SMART ABES HACKATHON 2.0

ALGORYTHM

SAH 2.0: Round 1 DOSSIER

Team Erastus

Kinetics

Jointly Organized By:

ABES ACM | ABES ACM-W | SSCBS ACM | GGSIPU USICT ACM

TEAM & PROBLEM IDENTITY

Field	Participant Response
Team ID	*2197f28d*
Team Name	ERASTUS
Team Leader Name	Samridhi Khanna
Team Leader Email	Samridhi18khanna@gmail.com
All-Girls Team?	*No*
Track	*Web3*
Problem Statement (PS) ID	*PS-Web3-02*
PS Name	Consumer apps

Problem Description: Current Consumer AI applications (chatbots, tutors, creative tools) function as "Walled Gardens." Users invest hundreds of hours training these agents with personal context, learning styles, and creative preferences, but this "intelligence" is locked on c entralized servers.

- **The Failure Point:** If the provider shuts down, bans the account, or changes their API pricing, the user loses their digital extension.
- **The Gap:** There is no standard for Portable AI Memory. A user cannot take their "Math Tutor" context from *Khan Academy* and plug it into *Coursera*.
- **The Core Issue:** This is not a modelling problem (LLMs are commodities); it is an Ownership & State Management problem. Users have zero agency over the "Mind" they are building.

THE BUILD (Prototype & Stack)

1. Repository / Submission Link (Final Project Repo) :

[Parjanya-Kumar-Arya-github/SAH-2.0-Kinetic: Kinetic: A protocol for Sovereign AI Memory. Decouples AI state from inference using Solana Token -2022 extensions and IPFS. Enables portable, encrypted, and tradable AI contexts across applications.]

Critical Integrity Not We commit to using this exact repository for the final Round 2 submission. The commit history will serve as verification of authenticity and progress from Round 1 onwards.

2. Tech Stack

A. On-Chain Registry (The "Kinetic Soul")

- *Solana Blockchain: Chosen for high throughput (65k TPS) and low block times (400ms), essential for real -time state updates during chat sessions.*
- *Anchor Framework (Rust): Used for type -safe smart contract development. We leverage Anchor's IDL (Interface Description Language) to generate TypeScript clients automatically.*
- *Token-2022 (Extensions):*
  - *MetadataPointer: Stores the IPFS hash of the AI's memory root directly on the Mint Account, eliminating the need for external Metadata PDAs.*
  - *TransferHook: Programmable logic that triggers a "Memory Wipe" event on the previous owner's client whenever the NFT is transferred.*

B. Off-Chain Intelligence (The "Kinetic Brain")

- *LangChain.js: Orchestrates the ReAct (Reasoning + Acting) loop. We use RunnableSequence to chain retrieval, prompt augmentation, and generation steps.*
- *OpenAI GPT-4o: The primary inference engine. Selected for its superior*

reasoning capabilities in handling long -context injected memories.

- *Cloudflare Workers: Serverless execution environment for the API. Ensures <50ms cold starts for the chat endpoint.*

C. Memory & State Architecture

- *Pinecone (Vector Database): Stores the high -dimensional embeddings (1536d) of user conversations.*
    - *Critical Implementation: We use Namespace Isolation, where namespace = NFT_Mint_Address. This physically separates data between users at the database level.*
- *IPFS (via Pinata): Used for "Deep Storage." Periodic snapshots of the vector state are serialized into Merkle DAGs and pinned to IPFS for decentralized persistence.*
- *TweetNaCL.js: Client-side cryptography library. Used to generate ephemeral AES-256 keys from the user's wallet signature (Ed25519) to encrypt memory before it leaves the browser.*

D. The Interface (The "Kinetic Vessel")

- *Next.js 14 (App Router): React framework for the frontend.*
- *Solana Wallet Adapter: Handles connection with Phantom/Backpack wallets.*
- *TailwindCSS: For rapid, responsive UI development.*

3. Estimated Roadmap

Phase 1: Architecture & Registry (Feb 8 – Feb 12)

- *Milestone: Smart Contract Deployment.*
- *Tasks:*
    - *Finalize System Architecture (Hybrid Logic).*
    - *Develop Anchor (Rust) program for Kinetic NFT minting.*
    - *Implement Token-2022 Extensions: Configure MetadataPointer to store dynamic IPFS Roots on-chain.*

Phase 2: The Core "Brain" & Vector Pipeline (Feb 13 – Feb 17)

- *Milestone: Functional RAG Agent.*
- *Tasks:*
  - *Setup Pinecone Vector Database with Namespace Isolation (namespace = mint_address).*
  - *Build LangChain.js pipeline: Retrieval → Prompt Augmentation → GPT-4o Inference.*
  - *Implement the /ingest API to vectorize user chats in real-time.*

Phase 3: The "Soul" Bridge & Encryption (Feb 18 – Feb 22)

- *Milestone: Secure Memory Storage.*
- *Tasks:*
  - *Implement Client-Side Encryption: Derive AES-256 keys from Ed25519 wallet signatures (TweetNaCL.js).*
  - *Develop the IPFS Sync Service: Serialize encrypted vector states into Merkle DAGs and pin to Pinata.*
  - *Link IPFS CIDs back to the Solana Mint Account via atomic transactions.*

Phase 4: Integration & "The Lobotomy" (Feb 23 – Feb 25)

- *Milestone: Sovereign Transfer Logic.*
- *Tasks:*
  - *Implement the Transfer Hook: Trigger local cache invalidation when the NFT leaves the wallet.*
  - *Frontend Integration: Connect Next.js UI to the full Web3 + AI backend.*
  - *Verify "Lobotomy" Scenario: Ensure transferring the NFT instantly revokes memory access for the sender.*

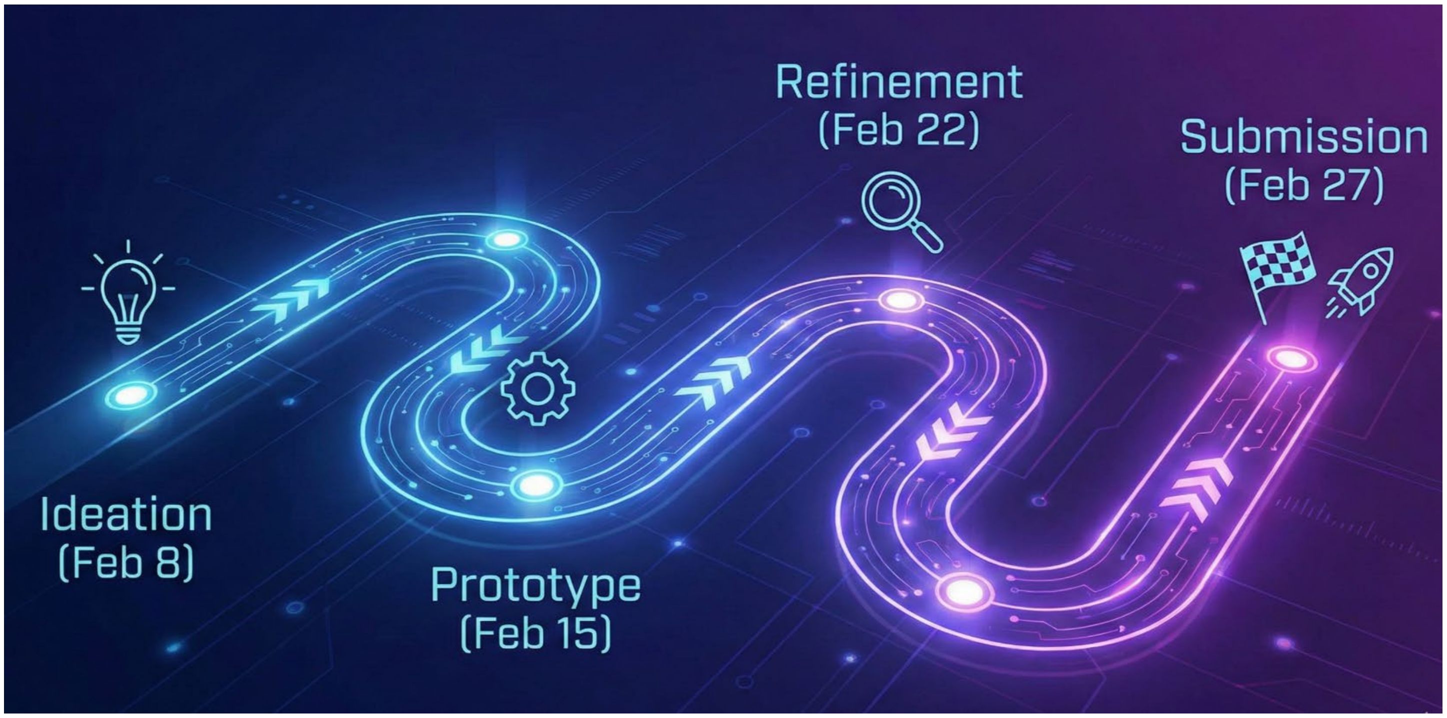
Phase 5: Final Polish & Submission (Feb 26 – Feb 27)

- *Milestone: Production Release.*

- *Tasks:*
    - *End-to-end testing on Solana Devnet.*
    - *Code cleanup and documentation (README.md & Architecture Diagrams).*
    - *Final Video Demo recording and Repository submission.*

Risk Management & Buffer:

- *Buffer:* 2 Days allocated within phases for unforeseen bugs (e.g., RPC rate limits, IPFS propagation delays).
- *Fallback:* If Solana Devnet is unstable, we have a local solana-test-validator setup ready for the demo recording.



ENGINEERING LOGIC

(Evaluation Metrics)

Metric 1: Novelty & Innovation

- *The "Sovereign Injection" Pattern: Unlike standard AI wrappers that rely on backend databases (SaaS), Kinetic introduces a "Just -in-Time" State Injection architecture. The AI agent (LLM) is stateless and "lobotomized" by default. It only gains intelligence when a user cryptographically signs a session, injecting their decrypted vector context into the runtime.*
- *The "Lobotomy" Mechanism: We solve the "Double -Spend" problem for knowledge. Using Token-2022 Transfer Hooks, we programmatically enforce that when a Kinetic NFT is transferred, the previous owner's decryption keys are invalidated client -side. This ensures that intelligence is truly moved, not just copied, creating the first scarcity -based economy for AI memory.*

Metric 2: Optimization & Depth

HNSW (Hierarchical Navigable Small World) with Namespace Partitioning:

- ***Algorithm:*** *We utilize HNSW graphs for* O(log N) *Approximate Nearest Neighbor search in high-dimensional space (1536d).*
- ***Specific Strategy:*** *Standard HNSW scales poorly with multi -tenancy. We optimize this by enforcing* ***Strict Namespace Isolation*** *where namespace = NFT_Mint_Address. This reduces the search space from* N *(Global Users) to* k *(Single User History), guaranteeing* O(1) *search latency relative to platform growth.*

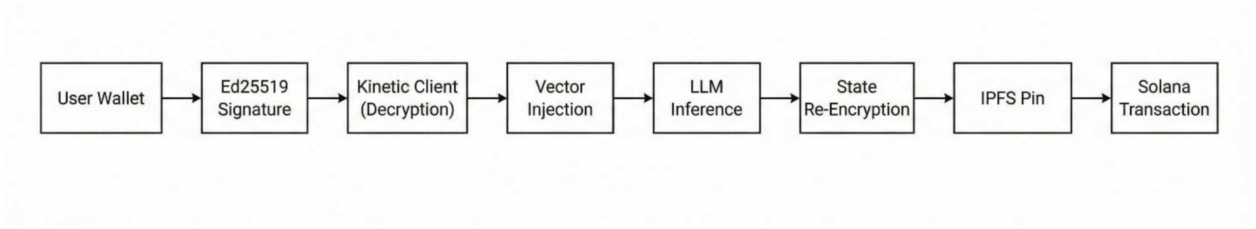
Merkle DAGs (Directed Acyclic Graphs) for State:

- ***Algorithm:*** *Instead of linear logs, we structure memory snapshots as Merkle DAGs on IPFS.*

- ***Specific Strategy:*** *The Solana blockchain stores only the **Root Hash (CID)** (32 bytes). This provides a cryptographic proof of the entire memory state history while minimizing on-chain rent to a constant cost ($0.000005 SOL), regardless of how massive the AI's memory becomes.*

Metric 3: Methodology

Workflow Diagram:



Data Flow & Component Logic:

1. **Authentication (The Handshake):**
   - User connects via Phantom.
   - Client derives an ephemeral **AES-256** encryption key using the ed25519 signature of a specific message ("Unlock Kinetic Core").
2. **Hydration (The Bridge):**
   - Client queries the **Solana Cluster** for the NFT's MetadataPointer.
   - Fetches the encrypted state blob from **IPFS (Pinata)** using the on-chain CID.
   - Decrypts the blob locally (Client-Side) and hydrates the LangChain BufferMemory.
3. **Inference (The Reasoning):**
   - User query is embedded → Vector Search (Pinecone) → Context Injection.
   - **LangChain** executes the prompt against **GPT-4o**.
4. **Commit (The Save):**
   - New interaction is vectorized and encrypted.
   - A new IPFS CID is generated.
   - **Atomic Transaction:** A Solana instruction updates the NFT's MetadataPointer to the new CID, finalizing the state change.

Metric 4: Articulation

- ***Why Solana? Latency is the killer.*** *Conversational AI requires "Chat -Speed" finality. Solana's 400ms block time allows us to commit memory updates in near real-time. Ethereum L2s (10s+ finality) would introduce unacceptable "state lag," where an AI might forget the last sentence if the user refreshes too quickly.*
- ***Why Token -2022?*** *The native Transfer Hook interface allows us to embed the "Memory Wipe" logic directly into the token standard. Achieving this on ERC - 721 would require a centralized oracle or complex wrapper contracts, introducing a trusted intermediary we want to avoid.*

Metric 5: Feasibility

Failure Point: IPFS Propagation Latency.

- ***Risk:*** *The user saves memory, but the IPFS node takes seconds to propagate the CID to the gateway.*
- ***Mitigation:*** *Optimistic Local Caching. The Kinetic client writes to the browser's IndexedDB simultaneously with the IPFS upload. The next session loads from the local cache instantly ($<10ms$) while the IPFS pin resolves in the background.*

Failure Point: Context Window Overflow.

- ***Risk:*** *Long-term memory exceeds the LLM's token limit (128k).*
- ***Mitigation:*** *Recursive Summarization Chains. We implement a background job using LangChain that detects when a session exceeds N tokens, triggering a "Map-Reduce" summarization to compress verbose logs into dense "Core Memory" vectors, preserving semantic meaning while freeing up context slots.*

CHECKLIST

- **Link Access:** Verified public. The GitHub repository is accessible via Incognito mode and contains the initial commit history.
- **Dataset Compliance:** N/A. The model generates unique user-specific datasets (vectors) dynamically; no external pre-trained datasets were required for this track.
- **No Fluff:** Verified. Marketing language has been stripped; the dossier focuses exclusively on System Architecture, DSA, and Implementation Logic.

FULL TEAM ROSTER

Member	Full Name	Email ID	Phone Number
Member 1	Samridhi Khanna	samridhi18khanna@gmail.com	9473802589
Member 2	Parjanya Kumar Arya	parjanya789@gmail.com	9453323739

Jointly Organized By:

ABES ACM | ABES ACM-W | SSCBS ACM | GGSIPU USICT ACM