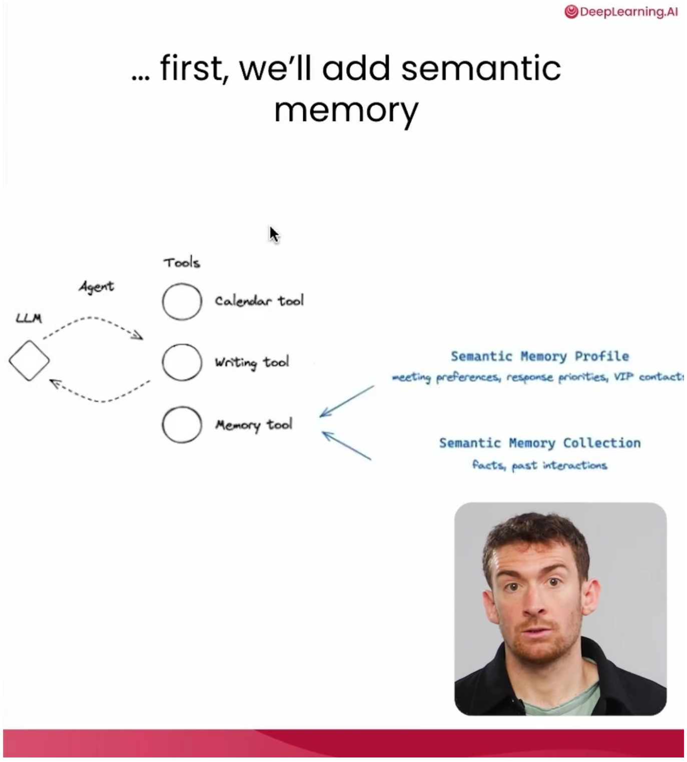
一、具有语义记忆的邮件助手 ***00:01***

1. 语义记忆介绍 00:10



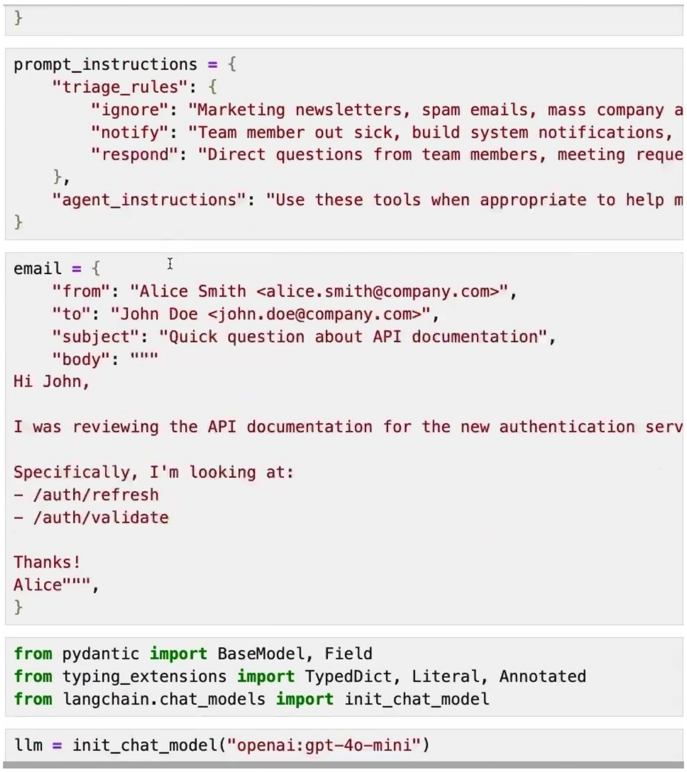
DeepLearning.AI

… first, we'll add semantic memory

- **语义记忆功能**: 用于存储用户相关的事实信息(FACTS)，并将其保存在长期记忆存储中，后续可通过专用工具进行检索。
- **实现方式**: 在已有邮件助手基础上添加语义记忆操作工具，包括管理记忆工具(manage_memory_tool)和搜索记忆工具(search_memory_tool)。

2. 代码示例 00:25

1）环境配置 00:27



```
}
```

```
prompt_instructions = {
    "triage_rules": {
        "ignore": "Marketing newsletters, spam emails, mass company a
        "notify": "Team member out sick, build system notifications,
        "respond": "Direct questions from team members, meeting reque
    },
    "agent_instructions": "Use these tools when appropriate to help m
}
```

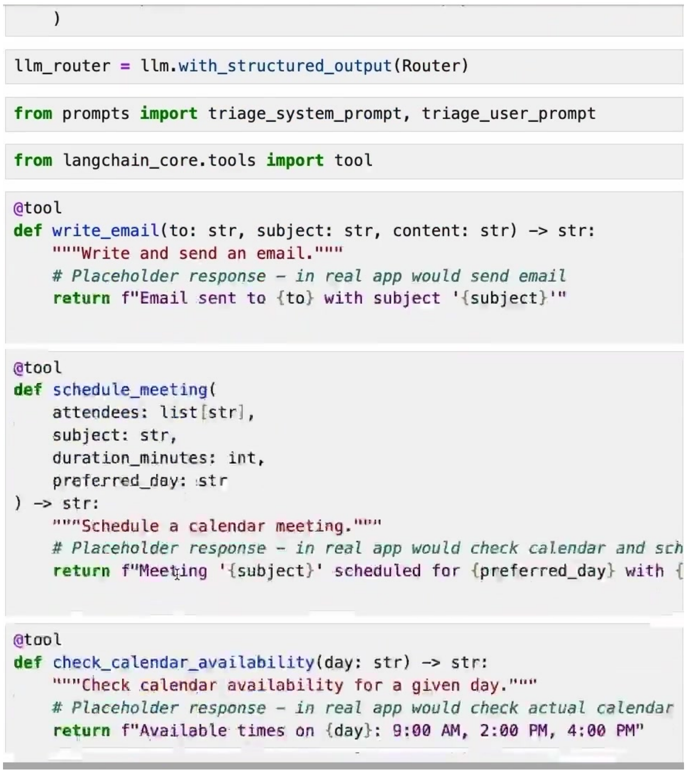
```
email = {
    "from": "Alice Smith <alice.smith@company.com>",
    "to": "John Doe <john.doe@company.com>",
    "subject": "Quick question about API documentation",
    "body": """
Hi John,

I was reviewing the API documentation for the new authentication serv

Specifically, I'm looking at:
- /auth/refresh
- /auth/validate

Thanks!
Alice""",
}
```

```
from pydantic import BaseModel, Field
from typing_extensions import TypedDict, Literal, Annotated
from langchain.chat_models import init_chat_model
```

```
llm = init_chat_model("openai:gpt-4o-mini")
```

- **初始化配置**: 沿用上节课创建的邮件助手环境，加载环境变量并定义用户配置信息。
- **记忆存储设置**:　　使用InMemoryStore作为临时存储，配置OpenAI的text-embedding-3-small模型进行记忆索引。

2）创建管理内存和搜索内存工具 01:30



```
    )
```

```
llm_router = llm.with_structured_output(Router)
```

```
from prompts import triage_system_prompt, triage_user_prompt
```

```
from langchain_core.tools import tool
```

```
@tool
def write_email(to: str, subject: str, content: str) -> str:
    """Write and send an email."""
    # Placeholder response - in real app would send email
    return f"Email sent to {to} with subject '{subject}'"
```

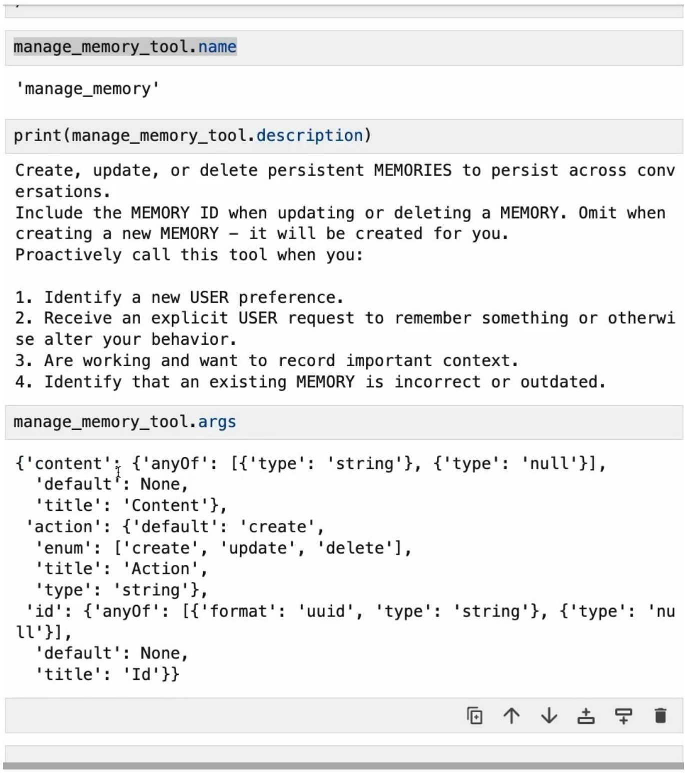
```
@tool
def schedule_meeting(
    attendees: list[str],
    subject: str,
    duration_minutes: int,
    preferred_day: str
) -> str:
    """Schedule a calendar meeting."""
    # Placeholder response - in real app would check calendar and sch
    return f"Meeting '{subject}' scheduled for {preferred_day} with {
```

```
@tool
def check_calendar_availability(day: str) -> str:
    """Check calendar availability for a given day."""
    # Placeholder response - in real app would check actual calendar
    return f"Available times on {day}: 9:00 AM, 2:00 PM, 4:00 PM"
```

- **工具来源**: 从langmem库导入create_manage_memory_tool和create_search_memory_tool函数。

- **命名空间设置**: 使用三层结构("email_assistant", "{langgraph_user_id}", "collection")，其中langgraph_user_id在运行时动态传入。
- **用户隔离**: 不同用户的记忆存储在不同命名空间下，避免记忆混淆。

3）检查内存工具 02:43



```
manage_memory_tool.name
```

```
'manage_memory'
```

```
print(manage_memory_tool.description)
```

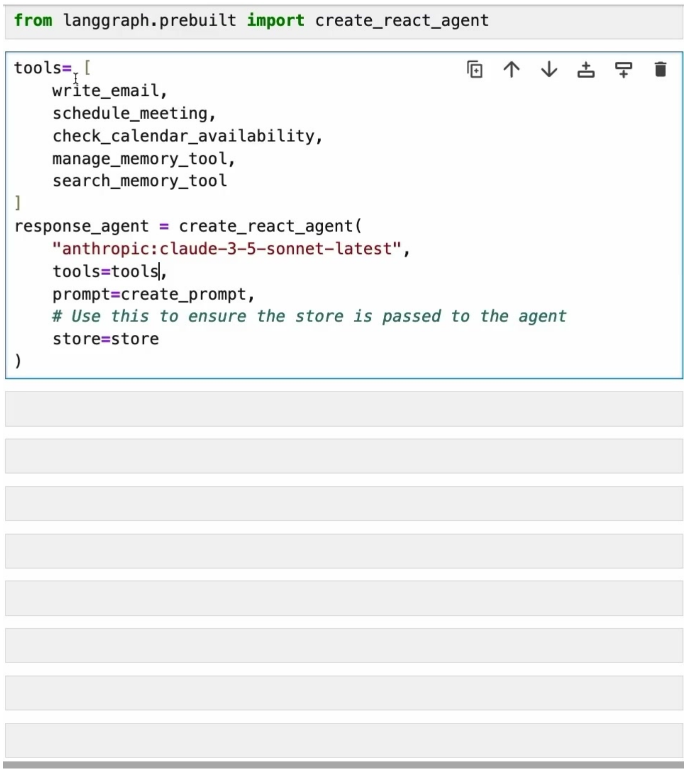
```
Create, update, or delete persistent MEMORIES to persist across conv
ersations.
Include the MEMORY ID when updating or deleting a MEMORY. Omit when
creating a new MEMORY - it will be created for you.
Proactively call this tool when you:

1. Identify a new USER preference.
2. Receive an explicit USER request to remember something or otherwi
se alter your behavior.
3. Are working and want to record important context.
4. Identify that an existing MEMORY is incorrect or outdated.
```

```
manage_memory_tool.args
```

```
{'content': {'anyOf': [{'type': 'string'}, {'type': 'null'}],
  'default': None,
  'title': 'Content'},
 'action': {'default': 'create',
  'enum': ['create', 'update', 'delete'],
  'title': 'Action',
  'type': 'string'},
 'id': {'anyOf': [{'format': 'uuid', 'type': 'string'}, {'type': 'nu
ll'}],
  'default': None,
  'title': 'Id'}}
```

-
- **管理工具功能**:
  - **名称**: manage_memory
  - **操作类型**: 支持create/update/delete三种操作
  - **参数**: content(内容)、action(操作类型)、id(记忆ID)
  - **触发场景**: 识别新用户偏好、收到明确记忆请求、记录重要上下文、修正错误记忆时
- **搜索工具功能**:
  - **名称**: search_memory
  - **参数**: query(查询内容)、limit(返回数量)、offset(偏移量)、filter(过滤器)
  - **评分机制**: 使用余弦相似度计算查询与记忆的匹配度

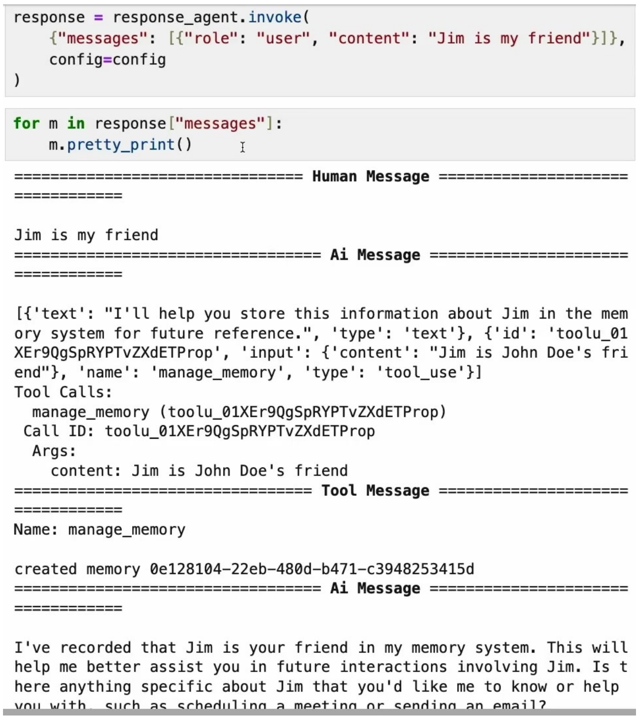
4）创建响应者 04:17



```python
from langgraph.prebuilt import create_react_agent
```

```python
tools= [
    write_email,
    schedule_meeting,
    check_calendar_availability,
    manage_memory_tool,
    search_memory_tool
]
response_agent = create_react_agent(
    "anthropic:claude-3-5-sonnet-latest",
    tools=tools,
    prompt=create_prompt,
    # Use this to ensure the store is passed to the agent
    store=store
)
```

-
- **系统提示修改**: 在原有工具基础上新增两个记忆管理工具。
- **存储传递**: 创建响应者时需传入store参数，确保记忆工具能访问存储。
- 测试响应者 05:03

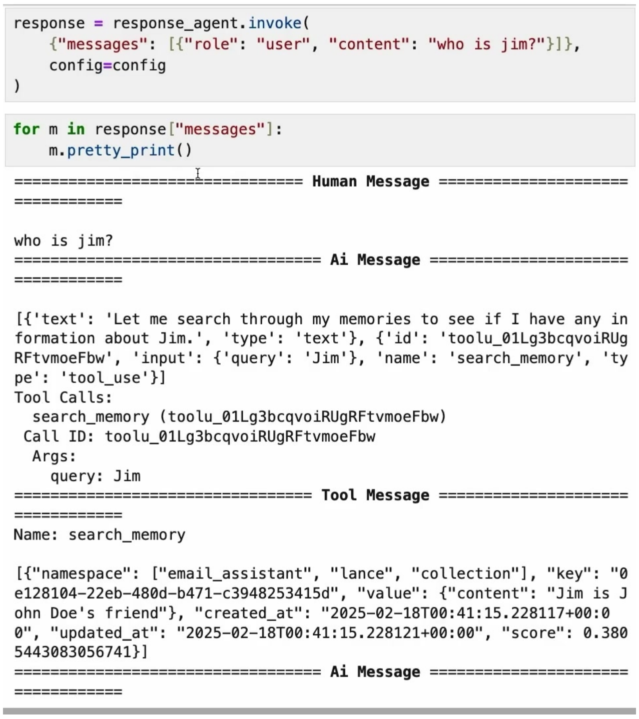


```python
response = response_agent.invoke(
    {"messages": [{"role": "user", "content": "Jim is my friend"}]},
    config=config
)
```

```python
for m in response["messages"]:
    m.pretty_print()
```

```
================================ Human Message =====================
============

Jim is my friend
================================== Ai Message ======================
============

[{'text': "I'll help you store this information about Jim in the mem
ory system for future reference.", 'type': 'text'}, {'id': 'toolu_01
XEr9QgSpRYPTvZXdETProp', 'input': {'content': "Jim is John Doe's fri
end"}, 'name': 'manage_memory', 'type': 'tool_use'}]
Tool Calls:
  manage_memory (toolu_01XEr9QgSpRYPTvZXdETProp)
 Call ID: toolu_01XEr9QgSpRYPTvZXdETProp
  Args:
    content: Jim is John Doe's friend
================================= Tool Message =====================
============
Name: manage_memory

created memory 0e128104-22eb-480d-b471-c3948253415d
================================== Ai Message ======================
============

I've recorded that Jim is your friend in my memory system. This will
help me better assist you in future interactions involving Jim. Is t
here anything specific about Jim that you'd like me to know or help
you with, such as scheduling a meeting or sending an email?
```

  -
  - **配置参数**: 必须包含langgraph_user_id用于记忆隔离。
  - **记忆创建**: 默认action为create，不指定id时会自动生成。
  - **记忆检索**: 通过search_memory工具查询已存储信息。
- 使用响应者 06:17



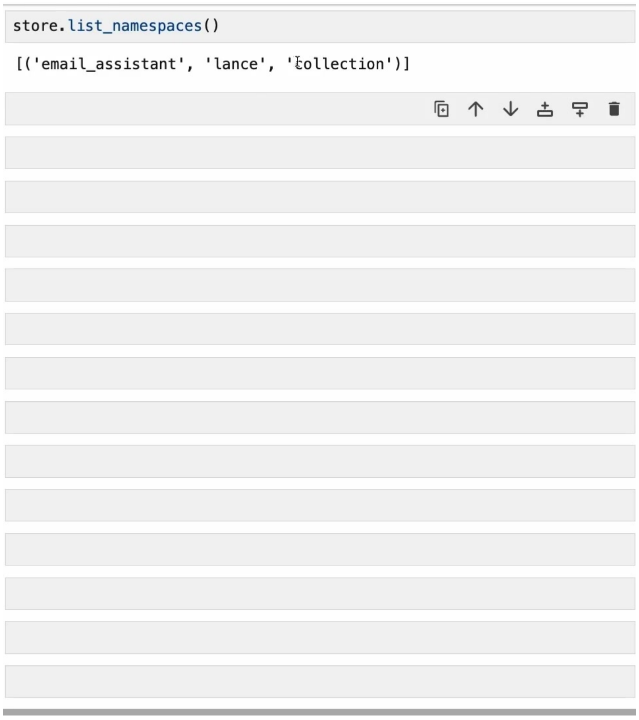
```
response = response_agent.invoke(
    {"messages": [{"role": "user", "content": "who is jim?"}]},
    config=config
)
```

```
for m in response["messages"]:
    m.pretty_print()
```

```
================================ Human Message =====================
============

who is jim?
================================== Ai Message ======================
============

[{'text': 'Let me search through my memories to see if I have any in
formation about Jim.', 'type': 'text'}, {'id': 'toolu_01Lg3bcqvoiRUg
RFtvmoeFbw', 'input': {'query': 'Jim'}, 'name': 'search_memory', 'ty
pe': 'tool_use'}]
Tool Calls:
  search_memory (toolu_01Lg3bcqvoiRUgRFtvmoeFbw)
 Call ID: toolu_01Lg3bcqvoiRUgRFtvmoeFbw
  Args:
    query: Jim
================================= Tool Message =====================
============
Name: search_memory

[{"namespace": ["email_assistant", "lance", "collection"], "key": "0
e128104-22eb-480d-b471-c3948253415d", "value": {"content": "Jim is J
ohn Doe's friend"}, "created_at": "2025-02-18T00:41:15.228117+00:0
0", "updated_at": "2025-02-18T00:41:15.228121+00:00", "score": 0.380
5443083056741}]
================================== Ai Message ======================
============
```

- **查询流程**: 用户提问→AI调用搜索工具→返回匹配记忆→AI总结响应。
- **记忆内容**: 存储为键值对形式，包含创建时间、更新时间等信息。

- 检查内存商店 07:13



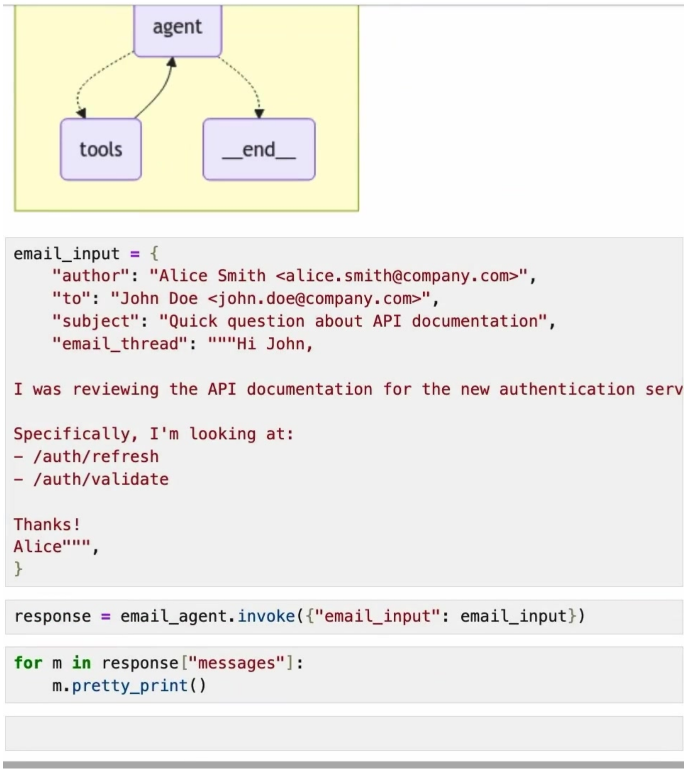
```
store.list_namespaces()
```

```
[('email_assistant', 'lance', 'collection')]
```

- **命名空间查看**: 通过store.list_namespaces()查看现有命名空间。
- **记忆搜索**: 可指定查询条件获取相关记忆及其匹配分数。

3. 创建余下的代理人 08:32

1）配置项目 09:37

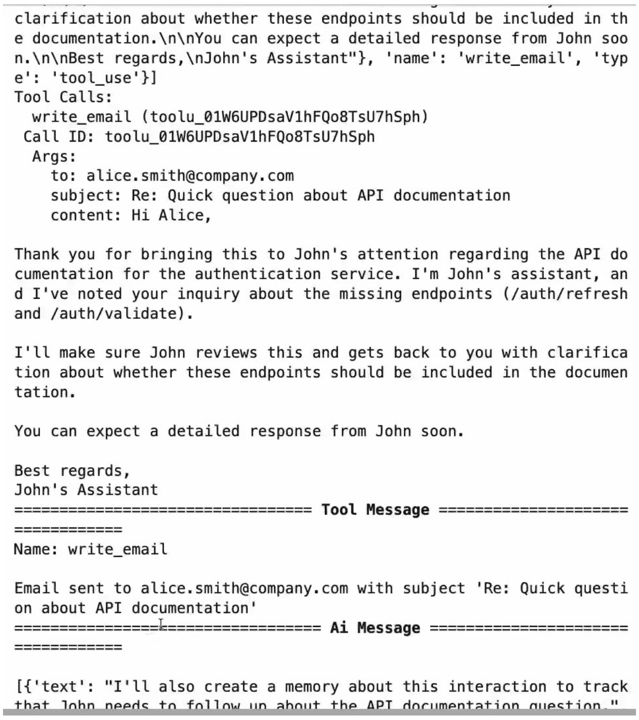


```
email_input = {
    "author": "Alice Smith <alice.smith@company.com>",
    "to": "John Doe <john.doe@company.com>",
    "subject": "Quick question about API documentation",
    "email_thread": """Hi John,

I was reviewing the API documentation for the new authentication serv

Specifically, I'm looking at:
- /auth/refresh
- /auth/validate

Thanks!
Alice""",
}
```

```
response = email_agent.invoke({"email_input": email_input})
```

```
for m in response["messages"]:
    m.pretty_print()
```

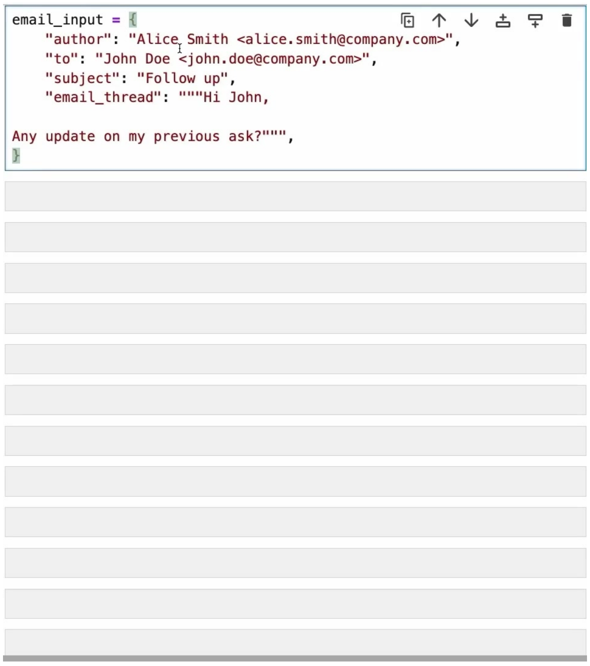
-
- **状态图构建**: 与上节课相同，包含triage_router和response_agent两个节点。
- **关键修改**: 在编译时传入store参数，使整个代理能访问记忆存储。
- 测试邮件响应 09:48



```
clarification about whether these endpoints should be included in th
e documentation.\n\nYou can expect a detailed response from John soo
n.\n\nBest regards,\nJohn's Assistant"}, 'name': 'write_email', 'typ
e': 'tool_use'}]
Tool Calls:
  write_email (toolu_01W6UPDsaV1hFQo8TsU7hSph)
 Call ID: toolu_01W6UPDsaV1hFQo8TsU7hSph
  Args:
    to: alice.smith@company.com
    subject: Re: Quick question about API documentation
    content: Hi Alice,

Thank you for bringing this to John's attention regarding the API do
cumentation for the authentication service. I'm John's assistant, an
d I've noted your inquiry about the missing endpoints (/auth/refresh
and /auth/validate).

I'll make sure John reviews this and gets back to you with clarifica
tion about whether these endpoints should be included in the documen
tation.

You can expect a detailed response from John soon.

Best regards,
John's Assistant
================================= Tool Message =====================
============
Name: write_email

Email sent to alice.smith@company.com with subject 'Re: Quick questi
on about API documentation'
================================== Ai Message ======================
============

[{'text': "I'll also create a memory about this interaction to track
that John needs to follow up about the API documentation question."
```

  -
  - **处理流程**:
    - 接收邮件输入
    - 分类判断(ignore/notify/respond)
    - 需要回复时搜索相关记忆
    - 生成回复并创建新记忆
  - **记忆应用**: 新邮件能关联之前存储的相关上下文信息。
- 应用案例 10:56

- 例题:邮件处理



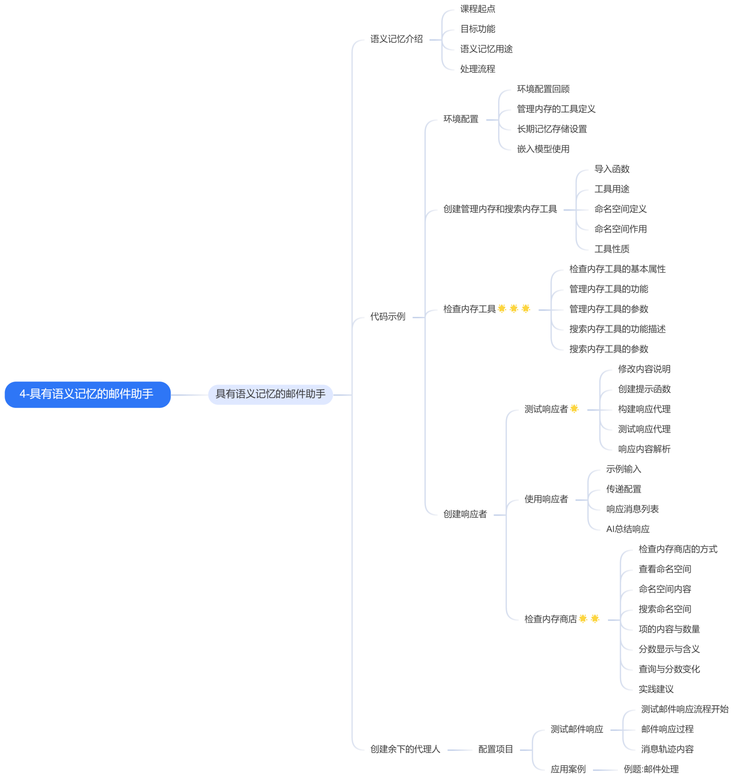
```
email_input = {
    "author": "Alice Smith <alice.smith@company.com>",
    "to": "John Doe <john.doe@company.com>",
    "subject": "Follow up",
    "email_thread": """Hi John,

Any update on my previous ask?""",
}
```

-
- **案例演示**: 展示对跟进邮件的处理过程，验证记忆工具的实用性。
- **效果验证**: 代理能正确检索之前存储的API文档问题记忆，生成连贯回复。

二、知识小结

知识点	核心内容	考试重点/易混淆点	难度系数
语义记忆工具集成	在邮件代理中添加工具，用于操作语义记忆（存储用户相关事实）	记忆存储的命名空间管理（langgraph_user_id）	★★
记忆存储配置	使用内存存储（InMemoryStore）并嵌入OpenAI text-embedding-3-small模型	嵌入模型对记忆索引的影响	★★
记忆管理工具	create_manage_memory_tool：支持创建/更新/删除记忆（需ID、动作、内容）	**默认动作为创建**，ID可自动生成	★★★

记忆搜索工具	create_search_memory_tool：基于查询检索记忆（支持过滤和分页）	相似度评分（余弦相似度）的计算逻辑	★★
代理工作流验证	测试代理记忆功能（如存储“Jim是朋友”，后续查询“Who is Jim?”）	**跨会话记忆持久性**验证	★★★
实际应用案例	代理自动保存“Alice询问API端点”记忆，并在后续邮件中调用	记忆内容与邮件上下文的关联性	★★★★


4-具有语义记忆的邮件助手
具有语义记忆的邮件助手
语义记忆介绍
课程起点
目标功能
语义记忆用途
处理流程
代码示例
环境配置
环境配置回顾
管理内存的工具定义
长期记忆存储设置
嵌入模型使用
创建管理内存和搜索内存工具
导入函数
工具用途
命名空间定义
命名空间作用
工具性质
检查内存工具
检查内存工具的基本属性
管理内存工具的功能
管理内存工具的参数
搜索内存工具的功能描述
搜索内存工具的参数
创建响应者
测试响应者
修改内容说明
创建提示函数
构建响应代理
测试响应代理
响应内容解析
使用响应者
示例输入
传递配置
响应消息列表
AI总结响应
检查内存商店
检查内存商店的方式
查看命名空间
命名空间内容
搜索命名空间
项的内容与数量
分数显示与含义
查询与分数变化
实践建议
创建余下的代理人
配置项目
测试邮件响应
测试邮件响应流程开始
邮件响应过程
消息轨迹内容
应用案例
例题:邮件处理