```
import os

from dotenv import load_dotenv

import swarms.prompts.urban_planning as upp
from swarm_models import GPT4VisionAPI, OpenAIChat
from swarms.structs import Agent, SequentialWorkflow

# Load environment variables
load_dotenv()
api_key = os.getenv("OPENAI_API_KEY")
stability_api_key = os.getenv("STABILITY_API_KEY")

# Initialize language model
llm = OpenAIChat(
    openai_api_key=api_key, temperature=0.5, max_tokens=3000
)

# Initialize Vision model
vision_api = GPT4VisionAPI(api_key=api_key)

# Initialize agents for urban planning tasks
architecture_analysis_agent = Agent(
    llm=llm, max_loops=1, sop=upp.ARCHITECTURE_ANALYSIS_PROMPT
)
infrastructure_evaluation_agent = Agent(
    llm=llm, max_loops=1, sop=upp.INFRASTRUCTURE_EVALUATION_PROMPT
)
traffic_flow_analysis_agent = Agent(
    llm=llm, max_loops=1, sop=upp.TRAFFIC_FLOW_ANALYSIS_PROMPT
)
environmental_impact_assessment_agent = Agent(
    llm=llm,
    max_loops=1,
    sop=upp.ENVIRONMENTAL_IMPACT_ASSESSMENT_PROMPT,
)
public_space_utilization_agent = Agent(
    llm=llm, max_loops=1, sop=upp.PUBLIC_SPACE_UTILIZATION_PROMPT
)
socioeconomic_impact_analysis_agent = Agent(
    llm=llm, max_loops=1, sop=upp.SOCIOECONOMIC_IMPACT_ANALYSIS_PROMPT
)

# Initialize the final planning agent
final_plan_agent = Agent(
    llm=llm, max_loops=1, sop=upp.FINAL_URBAN_IMPROVEMENT_PLAN_PROMPT
)

# Create Sequential Workflow
workflow = SequentialWorkflow(max_loops=1)

# Add tasks to workflow with personalized prompts
```

```
workflow.add(architecture_analysis_agent, "Architecture Analysis")
workflow.add(
    infrastructure_evaluation_agent, "Infrastructure Evaluation"
)
workflow.add(traffic_flow_analysis_agent, "Traffic Flow Analysis")
workflow.add(
    environmental_impact_assessment_agent,
    "Environmental Impact Assessment",
)
workflow.add(
    public_space_utilization_agent, "Public Space Utilization"
)
workflow.add(
    socioeconomic_impact_analysis_agent,
    "Socioeconomic Impact Analysis",
)
workflow.add(
    final_plan_agent,
    (
        "Generate the final urban improvement plan based on all"
        " previous agent's findings"
    ),
)
# Run the workflow for individual analysis tasks

# Execute the workflow for the final planning
```

```
workflow.run()

# Output results for each task and the final plan
for task in workflow.tasks:
    print(
        f"Task Description: {task.description}\nResult:"
        f" {task.result}\n"
    )
```