This chapter presents a comprehensive analysis of the requirements for the development of a Multi-chain Stablecoin Protocol based on the Collateralized Debt Position (CDP) mechanism. The chapter begins with a survey of the current Decentralized Finance (DeFi) landscape and analyzes existing solutions to identify limitations regarding cross-chain liquidity. Subsequently, it provides a functional overview of the proposed system through general and detailed use case diagrams. The core business processes, specifically the cross-chain state synchronization workflow, are illustrated to clarify the operation of the system. Finally, detailed functional descriptions of critical use cases and non-functional requirements regarding security, performance, and scalability are established to guide the subsequent design and implementation phases.

0.1 Status survey

The survey of the current technology landscape relies on three primary sources: (i) the needs of DeFi users seeking capital efficiency, (ii) existing single-chain stablecoin protocols, and (iii) current cross-chain infrastructure solutions.

Currently, the DeFi ecosystem exhibits severe fragmentation, particularly within lending protocols and CDP mechanisms. Under the prevailing model, users are restricted to establishing collateralized positions exclusively within a single blockchain network. Prominent platforms such as MakerDAO and Aave operate in isolated silos, preventing cross-chain collateral utility. Consequently, users face a complex and inefficient workflow when attempting to utilize capital across networks, often necessitating manual bridging procedures that incur high transaction costs and operational friction.

Analysis of Existing Systems

To identify the gap in the current market, this research analyzes two dominant categories of lending protocols.

Firstly, I want to talk about the isolated CDP protocols (e.g., MakerDAO). These represent the standard for decentralized stablecoins. They offer high security and proven economic models. However, they operate in strict isolation. A user with collateral on Ethereum cannot leverage this value to mint stablecoins on other networks (like Solana, Sui) and Layer 2 networks (like Arbitrum or Optimism) without physically bridging the underlying assets. This limitation forces users to choose between security (keeping assets on the one chain) and utility (using low-cost chains), resulting in significant capital inefficiency.

Secondly, "Cross-Chain Money Markets" (e.g., Radiant Capital) have effectively streamlined the user workflow, enabling seamless cross-chain borrowing without manual bridging steps. However, their architecture heavily relies on specific third-party interoperability layers, such as LayerZero, for message passing and asset transfers. This dependency introduces a critical single point of failure: the protocol's security is inextricably linked to the third-party bridge. Consequently, users are exposed to external systemic risks, and the protocol lacks sovereignty over its own verification logic.

Proposed Solution Analysis

The proposed system addresses the dependency risks of existing cross-chain markets by implementing a State Orchestration Architecture centered on Solana. Unlike protocols that outsource security entirely to general-purpose messaging layers, this solution maintains sovereignty over verification logic. The "Universal Wallet" mechanism ensures that the state is unified, while the validity of cross-chain requests is cryptographically verified on-chain (via Solana's Secp256k1 program) rather than relying solely on the trust assumptions of third-party bridges.

Table 1 highlights the strategic advantages of this approach, specifically regarding security sovereignty and state management.

Feature	**Isolated CDPs (e.g., MakerDAO)**	**Cross-Chain Markets (e.g., Radiant)**	**Proposed System (Universal USD)**
State Model	Isolated	Fragmented Pools	Unified Global State
Verification	Native On-chain Verification	Trusted Third-Party	On-chain Verification (Solana)
3rd Party Risk	None	High (Bridge dependency)	Minimized (Cryptographic proof required)
Capital Effic	Low (Trapped on one chain)	High	High

Bảng 1: Comparison of Lending Architectures

0.2 Functional Overview

The system is designed to function as a decentralized lending protocol that orchestrates state across the Solana blockchain and various EVM-compatible chains.

0.2.1 General use case diagram

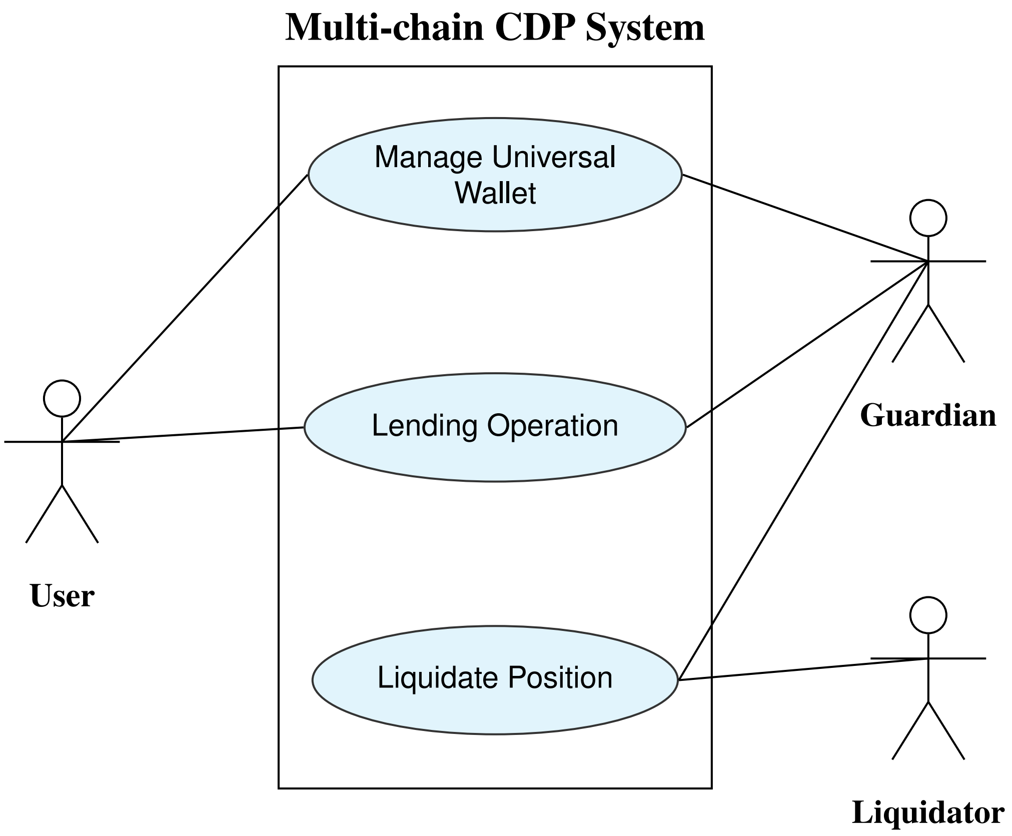
The system involves three primary actors:

(i) User: The borrower who holds collateral on EVM chains. They interact with the system by Lending Operations which include depositing assets, borrowing, repaying, and withdrawing, and manage their global debt position.

(ii) Guardian: A decentralized off-chain node responsible for listening to events on EVM chains, relaying signatures to Solana for verification, and synchronizing the execution results back to the EVM chains.

(iii) Liquidator: An actor who monitors the health factor of Universal Wallets on Solana. If a user's position becomes under-collateralized, the liquidator triggers the liquidation process.

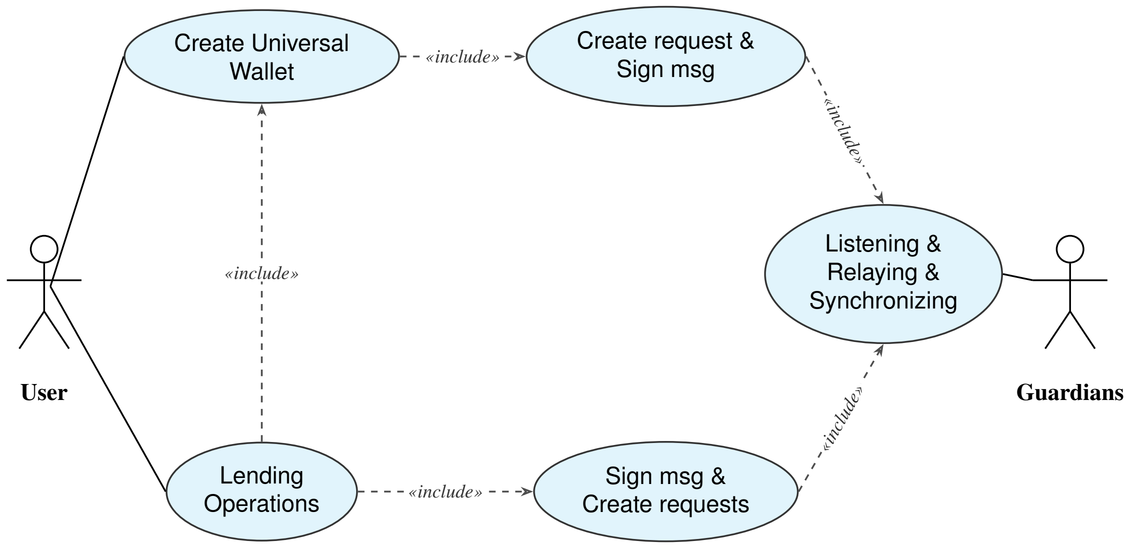
Figure 0.1 illustrates the general use cases.



Hình 0.1: General Use Case Diagram

0.2.2 Detailed use case diagram

To provide a general view of the system's operation, Figure 0.2 illustrates the decomposition of the core cross-chain workflows. This diagram emphasizes the dependency relationships between user actions and the underlying system processes, specifically highlighting the "Sign-then-Relay" mechanism. The diagram delineates several critical logical dependencies that govern the system's operation. Foremost among these is the dependency on the Universal Wallet; the core financial use cases Lending Operations containing "Deposit, Borrow, Repay, and Withdraw"



Hình 0.2: Detailed Use Case Diagram

maintain an *include* relationship with the "Create Universal Wallet" use case. This structure enforces a strict precondition, mandating that a user must establish a valid identity via a Universal Wallet PDA on Solana prior to executing any asset-related operations.

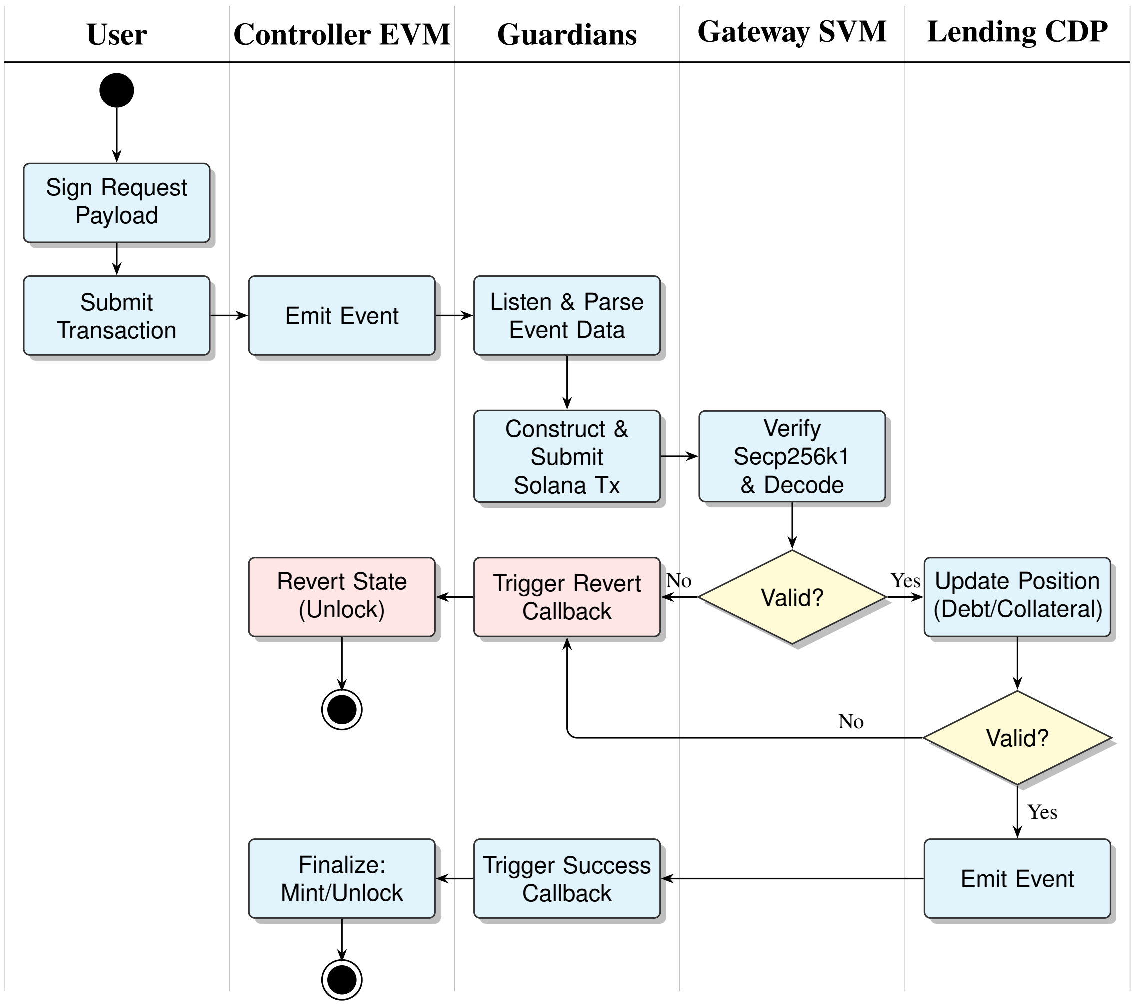
Furthermore, the security model is illustrated through the cryptographic authorization flow. Both the wallet creation and asset management workflows include the "Sign msg & Create requests" sub-use case. This relationship highlights that users do not interact directly with the Solana blockchain's state; instead, they generate and cryptographically sign messages within the EVM environment to authorize their intent. Consequently, this message creation process extends to include the "Listening & Relaying & Synchronizing" use case executed by the Guardian actor. This signifies that the lifecycle of a user's request is only finalized when the Guardian successfully intercepts the emitted event, relays the payload to Solana for verification, and synchronizes the execution result back to the source chain.

0.2.3 Business process

The system operates on a Cross-chain State Orchestration model, where the logic execution is decoupled from asset custody. The workflow involves five distinct entities: the User, the Controller EVM Contract, the Guardian Network, the Solana Gateway, and the Lending CDP Core.

As illustrated in Figure 0.3, the process follows a strict "Verify-then-Execute" lifecycle:

(i) Initiation (User in EVM): The process begins when the User cryptographically signs a specific request payload (containing the action type, amount, and



Hình 0.3: Business Process Activity Diagram

nonce) and submits the transaction to the Controller EVM Contract. The contract emits a event. Note that at this stage, no minting or unlocking occurs; the assets are simply locked or the request is queued.

(ii) Observation & Relay (Guardians): The Guardian network detects the event on the EVM chain. Instead of verifying the logic off-chain, the Guardian parses the event data and encapsulates the raw signature into a transaction destined for Solana.

(iii) Verification Layer (Solana Gateway): The Gateway Contract on Solana acts as the security checkpoint. It utilizes the native Secp256k1 program to verify that the signature matches the User's EVM address.

- If the signature is Invalid, the Gateway immediately signals a failure, bypassing the core logic.
- If the signature is Valid, the request is forwarded to the Lending CDP contract.

(iv) Logic Layer (Solana Lending CDP): The Lending CDP Contract attempts to update the user's position (e.g., increasing Debt). It performs critical checks such as Health Factor validation (e.g., Collateral Ratio $> 150\%$).

- If the logic holds (Logic OK), a event is emitted.
- If the logic fails (e.g., under-collateralized), the transaction is rejected.

(v) Synchronization & Finalization: Guardians listen for the outcome on Solana to trigger the corresponding callback on the EVM chain:

- If a event is captured, Guardians trigger the *Success Callback* on the EVM contract to finalize the action (e.g., Mint stablecoins).
- If the verification or logic failed, Guardians trigger the *Revert Callback* to unlock the user's assets and reset the nonce, ensuring funds are never stuck.

0.3 Functional Description

This section details the critical use cases of the system, covering the full lifecycle of a user's interaction from the initial identity creation to advanced position management.

0.3.1 Use Case: Create Universal Wallet

The creation of a Universal Wallet is the foundational procedure for establishing a user's cross-chain identity within the protocol. This process requires the coordination of four primary actors: the User, the off-chain Guardian, the EVM Contract, and the Solana Main Contract.

a, Pre-conditions

The execution of this use case is predicated on specific prerequisites. First, the User must possess a functional EVM wallet (e.g., MetaMask) with sufficient native tokens to cover the gas fees for the initial request transaction. Second, the User must hold a valid Solana wallet key pair, which is required to generate the cryptographic signature proving ownership of the destination address. Without these prerequisites, the binding process cannot commence.

b, Flow of Events

The workflow initiates on the EVM chain, where the User submits a transaction to the EVM Contract to formally request the creation of a wallet. This action triggers the contract to emit an event containing the user's EVM address, acting as a signal to the off-chain infrastructure. Simultaneously, to establish a proof of ownership for the destination, the User performs an off-chain action by signing a specific message containing their EVM address using their Solana private key. This signature is securely transmitted to the Guardian's API.

Subsequently, the Guardian captures the on-chain EVM event and cross-references it with the submitted off-chain Solana signature. Upon successful verification that the User controls both addresses, the Guardian constructs and submits a transaction to the Solana Main Contract. The process culminates when the Solana Main Contract allocates a new Program Derived Address (PDA) for the Universal Wallet, mapping the EVM identity to a unified storage state.

c, Post-conditions

Upon completion, a unique "Universal Wallet" structure is successfully initialized on the Solana blockchain. The User is thereafter authorized to perform cross-chain financial operations using this mapped identity.

0.3.2 Use Case: Deposit Collateral

This use case describes the mechanism by which users lock assets on an EVM chain to increase their collateral balance recorded on the Solana state.

a, Pre-conditions

The User must have previously initialized a Universal Wallet and must hold supported assets on the EVM chain sufficient for the deposit.

b, Flow of Events

The process begins with the User signing a payload and submitting the signature by calling the request Deposit function on the EVM Contract. Consequently, the assets are transferred to the contract's vault, and an event is emitted. The Guardian, monitoring the network, detects this event and reads the on-chain data. It then forwards the payload and the user's signature to the Solana Gateway.

On the Solana side, the Gateway contract verifies the signature and the integrity of the payload. Upon validation, the Solana Main Contract identifies the user's Universal Wallet and increments the collateral balance corresponding to the specific chain ID. Finally, the Guardian executes a synchronization step, updating the EVM contract to confirm that the deposit request has been synced, thereby allowing the user to proceed with subsequent actions.

c, Post-conditions

The specified collateral is securely locked in the EVM vault, and the collateral balance within the Universal Wallet on Solana is incremented to reflect this deposit.

0.3.3 Use Case: Mint Stablecoin

This process allows users to generate stablecoins against their collateral. This action mandates strict verification of the Health Factor on the Solana state to ensure system solvency.

a, Pre-conditions

The User must possess sufficient collateral such that the Health Factor remains above the minimum required ratio after the new debt is issued.

b, Flow of Events

The User initiates the process by signing a mint request payload and calling the request Mint function on the EVM Contract. Crucially, the contract immediately locks the User's mutex to prevent race conditions. The Guardian, upon listening to the emitted event, submits the signature and payload to Solana. The Main Contract verifies the signature and calculates the projected Health Factor.

If the Health Factor is valid, the Solana contract increases the User's debt balance. Observing the success event from Solana, the Guardian executes a callback transaction on the EVM Contract to physically mint the stablecoins to the User's wallet and release the mutex lock.

c, Alternative Flow

In the event that the Health Factor is insufficient, the Solana transaction will fail. Detecting this failure, the Guardian triggers a Revert function on the EVM Contract. This action ensures the User's mutex is unlocked without minting any tokens, restoring the system to its previous valid state.

d, Post-conditions

Stablecoins are minted to the User's address on the EVM chain, and the corresponding debt position is recorded in the Universal Wallet on Solana.

0.3.4 Use Case: Repay Debt

Users utilize this workflow to return stablecoins, thereby reducing their debt position and improving their Health Factor.

a, Pre-conditions

The User must hold a sufficient balance of the protocol's stablecoin on the EVM chain to cover the repayment amount.

b, Flow of Events

The User signs the payload and calls the request Repay function on the EVM Contract to submit the signature. The specified amount of stablecoins is locked immediately by the contract, and an event is emitted. The Guardian observes this event and relays the data and signature to Solana.

Upon receipt, the Solana Gateway verifies the signature, and the Main Contract processes the transaction by decreasing the User's debt balance in the Universal Wallet. To finalize the process, the Guardian confirms the state update back to the EVM chain, triggering the burning of the locked stablecoins and updating the user's nonce.

c, Post-conditions

The stablecoins are permanently burned on the EVM chain, and the debt balance recorded on Solana is decreased accordingly.

0.3.5 Use Case: Withdraw Collateral

This action allows users to retrieve their locked assets, provided that their remaining collateral is sufficient to support their outstanding debt.

a, Pre-conditions

The User must have sufficient free collateral, ensuring that the withdrawal does not cause the Health Factor to drop below the liquidation threshold.

b, Flow of Events

The User signs a withdraw request payload and submits it to the EVM Contract. The contract locks the User's mutex and, after verifying sufficient collateral exists in the EVM vault, emits an event. The Guardian forwards this request to Solana. The Main Contract performs a solvency check to determine if the remaining collateral covers the debt.

If the check passes, the Solana contract decreases the User's collateral balance. The Guardian then triggers the EVM Contract to transfer the requested assets from the vault back to the User's wallet.

c, Alternative Flow

If the withdrawal results in an insolvent position, the Solana transaction is rejected. Consequently, the Guardian executes a revert transaction on the EVM chain. This action keeps the assets locked in the vault and releases the mutex, preventing the user from withdrawing funds that secure active debt.

d, Post-conditions

The User receives the requested assets on the EVM chain, and the collateral balance within the Universal Wallet on Solana is reduced.

0.4 Non-functional Requirements

To ensure the Multi-chain CDP Protocol operates securely, efficiently, and reliably in a high-value financial environment, the system must adhere to the following strict non-functional requirements.

0.4.1 Security and Safety

Given the cross-chain nature of the protocol, security is the paramount requirement to prevent fund loss and bridge exploits. At the foundational level, the system mandates strict cryptographic compatibility. Specifically, the Solana Gateway must natively support and verify Secp256k1 signatures, which is the standard for EVM networks. This capability allows users to control their positions using existing Ethereum wallets without ever exposing their private keys to third parties. Furthermore, to safeguard against cross-chain replay attacks, the protocol implements a rigorous nonce management mechanism. Each transaction request must include a unique nonce tied to a specific Chain ID and User Address, allowing the Solana contract to deterministically reject any request with a nonce lower than or equal to the currently stored value.

In parallel with on-chain security, the off-chain infrastructure addresses the trust assumption of the Guardian network by decentralizing operations to prevent any single point of failure. The system requires a threshold of signatures, utilizing a Multisig or Threshold Signature Scheme from the Guardians before executing sensitive state changes, such as unlocking collateral. This distributed consensus effectively mitigates the risk associated with a single compromised Guardian node. Finally, the protocol ensures absolute financial safety through atomic revert capabilities. In the event of a failure during cross-chain synchronization, such as a Solana transaction reverting due to market slippage or an insufficient health factor, the system guarantees that the initial state on the EVM chain is successfully reverted. This automatic rollback releases the user's Mutex lock, preventing funds from being permanently frozen in a transitional state.

0.4.2 Performance and Efficiency

Regarding performance and efficiency, a primary objective is the minimization of end-to-end latency for cross-chain operations. The total duration for a generic user action, such as minting, is defined as the cumulative sum of the EVM confirmation time, the Guardian relay processing, the Solana finality period, and the final Guardian callback. To ensure a responsive and fluid user experience, the system is engineered to aim for a target execution time of under 30 seconds, assuming normal network conditions and excluding periods of extreme congestion on the Ethereum Mainnet.

Simultaneously, efficiency on the Solana Hub is governed by strict computational constraints. Given that the cryptographic verification of Secp256k1 signatures, necessary for authenticating EVM users, which is a computationally intensive operation, the Solana smart contracts must be rigorously optimized. This optimization is essential to ensure that instruction execution remains strictly within the block Compute Unit limit, thereby preventing transaction failures due to resource exhaustion and maintaining cost-effectiveness.

0.4.3 Reliability and Availability

To guarantee reliability and availability, the system is fundamentally designed around the principle of eventual consistency. This ensures that despite the asynchronous nature of distributed ledgers, the state maintained on the EVM Spokes will always eventually converge with the authoritative state on the Solana Hub. In scenarios involving network partitions or Guardian downtime, the system retains the capability to recover autonomously, processing any pending events once connectivity is restored. Furthermore, to handle the unreliability of network transmission, all Guardian operations are implemented with strict idempotency. This property ensures that submitting the same event multiple times, which is a common occurrence during network retries, does not result in the erroneous double-counting of debt or collateral. Finally, the infrastructure mandates a high availability standard of 99.9% for both the Guardian nodes and the Solana RPC endpoints. This rigorous uptime requirement is critical to prevent liquidation failures during periods of high market volatility, thereby preserving system solvency.

0.5 Conclusion

This chapter has provided a comprehensive analysis of the operational boundaries and functional necessities for the proposed Multi-chain Stablecoin Protocol. By critically evaluating the limitations of existing "Lock-and-Mint" bridges and single-chain CDP models, the study established the rationale for adopting a State Orchestration architecture, where Solana serves as the global state machine for liquidity scattered across EVM chains.

The functional analysis elucidated the complex workflows involving the Universal Wallet, emphasizing the pivotal role of the Guardian Network in maintaining cross-chain atomicity and data synchronization. Furthermore, the non-functional requirements have set strict constraints regarding cryptographic compatibility, system latency, and security against replay attacks. These specifications serve as the foundational blueprint for the architectural design and technical implementation strategies that will be presented in the subsequent chapters.