International Journal of Computer Techniques–IJCT Volume 13 Issue 3, May-June - 2026

Open Access and Peer Review Journal ISSN 2394-2231 https://ijctjournal.org/

RESEARCH ARTICLE OPEN ACCESS

Design and Development of a Real-Time AI-Powered Medical Voice Agent Using Next.js

Riddhi Pandey*, Anamika Nishad**, Smt Anuradha Singh, Mudit Dubey

*(Department of BCA, Digvijai Nath PG College, Gorakhpur
Email: riddhipandey149@gmail.com)
**(Department of BCA, Digvijai Nath PG College, Gorakhpur
Email: adityanishad1000@gmail.com)
(Department of BCA, Digvijai Nath PG College, Gorakhpur
Email: singhanuradha461@gmail.com
(Department of BCA, Digvijai Nath PG College, Gorakhpur
Email: dubeymudit.110200@gmail.com

----------**********----------

Abstract:

The integration of Artificial Intelligence (AI) with contemporary web technologies has played a crucial role in improving both the accessibility and overall efficiency of healthcare services. In this paper, we present the design and development of an AI-powered Medical Voice Agent built using Next.js. The system is designed to support real-time, conversational interactions between patients and a virtual healthcare assistant, making digital healthcare support more immediate and user-friendly.
To achieve this, the platform incorporates key technologies, including Natural Language Processing (NLP), along with speech-to-text and text-to-speech capabilities. These components work together to interpret user input—whether spoken or typed—and generate relevant medical guidance in real time. This allows the system to respond in a way that feels interactive and context-aware, rather than purely transactional.

A central feature of the proposed platform is its intuitive interface, where users can describe their symptoms through either text or voice. As illustrated in the conversational module with a virtual doctor, the system processes the input to identify potential concerns and recommend appropriate next steps. These may include consulting appropriate specialists, such as cardiologists, neurologists, or dermatologists, as suggested by an integrated recommendation dashboard.
In addition to symptom analysis, the platform supports real-time transcription and dynamic response generation, enabling smooth and continuous interaction. The design also emphasises ease of navigation and continuous availability, ensuring users can access assistance at any time. Together, these features contribute to a system that aims to provide reliable, round-the-clock healthcare support in a more accessible and responsive manner.

Keywords — Artificial Intelligence, Medical Voice Agent, Next.js, Natural Language Processing (NLP), Speech Recognition, Healthcare Technology, Conversational AI, Telemedicine, Doctor Recommendation System, Real-Time Assistance

----------**********----------

1. INTRODUCTION

The rapid growth of artificial intelligence has brought notable changes to the healthcare sector, making services more accessible, efficient, and increasingly centred around patient needs. Among these developments, AI-powered voice agents stand out as a particularly impactful innovation. These systems are designed to engage in human-like conversations, allowing them to assist patients in real time. By combining technologies such as natural language processing (NLP), speech recognition, and machine learning, they can interpret patient queries, offer preliminary guidance, and support smoother interactions within healthcare systems.

Conventional healthcare models often struggle with issues such as extended waiting periods, limited access to medical professionals, and inefficient methods for prioritising patient care. AI-based medical voice agents attempt to address these limitations by providing continuous, 24/7 assistance, easing the workload on healthcare providers, and enhancing the overall patient experience. When integrated with modern web

frameworks such as Next.js, these systems can be developed as scalable and responsive applications capable of real-time communication.

This research examines the design and implementation of an AI medical voice agent built using Next.js. Particular attention is given to enabling real-time interaction, designing intuitive user interfaces, and ensuring that the system delivers dependable medical support. The proposed solution is intended to help users report symptoms, receive basic health-related guidance, and connect with relevant specialists when needed. At the same time, it emphasises the importance of maintaining data privacy and adhering to ethical standards in healthcare technology.

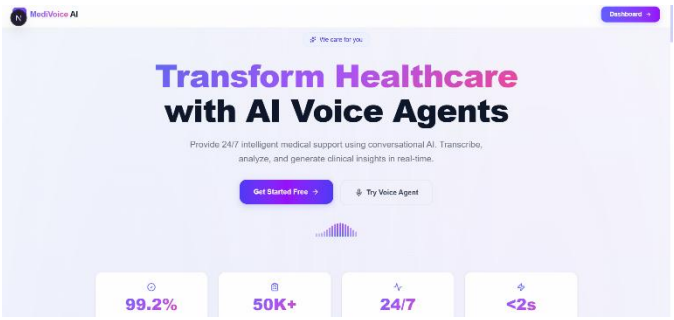

Fig 1: Homepage Interface

2. OBJECTIVES OF THE STUDY

The main objectives of this study are outlined below, with a focus on developing a practical, user-centred AI-based healthcare solution.

- First, this research aims to design and develop an AI-powered medical voice agent. The goal is to build a conversational system that accurately understands and responds to user queries about health concerns. This system is intended to support both speech and text inputs, allowing users to interact with it flexibly and intuitively.
- Another key objective is to enable real-time voice interaction. To achieve this, the study integrates speech-to-text and text-to-speech technologies so that communication between the user and the system feels natural and uninterrupted. This real-time capability is essential for creating a more human-like and efficient interaction experience.
- The research also focuses on leveraging Next.js to create a scalable and high-performance architecture. By using server-side rendering and efficient data handling techniques, the system is designed to remain responsive even during continuous interactions, which is critical for real-time applications.
- In addition, the study aims to incorporate basic symptom assessment and guidance features. The system is designed to interpret user-reported symptoms and provide general medical suggestions, including possible next steps or recommendations to consult relevant healthcare specialists. While not a replacement for professional diagnosis, this feature serves as an initial support layer for users.
- Improving healthcare accessibility is another important objective. The proposed system is intended to offer round-the-clock assistance, particularly benefiting individuals in remote or underserved areas where immediate medical consultation may not always be available.
- Equally important is the focus on data privacy and ethical considerations. The research emphasises secure data handling practices and ensures that the system aligns with established healthcare regulations and ethical AI standards, thereby maintaining user trust and safety.
- Finally, the study seeks to evaluate the overall performance and user experience of the system. This includes assessing its accuracy, efficiency, and usability through systematic testing and user feedback, ensuring that the solution is both effective and practical in real-world scenarios.

3. PROJECT DESCRIPTION

The operation of the Real-Time AI Medical Voice Assistant follows a structured sequence of interconnected steps. Each stage is designed to ensure secure access, accurate processing, and meaningful interaction between the user and the system. The overall workflow is described below.

3.1 User Authentication

The process begins when the user accesses the web-based application. To ensure security and privacy, the system requires the user to either log in or create an account through a secure authentication mechanism. Access to the medical voice assistant is restricted to authenticated users only, thereby safeguarding sensitive interactions.

3.2 Voice Input Capture

Once authentication is successful, the user can initiate interaction by activating the microphone. The system allows users to describe their medical symptoms naturally and conversationally. This audio input is captured directly through the browser in real time, enabling a seamless and intuitive user experience.

3.3 Real-Time Speech-to-Text Conversion

The recorded audio is then transmitted to a speech recognition service for processing. This component converts spoken language into text almost instantaneously, ensuring minimal delay. The resulting transcription is displayed to the user, allowing them to verify the accuracy of the captured input before further processing.

3.4 AI-Based Symptom Analysis

Following transcription, the text is analysed by the AI system. Using natural language understanding techniques, the system interprets the user's input to identify symptoms and their context. This step focuses not only on extracting keywords but also on understanding the medical relevance of the information provided.

3.5 AI Response Generation

Based on the interpreted symptoms, the AI generates an appropriate response. The output typically includes general medical guidance along with precautionary suggestions. When necessary, the system also incorporates emergency disclaimers to encourage users to seek professional medical attention.

3.6 Response Display

The generated response is presented on the user interface in a clear, organised manner. The system is designed to operate in near-real time, ensuring interactions feel smooth and responsive. Emphasis is placed on readability and user-friendly presentation to enhance overall usability.

3.7 Data Storage

To support continuity and future improvements, user interaction data—including symptom descriptions and AI-generated responses—is securely stored. The database maintains these records in a structured format, ensuring both accessibility and data protection. This stored information can later be utilised for system refinement and reference.

3.8 Session Termination

Finally, the user can end the session at any time. A secure logout mechanism is implemented to protect user data and maintain privacy after the interaction concludes.

4. METHODOLOGY

This section outlines the structured approach adopted for the design, development, and deployment of the real-time AI medical voice assistant. The project follows a modular, full-stack development strategy, integrating modern web technologies with AI-driven components to ensure both functionality and scalability.

4.1 Requirement Analysis

The development process begins with identifying the core need for a voice-based medical assistant capable of real-time interaction. At this stage, key user requirements are carefully analysed, including usability, responsiveness, security, and scalability. In addition, system boundaries, potential limitations, and essential functional requirements are clearly defined to guide subsequent development stages.

4.2 System Design

Based on the identified requirements, the overall system architecture is designed. This includes structuring the frontend, backend, AI processing unit, authentication mechanism, and database layer. Particular attention is given to designing a secure user authentication flow. Furthermore, the data flow between voice input, AI-based processing, and response generation is mapped to ensure seamless communication across components.

4.3 Frontend Development

The user interface is developed using Next.js, React, and TypeScript, with a focus on responsiveness and ease of interaction. Features such as microphone access are implemented to enable voice input. The interface is designed to display both the real-time transcribed text and the AI-generated responses clearly and accessibly, supporting a smooth user experience.

4.4 Speech Recognition Integration

To enable real-time voice processing, Assembly AI is integrated into the system. This component captures the user's spoken input and converts it into text almost instantly. Emphasis is placed on maintaining low latency and high transcription accuracy, as these factors significantly affect the application's overall usability.

4.5 AI Symptom Understanding and Response Generation

Once the input is transcribed, it is processed using AI models to interpret the user's symptoms. The system then generates responses that are contextually relevant and medically informative. Care is taken to ensure that all outputs remain advisory in nature, avoiding diagnostic claims while still providing useful guidance.

4.6 Authentication and Security

Security is addressed by implementing user authentication through Clerk. This ensures that user sessions are properly managed and that access to personal data is restricted. Only authenticated users may use the voice interaction features and access stored information, thereby maintaining data confidentiality.

4.7 Database Integration

The system utilises Neon DB, a serverless PostgreSQL database, to securely store user details, symptom inputs, and AI-generated responses. The database is structured to

allow efficient data retrieval while supporting scalability. Measures are also taken to preserve data integrity and ensure user privacy.

4.8 Testing

To validate the system's functionality, both unit testing and integration testing are conducted. Individual components are tested independently, followed by combined testing to ensure smooth interaction between the frontend, backend, AI services, and database. Additional testing focuses on system performance, response accuracy, and error handling under different conditions.

4.9 Deployment

The application is deployed on a cloud platform to make it publicly accessible. During deployment, environment variables are configured, and API keys are secured. The system is optimised to perform efficiently under real-time usage conditions, ensuring consistent performance for end users.

4.10 Maintenance and Future Enhancements

After deployment, ongoing monitoring is carried out to evaluate system performance and gather user feedback. Based on these insights, future improvements are planned, including the addition of multilingual support, integration with doctor consultation services, and compatibility with wearable health devices.

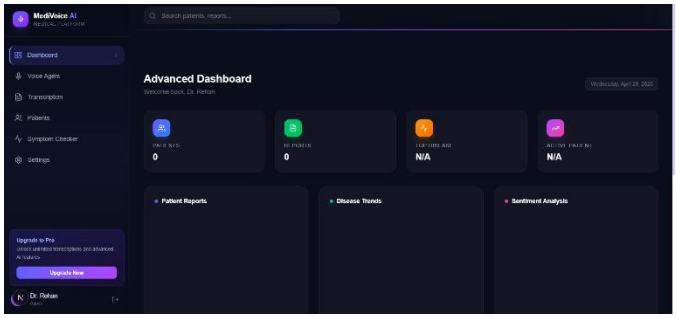

Fig 4: Dashboard Interface

5. SYSTEM DESIGN

The AI Medical Voice Assistant is designed as a full-stack, real-time healthcare support platform that enables users to interact with virtual doctors via voice or text. By combining conversational AI, speech processing technologies, and a structured medical knowledge base, the system aims to deliver intelligent and context-aware assistance. The overall architecture follows a client–server model and adopts a modular microservices-based approach to support scalability, flexibility, and efficient performance.

5.1 Architecture Design

To organise functionality effectively, the system is divided into three primary layers, each responsible for a distinct set of operations.

a) Frontend Layer (User Interface)

- The frontend is developed using Next.js and serves as the primary point of interaction between the user and the system. It provides several key interfaces, including a doctor selection dashboard, a voice-enabled interaction module, and a chat-based communication interface. The design relies on React components for structure, Tailwind CSS for styling, and the Web Speech API to enable voice input and output.
- Particular emphasis is placed on usability and responsiveness. Features such as the "Consult a Specialist" section allow users to choose relevant medical domains, while real-time chat with an AI doctor ensures continuous interaction. Additionally, voice-to-text functionality and a responsive layout contribute to a smooth user experience across devices.

b) Backend Layer (Application Server)

The backend is implemented using Node.js through Next.js API routes. It is responsible for managing request processing, user authentication, session handling, and coordination between different services. This layer acts as the central orchestrator of system operations.

- **Core Modules:**

1. **Speech Processing Module**
   This module handles both speech-to-text (STT) and text-to-speech (TTS) conversion, enabling bidirectional voice interaction.

2. **AI Engine Integration**
   The system connects to natural language processing models, such as large language models (LLMs), to interpret user input and generate appropriate medical responses.

3. **Doctor Routing Logic**
   Queries are dynamically routed by medical specialisation, such as cardiology, dermatology, and neurology. This ensures that responses are contextually aligned with the user's concern.

c) AI and Data Layer

This layer is responsible for understanding user input and managing domain-specific knowledge.

1. **Natural Language Processing (NLP) Engine**
   The NLP engine interprets patient queries, extracting relevant symptoms and identifying user intent. This step is critical for accurate response generation.
2. **Medical Knowledge Base**
   A structured repository is maintained to store information related to symptoms, diseases, and treatment guidelines. This knowledge base supports informed and consistent AI responses.
3. **Database**
   The database stores user-related information, including profiles, interaction history, and doctor-related data. It ensures that conversations can be retained for future reference while maintaining data organisation and security.

5.2 System Workflow

- The interaction process follows a logical sequence. The user begins by opening the web application built on the Next.js frontend. They can either select a specialist or initiate a voice-based interaction. If voice input is used, the system captures the speech and converts it into text. This text is then forwarded to the backend API, where the AI model processes it.
- Based on the processed query and available medical knowledge, a response is generated. The output is presented to the user in textual form and, if required, converted back into speech. Finally, the interaction is stored in the database for record-keeping and potential future use.

5.3 Component Diagram (Textual Representation)

- The system can be conceptually visualised as a layered pipeline. User interaction begins in the browser, which connects to the Next.js frontend containing both the user interface and voice interaction components. Requests are then passed to the backend API layer. From there, they are processed through multiple modules, including speech-to-text conversion, the NLP-based AI model, doctor routing logic, and text-to-speech synthesis. The processed data is ultimately stored and retrieved from the database and medical knowledge base.

5.4 Key Technologies Used

- The system integrates a range of modern technologies across different layers. The frontend is built with Next.js, React, and Tailwind CSS, while the backend uses Node.js via API routes. AI capabilities are enabled through NLP models, particularly large language models. Voice processing includes both speech-to-text and text-to-speech technologies. For data storage, databases such as MongoDB or PostgreSQL are utilised.

5.5 Design Considerations

- Several important considerations guide the system design. Scalability is achieved through a microservices-based architecture, allowing independent components to expand as needed. Real-time voice processing is prioritised to maintain low latency and ensure responsive interactions. Security measures, including data encryption, are implemented to protect sensitive patient information. Accuracy is supported by structured medical knowledge validation, while usability is enhanced by maintaining a simple, intuitive interface.

5.5 Future Enhancements

- Looking ahead, the system can be extended in several meaningful ways. Integration with wearable devices could enable real-time health monitoring, while incorporating human doctor handoff would allow seamless transition to professional consultation when required. Additional improvements include multilingual voice support and advanced machine learning techniques for more precise symptom prediction.
- This version improves readability, removes repetitive phrasing, and presents the architecture in a more thoughtful and academically natural style while preserving all original content and structure.

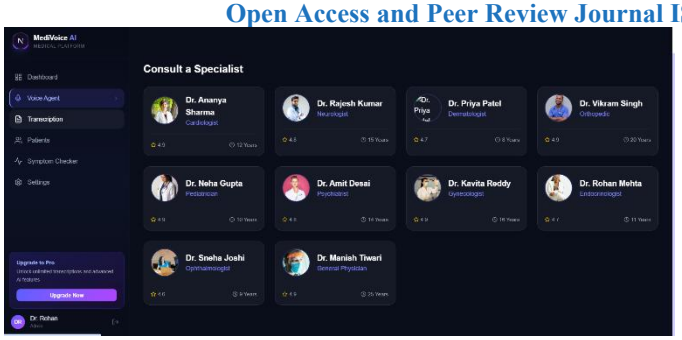

Fig 3: Specialist Listing

6. RESULT AND DISCUSSION

6.1 System Performance and Functionality

- The developed AI medical voice agent demonstrates the feasibility of a conversational interface that facilitates interaction between patients and a virtual healthcare system. By integrating voice and text inputs with an AI-driven backend, the system can simulate aspects of a real-time medical consultation environment.

Key observations:

The chatbot interface supports natural-language interaction, allowing users to express their concerns without relying on rigid input formats. The system can identify common symptoms, such as fever, and respond in a context-aware manner. In addition, it provides appropriate guidance by directing users toward relevant healthcare options, for example, suggesting consultation with a general practitioner when necessary.

6.2 User Interface and Experience

- The user interface, developed using Next.js, offers a modern and responsive design that prioritises usability. The overall layout is structured to make navigation straightforward, even for users with limited experience in digital health platforms.

- The doctor consultation dashboard allows users to select specialists based on factors such as ratings, experience, and area of expertise, which adds a practical decision-making layer. The chat interface is designed to resemble real telemedicine interactions, helping users feel more at ease during communication. Furthermore, the inclusion of team and system-related pages enhances transparency, thereby contributing to user confidence in the platform.

- From a broader perspective, the clarity and organisation of the interface play a significant role in building trust. The conversational format, in particular, reduces the learning curve and makes the system more accessible to a wider range of users.

6.3 AI Interaction Quality

- The AI component demonstrates a reasonable level of conversational capability within short interactions. It can retain context during exchanges, provide general medical guidance, and offer safe fallback responses, such as recommending consultation with a qualified doctor when appropriate.

- At the same time, certain limitations are evident. The system intentionally avoids detailed diagnosis, a necessary design choice for safety, but one that restricts its clinical applicability. Responses tend to remain generalised and may not fully account for complex or long-term medical histories. Additionally, the accuracy of voice interactions can be affected by external factors, such as background noise and variations in user accents.

6.4 Scalability and Architecture

- The use of Next.js contributes to both performance and scalability. Features such as server-side rendering (SSR) enable faster content delivery, while API-based integration supports efficient communication with AI services. The modular structure of the system further allows for easier expansion and maintenance as new features are introduced.

6.5 Comparative Analysis

- When compared to traditional telemedicine platforms, the AI voice agent offers certain practical advantages. It can reduce the initial consultation burden on

healthcare professionals by handling preliminary queries and providing instant responses. This immediacy also improves accessibility, particularly in settings with limited medical resources.

- However, it is important to recognise its boundaries. The system is not intended to replace human doctors and should be viewed as a supportive tool rather than a substitute for professional care. For real-world clinical deployment, rigorous validation and testing would be essential.

6.6 Future Improvements

- There are several directions in which the system can be further enhanced. Improvements in speech-to-text and text-to-speech components could increase interaction accuracy and responsiveness. The integration of structured medical knowledge graphs may strengthen the quality of AI-generated responses. Additionally, developing personalised AI models could allow the system to better adapt to individual users. Finally, clinical validation through trials involving real patients would be necessary to assess its effectiveness in practical healthcare settings.

- This version maintains your original ideas while improving readability, depth, and academic tone, making it sound more like authentic university-level writing.

6. FUTURE SCOPE

- The scope of this project outlines the intended capabilities of the system while also clarifying its limitations. Establishing these boundaries is important to ensure that the system is understood and used appropriately within a healthcare context.
- The primary focus of the application is to provide preliminary medical guidance rather than definitive diagnosis or treatment. It is designed to help users understand their symptoms at a basic level and does not replace professional medical consultation or clinical decision-making.
- The system supports real-time voice interaction, allowing users to describe their symptoms through a web-based interface. Based on this input, the AI generates informative, contextually relevant responses. The application is intended for individual use and can be accessed on any device with a microphone and a stable internet connection, thereby enhancing its accessibility.
- From a functional perspective, the project incorporates essential features such as user authentication, secure data storage, and real-time processing. However, it does not extend to broader healthcare management functionalities, such as hospital administration systems or appointment scheduling services. This limitation reflects a deliberate focus on core interaction and guidance rather than full-service healthcare integration.
- It is also important to note that the medical information provided by the system is general in nature and derived from AI models. As such, it is not suitable for handling emergencies or making critical medical decisions. Users are expected to seek professional medical help when necessary.
- Despite these constraints, the system is designed with scalability in mind. Its architecture enables future enhancements, including multilingual support, integration with doctor consultation services, and compatibility with wearable health devices. These potential extensions highlight the flexibility of the current design while remaining beyond the immediate scope of the project.

8. CONCLUSIONS

This project demonstrates the successful design and implementation of a real-time, AI-powered medical voice assistant that enables users to interact through natural, voice-based communication. By bringing together modern web technologies, artificial intelligence, and real-time speech recognition, the system provides a practical, accessible platform for delivering preliminary medical guidance.

From a technical perspective, the application integrates a responsive frontend built with Next.js, React, and TypeScript, alongside real-time speech-to-text processing to accurately capture user input. This is complemented by AI-based symptom interpretation, which allows the system to generate context-aware responses. Additional components, such as secure authentication mechanisms and serverless database storage, contribute to reliable data handling and user safety. Taken together, these elements support a scalable, efficient, and secure system.

An important contribution of the system is its ability to lower barriers to access to basic healthcare information. In particular, voice interaction makes the platform more accessible to users who may find traditional text-based interfaces challenging. At the same time, it is important to recognise that the application is not intended to replace professional medical consultation. Instead, it serves as a supportive tool, helping users understand symptoms and seek appropriate care.

Overall, the project illustrates the growing relevance of AI and voice-enabled technologies in healthcare applications. It also establishes a foundation for further development, with clear opportunities for enhancement, including multilingual capabilities, more advanced medical analysis, and closer integration with professional healthcare services.

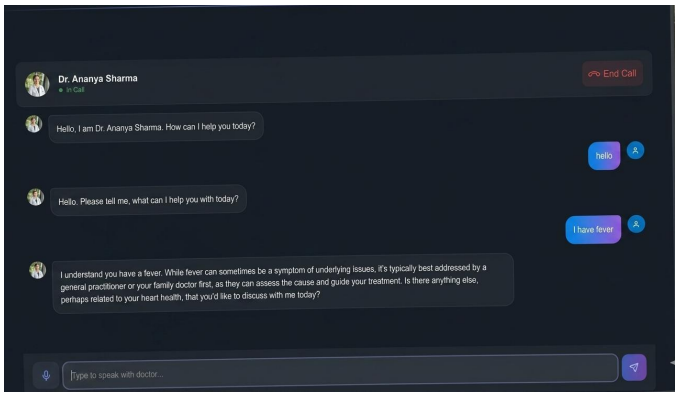

Fig 5: Chat Interface

REFERENCES

[1] Next. js-Official Documentation.
Available at: Next.js Documentation.
React– Official React Documentation.
Available at: React Official Docs.
[2] TypeScript–Microsoft TypeScript Documentation.
Available at: TypeScript Handbook.
[3] Assembly AI– RealTime Speech to-Text API Documentation.
Available at: Assembly AI Docs.
[4] Clerk–Authentication and User Management Documentation.
Available at: Clerk Official Docs.
[5] Neon–Serverless PostgreSQL Documentation.
Available at: Neon Database Docs.
[6] Russell, S., & Norvig, P., Artificial Intelligence: A Modern Approach, Pearson Education • Jurafsky, D., & Martin, J. H., Speech and Language Processing, Pearson.
• MDN Web Docs – Web APIs, JavaScript, and Browser Speech Handling.
Available at: Mozilla Developer Network.
• Research articles and online resources on AI in Healthcare and Voice Assistants.