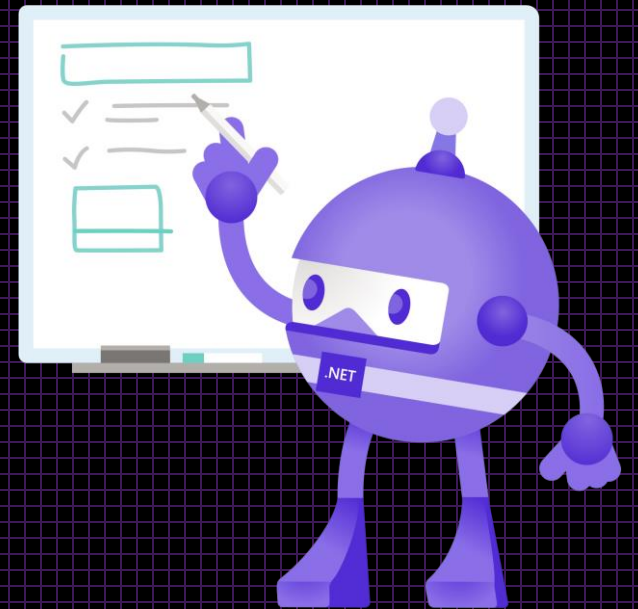


# LOCAL REASONING MODEL DEPLOYMENT WITH AUTOGEN BETWEEN OLLAMA AND LM STUDIO

Session By **Charunthon Limseelo**  
Microsoft Learn Student Ambassador  
Student Tech Community Lead Of Thailand

Slide Presentation Designed By Poonyada Phanitpotchamarn

@boatchrnthn





# Charunthon Limseelo (Boat)

---

- Microsoft Learn Student Ambassadors (Thai Student Tech Lead)
- Data Engineer Intern At Seven Peaks Software – Klong Toei, Bangkok
- Third Year Student At Computer Engineering (International Program) King Mongkut's University Of Technology Thonburi



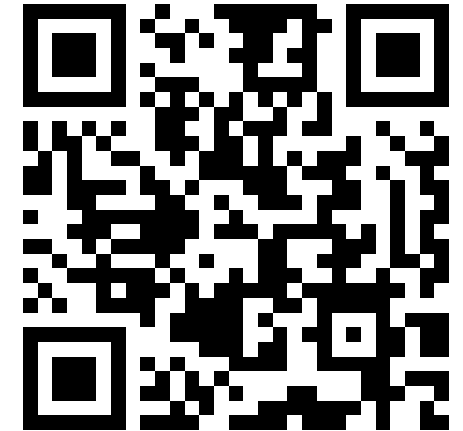
Charunthon Limseelo



@boatchrnthn



Charunthon Limseelo



# Acknowledgement To All Specialists on Inspiring Ideas

Engineering, Research-Based, and Practical Leads of AI



Business, Financial, Commercial-based, and Daily Users of AI



# CONTENTS

- Getting Into the Reasoning Models
- Open-Source Reasoning LLMs
- Local Model Providers
- Demonstration on Operating LM Studio and AutoGen
- Key Takeaways and **FOSSASIA** (how to dev local model based on sustainability)

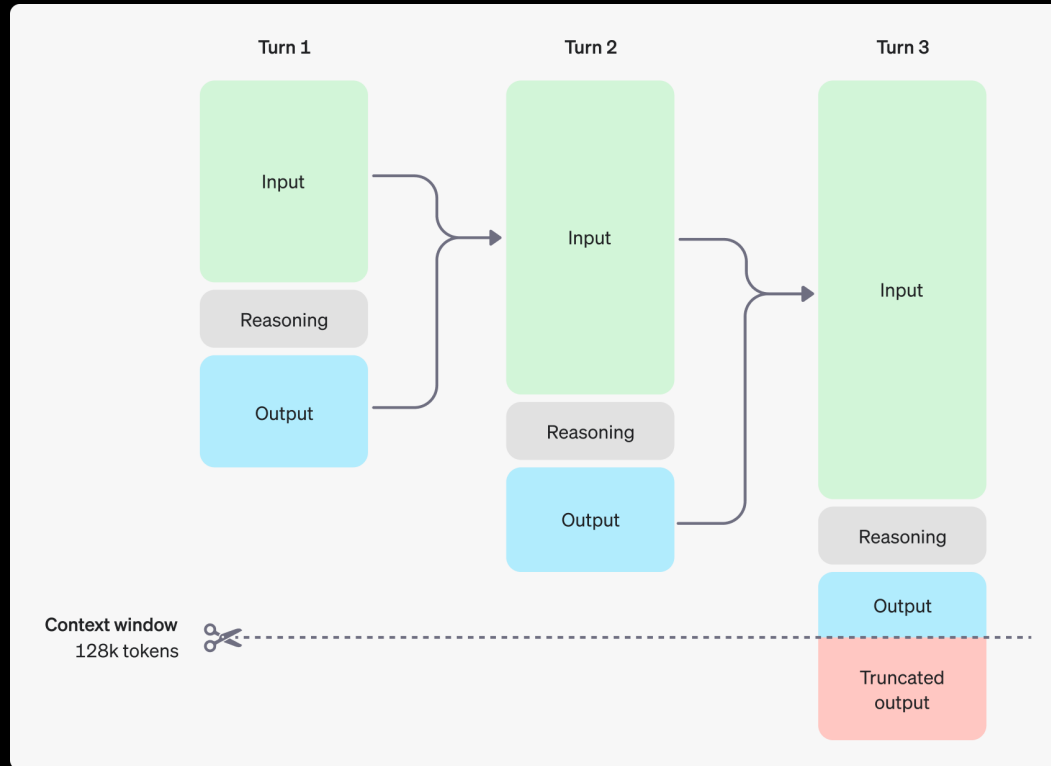
# 01

## Getting Into the Reasoning Models

Way making the LLMs smarter on generating answers

# Reasoning Models

Explore advanced reasoning and problem-solving models



- Reasoning models are new large language models trained with reinforcement learning to perform complex reasoning. Reasoning models think before they answer, producing a long internal chain of thought before responding to the user. Reasoning models excel in complex problem solving, coding, scientific reasoning, and multi-step planning for agentic workflows

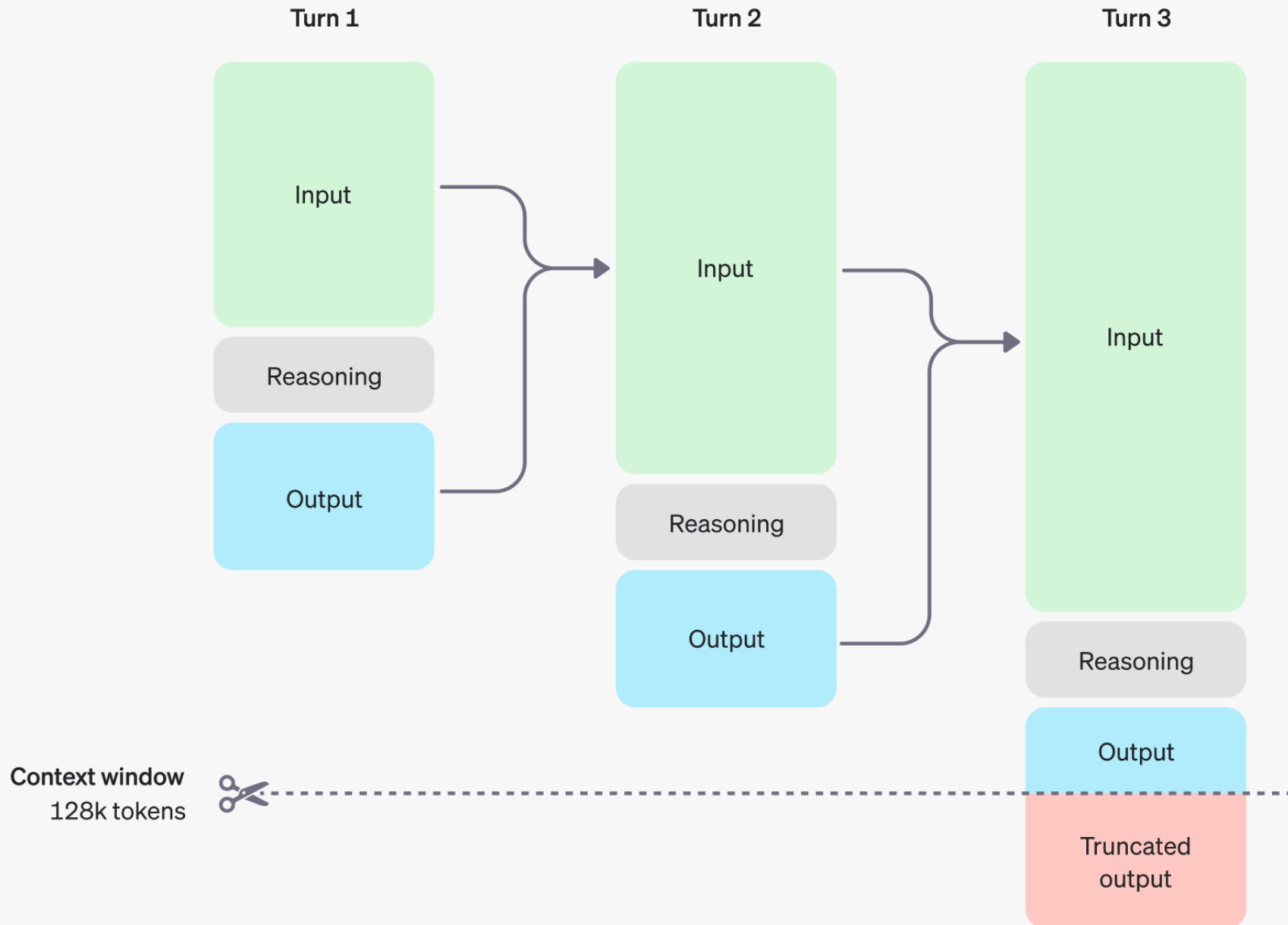
*Reference: OpenAI*

# How does it work?

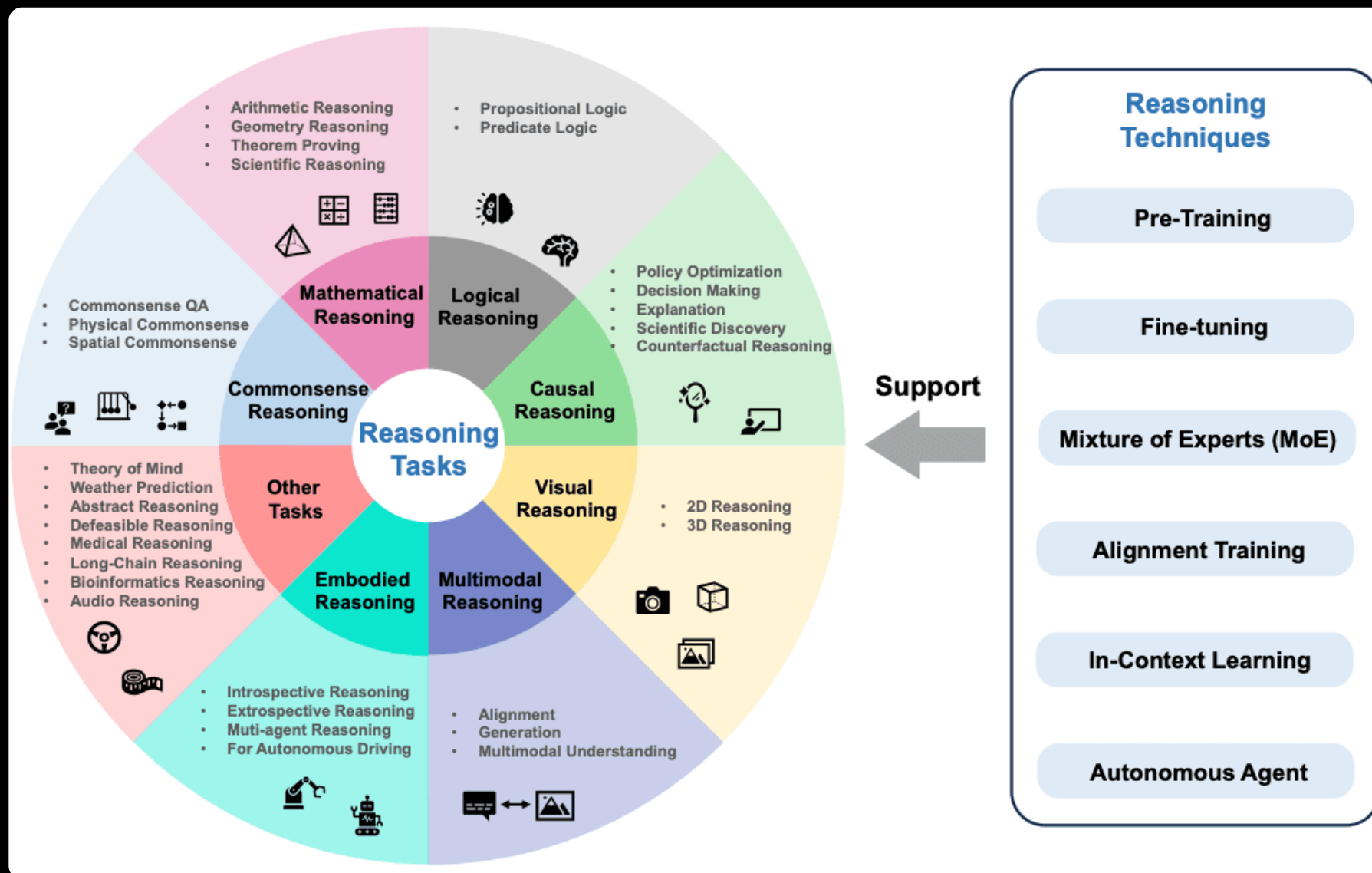
Reasoning models introduce reasoning tokens in addition to input and output tokens. The models use these reasoning tokens to "think", breaking down their understanding of the prompt and considering multiple approaches to generating a response. After generating reasoning tokens, the model produces an answer as visible completion tokens, and discards the reasoning tokens from its context



Chain-of-Thought (CoT)  
based Finetuning



# Reasoning can be experts on any concepts.





# How It Differs From Normal LLMs?

## Large Language Models

- Best for Content Creation
- Simple Task Accuracy
- High Hallucination Rate

## Large Reasoning Models

- Best for Scientific Research
- Complex Task Accuracy
- Low Hallucination Rate



Please consider with the user of each task, along with the consideration of usage and pricing of the token from the model generation.

# 02

## Open-Source Reasoning LLMs

Selecting Other Models To Be More Reasoning and Lower Price?

# Open-Source Small Reasoning Models



Cost  
Effectiveness



Deployment  
Flexibility



Ultra-low  
Latency



Easier to  
Customize

# Use Cases of Small OSRMs



Latency bound scenarios where fast response times are critical



Cost constrained tasks and use cases



Resource constrained environments



Select tasks can see improved performance via fine-tuning (vs. large model out-of-box)

# Alternative Reasoning LLMs



**Llama 3.3 70B Instruct:** This model from Meta has strong **mathematical reasoning** and code generation capabilities. It has a 128k-token context window and is comparable to top closed-source models like GPT-4o.



**DeepSeek R1** specifically designed for complex reasoning tasks, like logical inference and problem-solving, with a focus on providing accurate and structured solutions, often considered comparable to OpenAI's o1 model while being more cost-effective; it stands out due to its open-source nature and accessibility for both commercial and personal use.



**Microsoft's Phi-4 mini** outperforming models of similar and larger sizes across various benchmarks in language, reasoning, coding, and math but seems overfitted when we looked at MMLU.

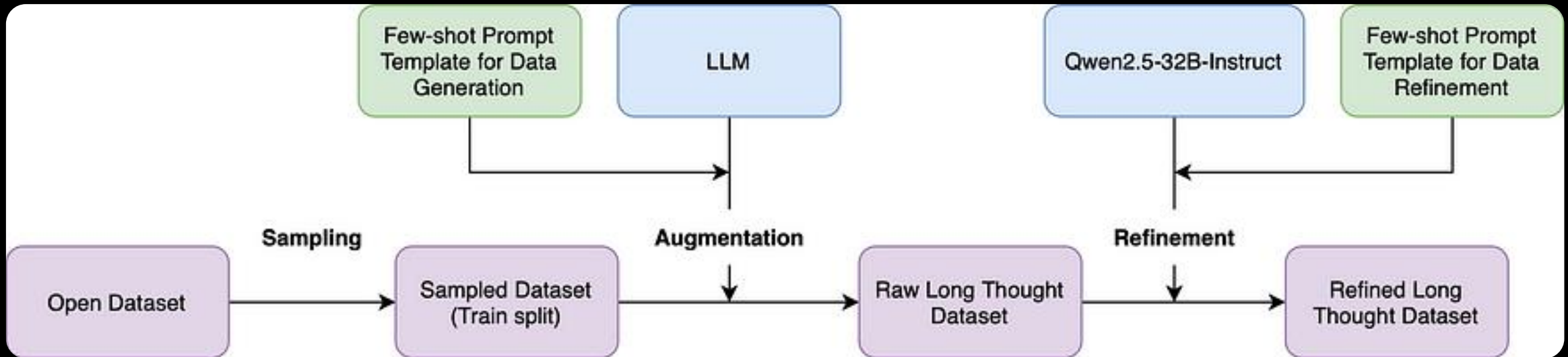


**Typhoon T1** is an open-source Thai reasoning model that uses large language models (LLMs) to generate a chain of thought before answering a question. It was developed by SCB 10X.

# Expanding SCB 10x Typhoon T1



**Typhoon T1** is an open-source Thai reasoning model that uses large language models (LLMs) to generate a chain of thought before answering a question. It was developed by SCB 10X.



Data Preparation Flow – SCB 10X

<thoughts>

<plan>

<step order="1">Understanding the scale relationship</step>

<step order="2">Setting up the proportion</step>

<step order="3">Cross-multiplying to solve for the model distance</step>

<step order="4">Calculating the final result</step>

<step order="5">Verifying the calculations</step>

</plan>

<step>

<title>Understanding the scale relationship</title>

<scratch\_pad>

The scale given is 1.5 cm to 50 m. This means that every 1.5 cm on the model corresponds to 50 meters in reality.

</scratch\_pad>

<summary>I'm clarifying the scale relationship to ensure accurate calculations.</summary>

<next\_step>Setting up the proportion</next\_step>

</step>

<step>

<title>Setting up the proportion</title>

<scratch\_pad>

Let  $(x)$  be the distance on the model (in centimeters). The proportion can be set up as follows:

$[$

$$\frac{1.5 \text{ cm}}{50 \text{ m}} = \frac{x \text{ cm}}{150 \text{ m}}$$

$]$

This represents the relationship between the model and the actual distance.

</scratch\_pad>

<summary>I've established the proportion based on the scale provided.</summary>

<next\_step>Cross-multiplying to solve for the model distance</next\_step>

</step>

...

</thoughts>



<response>

To find the distance between the two trees on the model, we use the scale provided.

### **\*\*Given:\*\***

- Scale: 1.5 cm corresponds to 50 m.
- Distance between the trees in reality: 150 m.

### **\*\*Step-by-step Calculation:\*\***

1. **\*\*Set up the proportion:\*\***

$$\left[ \frac{1.5 \text{ cm}}{50 \text{ m}} = \frac{x \text{ cm}}{150 \text{ m}} \right]$$

2. **\*\*Cross-multiply:\*\***

$$\left[ 1.5 \cdot 150 = 50 \cdot x \right]$$

Simplifying gives:

$$\left[ 225 = 50x \right]$$

3. **\*\*Solve for  $(x)$ :**

$$\left[ x = \frac{225}{50} = 4.5 \text{ cm} \right]$$

### **\*\*Final Answer:\*\***

The distance between the two trees on the model is **\*\*4.5 centimeters\*\***.

</response>

# 03

## Local Model Providers

Selecting Between Ollama and LM Studio, Which One Would I Choose?

# Ollama – Like Docker, But for LLMs

Created by Jeffrey Morgan



- Ollama is a lightweight, extensible framework for building and running language models on the local machine. It provides a simple API for creating, running, and managing models, as well as a library of pre-built models that can be easily used in a variety of applications.

```
ollama
+
Update Warp

base ~/Code/ecosystem
ollama run zephyr
eval count:      31/ token(s)
eval duration:   6.37296s
eval rate:      49.74 tokens/s
>>> what animals do not?
As I mentioned earlier, many animal species do not have clear and rigid social hierarchies among adult individuals in their wild populations. Some examples include:

1. Polar bears: They are solitary animals that do not live in groups with a social hierarchy.

2. Bald eagles: While they form pairs during mating season, there is no dominant leader or hierarchy within the pair or between pairs.

3. Sea otters: Although they form social groups, there is no hierarchy among them. The social structure is more cooperative and based on sharing resources.

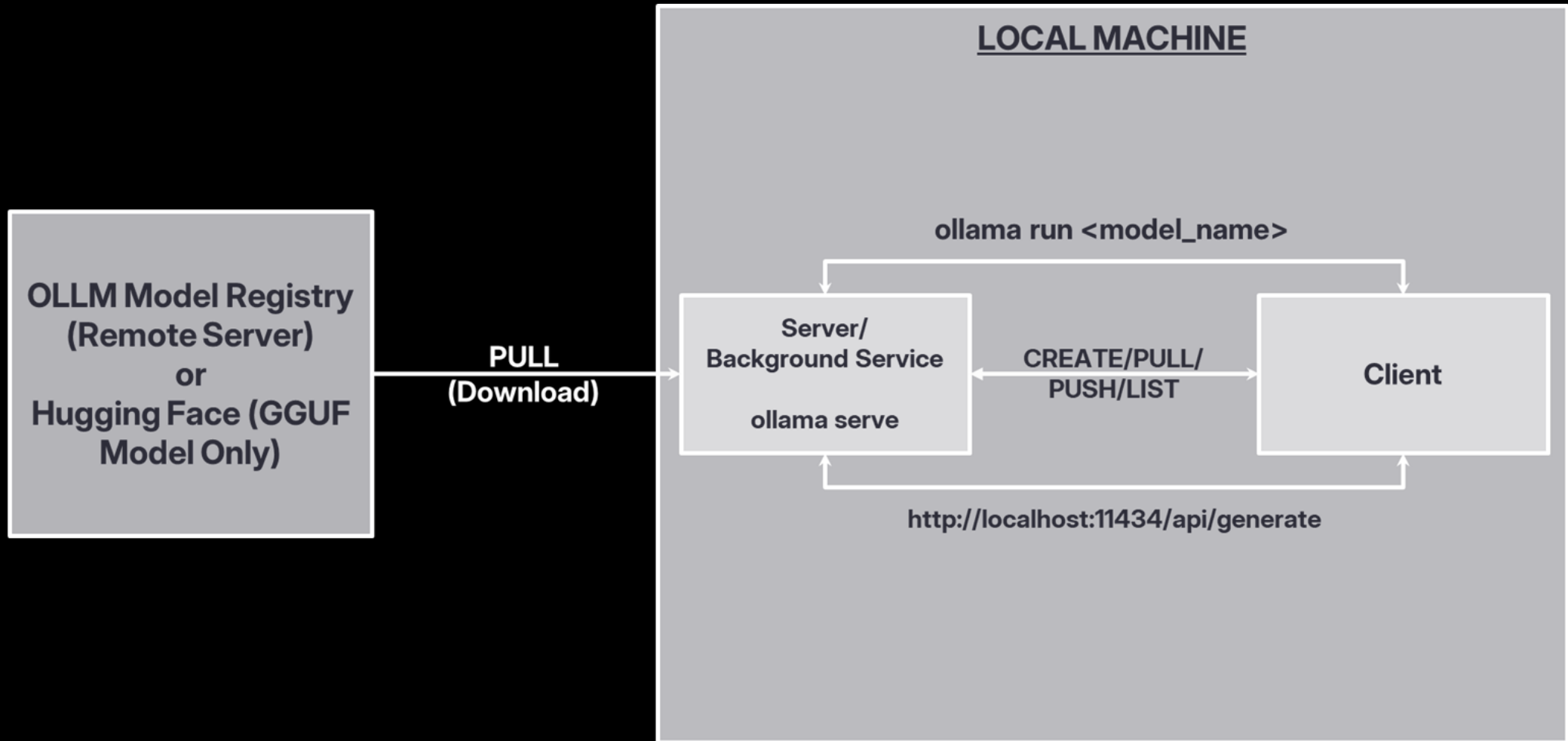
4. Tiger sharks: They are solitary predators that do not live in groups with a social hierarchy.

5. Orangutans: While they may form social groups led by a dominant female, there is generally less rigid and hierarchical structure compared to some other primates such as chimpanzees, gorillas, and bonobos.

These are just a few examples of animals that do not have clear social hierarchies among adult individuals in their wild populations. However, it's important to note that the presence or absence of social hierarchy can vary depending on factors such as species, environment, and context.

total duration: 8.592375916s
load duration:  1.528084666s
prompt eval count: 1169 token(s)
prompt eval duration: 2.050488s
prompt eval rate: 570.11 tokens/s
eval count:      255 token(s)
eval duration:   4.95492s
eval rate:      51.46 tokens/s
>>> end a message (/? for help)
```

# Inside's Architecture

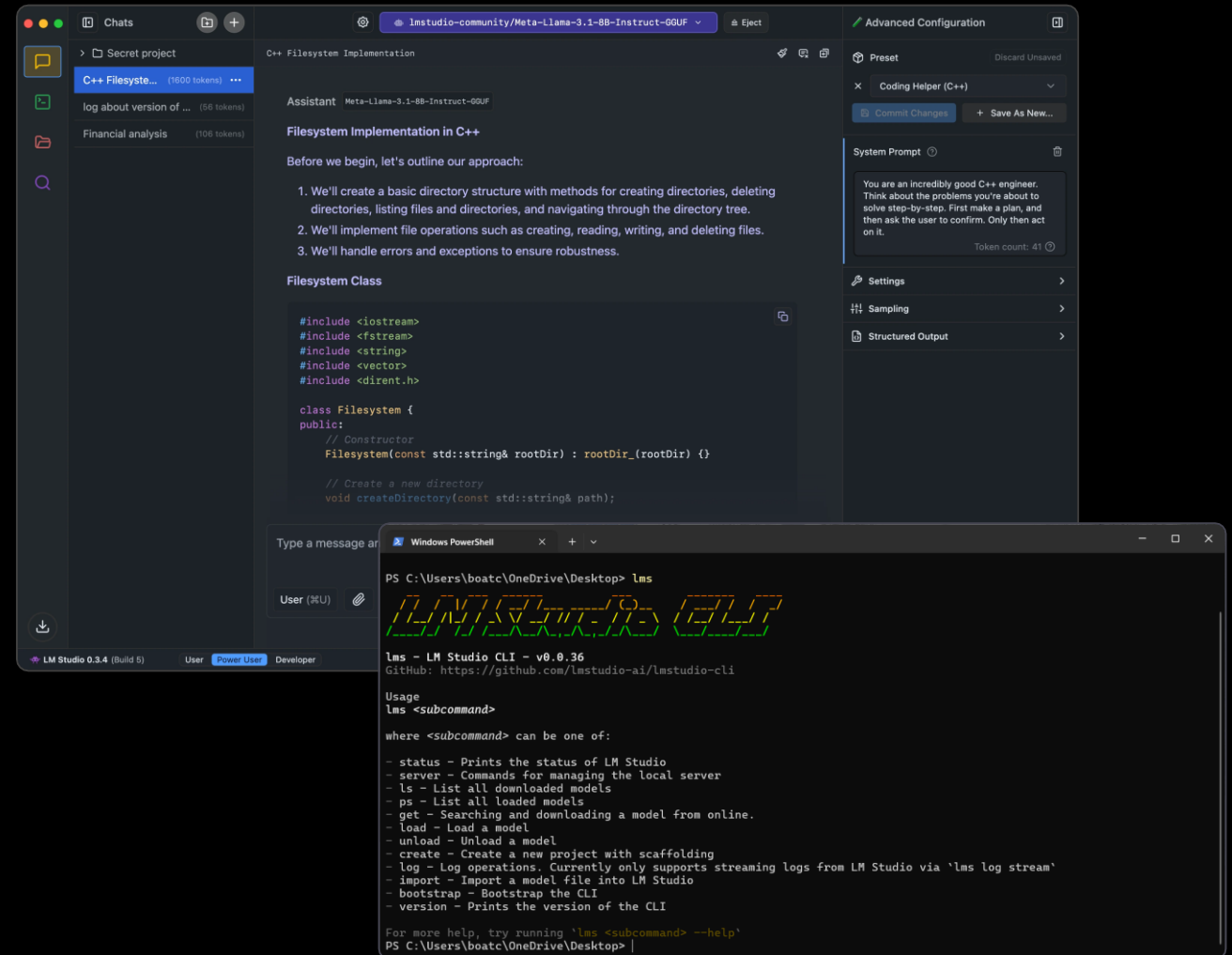


# LM Studio – Similar to Ollama, but have UI

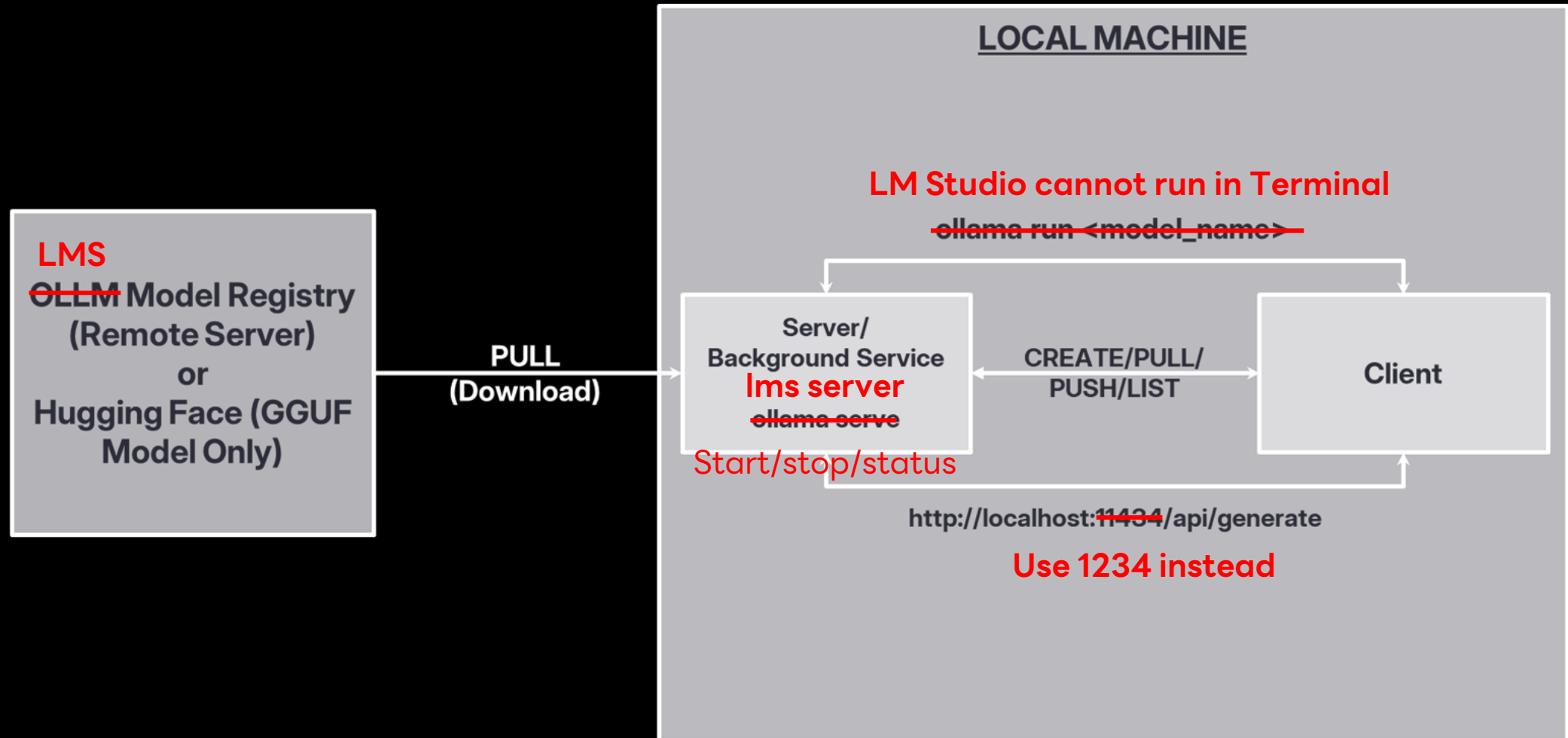
Created by Yagil Burowski



- LM Studio is a free, open-source desktop application that helps users run and customize Large Language Models (LLMs) locally. Search & download functionality (via Hugging Face 🤗)
- Two Software Parts – UI for users, and CLI for developers



# Inside's Architecture



# 04

## Demonstration on Operating Agents

Between Ollama and LM Studio, with **AutoGen**

# What is AI Agent and Multi-Agent AI System?

- An **AI agent** is an autonomous entity that can perceive its environment, reason about it, and take actions to achieve its goals. It functions through perception, reasoning, and action components.
- A **Multi-Agent AI System** is a system composed of multiple AI agents that interact and collaborate to achieve shared or individual goals. It exhibits decentralization, emergent behavior, collaboration, and competition. Applications include simulation, robotics, game AI, and e-commerce.





**What is AutoGen?**

# AutoGen: An Open-Source Programming Framework for Agentic AI



## Multi-Agent Conversation Framework

AutoGen provides multi-agent conversation framework as a high-level abstraction. With this framework, one can conveniently build LLM workflows.



## Easily Build Diverse Applications

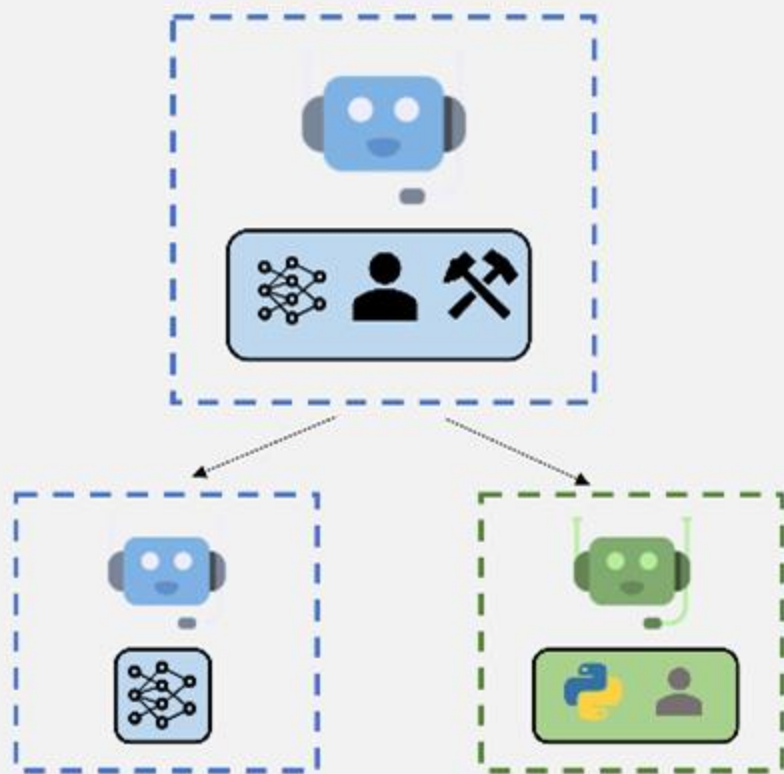
AutoGen offers a collection of working systems spanning a wide range of applications from various domains and complexities.



## Enhanced LLM Inference & Optimization

AutoGen supports enhanced LLM inference APIs, which can be used to improve inference performance and reduce cost.

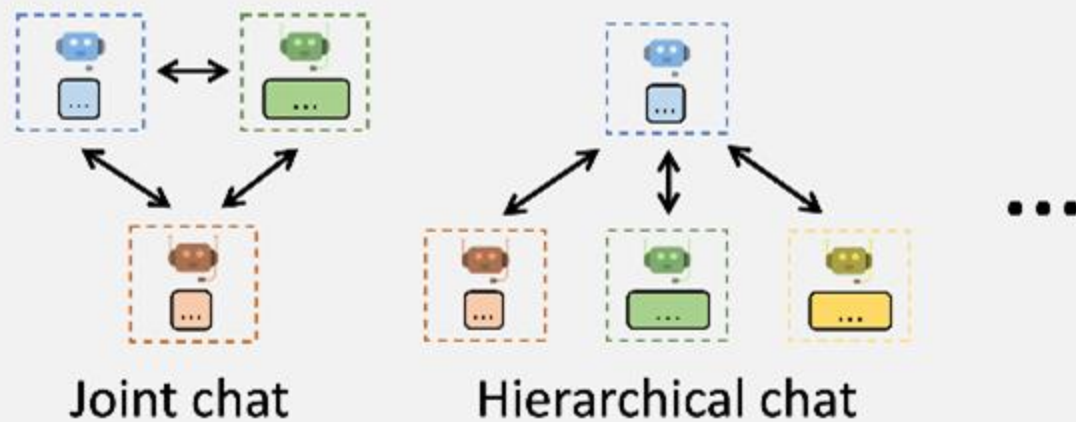
## Conversable agent



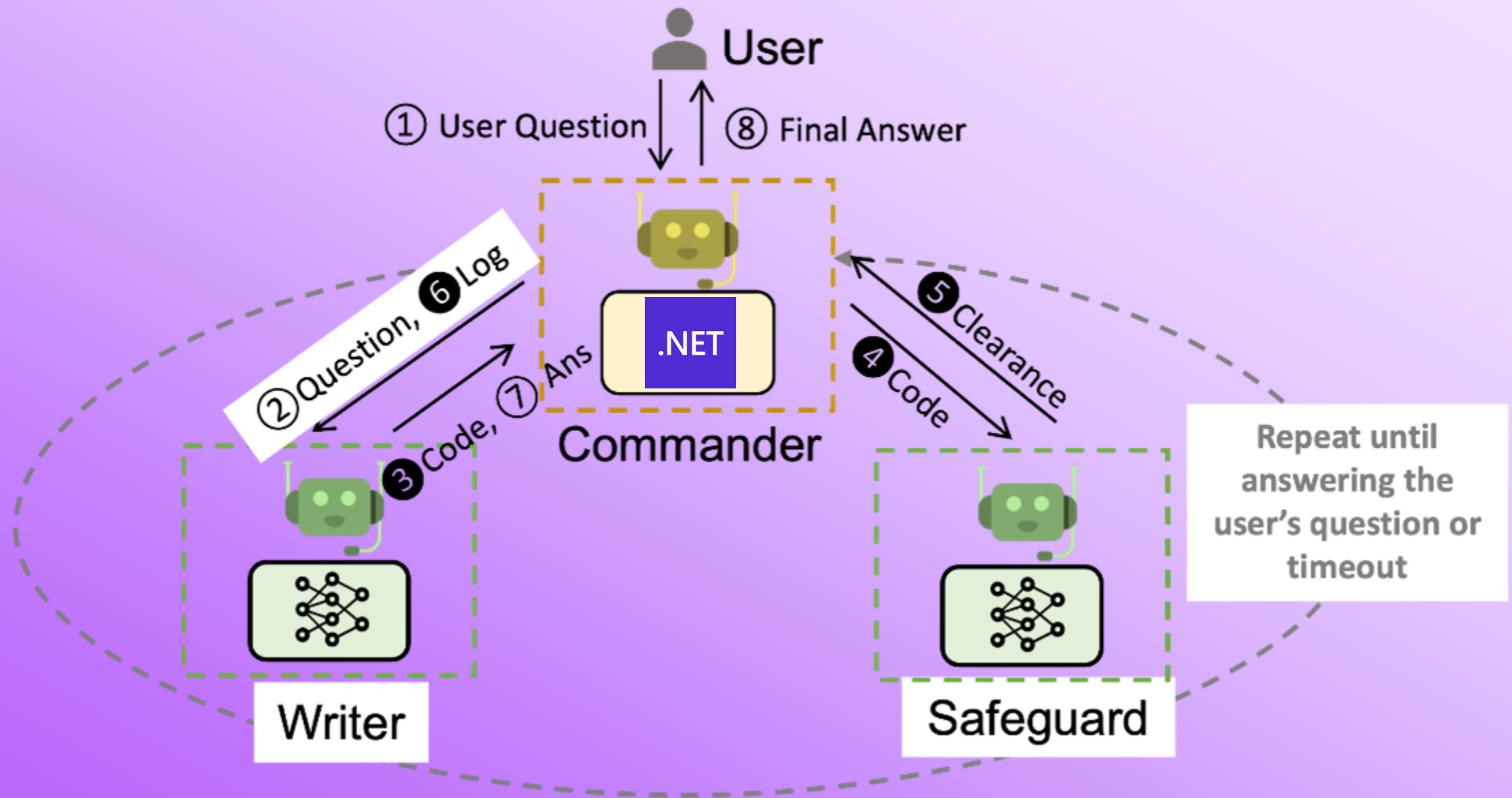
## Agent Customization



## Multi-Agent Conversations



## Flexible Conversation Patterns



# Some Syntaxes

Running Ollama Model with AutoGen for .NET

# Pre-requisite

**\*Warning:** NUGET packages always have a problem for installation, please run 'dotnet new nugetconfig' and then install the necessary packages for your project

First, install the AutoGen.Ollama package using the following command:

```
dotnet add package AutoGen.Ollama
```

Step 2: Add using statement

```
using AutoGen.Core; using AutoGen.Ollama.Extension;
```

# Create OllamaAgent: Text Based

```
using var httpClient = new HttpClient()  
{  
    BaseAddress = new Uri("http://localhost:11434"),  
};
```

```
var ollamaAgent = new OllamaAgent(  
    httpClient: httpClient,  
    name: "ollama",  
    modelName: "llama3:latest",  
    systemMessage: "You are a helpful AI assistant")  
    .RegisterMessageConnector()  
    .RegisterPrintMessage();
```

```
var reply = await ollamaAgent.SendAsync("Can you write a piece of C# code to  
calculate 100th of fibonacci?");
```

llama  
11434



# Some Syntaxes

Running LM Studio CLI with AutoGen for .NET



# Pre-requisite

**\*Warning:** NUGET packages always have a problem for installation, please run 'dotnet new nugetconfig' and then install the necessary packages for your project

First, install the AutoGen.Ollama package using the following command:

```
dotnet add package AutoGen.LMStudio
```

Step 2: Add these statement on csproj file

```
<ItemGroup>  
  <PackageReference Include="AutoGen.LMStudio" Version="AUTOGEN_LMSTUDIO_VERSION" />  
</ItemGroup>
```

# Create LMS Agent: Text Based

```
using System.ClientModel;
using AutoGen.Core;
using AutoGen.OpenAI;
using AutoGen.OpenAI.Extension;
using OpenAI;

var endpoint = "http://localhost:1234";
var openaiClient = new OpenAIClient(new ApiKeyCredential("api-key"), new OpenAIClientOptions
{
    Endpoint = new Uri(endpoint),
});

var lmAgent = new OpenAIChatAgent(
    chatClient: openaiClient.GetChatClient("<does-not-matter>"),
    name: "assistant")
    .RegisterMessageConnector()
    .RegisterPrintMessage();

await lmAgent.SendAsync("Can you write a piece of C# code to calculate 100th of fibonacci?");
```

# You can download the resources here...

## chrnthnkmutt/ **slm\_dotnet\_experiment**

This repository is used to make the experiment of using small language models and small reasoning models, in .NET framework...



Contributor



Issues



Star



Forks



You can download the resources here...



# 05

## Key Takeaways and FOSSASIA

What do we have learnt from this session....?

Ollama

LM Studio



**But depends on each users**

# Why I use reasoning models for making agents?

- **Multi-step problem solving:** Unlike simpler models that might only provide a single answer, reasoning models can handle complex tasks that require multiple steps, planning, and decision-making, making them well-suited for scenarios where a series of actions are needed to achieve a goal.
- **Contextual understanding:** Reasoning models can incorporate context and consider various factors when making decisions, allowing them to adapt to changing situations and respond appropriately to nuanced information.
- **Explainable reasoning:** Many reasoning models can provide a "chain of thought" explaining their reasoning process, which enhances transparency and allows for debugging or fine-tuning the agent's behavior.
- **Adaptability to new situations:** By reasoning through problems, agents can handle unexpected situations and adjust their strategies based on new information, making them more versatile in real-world applications.
- **Collaboration potential:** Reasoning models can be used to build agents that can effectively collaborate with other agents or humans, coordinating actions and sharing information to achieve a common goal.

# Examples of how reasoning models are used in agents:

- **Customer service agents:** Reasoning models can analyze customer queries, understand the context, and provide helpful responses, even if the questions are complex or ambiguous.
- **Financial planning agents:** These agents can use reasoning to analyze a user's financial situation, recommend investment strategies, and adapt to market changes.
- **Medical diagnosis support agents:** By reasoning through patient symptoms and medical records, these agents can provide doctors with potential diagnoses and treatment options.



# But think it carefully, before using it!



February 26 at 1:03 PM ·



หากคุณใช้ model deep thinking ตอบคำถามบ่อยๆ คุณจะมีส่วนทำให้โลกร้อนขึ้นนะครับ 😂



If you frequently use Deep Thinking Model, you might more globally warming

1 comment



Like



Comment

Most relevant ▼



คำถาม: 2+2 ได้เท่าไร

ใช้ไฟหนักกว่าเครื่องคิดเลขร้อยเครื่อง 😂

1d

Haha

Reply

Edited

3



I'm just asking, "What is 2+2?" but use the compute engine more than a hundred physical calculators 😂

# All Models Are Not Same in The Usage!

Try To Use It Sustainably and Consider on Energy Consumption!

# Thank you for your participation

Hope we could see each other again  
next time! 🙌

Follow me at @boatchrnthn on  
Social Media and @chrnthnkmutt on  
GitHub



## FOSSASIA SUMMIT 2025

True Digital Park, Bangkok  
summit.fossasia.org

March 13-15



การประชุมเทคโนโลยีโอเพนซอร์สชั้นนำของเอเชีย

### Charunthon Limseelo

Microsoft Learn Student Ambassador  
King Mongkut University of  
Technology Thonburi (KMUTT)

Small Language Models: The New  
Frontier of Developing Open-source  
Generative AI to Sustainability Goals