Microsoft

Introduce Phi-3

Kinfey Lo

Microsoft Senior Cloud Advocate

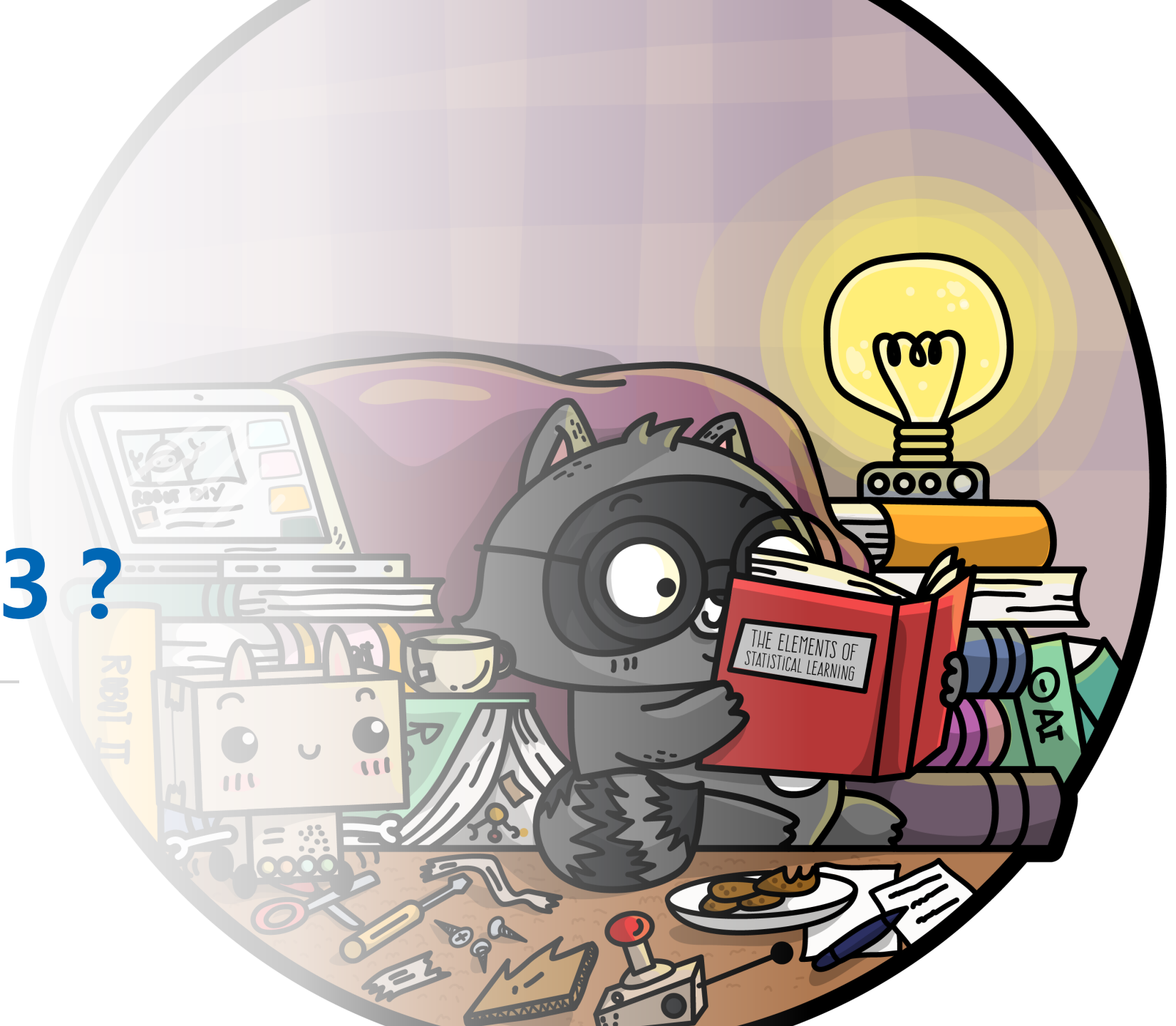
Generative AI

OpenAI

- GPT-3
- GPT-3.5
- GPT-4/4o
- DALLE
- text ada embedding
- Whisper

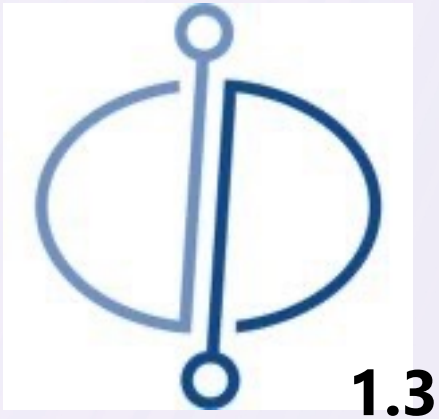
NVIDIA

What is Phi-3 ?



The development of the Microsoft Phi

Code, Text Completion, Chat



1.3B

microsoft/phi-1

Python coding model with perf. comparable to models 10x larger trained on 100x more data

1.3B

microsoft/phi-1_5

Natural language model
with NL comparable to models 10x larger trained on 30x more data and reasoning comparable to models 50x larger.

2.7B

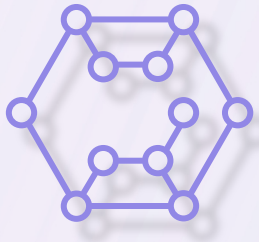
microsoft/phi-2

a 2.7 billion-parameter language model that demonstrates outstanding reasoning and language understanding capabilities, showcasing state-of-the-art performance among base language models with less than 13 billion parameters

Microsoft Phi 3 Family

Phi-3-mini

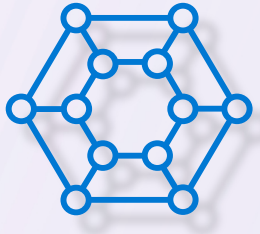
a 3.8B language model is available on Microsoft Azure AI Studio, Hugging Face, and Ollama. Phi-3 models significantly outperform language models of the same and larger sizes on key benchmarks (see benchmark numbers below, higher is better). Phi-3-mini does better than models twice its size, and Phi-3-small and Phi-3-medium outperform much larger models, including GPT-3.5



Phi-3-small & medium

Phi-3-small with only 7B parameters beats GPT-3.5T across a variety of language, reasoning, coding and math benchmarks.

Phi-3-medium with 14B parameters continues the trend and outperforms Gemini 1.0 Pro.



Phi-3-Vision

Phi-3-vision with just 4.2B parameters continues that trend and outperforms larger models such as Claude-3 Haiku and Gemini 1.0 Pro V across general visual reasoning tasks, OCR, table and chart understanding tasks.

Microsoft Phi 3 Family

Small Language Model, Best in class performance for size, with frictionless availability

Phi-3-mini-V
(3.8B + 0.3B)
Available today

Phi-3-small
(7B)
Available today

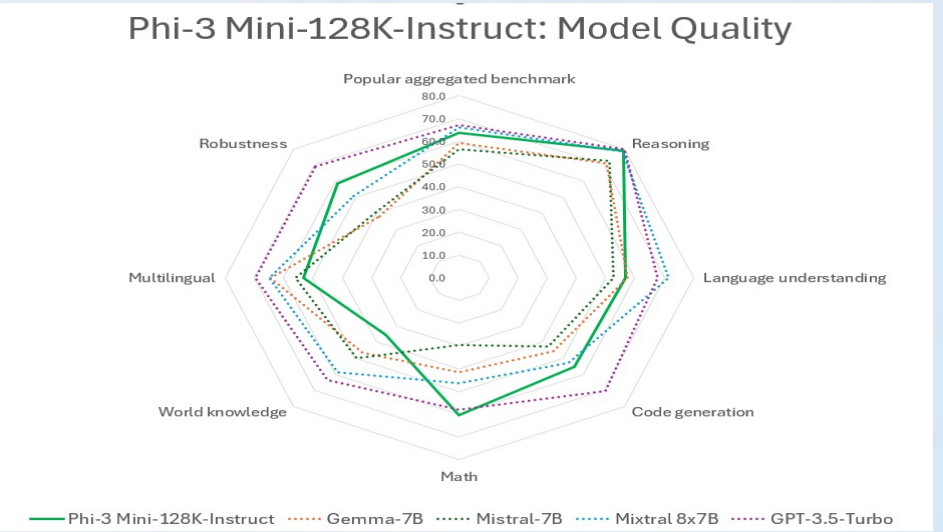
Phi-3-mini
(3.8B)
Available today

Azure AI Model Catalog

Hugging Face

Ollama

Core Value Proposition	✓ **Groundbreaking performance at a small size:** Faster and Cost Effective & Low costs in Fine Tuning. Excellent performance in latency bound scenarios where fast response times are critical ✓ **Unlocking new capabilities: Works best across r**esource constrained environments where local inference may be needed. Can deploy on edge & cloud with ONNX & TensorFlow Lite ✓ **Safe by Design**: Developed in accordance with Microsoft Responsible AI principles & trained on with high quality data
Description	Phi-3 Mini is a lightweight, state-of-the-art open model built upon datasets used for Phi-2 - synthetic data and filtered websites - with a focus on very high-quality, reasoning dense data. The model belongs to the Phi-3 model family and incorporates safety, alignment, and RLHF training.
Architecture	Phi-3 Mini has 3.8B parameters and is a dense decoder-only Transformer model.
Inputs	Text. It's best suited for prompts using chat format.
Context length	128K tokens - *"the first model in its weight class to support long-context of up to 128K"*
Outputs	Generated text in response to the input
Dates	Trained between Feb 2024 and April 2024
Status	This is a static model trained on an offline dataset. Future versions of the tuned models may be released as we improve models.
License	MIT



	Phi-3 Mini-128K-Instruct	Gemma-7B	Mistral-7B	Mixtral 8x7B	GPT-3.5-Turbo
Average across 80 benchmarks	57.7	52.6	47.2	56.2	62.2

Category	Phi-3 Mini-128K-Instruct	Gemma-7B	Mistral-7B	Mixtral 8x7B	GPT-3.5-Turbo
Popular aggregated benchmark	63.5	59.4	56.5	66.2	67.0
Reasoning	79.1	71.3	72.6	78.6	80.0
Language understanding	56.7	57.6	52.5	71.5	67.7
Code generation	55.5	45.6	42.9	52.7	70.4
Math	60.3	41.4	29.7	46.2	58.0
World knowledge	35.7	46.7	49.8	58.6	63.4
Multilingual	53.1	64.6	56.0	64.9	69.6
Robustness	58.6	38.4	40.6	51.0	69.3

Microsoft Phi 3

Phi-3-mini-V
(3.8B + 0.3B)

Phi-3-small
(7B)

Phi-3-mini
(3.8B)

ONNX Runtime

GGUF

Quantitative(INT4,FP16,FP32)

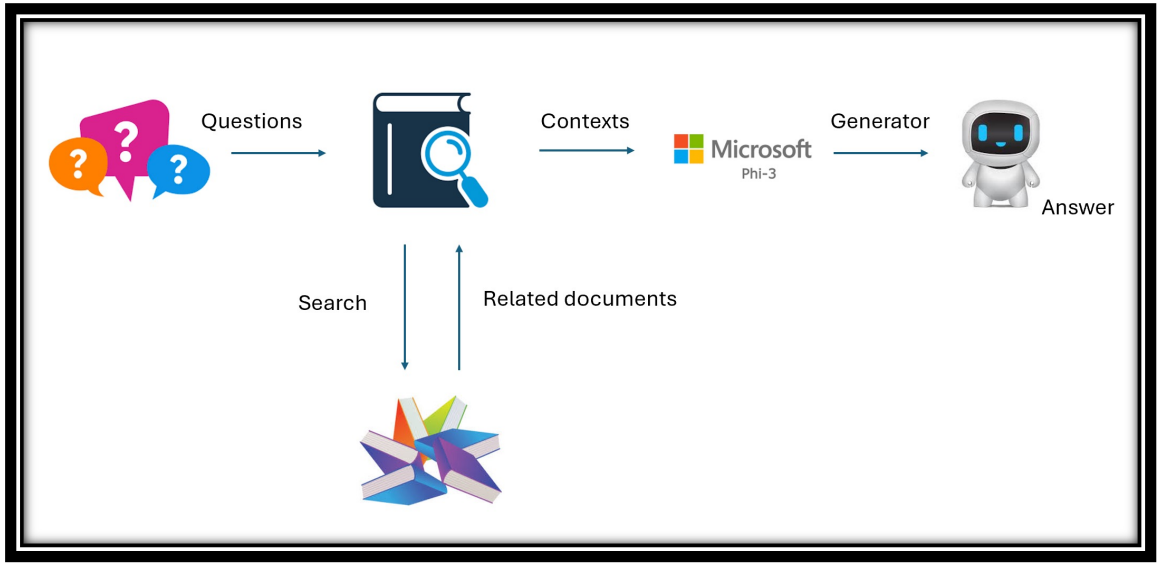
Demo : Call Phi-3-mini & Phi-3-Vision

Fine-tuning vs RAG


How about RAG?

RAG - Retrieval-Augmented Generation

RAG is data retrieval + text generation. The structured data and unstructured data of the enterprise are stored in the vector database. When searching for relevant content, the relevant summary and content are found to form a context, and the text completion capability of LLM/SLM is combined to generate content.



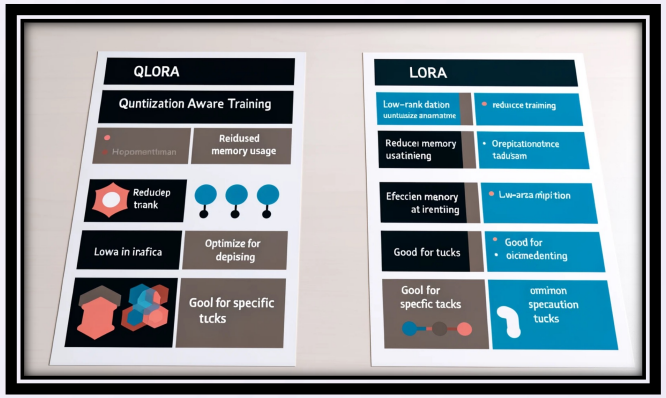
A vector database is a collection of data stored in mathematical form. Vector databases make it easier for machine learning models to remember previous inputs, enabling machine learning to be used to support use cases such as search, recommendations, and text generation. Data can be identified based on similarity metrics rather than exact matches, allowing computer models to understand the context of the data.

Vector database is the key to realizing RAG. We can convert data into vector storage through vector models such as text-embedding-3, jina-ai-embedding, etc.

Demo : RAG with Phi-3

Fine-tuning

Fine-tuning is based on improvement of a certain model. It does not need to start with the model algorithm, but data needs to be continuously accumulated. If you want more precise terminology and language expression in industry applications, fine-tuning is your better choice. But if your data changes frequently, fine-tuning can become complicated.



LoRA (Low-Rank Adaptation) and QLoRA (Quantized Low-Rank Adaptation) are both techniques used to fine-tune large language models (LLMs) using Parameter Efficient Fine Tuning (PEFT). PEFT techniques are designed to train models more efficiently than traditional methods.

LoRA is a standalone finetuning technique that reduces memory footprint by applying a low-rank approximation to the weight update matrix. It offers fast training times and maintains performance close to traditional fine-tuning methods.

QLoRA is an extended version of LoRA that incorporates quantization techniques to further reduce memory usage. QLoRA quantizes the precision of the weight parameters in the pre-trained LLM to 4-bit precision, which is more memory efficient than LoRA. However, QLoRA training is about 30% slower than LoRA training due to the additional quantization and dequantization steps.

QLoRA uses LoRA as an accessory to fix the errors introduced during quantization errors. QLoRA enables the fine-tuning of massive models with billions of parameters on relatively small, highly available GPUs. For example, QLoRA can fine-tune a 70B parameter model that requires 36 GPUs with only 2

RAG vs Fine-tuning

How to choose

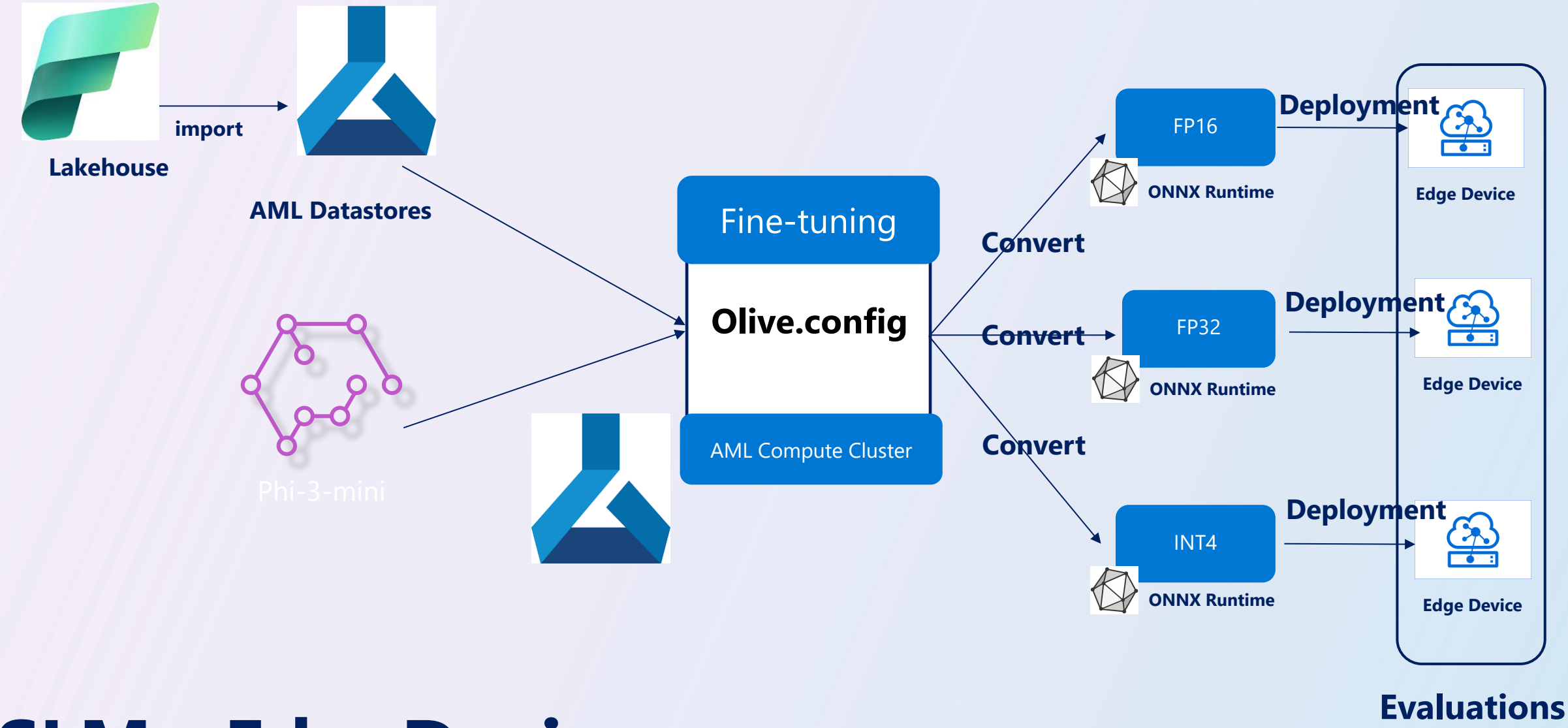
- If our answer requires the introduction of external data, RAG is the best choice
- If you need to output stable and precise industry knowledge, fine-tuning will be a good choice. RAG prioritizes pulling relevant content but might not always nail the specialized nuances.
- Fine-tuning requires a high-quality data set, and if it is just a small range of data, it will not make much difference. RAG is more flexible
- Fine-tuning is a black box, a metaphysics, and it is difficult to understand the internal mechanism. But RAG can make it easier to find the source of the data, thereby effectively adjusting hallucinations or content errors and providing better transparency.

Scenarios

- Vertical industries require specific professional vocabulary and expressions, Fine-tuning will be the best choice
- QA system, involving the synthesis of different knowledge points, RAG will be the best choice
- The combination of automated business flow RAG + Fine-tuning is the best choice

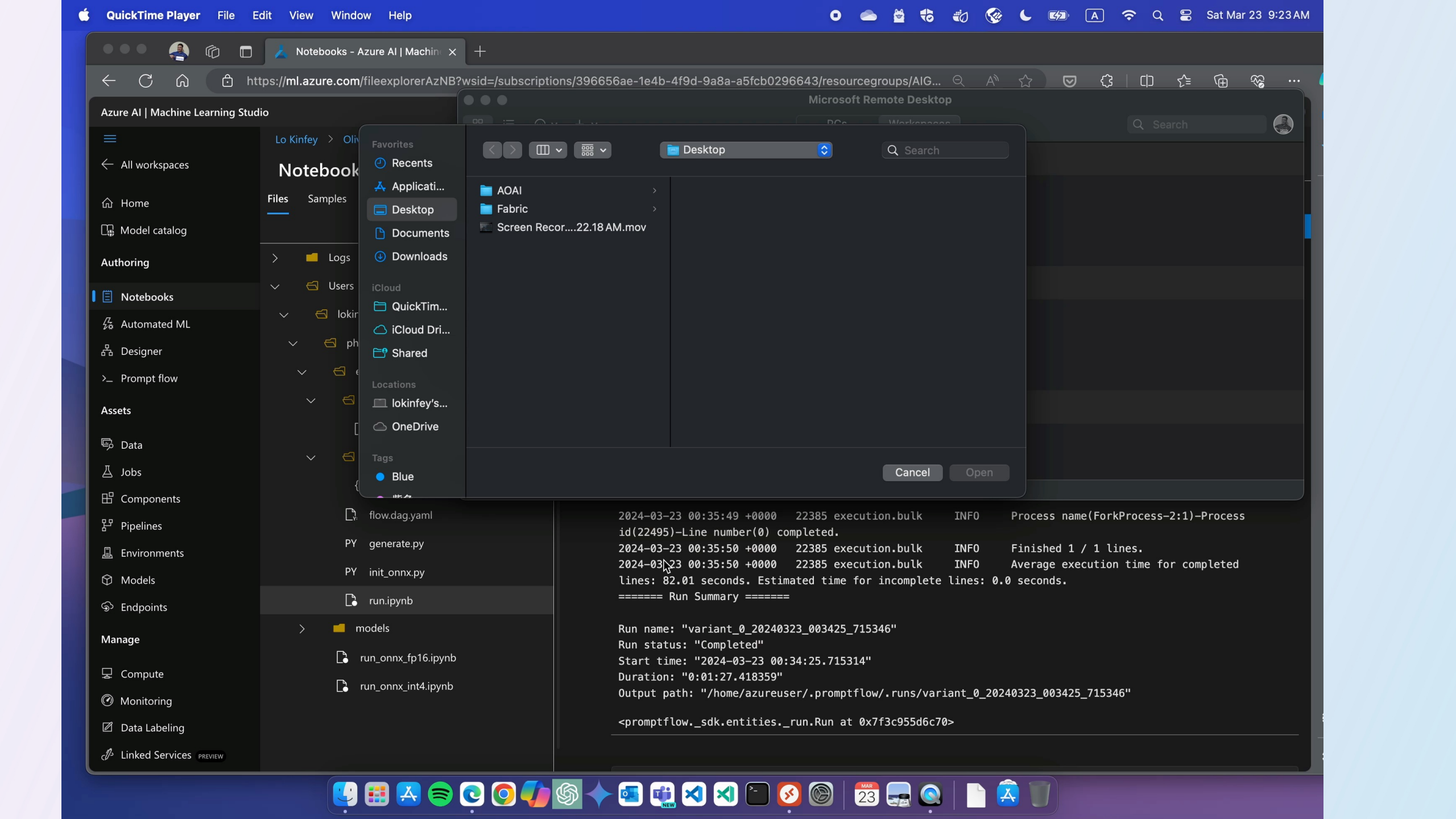
Microsoft Olive

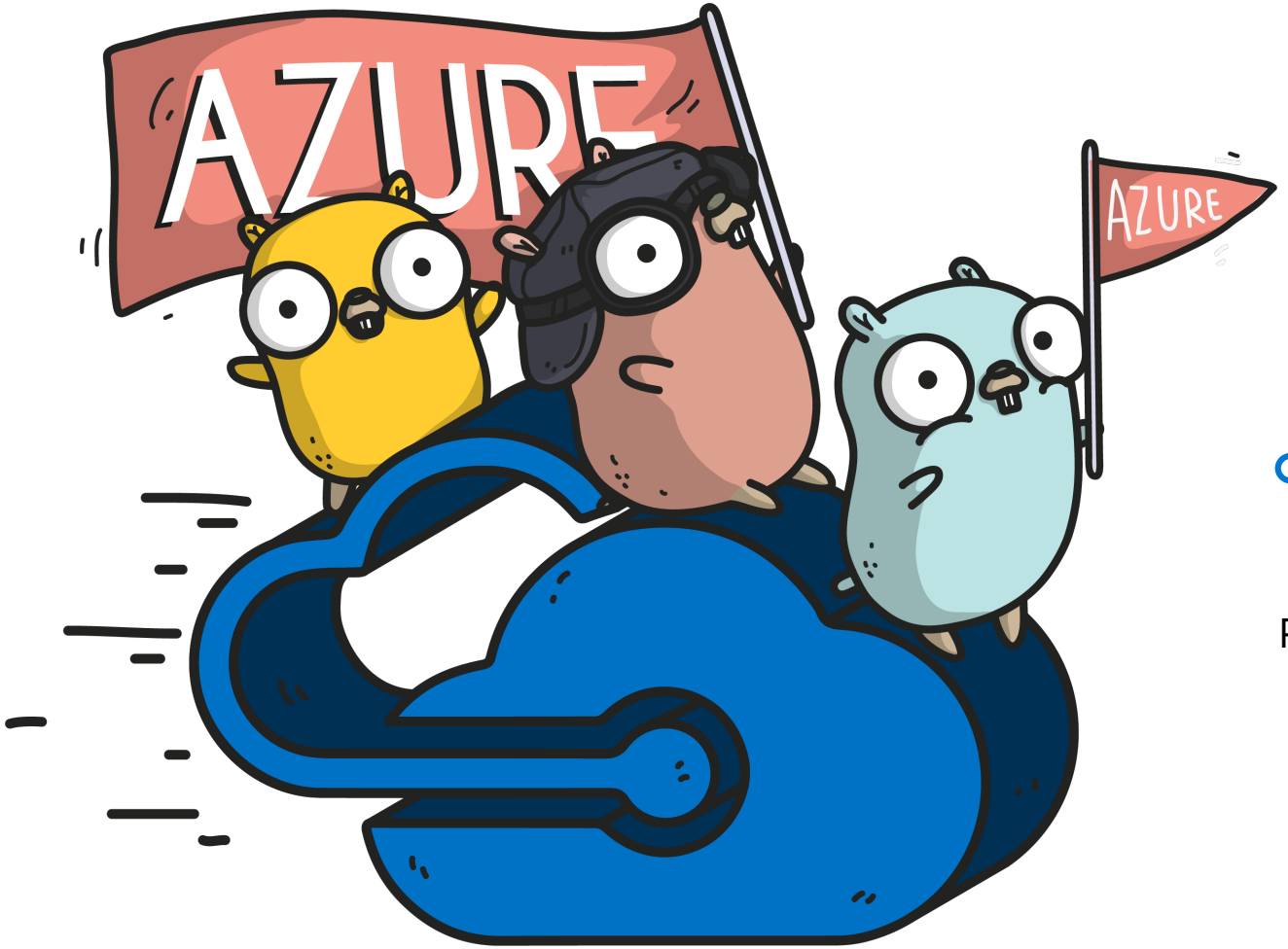
Olive.config as Service

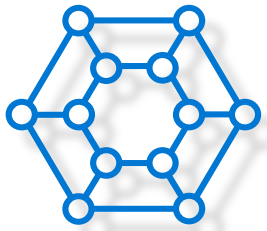


SLM with Edge Device

Demo

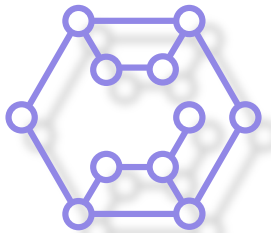

QuickTime Player File Edit View Window Help
Sat Mar 23 9:23 AM
Notebooks - Azure AI | Machine Learning Studio
https://ml.azure.com/fileexplorerAzNB?wsid=/subscriptions/396656ae-1e4b-4f9d-9a8a-a5fcb0296643/resourcegroups/AIG...
Azure AI | Machine Learning Studio
All workspaces
Home
Model catalog
Authoring
Notebooks
Automated ML
Designer
Prompt flow
Assets
Data
Jobs
Components
Pipelines
Environments
Models
Endpoints
Manage
Compute
Monitoring
Data Labeling
Linked Services PREVIEW
Lo Kinfey
Notebook
Files
Samples
Logs
Users
flow.dag.yaml
generate.py
init_onnx.py
run.ipynb
models
run_onnx_fp16.ipynb
run_onnx_int4.ipynb
Microsoft Remote Desktop
Search
Favorites
Recents
Applicati...
Desktop
Documents
Downloads
iCloud
QuickTim...
iCloud Dri...
Shared
Locations
lokinfey's...
OneDrive
Tags
Blue
Desktop
AOAI
Fabric
Screen Recor....22.18 AM.mov
Cancel
Open
2024-03-23 00:35:49 +0000 22385 execution.bulk INFO Process name(ForkProcess-2:1)-Process id(22495)-Line number(0) completed.
2024-03-23 00:35:50 +0000 22385 execution.bulk INFO Finished 1 / 1 lines.
2024-03-23 00:35:50 +0000 22385 execution.bulk INFO Average execution time for completed lines: 82.01 seconds. Estimated time for incomplete lines: 0.0 seconds.
======= Run Summary =======
Run name: "variant_0_20240323_003425_715346"
Run status: "Completed"
Start time: "2024-03-23 00:34:25.715314"
Duration: "0:01:27.418359"
Output path: "/home/azureuser/.promptflow/.runs/variant_0_20240323_003425_715346"
<promptflow._sdk.entities._run.Run at 0x7f3c955d6c70>





Phi-3-mini-V

(3.8B + 0.3B)



Phi-3-small

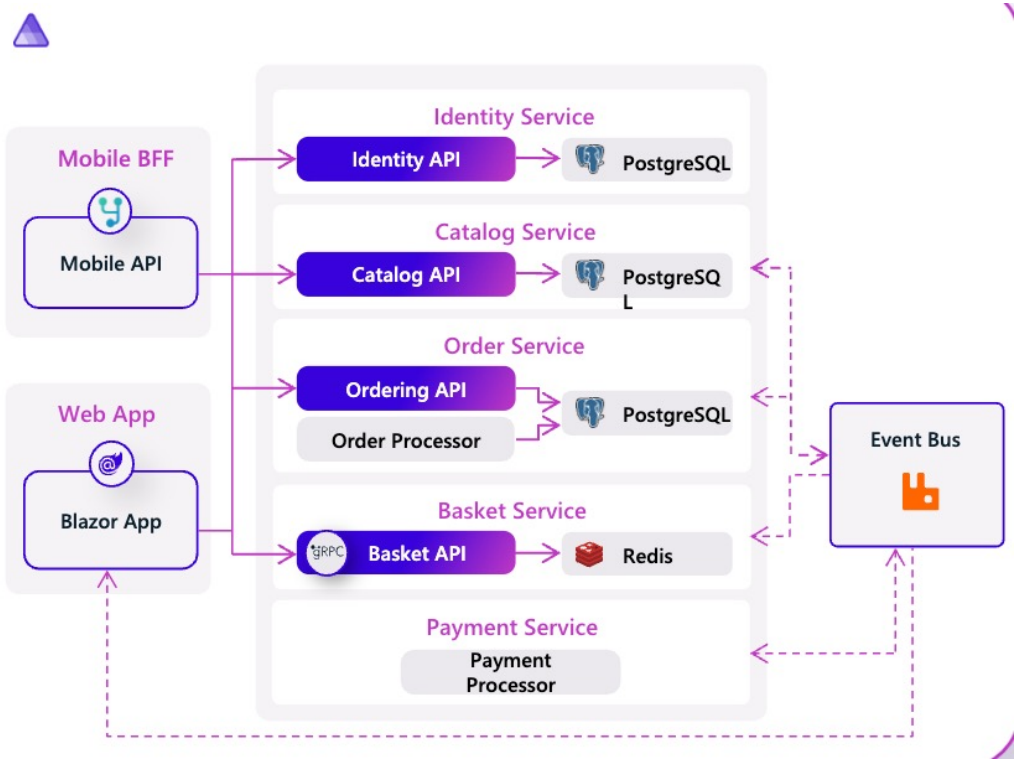
(7B)

Phi-3-mini

(3.8B)

Begin to talk about **Cloud Native**

What's .NET Aspire

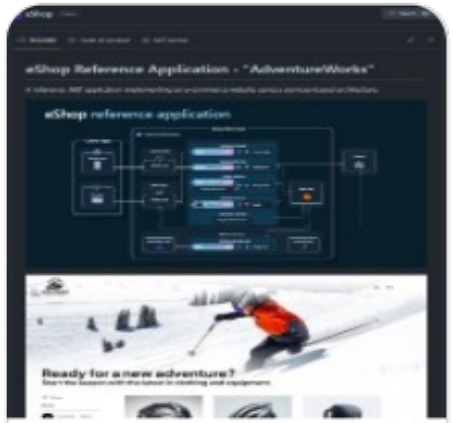


.NET Aspire is designed to improve the experience of building .NET cloud-native apps. It provides a consistent, opinionated set of tools and patterns that help you build and run distributed apps. .NET Aspire is designed to help you with:

- **Orchestration**: .NET Aspire provides features for running and connecting multi-project applications and their dependencies for local development environments.
- **Components:** .NET Aspire components are NuGet packages for commonly used services, such as Redis or Postgres, with standardized interfaces ensuring they connect consistently and seamlessly with your app.
- **Tooling:** .NET Aspire comes with project templates and tooling experiences for Visual Studio, Visual Studio Code, and the dotnet CLI to help you create and interact with .NET Aspire projects.

AI@.NET Aspire

Building Generative AI apps with .NET 8



Discover end-to-end sample applications and documentation

Learn

Semantic Kernel simplifies consuming AI components

Build

OpenAI

drant

Milvus

Growing AI ecosystem

Ecosystem

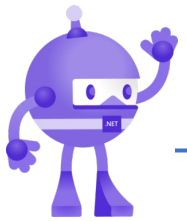
.NET Aspire and azd streamline deployment

Deploy

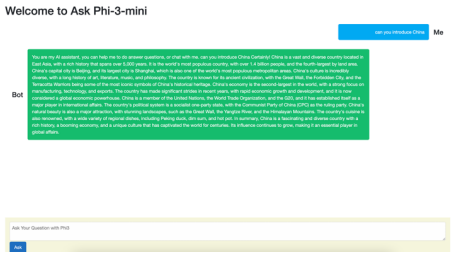
.NET Aspire + Azure + Semantic Kernel Easy monitoring and observability

Monitor

https://devblogs.microsoft.com/dotnet/build-gen-ai-with-dotnet-8/

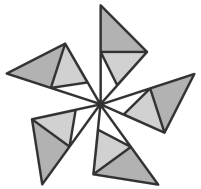


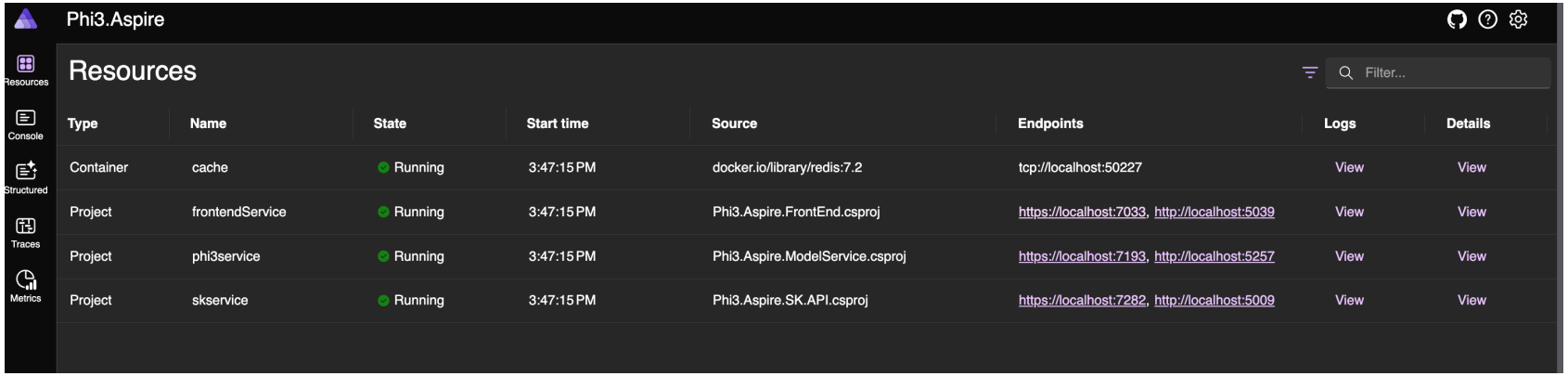
Ask



question

connect



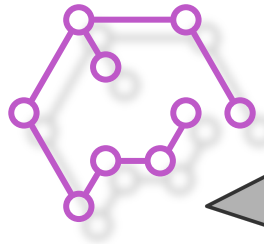


Blazor

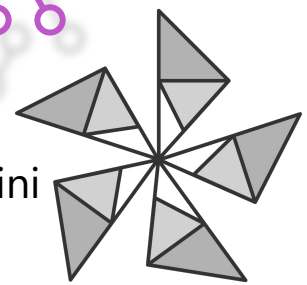
ChatUI Service

Semantic Kernel

AI Orchestration-Service

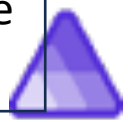


Phi-3-mini
(3.8B)



ONNX Runtime

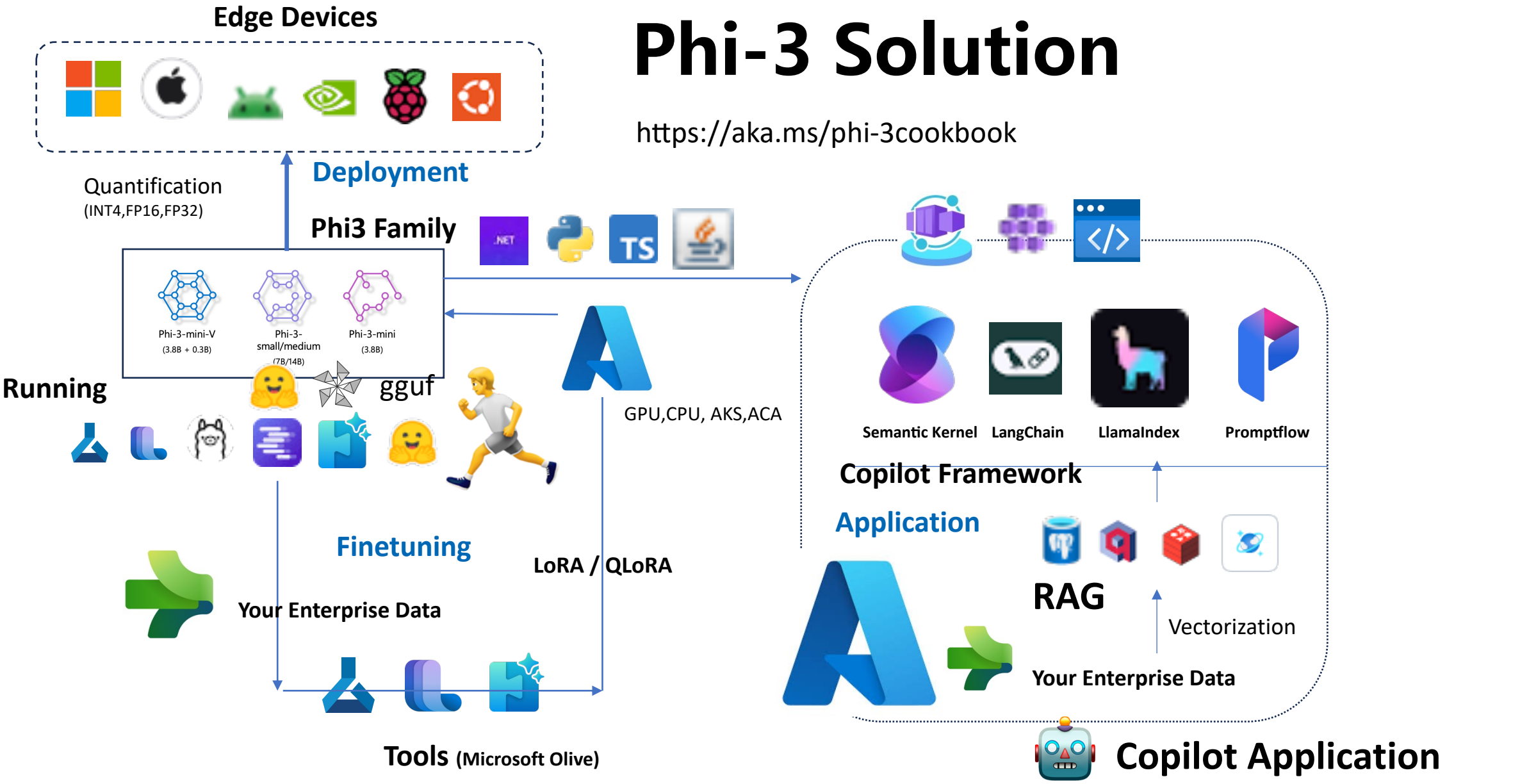
Phi-3-mini-Service



Demo : .NET Aspire with Phi3-mini

Phi-3 Solution

https://aka.ms/phi-3cookbook



Microsoft

THANK YOU

Phi-3 Cookbook