CONXT

Cross-AI Memory Layer

WHITE PAPER

Solving the Context Amnesia Problem in Modern AI Workflows

Version 1.0 · May 2025
conxt.dev

Executive Summary

Problem	AI models have no persistent memory across sessions or platforms — every conversation starts from zero.
Solution	Conxt captures, classifies, and persists conversational context as structured memory across all major AI platforms.
Approach	A Chrome extension silently monitors AI sessions and maps content into seven structured memory types via a FastAPI extraction engine.
Market	Professionals using AI daily — developers, PMs, researchers, consultants — and teams that need shared AI context.
Status	Live product. Chrome extension available. FastAPI engine deployed on Railway. Dashboard on Vercel. Team workspaces in active development.

Conxt is a cross-AI memory layer that solves the context amnesia problem inherent in all current large language model deployments. By capturing conversational context in real time and classifying it into seven structured memory types, Conxt gives professionals a persistent, platform-neutral memory that follows them across Claude, ChatGPT, Gemini, and Microsoft Copilot. This white paper describes the problem, the architecture, the competitive landscape, and the product roadmap.

1. The Problem: Context Amnesia in AI Workflows

1.1 The Stateless Nature of Large Language Models

Large language models process tokens within a fixed context window and produce output. When a session ends, nothing persists. The next conversation begins with no knowledge of prior interactions, user preferences, or organizational context. This is not a capability failure — it is an architectural constraint. As one benchmark study summarized: "LLMs are stateless — they start each interaction without any context or memory, leaving power untapped and insight lost" [1].

For casual use, this is a mild inconvenience. For professionals using AI as a core part of their daily workflow — developers, product managers, researchers, founders — it is a compounding productivity tax. Users spend 10–15 minutes per session re-establishing context they have already established dozens of times before.

1.2 Scale of the Problem

The magnitude of this friction is quantified in recent industry research:

- 32% of organizations cite output quality — directly linked to agents starting without context — as the single biggest barrier to production AI deployment (LangChain State of Agent Engineering, 2025) [2].
- 95% of enterprise generative AI pilots delivered zero measurable ROI, with failure attributed to context readiness rather than model quality (MIT NANDA, 2025) [3].
- A 2025 survey of 1,340 AI practitioners found that the absence of cross-session memory was the most frequently cited workflow friction point [2].

1.3 The Multi-Platform Compounding Effect

The problem is not confined to a single model. Professional AI users move fluidly between platforms based on task requirements. Claude excels at long-context reasoning. GPT-4 has deep integrations. Gemini connects to Google Workspace. Copilot sits inside developer toolchains. Each platform switch resets context entirely.

Even if one platform implemented persistent memory, that memory would not follow users to other platforms. The context problem is structural and cross-platform — it cannot be solved by any single AI provider.

As the LangChain State of Agent Engineering report observed: "The problem is not model capability. The problem is agents operating without sufficient context on each run" [2].

2. The Solution: Conxt

2.1 Product Overview

Conxt is a persistent, cross-platform memory layer for AI users. It operates as a Chrome extension that passively monitors AI conversations and extracts structured memory — without requiring users to change how they work. Extracted memories are stored in a user-controlled dashboard and made available across every AI session the user opens.

The core value proposition is simple: the model starts knowing who you are, what you have decided, and how you work — every time, on every platform.

2.2 The Seven Memory Types

A foundational design decision in Conxt is that memory is not monolithic. Different categories of conversational context have different value, different shelf lives, and different retrieval patterns. Conxt classifies all captured memory into seven distinct types:

Memory Type	Definition	Example
Decision	An explicit choice or conclusion reached during an AI session.	*"We are going with PostgreSQL over MongoDB for this project."*
Preference	A stylistic or workflow preference expressed by the user.	*"I prefer functional React components. Always use TypeScript strict mode."*
Entity	A named person, organization, system, or concept central to the work.	*"Our staging environment runs on Railway. The client is Acme Corp."*
Open Question	An unresolved issue or decision still under consideration.	*"We have not decided whether to use JWT or session-based auth yet."*
Coding Rule	A technical rule or constraint that governs how code is written.	*"Never use any Tailwind color utilities. All styles must be inline."*
Tool Choice	A specific tool, library, or platform selected for a task.	*"Using Supabase with pgvector for the memory store. Deploying to Vercel."*
Workflow	A recurring process or sequence of	*"Always test locally on port 3000 before*

Memory Type	Definition	Example
	steps used to accomplish a task.	*pushing. Download ZIPs from GitHub."*

This taxonomy is derived from analysis of hundreds of professional AI sessions and informed by cognitive science research on the categories of professional knowledge that most affect workflow continuity [4]. The classification system eliminates noise from the memory store and ensures that retrieval surfaces contextually appropriate information rather than raw conversation fragments.

2.3 How Capture Works

The Conxt Chrome extension operates using Manifest V3 content scripts that monitor conversation streams on supported AI platforms. When a session ends or a sufficient conversational unit is detected, the extension transmits the session hash and content to the Conxt extraction engine.

The extraction engine — built in FastAPI and deployed on Railway — uses a two-stage pipeline:

- Stage 1 — Redaction: A regex-based redaction scanner removes personally identifiable information, API keys, and other sensitive content before any LLM processing occurs.
- Stage 2 — Extraction: A structured Claude Vision prompt with hard caps per session and an ownership/significance filter classifies content into the seven memory types and returns structured JSON.

Extracted memories are written to a Supabase database with pgvector embeddings, enabling semantic retrieval in addition to structured filtering. All memories are surfaced in the user dashboard for review, approval, and editing before injection.

2.4 Cross-Platform Injection

When a user opens a new AI session, Conxt injects relevant memory context into the conversation — either automatically via the extension or on-demand via the dashboard. The injection is platform-neutral: the same memory store serves Claude, ChatGPT, Gemini, and Copilot.

This platform neutrality is the core architectural differentiator. Memory is stored outside any individual AI platform and belongs entirely to the user.

3. Technical Architecture

3.1 System Components

- Chrome Extension (MV3, JavaScript): Content scripts for AI platform monitoring, session hash generation, redaction scanner, and memory injection.
- Extraction Engine (FastAPI, Python, Railway): JWT-authenticated API endpoints for memory extraction, team management, context injection, and capture processing.
- Dashboard (Next.js 16, Vercel): User interface for memory review, team workspaces, and memory feed with real-time Supabase subscriptions.
- Database (Supabase + pgvector): Structured storage for memory records, team data, capture sessions, and vector embeddings for semantic retrieval.

3.2 Authentication

Authentication uses Supabase Auth with a supplementary JWT verification endpoint (/auth/verify/) implemented via PyJWT. The extension authenticates via the dashboard session, with tokens verified on the engine for all API calls. This architecture eliminates direct database access from the extension layer.

3.3 Data Model

The core data entity is the memory_record, which stores the classified memory type, structured content as JSON, source platform, confidence score, session identifier, reinforcement count, and archive/pin status. Memory records belong to a user and optionally to a team, enabling both personal and shared memory workspaces.

The sessionHash field enables deduplication — the engine returns already_processed for sessions it has seen, preventing duplicate extraction on re-visit.

3.4 Privacy and Security

Privacy is a first-class architectural concern in Conxt:

- All content is redacted before LLM processing. No raw conversation text is stored.
- Memory records are user-scoped by default. Team sharing is opt-in and explicit.
- Users review and approve memories before they become active in their feed.
- Users can archive, delete, or edit any memory record at any time.
- The extension does not transmit data on non-AI pages.

4. Team Workspaces

4.1 The Shared Context Problem

Individual memory is powerful. Shared memory is transformative. When a team works together with AI tools, the context problem multiplies: each person loses their own context between sessions, and they cannot share the context they have built with teammates.

The result is redundant re-onboarding, inconsistent AI outputs, and knowledge that is locked in individual conversation histories that no one else can access.

4.2 How Team Workspaces Work

Conxt team workspaces provide a shared memory feed that all members can contribute to and draw from. Key properties:

- Any team member can create memories that are visible to the whole team.
- Decisions, coding rules, and workflows established by one team member are available in the AI sessions of all others.
- Teams join via an auto-generated invite code (format: CX-XXXXXX). No manual user provisioning.
- Role-based access: owners can manage membership; members contribute and consume.
- Personal memories remain private. Only memories explicitly tagged to a team are shared.

4.3 Onboarding Impact

The most significant team use case is onboarding. When a new team member joins a Conxt workspace, they immediately have access to the accumulated AI context of the entire team — the decisions made, the tools chosen, the coding rules established, the workflows developed. What previously required weeks of informal knowledge transfer becomes available on day one.

5. Competitive Landscape

The memory and context problem in AI has attracted significant attention from both research institutions and commercial startups. The competitive landscape can be divided into three categories: developer infrastructure tools, single-platform memory features, and general capture products.

Product	Approach	Platform Neutral?	Structured Types?	Consumer UX?
Conxt	Passive capture → structured classification → cross-platform injection	Yes	Yes (7 types)	Yes
Mem0	Open-source developer SDK for agent memory infrastructure	Yes	No	No (developer tool)
Zep	Knowledge graph memory for enterprise AI agents	Yes	Partial	No (enterprise SDK)
Rewind.AI	Full on-device screen/audio capture and local vector search	No	No	Partial
Personal.ai	Personal knowledge base trained on user writing and conversations	No	No	Partial
OpenAI Memory	ChatGPT in-platform session memory	No	No	Yes (one platform)

5.1 Key Differentiators

Conxt occupies a distinct position in this landscape:

- Platform neutrality: Unlike OpenAI Memory or any first-party solution, Conxt works across all major AI platforms simultaneously.
- Consumer UX: Unlike Mem0, Zep, or LangMem — which are developer infrastructure tools — Conxt is designed for end users with no technical configuration required.
- Structured classification: The seven-type taxonomy provides structured, queryable memory rather than unstructured embeddings. This enables targeted retrieval and human-readable review.
- Human-in-the-loop: Users review and approve memories before they become active. This prevents hallucination propagation and gives users control over what the model knows.
- Passive capture: No workflow change required. The extension operates silently in the background.

6. Market Opportunity

6.1 Primary Market: AI-Native Professionals

The primary market is professionals who use AI as a core daily workflow tool. Based on current AI adoption data, this segment is growing rapidly:

- GitHub Copilot reported over 1.8 million paid subscribers as of Q4 2024, with enterprise adoption accelerating [5].
- ChatGPT reached 300 million weekly active users in February 2025 [6].
- Anthropic reported that Claude is used by over 90% of Fortune 500 companies for professional tasks [7].

Within this population, the highest-value users are those who use multiple AI platforms regularly — a behavior increasingly common among developers, product managers, researchers, and consultants who select tools based on task requirements rather than platform loyalty.

6.2 Secondary Market: AI-Native Teams

The team workspace feature opens a secondary market in organizations deploying AI at scale. The context problem is acute for teams because it multiplies across every member. Organizations that have invested in AI tooling but struggle to achieve consistent, context-aware outputs represent a strong secondary market.

"40% of enterprise applications will feature task-specific AI agents by 2026, up from less than 5% in 2025" (Gartner, August 2025) [8]. These agents fail without memory — and Conxt provides the memory layer.

7. Pricing Model

Conxt uses a freemium model designed to maximize individual adoption while monetizing power users and teams:

- Free Tier: Core memory capture and retrieval. Up to 500 memory records. Personal workspace only. Chrome extension included.
- Pro — $9/month: Unlimited memory records. Advanced filtering and semantic search. Memory analytics. Priority extraction.
- Team — $25/seat/month: Shared team workspace. Team memory feed. Role-based access. Onboarding memory packages. Priority support.

The pricing is positioned to be self-evident for any professional using AI regularly: the productivity recovery from eliminating context re-establishment pays for the subscription in the first week of use.

8. Product Roadmap

8.1 Current (Shipped)

- Chrome extension with MV3 content scripts for Claude, ChatGPT, Gemini, and Copilot.
- Seven-type memory classification engine (FastAPI, Railway).
- User dashboard with real-time memory feed (Next.js, Vercel).

- JWT authentication with Supabase Auth.
- Session deduplication via hash caching.
- Team workspace data model and API (teams, team_members, invite codes).

8.2 Near-Term (Next 90 Days)

- Stripe billing integration (Free / Pro / Team tiers).
- Team workspace dashboard UI.
- Invite link flow (/invite/[code] page).
- Memory injection API for direct context loading into AI sessions.
- Mobile app (Expo/React Native) for memory review on mobile.

8.3 Medium-Term (6–12 Months)

- Proactive memory surfacing: model-suggested relevant memories before session start.
- Memory relationship graph: linking related memories across types.
- API access for developers to build on the Conxt memory layer.
- Slack integration for team memory capture from non-AI conversations.
- Enterprise SSO and advanced admin controls.

9. Conclusion

The context amnesia problem is the most consequential unsolved friction in professional AI usage today. The models are extraordinary — the infrastructure around them has not kept pace. Every session starting from zero is a compounding tax on the productivity that AI is supposed to unlock.

Conxt solves this with a simple insight: memory should live outside the models, belong to the user, and work across every platform simultaneously. The seven-type classification system, passive capture architecture, and team workspace design together create a memory layer that is immediately useful, deeply personalized, and progressively more valuable with every session.

The models will continue to improve. Conxt builds the memory that makes that intelligence permanent.

conxt.dev

References

[1] Atlan Engineering. *"Memory Layer for AI Agents: How It Works and Why It Matters."* atlan.com, 2026. https://atlan.com/know/memory-layer-for-ai-agents/

[2] LangChain. *"State of Agent Engineering 2025."* LangChain Inc., 2025. https://www.langchain.com/stateofaiagents

[3] MIT NANDA Lab. *"Enterprise Generative AI Pilot Outcomes: 150 Executive Interviews."* MIT Sloan School of Management, 2025. https://mitsloan.mit.edu

[4] Chhikara, P. et al.. *"Mem0: Building Production-Ready AI Agents with Scalable Long-Term Memory."* ECAI 2025 / arXiv:2504.19413, 2025. https://arxiv.org/abs/2504.19413

[5] GitHub. *"GitHub Copilot: The AI-Powered Developer Platform."* GitHub Blog, 2024. https://github.blog/news-insights/product-news/github-copilot-the-ai-powered-developer-platform/

[6] OpenAI. *"ChatGPT — 300 Million Weekly Active Users."* OpenAI Blog, 2025. https://openai.com/blog

[7] Anthropic. *"Claude in the Enterprise."* Anthropic.com, 2025. https://www.anthropic.com/enterprise

[8] Gartner. *"Gartner Predicts 40% of Enterprise Apps Will Feature AI Agents by 2026."* Gartner Research, 2025. https://www.gartner.com/en/newsroom

[9] Tribe AI. *"Beyond the Bubble: How Context-Aware Memory Systems Are Changing the Game in 2025."* tribe.ai, 2025. https://www.tribe.ai/applied-ai/beyond-the-bubble-how-context-aware-memory-systems-are-changing-the-game-in-2025

[10] mem0.ai. *"State of AI Agent Memory 2026: Benchmarks, Architectures and Production Gaps."* mem0.ai Blog, 2026. https://mem0.ai/blog/state-of-ai-agent-memory-2026

[11] Prabhakar, A.. *"AI-Native Memory and the Rise of Context-Aware AI Agents."* ajithp.com, 2025. https://ajithp.com/2025/06/30/ai-native-memory-persistent-agents-second-me/

[12] The New Stack. *"Memory for AI Agents: A New Paradigm of Context Engineering."* thenewstack.io, 2025. https://thenewstack.io/memory-for-ai-agents-a-new-paradigm-of-context-engineering/