# **Project Report: Building a Simple Conversational AI**

## **Introduction**

This project focuses on developing a simple chatbot using a pre-trained GPT model (GPT-2 to generate text-based responses. The chatbot is implemented as a command-line interface and an optional web-based application using Streamlit.

## **Objectives**

* Implement a chatbot using a pre-trained GPT model.
* Provide a user-friendly interface for interactions.
* Allow customization of response generation parameters.
* Optionally fine-tune the model for improved domain-specific performance.

## **Methodology**

### **1. Environment Setup**

* Installed required libraries: transformers, torch, and streamlit.

### **2. Model Selection and Loading**

* Used GPT-2 for a locally hosted model via the Hugging Face transformers library.

from transformers import GPT2LMHeadModel, GPT2Tokenizer

model = GPT2LMHeadModel.from\_pretrained("gpt2")

tokenizer = GPT2Tokenizer.from\_pretrained("gpt2")

### **3. Chatbot Interface Implementation**

* Created a basic input/output loop for text-based interaction.
* Developed an optional web interface using Streamlit.

**Command-line Interface:**

while True:

user\_input = input("You: ")

if user\_input.lower() == "exit":

break

response = generate\_response(user\_input)

print(f"Bot: {response}")

**Streamlit Web Interface:**

import streamlit as st

from chatbot\_assignment import generate\_response

st.title("GPT Chatbot")

user\_input = st.text\_input("You: ")

if user\_input:

response = generate\_response(user\_input)

st.text\_area("Bot:", value=response, height=200)

### **4. Response Generation**

* Used the GPT model to process user input and generate responses.
* Adjusted parameters such as:  
  + **Temperature**: Controls creativity (lower for focused, higher for diverse responses).
  + **Max tokens**: Limits response length.

### **5. Testing and Iteration**

* Evaluated chatbot responses with various inputs.
* Adjusted parameters for improved coherence and relevance.
* Refined dialogue management based on testing feedback.

## **Challenges and Solutions**

| **Challenge** | **Solution** |
| --- | --- |
| API limitations for GPT-3 | Used GPT-2 for local testing |
| Coherence in long conversations | Tuned model parameters (e.g., context length) |
| Handling ambiguous inputs | Introduced fallback responses |

## **Conclusion**

The project successfully implemented a basic conversational AI using GPT models. The chatbot can generate coherent, context-aware responses and is accessible via both a command-line and web-based interface. Future work could focus on fine-tuning the model for specific domains and improving conversational memory.

## **Future Enhancements**

* Implement memory retention for better conversation flow.
* Fine-tune the model with real-world datasets.
* Expand the chatbot to support multiple languages.

## **References**

* Hugging Face Transformers Documentation