Creator Token Architecture Design

Protocol POC Requirements

1. **Creator Token Minting**
    - The protocol must allow creators to mint their own unique SPL token.
    - During token creation, creators must submit proof of identity or legitimacy (e.g., link to a YouTube intro video or signed message).
    - Token creation should include setting metadata such as token name, symbol, and optional description fields.
2. **Protocol Ownership of Mint**
    - The protocol (via PDA) will retain authority over the token mint and associated vault accounts.
    - Neither the creator nor any external party will have direct control over minting or supply updates.
    - This design prevents creators from arbitrarily changing token economics post-launch.
3. **Bonding Curve-Based Pricing**
    - The protocol must control token price using a bonding curve mechanism.
    - Token price increases with demand (buys) and decreases with supply (sells), based on a defined curve function (e.g., linear, exponential).
    - The bonding curve must be deterministic and calculated entirely on-chain.
4. **Anti-Whale Protections**

- The protocol should enforce rate limits or per-wallet token caps to prevent creators or fans from bulk buying.
- These protections ensure a fair and gradual distribution of tokens among genuine fans.

5. **Token Purchase by Fans**
   - Fans must be able to query and fetch the current price of a creator's token via a read-only on-chain call.
   - Fans can purchase a limited number of tokens directly from the bonding curve vault using SOL.
   - Token purchases trigger minting from the bonding curve PDA-controlled mint to the user's ATA.
6. **Token Sale (Sell-Back) by Fans**
   - Fans can sell their tokens back to the protocol at the current bonding curve price.
   - Upon sale, tokens are burned and the corresponding amount of SOL is returned to the fan from the protocol vault.

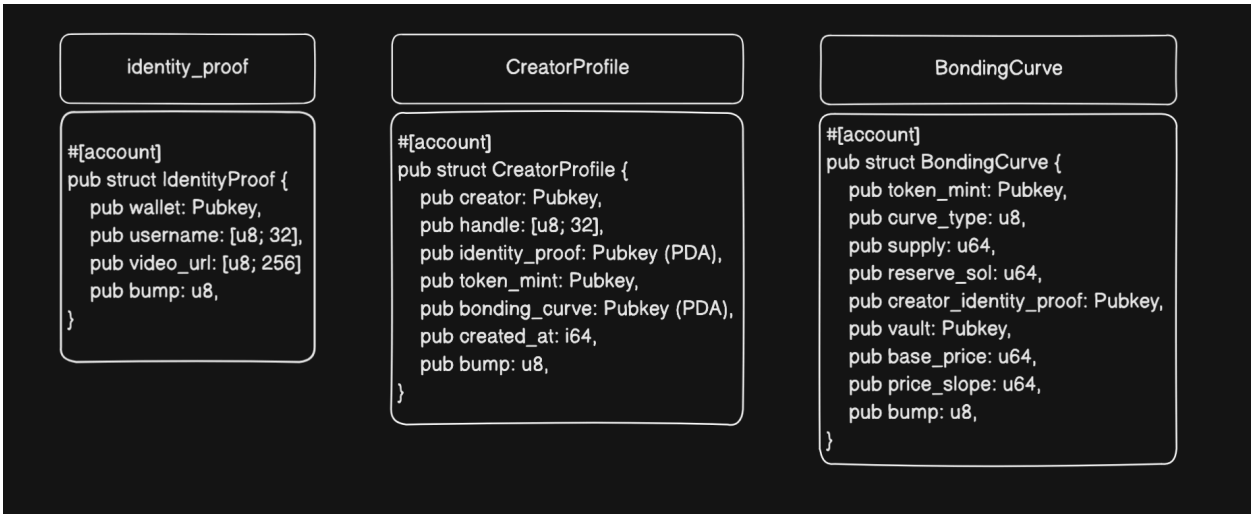
Accounts & their Uses

Vault Account : The `VaultAccount` is a **PDA (Program Derived Address)** that holds **SOL or SPL tokens** on behalf of the protocol and ensures secure custody and controlled flows of funds.

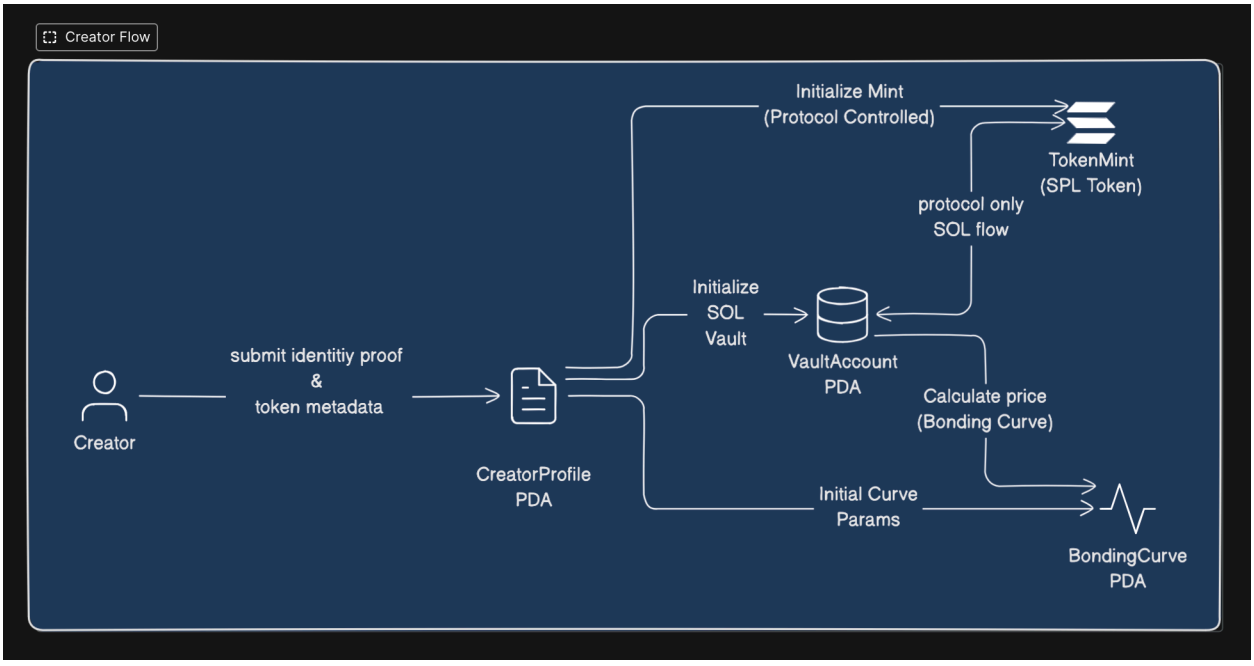
Creator Profile Account: The CreatorProfile account is a PDA that stores identity, metadata, and config for the creator and links them to their minted token.

Identity_proof : will be a PDA that is going to store will be storing the creator’s public key. Which they can then post on their youtube channel or some place as hard evidence that it was them who made the token.

The identity proof will store the creator's wallet address used to create the token, a chosen display name (or social handle) of the creator, a link to a video (YouTube, Twitter, etc.) where they verify their wallet. The fans can themselves verify whether the creator is real or not without requiring any third party to validate.



Creator Flow to Create a Token



1. Creator submits identity proof & token metadata

- **Action:** Creator provides identity (e.g., wallet signature, YouTube link) and details like token name, symbol, and initial supply preferences.
- **Purpose:** Prove legitimacy and define the token. Users should be able to validate through those validations that the creator is indeed real.

---

2. CreatorProfile PDA is initialized

- **Action:** A `CreatorProfile` PDA is created and stored on-chain.
- **Contains:** Creator wallet, social metadata, YouTube link, verification status.

---

3. VaultAccount PDA is initialized

- **Action:** A protocol-owned `VaultAccount` PDA is created.
- **Purpose:** This account will hold SOL received when fans buy tokens.
- **Security:** Only the protocol can access or move SOL from here.

---

4. TokenMint (SPL Token) is initialized

- **Action:** A new SPL Token mint is created by the protocol.
- **Ownership:** Controlled by the protocol — creators don't directly hold mint authority.
- **Link:** Tied to the CreatorProfile.

---

5. Initial Bonding Curve parameters are set

- **Action:** Creator chooses curve type, base price, multiplier.
- **Examples:** Linear curve (`price = base + multiplier * supply`).

---

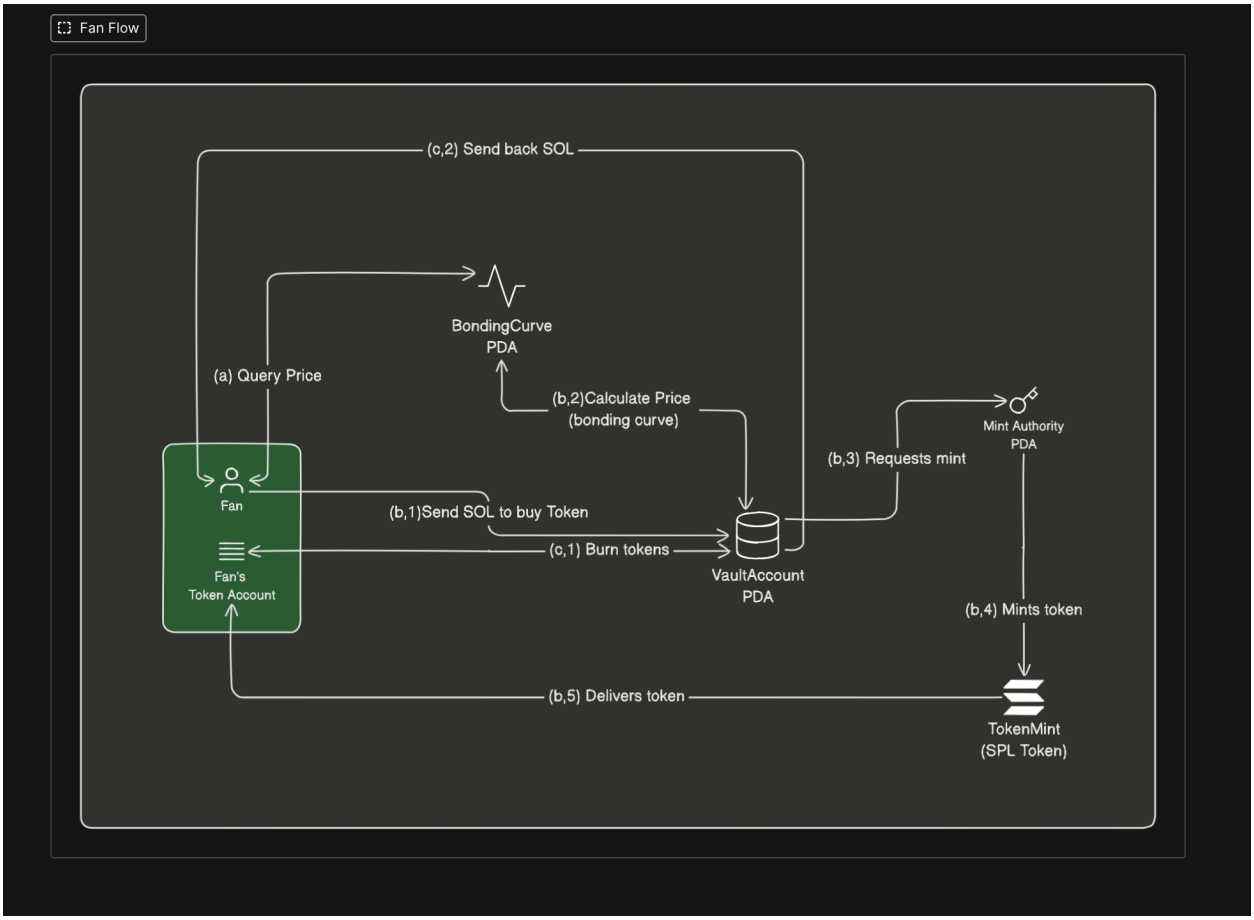
6. BondingCurve PDA is initialized

- **Action:** A `BondingCurve` PDA is created using initial parameters.
- **Purpose:** Stores curve config and `total_supply`, and handles price logic for buys/sells.

7. Protocol wires logic between components

- **Price Calculation:** When fans interact, token price is computed using the BondingCurve PDA.
- **Token Minting:** Token is minted from `TokenMint` when fan purchases.
- **Fund Flow:** Payment in SOL goes into `VaultAccount`.

Create Diagram for Fan flow



(a) Query Price Flow

a.1 — Fan queries token price

- **Action:** Fan initiates a query to view the current price of a creator's token.
- **Interaction:** Reads from the `BondingCurve PDA`.
- **Purpose:** Get the real-time price based on current supply and bonding curve logic.

(b) Buy Token Flow

b.1 — Fan sends SOL to buy token

- **Action:** Fan sends required SOL (price calculated from step `a.1`) to the protocol.
- **Destination:** SOL is sent to the `VaultAccount PDA`.

b.2 — Protocol recalculates price (bonding curve)

- **Action:** On-chain program reads `BondingCurve PDA` again to confirm current price based on updated supply.
- **Purpose:** Prevent price manipulation during transaction delay.

b.3 — Mint request is made

- **Action:** Protocol (via PDA) requests to mint new tokens from the SPL mint.
- **Authority:** Only `Mint Authority PDA` (controlled by protocol) can initiate mint.

b.4 — Tokens are minted

- **Action:** SPL tokens are minted into the protocol's controlled state.
- **Mint Source:** `TokenMint` PDA associated with the creator.

b.5 — Tokens delivered to fan

- **Action:** Newly minted tokens are transferred to the **fan's associated token account**.
- **Result:** Fan now owns the creator's token.

(c) Sell / Burn Token Flow

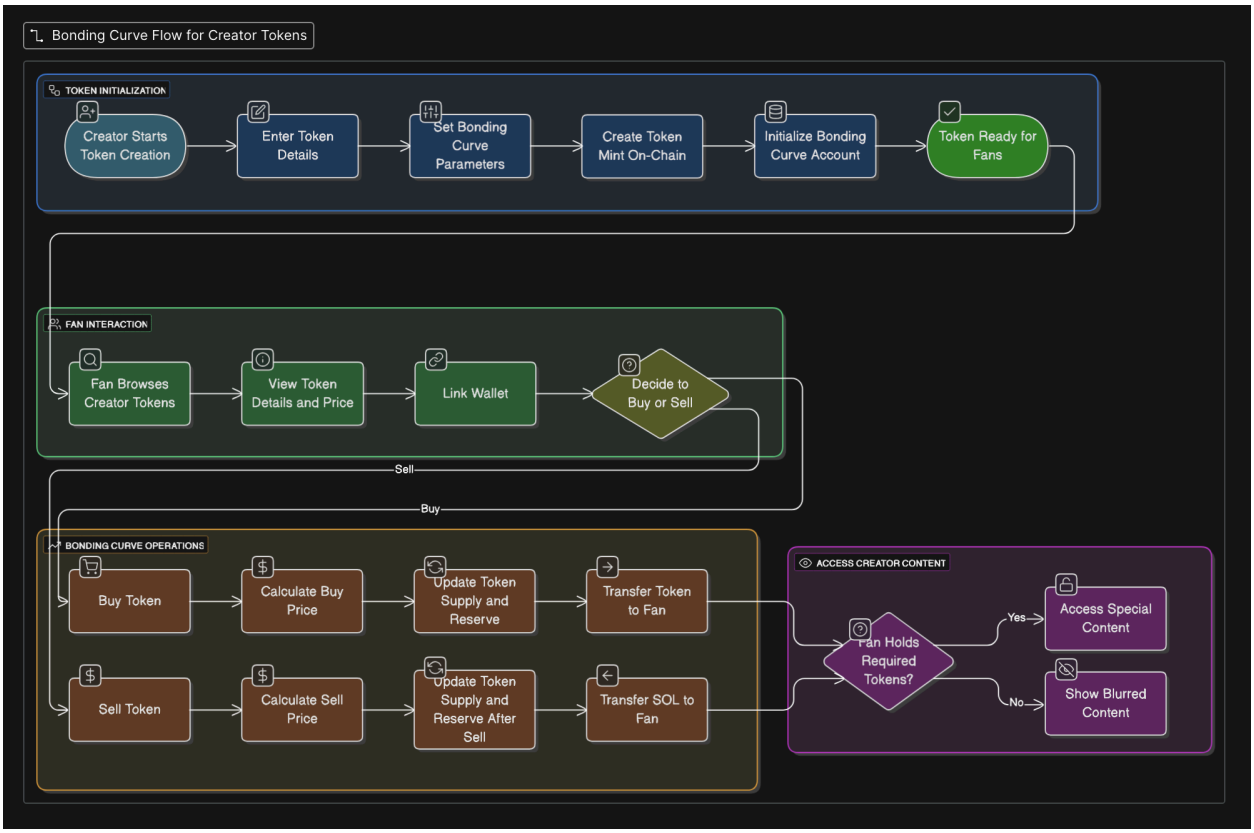
c.1 — Fan sends tokens to burn

- **Action:** Fan sends tokens back to protocol (via vault or burn handler).
- **Purpose:** Redeem current value in SOL by burning their tokens.

c.2 — Protocol sends back SOL

- **Action:** Using updated `BondingCurve PDA`, the protocol calculates the current redemption price.
- **Payment:** Corresponding SOL is sent back to the fan from `VaultAccount PDA`.

Bonding Curve Flow



Buy Token Flow

1. Buy Token

- Fan initiates a purchase action from the UI.
- On-chain program receives the intent to buy specific creator tokens.

2. Calculate Buy Price

- Based on the current supply, reserve, and bonding curve parameters, the program calculates the total cost.
- The formula can follow curves like:
  - Linear: `P = a * supply + b`
  - Exponential: `P = a * (supply)^2 + b`
- This price is retrieved from the **BondingCurve PDA**.

(NOTE : Formula is not yet finalized)

3. Update Token Supply and Reserve

- On successful payment:
  - `token_supply` increases by `X` (tokens bought),
  - `reserve_balance` increases by corresponding SOL amount.
- Both values are updated in the **BondingCurve PDA** account.

4. Transfer Token to Fan

- Tokens are minted (via `TokenMint` PDA) and transferred to the fan's token account.
- Mint authority is derived via PDA to remain protocol-controlled.

Sell Token Flow

5. Sell Token

- Fan initiates a token sale (burn) action from the UI.

6. Calculate Sell Price

- The program computes how much SOL the fan will receive for the tokens being burned.
- Uses the current bonding curve parameters from `BondingCurve PDA`.

7. Update Token Supply and Reserve After Sell

- `token_supply` is decreased by the number of tokens sold.
- `reserve_balance` is reduced by the SOL being refunded.
- Updates are saved back to the **BondingCurve PDA**.

8. Transfer SOL to Fan

- Equivalent SOL (based on bonding curve price) is sent back to the fan from the **VaultAccount PDA**.
- Tokens are burned to keep supply accurate.