A smart wallet is a next-generation digital wallet that combines traditional payment functionality with advanced blockchain technology and programmable features. Unlike basic cryptocurrency wallets that simply store and transfer digital assets, smart wallets can execute complex operations through smart contracts, automate transactions, and interact directly with decentralized applications. They offer enhanced security through features like multi-signature authentication, social recovery options, and customizable spending limits. Smart wallets also enable users to manage multiple cryptocurrencies, tokens, and NFTs in one interface, while providing advanced features like account abstraction, gas fee optimization, and batch transactions. This technology is revolutionizing how users interact with Web3 platforms by simplifying complex blockchain operations and offering a more user-friendly experience comparable to traditional financial services.

P1: Identify functional and non-functional requirements from the system description.

- Validity: 1-0/18 = 1
  - Functional, Non-Functional
- Consistency: 1-6/18 = 0.666
  - 9 functional requirements: Multi-Cryptocurrency Management, Smart Contract Execution, Automated Transactions, Integration with Decentralized Applications (dApps), Enhanced Security Mechanisms (Multi-Signature Authentication, Social Recovery Options), Customizable Spending Limits, Account Abstraction, Gas Fee Optimization, Batch Transactions.
  - 9 non-functional requirements: Usability, Performance, Reliability, Scalability, Security, Interoperability, Maintainability, Availability, Compliance.
  - Performance, Reliability, Scalability, Maintainability, Availability, Compliance not explicitly mentioned.

P2: Create a use case model for the system.

- Validity: 1-0/5 = 1
  - System Boundary (0.5), Actors (1.5), Use Cases (2), Relationships (1)
- Consistency: 1-0/8 = 1
  - Manage Assets, Execute Smart Contracts, Automate Transactions, Interact with dApps, Enhanced Security Features, Social Recovery, Optimize Gas Fees, Batch Transactions.
  - All covered by functional requirements.

P3: Create use case specifications for identified use cases.

- Validity: 1-(1*0.3)/6.5 = 0.953
  - Use Case Name (0.5), Actor (1), Precondition (1), Postcondition (1), Main Scenario (2), Alternative Scenario (1)
  - Alternative scenario having no reference to basic flow.

- Consistency: 1-0/8 = 1
  - Manage Assets, Execute Smart Contracts, Automate Transactions, Interact with dApps, Enhanced Security Features, Social Recovery, Optimize Gas Fees, Batch Transactions.

P4: Create a domain model based on use case specifications.

- Validity: 1-(0.5)/5 = 0.9
  - Classes (2), Attributes (1), Relationships (1), Multiplicities (1)
  - Must not have Data Types (-0.5), Operations (-0.5), Navigabilities (-0.5), Visibilities (-0.5)
  - Data types included.
- Consistency: 1-0/11= 1
  - 10 classes: User, Wallet, Asset, SmartContract, Transaction, dApp, SecuritySettings, RecoveryContact, BlockchainNetwork, BatchTransaction.
  - 11 relationships with multiplicities: User-Wallet (1-1), User-Asset (1-m), User-SecuritySettings (1-1), User-RecoveryContact (1-m), Wallet-Transaction (1-m), Wallet-SmartContract (1-m), Wallet-dApp (1-m), SmartContract-Transaction (1-1), BlockchainNetwork-Transaction (1-m), BlockchainNetwork-Asset (1-m), BatchTransaction-Transaction (1-m)

P5: Identify system operations from use case specifications.

- Validity: 1-0.2/1.6 = 0.875
  - Base Use Case (0.2), Operation Name (1), Parameters (0.2), Return (0.2)
  - No return specified
- Consistency: 1-10/30 = 0.666
  - 30 system operations
  - updateAssetBalance not specified in “Execute Smart Contracts”.
  - suspendAutomatedTransaction, updateAutomationCriteria not specified in “Automate Transactions”
  - disconnectFromDApp, recordDAppInteraction not specified in “Interact with dApps”
  - authenticateMultiSignature not specified in “Enhanced Security”
  - analyzeGasFees, suggestTransactionDelay not specified in “Optimize Gas Fees”
  - verifyBatchTransactionStatus, recordBatchTransaction not specified in “Batch Transactions”

P6: Create design sequence diagrams for system operations.

- Validity: 1-0.2/4.6 = 0.956
  - Base System Operation (0.2), Participants (1), Operations (1), Parameters (0.2), Return Type (0.2), Sequence of Operation Calls (2)
  - No return type
- Consistency: 1-1/6 = 0.833
  - 6 DSDs: sendAsset, executeSmartContract, setAutomationRules, initiateSocialRecovery, optimizeGasFees, createBatchTransaction
  - Participants: User, WalletService, SecurityService, BlockchainNetwork, TransactionService, AutomationService, GasOptimiazationService
  - optimizeGasFees not specified in system operations.

P7: Create design class diagrams based on the domain model and sequence diagrams.

- Validity: 1-(0.1+0.2+0.5)/5.2 = 0.846
  - Classes(1), Attributes with Types(0.5), Operations (1), Parameters with Types(0.2), Return Type (0.2), Relationship(1), Multiplicities(0.5), Navigabilities(0.5), Visibilities(0.3)
  - No parameter type, no return type, no navigabilities
- Consistency: 1-0/14 = 1
  - 14 classes: User, Wallet, Asset, SmartContract, Transaction, BatchTransaction, dApp, SecuritySettings, RecoveryContact, BlockchainNetwork, GasOptimizationService, AutomationService, SecurityService, TransactionService

P8: Develop a Java implementation for the system as specified in the class diagram and sequence diagrams.

- Validity: 1-(0.2+0.2)/5 = 0.92
  - Classes (0.5), Data types (0.5), Visibilities (0.5), Constructor (0.1), Getters (0.2), Setters (0.2), Methods (3)
  - Missing getters, setters
- Consistency: 1-0/8 = 1
  - 8 classes: User, Wallet, Asset, Transaction, SecuritySettings, TransactionService, dApp, Main

P9: Develop tests including unit tests, integration tests, and system tests for the implementation.

- Validity: 1-0/3 = 1
  - Unit tests (1), integration tests (1), system tests (1)
- Consistency: 1-0/6 = 1
  - 3 Unit Tests: UserTest, WalletTest, TransactionTest
  - 2 Integration Tests: UserWalletIntegrationTest, SmartContractIntegrationTest
  - 1 System Tests: SmartWalletSystemTest