PROJECT DOCUMENTATION

1.Title

AUTOMATED MEETING TRANSCRIPTION, TOPIC SEGMENTATION & SUMMARY GENERATION SYSTEM FOR BUSINESS MEETINGS & CORPORATE COMMUNICATION AUTOMATION

---

2. ABSTRACT

This project presents an automated system for analyzing meeting recordings by generating timestamp-aligned transcript segments and concise summaries. The system processes audio and transcript data to clean, segment, and align text with corresponding audio timestamps. It further extracts key discussion topics and produces structured summaries. The final output is stored in a machine-readable format, enabling efficient retrieval and analysis of long meeting recordings while significantly reducing manual review effort.

---

3. APPLICATION

The application is designed to assist users in efficiently reviewing long meeting or podcast recordings. Instead of listening to entire audio files, users receive timestamped transcript segments, extracted keywords, and concise summaries. The modular design allows easy extension and integration into other systems such as dashboards or analytics platforms.

---

4. SYSTEM ARCHITECTURE AND FLOW

System Workflow

Audio Files (audio_raw)

↓

Audio Preprocessing (audio_processed)

↓

Speech-to-Text & Timestamp Extraction

↓

Transcript Cleaning & Segmentation

↓

Keyword Extraction

↓

Summarization

↓

JSON Output (segments)

This flow ensures structured and scalable processing of meeting data.

---

5. PROJECT DIRECTORY STRUCTURE

```
project/
│
├── audio_raw/          # Original audio input files
├── audio_processed/    # Cleaned and normalized audio
├── transcripts/        # Transcript text files
├── segments/           # Timestamped JSON outputs
├── notebooks/          # Experiments and analysis
│
├── src/
│   ├── preprocessing.py       # Audio preprocessing logic
│   ├── transcription.py       # Speech-to-text and timestamps
│   ├── segmentation.py        # Transcript segmentation
│   ├── keyword_extraction.py  # Keyword identification
│   ├── summarization.py       # Summary generation
│   ├── ui_app.py              # User interface layer
│
├── docs/               # Documentation files
├── tests/              # Test cases
├── README.md           # Project overview
├── requirements.txt    # Dependencies
└── LICENSE             # License information
```

---

6. SETUP AND INSTALLATION

Software Requirements

- Python 3.x
- Windows or Linux operating system

Dependency Installation

pip install -r requirements.txt

---

7. CORE METHODOLOGY

This section explains the **end-to-end logic** implemented in the project.

Step 1: Audio Preprocessing

Module: preprocessing.py

- Raw audio files are loaded from the audio_raw directory.
- Noise reduction and normalization techniques are applied.
- Cleaned audio files are saved in audio_processed for accurate transcription.

Step 2: Speech-to-Text and Timestamp Extraction

Module: transcription.py

- Processed audio files are converted into text.
- Start and end timestamps are extracted for spoken segments.
- Generated transcripts are stored in the transcripts directory.

Step 3: Transcript Cleaning

Module: segmentation.py

- Transcript text is cleaned by removing empty lines and unwanted symbols.
- Ensures high-quality input for segmentation.

Step 4: Sentence Tokenization and Segmentation

Module: segmentation.py

- Cleaned transcripts are split into sentences.
- Sentences are grouped into fixed-size segments to represent meaningful discussion blocks.

Step 5: Timestamp Alignment

Module: segmentation.py

- Transcript segments are aligned with audio timestamps.
- Each segment is assigned a corresponding start time from the audio.

Step 6: Keyword Extraction

Module: keyword_extraction.py

- Important keywords are extracted from the transcript.
- Helps identify key discussion topics within the meeting.

Step 7: Summary Generation

Module: summarization.py

- Segmented transcript text is summarized into a concise form.
- The summary captures the main ideas discussed during the meeting.

Step 8: Output Generation

Module: segmentation.py

- All results are stored in structured JSON files:
  - Filename
  - Timestamped segments
  - Extracted keywords
  - Generated summary
- JSON outputs are saved in the segments directory.

---

8. USAGE INSTRUCTIONS

1. Place audio files in audio_raw/.
2. Run preprocessing to generate cleaned audio.
3. Execute the main pipeline script.
4. View timestamped JSON outputs in the segments/ directory.

---

10. RESULTS

- Accurate timestamp-aligned transcript segments

- Effective extraction of discussion keywords
- Concise and meaningful summaries
- Reduced effort in reviewing long recordings

---

11. CONCLUSION

The project successfully demonstrates an automated approach to meeting transcription analysis by integrating audio preprocessing, transcription, segmentation, keyword extraction, and summarization. Its modular design allows easy enhancement and adaptation for real-world applications.
