International Journal of Science, Strategic Management and Technology
Volume **02** Issue **08** August-2026 | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

AI Based Voice Authentication Banking System

Siva Pavin.R & Mayarudhran.S[1], MS.C.S.Selin Chandra[2]

Student, Department of Computer Science and Information Technology .Vels Institute of Science, Technology and Advanced Studies Chennai, India[1]

Assistant Professor, Department of Computer Science and Information Technology, Vels Institute of Science, Technology and Advanced Studies, Chennai, India[2]

https://doi.org/10.55041/ijsmt.v2i8.037

Cite this Article: Pavin.R, S. & Mayarudhran.S, (2026). AI Based Voice Authentication Banking System. International Journal of Science, Strategic Management and Technology, *02*(8), 1-9. https://doi.org/10.55041/ijsmt.v2i8.037

ABSTRACT: This project presents the development of a secure voice authentication system for banking applications using biometric speech recognition technology. The system enables users to perform secure login, registration, and financial transactions using their unique voice patterns instead of traditional passwords. The architecture integrates a frontend developed using Thymeleaf, HTML, CSS, and JavaScript for user interaction and audio recording through microphone access. The backend is built using Spring Boot in Java, which manages REST APIs, authentication logic, user accounts, and banking operations efficiently. A Python-based AI module using Flask processes voice recordings, extracts audio features using Librosa, and converts audio formats using FFmpeg. The extracted voice features are stored in MySQL database and compared during login using Euclidean distance to verify user identity. A threshold-based decision mechanism is used to determine authentication success or failure based on similarity between voice samples. The system also supports secure banking features such as user registration, login, and money transfer using mobile numbers. Overall, the project demonstrates the integration of web technologies, AI-based voice processing, and secure backend systems to build a modern authentication framework for security.

I. INTRODUCTION

In today's digital banking era, security has become a major concern as online transactions and financial services continue to grow rapidly. Traditional password-based authentication methods are increasingly vulnerable to fraud, hacking, and unauthorized access, making stronger biometric solutions essential for banking systems. This project introduces a Voice Authentication System that enables secure user verification using unique voice patterns instead of conventional credentials. The system integrates a web-based frontend, Spring Boot backend, MySQL database, and a Python AI module for voice processing and feature extraction. Users can record their voice for login and transactions, and the system compares extracted voice features using Euclidean distance with a threshold-based decision. This technology enhances banking security, reduces dependency on passwords, and provides a seamless and user-friendly authentication experience for modern applications. Overall, the system demonstrates integration of web technologies, artificial intelligence, and secure banking authentication methods effectively.

II. LITERATURE REVIEW

This project integrates biometric security, signal processing, and full-stack web development to create a secure banking environment. Previous research highlights the effectiveness of voice biometrics as a non-invasive and cost-effective authentication method compared to traditional passwords. The use of Mel-Frequency Cepstral Coefficients (MFCCs) is a well-documented standard for capturing unique vocal tract characteristics. Spring Boot provides a robust framework for managing complex banking transactions, while Python-based libraries like Librosa offer high precision in feature extraction. Overall, this project builds on proven cryptographic and signal-processing methods to create a modern, AI-driven security system.

1. Biometric Authentication in Banking

The use of biometrics in the financial sector allows for enhanced security by replacing or augmenting static passwords with physiological or behavioral traits. Numerous studies support that biometric-enabled systems significantly reduce the risk of identity theft and unauthorized access in digital banking.

2. Spring Boot and Microservices Architecture

The Spring Boot framework is a popular choice for banking systems due to its "convention over configuration" approach and high-level security features. Literature shows that Spring-based setups are ideal for handling RESTful APIs and secure transaction logic, making them suitable for scalable enterprise-level applications.

3. Voice Signal Processing (Librosa and MFCC)

Voice recognition technology relies on extracting unique features from audio signals. Previous projects have demonstrated how the Librosa library can effectively process audio to extract MFCCs, which represent the power spectrum of a sound. This approach is highly effective for distinguishing between different human voices.

4. Euclidean Distance for Pattern Matching

Distance-based algorithms are a standard solution for comparing feature vectors in biometric systems. Studies suggest that Euclidean distance provides a reliable mathematical foundation for determining the similarity between a stored voice template and a live sample, ensuring high accuracy in authentication.

5. Flask-Spring Boot Integration

For AI-driven applications, a polyglot architecture is essential for combining different language strengths. This project uses a Flask-based Python service to handle heavy AI processing while Spring Boot manages the core business logic. Such designs are commonly referenced in modern software engineering for their modularity and efficiency.

6. Integration and Innovation

While each component—voice processing, backend management, and database storage—has been individually studied, the innovation lies in the integration. Combining real-time WebM-to-WAV conversion, automated feature extraction, and banking transaction logic into a unified system aligns with modern trends in FinTech, where security and user experience are the primary goals.

III. METHODOLOGY

1. Frontend Interface and Audio Capture

The user interface is built using Thymeleaf, HTML5, and CSS3 to provide a seamless banking experience. JavaScript's MediaRecorder API is utilized to capture real-time audio through the device microphone. The captured audio is temporarily stored in WebM format and transmitted via an AJAX POST request to the Spring Boot backend.

2. Backend API and Request Handling

The Spring Boot application serves as the primary orchestrator. It hosts RESTful endpoints that manage user registration, login, and financial transactions. Upon receiving the audio data, the backend manages the file flow, ensuring the security of the session before passing the data to the AI processing service.

3. Audio Conversion and Preprocessing

Since the raw audio is captured in WebM format, the system utilizes FFmpeg to convert the files into high-fidelity WAV format. This step is critical because the signal processing library requires a standardized sample rate and bit depth to ensure that feature extraction remains consistent and accurate across different devices.

4. AI-Based Feature Extraction

The Python-based Flask service receives the WAV file and processes it using the Librosa library. The system extracts Mel-Frequency Cepstral Coefficients (MFCCs), which represent the unique spectral characteristics of the user's voice. These numerical feature vectors are then returned to the Spring Boot application as a JSON response.

5. Authentication and Similarity Matching

During registration, the feature vectors are stored in the MySQL database. During login attempts, the system calculates the Euclidean distance between the new live sample and the stored template. If the distance is below a predefined threshold, the user is authenticated; otherwise, access is denied to prevent unauthorized entry.

6. System Integration and Transaction Logic

The final stage involves integrating the authentication result with banking functionalities. Once verified, the user can perform operations such as checking their balance or transferring money via mobile numbers. The MySQL database ensures data persistence for user profiles, transaction history, and voice templates, completing the full-stack ecosystem.

IV. IMPLEMENTATION

The project was implemented using a multi-tier architecture, where a Spring Boot backend serves as the core controller for the banking application. The frontend was constructed using Thymeleaf and JavaScript to facilitate user registration and real-time voice recording via the MediaRecorder API. For the AI processing layer, a Flask-based Python service was

deployed to handle audio analysis. Integration between these two environments was achieved using REST APIs, allowing the Spring Boot server to pass recorded audio to Python and receive numerical feature vectors in return.

To handle audio compatibility, FFmpeg was integrated into the workflow to convert incoming WebM recordings into the WAV format required for analysis. The Python module utilized the Librosa library to perform signal processing, specifically extracting Mel-Frequency Cepstral Coefficients (MFCCs) that represent the user's unique vocal signature. These feature vectors, along with user credentials and account balances, were stored and managed within a MySQL database.

The authentication logic was implemented by calculating the Euclidean distance between the stored voice vector and the vector generated during the login attempt. A specific threshold value was calibrated to balance security and user convenience. Once authenticated, the system allows users to perform banking operations, such as transferring funds. All components were hosted on a local server environment, and rigorous testing was conducted to ensure that the voice matching algorithm correctly identified registered users while rejecting unauthorized access attempts.

V. WORKING PRINCIPLE

The core working principle of this project is based on biometric signal processing and multi-tier architecture to authenticate users through their unique vocal characteristics. It integrates a web-based frontend with a Spring Boot backend and a Python AI service to verify identity before allowing access to financial data.

Step-by-Step Operation:

1. Audio Capture and Preprocessing:

The user interacts with the Thymeleaf-based web interface to record their voice. The browser captures the audio stream, which is sent to the Spring Boot server. Because web browsers typically record in compressed formats, the system uses FFmpeg to transcode the audio into a linear WAV format. This ensures the raw signal is preserved for high-accuracy analysis.

2. Feature Extraction and AI Analysis:

The Spring Boot backend forwards the processed audio to a Flask-based AI module.

The Python service utilizes the Librosa library to perform a Fast Fourier Transform on the audio.

It extracts Mel-Frequency Cepstral Coefficients (MFCCs), converting the complex sound waves into a mathematical feature vector (a set of numbers representing the user's unique vocal tract shape).

These numerical vectors are returned to the main application for comparison.

3. Authentication and Transaction Execution:

The system compares the newly extracted vector with the one stored in the MySQL database during registration. Euclidean Distance Calculation: The system measures the "distance" between the two sets of numbers. If the distance is smaller than a specific threshold, the voices are considered a match.

Access Control: Upon a successful match, the Spring Boot application grants access to the banking dashboard, enabling the user to perform secure money transfers and balance inquiries via REST APIs.

3.1. SPRING BOOT (Java Framework)

Spring Boot is an open-source, Java-based framework used to create microservices and web applications. It serves as the backbone of the banking system, managing all core business logic and security protocols..

Key Features:

Embedded Server: Includes a built-in Tomcat server, eliminating the need for external web server deployment.

Auto-Configuration: Automatically configures dependencies, significantly reducing boilerplate code.

REST API Support: Provides robust tools for building the endpoints that connect the frontend and Python AI module.

Spring Security: Offers a comprehensive security framework to protect banking transactions and user data.

Why It's Used:

Simplifies the development of complex, production-ready backend systems.

Ensures high performance and scalability for financial operations.

Easily integrates with Hibernate and JPA for database management.

3.2. FLASK (Python AI Service)

Flask is a lightweight WSGI web application framework in Python. In this project, it acts as a specialized microservice dedicated to audio processing and artificial intelligence.

Key Features:

Lightweight Architecture: Minimalist design that allows for fast execution of Python scripts.

Extensibility: Easily integrates with scientific libraries like Librosa, NumPy, and SciPy.

Routing: Simple URL routing used to receive audio files and return feature vectors to the main server.

Why It's Used:

Enables the project to leverage Python's superior AI and signal-processing ecosystem.

Provides a clean separation of concerns between banking logic (Java) and AI logic (Python).

3.3. LIBROSA (Audio Analysis Library)

Librosa is a Python package for music and audio analysis. It is the primary tool used in this project to transform raw sound into mathematical data.

Key Features:

Feature Extraction: Capable of extracting MFCCs, chroma features, and spectral centroids.

Time-Series Analysis: Provides tools for audio normalization, resampling, and silence removal. Why It's Used:

Standardizes the voice recognition process with high-precision feature extraction. Allows the system to compare "voice prints" numerically using Euclidean distance. **MYSQL DATABASE**

MySQL is an open-source relational database management system (RDBMS) based on Structured Query Language (SQL). It is used to securely store and manage all persistent data for the banking application.

Key Features:

Data Structure: Uses tables with rows and columns to organize user and financial data.

Security: Offers robust data encryption and user-level access controls. Scalability: Capable of handling large volumes of transaction logs and user profiles.

Integration: Seamlessly connects with Spring Boot using Spring Data JPA.

Why It's Used:

Provides a reliable way to store sensitive information like account balances and passwords.

Ensures data integrity through ACID (Atomicity, Consistency, Isolation, Durability) properties.

Ideal for storing and retrieving high-dimensional voice feature vectors. In This Project:

Stores user profiles (username, mobile number, hashed passwords). Maintains the "voice print" templates (extracted MFCC vectors) for authentication.

Logs all banking transactions and updates account balances in real-time.

FFMPEG (AUDIO CONVERTER)

FFmpeg is a leading multimedia framework used for the transcoding, decoding, and encoding of audio and video files. In this project, it acts as a bridge between the frontend recording format and the backend processing requirements.

1. Audio Transcoding:

Function:

Converts the compressed WebM audio files captured by the web browser into uncompressed WAV files.

2. Key Features:

Format Support: Compatible with almost all audio codecs and containers (MP3, WebM, WAV, etc.). High Fidelity: Preserves the original audio quality during conversion to ensure AI accuracy.

Automation: Can be triggered automatically by the Spring Boot server using system-level commands.

VI. RESULT

The Voice Authentication Banking System successfully demonstrates a secure, biometric-based financial platform by integrating full-stack web development with machine learning. The system effectively utilizes voice recognition as a primary authentication factor, allowing users to register, log in, and perform banking transactions through their unique vocal signatures.

Key Results Achieved:

1. High-Accuracy Voice Authentication

The system successfully differentiates between registered users and unauthorized individuals. By utilizing Mel-Frequency Cepstral Coefficients (MFCC) and Euclidean distance calculations, the authentication module maintains a high success rate with minimal False Acceptance Rates (FAR).

2. Seamless Multi-Language Integration

The integration between the Spring Boot (Java) backend and the Flask (Python) AI service operates efficiently. REST APIs facilitate the rapid transfer of audio data and feature vectors, ensuring that the authentication process is completed in near real-time without significant latency.

3. Reliable Audio Processing Pipeline

The implementation of FFmpeg ensures that various browser-recorded formats are successfully converted into standardized 16kHz WAV files. This allows the Librosa library to extract consistent feature vectors regardless of the user's device or browser.

4. Secure Banking Functionality

The system provides a fully functional banking dashboard where users can view account balances and transfer funds. The MySQL database accurately manages relational data, ensuring that transaction logs are updated immediately upon successful voice verification.

5. Efficient Data Persistence

Voice templates are stored as numerical vectors within the database, rather than raw audio files. This reduces storage overhead and enhances privacy, as the original voice recording is processed and discarded while only the mathematical "voice print" is retained.

6. System Scalability and Security

The modular architecture allows for the future integration of additional security layers, such as OTP (One-Time Password) or Face Recognition. The current setup demonstrates a robust foundation for a "zero-trust" banking environment where identity is verified through behavior rather than just knowledge.

Output

VII. CONCLUSION

The Voice Authentication Banking System successfully demonstrates how the integration of full-stack web frameworks and artificial intelligence can create a secure, next-generation financial platform. By combining Spring Boot, Flask, and Python-based signal processing, the project provides a robust alternative to traditional knowledge-based authentication methods. The system effectively utilizes Mel-Frequency Cepstral Coefficients (MFCCs) to identify users based on their unique vocal characteristics, ensuring that banking transactions are both secure and user-friendly. The project highlights the potential of biometric AI in reducing financial fraud and enhancing the accessibility of digital banking. By automating the feature extraction process through Librosa and FFmpeg, the system achieves a high level of technical accuracy while maintaining a seamless user experience. Despite the complexities of cross-platform integration and audio normalization, the application operates effectively, providing reliable real-time authentication and transaction management. Moreover, the modular architecture of the system—separating the banking logic in Java from the AI processing in Python—ensures easy scalability. Future enhancements could include the addition of noise-cancellation algorithms, multi-factor authentication (MFA), or deep learning models for even greater precision. This project serves as a foundational model for secure biometric systems and offers valuable insights into the convergence of web technologies, database management, and machine learning in the FinTech industry.

REFERENCES

1. Technical Frameworks and Backend

Spring Boot: The primary Java framework used for building the RESTful banking APIs and managing server-side logic. Reference: Spring Framework Documentation (spring.io).

Flask: The Python micro-framework used to host the AI and audio processing service. Reference: Flask Documentation, Pallets Projects.

MySQL: The relational database management system used for storing user credentials and voice feature vectors. Reference: MySQL Official Reference Manual.

2. Audio Processing and AI Libraries

Librosa: The specialized Python library used for audio analysis and extraction of MFCC features. Reference: Librosa: Audio and Music Signal Analysis in Python (librosa.org).

FFmpeg: The multimedia framework used for converting WebM browser recordings into WAV format. Reference: FFmpeg Official Documentation and Command Line Tools.

Euclidean Distance Algorithms: Mathematical foundations used for biometric pattern matching and similarity scoring. Reference: "Pattern Recognition and Machine Learning" by Christopher Bishop.

3. Frontend Technologies

Thymeleaf: The Java template engine used for rendering the dynamic banking web pages. Reference: Thymeleaf Official Documentation.

MediaRecorder API: The browser-based JavaScript API used for capturing microphone input. Reference: MDN Web Docs - MediaStream Recording API.

4. Academic and Research Resources

Voice Biometrics: Research papers regarding Mel-Frequency Cepstral Coefficients (MFCC) and their reliability in speaker recognition.

Example: "Speaker Identification using MFCC and Vector Quantization" (IEEE Xplore).

Full-Stack Development: Books on integrating heterogeneous systems (Java and Python) via RESTful web services. Example: "Hands-On Microservices with Spring Boot and Cloud" by Magnus Larsson.

5. Technical Documentation and Community

GitHub Repositories: Various open-source implementations of signal processing and Spring-Python integration.

Tutorials and Forums: Technical guides from platforms like StackOverflow, GeeksforGeeks, and Baeldung for Spring Boot security and Python API development.

Practical Experimentation: Insights gained through the manual calibration of threshold values for the Euclidean distance matching during the testing phase.