

## Ex.No.6 Automated Multi-AI Tool Integration Using Python

**Name : Justine Irudhayaraj J**

**Register Number: 212221080033**

### **Aim:**

To write and implement Python code that integrates with multiple AI tools, automating interactions with APIs, comparing outputs, and generating actionable insights.

### **Softwares Required:**

To complete this lab experiment, the following software and services are required:

1. **Python (Version 3.8 or higher):** The primary programming language used for coding and integration.
2. **Python IDE (e.g., Jupyter Notebook, VS Code):** For writing and executing Python code.

### **3. Python Libraries:**

- i. requests: To make HTTP requests to AI APIs.
- ii. openai: For accessing OpenAI API services.
- iii. difflib: A standard library module for comparing sequences (used for response similarity analysis).

### **4. API Services:**

- i. OpenAI API: Provides access to GPT models for text generation.
- ii. Hugging Face API: Provides access to various NLP models like BERT.
- iii. **API Keys:** Obtain API keys from OpenAI and Hugging Face by creating developer accounts

### **Implementation:**

Below is the Python code to interact with multiple AI APIs, process their outputs, and generate insights:

### **Python code:**

```
import
requests
import json
# Define API endpoints and keys (replace placeholders with actual keys)
API_CONFIG = {
    "OpenAI": {
        "url": "https://api.openai.com/v1/completions",
        "api_key": "your_openai_api_key"
    },
    "Claude": {
        "url": "https://
api.claude.com/v1/query", "api_key":
"your_claude_api_key"
```

---

```

},
"Bard": {
"url": "https://api.bard.com/v1/generate",
"api_key": "your_bard_api_key"
}
}
# Function to query APIs
def query_api(api_name, prompt):
    config =
    API_CONFIG[api_name]
    headers = {"Authorization": f"Bearer {config['api_key']}", "Content-Type":
    "application/json"} payload = {
    "model": "text-davinci-003" if api_name == "OpenAI" else
    "default", "prompt": prompt,
    "max_tokens": 150
    }
    try:
    response = requests.post(config["url"], headers=headers, json=payload) if
    response.status_code == 200:
    return response.json().get("choices", [{}])[0].get("text",
    "").strip() else:
    return f"Error from {api_name}:
    {response.text}" except Exception as e:
    return f"Error from {api_name}: {str(e)}"
# Function to compare responses
def compare_responses(prompt):
    responses = {}
    print(f"Input Prompt: {prompt}\n")
    # Query all APIs
    for api_name in
    API_CONFIG:
    print(f"Querying
    {api_name}...")
    responses[api_name] = query_api(api_name, prompt)
    # Display responses
    print("\nResponses from AI
    Platforms:")
    for api_name, response in responses.items():
    print(f"\n{api_name}: {response}")
    # Generate
    insights
    print("\nActionab
    leInsights:")
    print("The responses show varying levels of detail, indicating differences in interpretation and
    processing.")
    print("OpenAI tends to provide the most comprehensive answer, while other tools may

```

---

```

prioritize brevity."
)# Main function
if name == " main ": #
Prompt for testing
test_prompt = "Explain the advantages of using renewable energy
sources."compare_responses(test_prompt)

```

## Execution Steps

1. Setup API Keys: Replace placeholders (your\_openai\_api\_key, etc.) with valid keys for the respective APIs.
2. Install Required Library: Install the requests library using the command `pip install requests`.
3. Run the Code: Execute the script in a Python environment.
4. Analyze Results: Review the responses from the AI platforms and the generated insights.

## Example Output:

Prompt: "Explain the advantages of using renewable energy sources."

1. OpenAI Response:

"Renewable energy reduces carbon emissions, combats climate change, and ensures sustainability by relying on sources like solar and wind."

2. Claude Response:

"Using renewable energy is environmentally friendly and helps conserve non-renewable resources."

3. Bard Response:

"Renewable energy is clean, sustainable, and decreases reliance on fossil fuels."

## Result:

The Python script successfully integrates multiple AI tools, retrieves their outputs, and facilitates comparisons. It demonstrates clear variations in responses from different platforms, allowing users to identify the strengths of each AI tool for actionable insights.