"""

pip3 install -U swarms

pip3 install -U chromadb

task -> Understanding Agent [understands the problem better] -> Summarize of the conversation ->

research agent that has access to internt perplexity -> final rag agent

Todo

- Use better llm -- gpt4, claude, gemini

- Make better system prompt

- Populate the vector database with q/a of past history

"""

```
from swarms import Agent, llama3Hosted, AgentRearrange
from pydantic import BaseModel
from swarms_memory import ChromaDB

# Initialize the language model agent (e.g., GPT-3)
llm = llama3Hosted(max_tokens=3000)


# Initialize Memory
memory = ChromaDB(
```

```
    output_dir="swarm_mechanic", n_results=2, verbose=True
)



# Output
class EvaluatorOuputSchema(BaseModel):
    evaluation: str = None
    question_for_user: str = None



# Initialize agents for individual tasks
agent1 = Agent(
    agent_name="Summary ++ Hightlighter Agent",
     system_prompt="Generate a simple, direct, and reliable summary of the input task alongside the highlights",
    llm=llm,
    max_loops=1,
)


# Point out that if their are details that can be added
# What do you mean? What lights do you have turned on.
agent2 = Agent(
    agent_name="Evaluator",
       system_prompt="Summarize and evaluate the summary and the users demand, always be interested in learning more about the situation with extreme precision.",
    llm=llm,
```

```
    max_loops=1,
    list_base_models=[EvaluatorOuputSchema],
)

# research_agent = Agent(
#     agent_name="Research Agent",
#       system_prompt="Summarize and evaluate the summary and the users demand, always be interested in learning more about the situation with extreme precision.",
#     llm=llm,
#     max_loops=1,
#     tool = [webbrowser]
# )

agent3 = Agent(
    agent_name="Summarizer Agent",
    system_prompt="Summarize the entire history of the interaction",
    llm=llm,
    max_loops=1,
    long_term_memory=memory,
)


# Task
task = "Car Model: S-Class, Car Year: 2020, Car Mileage: 10000, all my service lights are on, what should i do?"
```

```
# Swarm
swarm = AgentRearrange(
    agents=[agent1, agent2, agent3],
    flow=f"{agent1.agent_name} -> {agent2.agent_name} -> {agent3.agent_name}",
    memory_system=memory,
)

# Task
out = swarm.run(task)
print(out)
```