Assignment 2: User Stories & On-Chain Requirements

Name: Anudeep Avula

Address: 2SZyg3ZgvmpE2FkqKnhtyz5xj4twnVcv8vaDyrVJszuv
Project: MapsDotFun Real-Time Behavioral Clustering & Risk Visualization for Solana Tokens

Part A: Initial User & Function Mapping

1. Manual User Brainstorming

Category	Potential Users
Direct Users	Retail Solana Traders, Active Token Investors, Crypto Analysts / Researchers
Indirect Users / Beneficiaries	Token Founders & Teams, General Crypto Community, Content Creators
Administrators / Moderators	Platform Administrator (you), Community Moderators
Stakeholders	Solana Ecosystem Partners, Security Auditors, Token Holders, Integrators (API Consumers)

2. AI-Assisted User Prioritization (Summary)

1. **Retail Solana Traders** — need fast, trustworthy risk signals before trading.
2. **Active Token Investors** — depend on clear clustering and ownership data for investment decisions.
3. **Crypto Analysts / Researchers** — interpret and validate behavioral patterns for public reports.
4. **Platform Administrator** — manages indexing jobs, smart-contract anchors, and data consistency.

Final Selection & Rationale

Priority	User Type	Why Included
1	Retail Traders	Core proof of utility: real-time, visual risk detection.
2	Token Investors	Validate trustworthiness and influence scores.
3	Crypto Analysts	Provide depth, accuracy, and interpretability to analytics.
4	Platform Admin	Ensures system reliability and handles Merkle-root anchoring.

---

3. Core Function Mapping

User Type	Core Functions / Interactions
Retail Trader	Search token → view ownership graph → view risk score → filter wallets → bookmark token.
Token Investor	Compare tokens → set alerts → analyze historical risk trends.
Crypto Analyst	Access raw metrics API → export data → submit feedback on risk model accuracy.
Administrator	Deploy indexer program → initiate snapshot updates → anchor Merkle root on Solana → manage backend jobs.

---

4. Derived Core POC Requirements

For a working POC, the **critical interaction path** is:

Trader → Search Token → View Ownership Map + Risk Score.

Key Technical Requirements

- API to fetch real-time token holder snapshot (from Helius/SolanaFM).
- Backend service to process wallet graph and compute risk index.
- PostgreSQL schema for token records, holders, risk metrics.
- Visualization component (D3.js) to render clusters and ownership edges.
- Anchor program to store Merkle root and timestamp for proof-of-snapshot.

Part B : Adversarial Analysis & Granularity Check

1 Critique & Refine User Stories / Requirements:

- Some stories merged UI and backend logic (“view map and score”).
- Requirements needed clear state variables and specific on-chain functions.
- Anchoring mechanism must identify root hash and authority address.

Refinements Applied

Issue	Fix	Result
Combined actions	Split into atomic steps	Each story = one user action
Vague “store data on chain”	Defined explicit account + fields	Maps to Anchor schema
Missing system owner	Added admin authority for updates	Security and auth clarified

---

Part C : Granularity & Clarity Refinement Log

1. Final Manual Review & Refinement

Before	After	Rationale
“Trader views ownership map and risk score”	“Trader views ownership map.” “Trader views risk score.”	Split for atomicity.
“System creates on-chain record.”	“Admin initializes Solana account with mint address and snapshot root.”	Adds technical clarity.
“Risk algorithm calculates score.”	“Analytics engine computes risk score from wallet concentration and tx frequency.”	De-jargonized and specific.

Part D :Defining Potential On-Chain Requirements

1. Brainstorming On-Chain Requirements for Each User Story

User Story	Potential On-Chain Requirements
Admin initializes token snapshot	Create PDA account with mint_address, snapshot_root, timestamp. Store admin authority.
Admin updates snapshot	Instruction: update_snapshot(mint_address, new_root) → verifies authority & updates state.
Trader queries snapshot proof	Read-only RPC call fetches latest root; verify Merkle proof off-chain.
Trader views risk score	Risk index stored off-chain but anchored hash available on-chain for auditability.
Analyst validates data integrity	Use Merkle proof to confirm snapshot matches anchored root on Solana.

Process Appendix (Summary)

AI Prompts Used

- “List potential user types for MapsDotFun based on value proposition.”
- “Prioritize the most critical 2–5 user types for POC.”
- “Map key functions for these user types.”
- “Derive technical requirements from top interactions.”
- “Critique granularity and atomicity of stories vs value proposition.”

Key Decisions

- Focused POC on Retail Trader flow for clarity and impact.
- Limited on-chain scope to Merkle-root anchoring for verifiable integrity.
- Chose Helius API for indexing efficiency and fast iteration.

Refinement Rationale

- Each story reduced to single action and observable result.
- All on-chain steps map directly to Anchor instruction or account field.
- Language kept non-technical for stakeholder comprehension.