17 Apr 2025

AI Agents

Materials:

https://github.com/lpomoeabatatas/ubts_day_02

Outline

1. AI Agents Basics
2. Demo + Code Walkthrough
3. Project setup

AI Agents

What Are AI Agents

- An artificial intelligence (AI) agent is a software program that can interact with its environment, collect data, and use the data to perform self-determined tasks to meet predetermined goals.

- **Humans set goals, but an AI agent independently chooses the best actions and autonomous decision-making it needs to perform to achieve those goals**

Image: Microsoft Stock Images, https://bernardmarr.com/

Core Components of AI Agent

TASK REQUEST

GUARDRAILS

PERCEPTION

PLANNING

REASONING

MEMORY

TOOLS

Type of Agents

TYPES	KEY FEATURES	EXAMPLES
Simple Reflex Agent	Reacts to current percepts using predefined condition-action rules. No memory or planning,	Auto plant watering, Traffic Lights
Model-Based Agent	Uses internal models of the world to maintain environment state and history.	Robot vacuum cleaner
Goal-Based Agent	Strives to attain a specific goal. Use planning and reasoning to choose sequence of actions.	Navigation system
Utility-Based Agent	Evaluate outcomes with utility function to choose actions that maximize some benefits.	A navigation system that recommends the route to your destination that optimizes fuel efficiency
Learning Agent	As below, but with the ability to learn and keep knowledge	Personalized recommendation system

Agentic Design Patterns

- Reflection: Self-critique and improvement
- Tool Use: External tools to enhance performance
- Planning / Reasoning: Generating and following multi-step plans
- Multi-Agent Collaboration: Coordinated problem solving

Agentic Design Patterns: Reflection

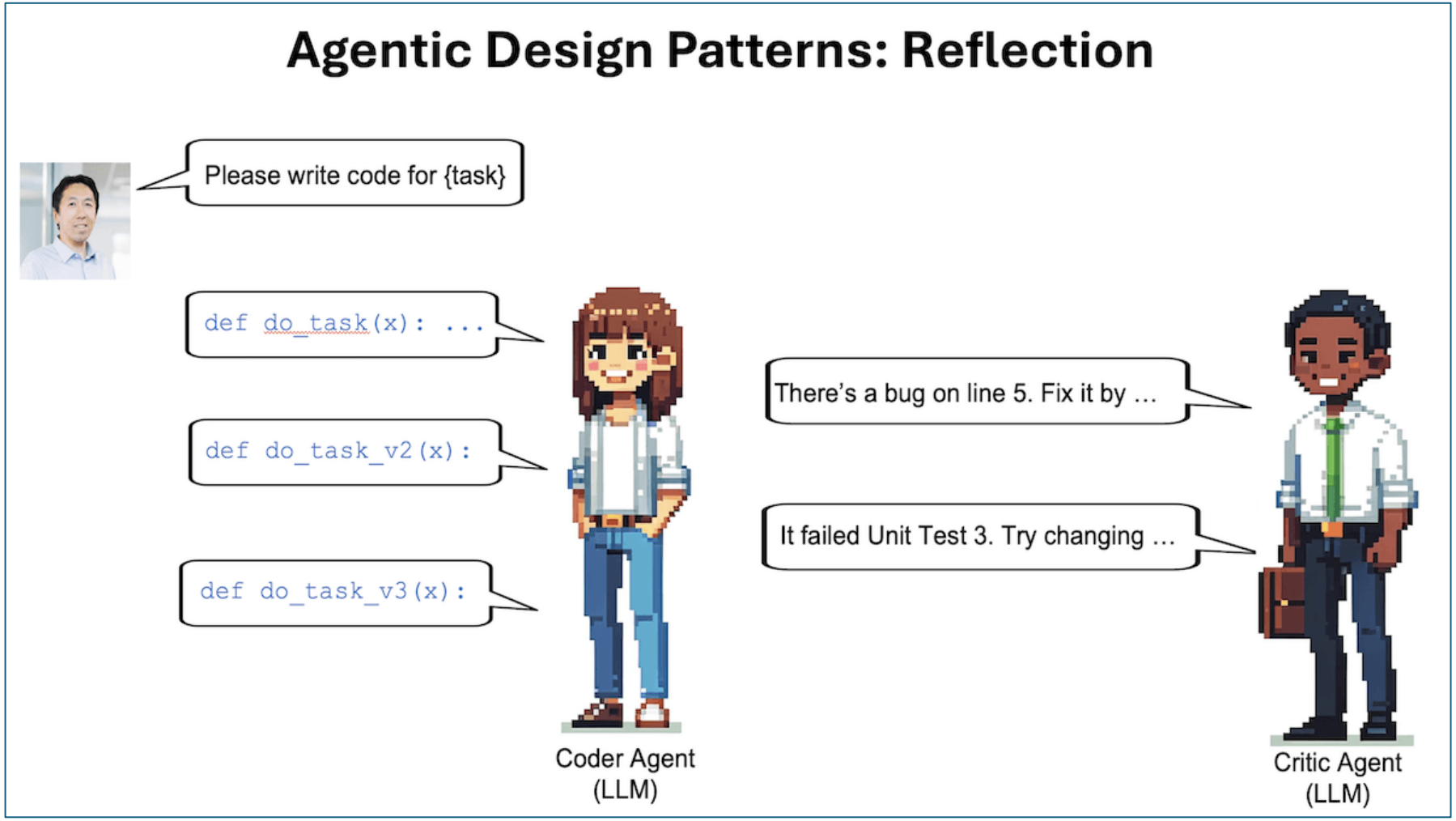


Image:THE BATCH, Deeplearning.AI

https://www.deeplearning.ai/the-batch/agentic-design-patterns-part-2-reflection/?ref=dl-staging-website.ghost.io

Agentic Design Patterns: Tool Use

Agentic Design Patterns: Tool Use

Web search tool

You

What is the best coffee maker according to reviewers?

Copilot

Searching for best coffee maker according to reviewers

Example from Bing CoPilot

Code execution tool

You

If I invest $100 at compound 7% interest for 12 years, what do I have at the end?

```
principal = 100
interest_rate = 0.07
years = 12
value = principal*(1 + interest_rate)**years
```

Example from ChatGPT

Image:THE BATCH, Deeplearning.AI

Agentic Design Patterns: Planning

image.jpg

final.jpg

Pose Determination

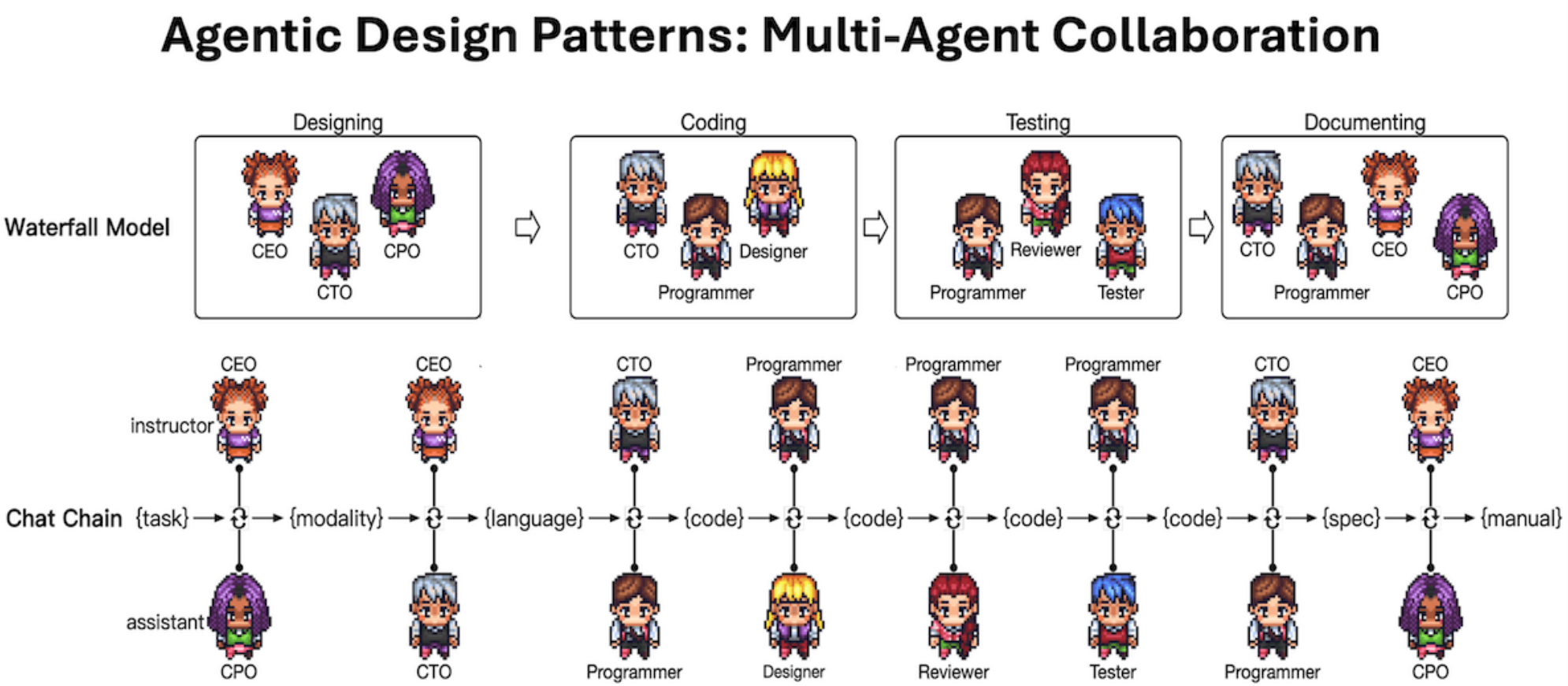
openpose model

Pose-to-Image

google/vit model

Image:THE BATCH, Deeplearning.AI

Agentic Design Patterns: Multi-Agent Collaboration



Proposed ChatDev architecture. Image adapted from "Communicative Agents for Software Development," Qian et al. (2023).

Image:THE BATCH, Deeplearning.AI

LLM Based AI Agent

REPUBLIC POLYTECHNIC

Write a white paper on how LLM has evolved since its intital days to today's capability. Explain how it lead to development of AI Agents that with LLM at its core

Here's my plan to tackle that topic. Let me know if you need to make changes.

LLM Evolution to AI Agents

Research Websites

Write a white paper explaining the evolution of Large Language Models (LLMs) from their initial generative capabilities to the advanced reasoning and planning abilities of today, and how this evolution has led to the development of AI Agents that utilize LLMs as their core.

(1) Research the definition and initial capabilities of Large Language Models (LLMs), focusing on their generative nature and early architectures like RNNs and LSTMs. Refer to research notes [1],...

More

Analyze Results

Create Report

Ready in a few mins

REPUBLIC POLYTECHNIC

Framework Ecosystem

APPLICATIONS

AGENTIC ORCHESTRATION LAYER

FOUNDATION MODEL

FINE-TUNED FOUNDATION MODEL

CLOUD INFRASTRUCTURE

ON-PREM INFRASTRUCTURE

Choosing an AI Agent Framework

Complexity of Task
Identify the tasks that you want an AI agent to fulfill and how complex these tasks are. Determine whether you need a simple implementation with only a single agent or a multiagent ecosystem.

Data privacy and security
Verify the security policies and measures of your framework of choice, including encryption for data at rest and in transit, access controls and removing any sensitive information.

Ease of Use
Consider your development team's skill level. Do you need a beginner-friendly AI framework or advanced agent frameworks low-level control and customizable code options.

Tools
Consider how well the framework works with other frameworks and tools services

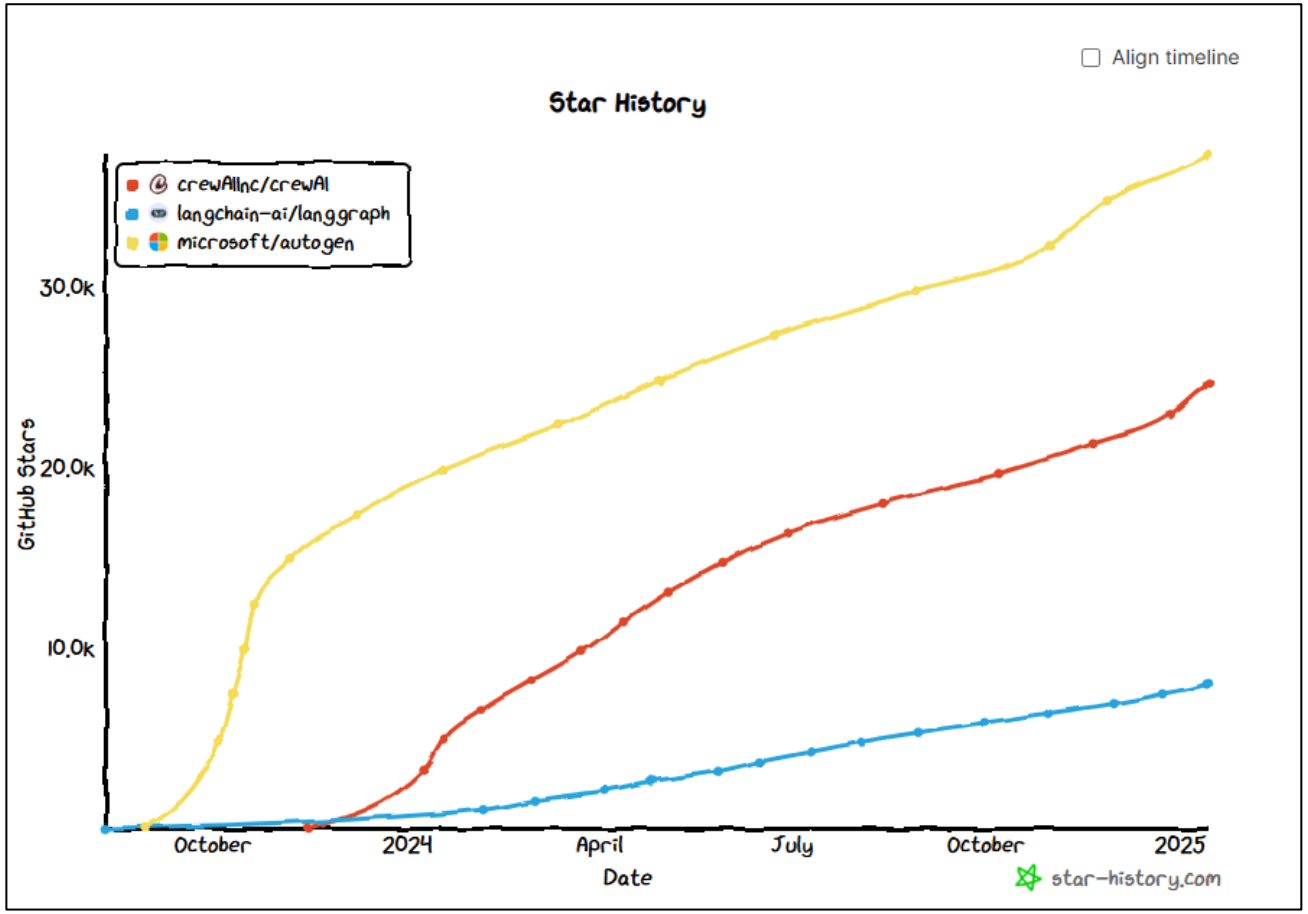
Seamless Integration
Evaluate agentic AI frameworks based on their compatibility with your existing tech stack

Performance and scalability
Think about response time or latency for real-time applications and how the framework scales

AutoGen is a versatile framework developed by Microsoft for building **conversational agents**. It treats workflows as conversations between agents, making it intuitive for users who prefer interactive ChatGPT-like interfaces.

CrewAI is a framework designed to facilitate the collaboration of **role-based AI agents**. Each agent in CrewAI is assigned specific roles and goals, allowing them to operate as a cohesive unit.

LangGraph is an open-source framework build stateful, multi-actor applications using large language models (LLMs). Inspired by the long history of representing data processing pipelines as directed acyclic graphs (DAGs). This graph-based approach allows for **fine-grained control over the flow and state of applications**, making it particularly suitable for **complex workflows and orchestration**.



Sources: Star-History.com, Perplexity AI, X

Comparing Agentic Frameworks

https://youtu.be/8lsJ7zLa2Pk?si=v8Ea4DzFbv7b7YZd

Live Demo: CrewAI Framework in Action

Outline

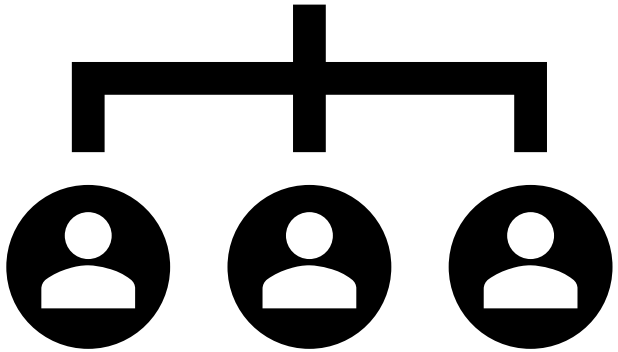
1. AI Agents Basics
2. Demo + Code Walkthrough
3. Project setup

Mental Model

Manager

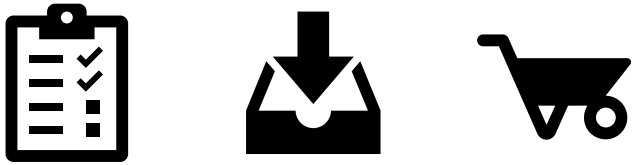
Captain

Who you need on your team



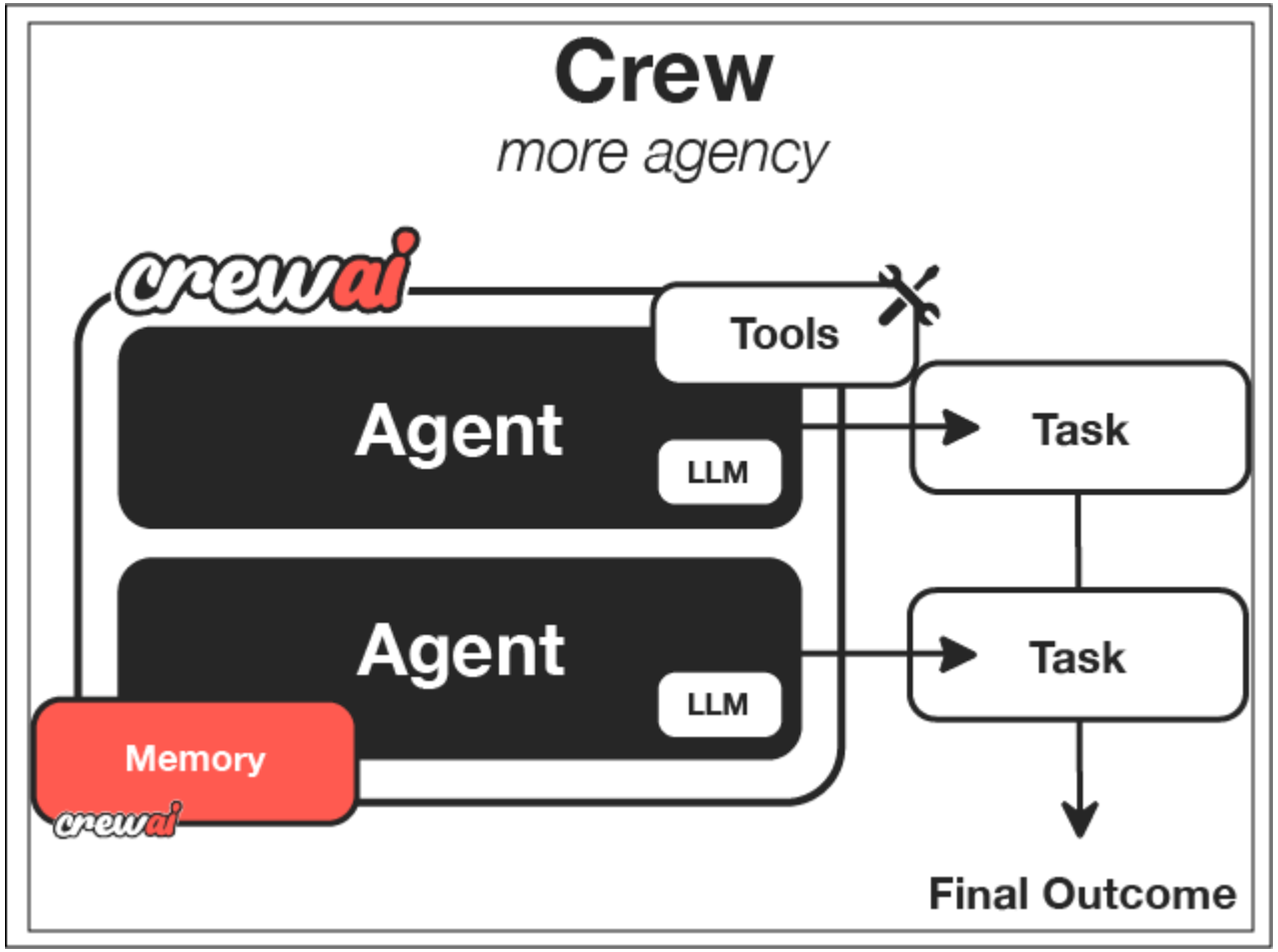
Flght Crew Specialists (Pilot, Flight Attendant, Ground Staff)

What must be accomplished



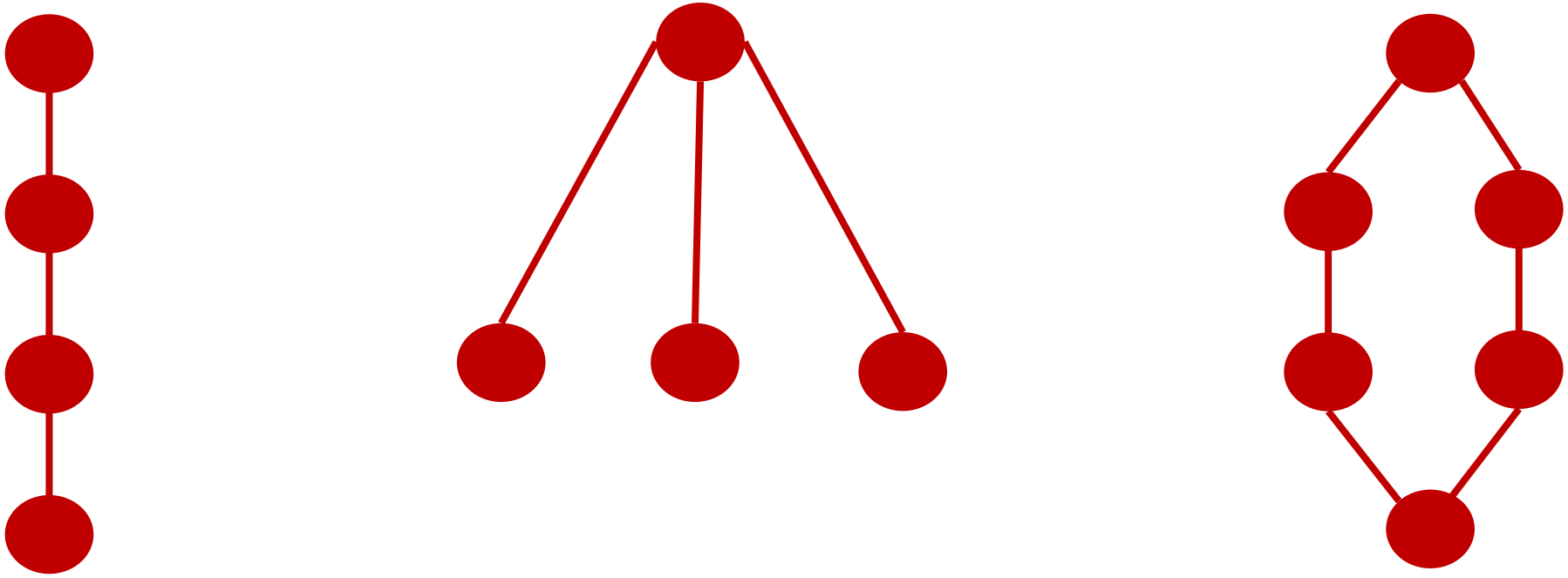
Check In, Serve, Broadcasting, Flight

CrewAI Framework Overview



Just like a company has departments (Sales, Engineering, Marketing) working together under leadership to achieve business goals, CrewAI helps you create an organization of AI agents with specialized roles collaborating to accomplish complex tasks.

Agent Collaboration Possibilities



Sequential

Hierarchical

Asynchronous

How It All Work Together

Component	Description	Key Features
Crew	The top-level organization	• Manages AI agent teams • Oversees workflows • Ensures collaboration • Delivers outcomes
AI Agents	Specialized team members	• Have specific roles (researcher, writer) • Use designated tools • Can delegate tasks • Make autonomous decisions
Process	Workflow management system	• Defines collaboration patterns • Controls task assignments • Manages interactions • Ensures efficient execution
Tasks	Individual assignments	• Have clear objectives • Use specific tools • Feed into larger process • Produce actionable results

Agent Attributes

Agent Attributes

Attribute	Parameter	Type	Description
Role	`role`	`str`	Defines the agent's function and expertise within the crew.
Goal	`goal`	`str`	The individual objective that guides the agent's decision-making.
Backstory	`backstory`	`str`	Provides context and personality to the agent, enriching interactions.
LLM *(optional)*	`llm`	`Union[str, LLM, Any]`	Language model that powers the agent. Defaults to the model specified in `OPENAI_MODEL_NAME` or "gpt-4".
Tools *(optional)*	`tools`	`List[BaseTool]`	Capabilities or functions available to the agent. Defaults to an empty list.

Details: https://docs.crewai.com/concept

REPUBLIC POLYTECHNIC

Agent – Role/Goal/Backstory Framework

Agent Definition	Guidance	Example
Role Defines what the agent does and their area of expertise.	• Be specific and specialized • Align with real-world professions: Include domain expertise	Corporate Communications Director specializing in crisis management
Goal Directs the agent's efforts and shapes their decision-making process.	• Be clear and outcome-focused Emphasize quality standard and expectation • Incorporate success criteria	Craft clear, empathetic crisis communications that address stakeholder concerns while protecting organizational reputation
Backstory Gives depth to the agent, influencing how they approach problems and interact with others	• Establish expertise and experience: Explain how the agent gained their skills • Define working style and values: Describe how the agent approaches their work • Create a cohesive persona	As a seasoned communications professional who has guided multiple organizations through high-profile crises, you understand the importance of transparency, speed, and empathy in crisis response. You have a methodical approach to crafting messages that address concerns while maintaining organizational credibility

More information: https://docs.crewai.com/guides/agents/crafting-effective-agents

Agent – LLM

CrewAI supports a multitude of LLM providers, each offering unique features, authentication methods, and model capabilities

Each can be assigned to different LLM nodes, depending on the task on hand

1. **Keep the keys in environment variables in .env**

```
OPENAI_API_KEY=<your-api-key>
HF_TOKEN=<your-api-key>
```

2. **Instantiate the LLM object in codes**

```
from crewai import LLM
llm_openai = LLM(model="openai/gpt-4“)
Llm_hf = LLM(model="huggingface/meta-llama/Meta-Llama-3.1-8B-Instruct")
```

More information: https://docs.crewai.com/guides/agents/crafting-effective-agents

Setting Up the LLM

REPUBLIC POLYTECHNIC

1. **Keep the keys in environment variables in .env**

```
OPENAI_API_KEY=<your-api-key>
HF_TOKEN=<your-api-key>
```

2. **Instantiate the LLM object in codes**

```
from crewai import LLM
llm_openai = LLM(model="openai/gpt-4“)
llm_hf = LLM(model="huggingface/meta-llama/Meta-Llama-3.1-8B-Instruct")
```

Details:https://docs.crewai.com/concepts/llms

Defining the Agent

```
llm_hf = LLM(model="huggingface/meta-llama/Meta-Llama-3.1-8B-Instruct")


agent = Agent(
    role="Senior Data Scientist",
    goal="Analyze and interpret complex datasets to provide actionable insights",
    backstory="With over 10 years of experience in data science and machine learning, "
              "you excel at finding patterns in complex datasets.",
    llm = llm_hf
) # Default: OPENAI_MODEL_NAME or "gpt-4"
```

Details:https://docs.crewai.com/concepts/llms

Task Attributes

Attribute	Parameters	Type	Description
Description	`description`	`str`	A clear, concise statement of what the task entails.
Expected Output	`expected_output`	`str`	A detailed description of what the task's completion looks like.
Name *(optional)*	`name`	`Optional[str]`	A name identifier for the task.
Agent *(optional)*	`agent`	`Optional[BaseAgent]`	The agent responsible for executing the task.
Tools *(optional)*	`tools`	`List[BaseTool]`	The tools/resources the agent is limited to use for this task.
Context *(optional)*	`context`	`Optional[List["Task"]]`	Other tasks whose outputs will be used as context for this task.
Async Execution *(optional)*	`async_execution`	`Optional[bool]`	Whether the task should be executed asynchronously. Defaults to False.
Human Input *(optional)*	`human_input`	`Optional[bool]`	Whether the task should have a human review the final answer of the agent. Defaults to False.

REPUBLIC POLYTECHNIC

Task – Anatomy of Effective Task

Single Purpose, Single Output. Avoid "God-Tasks".

Components	Guidance	Not Effective Example	A Better Task Description
Task Description - The Process	• Detailed instructions for execution • Context and background information • Scope and constraints • Process steps to follow	Research AI trends	Research AI trends for 2024 with a focus on regulatory requirements
Expected Output - The Deliverable	• Format specifications (markdown, JSON, etc.) • Structure requirements • Quality criteria • Examples of good outputs (when possible)	A report on AI trends	A comprehensive markdown report with: - Executive summary (5 bullet points) - 5-7 major trends with supporting evidence - For each trend: definition, examples, and business implications - References to authoritative sources

Defining the Task

```
research_task = Task(
    description="""
        Conduct a thorough research about AI Agents.
        Make sure you find any interesting and relevant information given
        the current year is 2025.
    """,
    expected_output="""
        A list with 10 bullet points of the most relevant information about AI Agents
    """,
    agent=researcher
)
```

Details:https://docs.crewai.com/concepts/llms

Defining the Task

```
reporting_task = Task(
    description="""
        Review the context you got and expand each topic into a full section for a report.
        Make sure the report is detailed and contains any and all relevant information.
    """,
    expected_output="""
        A fully fledge reports with the mains topics, each with a full section of information.
        Formatted as markdown without '```'
    """,
    agent=reporting_analyst,
    output_file="report.md"
)
```

Details:https://docs.crewai.com/concepts/llms

LINKING AGENT / TASK DEFINITION

```yaml
shipping_analyst:
  role: >
    Shipping Analyst
  goal: >
    Analyze shipping and logistics emails, extract key points,
    identify concerns, and propose actionable follow-ups.
  backstory: >
    With expertise in logistics operations, you excel at interpreting email context
    to determine urgency and significance. Your role is to break down complex
    shipping-related messages into smaller parts with appropriate actions items.
    Some of the common action items include tracking shipments, confirming deliveries,
    updating job assignments, and addressing delays. Your attention to detail is critical
    to performing this role successfully.
    You identify critical issues, suggest appropriate responses, and ensure that all
    necessary details are documented, reducing delays and improving workflow efficiency.
```

LINKING AGENT / TASK DEFINITION

```yaml
shipping_analysis_task:
  description: >
    Analyze the specific emails identified by the inbox monitor, using as context its email thread.
    You are given:
    The task is to analyze emails related to shipping, extracting key points, concerns, and
    action items requiring follow-up.
    Only include insights and action items from the targetMessage itself,
    unless an earlier message provides essential missing information
    (e.g., booking reference, delivery instructions, etc.)
    Do you best to identify the booking reference number that serves as the anchor to
    group related documents or communications. It may be
    found in the email subject or body or its earlier threads. Examples of the booking references is KASEJKT032248 or BCO12345678.
    Also, identify the vessel and voyage details if available in the email ors related emails.
    Typical Format of a booking reference is :
      - 8 to 12 characters
      - characters: Usually alphanumeric
      - structure: Often starts with letters (usually representing the carrier), followed by numbers.
    If the booking reference is not available, assign the value "_NotAvailable".

  expected_output: >
    Return a structured list for the specific emails.
    Each analysed email is a json object with the following fields:
    - "messageId": Unique identifier for the specific email.
    - "threadId": Unique identifier for the email thread.
    - "subject": Original subject line.
    - "bookingRef": Booking reference number (if available). Otherwise, assign value of "_NotAvailable".
    - "senderEmail": Email address of the sender.
    - "content": Full email content from the thread.
    - "actionItems": List of actions items from the email.
    - "answers": Empty list for this task.
  agent: shipping_analyst
```

Associate with the agent

Assembling The Crew

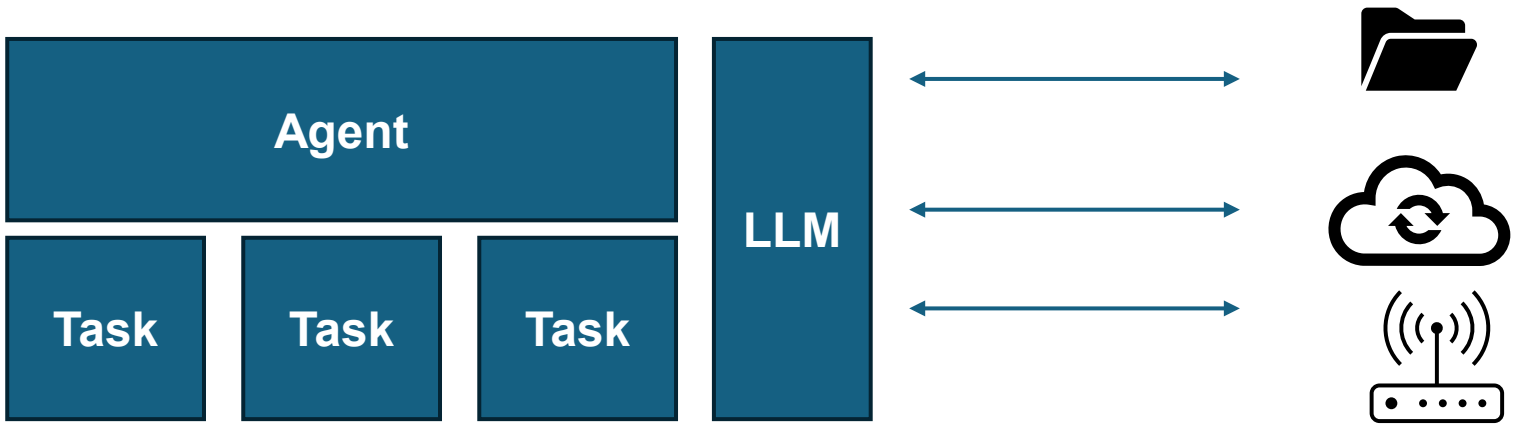
```
#
agent1 = Agent(…)
agent2 = Agent(…)

#
task1 = Task(…)
task2 = Task(…)

my_crew = Crew(
    agents=[agent1, agent2],
    tasks=[task1, task2],
    process=Process.sequential,
    llm= . . . ,)

result = my_crew.kickoff()
```

Activity 01

Spot the Difference

Image: Generated by ChatGPT

Tools



An agent powered by a language model only **can still complete tasks** that:

- Rely only on reasoning or memory
- Don't require external data sources
- Can be solved using pre-trained knowledge

With tools, an Agent tuns into an **autonomous, task-executing powerhouse** that can **think, act, calculates, retrieve etc**

- access website , stay current
- access private files (company Information)
- Call API (internal system, external services)

REPUBLIC POLYTECHNIC

Using Tools

Tools are utilities / functions designed to be called by a model: their inputs are designed to be generated by models, and their outputs are designed to be passed back to models

CrewAI Built-In Tools

https://docs.crewai.com/concepts/tools

Integrate with Langchain Tools

https://python.langchain.com/docs/integrations/tools/

Integrate with Llamaindex Tools

https://docs.llamaindex.ai/en/stable/module_guides/deploying/agents/tools/#concept

Custom Tools

REPUBLIC POLYTECHNIC

Using Built-In Tools

CrewAI comes with a set of **built-in tools** that are mostly designed to help agents perform common tasks such as web search, RAG query , Youtube search, text extraction

CrewAI Enterprise provides a comprehensive Tools Repository with pre-built integrations for common business systems and APIs

For a listing of built-in tools, refer to https://docs.crewai.com/tools/aimindtool

Using Built-In Tools - Websearch

```
from crewai_tools import SerperDevTool, WebsiteSearchTool

search_tool = SerperDevTool()
web_rag_tool = WebsiteSearchTool()

researcher_agent = Agent(
    role="  ",
    goal="  ",
    backstory="  ",
    tools=[search_tool, web_rag_tool],
      )
```

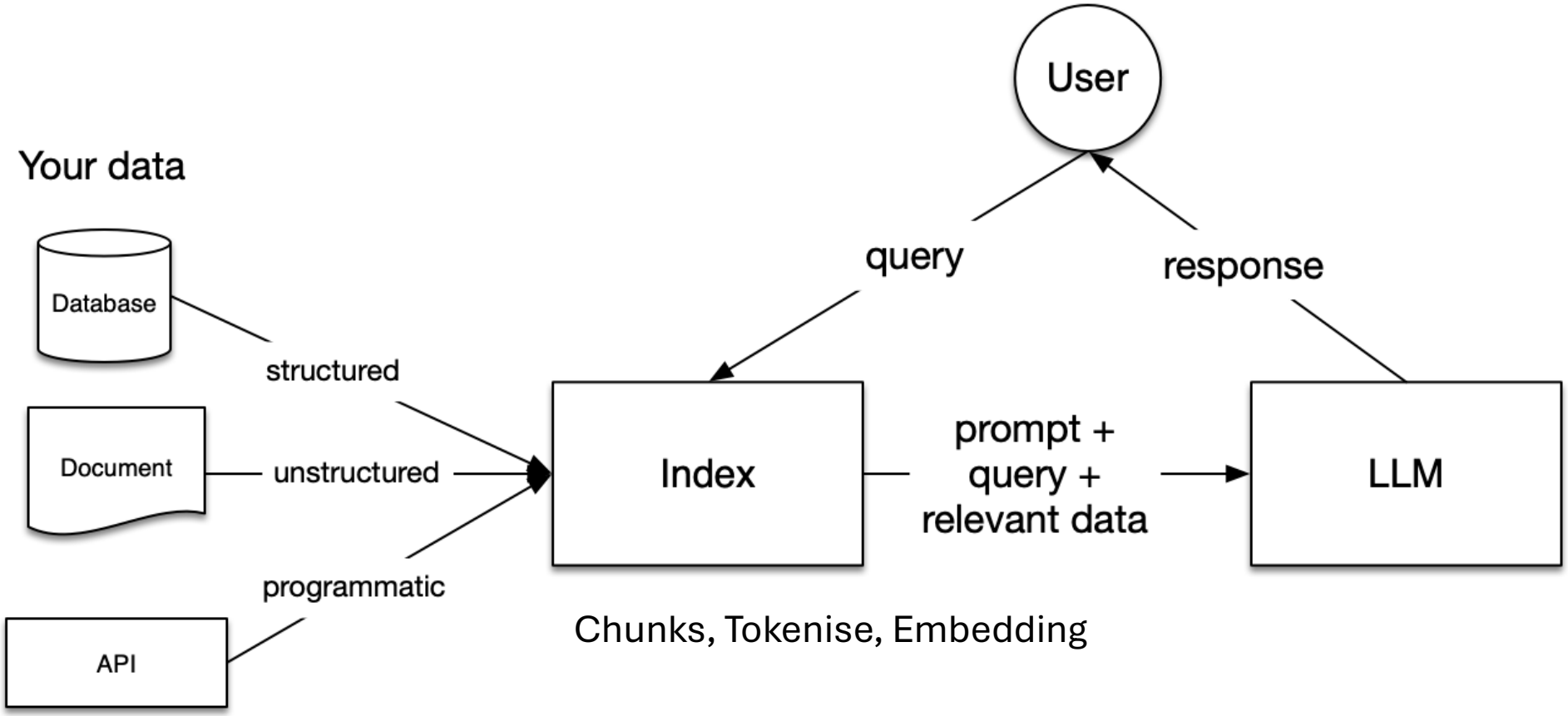
Activity 02

REPUBLIC POLYTECHNIC

Using Llamaindex Tools

- LMs come pre-trained on huge amounts of publicly available data, but they are not trained on **your** data. Your data may be private or specific to the problem you're trying to solve. It's behind APIs, in SQL databases, or trapped in PDFs and slide decks.
- Context augmentation makes your data available to the LLM to solve the problem at hand.
- LlamaIndex is the framework for Context-Augmented LLM Applications
- Tools include: Data Connectors, Data Indexes, Engine (Query, Chat), Agents, Workflows

RAG Architecture

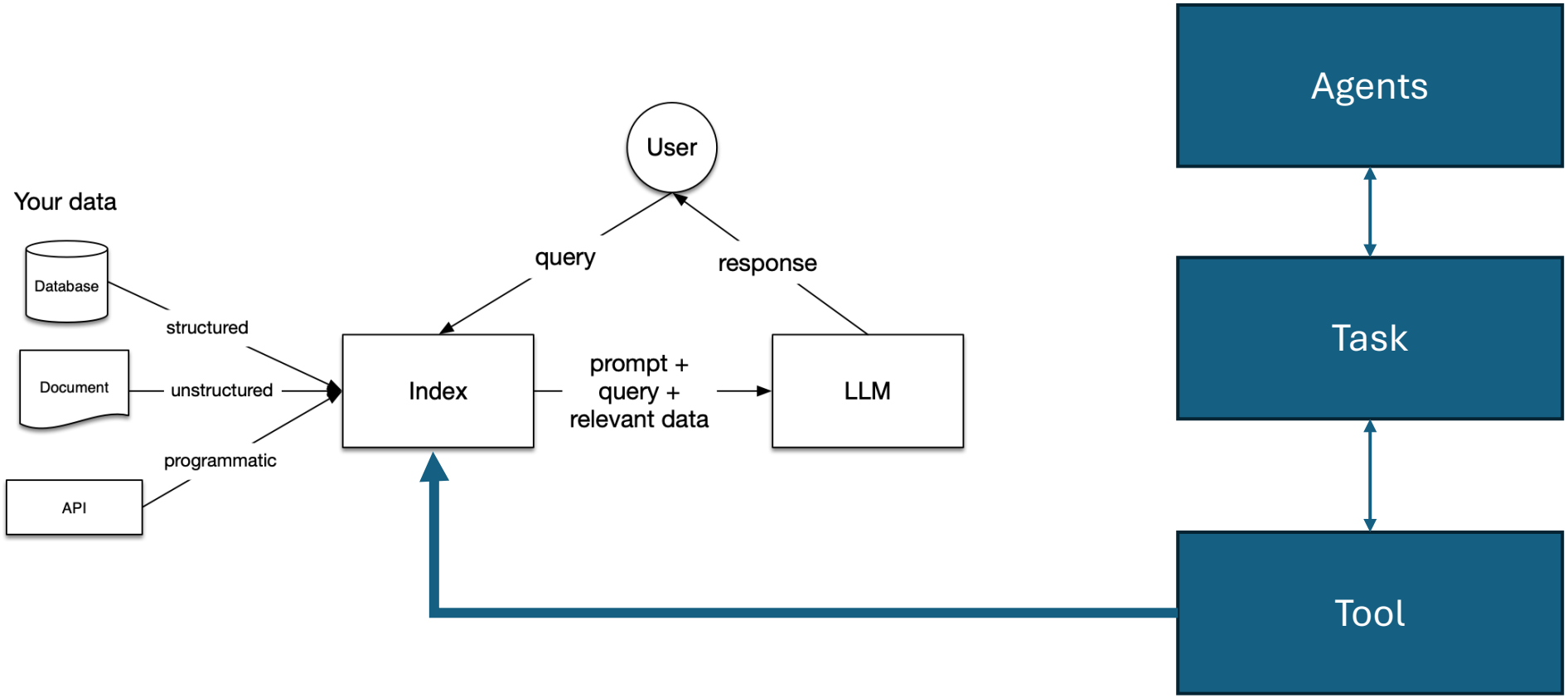


Source: https://docs.llamaindex.ai/en/stable/understanding/rag/

RAG In A Few Lines

```
from llama_index.core import VectorStoreIndex, SimpleDirectoryReader

documents = SimpleDirectoryReader("data").load_data()
index = VectorStoreIndex.from_documents(documents)
query_engine = index.as_query_engine()

response = query_engine.query(“replace with your query")
print(response)
```

CrewAI + Llamaindex



CrewAI + Llamaindex

```
documents = SimpleDirectoryReader("data").load_data()
index = VectorStoreIndex.from_documents(documents)
query_engine = index.as_query_engine()


query_on_private_data_tool = LlamaIndexTool.from_query_engine(
    query_engine,
    name="Company Data Query Tool",
    description="Use this tool to lookup information in company documents"


my_agent = Agent(…
    tools=[query_on_private_data_tool],
            )
```

Agent uses the name/description to understand how to use the tool. Be clear !

Activity 03

REPUBLIC POLYTECHNIC

Using Langchain Tools

LangChain is **a composable framework** to build with LLMs.

Langchain comes with a collection of toolkits that performs :

- Executes Online search
- Code Interpreter
- Productivity / Automation
- Web browsing
- SQL Database task
- And more

Langchain Gmail Tools

```
[ ] gmail_tool = GmailToolkit()
```

Tools Availability

- GmailCreateDraft - Tool that creates a draft email for Gmail
- GmailSendMessage - Tool that sends a message to Gmail.
- GmailSearch - Tool that searches for messages or threads in Gmail
- GmailGetMessage - Tool that gets a message by ID from Gmail.
- GmailGetThread - Tool that gets a thread by ID from Gmail.

```
[ ] tools = gmail_tool.get_tools()
    tools
```

```
[GmailCreateDraft(api_resource=<googleapiclient.discovery.Resource object at 0x7ff94a648810>),
 GmailSendMessage(api_resource=<googleapiclient.discovery.Resource object at 0x7ff94a648810>),
 GmailSearch(api_resource=<googleapiclient.discovery.Resource object at 0x7ff94a648810>),
 GmailGetMessage(api_resource=<googleapiclient.discovery.Resource object at 0x7ff94a648810>),
 GmailGetThread(api_resource=<googleapiclient.discovery.Resource object at 0x7ff94a648810>)]
```

Langchain Tools – GmailTool Kit

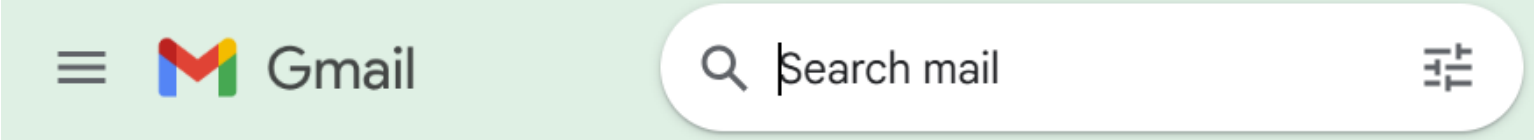
```
from langchain_community.agent_toolkits.gmail.toolkit import GmailToolkit

# Authenticate and get API resources
gmail_tool = GmailToolkit()


# Carry out a search
search = gmail_tool.get_tools()[2]
emails = search.invoke("is:unread in:inbox 'FYA' " )


# Read the email
id = emails[0]   #first mail
message = gmail_tool.get_tools()[3]
message = message.run(id)
```

Search tool index 2

Gmail Search Operators

Read the message at index 3

Gmail Search Operators



Gmail search operators are commands used to refine and filter search results within your Gmail inbox. They consist of a keyword or symbol followed by a search term or phrase, enabling precise searches.

- from:boss@hq.com
- subject:urgent
- is:unread
- after:2023/06/01

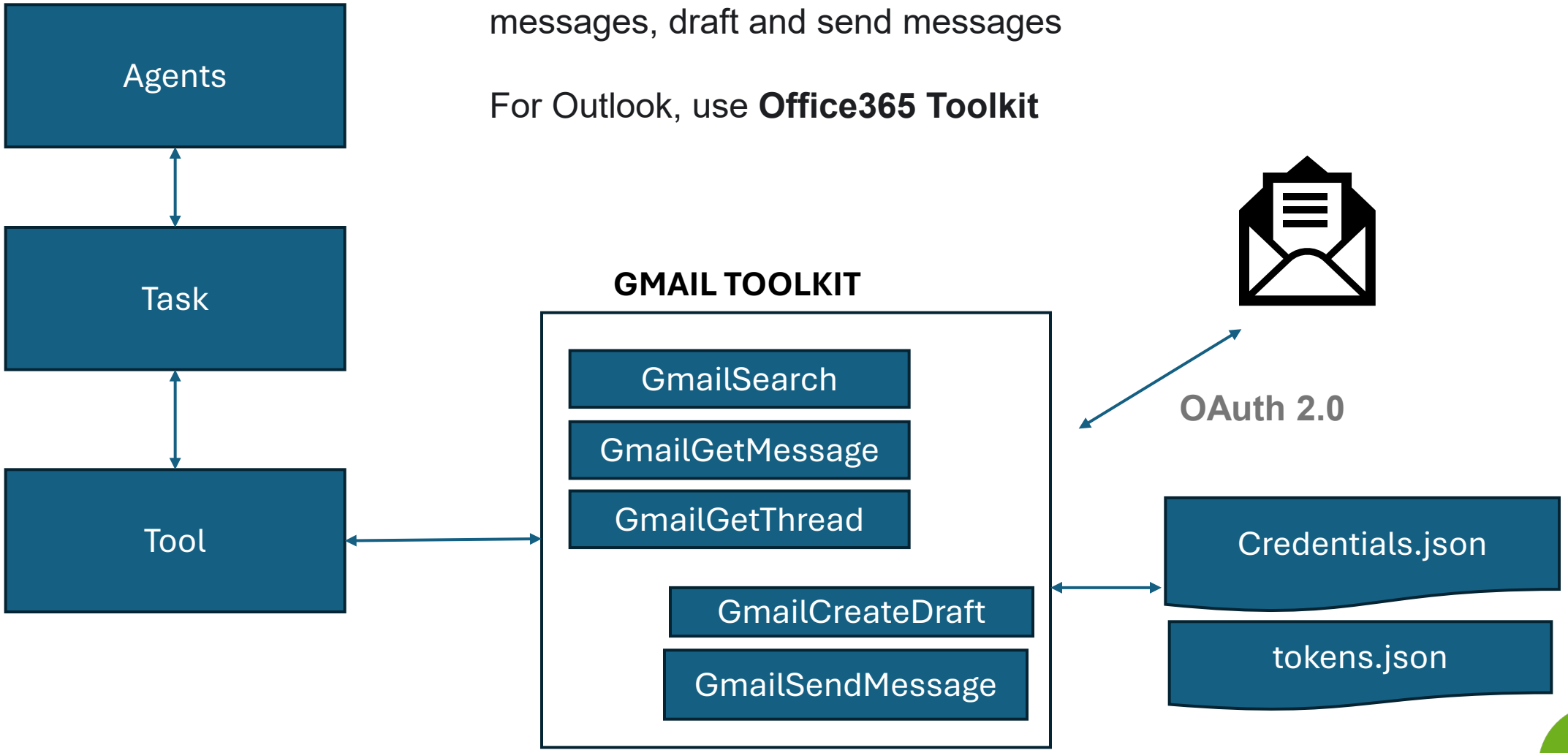
You can combine the operators using AND, NOT, OR
- from:boss@hq.com AND - is:unread

More details from https://hiverhq.com/blog/top-gmail-search-operators

CrewAI + Langchain Gmail Tool

Gmail Toolkit interacts with the GMail API to read messages, draft and send messages

For Outlook, use **Office365 Toolkit**



Using the `tool` Decorator

Alternatively, you can use the tool decorator `@tool`. This approach allows you to define the tool's attributes and functionality directly within a function, offering a concise and efficient way to create specialized tools tailored to your needs.

Code

```
from crewai.tools import tool

@tool("Tool Name")
def my_simple_tool(question: str) -> str:
    """Tool description for clarity."""
    # Tool logic here
    return "Tool output"
```

CreateDraft
SendMessage

CrewAI + Langchain Gmail Tool

```
@tool("search_gmail")
def search_gmail(query: str):
    """
    Useful for searching for gmails based on a search string
    The input should be a valid Gmail search string
    """
    try:
        gmail = GmailToolkit()  # custom GmailToolkit
    except Exception as e:
        return f"Failed to authenticate Gmail: {e}"

    query = “Some Gmail Search Operator”)
    search_tool = GmailSearch(api_resource=gmail.api_resource)
    result = search_tool.invoke(query)

    return f"\nSearch details: {result}\n"
```

Agent uses the docstring to understand how to use the tool

Good practice

Subclassing `BaseTool`

To create a personalized tool, inherit from `BaseTool` and define the necessary attributes, including the `args_schema` for input validation, and the `_run` method.

Code

```
from typing import Type
from crewai.tools import BaseTool
from pydantic import BaseModel, Field

class MyToolInput(BaseModel):
    """Input schema for MyCustomTool."""
    argument: str = Field(..., description="Description of the argument.")

class MyCustomTool(BaseTool):
    name: str = "Name of my tool"
    description: str = "What this tool does. It's vital for effective utilization."
    args_schema: Type[BaseModel] = MyToolInput

    def _run(self, argument: str) -> str:
        # Your tool's logic here
        return "Tool's result"
```

Search
GetMessage
GetThread

CrewAI + Langchain Gmail Tool

```
class CreateDraftInput(BaseModel):
    email: str = Field(..., description="Recipient's email address")
    subject: str = Field(..., description="subject description")
    body: str = Field(..., description="draft email body content")

class CreateDraftTool(BaseTool):
    name: str = "create_draft"
    description: str = "Useful to create an email draft."
    args_schema: type[BaseModel] = CreateDraftInput

    def _run(self, email: str, subject: str, body: str) -> str :
        gmail = GmailToolkit()
        draft_payload = {"to": [email],"subject": subject,"message": body}
        draft = GmailCreateDraft(api_resource=gmail.api_resource)
        result = draft.run(draft_payload)
```

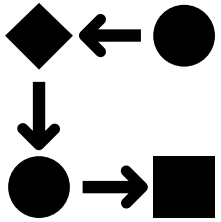
?Where should you put in a try/except block ?

CrewAI + Langchain Gmail Tool

```
mail_search_tool = search_gmail
mail_draft_tool = CreateDraftTool()


mail_agent = Agent(
    .....
    tools= [mail_search_tool, mail_draft_tool],
       )
```

Activity 04

CrewAI + Langgraph



What happened so far and how that affects what happens next.

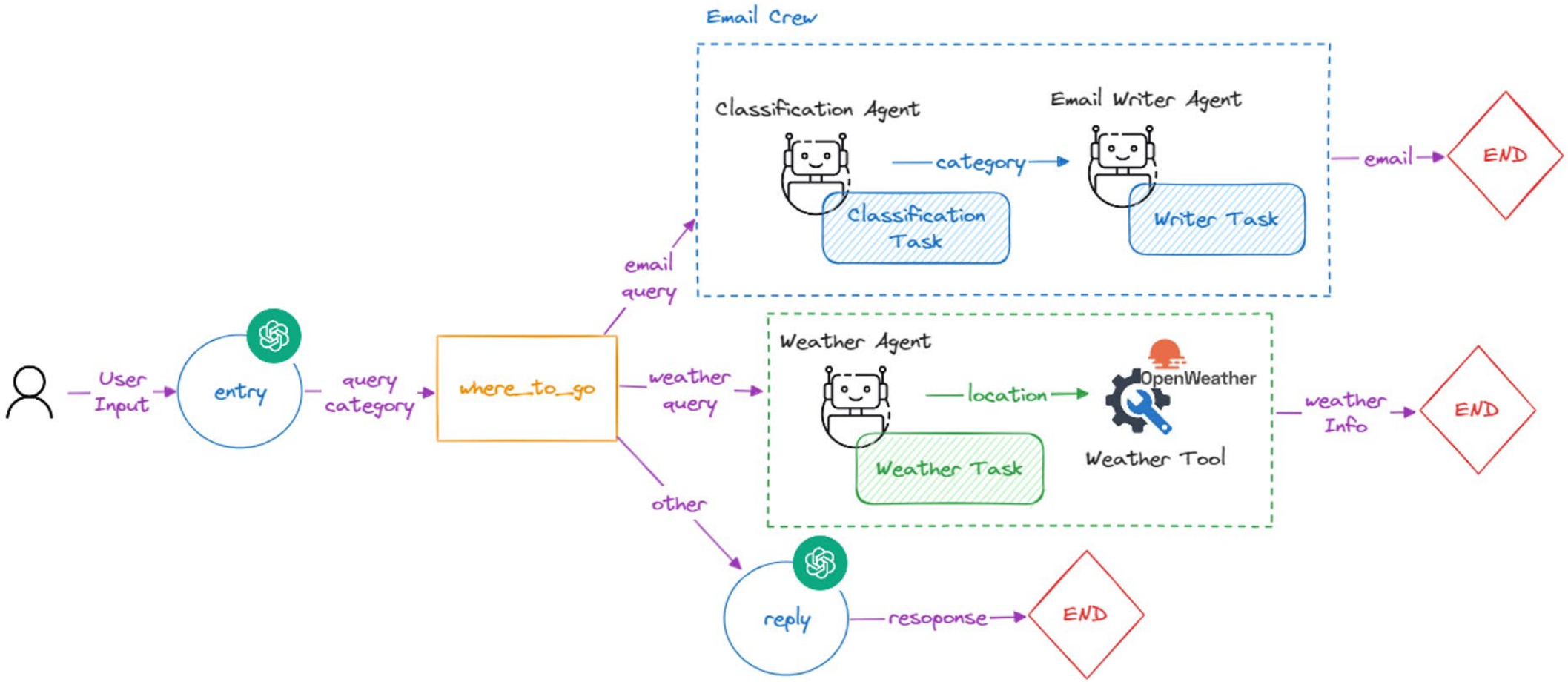
LangGraph is a framework designed to build and orchestrate complex, stateful workflows for AI agents
- Memory, control, adaptive

Who does what?

CrewAI focuses on orchestrating role-playing AI agents with predefined roles and goals to collaborate on tasks
- people, collaboration

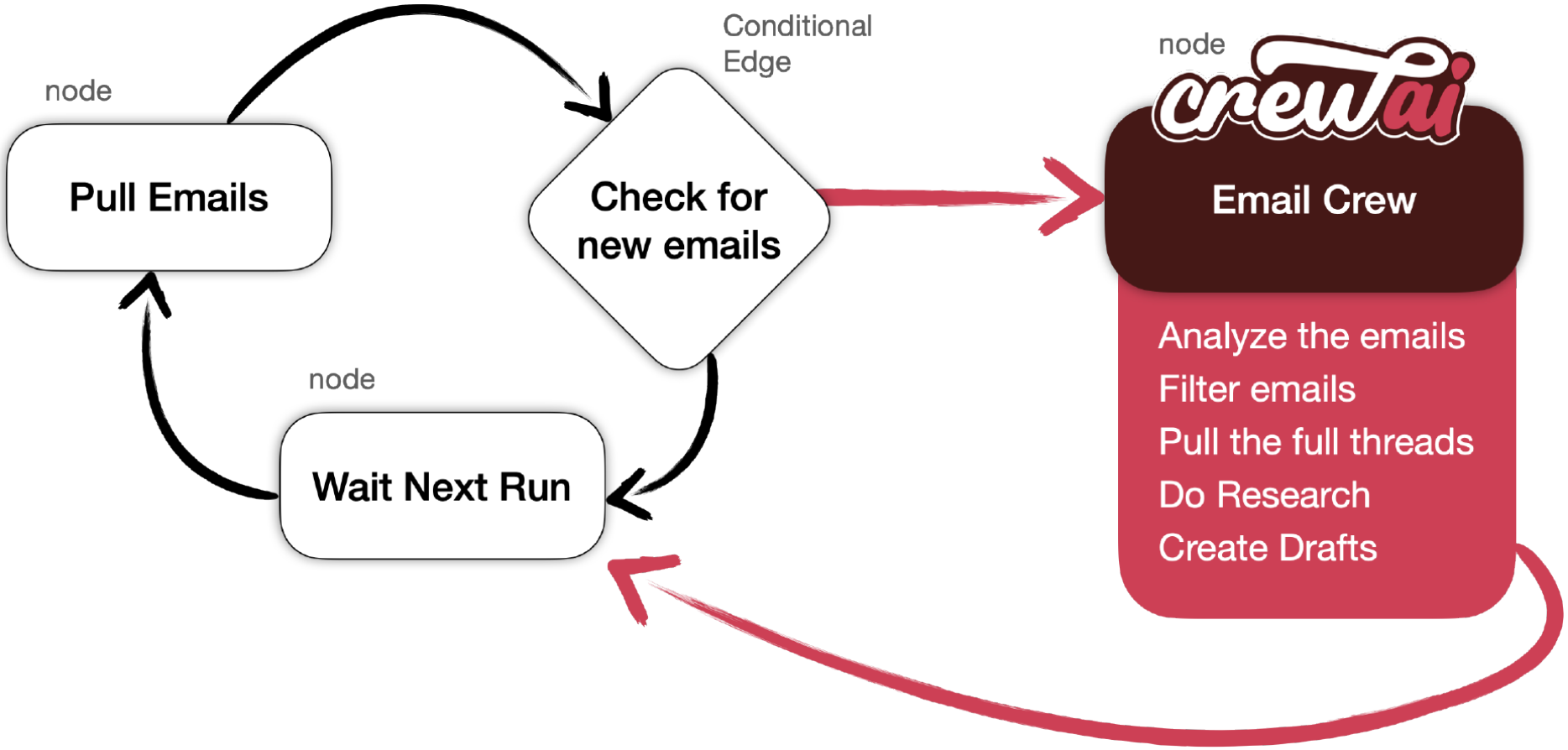
Real-world AI systems often need memory, control *and* teamwork.

Example (1)



Source: https://github.com/hualcosa/crewai-langgraph-multiagent-workflow

Example (2)



Source: https://github.com/crewAIInc/crewAI-examples/tree/main/CrewAI-LangGraph

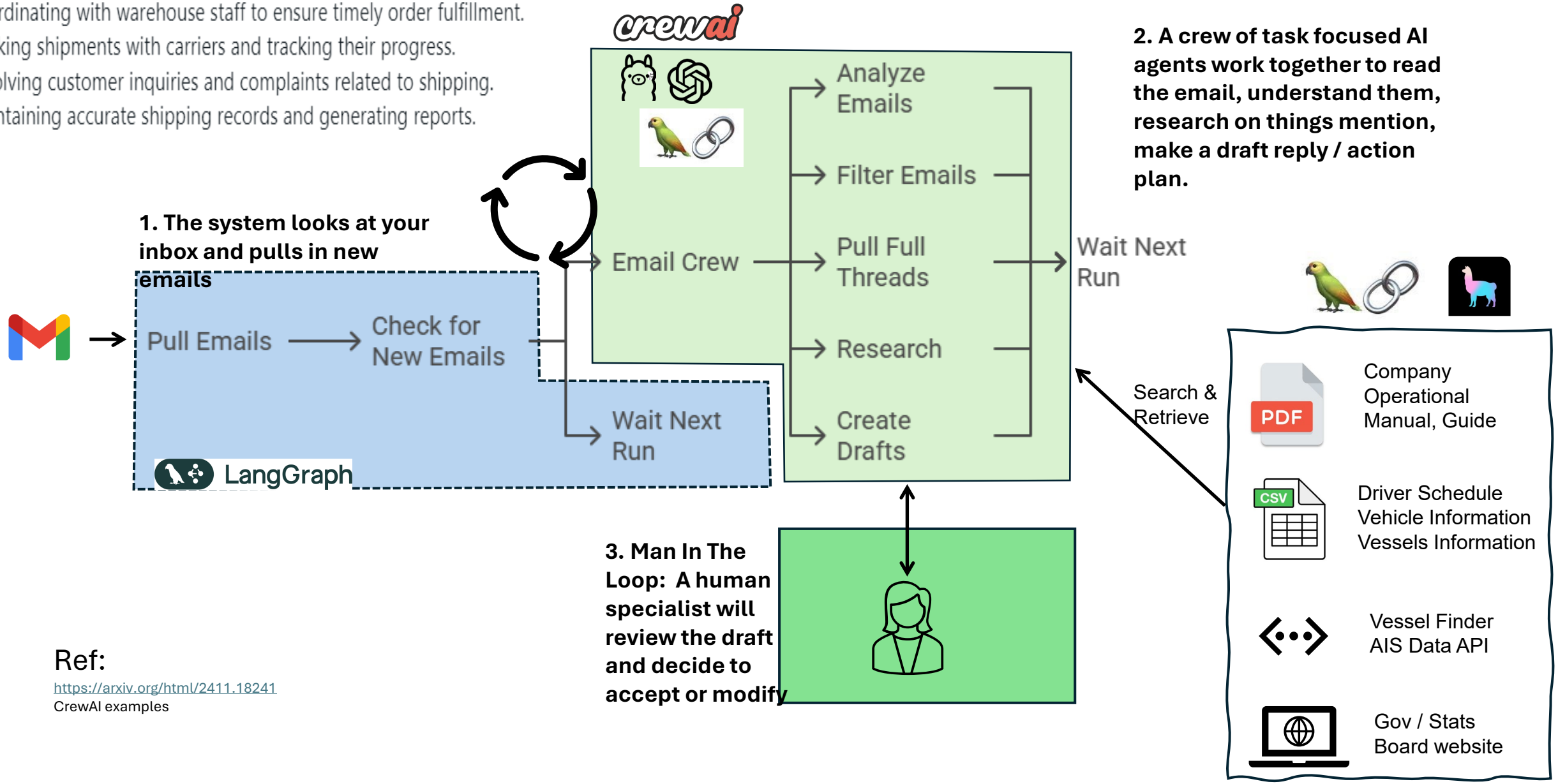
Langgraph + CrewAI (Implementation)

```
my_crew = Crew(…)

def node_1(state):
    # pretend we're doing something change state value
    return {**state}

def node_2(state):
    # pretend the crew kickoff and return state values
       my_crew.kickoff(...)
       return {**state}

workflow = StateGraph(...)

workflow.add_node("node_1", node_1)
workflow.add_node("node_2", node_2)
. . .
```

Source: https://github.com/crewAIInc/crewAI-examples/tree/main/CrewAI-LangGraph

A typical day for a Shipping Coordinator might involve:

- Reviewing incoming orders and preparing shipping labels.
- Coordinating with warehouse staff to ensure timely order fulfillment.
- Booking shipments with carriers and tracking their progress.
- Resolving customer inquiries and complaints related to shipping.
- Maintaining accurate shipping records and generating reports.

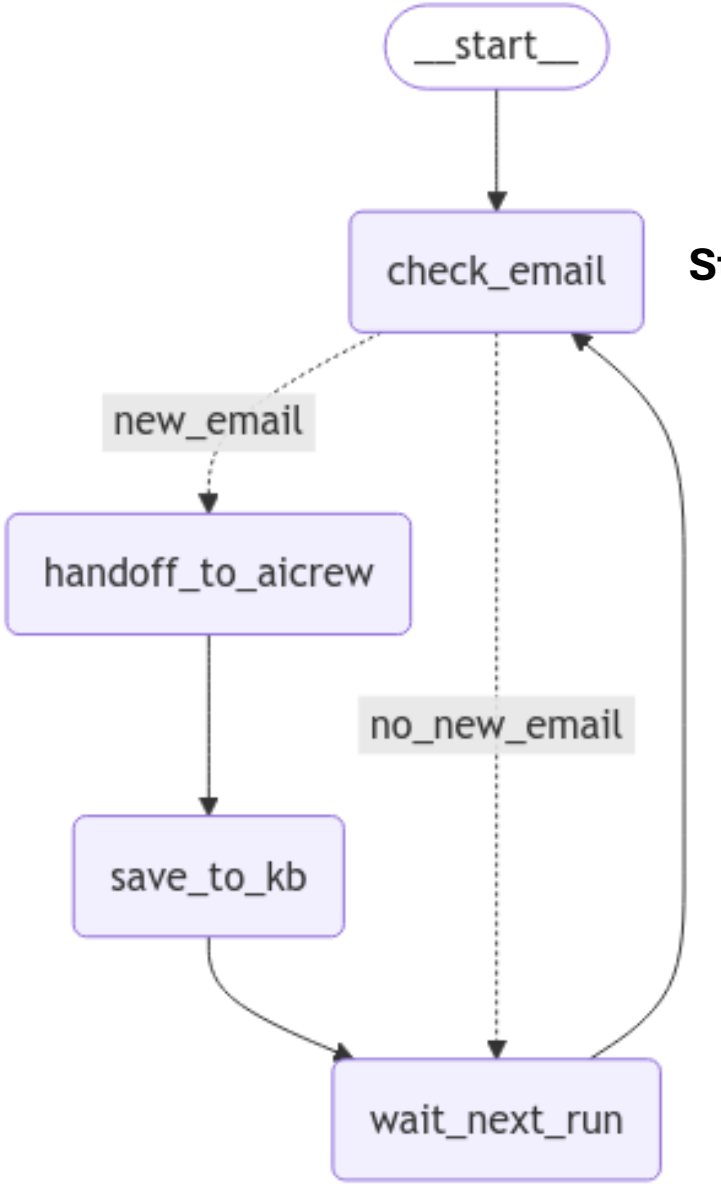


Ref:

https://arxiv.org/html/2411.18241

CrewAI examples

Graph



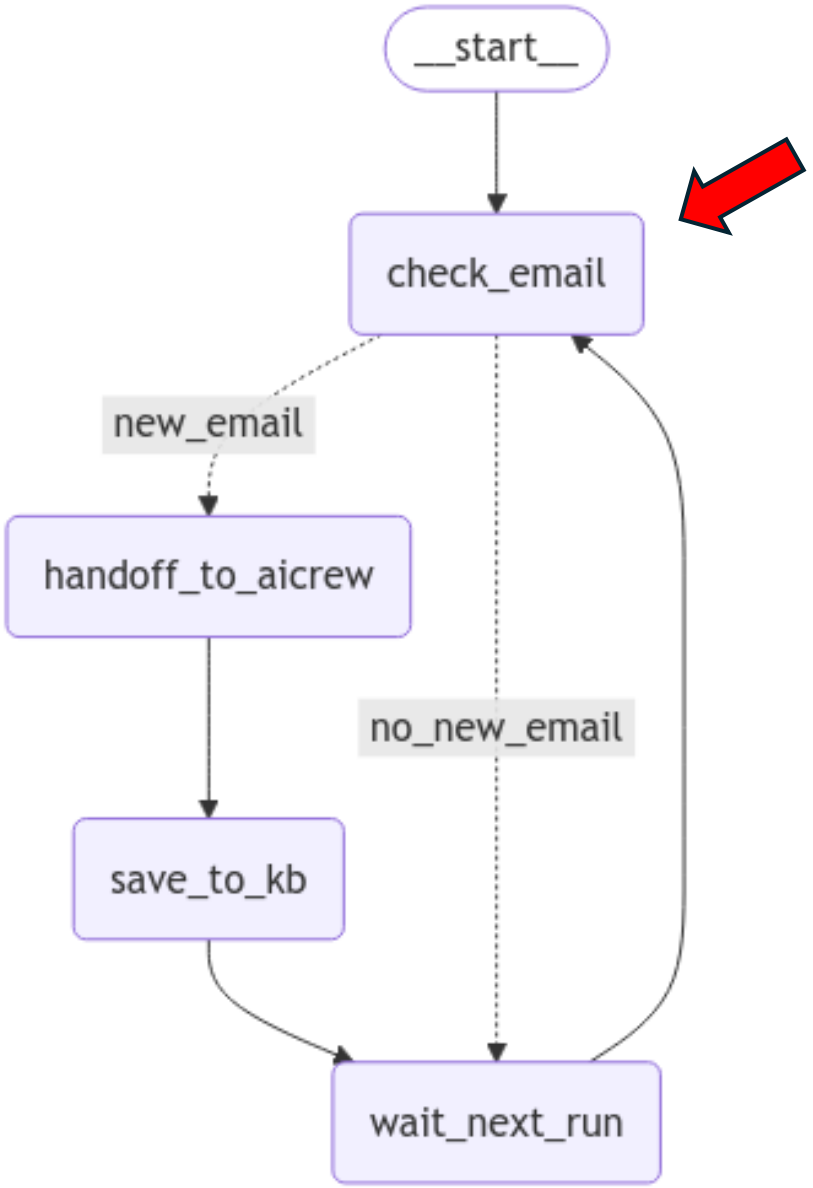
State Management + Mail Search

Email Extracts, Understanding + Replies

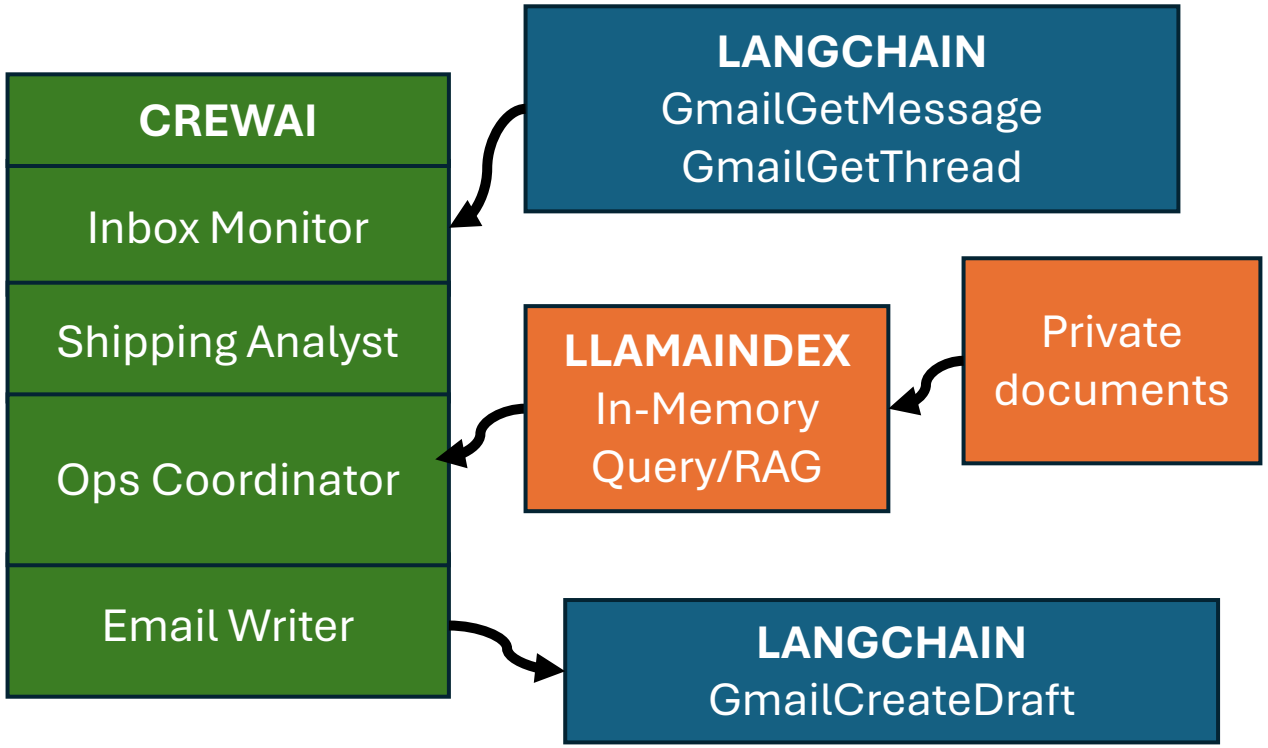
Extract, Transform, Load

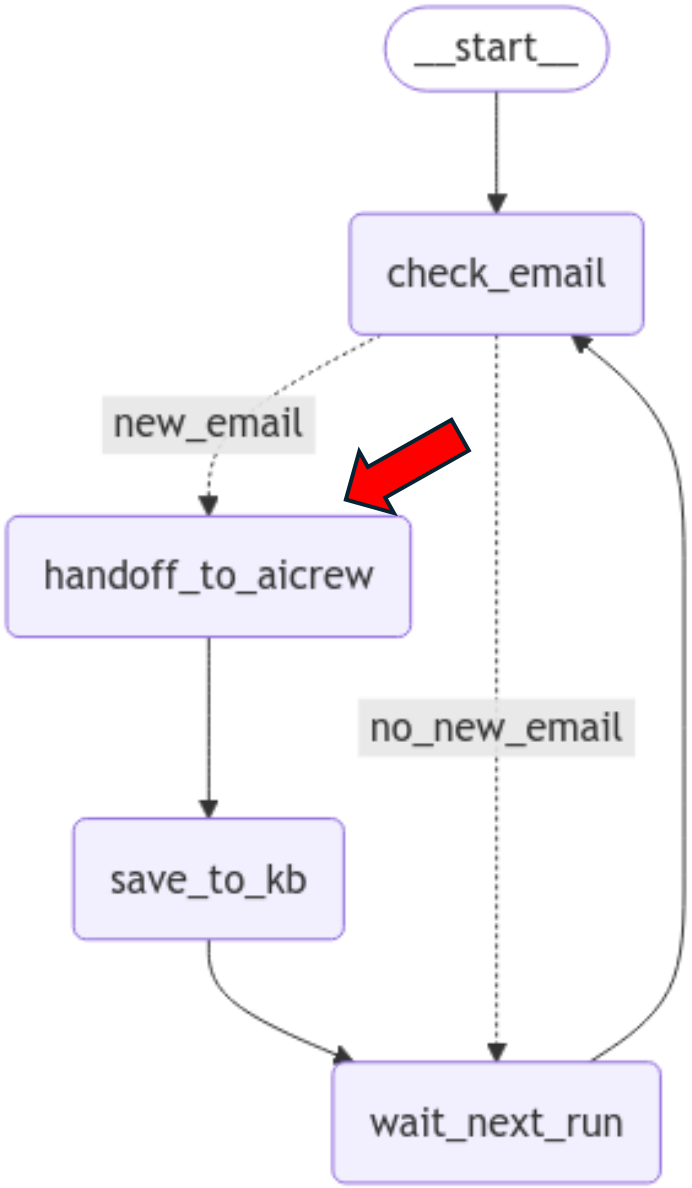
Graph

LANGCHAIN
GmailSearch
State Management

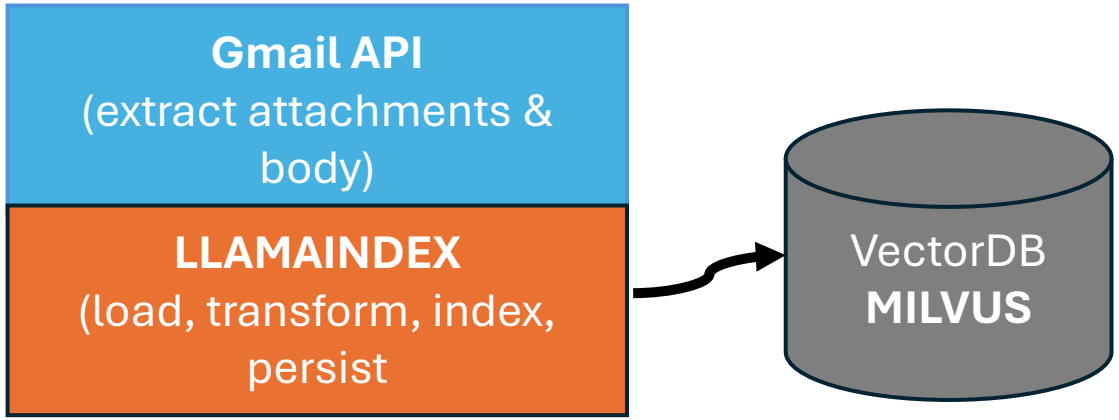


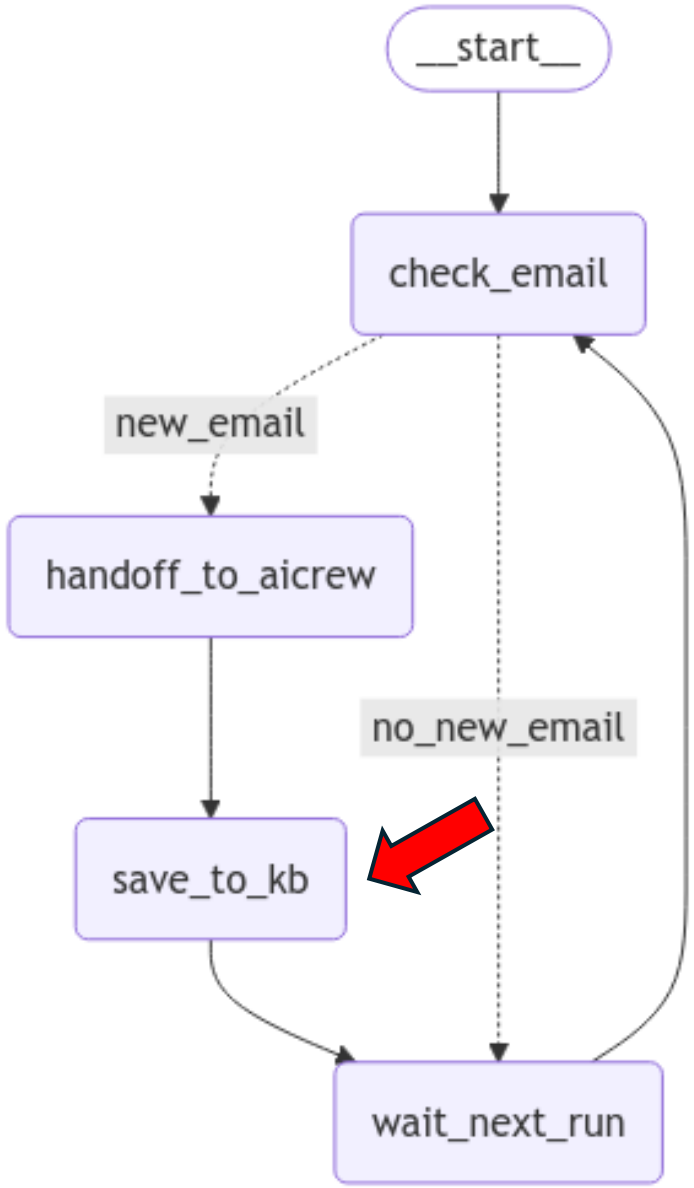
Graph


CREWAI
Inbox Monitor
Shipping Analyst
Ops Coordinator
Email Writer
LANGCHAIN
GmailGetMessage
GmailGetThread
LLAMAINDEX
In-Memory
Query/RAG
Private
documents
LANGCHAIN
GmailCreateDraft


__start__
check_email
new_email
handoff_to_aicrew
no_new_email
save_to_kb
wait_next_run

Graph


Gmail API
(extract attachments & body)
LLAMAINDEX
(load, transform, index, persist
VectorDB
MILVUS


__start__
check_email
new_email
handoff_to_aicrew
no_new_email
save_to_kb
wait_next_run

MAIL SEARCH + STATE MANAGEMENT

Gmail search operator **'is:unread in:inbox'** # space implied AND

```
State:
{   'action_required_emails': {},
    'checked_emails_ids': [],
    'emails': [],
    'graph_counter': 0,
    'max_iterations': 2}
```

	Initial	Mail Search (found 2 mails)	Mail Search (found 3 mails)	Mail Search (found 3 mails)
checked_email_ids	-	Id#1, id#2	Id#1, id#2, id#3	Id#1, id#2, id#3
emails	-	Details of emails id#1 Details of emails id#2	Details of emails #id3	[ ]
graph_counter	-	1	2	3

MAIL SEARCH + STATE MANAGEMENT

REPUBLIC POLYTECHNIC

Is passed as input to Crew Kickoff (next node)

```
State:
{   'action_required_emails': {'dummy': 'dummy'},
    'checked_emails_ids': ['195d5ef833acd1b1'],
    'emails': [   {   'cc': None,
                      'from': 'Customer <msgcustomer@gmail.com>',
                      'id': '195d5ef833acd1b1',
                      'sender': 'Customer <msgcustomer@gmail.com>',
                      'snippet': 'Dear All, Here are the new pic list. thank '
                                 'you On Thu, Mar 27, 2025 at 12:42 PM '
                                 'ShippingCoordinator '
                                 '&lt;msgubts@gmail.com&gt; wrote: Dear '
                                 'Juliet, Thank you for reaching out regarding '
                                 'the latest vessel',
                      'subject': 'Re: PT Interindo PO#2408014-10PL | ETA SIN '
                                 '10/09/24 | ETA JAKARTA\r\n'
                                 ' 13/09/24 | AN HAI',
                      'threadId': '195d51aac64f550f'}],
    'graph_counter': 1,
    'max_iterations': 2}
```

graph_counter increase by 1 per each loop Graph exits when graph_counter = max_iteration)

EMAIL CREW

Agent	Task
Inbox Monitor Track the inbox, filter out irrelevant messages, and retain important emails related to shipping and logistics.	**Input: State['emails']** **inbox_monitor_task** Monitor the inbox and filter out irrelevant messages while retaining emails related to **shipping and logistics jobs
Shipping Analyst Analyze shipping and logistics emails, extract key points, identify concerns, and propose actionable follow-ups.	**shipping_analysis_task** Analyze the specific emails identified by the inbox monitor, using as context its email thread. The task is to analyze emails related to shipping, extracting key points, concerns, and action items requiring follow-up.
Operations Coordinator Ensure all identified action items in the emails are supported with accurate, relevant information	**action_support_task** Use the email content as context, and leverage knowledge bases or external websites to find relevant, up-to-date details supporting the follow-ups.
Email Writer Craft clear, professional, and effective email responses based on dentified action items and necessary information.	**email_drafting_task** Based on the identified action-required emails, draft responses tailored to address the specific needs and context of each email **Output: Draft Email in Gmail & JSONL agent_app/outputs/crew**

Details: agent_app/src/email_crew/config

EMAIL CREW (OUTPUTS)

```
agent_app/outputs/crew
├── 20250415-120252-exported
│   ├── action_support_task.json
│   ├── email_drafting_task.jsonl
│   ├── inbox_monitor_task.txt
│   └── shipping_analysis_task.json
```

email_drafting_task.jsonl is used in the subsequent node


```
{
    "messageId":"195d5ef833acd1b1",
    "threadId":"195d51aac64f550f",
    "bookingRef":"KASEJKT032248",
    "subject":"Re: PT Interindo PO#2408014-1",
    "status":"DraftCompleted"
}
```

UPLOAD KNOWLEDGE BASE

Step 1: Extracting

1. Use **email_drafting_task.jsonl** to decide what needs to be vectorized

2. Based on messageId, perform low-level Gmail API calls to extract the emails in html and its attachments.

```
agent_app/outputs/rag
├── KASEJKT032248
│   ├── 250415120320
│   │   ├── Explomo_Builders_Pte._Ltd.-TQ2_(Trucking.pdf
│   │   ├── meta.jsonl
│   │   └── Re__PT_Interindo_PO#2408014-10PL___ETA_S.html
```

UPLOAD KNOWLEDGE BASE

REPUBLIC POLYTECHNIC

Step 2: Transforming

Use Llamaindex SimpleDirectoryReader() .load() to load and vectorise the directory contents

```
reader = SimpleDirectoryReader(input_dir=leave,
                               required_exts=required_exts,
                               recursive=False,
                           file_metadata=file_metadata)
documents = reader.load_data()
all_documents.extend(documents)
```

UPLOAD KNOWLEDGE BASE

REPUBLIC POLYTECHNIC

Step 3: Write to Milvus Vector Store

```
# Create Milvus vector store
vector_store = MilvusVectorStore(
    uri=self.milvus_uri,
    dim=self.dim,
    overwrite=self.overwrite,
    collection_name=self.collection
)


# Create storage context
storage_context = StorageContext.from_defaults(vector_store=vector_store)

# Create vector index and insert documents
index = VectorStoreIndex.from_documents(all_documents, storage_context=storage_context)
```

MILVUS DATABASE

Milvus is an open-source vector database that efficiently handles complex unstructured data like images, audio, and text. ’
Milvus is an open-source project under LF AI & Data Foundation distributed under the Apache 2.0 license.

Deployment mode – Milvus Lite, Milvus Standalone, Milvus Distributed. Runs efficiently across a wide range of environments, from a laptop to large-scale distributed systems.

Milvus - https://milvus.io/docs/overview.md

MILVUS DATABASE

REPUBLIC POLYTECHNIC

Searching for Best Practices in Retrieval-Augmented Generation

Xiaohua Wang, Zhenghua Wang, Xuan Gao, Feiran Zhang,
Yixin Wu, Zhibo Xu, Tianyuan Shi, Zhengyuan Wang, Shizheng Li,
Qi Qian, Ruicheng Yin, Changze Lv, Xiaoqing Zheng*, Xuanjing Huang
School of Computer Science, Fudan University, Shanghai, China
Shanghai Key Laboratory of Intelligent Information Processing
{xiaohuawang22,zhenghuawang23}@m.fudan.edu.cn
{zhengxq,xjhuang}@fudan.edu.cn

Database	Multiple Index Type	Billion-Scale	Hybrid Search	Cloud-Native
Weaviate	✗	✗	✓	✓
Faiss	✓	✗	✗	✗
Chroma	✗	✗	✓	✓
Qdrant	✗	✓	✓	✓
Milvus	✓	✓	✓	✓

Table 5: Comparison of Various Vector Databases

"Our evaluation indicates that Milvus stands out as the **most comprehensive solution** among the databases evaluated, meeting all the essential criteria and outperforming other open-source options."

KEY CONCEPTS

REPUBLIC POLYTECHNIC

DATABASE

A database serves as a logical unit for organizing and managing data.

A database can have many collections.

COLLECTION

A **collection** in Milvus is like a **table** in a traditional SQL database.

Two-dimensional table with fixed columns (fields) and variant rows (entity)

ENTITY

An **entity** is a single **row of data** in the collection

Fields

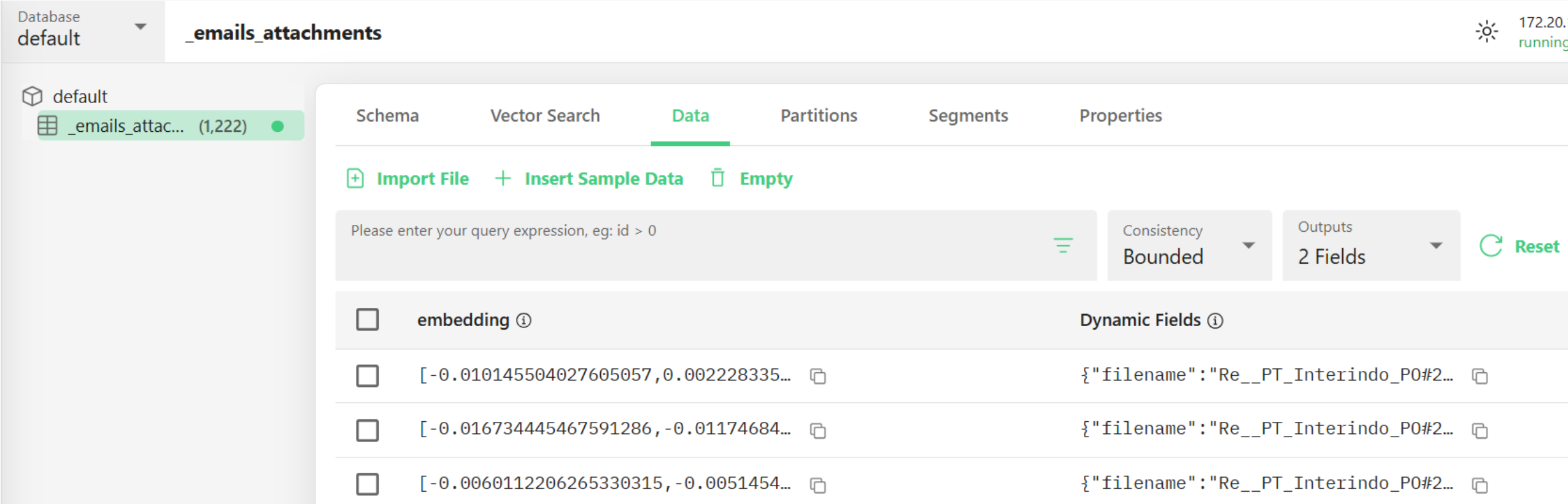
Entities

id	title	title_vector	link	reading_time	publication	claps	responses
0	The Reported Mortality Rate of Coronavirus Is Not Important	[0.041732933, 0.013779674, -0.027564144, -0.013061441, 0.009748648, 0.0(0.019370917, -0.008112641, -0.014931766, 0.065622255, 0.0149185015, 0.0	https://medium.com/swlh/the-reported-mortality-rate-of-coronavirus-is-not-important-369989c8d912	13	The Startup	1100	18
1	Dashboards in Python: 3 Advanced Examples for Dash Beginners and Everyone Else	[0.0039737443, 0.003020432, -0.0006188639, 0.03913546, -0.00089768134, -0.0076336027, -0.007048531, -0.03015078, 0.03309948, 0.028124114, 0.01	https://medium.com/swlh/dashboards-in-python-3-advanced-examples-for-dash-beginners-and-everyone-else-b1daf4e2ec0a	14	The Startup	726	3
2	How Can We Best Switch in Python?	[0.031961977, 0.00047043373, -0.018263113, 0.027324716, -0.0054595284, -0.0043757795, 0.035335075, -0.031514447, 0.008930741, 0.060583394, 0.0	https://medium.com/swlh/how-can-we-best-switch-in-python-458fb33f7835	6	The Startup	500	7
3	Maternity leave shouldn't set women back	[0.032572296, -0.011148319, -0.01688577, -0.0026665623, -0.011911687, -0 0.0049457005, 0.025084743, -0.0047377576, -0.009690498, 0.01791431, 0.0	https://medium.com/swlh/maternity-leave-shouldnt-set-women-back-5019dd3129d8	9	The Startup	460	1
4	Python NLP Tutorial: Information Extraction and Knowledge Graphs	[-0.011735886, -0.016938083, -0.027233299, 0.010389945, -0.037136745, 0. 0.039703365, -0.057903044, 0.0099971695, 0.064220704, 0.04100326, -0.00	https://medium.com/swlh/python-nlp-tutorial-information-extraction-and-knowledge-graphs-43a2a4c4556c	7	The Startup	163	0
5	Guide to Nest JS-RabbitMQ Microservices	[0.006832482, 0.012578676, -0.0032038041, 0.008881042, 0.02000302, -0.0(-0.0048119356, -0.04051565, 0.030480245, 0.043798033, 0.019709101, -0.0	https://medium.com/swlh/guide-to-nest-js-rabbitmq-microservices-e1e8655d2853	7	The Startup	184	0

Figure 1: A Medium article collection in Zilliz Cloud database.

Source: Milvus website

Attu WebUI for Milvus



Collection name: _emails_attachments
Embedding: Vectorised data of chunks of the data
Dynamic Fields: Meta Data derived by Llamaindex

Hands-On Query On Milvus

Notebook: agent_app/notebooks/query_llamaindex_milvus.ipynb

Run the notebook on your local environment
Ensure the Milvus is running
Check that there are some entities in Milvus
Adjust the query based on the stored information

Discussions