```
import os


from dotenv import load_dotenv

from examples.demos.plant_biologist_swarm.prompts import (

    diagnoser_agent,

    disease_detector_agent,

    growth_predictor_agent,

    harvester_agent,

    treatment_recommender_agent,

)


from swarms import Agent, ConcurrentWorkflow

from swarm_models.gpt_o import GPT4VisionAPI




# Load the OpenAI API key from the .env file

load_dotenv()


# Initialize the OpenAI API key

api_key = os.environ.get("OPENAI_API_KEY")


# GPT4VisionAPI

llm = GPT4VisionAPI(

    max_tokens=4000,

)
```

```
# Initialize Diagnoser Agent
diagnoser_agent = Agent(
    agent_name="Diagnoser Agent",
    system_prompt=diagnoser_agent(),
    llm=llm,
    max_loops=1,
    dashboard=False,
    streaming_on=True,
    verbose=True,
    # saved_state_path="diagnoser.json",
    multi_modal=True,
    autosave=True,
)

# Initialize Harvester Agent
harvester_agent = Agent(
    agent_name="Harvester Agent",
    system_prompt=harvester_agent(),
    llm=llm,
    max_loops=1,
    dashboard=False,
    streaming_on=True,
    verbose=True,
    # saved_state_path="harvester.json",
    multi_modal=True,
    autosave=True,
)


# Initialize Growth Predictor Agent
growth_predictor_agent = Agent(
    agent_name="Growth Predictor Agent",
    system_prompt=growth_predictor_agent(),
    llm=llm,
    max_loops=1,
    dashboard=False,
    streaming_on=True,
    verbose=True,
    # saved_state_path="growth_predictor.json",
    multi_modal=True,
    autosave=True,
)


# Initialize Treatment Recommender Agent
treatment_recommender_agent = Agent(
    agent_name="Treatment Recommender Agent",
    system_prompt=treatment_recommender_agent(),
    llm=llm,
    max_loops=1,
    dashboard=False,
    streaming_on=True,
    verbose=True,
    # saved_state_path="treatment_recommender.json",
    multi_modal=True,
    autosave=True,
)


# Initialize Disease Detector Agent
disease_detector_agent = Agent(
    agent_name="Disease Detector Agent",
    system_prompt=disease_detector_agent(),
    llm=llm,
    max_loops=1,
    dashboard=False,
    streaming_on=True,
    verbose=True,
    # saved_state_path="disease_detector.json",
    multi_modal=True,
    autosave=True,
)
agents = [
    diagnoser_agent,
    disease_detector_agent,
    treatment_recommender_agent,
    growth_predictor_agent,
    harvester_agent,
]
```

```
# Create the Concurrent workflow

workflow = ConcurrentWorkflow(

    agents=agents,

    max_loops=1,

)


workflow.run("Diagnose the plant disease.")
```