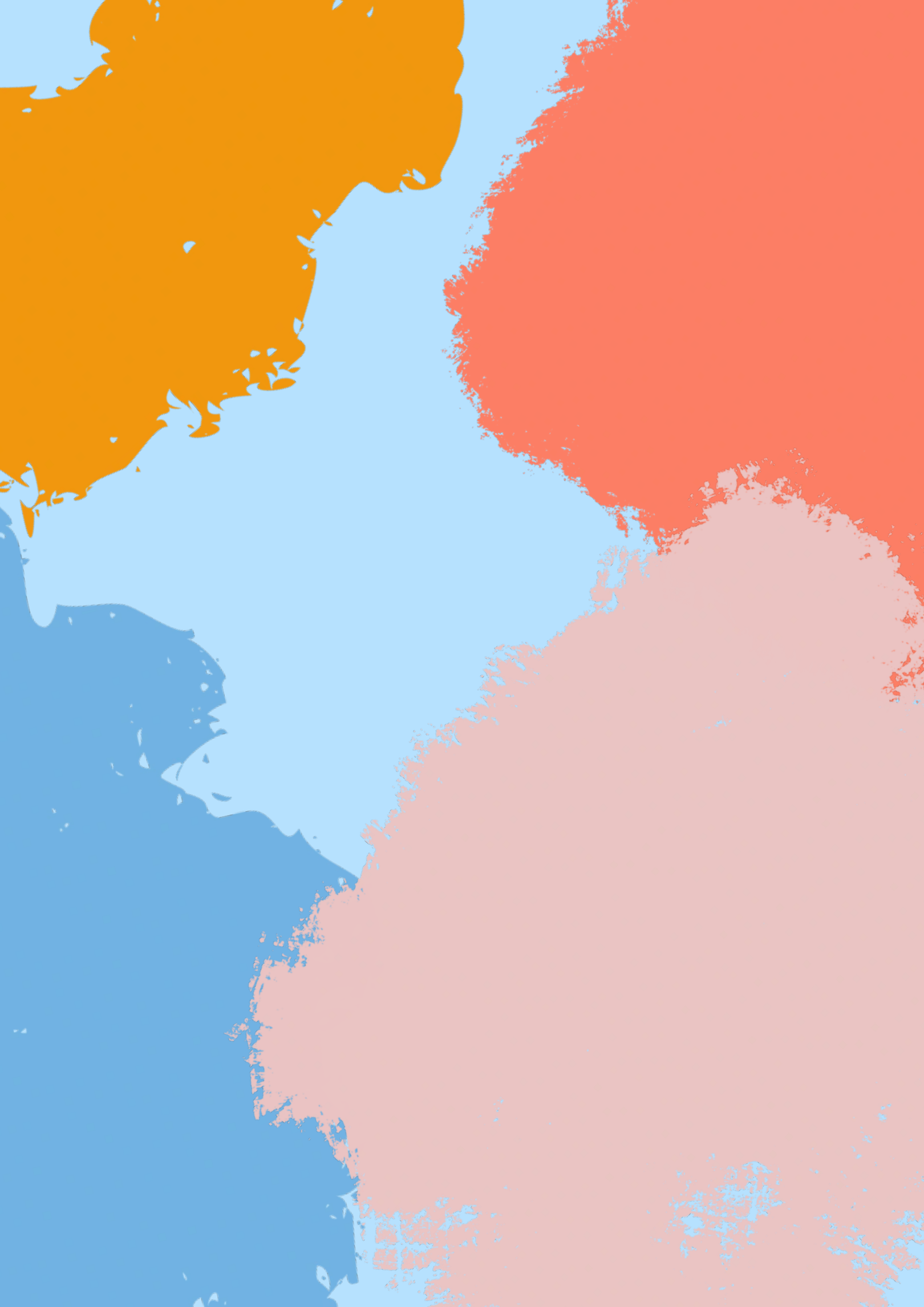


# AutoGen

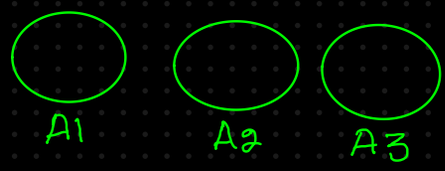
A framework for building AI agents and applications

AG

0.4 version

What is autogen?

a framework for building multi agents AI system.




Why does it matter?

Autogen lets multiple agents specialize and work together as a unit

team of superheroes

Autogen v0.4

async foundation

faster and more flexible for complex tasks.

layered API design

Agent chat API

Quick, task focused apps

Core API

low level control

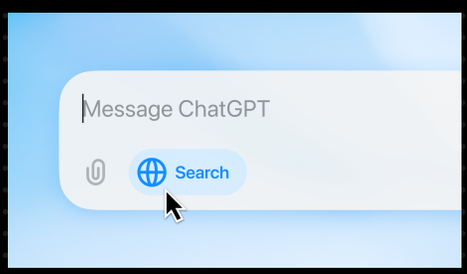
Better Observability

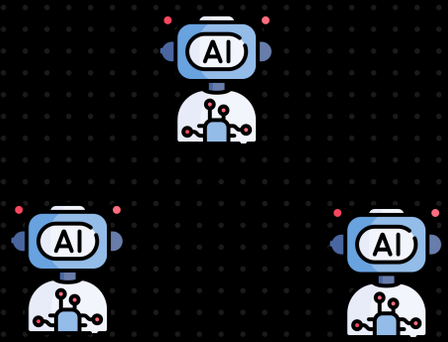
# AutoGen

A framework for building AI agents and applications

AutoGen is a comprehensive framework developed by Microsoft for building AI agents and applications. It facilitates the creation of both single and multi-agent systems capable of complex interactions and workflows.

writing blogs
debugging
analyzing

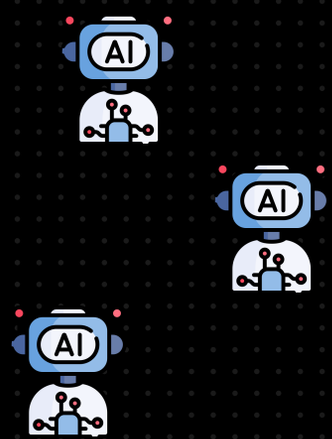



Autogen is a framework developed by microsoft to create multi-agent AI systems.

2023
|
2025 v0.4

Single AI
Lone worker



Why it matters?

Smarter & scalable systems

flexibility ✓

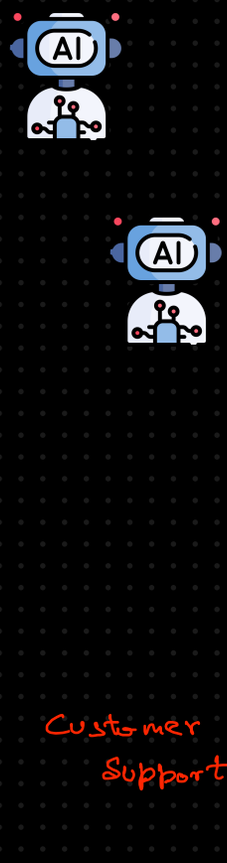
Microsoft AI ecosystem

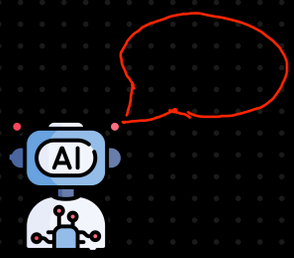
- Azure
- Semantic Kernel

# Autogen v0.4

early 2025

1. async foundation
2. Layered design
   - Core
   - Agent Chat
3. Better Observability

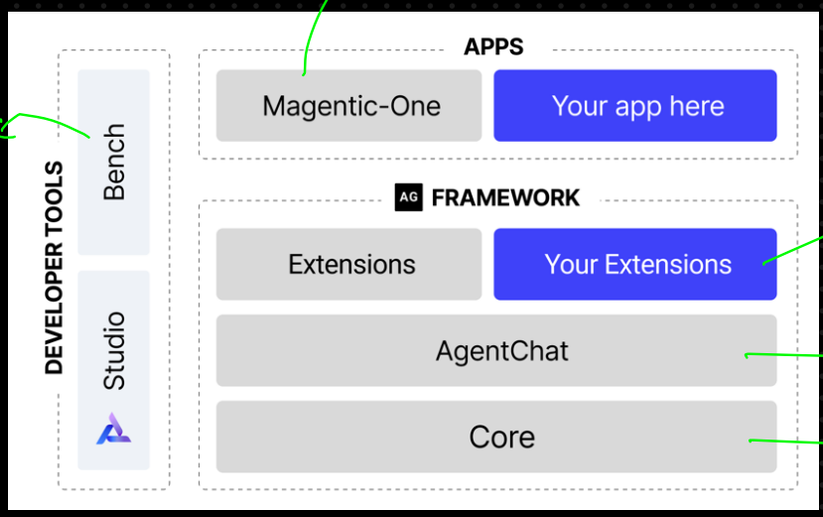



Autogen vs AG2

# Setting Up Microsoft Autogen in VS Code

# Autogen Architecture




pre built application

testing ground

adding flexibility

high level layer

foundation
→ asynch
→ event driven

# Autogen v0.4 vs Other AI Frameworks

Lang Chain, CrewAI, microsoft kernel, llama Index

| | Autogen AG | Lang Chain 🦜🔗 |
|---|---|---|
| Architecture | event-driven async | Chain-based sequential execution |
| multi agent Capabilities | ✓ | better for a single agent workflow |
| Observability | ✓ | depth missing |
| Scalability | async foundation handling complex agent network efficiently | synch. design struggles with large scale |
| Use case Suitability | Collaborative AI /team | data processing / Q-A tools |

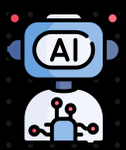
| | Autogen AG | Crew AI |
|---|---|---|
| Flexibility | Core Agent-chat | predefined workflows, less flexible for customization |
| Customization | event system & we can do tweaks | template-based approach |
| Maturity | matured | newer & growing |
| Learning Curve | requires bit more learning | Simpler |
| Use Case | complex projects | straightforward team task |

| | Autogen AG | Semantic Kernel |
|---|---|---|
| Focus | multi agent collaboration | integrate AI into existing apps |
| Integration | emphasizes standalone agent network | embedding AI into systems |
| Programming Model | event-driven | function-composition paradigm. |
| Memory managment | context management is sophisticated | skill based memory |
| Use Case | multi agents people | in-app specific AI enhancement |

| | Autogen AG | Llama Index |
|---|---|---|
| Primary Purpose | agents for task & conversation | Knowledge retrieval and document integration |
| Data Handling | focused on agent interaction | excels with structured or unstructured data sources |
| Use Case | multi-agent or dynamic agent teams | Llama index knowledge based AI |

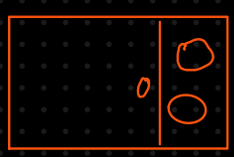
# Async Functionality in Python

→ Autogen uses async functionality for real-time agent interactions.

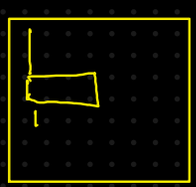



→ API

## Synchronous programming



Microwave



## Asynchronous programming

Coffee → 3

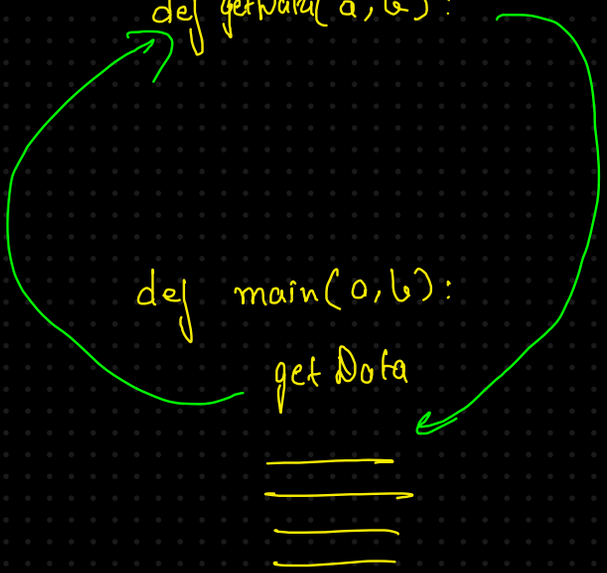
Toggle → 2

5 min

3 min

## Subroutine

```
def getData(a,b):

def main(a,b):
    getData
```



main → getData → completed → main

## Co-routine

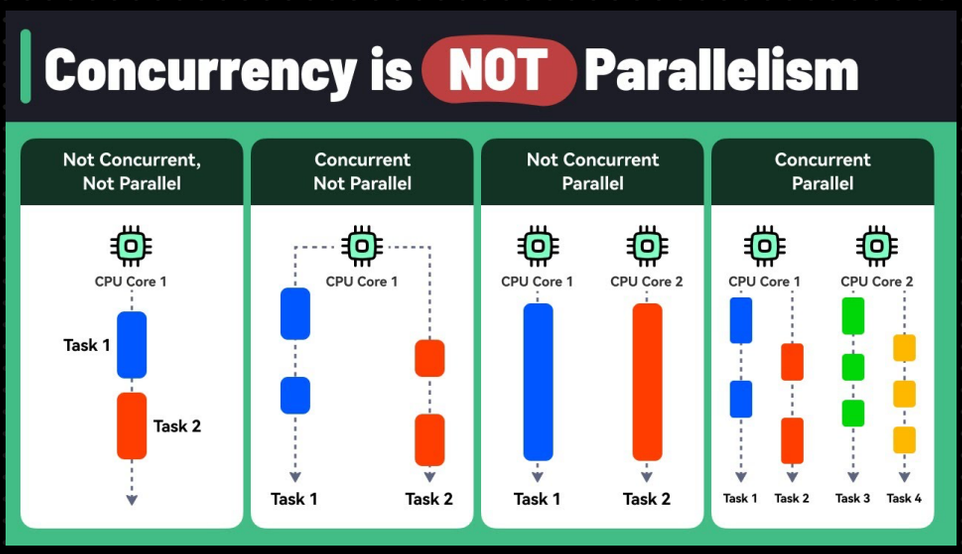
```
async def getData

def main():
    getData
```

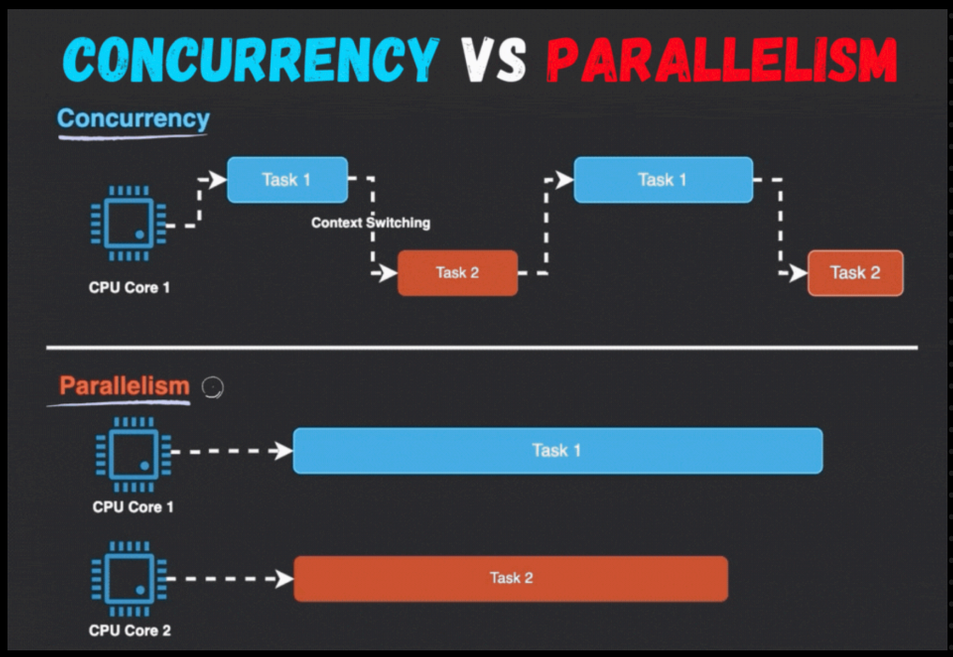
# Parallelism

running multiple tasks simultaneously using multiple threads or processes.

# Concurrency

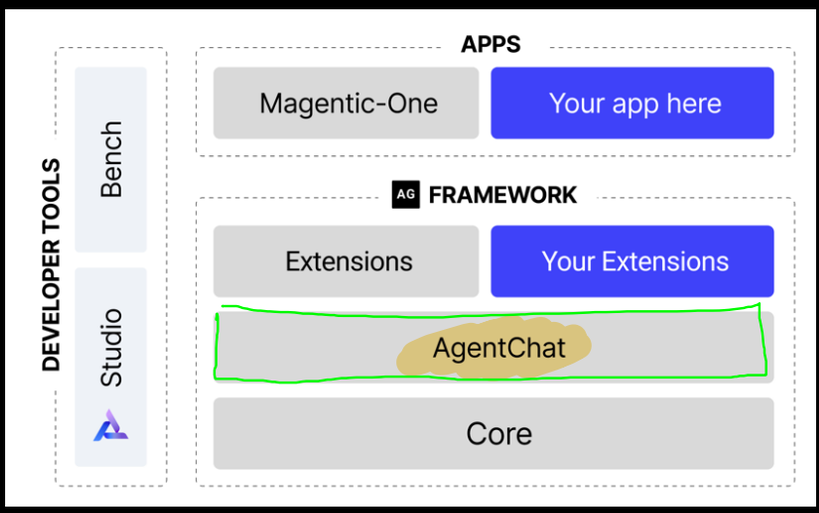
Managing multiple task that can start, run & finish with overlapping times.



# Async Functionality in Python - Practical

# Introduction to Agent Chat API




Agent chat provides a modern async powered way to handle agent conversation

What it does?

→ define agents

→ enable them to chat, collaborate and solve problems.

→ event driven i.e. response to task as they come

Key features

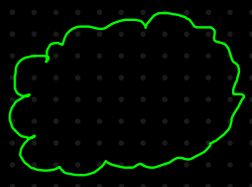
→ customization with prompts

→ connect it to LLM's

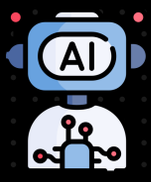
Why it Matters

→ makes dev fast and flexible

# Configuring LLMs ( OpenAI, Gemini, llama)



OpenAI
Gemini
llama




import / install library

```python
from autogen_ext.models.openai import OpenAIChatCompletionClient

openai_model_client = OpenAIChatCompletionClient(
    model="gpt-4o-2024-08-06",
    # api_key="sk-...", # Optional if you have an OPENAI_API_KEY
)
```

Class in our lib which connects to openAI models

gpt 4o , gpt 3.5-t

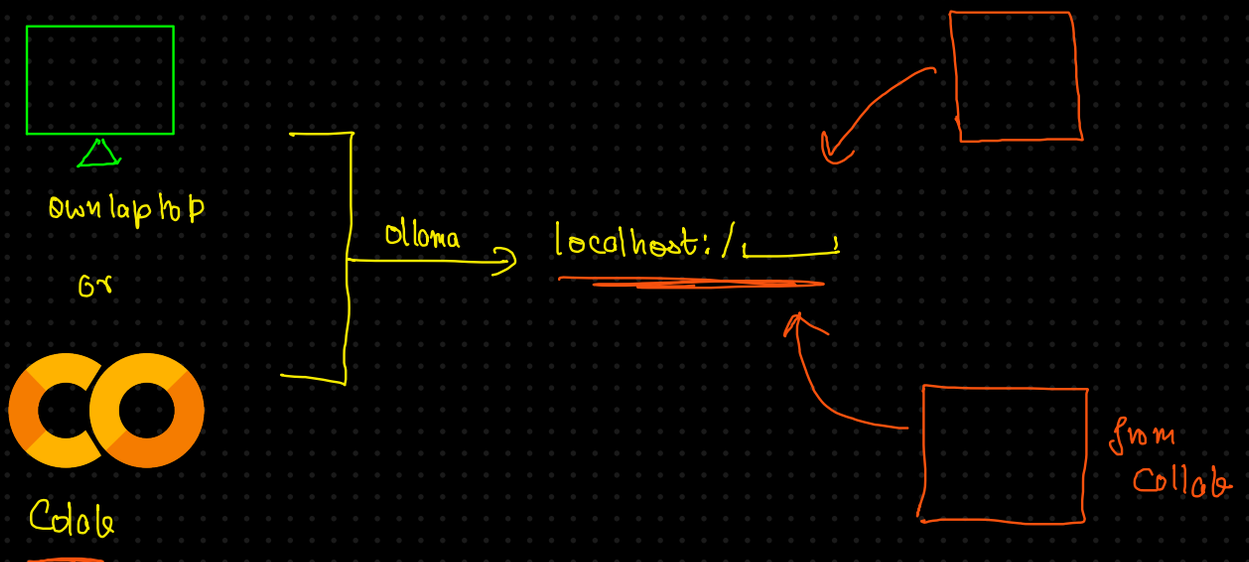
to authorize & use

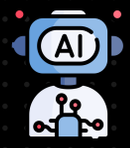
## Gemini

☐ Google cloud account ⇒ 3 months $300 credits

Google AI studio ⟶ Gemini Key

Ollama → local model




AI