```python
from pydantic import BaseModel, Field

from swarm_models import OpenAIChat

from swarms import Agent

import os



# Initialize the schema for the person's information

class Schema(BaseModel):

    name: str = Field(..., title="Name of the person")

    agent: int = Field(..., title="Age of the person")

    is_student: bool = Field(

        ..., title="Whether the person is a student"

    )

    courses: list[str] = Field(

        ..., title="List of courses the person is taking"

    )



# Convert the schema to a JSON string

tool_schema = Schema(

    name="Tool Name",

    agent=1,

    is_student=True,

    courses=["Course1", "Course2"],

)
```

```
# Define the task to generate a person's information
task = (
    "Generate a person's information based on the following schema:"
)

# Initialize the agent
agent = Agent(
    agent_name="Person Information Generator",
    system_prompt=(
        "Generate a person's information based on the following schema:"
    ),
    # Set the tool schema to the JSON string -- this is the key difference
    # tool_schema=tool_schema,
    llm=OpenAIChat(
        openai_api_key=os.getenv("OPENAI_API_KEY"),
    ),
    max_loops=3,
    autosave=True,
    dashboard=False,
    streaming_on=True,
    verbose=True,
    interactive=True,
    # Set the output type to the tool schema which is a BaseModel
    # output_type=tool_schema,  # or dict, or str
    metadata_output_type="json",
    # List of schemas that the agent can handle
    list_base_models=[tool_schema],

    function_calling_format_type="OpenAI",

    function_calling_type="json",  # or soon yaml

)


# Run the agent to generate the person's information

generated_data = agent.run(task)


# Print the generated data

print(f"Generated data: {generated_data}")
```