IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Echomind: An Intelligent Voice-Driven Conversational Ai Avatar System

[1]Rahmath Ali Khan, [2]Amman Abdul Latheef, [3]Syed Daniyal Mubarak, [4]Syed Minhaj Ur Rahman, [5]Dr. Md Azizuddin

[1, 2, 3, 4]Student, [5]Principal

[1,2,3,4]Department of Computer Science,

[1,2,3,4]Mumtaz College of Engineering and Technology, Hyderabad, Telangana, India

[5]Department of Computer Science, Mumtaz College of Engineering and Technology, Hyderabad, Telangana, India

Abstract: This paper presents EchoMind, an Android-based conversational avatar system built upon four tightly coupled technical components: on-device automatic speech recognition, a cloud-hosted large language model for response generation, a neural text-to-speech engine for vocal output, and a Unity-rendered three-dimensional character for visual feedback. Rather than developing proprietary deep learning models from the ground up, EchoMind adopts an orchestration-based architecture that integrates established third-party services through well-defined interfaces, making the system both feasible as a final-year academic project and straightforward to extend. The interaction pipeline begins when the user activates the microphone; the captured audio is transcribed locally, a rolling context summary is computed on the client side, and the consolidated prompt is dispatched to the language model. The generated textual reply is subsequently synthesized into natural-sounding speech and delivered through an animated avatar capable of transitioning among idle, greeting, listening, and speaking states. A lightweight local memory buffer preserves conversational continuity across successive turns without imposing significant computational overhead. The overall architecture demonstrates that coherent, human-friendly voice interfaces can be constructed by composing existing Artificial Intelligence (AI) services and game-engine tooling, entirely bypassing the need to train a bespoke foundation model.

Index Terms - Conversational avatar, Speech recognition, Lip-sync, Neural text-to-speech, Unity, Mobile AI.

I. INTRODUCTION

Recent advances in voice-driven interfaces have substantially reshaped user expectations in human-computer interaction. Contemporary users are increasingly disinclined to parse lengthy textual outputs from a screen; instead, they expect software to listen attentively, formulate a response, and deliver it through a recognisable speaking character. This behavioural shift carries significant implications for educational tools, technology demonstration applications, and interaction design research, as it redirects interface philosophy away from purely text-based communication toward what researchers describe as embodied conversational interaction.

EchoMind is designed in that direction. The application does not attempt to build a new foundation model. Instead, it combines existing components into one clear pipeline: microphone input, speech recognition, language generation, voice synthesis, and avatar animation. This approach keeps the project realistic, maintainable, and appropriate for a student-level product while still demonstrating modern Artificial Intelligence (AI) capabilities.

The animated character occupies a central role in the overall design philosophy. Whereas a conventional text chatbot confines all output to printed words, an avatar that displays distinct facial states for listening, processing, and speaking transforms a transactional exchange into something closer to genuine conversation. For this reason, the system gives equal priority to linguistic accuracy and to visual responsiveness. At each stage of the interaction cycle, clear on-screen indicators inform the user of the system's current operational state, a feature that measurably reduces confusion and lowers the cognitive effort required to use the application.

Dialogue continuity constitutes a further design priority. Without access to prior context, a conversational system produces responses that feel disconnected and repetitive, undermining the illusion of a coherent exchange. To counter this, EchoMind maintains a compact rolling summary of recent interaction turns and injects it into each subsequent request. The mechanism is deliberately kept lightweight to accommodate the memory and processing constraints characteristic of mobile deployment environments.

II. Related Work

Research into animated talking avatars spans several decades and has progressed through distinct technological generations. The earliest approaches relied on finite sets of handcrafted viseme shapes and deterministic mapping rules that translated phoneme sequences into approximate mouth positions. While computationally economical, these rule-driven methods produced rigid, noticeably mechanical facial motion. Subsequent work shifted toward data-driven and deep-learning methodologies, enabling audio-conditioned talking-head synthesis in which facial dynamics are predicted directly from raw speech signals, yielding substantially higher perceptual realism. Importantly, contemporary studies establish that convincing speaker simulation extends beyond the oral region; coordinated head movement, natural eye behaviour, and contextually appropriate facial expressions collectively determine whether a viewer perceives the character as lifelike [1], [2].

At the implementation level, facial animation in avatar systems is predominantly achieved through blendshape or morph-target deformation. Under this paradigm, the three-dimensional character mesh is accompanied by a library of predefined facial configurations, each representing a specific expression or articulatory posture. At runtime, these configurations are blended according to continuously updated weight values, producing smooth transitions between mouth positions. Viseme-based synchronisation pipelines leverage this mechanism by mapping discrete speech-sound categories onto the corresponding blendshape weights. Practical deployments confirm that this approach yields acceptable perceptual fidelity in real-time settings, provided the audio production pipeline maintains stable, low-latency output — a condition readily achievable within controlled game-engine environments [4].

On the speech synthesis side, neural Text-To-Speech (TTS) services have undergone marked improvements in both naturalness and prosodic expressiveness over recent years. Within avatar applications, their significance extends beyond mere intelligibility; consistent pronunciation, well-modulated intonation, and affectively appropriate delivery are all prerequisites for sustaining the illusion of a coherent personality. Services such as ElevenLabs have gained traction in this domain due to their ability to produce perceptually convincing vocal output on demand [3]. A consequential practical constraint follows from this dependency: because the lip-synchronisation subsystem derives its timing cues directly from the synthesised audio stream, the TTS engine and the animation controller must operate with tightly coupled temporal alignment.

Conversational memory management represents a fourth technical dimension that directly governs perceived interaction quality. Systems that handle each user utterance as a context-free, isolated input inevitably produce incoherent responses when topics evolve across multiple turns. Research into long-horizon dialogue architectures demonstrates that selective retrieval strategies — in which only the most contextually relevant prior exchanges are surfaced for the current response — outperform naive full-history approaches in both coherence and computational efficiency [5]. EchoMind draws upon this insight by maintaining a condensed local summary of recent dialogue rather than appending an unbounded conversation log to every model request.

III. System Design

EchoMind is structured around a three-tier modular decomposition that enforces clear separation of concerns across its constituent subsystems. The Android application tier assumes responsibility for all device-level operations: presenting the User Interface (UI), acquiring microphone permission, verifying network connectivity, executing on-device speech recognition, and rendering the transcribed text for user confirmation. The Unity engine tier manages the three-dimensional character, including scene composition, animation state transitions, and real-time rendering. Bridging these two tiers, the AI services layer receives the recognised speech, interacts with the language model, obtains synthesised audio from the TTS service, and routes the resulting output back to the animation engine. This layered topology ensures that modifications to any individual subsystem do not propagate structural changes to the others.

User interaction flow

Interaction is initiated when the user activates the microphone control, at which point the application transitions into its listening state and begins buffering the incoming audio stream. Crucially, the user retains explicit authority to terminate audio capture at will; this design choice reflects the principle that voice-activated systems must never override user agency over the microphone. While the system is in the listening state, a real-time audio-intensity visualisation and a textual status indicator provide immediate confirmation that the device is actively capturing input, reducing the uncertainty that commonly arises during silent pauses in speech.

Context memory

EchoMind manages conversational history through a compact on-device data structure that retains a condensed representation of recent dialogue turns. Rather than transmitting the full exchange verbatim to the language model with each request, the system incrementally updates a summarised snapshot of the most recent interactions. This snapshot is concatenated with the current user utterance to form the complete prompt before network dispatch. The resulting approach achieves a workable balance: the model receives sufficient contextual grounding to produce coherent follow-up responses, while the prompt length and associated inference latency remain well within acceptable bounds for a mobile deployment.

Avatar rendering

The three-dimensional character is developed and maintained as a self-contained asset, deliberately decoupled from the scene environment so that facial geometry, body rig, gesture animations, and background elements can each be iterated independently. The avatar operates across four distinct behavioural states — idle, greeting, listening, and talking — which are activated programmatically in response to the current phase of the interaction cycle. This state-driven approach lends the character a sense of responsiveness and presence that would be absent from a static visual element, reinforcing the perception that the system is an active participant in the conversation rather than a passive display surface.

Lip-sync strategy

Facial lip synchronisation in EchoMind is driven entirely by the audio output produced by the Text-To-Speech (TTS) engine, eliminating the need to pre-animate responses or maintain a manually curated library of word-level mouth sequences. The synchronisation pipeline analyses the incoming audio stream and maps detected phonetic activity to a compact set of viseme shapes, which are subsequently applied as blendshape weights on the avatar mesh. Because a relatively small viseme vocabulary is sufficient to represent the dominant visible articulatory movements of English speech, this strategy delivers perceptually adequate lip motion without incurring the computational overhead associated with full phoneme-to-geometry solvers. A further advantage is generality: the same pipeline handles any text generated by the language model without requiring per-sentence configuration.

Table 1 Core modules of EchoMind

Module	Function
Android User Interface (UI)	Handles microphone permission, listening state, and text display.
Speech Recognition	Transcribes the user's spoken input into text directly on the device without sending raw audio to a remote server.
Context Memory	Maintains an incrementally updated dialogue summary that supplies the model with relevant prior context on each request.
Large Language Model (LLM) Interface	Submits the combined user query and context summary to the hosted language model and retrieves the generated reply.
Neural TTS	Converts the model's textual reply into a naturalistically synthesised audio stream for avatar playback.
Unity Avatar Engine	Handles real-time scene rendering, behavioural state transitions, blendshape-driven lip synchronisation, and all character animations.

IV. Implementation Details

The working prototype takes the form of an Android application developed in Kotlin, employing Jetpack Compose as the declarative UI toolkit. This technology selection is motivated by Compose's reactive rendering model, which automatically propagates state changes — such as the transition from listening to processing — into the visual hierarchy without requiring imperative view manipulation. The primary screen presents a centrally positioned microphone activation control, a scrollable transcription area at the bottom of the viewport, and a compact status bar that communicates the system's current operational phase, keeping the visual layout uncluttered and immediately comprehensible to first-time users.

Microphone access is solicited at runtime in accordance with Android's permission model, which mandates that sensitive hardware capabilities be requested in context rather than declared silently at installation. Prior to each AI inference cycle, the application performs a lightweight network reachability probe. This precautionary step is necessary because, although both speech transcription and the UI layer are capable of operating offline, the response generation step and the TTS synthesis step each require active internet connectivity when routed through external Application Programming Interfaces (APIs). Surfacing a connectivity failure before submitting a network request yields a significantly more informative error message than the timeout exception that would otherwise reach the user.

The conversational memory subsystem is realised as a lightweight in-process buffer residing entirely on the client device. An asynchronous background routine periodically distils the accumulated dialogue into a condensed summary, ensuring that the buffer remains bounded in size regardless of session duration. Beyond supporting contextual coherence, this local cache provides a secondary benefit: frequently repeated queries or recently answered questions can be served from the cache directly, reducing the frequency of redundant round-trips to the remote language model and improving perceived response latency.

The character asset production pipeline originates in Blender, where the humanoid model is sculpted, UV-unwrapped, textured, and rigged with a skeletal hierarchy suitable for real-time animation. Once the authoring phase is complete, the model is exported in FBX format and imported into the Unity project, where it is coupled to an Animator Controller managing the four primary behavioural states. The idle and listening clips are designed to loop seamlessly, the greeting animation plays once on session initialisation, and multiple talking variants are available to prevent repetitive motion during extended speech output.

The listening state additionally incorporates gentle procedural head movement to further convey attentiveness.

The lip-synchronisation implementation operates on the principle that the TTS audio signal contains sufficient acoustic information to drive plausible mouth motion without access to the underlying phoneme transcript. During playback, the audio stream is continuously analysed and its dominant phonetic characteristics are resolved against a small inventory of viseme identifiers. Each identifier maps directly to a corresponding blendshape weight on the character mesh, which Unity applies at the frame level. This per-frame mapping strategy eliminates the requirement for pre-computed word-level animation curves, allowing the avatar to articulate any language model output in real time with a fixed, manageable engineering overhead.

The environment scene is authored and managed entirely within Unity, kept strictly separate from the character asset file. This architectural boundary means that the background can be redesigned, replaced, or extended at any point without necessitating a re-export of the avatar mesh or its animation data. In the current iteration, EchoMind presents a softly lit forest setting chosen for its visual tranquillity; the deliberate minimalism of the scene directs the viewer's attention toward the character and away from environmental detail, supporting rather than competing with the conversational interaction.

V. Preliminary Evaluation And Discussion

The present implementation is best characterised as a functional proof-of-concept rather than a production-grade release. Its primary evaluative purpose is to verify end-to-end pipeline integrity: confirming that a spoken user utterance can be reliably transcribed, contextualised, processed by the language model, converted to audio, and rendered as synchronised avatar speech within a single coherent application. Throughout the development cycle, the modular decomposition of responsibilities across the Android client, the AI services layer, and the Unity scene proved instrumental in maintaining architectural stability; changes within one tier did not cascade into regressions elsewhere.

A principal strength of the architecture is its inherent testability. Because each pipeline stage exposes a well-defined interface independent of its neighbours, individual components can be exercised in isolation during development and debugging. The transcription module can be validated against pre-recorded audio files before the language model is connected; the avatar animation states can be driven by synthetic audio during rig verification before the TTS service is integrated. This incremental integration strategy substantially reduces the diagnostic complexity that typically confronts small development teams when a multi-component system is assembled and tested as a monolith.

A second notable strength is the project's tractable scope. By adopting an integration-centric philosophy, EchoMind sidesteps the prohibitive data and compute requirements associated with training custom speech recognition models, generative face synthesis networks, or large language models from scratch. The resulting system nonetheless achieves a compelling demonstration of state-of-the-art AI capabilities on commodity mobile hardware. This characteristic makes the project well-suited to the constraints of an academic semester timeline and positions it as an instructive reference for final-year engineering teams seeking technically credible yet feasibly deliverable project specifications.

The prototype's limitations are equally transparent. First, the overall interaction quality is contingent on the reliability and latency of third-party API providers; a degraded or unavailable external service directly impairs the user experience in ways outside the development team's control. Second, the fidelity of avatar animation is bounded by the quality of the underlying facial rig and the completeness of the blendshape set authored during the character production phase. Third, the current memory subsystem is optimised for short to medium-length exchanges and lacks the retrieval sophistication needed for persistent long-term personalisation across separate sessions.

Notwithstanding these constraints, the prototype meaningfully advances the case for embodied voice interfaces as a viable interaction paradigm for mobile AI applications. The findings indicate that equipping a conversational agent with a visible animated character, explicit microphone state signalling, and a coherent response delivery loop substantially improves the intuitiveness of the interaction without demanding a commensurate increase in backend complexity. The system is both functional and demonstrably more approachable than a text-only equivalent, suggesting that the design principles applied here merit consideration in future, more mature implementations.

VI. Conclusion

This paper has described the design, implementation, and preliminary evaluation of EchoMind, an Android-hosted conversational avatar system that unifies on-device speech recognition, context-aware language model inference, neural speech synthesis, and real-time three-dimensional character animation within a single cohesive application. The system's architecture is organised as a set of loosely coupled, independently evolvable modules, ensuring that advances in any individual component — whether in transcription accuracy, language model capability, vocal naturalness, or rendering fidelity — can be incorporated without restructuring the surrounding system.

The central engineering principle underpinning EchoMind is pragmatic composability. Rather than investing development effort in training bespoke AI models, the project treats established third-party services as building blocks and focuses engineering effort on the orchestration layer that coordinates them into a unified, perceptually coherent experience. This strategy dramatically reduces the time-to-demonstration while still affording an authentic exploration of the technical challenges inherent in embodied conversational systems, making it an exemplary model for applied final-year engineering research.

Several directions are identified for future development. On the expressive side, integrating emotion-conditioned TTS parameters and a broader set of facial action units would enable the avatar to reflect sentiment in both vocal tone and facial configuration. On the memory side, replacing the current rolling-summary mechanism with a vector-store retrieval architecture would support persistent personalisation across multiple sessions. Additionally, incorporating an on-device fallback model for selected language tasks would reduce dependency on network availability and extend usability in connectivity-constrained environments. Collectively, these enhancements would advance EchoMind from a technically sound demonstration toward a deployable, user-facing AI companion platform.

VII. Acknowledgment

The authors gratefully acknowledge the academic guidance and institutional support extended by the faculty of Mumtaz College of Engineering and Technology throughout the duration of this project. The work also drew upon a broad body of publicly accessible technical documentation and peer-reviewed literature spanning conversational AI, neural speech synthesis, real-time lip synchronisation, and three-dimensional avatar engineering, the contributions of whose original authors are recognised through the citations provided.

References

[1] A. Chatziagapi et al., "Seeing the Sound: Multilingual Lip Sync for Real-Time Face-to-Face Translation," Computers, vol. 14, no. 1, p. 7, 2025.

[2] A. Chatziagapi et al., "AV-Flow: Transforming Text to Audio-Visual Human-like Interactions," arXiv preprint arXiv:2502.13133, 2025.

[3] ElevenLabs, "Text to Speech (product guide)," ElevenLabs Documentation. [Online]. Available: https://elevenlabs.io/docs/eleven-creative/playground/text-to-speech

[4] A. Chitimalli, "Build Real-Time AI Avatars with Lip Sync Using Agora ConvoAI & RPM," Agora Blog, 2026. [Online]. Available: https://www.agora.io/en/blog/build-real-time-ai-avatars-with-lip-sync-using-agora-convoai-rpm/

[5] R. Sumida, K. Inoue, and T. Kawahara, "Enhancing Long-term RAG Chatbots with Psychological Models of Memory Importance and Forgetting," Dialogue & Discourse, vol. 16, no. 2, pp. 74-110, 2025.

[6] I. R. Ali and A. H. Basori, "Expression of Avatar through Lip Synchronization and MPEG4 in Virtual Reality Based On XFace Toolkit: Present and Future," Procedia - Social and Behavioral Sciences, vol. 97, pp. 700-706, 2013.

[7] Oculus, "OVRLipSync SDK Documentation," Meta/Oculus Developer Resources. [Online]. Available: https://developer.oculus.com/