```
!pip install transformers
```

```
Requirement already satisfied: transformers in
/usr/local/lib/python3.11/dist-packages (4.51.3)
Requirement already satisfied: filelock in
/usr/local/lib/python3.11/dist-packages (from transformers) (3.18.0)
Requirement already satisfied: huggingface-hub<1.0,>=0.30.0 in
/usr/local/lib/python3.11/dist-packages (from transformers) (0.31.2)
Requirement already satisfied: numpy>=1.17 in
/usr/local/lib/python3.11/dist-packages (from transformers) (2.0.2)
Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.11/dist-packages (from transformers) (24.2)
Requirement already satisfied: pyyaml>=5.1 in
/usr/local/lib/python3.11/dist-packages (from transformers) (6.0.2)
Requirement already satisfied: regex!=2019.12.17 in
/usr/local/lib/python3.11/dist-packages (from transformers)
(2024.11.6)
Requirement already satisfied: requests in
/usr/local/lib/python3.11/dist-packages (from transformers) (2.32.3)
Requirement already satisfied: tokenizers<0.22,>=0.21 in
/usr/local/lib/python3.11/dist-packages (from transformers) (0.21.1)
Requirement already satisfied: safetensors>=0.4.3 in
/usr/local/lib/python3.11/dist-packages (from transformers) (0.5.3)
Requirement already satisfied: tqdm>=4.27 in
/usr/local/lib/python3.11/dist-packages (from transformers) (4.67.1)
Requirement already satisfied: fsspec>=2023.5.0 in
/usr/local/lib/python3.11/dist-packages (from huggingface-
hub<1.0,>=0.30.0->transformers) (2025.3.2)
Requirement already satisfied: typing-extensions>=3.7.4.3 in
/usr/local/lib/python3.11/dist-packages (from huggingface-
hub<1.0,>=0.30.0->transformers) (4.13.2)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.11/dist-packages (from requests->transformers)
(3.4.2)
Requirement already satisfied: idna<4,>=2.5 in
/usr/local/lib/python3.11/dist-packages (from requests->transformers)
(3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.11/dist-packages (from requests->transformers)
(2.4.0)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.11/dist-packages (from requests->transformers)
(2025.4.26)
```

```
from transformers import AutoModelForCausalLM, AutoTokenizer
import torch

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

```
model = AutoModelForCausalLM.from_pretrained("microsoft/DialoGPT-
small")
```

```
/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
secret in your Google Colab and restart your session.
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(
```


```
{"model_id":"29e32c0fc5d5445ea7d82697c54f9e21","version_major":2,"vers
ion_minor":0}

{"model_id":"67789fa28ea144d2970a7a3cc9543041","version_major":2,"vers
ion_minor":0}

{"model_id":"1b721e1777f24d8db8c5ce937e944081","version_major":2,"vers
ion_minor":0}

{"model_id":"fda202a3087e4b448581fd37fe334e46","version_major":2,"vers
ion_minor":0}

{"model_id":"ac1249fb56824b14a250df05e9c4ab79","version_major":2,"vers
ion_minor":0}

{"model_id":"7371da9a33d74b9fa6cd04e9cb0d7daa","version_major":2,"vers
ion_minor":0}
```


```
# Chat history
chat_history_ids = None
step = 0

print("ChatBot: Hello! I'm your Gen AI assistant. Type 'quit' to
exit.")

while True:
    # Get user input
    user_input = input("You: ")
    if user_input.lower() == "quit":
        print("ChatBot: Goodbye!")
        break

    # Encode user input
    new_input_ids = tokenizer.encode(user_input + tokenizer.eos_token,
return_tensors='pt')

    # Append to chat history (if not first turn)
```

```
    bot_input_ids = torch.cat([chat_history_ids, new_input_ids], dim=-
1) if step > 0 else new_input_ids

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

    # Decode and print response
    response = tokenizer.decode(chat_history_ids[:,
bot_input_ids.shape[-1]:][0], skip_special_tokens=True)
    print(f"ChatBot: {response}")
    step += 1
```

```
ChatBot: Hello! I'm your Gen AI assistant. Type 'quit' to exit.

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
observe unexpected behavior. Please pass your input's `attention_mask`
to obtain reliable results.

ChatBot: Hi
ChatBot: How are you
ChatBot: I'm sorry
ChatBot: I'm sorry
```