individual_assignment_day1_V1

September 4, 2025

1 OpenAI APIs Assignment: Prompt Engineering & Agent Pattern: Tool Use

1.1 Assignment Overview

This assignment focuses on the following scenarios: * Tool Use with Wikipedia: Implementing function calling with Wikipedia integration

1.2 Assignment Structure: Tool Use with Wikipedia Integration

Objective: Create a research assistant that can search Wikipedia and provide comprehensive information using OpenAI's function calling capabilities.

1.3 Requirements:

- Wikipedia Search Function: Implement a function to search Wikipedia articles
- Content Extraction: Extract and summarize article content
- Function Calling: Use OpenAI's function calling to integrate Wikipedia data
- Structured Output: Return information in a structured format
- Error Handling: Handle cases where articles don't exist or are ambiguous

1.4 Key Implementation Tasks:

“‘# Required libraries import wikipedia import openai from dotenv import load_dotenv

2 Function 1: Wikipedia Search

def search_wikipedia(query): “ “ “Search Wikipedia for articles related to the query” “ ” # Students implement Wikipedia search using wikipediaapi # Return structured results with titles and descriptions

3 Function 2: Content Extraction

def get_wikipedia_content(title): “ “ “Get full content of a Wikipedia article” “ ” # Students implement content extraction # Handle missing articles and disambiguation

4 Function 3: OpenAI Function Calling

def handle_wikipedia_research(query): “ ““Complete function calling workflow”“ ” # Students implement the full workflow: # 1. Send query to OpenAI with function definitions # 2. Handle function calls (search_wikipedia, get_wikipedia_content) # 3. Execute requested functions # 4. Send results back to OpenAI for final response “‘

4.1 You can use one of these test cases to show:

- “What is artificial intelligence?”
- “Tell me about the history of Singapore”
- “Explain quantum computing”

4.2 How to submit your assignment to Canvas

Submit the code with the input and output as a pdf file into Canvas

[1]:
```
#Here is an example of the Wikipedia Search Function and you may need to␣
 ↪install the wikipedia library API
# import wikipedia

# def get_article(search_term):
#     results = wikipedia.search(search_term)
#     first_result = results[0]
#     page = wikipedia.page(first_result, auto_suggest=False)
#     return page.content

# article = get_article("What is artificial intelligence?")
# #print(article[:1000])
# #article is very long, so let's just print a preview

# from IPython.display import display, HTML

# article = get_article("What is artificial intelligence?")
# html_content = f"""
# <div style="background-color: #f8f9fa; padding: 20px; border-radius: 8px;␣
 ↪border-left: 4px solid #007bff;">
#     <h3 style="color: #007bff; margin-top: 0;">What is artificial␣
 ↪intelligence? Answer Preview</h3>
#     <p style="line-height: 1.6; text-align: justify;">{article[:1000]}...</p>
#     <small style="color: #6c757d;">(Showing first 1000 characters)</small>
# </div>
# """
# display(HTML(html_content))
```

I used GPT-4.0 to produce sample codes for creation of some functions which I modified to suit assginment requirements. I am responsible for the content and quality of the submitted work.

[ ]:
```python
# Function 1
import requests

def search_wikipedia(query):
    """
    REST API function that returns title, and description
    for the top 5 search results.

    Args:
        query (str): The query to search for on Wikipedia

    Returns:
        list: A list of dictionaries containing 'title' and'description'
              for up to 5 Wikipedia articles
    """
    language_code = "en"
    base_url = f"https://api.wikimedia.org/core/v1/wikipedia/{language_code}/search/page"

    headers = {
        'User-Agent': 'MyApp/1.0 (your.email@example.com)',
    }

    params = {
        'q': query,
        'limit': 5
    }

    try:
        response = requests.get(base_url, headers=headers, params=params)
        response.raise_for_status()
        data = response.json()

        if 'pages' not in data or not data['pages']:
            return []

        articles = []

        for page_data in data['pages']:
            title = page_data.get('title')
            raw_description = page_data.get('description')
            description = raw_description.strip() if raw_description and raw_description.strip() else 'No description available'


            article_info = {
                'title': title,
```

```
                'description': description

            }
            articles.append(article_info)


        return articles

    except Exception as e:
        print(f"Error querying Wikimedia REST API: {e}")
        return []



# Example usage and testing
if __name__ == "__main__":
    query = "What is artificial intelligence"
    print(f"Searching for: {query}")
    print("=" * 50)

    articles = search_wikipedia(query)

    for i, article in enumerate(articles, 1):
        print(f"\n--- Article {i} ---")
        print(f"Title: {article['title']}")
        print(f"Description: {article['description']}...")
```

```
Searching for: What is artificial intelligence
==================================================

--- Article 1 ---
Title: Artificial intelligence
Description: Intelligence of machines…

--- Article 2 ---
Title: Artificial general intelligence
Description: Type of AI with wide-ranging abilities…

--- Article 3 ---
Title: Applications of artificial intelligence
Description: No description available…

--- Article 4 ---
Title: Friendly artificial intelligence
Description: AI to benefit humanity…

--- Article 5 ---
Title: Symbolic artificial intelligence
```

Description: Methods in artificial intelligence research…

[3]:
```python
# helper function
def process_articles(articles):
    """
    Process a list of Wikipedia articles and return their titles.

    Args:
        articles (list): A list of dictionaries containing article information.

    Returns:
        list: A list containing titles of articles
    """
    titles = []
    for article in articles:
        titles.append(article.get('title'))
    return titles
```

[4]:
```python
# 2nd function
import wikipedia

def get_wikipedia_content(title):
    """
    Using Wikipedia python library returns contents
    for the top 3 search results.

    Args:
        title (str): The title of the Wikipedia article to retrieve content for.

    Returns:
        content (str): The content of the Wikipedia article
    """



    try:
        # Use wikipedia package to get full content
        page = wikipedia.page(title, auto_suggest=False)

        content = page.content

    except wikipedia.exceptions.DisambiguationError as e:
        # Try first disambiguation option
        try:
            page = wikipedia.page(e.options[0], auto_suggest=False)
            content = page.content
        except:
            print(f"Disambiguation error occurred for '{title}'")
```

```
            return []
    except wikipedia.exceptions.PageError:
        print(f"Missing article occurred for '{title}'")
        return []
    except Exception as e:
        print(f"Error retrieving Wikipedia content for '{title}': {e}")
        return []

    return content


# Example usage and testing
if __name__ == "__main__":
    title = "Artificial Intelligence"
    print(f"Extracting content for: {title}")
    print("=" * 50)

    content = get_wikipedia_content(title)

    print(f"Content for '{title}':\n{content[:500]}...")  # Print first 500␣
 ↪characters of content
```

```
Extracting content for: Artificial Intelligence
==================================================
Content for 'Artificial Intelligence':
Artificial intelligence (AI) is the capability of computational systems to
perform tasks typically associated with human intelligence, such as learning,
reasoning, problem-solving, perception, and decision-making. It is a field of
research in computer science that develops and studies methods and software that
enable machines to perceive their environment and use learning and intelligence
to take actions that maximize their chances of achieving defined goals.
High-profile applications of AI incl…
```

[5]:
```
from agents import Agent, trace, Runner, function_tool
from agents.model_settings import ModelSettings
from IPython.display import Markdown
import json
from IPython.display import display, HTML

# 1. Define your tool (Function 3 orchestrator)
@function_tool
def handle_wikipedia_research(query: str):
    """
    Handles the Wikipedia research process for a given query.
    Uses search_wikipedia() + get_wikipedia_content() internally.
    """
```

```
    articles = search_wikipedia(query)          # Function 1
    titles = process_articles(articles)         # Helper

    contents = []
    for title in titles:
        content = get_wikipedia_content(title)  # Function 2
        contents.append(content)

    return json.dumps(contents)


# 2. Instructions for the agent
INSTRUCTIONS = (
    "You are a research assistant. You are to use the extracted information␣
 ↪from Wikipedia "
    "to answer user queries. Your goal is to provide accurate and concise␣
 ↪answers based on "
    "the retrieved content from Wikipedia. If you do not know the answer,␣
 ↪please say so."
)

# 3. Create the Agent object
research_agent = Agent(
    name="Research agent",
    instructions=INSTRUCTIONS,
    model="gpt-4o-mini",
    model_settings=ModelSettings(tool_choice="required"),
    tools=[handle_wikipedia_research],  #  pass the wrapped tool
)

# 4. Run the agent
message = (
    "You are a research assistant. You have to provide a simple explanation of␣
 ↪neural networks. "
    "The goal is to help users understand complex topics easily. The target␣
 ↪audience is elementary school students."
)

with trace("Research_Agent_Search"):
    result = await Runner.run(research_agent, message)

# 5. Display the answer
display(Markdown(result.final_output))

# final=result.final_output

# html_content = f"""
```

```
# <div style="background-color: #f8f9fa; padding: 20px; border-radius: 8px;␣
 ↪border-left: 4px solid #007bff;">
#     <h3 style="color: #007bff; margin-top: 0;">What is artificial␣
 ↪intelligence? Answer Preview</h3>
#     <p style="line-height: 1.6; text-align: justify;color: #6c757d;">{final}..
 ↪.</p>
#     <small style="color: #6c757d;">(Showing first 1000 characters)</small>
# </div>
# """
# display(HTML(html_content))
```

4.2.1 Simple Explanation of Neural Networks

Imagine your brain is like a big web of lights (neurons). When you learn something new, some lights turn on, and they pass messages to each other through connections (like wires). This helps you think, remember, or recognize things.

What are Neural Networks?

1. **Neurons**: Just like the brain, a neural network has units called "neurons." Each neuron can receive information, do some work with it, and send it to other neurons.
2. **Layers**: Neurons are organized into **layers**. The first layer takes in new information (like pictures), and the last layer tells you the answer (like "This is a cat!"). There can be many layers in between that help process the information.
3. **Learning**: When the network learns, it adjusts the strengths of the connections between neurons. This way, it becomes better at solving problems.

4.2.2 How Does Learning Happen?

- The network starts by making guesses.
- If it guesses wrong, it learns from its mistake, kind of like how you practice a sport or a game until you get better at it.

4.2.3 Why Use Neural Networks?

Neural networks are great for tasks like:

- **Recognizing faces** in photos
- **Understanding speech** when you talk to devices
- **Helping robots** figure out how to move and act

In short, neural networks help computers think a little bit like us. They're used in many smart technologies today, making things easier and more fun!