```
import os

from dotenv import load_dotenv

from swarms import Agent, SequentialWorkflow

from swarm_models import OpenAIChat


load_dotenv()


# Get the OpenAI API key from the environment variable

api_key = os.getenv("GROQ_API_KEY")


# Model

model = OpenAIChat(

    openai_api_base="https://api.groq.com/openai/v1",

    openai_api_key=api_key,

    model_name="llama-3.1-70b-versatile",

    temperature=0.1,

)




# Initialize specialized agents

data_extractor_agent = Agent(

    agent_name="Data-Extractor",

    system_prompt=None,

    llm=model,

    max_loops=1,

    autosave=True,
    verbose=True,
    dynamic_temperature_enabled=True,
    saved_state_path="data_extractor_agent.json",
    user_name="pe_firm",
    retry_attempts=1,
    context_length=200000,
    output_type="string",
)

summarizer_agent = Agent(
    agent_name="Document-Summarizer",
    system_prompt=None,
    llm=model,
    max_loops=1,
    autosave=True,
    verbose=True,
    dynamic_temperature_enabled=True,
    saved_state_path="summarizer_agent.json",
    user_name="pe_firm",
    retry_attempts=1,
    context_length=200000,
    output_type="string",
)

financial_analyst_agent = Agent(
    agent_name="Financial-Analyst",
    system_prompt=None,
    llm=model,
    max_loops=1,
    autosave=True,
    verbose=True,
    dynamic_temperature_enabled=True,
    saved_state_path="financial_analyst_agent.json",
    user_name="pe_firm",
    retry_attempts=1,
    context_length=200000,
    output_type="string",
)

market_analyst_agent = Agent(
    agent_name="Market-Analyst",
    system_prompt=None,
    llm=model,
    max_loops=1,
    autosave=True,
    verbose=True,
    dynamic_temperature_enabled=True,
    saved_state_path="market_analyst_agent.json",
    user_name="pe_firm",
    retry_attempts=1,
    context_length=200000,
    output_type="string",
)


operational_analyst_agent = Agent(
    agent_name="Operational-Analyst",
    system_prompt=None,
    llm=model,
    max_loops=1,
    autosave=True,
    verbose=True,
    dynamic_temperature_enabled=True,
    saved_state_path="operational_analyst_agent.json",
    user_name="pe_firm",
    retry_attempts=1,
    context_length=200000,
    output_type="string",
)


# Initialize the SwarmRouter
router = SequentialWorkflow(
    name="pe-document-analysis-swarm",
            description="Analyze documents for private equity due diligence and investment
decision-making",
    max_loops=1,
    agents=[
        data_extractor_agent,
        summarizer_agent,
        financial_analyst_agent,
        market_analyst_agent,
        operational_analyst_agent,
    ],
    output_type="all",
)


# Example usage
if __name__ == "__main__":
    # Run a comprehensive private equity document analysis task
    result = router.run(
        "Where is the best place to find template term sheets for series A startups. Provide links and references",
        img=None,
    )
    print(result)
```