AI Meeting Summarizer & Sharer

Approach, Process, and Tech Stack Documentation

1. Overview

The AI Meeting Summarizer is a Streamlit-based web application that allows users to:

- Upload or paste meeting transcripts.
- Generate concise summaries using an LLM.
- Edit, save, and share summaries via email.
- Manage previously saved summaries (view, delete individually, or delete all).

The goal is to **simplify knowledge sharing** after meetings, while ensuring data persistence and secure integration with external APIs.

2. Approach

The design follows three core principles:

1. **Simplicity in UI/UX**
   - A clean, tab-based interface .
   - Minimal inputs: transcript, instructions, recipients.
   - One-click actions for generation, saving, emailing, and deletion.
2. **Separation of Concerns**
   - LLM interaction handled by a dedicated llm client.
   - Email sending handled by email_utils.
   - Database operations abstracted in database.py.

3. Process

Step 1: Transcript Input

- Users can **upload a .txt file** or paste text directly.
- The input transcript is stored in Streamlit session state.

Step 2: Summary Generation

- The transcript and user-provided instruction are combined into a **prompt**.
- The llm_client.generate_summary() function calls the LLM API (Groq) to produce a structured summary.
- The summary is editable inside the UI before saving.

Step 3: Data Persistence

- Summaries are stored in a SQLite database via SQLAlchemy ORM (database.py).
- Schema includes fields: title, prompt, original_transcript, generated_summary, edited_summary, and created_at.

Step 4: Sharing via Email

- Users enter recipient addresses and subject line.
- Email validation ensures correctness. email_utils.send_summary_email() sends formatted HTML email.

Step 5: Managing Saved Summaries

- Saved summaries are displayed under the **Saved Summaries** tab.
- Each entry is expandable, showing the prompt and generated summary.

4. Tech Stack

Frontend & UI

- Streamlit
- Streamlit Session State

Backend & Business Logic

- Python 3.10+
- LLM Integration: Groq
- Email Sending: email_utils
- Email Validation: email_validator library.

Database & Persistence

- **SQLite**
- **SQLAlchemy ORM**

5. Future Enhancements

- **Search & Filter** saved summaries (by title, keyword, or date).
- **Role-based Access Control** for shared summaries.