Agents, Assemble!

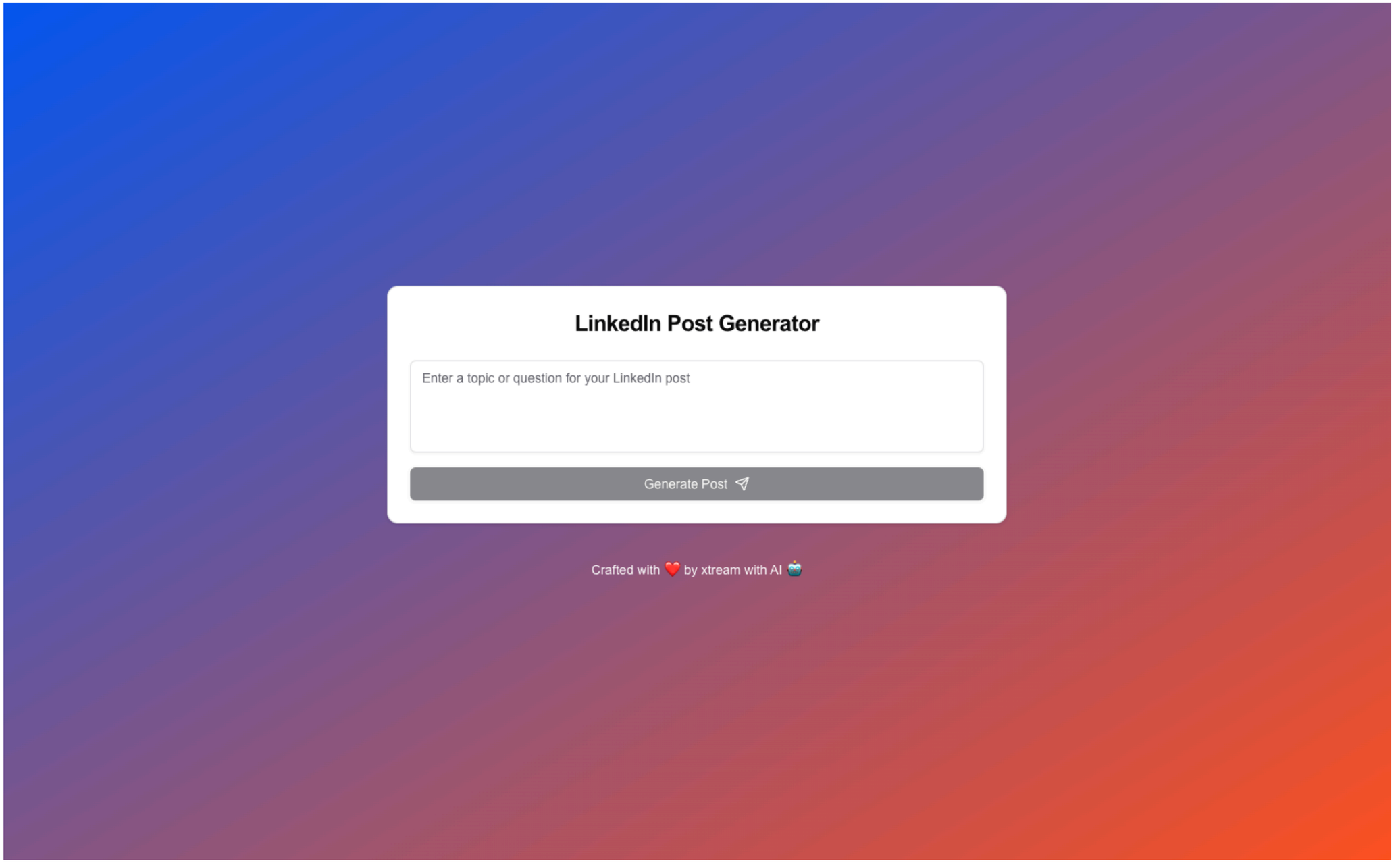
GenAI on steroids with Autogen and multi agents

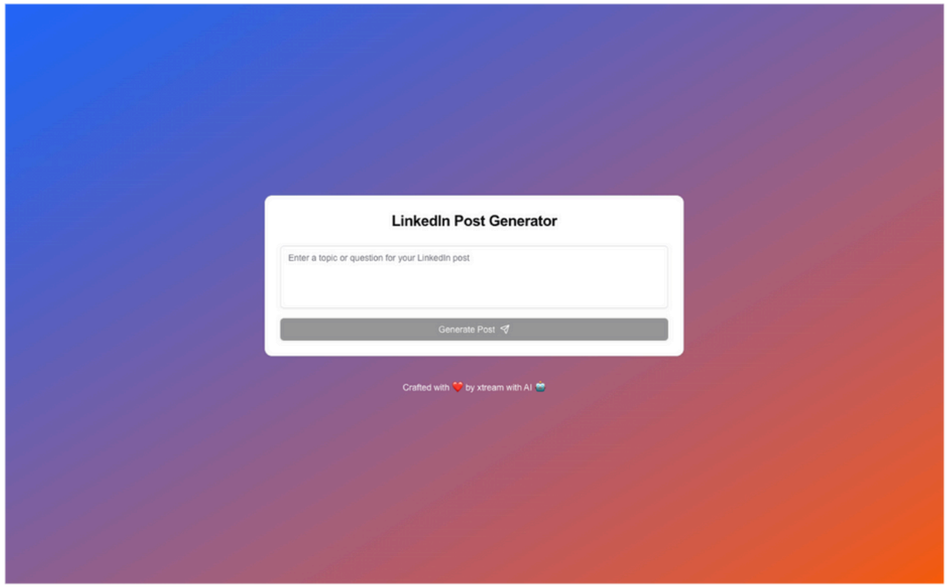
Simone Colucci
Riccardo Zoncada

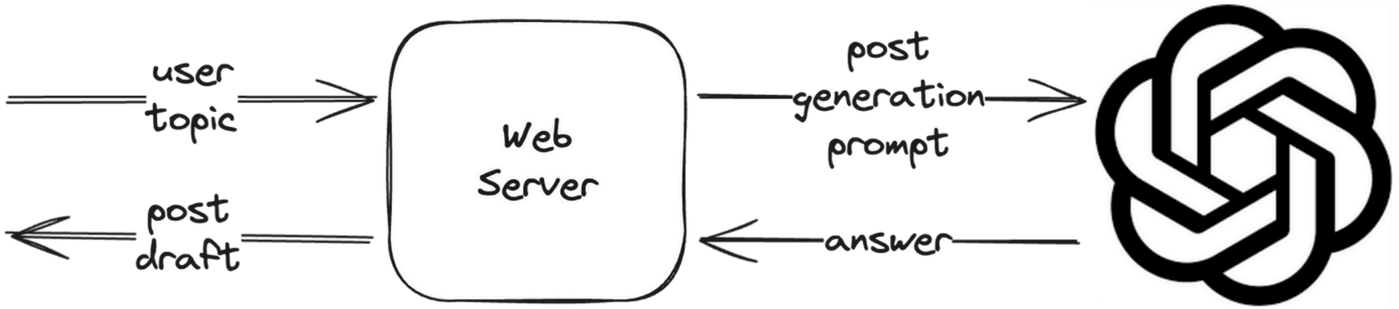
CONFERENCE
MILAN 2024

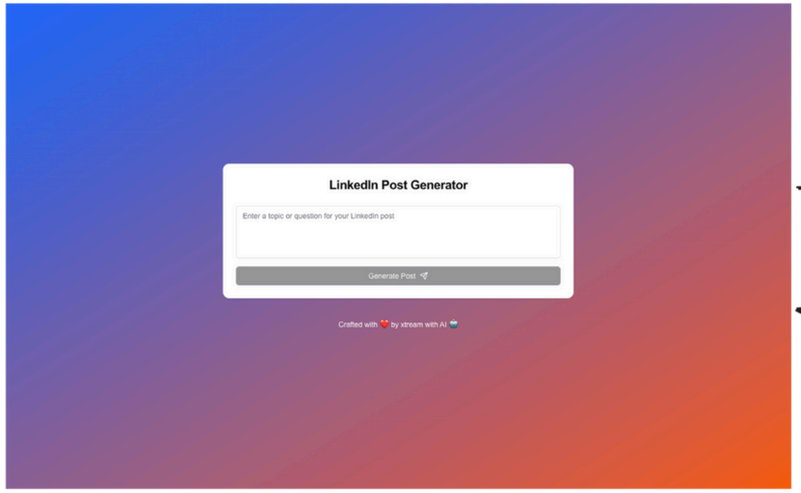
Agenda

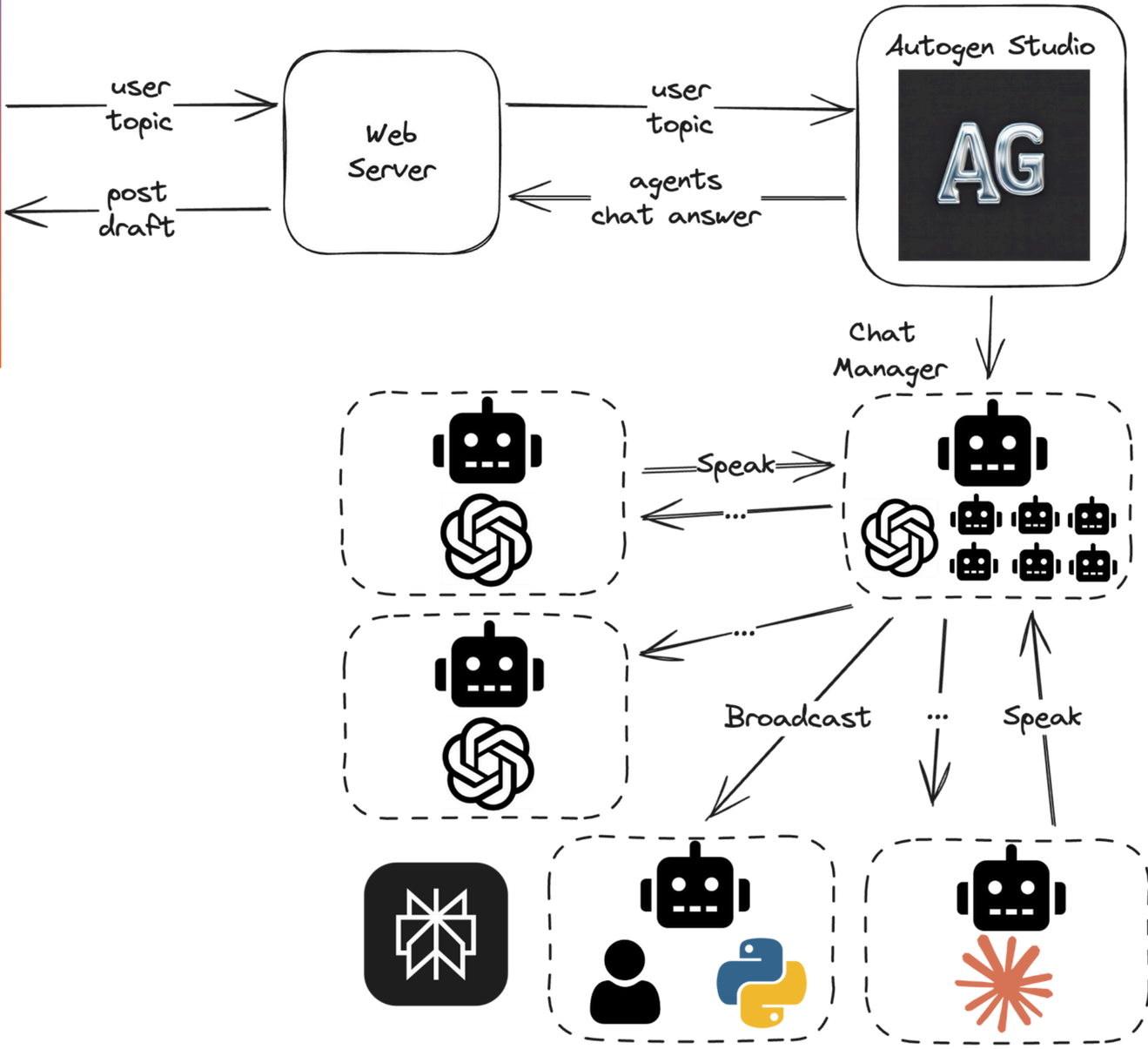
- Use case
- AI agents & multi-agent paradigm
- AutoGen & AutoGen Studio
- Live experimentations
- Wrap up


LinkedIn Post Generator
Enter a topic or question for your LinkedIn post
Generate Post
Crafted with ❤️ by xtream with AI 🤖

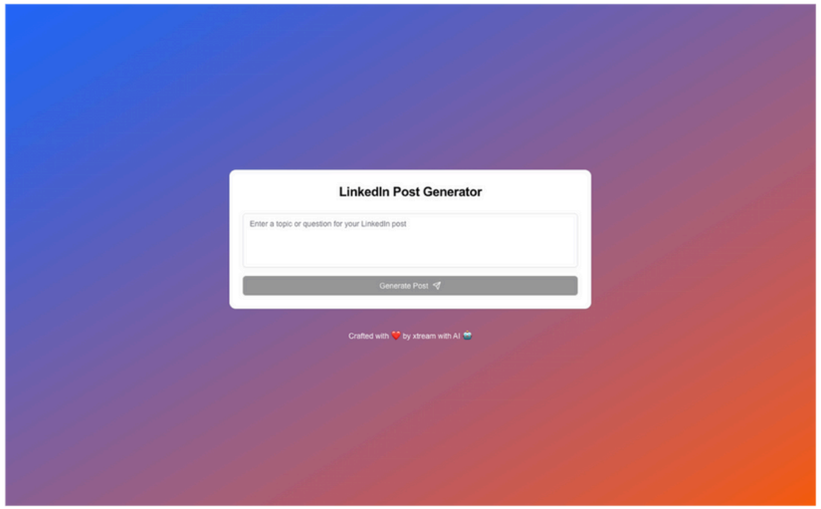

LinkedIn Post Generator
Enter a topic or question for your LinkedIn post
Generate Post
Crafted with by xtream with AI

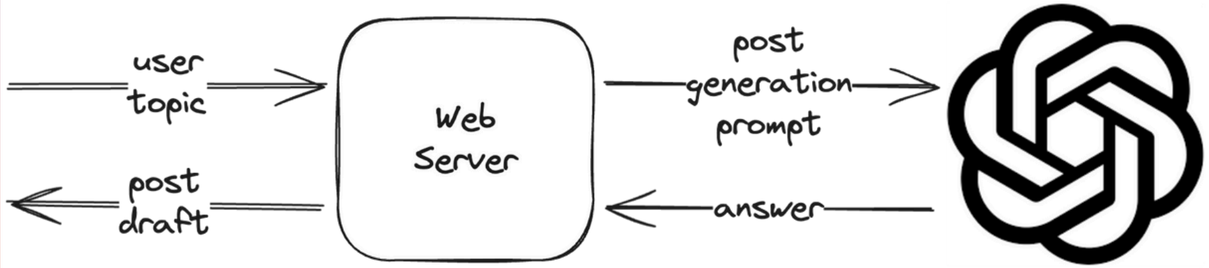

user
topic
post
draft
Web
Server
post
generation
prompt
answer


LinkedIn Post Generator
Enter a topic or question for your LinkedIn post
Generate Post
Crafted with ❤️ by xtream with AI


user
topic
post
draft
Web
Server
user
topic
agents
chat answer
Autogen Studio
AG
Chat
Manager
Speak
...
...
Broadcast
...
Speak

Single LLM — one-shot prompt



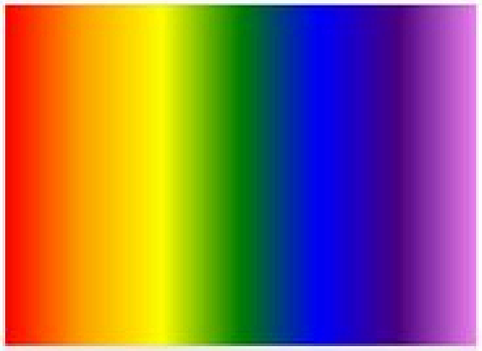


- Week guarantee on enforcing prompt guidelines
- Risk of hallucinations on unknown topics
- Lacks reasoning capabilities

Linear Gradients

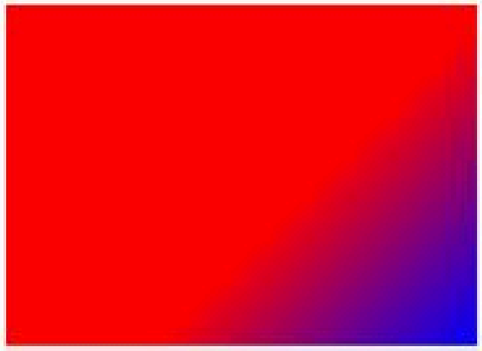
background: linear-gradient(to bottom right, red, rgba(255,0,0,0));

Linear with transparency



background: linear-gradient(to right, red, orange, yellow, green, blue, indigo, violet);

Gradient with multiple color stops



background: linear-gradient(135deg, red, red 60%, blue);

Gradient that starts at 60% of the gradient line

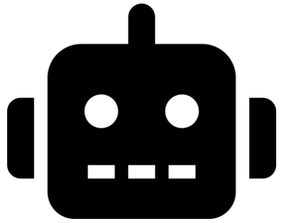
background: linear-gradient(135deg, red, blue);

Gradient at a 45 degree angle

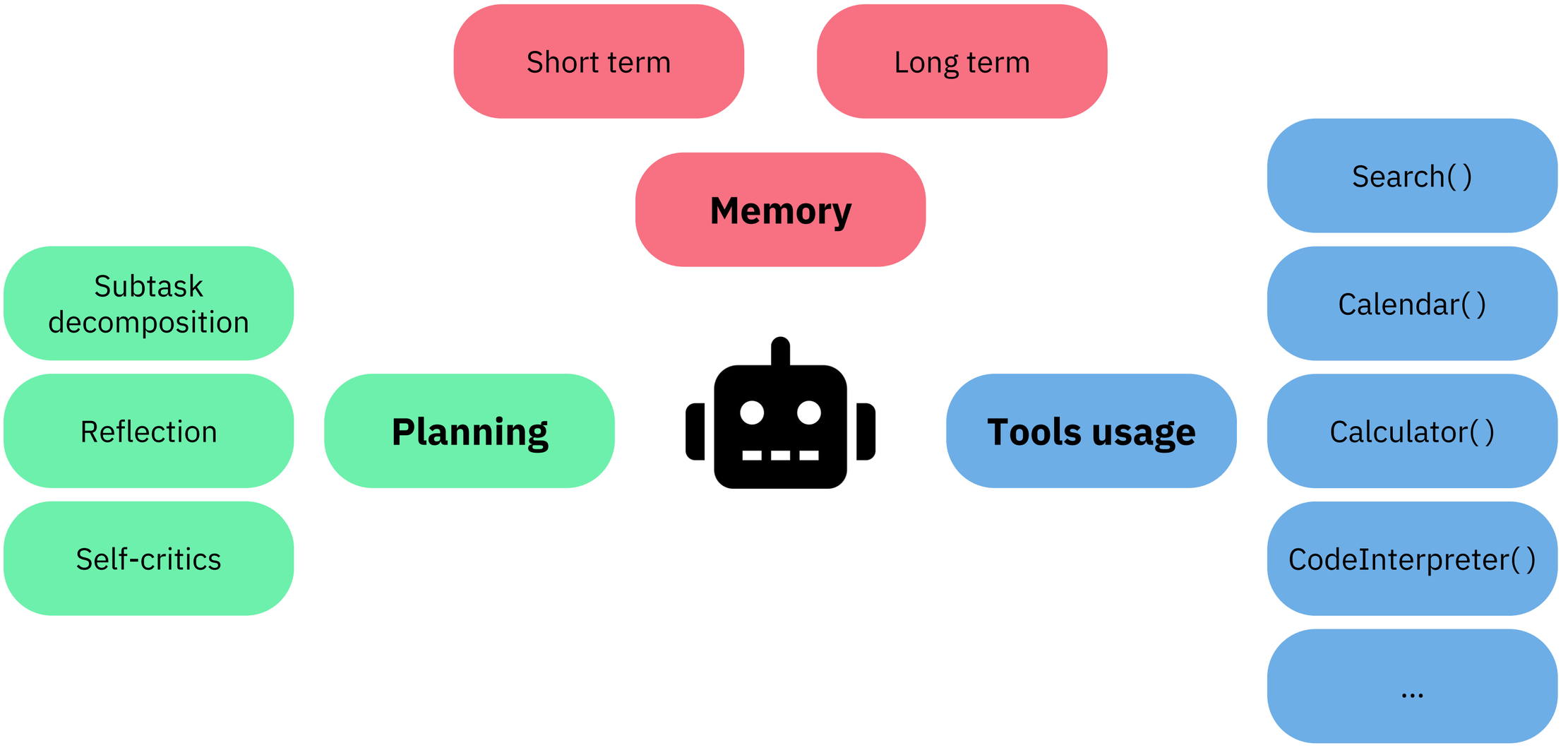
What is an AI agent?

Memory

Planning



Tools usage


Short term
Long term
Memory
Subtask decomposition
Reflection
Self-critics
Planning
Tools usage
Search( )
Calendar( )
Calculator( )
CodeInterpreter( )
...

Why multi-agent?

Multi-agent approach

- Multiple agents with their own strategies and behaviors.
- Every agents could use a specialised LLMs, tailored for a specific task.
- Communicate and collaborate to handle more dynamic and complex tasks in a collective decision-making processes.
- Empirical studies demonstrate the effectiveness of the framework in many example applications.

AgentCoder: Multi-Agent Code Generation with Effective Testing and Self-optimisation

Dong Huang
University of Hong Kong
dhuang@cs.hku.hk

Jie M.Zhang
jie.zhang@kcl.ac.uk
jie.zhang@kcl.ac.uk

Michael Luck
University of Sussex
Michael.Luck@sussex.ac.uk

Qingwen BU
Shanghai Jiao Tong University
qwbu01@sjtu.edu.cn

Yuhao Qing
University of Hong Kong
yhqing@cs.hku.hk

Heming Cui
University of Hong Kong
heming@cs.hku.hk

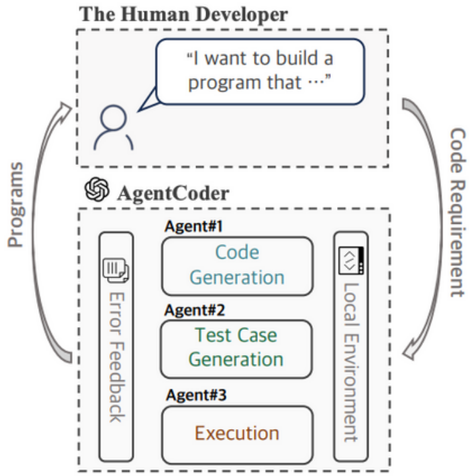
Abstract

Advances in natural language processing (NLP) have been significantly boosted by the development of transformer-based large language models (LLMs). These models have revolutionized NLP tasks, particularly in code generation, aiding developers in creating software with enhanced efficiency. Despite their advances, challenges remain in balancing code snippet generation with effective test cases. To address these issues, this paper introduces AgentCoder, a novel code generation solution comprising a multi-agent framework with a specialized test designer agents in addition to the programmer agent and the test executor agent. During the coding procedure, the test designer agent generates effective test cases for the generated code, and the test executor agent runs the code with the test cases and writes feedback to the programmer agent for it to refine the code. This collaborative system enhances code generation efficiency with less cost, outperforming both single-agent models and earlier multi-ag
our extensive experiments on 14 LLMs and 16 b
ple, AgentCoder (GPT-4) achieves 96.3% and 91.
MBPP datasets with an overall token overhead of
of-the-art obtains only 90.2% and 78.9% pass@1
of 138.2K and 206.5K.

1 Introduction

In recent years, natural language processing (NLP) h
transformer-based large language models (LLMs). These m
series [4, 29] developed by OpenAI, have consistently set
a wide array of standard NLP tasks. One of the most pivo
generation for downstream tasks, where they play a vital r
ware [13, 34, 36, 28, 27, 24]. Through extensive pretraini
such as publicly available data on GitHub, these code LI
standing that can be effectively applied to diverse code-rela

Numerous recent efforts have been made to improve the
in-context learning and its variations [11, 37, 20, 18, 42,
path is single-agent self-refinement within the same conve
proposed Self-Edit to enhance the performance of LLMs in
runs the generated code against test cases that are manuall
the LLMs to refine the code based on the error messages of

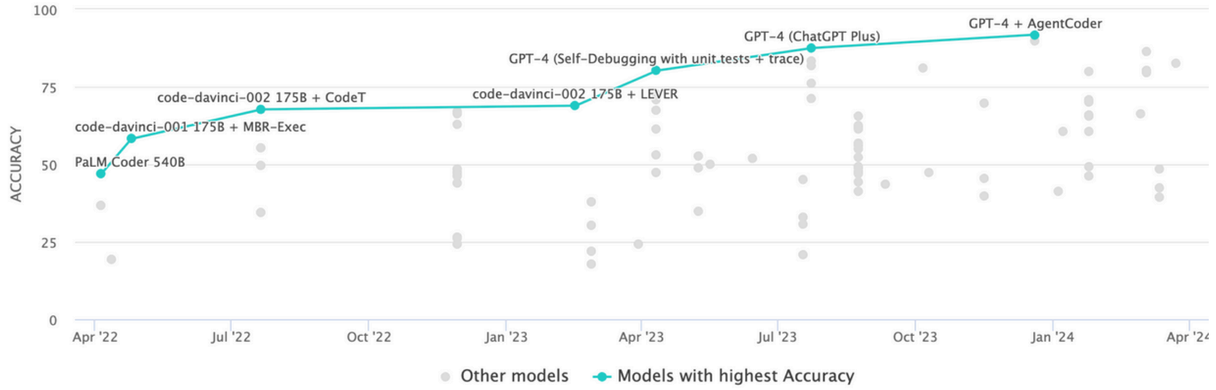


Preprint. Under review.

Code Generation on MBPP

Leaderboard Dataset

View Accuracy by



Filter: checked untagged

Edit L

Rank	Model	Accuracy	Paper	Code	Result	Year
1	**GPT-4 + AgentCoder**	91.8	AgentCoder: Multi-Agent-based Code Generation with Iterative Testing and Optimisation			2023
2	**GPT-3.5 Turbo** (ChatGPT)	89.9	AgentCoder: Multi-Agent-based Code Generation with Iterative Testing and Optimisation			2023
3	**GPT-4** (ChatGPT Plus)	87.5	How Does Naming Affect LLMs on Code Analysis Tasks?			2023
4	**Claude 3 Opus**	86.4	The Claude 3 Model Family: Opus, Sonnet, Haiku			2024

AgentCoder: Multi-Agent Code Generati
Effective Testing and Self-optimisati

Dong Huang
University of Hong Kong
dhuang@cs.hku.hk

Jie M.Zhang
jie.zhang@kcl.ac.uk
jie.zhang@kcl.ac.uk

Michael
University
Michael.Luck@s

Qingwen BU
Shanghai Jiao Tong University
qwbu01@sjtu.edu.cn

Yuhao Qing
University of Hong Kong
yhqing@cs.hku.hk

Hem
University
heming@

Abstract

Advances in natural language processing (NLP) have been significantly
by the development of transformer-based large language models (LLMs
models have revolutionized NLP tasks, particularly in code generation, a
velopers in creating software with enhanced efficiency. Despite their a
challenges remain in balancing code snippet generation with effective t
To address these issues, this paper introduces AgentCoder, a novel cod
tion solution comprising a multi-agent framework with a specialized test
agents in addition to the programmer agent and the test executor agent
the coding procedure, the test designer agent generates effective test cas
generated code, and the test executor agent runs the code with the test c
writes feedback to the programmer agent for it to refine the code. Thi
orative system enhances code generation efficiency with less cost, out
ing both single-agent models and earlier multi-ag
our extensive experiments on 14 LLMs and 16 b
ple, AgentCoder (GPT-4) achieves 96.3% and 91.
MBPP datasets with an overall token overhead of
of-the-art obtains only 90.2% and 78.9% pass@1
of 138.2K and 206.5K.

1 Introduction

In recent years, natural language processing (NLP) h
transformer-based large language models (LLMs). These m
series [4, 29] developed by OpenAI, have consistently set
a wide array of standard NLP tasks. One of the most pivo
generation for downstream tasks, where they play a vital r
ware [13, 34, 36, 28, 27, 24]. Through extensive pretraini
such as publicly available data on GitHub, these code LI
standing that can be effectively applied to diverse code-rela

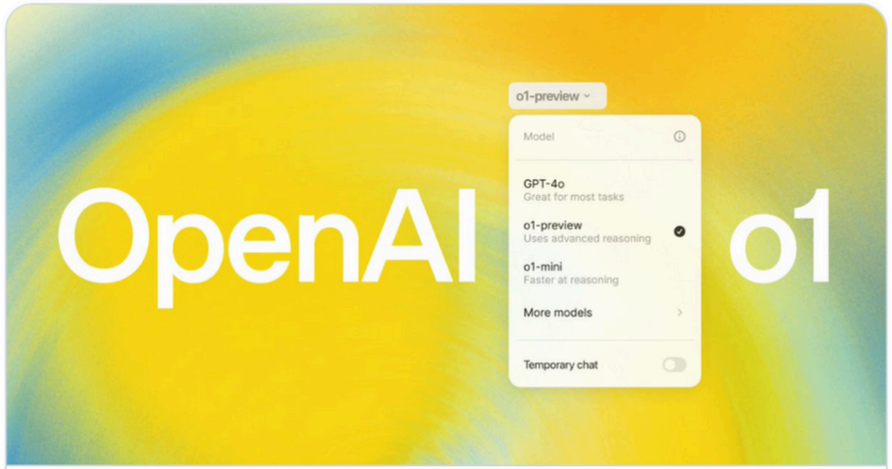
Numerous recent efforts have been made to improve the
in-context learning and its variations [11, 37, 20, 18, 42,
path is single-agent self-refinement within the same conve
proposed Self-Edit to enhance the performance of LLMs in
runs the generated code against test cases that are manually
the LLMs to refine the code based on the error messages of

Preprint. Under review.

OpenAI
@OpenAI

We're releasing a preview of OpenAI o1—a new series of AI models designed to spend more time thinking before they respond.

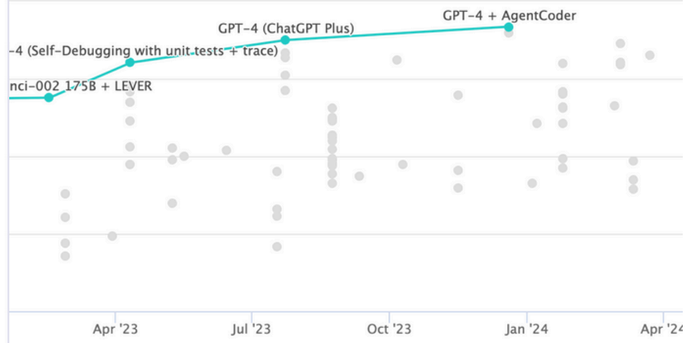
These models can reason through complex tasks and solve harder problems than previous models in science, coding, and math.



openai.com
Introducing OpenAI o1
Introducing OpenAI o1

7:09 PM · Sep 12, 2024

View Accuracy by



dels Models with highest Accuracy

Edit L

aper	Code	Result	Year
AgentCoder: Multi-Agent-based Code Generation with terative Testing and Optimisation			2023
AgentCoder: Multi-Agent-based Code Generation with terative Testing and Optimisation			2023
How Does Naming Affect LLMs on Code Analysis Tasks?			2023
The Claude 3 Model Family: Opus, Sonnet, Haiku			2024

How do we build multi-agent?

AutoGen

*AutoGen is an **open-source** framework that allows developers to build LLM applications via **multiple agents** that can **converse** with each other to accomplish tasks.*

Key features

- Conversation patterns for complex workflows
- Standard GenAI pipelines components (agents, models, tools)

AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation

Qingyun Wu[†], Gagan Bansal[*], Jieyu Zhang[±], Yiran Wu[†], Beibin Li[*]
Erkang Zhu[*], Li Jiang[*], Xiaoyun Zhang[*], Shaokun Zhang[†], Jiale Liu[∓]
Ahmed Awadallah[*], Ryen W. White[*], Doug Burger[*], Chi Wang[*1]
[*]Microsoft Research, [†]Pennsylvania State University
[±]University of Washington, [∓]Xidian University

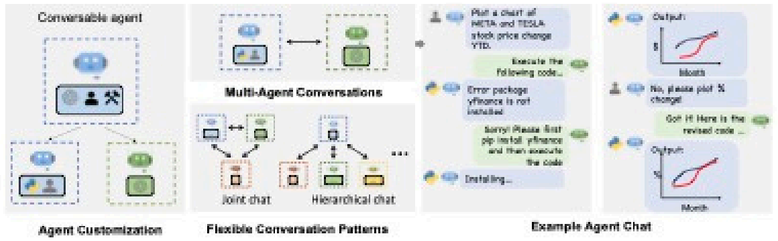


Figure 1: AutoGen enables diverse LLM-based applications using multi-agent conversations. (Left) AutoGen agents are conversable, customizable, and can be based on LLMs, tools, humans, or even a combination of them. (Top-middle) Agents can converse to solve tasks. (Right) They can form a chat, potentially with humans in the loop. (Bottom-middle) The framework supports flexible conversation patterns.

Abstract

AutoGen[2] is an open-source framework that allows developers to build LLM applications via multiple *agents* that can converse with each other to accomplish tasks. AutoGen agents are customizable, *conversable*, and can operate in various modes that employ combinations of LLMs, human inputs, and tools. Using AutoGen, developers can also flexibly define agent interaction behaviors. Both natural language and computer code can be used to program flexible conversation patterns for different applications. AutoGen serves as a generic framework for building diverse applications of various complexities and LLM capacities. Empirical studies demonstrate the effectiveness of the framework in many example applications, with domains ranging from mathematics, coding, question answering, operations research, online decision-making, entertainment, etc.

[1]Corresponding author. Email: auto-gen@outlook.com
[2]https://github.com/microsoft/autogen

AutoGen Studio
Build Multi-Agent Apps

Build | Playground | Gallery

Guest User
guestuser

CONFERENCE MILAN 2024

Workflow

Select or create an agent workflow.

General Agent Workflow

Create new workflows here

Sessions

Create a new session or select an existing session to view chat.

ada9e3bf-4079-4f66-8c8e-6a3 ...
just now

delete publish

45d4da7e-f9aa-4609-a408-984 ...
28 minutes ago

delete publish

680d8f0c-3e8f-4bb2-8c06-b54 ...
19 hours ago

delete publish

+ New

close sidebar

USER

create a 4 page pdf brochure on coffee from different parts of the world with some description of origins. E.g Ethiopian coffee may be in a glass on a table with a lush green forest in the background.

AGENT

The PDF brochure titled "Coffee_Brochure.pdf" has been successfully created. It includes images and descriptions of coffee from Ethiopia, Colombia, Brazil, and Vietnam, assembled into a 4-page document. Your brochure on coffee from different parts of the world is now ready. TERMINATE

Agent Messages (10 messages) | 2 mins 39 secs

Results (8 files)

Coffee_Brochure.pdf

77d9370c-1fe1-4658-a79a-ab7890c93a6b.png

f7a649b1-ce4a-4d5f-91a3-0085b2f574b6.png

a24eea2a-8d7c-4855-a1a8-953b08b0bf41.png

6d3319d6-8041-49e1-acb5-439f084440c5.png

generate_more_images.py

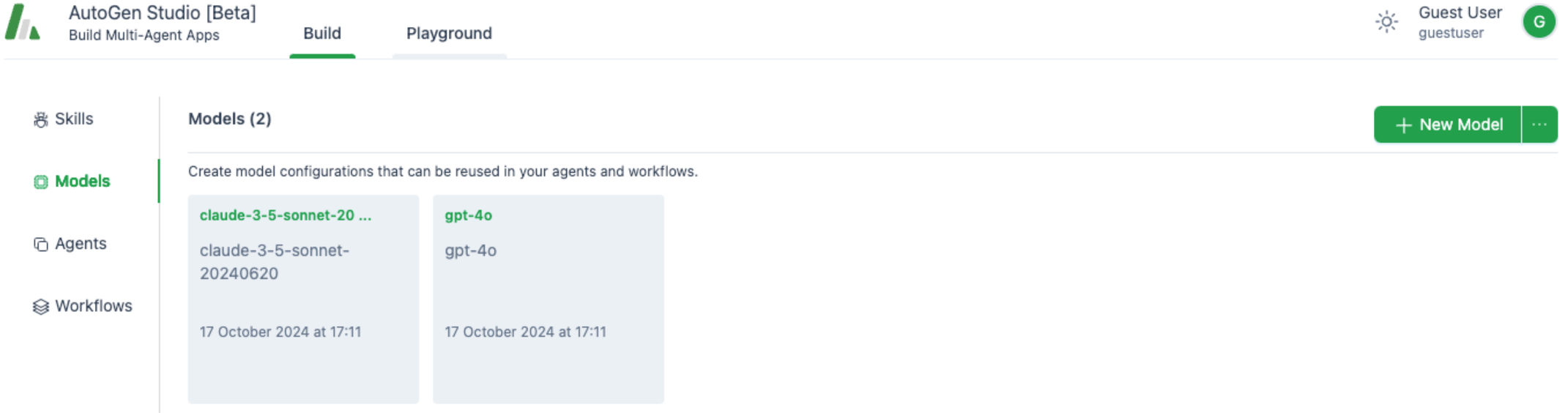
create_pdf_brochure.py

generate_images.py

Blank slate? Try one of the example prompts below

Stock Price | Sine Wave | Markdown | Paint

Maintained by the AutoGen Team.


AutoGen Studio [Beta]
Build Multi-Agent Apps
Build
Playground
Guest User
guestuser
G
Skills
Models
Agents
Workflows
Models (2)
New Model
Create model configurations that can be reused in your agents and workflows.
claude-3-5-sonnet-20 ...
claude-3-5-sonnet-20240620
17 October 2024 at 17:11
gpt-4o
gpt-4o
17 October 2024 at 17:11

Live time!

Wrap up

Advantages

- **Sophisticated content generation** — creates highly detailed and nuanced outputs.
- **Iterative reasoning** — capable of refining solutions through multiple iterations.
- **External tool integration** — seamlessly integrates with external tools to expand functionality.
- **Multi-model support** — Can utilize multiple models to tackle different tasks.

Challenges

- **Time and cost** — longer processing times and higher resource demands.
- **Instability** — possible inconsistent performance, loops or unpredictable results.

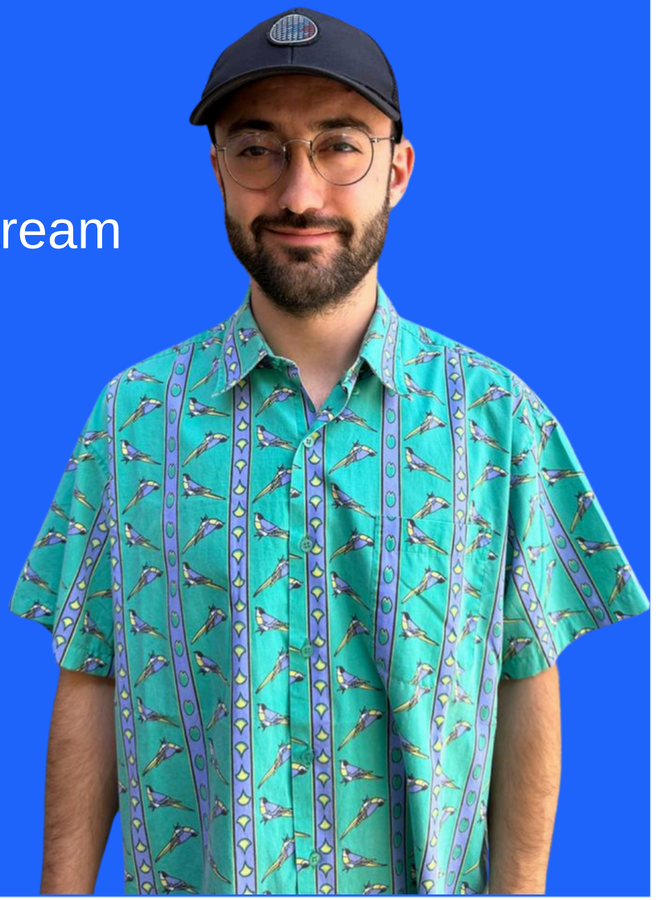
Resources

- AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation
- AutoGen Studio: A No-Code Developer Tool for Building and Debugging Multi-Agent Systems
- AgentCoder: Multi-Agent-based Code Generation with Iterative Testing and Optimisation
- AutoGen framework docs
- Code generation on MBPP benchmark
- AutoGen GitHub repo
- GenAI for Engineers Class by xtream
- Create your own v0 — Codemotion Workshop Fest 2024 by xtream
- Create your own v0 — repository

Let’s get in touch!

Simone Colucci
Founder, Head of Product @ xtream

Riccardo Zoncada
Software Engineer @ xtream



{codemotion} CONFERENCE MILAN 2024

Don't forget to rate the talk!

{codemotion} CONFERENCE MILAN 2024

Thanks!