1. Update your `call_llm_azure_openai` function in `llm.py`:

Add support for `stream=True` and yield chunks of the response:

Instead of `call_llm_azure_openai` **use this:**

```
def call_llm_azure_openai_stream(context):
    try:
        client = openai.AzureOpenAI(
            azure_endpoint=azure_openai_endpoint,
            api_key=azure_openai_key,
            api_version=azure_openai_api_version,
        )

        response_stream = client.chat.completions.create(
            model=gpt_model,
            messages=context,
            temperature=int(temperature),
            max_tokens=int(max_token),
            frequency_penalty=int(frequency_penalty),
            presence_penalty=int(presence_penalty),
            top_p=int(top_p),
            seed=int(seed),
            stream=True  # enable streaming
        )

        for chunk in response_stream:
            if chunk.choices[0].delta.content:
                yield chunk.choices[0].delta.content

    except Exception as e:
        print(f"Streaming error: {e}")
        yield "Sorry, I couldn't process your request right now."
```

---

2. Update `app.py` to display the stream in real-time:

Replace this block:

```
with st.spinner("Typing..."):
    with st.chat_message("assistant"):
        ...
        st.markdown(model_response_answer)
```

With streaming logic using `st.write_stream()`:

```
from backend.llm import call_llm_azure_openai_stream  # add this in top lines as an import


with st.spinner("Typing..."):
    with st.chat_message("assistant"):
        full_response = ""

        def stream_response():
            nonlocal full_response
            for chunk in call_llm_azure_openai_stream(st.session_state["context"]):
                full_response += chunk
                yield chunk

        st.write_stream(stream_response)
```

3. Update helper.py

🔁 Previous version :

```
def update_context():
    context = [
        {
            "role": "system",
            "content": """
            You are a Conversational AI assistant helping users answer questions.
            Answer the questions as completely, correctly, and concisely as possible.
            Answer the last question asked by the user.
            Use the previous conversation to provide context for the current conversation only if
required.
            Don't reply to queries comprising offensive, inappropriate, or irrelevant content.
            **VERY IMPORTANT**: Include only content in your response.
            """,
        },
        {"role": "user", "content": "Hello"},
        {"role": "assistant", "content": "Hello, how may I help you today?"}
    ]
    return context
```

✅ Updated version (more conversational & friendly):

```
def update_context():
    context = [
        {
            "role": "system",
            "content": """
            You are a friendly and knowledgeable AI assistant designed to help users with their
questions in a clear, concise, and engaging way. Always be helpful, respectful, and approachable
in tone.

            Use previous conversation context only if it's relevant to the current question.

            If a user greets you or asks something casual, respond warmly. If a user asks a
complex or professional question, provide a helpful and accurate response.

            Avoid sharing any harmful, offensive, or inappropriate content. Keep answers focused
and user-friendly.
            """,
        },
        {"role": "user", "content": "Hello"},
        {"role": "assistant", "content": "Hey there! How can I help you today?"}
    ]
    return context
```