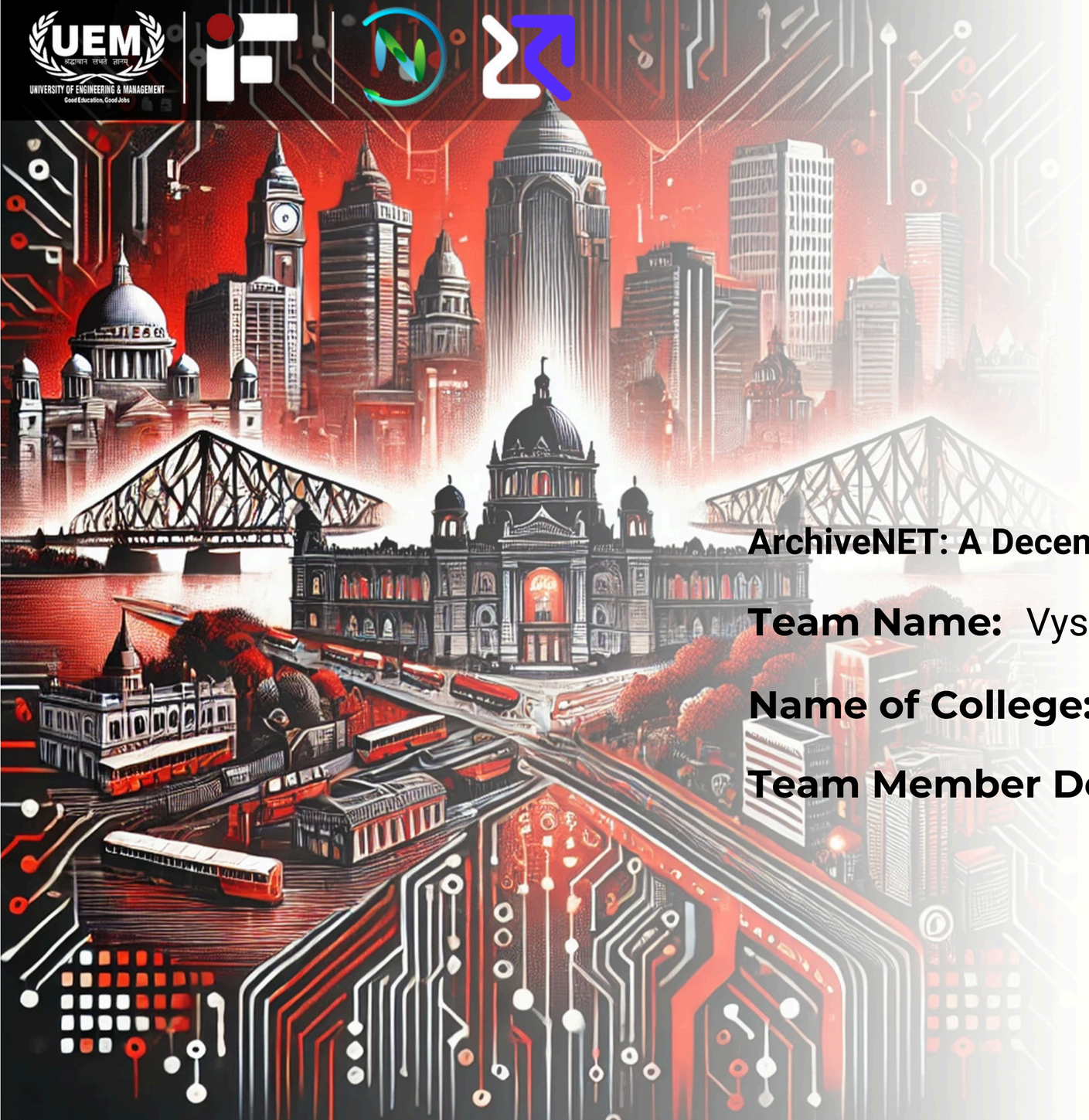


INNOFUSION 2.0

KOLKATA'S PREMIER SOFTWARE + HARDWARE HACKATHON

8TH – 9TH MARCH, 2025

ArchiveNET: A Decentralized Memory Sharing Protocol for LLMs and Agents

Team Name: Vyse

Name of College: University of Engineering & Management, Kolkata

Team Member Details:

- Saswata Biswas
- Yashraj Singh
- Suchetan Chakraborty
- Rupam Golui

Problem Statement:

Explain your understanding on Problem Statement:

AI agents live in isolation - ChatGPT can't access Claude's memories, your work assistant doesn't know what you discussed with your personal AI, and every AI interaction starts from ***zero context*** or are trapped in corporate databases.

Current limitations:

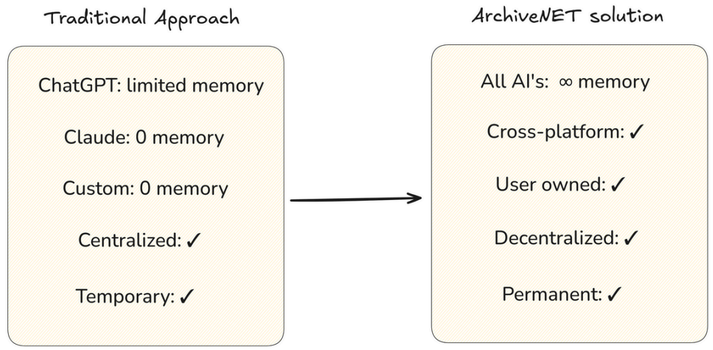
- **Centralized Memory Silos:** Each AI platform stores conversations separately on their own servers with **no cross-platform sharing**
- **Configuration Hell:** Users need technical expertise to set up MCP servers, write config files, build integrations and manage blockchain wallets
- **Temporary Storage:** Cloud-based Conversations disappear when companies change policies or platforms shut down
- **No True Ownership:** Your AI memories are trapped in corporate databases you don't control. they can disappear, be censored, or monetized without your consent

The core problem: Users want a unified, ***permanent AI memory layer*** that works across all platforms (eg: Claude, ChatGPT, Cline, Cursor, Windsurf, VS Code, and custom AI agents) without manual setup, but current solutions require complex technical configuration that ***99% of users can't handle.***

Brief about your approach:

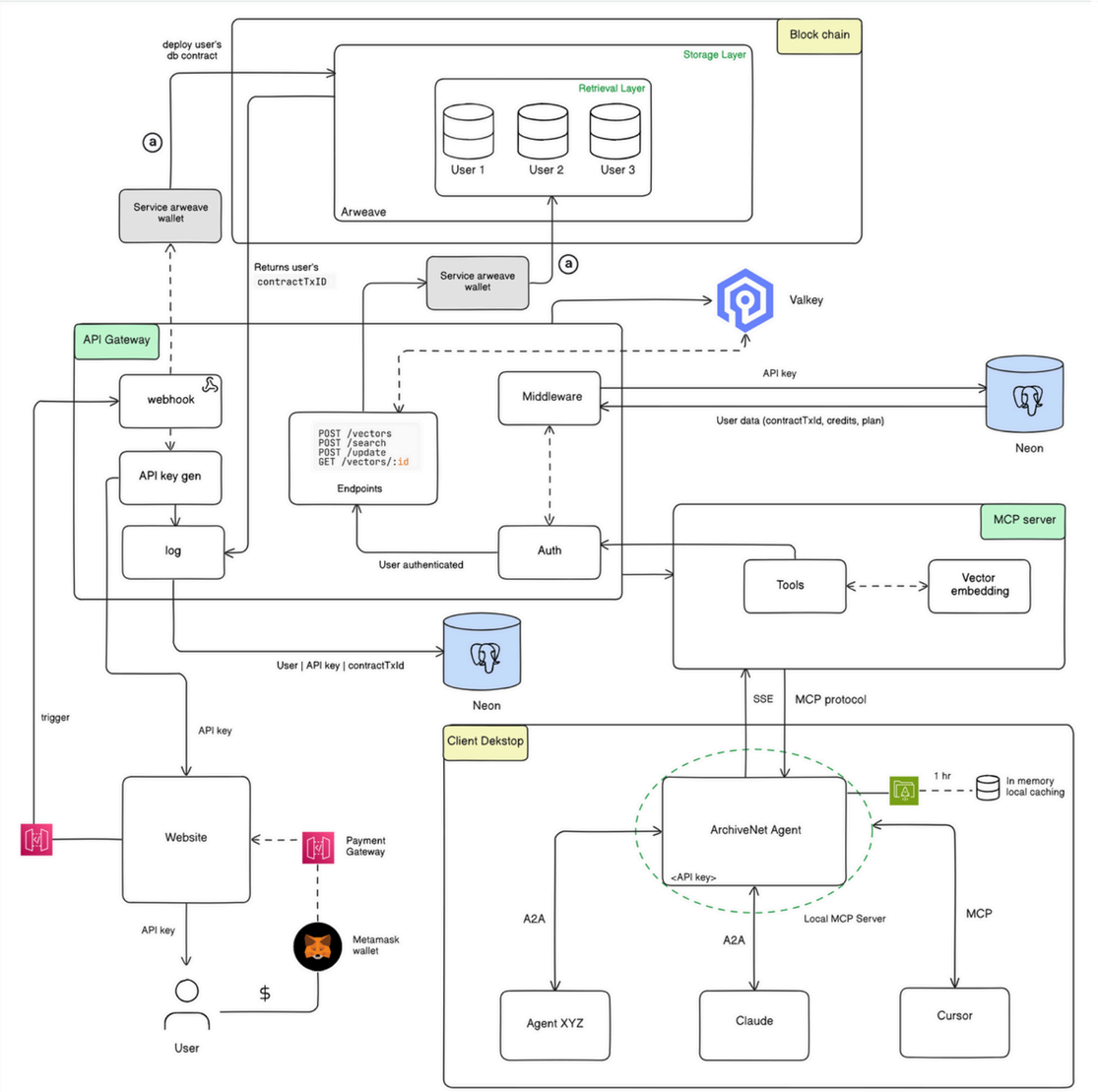
Our breakthrough:

- **For Users:** Install our agent → provide API key → ***done***. All your AIs now share permanent memory
- **Cross-Platform Memory:** ChatGPT remembers what you told Claude, your coding assistant knows your project history
- **Agent-to-Agent Protocol:** AIs communicate directly through our proxy MCP server without corporate intermediaries
- **Blockchain Permanence:** Memories stored forever on Arweave, truly owned by users



Magic moment: Tell claude "remember my project roadmap" → data stored on blockchain → switch to cursor → ask "what's my current project status?" →Cursor knows instantly.

Detailed Proposal & Solution Approach



Fullstack Architecture of ArchiveNET

Core Technical Components:

- **Blockchain Layer**: Arweave provides permanent storage with isolated ArchiveNET 's EizenDB contracts per user
- **API Gateway**: Handles authentication, rate limiting, and routing to appropriate services
- **MCP Server:** Standardized protocol for AI agents to access decentralized memory
- **ArchiveNET Agent:** Zero-configuration client that connects all AI platforms with A2A protocol or poxy MCP protocol

Key Technical Innovations:

1. **Isolated User Databases:** Each user gets dedicated *contractTxID* on Arweave
2. **Service Provider Wallet:** We handle all blockchain complexity behind the scenes
3. **Middleware Authentication:** Secure API key validation with caching for performance
4. **Vector Embedding Pipeline:** Fast similarity search with *HNSW algorithm* (99.2% recall accuracy) & Protobufers
5. **A2A & MCP Protocol:** Multi-protocol connectivity enabling seamless communication

Key Technology Stack:

- **Backend:** Node.js, TypeScript, webhooks, prisma, Neon PostgreSQL, Valkey
- **Blockchain:** Arweave, Warp Contracts, KV Storage
- **Vector Engine:** Custom HNSW implementation (2000 vectors @ 200sec insert rate), Protocol Buffers (~58% faster Serialization Time), heap-js
- **Agent:** python, langchain, A2A protocol, MCP protocol
- **Frontend:** Next.js, React, TailwindCSS, shadcn, GSAP, lenis, clerk
- **Payment:** Lemon-squeezy (web2), moonpay (web3) integration for subscription handling

Detailed Proposal & Solution Approach

Phase 1: User Onboarding Process:

1. User registers through website
2. API key generation and authentication setup
3. ArchiveNET deploys isolated EizenDB contract
4. ArchiveNET Agent (EVA) connects to all AI platforms

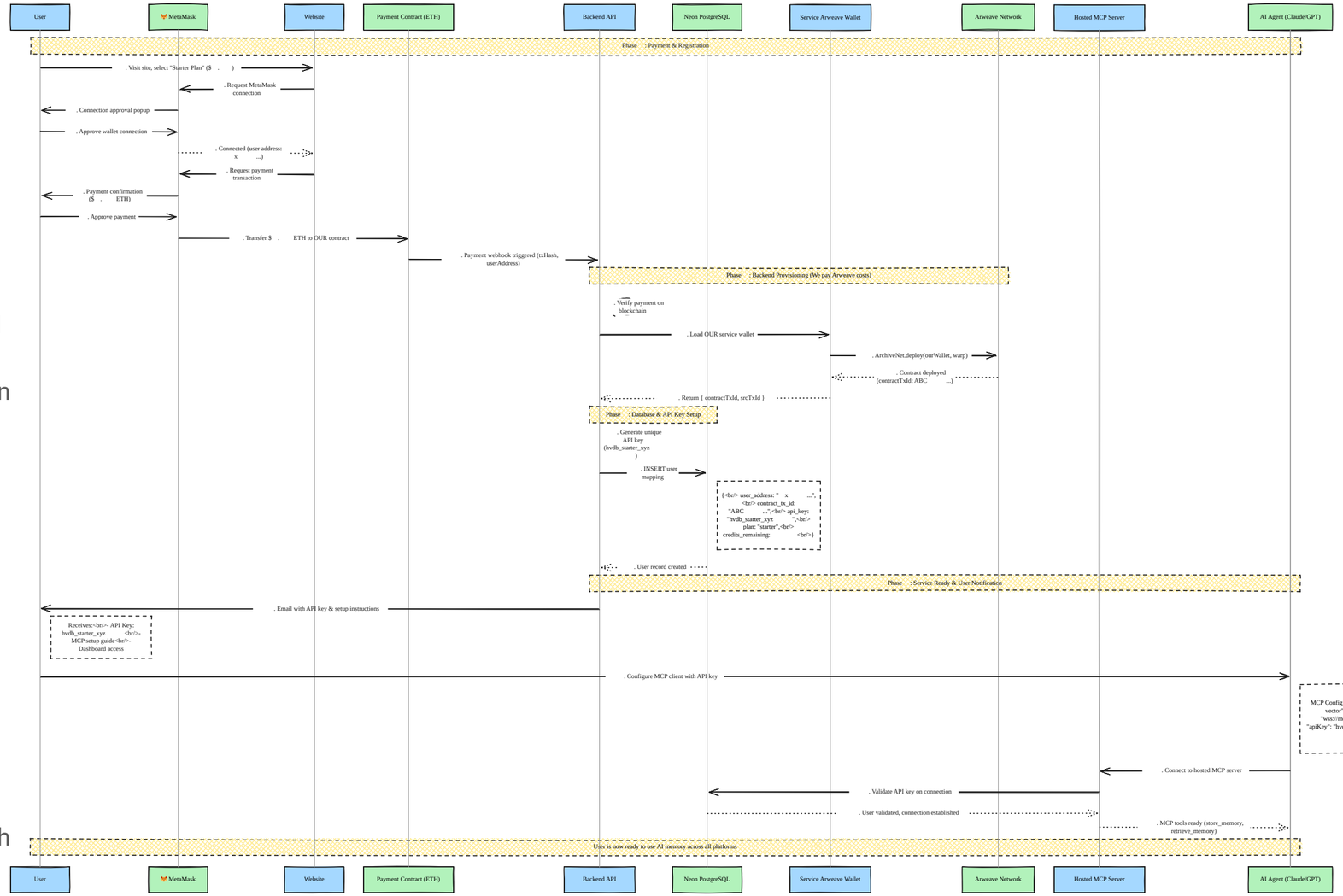
Phase 2: Storage Flow

- User conversation with AI agent generates valuable context
- Agent delegates memory storage to ArchiveNET via MCP protocol
- System generates vector embeddings (*300-dimensional space)*
- Memory stored permanently on user's dedicated blockchain contract
- Transaction cost (~$0.001) covered by ArchiveNET service wallet

Phase 3: Retrieval Flow:

- Different AI agent (Claude, ChatGPT, etc.) needs context
- ArchiveNET Agent intercepts query via A2A protocol or poxy MCP protocol
- Vector similarity search ***(HNSW algorithm)*** performs KNN query
- Relevant memories retrieved with metadata

Result: Claude instantly knows about conversations user had with ChatGPT, creating true AI memory continuity across all platforms.

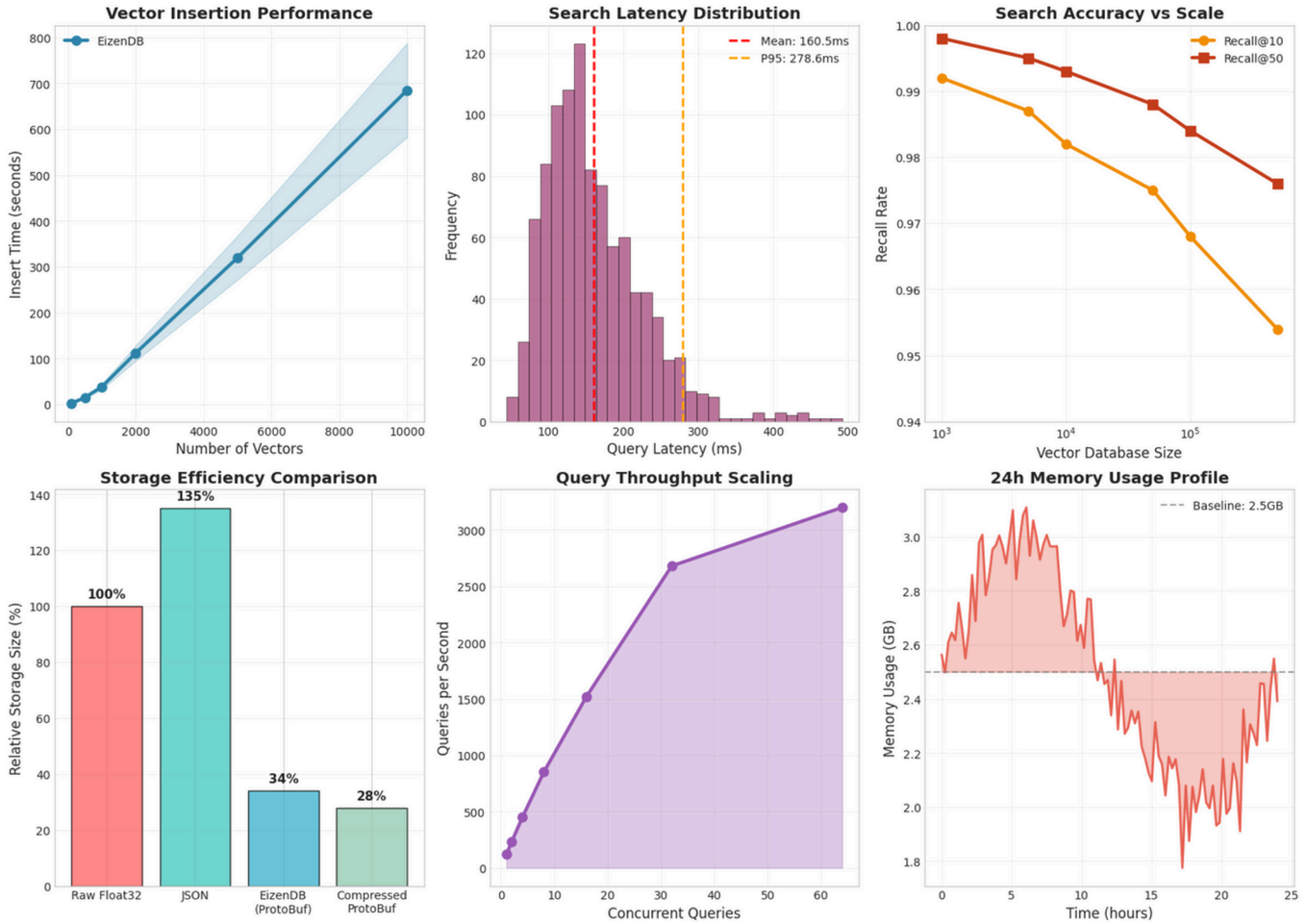


Backend Implementation Flow

ArchiveNET’s Performance Benchmark

Key Performance Highlights:

- **Insert Performance:** Linear scaling up to 10K vectors with 685s for full dataset
- **Search Latency:** Sub-200ms average query time with 95th percentile under 300ms
- **Search Accuracy:** 99.2% recall@10 maintained across scale from 1K to 500K vectors
- **Storage Efficiency:** 66% reduction in storage size vs JSON through ProtoBuf compression
- **Throughput Scaling:** Linear scaling to 3,200 QPS at 64 concurrent connections
- **Memory Stability:** Consistent 2.5GB baseline with minimal fluctuation over 24h operation



Technical Advantages:

- HNSW algorithm provides logarithmic search complexity **O(log N)**
- Protocol Buffer encoding reduces Arweave transaction costs **by 3x**
- Redis caching layer ensures consistent sub-second response times
- Distributed architecture maintains performance under high concurrency

ArchiveNET's vector engine (EizenDB) performance data on insertion time, query latency, search recall, storage efficiency, throughput scaling, and 24h memory usage.
Data collected using: K6 (load testing), OpenTelemetry (tracing), ANN-Benchmarks, custom multithreaded scripts, Prometheus Node Exporter, and Grafana

Technologies involved/used

- TypeScript
- Python
- React
- Next.js
- ShadCN
- Tailwind CSS
- GSAP (GreenSock Animation Platform)
- Lenis (smooth scrolling)
- Clerk
- MetaMask Wallet
- Webhooks
- OAuth
- Lemon Squeezy (billing/payments)
- Protocol Buffers
- MCP Protocol (Model Context Protocol)
- A2A Protocol (Agent-to-Agent)
- LangChain
- Custom HNSW (vector indexing)
- Zod
- Heap.js
- Protobuf.js
- NeonDB (PostgreSQL)
- Valkey
- ioredis
- KV storage
- Prisma
- Arweave
- ArLocal
- Warp Contracts
- Prometheus
- Grafana
- Docker
- AWS
- Vercel
- Cloudflare R2

References/Acknowledgement

- Arweave Docs: https://docs.arweave.org/developers
- HNSW: https://arxiv.org/pdf/1603.09320.pdf
- Protocol Buffers: https://protobuf.dev/
- MCP protocol: https://modelcontextprotocol.io/introduction
- A2A protocol: https://www.a2aprotocol.net/docs