Project Name: Meeting Minute Generator – AI-Powered Audio to Text Summarizer

Problem Statement:
Teams often spend a lot of time taking manual notes during meetings, which is inefficient and can lead to missed details. This project solves that by automatically converting meeting audio recordings into structured, concise minutes.

Objectives:
- Convert speech from meeting recordings into accurate text.
- Summarize the transcription into short, clear bullet points.
- Deliver results quickly through an API and user interface.

Dataset and Input:
- Input: Audio files in formats like WAV or MP3.
- Typical file length: up to 60 minutes.
- Output: Clear meeting minutes in text form.

Technologies Used:
- Whisper model for high-quality speech-to-text conversion.
- GPT-3 for generating summaries from transcriptions.
- FastAPI for backend REST API development.
- PostgreSQL for storing meeting data.
- AWS S3 for audio file storage.
- Python, Pandas, NumPy for data processing.

System Workflow:

1. User uploads audio via FastAPI.
2. File is stored in AWS S3.
3. Whisper transcribes the audio into text.
4. GPT-3 summarizes the text into concise meeting notes.
5. PostgreSQL stores the summaries.
6. API serves the results back to the user.

Model Choice and Reasoning:
Whisper was chosen for its multi-accent, high-accuracy transcription capabilities. GPT-3 was selected for its ability to produce human-like summaries without extensive fine-tuning. FastAPI was used to ensure a fast and scalable backend.

Performance and Scalability:
- Batched audio processing to reduce latency.
- Asynchronous API requests to avoid delays.
- Cloud storage to enable scaling and reduce local storage load.
- Modular architecture to scale components independently.

Results:
- Transcribed and summarized over 60 meetings during testing.
- Achieved about 90% accuracy in extracting key points compared to human notes.
- Reduced time per meeting summary from 1 hour to under 2 minutes.

Deployment Steps:

- Clone the repository.
- Configure environment variables for OpenAI and AWS.
- Install dependencies with pip.
- Run the API server.
- Use endpoints to upload audio and retrieve summaries.

Future Improvements:

- Add speaker identification to track who said what.
- Fine-tune summarization for specific domains.
- Deploy in containers for large-scale enterprise use.
- Add monitoring and automated scaling in production.