CrewAI Week-3.

Crew has several offering→

→ Crew Enterprise - A multi-agent platform for deploying, running and monitoring AgenticAI

→ CrewAI UI studio → A no-code/lo-code product for creating multi-agent solution.

→ CrewAI open-source framework - "Orchestrating high performance AI agents with ease and scale.

→ Two different approaches →

① CrewAI Crews → Autonomous solution with AI teams of agents with different roles.

② CrewAI flows → structured automations by dividing complex tasks into precise workflow. are more fixed workflows.

Usecase →

① "Choose Crew when: You need autonomous problem-solving, "creative collaboration or exploratory tasks".

② "Choose ~~[illegible]~~ flows when: You require deterministic outcomes, auditability, or precise control over execution'

[CrewAI → CrewAI is a lean, lighting-fast Python framework built entirely from scratch - completely independent of Langchain or other agent frameworks]

~~[illegible]~~

Five step to create crewAI project. (14)

→ Agent → an autonomous unit, with an (16) LLM, a role, a goal, a backstory, memory tools

→ Task → a specific assignment to be carried Out, with a description, expected output, agent.

→ Crew → a team of Agents and Tasks; either:
sequential: run tasks in order they are defined.
hierarchical: use a manager LLM to assign.

Difference by OpenAI SDK →
Lightweight, but somewhat more opinionated than OpenAI Agents SDK - more terminologies marginally more prescriptive.
... and with an ability to get much more prescriptive

Agents & Task can be created by code, setting the backstory, description, expected, output, etc.
Or you can define each in the YAML file that's provided when you create the code:

```
researcher:        → can add as different file for better understandin
  role:>
    senior Financial Researcher
  goal:>
    Research companies, news and potential
  backstory:>
    Your . - - -  . . . . info.
  llm: gemini-2.0-flash
```

YAML file.

[agent = Agent(config=self.agents_config
['researcher'])]

↑ agent in YAML file

In addition to YAML we also crewAI.py file.

→ CrewAI uses the super-simple LiteLLM under the hood to interface with almost any LLM, set key in .env file

{ llm = LLM(model = "gemini/gemini-2-0-flash") }

Even simpler than OpenAI SDK.

CrewAI projects & UV projects :

① CrewAI is already installed
→ uv tool install crewai.

② Create a new project with
→ crewai create crew my-crew]

{ Create a directory structure with subdirectories }

③ Run with
→ ~~crew~~crewai run.

crewai uses (UV.)

Five step to create crewAI project.

① Create the project with:
   Crew create crew my-project.

② Fill in the config yaml files project to define the Agents and tasks

③ Complete the crew.py module to create the Agents, Tasks, and crew, refrencing the config.

④ update main.py to set any config and run.

Note → Serper is an free open-source platform that allows to run lighting fast google search at an unbeatable price.

→ Go to serper > signup > Api key > copy the clipkey > put it in .env file.

→ Project 4 → stock picker.

Using all the five steps +
along with new features -

① structured output
② custom tool → Imp.
③ hierarchical process

→ step 2 → change the name of the tool.

→ Project 5 → Developer Agents +

→ feature of crew that uses memory.

① ~~Short term memory → temporarily stores retired. RAG is called in embeddings~~

Types of Memory →

(1) Short term memory → Temporary stores recent infor and outcomes using RAG. enabling ~~info~~ ~~but~~ agent to access relevant information during the current execution

(2) Long-Term memory → Preserve valuable insights and learning ; building knowledge over time.

(3) Entity memory → Info about people, places and concepts encountered during tasks, facilitation deeper understanding and relationship mapping. Use RAG for storing entity information

(3) contextual memory) → main the context by combining by ~~[illegible]~~ the above

(4) User memory → stores user-specific info ~~When and~~ proffession and user expr.

memory is added with crew.py file

code-execution = True put in docker current

Project 5 → Engineering Team project

Agents →
Engineer Leads, Backend Engineer, Frontend Engineer, Test Engineer.