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


class AgentSpec(BaseModel):
    agent_name: str = Field(
        ...,
        description="The name of the agent",
    )
    system_prompt: str = Field(
        ...,
        description="The system prompt for the agent",
    )
    agent_description: str = Field(
        ...,
        description="The description of the agent",
    )
    max_tokens: int = Field(
        ...,
        description="The maximum number of tokens to generate in the API response",
    )
    temperature: float = Field(
        ...,
```

```
        description="A parameter that controls the randomness of the generated text",
    )
    context_window: int = Field(
        ...,
        description="The context window for the agent",
    )
    task: str = Field(
        ...,
        description="The main task for the agent",
    )


class SwarmSpec(BaseModel):
    multiple_agents: List[AgentSpec] = Field(
        ...,
        description="The list of agents in the swarm",
    )


def create_agent(
    agent_name: str,
    system_prompt: str,
    agent_description: str,
    max_tokens: int,
    temperature: float,
    context_window: int,
```

```
):
    return Agent(
        agent_name=agent_name,
        system_prompt=system_prompt,
        agent_description=agent_description,
        max_tokens=max_tokens,
        temperature=temperature,
        context_window=context_window,
    )



# Example usage:
# Initialize the function caller
model = OpenAIFunctionCaller(
      system_prompt="You're an agent creator, you're purpose is to create an agent with the user provided specifications. Think of relevant names, descriptions, and context windows for the agent. You need to provide the name of the agent, the system prompt for the agent, the description of the agent, the maximum number of tokens to generate in the API response, the temperature for the agent, the context window for the agent, and the model name for the agent from huggingface.",
    max_tokens=3000,
    temperature=0.8,
    base_model=SwarmSpec,
    parallel_tool_calls=False,
)
```

```
def parse_json_for_agents_then_create_agents(
    function_call: dict,
) -> str:
    agents = []
    for agent in json["multiple_agents"]:
        agents.append(
            create_agent(
                agent["agent_name"],
                agent["system_prompt"],
                agent["agent_description"],
                agent["max_tokens"],
                agent["temperature"],
                agent["context_window"],
                # agent["model_name"]
            )
        )
    return agents


# The OpenAIFunctionCaller class is used to interact with the OpenAI API and make function calls.
out = model.run(
    "Create a swarm of agents to generate social media posts. Each agent should have it's own social media"
)
```