



Microsoft Build





Thank you to our Premium Partners!

ClickHouse

cohere

CONFLUENT

TADO

JETBRAINS

Lucid

ON

new relic.

POSTMAN

flake

SPACE AND TIME

Syncfusion

Thank you to our Premium Partners!

ClickHouse

cohere

CONFLUENT

DATADOG

JETBRAINS

Lucid

NEON

new relic.

POSTMAN

snowflake

SPACE AND TIME

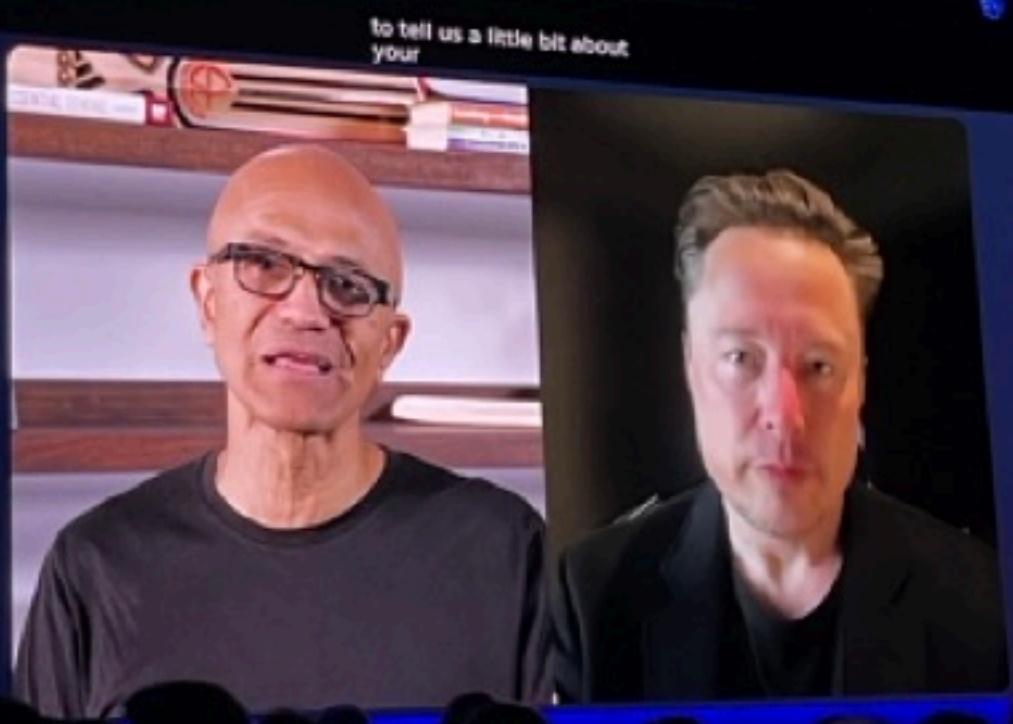
Syncfusion

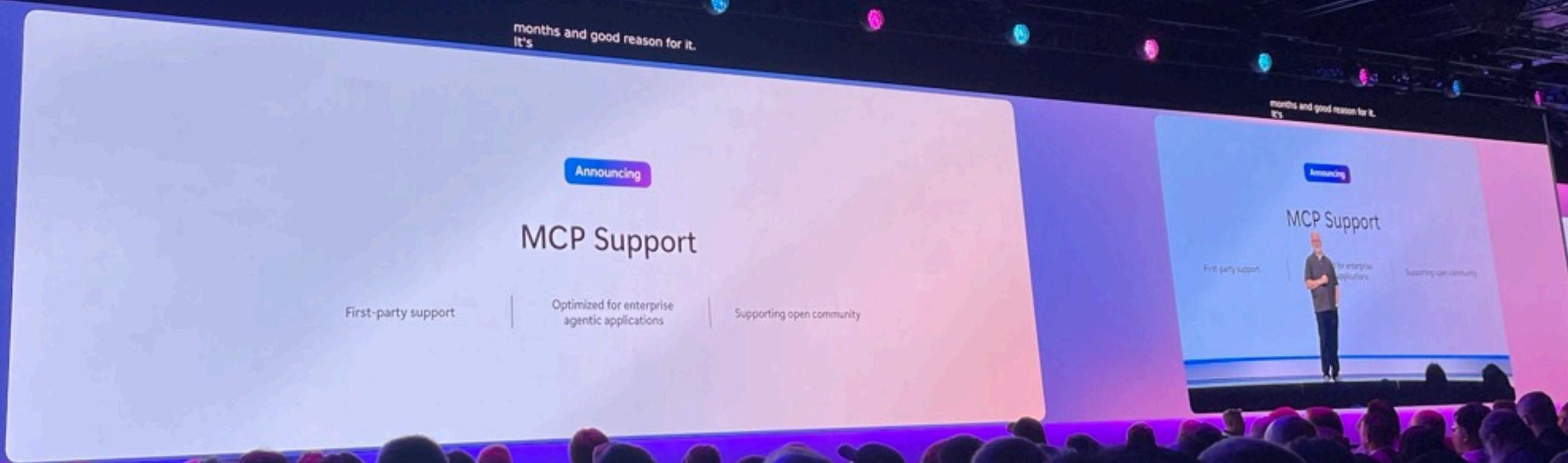


various form factors that
developers use for software

various form factors that
developers use for software

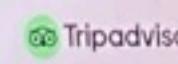
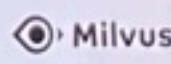
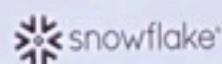
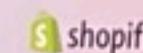
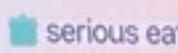
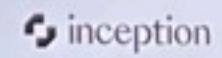
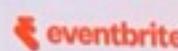
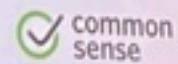
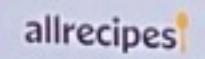






right now.
Wait to hear all of the

Our collaborators



right now.
Wait to hear all of the

O'REILLY



So there is a fantastic talk at
4:30-5:30 today, how

BRK240

Turn your website
into an AI app
(NLWeb)

Monday, May 19
4:30 PM – 5:30 PM
Arch, 705 Pike, Level 6, Room 606

DEM575

Making Agent Memory
Better and Agent Actions
Faster with TypeAgent

Monday, May 19
1:45 PM – 2:00 PM
Arch, 705 Pike, Level 4, Hub, Theater 8

So there is a fantastic talk at
4:30-5:30 today, how

BRK240
DEM575

Turn your website
into an AI app
(NLWeb)

Monday, May 19
4:30 PM – 5:30 PM
Arch, 705 Pike, Level 6, Room 606

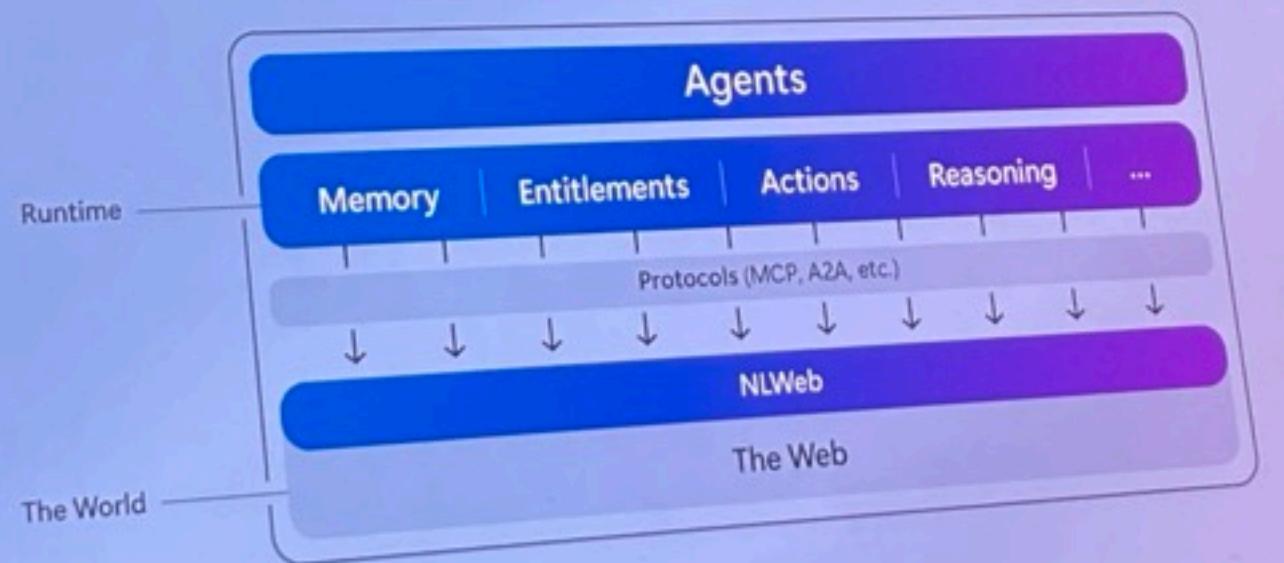
Making Agent Memory
Better and Agent Actions
Faster with TypeAgent

Monday, May 19
1:45 PM – 2:00 PM
Arch, 705 Pike, Level 4, Hub, Theater 8



Build and see you next year.
[Applause]

Building the open agentic web



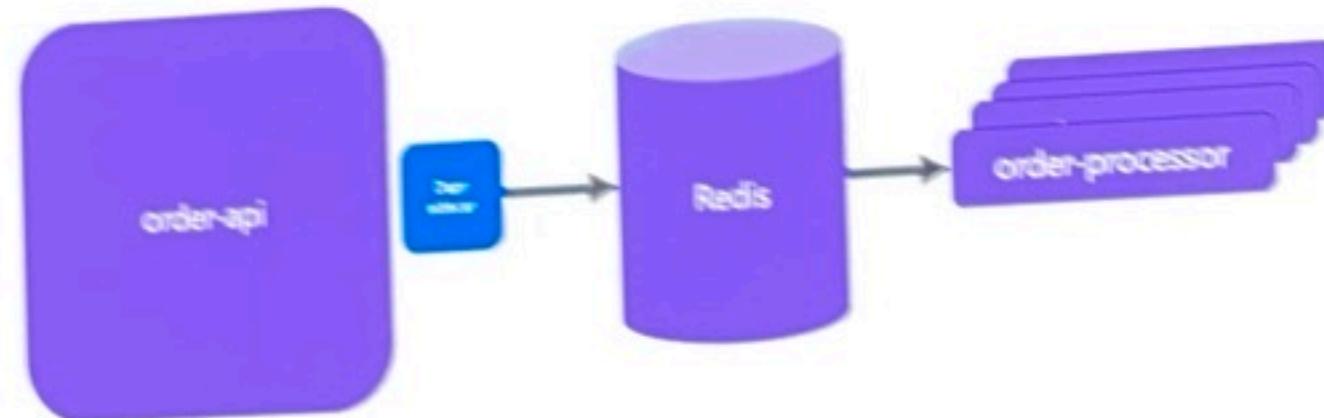
>> Let's roll the video.
>>

>> Let's roll the video.
>>



RUNS TO PICTURE THAT FOR EVERY MICROSERVICE
YOU COULD HAVE EFFECTIVELY RIDING
SUCH AS

Sidecars + Redis



**REPLACE ALL THAT CHAT WITH. INSTEAD,
JUST WRITE THAT ONE LINE OF**



The screenshot shows a Mac OS X desktop environment with a terminal window and a code editor window.

The terminal window is titled "order-api" and contains the command:

```
curl -X POST http://localhost:8080/orders
```

The code editor window displays the file "Program.cs" with the following content:

```
1  var app = builder.Build();
2  var http = app.Services.GetRequiredService<IHttpClientFactory>().CreateClient();
3  var dapr = app.Services.GetRequiredService<DaprClient>();
4
5  app.MapPost("/orders", async (Order order) =>
6  {
7      // baseline synchronous fan-out (3 HTTP calls)
8      // await http.PostAsJsonAsync("http://localhost:6001/reserve", order); // call to inventory-api
9      // await http.PostAsJsonAsync("http://localhost:6002/charge", order); // call to billing-api
10     // await http.PostAsJsonAsync("http://localhost:6003/ship", order); // call to shipping-api
11
12     await dapr.PublishEventAsync("orders-pubsub", "orders", order);
13
14     return Results.Accepted();
15 };
16
17 app.Run("http://0.0.0.0:80");
18
19 record Order(int OrderId, decimal Amount);
20
21 #region
22 // await dapr.PublishEventAsync("orders-pubsub", "orders", order);
23 #endregion
```

PARAMETERS FOR EVERYTHING TO RUN
SMOOTHLY. SO I'M GONNA SPECIFY THAT

iTerm2 Shell Edit View Session Scripts Profiles Toolbelt Window Help

zsh in order-api

```
< zsh in order-api (-zsh)
  Sending HTTP request POST http://localhost:6001/reserve
info: System.Net.Http.HttpClient.Default.ClientHandler[101]
Received HTTP response headers after 17.2026ms - 200
info: System.Net.Http.HttpClient.Default.LogicalHandler[101]
End processing HTTP request after 22.501ms - 200
info: System.Net.Http.HttpClient.Default.LogicalHandler[100]
Start processing HTTP request POST http://localhost:6002/charge
info: System.Net.Http.HttpClient.Default.ClientHandler[100]
  Sending HTTP request POST http://localhost:6002/charge
info: System.Net.Http.HttpClient.Default.ClientHandler[101]
Received HTTP response headers after 5.2581ms - 200
info: System.Net.Http.HttpClient.Default.LogicalHandler[101]
End processing HTTP request after 5.3156ms - 200
info: System.Net.Http.HttpClient.Default.LogicalHandler[100]
Start processing HTTP request POST http://localhost:6003/ship
Info: System.Net.Http.HttpClient.Default.ClientHandler[100]
  Sending HTTP request POST http://localhost:6003/ship
Info: System.Net.Http.HttpClient.Default.ClientHandler[101]
Received HTTP response headers after 4.6106ms - 200
Info: System.Net.Http.HttpClient.Default.LogicalHandler[101]
End processing HTTP request after 4.6605ms - 200
info: Microsoft.AspNetCore.Http.Result.AcceptedResult[1]
Setting HTTP status code 202.
Info: Microsoft.AspNetCore.Routing.EndpointMiddleware[1]
Executed endpoint 'HTTP: POST /orders'
info: Microsoft.AspNetCore.Hosting.Diagnostics[2]
Request finished HTTP/1.1 POST http://localhost/orders - 202 0
- 89.9702ms
*Info: Microsoft.Hosting.Lifetime[0]
Application is shutting down...
alexmang> order-api > dapr run --app-id order-api --dotnet
```

zsh in order-processor (-zsh)

```
demo> cd order-processor
order-processor> code .
alexmang> order-processor
```

in zsh at 11:23:22

THERE'S A TYPO SOMEWHERE, COMPONEN~~T~~ IT'S DEFINITELY A TYPO,

iTerm2 Shell Edit View Session Scripts Profiles Toolbelt Window Help

zsh in order-api

```
< zsh in order-api (-zsh)
  addresses that sidecar will listen to
    --enable-api-logging
  ty. Valid values are: true or false
    --enable-app-health-check
  application using the protocol defined with app-protocol
    --enable-profiling
  HTTP endpoint
    -h, --help
    -k, --kubernetes
  ce against Kubernetes environment.
    --log-level string
  es are: debug, info, warn, error, fatal, or panic (default "info")
    -M, --metrics-port int
  (default -1)
    --placement-host-address string
  service. Format is either <hostname> for default port or <hostname>:
  sport> for custom port (default "localhost")
    --profile-port int
  ver to listen on (default -1)
    --resources-path strings
  tory
    -f, --run-file string
  s for the list of apps to run
    -u, --unix-domain-socket string
  dir. If specified, Dapr API servers will use Unix Domain Sockets

Global Flags:
  --log-as-json      Log output in JSON format
  --runtime-path string
  on directory

unknown flag: --dotnet
```

zsh in order-processor (-zsh)

```
demo cd order-processor
order-processor code .
alexmang order-processor
```

in zsh at 11:25:11

```
    --enable-app-health-check          Enable health checks for the application using the protocol defined with app-protocol
    --enable-profiling                Enable pprof profiling via a
                                     Print this help message
                                     Run the multi-app run template against Kubernetes environment.
    --log-level string                The log verbosity. Valid values are: debug, info, warn, error, fatal, or panic (default "info")
    -M, --metrics-port int           The port of metrics on dapr
                                     (default -1)
    --placement-host-address string   The address of the placement service. Format is either <hostname> for default port or <hostname>:port> for custom port (default "localhost")
    --profile-port int               The port for the profile server to listen on (default -1)
    --resources-path strings          The path for resources directory
    -f, --run-file string            Path to the run template file for the list of apps to run
    -U, --unix-domain-socket string  Path to a unix domain socket dir. If specified, Dapr API servers will use Unix Domain Sockets
```

Global Flags:

```
    --log-as-json                    Log output in JSON format
    --runtime-path string             The path to the dapr runtime installation directory
```

```
unknown flag: --dotnet
```

```
order-api ➤ dapr run --app-id order-api --app-port 80 --resources-path ../components --dotnet run
alexmang ➤ order-api ➤ ➤ in zsh at 11:25:20
```

**BOMBARDING THE APPLICATION AND CALLING
A LOT OF**

```
iTerm2 Shell Edit View Session Scripts Profiles Toolbar Window Help 4/4 10:22 AM Mon May 19 11:26
zsh in demo
< zsh in order-api (detached)
ins: A record lookup error: lookup localhost: i/o timeout app_id=order-api instance=Alexs-M Pro.local scope=dapr.runtime.actors.placement client type=log ver=1.15.5
[ERR0[0076] Failed to connect to placement dns: //localhost:50005: failed to create placement client: rpc error: code = Unavailable desc = ins: A record lookup error: lookup localhost: i/o timeout app_id=order-api instance=Alexs-M Pro.local scope=dapr.runtime.actors.placement client type=log ver=1.15.5
[ERR0[0076] Failed to connect to placement dns: //localhost:50005: failed to create placement client: rpc error: code = Unavailable desc = ins: A record lookup error: lookup localhost: i/o timeout app_id=order-api instance=Alexs-M Pro.local scope=dapr.runtime.actors.placement client type=log ver=1.15.5
[ERR0[0076] Failed to connect to placement dns: //localhost:50005: failed to create placement client: rpc error: code = Unavailable desc = ins: A record lookup error: lookup localhost: i/o timeout app_id=order-api instance=Alexs-M Pro.local scope=dapr.runtime.actors.placement client type=log ver=1.15.5
[ERR0[0077] Failed to connect to placement dns: //localhost:50005: failed to create placement client: rpc error: code = Unavailable desc = dns: A record lookup error: lookup localhost: i/o timeout app_id=order-api instance=Alexs-M Pro.local scope=dapr.runtime.actors.placement client type=log ver=1.15.5
[ERR0[0077] Failed to connect to placement dns: //localhost:50005: failed to create placement client: rpc error: code = Unavailable desc = dns: A record lookup error: lookup localhost: i/o timeout app_id=order-api instance=Alexs-M Pro.local scope=dapr.runtime.actors.placement client type=log ver=1.15.5
zsh in order-processor (order-processor)
< zsh in order-processor (order-processor)
dns: A record lookup error: lookup localhost: i/o timeout app_id=order-processor instance=Alexs-M Pro.local scope=dapr.runtime.actors.placement client type=log ver=1.15.5
[ERR0[0046] Failed to connect to placement dns: //localhost:50005: failed to create placement client: rpc error: code = Unavailable desc = dns: A record lookup error: lookup localhost: i/o timeout app_id=order-processor instance=Alexs-M Pro.local scope=dapr.runtime.actors.placement client type=log ver=1.15.5
[ERR0[0046] Failed to connect to placement dns: //localhost:50005: failed to create placement client: rpc error: code = Unavailable desc = dns: A record lookup error: lookup localhost: i/o timeout app_id=order-processor instance=Alexs-M Pro.local scope=dapr.runtime.actors.placement client type=log ver=1.15.5
[ERR0[0046] Failed to connect to placement dns: //localhost:50005: failed to create placement client: rpc error: code = Unavailable desc = dns: A record lookup error: lookup localhost: i/o timeout app_id=order-processor instance=Alexs-M Pro.local scope=dapr.runtime.actors.placement client type=log ver=1.15.5
[ERR0[0047] Failed to connect to placement dns: //localhost:50005: failed to create placement client: rpc error: code = Unavailable desc = dns: A record lookup error: lookup localhost: i/o timeout app_id=order-processor instance=Alexs-M Pro.local scope=dapr.runtime.actors.placement client type=log ver=1.15.5
[ERR0[0047] Failed to connect to placement dns: //localhost:50005: failed to create placement client: rpc error: code = Unavailable desc = dns: A record lookup error: lookup localhost: i/o timeout app_id=order-processor instance=Alexs-M Pro.local scope=dapr.runtime.actors.placement client type=log ver=1.15.5
zsh in demo (-zsh)
Last login: Mon May 19 11:21:24 on ttys006
deno curl -X POST http://localhost:80/orders -H "Content-Type: application/json" -d '{"orderId":12,"amount":125.0}' alexmang ➜ deno bombardier -c 50 n 10000 \
-m POST \
-H "Content-Type: application/json" \
-b '{"orderId":42,"amount":99.0}' \
http://localhost:80/orders
```

TO YOUR HARDWARE LIKE THE BEST POSSIBLE MODEL THAT WILL RUN ON IT, RIGHT.

Hello, Foundry Local Explore. Build. Test. Ship... Locally.

Deliver the best model for your users' hardware

Foundry Local auto-selects the best model variant (CPU, GPU, NPU) for the user's device.

Optimised runtimes (ONNX, WebGPU) ensure performance even on modest hardware

Optimized models for performance and quality

Supports advanced quantized ONNX models and device-optimised LLMs (e.g. Phi-4, Mistral).

Models are dynamically pulled from the Azure AI Foundry catalog—no bundling required

Built-in model and service management

Model management service handles download, versioning, and runtime loading

CLI and SDKs simplify integration and lifecycle management

Integrates into the tools and frameworks you love

Integrates with .NET Aspire, Azure OpenAI, Semantic Kernel, LangChain, and more.

OpenAI-compatible APIs for chat and generation streamline adoption.



```
using Microsoft.AI.Foundry.Local;
using OpenAI;
using OpenAI.Chat;
using System.ClientModel;

var alias = "qwen2.5-0.5b";
var manager = await FoundryManager.StartModelAsync(alias);
var model = await manager.GetModelInfoAsync(alias);

ApiKeyCredential key = new ApiKeyCredential(manager.ApiKey);
OpenAIClient client = new OpenAIClient(key, new OpenAIClientOptions
{
    Endpoint = manager.Endpoint
});

var chatClient = client.GetChatClient(model?.ModelId);

CollectionResult<StreamingChatCompletionUpdate> completionUpdates =
    chatClient.CompleteChatStreaming("Why is the sky blue");

Console.WriteLine("[ASSISTANT]: ");
foreach (StreamingChatCompletionUpdate completionUpdate in completionUpdates)
{
    if (completionUpdate.ContentUpdate.Count > 0)
    {
        Console.WriteLine(completionUpdate.ContentUpdate[0].Text);
    }
}
Console.WriteLine();
```

APPLICATION. SO THE CLIENT APPLICATION RECEIVES THAT AND THEN PUTS THAT

Hello, Foundry Local Explore. Build. Test. Ship.

Deliver the best model for your users' hardware

Foundry Local auto-selects the best model variant (CPU, GPU, NPU) for the user's device

Optimized runtimes (ONNX, WebGPU) ensure performance even on modest hardware

Optimized models for performance and quality

Supports advanced quantized ONNX models and device-optimized LLMs (L. Phi-4, Qwen, Mistral)

Models are dynamically pulled from the Azure AI Foundry catalog—no building required

Built-in model and service management

Model management service handles download, versioning, and runtime management

SDKs simplify integration and lifecycle management

Integrates into the tools and frameworks you love

Integrates with .NET Aspire, Azure OpenAI, Semantic Kernel, LangChain, and

AI-compatible APIs for chat and generation streamline adoption



.NET Aspire app host

```
var builder = DistributedApplication.CreateBuilder(args);  
  
var foundryLocal = builder.AddAzureAIFoundryLocalResource("ai");  
  
var chatModel = foundryLocal.AddModel("chat", "gwen2.5-0.5b");  
  
builder.AddProject<Projects.AIFoundryLocal_Web>("aifoundrylocal-web")  
    ....WithReference(chatModel)  
    ....WaitFor(chatModel);  
  
builder.Build().Run();
```

Client app

```
var builder = WebApplication.CreateBuilder(args);  
builder.Services.AddRazorComponents().AddInteractiveServerComponents();  
  
builder.AddServiceDefaults();  
  
builder.AddChatCompletionsClient("chat")  
    ....AddChatClient()  
    ....UseFunctionInvocation()  
    ....UseLogging();
```

NGU, BUT IT SHOULD BE ALL GOOD TO
PULL DOWN A CP

The screenshot shows a web-based monitoring and management interface for a local Foundry demo application. The main title bar reads "LocalFoundryDemo Resources". The left sidebar contains navigation links for "LocalFoundryDemo", "Resources", "Console", "Structured", "Traces", and "Metrics". The "Resources" section is currently selected and displays a table of running services. The table has columns for "Name", "State", "Start time", "Source", "URLs", and "Actions". There are three entries:

Name	State	Start time	Source	URLs	Actions
chat	Running	-	qwen2.5-0.5b		[Edit] [More]
ai	Running	-			[Edit] [More]
aichatweb-app	Running	11:45:44	LocalFoundryDemo.Web.csproj	https://localhost:7016 [+1]	[Edit] [More]

The bottom status bar shows the Windows taskbar with various pinned icons and the system tray indicating network and battery status.

ONE, WHICH IS THAT REQUEST THAT
I MADE TO THE CHAT RESOURCE AND

The screenshot shows a browser window displaying a trace visualization for the 'aichatweb-app: chat qwen2.5-0.5b-instruct-generic-cpu' resource. The main interface includes a sidebar with 'Resources', 'Console', 'Structured', 'Traces' (selected), and 'Metrics'. The main panel shows a timeline with two spans: 'aichatweb-ap' (duration 3.91s) and 'aichatweb-' (duration 7.83s). A detailed view of the second span is open, showing its duration of 15.66s, start time at 0µs, and various attributes:

Name	Value
SpanId	bcb3fd13d4d2b3c3
Name	chat qwen2.5-0.5b-instruct-generic-cpu
Kind	Client
az.namespace	Microsoft.CognitiveServices
gen_ai.operation.name	chat
gen_ai.request.model	qwen2.5-0.5b-instruct-generic-cpu

COOL SO JUST TO CLOSE OUT, WE HAVE QUITE A FEW SESSIONS

Closing out

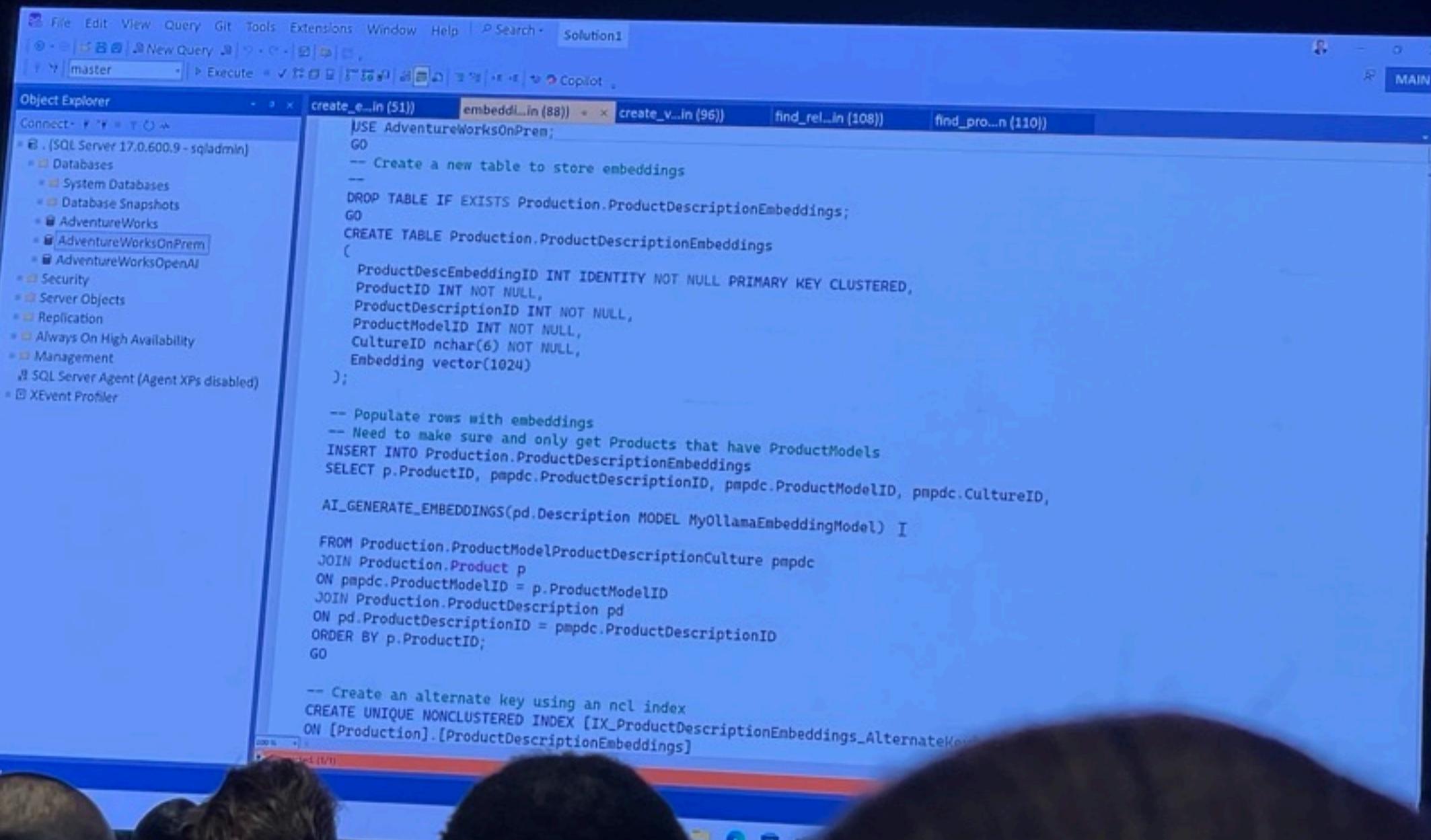
Resources:

- GH: <https://github.com/microsoft/Foundry-Local>
- Docs: aka.ms/foundry-local-docs

Upcoming sessions on Foundry Local:

Code	Session Title	Type	Date/Time
BRK146	Bring AI Foundry to Local: Building cutting-edge on-device AI experiences	Breakout	5/19 1:30pm-2:30pm
DEM526	Build and ship cross platform apps on-device with AI Foundry and ONNX	Demo	5/19 4:45pm-5:00pm
LAB306	Integrating and Enhancing Applications with .NET Aspire	Lab	5/19 2:30pm-3:45pm 5/21 8:45am-10am
LAB307	Building GenAI Apps with C#	Lab	5/19 6pm -7.15pm 5/20 3.30pm-4:45pm

FUNCTION TO GENERATE EMBEDDINGS BASED ON



```
USE AdventureWorksOnPrem;
GO
-- Create a new table to store embeddings
--
DROP TABLE IF EXISTS Production.ProductDescriptionEmbeddings;
GO
CREATE TABLE Production.ProductDescriptionEmbeddings
(
    ProductDescEmbeddingID INT IDENTITY NOT NULL PRIMARY KEY CLUSTERED,
    ProductID INT NOT NULL,
    ProductDescriptionID INT NOT NULL,
    ProductModelID INT NOT NULL,
    CultureID nchar(6) NOT NULL,
    Embedding vector(1024)
);

-- Populate rows with embeddings
-- Need to make sure and only get Products that have ProductModels
INSERT INTO Production.ProductDescriptionEmbeddings
SELECT p.ProductID, pmpdc.ProductDescriptionID, pmpdc.ProductModelID, pmpdc.CultureID,
AI_GENERATE_EMBEDDINGS(pd.Description MODEL MyOllamaEmbeddingModel) I
FROM Production.ProductModelProductDescriptionCulture pmpdc
JOIN Production.Product p
ON pmpdc.ProductModelID = p.ProductModelID
JOIN Production.ProductDescription pd
ON pd.ProductDescriptionID = pmpdc.ProductDescriptionID
ORDER BY p.ProductID;
GO

-- Create an alternate key using an ncl index
CREATE UNIQUE NONCLUSTERED INDEX [IX_ProductDescriptionEmbeddings_AlternateKey]
ON [Production].[ProductDescriptionEmbeddings]
```

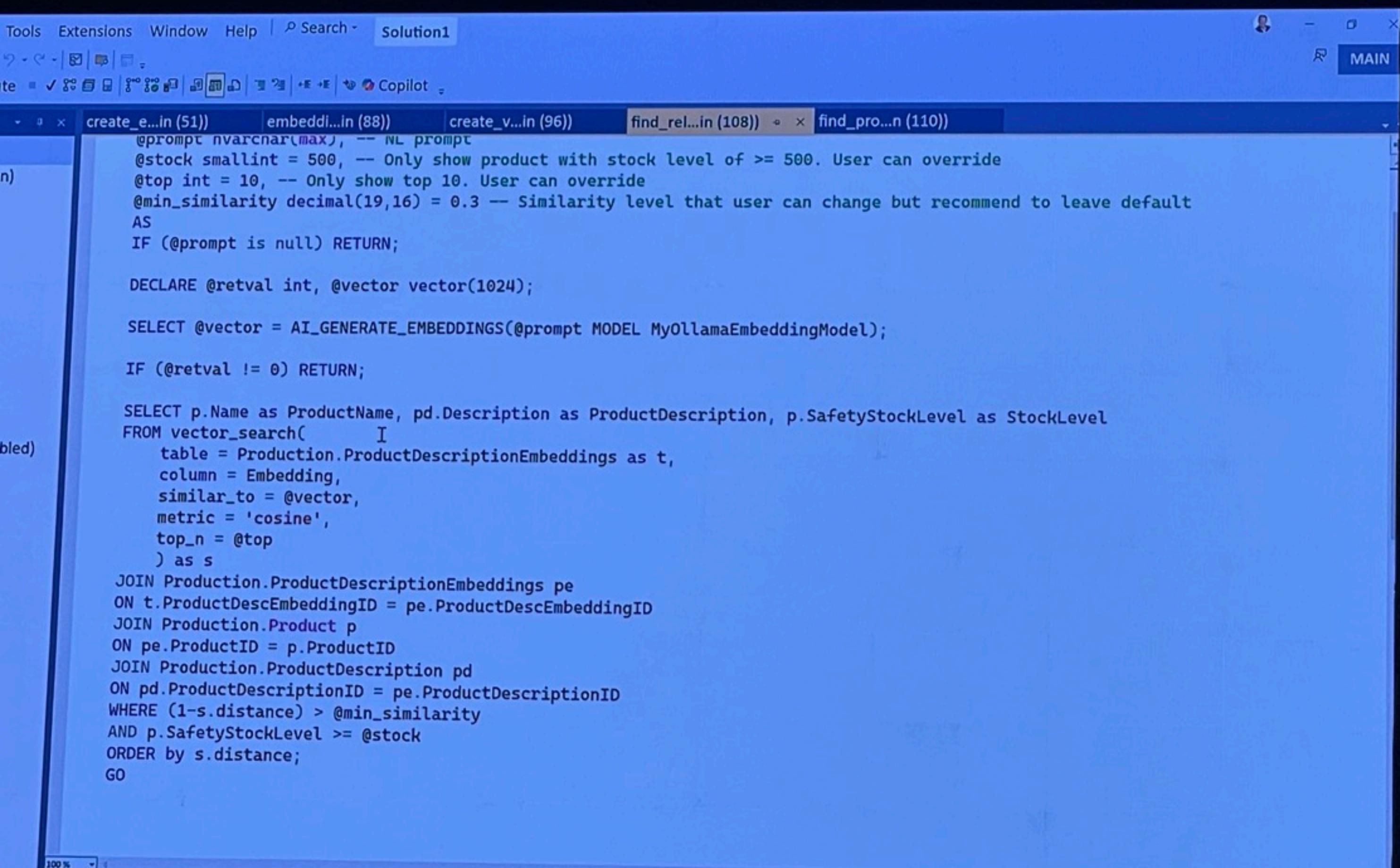
A VECTOR INDEX? WELL, GUESS WHAT? WE'VE GOT THE ABILITY TO DEFINE

The screenshot shows a Microsoft SQL Server Management Studio (SSMS) interface. The title bar reads "Solution1". The menu bar includes "Query", "Git", "Tools", "Extensions", "Window", "Help", and a search bar. The toolbar contains icons for "New Query", "Execute", and "Copilot". Below the toolbar, there are four tabs: "create_e...in (51)", "embeddi...in (88)", "create_v...in (96)" (which is selected), and "find_rel...in". The main pane displays the following T-SQL code:

```
USE [AdventureWorksOnPrem];
GO
CREATE VECTOR INDEX product_vector_index
ON Production.ProductDescriptionEmbeddings (Embedding)
WITH (METRIC = 'cosine', TYPE = 'diskann', MAXDOP = 8);
GO
```



ON THE MODEL I JUST DEFINED. ON
TOP OF THAT, I WILL



The screenshot shows a Microsoft Visual Studio interface with a dark theme. The top menu bar includes Tools, Extensions, Window, Help, Search, and Solution1. The main window displays a T-SQL script:

```
create_e...in (51) | embeddi...in (88)) | create_v...in (96)) | find_rel...in (108)) | find_pro...n (110))
@prompt nvarchar(max), -- NL prompt
@stock smallint = 500, -- Only show product with stock level of >= 500. User can override
@top int = 10, -- Only show top 10. User can override
@min_similarity decimal(19,16) = 0.3 -- Similarity level that user can change but recommend to leave default
AS
IF (@prompt is null) RETURN;

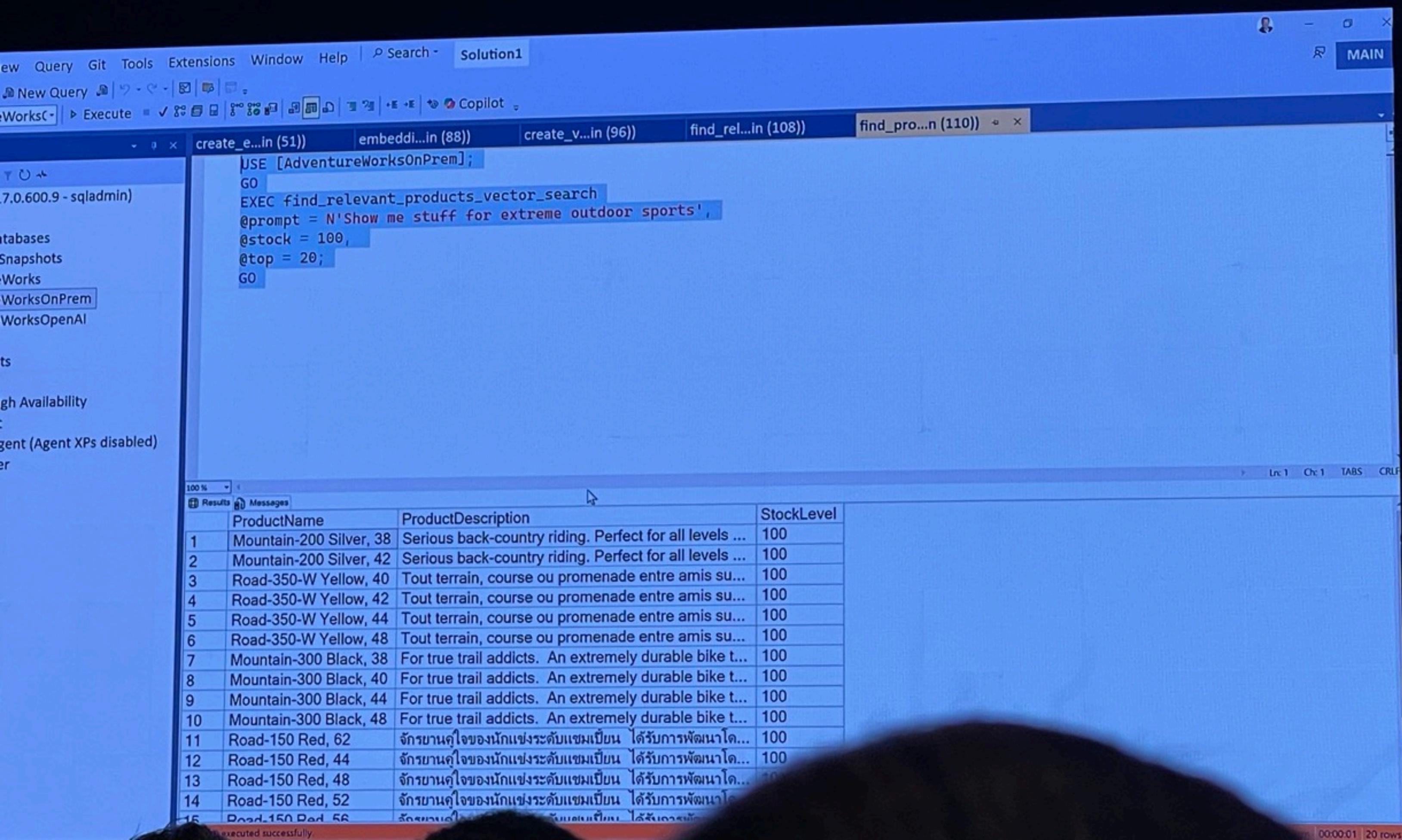
DECLARE @retval int, @vector vector(1024);

SELECT @vector = AI_GENERATE_EMBEDDINGS(@prompt MODEL MyOllamaEmbeddingModel);

IF (@retval != 0) RETURN;

SELECT p.Name as ProductName, pd.Description as ProductDescription, p.SafetyStockLevel as StockLevel
FROM vector_search(
    I
    table = Production.ProductDescriptionEmbeddings as t,
    column = Embedding,
    similar_to = @vector,
    metric = 'cosine',
    top_n = @top
) as s
JOIN Production.ProductDescriptionEmbeddings pe
ON t.ProductDescEmbeddingID = pe.ProductDescEmbeddingID
JOIN Production.Product p
ON pe.ProductID = p.ProductID
JOIN Production.ProductDescription pd
ON pd.ProductDescriptionID = pe.ProductDescriptionID
WHERE (1-s.distance) > @min_similarity
AND p.SafetyStockLevel >= @stock
ORDER by s.distance;
GO
```

PASSING A SIMPLE PROMPT HERE IN ENGLISH, WHAT YOU CAN SEE IS I'M



The screenshot shows a Microsoft SQL Server Management Studio (SSMS) interface. The title bar says "Solution1". The menu bar includes "File", "Query", "Git", "Tools", "Extensions", "Window", "Help", and "Search - Solution1". The toolbar has icons for "New Query", "Execute", and "Copilot". The left sidebar shows database connections and objects: "7.0.600.9 - sqladmin", "AdventureWorksOnPrem", "AdventureWorks", "AdventureWorksLT", "AdventureWorksLTReport", and "AdventureworksDW". The main area has tabs: "create_e...in (51)", "embeddi...in (88)", "create_v...in (96)", "find_rel...in (108)", and "find_pro...n (110)". The "find_pro...n (110)" tab is active and contains the following T-SQL code:

```
USE [AdventureWorksOnPrem];
GO
EXEC find_relevant_products_vector_search
@prompt = N>Show me stuff for extreme outdoor sports',
@stock = 100,
@top = 20;
GO
```

The results grid shows 20 rows of products:

	ProductName	ProductDescription	StockLevel
1	Mountain-200 Silver, 38	Serious back-country riding. Perfect for all levels ...	100
2	Mountain-200 Silver, 42	Serious back-country riding. Perfect for all levels ...	100
3	Road-350-W Yellow, 40	Tout terrain, course ou promenade entre amis su...	100
4	Road-350-W Yellow, 42	Tout terrain, course ou promenade entre amis su...	100
5	Road-350-W Yellow, 44	Tout terrain, course ou promenade entre amis su...	100
6	Road-350-W Yellow, 48	Tout terrain, course ou promenade entre amis su...	100
7	Mountain-300 Black, 38	For true trail addicts. An extremely durable bike t...	100
8	Mountain-300 Black, 40	For true trail addicts. An extremely durable bike t...	100
9	Mountain-300 Black, 44	For true trail addicts. An extremely durable bike t...	100
10	Mountain-300 Black, 48	For true trail addicts. An extremely durable bike t...	100
11	Road-150 Red, 62	จักรยานคุณภาพของนักแข่งระดับแชมป์เปี้ยน ได้รับการพัฒนาโดย...	100
12	Road-150 Red, 44	จักรยานคุณภาพของนักแข่งระดับแชมป์เปี้ยน ได้รับการพัฒนาโดย...	100
13	Road-150 Red, 48	จักรยานคุณภาพของนักแข่งระดับแชมป์เปี้ยน ได้รับการพัฒนาโดย...	100
14	Road-150 Red, 52	จักรยานคุณภาพของนักแข่งระดับแชมป์เปี้ยน ได้รับการพัฒนาโดย...	100
15	Road-150 Red, 56	จักรยานคุณภาพของนักแข่งระดับแชมป์เปี้ยน ได้รับการพัฒนาโดย...	100

At the bottom, it says "executed successfully." and "00:00:01 20 rows".

TO CHINESE LANGUAGE AND EXECUTE THIS PROCEDURE, I'M GOING

The screenshot shows a SQL Server Management Studio (SSMS) interface. On the left, the Object Explorer pane displays a connection to 'SQL Server 17.0.600.9 - sqladmin' with various database and security options. The main area contains a query window with the following T-SQL code:

```
USE [AdventureWorks];
GO

EXEC find_relevant_products_vector_search
@prompt = N'Show me stuff for extreme outdoor sports',
@stock = 100,
@step = 20;
GO

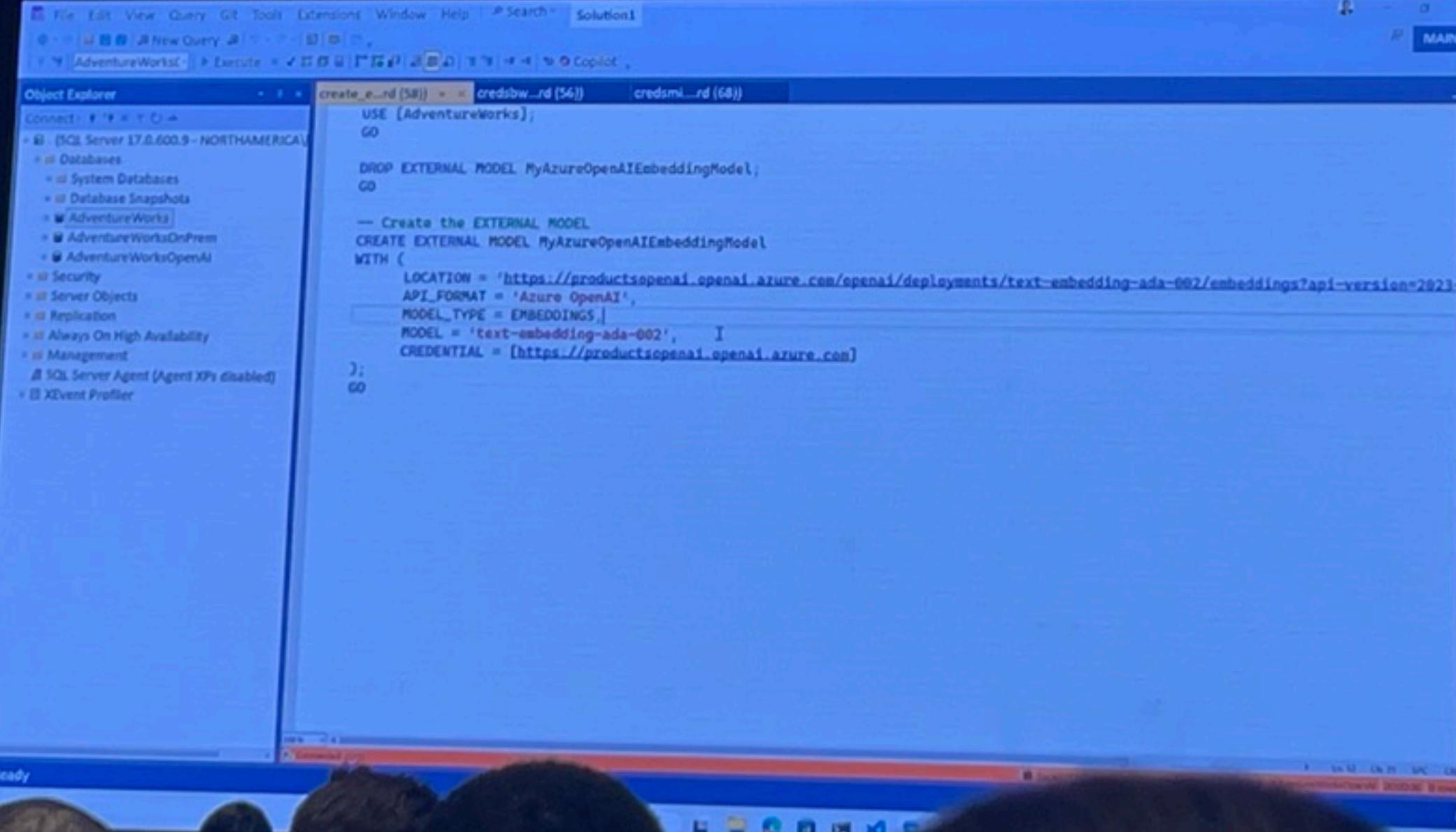
-- Do the same prompt but in Chinese
EXEC find_relevant_products_vector_search
@prompt = N'请向我显示极限户外运动的装备',
@stock = 100,
@step = 20;
GO
```

Below the code, the Results pane displays a table of products:

	ProductName	ProductDescription	StockLevel
1	Mountain-300 Black, 38	适用于真正的越野车迷。此自行车极其耐用，无论身处何地，地形如何复杂，一切均在掌控之中。真正物超所值！	100
2	Mountain-300 Black, 40	适用于真正的越野车迷。此自行车极其耐用，无论身处何地，地形如何复杂，一切均在掌控之中。真正物超所值！	100
3	Road-450 Red, 58	真正的多项运动自行车，骑乘自如，设计新颖，符合空气动力学的设计给您带来专业车手的体验。极佳的传动装...	100
4	Road-450 Red, 60	真正的多项运动自行车，骑乘自如，设计新颖，符合空气动力学的设计给您带来专业车手的体验。极佳的传动装...	100
5	Mountain-200 Silver, 38	适用于环境恶劣的野外骑行，可应对各种比赛的完美赛车。使用与 Mountain-100 相同的 HL 车架。	100
6	Mountain-200 Silver, 42	适用于环境恶劣的野外骑行，可应对各种比赛的完美赛车。使用与 Mountain-100 相同的 HL 车架。	100
7	Mountain-400-W Silver, 38	此自行车具有优越的性价比，它灵敏且易于操控。越野骑乘也可轻松胜任。	100
8	Mountain-400-W Silver, 40	此自行车具有优越的性价比，它灵敏且易于操控。越野骑乘也可轻松胜任。	100
9	Mountain-400-W Silver, 46	此自行车具有优越的性价比，它灵敏且易于操控。越野骑乘也可轻松胜任。	100
10	HL Mountain Front Wheel	高性能的山地车备用轮。	500
11	HL Mountain Rear Wheel	高性能的山地车备用轮。	500
12	ML Crankset	高精度的曲臂。	500
13	Road-750 Black, 58	入门级成人自行车 确保越野旅行或公路骑乘的舒适，快拆式车轮和轮胎。	100
14	Road-750 Black, 44	入门级成人自行车 确保越野旅行或公路骑乘的舒适，快拆式车轮和轮胎。	100
15	Road-750 Black, 48	入门级成人自行车 确保越野旅行或公路骑乘的舒适，快拆式车轮和轮胎。	100



AZURE RESOURCES. SO HOW DO YOU TYPICALLY DO IT TODAY? LET'S START



```
USE [AdventureWorks];
GO

DROP EXTERNAL MODEL MyAzureOpenAIEmbeddingModel;
GO

-- Create the EXTERNAL MODEL
CREATE EXTERNAL MODEL MyAzureOpenAIEmbeddingModel
WITH (
    LOCATION = 'https://productsopenai.openai.azure.com/openai/deployments/text-embedding-ada-002/embeddings?api-version=2023-05-15',
    API_FORMAT = 'Azure OpenAI',
    MODEL_TYPE = EMBEDDINGS,
    MODEL = 'text-embedding-ada-002',
    CREDENTIAL = [https://productsopenai.openai.azure.com]
);
GO
```



AZURE THAT USES MANAGED IDENTITIES. A LOT MORE SECURE. HOW DO YOU SET

The screenshot shows the Microsoft Azure portal interface. At the top, there's a dark banner with the text "AZURE THAT USES MANAGED IDENTITIES. A LOT MORE SECURE. HOW DO YOU SET". Below this, the Azure navigation bar includes "Microsoft Azure (Preview)", "Report a bug", a search bar ("Search resources, services, and docs (G+/)", "Copilot"), and a user profile ("bobward@microsoft.com", "MICROSOFT (MICROSOFTONMICROSOFT.COM)"). The main content area displays a resource named "productdata | Microsoft Entra ID and Purview" under "SQL Server - Azure Arc". The left sidebar lists several sections: Overview, Activity log, Access control (IAM), Diagnose and solve problems, Resource visualizer, Settings (which is currently selected), Microsoft Entra ID and Purview (under Settings), Best practices assessment, Updates, Properties, Security (which is expanded to show Data management, Monitoring, Migration, Automation, and Help). A message in the center states "Azure Active Directory (Azure AD) is now Microsoft Entra ID. Learn more".

MODEL CONTEXT PROTOCOL, SERVER. GIVE

```
8
9     [McpServerTool, Description("")]
10    Query the database to find claims data for customers.
11    Columns:
12        [Key] id (int): internal claim id
13        [ForeignKey: to customers.id] customer_id (int): customer id
14        claim_type (string): claim type (auto, home, life, health, e)
15        claim_date (DateTime): claim date
16        details (string): details and notes about the claim added by
17        "")]
18    public static Claim[] GetClaims(
19        [Description("OData $filter string, e.g., claim_type eq 'a")]
20        CancellationToken? cancellationToken = null) =>
21        GetFiltered<Claim>("Claim", filter, cancellationToken);
22
23    [McpServerTool, Description("Upsert a Claim record; returns up")]
24    public static Claim UpsertClaim(
25        [Description("Claim ID (int)")] int id,
26        [Description("Customer ID (int)")] int customer_id,
27        [Description("Claim type (nvarchar(100)))")] string claim_type,
28        [Description("Claim date (datetime)")] DateTime c]
```

THESE CAPABILITIES AND MADE MINIMUM
APPLICATIONS CODE CHANGES, USE THE



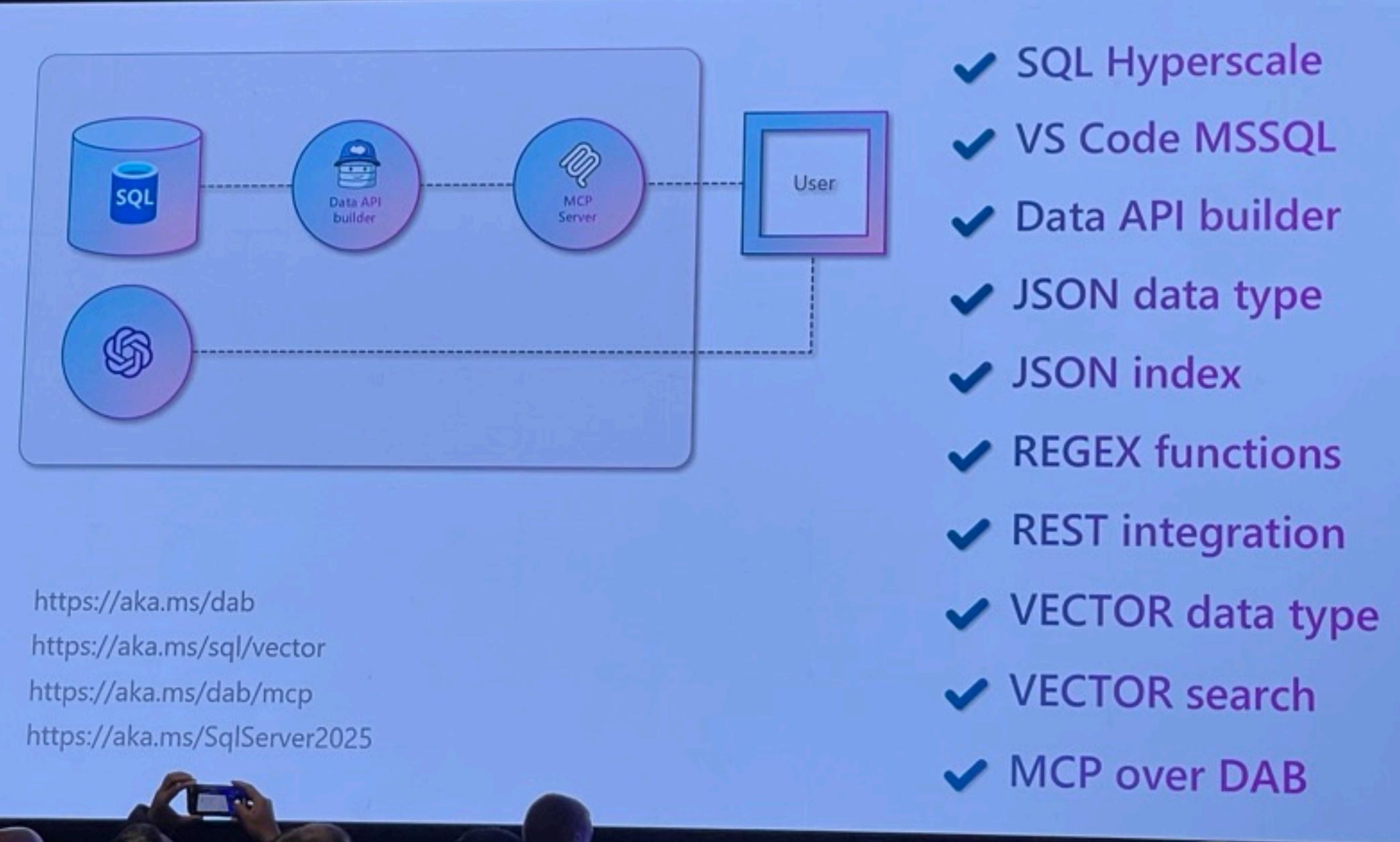
- ✓ SQL Hyperscale
- ✓ VS Code MSSQL
- ✓ Data API builder
- ✓ JSON data type
- ✓ JSON index
- ✓ REGEX functions
- ✓ REST integration
- ✓ VECTOR data type
- ✓ VECTOR search

<https://aka.ms/dab>
<https://aka.ms/sql/vector>
<https://aka.ms/dab/mcp>
<https://aka.ms/SqlServer2025>

MCP over DAB



DATABASE AI AND DEVELOPER EXPERIENCES TO BUILD THIS



TO HELP YOU GATHER MORE KNOWLEDGE
ABOUT ENTITIES. RANKING WILL HELP

Semantic Operators

azure_ai extension

generate()

Perform arbitrary ChatGPT-style generation

is_true()

Evaluate filter conditions and joins in natural language

extact()

Extract semantic relationships hidden in your data

rank()

Improve search accuracy with semantic reranking

```
1
2 SELECT azure_ai.generate('Summarize
  this feedback: ' || feedback_text)
3 FROM customer_feedback
4 WHERE created_at >
  now() - interval'7 days';
```



Microsoft

YOU ASKED AS WELL. COPILOT'S REALLY PHENOMENAL

The screenshot shows a PostgreSQL client interface with a dark theme. On the left, there's a sidebar with icons for connections, databases, schemas, tables, and views. The main area shows a query editor with a file named '1_pg_query.sql' containing the following SQL code:

```
4.248.28.54 | build_2025
SELECT
    r1.id,
    r1.product_id,
    r1.user_name,
    r1.rating,
    r1.review_text,
    r1.created_at
FROM public.reviews AS r1
JOIN public.reviews AS r2
    ON r1.product_id = r2.product_id
WHERE
    md5(lower(r1.review_text)) = md5(lower(r1.rev
    AND lower(r1.review_text) LIKE '%' || lower('
    AND extract(year FROM r1.created_at) = extract
ORDER BY
    random(),
    length(r1.review_text) DESC;
```

To the right of the query editor is a 'CHAT' panel displaying an AI-generated explanation of the query's execution plan:

- Seq-scan of reviews (r1) with three per-row expressions
 - md5(lower(review_text)) = ... is a no-op (always true)
 - extract(year FROM created_at) = ... is a no-op
 - lower(review_text) LIKE '%great%' forces a full scan
- Seq-scan of reviews again (r2) to build a hash table on product_id.
- A hash-join between r1 and r2 on product_id. This "self-join" multiplies rows for every other review of the same product, yielding ~7 482 rows.
- A full in-memory sort by random() and length(review_text)::
 - random() can't use any index—PostgreSQL must pull all rows into RAM and quicksort them.

Total times:
• Planning: ~0.6 ms
• Execution: ~38 ms

If your reviews table grows, this will degrade quickly.

— Recommendations —

- Remove redundant predicates and the self-join if you only care about filtering r1.

At the bottom, there are buttons for 'Add Context...' and '1_pg_query.sql Current file'. A status bar at the bottom right says '@pgsql Helps you query, explore, and interact with PostgreSQL'.



IT AGAIN, WE'LL GET THE SAME SET OF STEPS EXPLAINING THE

The screenshot shows a PostgreSQL client interface with a dark theme. On the left, there's a sidebar with icons for connections, databases, schemas, tables, and query history. The main area has tabs for 'Untitled-1' (containing a SQL query), 'PROBLEMS', and 'POSTGRES QUERY RESULTS'. The 'POSTGRES QUERY RESULTS' tab is active, displaying a table with columns: id, product_id, user_name, rating, and review_text. The table contains 10 rows of sample data. To the right of the results, a 'CHAT' panel is open, showing a connection status message and a detailed explanation of the query's execution plan. A red box highlights the 'Execution ~8.5 ms.' part of the plan summary. Below the plan, there's a section for 'Bottlenecks & Recommendations' with one item listed.

4.248.28.54 | build_2025

```
1  SELECT id, product_id, user_name, rating, review_text
2   FROM public.reviews
3   WHERE lower(review_text) LIKE '%great%'
4     AND product_id IN (
5       SELECT product_id
6         FROM public.reviews
7         GROUP BY product_id
8         HAVING COUNT(*) > 1
9     )
```

ORDER BY random() length(review_text) DESC

	id	product_id	user_name	rating	review_text
1	1109	47	Michael Nelson	5	Great product!
2	579	25	Megan Brown	3	Not great.
3	947	40	Brian Hall	4	Good product.
4	586	25	Brian Hunter	4	Great product!
5	1162	50	Sarah Johnson	5	Great product!
6	369	16	Linda Martinez	4	Good product.
7	1256	53	Sofia Baker	4	Great product!
8	876	37	Ursula Scott	4	Good product.
9	1246	53	Jack Lopez	4	Great product!

Connected to PMVM/build_2025 (read/write)

Here's what the EXPLAIN ANALYZE shows for the simplified query:

- Two sequential scans
 - One to build a hash-aggregate of `product_id` with `COUNT()>1` (1.2 ms, 1 327 rows)
 - One to filter reviews by `lower(review_text) LIKE '%great%'` (6.3 ms, 316 rows)
- A hash-join on `product_id` (joining the 316 filtered rows to the ~56 "duplicate" `product_ids`) finishes in ~7.9 ms.
- A quicksort of the 316 result-rows by `random()` and `length(review_text)` takes <0.1 ms.

Total: Planning ~0.8 ms, Execution ~8.5 ms.

Bottlenecks & Recommendations

- Full table scans for both the LIKE filter and the grouping

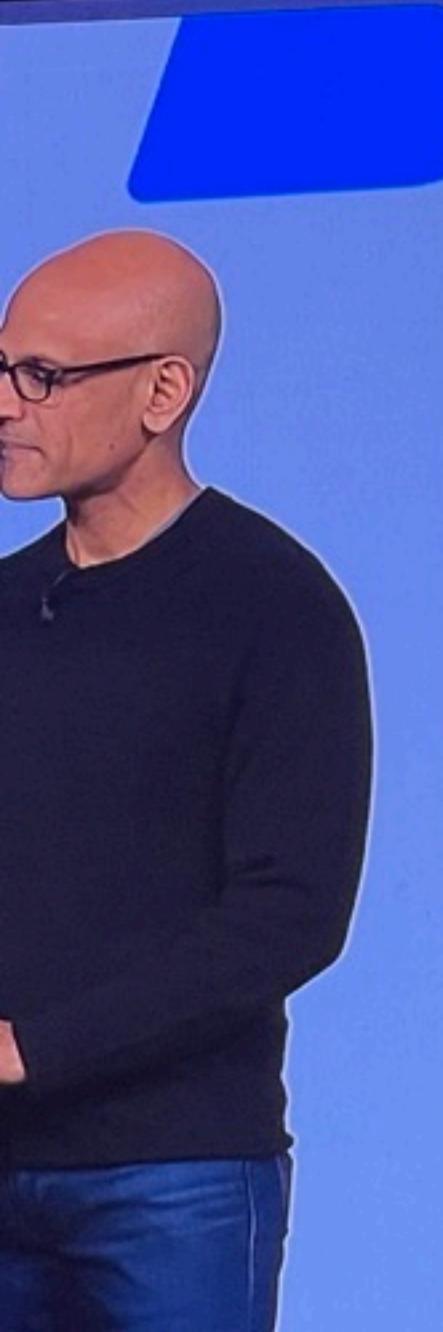
Add Context... 1_pg_query.sql Current file

@pgsql Helps you query, explore, and interact with PostgreSQL





head over to VS code.
I'm going to introduce you to SE



A screenshot of a Microsoft Visual Studio Code interface running on a large screen. The interface shows a PostgreSQL connection named 'semantic' with a query editor containing the following SQL code:

```
1 SELECT name,
2        azure_ai.generate('Generate description for the event: ' || name)
3 FROM events
4 LIMIT 5;
```

The results tab displays five rows of generated descriptions for various events:

name	generate
GenAI Summit 2025	The GenAI Summit 2025 is a premier event bringing together industry leaders...
DataFest West	DataFest West is an exciting event focused on data science, analytics, and ...
Cloud Innovators Day	Cloud Innovators Day is a dynamic event dedicated to exploring the latest a...
TechFusion Expo	TechFusion Expo is a premier technology event that brings together industry...
AI & Cloud Convergence	The AI & Cloud Convergence event explores the integration of artificial int...

TO CLOSE OFF AND ACTUALLY HOPE YOU
WILL ALL COME AT

Thank you



<https://aka.ms/BuildOpenHackResources>



WHERE YOU HAVE MANY AGENTS. A MICROSOFT

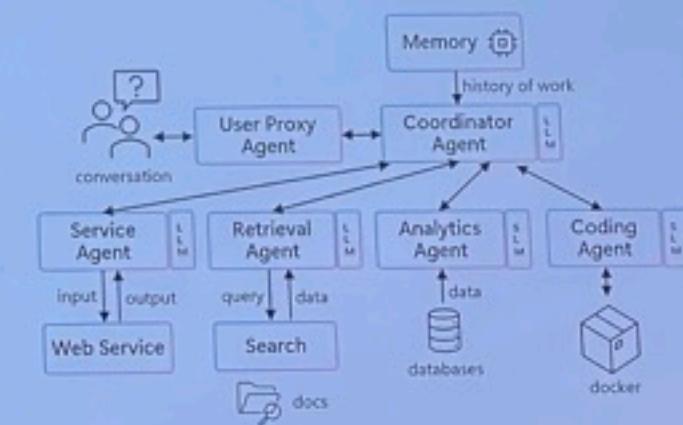
Spectrum of Agentic Solutions



No Tools Agent
Narrow one-shot task
Ex: summarize text



Single Agent
Clearly scoped iterative task
Ex: providing an answer with supporting evidence to a complex question



Multi-agent Systems
Wide scope complex use case requiring diverse skills
Ex: Propose 2 Instagram marketing campaigns including assets that would leverage the top 2 recent trends in our past quarter US Sales to boost our mailing list user base and predict the impact of each campaign



THE FOUNDRY. WHICH ALLOW YOU TO
USING YOUR MODEL ANDS

Microsoft has different ways of building agent systems



IaaS

Infrastructure-as-a-service



PaaS

Platform-as-a-service



SaaS

Software-as-a-service



Semantic Kernel
Agent framework



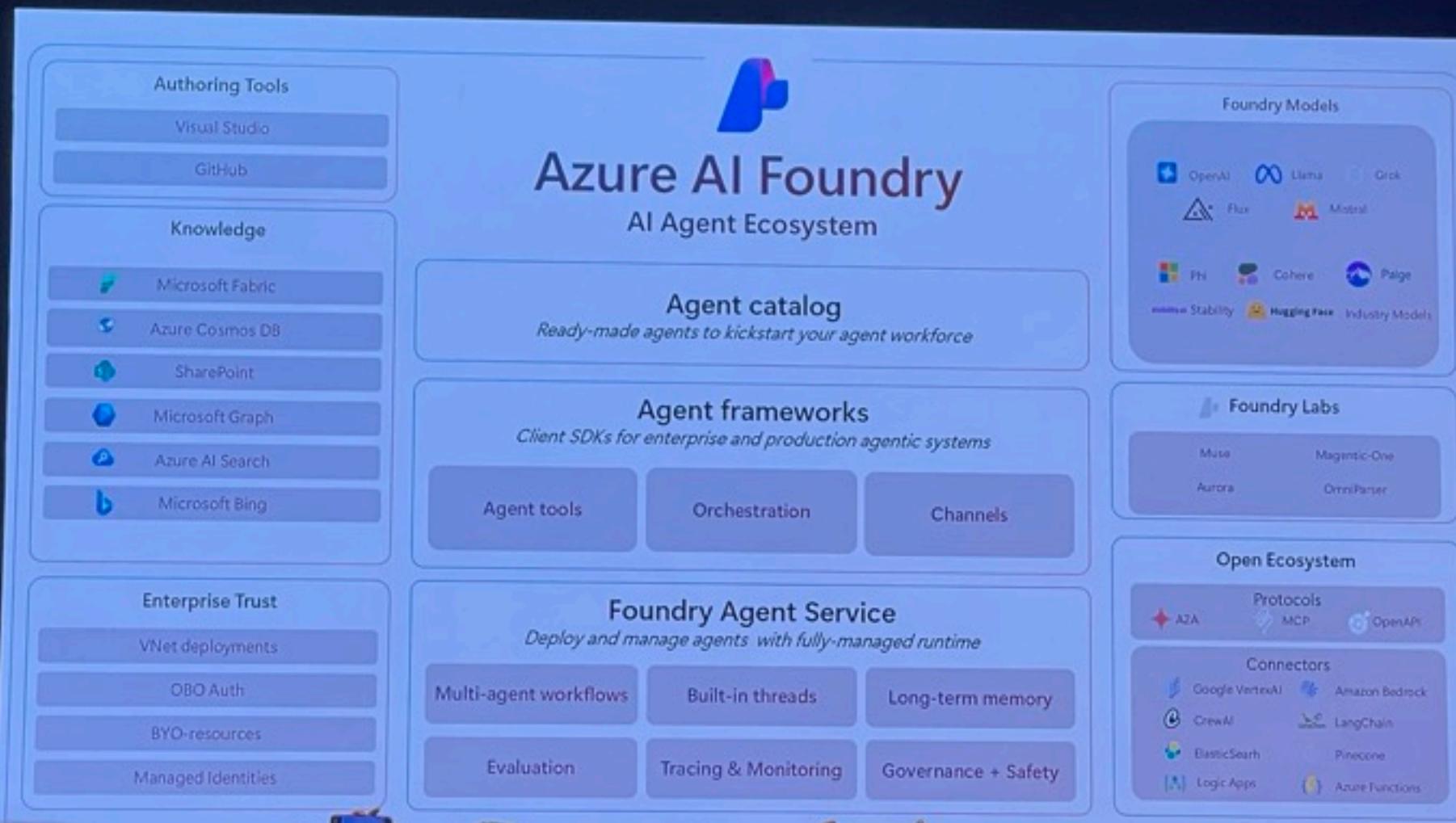
Azure AI Foundry
Agent service



Copilot Studio
Agents

Control, visibility, and customization

USING YOUR MODEL ANDS YOUR VECTOR STORES AND TOOLS IN A PRO- CCODE



PIECE I'M USING THE PROCESS FRAMEWORK. I'M USING

```
3 AssistantClient client = new Agent();
4
5 // create the single agents
6 var teacherAgent = await client.CreateAssistantAsync("gpt-4o", new()
7 {
8     Description = "A math teacher assistant",
9     Name = "Teacher",
10    Instructions = "You are a teacher that create pre-school math question for stu
11 });
12 Console.WriteLine($"Creating agent {teacherAgent.Value.Name} ({teacherAgent.Value.
13
14 var studentAgent = await client.CreateAssistantAsync("gpt-4o", new()
15 {
16     Description = "A student assistant",
17     Name = "Student",
18     Instructions = "You are a student that answer question from teacher, when teac
19 });
20
21 Console.WriteLine($"Creating agent {studentAgent.Value.Name} ({studentAgent.Value.
22
```

HANSELMAN'S TALK YESTERDAY HE USED THE SAME SDK WHERE HE AND

```
File Edit Selection View Go Run Terminal Help workflow [Administrator]
EXPLORER
WORKFLOW
Extensions
Agents.cs
ClientOptions.cs
Ext.cs
ModuleInitializer.cs
Process
Workflows.cs
.env
.gitignore
Makefile
Program.cs
README.md
workflow.csproj
workflow.sln

Process > Workflows.cs > Workflows > Build
public class TwoAgentMathState
{
    public List<ChatMessageContent>? TeacherMessages { get; set; }

    1 reference
    public static class Workflows
    {
        1 reference
        public static FoundryProcessBuilder<T> Build<T>(Assistant studentAgent, Assistant teacherAgent) where T : class, new()
        {
            var studentDefinition = new AgentDefinition { Id = studentAgent.Id, Name = studentAgent.Name, Type = AzureAIAGentFactor };
            var teacherDefinition = new AgentDefinition { Id = teacherAgent.Id, Name = teacherAgent.Name, Type = AzureAIAGentFactor };

            // Define the process with a state type
            var processBuilder = new FoundryProcessBuilder<T>("two_agent_math_chat");

            // Create a thread for the student
            processBuilder.AddThread("Student", KernelProcessThreadLifetime.Scoped);
            processBuilder.AddThread("Teacher", KernelProcessThreadLifetime.Scoped);

            // Add the student
            var student = processBuilder.AddStepFromAgent(studentDefinition);

            // Add the teacher
            var teacher = processBuilder.AddStepFromAgent(teacherDefinition);

            //***** Orchestrate *****/
            // When the process starts, activate the student agent
            processBuilder.OnProcessEnter().SendEventTo(
                student,
                thread: "_variables_.Student",
                messagesIn: ["_variables_.TeacherMessages"],
                inputs: new Dictionary<string, string> { });
        }
    }
}

public class Program
{
    public static void Main()
    {
        var workflow = Workflows.Build<TwoAgentMathState>(new Assistant(), new Assistant());
    }
}
```

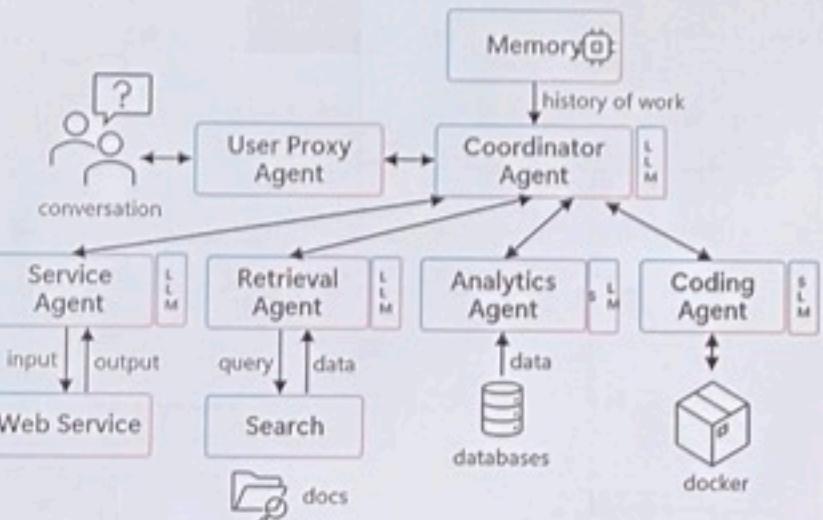
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL FOCUS

PS C:\Users\salmanq\code\build25\workflow>

Salman Quazi Q:\Build25\Workflow 1n 23, Col 57 Spaces:4 UTF-8 CRLF 3:26 PM 12/20/2025

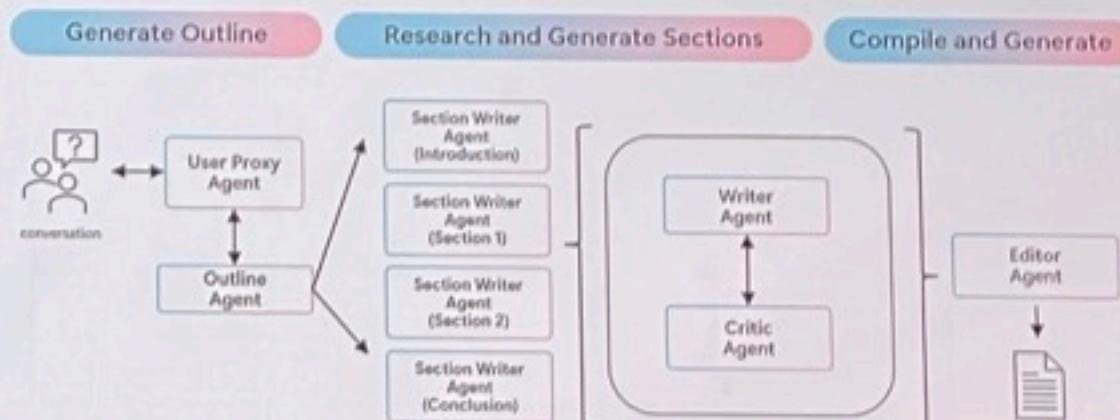
THE SAME CAPABILITIES THAT SALMAN TALKED ABOUT.

Agent systems with Semantic Kernel



Agent Framework
LLM-driven orchestration
Creative reasoning and decision-making

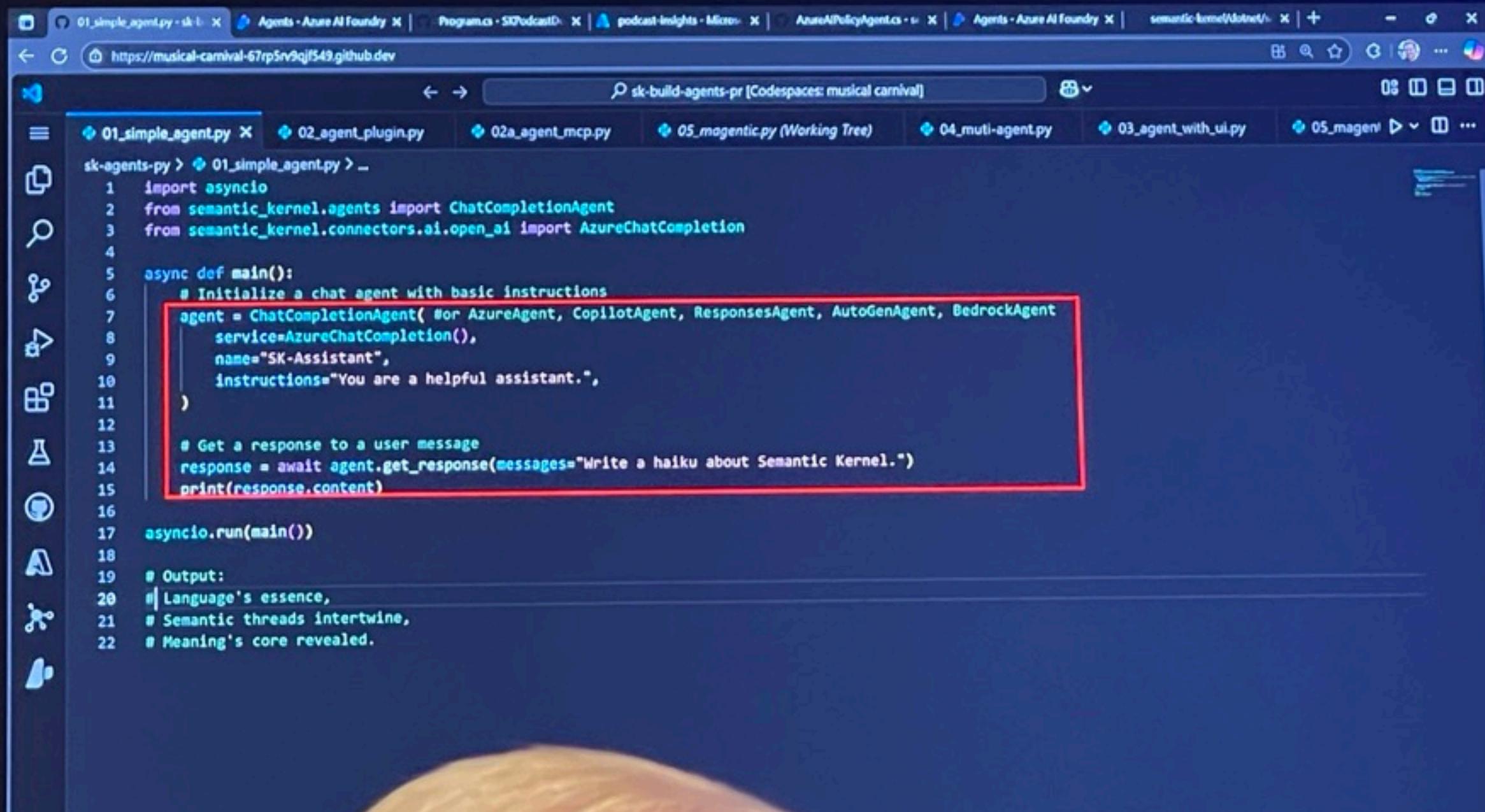
Ex: Propose 2 Instagram marketing campaigns including assets that would leverage the top 2 recent trends in our past quarter US Sales to our mailing list user base and predict the impact of the campaign



Process Framework
Workflow-driven orchestration
Deterministic and predictable execution pattern

Ex: Execution the Deep Research process to generate a document outline, write and refine section and synthesize into final document

WE CAN WE CREATE OUR AGENT. WE CONNECT IT TO



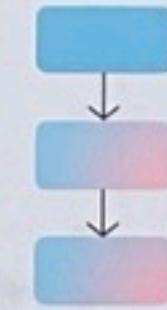
The screenshot shows a terminal window with a dark theme. The title bar reads "sk-build-agents-pr [Codespaces: musical carnival]". The window displays a Python script named "01_simple_agent.py". A red rectangular box highlights the following code block:

```
    1 import asyncio
    2 from semantic_kernel.agents import ChatCompletionAgent
    3 from semantic_kernel.connectors.ai.openai import AzureChatCompletion
    4
    5 async def main():
    6     # Initialize a chat agent with basic instructions
    7     agent = ChatCompletionAgent(#or AzureAgent, CopilotAgent, ResponsesAgent, AutoGenAgent, BedrockAgent
    8         service=AzureChatCompletion(),
    9         name="SK-Assistant",
   10         instructions="You are a helpful assistant.",
   11     )
   12
   13     # Get a response to a user message
   14     response = await agent.get_response(messages="Write a haiku about Semantic Kernel.")
   15     print(response.content)
   16
   17 asyncio.run(main())
   18
   19 # Output:
   20 # Language's essence,
   21 # Semantic threads intertwine,
   22 # Meaning's core revealed.
```

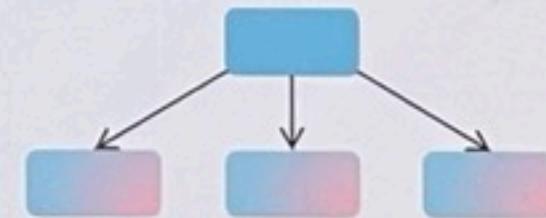
SEQUENCE. CONCURRENT. ALL OF THE AGENTS WORKING IN PARALLEL

Multi-Agent Orchestration Patterns

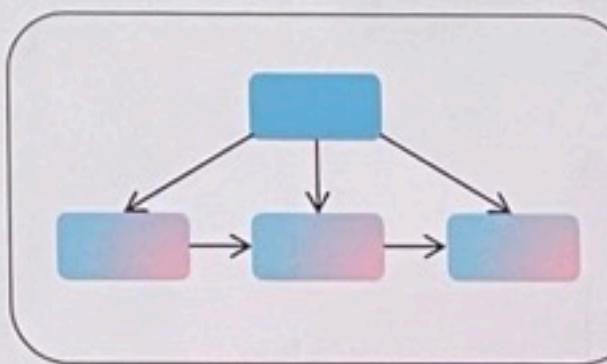
Sequential



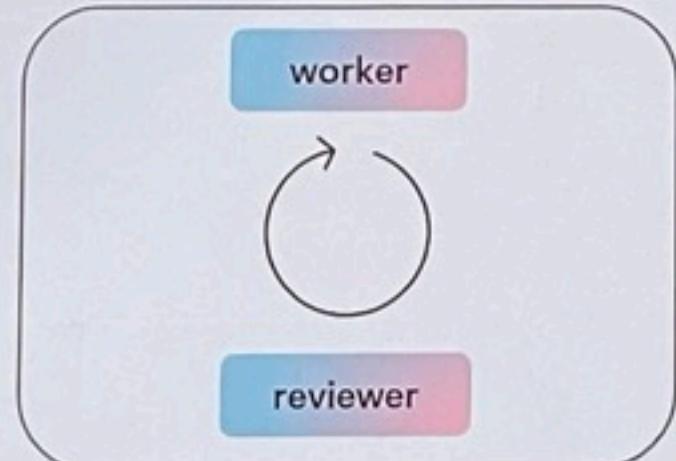
Concurrent



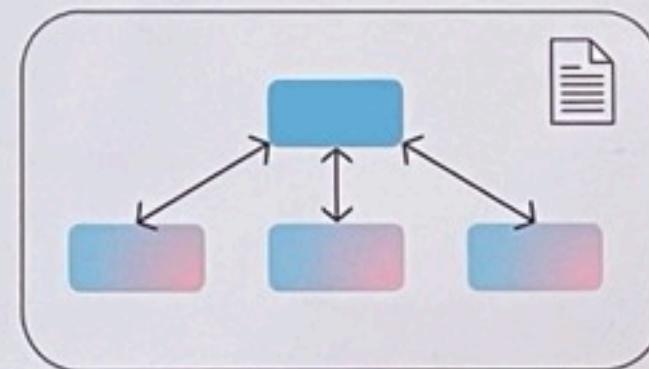
Handoff



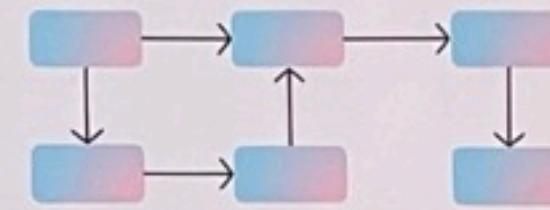
Group Chat



Magnetic



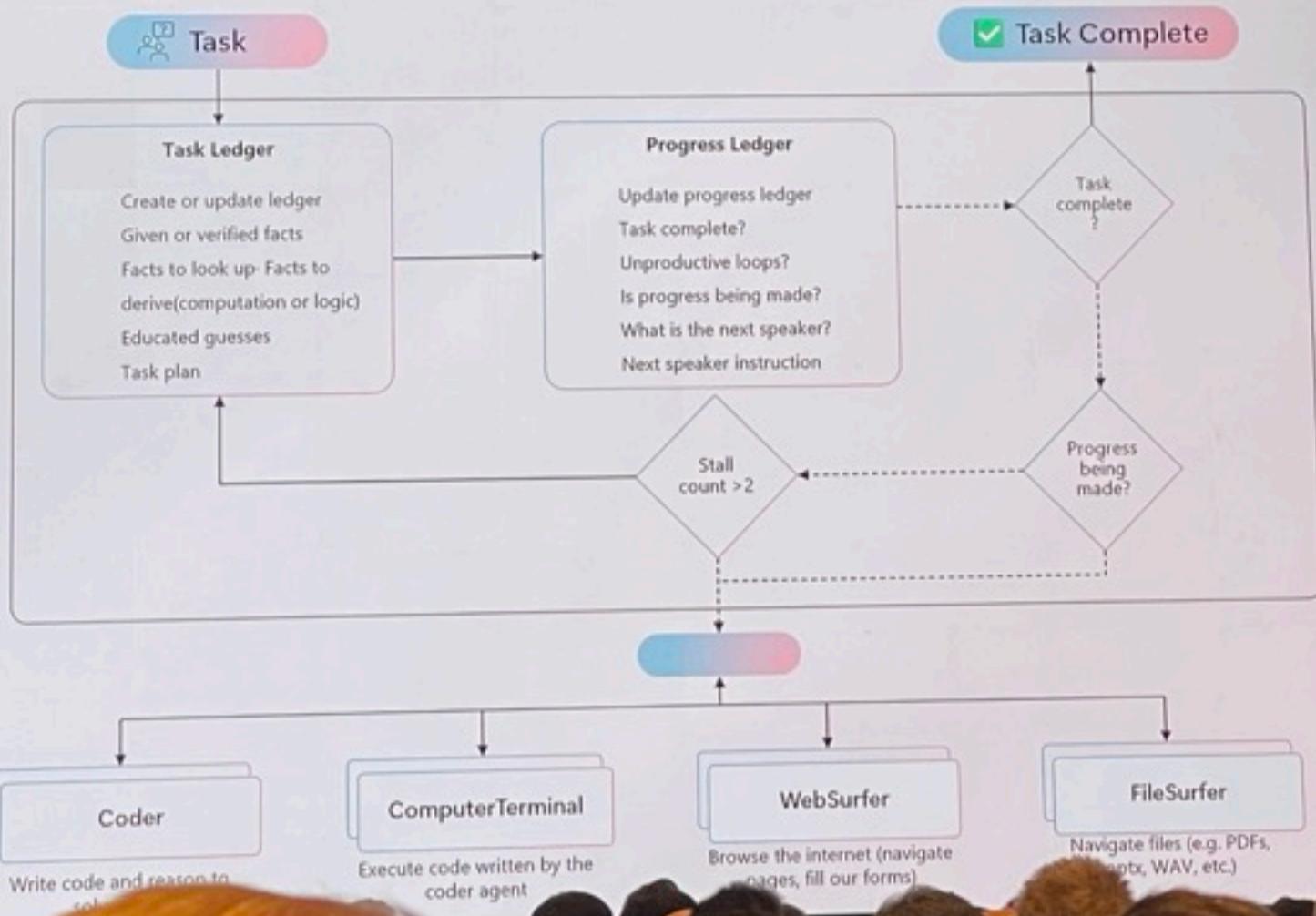
Workflow Process



TASK LEDGER AND CHECKS IF THE AGENTS
IS MAKING PROGRESS TOWARD THAT GOAL.

Magentic One

<https://aka.ms/magnetic-one>



The screenshot shows a Microsoft Visual Studio Code (VS Code) interface with the following details:

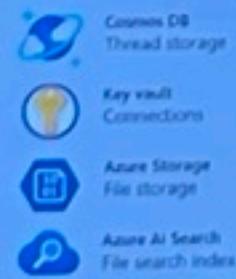
- Title Bar:** Shows multiple tabs including "03_agent_with_uipy", "Assistant", "Agents - Azure AI For", "Program.cs - SKPodcast", "podcast-insights - M", "AzureAI PolicyAgent", "Agents - Azure AI For", and "semantic-kernel/dot".
- Header Bar:** Shows the URL "https://upgraded-fiesta-rvx6r4qqg2p9q7.github.dev".
- Left Sidebar (Explorer):** Shows the project structure for "SKPOCASTDOTNET [CODESPACE]". It includes files like ".vscode", "Agents", "ContentAgent.cs", "LinkVerificationAgent.cs", "SummaryAgent.cs", "VoiceAgents.cs", "bin", "Filters", "Helpers", "Models", "obj", "output", "Plugins", "Services", ".env", ".gitignore", "881.txt", "podcast.process.yaml", and "Program.cs". "Program.cs" is currently selected.
- Central Editor Area:** Displays the content of "Program.cs".

```
21 class Program  
22     async static Task Main(string[] args)  
23     {  
24         process  
25             .OnInputEvent("Start")  
26             .SendEventTo(new ProcessFunctionTargetBuilder(getTranscriptStep));  
27         getTranscriptStep  
28             .OnFunctionResult()  
29             .SendEventTo(new ProcessFunctionTargetBuilder(contentAgentsStep));  
30         contentAgentsStep  
31             .OnFunctionResult()  
32             .SendEventTo(new ProcessFunctionTargetBuilder(linkVerificationAgentStep));  
33         linkVerificationAgentStep  
34             .OnFunctionResult()  
35             .SendEventTo(new ProcessFunctionTargetBuilder(summaryAgentsStep));  
36         summaryAgentsStep  
37             .OnFunctionResult()  
38             .StopProcess();  
39     }  
40     KernelProcess kernelProcess = process.Build();
```
- Right Panel (Preview):** Shows a preview of "podcast.process.yaml". It contains two sections:
 - ContentAgent:** Analyzes podcast transcript and generate comprehensive show notes.
 - LinkVerificationAgent:** Extracts and verifies links mentioned in podcast transcripts.
- Bottom Status Bar:** Shows "dotnet" and other status indicators.
- Bottom Taskbar:** Shows "PROBLEMS", "OUTPUT", "DEBUG CONSOLE", "TERMINAL", "PORTS", "AZURE", "POLYGLOT NOTEBOOK", "SPELL CHECKER", and "dotnet".
- Bottom Activity Bar:** Shows "03-mini", "Layout: US", and "Prettier".

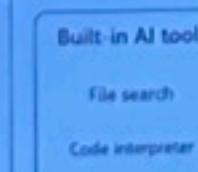
A DEMO FAIL. SO TO BRING IT ALL
TOGETHER WE SAW AZURE AZURE AI

Agent System with Azure AI Foundry

BYO resources



Azure AI Foundry Standard setup



Semantic Kernel Multi-agent orchestrator

Azure Container Apps



YOU CAN SELECT DIFFERENT AGENTS AND TOOLS. YOU CAN STORE INCAS MOW

Agent System with Azure AI Foundry

BYO resources

- Cosmos DB Thread storage
- Key vault Connections
- Azure Storage File storage
- Azure AI Search File search index

Azure AI Foundry Standard setup

- Built-in AI tools
- File search
- Code interpreter

Agents



Models



Bot Service
Channels

AI tool resources

- Azure AI Search
- Logic Apps
- Grounding with Bing Search
- Azure Functions

Semantic Kernel Multi-agent orchestrator

Azure Container Apps



Fabric

SharePoint

External APIs

MCP s

A2A Servers

TO YOUR APS. NET CORE APPS. TO DO
THAT WE'LL ADD A BUNCH OF AUTHENTICATION

Authentication scaffolding

- Based on `dotnet scaffold`
- Update existing ASP.NET Core & Blazor Identity scaffolders
- Add ASP.NET Core Identity Endpoints
- Add Entra ID authentication
- Add authentication to Blazor Hybrid and .NET MAUI apps

Pick a scaffolding command:

ASP.NET Core Identity (dotnet-scaffold-aspnet)
> Blazor Identity (dotnet-scaffold-aspnet)





Microsoft Learn

Thanks for joining us!

Share your thoughts at aka.ms/build/evals

Explore Learn On-demand!

Join us on Level 3 to take the Microsoft Build Challenge in-person, find more labs, earn a Microsoft Applied Skill, or even complete a practice assessment.

Meet us in the Hub!

Engage with experts, explore our new AI-enabled capabilities, and maybe even get celebrated! Visit us in SCC Arch, Level 4!

Level up your AI game!

Immerse yourself in the latest AI innovations and build your skills. Discover everything you can achieve with interactive training, credentials, and resources at: aka.ms/LearnAtBuild.

talk in terms of robotics, if you
take away all robotics and



work into the boxes and lines. At
the heart of our system is

