graph LR A[Student] --> B[Gateway] C[Sensei] --> B B --> D[Registry] B --> E[Booking] B --> F[Privacy] E --> G[Token] E --> H[NFT] F --> I[FHEVM] G --> J[LayerZero]

Simplified System Overview (Preview-Friendly)

graph TB subgraph "Users" Student[Student] Sensei[Sensei] end subgraph "Core System" Gateway[Gateway] Registry[Registry] Token[Token] Booking[Booking] NFT[NFT] Privacy[Privacy] end subgraph "External" FHEVM[FHEVM] LayerZero[LayerZero] end Student --> Gateway Sensei --> Gateway Gateway --> Registry Gateway --> Booking Gateway --> Privacy Registry --> Booking Booking --> Token Booking --> NFT Privacy --> FHEVM Token --> LayerZero

Simplified Knowledge Session Flow (Preview-Friendly)

sequenceDiagram participant Student participant Gateway participant Registry participant Booking participant Session participant Token participant NFT Student->>Gateway: bookSession Gateway->>Registry: verifySensei Registry-->>Gateway: senseiProfile Gateway->>Booking: bookSessionWithETH Booking->>Session: createSession Session-->>Booking: sessionCreated Sensei->>Gateway: acceptSession Gateway->>Session: acceptSession Sensei->>Gateway: startSession Gateway->>Session: startSession Sensei->>Gateway: completeSession Gateway->>Session: completeSession Session->>Token: completeKnowledgeSession Session->>NFT: createSessionNFT

Overall System Architecture (Complex Flow)

graph TB subgraph "Entry Points" Student["Student User - Blue Node"] Sensei["Sensei User - Purple Node"] end subgraph "Initial Processing" Auth["Authentication - Black Node"] Validation["Input Validation - Black Node"]

ProfileCheck["Profile Verification - Black Node"] end subgraph "Core Decision Point" Router["Request Router - Orange Node"] end subgraph "Top Processing Flow" TopProcessor1["Session Booking - Black Node"] TopProcessor2["Payment Processing - Black Node"] TopProcessor3["Token Minting - Black Node"] end subgraph "Central Processing Pipeline" CentralNode1["Knowledge Session - Black Node"] CentralNode2["AI Processing - Black Node"] CentralNode3["Privacy Manager - Black Node"] CentralNode4["FHEVM Integration - Black Node"] CentralNode5["Cross-Chain Prep - Black Node"] CentralNode6["Final Processing - Black Node"] end subgraph "Critical Processing Hub" GreenHub["Token Economy Hub - Green Node"] end subgraph "Bottom Processing Flow" BottomProcessor1["Session Management - Black Node"] BottomProcessor2["Quality Assessment - Black Node"] BottomProcessor3["NFT Creation - Black Node"] end subgraph "Output Processing" OutputNode1["Data Aggregation - Black Node"] OutputNode2["Cross-Chain Sync - Black Node"] OutputNode3["Final State Update - Black Node"] end subgraph "Final Destination" DarkBlue["External System - Dark Blue Node"] end subgraph "Secondary Processing" SecondaryNode1["Error Handling - Black Node"] SecondaryNode2["Logging - Black Node"] SecondaryNode3["Analytics - Black Node"] end %% Entry Point Connections Student --> Auth Sensei --> ProfileCheck %% Initial Processing Auth --> Validation ProfileCheck --> Router Validation --> Router %% Top Flow (Solid Lines) Router --> TopProcessor1 TopProcessor1 --> TopProcessor2 TopProcessor2 --> TopProcessor3 TopProcessor3 --> CentralNode1 %% Central Processing (Vertical Flow) CentralNode1 --> CentralNode2 CentralNode2 --> CentralNode3 CentralNode3 --> CentralNode4 CentralNode4 --> CentralNode5 CentralNode5 --> CentralNode6 %% Green Hub Connections (Multiple Outputs) CentralNode2 --> GreenHub GreenHub --> DarkBlue GreenHub --> DarkBlue GreenHub --> DarkBlue %% Bottom Flow (Solid Lines) Router --> BottomProcessor1 BottomProcessor1 --> BottomProcessor2 BottomProcessor2 --> BottomProcessor3 BottomProcessor3 --> CentralNode6 %% Dashed Connections (Secondary/Async) Router -.-> SecondaryNode1 TopProcessor2 -.-> SecondaryNode2 CentralNode3 -.-> SecondaryNode3 SecondaryNode1 -.-> OutputNode1 SecondaryNode2 -.-> OutputNode2 SecondaryNode3 -.-> OutputNode3 %% Output Processing CentralNode6 --> OutputNode1 OutputNode1 --> OutputNode2 OutputNode2 --> OutputNode3 OutputNode3 --> DarkBlue %% Cross-Connections TopProcessor3 -.-> CentralNode3 BottomProcessor2 -.-> CentralNode4 CentralNode5 -.-> OutputNode2 %% Final Connections OutputNode1

--> DarkBlue OutputNode3 --> DarkBlue

Smart Contract Interaction Matrix

graph LR subgraph "User Interface Layer" UI1[Web App] UI2[Mobile App] UI3[CLI Tool] end subgraph "Gateway Layer" Gateway1[SenseiGateway] Gateway2[API Gateway] Gateway3[Event Handler] end subgraph "Core Business Logic" Core1[SenseiRegistry] Core2[BookingSystem] Core3[KnowledgeSession] Core4[SenseiToken] end subgraph "Asset Management" Asset1[LessonNFT] Asset2[SensayAI] Asset3[PrivacyManager] end subgraph "Advanced Features" Adv1[SenseiCrossChain] Adv2[FHEVM Integration] Adv3[LayerZero Bridge] end subgraph "External Systems" Ext1[FHEVM Network] Ext2[LayerZero Protocol] Ext3[IPFS Storage] Ext4[Oracle Services] end %% Primary Connections (Solid) UI1 --> Gateway1 UI2 --> Gateway2 UI3 --> Gateway3 Gateway1 --> Core1 Gateway2 --> Core2 Gateway3 --> Core3 Core1 --> Core2 Core2 --> Core3 Core3 --> Core4 Core4 --> Asset1 Core3 --> Asset2 Core2 --> Asset3 %% Secondary Connections (Dashed) Gateway1 -.-> Adv1 Gateway2 -.-> Adv2 Gateway3 -.-> Adv3 Asset3 -.-> Ext1 Adv1 -.-> Ext2 Adv2 -.-> Ext1 %% Cross-Connections Core4 -.-> Asset1 Core3 -.-> Asset2 Core2 -.-> Asset3 %% External Integrations Ext1 -.-> Core4 Ext2 -.-> Adv1 Ext3 -.-> Asset1 Ext4 -.-> Core4

Knowledge Session Flow (Detailed)

sequenceDiagram participant Student participant Gateway participant Registry participant Booking participant Session participant Token participant NFT participant Privacy participant FHEVM participant CrossChain Note over Student,CrossChain: Session Booking Phase Student->>Gateway: bookSession(senseiId, details, payment) Gateway->>Registry: verifySensei(senseiId) Registry-->>Gateway: senseiProfile alt Payment Method Selection Gateway->>Booking: bookSessionWithETH{value: price} else Token Payment Gateway->>Booking: bookSessionWithToken(price, token) end Booking->>Session: createSession(sessionId, details) Session-->>Booking: sessionCreated Gateway->>Student: sessionBooked(sessionId) Note over Session: Session State: PENDING Note over Student,CrossChain: Session Acceptance Phase Sensei-

>>Gateway: acceptSession(sessionId) Gateway->>Session: acceptSession(sessionId) Session->>Session: state = ACCEPTED Note over Student,CrossChain: Session Execution Phase Sensei->>Gateway: startSession(sessionId) Gateway->>Session: startSession(sessionId) Session->>Session: state = IN_PROGRESS Note over Student,CrossChain: Session Completion Phase Sensei->>Gateway: completeSession(sessionId, quality) Gateway->>Session: completeSession(sessionId, quality) par Parallel Processing Session->>Token: completeKnowledgeSession(sessionId, sensei, student, price) Session->>NFT: createSessionNFT(sessionId, details, quality) Session->>Privacy: uploadEncryptedKnowledge(encryptedData) end Token->>Token: mint tokens to contract NFT->>NFT: create lesson NFT Privacy->>FHEVM: store encrypted knowledge Session->>Session: state = COMPLETED Note over Student,CrossChain: Post-Session Processing Privacy->>CrossChain: sync encrypted data CrossChain->>CrossChain: prepare cross-chain message Note over Student,CrossChain: Cross-Chain Propagation CrossChain->>CrossChain: _lzSend(message, targetChain) CrossChain-->>CrossChain: MessagingReceipt

Token Economy Flow (Complex)

flowchart TD subgraph "Input Sources" ETH[ETH Payments] Sessions[Session Completions] CrossChain[Cross-Chain Transfers] External[External Integrations] end subgraph "Processing Engine" MintRate[Current Mint Rate] Rebase[Rebase Logic] Demand[Demand Calculation] Backing[Backing Ratio] end subgraph "Value Distribution" SenseiEarnings[Sensei Earnings] PlatformFees[Platform Fees] StudentRewards[Student Rewards] Treasury[Treasury Pool] end subgraph "Economic Controls" MinRate[Minimum Rate: 100] MaxRate[Maximum Rate: 1000] MinBacking[Minimum Backing: 100%] RebaseInterval[Rebase Interval: 24h] end subgraph "Dynamic Adjustments" RateAdjust[Rate Adjustment] BackingAdjust[Backing Adjustment] SupplyAdjust[Supply Adjustment] QualityBonus[Quality Bonuses] end subgraph "Output Mechanisms" TokenMinting[Token Minting] TokenBurning[Token Burning] CrossChainSync[Cross-Chain Sync] ExternalAPIs[External APIs] end %% Primary Flow ETH --> MintRate Sessions --> Demand CrossChain --> Backing External --> MintRate Demand --> Rebase Rebase --> MintRate MintRate --> RateAdjust Backing --> BackingAdjust BackingAdjust --> SupplyAdjust RateAdjust --> TokenMinting SupplyAdjust -->

TokenBurning %% Distribution Flow Sessions --> SenseiEarnings Sessions --> PlatformFees Sessions --> StudentRewards Sessions --> Treasury %% Control Flow MinRate --> MintRate MaxRate --> MintRate MinBacking --> Backing RebaseInterval --> Rebase %% Quality Flow Sessions --> QualityBonus QualityBonus --> SenseiEarnings %% Output Flow TokenMinting --> ExternalAPIs TokenBurning --> ExternalAPIs CrossChainSync --> ExternalAPIs %% Feedback Loops TokenMinting -.-> Demand TokenBurning -.-> Backing ExternalAPIs -.-> Sessions

NFT Creation & Minting Flow (Detailed)

flowchart TD A[Session Completed] --> B[Quality Assessment] B --> C[NFT Metadata Creation] C --> D{isPublicMintable?} D -->|Yes| E[Public NFT] D -->|No| F[Private NFT] E --> G[Set Mint Price] F --> H[Student Only Access] G --> I[Calculate Price by Quality] I --> J[Store in LessonNFT Contract] J --> K{Student Wants to Mint?} K -->|Yes| L[Pay with SenseiTokens] K -->|No| M[NFT Remains Unminted] L --> N[Transfer Tokens to Contract] N --> O[Mint NFT to Student] O --> P[Update Metadata: isMinted = true] M --> Q[NFT Available for Public Mint] Q --> R[Anyone Can Mint with Tokens] subgraph "Quality-Based Pricing" S[Quality 1-3: Low Price] T[Quality 4-6: Medium Price] U[Quality 7-8: High Price] V[Quality 9-10: Premium Price] end B --> S B --> T B --> U B --> V

FHEVM Privacy System (Complex)

graph TB subgraph "Data Input Layer" RawData[Raw Knowledge Data] Metadata[Session Metadata] QualityScores[Quality Scores] UserPreferences[User Preferences] end subgraph "Encryption Layer" Encryptor[Data Encryptor] EncryptedData[Encrypted Data] PublicHash[Public Hash] EncryptionKey[Encryption Keys] end subgraph "FHEVM Processing Engine" TFHE[TFHE Library] EncryptedTypes[euint64, ebytes256, ebool] HomomorphicOps[Add, Div, Mul, Sub, Cmp] ZeroKnowledge[Zero-Knowledge Proofs] end subgraph "Privacy Manager Core" Upload[uploadEncryptedKnowledge] Verify[verifyEncryptedKnowledge] Process[processKnowledgeForAI] Cleanup[cleanupOldKnowledge] Analytics[Privacy-Preserving Analytics] end subgraph "Encrypted State

Management" TotalValue[encryptedTotalKnowledgeValue] AvgQuality[encryptedAverageKnowledgeQuality] Contributions[EncryptedKnowledgeContribution] QualityDistribution[Encrypted Quality Distribution] UserPatterns[Encrypted User Patterns] end subgraph "AI Integration" AITraining[Secure AI Training] ModelInference[Private Model Inference] KnowledgeAggregation[Knowledge Aggregation] QualityAssessment[Quality Assessment] end subgraph "Data Lifecycle" DataIngestion[Data Ingestion] DataProcessing[Data Processing] DataStorage[Encrypted Storage] DataRetrieval[Secure Retrieval] DataCleanup[Data Cleanup] end %% Data Flow RawData --> Encryptor Metadata --> Encryptor QualityScores --> Encryptor UserPreferences --> Encryptor Encryptor --> EncryptedData EncryptedData --> PublicHash Encryptor --> EncryptionKey EncryptedData --> TFHE TFHE --> EncryptedTypes EncryptedTypes --> HomomorphicOps HomomorphicOps --> ZeroKnowledge %% Privacy Manager Flow HomomorphicOps --> Upload Upload --> Verify Verify --> Process Process --> Analytics %% State Management Upload --> TotalValue Upload --> AvgQuality Upload --> Contributions Upload --> QualityDistribution Upload --> UserPatterns %% AI Integration Process --> AITraining AITraining --> ModelInference ModelInference --> KnowledgeAggregation KnowledgeAggregation --> QualityAssessment %% Data Lifecycle RawData --> DataIngestion DataIngestion --> DataProcessing DataProcessing --> DataStorage DataStorage --> DataRetrieval DataRetrieval --> DataCleanup %% Feedback Loops Analytics -.-> HomomorphicOps QualityAssessment -.-> HomomorphicOps DataCleanup -.-> EncryptedState

Cross-Chain Messaging (Detailed)

sequenceDiagram participant SourceChain participant SenseiCrossChain participant LayerZero participant TargetChain participant DestinationContract participant ExternalSystem Note over SourceChain,ExternalSystem: Message Preparation Phase SourceChain->>SenseiCrossChain: crossChainTransfer(recipient, amount, targetChain) SenseiCrossChain->>SenseiCrossChain: validateParameters(recipient, amount, targetChain) SenseiCrossChain->>SenseiCrossChain: prepareMessage(recipient, amount, targetChain) Note over SourceChain,ExternalSystem: Message Sending Phase SenseiCrossChain->>SenseiCrossChain: _lzSend(message, targetChain) SenseiCrossChain-

>>LayerZero: sendMessage(targetChain, message) LayerZero-->>SenseiCrossChain: MessagingReceipt Note over LayerZero: Message Propagation & Validation LayerZero->>LayerZero: validateMessage(message) LayerZero->>LayerZero: propagateMessage(targetChain) Note over SourceChain,ExternalSystem: Message Delivery Phase LayerZero->>TargetChain: deliverMessage(message) TargetChain->>DestinationContract: _lzReceive(message) Note over SourceChain,ExternalSystem: Message Processing Phase DestinationContract->>DestinationContract: validateMessage(message) DestinationContract->>DestinationContract: processCrossChainMessage() alt Token Transfer Operation DestinationContract->>DestinationContract: mintTokens(recipient, amount) DestinationContract->>DestinationContract: updateTokenSupply(amount) else Data Synchronization DestinationContract->>DestinationContract: updateCrossChainData(data) DestinationContract->>DestinationContract: syncStateChanges() else Contract State Update DestinationContract->>DestinationContract: updateContractState(newState) DestinationContract->>DestinationContract: emitStateUpdateEvent() end Note over SourceChain,ExternalSystem: Confirmation Phase DestinationContract-->>TargetChain: processingSuccess TargetChain-->>LayerZero: deliveryConfirmation LayerZero-->>SourceChain: finalConfirmation Note over SourceChain,ExternalSystem: Post-Processing SenseiCrossChain->>SenseiCrossChain: updateLocalState(success) SenseiCrossChain->>ExternalSystem: notifyExternalSystems(success) Note over SourceChain,ExternalSystem: Error Handling (if needed) alt Message Delivery Failed LayerZero-->>SourceChain: deliveryFailure SenseiCrossChain->>SenseiCrossChain: handleDeliveryFailure() SenseiCrossChain->>SourceChain: initiateRetry() end

Security Model (Comprehensive)

graph TB subgraph "Access Control Layer" Owner[Contract Owner] AuthorizedMinters[Authorized Minters] AuthorizedBurners[Authorized Burners] SenseiOnly[Sensei-Only Functions] StudentOnly[Student-Only Functions] AdminOnly[Admin-Only Functions] end subgraph "Security Mechanisms" ReentrancyGuard[Reentrancy Protection] Ownable[Ownership Control] Pausable[Emergency Pause] RateLimiting[Rate Limiting]

TimeoutMechanisms[Timeout Mechanisms] BlacklistSystem[Blacklist System] end subgraph "Input Validation" AddressValidation[Address Validation] AmountValidation[Amount Validation] StateValidation[State Validation] QualityValidation[Quality Validation] StringValidation[String Validation] ArrayValidation[Array Validation] end subgraph "Economic Security" BackingRatio[Backing Ratio Checks] MintRateLimits[Mint Rate Limits] SessionTimeouts[Session Timeouts] PaymentVerification[Payment Verification] OverflowProtection[Overflow Protection] UnderflowProtection[Underflow Protection] end subgraph "State Machine Security" StateTransitions[Valid State Transitions] StateValidation[State Validation] StateLocking[State Locking] StateRollback[State Rollback] end subgraph "Privacy & Encryption" DataEncryption[Data Encryption] KeyManagement[Key Management] AccessControl[Access Control] AuditLogging[Audit Logging] end subgraph "Cross-Chain Security" MessageValidation[Message Validation] SignatureVerification[Signature Verification] ReplayProtection[Replay Protection] ChainValidation[Chain Validation] end %% Access Control Flow Owner --> AuthorizedMinters Owner --> AuthorizedBurners Owner --> AdminOnly AuthorizedMinters --> ReentrancyGuard AuthorizedBurners --> ReentrancyGuard AdminOnly --> Ownable %% Security Mechanism Flow ReentrancyGuard --> InputValidation Ownable --> StateMachineSecurity Pausable --> EconomicSecurity %% Input Validation Flow AddressValidation --> StateValidation AmountValidation --> EconomicSecurity QualityValidation --> StateValidation %% Economic Security Flow BackingRatio --> StateMachineSecurity MintRateLimits --> EconomicSecurity PaymentVerification --> StateMachineSecurity %% State Machine Flow StateTransitions --> StateValidation StateValidation --> StateLocking StateLocking --> StateRollback %% Privacy Flow DataEncryption --> KeyManagement KeyManagement --> AccessControl AccessControl --> AuditLogging %% Cross-Chain Flow MessageValidation --> SignatureVerification SignatureVerification --> ReplayProtection ReplayProtection --> ChainValidation

Smart Contract Relationships

graph LR subgraph "Core Contracts" Registry[SenseiRegistry] Token[SenseiToken] Gateway[SenseiGateway] end subgraph "Session Management" Booking[BookingSystem] Session[KnowledgeSession] end subgraph "Digital

Assets" NFT[LessonNFT] AI[SensayAI] end subgraph "Advanced Features" Privacy[PrivacyManager] CrossChain[SenseiCrossChain] end Registry --> Booking Registry --> Session Registry --> AI Booking --> Token Session --> Token Session --> NFT Gateway --> Privacy Gateway --> CrossChain Privacy --> Token CrossChain --> Token AI --> Registry NFT --> Token

Token Economy System

graph TB subgraph "Token Supply Management" MintRate[Current Mint Rate] Rebase[Rebase Logic] Demand[Demand Calculation] end subgraph "Value Sources" ETHBacking[ETH Backing] KnowledgeValue[Knowledge Value] SessionPayments[Session Payments] end subgraph "Distribution" SenseiEarnings[Sensei Earnings] PlatformFees[Platform Fees] StudentRewards[Student Rewards] end subgraph "Economic Controls" MinRate[Minimum Rate: 100] MaxRate[Maximum Rate: 1000] BackingRatio[Minimum Backing: 100%] end ETHBacking --> MintRate KnowledgeValue --> Demand Demand --> Rebase Rebase --> MintRate SessionPayments --> KnowledgeValue SessionPayments --> ETHBacking MintRate --> Distribution KnowledgeValue --> Distribution Distribution --> SenseiEarnings Distribution --> PlatformFees Distribution --> StudentRewards MinRate --> MintRate MaxRate --> MintRate BackingRatio --> Rebase

FHEVM Privacy System

graph TB subgraph "Data Encryption" RawData[Raw Knowledge Data] EncryptedData[Encrypted Data] PublicHash[Public Hash] end subgraph "FHEVM Processing" TFHE[TFHE Library] EncryptedTypes[euint64, ebytes256] HomomorphicOps[Add, Div, Mul] end subgraph "Privacy Manager" Upload[uploadEncryptedKnowledge] Verify[verifyEncryptedKnowledge] Process[processKnowledgeForAI] Cleanup[cleanupOldKnowledge] end subgraph "Encrypted State" TotalValue[encryptedTotalKnowledgeValue] AvgQuality[encryptedAverageKnowledgeQuality] Contributions[EncryptedKnowledgeContribution] end RawData --> EncryptedData EncryptedData --> PublicHash EncryptedData --> TFHE TFHE --> EncryptedTypes

EncryptedTypes --> HomomorphicOps HomomorphicOps --> Upload Upload --> Verify Verify --> Process Process --> Cleanup Upload --> TotalValue Upload --> AvgQuality Upload --> Contributions TotalValue --> HomomorphicOps AvgQuality --> HomomorphicOps

Cross-Chain Messaging

sequenceDiagram participant SourceChain participant SenseiCrossChain participant LayerZero participant TargetChain participant DestinationContract SourceChain->>SenseiCrossChain: crossChainTransfer(recipient, amount, targetChain) SenseiCrossChain->>SenseiCrossChain: _lzSend(message, targetChain) SenseiCrossChain->>LayerZero: sendMessage(targetChain, message) LayerZero-->>SenseiCrossChain: MessagingReceipt Note over LayerZero: Message Propagation LayerZero->>TargetChain: deliverMessage(message) TargetChain->>DestinationContract: _lzReceive(message) DestinationContract->>DestinationContract: processCrossChainMessage() alt Token Transfer DestinationContract->>DestinationContract: mintTokens(recipient, amount) else Data Sync DestinationContract->>DestinationContract: updateCrossChainData(data) end DestinationContract-->>TargetChain: success TargetChain-->>LayerZero: confirmation LayerZero-->>SourceChain: delivery confirmation

Security Model

graph TB subgraph "Access Control" Owner[Contract Owner] AuthorizedMinters[Authorized Minters] AuthorizedBurners[Authorized Burners] SenseiOnly[Sensei-Only Functions] StudentOnly[Student-Only Functions] end subgraph "Security Features" ReentrancyGuard[Reentrancy Protection] Ownable[Ownership Control] Pausable[Emergency Pause] RateLimiting[Rate Limiting] end subgraph "Input Validation" AddressValidation[Address Validation] AmountValidation[Amount Validation] StateValidation[State Validation] QualityValidation[Quality Validation] end subgraph "Economic Security" BackingRatio[Backing Ratio Checks] MintRateLimits[Mint Rate Limits] SessionTimeouts[Session Timeouts] PaymentVerification[Payment Verification] end

Owner --> AuthorizedMinters Owner --> AuthorizedBurners AuthorizedMinters --> ReentrancyGuard AuthorizedBurners --> ReentrancyGuard SenseiOnly --> StateValidation StudentOnly --> StateValidation AddressValidation --> InputValidation AmountValidation --> InputValidation BackingRatio --> EconomicSecurity MintRateLimits --> EconomicSecurity SessionTimeouts --> EconomicSecurity

Data Flow Architecture (Complex)

flowchart LR subgraph "User Interface Layer" UI1[Web Interface] UI2[Mobile App] UI3[API Client] UI4[CLI Tool] end subgraph "Authentication Layer" Auth1[Wallet Connection] Auth2[Signature Verification] Auth3[Session Management] Auth4[Role Assignment] end subgraph "Gateway Processing" Gateway1[Request Validation] Gateway2[Input Sanitization] Gateway3[Rate Limiting] Gateway4[Request Routing] end subgraph "Business Logic Layer" Logic1[Session Management] Logic2[Token Operations] Logic3[NFT Management] Logic4[AI Processing] Logic5[Privacy Management] Logic6[Cross-Chain Logic] end subgraph "Data Storage Layer" Storage1[Blockchain State] Storage2[IPFS Metadata] Storage3[Encrypted Knowledge] Storage4[User Profiles] Storage5[Session History] end subgraph "External Integrations" Ext1[FHEVM Network] Ext2[LayerZero Protocol] Ext3[Oracle Services] Ext4[Analytics Services] Ext5[Notification Services] end subgraph "Output Layer" Output1[Event Emission] Output2[State Updates] Output3[Cross-Chain Messages] Output4[External API Calls] Output5[User Notifications] end %% Primary Data Flow UI1 --> Auth1 UI2 --> Auth2 UI3 --> Auth3 UI4 --> Auth4 Auth1 --> Gateway1 Auth2 --> Gateway2 Auth3 --> Gateway3 Auth4 --> Gateway4 Gateway1 --> Logic1 Gateway2 --> Logic2 Gateway3 --> Logic3 Gateway4 --> Logic4 Logic1 --> Storage1 Logic2 --> Storage2 Logic3 --> Storage3 Logic4 --> Storage4 Logic5 --> Storage5 %% Secondary Data Flow Logic1 -.-> Ext1 Logic2 -.-> Ext2 Logic3 -.-> Ext3 Logic4 -.-> Ext4 Logic5 -.-> Ext5 %% Output Flow Storage1 --> Output1 Storage2 --> Output2 Storage3 --> Output3 Storage4 --> Output4 Storage5 --> Output5 %% Cross-Connections Logic1 -.-> Logic2 Logic2 -.-> Logic3 Logic3 -.-> Logic4 Logic4 -.-> Logic5 Logic5 -.-> Logic6 %% External Feedback Ext1 -.-> Logic1 Ext2 -.-> Logic2 Ext3 -.-> Logic3 Ext4 -.-> Logic4 Ext5 -.-> Logic5

Economic Incentives (Detailed)

graph TB subgraph "Sensei Incentives" SessionPayments[Session Payments] QualityBonuses[Quality Bonuses] Reputation[Reputation Building] TokenRewards[Token Rewards] NetworkEffects[Network Effects] CrossChainRewards[Cross-Chain Rewards] end subgraph "Student Incentives" KnowledgeAccess[Knowledge Access] NFTOwnership[NFT Ownership] TokenEarnings[Token Earnings] QualityAssurance[Quality Assurance] LearningProgress[Learning Progress] CommunityAccess[Community Access] end subgraph "Platform Incentives" PlatformFees[Platform Fees] NetworkGrowth[Network Growth] DataValue[Data Value] CrossChainFees[Cross-Chain Fees] EcosystemExpansion[Ecosystem Expansion] InnovationRewards[Innovation Rewards] end subgraph "Economic Mechanisms" DynamicMinting[Dynamic Minting Rate] RebaseLogic[Rebase Logic] BackingRatio[Backing Ratio] DemandResponse[Demand Response] QualityMultipliers[Quality Multipliers] VolumeDiscounts[Volume Discounts] end subgraph "Token Utility" PaymentMethod[Payment Method] GovernanceRights[Governance Rights] StakingRewards[Staking Rewards] LiquidityProvision[Liquidity Provision] CrossChainBridge[Cross-Chain Bridge] DefiIntegration[DeFi Integration] end subgraph "Network Effects" UserGrowth[User Growth] ContentQuality[Content Quality] NetworkLiquidity[Network Liquidity] CrossChainAdoption[Cross-Chain Adoption] EcosystemPartnerships[Ecosystem Partnerships] InnovationAttraction[Innovation Attraction] end %% Primary Incentive Flow SessionPayments --> TokenRewards QualityBonuses --> TokenRewards Reputation --> SessionPayments NetworkEffects --> SessionPayments KnowledgeAccess --> NFTOwnership NFTOwnership --> TokenEarnings QualityAssurance --> KnowledgeAccess LearningProgress --> CommunityAccess PlatformFees --> NetworkGrowth NetworkGrowth --> DataValue DataValue --> CrossChainFees EcosystemExpansion --> InnovationRewards %% Economic Mechanism Flow DynamicMinting --> DemandResponse RebaseLogic --> BackingRatio BackingRatio --> DemandResponse DemandResponse --> DynamicMinting QualityMultipliers --> TokenRewards VolumeDiscounts --> TokenEarnings %% Token Utility Flow TokenRewards --> PaymentMethod TokenRewards --> GovernanceRights TokenRewards --> StakingRewards TokenRewards --> LiquidityProvision TokenRewards --> CrossChainBridge

TokenRewards --> DefiIntegration %% Network Effects Flow UserGrowth --> ContentQuality ContentQuality --> NetworkLiquidity NetworkLiquidity --> CrossChainAdoption CrossChainAdoption --> EcosystemPartnerships EcosystemPartnerships --> InnovationAttraction InnovationAttraction --> UserGrowth %% Feedback Loops TokenRewards -.-> SessionPayments TokenEarnings -.-> KnowledgeAccess InnovationRewards -.-> EcosystemExpansion NetworkLiquidity -.-> TokenRewards