Assignment 2

User Stories & On-Chain Requirements

Part A: User Stories & On-Chain Requirements Document:

1.A preliminary, comprehensive list of all potential user types.

Direct Users

- Casual Gamers
- Competitive/Esports Gamers
- Web3 and GameFi Enthusiasts
- Streamers/Influencers

Indirect Users / Beneficiaries

- Game Developers & Studios
- In-game Item Creators / Asset Designers
- Viewers / Fans

Administrators / Moderators

- Platform Developers
- Community Moderators

Stakeholders

- Token Holders / Investors
- Gaming Guilds & DAOs
- Brand Sponsors / Advertisers

2. Analyze the AI's recommendations and rationale. Decide on your final, prioritized list of the top 2-5 key user types. Document your final list and a brief explanation of your decision-making process, noting where you agreed or disagreed with the AI's suggestions and why.

Final Prioritized List of Key User Types

1. **Gamers (Casual and Competitive)**
2. **Web3 and GameFi Enthusiasts**
3. **Token Holders / Traders**

Decision-Making Process:

After going through the AI's suggestions, I agreed with most of them since they fit well with my project's goal of rewarding players and building a player-driven economy. I decided to combine casual and competitive gamers into one group "Gamers" because they both use the system in similar ways, just with different levels of playtime. I also added token holders and traders, as they're important for testing the trading and liquidity side of the project. This mix covers both the people who play and the ones who drive the token economy, making it a well-rounded focus for the Proof-of-Concept.

3. A list of key functions and interactions mapped to your prioritized user types.

1. Gamers (Casual and Competitive)

- Earn tokens based on playtime.
- Redeem tokens for in-game rewards or items.
- Track token balance and rewards in a dashboard.
- Trade or sell tokens on supported marketplaces.

2. Web3 & GameFi Enthusiasts

- Connect wallet to earn, store, and trade tokens.
- Explore games using the reward system.
- Stake or hold tokens to support the ecosystem.
- Engage in community discussions.

3. Token Holders / Traders

- Buy, sell, or trade tokens on exchanges or marketplaces.
- Provide liquidity or stake tokens for rewards.
- Track token performance and value.

4. An initial list of key technical requirements derived from the core user interactions.

1. Playtime Tracking and Token Reward System

Goal: Automatically reward players with tokens based on their playtime.

Technical Requirements:

- User Authentication System:
  To identify each player (e.g., using wallet-based login or player ID).
- Playtime Tracking Mechanism:
  A timer or activity tracker that records how long a player stays active in the game session.
- Smart Contract for Token Distribution:
  A blockchain contract that mints or transfers tokens to players according to tracked playtime.
- Backend Logic & API:
  Server or script to process session data and trigger the token reward transactions.
- Player Dashboard (Frontend):
  UI to display total playtime, tokens earned, and reward history.

2. **Token Trading / Marketplace Interaction**

Goal: Allow players or traders to trade earned tokens.

Technical Requirements:

- Blockchain Token (ERC-20/SPL Standard):
  The reward token must be deployed on a blockchain that supports trading .
- Basic Marketplace or Exchange Interface:
  Simple UI to list tokens, view price, and trade (buy/sell) between players.
- Wallet Integration:
  Integration with wallets (e.g., MetaMask, Phantom) to hold, send, and receive tokens.
- Transaction Handling System:
  Smart contract functions or APIs to manage token transfers and marketplace interactions.

- Database or Ledger (optional):
  To store transaction history and player token balances off-chain for quick retrieval.

5. Document the AI critique, your analysis, and the rationale for your refined user stories and technical requirements.

1. AI Critique Summary

The AI feedback identified the following:

- Strengths:
  - User stories align well with the value proposition: rewarding playtime, token trading, and creating a player-driven economy.
  - Technical requirements cover key components: blockchain smart contracts, wallet integration, backend logic, and dashboard.

- Weaknesses / Gaps:
  - Technical requirements were not granular enough for actual implementation (missing API endpoints, database schemas, smart contract functions).
  - Playtime tracking, redemption system, and fraud prevention mechanisms were not fully specified
  - Marketplace mechanics and tokenomics needed more clarity.

2. My Analysis

After reviewing the AI critique and the project goals:

- The core value loop (playtime → token reward → trading/redeeming) is correct and aligns with the value proposition.
- Granularity gaps would make it difficult to directly build a PoC without additional specification:
  - Need explicit data flows, backend APIs, and database schemas.
  - Smart contract logic must be defined for token minting, redemption, and marketplace trades.
  - Fraud prevention, anti-cheat measures, and minimal tokenomics rules are necessary to validate feasibility.

3. Refined User Stories

Gamers (Casual and Competitive)

- Earn tokens automatically based on playtime.
- View your token balance and reward history on a dashboard.
- Redeem tokens for in-game items or boosts.
- Trade earned tokens with other players.

Web3 & GameFi Enthusiasts

- Connect your wallet to store and trade tokens.
- Discover games that integrate with the system.
- Stake or hold tokens to participate in governance or ecosystem activities.

Token Holders / Traders

- Buy, sell, and trade tokens via the marketplace or external exchanges.
- Track token performance and liquidity.
- Stake tokens to earn rewards or participate in governance.

4. Refined Technical Requirements

Playtime Tracking and Token Reward System

- User Authentication: Wallet-based login with API endpoints.
- Playtime Tracking: Session timers, anti-fraud checks.
- Smart Contract: rewardPlayer(address, playtime) → tokens minted.
- Backend API: Processes session data, triggers token minting.
- Database: users, sessions, rewards tables.
- Dashboard: Displays playtime, token balance, reward history.

Token Trading / Marketplace

- Token Deployment: ERC-20/SPL standard token.
- Marketplace Interface: UI for buy/sell, smart contract for order execution.
- Wallet Integration: Connect wallets to trade and hold tokens.
- Transaction Handling: Smart contract or API to validate transfers.
- Database: transactions table to record trades.

6. A list of your user stories, each followed by a bulleted list of its potential on-chain requirements.

Gamers / Web3 & GameFi Enthusiasts

1. Sign in to the game using your wallet

- Use Solana wallet (e.g., Phantom) for authentication.
- Record player address on-chain.
- Initialize player account/profile if first login.

2. Play the game to earn tokens automatically based on playtime

- Solana smart contract (program) mints or transfers tokens based on validated playtime.
- Store cumulative playtime per player on-chain.
- Apply rate limits for token issuance.
- Emit an event for each token reward transaction.

3. Check your total tokens and playtime on your dashboard

- Solana token account balance query (getTokenAccountBalance).
- Maintain reward history on-chain (or off-chain via events).

4. Redeem your tokens for in-game items or boosts

- Function to burn or lock tokens upon redemption.
- Map redeemed tokens to specific in-game items or assets.
- Emit a redemption event.

5. Trade tokens with other players in the marketplace

- SPL token contract supports transfers between addresses.
- Marketplace Solana program to handle trades.
- Track trade transaction history.
- Ensure trades fail if balances are insufficient.

6. Stake or hold tokens for governance or ecosystem benefits

- Solana program to lock staked tokens.
- Track staked amount and duration per player.

- Enable voting or reward distribution based on staked tokens.

Token Holders / Traders

7. Buy tokens from the marketplace or other players

- SPL token transfer function for purchases.
- Marketplace program handles escrow and settlement
- Emit purchase confirmation event.

8. Sell or trade tokens to other users or platforms

- Transfer tokens via SPL token contract.
- Marketplace program manages trade and events.
- Optional: validate minimum balance or price.

9. Monitor token value and performance

- Record all on-chain transactions for wallet.
- Emit events for transfers/trades.
- Optional: integrate price oracle for token value tracking.

10. Stake tokens to earn rewards or participate in governance

- Lock tokens in Solana staking program.
- Calculate and distribute rewards periodically.
- Record governance participation on-chain.

Part B:

AI Prompt:
My project's value proposition is [summarize from Assignment 1]. Here is a brainstormed list of all potential user types: [List from Step 1]. 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.

AI Output:

1. Casual Gamers

They represent the largest segment of the gaming population and are ideal for testing the token reward mechanism. If casual players find value in earning and using tokens for their playtime, it validates the system's ability to drive engagement and retention.

2. Web3 and GameFi Enthusiasts

As early adopters familiar with blockchain and tokenized systems, they're more likely to test, give feedback, and help refine the technical aspects of the PoC. Their engagement helps validate the smart contract logic, tokenomics, and marketplace functionality.

3. Competitive/Esports Gamers

These players spend significant time in games and can stress-test the "time-based" reward logic. Their participation helps prove that the system can handle intensive play sessions and still remain fair, scalable, and motivating.

AI Prompt:
For a project with this value proposition [summarize] and focusing on these prioritized user types [list your final selected types from Step 2], help map out the key functions or interactions each user would need to perform.

AI Output:

1. Gamers (Casual and Competitive)

Key Functions / Interactions:

- Play games and automatically earn tokens based on their playtime.
- Track their earned tokens and rewards through a dashboard or in-game interface.
- Redeem tokens for in-game items, cosmetics, or upgrades.
- Trade or sell tokens to other players or on a marketplace.
- Participate in leaderboard systems or special events tied to token rewards.

2. Web3 and GameFi Enthusiasts

Key Functions / Interactions:

- Connect their crypto wallets to the platform to earn, store, and trade tokens.
- Explore game integrations and analyze token performance or staking options.
- Provide early feedback, test features, and engage with community discussions.
- Possibly stake tokens to support ecosystem growth or governance decisions.

3. Token Holders / Traders

Key Functions / Interactions:

- Buy, sell, or trade tokens earned by players on supported marketplaces.
- Provide liquidity to the token ecosystem through exchanges or staking pools.
- Monitor token price trends and participate in governance if applicable.
- Engage in speculation or long-term investment as the ecosystem grows.

AI Prompt:
Based on these top 1-2 critical user interactions [describe interactions], what are the key technical requirements needed to build a proof-of-concept?

AI Output:

1. Playtime Tracking and Token Reward System

Goal: Automatically reward players with tokens based on their playtime.

Technical Requirements:

- User Authentication System:
  To identify each player (e.g., using wallet-based login or player ID).

- Playtime Tracking Mechanism:
  A timer or activity tracker that records how long a player stays active in the game session.
- Smart Contract for Token Distribution:
  A blockchain contract that mints or transfers tokens to players according to tracked playtime.
- Backend Logic & API:
  Server or script to process session data and trigger the token reward transactions.
- Player Dashboard (Frontend):
  UI to display total playtime, tokens earned, and reward history.

2. Token Trading / Marketplace Interaction

Goal: Allow players or traders to trade earned tokens.

Technical Requirements:

- Blockchain Token (ERC-20/SPL Standard):
  The reward token must be deployed on a blockchain that supports trading (e.g., Ethereum, Solana, Aptos).
- Basic Marketplace or Exchange Interface:
  Simple UI to list tokens, view price, and trade (buy/sell) between players.
- Wallet Integration:
  Integration with wallets (e.g., MetaMask, Phantom) to hold, send, and receive tokens.
- Transaction Handling System:
  Smart contract functions or APIs to manage token transfers and marketplace interactions.
- Database or Ledger (optional):
  To store transaction history and player token balances off-chain for quick retrieval.

AI Prompt:

Review my core user functions/stories [Input from Part A, Step 3] and requirements [Input from Part A, Step 4]. Considering my project's refined value proposition [summarize from Assignment 1], 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?

AI Output:

✅ What’s Strong About Your Current User Stories

- Clear alignment with your value proposition:
  The stories (especially for gamers) directly match your core promise — rewarding playtime, token trading, and creating value through engagement.

- Logical user flow:
  You’ve mapped how tokens are *earned* → *tracked* → *traded*, which covers both the engagement and liquidity sides.

- Technical feasibility:
  You’ve identified blockchain (smart contracts, wallet integration), backend (tracking logic), and frontend (dashboard/UI) — meaning the structure is realistic for a Proof-of-Concept (PoC).

⚙️ Where It Falls Slightly Short (Granularity Gaps)

Your technical requirements are conceptually strong but not yet *implementation-level granular*.
To map these into an actual build plan (database schemas, APIs, blockchain programs), you’ll need:

1. Clear data flow definition —
   How does the *playtime tracker* communicate with the *smart contract*?
   - Example: POST /api/reward endpoint sends {player_id, playtime} → backend verifies → triggers mintTokens(player_id, amount) on-chain.
2. Database schema clarity —
   Define what’s stored off-chain. Example tables:
   - users (player_id, wallet_address, total_playtime, total_tokens)

    - sessions (session_id, start_time, end_time, tokens_earned)
    - transactions (tx_id, type [earn/trade/redeem], timestamp, value)

3. Blockchain program specifics —
   The *Smart Contract for Token Distribution* needs concrete logic, e.g.:
    - Function: rewardPlayer(address player, uint playtime)
    - Rule: 1 token = X minutes (configurable)
    - Event: TokenMinted(player, amount)
4. Marketplace mechanics —
   “Trade or sell tokens” needs clarity on *how*:
    - Peer-to-peer (direct wallet trades)?
    - Listing on an in-app marketplace?
    - Simple swap via a DEX-like interface?
5. Redemption system missing:
   Since part of your value prop is redeeming tokens for *in-game items or boosts*, there should be a minimal version for that:
    - Backend endpoint: /api/redeem
    - Smart contract or off-chain inventory that verifies item ownership.

What’s Missing or Unclear

Area	What’s Missing	Why It Matters

Game Integration Layer	How is playtime tracked across games? (SDK? API? Plugin?)	Needed to generalize beyond one demo game.
Security & Fraud Prevention	How to ensure players can't fake playtime or idle farm?	Vital for token fairness and sustainability.
Tokenomics Parameters	How many tokens per minute, supply cap, inflation control	Critical for balancing economy.
Admin/Developer Interface	Ability to monitor users, token distribution, and usage stats	Needed for debugging and governance in PoC.

Part C: Granularity & Clarity Refinement

- **De-jargon:** Removed technical terms like "API," "smart contract," or "on-chain" and replaced them with simple, non-technical language.
- **Atomicity:** Split combined actions into separate steps.
- **Clarity:** Each story clearly states what the user does and the outcome they get.
- **No Overlap:** Removed redundancy between "connect wallet" and "store tokens" by combining clarity while keeping steps atomic.

Before	**After**	**Rationale**
Earn tokens automatically based on playtime.	Play the game to earn tokens automatically based on how long you play.	Added context for clarity; outcome is obvious.
View token balance and reward history in a dashboard.	Check your total tokens and playtime on your dashboard.	Simplified language; made it easy for non-technical users.
Redeem tokens for in-game items or boosts.	Redeem your tokens to get in-game items, cosmetics, or boosts.	Added examples to clarify outcome.
Connect wallet to store and trade tokens.	Connect your digital wallet to store tokens safely.	Removed jargon; split “store” and “trade” into separate story.
Trade earned tokens with other players.	Trade tokens with other players in the marketplace.	Clarified where trading happens; improved context.