Langchain Process

Friday, 28 March 2025 4:30 PM

- Create a prompt template
- Create a llm wrapper
- Create a chain and link these using |
- Get the result by invoking the chain like this :
    - res = chain.invoke(input = {"information":information})
- Use output parser after calling llm so that it handles parsing the output and give us properly

Creating agent:

-
```
tools_for_agent = [
    Tool(
        name="Crawl Google 4 linkedIn profile page.",
        func="?",
        description="Used when you need to get the linkedin URL."
    )
]
```

- Now Harrison Chase 17 is the username of Harrison Chase in the prompt tab.
  And Harrison Chase is the co-founder and creator of Link Chain and Slash react.
  Here is a prompt that Harrison Chase wrote, which is a super popular prompt used for react prompting.
  And it's actually going to be the reasoning engine of our agent.

  It will include our tool names and our tool descriptions and what we want our agent to do.
  And luckily for us, Linkchain is going to be plugging in those values for us after we initialize the

```
react_prompt = hub.pull("hwchase17/react")
```

- And now we want to use the function Create React agent, which is going to accept our react prompt that we just saw. It's going to accept our tools and it's going to accept our LLM.

```
agent = create_react_agent(llm=llm, tools=tools_for_agent, prompt=react_prompt)
```

- But now we want to provide it also the runtime.
  So how to run in loops.
  And this is going to be our final agent.
  So we want to create something which is called an agent executor.
  It's going to receive that agent.
  It's going to receive a list of tools because those are actually the tools that will be invoked.
  And we want to.
  Supply verbose equals true.
  So we'll see extensive logging and we'll understand a bit more how this agent is working.

```
agent_executor = AgentExecutor(agent=agent, tools=tools_for_agent, verbose=True)
```

- last thing we want to do is to invoke the agent.
  And lastly what we want to do is simply to parse out the result.
  So we'll go and take the result in the output key and simply return it to the user.

```
result = agent_executor.invoke(
    input={"input":prompt_template.format_prompt(name=name)}
)
```

- How react prompt works: