PG9: Take the Institution name as input. Use Pydantic to define the schema for the desired output and create a custom output parser. Invoke the Chain and Fetch Results. Extract the below Institution related details from Wikipedia: The founder of the Institution. When it was founded. The current branches in the institution . How many employees are working in it. A brief 4-line summary of the institution

Soln:

Approach 1: Using Cohere and LangChain

```
# Install the langchain-cohere library (command to be run in the
terminal, not Python code)
# pip install -U langchain-cohere

# Import necessary modules from langchain and pydantic
from langchain.prompts import PromptTemplate # For creating prompt
templates
from langchain.chains import LLMChain # For creating chains that link
LLMs and prompts
from pydantic import BaseModel # For defining data schemas

# Define Pydantic schema for the desired output
class InstitutionDetails(BaseModel):
    """
    Pydantic model to structure the output data for institution
details.
    """
    founder: str # Founder of the institution (string)
    founded: str # Year/date when the institution was founded (string)
    branches: int # Number of current branches (integer)
    employees: int # Number of employees working in the institution
(integer)
    summary: str # A 4-line brief summary of the institution (string)
```

```
# Define the prompt template for GPT-3
prompt_template = """
Given the name of an institution, extract the following details from
Wikipedia:
1. Founder of the institution
2. When it was founded
3. Current branches of the institution
4. How many employees work in it
5. A 4-line brief summary of the institution

Institution: {institution_name}
"""

import getpass
```

```
!pip install langchain-cohere

import os

# Check if the COHERE_API_KEY environment variable is already set
if not os.environ.get("COHERE_API_KEY"):
    # If not set, prompt the user to enter their Cohere API key and set
it as an environment variable
    os.environ["COHERE_API_KEY"] = getpass.getpass("Enter API key for
Cohere: ")

# Import the ChatCohere class from the langchain_cohere library
from langchain_cohere import ChatCohere

# Initialize the ChatCohere model with a specific model version
(command-r7b-12-2024)
model = ChatCohere(model="command-r7b-12-2024")
```

```
# Setup Langchain with the prompt and model

# Create a PromptTemplate object, specifying input variables and the
template
prompt = PromptTemplate(input_variables=["institution_name"],
template=prompt_template)

# Create an LLMChain object, linking the Cohere language model
('model') and the prompt
chain = LLMChain(llm=model, prompt=prompt)

# Function to fetch institution details using GPT-3
def fetch_institution_details(institution_name: str):
    """
    Fetches institution details using the Langchain chain and GPT-3
model.

    Args:
        institution_name (str): The name of the institution to fetch
details for.

    Returns:
        str: The result from the LLMChain run, containing institution
details.
    """
    # Run the LLMChain with the institution name as input and get the
result
    result = chain.run(institution_name=institution_name)
    return result
```

```python
# Take institution name input from the user
institution_name = input("Enter the institution name: ")

# Call the function to fetch institution details, passing the user
input
institution_details = fetch_institution_details(institution_name)

# Print the fetched institution details
print(institution_details)
```

Output:

```
Enter the institution name:

ATME College of Engineering
<ipython-input-5-df0c7c7de135>:21: LangChainDeprecationWarning: The method `Chain.run` was deprecated
in langchain 0.1.0 and will be removed in 1.0. Use :meth:`~invoke` instead.
  result = chain.run(institution_name=institution_name)
Here are the details extracted from Wikipedia for ATME College of Engineering:

**1. Founder:**

* The information about the founder of ATME College of Engineering is not readily available in the provided
context. You would need to search for specific Wikipedia pages or sources related to the college to find this
information.

**2. Founding Date:**

* Similarly, the founding date is not mentioned in the given text.

**3. Current Branches:**

* The source doesn't explicitly state the current branches. You would need to consult the college's Wikipedia
page or other reliable sources for this information.

**4. Number of Employees:**

* Employee count is not provided in the context.

**5. Brief Summary:**

* Unfortunately, a concise summary cannot be generated based on the information given.

**Important Note:**

* The above information is based solely on the content you provided. To obtain accurate and up-to-date details,
it's crucial to consult the official Wikipedia page for ATME College of Engineering or other reliable sources.
```

Approach 2 Using WikiPediaAPIWrapper

```
%pip install --upgrade --quiet  wikipedia
```

```
from langchain_community.tools import WikipediaQueryRun
from langchain_community.utilities import WikipediaAPIWrapper
```

```
from pydantic import BaseModel, Field
import re

# Step 1: Define the Pydantic schema
class InstitutionDetails(BaseModel):
    founder: str = Field(..., description="Founder of the institution")
    founded_year: str = Field(..., description="Year the institution
was founded")
    branches: list[str] = Field(..., description="Current branches in
the institution")
    employees: str = Field(..., description="Number of employees in the
institution")
    summary: str = Field(..., description="A brief 4-line summary of
the institution")

# Step 2: Create a custom output parser
def parse_wikipedia_content(content: str) -> InstitutionDetails:
    founder_match = re.search(r"Founded by\s*([\w\s,]+)", content)
    founded_year_match = re.search(r"Established in\s*(\d{4})",
content)
    branches_match = re.findall(r"(\b[A-Z][a-zA-Z\s]+ Campus\b)",
content)
    employees_match = re.search(r"(\d{3,6})\s*employees", content)

    summary_sentences = content.split(". ")[:4]  # Extract first 4
sentences

    return InstitutionDetails(
        founder=founder_match.group(1) if founder_match else "Not
Found",
        founded_year=founded_year_match.group(1) if founded_year_match
else "Not Found",
        branches=branches_match if branches_match else ["Not Found"],
        employees=employees_match.group(1) if employees_match else "Not
Found",
        summary=". ".join(summary_sentences)
    )

# Step 3: Fetch details from Wikipedia
wiki = WikipediaQueryRun(api_wrapper=WikipediaAPIWrapper())
institution_name = "Apple Company"
wiki_content = wiki.run(institution_name)

# Step 4: Parse and display results
institution_details = parse_wikipedia_content(wiki_content)
```

```python
print(institution_details.model_dump_json(indent=4))
```

Output:

```
{
    "founder": "Not Found",
    "founded_year": "Not Found",
    "branches": [
        "Not Found"
    ],
    "employees": "Not Found",
    "summary": "Page: Apple Inc.\nSummary: Apple Inc. is an American
multinational corporation and technology company headquartered in
Cupertino, California, in Silicon Valley. It is best known for its
consumer electronics, software, and services. Founded in 1976 as Apple
Computer Company by Steve Jobs, Steve Wozniak and Ronald Wayne, the
company was incorporated by Jobs and Wozniak as Apple Computer, Inc"
}
```