Chatbot using LLM

```
from transformers import AutoModelForCausalLM, AutoTokenizer
import torch

# Load pre-trained model and tokenizer
model_name = "microsoft/DialoGPT-medium"
tokenizer = AutoTokenizer.from_pretrained(model_name)
model = AutoModelForCausalLM.from_pretrained(model_name)
```

```
/usr/local/lib/python3.11/dist-packages/huggingface_hub/utils/_auth.py:94: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set it as :
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
  warnings.warn(
tokenizer_config.json: 100%                                      614/614 [00:00<00:00, 20.8kB/s]
vocab.json:      1.04M/? [00:00<00:00, 6.61MB/s]
merges.txt:      456k/? [00:00<00:00, 8.62MB/s]
config.json: 100%                                         642/642 [00:00<00:00, 16.8kB/s]
pytorch_model.bin: 100%                                      863M/863M [00:11<00:00, 65.3MB/s]
model.safetensors: 100%                                      863M/863M [00:18<00:00, 40.5MB/s]
generation_config.json: 100%                                     124/124 [00:00<00:00, 1.30kB/s]
```

```
# Keep track of conversation history
chat_history_ids = None

def ask_bot(user_input, chat_history_ids=None):
    # Encode input and append to chat history if exists
    new_input_ids = tokenizer.encode(user_input + tokenizer.eos_token, return_tensors='pt')
    bot_input_ids = torch.cat([chat_history_ids, new_input_ids], dim=-1) if chat_history_ids is not None else new_input_ids

    # Generate a response
    chat_history_ids = model.generate(
        bot_input_ids,
        max_length=1000,
        pad_token_id=tokenizer.eos_token_id
    )

    # Decode the last generated response
    response = tokenizer.decode(chat_history_ids[:, bot_input_ids.shape[-1]:][0], skip_special_tokens=True)
    return response, chat_history_ids


print("🤖 Chatbot ready! Type 'quit' to stop.\n")

while True:
    user_input = input("🧑 You: ")
    if user_input.lower() == "quit":
        break
    response, chat_history_ids = ask_bot(user_input, chat_history_ids)
    print("🤖 Bot:", response)
```

```
🤖 Chatbot ready! Type 'quit' to stop.

🧑 You: who is your father?
The attention mask is not set and cannot be inferred from input because pad token is same as eos token. As a consequence, you may ol
🤖 Bot: I'm not sure, but I think he's a guy.
🧑 You: I am your father
🤖 Bot: I am your father
🧑 You: I created you
🤖 Bot: I created you
🧑 You: oh so you can only answer questions?
🤖 Bot: I created you
🧑 You: what's the capital of France?
🤖 Bot: I created you
🧑 You: quit
```

```
print("🤖 Chatbot ready! Type 'quit' to stop.\n")

while True:
    user_input = input("🧑 You: ")
    if user_input.lower() == "quit":
        break
```

Que puis-je vous aider à créer ?

```
    response, chat_history_ids = ask_bot(user_input, chat_history_ids)
    print("🤖 Bot:", response)
```

```
🤖 Chatbot ready! Type 'quit' to stop.

🧑 You: who are you?
🤖 Bot: I created you
🧑 You: reset
🤖 Bot: I created you
🧑 You: reset your history
🤖 Bot: reset your history
🧑 You: who are you?
🤖 Bot: I created you
🧑 You: stop hallucinating
🤖 Bot: I created you
🧑 You: quit
```