© APR 2026 | IRE Journals | Volume 9 Issue 10 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I10-1716492

Automated Text Note Generator

GIRISH LAWATE[1], YASH PATIL[2], PROF. DEEPTI VIJAY CHANDRAN[3]
[1, 2, 3] *Department of Computer Engineering SIGCE*

Abstract- ***In modern educational environments, students often face difficulty in maintaining accurate and complete notes while simultaneously focusing on lecture comprehension. This challenge becomes more significant in multilingual classrooms, where teachers may switch between languages such as English and Hindi during instruction. Manual note-taking in such scenarios is often incomplete, inconsistent, and cognitively demanding. To address this problem, this paper presents a real-time speech-to- text note generation system designed specifically for classroom environments. The proposed system captures a teacher's speech during live sessions, transcribes it into text in real time, and synchronizes the generated notes across connected student devices. The solution is built using a Flutter-based frontend for cross-platform accessibility, a NestJS backend for session management and synchronization, and the Deepgram API for low-latency multilingual speech recognition. The system supports bilingual speech input, automatic language adaptation, and collaborative note sharing among multiple users within a classroom session. Experimental observations from prototype testing indicate that the system achieves an average transcription accuracy of approximately 80–90% under moderate classroom conditions, with a response latency of 2–5 seconds depending on network quality and background noise. The system reduces the burden of manual note-taking, improves accessibility for students with hearing difficulties, and contributes to more inclusive and efficient learning environments. The work demonstrates the practical application of natural language processing and real-time communication technologies in educational technology solutions.***

Index Terms—Speech-to-Text, Natural Language Processing, Real-Time Systems, Multilingual Recognition, Deepgram API, Classroom Automation, Educational Technology

I. INTRODUCTION

The process of note-taking plays an essential role in the learning process. Students often rely on lecture notes for revision, concept reinforcement, and examination preparation. However, in traditional classroom settings, note-taking is a manual and attention-intensive task. Students must divide their focus between listening, understanding, and writing, which often leads to missed content and incomplete notes.

The challenge becomes even more significant in real-time classroom environments where lectures are delivered at a fast pace and where teachers naturally switch between multiple languages. In Indian classrooms, for example, bilingual de- livery involving English and Hindi is common. While this improves teacher-student interaction, it creates difficulties for standard monolingual note-generation systems and for students who may not be equally comfortable in both languages.

Recent advances in speech recognition, natural language processing, and cloud-based APIs have enabled the development of systems capable of converting spoken language into text with high speed and improved accuracy. These technologies create an opportunity to design intelligent classroom tools that can capture lectures in real time and transform them into synchronized notes accessible on student devices. This paper proposes a Speech-to-Text Note Generator for Real-Time Multilingual Classroom Environments. The system is intended to support teachers during live lectures by continuously converting speech into textual notes and instantly distributing them to students participating in the same session. The objective is not merely transcription, but the creation of a collaborative academic support tool that improves learning quality, accessibility, and digital classroom interaction.

A. Problem Statement

In traditional classroom settings, students frequently miss important lecture content due to the following reasons:

- Fast-paced speech delivery by instructors
- Difficulty in understanding bilingual or multilingual lectures
- Distractions and reduced concentration during long sessions

- Varying note-taking speed and skill among students
- Lack of accessible support for students with hearing or learning difficulties

As a result, there is a need for a system that can automatically capture classroom speech, transcribe it in real time, and make the generated notes instantly available to all students.

B. Objectives

The major objectives of the proposed system are:

- To develop a real-time speech-to-text system for class- room lectures
- To support multilingual or bilingual speech recognition, especially English and Hindi
- To ensure low latency and acceptable transcription accuracy
- To enable synchronized note sharing among all students in a session
- To reduce the burden of manual note-taking
- To improve classroom accessibility and inclusion

C. Scope of the Work

The system is designed for classroom-oriented lecture tran- scription and note sharing. It supports live sessions initiated by a teacher and joined by students through a mobile or web- compatible interface. The present scope focuses on:

- Real-time lecture transcription
- Classroom note synchronization
- Bilingual classroom speech handling
- Cross-platform frontend access

Features such as lecture summarization, advanced semantic search, speaker diarization, and offline processing are considered for future enhancement.

II. RELATED WORK

Speech recognition systems have gained substantial attention over the past decade due to improvements in deep learning, cloud computing, and real-time streaming technologies. Major cloud providers such as Google, Microsoft, and Deep- gram offer APIs for speech-to-text conversion, each optimized for different application scenarios.

Google Speech-to-Text provides robust multilingual sup- port and integration with cloud-based applications. Microsoft Azure Speech Services offers enterprise-grade transcription and speaker analytics features. However, these systems are generally designed for broader use cases such as voice assistants, call-center processing, and media transcription. They do not directly address the classroom requirement of synchronized note sharing among students in a live lecture environment.

Deepgram has emerged as a strong alternative for real-time transcription due to its low-latency streaming support, scalable API architecture, and support for multiple languages. Its ability to process continuous speech streams makes it suitable for classroom environments, where speech is dynamic and often includes domain-specific terms.

Existing research in educational technology has also explored automatic lecture capture systems, note assistance tools, and audio-based learning support. Many of these solutions focus on recording lectures for later review rather than generating live synchronized notes. Others lack support for dynamic language switching in multilingual classrooms. Thus, there remains a gap for a practical, classroom-centric, multilingual, real-time transcription platform.

A. Comparison with Existing Systems

Table I presents a basic comparison of existing solutions with the proposed system.

TABLE I
COMPARISON OF EXISTING SOLUTIONS WITH PROPOSED SYSTEM

Feature	Google STT	Azure Speech	Proposed Sys- tem
Real-time transcription	Yes	Yes	Yes
Multilingual support	Yes	Yes	Yes
Classroom note synchronization	No	No	Yes
Teacher-student live session model	No	No	Yes
Focused on educa-	Limited	Limited	Yes

© APR 2026 | IRE Journals | Volume 9 Issue 10 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I10-1716492

tional workflow			
Collaborative note sharing	No	No	Yes

III. SYSTEM ARCHITECTURE

The proposed system follows a three-layer architecture consisting of the frontend layer, backend layer, and speech- processing layer. The layered design ensures modularity, main- tainability, and scalability.

A. Frontend Layer

The frontend is developed using Flutter, which enables cross-platform deployment on Android, iOS, and potentially web platforms. The frontend provides separate interfaces for teachers and students.

Teacher-side features:

- Start and end live classroom sessions
- Monitor transcription in real time
- Share session code with students

Student-side features:

- Join classroom session using code or identifier
- View notes as they are generated
- Access synchronized lecture transcript on their device

B. Backend Layer

The backend is built using NestJS, a scalable Node.js framework suited for structured API and real-time service development. The backend performs the following operations:

- User authentication and session control
- Teacher-student session mapping
- Reception and forwarding of speech stream metadata
- Real-time note synchronization using WebSocket or socket-based communication
- Data storage and management for session records

C. Speech Processing Layer

The speech-processing layer uses the Deepgram API for streaming speech recognition. Audio captured from the teacher's device is streamed in real time, processed by Deep- gram, and converted into text. The generated text is returned in chunks or partial transcripts, which are then sent to the backend and finally displayed to all students in the session.

D. Architecture Diagram

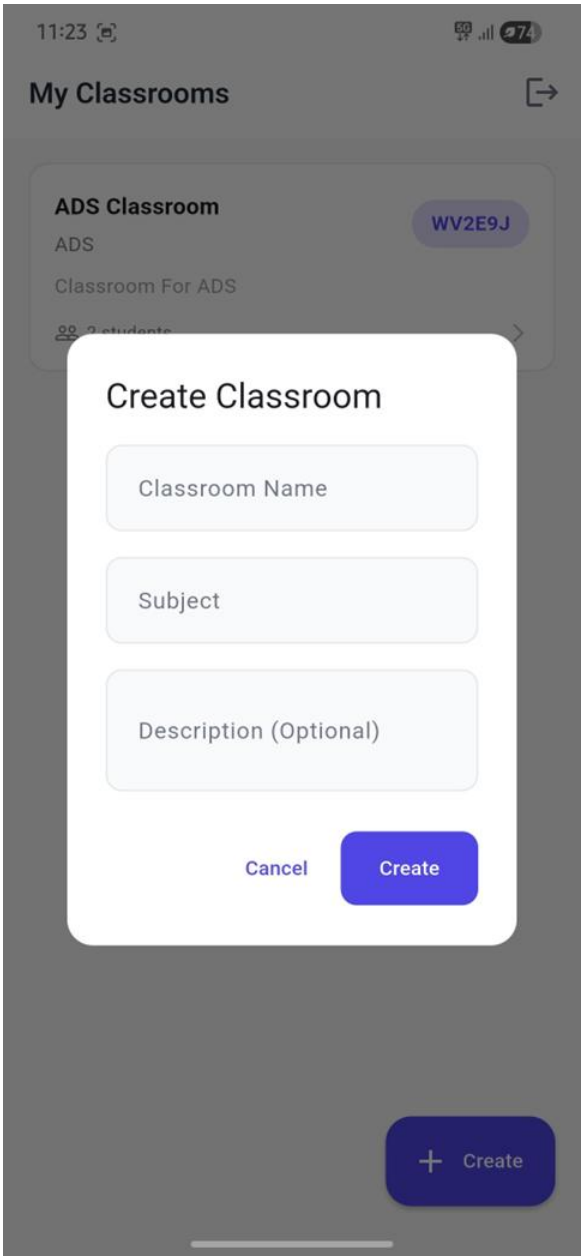

Fig. 1. Overall System Architecture

E. System Flow

The overall flow of the system can be summarized as follows:

1) The teacher logs in and creates a lecture session.

2) Students join the same session using the generated session identifier.
3) The teacher's speech is captured through the device microphone.
4) Audio is streamed to the backend and forwarded to the Deepgram API.
5) Deepgram converts speech into text in real time.
6) The backend receives the transcribed text and broadcasts it to connected student devices.
7) Students view continuously updated lecture notes during the class.

IV. METHODOLOGY

The methodology of the proposed work is centered around live audio capture, real-time streaming, speech recognition, text synchronization, and note display.

A. Workflow of the Proposed System

The working process of the system consists of the following stages:

1) Session Initialization: The teacher initiates a classroom session through the frontend application.
2) Audio Acquisition: The application captures the teacher's speech through the microphone.
3) Streaming and Preprocessing: Captured audio is pack- etized and transmitted for processing.
4) Speech Recognition: The speech-processing API con- verts spoken language into textual output.
5) Synchronization: The backend distributes the generated text to all active student devices.
6) Storage and Access: Notes may be stored for later access and review.

B. Processing Pipeline

The processing pipeline of the system can be represented mathematically as:

$S(t) \rightarrow A(t) \rightarrow P(A) \rightarrow T(A) \rightarrow D(T)$

where:

- S(t) denotes the spoken lecture input over time,
- A(t) represents captured audio data,
- P (A) refers to preprocessing and stream preparation,
- T (A) denotes the transcription output,
- D(T) is the displayed text synchronized to student de- vices.

A simplified pipeline is shown below:

Speech Input → Audio Stream → Deepgram API → Text Output → Fron

C. Functional Modules

The proposed system can be divided into the following modules:

1) Authentication Module: This module handles user regis- tration and login for both teachers and students. It ensures that only authenticated users can create or join classroom sessions.
2) Session Management Module: This module creates unique session identifiers, manages active lecture rooms, tracks joined students, and closes sessions once the lecture ends.
3) Audio Streaming Module: This module captures contin- uous audio from the teacher's microphone and transmits it in a suitable format for real-time processing.
4) Transcription Module: This module interacts with the speech recognition API to obtain live text from streaming audio.
5) Synchronization Module: This module ensures that all students receive updated notes with minimal delay. It is responsible for real-time broadcasting and efficient frontend updates.

V. IMPLEMENTATION DETAILS

A. Frontend Implementation Using Flutter

Flutter was selected as the frontend framework due to its cross-platform support, rich UI capabilities, and reactive widget-based architecture. The frontend contains:

- Login and registration screens
- Session creation and join screens
- Live transcript display panel
- Session status indicators

The use of Flutter also ensures that the same codebase can be adapted for multiple platforms, reducing development effort.

B. Backend Implementation Using NestJS

NestJS provides a modular backend structure with support for REST APIs and real-time communication patterns. Its controller-service architecture simplifies management of:

- User endpoints

- Session endpoints
- Real-time event handling
- Integration with third-party speech APIs

Socket-based communication is used for real-time delivery of notes from the backend to all connected student devices.

C. Speech Recognition Integration

Deepgram API is used in streaming mode to handle real- time speech recognition. The API receives the audio stream and responds with text fragments, which may include inter- mediate and finalized transcription outputs. These outputs are processed and displayed in the user interface.

D. Database and Storage Considerations

Although the primary goal of the system is live note genera- tion, storage support is important for future reference. Session details, teacher identity, timestamps, and generated notes can be stored in a database for later retrieval. A lightweight rela- tional or NoSQL storage system can be integrated depending on system scale.

VI. SPEECH PROCESSING IN MULTILINGUAL CLASSROOM ENVIRONMENTS

One of the most important aspects of the system is its ability to operate in multilingual environments. In many class- rooms, teachers naturally alternate between English and Hindi, depending on the topic, student understanding, and teaching style.

A. Audio Capture

Audio is captured continuously using the device micro- phone. Since classroom speech is ongoing and dynamic, the system must handle a continuous stream rather than isolated commands.

B. Noise Handling

Classroom environments are often affected by background disturbances such as student conversations, fan noise, corridor sounds, and microphone variations. To improve recognition quality, basic preprocessing and noise reduction strategies are used before sending audio data to the speech engine. Although advanced denoising is not fully implemented in the current version, the system attempts to maintain stable transcription under moderate background noise.

C. Language Detection and Switching

A major challenge in multilingual speech recognition is handling language switching within the same sentence or lecture. The proposed system supports Hindi and English input by relying on the multilingual capabilities of the selected speech- recognition engine. This allows the system to adapt to bilingual teaching patterns and generate notes without requiring the user to manually select the language for every utterance.

D. Challenges in Multilingual Speech Recognition

Despite recent improvements in cloud-based speech recognition, multilingual classroom scenarios remain difficult due to:

- Code-switching between languages
- Different accents and pronunciation styles
- Technical terminology mixed with regional language
- Environmental noise and microphone inconsistency These challenges affect transcription accuracy and must be addressed in future system enhancements.

VII. EXPERIMENTAL RESULTS AND PERFORMANCE EVALUATION

To evaluate the practical performance of the system, a prototype implementation was tested under classroom-like conditions. The evaluation focused on two important factors: transcription accuracy and response latency.

A. Evaluation Parameters

The following parameters were considered:

- Accuracy of recognized text
- Delay between spoken words and displayed text
- Behavior under bilingual lecture input
- Performance under mild background noise

B. Observed Results

During testing, the system demonstrated the following ap- proximate performance:

- Transcription Accuracy: 80–90%
- Response Latency: 2–5 seconds
- Bilingual Handling: Effective in moderate English-Hindi switching scenarios

© APR 2026 | IRE Journals | Volume 9 Issue 10 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I10-1716492

C. Performance Table

TABLE II
PROTOTYPE PERFORMANCE OBSERVATIONS

Metric	Observed Range
Transcription Accuracy	80–90%
Average Latency	2–5 seconds
Performance in Quiet Environment	High
Performance in Noisy Environment	Moderate
Multilingual Speech Handling	Effective under normal switching

D. Discussion of Results

The results show that the proposed system is suitable for practical classroom use, especially in controlled or moderately noisy environments. The latency is low enough to support live note synchronization without causing major disruption to the classroom experience. While the transcription accuracy is not perfect, it is sufficient to provide meaningful assistance to students and significantly reduce the amount of manual note-taking required.

However, as background noise increases, recognition quality decreases. Similarly, heavy code-switching and unclear pronunciation may reduce text quality. These limitations indicate that further model tuning, noise suppression, and domain adaptation would improve performance.

VIII. APPLICATIONS OF THE PROPOSED SYSTEM

A. Classroom Learning

The primary application of the system is in educational institutions where teachers conduct live lectures. Students can use the generated notes for real-time understanding and later revision.

B. Accessibility Support

The system is highly useful for students with hearing impairments or auditory processing difficulties. Live textual display improves accessibility and inclusiveness in the class- room.

C. Hybrid and Online Education

In blended or hybrid learning scenarios, the same system can be used to support remote learners by making lecture text available in real time.

D. Lecture Archiving

Generated notes can be stored for future use, creating a searchable archive of classroom sessions that benefits both teachers and students.

E. Collaborative Learning

The synchronized note-sharing model supports collaborative learning, as all students receive a common lecture record, reducing dependency on individual note quality.

IX. LIMITATIONS

Although the system demonstrates promising performance, it has several limitations:

- It is sensitive to high levels of background noise.
- It requires a stable internet connection for real-time API communication.
- The current implementation supports only limited multi- lingual capability.
- It does not yet include summarization, topic extraction, or keyword highlighting.
- Recognition accuracy may drop for domain-specific ter- minology or strong accents.

These limitations are common in real-time speech systems and provide direction for future development.

X. FUTURE WORK

Several enhancements can improve the usefulness and robustness of the proposed system:

- Automatic Summarization: Integrating NLP-based summarization to generate concise lecture summaries.
- Keyword Extraction: Highlighting important terms, definitions, and concepts.
- Offline Mode: Providing local transcription support for environments with poor internet connectivity.
- Additional Language Support: Extending the system to regional languages beyond Hindi and English.

- LMS Integration: Connecting the system with learning management platforms such as Google Classroom or Moodle.
- Analytics Dashboard: Giving teachers access to session history, participation data, and note usage insights.
- Improved Noise Suppression: Enhancing classroom robustness using advanced preprocessing or AI-based denoising.

XI. CONCLUSION

This paper presented a real-time speech-to-text note generator designed for multilingual classroom environments. The system addresses an important educational challenge by reducing the burden of manual note-taking and improving lecture accessibility for students. By combining a Flutter frontend, NestJS backend, and Deepgram-based speech recognition pipeline, the proposed solution enables real-time transcription and synchronized note sharing during live classes.

The prototype results demonstrate that the system can achieve practical transcription quality with acceptable latency, making it suitable for educational use in real-world scenarios. The inclusion of bilingual classroom support increases its relevance in multilingual teaching contexts. Although the system has limitations related to noise, connectivity, and lan- guage complexity, it establishes a strong foundation for future educational technology tools that combine natural language processing, cloud services, and collaborative interfaces.

Overall, the proposed work demonstrates how speech recognition technology can be effectively adapted for classroom assistance, accessibility improvement, and digital transformation in education.

ACKNOWLEDGMENT

We would like to express our sincere gratitude to our institution, faculty members, and mentors for their continuous guidance, encouragement, and support throughout the development of this project. Their valuable suggestions and academic direction played an important role in the successful completion of this work.

REFERENCES

[1] Deepgram, “Speech-to-Text API Documentation,” Available: https:// developers.deepgram.com/. Accessed: Apr. 2026.

[2] D. Jurafsky and J. H. Martin, Speech and Language Processing, 3rd ed. draft, Stanford University, 2023.

[3] Google Cloud, “Speech-to-Text Documentation,” Available: https:// cloud.google.com/speech-to-text/docs. Accessed: Apr. 2026.

[4] Microsoft Azure, “Azure AI Speech Documentation,” Available: https:// learn.microsoft.com/en-us/azure/ai-services/speech-service/. Accessed: Apr. 2026.

[5] NestJS, “NestJS Official Documentation,” Available: https://docs.nestjs. com/. Accessed: Apr. 2026.

[6] Flutter, “Flutter Official Documentation,” Available: https://docs.flutter. dev/. Accessed: Apr. 2026.

[7] A. Graves, A.-r. Mohamed, and G. Hinton, “Speech Recognition with Deep Recurrent Neural Networks,” in Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP), 2013, pp. 6645–6649.

[8] T. N. Sainath and C. Parada, “Convolutional Neural Networks for Small- Footprint Keyword Spotting,” in Proc. Interspeech, 2015, pp. 1478–1482.