Vol. 2, No. 2, September 2025

IJAIS International Journal of Artificial Intelligence and Science

Published By Asosiasi Doktor Sistem Informasi Indonesia

Speech-Driven Visitor Notification System Using Telegram Bot and Voice Activity Detection for Real-Time Retail Applications

Aria Setiaji[1], Aria Hendrawan[2], Bernadus Very Christioko[3], Lenny Margaretta Huizen[4]

[1,2,3,4]Department of Information Technology and Communication, Universitas Semarang, Indonesia

Email: ariasetiaji34@gmail.com[1], ariahendrawan@usm.ac.id[2], very@usm.ac.id[3], lenny@usm.ac.id[4]

Received: July 27, 2025

Revised: August 15, 2025

Accepted: Sept. 17, 2025

Published: Sept. 30, 2025

Corresponding Author:

Author Name*:

Aria Hendrawan

Email*:

ariahendrawan@usm.ac.id

DOI: 10.63158/IJAIS.v2i2.44

Abstract. In the retail industry, fast and responsive service is essential for maintaining customer satisfaction and loyalty. A key challenge faced by store owners is delayed responses to customer arrivals, leading to dissatisfaction and potential lost sales. This project develops an automatic notification system using Speech-to-Text technology and a Telegram bot to detect voice keywords and send real-time notifications to store owners. The system was developed using the prototype methodology, allowing for iterative testing and refinement to ensure it met user needs and functional requirements. It integrates the Deepgram API for accurate speech transcription, the Telegram Bot API for notifications, and a web interface for managing keywords and monitoring system status. To enhance efficiency, a Voice Activity Detection (VAD) module was added, ensuring that only human speech is processed, thereby reducing unnecessary processing. Experimental results showed that the system achieved 100% accuracy in quiet environments and 80% in noisy conditions. The system's response time was also impressive, with an average time of 3.72 seconds in quiet conditions and 3.8 seconds in noisy environments. Word Error Rate (WER) and Character Error Rate (CER) evaluations indicated perfect accuracy in quiet conditions (WER 0%, CER 0%) and slight errors in noisy conditions (WER 13.33%, CER 12.5%). Overall, the system effectively improved service speed and responsiveness, offering store owners a valuable tool for enhancing customer experience in retail environments.

Keywords: Speech-to-Text, Telegram Bot, Voice Activity Detection, Retail Automation, Real-Time Notifications

IJAIS International Journal of Artificial Intelligence and Science
Publsihed By Asosiasi Doktor Sistem Informasi Indonesia

1. INTRODUCTION

Service quality plays a critical role in maintaining customer satisfaction, loyalty, and profitability in the retail industry. Fast, accurate, and responsive service directly influences customers' perceptions and purchase intentions [1]. When customers receive timely attention, they are more likely to return, while poor responsiveness can result in dissatisfaction and reduced retention rates. Moreover, environmental factors such as store layout, product availability, and human interaction also contribute to perceived service quality [2].

A survey conducted among 30 visitors at Barokah Store revealed that 43.3% of respondents experienced visiting the store when no attendant was available, and 40% admitted leaving without purchasing due to delayed service. These findings highlight that response delays can significantly impact customer loss and revenue decline.

With advancements in information technology, automation has become a feasible approach to improving retail operations. Prior studies have utilized ultrasonic sensors to detect customer arrivals and notify attendants [3]. However, such sensor-based systems are limited by their placement flexibility and susceptibility to environmental interference. In contrast, speech recognition technology offers a more natural and intuitive solution without the need for additional hardware, as successfully demonstrated in home automation and control systems [4]. Nevertheless, challenges such as background noise and accent variations persist, necessitating robust noise suppression algorithms [5].

Telegram bots have gained popularity for real-time automation and notification systems due to their simplicity and reliability [6]. Previous implementations have utilized Telegram for IoT-based security systems [7] and automated watering systems [8], while speech recognition has proven effective for automated voice control applications [9]. However, integration of both technologies—speech recognition and Telegram notification—within a retail context has not been extensively explored.

This research bridges that gap by designing an automatic visitor notification system that employs speech recognition to detect predefined voice keywords and transmit notifications through Telegram. The system also includes a web dashboard for managing

keywords and monitoring operational status, thereby improving service efficiency and enhancing customer experience in retail environments.

2. METHODS

2.1 Development method

The system was developed using the prototype development methodology, a widely adopted iterative software development approach that emphasizes the early creation of a working model (prototype) for user evaluation. This approach allows for continuous user feedback and iterative refinements, leading to the development of a system that better meets end-user requirements and expectations [10]. By enabling direct interaction with early versions of the system, the prototype model helps bridge communication gaps between developers and stakeholders, particularly useful in systems where user experience and real-time interaction play a pivotal role. According to established research [11], the prototyping model typically comprises the following five core phases, each of which was implemented in this study:

1) Initial Requirement Identification: In the first phase, preliminary system requirements were gathered through informal interviews and observation sessions with store staff and potential users. The main focus was to determine key system functionalities, including real-time keyword detection through voice input, timely notification delivery to store personnel, and usability within a physical retail environment. Requirements such as offline readiness, latency constraints, and privacy concerns were also noted during this stage.
2) Rapid Design: Based on the gathered requirements, a preliminary system design was constructed. This included designing an intuitive user interface for the web dashboard and mapping the basic workflow of the system. The focus was on creating a minimal yet functional prototype that could demonstrate the essential interaction flows between speech recognition, keyword filtering, and messaging components.
3) Prototype Construction: A functional prototype of the system was developed, integrating key components such as the Deepgram Speech-to-Text API for audio transcription and the Telegram Bot API for delivering notifications. The backend was developed using Python and Flask, ensuring modularity and ease of future enhancements. This initial version provided a basic, end-to-end working system

capable of recording audio, transcribing it, detecting specific keywords, and issuing real-time notifications.

4) User Evaluation: The prototype was deployed and tested within an operational retail store to evaluate its performance under real-world conditions. Store staff interacted with the system over multiple days, and both quantitative data (response time, accuracy of keyword detection) and qualitative feedback (user satisfaction, perceived usefulness) were collected. This phase highlighted critical issues such as background noise interference, false positives, and the need for keyword customization.

5) Refinement and Enhancement: Based on the insights gained during the evaluation phase, the system underwent several iterations of improvement. Enhancements included noise filtering, expanded keyword customization options, optimization of notification latency, and improvements in the dashboard's UI for better usability. These refinements continued until the prototype evolved into a stable and fully functional final system.

2.2 System Flow

The operational flow of the notification system is depicted in Figure 1. The process begins with the initialization of the system, wherein the audio capture module is activated. The system continuously records environmental audio using a connected microphone. Once audio data is captured, it is transmitted to the Deepgram API, which performs real-time transcription using advanced machine learning models optimized for speech recognition.

After transcription, the system parses the resulting text to detect the presence of any predefined target keywords, which are stored in the system's backend database. These keywords are typically phrases such as "excuse me", "hello", or similar utterances frequently used by customers to initiate interaction with store staff. If a match is found between the transcribed text and the keyword list, the system triggers a notification via the Telegram messaging platform, alerting store personnel instantly.

This cycle—audio capture → transcription → keyword matching → notification delivery—repeats continuously, ensuring real-time responsiveness. The system is designed to run autonomously, with minimal user intervention, and can be paused or reconfigured through a dedicated web-based dashboard.

IJAIS International Journal of Artificial Intelligence and Science
Publsihed By Asosiasi Doktor Sistem Informasi Indonesia

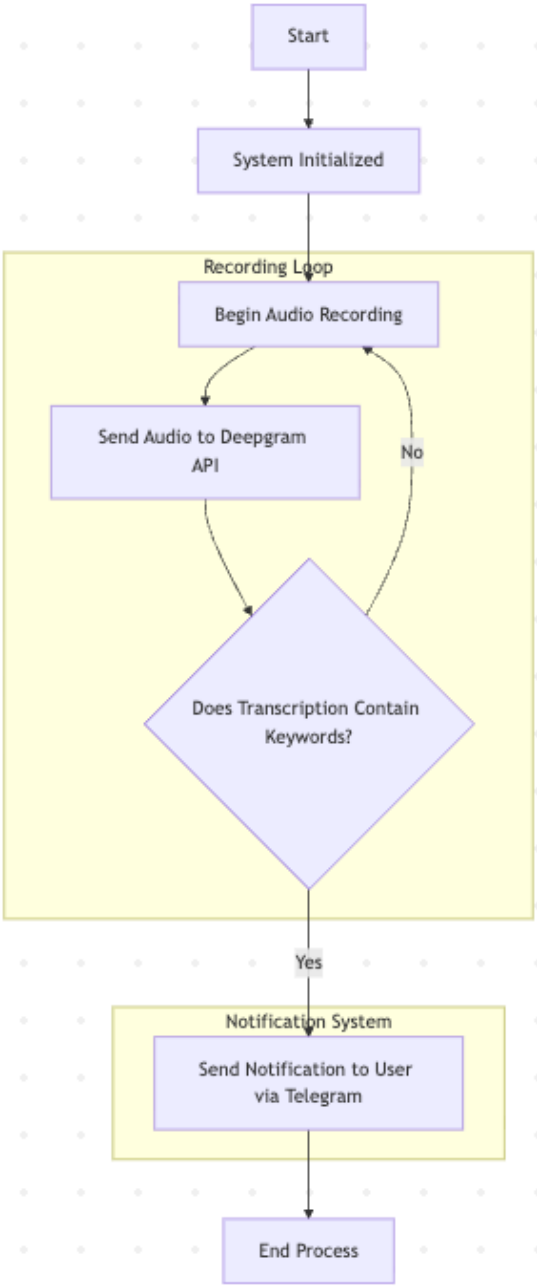

Figure 1. System Flowchart

2.3 System Functional

The system's primary functionalities and interactions are outlined using a Use Case Diagram (Figure 2). This visual representation illustrates the key roles and actions within the system, primarily focusing on the store owner (actor) and the automated notification system. The main use cases include:

1) Start and Stop Recording: The user can manually start or stop the recording process through the dashboard.

IJAIS International Journal of Artificial Intelligence and Science
Published By Asosiasi Doktor Sistem Informasi Indonesia

2) Real-Time Transcription: The system listens and sends captured audio to Deepgram's API for speech-to-text processing.
3) Keyword Detection: After transcription, the system filters for predefined keywords.
4) Notification Dispatch: Upon keyword detection, the system sends a notification to the store owner or assigned personnel via Telegram.
5) View System Status and Logs: The web dashboard displays current system status, notification logs, and detected keywords.

This modular interaction ensures simplicity and clarity in user operations while providing transparency in system functionality.

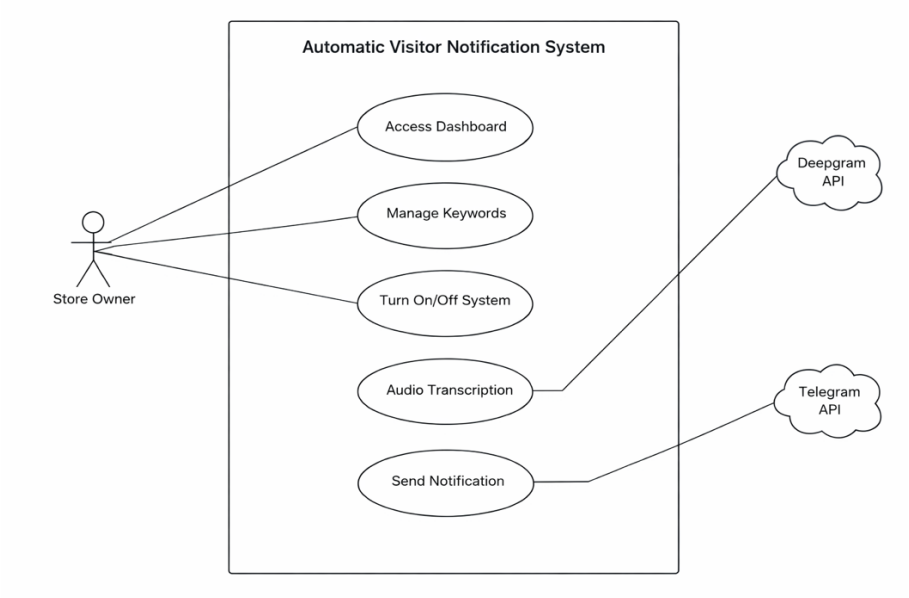

Figure 2. Use Case Diagram

2.4 System Architecture

The system's internal structure is designed based on the Model-View-Controller (MVC) architecture, which promotes separation of concerns and improves maintainability and scalability. As shown in Figure 3, the architecture is further enhanced by the inclusion of a Service Helper module that abstracts communication with external APIs and services.

1) Model: Responsible for managing application data, such as the list of keywords, user preferences, and system logs. It also handles database interactions to persistently store these elements.

IJAIS International Journal of Artificial Intelligence and Science
Publsihed By Asosiasi Doktor Sistem Informasi Indonesia

2) View: Provides the graphical interface (web dashboard) accessed by store owners or staff. It enables users to monitor system activity, view detected keywords, modify settings, and manually control the system.
3) Controller: Serves as the central logic unit that processes incoming requests from the view, interacts with the model to fetch or update data, and initiates actions like recording or sending notifications.
4) Service Helper: Encapsulates external service integrations including:
    a) Audio Recording Module: Captures real-time audio using device microphones.
    b) Deepgram API Client: Handles audio streaming and receives transcription data.
    c) Telegram Bot API Client: Formats and sends notification messages based on keyword detection events.

This architectural approach ensures the system is loosely coupled, making it easier to update individual components without affecting others, and supports future enhancements such as multi-language support or alternate notification channels.

2.5 System Requirements

The successful development, deployment, and operation of the proposed speech-recognition-based notification system depend on a well-balanced combination of hardware and software components. Each component is carefully selected based on its compatibility, performance, scalability, and ease of integration within a real-time, user-interactive environment. These requirements are categorized into hardware and software specifications as follows:

2.5.1. Hardware Requirements

To ensure smooth performance and real-time responsiveness, the system requires the following hardware elements:

1) Laptop or Desktop Computer

This serves as the central processing unit for the system. It hosts the web-based dashboard, executes the backend services, handles real-time audio data processing, and manages communication with external APIs such as Deepgram and Telegram. The hardware should have at least a dual-core processor, 8 GB RAM, and stable internet

IJAIS International Journal of Artificial Intelligence and Science
Publsihed By Asosiasi Doktor Sistem Informasi Indonesia

connectivity to ensure optimal performance during concurrent tasks such as transcription and data logging.

2) Microphone

A high-quality, omnidirectional microphone is essential to accurately capture audio input from the environment. Since the system operates in potentially noisy retail spaces, the microphone should offer basic noise-canceling capabilities or be positioned optimally to reduce interference. The clarity of captured audio directly affects transcription accuracy and overall system reliability.

3) Smartphone

A modern smartphone is required to receive real-time notifications through the Telegram application. Additionally, the phone may serve as a remote interface to access the system dashboard for monitoring alerts and updating configurations. For convenience, the Telegram app should be pre-installed, and push notifications must be enabled.

2.5.2. Software Requirements

The software stack of the system is designed with a focus on flexibility, modularity, and scalability. It is built using open-source and cloud-based technologies to maintain a cost-effective yet powerful architecture.

1) Programming Language – Python

Python is chosen as the primary development language due to its extensive support for audio processing, API integrations, and rapid application development. Libraries such as pyaudio, wave, and threading facilitate real-time audio capture, while its syntax promotes developer productivity and maintainability.

2) Web Framework – Flask

Flask is utilized to create a lightweight web server and RESTful API for system interaction. It enables rapid development of the web-based dashboard and supports the integration

IJAIS International Journal of Artificial Intelligence and Science
Publsihed By Asosiasi Doktor Sistem Informasi Indonesia

of external services like Deepgram and Telegram through simple HTTP interfaces. Flask's modular structure also supports future feature expansions with minimal rework.

3) Speech Recognition Service – Deepgram API

The system uses the Deepgram API for its high-performance speech-to-text transcription capabilities. Deepgram is selected based on its support for real-time processing, advanced acoustic models, speaker diarization, and language-specific noise filtering. These features are critical in ensuring accurate keyword detection in dynamic and potentially noisy retail environments.

4) Messaging Platform – Telegram Bot API

To deliver notifications instantly and securely, the system integrates the Telegram Bot API. This platform offers encrypted messaging, custom bot development, and flexible configuration, allowing the system to send real-time alerts to designated users or groups with minimal delay.

5) Supporting Libraries and Dependencies

Additional Python libraries such as: 1) requests: Facilitates HTTP communication with external APIs. 2) Flask-CORS: Manages cross-origin resource sharing for web interfaces. 3) dotenv: For secure management of environment variables (API keys, tokens). 4) threading and multiprocessing: To handle concurrent recording and processing tasks. These libraries collectively support asynchronous operations, secure API integration, and smooth data handling across system components.

3. RESULTS AND DISCUSSION

3.1 System Implementation

The automatic notification system was developed modularly using the Python programming language and Flask framework. The system architecture consists of three main components: Audio transcription using Deepgram API, Notification delivery via Telegram Bot API, Web-based dashboard for managing keywords and system status, and the audio transcription function converts user speech input into text by sending WAV audio files to the Deepgram API and receiving transcription results in JSON format. These results are compared against stored keywords in the system database. When a match is

IJAIS International Journal of Artificial Intelligence and Science
Publsihed By Asosiasi Doktor Sistem Informasi Indonesia

found, a Telegram notification is sent automatically using an HTTP POST request containing the message payload and chat ID.

The prototype workflow began with continuous audio recording. Each audio segment was sent to the Deepgram API for transcription, and the result was compared with stored keywords. When a match was found, a notification message was sent to the store attendant via Telegram. The web-based dashboard allowed users to manage keywords and control the system's operational status. This made the system user-friendly even for non-technical users. Figure 4 shows the appearance of the system's dashboard interface.

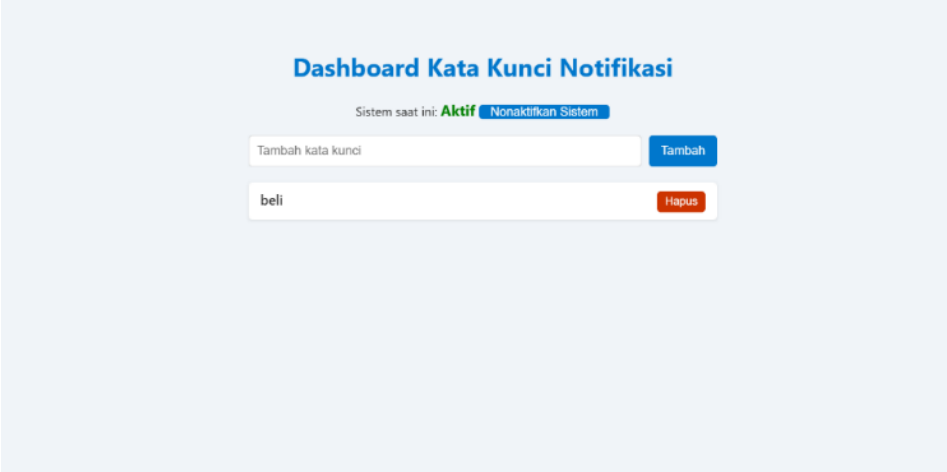

Figure 3. System Interface

3.2 Prototype Testing

Initial prototype testing was conducted in a controlled environment to evaluate the system's core functionality. The system successfully detected the keyword "beli" (buy) with a 100% success rate under optimal conditions: a quiet environment with a distance of approximately 10 cm between the sound source (speaker) and the microphone. The average response time between the user speaking the keyword and receiving the Telegram notification ranged from 4 to 6 seconds, which is within the expected performance range for real-time applications. In addition to the primary functionality, the web-based dashboard was also evaluated. It was found to be fully operational, allowing users to modify keywords and monitor the system's status. These results confirm that the prototype fulfills the basic requirements of a speech recognition-based notification system, providing accurate keyword detection and timely notifications.

3.3 System Improvement

Following the initial testing phase, several enhancements were made to improve the system's efficiency and robustness. One key improvement was the integration of a Voice Activity Detection (VAD) module. The VAD module detects human speech activity in real-time, ensuring that only audio segments containing speech are sent for transcription. This refinement helps to:

1) Optimize API Usage: By eliminating the need to process periods of silence or background noise, the system reduces unnecessary calls to the Deepgram API, which in turn improves overall system efficiency.
2) Reduce Latency: The VAD module helps in minimizing the time between detecting speech and sending audio for transcription, leading to faster response times.
3) Improve Data Processing: The system now processes only relevant audio, avoiding unnecessary transcription of non-speech sounds, thereby streamlining the overall operation.

These improvements contribute to making the system more reliable and efficient, particularly in environments with fluctuating noise levels or when only intermittent speech is present.

3.4 Final Testing

To evaluate the real-world performance of the system after the VAD module was integrated, final testing was conducted at Barokah Store under two environmental conditions: quiet and noisy. The phrase “aku mau beli” (I want to buy), with the keyword “beli,” was chosen for testing purposes. The system’s performance was evaluated based on accuracy, response time, and transcription errors. Each condition was tested 10 times to provide a comprehensive assessment of system performance. The metrics used for evaluation included Word Error Rate (WER) and Character Error Rate (CER), which are standard metrics for assessing the quality of speech recognition systems.

Table 1. Final Testing Results

Condition	Accuracy	Avg. Response Time	WER	CER
Quiet	100%	3.72 s	0%	0%
Noisy	80%	3.8 s	13.33%	12.5%

IJAIS International Journal of Artificial Intelligence and Science
Publsihed By Asosiasi Doktor Sistem Informasi Indonesia

3.5. Discussion

The results from this study highlight the effectiveness and practicality of integrating speech recognition technology with Telegram-based notifications to improve service quality in retail environments. The system prototype demonstrated solid performance in detecting keywords from audio input, with a 100% accuracy rate in a controlled, quiet environment. These results are promising, confirming that speech recognition can be effectively applied to automate and expedite customer service tasks, addressing issues such as delayed service, which has been identified as a key factor leading to customer loss in retail settings.

The system's performance under quiet conditions was excellent, with a 100% accuracy rate and minimal response time (3.72 seconds). This success aligns with prior research in speech recognition systems, where optimal conditions, such as minimal background noise and clear audio input, significantly enhance transcription accuracy and speed [1][2]. The perfect accuracy in a controlled environment suggests that the core framework for keyword detection and notification delivery is robust and reliable.

However, the system's performance in noisy environments—while still functional—showed some degradation. The accuracy dropped to 80%, and transcription errors were noted with a Word Error Rate (WER) of 13.33% and a Character Error Rate (CER) of 12.5%. This decline emphasizes the challenges that speech recognition systems face when exposed to background noise, a limitation widely acknowledged in the literature [5]. Despite improvements such as the integration of the Voice Activity Detection (VAD) module to filter out periods of silence and irrelevant audio, background noise and accent variations remain significant factors influencing the system's accuracy. The VAD's role in reducing unnecessary audio processing was critical in mitigating some of these challenges, but further refinement is required to enhance the system's resilience to environmental noise.

The average response time between the detection of a keyword and the dispatch of the Telegram notification (3.72 seconds in quiet conditions and 3.8 seconds in noisy conditions) is within the expected range for real-time applications. This suggests that the system can operate efficiently without introducing significant delays, which is crucial for maintaining high levels of service in retail settings where immediate attention is often

required. The ability to deliver notifications almost instantaneously, coupled with real-time transcription, aligns with the system's goal of improving customer interaction by ensuring that store attendants are alerted without unnecessary delay.

Moreover, the integration of Telegram notifications provides an effective communication channel that does not require the use of specialized devices or interfaces, making it accessible and easy for store personnel to adopt. The use of an existing messaging platform enhances the scalability of the system, ensuring that it can be deployed in a wide range of retail environments without the need for costly proprietary hardware.

The integration of the VAD module represents a significant improvement in system performance, particularly in terms of efficiency. By ensuring that only audio containing human speech is sent for transcription, the VAD reduces the processing load on the Deepgram API, optimizing both resource usage and response times. This improvement is crucial for the system's scalability, especially in larger retail environments where multiple audio sources may be present simultaneously. Furthermore, reducing unnecessary transcription helps lower the overall operational costs by minimizing API usage, an important consideration for real-time systems relying on cloud-based services with usage-based pricing.

The system's modular architecture, built using Python and Flask, also enhances its maintainability and extensibility. The separation of concerns between the different system components (audio capture, transcription, keyword detection, and notification delivery) allows for future upgrades and feature expansions, such as multi-language support or enhanced noise reduction capabilities. This flexibility is essential for adapting the system to a wider range of retail environments with varying operational requirements.

Although the system has demonstrated its potential in both controlled and operational settings, the user feedback from the Barokah Store deployment highlighted areas for further refinement. Staff members reported that the ability to customize keywords and adjust system settings through the web-based dashboard greatly enhanced their experience, offering a user-friendly interface that required minimal technical knowledge. However, the need for better noise handling in noisy environments, as seen in the testing

phase, was a common concern. Store employees in high-traffic areas with multiple customers were particularly sensitive to the system's occasional misinterpretation of speech due to background noise.

Given these insights, future developments could focus on improving speech recognition accuracy in noisy environments. This could involve incorporating more advanced noise suppression algorithms or training the transcription models to better handle specific retail-related sounds, such as background chatter or product displays. Additionally, integrating the system with other sensors (e.g., motion detectors or camera-based systems) could further enhance its accuracy by providing additional context to the speech recognition process, helping to distinguish between relevant customer interactions and irrelevant noise.

The successful integration of speech recognition and Telegram notifications in this study opens up possibilities for further applications in the retail industry. Beyond customer service, similar systems could be developed to automate tasks such as inventory management, employee attendance monitoring, or even customer behavior analysis, thus offering a comprehensive suite of tools to improve operational efficiency. Moreover, this system could be extended to incorporate machine learning algorithms that continuously improve keyword detection and transcription accuracy over time. By learning from real-world interactions, the system could refine its responses to accommodate regional accents, dialects, and store-specific terminologies, further enhancing the customer experience.

4. CONCLUSION

The automatic customer notification system at Toko Barokah, based on speech recognition and the Telegram bot, has been successfully developed and meets the objectives outlined in the research. The system efficiently detects voice keywords in real time using the Deepgram API and sends automatic notifications to store attendants via Telegram. The integration of the Voice Activity Detection (VAD) module, utilizing the webrtcvad library, significantly enhanced processing efficiency by ensuring that only human speech was recorded. This improvement led to faster response times and reduced API usage. Testing results demonstrated that the system achieved high accuracy and

responsiveness in both quiet and noisy environments. The prototype-based approach facilitated real-world testing within the store environment, resulting in a functional, user-friendly system that effectively meets the needs of the store and its staff.

ACKNOWLEDGMENT

The author would like to express gratitude to Universitas Semarang and Barokah Store for their support and opportunity to conduct this research and system testing. Special appreciation is also extended to the academic supervisor for the guidance and direction provided throughout the research process.

REFERENCES

[1] R. Hamid, D. L. Radji, and Y. L. Ismail, "Oikos-Nomos: Jurnal Kajian Ekonomi Pengaruh Empathy dan Responsiveness Terhadap Minat Kunjungan Ulang Pelanggan," *Oikos-Nomos: Jurnal Kajian Ekonomi*, vol. 5, no. 1, pp. 23-34, 2020.

[2] F. R. Sinay, R. Facey, and G. Rehatta, "Pengaruh Tata Letak Produk Terhadap Keputusan Pembelian Pada Minimarket Alfamidi Depok Lima Desa Poka Kecamatan Kota Ambon," *Jurnal Ilmiah Manajemen*, vol. 12, no. 1, pp. 67-81, 2024.

[3] A. H. Pratama Subakti, A. M. Hara Pardede, and M. A. Syari, "Rancangan Sistem Notifikasi Kedatangan Pembeli Dengan Suara Menggunakan Arduino," *Jurnal Teknologi dan Sistem Informasi*, vol. 10, no. 1, pp. 45-58, Jan. 2023.

[4] M. Salman Farizi et al., "Implementasi Speech Recognition Pada Sistem Kendali Perangkat Elektronik Rumah Berbasis IoT (Internet of Things) dan Mobile Application," *Jurnal Teknologi Informasi*, vol. 15, no. 3, pp. 112-121, 2022.

[5] H. Wijaya, "Teknologi Pengenalan Suara tentang Metode, Bahasa dan Tantangan: Systematic Literature Review," *bit-Tech*, vol. 7, no. 2, pp. 533–544, Dec. 2024, doi: 10.32877/bt.v7i2.1888.

[6] Z. Ardila Safitri, E. Haerani, R. Muzawi, M. Affandes, and Pizaini, "Intrusion Detection System (IDS) Pada Snort Dengan Bot Telegram Sebagai Sistem Notifikasi Terhadap Serangan Syn Flood dan Ping Of Death," *Jurnal Keamanan Siber*, vol. 18, no. 2, pp. 76-84, Jun. 2024.

IJAIS International Journal of Artificial Intelligence and Science
Publsihed By Asosiasi Doktor Sistem Informasi Indonesia

[7] R. A. Darmawan, M. Ulfah, and A. S. Irtawaty, "Sistem Keamanan Kotak Amal Berbasis NodeMCU Menggunakan Bot Telegram," *Jurnal Teknik Elektro dan Komputasi (ELKOM)*, vol. 5, no. 1, pp. 28–37, Mar. 2023, doi: 10.32528/elkom.v5i1.9370.

[8] S. Prasetyo and S. Abdullah, "Rancang Bangun Penyiram Tanaman Otomatis Berbasis Internet of Things Menggunakan NodeMCU dan Telegram," *Jurnal Restikom: Riset Teknik Informatika dan Komputer*, vol. 3, no. 2, pp. 51–59, 2021.

[9] A. F. Muhammad, "Developing Dataset Management of Speech Recognition Based Automatic English-Speaking Skill Testing System," *Journal of Educational Sciences*, vol. 7, no. 2, pp. 296-305, Apr. 2023, doi: 10.31258/jes.7.2.p.296-305.

[10] D. Naufal Kurozy, R. Gilang Pratama, and A. E. Muhammad, "Penerapan Metode Prototype Pada Perancangan Sistem Pendaftaran Mahasiswa Baru," *Jurnal Pustaka Nusantara Multidisplin*, vol. 3, no. 1, pp. 45-59, 2025.

[11] N. Renaningtias and D. Apriliani, "Penerapan Metode Prototype Pada Pengembangan Sistem Informasi Tugas Akhir Mahasiswa," *Jurnal Rekursif*, vol. 9, no. 1, pp. 101-112, 2021.