

1. Introduction

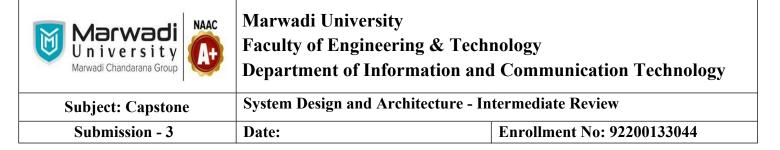
- Purpose: Explain *why* this system exists in simple terms.
 - Example: "The project aims to develop a unified voice analysis system for comparing and managing voice data efficiently."
- Scope: What the system will do.
 - Example: "The system allows users to upload voice data, search and match voices, and compare two audio files for similarity."
- Stakeholders: Identify users who will benefit.
 - Example: "Target users include security personnel, researchers, and audio analysts."

2. Problem Statement

- Describe the problem your system solves.
 - Example points:
 - 1. Manual voice management is time-consuming.
 - 2. Searching for a voice in large datasets is error-prone.
 - 3. Comparing voices manually is impractical for large libraries.
 - 4. Lack of a scalable, reliable solution to store and analyze voice data.

3. Objectives

- Use **bullet points** for clarity:
 - 1. Develop a modular voice management system.
 - 2. Enable uploading and storing voice files securely.
 - 3. Allow searching and matching of voice data efficiently.
 - 4. Provide accurate comparison between two voice samples



4. Relevance to ICT Domain

- Explain why this is important for ICT engineering.
 - o Points:
 - 1. Involves data storage and database management (SQLite/MySQL).
 - 2. Uses **signal processing** (MFCC, DTW, cosine similarity).
 - 3. Implements software engineering best practices (modular design, reusable code).
 - 4. Can integrate with **cloud services** for future scalability.
 - 5. Supports AI/ML applications in audio processing.

5. Feasibility Analysis

- Technical Feasibility:
 - 1. System uses Python, Streamlit, SQLite all open-source and easy to deploy.
 - 2. Audio processing with Librosa and Numpy lightweight and well-documented.

Operational Feasibility:

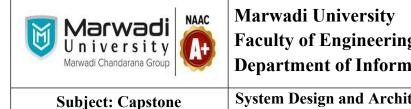
- 1. Users can upload and compare files with simple UI.
- 2. No advanced technical knowledge required to use system.

Economic Feasibility:

- 1. Minimal cost as system relies on free/open-source tools.
- 2. Scalable on cloud infrastructure if needed later.

6. Market / User Needs Analysis

- Identify real-world need for the system:
 - 1. Security agencies need accurate voice identification.
 - 2. Educational institutes and researchers need voice datasets management.
 - 3. Companies need voice authentication systems for secure access.



Faculty of Engineering & Technology
Department of Information and Communication Technology

System Design and Architecture - Intermediate Review

Submission - 3 Enrollment No: 92200133044 Date:

7. Literature Review

- Review existing solutions and why your system improves on them:
 - 1. Traditional manual matching is slow and error-prone.
 - 2. Some AI-based systems require expensive hardware.
 - 3. Open-source projects may lack a user-friendly interface.
 - 4. Your system provides modular, scalable, and cost-effective solution.
- Include citations (IEEE, ACM, or official Python/Librosa docs).

8. System Design / Modular Design

- Overview: Divide system into modules for clarity.
 - **Modules / Components:**
 - 1. Front-End: Streamlit interface for uploading, searching, and comparing audio.
 - 2. **Back-End:** Python scripts handling audio processing and logic.
 - 3. Database Layer: SQLite database to store voice metadata.
 - 4. Audio Processing Module: Extract MFCC features, calculate similarity.
 - 5. **Comparison Engine:** Cosine similarity + DTW for matching.

Benefits of Modularity:

- Easy to maintain and update.
- Components can be reused in other projects.
- New modules (like cloud storage) can be added later.

Include Diagram:

- Use Draw.io / Lucidchart.
- Label each module.
- Show data flow (upload \rightarrow process \rightarrow store \rightarrow compare \rightarrow output).



Marwadi University

Faculty of Engineering & Technology

Department of Information and Communication Technology

Subject: Capstone System Design and Architecture - Intermediate Review

Submission - 3 Date: Enrollment No: 92200133044

9. Technology Stack

• Languages: Python (main logic), SQL (database).

• Frameworks / Libraries:

- 1. Streamlit UI development.
- 2. Librosa Audio feature extraction.
- 3. Numpy / Scipy Calculations.
- 4. SQLite Lightweight database.

• Justification:

- o Open-source, easy to use, lightweight, scalable.
- o Well-documented libraries ensure reliable implementation.

Optional Future Additions:

- Cloud storage (AWS, GCP) for scalability.
- o AI models for voice recognition.

10. Scalability Plan

• **Horizontal scaling:** Use multiple servers if user/data load increases.

Database scaling:

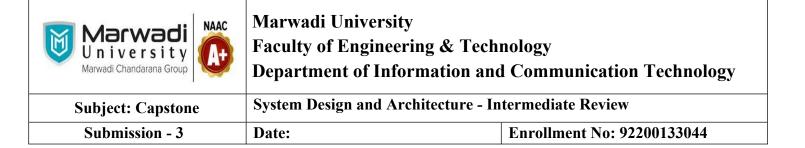
o Use SQLite initially, then migrate to MySQL/PostgreSQL for larger datasets.

• Performance optimization:

- o Cache frequent queries.
- Use batch processing for large audio uploads.

• Reliability:

- Backup database regularly.
- Error handling in audio processing.



11. Conclusion

- Summarize system's technical design and benefits:
 - 1. Modular and scalable design ensures maintainability.
 - 2. Technology stack balances performance and cost.
 - 3. System meets user needs efficiently and reliably.