

joaomdmoura / crewAI

Search

Notifications

<> Code

Issues 86

Pull requests 20

Actions

Projects

Security

Insights



Framework for orchestrating role-playing, autonomous AI agents. By fostering collaborative intelligence, CrewAI empowers agents to work together seamlessly, tackling complex tasks.

[crewai.io](#)

MIT license

7.8k stars

846 forks

99 watching

4 Branches

14 Tags

Activity

Public repository

main

4 Branches

14 Tags

🔍 Go to file

Go to file

+

Admin

joaomdmoura 4 days ago

⋮

🕒

📁 .cache/plugin/social	adding manager_llm	2 weeks ago
📁 .github/workflows	preparing new version 0.5.0	2 weeks ago
📁 docs	adding function calling llm s...	last week
📁 src/crewai	adding more error logging a...	4 days ago
📁 tests	Cutting new version with to...	4 days ago
📄 .gitignore	updating codeignore	last week
📄 .pre-commit-config.y...	Update autoflake args	2 months ago
📄 LICENSE	Updating Readme	3 months ago
📄 README.md	updating readme	last week
📄 crewAI.excalidraw	Adding Excalidraw diagram	3 months ago
📄 mkdocs.yml	Adding new crew specific do...	2 weeks ago
📄 poetry.lock	Cutting new version with to...	4 days ago

pyproject.toml

adding more error logging a...

4 days ago



crewAI

crewAI: Cutting-edge framework for orchestrating role-playing, autonomous AI agents. By fo
empowers agents to work together seamlessly, tackling compl

[Homepage](#) | [Documentation](#) | [Chat with Docs](#) | [Examp](#)

 Stars < 7.8k License MIT

Table of contents

- [Why CrewAI?](#)
- [Getting Started](#)
- [Key Features](#)
- [Examples](#)
 - [Quick Tutorial](#)
 - [Trip Planner](#)
 - [Stock Analysis](#)
- [Connecting Your Crew to a Model](#)
- [How CrewAI Compares](#)
- [Contribution](#)

- [Hire CrewAI](#)
- [Telemetry](#)
- [License](#)

Why CrewAI?

The power of AI collaboration has too much to offer. CrewAI is designed to enable AI agents to ass cohesive unit - much like a well-oiled crew. Whether you're building a smart assistant platform, an multi-agent research team, CrewAI provides the backbone for sophisticated multi-agent interaction

Getting Started

To get started with CrewAI, follow these simple steps:

1. Installation

```
pip install crewai
```

The example below also uses DuckDuckGo's Search. You can install it with `pip` too:

```
pip install duckduckgo-search
```

2. Setting Up Your Crew

```
import os
from crewai import Agent, Task, Crew, Process

os.environ["OPENAI_API_KEY"] = "YOUR_API_KEY"

# You can choose to use a local model through Ollama for example. See ./docs/how-to/llm-co
# from langchain_community.llms import Ollama
# ollama_llm = Ollama(model="openhermes")

# Install duckduckgo-search for this example:
# !pip install -U duckduckgo-search

from langchain_community.tools import DuckDuckGoSearchRun
search_tool = DuckDuckGoSearchRun()

# Define your agents with roles and goals
researcher = Agent(
```

```

role='Senior Research Analyst',
goal='Uncover cutting-edge developments in AI and data science',
backstory="""You work at a leading tech think tank.
Your expertise lies in identifying emerging trends.
You have a knack for dissecting complex data and presenting actionable insights."""
verbose=True,
allow_delegation=False,
tools=[search_tool]
# You can pass an optional llm attribute specifying what mode you wanna use.
# It can be a local model through Ollama / LM Studio or a remote
# model like OpenAI, Mistral, Anthropic or others (https://python.langchain.com/docs/integrations/llms/ollama)
#
# Examples:
#
# from langchain_community.llms import Ollama
# llm=ollama_llm # was defined above in the file
#
# from langchain_openai import ChatOpenAI
# llm=ChatOpenAI(model_name="gpt-3.5", temperature=0.7)
)
writer = Agent(
    role='Tech Content Strategist',
    goal='Craft compelling content on tech advancements',
    backstory="""You are a renowned Content Strategist, known for your insightful and engaging
    You transform complex concepts into compelling narratives."""
    verbose=True,
    allow_delegation=True,
    # (optional) llm=ollama_llm
)

# Create tasks for your agents
task1 = Task(
    description="""Conduct a comprehensive analysis of the latest advancements in AI in 2024.
    Identify key trends, breakthrough technologies, and potential industry impacts.
    Your final answer MUST be a full analysis report""",
    agent=researcher
)

task2 = Task(
    description="""Using the insights provided, develop an engaging blog
    post that highlights the most significant AI advancements.
    Your post should be informative yet accessible, catering to a tech-savvy audience.
    Make it sound cool, avoid complex words so it doesn't sound like AI.
    Your final answer MUST be the full blog post of at least 4 paragraphs.""",
    agent=writer
)

# Instantiate your crew with a sequential process
crew = Crew(

```

```
agents=[researcher, writer],
tasks=[task1, task2],
verbose=2, # You can set it to 1 or 2 to different logging levels
)

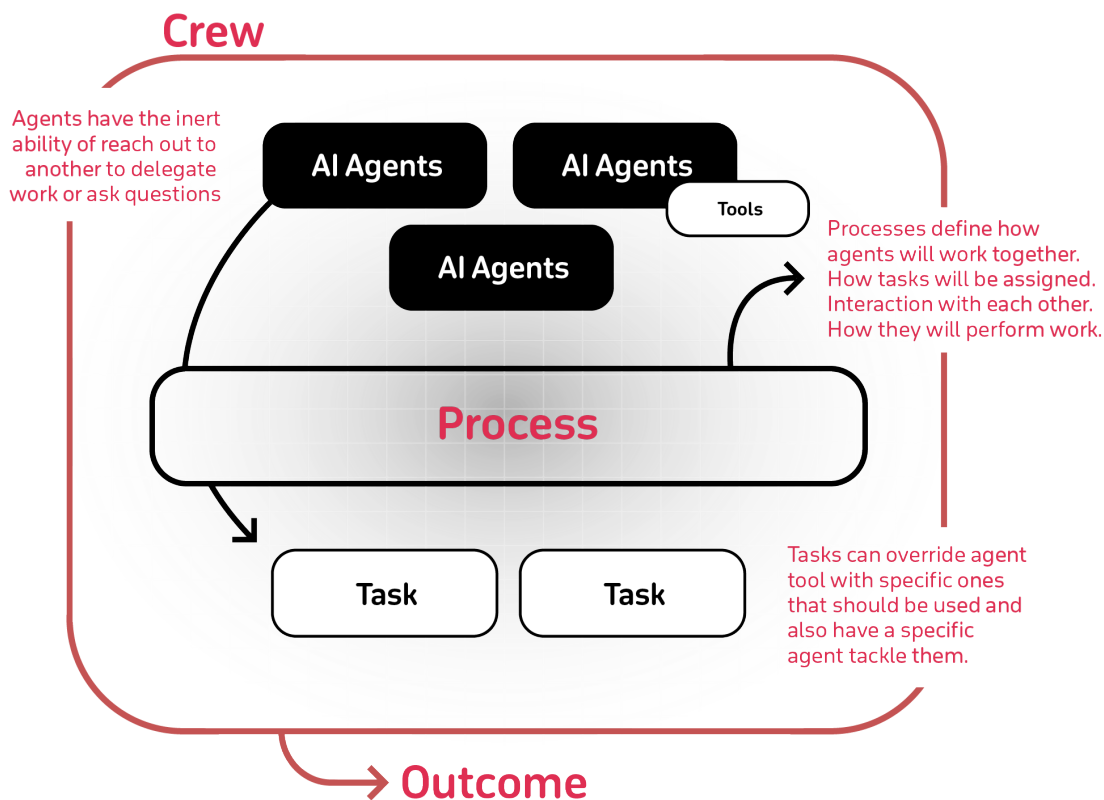
# Get your crew to work!
result = crew.kickoff()

print("#####")
print(result)
```

In addition to the sequential process, you can use the hierarchical process, which automatically ass properly coordinate the planning and execution of tasks through delegation and validation of resu

Key Features

- **Role-Based Agent Design:** Customize agents with specific roles, goals, and tools.
- **Autonomous Inter-Agent Delegation:** Agents can autonomously delegate tasks and inquire a solving efficiency.
- **Flexible Task Management:** Define tasks with customizable tools and assign them to agents d
- **Processes Driven:** Currently only supports `sequential` task execution and `hierarchical` proc consensual and autonomous are being worked on.
- **Works with Open Source Models:** Run your crew using Open AI or open source models refer details on configuring you agents' connections to models, even ones running locally!



Examples

You can test different real life examples of AI crews in the [crewAI-examples repo](#):

- [Landing Page Generator](#)
- [Having Human input on the execution](#)
- [Trip Planner](#)
- [Stock Analysis](#)

Quick Tutorial



Trip Planner

[Check out code for this example](#) or watch a video below:



Stock Analysis

[Check out code for this example](#) or watch a video below:



Stock Analysis with AI Agents

Connecting Your Crew to a Model

crewAI supports using various LLMs through a variety of connection options. By default your agent connects to the model. However, there are several other ways to allow your agents to connect to models. For example, you can use a local model via the Ollama tool.

Please refer to the [Connect crewAI to LLMs](#) page for details on configuring your agents' connection options.

How CrewAI Compares

- **Autogen:** While Autogen excels in creating conversational agents capable of working together, orchestrating agents' interactions requires additional programming, which can become cumbersome as the number of tasks grows.
- **ChatDev:** ChatDev introduced the idea of processes into the realm of AI agents, but its implementation is limited and not geared towards production environments, which can hinder scalability.

CrewAI's Advantage: CrewAI is built with production in mind. It offers the flexibility of Autogen's conversational approach of ChatDev, but without the rigidity. CrewAI's processes are designed to be dynamic, adapting to both development and production workflows.

Contribution

CrewAI is open-source and we welcome contributions. If you're looking to contribute, please:

- Fork the repository.
- Create a new branch for your feature.
- Add your feature or improvement.
- Send a pull request.
- We appreciate your input!

Installing Dependencies

 **README**  MIT license

```
poetry install
```

Virtual Env

```
poetry shell
```

Pre-commit hooks

```
pre-commit install
```

Running Tests

```
poetry run pytest
```

Running static type checks

```
poetry run pyright
```

Packaging

```
poetry build
```

Installing Locally

```
pip install dist/*.tar.gz
```

Hire CrewAI

We're a company developing crewAI and crewAI Enterprise, we for a limited time are offer consultancy early access to our enterprise solution If you are interested on having access to it and hiring weekly at joao@crewai.com.

Telemetry

CrewAI uses anonymous telemetry to collect usage data with the main purpose of helping us improve the most used features, integrations and tools.

There is NO data being collected on the prompts, tasks descriptions agents backstories or goals nor any data that is being processed by the agents, nor any secrets and env vars.

Data collected includes:


- Version of crewAI
 - So we can understand how many users are using the latest version
- Version of Python
 - So we can decide on what versions to better support
- General OS (e.g. number of CPUs, macOS/Windows/Linux)
 - So we know what OS we should focus on and if we could build specific OS related features
- Number of agents and tasks in a crew
 - So we make sure we are testing internally with similar use cases and educate people on the
- Crew Process being used
 - Understand where we should focus our efforts
- If Agents are using memory or allowing delegation
 - Understand if we improved the features or maybe even drop them
- If Tasks are being executed in parallel or sequentially
 - Understand if we should focus more on parallel execution
- Language model being used
 - Improved support on most used languages
- Roles of agents in a crew
 - Understand high level use cases so we can build better tools, integrations and examples
- Tools names available
 - Understand out of the publically available tools, which ones are being used the most so we

Users can opt-in sharing the complete telemetry data by setting the `share_crew` attribute to `True`

License

CrewAI is released under the MIT License.

Releases 14

 **v0.11.1** Latest


4 days ago

+ 13 releases

Packages

No packages published

Used by 162



+ 154

Contributors 22



+ 8 contributors

Deployments 61

 **github-pages** 4 days ago

+ 60 deployments

Languages

- Python 100.0%