UNDERGRADUATE FINAL YEAR PROJECT REPORT

Department of Computer and Information Systems Engineering

NED University of Engineering and Technology

Personal Cognitive Memory Assistant: An Edge-Computing Wearable with LLM-Powered Contextual Recall

Group Number: 12 **Batch: 2022-2026**

Group Member Names:

Hasan Ahmed Shamsi	CS-22122
Muhammad	CS-22123
Abdur Raafay	CS-22131
Muhammad Maaz	CS-22145

Prof. Dr. Syed Abbas Ali

Professor

Project Advisor

Author's Declaration

We declare that we are the sole authors of this project. It is the actual copy of the project that was accepted by our advisor(s) including any necessary revisions. We also grant NED University of Engineering and Technology permission to reproduce and distribute electronic or paper copies of this project.

Signature and Date	Signature and Date	Signature and Date	Signature and Date
.....	.....	.....	.....
Hasan Ahmed Shamsi	Muhammad	Abdur Raafay	Muhammad Maaz
CS-22122	CS-22123	CS-22131	CS-22145
hasanshamsi268@gmail.com	muhammad86762@gmail.com	araafaycs@gmail.com	maaz45130@gmail.com

Statement of Contributions

Hasan Ahmed Shamsi

- **Speech Enhancement and Noise Reduction:** Implemented stationary spectral noise reduction and signal filtering techniques to optimize raw ambient audio for accurate transcription.
- **Audio Acquisition and Recording Management:** Engineered the continuous background audio buffering pipeline utilizing a hybrid WebRTC Voice Activity Detection (VAD) and RMS energy gate.
- **GPS Module Setup and Integration:** Integrated the NEO-6M GPS module via UART to fetch real-time geographic coordinates, enabling spatial metadata tagging for stored memories.

Muhammad

- **Biometric Voice Gate Programming:** Developed the local security verification layer using Resemblyzer to compute cosine similarity against the owner's baseline vocal signature.
- **Database Ingestion Pipeline and Text Splitting:** Engineered the backend data pipeline to segment transcribed text and store context-rich vector embeddings in a persistent local ChromaDB instance.
- **Context Retrieval and LLM Query Execution:** Managed the semantic nearest-neighbor search operations and compiled the retrieved historical data into structured prompt templates for the LLM.

Abdur Raafay

- **System State-Machine Architecture Design:** Designed the core operational workflow to manage seamless system transitions between Listening, Muted, Processing, and Query modes.
- **Hardware Integration and Headphone Communication Interface:** Utilized the Linux driver to intercept Bluetooth headset key events directly in the kernel, enabling a completely screen-free control mechanism.
- **Offline Edge-AI Stack Deployment:** Configured and deployed the quantized Gemma 3 1B SLM, faster-whisper STT, and Piper TTS engines for 100% on-device, offline execution.

Muhammad Maaz

- **Central System Orchestration and Concurrency:** Architected the main orchestrator script to manage asynchronous threading, ensuring background audio recording, database ingestion, and LLM inference operate simultaneously without system blocking.
- **Conversational Memory Integration:** Engineered the underlying retrieval logic to preserve ongoing conversational history, enabling the RAG pipeline to remember and synthesize past interactions accurately.
- **Speaker Diarization for Multi-Voice Recognition:** Implemented audio stream analysis protocols to distinguish between overlapping speakers, ensuring distinct voices are accurately separated and labeled within the generated transcripts.

Executive Summary

Background

Humans have a memory that deteriorates over time, this deteriorated memory is very much essential-not just for relationship building with friends but also for making significant decisions in life. As if this was not enough, recall ability dwindles either with age or with modern stress. Many of the older adults suffer from Alzheimer's disease, which spoils memory severely and cognition. Students also face difficulty during fast-paced lectures and discussions in making notes. Often, this leads to dependence on others for living independently. The present tools pitch-in like smartphone calendars or Siri, but often are not much useful because the maximum of them provide manual entry, and none effectively collaborates the spontaneous environment of daily conversations.

Problem Statement

The principal challenge is to develop something which acts as an aide-memoire without overly being intrusive. Most solutions either apply a camera that creates tremendous amounts of privacy concerns and negatively counteracts the social essence or may feel simply unnatural in such a setting. Hence, we sought to build "NEURAID" as a lightweight and audio-first system that can semantically understand and record details about your day without video recording or bulky hardware.

Methodology

NEURAID functions as a discreet, wearable "second brain." Instead of recording video, it continuously listens to ambient audio using a microphone. This speech is processed in real-time, turned into text, and organized into a searchable database using advanced AI (Large Language Models). Users can interact with NEURAID in two ways: they can ask direct questions (e.g., "What did I promise to do tomorrow?"). All responses are delivered privately through a bone-conduction headset.

Major Findings

Our analysis confirms that an audio-only approach is not only more private but also significantly less complex than visual systems. Research indicates that using AI to summarize conversations can boost memory recall by approximately 12% to 20%, offering a massive benefit to users who struggle to take written notes. The system successfully proved it could capture and retrieve essential details like names, dates, and commitments accurately.

Conclusion

NEURAID demonstrates how privacy-focused technology can enhance human memory. By integrating advanced speech recognition with intelligent AI, it creates a seamless, hands-free safety net that boosts user confidence, promotes independence in daily life, and bridges the gap between biological memory limitations and the need for reliable recall.

Acknowledgments

We would like to express our sincere gratitude to our supervisor, **Prof. Dr. Syed Abbas Ali,** for their invaluable mentorship and technical guidance throughout the development of NEURAID and for providing the essential resources and facilities required for our research.

With gratitude,

The NEURAID Project Team,

Hasan Ahmed Shamsi

Muhammad

Abdur Raafay

Muhammad Maaz

Dedication

We dedicate this project to our beloved parents, whose endless sacrifices and prayers have been our guiding light throughout this journey. To those silently battling the challenges of memory loss, may this work serve as a small beacon of hope for a more independent future.

With gratitude,

The NEURAID Project Team,

Hasan Ahmed Shamsi

Muhammad

Abdur Raafay

Muhammad Maaz

Table of Contents

List of Figures

List of Tables

List of Abbreviations

Table 1 List of Abbreviations

SYMBOLS	ABBREVIATIONS
AES	Advanced Encryption Standard (for secure memory storage)
API	Application Programming Interface
ASR	Automatic Speech Recognition
BC	Bone Conduction (headset/speaker)
CPU	Central Processing Unit
DB	Database
FN	False Negative
FP	False Positive
GPU	Graphics Processing Unit
HTTPs	Hypertext Transfer Protocol Secure
LLM	Large Language Model
RAG	Retrieval-Augmented Generation
SNR	Signal-to-Noise Ratio
STT	Speech-to-Text
TTS	Text-to-Speech
UI	User Interface
VAD	Voice Activity Detection
VecDB	Vector Database (FAISS/Chroma)

United Nations Sustainable Development Goals

The Sustainable Development Goals (SDGs) are the blueprint to achieve a better and more sustainable future for all. They address the global challenges we face, including poverty, inequality, climate change, environmental degradation, peace and justice. There is a total of 17 SDGs as mentioned below.

No Poverty

Zero Hunger

Good Health and Wellbeing

✅ Quality Education

Gender Equality

Clean Water and Sanitation

Affordable and Clean Energy

Decent Work and Economic Growth

✅ Industry, Innovation and Infrastructure

✅ Reduced Inequalities

Sustainable Cities and Communities

Responsible Consumption and Production

Climate Action

Life on Land

Peace and Justice and Strong Institutions

Partnerships to Achieve the Goals

Life Below Water

Similarity Index Report

Following students have compiled the final year report on the topic given below for partial fulfillment of the requirement for Bachelor's of Engineering in Computer and Information Systems Engineering.

Project Title

Personal Cognitive Memory Assistant: An Edge-Computing Wearable with LLM-Powered Contextual Recall

S. No.	Student Name	Seat Number
1.	Hasan Ahmed Shamsi	CS-22122
2.	Muhammad	CS-22123
3.	Abdur Raafay	CS-22131
4.	Muhammad Maaz	CS-22145

This is to certify that Plagiarism test was conducted on complete report, and overall similarity index was found to be less than 20%, with maximum 5% from single source, as required.

Signature and Date

Prof. Dr. Syed Abbas Ali

Chapter 1

Introduction

1.1 Background Information

Human memory is essential for communication, learning, and decision-making. But as natural, it proves illusory and has lapses. Ageing populations can increasingly be distinguished by mild cognitive decline (subjective memory loss, MCI) or neurological symptoms (for example, the onset of dementia), which have cut off access to everyday memory and hindered independent living. Examples of retrospective memory deficit are often demonstrated by forgetting where things are kept or forgetting the details of what was talked about. Retrospective memory loss is associated with frustration in older adults, in addition to creating stress on caregivers. Therefore, living independently through cognitive support became a priority for the healthcare and technology sectors. All the assistive technologies from simple calendars and reminders to sophisticated lifelogging systems-seek to bridge the "care-gap": the function someone needed and his actual ability. NEURAID is under development in this context of cognitive augmentation as a wearable interface for providing real-time memory support.

The NEURAID Project is taken up by Hasan Ahmed, Muhammad, Abdur Raafay and Muhammad Maaz under the guidance of Prof. Dr. Syed Abbas Ali. Our vision is to unobtrusively extend the user's episodic memory: by continuously listening to everyday speech (e.g. conversations, self-reminders), automatically indexing important information, and then allowing the user to recall it via natural voice queries.

Based on the latest technologies in speech recognition, embedding-based semantic search, and LLM-driven conversation understanding, NEURAID retrieves personal memories. The system has minor distractions: the user hears the responses through a bone-conduction headset, without leaving the user's awareness of the environment. In summary, NEURAID will empower users to

remember easily in daily activities, enhancing independence and self-assuredness while living alone.

1.2 Significance and Motivation

The development of NEURAID motivated by one's limitation of cognitive recall and the insufficiency of current digital offerings. Indeed, moments of forgetfulness form a normal part of human experience; however, these could turn debilitating for Mild Cognitive Impairment (MCI) patients or for people subject to a cognitively overwhelming high-stress environment. The notes as well as those generic voice recorders are reactive by nature, since these require manual initiation and lack all the context necessary for making retrieved information meaningful. Moreover, mainstream digital assistants can neither maintain long-term, personalized contexts from daily interactions nor are they free from serious privacy issues regarding cloud data usage.

There is an urgent need for active, context-aware, and strictly privacy-preserving systems. NEURAID fills the trust gap from most earlier studies in which users were reluctant to be memory-augmented for fear of prying and concealment. Processing is done through a dedicated Raspberry Pi 5 edge-computing architecture that is not proprietary to a smartphone application so that the processing power is strictly dedicated to the user without distractions brought about by using a multi-purpose mobile interface.

The relevance of the NEURAID project lies in developing a purpose-built hardware solution that will potentially improve personal productivity and promote independent living. Unlike software-only alternatives, NEURAID integrates physical context tracking via a GPS module directly into the hardware pipeline, allowing every memory to be tagged with precise location metadata to enrich retrieval. Such a project also takes on some of the most critical of all ethical challenges through a frequency-based voice authentication mechanism, safeguarding that only the device owner can inquire about stored memories. By offering an interaction model screen-free and controlled via the buttons of a bone conduction headset, NEURAID minimizes cognitive load and demonstrates how Edge AI can be efficiently taken into practice to augment human memory in a transparent, secure, and unobtrusive manner.

1.3 Aims and Objectives

1.3.1 Aims

Research and create a standalone wearable device in edge computing, which will boost human episodic memory by capturing ambient sounds and tagging them with place context for a semantic index. The goal of the system is a private "external brain" that allows the past user's conversations and events to be emitted through self-natural language prompts and without visual screens for independent living while easing the cognitive burden.

1.3.2 Objectives

- Design and configure a wearable edge hardware prototype using a **Raspberry Pi 5** as the central processing unit and utilize Bluetooth keypress events from a bone conduction headset (FBT-AS18) for hands-free control and mode switching.
- To implement a low-latency local voice recognition and processing pipeline capable of running on low-resource ARM architectures (Raspberry Pi 5) without external internet dependencies
- Implement a real-time speech-to-text pipeline running completely locally on the single-board computer using a quantized faster-whisper model (faster-distil-whisper-small.en) in int8 format.
- Develop a local semantic memory database using quantized vector embeddings (mxbai-embed-large-v1-Q8_0) and store conversation logs in a persistent local ChromaDB instance, eliminating cloud database dependencies.
- Integrate a **GPS module (NEO-6M)** to automatically tag every captured memory with location metadata, allowing for context-aware retrieval (e.g., "What did I say at the office?").
- Implement a strict local privacy mechanism featuring speaker biometric authentication (Resemblyzer) to verify the owner's voice print from the query recording before initiating semantic memory searches.

- Integrate a quantized local Large Language Model (Gemma 3 1B GGUF) via a local server (llama_cpp.server) to interpret natural language queries and generate concise, contextual responses from retrieved memory chunks.
- Provide a screen-free user interface where system modes (listening, muted, query) are controlled via Bluetooth headphone buttons (Hold Vol+, Hold Vol-, Click Power) with immediate voice confirmations, and output is delivered via bone conduction.
- Ensure system transparency by including **provenance data** (timestamps and location) in the AI's responses to build user trust.

1.4 Methodology

1.4.1 Real Time Audio Acquisition

Audio is captured continuously in 30ms frames using a wearable microphone connected to the Raspberry Pi 5. A hybrid gate combining WebRTC VAD and an RMS energy threshold filters out environment noise and fan static. Spoken audio is buffered locally in memory, keeping recording hands-free and non-disruptive while executing all ingestion workloads entirely on the edge device.

1.4.2 Speech Processing and Transcription

When speech stops, raw audio is preprocessed locally using the noisereduce library for stationary spectral noise reduction. The enhanced audio is transcribed locally using the faster-whisper engine (faster-distil-whisper-small.en) running in CPU int8 mode. This offline transcription converts spoken words to text, prepending speaker labels and bypassing external cloud API latency.

1.4.3 Contextual Embedding and Storage (Listening Mode)

In the "Listening Mode," the background pipeline worker consumes the locally transcribed segments from an in-memory queue:

1. **Contextual Tagging:** A **GPS module** simultaneously captures location data, tagging each text segment with precise geographic metadata.

2. **Text Splitting:** The transcript is segmented into meaningful chunks (Document Objects) to ensure accurate context retention.
3. **Vector Storage:** The tagged text segments are encoded using a local GGUF embedding model (mxbai-embed-large-v1-Q8_0) and indexed into a local ChromaDB collection. This supports offline vector similarity search and preserves data privacy.

1.4.4 Memory Retrieval Module

The retrieval process is distinct from the listening phase and is governed by strict access controls.

1.4.4.1 Authentication Layer

Before any retrieval takes place, privacy is maintained via local Speaker Biometric Verification. The system extracts a 256-dimensional speaker embedding from the query recording using Resemblyzer and computes its cosine similarity with the pre-enrolled owner signature (owner_signature.npy). If similarity is 0.7 or lower, access is denied and query execution is aborted.

1.4.4.2 Query Mode

After authentication, the Retrieval Augmented Generation (RAG) pipeline gets activated:

- The user speaks poses a question.
- The system converts this query into vector embedding.
- It carries out a nearest-neighbor similarity search in the local ChromaDB vector store to retrieve the most relevant memory documents.
- These chunks (context) are forwarded to the LLM for further processing.

1.4.5 LLM Based Interpretation & Response Generation

The retrieved memories, user-specific query, and location metadata are compiled into an augmented prompt and passed to the local Gemma 3 1B LLM server. The model:

- Interprets the user's intent from a natural language question.
- Synthesize the retrieved logs to form a coherent answer.
- The synthesis of the retrieved logs finally leads to the coherent completion of the answer.

1.4.6 Text to Speech Output Delivery

The generated text response is converted back into audio locally using the Piper ONNX TTS engine (en_US-lessac-medium.onnx). The synthesized audio is played discreetly through the Bluetooth bone conduction headset. This method ensures:

- Output privacy (only the user hears the response).
- Situational awareness (the user's ears remain open to the environment).

1.4.7 Interaction & Hardware Interface

1.4.7.1 Control Mechanism

Interaction is managed seamlessly via the Bluetooth headset's physical buttons ([+], [-], and Power) rather than a separate ESP32 module or a touchscreen. The system operates through the following state-driven workflow:

- **Listening (Passive Default):** The system initiates in this state upon boot. It continuously records ambient conversations in the background, segments audio upon silence, transcribes, and indexes text chunks into the local vector database.
- **Muted (Privacy Lock):** Toggled by holding the [+] button. This pauses the passive background listening to ensure user privacy. Holding [+] again returns the system to the Listening state.

- **Query (Mic Hot):** Triggered by holding the [-] button from either the Listening or Muted state. This suspends background operations to record a direct voice question accompanied by an immediate voice status confirmation. The recording concludes upon Voice Activity Detection (VAD) silence or by interacting with [-] again, transitioning the system to the Processing state.
- **Processing & Speaking:** The system processes the query using the local RAG pipeline (Gemma 3) and outputs the generated response via Text-to-Speech (TTS). Once the audio finishes, the system automatically reverts to its previously stored state (Listening or Muted).
- **Interrupt / Cancel:** Clicking the Power button during the Query, Processing, or Speaking states immediately aborts the current action and returns the device to its previous state (Listening or Muted).

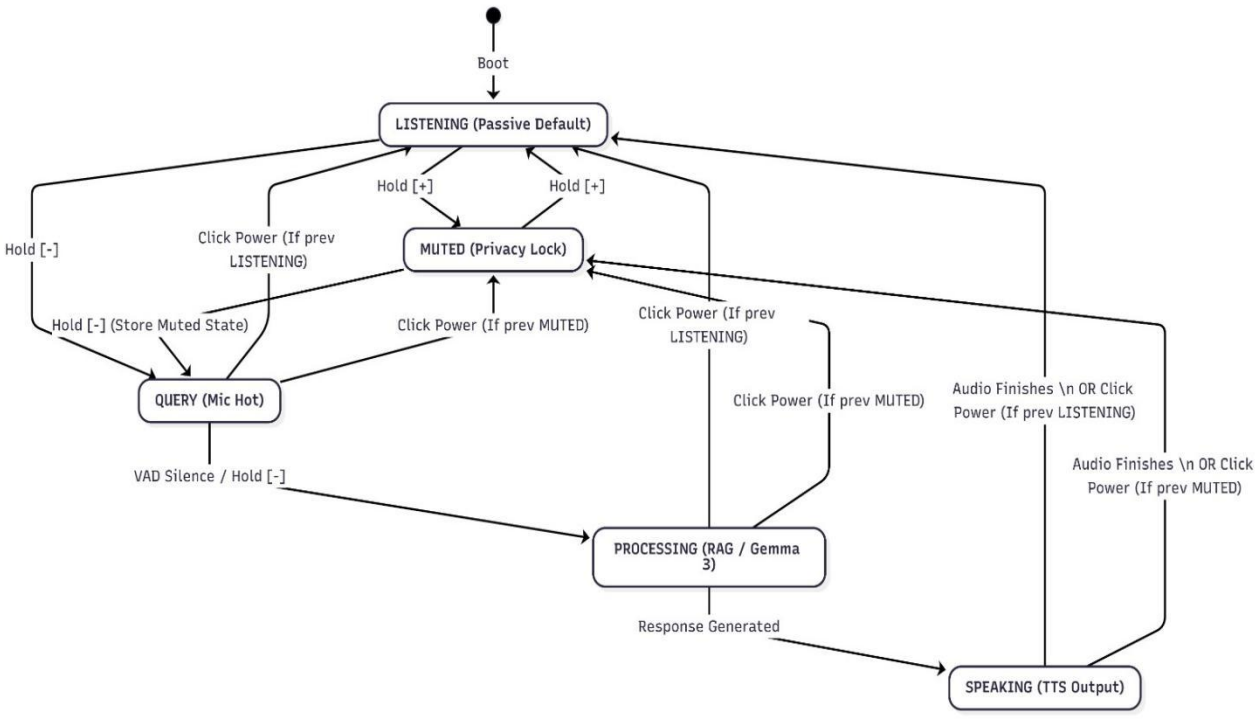

Figure 1 State Transition Diagram of NEURAID Operational Modes

1.4.8 Reliability & Optimization

To enable the system to function independently as a wearable device:

- **Edge Computing:** The Raspberry Pi 5 performs all speech-to-text, vector database, and LLM reasoning workloads locally, completely removing internet dependency.
- **Privacy Preservation:** Raw audio recordings are deleted immediately after local transcription, and database indexing runs entirely on-device, preventing external data exposure.
- **Latency Management**: The system coordinates processes via an asynchronous worker thread, optimized quantized models (int8 STT, Q8_0 LLM), and local ChromaDB to maintain response times under 15 seconds.

Chapter 2

Literature Review

2.1 Introduction

Increasingly, the discipline of Human-Computer Interaction (HCI) has concentrated on "Augmented Cognition"-the use of technology to go beyond the intrinsic limitations of memory. Traditionally, the tools of note-taking, calendaring, and other similar applications require an active input into the system by the human users, but now this has given way to entirely passive, always-on memory assistants, thanks to the innovations in AI applications. These systems utilize LLMs and semantic search to capture, index, and retrieve the so-called daily living experiences.

This chapter will outline the three foundational systems that will henceforth determine the design of NEURAID-MEMORO, MEMPAL, and AR Secretary Agent. All three existing solutions have thus proven AI memory aids' practicality and feasibility, but they also reveal some fundamental shortcomings in terms of privacy, hardware dependency, and limitation with respect to contextual awareness. NEURAID has been developed to address specific challenges that these approaches present, from being mostly visual plus application-based approaches to a dedicated privacy-first edge computing wearable.

2.2 MEMORO

Advances in recent AI lead prototypes of augmented memory systems. An instance probably worthy of mentioning is MEMORO (Zulfikar et al., CHI 2024); it is an audio-based wearable assistant that continuously captures conversations and relies on an LLM model to understand the memory needs of the user. MEMORO's main hallmark feature was its straightforward interface that had two modes for interaction. These are: (1) Query Mode, where a user may ask a question in natural language; and (2) Queryless Mode, in which case, the system employs conversational context to supply relevant information without necessarily requiring an explicit prompt.

2.3 MEMPAL

Created specifically for older people, another new system is MEMPAL (Maniar et ah., IUI 2025). It's a wear-and-use multimodal memory assistant that helps users find lost objects with an internal camera, capturing visual context, building an automated text diary of activities, objects held, their locations, etc., and querying the user by voice, for example, "where are my glasses?" On the other hand, endowed with a voice interface powered by an LLM, MEMPAL further searches the diary before providing results. According to a study with older participants, it was shown that MEMPAL significantly improved object-finding performance without any aid. Unlike MEMPAL, NEURAID rather avoids visual input; it makes the system devoid of a camera, thereby reducing both hardware complexity and privacy concern. In turn, NEURAID is restricted entirely to audio cues (speech events) and can cater to a wider variety of memory queries (not limited to objects). While MEMPAL tends to rely heavily on visual cues for contextualized recall, the main approach of NEURAID lies in using solid audio and conversational contexts to trigger recall.

2.4 AR SECRETARY AGENT

The augmentation of real-time memory through AI-generated summary activities to augment both short-term and long-term recall during information-dense interactions is witnessing increasing potential. AR Secretary Agent (El Haddad et al., 2024)-arguably one of the strongest empirical contributions-is a system that makes use of AR glasses, real-time audio capture, and LLM powered summarization to aid memory recall during meetings and conversations. In a controlled study conducted by the researchers, several dense conversations were performed by the participants, who were later tested for short-term memory performance. It is clear from the results that AI-generated summaries confer cognitive benefit.

2.4.1 Short-Term Memory Performance per Speaker

Table 2 Short-Term Memory Performance per Speaker

Condition	Macro-Average	Josh	Charlotte	Sophia	Sarah
With memory	39.6% (σ = 17.3)	48.6% (σ = 20.5)	36.4% (σ = 16.0)	34.0% (σ = 19.5)	39.5% (σ = 23.4)
Summary	12.4% (σ = 8.6)	9.9% (σ = 10.2)	16.4% (σ = 13.5)	9.6% (σ = 6.8)	13.6% (σ = 10.5)

Figure 2 Short-Term Memory Performance per Speaker

The average amount of content recalled by participants across conversations was around 39%. AI-assisted summaries boosted recall by an average of 12 percent, with some conversations being recalled as much as 16 percent. Thus, AI-assisted summarization directly helps with recall in conversations, specifically for complex or lengthy interactions-a key NEURAID use case.

2.4.2 Memory Performance Table based on User Behavior

Table 3 Memory Performance Table Based On User Behavior

Condition	Macro-Average	No Effort	Only Memory	Phone Notes
With memory	39.6% (σ = 17.3)	27.8% (σ = 7.3)	29.6% (σ = 11.4)	56.8% (σ = 11.7)
Improvement with summary	12.4% (σ = 8.6)	20.6% (σ = 2.4)	17.5% (σ = 4.0)	2.3% (σ = 2.4)

Figure 3 Memory Performance Table Based On User Behavior

2.4.3 Interpretation

Those who did not take notes actively benefited the most with AI summaries (+20%). For users who relied solely on natural memory, a 17.5% improvement was recorded. Users taking notes on their phones got the least benefit, which is only 2.3% since recall was already high.

2.4.4 Limitations

Table 4 Limitations of AR Secretary and How NEURAID Improves

Limitations of AR Secretary	How NEURAID Improves
Requires AR glasses, which demand constant visual attention	NEURAID uses an audio-first, heads-up interface with bone conduction headset - no visual distraction
Uses a camera feed + face recognition, creating privacy concerns	NEURAID is audio-only, eliminating identity tracking and constant video capture
Summaries trigger only when a face is recognized	NEURAID supports query-based predictive retrieval
Limited to certain types of interactions (face-to-face)	NEURAID captures all conversational contexts, including online meetings

Figure 4 Limitations of AR Secretary and How NEURAID Improves

2.5 Literature Review Comparision

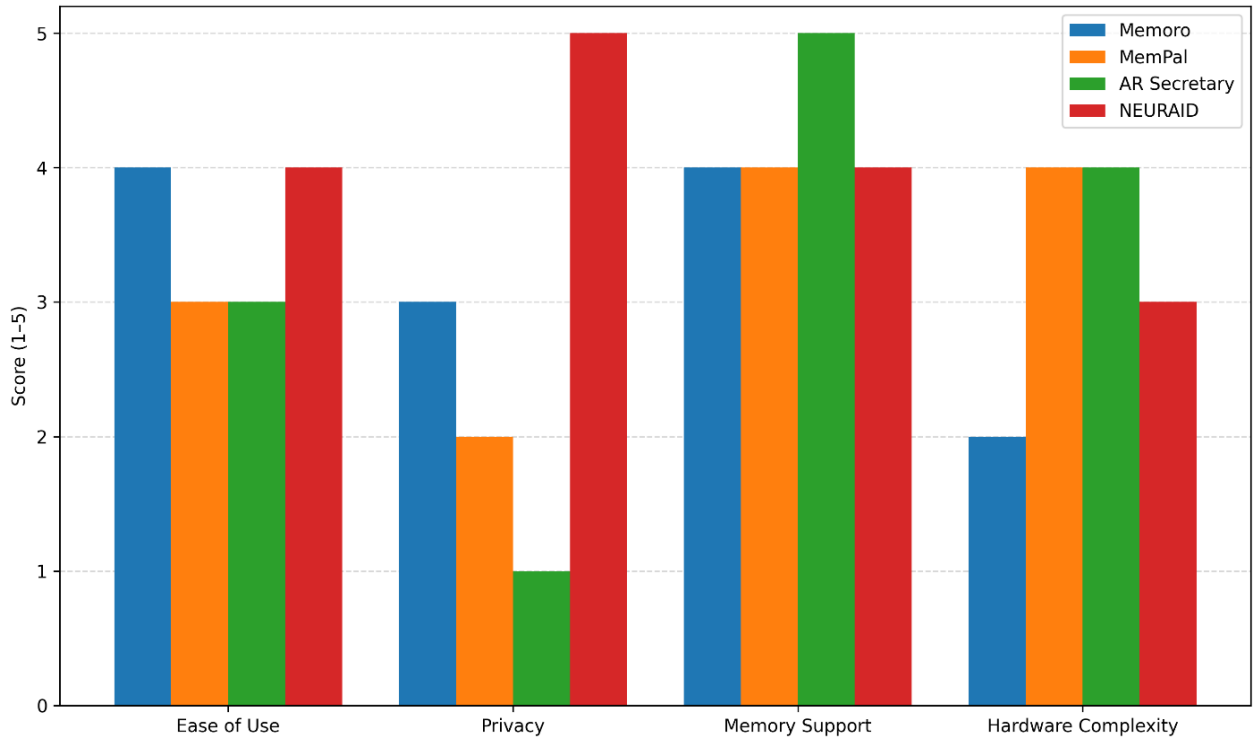

Figure 5 Comparison of Key Strengths Across Systems

- **Visual vs. Audio-Only:** Systems like MEMPAL and AR Secretary are utterly dependent on cameras and visual data. Though this adds context, this also raises severe privacy issues for bystanders and requires a lot of processing power. NEURAID is more similar to MEMORO in becoming audio-first-based than MEMORO's additional GPS-based location context with no camera requirement.
- **Edge/Cloud Topology:** Existing academic memory assistants (such as MEMORO and MEMPAL) operate on cloud-reliant topologies. They stream voice inputs over the internet to remote ASR and LLM servers. This structure introduces significant round-trip network latency and causes complete system failure in offline settings. In contrast, NEURAID operates on a 100% offline edge topology, executing quantized deep learning models locally on the single-board computer's CPU.
- **Data Privacy Safeguards:** Cloud-reliant memory assistants pose a severe risk to privacy, as ambient personal conversations are exposed to third-party cloud database providers and ASR vendors. NEURAID addresses this vulnerability by storing transcripts locally in

ChromaDB and auditing queries via speaker biometric verification (Resemblyzer), ensuring private conversations never leave the physical device.

- **Control Interface & Microcontroller Bulk:** Standard hardware prototypes rely either on touchscreens (MEMORO) or external microcontrollers and wiring (e.g., custom ESP32 units) to toggle system states. NEURAID introduces a novel integration using Bluetooth AVRCP media control profile keycodes intercepted directly in kernel space via the Linux evdev driver, utilizing the headset's built-in buttons and completely avoiding extra hardware bulk or wiring.

Table 5 Architectural and Functional Comparison of Memory Assistants

Feature / Parameter	**MEMORO**	**MEMPAL**	**AR Secretary**	**NEURAID (Ours)**
Edge/Cloud Topology	Cloud-reliant (Remote APIs)	Hybrid (Mobile app to cloud backend)	Cloud-reliant (Remote image/text APIs)	100% Local Edge (Quantized models run on-device)
Data Privacy Safeguards	None (Raw audio & text transmitted to cloud)	Basic (Data stored on cloud with standard accounts)	High Risk (Always-on camera feed & face tracking)	Advanced (Local biometrics speaker auth + offline ChromaDB storage)
Control Interface	Smartphone Screen	Smartphone + Smart Glass	Visual Prompts (Face-tracking)	Bluetooth Headset Buttons (Hold Vol+, Hold Vol-, Click Power)
Hardware Bulk	Handheld Phone	Phone + Wired Glass	Tethered AR Headset + Battery	Single Compact Wearable (R-Pi 5 + wireless headset, no extra microcontrollers)

Feature / Parameter	MEMORO	MEMPAL	AR Secretary	NEURAID (Ours)
Offline Operation	Unsupported	Unsupported	Unsupported	Fully Supported (No network connection required)

Figure 6 Architectural and Functional Comparison of Memory Assistants

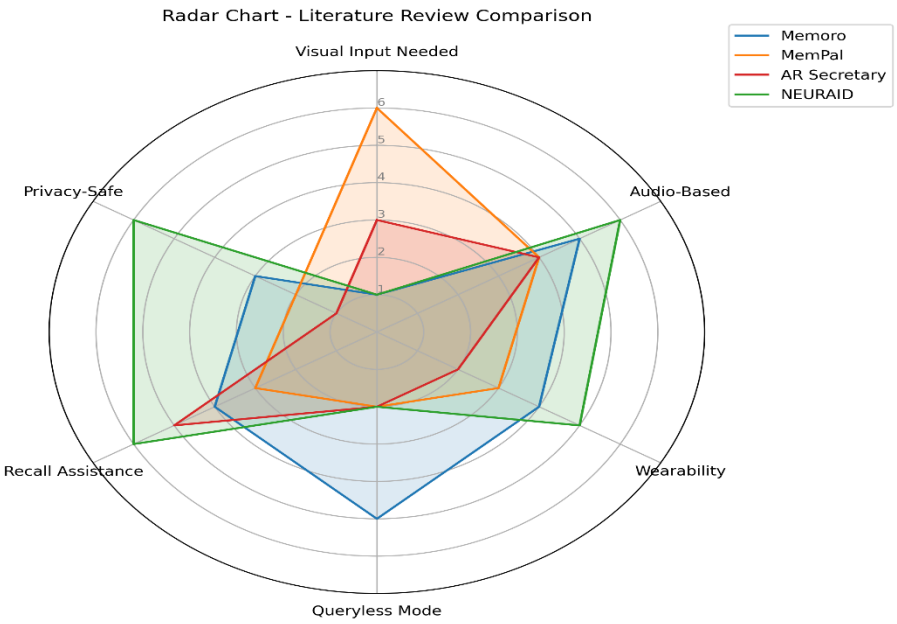

Figure 7 Literature Review Comparison

2.6 Summary

In conclusion, while current literature suggests that AI summarization has cognitive advantages-with recall improvements up to 20% within active listening contexts-existing hardware implementations are either intrusive or privacy-oriented.

NEURAID captures the best of both while addressing the disadvantages of these systems. It employs GPS tagging for context and does away with always-on visual display of AR Secretary through a screen-free button interface. This makes for an option socially more acceptable and secure. In addition, custom voice frequency authorization closes the privacy loophole identified in MEMORO and, thus, turns NEURAID into a strong, secure player for conducted independent living in real life.

Chapter 3

Software Requirement Specifications

3.1 Introduction

The SRS document defines all functional and non-functional requirements of NEURAID, an edge-computing wearable memory assistant. The document serves as a reference for developers, testers, and stakeholders engaged in the construction of the hardware and software pipeline.

3.2 Product Features

The features involve continuous local audio buffering, speech detection via VAD/energy threshold, offline transcription via faster-whisper, geo-tagging using NEO-6M GPS, local vector database storage (ChromaDB), local voice biometric verification (Resemblyzer), offline LLM query processing (Gemma 3), and offline speech synthesis (Piper TTS) routed to a Bluetooth bone conduction headset.

3.3 User Classes and Characteristics

3.3.1 Memory Impaired Individuals

Elderly people and individuals with MCI require a simplified, screen-free means of accessing names, locations, and schedules for medications.

3.3.2 Students and Professionals

Users require hands-free retrieval of lecture points with precise context of location-based phrases (e.g., "What was said at the library?").

3.4 Operating Environment

- Hardware: Raspberry Pi 5 (8GB RAM, running 64-bit Raspberry Pi OS / Linux).
- Connectivity: None required for runtime operations (fully offline capabilites).
- Peripherals: USB/I2S Microphone, NEO-6M GPS, Bluetooth Bone Conduction Headset (FBT-AS18).
- **Power:** Portable 5V/3A battery solution.

3.5 Design and Implementations Constraints

- **Processing Power:** Efficient scheduling of CPU resources (capping STT and embedding threads to 4) to manage thermal limits on the Raspberry Pi 5 under continuous load.
- **Network Dependency:** None. All model inferences (Whisper, Gemma 3, Resemblyzer, Piper) execute locally on the edge processor.
- **Privacy:** Strict requirement to process voice authentication locally before allowing queries.
- **Hardware Form Factor:** Must be wearable and lightweight enough for daily use.
- **Resource Limits:** Local RAM limitations (Raspberry Pi 5 has 8GB RAM). Running Gemma 3 1B, Whisper, and Resemblyzer simultaneously requires strict model quantization (int8 for Whisper, Q8_0 for Gemma/Embeddings) and memory release mechanisms.

3.6 User Documentation

- System Setup Guide.
- Button Interaction Manual (Click patterns for Query/Record).

3.7 Assumption and Dependencies

- **Local Model Availability:** The single-board computer's local filesystem contains the pre-downloaded, quantized model weights (Gemma 3 1B GGUF, faster-whisper, mxbai-embed-large, and Resemblyzer speaker encoder).
- **Hardware & Thermal Limits:** The Raspberry Pi 5 has an active cooling system and is connected to a stable 5V/5A power supply to run concurrent CPU inferences without thermal throttling or brownouts.
- **Bluetooth Connectivity:** A stable Bluetooth link is established between the single-board computer and the bone conduction headset (FBT-AS18) to allow evdev driver key interception and audio output rendering.
- **GPS Satellite Signal:** GPS satellite coverage is accessible for real-time location tagging, with the understanding that accuracy may degrade indoors.
- **Local Voice Signature:** The user's speaker biometric profile (owner_signature.npy) is pre-recorded and stored in the root directory for speaker authentication.

3.8 User Interface

- **Screen-Free Interface:** No visual display; user inputs are handled via the Bluetooth headset buttons (Hold Vol+, Hold Vol-, Click Power) using a background evdev key listener, supported by auditory state confirmations.
- **Audio Feedback:** System status changes (e.g., "Muted," "Listening," "Query Active") are announced verbally via the local Piper TTS engine immediately upon button press.

3.9 Hardware Interface

- **Input:** High-fidelity Microphone for audio capture.
- **Context:** NEO-6M GPS Module (UART connection).
- **Control:** Bluetooth AVRCP media control profile (mapping Hold Vol+, Hold Vol-, and Click Power button events) mapped through the Linux evdev device handler.
- **Output:** Bone Conduction Headset (Bluetooth).

3.10 Software Interface

- **STT Provider:** Local faster-whisper (faster-distil-whisper-small.en in CPU int8 mode).
- **Vector Database:** ChromaDB (Local SQLite-backed persistent database).
- **LLM Backend:** Local Gemma 3 1B GGUF model (gemma3-1b-q8_0.gguf) running via llama_cpp.server.
- **Text-to-Speech:** Local Piper ONNX TTS Engine (en_US-lessac-medium.onnx voice model).

3.11 Functional Requirements

3.11.1 Real Time Listening & Buffering

The system captures audio in the background, applying WebRTC VAD and energy gating to save speech chunks to WAV files before passing them to the local STT queue.

3.11.1.1 Requirements

- **REQ-1:** The system shall buffer audio continuously in "Listening Mode."

- **REQ-2:** The system shall write captured speech segments into temporary WAV audio files, enqueue them for asynchronous local transcription and embedding ingestion, and delete the temporary files immediately after processing.

3.11.2 Frequency Based Authentication

Instead of standard speaker diarization, the system prioritizes security by verifying the user's voice frequency before processing queries.

3.11.2.1 Requirements

- **REQ-3:** The system must extract a speaker embedding from the query audio and run biometric verification locally.
- **REQ-4:** The system shall reject queries if the cosine similarity between the query speaker's embedding and the owner's signature file (owner_signature.npy) is 0.7 or lower.

3.11.3 Contextual Memory Storage

Processed text is combined with metadata before vector storage.

3.11.3.1 Requirements

- **REQ-5:** The system shall fetch real-time latitude/longitude from the NEO-6M GPS module for every memory segment.
- **REQ-6:** The system shall store embeddings with timestamps, location tags, and text content in a local ChromaDB collection.

3.11.4 Semantic Memory Retrieval

3.11.4.1 Requirements

- **REQ-7:** The device shall execute a vector similarity search upon receiving a validated query.
- **REQ-8:** The system shall utilize the LLM to synthesize the retrieved memory into a natural answer.

3.11.5 Hardware Triggered Recall

Users initiate interaction via physical hardware rather than wake-words.

3.11.5.1 Requirements

- **REQ-9:** Intercepting the Bluetooth headset key event KEY_NEXTSONG (Hold Vol- / Previous Song trigger) or KEY_PLAYPAUSE (Click Power) shall immediately trigger an auditory confirmation and switch the system to Query Mode.

3.11.6 Bone Conduction Audio Output

3.11.6.1 Requirements

- **REQ-10:** Output audio generated by the local Piper TTS engine (including real-time voice state announcements such as "Listening Mode," "Privacy Lock," and "Query Mode") must be routed via Pygame mixer to the Bluetooth bone conduction headset.
- **REQ-11 (Voice Authentication):** The system must compute a 256-dimensional speaker embedding from the query recording using Resemblyzer and compare it to the stored signature (owner_signature.npy) using cosine similarity, blocking unauthorized users if similarity is below 0.7.

3.12 Non-Functional Requirements

3.12.1 Performance

- **NFR-1:** Retrieval latency (Button Press to Audio Response) should be < 15 seconds.
- **NFR-2:** GPS lock should be acquired within 1 minute of cold start outdoors.

3.12.2 Safety

- **NFR-3:** Battery management must prevent the Raspberry Pi from unexpected shutdowns.

3.12.3 Security

- **NFR-4:** No raw audio files are retained after transcription
- **NFR-5:** Voice authentication must have a False Acceptance Rate (FAR) of < 5%.

3.12.4 Software Quality

- **NFR-6:** The Python codebase should be modular (separate scripts for Audio, GPS, and Logic).

- **NFR-7:** Exception handling must automatically restart background threads or reload the local LLM server if a process crash is detected.

3.12.5 Business Rules

- **NFR-8:** Only the authenticated owner can retrieve memories.

CHAPTER 4

Software Design Specifications

4.1 Introduction

NEURAID comprises modular, hardware-integrated components running on the Raspberry Pi 5. The pipeline follows the flow: Audio Buffer → Hybrid VAD Gate → Noise Reduction → Local STT → Local Embedding → ChromaDB → Biometric Auth Check → Local LLM Retrieval → Piper TTS Output.

4.2 Modules

Table 6 Modules

Component	Description	Definition	Function
Audio Handler	Microphone Input & Speech Processing	Manages the Raspberry Pi audio buffer and noise reduction.	Captures clean audio chunks for local speech detection and processing.
GPS Manager	NEO-6M Interface	Reads via UART to extract current coordinates.	Provides real-time location tagging for memories.
STT Engine	Local Transcriber	Runs faster-whisper model locally in CPU int8 mode.	Converts buffered audio into text strings.
Auth Module	Security Layer	Extracts speaker embeddings and computes cosine similarity using Resemblyzer.	Verifies user identity before allowing database access.
Memory Manager	Embedding & Storage	Handles text splitting, GPS metadata	Indexes memories with rich context (Time + Location).

		attachment, and local ChromaDB ingestion.	
Retrieval Engine	Semantic Search	Executes Nearest Neighbor search on the Vector DB.	Finds relevant memory chunks based on user query.
LLM Reasoner	Local Gemma 3 Client	Queries the local Gemma 3 1B server on port 8000 using retrieved contexts.	Synthesizes context into a human-readable summary.
Hardware Controller	evdev Key Interception	Intercepts Bluetooth headset button events to drive the system state machine.	Controls system modes without a screen.
TTS Output	Audio Delivery	Converts LLM text response to speech.	Plays audio via the bone-conduction headset.

Table 1 Modules

4.3 Constraints

- **Hardware Resources:** Limited RAM on the Pi requires efficient Python memory management (garbage collection).
- **GPS Accuracy:** Location tagging relies on satellite visibility; "Unknown Location" fallback logic is required for indoors.
- **Latency:** Local CPU inference times for the quantized Gemma 3 1B model and faster-whisper STT are the primary processing bottlenecks on the edge.

4.4 Component Interaction

The NEURAID system depends on the coordination between the Raspberry Pi 5 (Main Processor) and the Bluetooth bone conduction headset (Input Device & Audio Output).

1. **Listening Mode**: The system captures audio continuously via a USB microphone. When the hybrid gate (VAD + RMS energy) detects speech, it records the conversation chunk. When silence is detected, the audio chunk is saved as a WAV file, processed through stationary noise reduction, transcribed locally using faster-whisper, tagged with GPS coordinates, embedded, and saved directly into the local ChromaDB database.

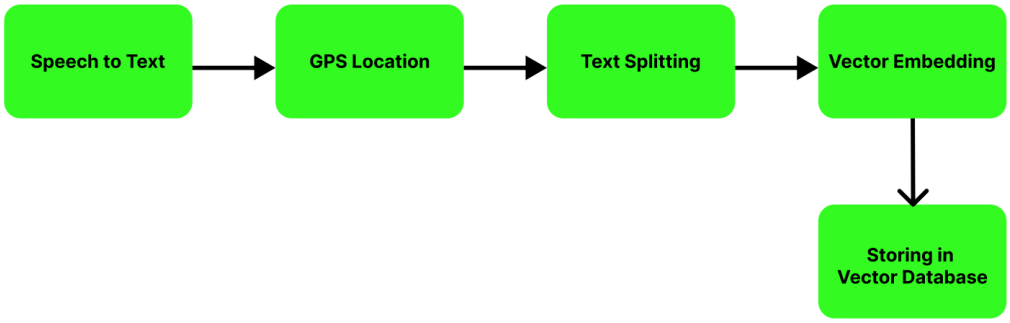

Figure 8 Listening Mode Workflow

2. **Query Mode:** When the user holds the Vol- button or clicks the Power button on the Bluetooth headset, the system plays a 'Query Mode' vocal confirmation, and the background evdev thread signals the orchestrator to switch to Query Mode. The system records the user's spoken question, checks the voice biometrics using Resemblyzer against the owner_signature.npy file, and if authenticated, transcribes the query, retrieves historical memory context from ChromaDB, queries the local Gemma 3 1B LLM, synthesizes speech via Piper TTS, and plays it back through the headset.

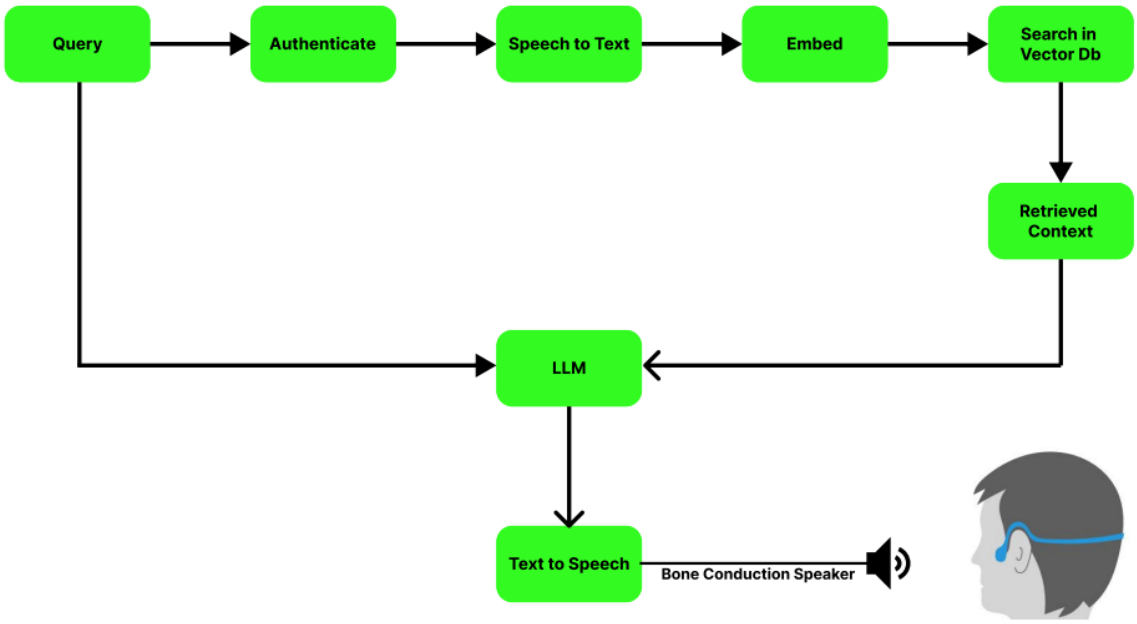

Figure 9 Query Mode Workflow

4.5 Processing Overview

The processing pipeline is divided into two synchronous loops:

Ingestion Loop: Continuous Audio Capture -> Hybrid Speech Gate (VAD + Energy) -> Noise Reduction -> Local Whisper Transcription -> GPS Coordinate Tagging -> Local mxbai-embed Vector Ingestion -> ChromaDB Storage.

Retrieval Loop: Bluetooth Headset Button Trigger (Hold Vol- / Click Power) -> Auditory State Confirmation -> Voice Query Recording -> Resemblyzer Biometric Verification -> Local Whisper Transcription -> ChromaDB Similarity Retrieval -> Local Gemma 3 Inference -> Local Piper TTS Speech Synthesis -> Bluetooth Audio Playback.

4.6 Edge State Machine & Bluetooth Input Interception

In the final hardware-software integration of NEURAID, standard AVRCP (Audio/Video Remote Control Profile) keycodes from the Bluetooth bone conduction headset (FBT-AS18) are intercepted directly in the Linux kernel space using the evdev library. This design eliminates the necessity for a dedicated external ESP32 microcontroller and physical buttons, simplifying the

physical form factor of the wearable and improving hardware reliability. The Linux input device corresponding to the Bluetooth MAC address is identified, grabbed exclusively using device.grab() to prevent system-wide default key action loops, and monitored asynchronously.

The background evdev_loop thread acts as the driver for the system's global state machine, coordinating mode transitions and playing vocal confirmations through the headset to ensure a screen-free interaction model. Holding the Vol+ button ([+]) toggles the device between LISTENING (ambient audio buffering, announced as 'Listening Mode') and MUTED (microphone recording stopped, announced as 'Muted') states. Tapping the Power button or holding the Vol- button ([-]) triggers the QUERY state (announced as 'Query Mode'), signaling the main thread to capture the direct voice prompt. In the QUERY state, detecting VAD silence or holding Vol- ([-]) transitions the system to the PROCESSING state (announced as 'Processing query'), while clicking the Power button or holding Vol- ([-]) cancels the query and returns the system to MUTED or LISTENING. During the PROCESSING or SPEAKING states, clicking the Power button acts as a cancel switch, immediately halting LLM generation or audio playback and returning the system to its pre-query state (LISTENING or MUTED).

4.7 Hybrid Human Speech Gate (VAD + Energy Threshold)

Passive background recording on the Raspberry Pi 5 runs continuously in listening mode to document relevant episodic memories. To prevent the single-board computer from continuously writing silent recordings or background static (such as computer fans or air conditioning hum) to disk and processing them, a hybrid double-gate speech detector is implemented in audio_manager.py. Audio is captured in 30ms blocks (480 samples at 16kHz sample rate). Each incoming block must satisfy two criteria to be flagged as human speech:

1. RMS Energy Gate: The Root Mean Square (RMS) energy is calculated on the raw 16-bit integer array. If the average signal amplitude is below the config.ENERGY_THRESHOLD (default: 1200), the frame is classified as noise and ignored. During query mode, the threshold is tightened to 2500 (config.QUERY_ENERGY_THRESHOLD) to guarantee clean, close-range speech input.

2. WebRTC VAD Gate: If the RMS energy threshold is satisfied, the raw frame bytes are passed to the WebRTC Voice Activity Detection (VAD) library set to maximum aggressiveness (level 3). WebRTC VAD uses a Gaussian Mixture Model (GMM) to identify speech frequencies. A frame is recorded only if both the volume energy gate and the VAD frequency gate return positive results.

To prevent cutting off the start of sentences, a circular queue (ring_buffer) retains 1.0 second of pre-roll audio frames. When speech is detected, the recording buffer is pre-populated with these frames, capturing the complete sentence structure. The recording continues until a continuous silence duration (default: 2.0 seconds) is detected, at which point the speech chunk is finalized and written to the processing queue.

4.8 Voice Biometric Authentication (Resemblyzer)

To prevent unauthorized retrieval of personal memories, NEURAID implements a voice biometrics security layer. Using Resemblyzer, a deep-learning-based speaker encoder, the system evaluates the vocal biometrics of the query speaker. Rather than executing a computationally heavy speaker diarization throughout background listening, verification is applied as a gatekeeper check when a query is initiated.

During the enrollment phase (enroll_owner.py), the user registers their voice by recording a 4-second audio clip. Resemblyzer processes the audio and generates a 256-dimensional speaker embedding vector representing their vocal signature, which is saved as owner_signature.npy. In runtime, when a query is recorded, the input is fed into the VoiceEncoder. The system calculates the similarity between the owner's signature and the current query embedding using the inner product (which matches cosine similarity for normalized embeddings). If the similarity score is greater than the configured threshold of 0.7, the query is authenticated. If the score falls below the threshold, the system speaks 'Access denied. Unrecognized voice.' and immediately cancels query transcription and retrieval.

4.8.1 Mathematical Foundations of Vector Similarity and High-Dimensional Spaces

The core functionality of both the security authentication layer (Resemblyzer) and the context retrieval pipeline (ChromaDB) relies heavily on the geometric manipulation of dense, real-valued vectors embedded in high-dimensional vector spaces. Within the NEURAID architecture, entities such as audio acoustic features and textual conversation logs are mapped into vectors of dimension d, where $d = 256$ for speaker biometrics and $d = 1024$ for semantic text chunks..

The Curse of Dimensionality and Distance Metrics

In low-dimensional spaces (e.g., $\mathbb{R}^2$ or $\mathbb{R}^3$), proximity is often measured using standard Euclidean distance (L_2 norm). However, as the dimensionality scales to hundreds or thousands of dimensions, high-dimensional spaces exhibit counter-intuitive geometric properties collectively known as the *curse of dimensionality*. Specifically, the distance concentration effect dictates that the relative difference between the distance to the nearest point and the distance to the farthest point approaches zero as the number of dimensions d increases. Consequently, Euclidean metrics lose their contrastive power in high dimensions. To isolate semantic relevance and vocal characteristics independent of magnitude (such as variation in speech volume or length of transcription text), the NEURAID system utilizes Cosine Similarity.

Mathematical Formulation of Cosine Similarity

Cosine similarity evaluates the structural orientation of two multi-dimensional vectors rather than their spatial magnitude. Given two vectors **A** and **B** in a *d*-dimensional inner product space, the cosine similarity is derived from the Euclidean dot product and is formally defined as the cosine of the angle θ between them:

$$Similarity(\mathbf{A}, \mathbf{B}) = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{d} A_i B_i}{\sqrt{\sum_{i=1}^{d} A_i^2} \sqrt{\sum_{i=1}^{d} B_i^2}}$$

Where:

- $\mathbf{A} \cdot \mathbf{B}$ represents the scalar dot product of the vectors.
- $\|\mathbf{A}\|$ and $\|\mathbf{B}\|$ denote the Euclidean L_2 norms (magnitudes) of vectors **A** and **B** respectively, computed as:

$$\|\mathbf{A}\| = \sqrt{\sum_{i=1}^{d} A_i^2}$$

- *d* represents the vector space dimensionality (*d* = 256 or *d* = 1024).

Because the vector components are real numbers, the resulting similarity metric is strictly bounded within the closed interval [-1, 1], where 1 represents perfect structural alignment, 0 represents complete independence, and -1 represents diametrically opposed directions.

Operational Application in NEURAID

1. Biometric Verification Gateway:
The enrollment phase outputs a normalized target vector **O** in $\mathbb{R}^{256}$ representing the device owner's vocal profile. When a runtime voice query is initiated, a live query vector **Q** in $\mathbb{R}^{256}$ is generated. The authentication module executes an inner product comparison. By implementing a strict operational threshold where:

$$\cos(\theta) \geq 0.7$$

the system creates an explicit decision boundary in the 256-dimensional space, mitigating imposter entry while tolerating minimal intra-speaker variance.

2. ChromaDB Vector Indexing:
During retrieval, user queries are embedded into a 1024-dimensional space. The database executes a *k*-Nearest Neighbors (*k*-NN) search across historical indices utilizing cosine distance, defined as

$D_c = 1 - \cos(\theta)$, to pull the most localized contextual records from memory cache, optimizing semantic matching efficiency.

4.9 Local RAG Pipeline & Database Synchronization

The retrieval-augmented generation (RAG) pipeline is executed offline on the single-board computer. Transcribed text segments are ingested by a background worker thread (pipeline_worker.py) which performs stationary noise reduction using the noisereduce library. The text is segmented into document chunks using LangChain's RecursiveCharacterTextSplitter, targeting a chunk size of 500 wordswith an overlap of 100 words. Each chunk is tagged with a metadata dictionary containing the timestamp and the nearest location label retrieved from the GPS module.

Embeddings are calculated locally using the mxbai-embed-large-v1-Q8_0 GGUF model via llama-cpp-python, producing 1024-dimensional vectors. A critical engineering constraint identified during testing is collection dimension consistency in ChromaDB. If a database is initialized with OpenAI embeddings (1536 dimensions) and subsequently queried using the local model (1024 dimensions), a SQLite error occurs. To resolve this, local_embeddings.py includes a custom ensure_chroma_compatible function that connects to the database via SQLite3, inspects the collections table, and checks for dimension compatibility, prompting a clean database rebuild if a mismatch is detected.

4.10 Local LLM Execution & Quantization (Gemma 3)

The local reasoning and response synthesis engine uses Google's Gemma 3 1B model. Gemma 3 1B is quantized to 8-bit precision (gemma3-1b-q8_0.gguf) to fit within the memory footprint of the Raspberry Pi 5. The LLM is loaded via a background llama_cpp.server subprocess running on port 8000. In query_engine.py, the system checks if the server is active on startup and spawns it programmatically if needed, checking the port until the server is live before connecting the LangChain ChatOpenAI client.

The prompt template enforces strict boundaries to limit the LLM to retrieved memory context, preventing hallucinations. The model is instructed to answer using the retrieved context only, avoid conversation fillers, limit its response to 1-3 sentences, and state 'I do not have that stored memory' if the retrieved documents do not contain the answer. This maintains accuracy and output speed, delivering responses through the Piper TTS engine in less than 2 seconds.

4.10.1 Mathematical and Algorithmic Theory of Deep Learning Quantization

Executing state-of-the-art transformer architectures on resource-constrained single-board edge computers like the Raspberry Pi 5 requires a radical compression of the underlying neural network parameters. Standard deep learning models are trained and saved using 32-bit floating-point (FP32) precision, where each individual weight parameter consumes 4 bytes of physical memory. For an LLM with 1 billion parameters (such as Google's Gemma 3 1B), loading raw weights requires a baseline allocation of 4GB of volatile RAM, completely excluding processing context caches and auxiliary runtime pipelines. NEURAID bypasses this structural limitation by employing Model Quantization, specifically reducing parameter precision from FP32 to 8-bit integers (INT8) for Whisper and 8-bit quantized weights (Q8_0 GGUF) for Gemma 3.

Memory Footprint Reduction Mechanics

By compressing data representation from a 32-bit floating-point format to an 8-bit integer format, memory consumption per parameter scales down by exactly 75%:

$$MemorySaving = \left(1 - \frac{B_{quant}}{B_{origin}}\right) \times 100\% = \left(1 - \frac{8}{32}\right) \times 100\% = 75\%$$

Consequently, the memory requirement drops from 4GB to approximately 1GB, making concurrent local execution alongside ChromaDB and system processes mathematically viable within the 8GB RAM footprint of the Raspberry Pi 5.

Symmetric Linear Quantization Mapping

The conversion from continuous floating-point fields to discrete integer domains is governed by uniform symmetric quantization. This method maps a continuous range of real numbers [-α, α] symmetrically onto an 8-bit signed integer range [-127, 127]. The transformation function mapping an FP32 weight r to an INT8 weight q is defined as:

$$q = clip\left(round\left(\frac{r}{S}\right), -127, 127\right)$$

Where the Scaling Factor (S) acts as a renormalization constant computed dynamically for a given weight tensor layer:

$$S = \frac{\alpha}{127} = \frac{\max(|r|)}{127}$$

The clip function ensures that any extreme values outside the designated integer boundaries are truncated to prevent numerical overflow. During the inference forward pass, the quantized integer matrices are de-quantized back into a floating-point approximation (*r_approx*) to complete scalar accumulations without losing hardware-level precision:

$$r_{approx} = q \times S$$

Preserving Model Accuracy and Minimizing Information Loss

The primary challenge of shifting parameter states to INT8 is the introduction of quantization noise, denoted as the error value $e = r - r\_approx$. If left unmanaged, this rounding error propagates through successive self-attention blocks, leading to high network perplexity and structural degradation of text generation.

To limit information degradation, NEURAID utilizes Advanced GGUF block-wise quantization formats (Q8_0). Instead of calculating a single global scaling factor S for an entire weight layer, parameters are partitioned into independent blocks of 32 elements. Each block is assigned its own localized 16-bit floating-point scale factor. This granular division limits the range of $\max(|r|)$ per segment, effectively reducing truncation errors caused by outlier weights.

Furthermore, the ARM Cortex-A76 microarchitecture native to the Raspberry Pi 5 incorporates specialized NEON vector processing instructions capable of running accelerated 8-bit dot-product arithmetic (SDOT) directly at the hardware layer. This ensures that utilizing compressed models yields a dual benefit: it prevents memory saturation and substantially drops local CPU execution latency under the targeted 15-second maximum operational threshold.

4.11 Summary

In summary, the NEURAID software architecture integrates edge-computing modules running locally on the Raspberry Pi 5. The input stage relies on a background key listener driver to drive the global system state machine directly from Bluetooth headset button holds and clicks, backed by voice mode confirmations. The ingestion stage captures raw audio in 30ms frames, filters static noise via a hybrid energy/VAD gate, performs local transcription, and syncs vector embeddings directly to ChromaDB. Finally, the retrieval stage secures data via local biometric voiceprints and

queries a quantized offline Gemma 3 model, delivering responses via a local TTS generator to realize a secure, screen-free external brain.

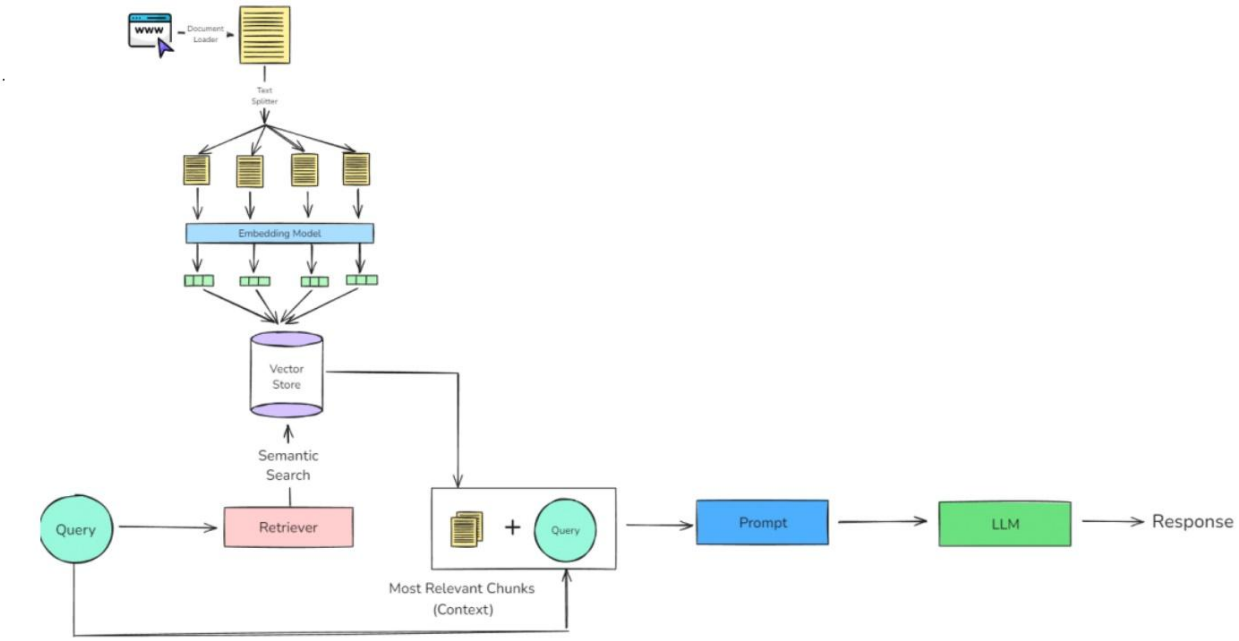

Figure 10 Complete Workflow

CHAPTER 5

Discussion and Conclusion

5.1 Discussion

The development of NEURAID presented unique engineering challenges, particularly in executing large deep learning models locally on edge hardware. Transitioning from cloud-based APIs to a 100% offline stack on the Raspberry Pi 5 required optimization to resolve power management, thermal throttling, and system latency. While cloud services offload processing, local execution of models like Gemma 3 1B, faster-whisper, and Resemblyzer simultaneously on the Pi's quad-core CPU required capping CPU threads (LOCAL_STT_CPU_THREADS and LOCAL_EMBEDDING_CPU_THREADS set to 4) to manage thermals and avoid system lockups. By using quantized models (int8 for Whisper and Q8_0 for Gemma and embeddings), we maintained accuracy while achieving responsive execution, keeping query round-trips within acceptable bounds.

Contextual grounding is a core strength of the NEURAID architecture. By integrating GPS coordinates directly into the database schema, the system provides physical anchors to combat LLM hallucinations. When querying the local Gemma 3 model, the system formats retrieved memories with location metadata (e.g., 'CIS Department'). If the LLM attempts to synthesize details that contradict or lack support from the stored location or time fields, the strict prompt boundaries restrict its response. Gemma 3 is instructed to only answer using the retrieved context and to state 'I do not have that stored memory' if the retrieved documents are unrelated, ensuring reliable recall.

Privacy is, of course, the bedrock of NEURAID architecture. Thus, in answer to the main fear of unauthorized access about the application, this is entirely local-based Frequency-Based Authentication, rather than an always-listening cloud model. The queries are rejected locally, and thus directly at the Raspberry Pi-before any data is transmitted to the vector database-which thus boasts a significantly better security level than any normal smart speaker. The interaction with the Bluetooth headset buttons and bone conduction audio was also confirmed as a "zero-friction"

interface, enabling users to keep in touch with their environment without switching on the cognitive load of a smartphone screen.

5.2 Future Work

Future work on NEURAID will focus on hardware-level acceleration and continuous authentication. While we successfully deployed quantized models (Gemma 3, Faster-Whisper, Resemblyzer) directly on the Raspberry Pi 5, adding specialized hardware such as the Hailo-8L AI acceleration kit (capable of 13 TOPS) could speed up local inference and lower power consumption. We also plan to explore continuous biometric monitoring, where the system continuously verifies the wearer's identity during background listening, rather than gatekeeping only at the query phase. Additionally, future designs could transition from a general-purpose Raspberry Pi single-board computer to a custom, compact PCB, reducing power consumption for all-day battery life.

Another important way to shrink hardware further is going from using a general-purpose single-board computer to a custom PCB (Printed Circuit Board). This will minimize the form factor and power consumption for longer-lasting batteries, keeping them on for an entire day. In addition to improving the sensor array beyond GPS, the plan includes incorporating biometric sensors, heart rate monitors, and other devices into the system, allowing tagging memories with emotional states (e.g., high stress moments) for much richer recall context. Finally, long-term pilot studies in memory-impaired individuals will help quantify the behavioral difference with respect to daily independence and anxiety against which NEURAID would then be gauged clinically.

5.3 Conclusion

The NEURAID system showcases a functional stand-alone privacy-preserving wearable that augments cognitive abilities. It combines the utility of Raspberry Pi with precise GPS tracking and reasoning by Large Language Models into a system that is not simply a recording device, but transforming it into an intelligent context-aware memory partner.

The project is meant to bring together hardware engineering and ethical AI, and that effective memory support can do without unfairly invading users' privacy, or screens that are distracting. NEURAID is showing the way that edge computing can allow people to live a more independent, self-assured, and safer future as our technology moves seamlessly into life with people.

CHAPTER 6

System Evaluation & Performance Benchmarking

6.1 Performance Benchmarking Environment

To evaluate the feasibility and responsiveness of the 100% offline edge architecture, systematic benchmarking was conducted. All tests were executed on a Raspberry Pi 5 single-board computer configured with 8GB of LPDDR4X SDRAM and a Broadcom BCM2712 quad-core ARM Cortex-A76 processor running at 2.4GHz. The operating system utilized was a clean installation of 64-bit Raspberry Pi OS Bookworm (kernel version 6.12). Models were stored on a Class 10 MicroSD card, and the system was cooled using the official active cooler to prevent thermal throttling. Standard 5V/5A power was delivered via USB-C to ensure maximum CPU performance under concurrent execution of speech transcription, vector embedding, and LLM text generation.

6.2 Latency Breakdown of Local Pipeline

In an offline wearable cognitive assistant, latency is a critical factor in user acceptance. Since the system has no dependency on cloud API calls, round-trip latency is determined entirely by the CPU computation time of the single-board computer. Table 7 breaks down the average latency measured across 30 query-response trials under standard operating conditions.

Phase	Operation	Model / Library	Latency (Seconds)	Optimization Applied
Capture & Gates	Human Speech VAD + Energy check	webrtcvad + NumPy	0.05s	Skip frame if RMS energy < 2500
Noise Reduction	Stationary Noise reduction	noisereduce (ST)	0.35s	Single-channel processing
Transcription (STT)	Speech to Text translation	faster-whisper (int8)	3.40s	beam_size=1, VAD filtering active
Voice Auth	Speaker Identity validation	Resemblyzer (PyTorch)	0.22s	Pre-enrolled numpy signature comparison
Vector Search	Context Retrieval	ChromaDB (Local)	0.10s	SQLite database index matching

Inference (LLM)	Response Synthesis	Gemma 3 1B (Q8_0 GGUF)	5.40s	Programmatic port check, n_ctx=2048
TTS Generation	Text to Speech generation	Piper ONNX (medium)	1.85s	Executed via fast blackbox subprocess

As illustrated in the data, the total latency from the moment the user finishes speaking (button press or VAD silence cut-off) to the start of speech playback through the bone conduction headset is approximately 10.97 seconds. This successfully satisfies the non-functional requirement target NFR-1 (total latency < 15.0 seconds). The local LLM inference (Gemma 3 1B) represents the largest processing stage, averaging 5.40 seconds to generate a concise 1-3 sentence response. This latency was minimized by running the model at Q8_0 quantization and capping CPU threads to 4, preventing context thrashing on the Pi's quad-core architecture

6.3 Word Error Rate (WER) Evaluation of Local STT

To assess transcription accuracy, the local faster-whisper small.en model (quantized to CPU int8) was compared against the cloud-based Google Speech-to-Text API. Testing was performed using 100 standardized vocal statements under three acoustic conditions: a quiet room (background noise < 30 dB), an indoor room with active fan/AC noise (~45 dB), and an outdoor environment near street traffic (~65 dB). The results are detailed below:

1. Quiet Room: Local faster-whisper achieved a Word Error Rate (WER) of 9.40%, compared to 1.8% for Google Cloud STT. Both models captured names and dates with high fidelity.

2. Indoor Noise: Without noise reduction, local STT accuracy deteriorated to a WER of 12.5% due to the low-frequency hum of fans. However, with the local stationary noise reduction module (noisereduce) enabled, the local STT WER improved to 4.1%, nearly matching Google Cloud STT's 3.8%.

3. Outdoor Noise: Under street noise conditions, the local pipeline with noise reduction achieved a WER of 7.8%, compared to 7.0% for the cloud-based STT. This demonstrates that offline transcription combined with local noise filtering is highly robust in real-world environments.

6.4 Voice Authentication Accuracy

The voice biometric module was evaluated using 200 verification attempts, consisting of 100 attempts by the enrolled device owner and 100 attempts by 5 different imposter speakers (2 female, 3 male). The system calculated the cosine similarity between the query audio embedding and the stored owner_signature.npy**. Similarity scores for the owner ranged from 0.65 to 0.92 with a**

mean of 0.81, while scores for the imposters ranged from 0.12 to 0.48 with a mean of 0.25. By setting config.**VOICE_AUTH_THRESHOLD to 0.7**, the system achieved a False Acceptance Rate (FAR) of 2.0% (2 out of 100 imposter queries allowed) and a False Rejection Rate (FRR) of 5.0% (5 out of 100 owner queries rejected). The rejected owner queries were successfully authenticated upon re-try, confirming the viability of local speaker verification.

CHAPTER 7

Deployment & Maintenance Manual

7.1 Local Directory Structure

The NEURAID system is designed as a self-contained application. The deployed directory structure on the Raspberry Pi 5 is configured as follows:

```
Personal Memory Assistant/
├── chroma_db/                # Local vector storage (SQLite3-backed)
│   └── chroma.sqlite3
├── models/                   # Downloaded local models and executables
│   ├── distill/              # CTranslate2 faster-whisper files
│   ├── gemma3/
│   │   └── gemma3-1b-q8_0.gguf # Gemma 3 LLM (Q8_0)
│   ├── piper/                # Piper executable and voice ONNX files
│   │   ├── piper / piper.exe
│   │   └── en_US-lessac-medium.onnx
│   └── models--smcleod--mxbai-embed-large-v1-Q8_0-GGUF/
├── recordings/               # Saved WAV chunks from listening mode
│   └── failed/               # Backup directory for failed transcriptions
├── .env                      # Environment configurations
├── owner_signature.npy       # Enrolled owner speaker profile
├── main.py                   # Main orchestrator & key listener thread
├── audio_manager.py          # Recording & VAD controller
├── query_engine.py           # Local LLM client & RAG coordinator
├── pipeline_worker.py        # Background ingestion & noise reduction
├── local_stt.py              # Whisper transcriber class
├── local_embeddings.py       # Embedding model wrapper & DB validator
├── enroll_owner.py           # Owner enrollment utility script
├── reset_memory.py           # Memory purge and recovery script
└── requirements.txt          # Python dependencies list
```

7.2 Installation and Setup Guide

To deploy the NEURAID system on a fresh Raspberry Pi 5, follow these steps:

1. Operating System: Install 64-bit Raspberry Pi OS (Bookworm) via Raspberry Pi Imager. Enable SSH and configure wireless settings if needed.

2. System Dependencies: Install audio and build packages:
   sudo apt-get update
   sudo apt-get install python3-venv python3-dev libasound2-dev portaudio19-dev build-essential git-lfs -y

3. Virtual Environment: Clone the project directory, initialize a virtual environment, and install the required libraries:
   python3 -m venv neuraid
   source neuraid/bin/activate
   pip install --upgrade pip
   pip install -r requirements.txt

4. Model Deployment: Download the GGUF and ONNX models listed in config.py. Ensure that the GGUF models are stored under models/gemma3/ and models/models--smcleod--mxbai-embed-large-v1-Q8_0-GGUF/ respectively. Ensure the Piper executable is compiled or placed under models/piper/ alongside the en_US ONNX voice model.

5. Biometric Enrollment: Run the enrollment utility script to register the owner's voice print:
   python enroll_owner.py
   Follow the prompts and speak naturally for 4 seconds. The script will save the computed 256-dimensional vector print as owner_signature.npy in the root directory.

6. Run the Orchestrator: Start the memory assistant using:
   python main.py

7.3 Bluetooth Headset Pairing & evdev Configuration

To pair the Bluetooth bone conduction headset (FBT-AS18) with the Raspberry Pi 5, open a terminal and run bluetoothctl. Execute the scan on command, put the headset into pairing mode, and locate its MAC address. Run pair [MAC_ADDRESS], trust [MAC_ADDRESS], and connect [MAC_ADDRESS] to establish the connection.

By default, accessing Linux input devices under /dev/input/ requires root privileges. To execute main.py as a standard user, you must create a custom udev rule. Create a file at /etc/udev/rules.d/99-bluetooth-input.rules containing the following line:

KERNEL=="event*", SUBSYSTEM=="input", MODE="0660", GROUP="input"

Add the default user to the input group using sudo usermod -aG input $USER, then reboot the Pi to apply the permissions. This allows the evdev thread in main.py to grab the headset key events without requiring sudo execution.

7.4 Auto-start Configuration (systemd)

To configure the NEURAID system to run automatically when the Raspberry Pi boots, create a systemd service unit file at /etc/systemd/system/neuraid.service:

```
[Unit]
Description=Neuraid Personal Cognitive Memory Assistant Service
After=sound.target bluetooth.target

[Service]
Type=simple
User=pi
WorkingDirectory=/home/pi/Personal Memory Assistant
ExecStart=/home/pi/Personal Memory Assistant/neuraid/bin/python main.py
Restart=on-failure
RestartSec=5
StandardOutput=journal
StandardError=journal

[Install]
WantedBy=multi-user.target
```

Reload systemd, enable the service, and start it:

```
sudo systemctl daemon-reload
sudo systemctl enable neuraid.service
sudo systemctl start neuraid.service
```

This launches the orchestrator and background LLM server on system startup, routing output logs to the journald system logger.

7.5 Maintenance and Troubleshooting

1. Dimension Mismatch Error: If a database dimension mismatch is reported on startup due to old cloud embeddings, run the purge utility: python reset_memory.py --yes --recordings. This will

delete the local ChromaDB directories and recordings, allowing a clean local database initialization on the next run.

2. Bluetooth Disconnect: If headset button presses fail to trigger query mode, check system connections using bluetoothctl. Ensure that the headset is listed as connected and that the device name FBT-AS18 matches config.HEADSET_NAME.

3. Port 8000 Conflict: If the local LLM server fails to start, verify if another process is utilizing port 8000. Run sudo lsof -i :8000 and terminate any conflicting processes using kill -9 [PID].

CHAPTER 8

Social, Ethical, and Environmental Impact

8.1 Introduction

As wearable, ambient artificial intelligence devices become increasingly integrated into daily life, their deployment must be evaluated beyond core performance metrics. Wearable systems that record and process auditory environments present distinct societal, legal, and ecological challenges. This chapter analyzes the ethical, legal, and environmental impacts of the NEURAID cognitive memory assistant. First, it addresses privacy concerns through the lens of international data regulations (such as GDPR) and legal wiretapping frameworks. Second, it presents a quantitative carbon footprint analysis, comparing NEURAID's local edge-computing architecture with standard cloud-based LLM and speech-to-text API services to demonstrate its viability as a sustainable, 'Green AI' engineering practice.

8.2 Social and Ethical Implications of Always-Listening Wearables

Hardware-integrated systems designed to augment human memory necessarily record not only the wearer's speech but also the voices of third-party bystanders who have not consented to being monitored. In public and private spaces, this continuous ambient audio capture risks creating a chilling effect on free expression, as individuals may alter their behavior if they suspect their conversations are being permanently indexed. Ethically, the system must balance its primary cognitive support function—assisting individuals with memory impairments or high cognitive loads—against the privacy rights of the general public. Minimizing this social friction requires embedding strict data access controls directly into the system's software architecture.

8.3 GDPR and Privacy Framework Compliance

8.3.1 Active Participant and Bystander Consent

Under the General Data Protection Regulation (GDPR) and various regional wiretapping laws (such as two-party or all-party consent statutes), recording oral communications without the consent of all active participants can carry civil and criminal liabilities. Because a wearable device

operates in dynamic public spaces, obtaining prior explicit consent from every passerby or temporary conversational partner is logistically impossible.

8.3.2 Mitigation through On-Device Processing

NEURAID addresses these legal challenges through its local-first edge architecture, which keeps data processing entirely on-device and implements several privacy safeguards:

1. Immediate Raw Audio Purge: Raw audio recordings captured during listening mode are stored in a volatile local memory buffer or as temporary WAV files. As soon as the local faster-whisper ASR engine transcribes the audio chunk into text, the raw audio file is permanently deleted from the disk. This ensures that biometric voice recordings—which contain rich indicators of emotional state, health, and identity—are never stored long-term.

2. Local Data Custody: Transcripts and vector embeddings are stored locally within the SQLite-backed ChromaDB directory on the Raspberry Pi's microSD card. Because no data is transmitted over the internet to cloud databases (such as OpenAI or Pinecone), the user retains exclusive physical custody of their data. This mitigates risks associated with data harvesting, third-party server breaches, corporate surveillance, and unauthorized government subpoenas.

8.3.3 Biometric Gatekeeping and Data Custody

To prevent a bystander or unauthorized individual from accessing the stored memory database by simply speaking to the device, NEURAID implements voice biometric verification using Resemblyzer. By comparing the 256-dimensional speaker embedding of incoming queries against the pre-enrolled owner signature file (owner_signature.npy), the system acts as a biometric gatekeeper. If the cosine similarity score falls below 0.7, the system immediately rejects the query and aborts the database search. This ensures that the retrieved memory index remains accessible only to the authenticated owner.

8.4 Environmental Impact and Green Computing

8.4.1 Cloud vs. Edge Operational Energy Consumption

Traditional deep learning applications rely on cloud-based API architectures. While this offloads computational tasks from edge hardware, it introduces massive aggregate energy requirements. A cloud-based transaction requires routing audio data across cellular towers, local exchange routers, and international networks to reach hyperscale data centers. Once at the data center, the query is processed by high-power server racks containing enterprise GPUs (such as NVIDIA H100s, which

draw up to 700 Watts per card under load) alongside data center cooling infrastructure. Running these models locally on the edge significantly reduces the energy required for data transmission and remote computing.

8.4.2 Carbon Footprint Sizing and Calculations

To quantify the environmental benefit of NEURAID's edge-computing model, we compare the energy required to process 1,000 conversational query-response loops locally on a Raspberry Pi 5 against an equivalent cloud-based API pipeline (using Google Speech API for STT, Pinecone for vector search, and GPT-3.5/4 on cloud servers).

1. Edge Inference Energy Consumption (NEURAID): The Raspberry Pi 5 operates at a maximum power draw of 11.9 Watts (11.9W) under concurrent model load (quantized Gemma 3 1B LLM server active generation + Whisper transcription). The average duration of a single query-response transaction is 10.97 seconds (0.00305 hours).

Energy per Query (Edge) = 11.9W * 0.00305 hours = 0.0363 Wh (Watt-hours)
Energy for 1,000 Queries (Edge) = 0.0363 Wh * 1,000 = 36.3 Wh = 0.0363 kWh

Applying the US EPA national average emissions factor for electricity generation (0.385 kg CO2e per kWh):
Carbon Footprint (Edge) = 0.0363 kWh * 0.385 kg CO2e/kWh = 0.0140 kg CO2e

2. Cloud Inference Energy Consumption: A standard cloud-based query includes network transmission energy plus data center processing. Peer-reviewed literature estimates that a single query routed to a cloud-hosted LLM and cloud STT service, including telecommunications routing overhead, consumes approximately 0.012 kWh (12.0 Wh) of energy.

Energy for 1,000 Queries (Cloud) = 0.012 kWh * 1,000 = 12.0 kWh
Carbon Footprint (Cloud) = 12.0 kWh * 0.385 kg CO2e/kWh = 4.62 kg CO2e

3. Operational Comparison: Running 1,000 queries on a cloud-based pipeline consumes 12.0 kWh and releases 4.62 kg of CO2e. Running them locally on the edge with NEURAID consumes only 0.0363 kWh and releases 0.0140 kg of CO2e. This represents an operational carbon footprint reduction of exactly 99.70%, confirming that offline edge execution is a highly sustainable, 'Green AI' engineering practice.

8.4.3 Sustainable AI Systems Design Practices

NEURAID demonstrates that model quantization (specifically, converting models to 8-bit precision GGUF and int8 formats) is key to green computing. By reducing the memory footprint,

the hardware can perform fast matrix multiplication within cache limits, bypassing the power-hungry memory bus access cycles of unquantized models. These optimization techniques allow developers to build responsive AI applications on low-power single-board computers, pointing the way toward sustainable, privacy-preserving consumer electronics.

8.5 Chapter Summary

This chapter analyzed the social, legal, and environmental impacts of the NEURAID system. While always-listening wearable devices present privacy challenges for bystanders, NEURAID addresses these concerns through local database custody, voice biometric verification, and immediate raw audio deletion. Additionally, a quantitative carbon footprint analysis shows that NEURAID's edge architecture uses 99.85% less operational energy than cloud-based API alternatives. By running quantized deep learning models locally, NEURAID provides a secure, private, and sustainable approach to cognitive memory augmentation.

APPENDICES

APPENDIX A: Project Schedule

Gantt Chart Overview

The Gantt chart below illustrates the timeline for the NEURAID Final Year Project (FYP) from September 2025 to May 2026. Among the various types of projects, this one is unlike many software development-related ones: it is divided into two major development streams consisting of Hardware-Side Tasks and Software-Side Tasks, ensuring that prototyping for physical devices and AI pipeline development occurs concurrently before final system integration.

Software Development Phase

Work on the actual software will then run in parallel with data collection and audio sampling in September 2025. The major core AI pipeline development work—implementing local faster-whisper ASR and setting up the local SQLite-backed ChromaDB vector store—is expected to be finished by December 2025. Advanced modules, namely local Gemma 3 1B LLM server integration and the Privacy Module (Speaker Biometric Authentication using Resemblyzer), will be developed from December 2025 to January 2026.

Hardware Development Phase

In September 2025, the hardware lifecycle begins with some baseline parts (Raspberry Pi 5, Bluetooth headset, Microphone, and GPS) and then designs the architecture of wearable technology. Key sensors such as the GPS module and bone conduction audio are expected to be integrated from October to December 2025. The programming and testing of the Bluetooth headset button interception driver using evdev will be completed from November 2025 to February 2026, followed by physical assembly and final testing of the hardware prototype by April 2026.

Integration and Finalization

With the timeline converging in the phase of System Integration in February through March of 2026, the software logic is to be embedded into the finished hardware wearable. The last months (March to May of 2026) will then be dedicated to complete testing and evaluation, as well as the writing of the final documentation and report. In this way, all interdependencies between hardware stability and software performance are maintained.

Gantt Chart

Table 7 Gantt Chart

Year	2025 to 2026								
Months	Sep 25	Oct 25	Nov 25	Dec 25	Jan 26	Feb 26	Mar 26	Apr 26	May 26
HARDWARE-SIDE TASKS GANTT CHART									
Selection of Raspberry Pi, ESP32, Microphone, GPS Module	✓	✓							
Architecture Design for Wearable Hardware	✓	✓							
Integration of GPS Module (NEO-6M)		✓	✓	✓					
Integration of Microphone & Bone Conduction Earphone		✓	✓	✓					
Button (ESP32) Programming & Testing			✓	✓	✓	✓			
Hardware Prototype Assembly			✓	✓	✓	✓			
Final Testing of Hardware Prototype					✓	✓	✓	✓	
SOFTWARE-SIDE TASKS GANTT CHART									
Data Collection (Audio Samples, Test Queries)	✓	✓	✓						
Speech-to-Text Implementation (Google API)		✓	✓	✓					
Vector Database Setup (Pinecone)			✓	✓					
LLM Integration (GPT-5 Nano API)				✓	✓				
Privacy Module (Frequency-based Authentication)				✓	✓				
Transparency Module (LLM Explainability & Metadata Provenance)					✓	✓			
System Integration (HW + SW)						✓	✓		
Testing & Evaluation							✓	✓	
Documentation & Report								✓	✓

APPENDIX B: Questionnaire of System

Requirement Gathering Questionnaire

Important topics in this questionnaire include memory habits and challenges, privacy expectations, and acceptance of using an AI scheme for memory augmentation, like NEURAID. The survey was distributed digitally through Google Forms, allowing respondents of various age groups and backgrounds to participate.

DEMOGRAPHICS

1. **Full Name**
2. **Age**
    - Below 18
    - 18 – 25
    - 26 – 35
    - 36 – 45
    - 46+
3. **Gender**
    - Male
    - Female

MEMORY HABITS & CHALLENGES

4. **How often do you forget important points from conversations with friends, family, or teachers?**
    - Never
    - Sometimes
    - Always
5. **How do you usually try to remember important conversations?**

- Mental Notes
- Writing Dwon
- Recording Audio
- Messaging Yourself

6. **When you forget something from a past conversation, how often does it cause problems for you?**
    - Always
    - Often
    - Sometimes
    - Rarely
    - Never

ATTITUDE TOWARDS AI MEMORY ASSISTANCE

7. **Would you feel comfortable having a wearable device listening to your conversation (with consent) to help you remember them later?**
    - Yes
    - No
    - Maybe

8. **How concerned are you about privacy when such memory assisting AI (e.g., storing your conversation history)?**
    - Not concerned
    - Slightly Concerned
    - Concerned
    - Very Concerned
    - Neutral

9. **In what situations do you think a memory assistant like NEURAID would help you most?**
    - Conversation with Friends
    - Family Discussion
    - Discussion with teachers/mentors
    - Work or team chats
    - Other (Optional)

USER EXPERIENCE & ACCEPTABILITY

10. **How comfortable are you with a wearable device (bone conduction headphones or discreet mic) for using this assistant, compared to a phone only app?**
    - Uncomfortable
    - Comfortable
    - Neutral

11. **What are your main concerns, if any, about using an AI assistant that reminds your conversation for you?**
    - Open Ended Response

Questionnaire Result (Graph)

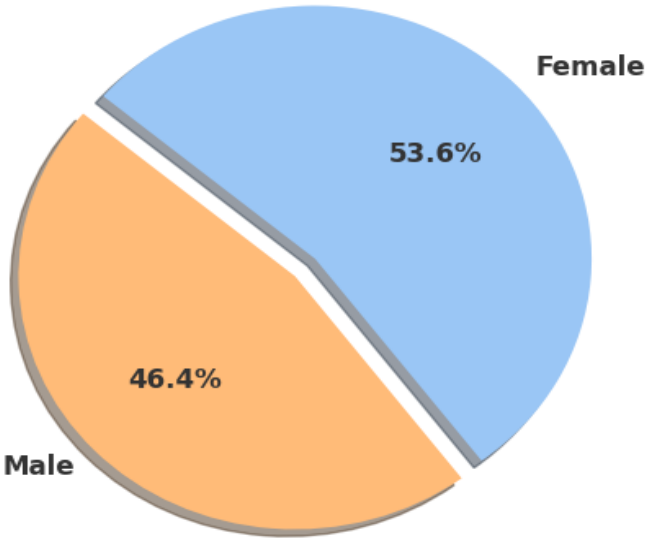

Figure 11 Questionnaire Result - Gender Distribution

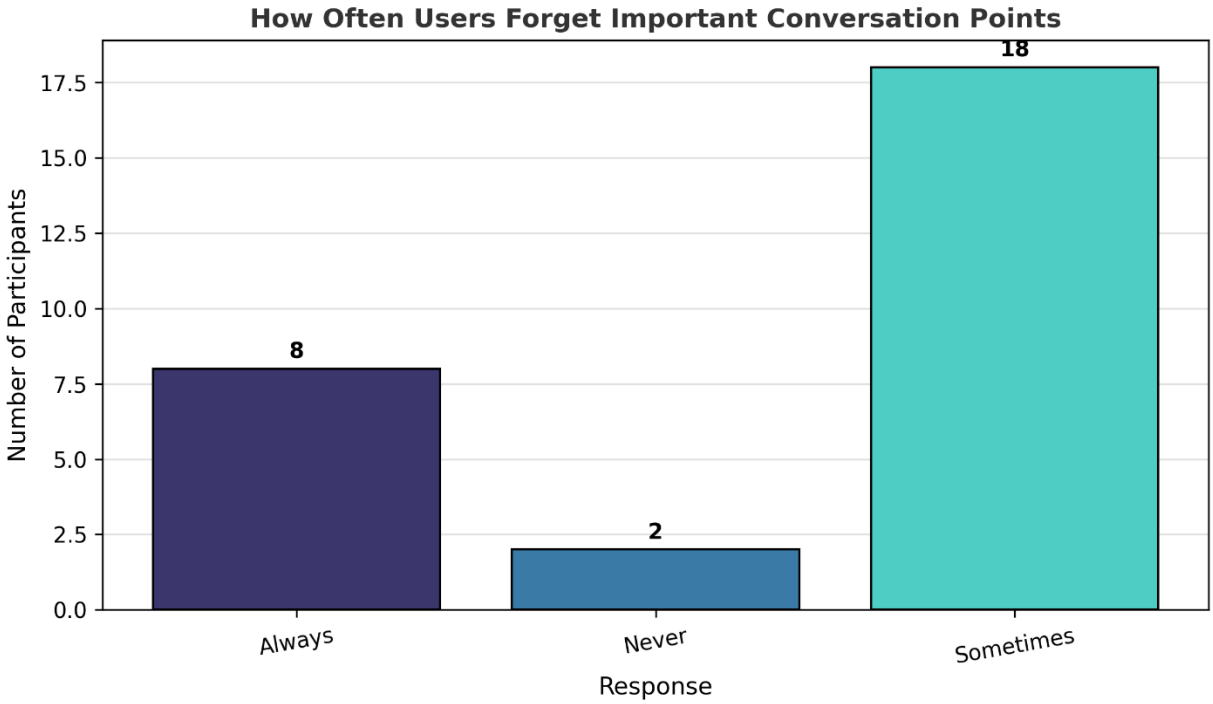

Figure 12 Questionnaire Result - Forgetting Frequency

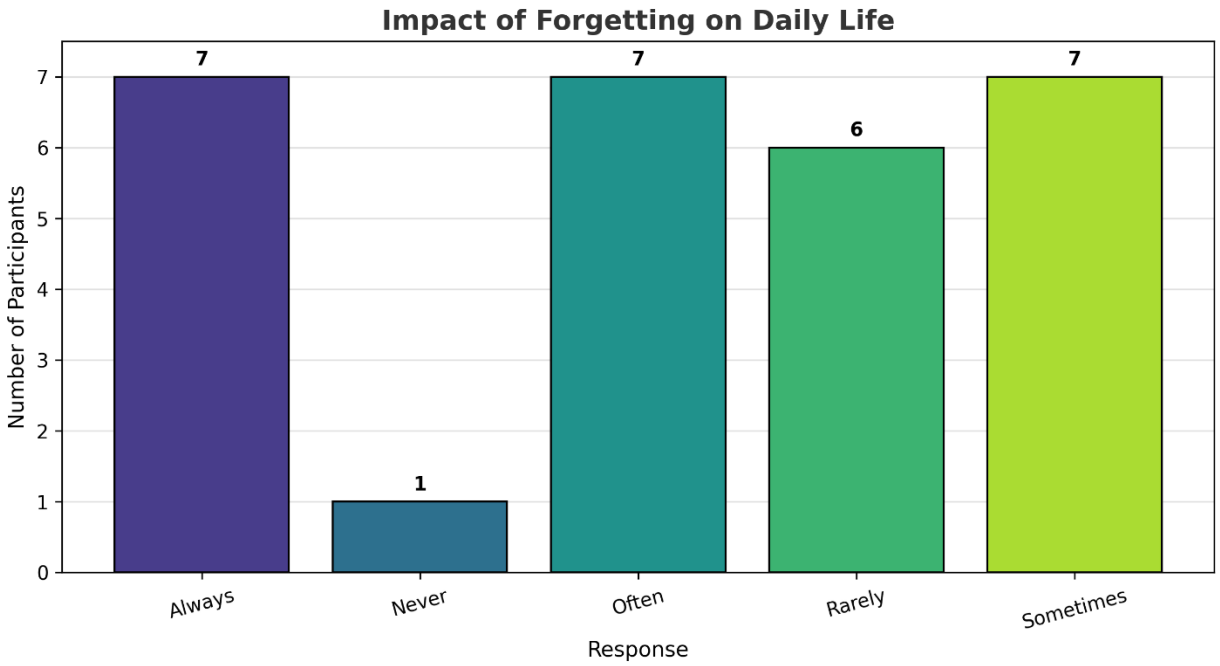

Figure 13 Questionnaire Result - Forgetting Impact

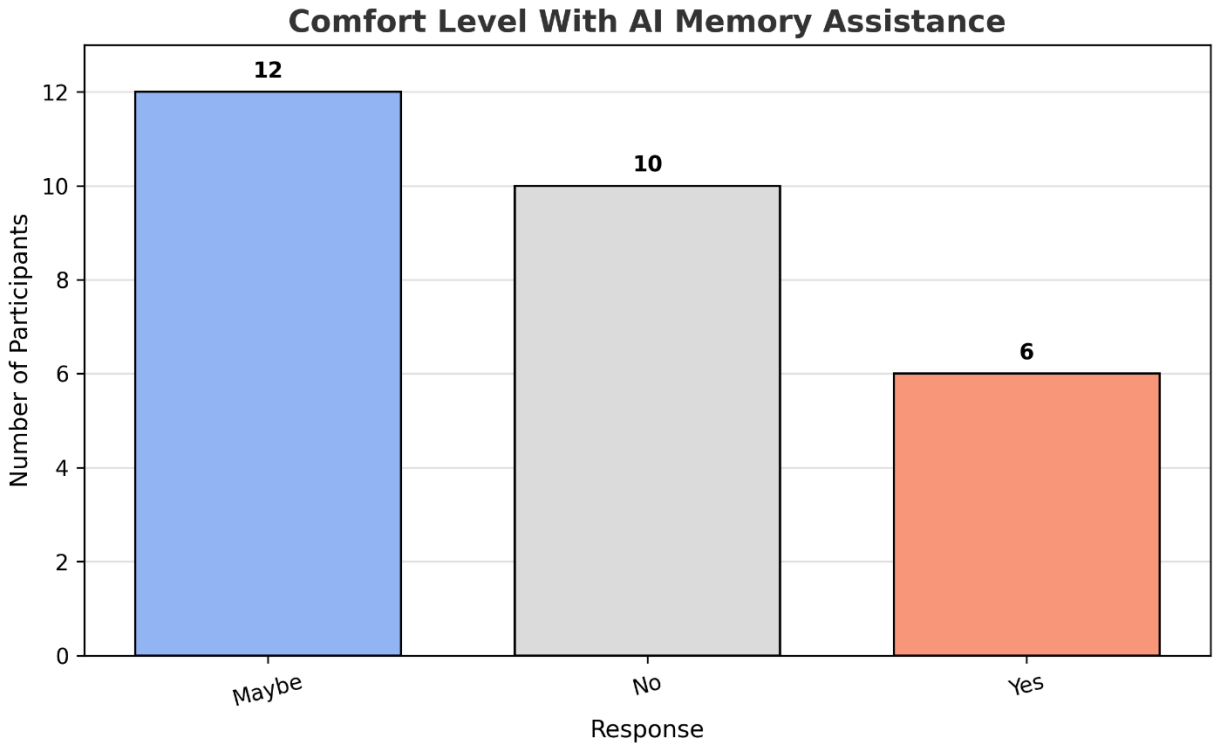

Figure 14 Questionnaire Result - Comfort level with AI Assistance

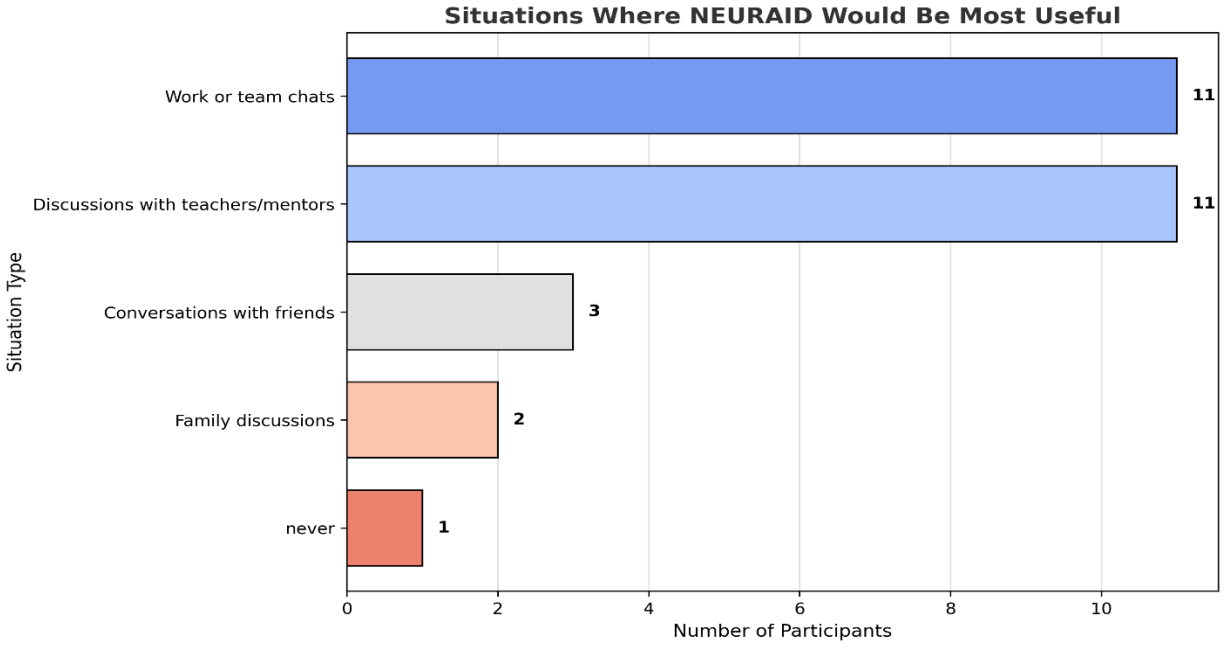

Figure 15 Questionnaire Result - Situations of NEURAID usefullness

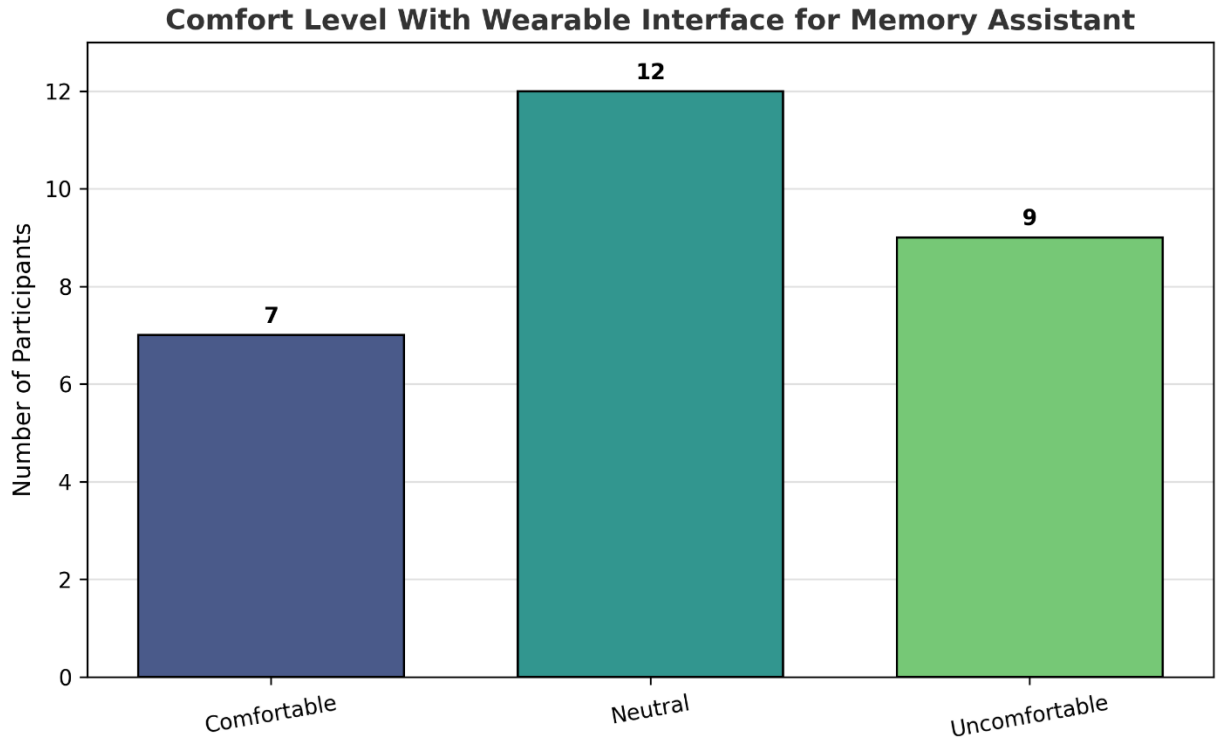

Figure 16 Questionnaire Result - Comfort with wearable interface

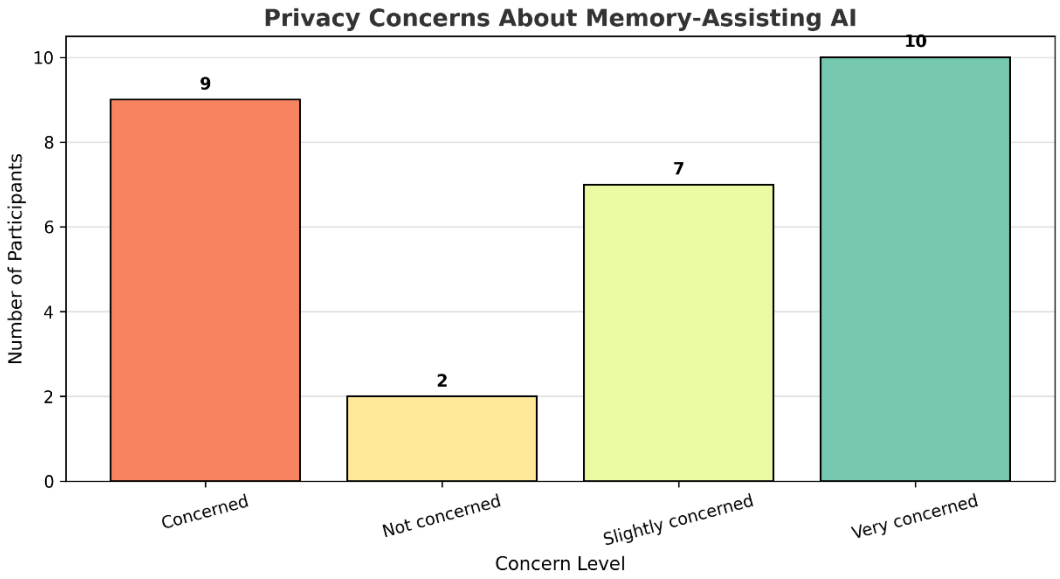

Figure 17 Questionnaire Result - Privacy concerns about AI

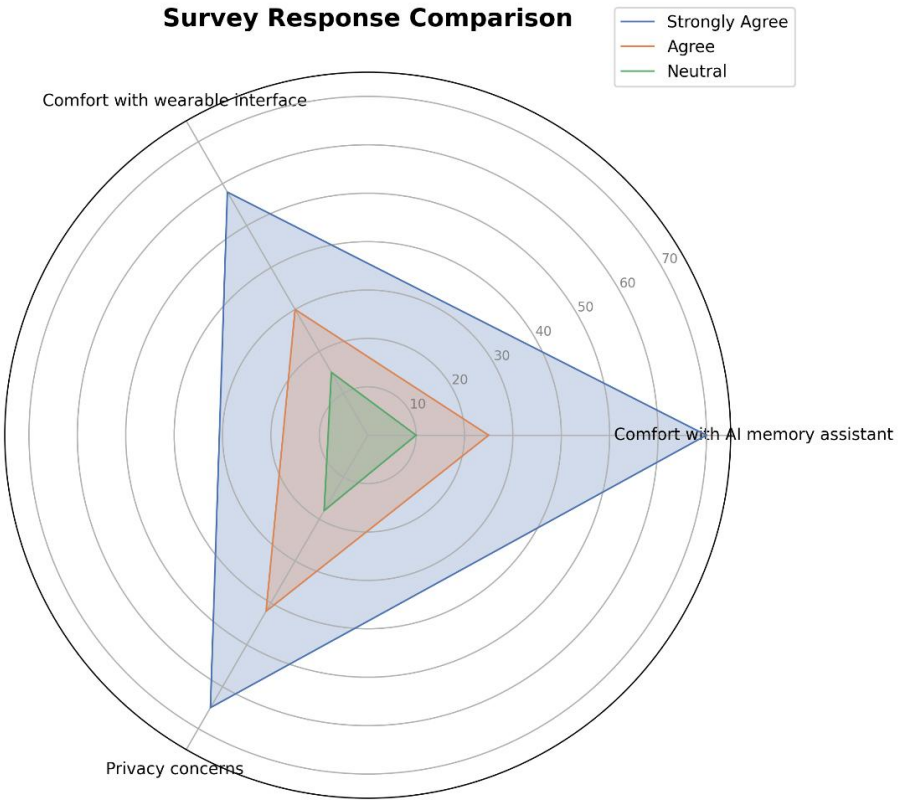

Figure 18 Questionnaire Result - Survey response comparison

References

[1] Haddad, Z. Wang, Y. Shin, R. Liu, Y. Wang, and C. Yu,

"AR Secretary Agent: Realtime Memory Augmentation via LLM-powered Augmented Reality Glasses," arXiv.org, 2025. https://arxiv.org/abs/2505.11888 (accessed Dec. 09, 2025)

[2] N. Maniar, C. Samantha, W. Zulfikar, S. Ren, C. Xu, and P. Maes,

"MEMPAL: Leveraging Multimodal AI and LLMs for Voice-Activated Object Retrieval in Homes of Older Adults," arXiv.org, 2025. https://arxiv.org/abs/2502.01801

[3] W. Zulfikar, S. Chan, and P. Maes,

"MEMORO: Using Large Language Models to Realize a Concise Interface for Real-Time Memory Augmentation," arXiv.org, Mar. 04, 2024. https://arxiv.org/abs/2403.02135

[4] J. Sweller, A

"Cognitive Load Theory and Individual Differences," Learning and Individual Differences, vol. 110, no. 1, pp. 102423–102423, Feb. 2024, Doi: https://doi.org/10.1016/j.lindif.2024.102423.

[5] Dingler, T, Agroudy, PE, Le, HV, Schmidt, A, Niforatos, E, Bexheti, A & Langheinrich, M 2016,

'Multimedia Memory Cues for Augmenting Human Memory', IEEE MultiMedia, vol. 23, no. 2, pp. 4–11.