**Ex.No: 7**

**DEVELOP A PROMPT-BASED APPLICATION TAILORED TO PERSONAL NEEDS USING LARGE LANGUAGE MODELS**

**AIM:**

To design and implement a Python-based application using Large Language Models (LLMs) that is tailored to the user’s personal needs, fostering creativity and practical problem-solving skills through prompt engineering.

**EXPLANATION:**

Prompt-based applications use **carefully designed instructions** to guide AI models like GPT or BERT in performing specific tasks. By tailoring prompts to personal or practical requirements, LLMs can be used to:

* Generate creative content (stories, poetry, scripts)
* Assist in task automation (scheduling, summaries, code snippets)
* Solve domain-specific problems (education, research, business)
* Provide personalized learning or productivity tools

The process involves:

1. **Identifying personal needs or problems.**
2. **Designing prompts** to instruct the AI effectively.
3. **Developing an interactive application** where the user inputs prompts and receives AI-generated solutions.
4. **Refining prompts** to improve relevance, coherence, and creativity of outputs.

**ALGORITHM:**

**STEP 1:** Install and import necessary Python libraries (openai, transformers, tkinter for GUI, if required).  
**STEP 2:** Collect the user’s personal requirements or problems.  
**STEP 3:** Create functions to generate AI responses using pre-trained LLMs.  
**STEP 4:** Design prompts based on the user’s needs.  
**STEP 5:** Take user input dynamically and pass it to the LLM via the prompt.  
**STEP 6:** Display the AI-generated solution/output.  
**STEP 7:** Refine prompts if the output is unclear or incomplete.

**PROGRAM EXAMPLE (Python):**

# Prompt-based personal assistant using OpenAI GPT

# Developed by:

# Register Number:

import openai

# Setup API key

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

# Function to get AI response based on user prompt

def get\_ai\_response(user\_prompt):

response = openai.Completion.create(

engine="text-davinci-003",

prompt=user\_prompt,

max\_tokens=150,

temperature=0.7

)

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

# Example: Personalized application

print("Welcome to your personal AI assistant!")

name = input("Enter your name: ")

need = input("Describe your requirement (e.g., generate ideas, summarize text, solve a problem): ")

# Tailored prompt

prompt = f"Hello AI, {name} needs help to {need}. Provide a creative and practical solution."

output = get\_ai\_response(prompt)

print("\n--- AI Response ---")

print(output)

**Sample Output:**

Welcome to your personal AI assistant!

Enter your name: Siva

Describe your requirement (e.g., generate ideas, summarize text, solve a problem): plan a study schedule for competitive exams

--- AI Response ---

Hello Siva! Here’s a practical study schedule for your competitive exams:

- Morning (6:00 AM - 8:00 AM): Focus on high-priority subjects

- Midday (10:00 AM - 12:00 PM): Practice mock tests and problem-solving

- Afternoon (2:00 PM - 4:00 PM): Review notes and revise weak topics

- Evening (6:00 PM - 8:00 PM): Group study or discussion sessions

- Night (9:00 PM - 10:00 PM): Summarize the day’s learning and plan tomorrow

Remember to include short breaks to stay productive and avoid burnout!

**RESULT:**

The prompt-based application successfully generated personalized solutions tailored to the user’s needs. By modifying the prompt, outputs could be made **more creative, detailed, or practical** depending on the requirement.

**INFERENCE:**

* Prompt design is crucial for **relevant and actionable AI outputs**.
* Tailoring prompts to personal needs enhances **creativity and practical problem-solving**.
* Applications using LLMs can be flexible, interactive, and scalable for various domains.