AI-Powered Meeting Summarizer Using Azure OpenAI API

1. Objectives

- Implement Azure OpenAI's GPT Models: Leverage AzureOpenAI's GPT models to process and summarize meeting transcripts.
- Develop a Functional Application: Create an application that ingests meeting transcripts and produces concise summaries.
- Enhance Productivity: Provide users with quick insights from lengthy meetings, aiding efficient information dissemination.

2. Problem Statement

- In today's fast-paced work environment, attending every meeting or reviewing lengthy meeting notes can be challenging. An AI-powered tool that automatically generates concise summaries from meeting transcripts is invaluable.
- This application aims to process raw meeting transcripts and produce clear, concise summaries highlighting key points, decisions, and action items.

3. Inputs / Shared Artifacts

- Azure OpenAI Resource:
  - API key
  - Endpoint URL
  - Deployment name (to be set as environment variables).

4. Expected Outcomes

- Functional Application: Users can upload meeting transcripts and receive summarized versions.
- User Interface: Simple UI (command-line or web-based) for uploading transcripts and displaying summaries.

5. Concepts Covered

- Using GPT models for text summarization.
- Processing large textual inputs (meeting transcripts).
- Designing user interfaces for file input and result display.
- Integrating Azure OpenAI API in applications.
- Writing clear and concise documentation.

Example: Step-by-Step Instructions with Code Snippet

```
from openai import AzureOpenAI
import os

api_version = "2024-07-01-preview"
client = AzureOpenAI(
    api_version=api_version,
    azure_endpoint=os.getenv("AZURE_OPENAI_ENDPOINT"),
    api_key=os.getenv("AZURE_OPENAI_API_KEY"),
)



# Step 2: Load transcript text (example: read from a file)
with open("meeting_transcript.txt", "r") as file:
    transcript = file.read()

# Step 3: Craft prompt for summarization
prompt = f"Summarize the following meeting transcript with key points,
decisions, and action items:\n\n{transcript}"

# Step 4: Call OpenAI ChatCompletion API
response = client.chat.completions.create(
    model="gpt-4o-mini",
    messages=[
        {"role": "system", "content": "You are a helpful assistant
specialized in summarizing meeting notes."},
        {"role": "user", "content": prompt}
    ],
    temperature=0.3,
    max_tokens=500
)

# Step 5: Extract and display summary
summary = response.choices[0].message.content.strip()
print("Meeting Summary:\n")
print(summary)
```

Final Submission Checklist

- Submit your application code with clear comments.
- Include sample meeting transcript files and example outputs. (5 samples)