RecallAR: An Augmented Reality Memory Layer for Social Interactions

Collin Boler
collinboler@princeton.edu
Princeton University
Princeton, New Jersey, USA

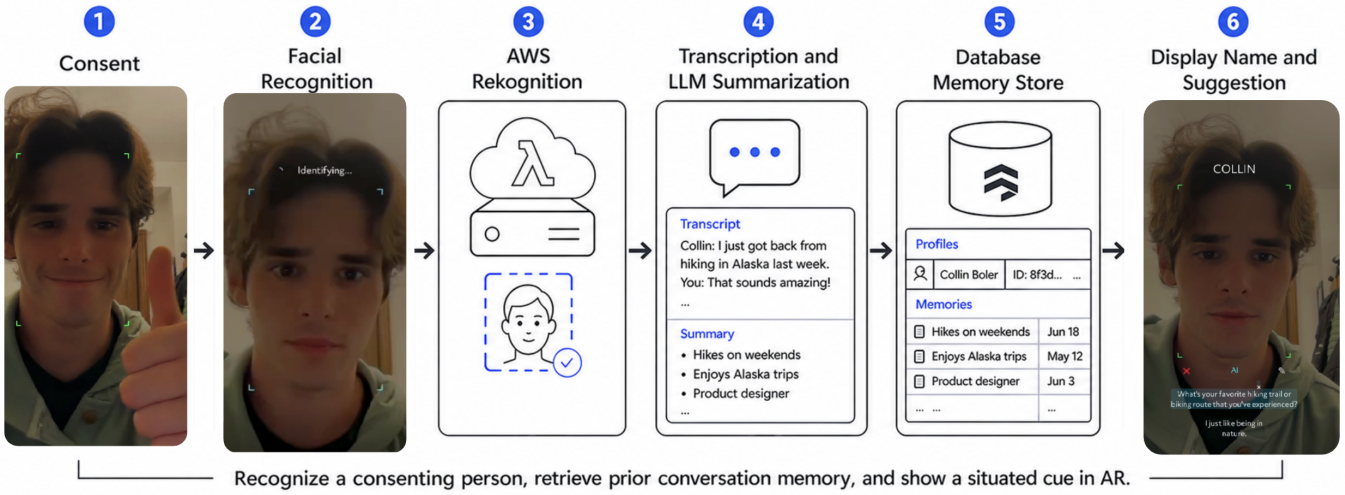

Figure 1: RecallAR pipeline from enrollment to in-situ memory cue.

Abstract

Forgetting names and earlier conversation details is a small but common failure that can make social and professional interactions feel awkward. RecallAR is an augmented reality prototype that explores whether smart glasses can provide a timely, lightweight memory cue while a person is physically in front of the wearer. The system recognizes consenting enrolled people through a Spectacles camera feed, retrieves their stored profile and recent conversation memories, and displays a name plus one relevant conversation nudge in the wearer's field of view. Unlike general lifelogging tools, RecallAR focuses on social continuity: helping the wearer remember who someone is and what would be natural to ask next. The prototype integrates Snap Spectacles, Lens Studio TypeScript, AWS Rekognition, Firebase Firestore, and OpenAI-backed transcription and summarization endpoints. This paper describes the design decisions, implementation, limitations, and ethical constraints of a situated AR memory assistant.

1 Introduction

People often recognize a face before they remember a name. They may also remember that they have met someone before without remembering the detail that would make the next conversation feel personal: a project deadline, a recent trip, a hobby, or something the person cared enough to mention last time. This problem is ordinary, but it matters. Forgetting names can create social friction, and forgetting personal details can make repeated interactions feel less attentive than they should. In professional settings, these lapses can also weaken networking, mentoring, and relationship-building.

RecallAR addresses this problem by treating augmented reality as a situated social memory layer. The key idea is not face recognition alone. Face recognition identifies who is in front of the wearer; it does not decide what information is useful in that moment. RecallAR combines identity retrieval with transcript-grounded memory so that the system can show a short, socially relevant cue at the time and place where the cue is useful. For example, rather than simply showing "Mike," the system can show "Mike - ask how marathon training is going" if a prior conversation included that detail.

The perspective behind RecallAR is that memory support should be contextual and glanceable. A phone contact list can store names and notes, but it requires the user to stop, search, and look away from the interaction. A general meeting summarizer can capture conversation history, but it is usually detached from the physical moment of seeing the person again. RecallAR instead uses the presence of a face in the camera view as the trigger for retrieval. It aims to keep the interaction in the real world by placing a small cue near the person rather than turning the social situation into an app session.

This framing also changes what counts as success. A technically impressive system that displays too much information could still fail socially, because the wearer might become visibly distracted or the other person might feel watched. RecallAR therefore treats restraint as part of the design problem. The interface should help the wearer recover one useful thread, not narrate everything it

knows. In early versions of the concept, I imagined a richer profile card with many stored facts. I moved away from that design because the most realistic social use case is not studying a person's file, but recovering enough context to make the next sentence feel attentive.

This project is scoped to non-medical everyday use. Medical use cases, such as supporting people with dementia-related memory loss, involve clinical, ethical, and evaluation challenges that fall outside the timeframe of this project. It is not intended as a clinical memory aid or a system for identifying bystanders. The prototype is designed around consenting people, with visible controls for naming, editing, deletion, and generating reminders. The main contribution is an end-to-end prototype of a consent-aware AR workflow for social recall: enrollment, face matching, conversation capture, memory extraction, and in-situ display.

2 Background

RecallAR builds on several strands of prior work. The Remembrance Agent explored continuous information retrieval for augmented memory, displaying short summaries of notes and documents that might be relevant to a wearer's current context [5]. RecallAR shares the goal of just-in-time recall, but shifts the retrieval target from documents to interpersonal memory and uses face presence as a situated trigger.

Lifelogging and wearable memory aids such as SenseCam showed that passive capture can support later autobiographical memory [3, 6]. Those systems are valuable for reviewing the past, but RecallAR focuses on acting during a live social encounter. Instead of asking the user to browse captured history later, it extracts small memory nuggets that can be surfaced while the person is present.

The project also connects to context-aware computing. Dey, Abowd, and colleagues describe systems that sense context and use it to provide relevant services [2]. In RecallAR, context includes identity, physical co-presence, recent conversation history, and whether the user is currently in a face-tracked interaction. This context narrows what the system should show: not a complete profile, but a brief cue that helps the current conversation.

Existing commercial products cover parts of this space. Smart glasses and AR development platforms provide camera access, hand tracking, and head-worn display primitives [7, 8]. Cloud APIs can perform face search [1], speech-to-text [4], and text summarization. RecallAR's novelty is not inventing those components individually. It is the integrated workflow: consented face enrollment, recognition, transcript memory, and a small AR nudge designed for social continuity rather than surveillance or general productivity.

There is also an important distinction between remembering facts and supporting relationships. Many note-taking or customer relationship management systems store information about people, but they are usually designed for explicit lookup before or after an interaction. RecallAR explores a different moment: the instant when the wearer has already encountered the person and needs a cue without stepping out of the conversation. This makes the physical situation central. The person, the camera, the display, and the prior memory all meet in the same place.

Prior work also suggests why this problem is ethically sensitive. Wearable memory systems can shift from assistance to monitoring if capture is hidden or too broad. For that reason, the relevant gap is not simply "put face recognition in AR." The more interesting challenge is whether a system can provide useful contextual memory while making consent, deletion, and social appropriateness first-class parts of the workflow.

3 Design Approach

RecallAR was designed around a simple repeated scenario: the wearer sees someone, the system recognizes the person, and the interface provides just enough information to help the wearer continue naturally. This led to three design goals.

First, the cue should be glanceable. The main UI shows the person's name above or near the tracked face, plus a short memory nudge below it. The design avoids a large dashboard because the user is supposed to remain in a conversation, not read a profile. A good cue should be readable in a second or two and then fade into the background.

Second, the system should make consent and control part of the interaction instead of hiding them in settings. Unknown faces are not automatically committed as permanent identities. The prototype stages enrollment and waits for a thumbs-up confirmation before saving a new person. The app also supports editing a name and deleting a stored identity. This matters because face recognition in public or semi-public space can easily feel invasive. RecallAR's intended use is narrower: small groups where people agree to be enrolled and where the wearer has a legitimate reason to remember prior interactions.

Third, the memory cue should be grounded in conversation history. During a recognized interaction, the system can capture speech, produce a transcript, extract concise "memory nuggets," and save them under that person's record. Later, the system retrieves recent memories and can generate one next-question suggestion. The design goal is to avoid generic prompts like "Ask about work" and instead show something grounded, such as "Ask how the robotics demo went." This is the difference between an identification system and a social memory assistant.

The user experience has two main flows. In the enrollment flow, the wearer looks at a consenting person, confirms enrollment with a thumbs-up gesture, enters or says the person's name, and saves the record. In the recognition flow, the wearer looks at an enrolled person, the system scans periodically, and the AR overlay appears with the person's name and recent memory cue. A small set of controls supports rescan, edit, delete, transcript viewing, and suggestion generation.

The enrollment flow is intentionally slower than the recognition flow. This is a deliberate tradeoff. Recognition should feel almost automatic once a relationship exists, but enrollment is the moment when the system creates a durable identity record. I designed the thumbs-up gesture as a lightweight "green light" because it is visible, fast, and natural on Spectacles. After the gesture, the wearer can name the person using the on-device keyboard or speech input. If they abandon the flow, the pending face is not committed as a permanent Rekognition or Firestore record. This avoids filling the database with accidental "unknown person" entries and gives the wearer a clean way to back out.

The recognition flow prioritizes stability over constant novelty. The app locks onto a matched person and keeps the card visible

while periodically re-verifying identity in the background. This avoids a common failure mode in AR demos: the system appears to work, but every small tracking failure makes the overlay flash or reset. RecallAR includes hysteresis for mismatches and face-lost events so a single bad frame does not immediately replace the name or erase the cue. In practice, these small interaction details make the system feel less like a diagnostic tool and more like something that could sit quietly inside a real conversation.

The memory design also avoids presenting raw transcripts by default. Raw transcripts are useful for debugging and review, but they are too long and too private for the primary overlay. The front-end instead shows short nuggets and exposes only real time transcription. This separation reflects the main design claim: the system should turn conversation history into a small, timely reminder, not make the wearer read a meeting log while talking to someone.

Finally, the UI uses direct manipulation controls for sensitive actions. Editing a name is attached to the recognized person, deletion is attached to the same face-anchored context, and suggestion generation is shown as a specific action rather than a constantly updating stream. These decisions make the system more legible. The wearer can tell when the app is simply showing stored information, when it is recording or processing conversation, and when it is about to modify a record.

4 Implementation

RecallAR is implemented as a Snap Spectacles lens in Lens Studio using TypeScript. The main front-end orchestrator is RecallARLens.ts, which manages camera initialization, face tracking, network calls, AR text placement, enrollment state, and conversation memory state. The lens uses Spectacles camera frames for recognition and a Head Binding component for continuous face tracking. Recognition runs periodically rather than every frame, while the UI position is updated from face landmarks and Rekognition bounding boxes so the overlay appears visually attached to the person.

The front-end has two levels of visual feedback. A screen-space HUD can display status text, while the main experience uses head-locked and face-anchored text for the name, memory nudge, transcript preview, and controls. The code smooths bounding-box movement to reduce snapping, debounces face-lost events to avoid flicker, and re-verifies identity on an interval so the UI can recover when a different person enters the frame. These details are important because an AR prototype can be technically correct but feel unusable if labels jump around or disappear during normal head motion.

The camera pipeline starts with the Spectacles Camera Module. The app requests the left color camera and encodes frames as low-quality JPEG Base64 strings to reduce payload size. Each scan sends the image to a backend endpoint. The default recognition threshold in the current prototype is 85 percent similarity, with a more permissive manual rescan path for borderline cases. The client also handles no-face and no-match cases separately, so it can retry poor frames without immediately treating every failed scan as a new person.

The recognition loop is split into a fast local tracking layer and a slower cloud recognition layer. Local face tracking updates the position of the overlay every frame, which is what makes the label feel spatially attached to the face. Cloud recognition runs on a timed scan interval because sending every frame to Rekognition would increase latency, cost, and network load. This split is important for a wearable device. The wearer needs the overlay to move smoothly, but the actual identity does not need to be recomputed sixty times per second.

Several implementation details were added after early testing. The app uses a no-bounding-box retry budget when Rekognition cannot find a usable face in the submitted frame. It also includes a self-healing timeout that restarts scanning after a temporary failure. When a person is already locked, the app waits for repeated contradictory recognition results before switching identities. These mechanisms are not part of the conceptual demo, but they are necessary for making the interaction believable on-device, where lighting, Wi-Fi, and head movement are inconsistent.

The backend is an AWS Lambda function exposed through API Gateway. The Lambda acts as a proxy between the lens and cloud services so that AWS credentials remain server-side. Its /search endpoint calls Amazon Rekognition SearchFacesByImage, returns the best match, confidence score, and bounding box, and applies the similarity threshold passed by the client. Its /enroll endpoint creates the Rekognition collection if needed and indexes a face. Its /delete endpoint removes a face from the collection, and its /upload endpoint stores an enrolled face image in S3.

Person records and memories are stored in Firebase Firestore. A person record includes the Rekognition face id, display name, enrollment note, timestamps, and optional photo URL. Recent memory records are stored under the person and include the raw transcript, extracted nuggets, creation time, and a date label. The Spectacles client uses Firestore's REST API to fetch a person by face id, update last_seen_at, and query recent memories in reverse chronological order.

Using Firestore as the memory database keeps the face-recognition layer separate from the social-memory layer. Rekognition stores face embeddings and returns a face id; Firestore stores the human-readable identity and memory history associated with that id. This separation made the system easier to debug and safer to modify. For example, deleting a person requires deleting the Rekognition face and the Firestore record, while changing a person's display name only touches Firestore. It also means that memory retrieval can evolve independently of the recognition model.

Conversation memory is implemented through additional Lambda endpoints. The /conversation-diarize endpoint sends audio to an OpenAI transcription model and normalizes the response into speaker turns. The /extract-memories endpoint asks GPT-4o-mini to extract three to five concise facts from a transcript, focused on interests, plans, opinions, or experiences. The /conversation-suggestion endpoint generates one short next question or talking point from stored transcript history. These endpoints return structured JSON rather than free-form prose so the lens can display small cues reliably.

The memory extraction prompt is intentionally narrow. It asks for concise facts about plans, interests, opinions, and experiences rather than a complete summary. This keeps the downstream UI simple and reduces the chance that a long generated summary will contain details that are not useful in the next encounter. The suggestion endpoint is similarly constrained to exactly one short

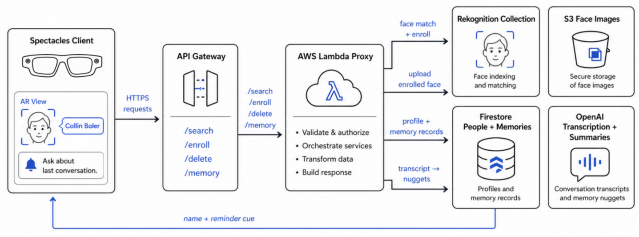

Figure 2: Backend architecture and API Calls.

question or talking point. In a conversational AR interface, quantity is not automatically helpful. One grounded prompt is more usable than five possible directions competing for attention.

On the Spectacles side, the app integrates several input modes. Hand tracking is used for the thumbs-up consent signal. UIKit buttons provide pinchable controls for rescan, editing, and deletion. Speech recognition is used for conversation capture and can also support name entry. These input methods are chosen because they fit a head-worn context where pulling out a phone or using a full keyboard would break the interaction. The implementation still includes a virtual keyboard fallback because speech is not always available in Lens Studio preview and can be unreliable in noisy spaces.

The result is a full loop: a consenting person is enrolled, their face id becomes the key for a Firestore profile, conversation snippets become memory records, and later face recognition retrieves those memories into an AR cue. The system is interactive because the wearer can trigger enrollment, edit or delete identity records, rescan faces, and request suggestions. It is situated because the cue is only meaningful when anchored to someone physically present in the camera view.

5 Limitations and Future Work

RecallAR has several limitations. The first is recognition reliability. Face recognition quality depends on lighting, camera angle, motion blur, and whether the enrolled face image is representative. The current threshold is tunable, but any threshold creates a tradeoff between missed recognitions and false matches. A false match is especially costly in a social memory app because the system might confidently show the wrong name or cue.

The second limitation is privacy. Even with consent gates, a wearable camera that recognizes people and stores conversation memories needs careful boundaries. The prototype assumes small, consenting groups. A production version would need stronger deletion tools, visible recording indicators, enrollment review, encrypted storage, retention limits, and clearer ways for the other person to know what is being captured. It should also avoid bystander identification by design.

The third limitation is memory quality. LLM-generated nuggets can be useful, but they can also miss nuance, over-compress a conversation, or generate a suggestion that feels too personal for the moment. Future versions should let users approve or edit extracted memories before they are saved. They should also rank reminders based not only on recency, but on social appropriateness, relationship context, and whether a topic was already discussed.

A fourth limitation is group interaction. The current prototype focuses on one face at a time, which matches the simplest use case but does not cover many realistic social settings. At a conference, classroom, or group meeting, several enrolled people may be visible. A future version would need to decide which person to prioritize, how to avoid cluttering the wearer's view, and how to handle overlapping conversation history. It may be better to show no cue at all than to overload the wearer with labels in a group.

There is also a social-performance limitation. Even if the cue is accurate, the wearer must use it gracefully. Asking a question that sounds too obviously machine-generated could make the interaction feel less authentic. This suggests that future systems should support subtle reminders rather than scripts. The best cue may be a topic phrase rather than a full sentence, leaving the wearer to phrase the follow-up naturally.

Future work should include a small user study in which pairs or small groups use RecallAR over repeated meetings. Useful measures would include recognition accuracy, time to display, perceived awkwardness, trust, and whether the cue actually helped the wearer ask better follow-up questions. I would also like to explore on-device processing where possible, stronger audio consent flows, and a more polished AR layout that adapts to distance and group conversations.

6 Conclusion

RecallAR is an AR prototype for remembering names and conversation details in everyday social settings. It incorporates intelligence through face recognition, speech transcription, and LLM-based extraction of memory nuggets and conversation suggestions. It is interactive because users can enroll people, confirm consent with a gesture, edit or delete records, rescan faces, and request context-specific prompts. It is situated in physical space because the system uses the Spectacles camera and face tracking to display cues only when a person is present in front of the wearer. The project shows how AR can move beyond displaying information to supporting social continuity, while also making clear that accuracy, consent, and restraint are central design requirements for this kind of system.

Acknowledgments

I would like to extend my thanks to Parastoo Abtahi for helping to advise this project as part of Princeton's AR Meets AI Independent Work Seminar. ChatGPT was utilized to help generate some sections of this work, including text, code, diagrams, and citations.

References

[1] Amazon Web Services. 2024. SearchFacesByImage: Amazon Rekognition API Reference. https://docs.aws.amazon.com/rekognition/latest/APIReference/API_SearchFacesByImage.html

[2] Anind K. Dey, Gregory D. Abowd, and Daniel Salber. 2001. A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications. *Human-Computer Interaction* 16, 2-4 (2001). doi:10.1207/S15327051HCI16234_02

[3] Steve Hodges, Lyndsay Williams, Emma Berry, Shahram Izadi, James Srinivasan, Alex Butler, Gavin Smyth, Narinder Kapur, and Ken Wood. 2006. SenseCam: A Retrospective Memory Aid. In *UbiComp*. https://www.microsoft.com/en-us/research/wp-content/uploads/2016/11/SenseCam-Ubicomp-2006-camera-ready.PD_.pdf

[4] OpenAI. 2024. Speech to Text Guide and Transcription API Documentation. https://platform.openai.com/docs/guides/speech-to-text

[5] Bradley J. Rhodes. 1997. The Wearable Remembrance Agent: A System for Augmented Memory. *Personal Technologies* (1997). https://www.bradleyrhodes.com/

Papers/wear-ra.html
[6] Abigail Sellen, Andrew Fogg, Mike Aitken, Steve Hodges, Carsten Rother, and Ken Wood. 2007. Do Life-Logging Technologies Support Memory for the Past? An Experimental Study Using SenseCam. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI)*. https://www.microsoft.com/en-us/research/wp-content/uploads/2007/09/CHI07_SensecamMemory.pdf
[7] Snap Inc. 2024. Camera Module: Spectacles APIs. https://developers.snap.com/spectacles/about-spectacles-features/apis/camera-module
[8] Snap Inc. 2024. Internet Access: Spectacles APIs. https://developers.snap.com/spectacles/about-spectacles-features/apis/internet-access