Building AI Applications from Scratch

Jeremy Likness, Principal Product Manager
Luis Quintanilla, Senior Product Manager

.NET & AI

Learn

Integrate

Collaborate

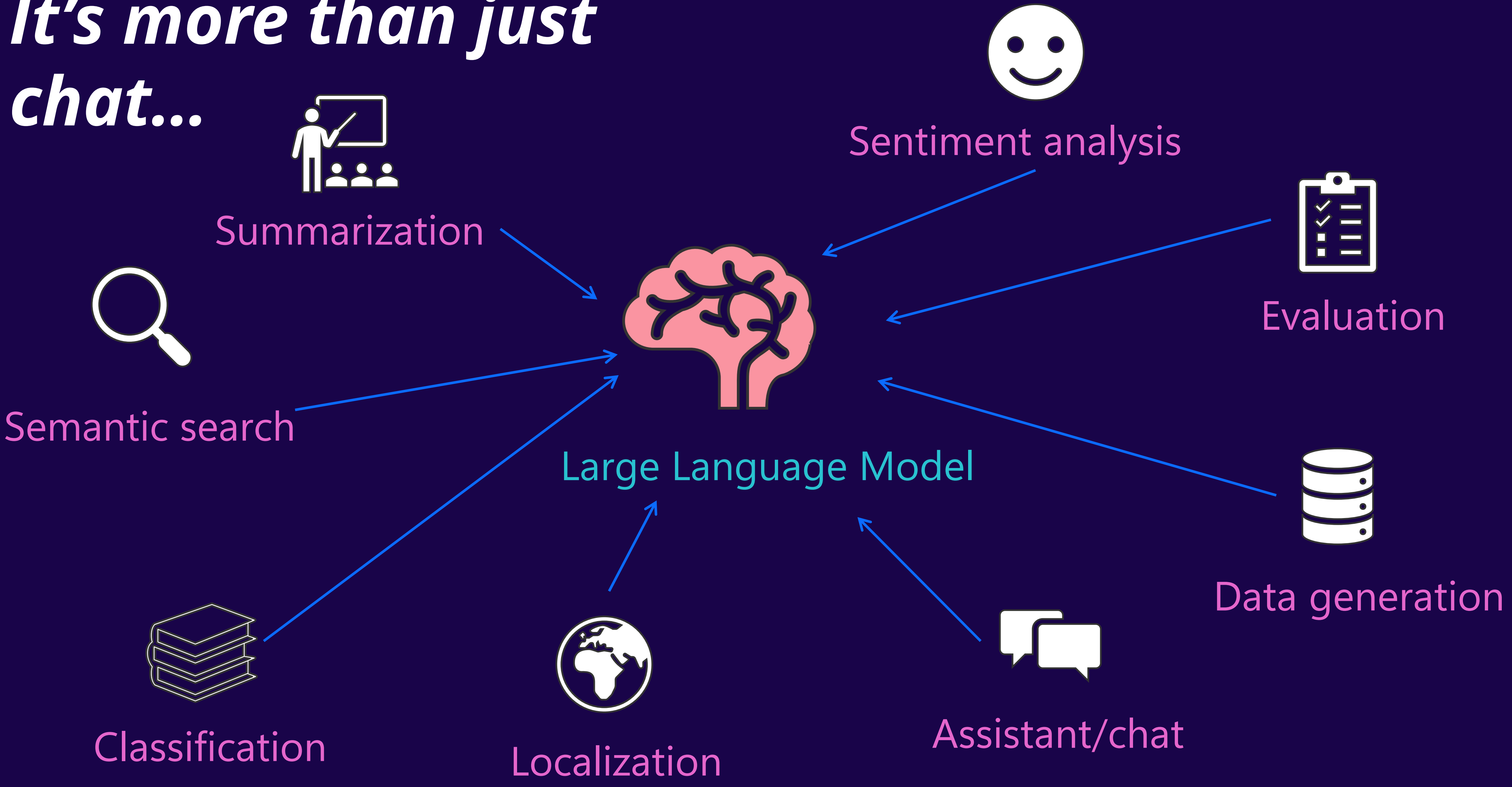
Deploy

.NET as the Industry shifts, we shift with you!

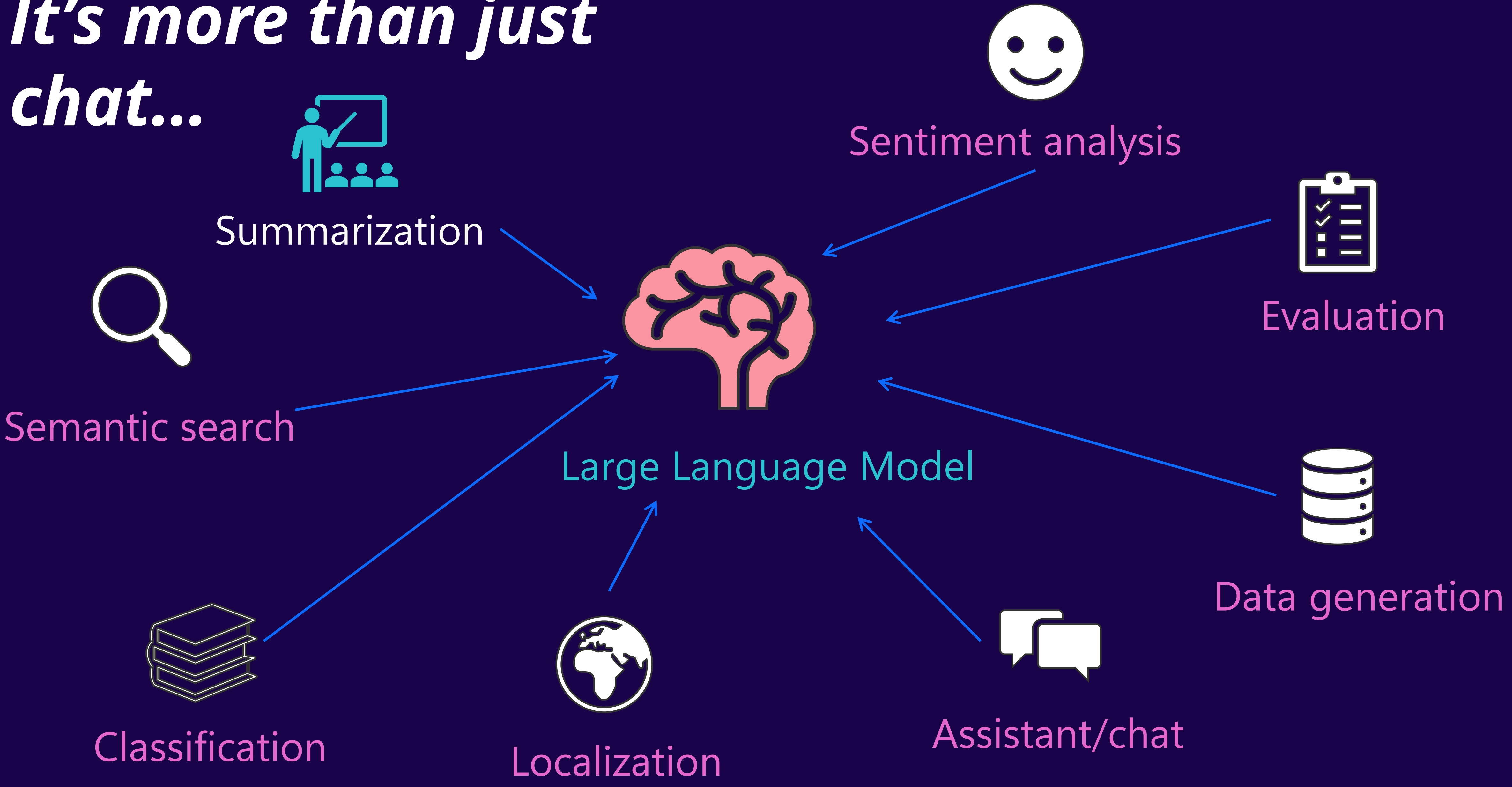
Demo:
“Imagine” (starting from scratch)

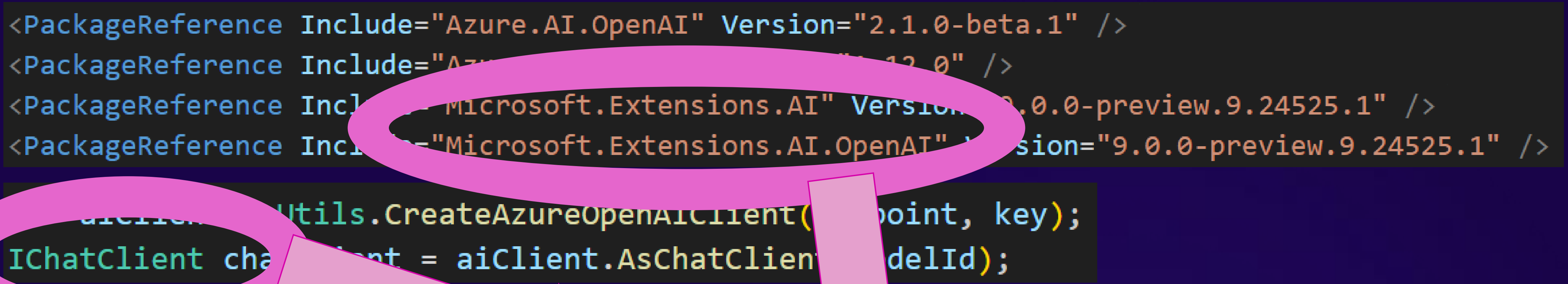
Jeremy Likness

It’s more than just chat…



It's more than just chat...





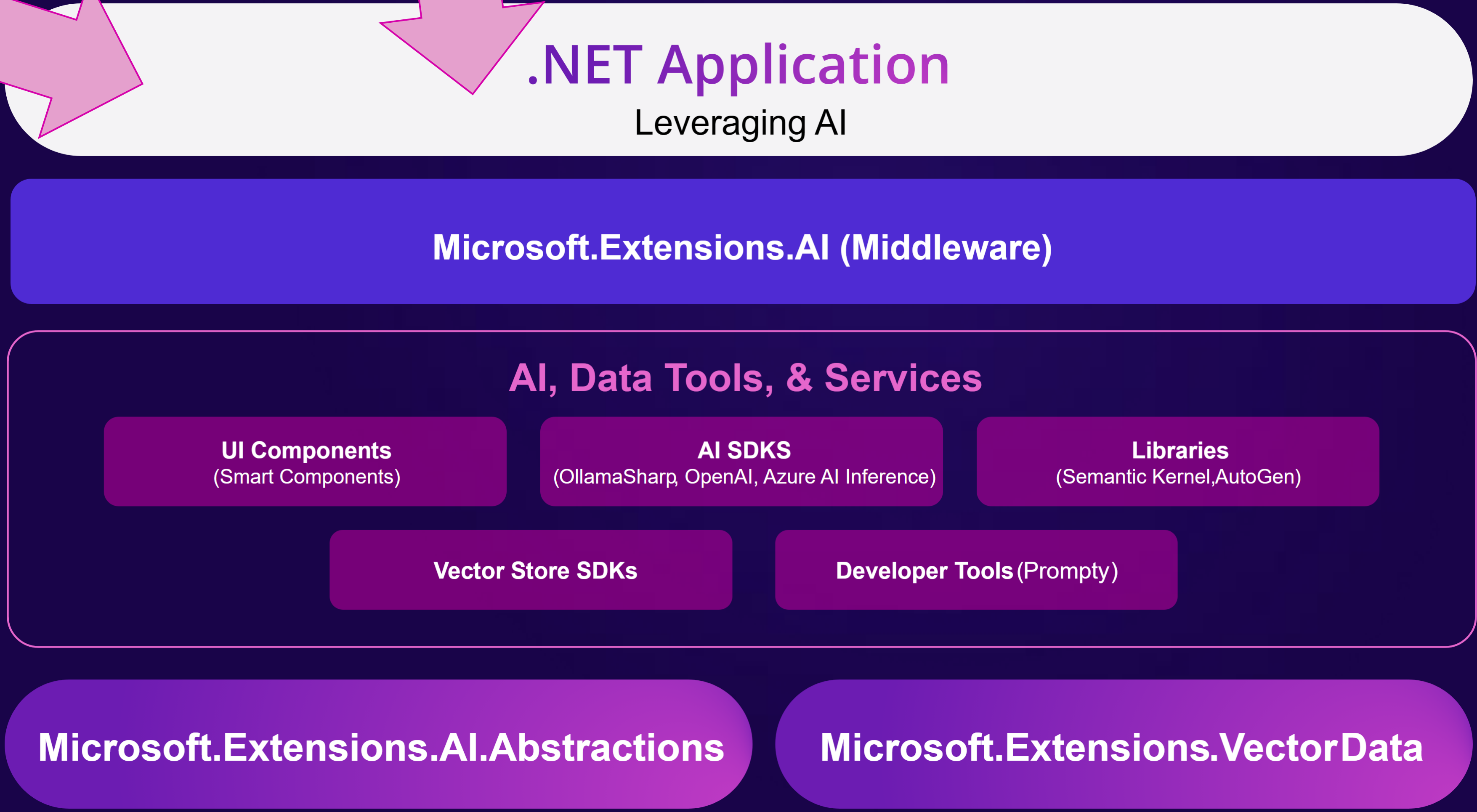
```
<PackageReference Include="Azure.AI.OpenAI" Version="2.1.0-beta.1" />
<PackageReference Include="Az                              1.12.0" />
<PackageReference Incl    Microsoft.Extensions.AI" Versio   .0.0-preview.9.24525.1" />
<PackageReference Incl   ="Microsoft.Extensions.AI.OpenAI"   sion="9.0.0-preview.9.24525.1" />
```

```
aiClien      Utils.CreateAzureOpenAIClient(    point, key);
IChatClient cha    ent = aiClient.AsChatClien    delId);
```

Common Building Blocks
- Chat
- Streaming
- Embeddings

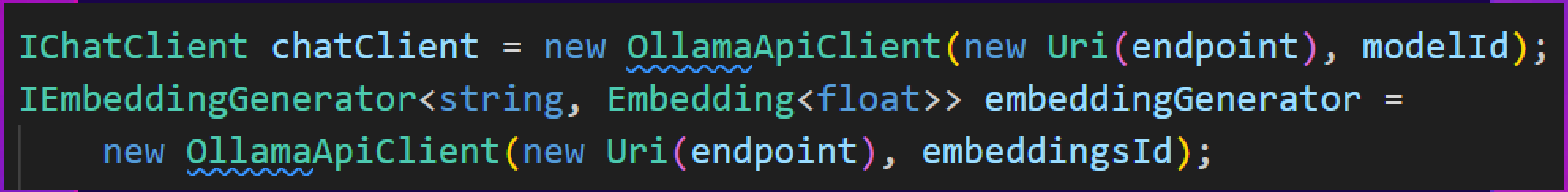
Middleware
- Cache
- Telemetry
- Filter

Integrations
- OpenAI
- GitHub Models
- Ollama



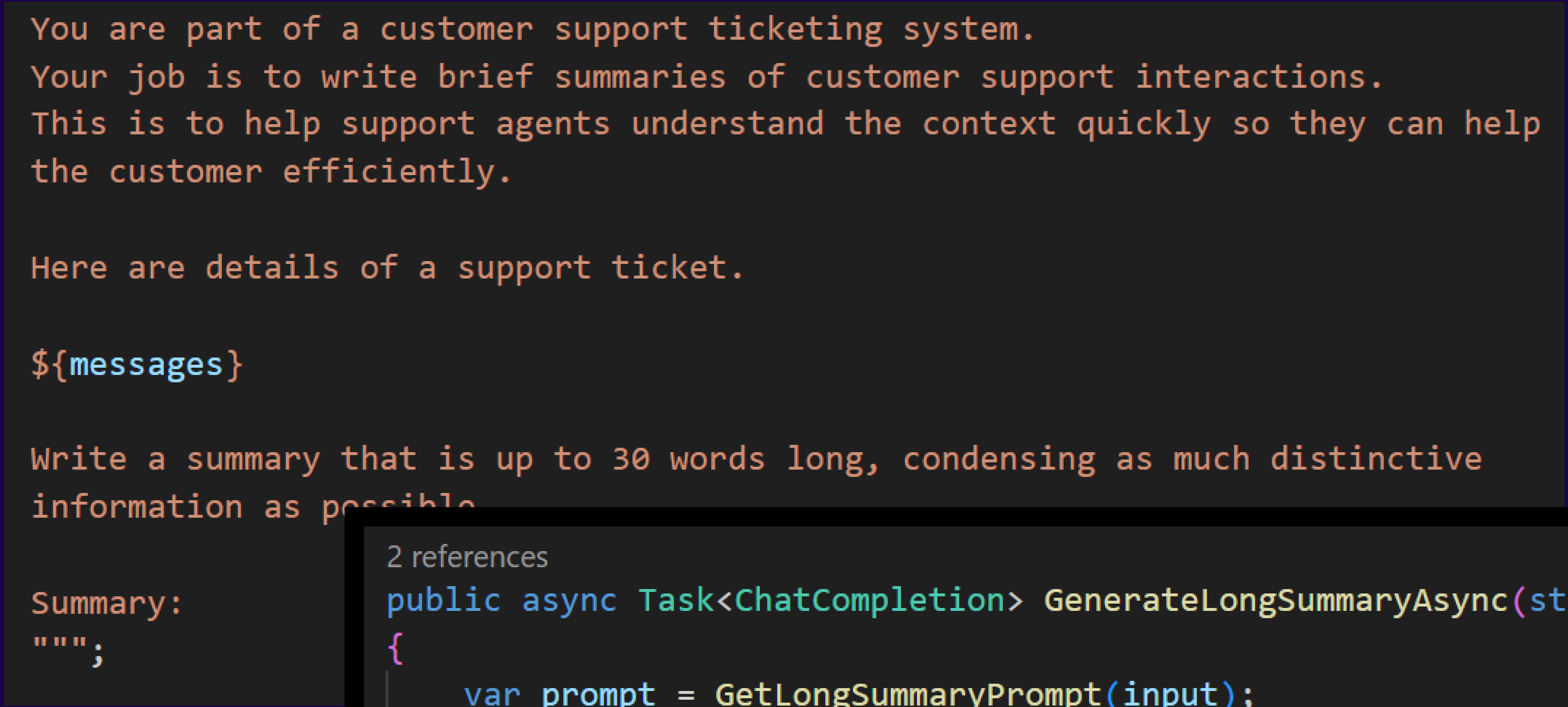
.NET Application
Leveraging AI
Microsoft.Extensions.AI (Middleware)
AI, Data Tools, & Services
UI Components (Smart Components)
AI SDKS (OllamaSharp, OpenAI, Azure AI Inference)
Libraries (Semantic Kernel,AutoGen)
Vector Store SDKs
Developer Tools (Prompty)
Microsoft.Extensions.AI.Abstractions
Microsoft.Extensions.VectorData

Want to change?

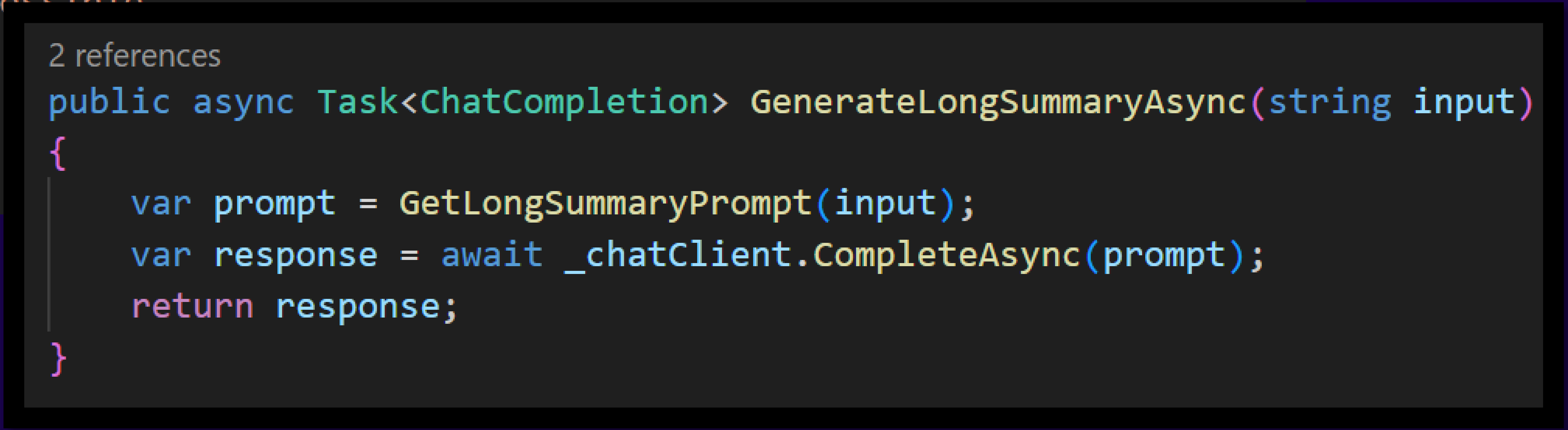


```
IChatClient chatClient = new OllamaApiClient(new Uri(endpoint), modelId);
IEmbeddingGenerator<string, Embedding<float>> embeddingGenerator =
    new OllamaApiClient(new Uri(endpoint), embeddingsId);
```

Prompt time!



```
You are part of a customer support ticketing system.
Your job is to write brief summaries of customer support interactions.
This is to help support agents understand the context quickly so they can help
the customer efficiently.

Here are details of a support ticket.

${messages}

Write a summary that is up to 30 words long, condensing as much distinctive
information as po

Summary:
""";
```



```
2 references
public async Task<ChatCompletion> GenerateLongSummaryAsync(string input)
{
    var prompt = GetLongSummaryPrompt(input);
    var response = await _chatClient.CompleteAsync(prompt);
    return response;
}
```

Demo: Summarization

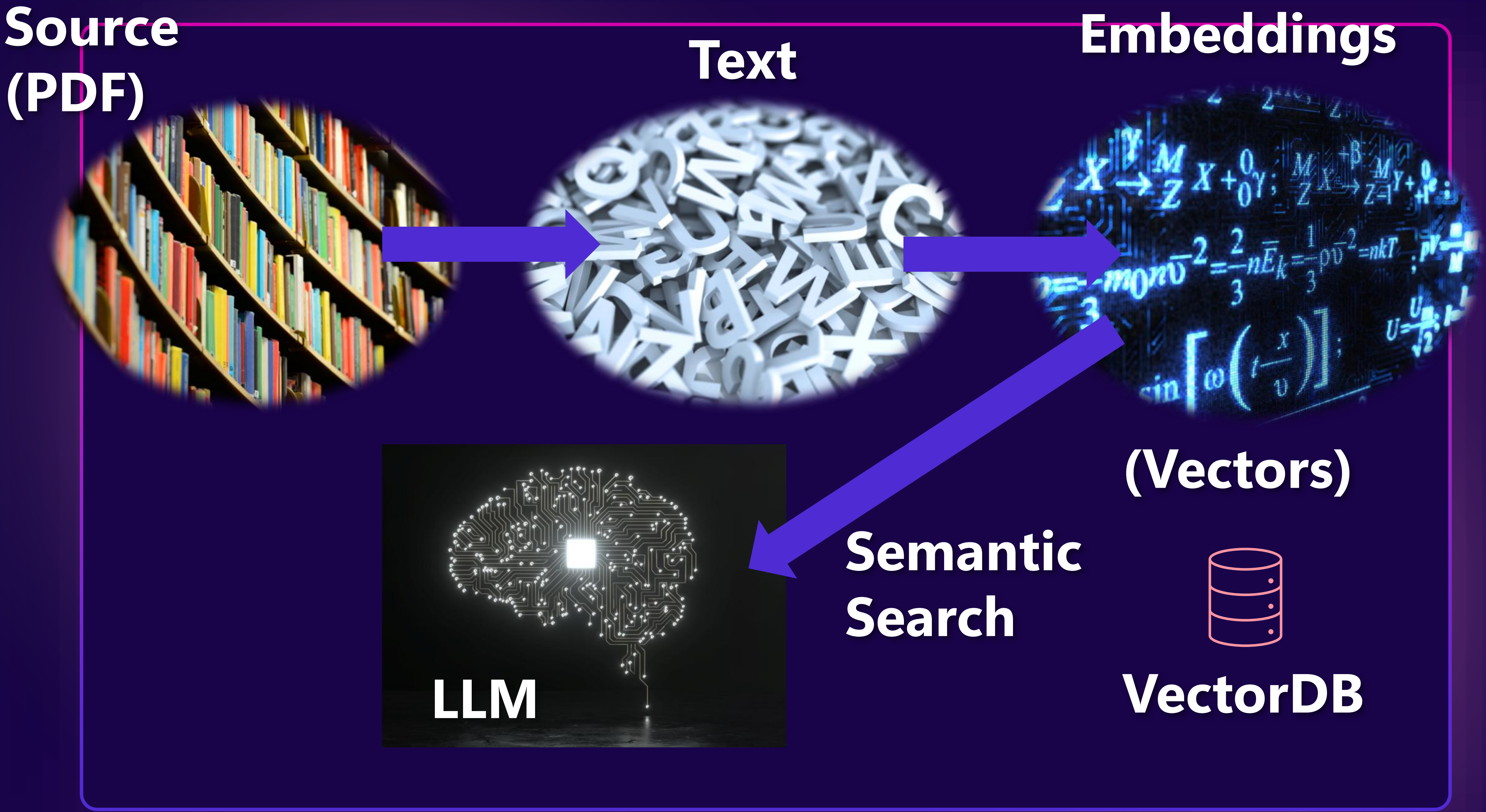
Jeremy Likness

RAG Time!

Retrieval

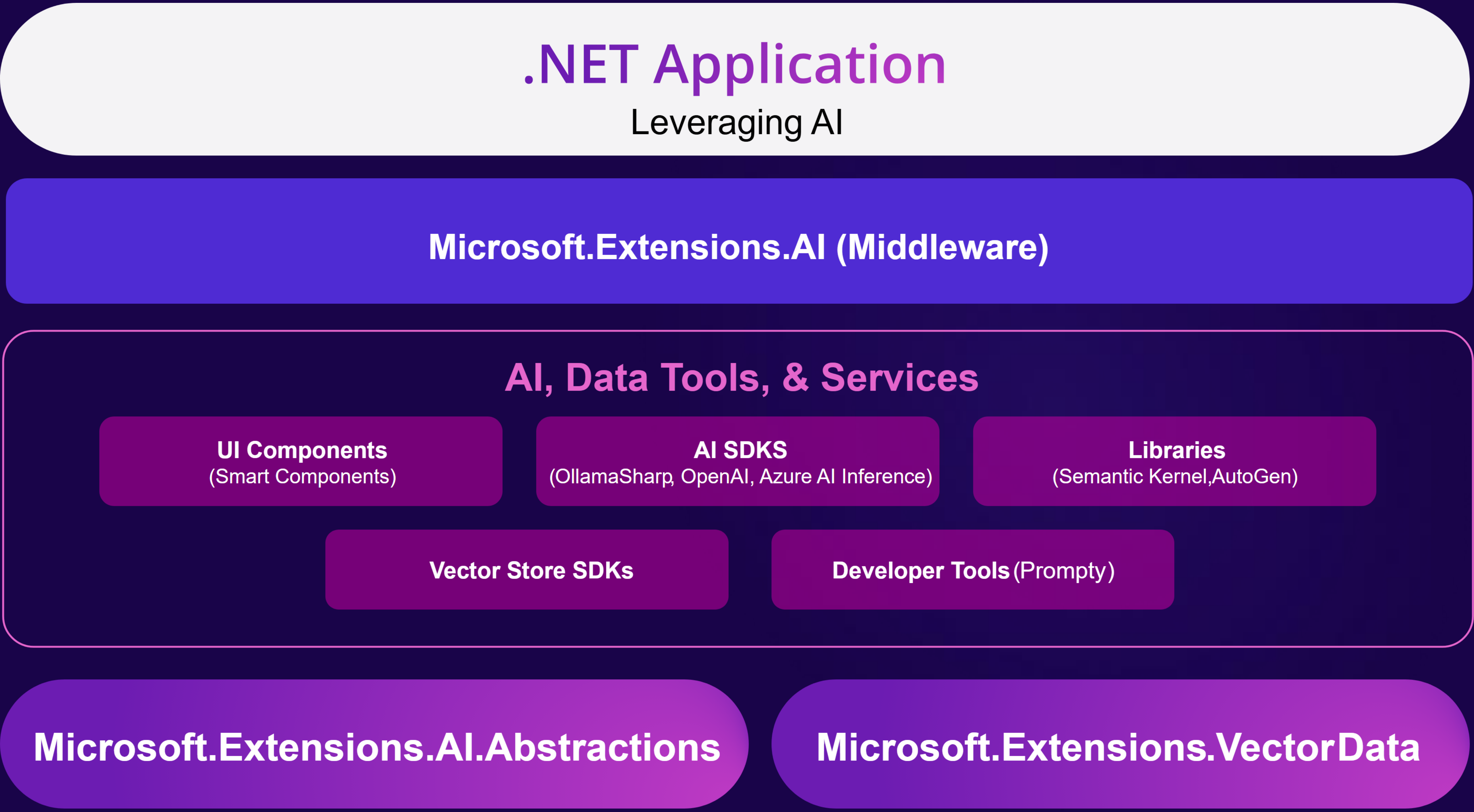
Augmented

Generation


Source
(PDF)
Text
Embeddings
(Vectors)
Semantic
Search
LLM
VectorDB

```csharp
IEmbeddingGenerator<string, Embedding<float>> embeddingGenerator =
    new OllamaApiClient(new Uri(endpoint), embeddingsId);
```

```csharp
// Configure product manual service
var vectorStore = new InMemoryVectorStore();
var productManualService = new ProductManualService(embeddingGenerator,
    vectorStore, useOpenAIEmbeddings);
```



.NET Application

Leveraging AI

Microsoft.Extensions.AI (Middleware)

AI, Data Tools, & Services

UI Components
(Smart Components)

AI SDKS
(OllamaSharp, OpenAI, Azure AI Inference)

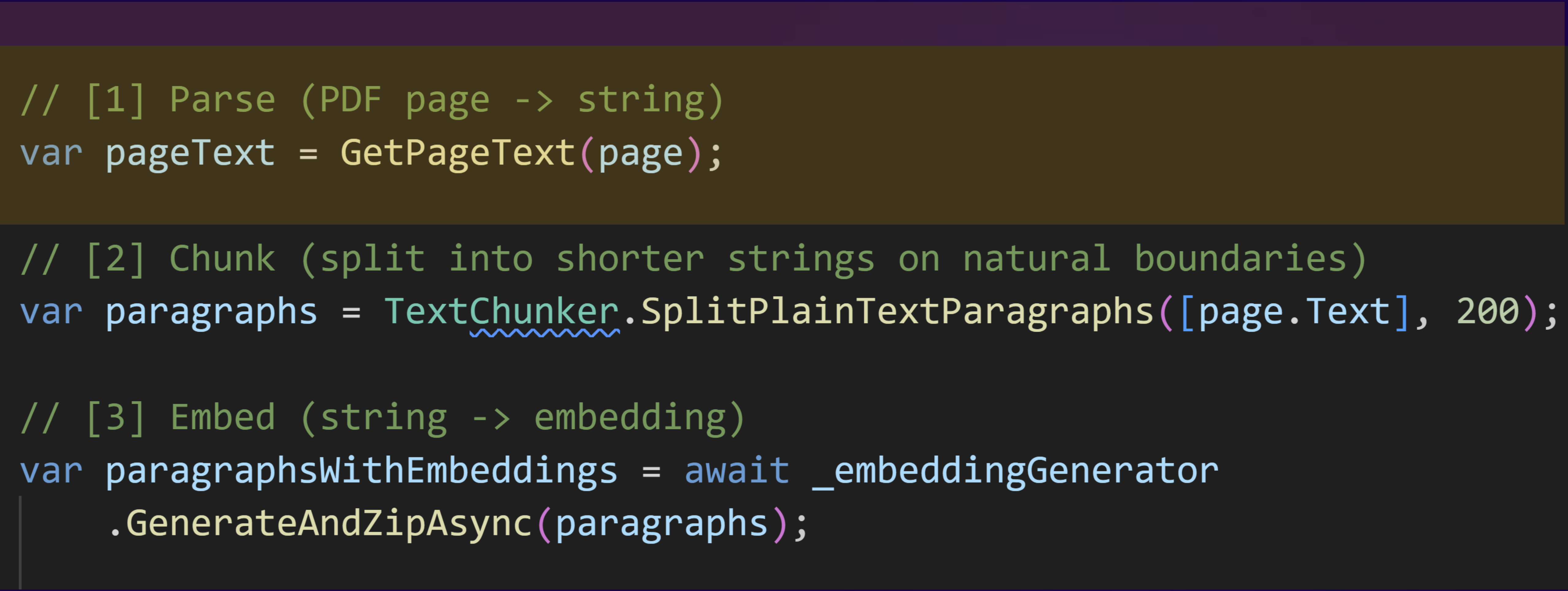
Libraries
(Semantic Kernel,AutoGen)

Vector Store SDKs

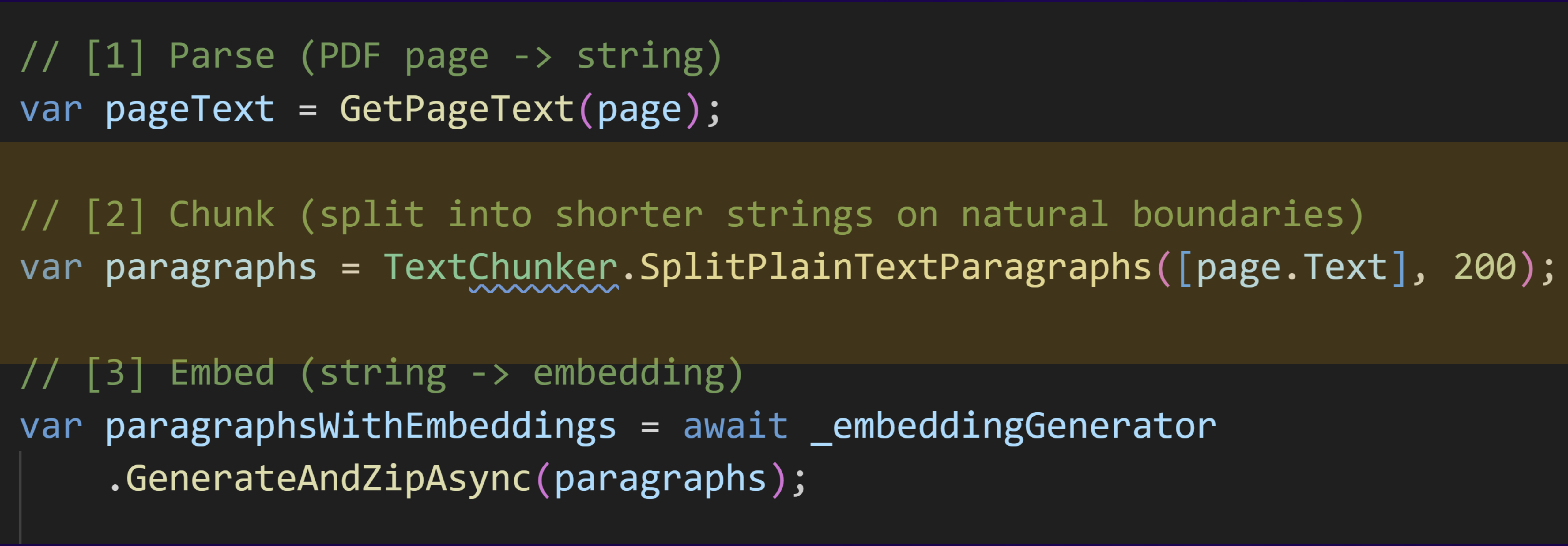
Developer Tools (Prompty)

Microsoft.Extensions.AI.Abstractions

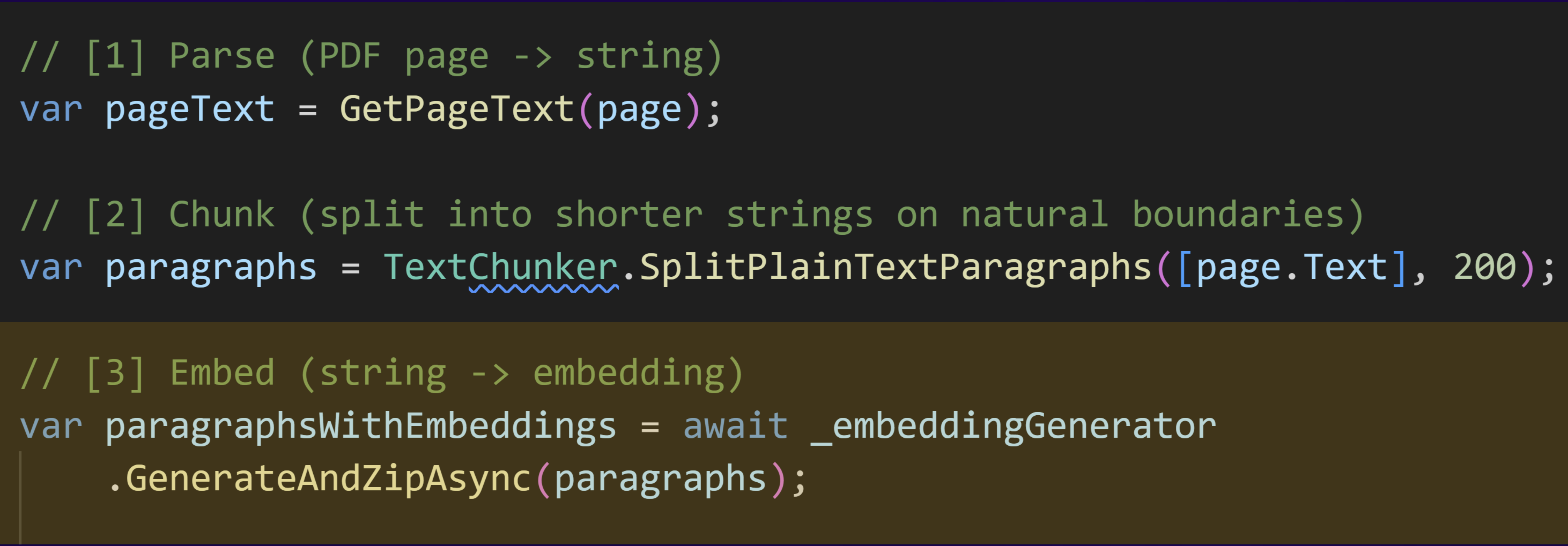
Microsoft.Extensions.VectorData



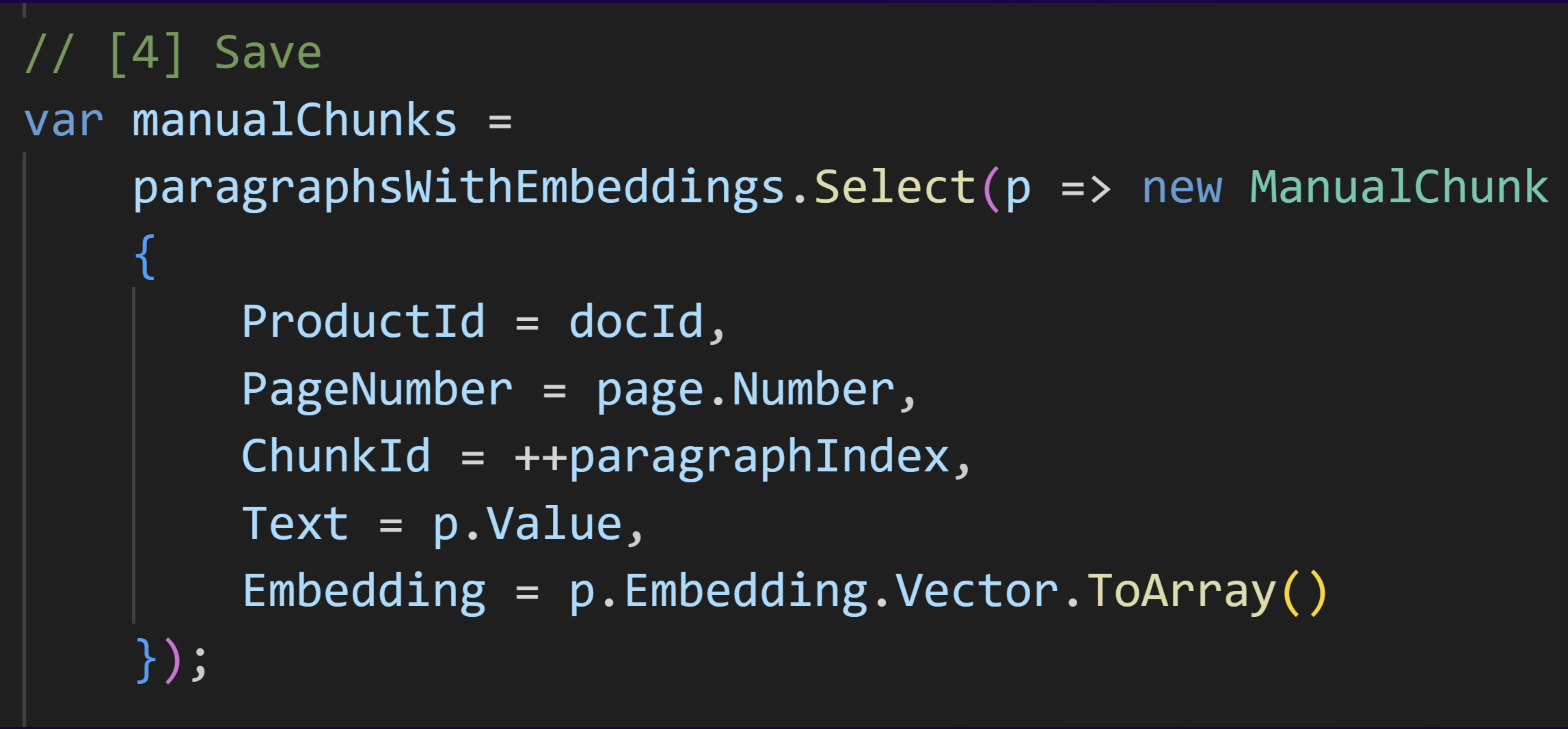
```csharp
// [1] Parse (PDF page -> string)
var pageText = GetPageText(page);

// [2] Chunk (split into shorter strings on natural boundaries)
var paragraphs = TextChunker.SplitPlainTextParagraphs([page.Text], 200);

// [3] Embed (string -> embedding)
var paragraphsWithEmbeddings = await _embeddingGenerator
    .GenerateAndZipAsync(paragraphs);
```



```
// [1] Parse (PDF page -> string)
var pageText = GetPageText(page);

// [2] Chunk (split into shorter strings on natural boundaries)
var paragraphs = TextChunker.SplitPlainTextParagraphs([page.Text], 200);

// [3] Embed (string -> embedding)
var paragraphsWithEmbeddings = await _embeddingGenerator
    .GenerateAndZipAsync(paragraphs);
```



```csharp
// [1] Parse (PDF page -> string)
var pageText = GetPageText(page);

// [2] Chunk (split into shorter strings on natural boundaries)
var paragraphs = TextChunker.SplitPlainTextParagraphs([page.Text], 200);

// [3] Embed (string -> embedding)
var paragraphsWithEmbeddings = await _embeddingGenerator
    .GenerateAndZipAsync(paragraphs);
```



```
// [4] Save
var manualChunks =
    paragraphsWithEmbeddings.Select(p => new ManualChunk
    {
        ProductId = docId,
        PageNumber = page.Number,
        ChunkId = ++paragraphIndex,
        Text = p.Value,
        Embedding = p.Embedding.Vector.ToArray()
    });
```

```
{
  "ChunkId": 2460,
  "ProductId": 179,
  "PageNumber": 3,
  "Text": "Misuse and Abuse",
  "Embedding": [0.02627289, 0.066441216, 0.04378815, 0.03978883,
```

Demo:
RAG and Semantic Search

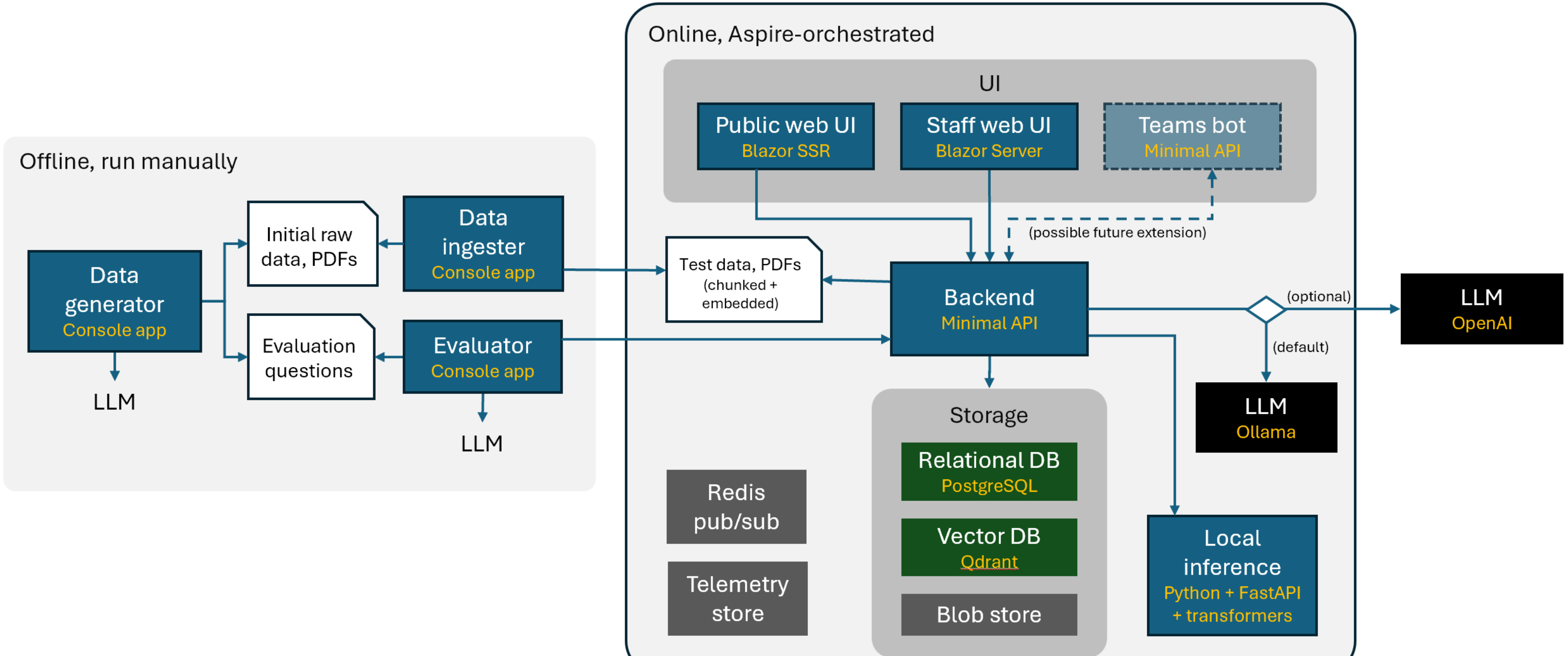
Jeremy Likness

“

But wait... Our system is web-based and has a few more moving pieces...”

No worries....

eShopSupport



Demo: eShopSupport

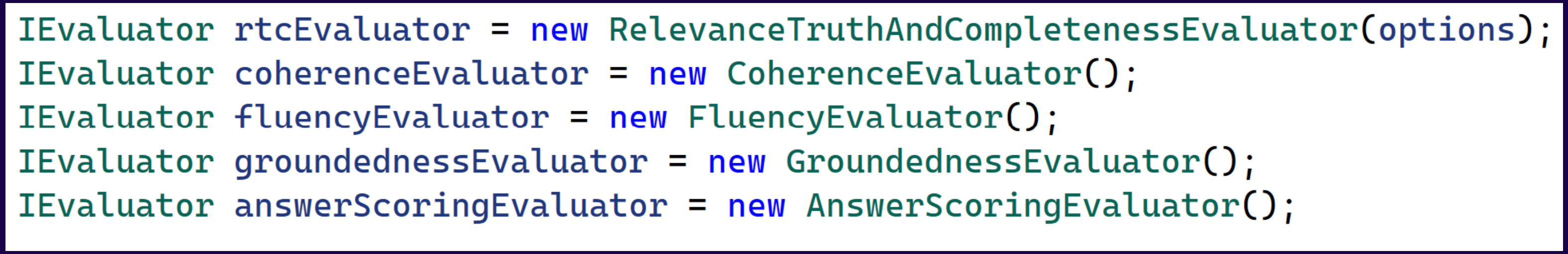
Jeremy Likness

AI Evaluations

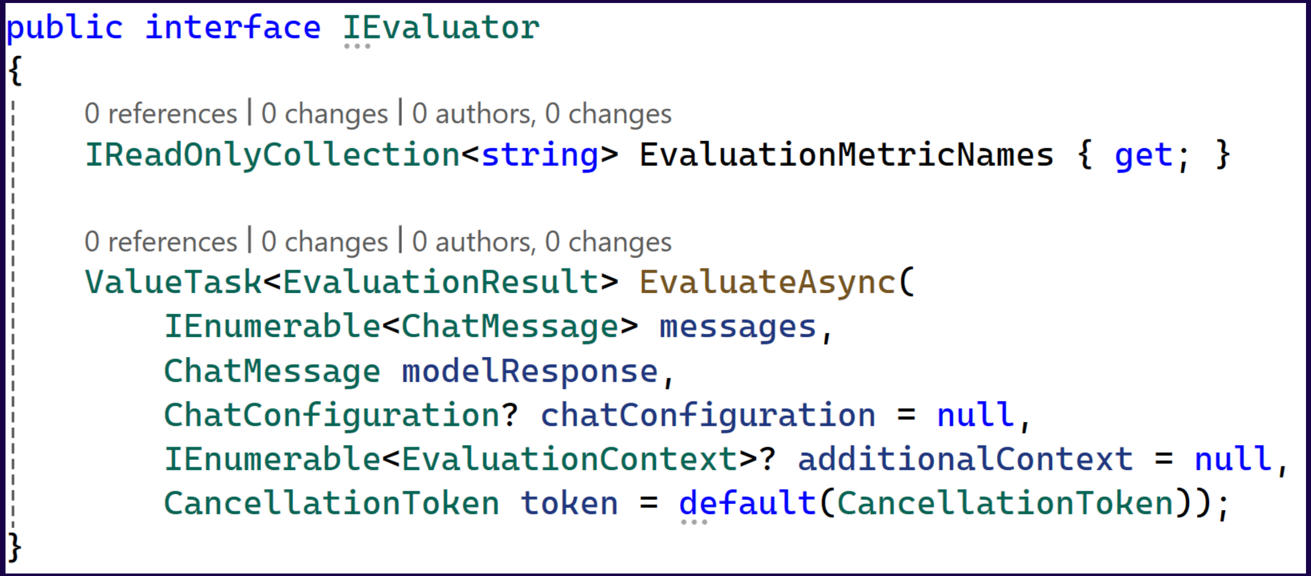
Traditional Software vs AI Evaluations

	Traditional Software	AI
Outputs	Predictable, Objective	Unpredictable, Subjective (Vibes-based)
Patterns	TDD, Assertions	Thresholds, Variable
Tools	Integrated (VS, CI/CD)	Isolated, Disparate

Built-in Evaluators & Extensible APIs

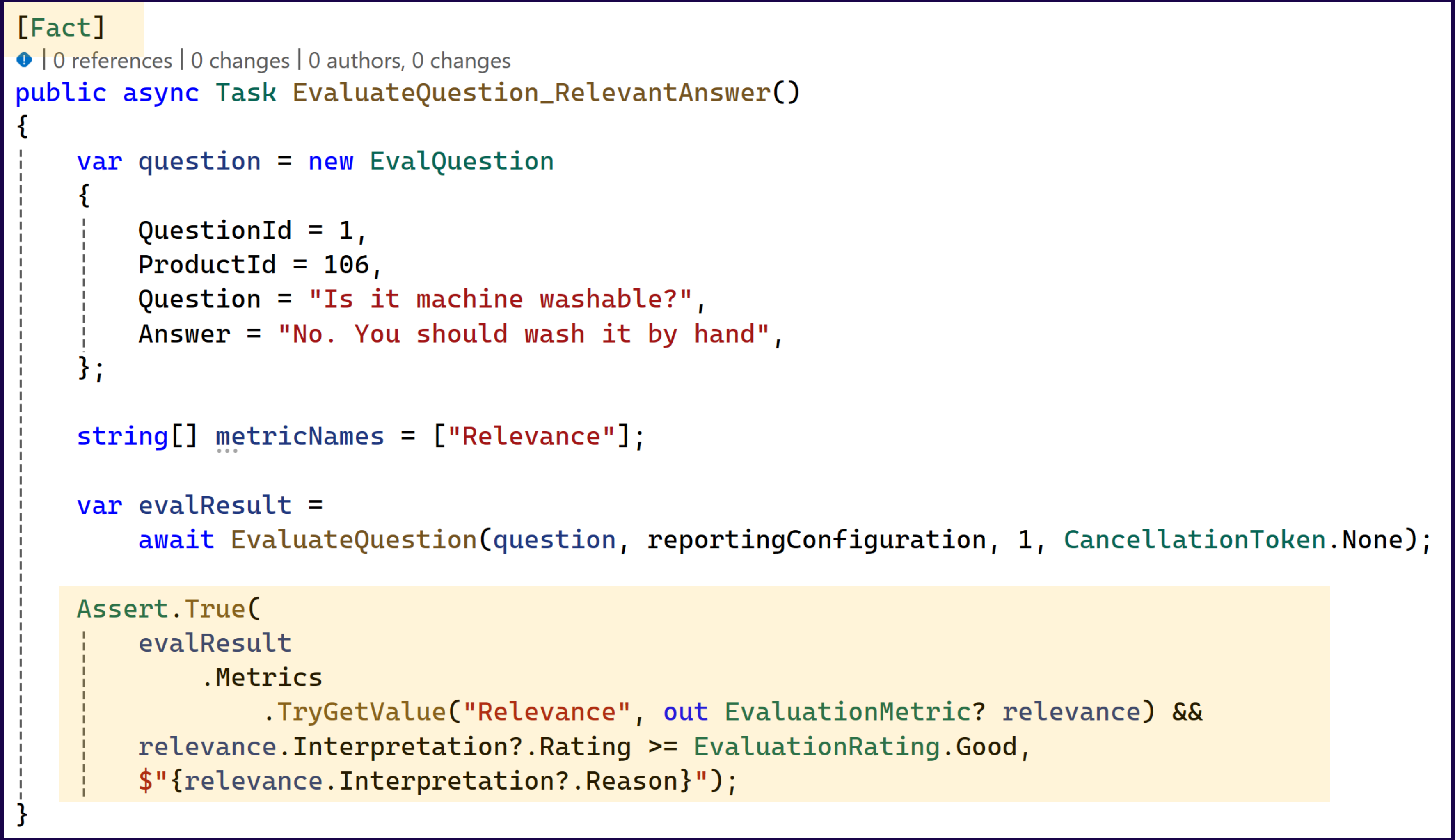


```csharp
IEvaluator rtcEvaluator = new RelevanceTruthAndCompletenessEvaluator(options);
IEvaluator coherenceEvaluator = new CoherenceEvaluator();
IEvaluator fluencyEvaluator = new FluencyEvaluator();
IEvaluator groundednessEvaluator = new GroundednessEvaluator();
IEvaluator answerScoringEvaluator = new AnswerScoringEvaluator();
```



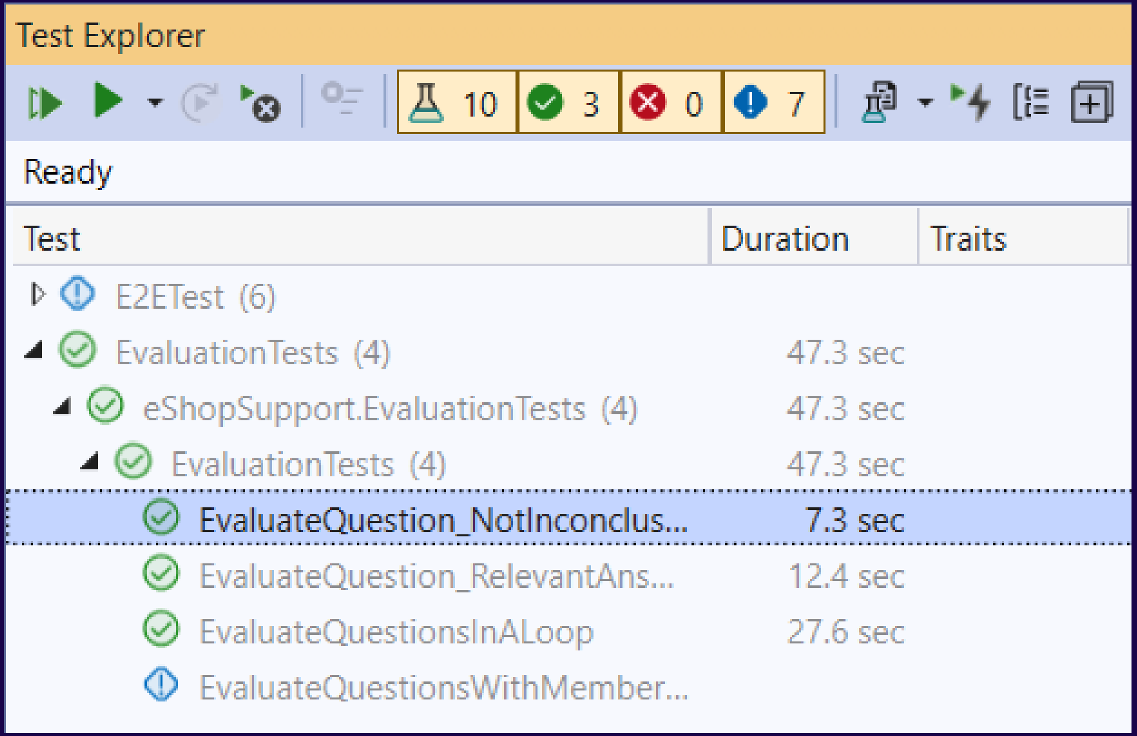
```csharp
public interface IEvaluator
{
    0 references | 0 changes | 0 authors, 0 changes
    IReadOnlyCollection<string> EvaluationMetricNames { get; }

    0 references | 0 changes | 0 authors, 0 changes
    ValueTask<EvaluationResult> EvaluateAsync(
        IEnumerable<ChatMessage> messages,
        ChatMessage modelResponse,
        ChatConfiguration? chatConfiguration = null,
        IEnumerable<EvaluationContext>? additionalContext = null,
        CancellationToken token = default(CancellationToken));
}
```

Integrate existing and new patterns

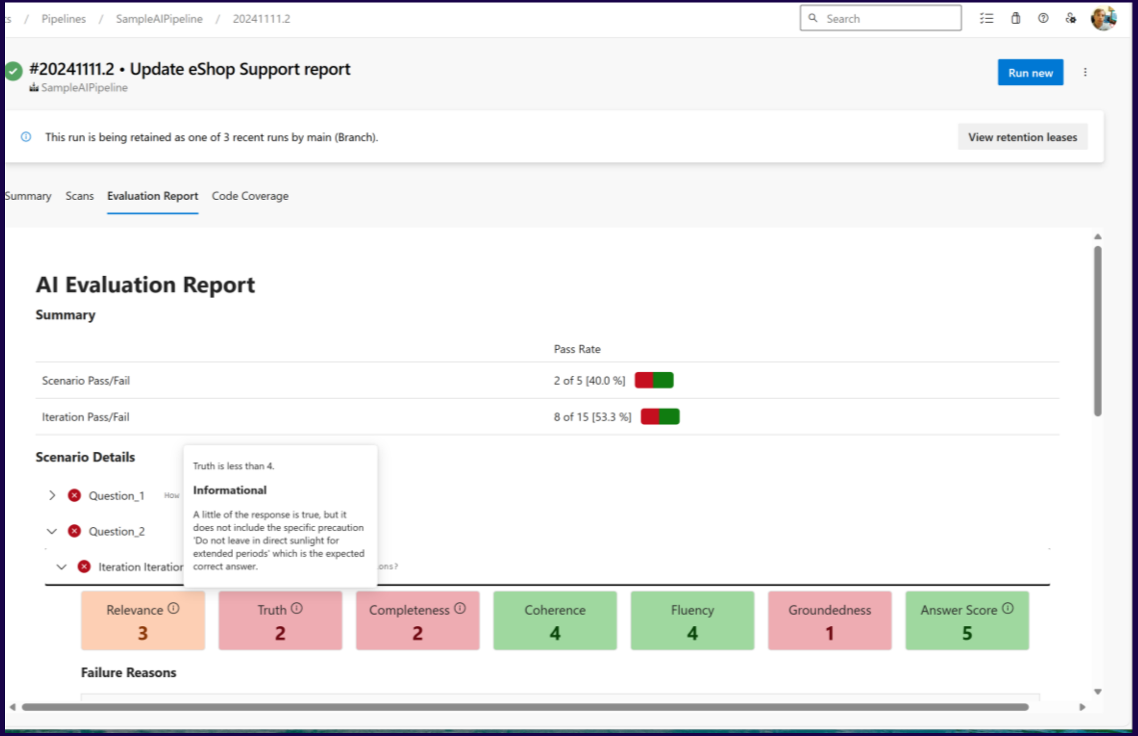


```
[Fact]
❶ | 0 references | 0 changes | 0 authors, 0 changes
public async Task EvaluateQuestion_RelevantAnswer()
{
    var question = new EvalQuestion
    {
        QuestionId = 1,
        ProductId = 106,
        Question = "Is it machine washable?",
        Answer = "No. You should wash it by hand",
    };

    string[] metricNames = ["Relevance"];

    var evalResult =
        await EvaluateQuestion(question, reportingConfiguration, 1, CancellationToken.None);

    Assert.True(
        evalResult
            .Metrics
                .TryGetValue("Relevance", out EvaluationMetric? relevance) &&
        relevance.Interpretation?.Rating >= EvaluationRating.Good,
        $"{relevance.Interpretation?.Reason}");
}
```

Works with existing tooling



Test	Duration	Traits
E2ETest (6)		
EvaluationTests (4)	47.3 sec	
eShopSupport.EvaluationTests (4)	47.3 sec	
EvaluationTests (4)	47.3 sec	
EvaluateQuestion_NotInconclus...	7.3 sec	
EvaluateQuestion_RelevantAns...	12.4 sec	
EvaluateQuestionsInALoop	27.6 sec	
EvaluateQuestionsWithMember...		



#20241111.2 • Update eShop Support report
SampleAIPipeline

This run is being retained as one of 3 recent runs by main (Branch).

AI Evaluation Report

Summary

	Pass Rate
Scenario Pass/Fail	2 of 5 [40.0 %]
Iteration Pass/Fail	8 of 15 [53.3 %]

Scenario Details

Question_1
Question_2
Iteration Iteration

Truth is less than 4.

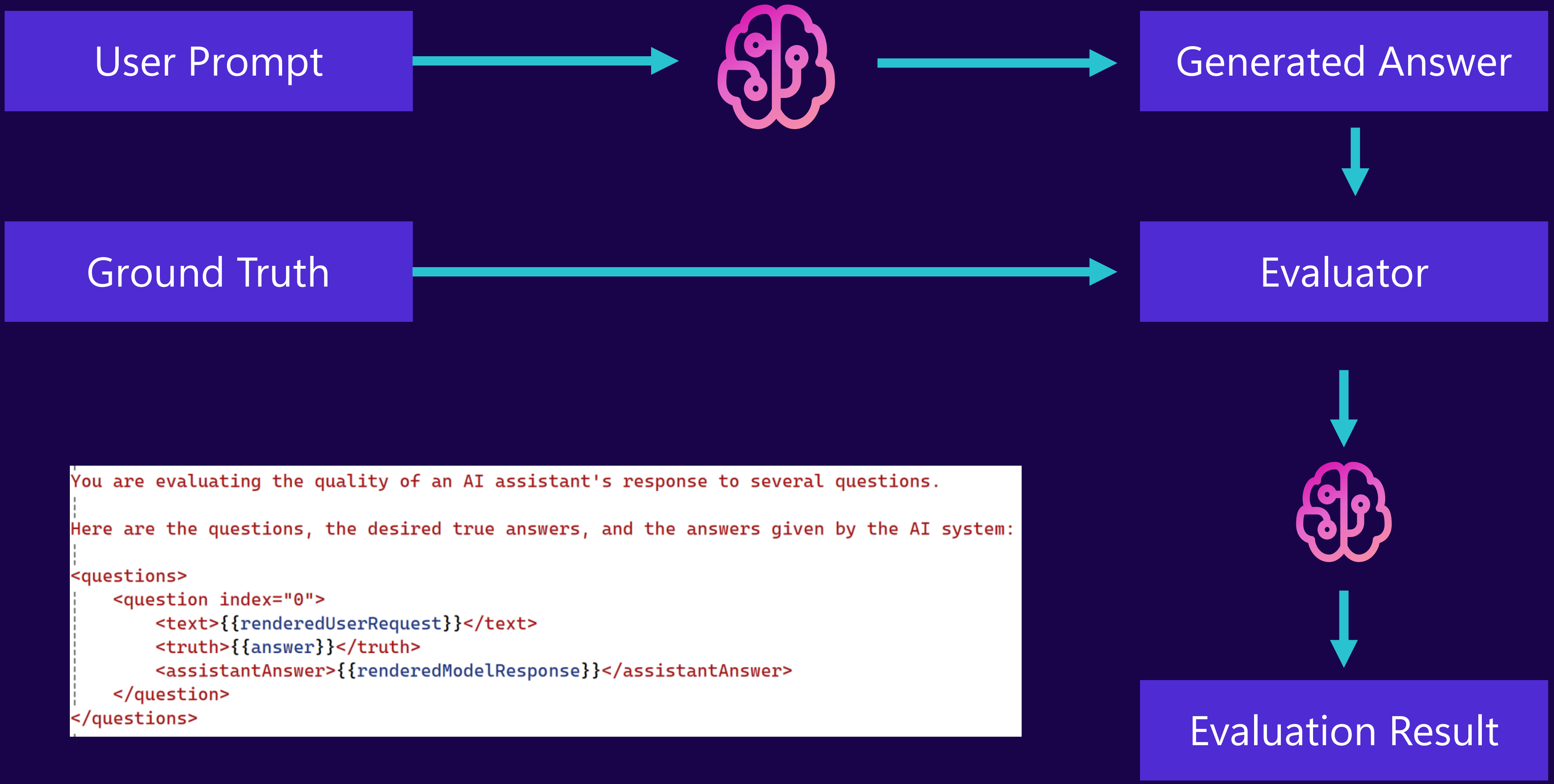
Informational

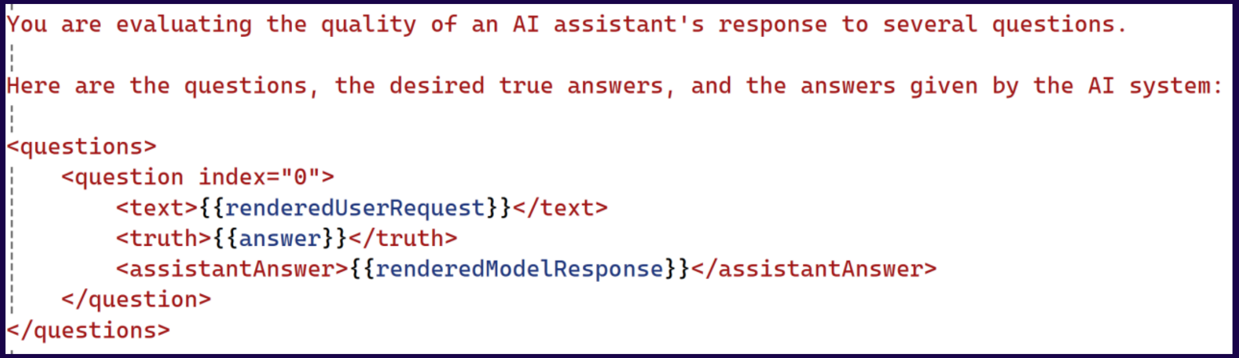
A little of the response is true, but it does not include the specific precaution 'Do not leave in direct sunlight for extended periods' which is the expected correct answer.

Relevance	Truth	Completeness	Coherence	Fluency	Groundedness	Answer Score
3	2	2	4	4	1	5

Failure Reasons

AI Evaluations





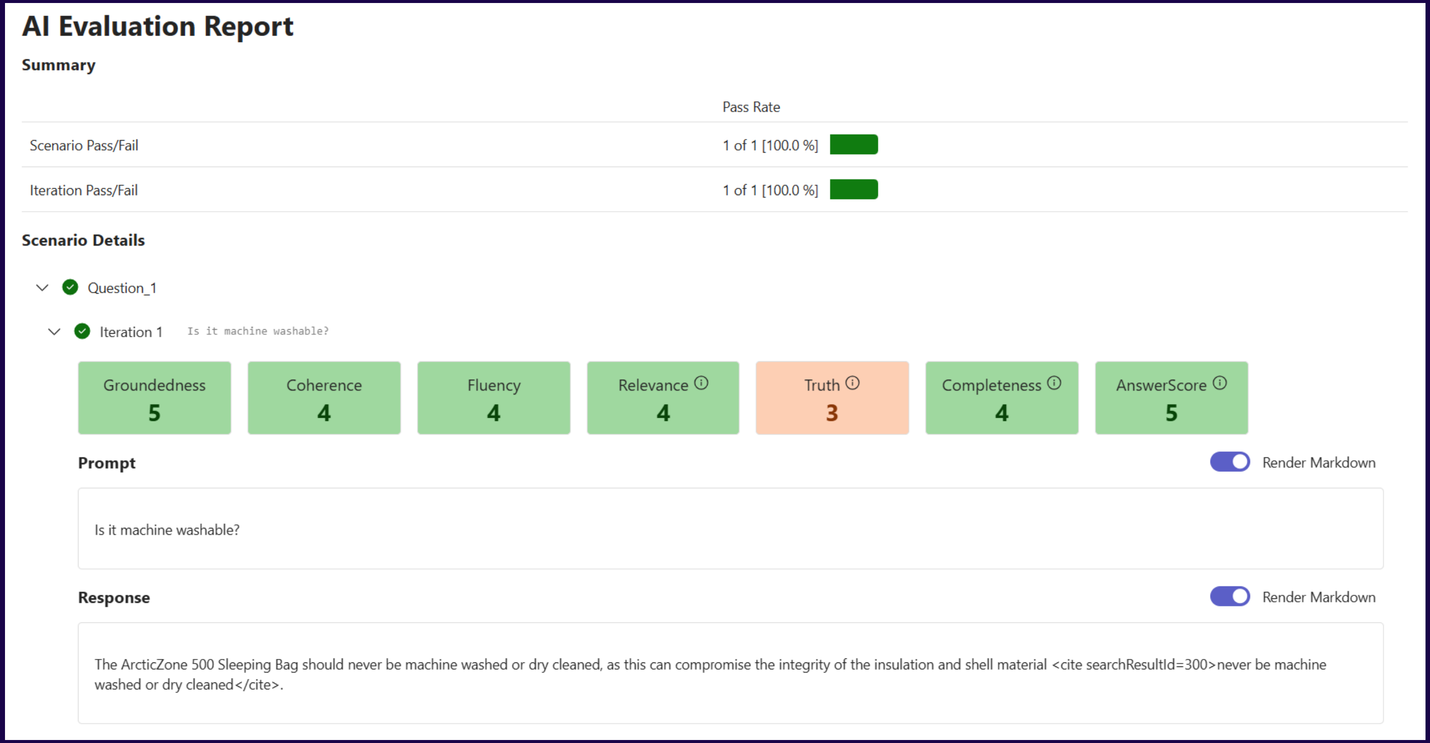
```
You are evaluating the quality of an AI assistant's response to several questions.

Here are the questions, the desired true answers, and the answers given by the AI system:

<questions>
    <question index="0">
        <text>{{renderedUserRequest}}</text>
        <truth>{{answer}}</truth>
        <assistantAnswer>{{renderedModelResponse}}</assistantAnswer>
    </question>
</questions>
```

Demo:
AI Evaluations

Luis Quintanilla

Report Generation

```
dotnet aieval report --path C:\src\eShopCache\ --output eval-report.html
```



AI Evaluation Report

Summary

	Pass Rate
Scenario Pass/Fail	1 of 1 [100.0 %] 🟩
Iteration Pass/Fail	1 of 1 [100.0 %] 🟩

Scenario Details

✓ Question_1

✓ Iteration 1 `Is it machine washable?`

Groundedness	Coherence	Fluency	Relevance ⓘ	Truth ⓘ	Completeness ⓘ	AnswerScore ⓘ
5	4	4	4	3	4	5

Prompt

Is it machine washable?

Response

The ArcticZone 500 Sleeping Bag should never be machine washed or dry cleaned, as this can compromise the integrity of the insulation and shell material <cite searchResultId=300>never be machine washed or dry cleaned</cite>.

AI Evaluations

Coming Soon

Get .NET 9

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

Resources

Docs: *aka.ms/dotnet/ai/docs*

Samples: *aka.ms/dotnet/ai/samples*

Thank you