

# Turbin3 User Stories & On-Chain Requirements

## Prioritized list of top key users :

1. Content Creators
2. Fans / Users interacting with Creator tokens

## Creator Flows

### User Story : "Creator Profile Setup"

#### Potential On-Chain Requirements

- Need some front-end for the user to interact with
- Need validations for creators to sign up
- Need the creator to link their wallet
- We will store creators basic details and with that, their unlisted launch video link posted on Youtube which will be used by the users to identify their creators validity.
- Creators information like `handle` , `bio` , `social_links` URLs will be stored on database.

### User Story : "Creator Token Creation"

#### Potential On-Chain Requirements

- A frontend for the creator with required fields to create a token
- Let user pick basic token details which they will be creation
- Use solana spl token to create mints

## **User Story : "Token Dashboard"**

### **Potential On-Chain Requirements**

- Creators can check how their token is doing
- Creators can check the engagement with users
- Creators will have graphs about the health of their token
- We will store graph details off-chain

## **User Story : "Community Engagement"**

### **Potential On-Chain Requirements**

- Creators can access community engagement page where they can post unique content for their fans for the amount they have spend
- Post functionality where they can select the amount of token user should have to view the post.
- Fans will be shown blurred image / content if they do not have the token amount required which the user is raising

## **Fan Flows**

### **User Story : "Browse & Discover Creators"**

#### **Potential On-Chain Requirements**

- Fans will login from their account with basic details like user name and password
- Fans will be able to link their wallets
- Fans can search their creator tokens
- Fans can interact with token details like price, amount and basic information about the token.
- Data used here will be stored off-chain, while some specific token data may be fetched on-chain

## **User Story : "Buy Creator Token"**

### **Potential On-Chain Requirements**

- Fans will be able to buy tokens directly after browsing by interacting and signing contracts.
- Fans will be able to check out creator specific content for the amount of token they hold of their creator.

# **Process Appendix**

## **Manual Brainstorm**

**Direct Users** : Content creators, Business owners, Individuals who wants to raise funding for their ideas, Non-government organizations

**Indirect Users / Beneficiaries** : Project owners, Fans

**Administrators / Moderators** : I will be managing the system, if this goes large scale I would like community moderators to pitch in ideas to various problems.

**Stakeholders** : Partners

### **Prioritized list of top key users :**

After taking suggestions from AI I feel that the early key users I should be focusing on should be

1. Content Creators
2. Fans / Users interacting with Creator tokens

I do not disagree with AI at this point because I initially had the same plan, and as the AI did not suggest anything new there is no point of me to drag this on. Maybe in the future we can make it for a broader audience like NGOs or Small Business Owners.

## **Core Features**

## Content Creators — Key Functions

These users are the supply side. Their primary goal is to **launch tokens, manage them, and build fan engagement**.

Function	Description
<b>Identity Verification</b>	Upload docs/social proofs to verify real identity (ensures trust in the token).
<b>Creator Profile Setup</b>	Configure public profile with bio, social links, token details, etc.
<b>Token Creation</b>	Mint an SPL (Solana) token with name, symbol, supply, and distribution rules.
<b>Perk Definition</b>	Assign utility to token (e.g., gated content, voting rights, shoutouts, merch access).
<b>Fundraising Round / Token Sale</b>	Launch an initial sale or bonding curve model where fans buy in.
<b>Analytics Dashboard</b>	Track token holders, engagement, price movement, and income generated.
<b>Reward/Perk Delivery Tools</b>	Provide gated access to perks via wallet integration or off-chain link/email.
<b>Community Engagement</b>	Post updates, polls, or interact with fans via token-gated channels.

## Fans (Users Interacting with Creator Tokens) — Key Functions

These users are the demand side. Their role is to **discover, purchase, and engage with creator tokens**.

Function	Description
<b>Browse &amp; Discover Creators</b>	View creator profiles with token info, perks, and verification badge.

Function	Description
<b>Connect Wallet</b>	Use Phantom or similar Solana wallet to interact with platform.
<b>Buy Creator Token</b>	Participate in initial sale or buy from DEX (Raydium/Orca) integration.
<b>Token Perk Access</b>	Unlock gated content, Discord roles, events, etc., based on token holdings.
<b>Sell/Trade Tokens</b>	Off-ramp or trade tokens if needed (price appreciation, speculation).
<b>Track Token Holdings</b>	See how many tokens they hold per creator and any changes in value.
<b>Receive Updates from Creators</b>	Follow creators, get notified of new perks, or voting events.

## Example Interactions / Flow (MVP)

### For Creators:

1. Sign up → Verify Identity → Setup Profile → Launch Token → Promote to fans → Track & Deliver perks

### For Fans:

1. Connect Wallet → Browse Creators → Buy Token → Access Perks → Stay Engaged → (Optional) Trade token later

### Part C :

## Review of Each User Story & Missing Granularity

### Creator Flows

#### 1. Identity Verification

- **Good:** aligns with the trust layer
- **Missing granularity:**

- What docs? Social account linking or actual KYC?
  - Where is the data stored (e.g., S3, IPFS)?
  - Who approves — manual or auto?
  - Technical Mapping:
    - DB: `creators` table with `verification_status` , `kyc_doc_url`
    - API: `POST /verify-identity` , `GET /creator/:id/verification`
    - Admin portal for moderation (if manual)
- 

## 2. Creator Profile Setup

- **Good**, but:
    - Need to define editable fields: name, bio, profile image, social links, etc.
    - Should include linking to identity verification
  - Technical Mapping:
    - DB: `creator_profiles` with columns like `handle` , `bio` , `social_links` , etc.
    - API: `POST /profile` , `GET /profile/:id`
- 

## 3. Token Creation

- **Good**, core to value proposition
  - **Missing details**:
    - Can creator choose supply? Symbol?
    - Any vesting or reserve rules?
    - Mint on demand or pre-generated mint addresses?
  - Technical Mapping:
    - On-chain: Use `@solana/spl-token` to create mint account
    - Backend: call Anchor program or use a service like [Strata Protocol]
    - DB: `creator_tokens` table ( `mint_address` , `symbol` , `supply` , `creator_id` )
-

## 4. Perk Definition

- **Important but needs clarity:**
    - What types of perks? Gated content? Discord access?
    - Thresholds? (e.g., hold 100 tokens for reward X)
  - Technical Mapping:
    - DB: `perks` table with `title`, `description`, `token_requirement`, `url`, `type`
    - API: `POST /creator/:id/perks`, `GET /perks/:creatorId`
    - Gating logic: frontend + backend token balance check
- 

## 5. Fundraising Round / Token Sale

- **Crucial for PoC**, but not detailed enough
  - Missing:
    - One-time sale or bonding curve?
    - Use case: Creators get SOL, fans get tokens
  - Technical Mapping:
    - On-chain: basic escrow program, or Strata, will use bonding curve
    - DB: `sales` table with `creator_id`, `price`, `start`, `end`
    - API: `POST /launch-sale`, `POST /buy-token`
- 

## 6. Analytics Dashboard

- **Nice-to-have for MVP**, but **not required** in PoC
  - Technical Mapping:
    - Use RPC calls to fetch token balances
    - Off-chain DB to track purchases + holders
    - API: `GET /dashboard/:creatorId`
- 

## 7. Reward/Perk Delivery Tools

- **Important for utility**, but you should **define how they're delivered**:
    - URL unlocked? Discord role? NFT airdrop?
  - Technical Mapping:
    - Token gate check on frontend or API
    - API: `GET /redeem-perk/:perkId`
- 

## 8. Community Engagement

- **Too vague** in PoC
    - Maybe start with: creator posts, polls, or token-gated comments
  - Technical Mapping:
    - DB: `posts` table linked to `creator_id`
    - API: `POST /creator/:id/post` , `GET /feed/:creatorId`
- 

## Fan Flows

### 1. Browse & Discover Creators

- Solid. Basic search + listing page.
  - DB: `creator_profiles`
  - API: `GET /creators`
- 

### 2. Connect Wallet

- Well-defined; no issues
  - Solana Wallet Adapter
- 

### 3. Buy Creator Token

- Important, but needs precise mechanism:
  - From sale vault? P2P DEX?
- Technical Mapping:



- Smart contract or backend logic
- UI to sign + send SOL, get tokens

---

## 4. Token Perk Access

- Covered above
- Frontend token check or backend gated endpoint

---

## 5. Sell / Trade Tokens

- **Out of scope for PoC unless using DEX integration**
- Can be handled via link to external DEX like Jupiter/Raydium

---

## 6. Track Token Holdings

- Use `getTokenAccountsByOwner` RPC call
- Optional DB mirror to cache results

---

## 7. Receive Updates from Creators

- Start simple: follow creators + view a feed
- DB: `follows` , `posts`
- API: `GET /feed` , `POST /follow/:creatorId`

---

## What's Missing or Unclear

Issue	Suggestion
<b>Unclear token mechanics</b>	Define mint rules: fixed or dynamic supply, can fans resell? Any fees?
<b>Verification Levels</b>	Add clear tiers: unverified, verified-basic (social), verified-KYC
<b>Perk Delivery UX</b>	Define how perks are delivered — link, code, access token, etc.
<b>Redemption Mechanics</b>	Can perks be redeemed once? Are they logged in DB?

Issue	Suggestion
<b>Community tools</b>	Clarify what engagement actually means at MVP: feed, comments, votes?

## Part C Refinement Log

Removing unwanted emoji

## Part D : Defining Potential On-Chain Requirements

### Creator Flows

#### User Story : "Creator Profile Setup"

##### Potential On-Chain Requirements

- Need some front-end for the user to interact with
- Need validations for creators to sign up
- Need the creator to link their wallet
- We will store creators basic details and with that, their unlisted launch video link posted on Youtube which will be used by the users to identify their creators validity.
- Creators information like `handle` , `bio` , `social_links` URLs will be stored on database.

#### User Story : "Creator Token Creation"

##### Potential On-Chain Requirements

- A frontend for the creator with required fields to create a token
- Let user pick basic token details which they will be creation
- Use solana spl token to create mints

#### User Story : "Token Dashboard"

### **Potential On-Chain Requirements**

- Creators can check how their token is doing
- Creators can check the engagement with users
- Creators will have graphs about the health of their token
- We will store graph details off-chain

### **User Story : "Community Engagement**

#### **Potential On-Chain Requirements**

- Creators can access community engagement page where they can post unique content for their fans for the amount they have spend
- Post functionality where they can select the amount of token user should have to view the post.
- Fans will be shown blurred image / content if they do not have the token amount required which the user is raising

### **Fan Flows**

### **User Story : "Browse & Discover Creators"**

#### **Potential On-Chain Requirements**

- Fans will login from their account with basic details like user name and password
- Fans will be able to link their wallets
- Fans can search their creator tokens
- Fans can interact with token details like price, amount and basic information about the token.
- Data used here will be stored off-chain, while some specific token data may be fetched on-chain

### **User Story : "Buy Creator Token"**

## Potential On-Chain Requirements

- Fans will be able to buy tokens directly after browsing by interacting and signing contracts.
- Fans will be able to check out creator specific content for the amount of token they hold of their creator.

## Top 2 Critical Interactions

### 1. Content Creator Launches a Token (with identity verification)

This validates your **supply-side value proposition**: creators can mint their own identity-backed tokens and offer perks.

### 2. Fans Buy Tokens and Access Creator Perks

This tests the **demand-side**: fans become token holders and get value beyond speculation — unlocking perks, gated content, or access.

## AI Prompts

### Prompt 1 :

My project's value proposition is

A crypto-native platform that enables content creators to launch their own identity-verified tokens, allowing fans to financially and socially invest in their growth.

We empower creators with alternative monetization tools that go beyond ad revenue or centralized platforms. Fans become stakeholders, not just supporters gaining access, perks, and potential upside as creators succeed.

### **Key Value Areas:**

#### 1. Alternative Monetization for Underserved Creators

Empower mid-tier and niche creators to raise capital without relying on platform gatekeepers.

#### 2. Fan Financial Participation (Ownership Culture)

Turn loyal fans into stakeholders with social tokens that represent access, status, and long-term creator alignment.

### 3. Decentralized, Global Discovery & Backing

Use low-fee, permissionless Solana infrastructure to allow fans anywhere in the world to support and invest in creators directly.

. Here is a brainstormed list of all potential user types:

**Direct Users** : Content creators, Business owners, Individuals who wants to raise funding for their ideas, Non-government organizations

**Indirect Users / Beneficiaries** : Project owners, Fans

**Administrators / Moderators** : I will be managing the system, if this goes large scale I would like community moderators to pitch in ideas to various problems.

**Stakeholders** : Partners

. Based on the value proposition, which 2-5 of these user types are the most critical to focus on for an initial Proof-of-Concept? For each user you recommend, provide a brief rationale explaining why they are essential for proving the project's core value.

### Output :

For your initial **Proof-of-Concept (PoC)**, it's important to focus on a tightly scoped group of users who directly validate your **core value proposition**: empowering creators with crypto-native monetization tools, while enabling fans to participate in their growth. Based on this, here are the **top 3 user types** you should focus on initially:

---

### ◆ 1. Content Creators (Direct Users)

#### ✓ Why essential:

- They are the **primary value creators** on your platform.
- Their adoption and engagement prove whether your tokenization tools offer **real, viable monetization alternatives** to ads or centralized platforms.

- A successful PoC needs 3–5 creators who launch and promote their tokens, providing feedback on usability, appeal, and onboarding friction.
- 

## ◆ 2. Fans (Indirect Users / Beneficiaries)

### ✓ Why essential:

- The core innovation lies in **turning fans into stakeholders**, making them socially and financially invested.
  - Fans validate whether the model creates genuine demand for creator tokens — i.e., are they willing to buy tokens to support or access perks?
  - A working PoC needs **fan participation to demonstrate creator-fan token economies in action**.
- 

## ◆ 3. You (Administrator/Moderator)

### ✓ Why essential:

- At PoC stage, you're the only one ensuring smooth operation: from onboarding users, managing support, gathering feedback, to possibly tweaking governance or token logic manually.
  - You'll also be handling **trust, moderation, and verification** (especially if identity verification is a feature).
  - Your perspective is critical to validate **technical feasibility, creator trust, and ops burden**.
- 

## Optional (Phase-2 Consideration for MVP)

## ◆ 4. Partners (Stakeholders)

- Might not be essential for PoC, but can be useful if you need initial token liquidity (e.g. seed funders, small DAOs) or marketing reach (crypto media or tool partners).
- 

## ⊘ Not needed yet for PoC:

- **Business owners / NGOs / idea fundraisers** – these dilute the early focus and may have different needs compared to creators with active communities.
  - **Community Moderators** – only relevant when user base grows or community governance becomes active.
- 

## ✓ PoC Recommendation Summary

User Type	Role in PoC Validation
<b>Content Creators</b>	Validate creator-side monetization tools & experience.
<b>Fans</b>	Validate token demand, fan alignment, and participation.
<b>Administrator (You)</b>	Ensure smooth ops, feedback loop, creator onboarding.

Let me know if you want help crafting user flows or experiments to validate with these 3 groups.

### Prompt 2 :

For a project with this value proposition that is "A crypto-native platform that enables content creators to launch their own identity-verified tokens, allowing fans to financially and socially invest in their growth.

We empower creators with alternative monetization tools that go beyond ad revenue or centralized platforms. Fans become stakeholders, not just supporters gaining access, perks, and potential upside as creators succeed." focusing on these prioritized user types ,Content Creators, Fans / Users interacting with Creator tokens, help map out the key functions or interactions each user would need to perform.

### Output 2 :

Absolutely — mapping the **key functions and interactions** for your prioritized user types (Content Creators & Fans) is critical to shaping your MVP / Proof of Concept.

---

## Value Proposition Focus

- Empower **creators** with tools to **launch identity-verified tokens** and monetize.
- Enable **fans** to **buy, interact with,** and **benefit** from those tokens socially and financially.



### 1. Content Creators — Key Functions

These users are the supply side. Their primary goal is to **launch tokens, manage them,** and **build fan engagement.**

Function	Description
<b>Identity Verification</b>	Upload docs/social proofs to verify real identity (ensures trust in the token).
<b>Creator Profile Setup</b>	Configure public profile with bio, social links, token details, etc.
<b>Token Creation</b>	Mint an SPL (Solana) token with name, symbol, supply, and distribution rules.
<b>Perk Definition</b>	Assign utility to token (e.g., gated content, voting rights, shoutouts, merch access).
<b>Fundraising Round / Token Sale</b>	Launch an initial sale or bonding curve model where fans buy in.
<b>Analytics Dashboard</b>	Track token holders, engagement, price movement, and income generated.
<b>Reward/Perk Delivery Tools</b>	Provide gated access to perks via wallet integration or off-chain link/email.
<b>Community Engagement</b>	Post updates, polls, or interact with fans via token-gated channels.

Optional (Later Phase):

- Enable DAO-like governance (fans vote on decisions).
- Launch NFTs tied to the token for extra perks.



## 2. Fans (Users Interacting with Creator Tokens) — Key Functions

These users are the demand side. Their role is to **discover**, **purchase**, and **engage with creator tokens**.

Function	Description
<b>Browse &amp; Discover Creators</b>	View creator profiles with token info, perks, and verification badge.
<b>Connect Wallet</b>	Use Phantom or similar Solana wallet to interact with platform.
<b>Buy Creator Token</b>	Participate in initial sale or buy from DEX (Raydium/Orca) integration.
<b>Token Perk Access</b>	Unlock gated content, Discord roles, events, etc., based on token holdings.
<b>Sell/Trade Tokens</b>	Off-ramp or trade tokens if needed (price appreciation, speculation).
<b>Track Token Holdings</b>	See how many tokens they hold per creator and any changes in value.
<b>Receive Updates from Creators</b>	Follow creators, get notified of new perks, or voting events.

Optional (Later Phase):

- **Vote on Creator Proposals:** Use tokens for governance if implemented.
- **Referral / Affiliate Program:** Earn tokens for bringing in other fans.

### Example Interactions / Flow (MVP)

#### For Creators:

1. Sign up → Verify Identity → Setup Profile → Launch Token → Promote to fans → Track & Deliver perks

#### For Fans:

1. Connect Wallet → Browse Creators → Buy Token → Access Perks → Stay Engaged → (Optional) Trade token later

## ✓ MVP Checklist Based on User Interactions

Feature	Creator	Fan
Identity Verification	✓	✗
Token Launch (Minting)	✓	✗
Token Purchase UI	✗	✓
Wallet Connect	✓	✓
Creator Profile Page	✓	✓
Perk Management (e.g., unlock content)	✓	✓
Token Holding Dashboard	✓	✓
Notifications / Updates	✓	✓

Would you like wireframes or a table mapping these features to frontend/backend components or Solana primitives (e.g., PDAs, mint accounts, metadata)?

### Prompt 3:

Based on these top 1-2 critical user interactions from above, what are the key technical requirements needed to build a proof-of-concept?

### Output 3:

Great — for a focused **Proof-of-Concept (PoC)**, the top **1–2 most critical user interactions** are:

## 🔑 Top 2 Critical Interactions

### 1. Content Creator Launches a Token (with identity verification)

This validates your **supply-side value proposition**: creators can mint their own identity-backed tokens and offer perks.

## 2. Fans Buy Tokens and Access Creator Perks

This tests the **demand-side**: fans become token holders and get value beyond speculation — unlocking perks, gated content, or access.

---

## Key Technical Requirements to Support These Interactions

### 1. Solana Wallet Integration

- **Purpose:** Let users (both creators & fans) connect wallets to prove ownership and sign transactions.
  - **Tech:**
    - Use [Solana Wallet Adapter](#) with Phantom/Solflare
    - Frontend: React hooks for wallet connect/disconnect
    - Backend: Identify users via their public key (no email needed)
- 

### 2. Token Minting Logic for Creators

- **Purpose:** Enable creators to mint their own SPL tokens with supply settings and metadata.
  - **Tech:**
    - Use [@solana/spl-token](#) library
    - Creator signs a mint transaction (maybe 100,000 tokens, or use bonding curve later)
    - Optionally store metadata via Metaplex token metadata standard
- 

### 3. Basic Identity Verification (Manual or Light KYC)

- **Purpose:** Build trust in creator tokens. For PoC, keep this simple.
- **Tech:**
  - Form with social links, optional doc upload (store on Arweave/IPFS or basic S3 backend)

- Admin dashboard (just for you) to manually mark creators as “verified”
  - Add a verification badge to profiles
- 

#### ✓ 4. Token Sale / Distribution Mechanism

- **Purpose:** Let fans acquire creator tokens
  - **Tech Options:**
    - **Simplest:** Fans send SOL to a designated wallet, backend sends them tokens (pseudo bonding curve)
    - **Better:** Create a simple on-chain sale using a basic smart contract (e.g., PDA escrow vault)
    - **Even Better:** Use open-source bonding curve protocols (e.g., [Strata](#))
- 

#### ✓ 5. Perk Access / Token-Gated Content

- **Purpose:** Let fans access exclusive content/features based on token holdings
  - **Tech:**
    - Check wallet for minimum token balance (client-side or via backend validation)
    - Unlock link, video, download, or Discord role using token-gate logic
    - Optional: Integrate with [Collab.Land](#) or custom bot for Discord roles
- 

#### ✓ 6. Minimal Database (Off-chain)

- **Purpose:** Store creator profiles, token metadata, perks, verification status
  - **Tech:**
    - PostgreSQL or Firebase (lightweight)
    - Tables: users, tokens, perks, transactions (optional)
    - Or use Supabase for auth + DB + storage
- 

#### ✓ 7. Frontend UI (React/Next.js)

- **Pages Needed:**
  - Creator dashboard: Profile setup, token mint, view holders
  - Fan explore page: Discover creators, buy tokens
  - Gated perks: Viewable only if fan owns X tokens
- **Libraries:**
  - Solana Wallet Adapter
  - SPL-Token or Anchor client if you use a smart contract
  - Tailwind, ShadCN, or your preferred UI lib

---

## Optional Add-ons (If Time Permits)

- Show token price using DEX integration (e.g., Jupiter Aggregator)
- Token holder leaderboard
- Notifications (e.g., new perk posted)

---

## Minimal Viable PoC Stack Summary

Layer	Tech
<b>Frontend</b>	React / Next.js + Solana Wallet Adapter
<b>Smart Contracts</b>	SPL-Token + (optional: Anchor or Strata Protocol)
<b>Backend</b>	Node.js/Express OR serverless functions
<b>Storage</b>	Supabase / PostgreSQL for user metadata
<b>Verification</b>	Manual admin panel with form uploads
<b>Blockchain</b>	Solana mainnet-dev or devnet
<b>Hosting</b>	Vercel / Render / Netlify for frontend, optional backend

---

Would you like a complete breakdown of how to technically implement one of these flows (e.g., token minting or perk-gated access)?

#### **Prompt 4 :**

Review my core user functions/stories Identity Verification  
Creator Profile Setup  
Token Creation  
Perk Definition  
Fundraising Round / Token Sale  
Analytics Dashboard  
Reward/Perk Delivery Tools  
Community Engagement  
  
Browse & Discover Creators  
Connect Wallet  
Buy Creator Token  
Token Perk Access  
Sell/Trade Tokens  
Track Token Holdings  
Receive Updates from Creators  
and requirements

### **1. Content Creator Launches a Token (with identity verification)**

This validates your **supply-side value proposition**: creators can mint their own identity-backed tokens and offer perks.

### **2. Fans Buy Tokens and Access Creator Perks**

This tests the **demand-side**: fans become token holders and get value beyond speculation — unlocking perks, gated content, or access.

. Considering my project's refined value proposition A crypto-native platform that enables content creators to launch their own identity-verified tokens, allowing fans to financially and socially invest in their growth.

We empower creators with alternative monetization tools that go beyond ad revenue or centralized platforms. Fans become stakeholders, not just supporters gaining access, perks, and potential upside as creators succeed, do these stories truly hit the mark? Are the requirements granular enough to map to specific technical components (e.g., database schemas, API endpoints, specific blockchain programs)? What's missing or unclear?

#### Output 4:

You're doing a great job aligning your user stories with your value proposition.  
Let's go step-by-step and assess:

---

### Overall Alignment with Value Proposition

Your refined value prop:

A crypto-native platform enabling content creators to launch identity-verified tokens, empowering fans to financially and socially invest in their growth.

You're tackling this with:

- Creator onboarding + tokenization → aligns with "**crypto-native monetization**"
- Fan purchasing + perks → aligns with "**fans as stakeholders with upside and utility**"

 Your current user stories **do hit the mark thematically and functionally**.

But to build a **scalable, testable PoC**, a few stories need to be:


1. **More granular** (to map to code/db/smart contract)
  2. **Sequenced** into subflows (to clarify dependencies)
  3. **Explicit about success criteria** (e.g. for a "verified creator")
- 

### Review of Each User Story & Missing Granularity


#### Creator Flows

#### 1. Identity Verification


- **Good:** aligns with the trust layer
- **Missing granularity:**

- What docs? Social account linking or actual KYC?
  - Where is the data stored (e.g., S3, IPFS)?
  - Who approves — manual or auto?
  -  **Technical Mapping:**
    - DB: `creators` table with `verification_status` , `kyc_doc_url`
    - API: `POST /verify-identity` , `GET /creator/:id/verification`
    - Admin portal for moderation (if manual)
- 

## 2. **Creator Profile Setup**


- **Good**, but:
    - Need to define editable fields: name, bio, profile image, social links, etc.
    - Should include linking to identity verification
  -  **Technical Mapping:**
    - DB: `creator_profiles` with columns like `handle` , `bio` , `social_links` , etc.
    - API: `POST /profile` , `GET /profile/:id`
- 

## 3. **Token Creation**


- **Good**, core to value proposition
  - **Missing details:**
    - Can creator choose supply? Symbol?
    - Any vesting or reserve rules?
    - Mint on demand or pre-generated mint addresses?
  -  **Technical Mapping:**
    - On-chain: Use `@solana/spl-token` to create mint account
    - Backend: call Anchor program or use a service like [Strata Protocol]
    - DB: `creator_tokens` table ( `mint_address` , `symbol` , `supply` , `creator_id` )
-




## 4. Perk Definition

- **Important but needs clarity:**
    - What types of perks? Gated content? Discord access?
    - Thresholds? (e.g., hold 100 tokens for reward X)
  -  Technical Mapping:
    - DB: `perks` table with `title`, `description`, `token_requirement`, `url`, `type`
    - API: `POST /creator/:id/perks`, `GET /perks/:creatorId`
    - Gating logic: frontend + backend token balance check
- 


## 5. Fundraising Round / Token Sale

- **Crucial for PoC**, but not detailed enough
  - Missing:
    - One-time sale or bonding curve?
    - Use case: Creators get SOL, fans get tokens
  -  Technical Mapping:
    - On-chain: basic escrow program, or Strata
    - DB: `sales` table with `creator_id`, `price`, `start`, `end`
    - API: `POST /launch-sale`, `POST /buy-token`
- 


## 6. Analytics Dashboard

- **Nice-to-have for MVP**, but **not required** in PoC
  -  Technical Mapping:
    - Use RPC calls to fetch token balances
    - Off-chain DB to track purchases + holders
    - API: `GET /dashboard/:creatorId`
- 

## 7. Reward/Perk Delivery Tools



- **Important for utility**, but you should **define how they're delivered**:
    - URL unlocked? Discord role? NFT airdrop?
  -  **Technical Mapping**:
    - Token gate check on frontend or API
    - API: `GET /redeem-perk/:perkId`
- 

## 8. **Community Engagement**


- **Too vague** in PoC
    - Maybe start with: creator posts, polls, or token-gated comments
  -  **Technical Mapping**:
    - DB: `posts` table linked to `creator_id`
    - API: `POST /creator/:id/post` , `GET /feed/:creatorId`
- 

## **Fan Flows**


### 1. **Browse & Discover Creators**

- Solid. Basic search + listing page.
  -  DB: `creator_profiles`
  -  API: `GET /creators`
- 

### 2. **Connect Wallet**


- Well-defined; no issues
  -  Solana Wallet Adapter
- 

### 3. **Buy Creator Token**

- Important, but needs precise mechanism:
  - From sale vault? P2P DEX?
-  **Technical Mapping**:

- Smart contract or backend logic
- UI to sign + send SOL, get tokens


#### 4. Token Perk Access

- Covered above
-  Frontend token check or backend gated endpoint



#### 5. Sell / Trade Tokens

- **Out of scope for PoC unless using DEX integration**
- Can be handled via link to external DEX like Jupiter/Raydium




#### 6. Track Token Holdings



-  Use `getTokenAccountsByOwner` RPC call
- Optional DB mirror to cache results

#### 7. Receive Updates from Creators

- Start simple: follow creators + view a feed
-  DB: `follows` , `posts`
-  API: `GET /feed` , `POST /follow/:creatorId`

### What's Missing or Unclear

Issue	Suggestion
 <b>Unclear token mechanics</b>	Define mint rules: fixed or dynamic supply, can fans resell? Any fees?
 <b>Verification Levels</b>	Add clear tiers: unverified, verified-basic (social), verified-KYC
 <b>Perk Delivery UX</b>	Define how perks are delivered — link, code, access token, etc.

Issue	Suggestion
 <b>Redemption Mechanics</b>	Can perks be redeemed once? Are they logged in DB?
 <b>Community tools</b>	Clarify what engagement actually means at MVP: feed, comments, votes?

## ✓ Recommendation

You're 80% there — your stories are strong and well-aligned with your value proposition. To go from story → **technical spec**, you just need:

- More granularity per feature (as above)
- Clear **data models**, **API surface**, and **smart contract responsibilities**
- Minimum set of success criteria per interaction (e.g. creator launches token, 5 fans buy, 1 perk redeemed)

Would you like me to help you turn these refined stories into:

- ✓ DB schema
- ✓ API endpoint list
- ✓ Smart contract architecture?

Just say the word.