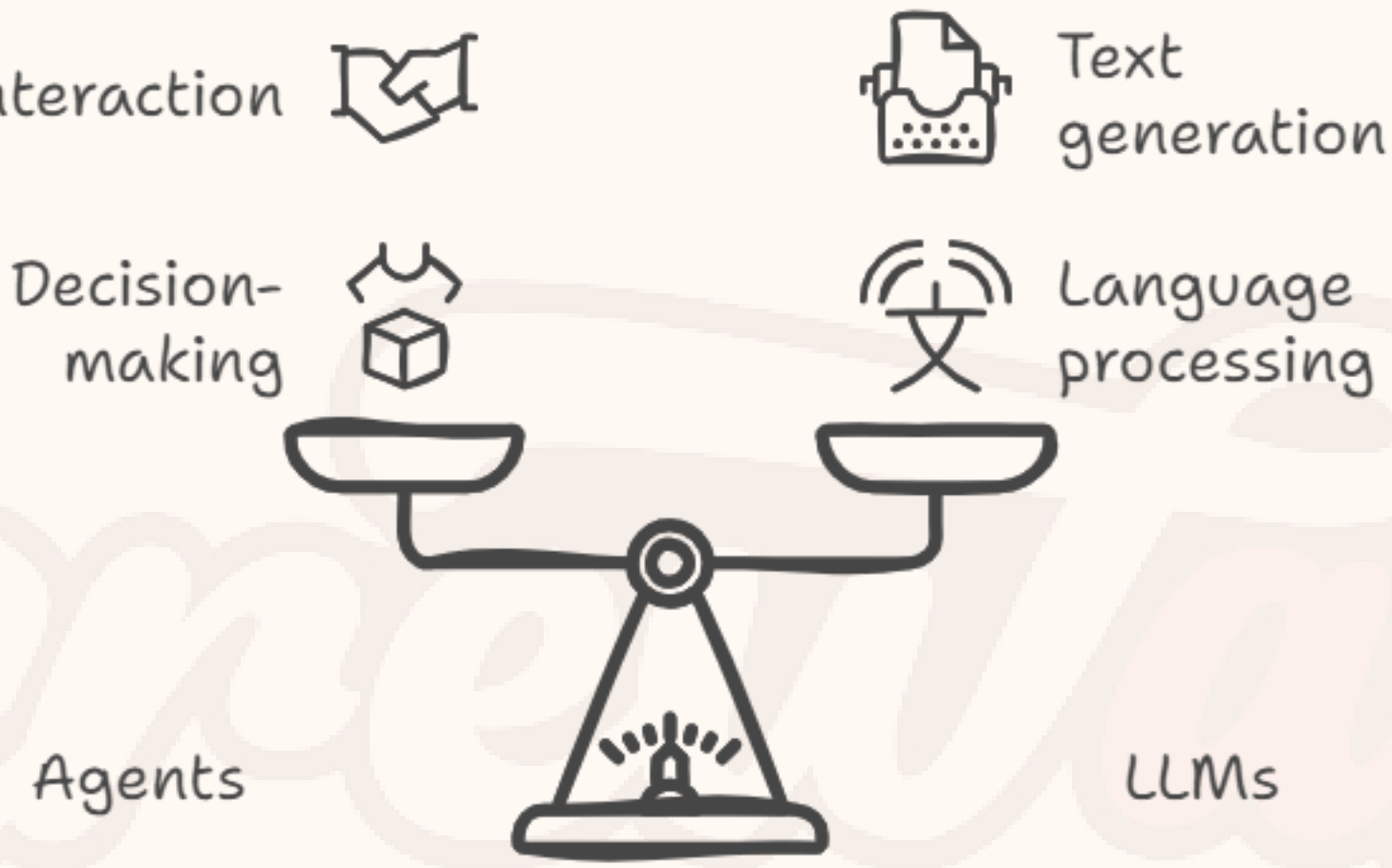
Building Multi Agent Systems with CrewAI

Bhavishya Pandit

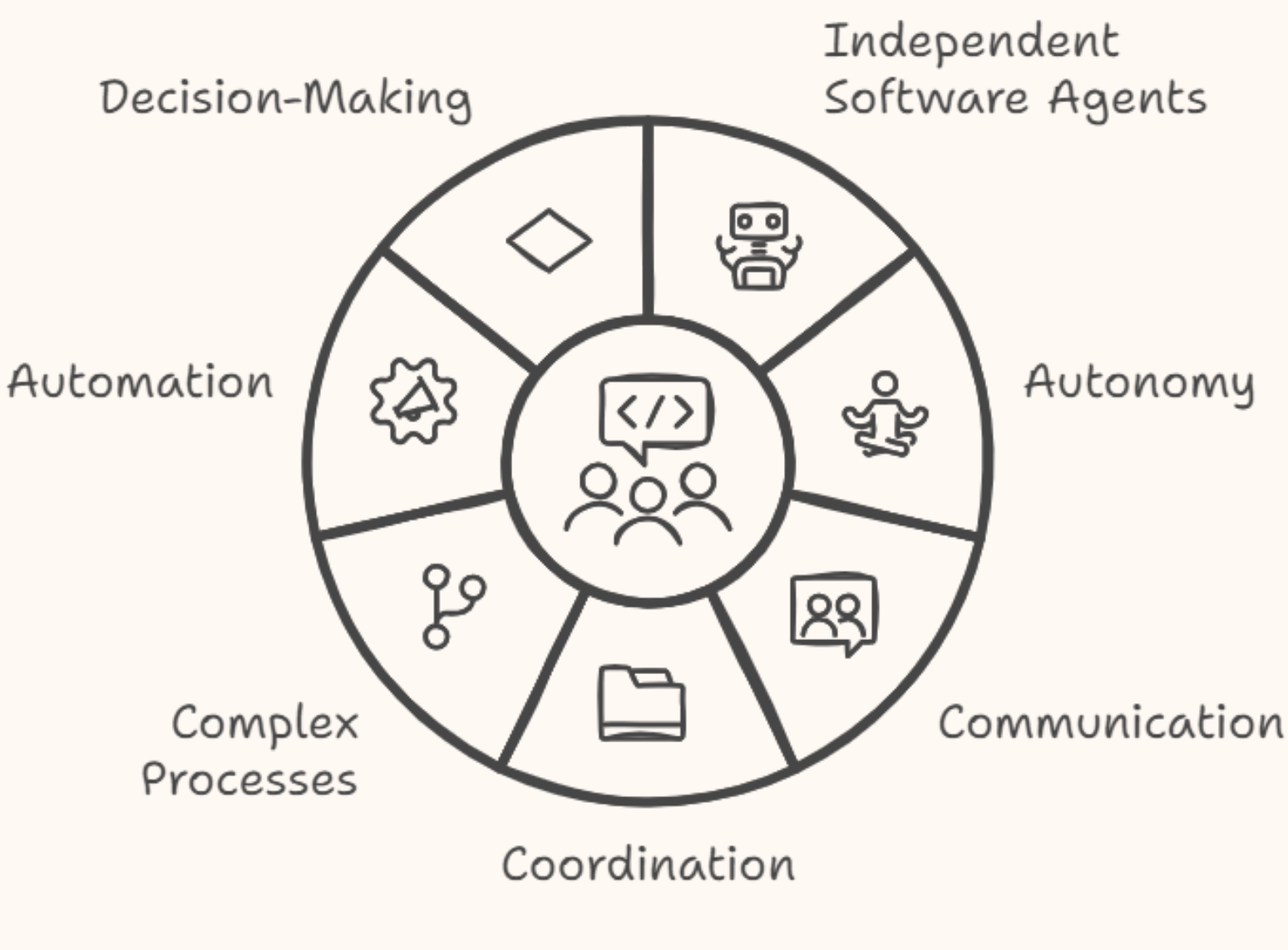
Welcome to Crew AI



CrewAI is an open-source Python framework designed to support developing and managing multi-agent AI systems.

CrewAI improves these AI systems by assigning specific roles, enabling autonomous decision-making, and facilitating communication between agents. This approach allows them to tackle complex problems more effectively than individual agents working alone.

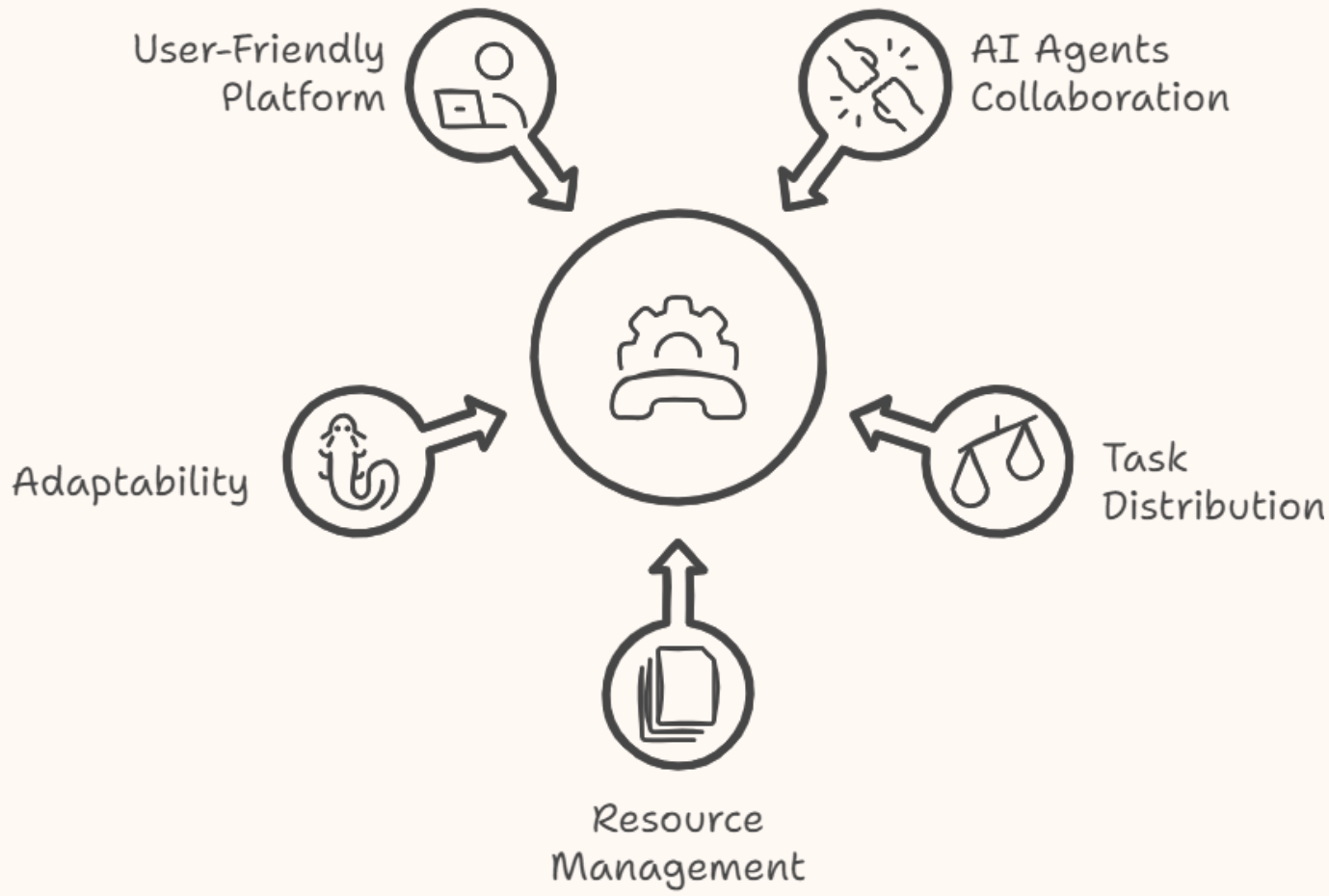
What are Multi-Agent Systems



Multi-agent systems involve multiple independent software agents that work together to solve tasks or problems.

Each agent acts autonomously but can communicate with others to share information and coordinate actions, making them useful for handling complex processes.

Benefits of Using CrewAI



- **Enhanced Creativity**: Crewai AI can generate innovative ideas and concepts, helping teams brainstorm more effectively.
- **Time Efficiency**: It automates routine tasks, allowing users to focus on more strategic and creative work.
- **Collaborative Features**: Supports team collaboration by enabling multiple users to work together seamlessly.
- **Scalability**: Easily adapts to different team sizes and project scopes, making it suitable for both small startups and large enterprises.

A Quick Glimpse at CrewAI Code

Let's roll up our sleeves and build a workflow using CrewAI tools to scrape the content from the website and then perform RAG on it.

Step 1: Install the crewai-tools and crewai packages

```
pip install crewai-tools crewai
```

Step 2: Scraping a website

```
from crewai_tools import ScrapeWebsiteTool, FileWriterTool,
TXTSearchTool
import requests

# Initialize the tool, potentially passing the session
tool =
ScrapeWebsiteTool(website_url='https://en.wikipedia.org/wiki/Artificial_
intelligence')

# Extract the text
text = tool.run()
print(text)
```

Bhavishya Pandit

Step3: Write the extracted text to a file

```
# Initialize the tool
file_writer_tool = FileWriterTool()

# Write content to a file in a specified directory
result = file_writer_tool._run(filename='ai.txt', content = text,
directory = '', overwrite=True)
print(result)
```

Step 4: Set up the text search tool

```
import os
from crewai_tools import TXTSearchTool

os.environ['OPENAI_API_KEY'] = 'API-KEY'

# Initialize the tool with a specific text file, so the agent can
search within the given text file's content
tool = TXTSearchTool(txt='ai.txt')
```

Step 5: Create an agent for the task and execute it

```
from crewai import Agent, Task, Crew

context = tool.run('What is natural language processing?')

data_analyst = Agent(
    role='Educator',
    goal=f'Based on the context provided, answer the question - What is Natural Language Processing? Context - {context}',
    backstory='You are a data expert',
    verbose=True,
    allow_delegation=False,
    tools=[tool]
)

test_task = Task(
    description="Understand the topic and give the correct response",
    tools=[tool],
    agent=data_analyst,
    expected_output='Give a correct response'
)

crew = Crew(
    agents=[data_analyst],
    tasks=[test_task]
)

output = crew.kickoff()
```

Output

```
output.raw
```

```
'Natural language processing (NLP) allows programs to read, write and communicate in human languages such as English. Specific problems include speech recognition, speech synthesis, machine translation, information extraction, information retrieval and question answering.\n\nEarly work, based on Noam Chomsky\'s generative grammar and semantic networks, had difficulty with word-sense disambiguation unless restricted to small domains called "micro-worlds" (due to the common sense knowledge problem). Margaret Masterman believed that it was meaning and not grammar that was the key to understanding languages, and that thesauri and not dictionaries should be the basis of computational language structure.\n\nModern deep learning techniques for NLP include word embedding (representing words, typically as vectors encoding their meaning), transformers (a deep learning architecture using an attention mechanism), and others. In 2019, generative pre-trained transformer (or "GPT") language models began to generate coherent text, and by 2023, these models were able to get human-level scores on the bar exam, SAT test, GRE test, and many other real-world applications.<'
```

How many different CrewAI tools have you used?

Comment below

Bhavishya Pandit

Follow for more AI/ML posts

SAVE

LIKE

SHARE

Bhavishya Pandit