Part A: Initial User & Function Mapping

1. Manual User Brainstorming

Objective:

Identify all potential user types who might interact with or benefit from StreamVault.

Context Recap – Value Proposition:

StreamVault is a fully on-chain subscription vault protocol on Solana. It enables users to prepay into programmable escrow vaults that release funds over time or based on milestone completion. Users may cancel anytime and reclaim unused funds, while providers receive predictable, trustless revenue. It serves creators, DAOs, SaaS tools, and developers needing flexible, composable subscription primitives.

User Typology:

Category	User Type
Direct Users	Web3 Content Creators
	DAOs / Treasury Operators
	Freelancers / Crypto-Native Service Providers
	Subscription-Based dApp Builders
	Web3 SaaS Tool Developers
Indirect Beneficiaries	Subscribers / Content Consumers
	Clients of Freelancers
	DAO Token Holders / Community Participants
Administrators	Protocol Developers (e.g., founding team)
	Governance Actors (future DAO voting participants)
	Community Moderators
Stakeholders	Solana Ecosystem Partners
	Grant Reviewers / Investors
	Security Auditors
	Dev Tooling Maintainers

---

2. AI-Assisted User Prioritization

Prompt to AI:

My project's value proposition is:

“StreamVault is a fully on-chain subscription vault protocol on Solana. It allows users to prepay into programmable escrow vaults that unlock funds over time or upon hitting on-chain milestones. Users can cancel subscriptions anytime and reclaim unused funds; providers benefit from trustless, milestone-based payouts. It is composable and ideal for DAOs, content creators, SaaS tools, and subscription ecosystems.”

Here is a brainstormed list of all potential user types:

- Web3 Content Creators
- DAOs / Treasury Operators
- Freelancers / Crypto-Native Service Providers
- Subscription-Based dApp Builders
- Web3 SaaS Tool Developers
- Subscribers / Content Consumers
- Clients of Freelancers
- DAO Token Holders
- Protocol Developers
- Governance Participants
- Community Moderators
- Solana Ecosystem Partners
- Grant Reviewers / Investors
- Security Auditors
- Dev Tooling Maintainers

Based on the value proposition, which 2-5 of these user types are most critical to focus on for a proof-of-concept? Provide a rationale for each.

AI Response (Summarized):

Recommended POC User Types:

1. **Web3 Content Creators**
   Reason: They are ideal early adopters for milestone-based subscriptions and will directly use the protocol. Their needs (cancellations, refunds, unlocks) align tightly with the value prop.
2. **Freelancers / Service Providers**
   Reason: They benefit from milestone-triggered payments and trust-minimized escrow. This validates StreamVault’s use as a modular payout mechanism.

3. **DAOs / Treasury Operators**
   Reason: DAOs need trustless payment flows for recurring services. StreamVault allows more flexible and accountable treasury disbursement.
4. **Subscribers / Consumers**
   Reason: Their ability to reclaim unused funds validates the “cancel-anytime” aspect, which is essential to prove the protocol is fair and user-centric.
5. **Web3 SaaS Builders**
   Reason: As protocol integrators, they validate the composability and developer-friendliness of StreamVault.

Manual Synthesis & Final POC User Selection:

We agree with most of the AI's prioritization but will **focus on the following 3 user types** for the POC:

1. **Web3 Content Creators** – They validate milestone-based unlocking and composability.
2. **Freelancers / Service Providers** – They are real users of escrow logic and benefit from trustless milestone triggers.
3. **DAOs / Treasury Managers** – Their use case helps prove refundability, cancelability, and recurring service flows.

Rationale:
These three represent distinct yet overlapping user bases that align tightly with StreamVault's programmable escrow logic. Together, they test the core mechanics of deposits, time-based or milestone-based unlocks, and refunds.

---

3. Core Function Mapping by User Type

Prompt to AI:

For a project with the value proposition described above and focusing on these prioritized user types:

- Web3 Content Creators
- Freelancers / Service Providers
- DAOs / Treasury Managers

Help map out the key functions or interactions each user would need to perform.

AI Output – User Functions:

Freelancer / Provider - Share a vault address with DAO/client.
- Track unlocks tied to milestone proof.
- Claim partial funds upon work submission.

DAO / Treasury Manager - Create a vault for recurring contributor payment.
- Approve or reject milestone completion.
- Cancel the vault if contributor is non-performing.
- Reclaim unused funds to treasury.

4. Deriving Core POC Requirements

Manual Step: Select Critical User Interactions

For the Proof-of-Concept, we choose:

1. **Vault Creation with Milestone Configuration** (by a DAO or creator)
2. **Cancel Vault and Reclaim Unused Funds** (by user/DAO)

Prompt to AI:

Based on these two critical interactions:

- Vault Creation with milestone configuration
- Vault Cancellation and refund of unused funds

What are the key technical requirements needed to build a proof-of-concept?

AI Output – POC Technical Requirements:

1. Vault Creation with Milestone Logic

- Instruction to create a new Vault PDA storing:
    - Creator/DAO public key
    - Receiver public key
    - Total deposit amount
    - Milestone unlock schedule (e.g., block time or custom signal)
    - Funds remaining
- Must validate input milestone format at creation time
- Funds held in escrow under a Vault account controlled by the program
- Event logging (e.g., VaultCreated, MilestoneDefined)

2. Vault Cancellation + Refund Logic

- Instruction to cancel vault if not all milestones completed
- Calculate unused funds based on elapsed time or milestone index
- Transfer unused funds back to creator/DAO
- Mark vault as "cancelled" to prevent further withdrawals

- Emit VaultCancelled and FundsRefunded events

Part B: Adversarial Analysis & Granularity Check

1. AI Critique of Stories & Requirements

Prompt to AI:

Review my user functions:

- Vault creation with milestone unlock schedule
- Vault cancellation and refund logic

And the associated technical requirements.
Do these stories truly validate the value proposition? Are the requirements granular enough? Are any parts unclear or missing?

AI Critique Output:

- **Clear Value Fit:** Both interactions strongly align with StreamVault's core value proposition of trustless, cancelable, milestone-based payments.
- **Missing Edge Case Handling:** You may need to clarify behavior for:
  - Missed milestones (i.e., did the user not complete it?)
  - What defines "milestone complete" in a trustless setting?
- **Granularity:** The requirements are generally well-scoped and map cleanly to PDA creation, SOL transfers, and event logs.
- **Implied Trust Layer:** If milestone completion depends on an off-chain signal (e.g., DAO votes or API), clarify how that is validated (e.g., via multisig, oracle, or proof uploads).

Manual Refinement Summary:

- Added "milestone validation" logic to clarify how an unlock is triggered (off-chain vs. DAO approval).
- Defined that POC will assume a **manual milestone approval** by a designated authority (DAO signer or Creator).

Conclusion: Part A + B Final Output Summary

Component	Summary
Final POC Users	Web3 Content Creators, Freelancers, DAO Treasury Managers
Critical POC Interactions	Vault Creation with Milestone Unlocks; Vault Cancellation & Refund
Derived Technical Requirements	PDA creation, milestone schedule storage, cancellation logic, refunds
Refinement Notes	Added milestone approval method; clarified edge-case handling