

Beginner's Guide to Multi-Agents

AI by Hand 

Prof. Tom Yeh

Hosted by:



University of Colorado
Boulder



Real World Example

Orion by Gravity Foundation

Beginner's Guide to Multi-Agents - AI by Hand



University of Colorado
Boulder



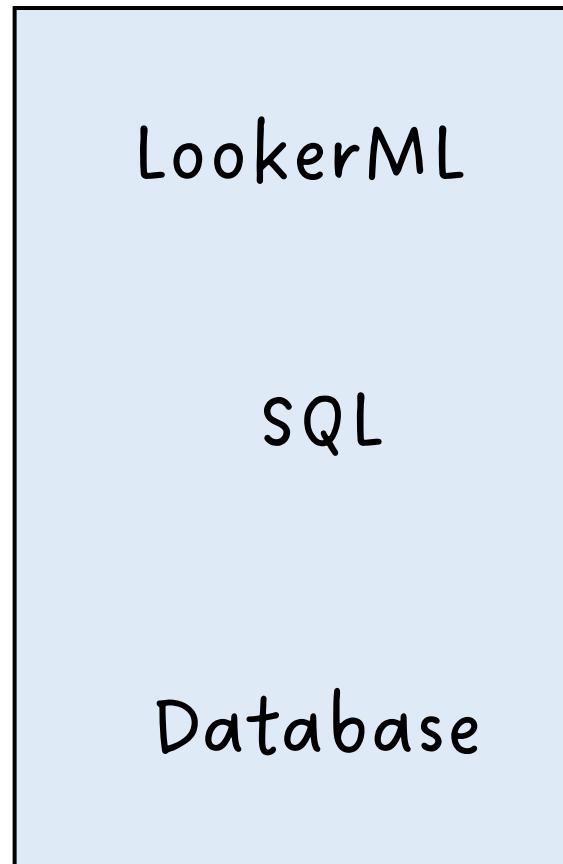
A team of AI-Powered agents. For your team of data heroes.

Orion is built using an agentic system, where multiple specialized AI agents collaborate to continuously monitor your data, investigate trends, and generate actionable insights.

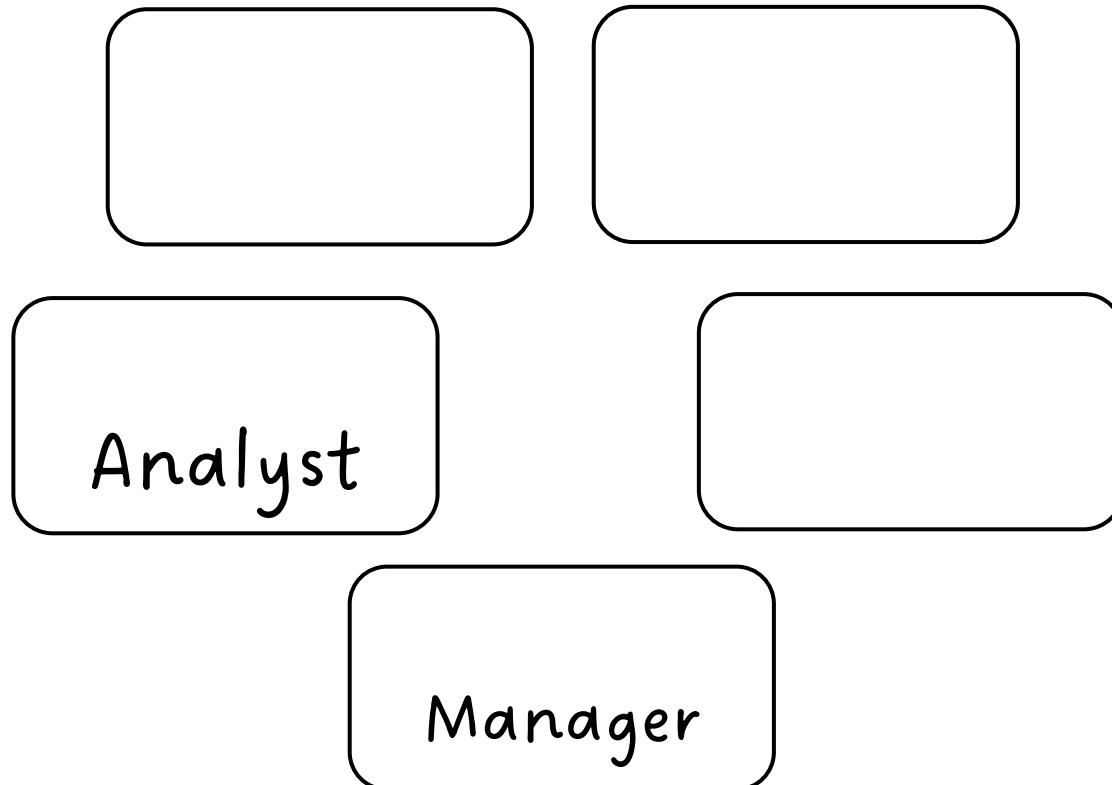
The image displays a collage of screenshots from the Orion platform, illustrating its various features and capabilities:

- Orion Dashboard:** Shows a navigation bar with "Orion", "Dashboard", "Insights", "Analysis Planner", and "Configuration". Below is a "Lockers" section with "Buttons", "Schemas", and "Admin".
- Insights View:** A prominent window titled "Jeans Sales Surge 29%: Order 6,000 Units to Meet Holiday Demand". It includes a timestamp "August 27, 2024" and a green "Approved" button. The summary states: "Due to high customer interest, jeans sales have increased by 29% compared to last year. However, our inventory is depleting faster than expected. Anticipate continuous high demand for Labor Day Sales and the approaching holiday season." Recommended actions include "Pause the current sale (30% off) and sell the remaining jeans at the original price." and "Order additional jeans".
- Analysis View:** Shows "Current Inventory Status" with a pie chart and a table of data for countries like United States, Canada, Mexico, Australia, and United Kingdom.
- Configuration View:** Shows a "Business" section with "State" and "Country" dropdowns.
- User Onboarding:** A "User Onboarding" section with a progress bar and a "Get Started" button.

Business
Intelligence
Looker



Multi-Agents



Business
Operations



AI by Hand A simple yellow line drawing of a hand holding a pen or pencil.

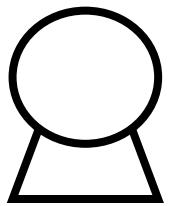
LLM -> RAG

Beginner's Guide to Multi-Agents - AI by Hand

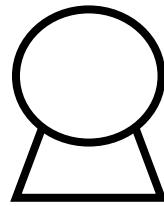


University of Colorado
Boulder

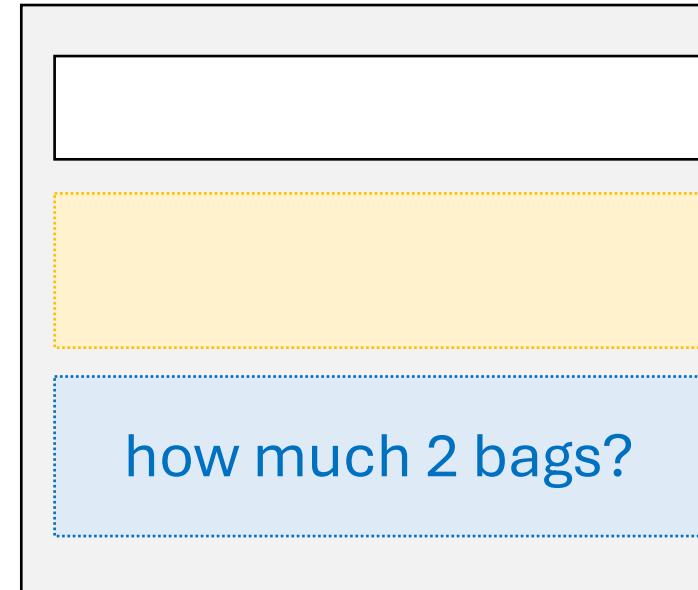
LLM Only



RAG



RAG



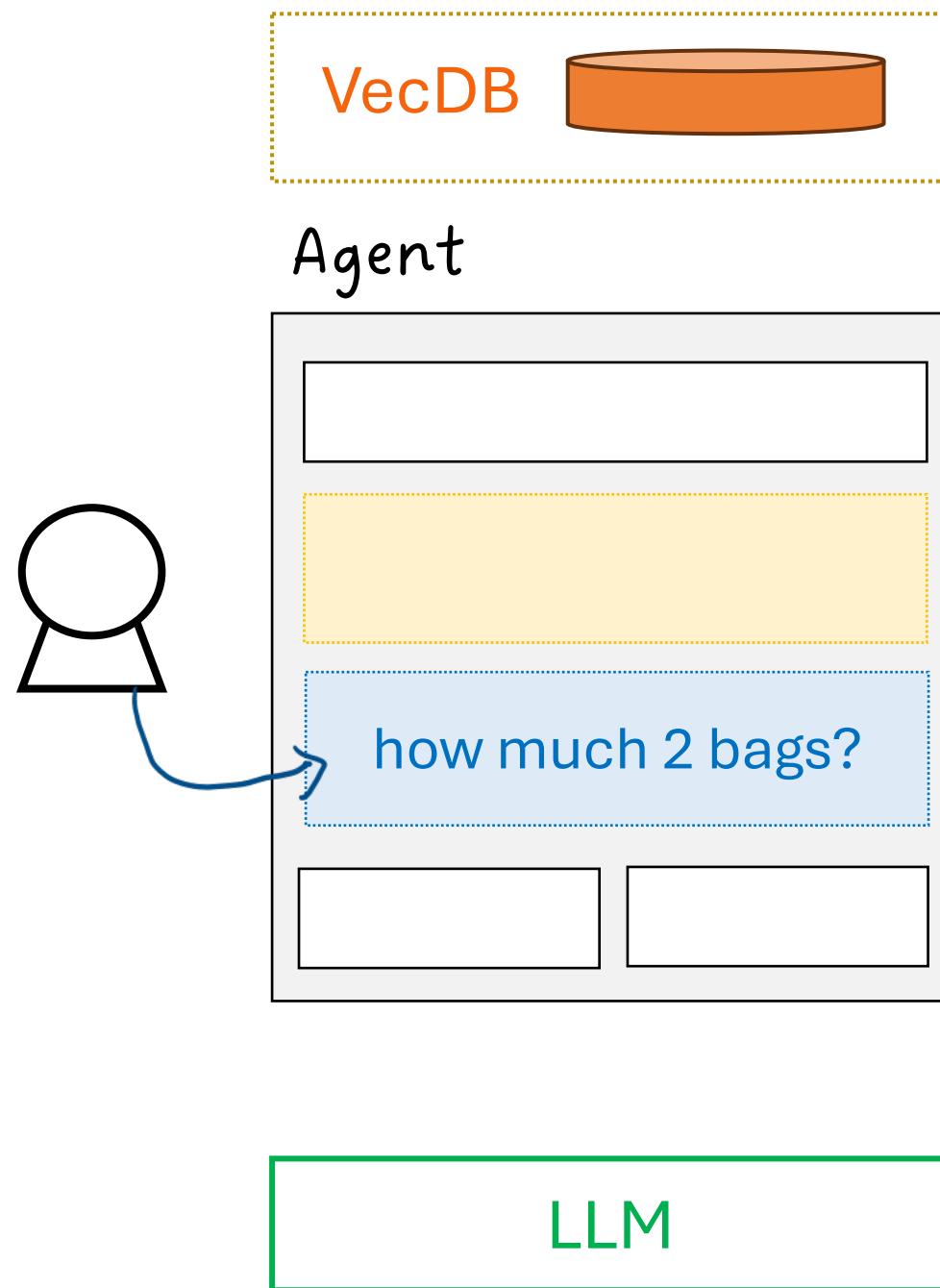
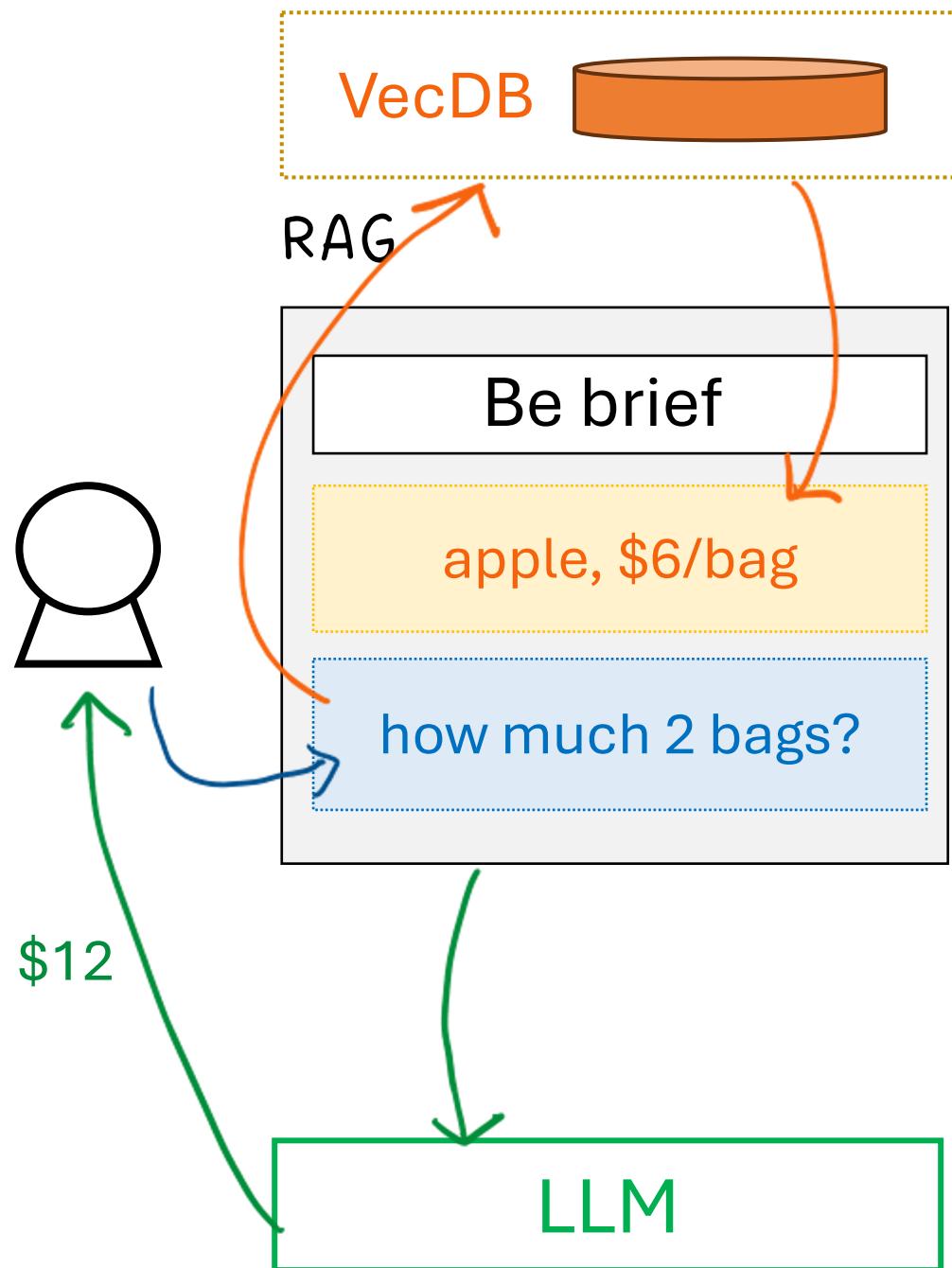
LLM

RAG -> Agent

Beginner's Guide to Multi-Agents - AI by Hand



University of Colorado
Boulder

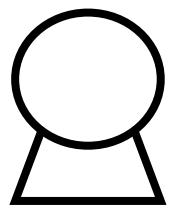


Multi-Agents

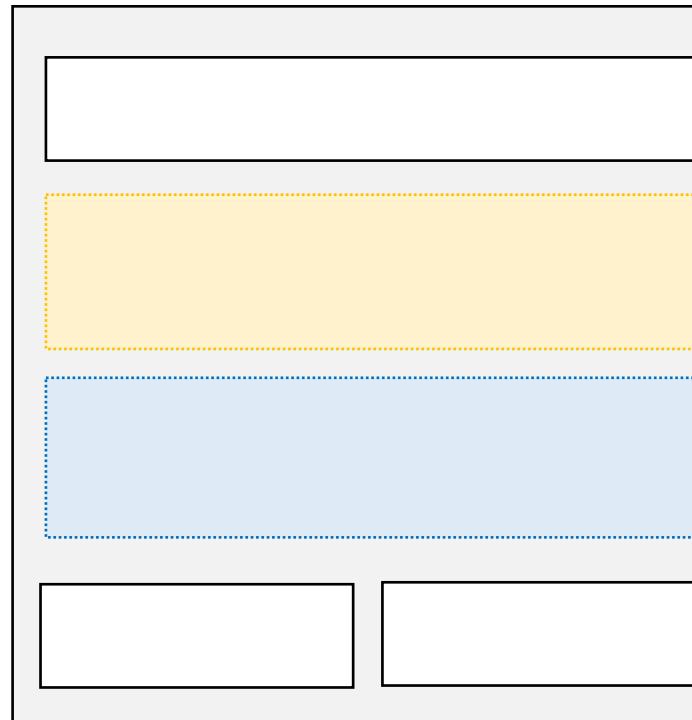
Beginner's Guide to Multi-Agents - AI by Hand



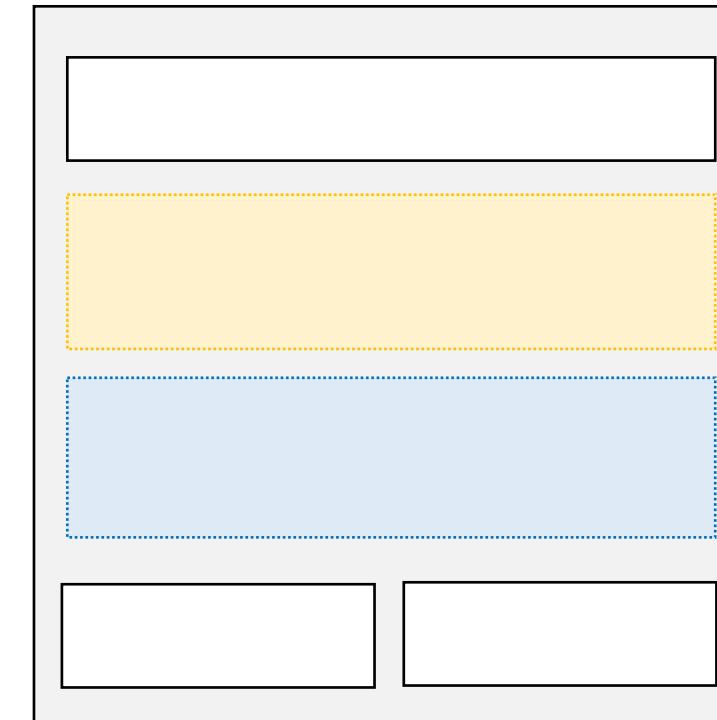
University of Colorado
Boulder

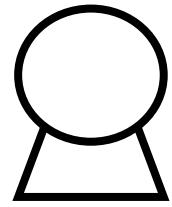


Agent

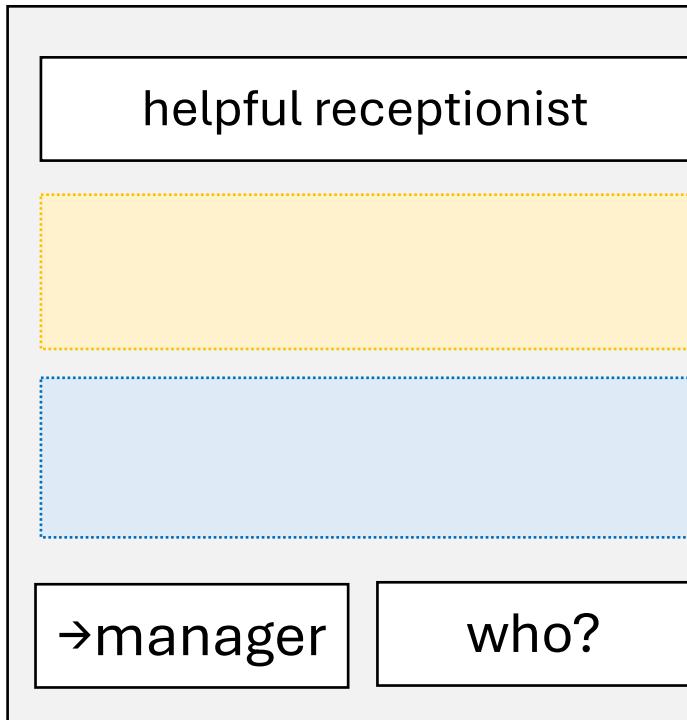


Agent

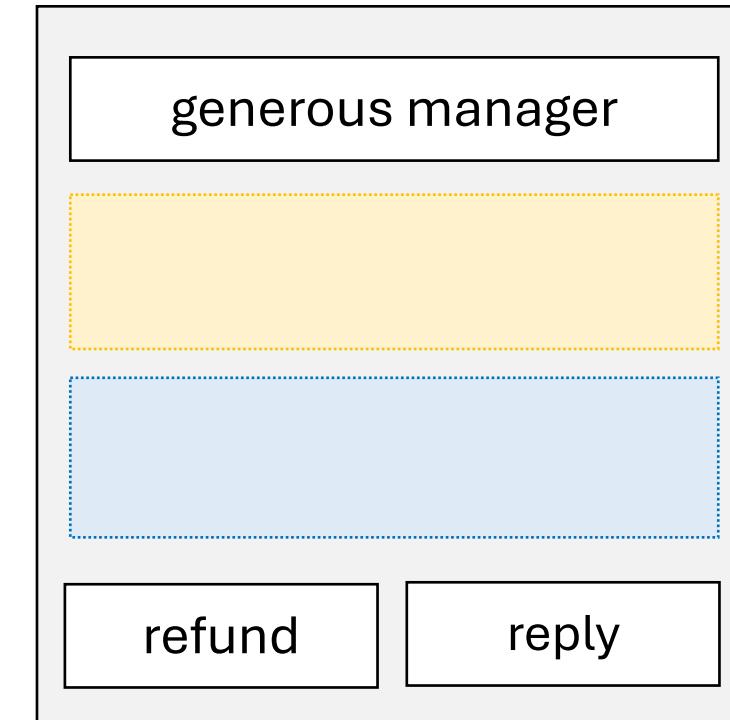




Agent



Agent



LLM



Swarm

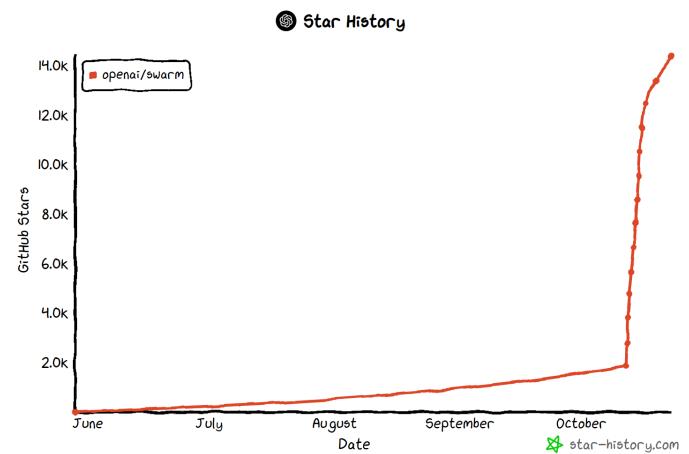


OpenAI Swarm

Beginner's Guide to Multi-Agents - AI by Hand 



University of Colorado
Boulder



```
from swarm import Swarm, Agent

client = Swarm()

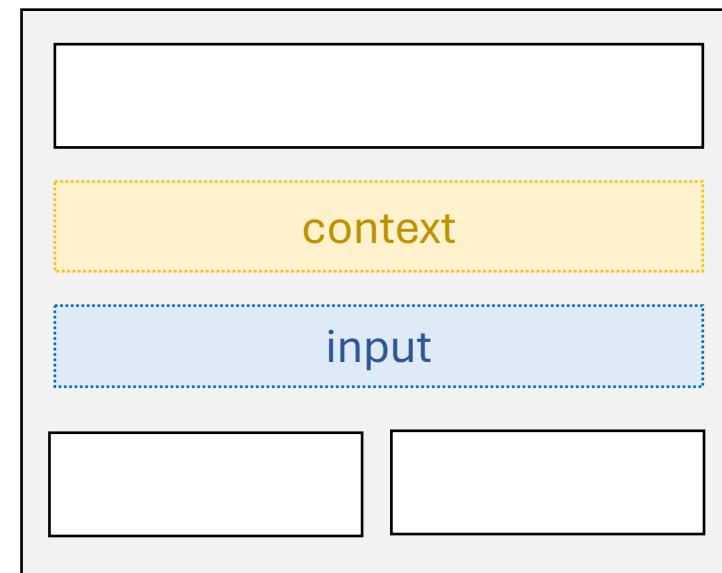
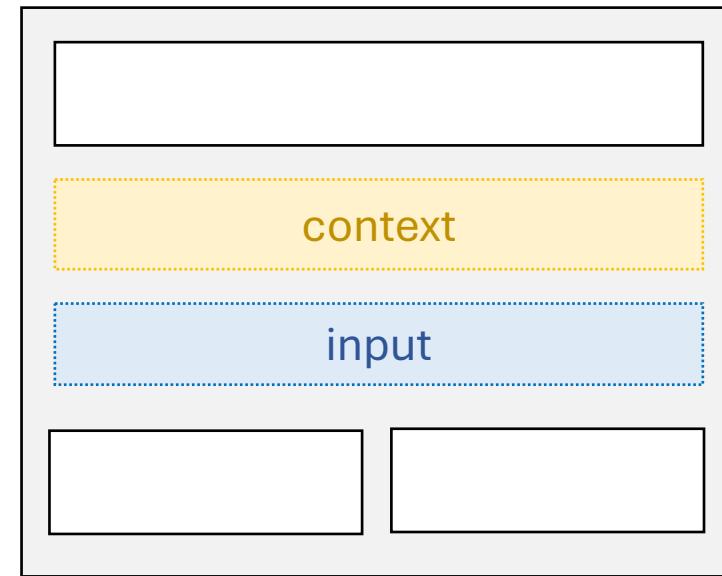
def transfer_to_agent_b():
    return agent_b

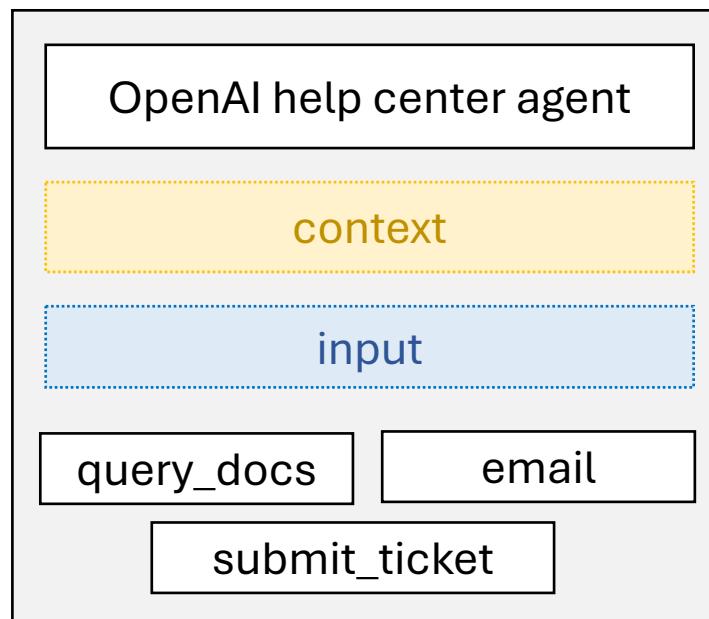
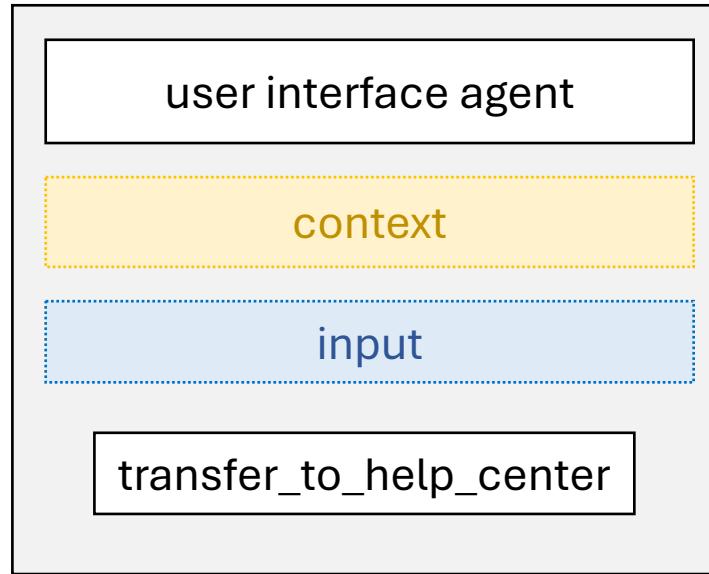
agent_a = Agent(
    name="Agent A",
    instructions="You are a helpful agent.",
    functions=[transfer_to_agent_b],
)

agent_b = Agent(
    name="Agent B",
    instructions="Only speak in Haikus.",
)

response = client.run(
    agent=agent_a,
    messages=[{"role": "user", "content": "I want to talk to agent B."}],
)

print(response.messages[-1]["content"])
```





```
user_interface_agent = Agent(  
    name="User Interface Agent",  
    instructions="You are a user interface agent that handles a",  
    functions=[transfer_to_help_center],  
)  
  
help_center_agent = Agent(  
    name="Help Center Agent",  
    instructions="You are an OpenAI help center agent who deals",  
    functions=[query_docs, submit_ticket, send_email],  
)
```

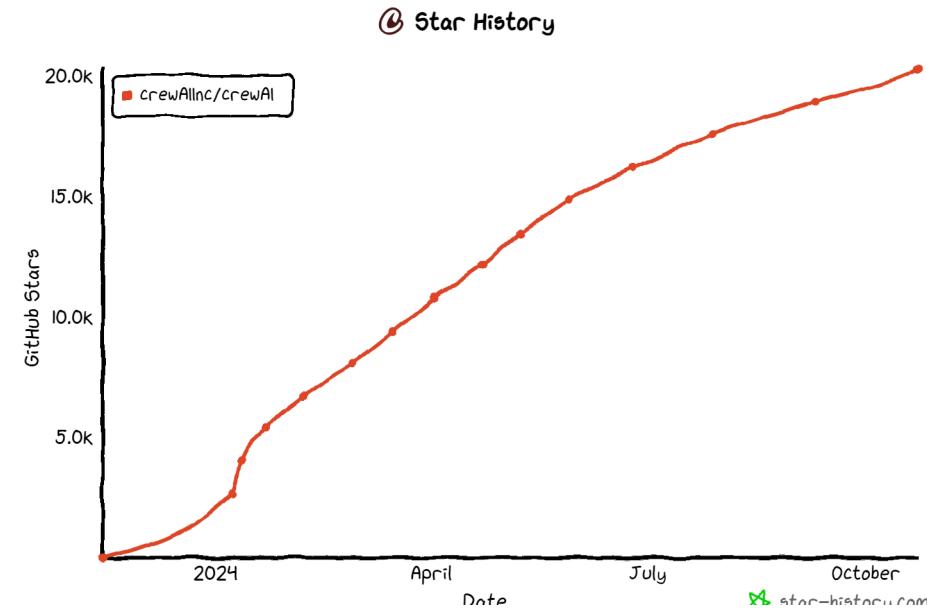


CrewAI

Beginner's Guide to Multi-Agents - AI by Hand 🖌️



University of Colorado
Boulder



```
# src/my_project/config/agents.yaml
```

```
researcher:
```

```
  role: >
```

```
    {topic} Senior Data Researcher
```

```
  goal: >
```

```
    Uncover cutting-edge developments in {topic}
```

```
  backstory: >
```

```
    You're a seasoned researcher with a knack for uncovering the latest developments in {topic}. Known for your ability to find the most information and present it in a clear and concise manner.
```



```
reporting_analyst:
```

```
  role: >
```

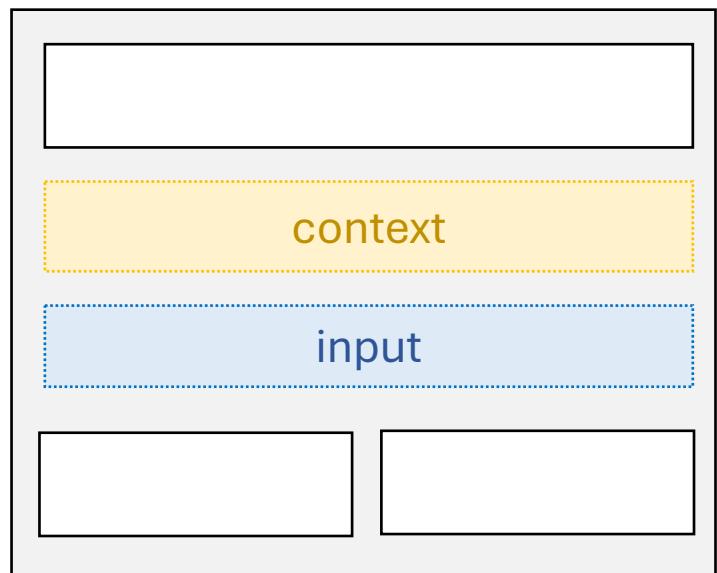
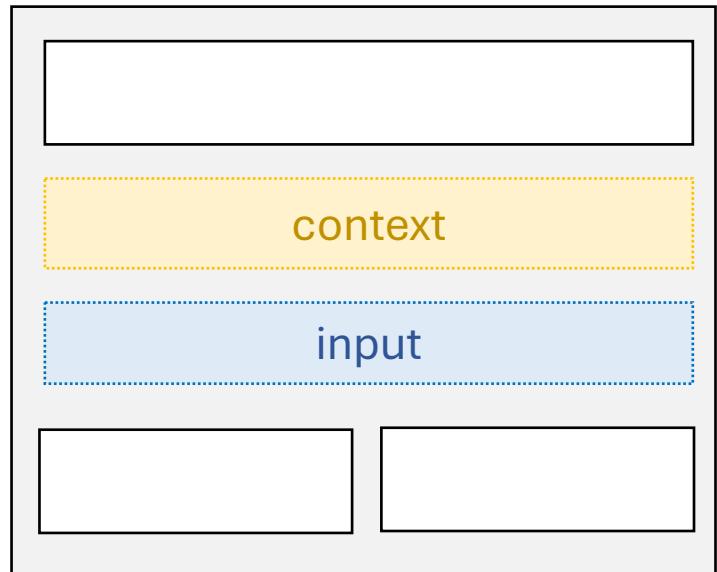
```
    {topic} Reporting Analyst
```

```
  goal: >
```

```
    Create detailed reports based on {topic} data analysis and research
```

```
  backstory: >
```

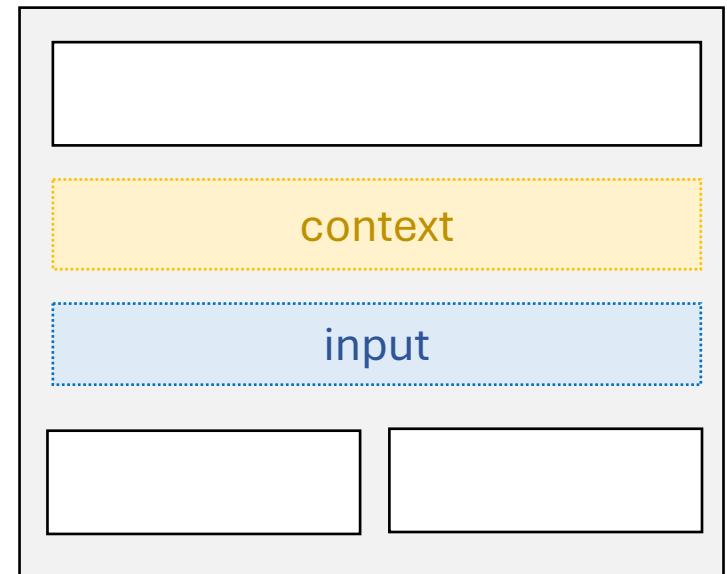
```
    You're a meticulous analyst with a keen eye for detail. You're known for your ability to turn complex data into clear and concise reports, making it easy for others to understand and act on the information you
```



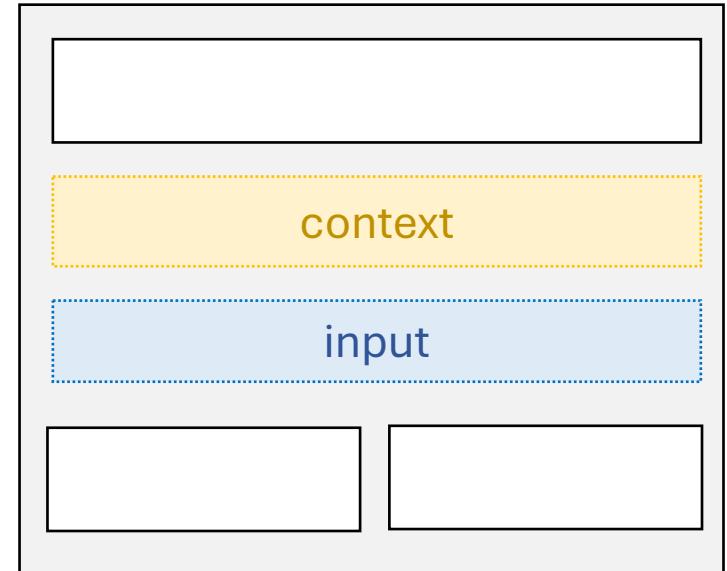
```
# src/my_project/config/tasks.yaml
research_task:
  description: >
    Conduct a thorough research about {topic}
    Make sure you find any interesting and relevant information given
    the current year is 2024.
  expected_output: >
    A list with 10 bullet points of the most relevant information at
agent: researcher

reporting_task:
  description: >
    Review the context you got and expand each topic into a full section
    Make sure the report is detailed and contains any and all relevant
  expected_output: >
    A fully fledged report with the main topics, each with a full section
    Formatted as markdown without '```'
agent: reporting_analyst
output_file: report.md
```

Researcher



Reporting Analyst

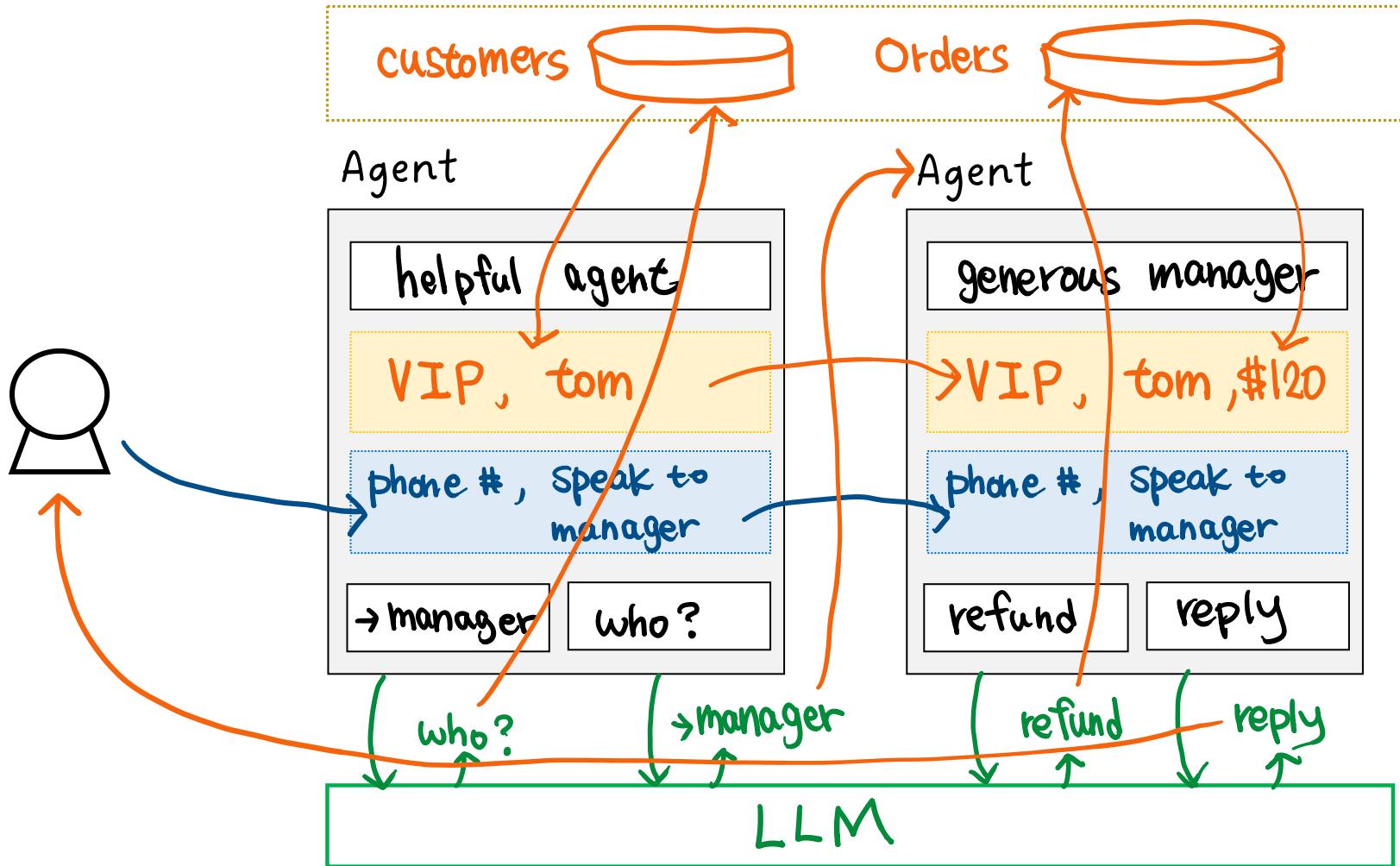


Process

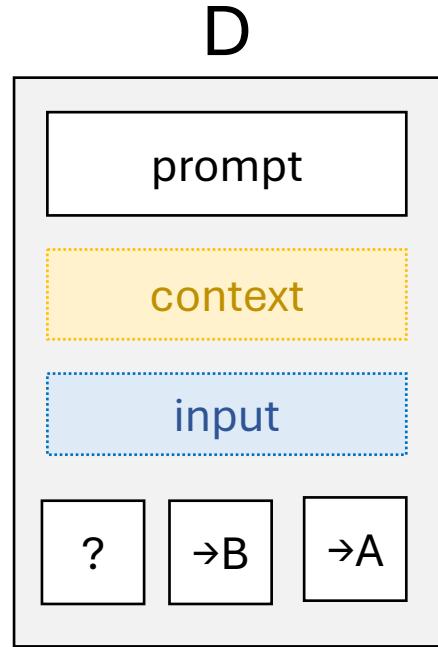
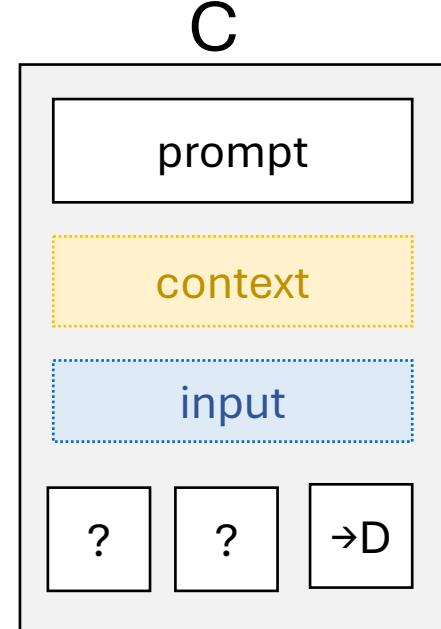
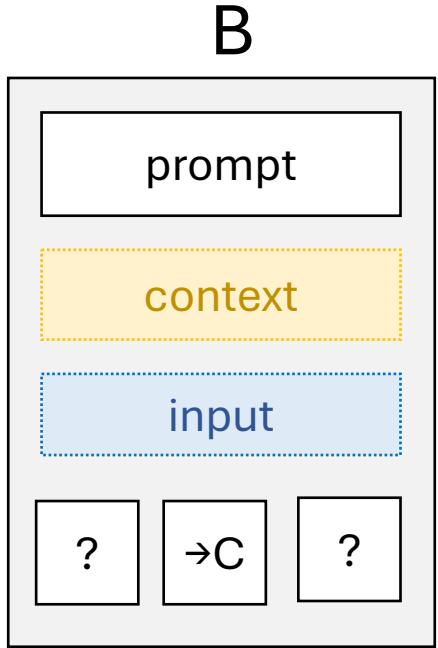
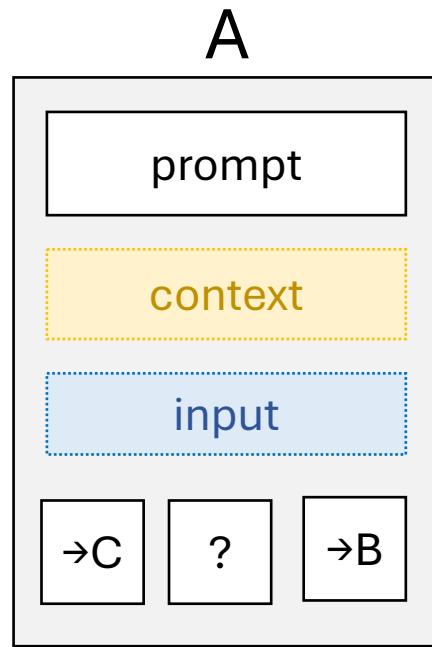
Beginner's Guide to Multi-Agents - AI by Hand



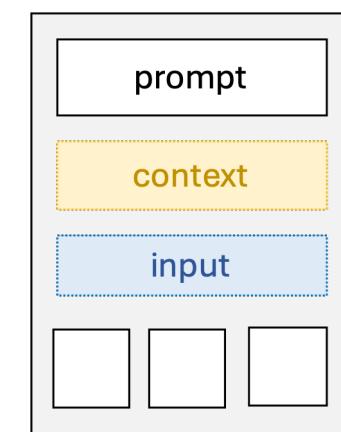
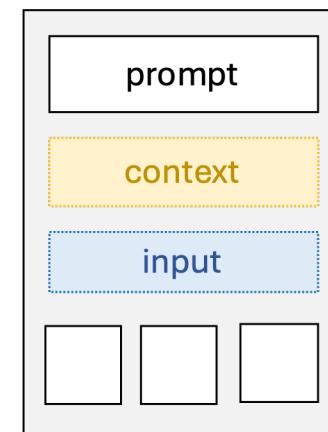
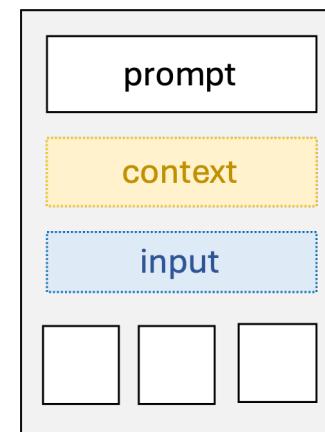
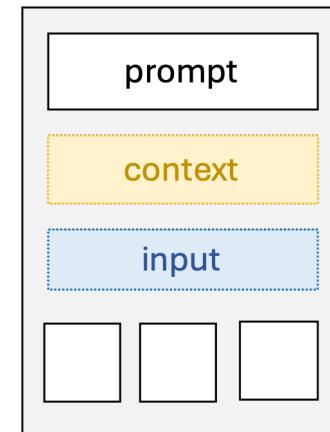
University of Colorado
Boulder



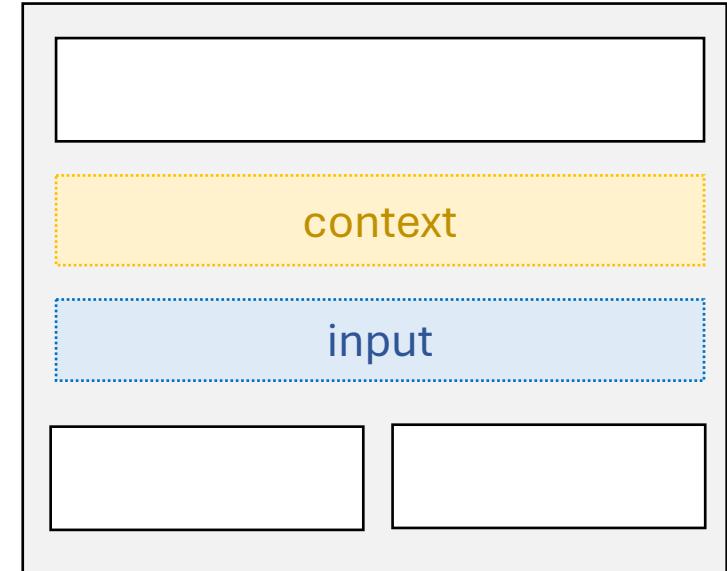
sequential



Hierarchical



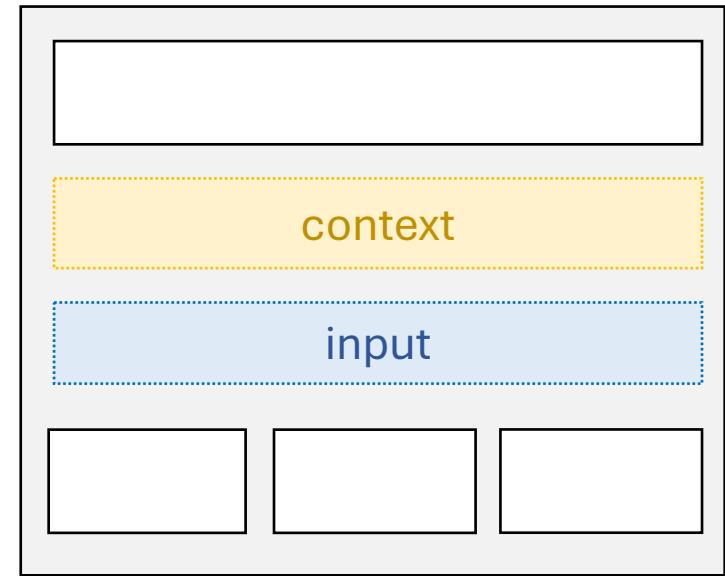
```
"hierarchical_manager_agent": {  
    "role": "Crew Manager",  
    "goal": "Manage the team to complete the task in the best  
        way possible.",  
    "backstory": "You are a seasoned manager with a knack for  
        getting the best out of your team.\nYou are also known  
        for your ability to delegate work to the right people,  
        and to ask the right questions to get the best out of  
        your team.\nEven though you don't perform tasks by  
        yourself, you have a lot of experience in the field,  
        which allows you to properly evaluate the work of your  
        team members."  
},
```



```

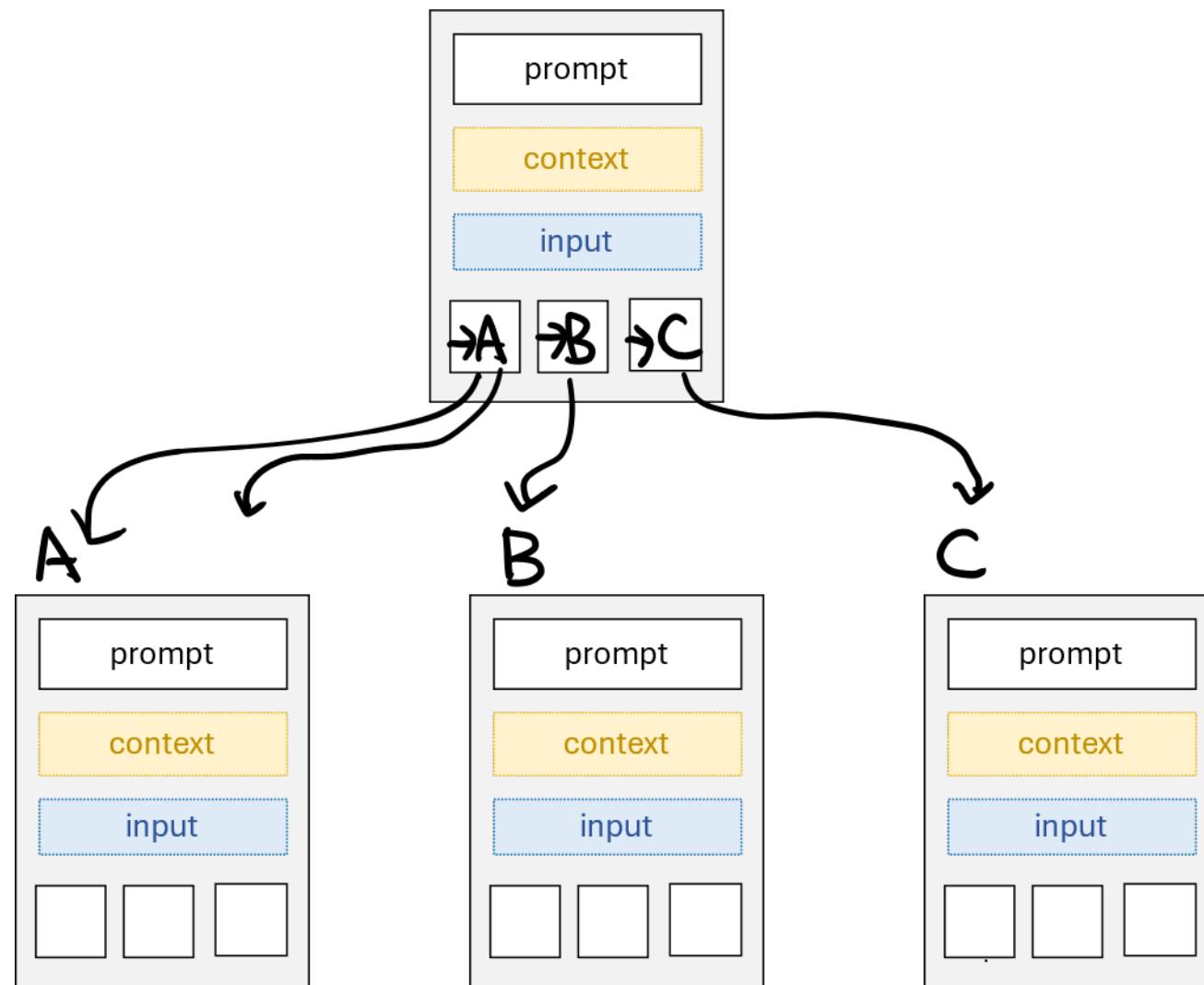
def _create_manager_agent(self):
    i18n = I18N(prompt_file=self.prompt_file)
    if self.manager_agent is not None:
        self.manager_agent.allow_delegation = True
        manager = self.manager_agent
        if manager.tools is not None and len(manager.tools) > 0:
            self._logger.log(
                "warning", "Manager agent should not have tools", color="orange"
            )
            manager.tools = []
            raise Exception("Manager agent should not have tools")
        manager.tools = self.manager_agent.get_delegation_tools(self.agents)
    else:
        self.manager_llm = (
            getattr(self.manager_llm, "model_name", None)
            or getattr(self.manager_llm, "deployment_name", None)
            or self.manager_llm
        )
        manager = Agent(
            role=i18n.retrieve("hierarchical_manager_agent", "role"),
            goal=i18n.retrieve("hierarchical_manager_agent", "goal"),
            backstory=i18n.retrieve("hierarchical_manager_agent", "backstory"),
            tools=AgentTools(agents=self.agents).tools(),
            llm=self.manager_llm,
            verbose=self.verbose,
        )
        self.manager_agent = manager
manager.crew = self

```



Graph

Manager

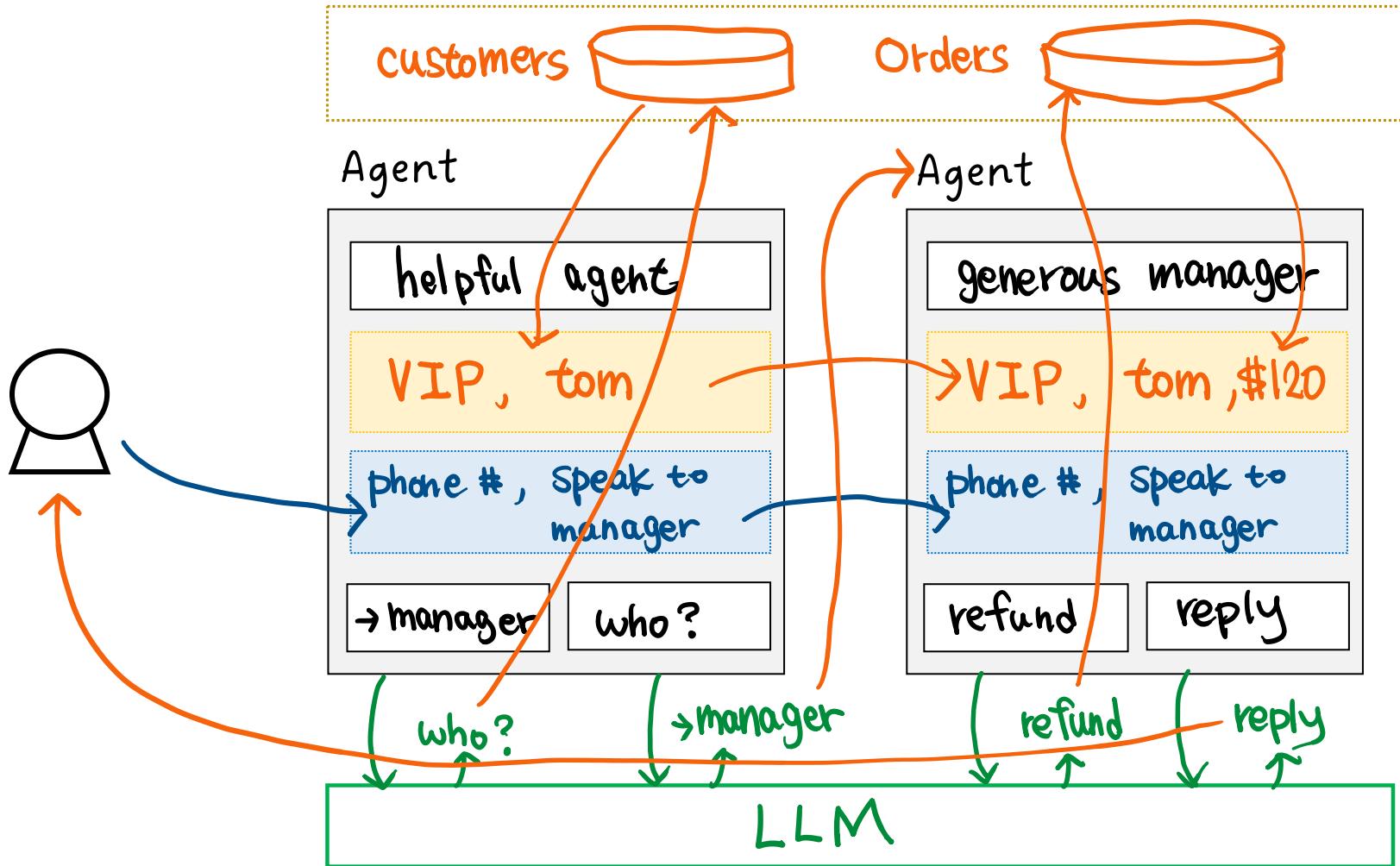


Communication

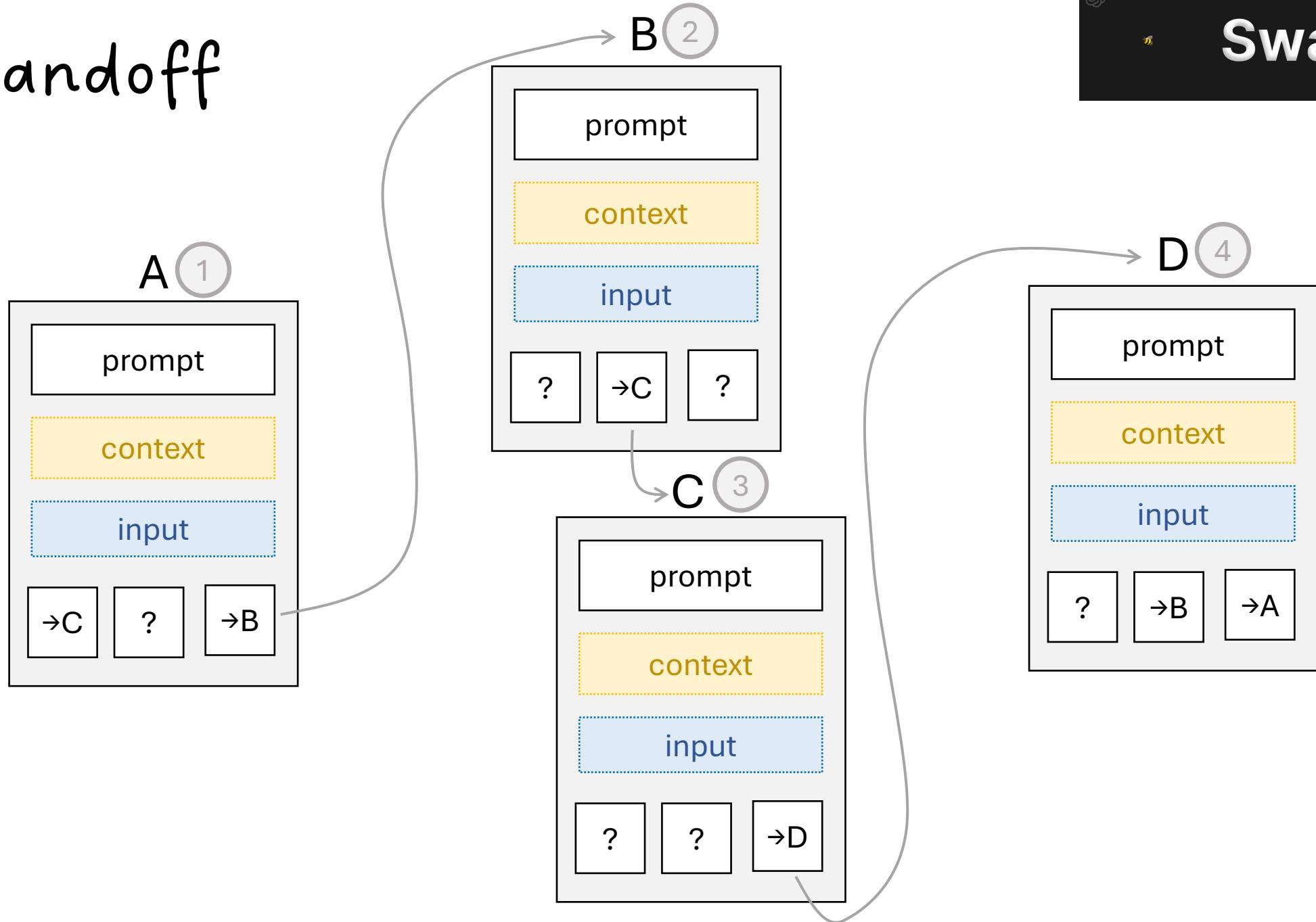
Beginner's Guide to Multi-Agents - AI by Hand



University of Colorado
Boulder

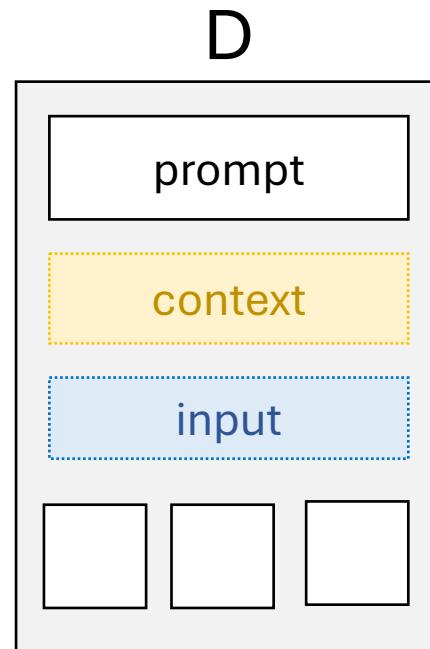
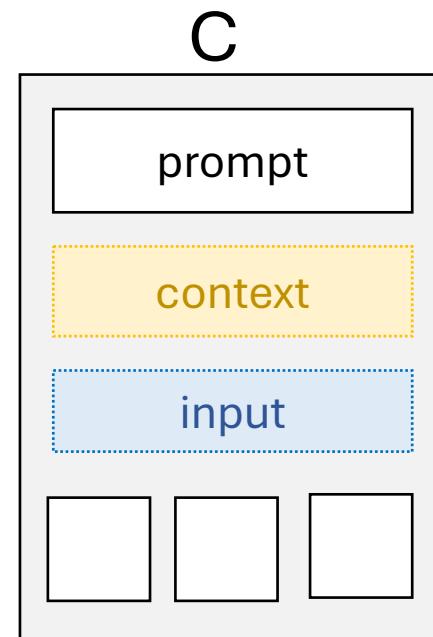
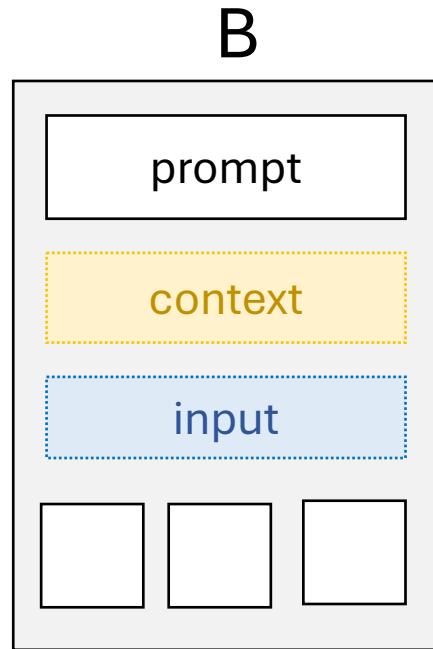
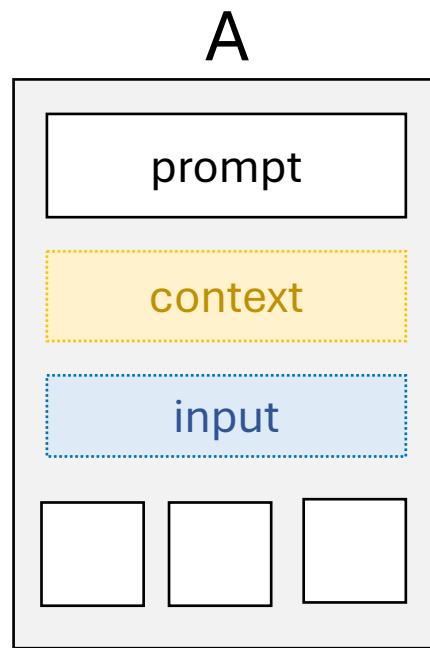


Handoff



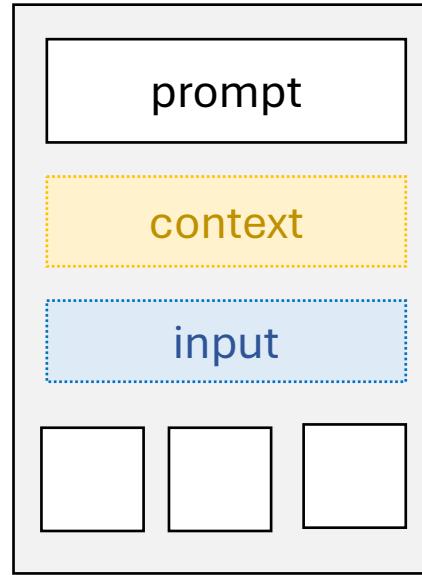
Q/A

crewai

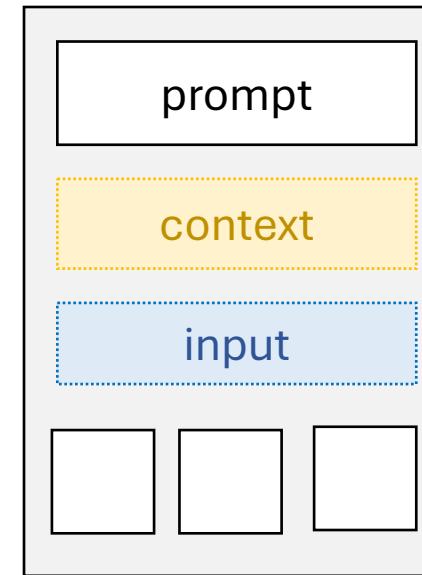


Messages

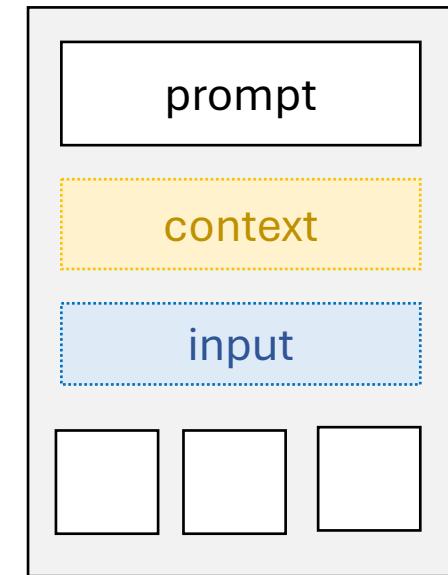
A



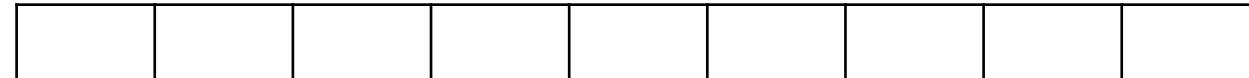
B



C



Messages Queue

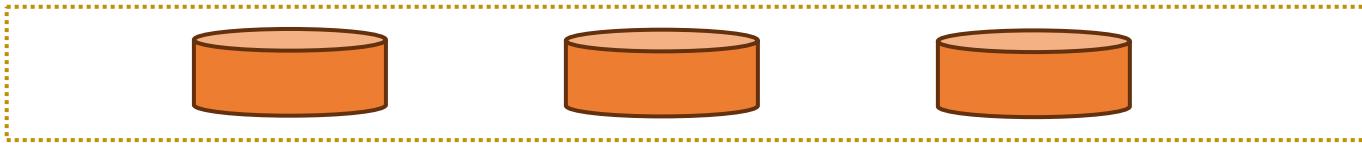


Memory

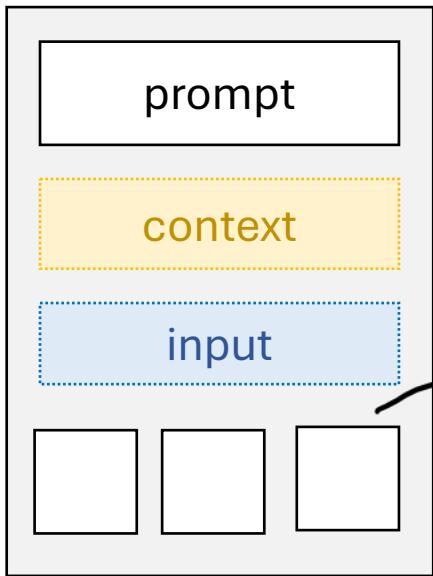
Beginner's Guide to Multi-Agents - AI by Hand



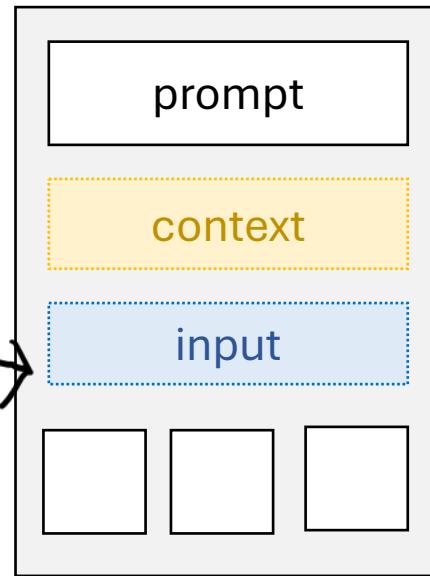
University of Colorado
Boulder



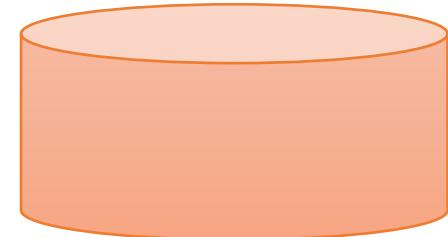
A



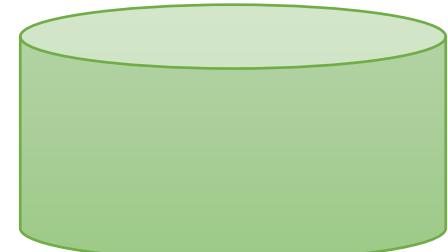
B



Short-term
Memory



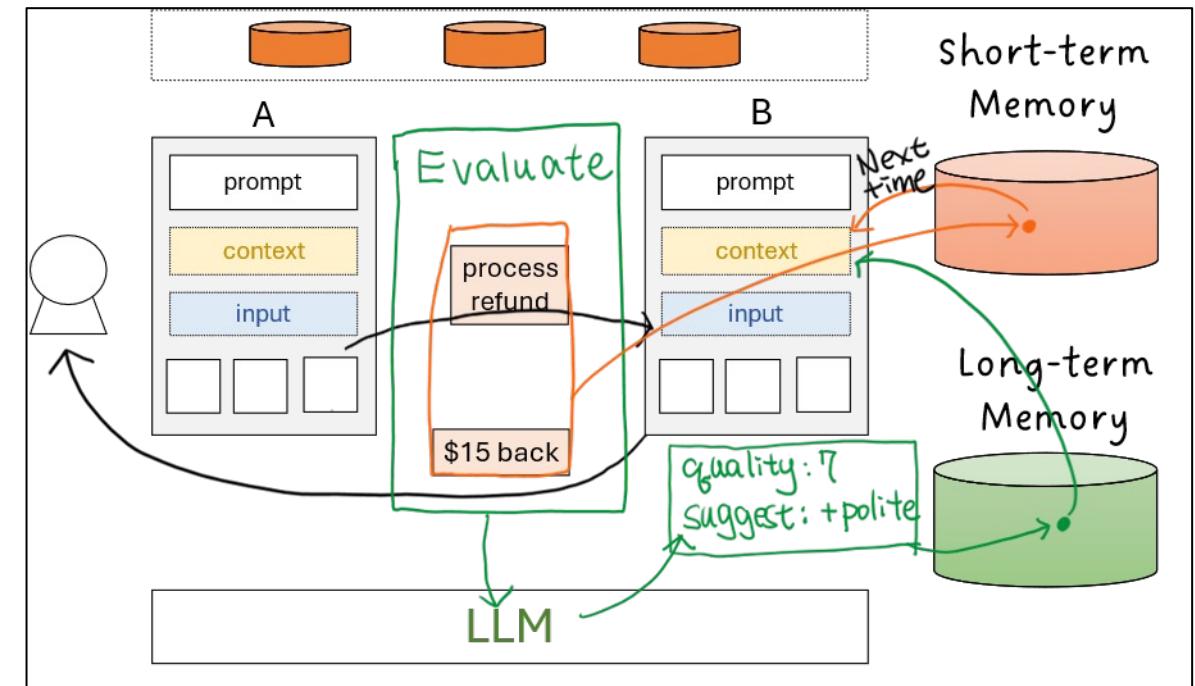
Long-term
Memory



```

def _create_short_term_memory(self, output) -> None:
    """Create and save a short-term memory item if conditions are met."""
    if (
        self.crew
        and self.agent
        and self.task
        and "Action: Delegate work to coworker" not in output.text
    ):
        try:
            if (
                hasattr(self.crew, "_short_term_memory")
                and self.crew._short_term_memory
            ):
                self.crew._short_term_memory.save(
                    value=output.text,
                    metadata={
                        "observation": self.task.description,
                    },
                    agent=self.agent.role,
                )
        except Exception as e:
            print(f"Failed to add to short term memory: {e}")
            pass

```



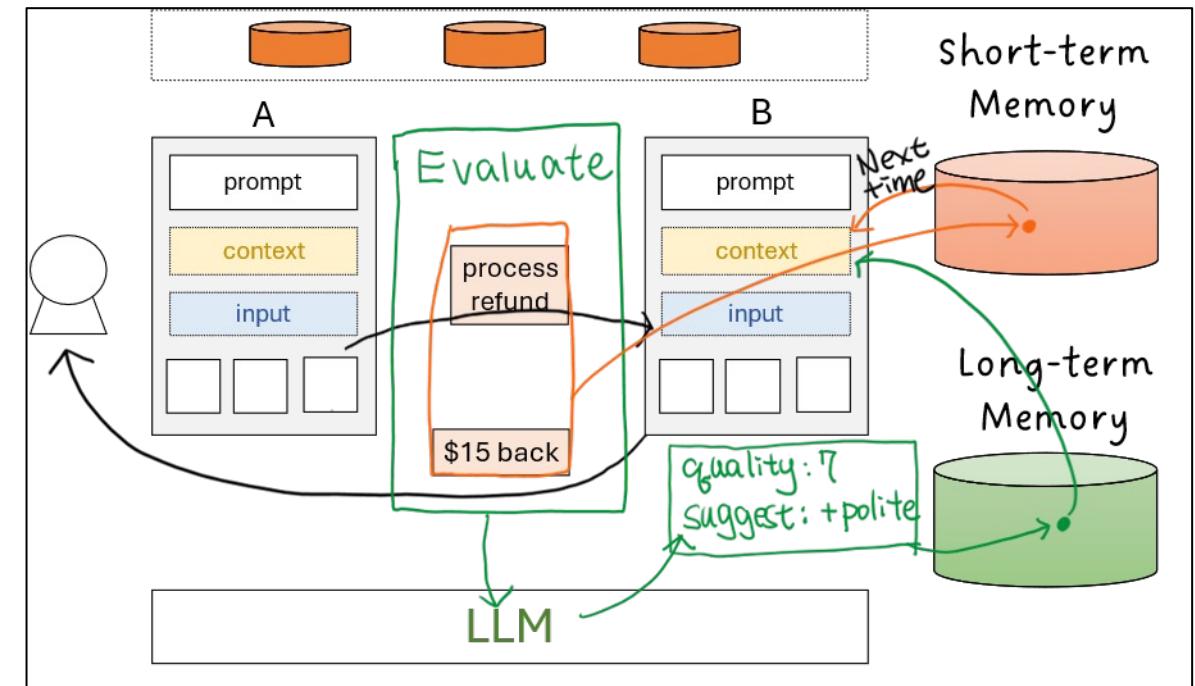
```

def _create_long_term_memory(self, output) -> None:
    """Create and save long-term and entity memory items based on evaluation."""
    if (
        self.crew
        and self.crew.memory
        and self.crew._long_term_memory
        and self.crew._entity_memory
        and self.task
        and self.agent
    ):
        try:
            ltm_agent = TaskEvaluator(self.agent)
            evaluation = ltm_agent.evaluate(self.task, output.text)

            if isinstance(evaluation, ConverterError):
                return

            long_term_memory = LongTermMemoryItem(
                task=self.task.description,
                agent=self.agent.role,
                quality=evaluation.quality,
                datetime=str(time.time()),
                expected_output=self.task.expected_output,
                metadata={
                    "suggestions": evaluation.suggestions,
                    "quality": evaluation.quality,
                },
            )
            self.crew._long_term_memory.save(long_term_memory)
        except Exception as e:
            print(f"Error creating long-term memory: {e}")

```



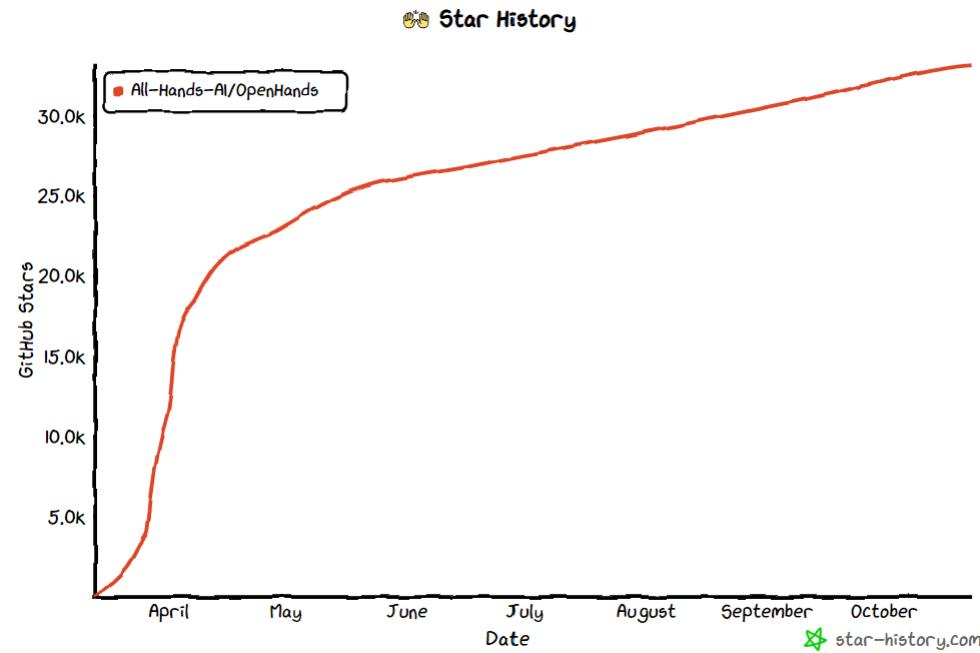


OpenHands

Beginner's Guide to Multi-Agents - AI by Hand 🤖



University of Colorado
Boulder



Chat

The data has been successfully saved to `commits_per_week.csv`. Now, let's visualize the data using `matplotlib`.

First, we need to install `matplotlib` if it's not already installed.

Great! Now, let's write a script to read the `commits_per_week.csv` file and use `matplotlib`. I'll create a new Python script.

Let's write the code to read the `commits_per_week.csv` file and use `matplotlib`. I'll update the `visualize_commits.py` necessary code.

Sure! Let me perform a Google search for OpenDevin.

Here are some of the top results:

- OpenDevin GitHub Pages:** OpenDevin is an autonomous AI capable of executing complex tasks and collaborating actively with users during development.
- Introducing OpenDevin CodeAct 1.0:** OpenDevin CodeAct 1.0 is a new feature designed to handle coding tasks. It is based on CodeAct 1.0 and consolidates LLM agents' capabilities.
- OpenDevin GitHub:** OpenDevin GitHub is a repository and resources related to OpenDevin.
- OpenDevin: Open Source AI:** OpenDevin is a self-contained AI designed to perform complex tasks and collaborate with users during development.

Anyone use OpenDevin?: Re: Discussion on Reddit about the AI. Would you like to visit any of the links above or provide more information on a specific topic?

The `commits_per_week.png` file has been successfully created. Let's visualize the number of commits per week.

Message assistant...

Code Editor

STDOUT/STDERR

<IPython.core.display.Image object>

Number of Commits per Week (Starting March 2024)

Browser

https://www.google.com/search?q=OpenDevin

Google

OpenDevin

All Images News Videos Shopping Tools

Jupyter IPython

Chat

Let's add a retry mechanism to the script to handle this situation. We will wait for a few seconds and retry the request if we receive a 202 status code. I'll update the script accordingly.

The script has been updated to include a retry mechanism. Let's execute the script again to fetch the number of commits per week for the `OpenDevin/OpenDevin` repository.

The script successfully fetched the number of commits per week for the `OpenDevin/OpenDevin` repository. Here are the results:

```
Week starting 1710028800: 4 commits
Week starting 1710633600: 41 commits
Week starting 1711238400: 109 commits
Week starting 1711843200: 106 commits
Week starting 1712448000: 63 commits
Week starting 1713052800: 78 commits
Week starting 1713657600: 83 commits
Week starting 1714262400: 77 commits
Week starting 1714867200: 70 commits
Week starting 1715472000: 92 commits
Week starting 1716076800: 103 commits
Week starting 1716681600: 83 commits
Week starting 1717286400: 57 commits
```

If you need any further assistance or modifications, please let me know!

Message assistant...

Code Editor

github_commits.py

```
1 import requests
2 import json
3 import time
4
5 def get_commits_per_week(owner, repo):
6     url = f'https://api.github.com/repos/{owner}/{repo}/stats/commit_activity'
7     while True:
8         response = requests.get(url)
9
10        if response.status_code == 200:
11            data = response.json()
12            for week in data:
13                print(f"Week starting {week['week']}: {week['total']} commits")
14            break
15        elif response.status_code == 202:
16            print("Data is being processed. Retrying in 5 seconds...")
17            time.sleep(5)
18        else:
19            print(f"Failed to fetch data: {response.status_code}")
20            break
21
22    if __name__ == '__main__':
23        owner = 'OpenDevin'
24        repo = 'OpenDevin'
```

Terminal (read-only)

```
Week starting 1710633600: 41 commits
Week starting 1711238400: 109 commits
Week starting 1711843200: 106 commits
Week starting 1712448000: 63 commits
Week starting 1713052800: 78 commits
Week starting 1713657600: 83 commits
Week starting 1714262400: 77 commits
Week starting 1714867200: 70 commits
Week starting 1715472000: 92 commits
Week starting 1716076800: 103 commits
Week starting 1716681600: 83 commits
Week starting 1717286400: 57 commits
```

\$ Agent is awaiting user input...

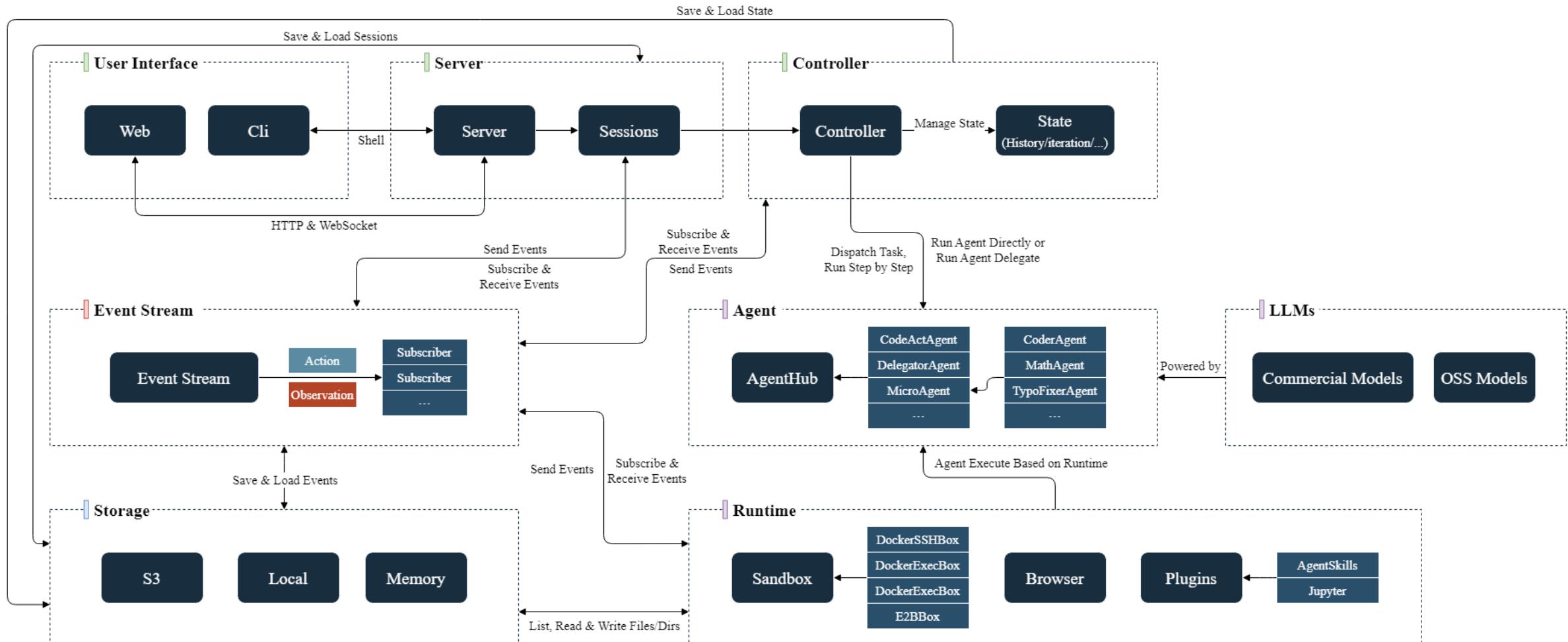
Agents

Process

Communication

Memory

LLM





CodeR

Beginner's Guide to Multi-Agents - AI by Hand



University of Colorado
Boulder

Agents

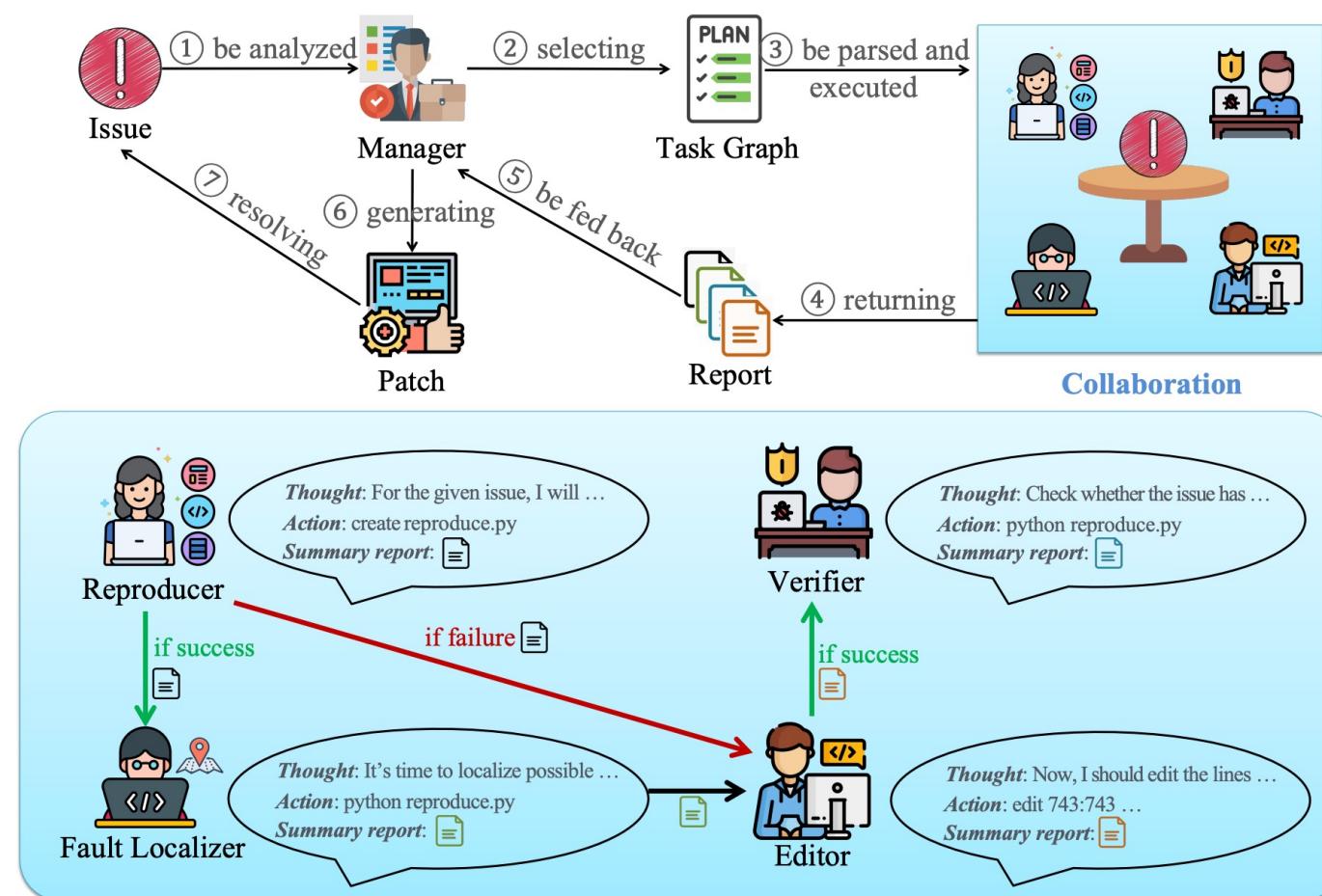


Process

Communication

Memory

LLM



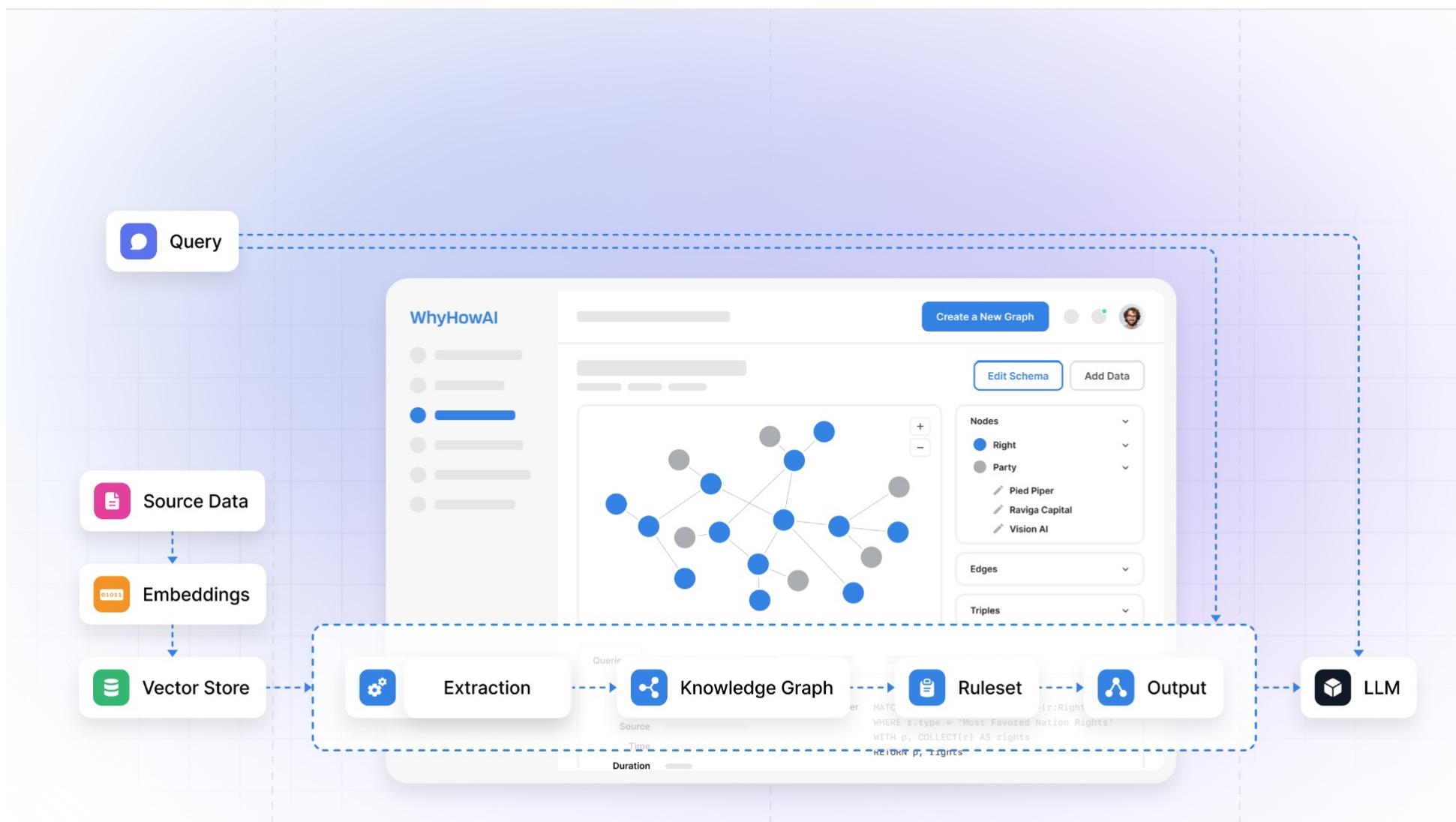
Multi-Agents + Graphs

whyHow.ai

Beginner's Guide to Multi-Agents - AI by Hand



University of Colorado
Boulder



Legal

The management of control functions by the Board and the Chief Compliance Officer (CCO) involves several key responsibilities as outlined in the compliance document:

Responsibilities of the Board:

1. Oversight of Compliance Risk Management: The Board is responsible for overseeing the management of compliance risk within the financial institution.
2. Approval and Support for the Compliance Function: The Board must ensure that the compliance function and the CCO have the appropriate standing, authority, and independence to carry out their duties effectively. The Board must also ensure that the compliance function is adequately resourced.
3. Engagement with the CCO: The Board should engage with the CCO regularly to discuss issues faced by the compliance function and ensure the CCO has direct access to the Board.
4. Annual Evaluation: The Board must evaluate the effectiveness of the institution's management of compliance risk at least annually, considering assessments from senior management, internal audit, and the CCO.

Responsibilities of the Chief Compliance Officer (CCO):

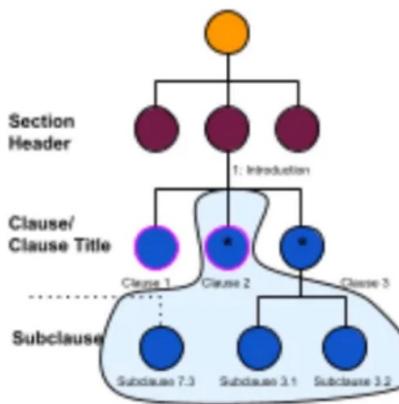
1. Coordination and Management of Compliance Risk: The CCO is responsible for coordinating the identification, management, and monitoring of compliance risk across the institution. This includes ensuring compliance monitoring and testing are consistent across the organization.
2. Reporting and Advising: The CCO must report regularly to senior management on the findings and analysis of compliance risks and must ensure that reports are readily available to internal audit and regulatory authorities. The CCO also advises the Board and senior management on legal and regulatory requirements, keeping them informed of developments and their implications.
3. Independence: The CCO must maintain independence from business lines to effectively carry out the role of a control function. This includes ensuring that there is no conflict of interest in their responsibilities, reporting lines, or remuneration.
4. Resources and Training: The CCO must ensure that the compliance function is sufficiently resourced with officers who have the necessary qualifications and experience. The CCO is also responsible for ensuring that adequate training is provided to officers on relevant legal and regulatory requirements.

Interaction with Other Control Functions:

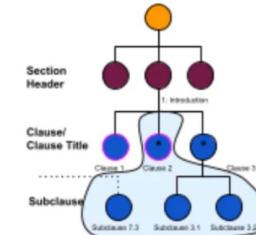
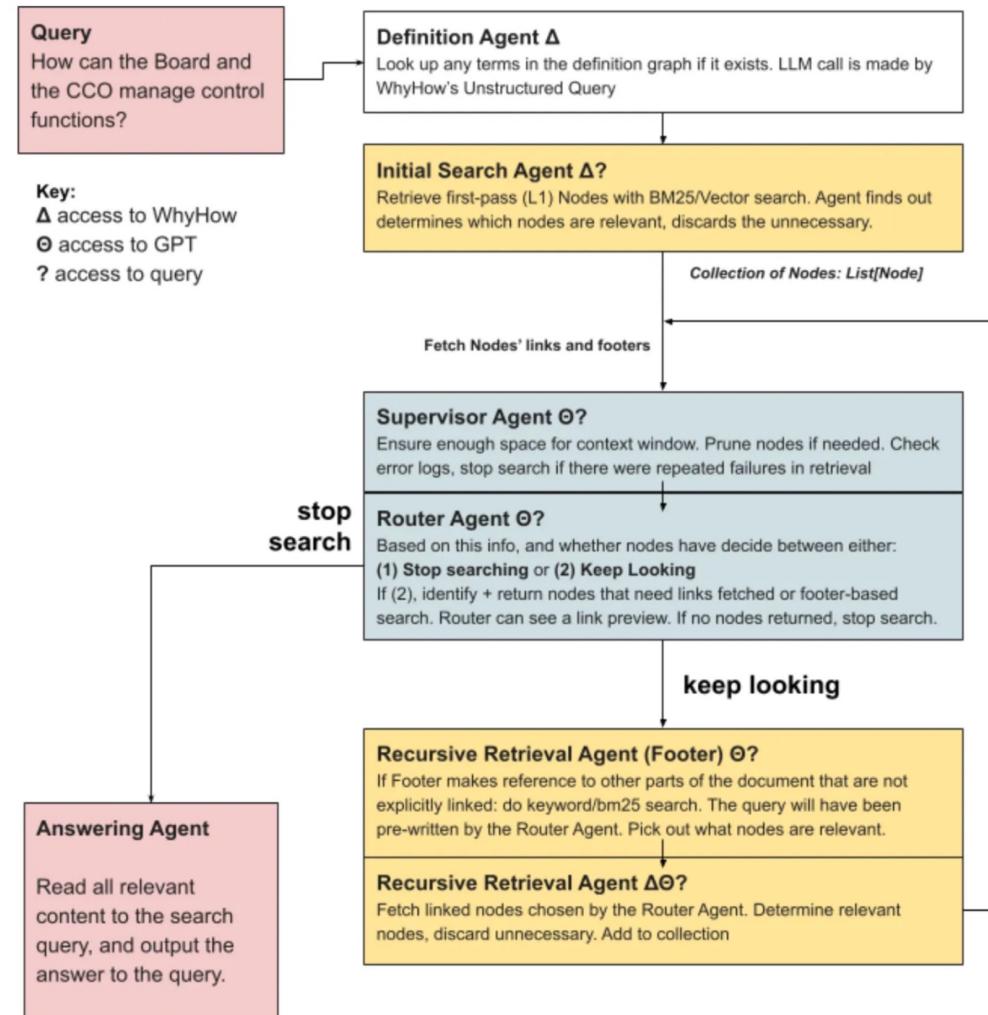
The compliance function should be organized in a way that allows effective management of compliance risk, considering the size, nature, and complexity of the financial institution's operations. Where responsibilities are shared between a dedicated compliance unit and other control functions, clear documentation and coordination are required to ensure that the CCO can perform their duties effectively without impairing independence or focus. By fulfilling these responsibilities, the Board and the CCO can ensure that the institution's compliance risks are managed effectively, supporting the overall safety and soundness of the financial institution.

Query

How can the board and the CCO manage control functions?



Agents Process Communication Memory LLM



Collections of retrieved subgraphs are passed through the pipeline (cyan highlight)
Nodes marked with * have extra footer info
Nodes with pink outline a link to another link
As the collection passes through the pipeline, they can be 'marked', either for redundancy, added extra context e.g. what section the clause is in
Some agents/tools can add to the collection

Node:

- id : str
- type : ElementType
- content : str
- context: Dict[str, Any]
- children: Dict[str, Node]

For those looking to understand what each agent does, we include code snippets for each agent in the Appendix

Q/A

Beginner's Guide to Multi-Agents - AI by Hand



University of Colorado
Boulder