscape

- **Semantic Kernel**: High-level orchestration framework
- **AutoGen**: Multi-agent conversation framework
- **Fragmented Ecosystem**: Different APIs, patterns, abstractions

The Challenge

- **Inconsistent APIs**: Each framework had unique approaches
- **Integration Complexity**: Difficult to combine tools
- **Provider Lock-in**: Hard to switch AI providers
- **Learning Curve**: Multiple frameworks to master

Microsoft Agent Framework

Unified Multi-Agent Platform

- **Consolidation**: Replaces Semantic Kernel & AutoGen
- **Built on Microsoft.Extensions.AI**: Leverages unified abstractions
- **Multi-Agent Orchestration**: Coordinate multiple AI agents
- **Event-Driven Architecture**: Flexible agent communication

Microsoft Agent Framework (continued)

Key Features

- **Agent Types**: Task-based, conversational, autonomous
- **Memory Systems**: Short-term and long-term context
- **Tool Integration**: Function calling and external tools
- **Provider Agnostic**: Works with any AI backend

Learn more: Microsoft Agent Framework

DEMOS

Questions ?

Resources

Links

- .NET Aspire
- What's new in .NET Aspire 9.5
- Microsoft.Extensions.AI
- Microsoft Agent Framework
- Aspirify (legacy; unavailable)
- Aspire Samples
- eShopLite

Follow Chris Ayers

Feedback