```
from swarm_models.openai_function_caller import OpenAIFunctionCaller
from pydantic import BaseModel, Field
from typing import List


class Observation(BaseModel):
    observation: str = Field(
        ...,
        description="What are you seeing in the image?",
    )
    summary_of_observation: str = Field(
        ...,
        description="The summary of the observation/ img",
    )


class Sequence(BaseModel):
    goal: str = Field(
        ...,
        description="The goal of the mission",
    )
    observation: List[Observation] = Field(
        ...,
        description="The observations of the agent",
    )
    action: str = Field(
```

```
        ...,
        description="Take an action that leads to the completion of the task.",
    )


class GoalDecomposer(BaseModel):
    goal: str = Field(
        ...,
        description="The goal of the task",
    )
    sub_goals: List[str] = Field(
        ...,
        description="The sub goals of the mission",
    )


# Given the task t, observation o, the sub-goals
# sequence g1, g2, g3, ..., gn can be formulated as:


class KGP(BaseModel):
    task: str = Field(
        ...,
        description="The task to be accomplished",
    )
    observation: str = Field(
```

```
        ...,
        description="The observation of the task",
    )
    sequence: List[GoalDecomposer] = Field(
        ...,
        description="The sequence of goals to accomplish the task",
    )


# Example usage:
# Initialize the function caller
model = OpenAIFunctionCaller(
      system_prompt="You're an autonomous agent, you're purpose to accomplish a task through understanding your goal, observing the environment, and taking actions that lead to the completion of the task.",
    max_tokens=500,
    temperature=0.5,
    base_model=KGP,
    parallel_tool_calls=False,
)


# The OpenAIFunctionCaller class is used to interact with the OpenAI API and make function calls.
out = model.run(
    "We need to craft a diamond pickaxe to mine the obsidian."
)
```

```
print(out)
```