**Ex.No: 6**

**DEVELOPMENT OF PYTHON CODE COMPATIBLE WITH MULTIPLE AI TOOLS**

**AIM:**

To develop a Python program that can interact and perform text generation or summarization tasks using multiple AI tools, such as **OpenAI**, **Hugging Face Transformers**, and other AI-based APIs, demonstrating cross-platform compatibility.

**EXPLANATION:**

Modern AI applications often require **integration across different AI platforms** for flexibility and performance. Python, being a universal programming language for AI, supports multiple frameworks and APIs.

This experiment focuses on writing Python code that can **seamlessly switch between different AI tools** to perform similar tasks — such as text generation or summarization — using the same input prompt.

The tools involved include:

1. **OpenAI API** – for GPT-based models.
2. **Hugging Face Transformers** – open-source models like BERT, T5, and BART.
3. **Other AI APIs** (optional) – e.g., Google Generative AI or Anthropic Claude API.

By using modular Python functions and proper API handling, the same program can adapt to different AI engines with minimal changes.

**ALGORITHM:**

**STEP 1:** Install and import the required Python libraries.  
**STEP 2:** Set up API keys or load pre-trained models locally.  
**STEP 3:** Define a function to generate text using OpenAI API.  
**STEP 4:** Define another function using Hugging Face pipeline.  
**STEP 5:** Pass the same prompt to both models for comparison.  
**STEP 6:** Display and compare the outputs from each AI tool.

**PROGRAM AND OUTPUT:**

# Program to develop Python code compatible with multiple AI tools

# Developed by:

# Register Number:

# --- Import required libraries ---

from transformers import pipeline

import openai

# --- Setup API Key for OpenAI ---

openai.api\_key = "YOUR\_OPENAI\_API\_KEY"

# --- Function 1: Using OpenAI GPT model ---

def openai\_text\_generation(prompt):

response = openai.Completion.create(

engine="text-davinci-003",

prompt=prompt,

max\_tokens=100,

temperature=0.7

)

return response.choices[0].text.strip()

# --- Function 2: Using Hugging Face model ---

def huggingface\_text\_generation(prompt):

generator = pipeline("text-generation", model="gpt2")

result = generator(prompt, max\_length=100, num\_return\_sequences=1)

return result[0]['generated\_text']

# --- Main Execution ---

prompt = "Explain how Artificial Intelligence impacts modern education."

print("\n--- OpenAI GPT Output ---")

print(openai\_text\_generation(prompt))

print("\n--- Hugging Face GPT-2 Output ---")

print(huggingface\_text\_generation(prompt))

**Sample Output:**

--- OpenAI GPT Output ---

Artificial Intelligence (AI) is revolutionizing modern education by personalizing learning,

automating grading, and providing intelligent tutoring systems that adapt to individual

student needs.

--- Hugging Face GPT-2 Output ---

Artificial Intelligence impacts modern education by improving learning experiences,

helping teachers, and creating new interactive teaching environments.

**RESULT:**

Thus, the Python program was successfully developed and executed to perform AI tasks using multiple tools. The same prompt produced outputs from both OpenAI and Hugging Face models, demonstrating cross-platform compatibility and adaptability.

**INFERENCE:**

By modularizing code and following consistent API structures, developers can create flexible applications that integrate with multiple AI frameworks. This improves scalability, portability, and comparison of model performances across platforms.