

> Ziel: Hybrid Perps DEX auf Base, mit on-chain Matching (Continuous + Batch Auction), Cross-Margin-Engine, Vault/Insurance-Backstop und upgradableem Stack. V1 bleibt als Legacy-Pfad (Sepolia), V2 ist Standard.

## 1. MOTIVATION & ZIELE

- MEV-minimiertes, faireres Matching durch Hybrid-Modell (Continuous + periodische Auctions).
- Strenge Risiko-Kontrollen: Cross-Margin, OI-/Exposure-Caps, Funding, Partial Liquidations.
- Kapital-Effizienz und Resilienz: LP-Vault + Insurance Fund als Residual Counterparty und Backstop.
- Sicherheit & Upgradeability: UUPS + Timelock (später), modulare Oracles (Router), audits-once patterns.
- Deployment-Fokus: Base (8453) als primäre Chain; V1 bleibt auf Sepolia als Untermenü.

## 2. SYSTEMÜBERSICHT

- **\*\*OrderbookV2 (UUPS)\*\***: Tick-basierte Price-Time-Priority, Continuous Matching, Batch Auction mit Clearing-Preis, optionale Vault-Residual-Fills, Stop-Trigger via Keeper.
- **\*\*PerpEngineV2 (UUPS)\*\***: Cross-Margin, Funding, Liquidation (partial), OI-/Exposure-Caps, Maker/Taker-Gebühren-Split (Vault/Insurance/Treasury), ADL-Flag.
- **\*\*OracleRouter (UUPS)\*\***: Multi-Source Aggregation, MaxAge/MaxDeviation, optional TWAP.
- **\*\*Vault (UUPS, ERC20 Shares)\*\***: USDC (6d) Ein-/Auszahlung, optional Engine-Allocation.
- **\*\*InsuranceFund (UUPS)\*\***: USDC Treasury, optional Engine-Allocation, Backstop für Defizite.
- **\*\*Keepers\*\***: Auction-Executor, Stop-Trigger, Funding-Updater, Liquidation, Oracle-Updater.
- **\*\*Indexer/WS/API\*\***: Dual-Stack V1/V2 Endpoints; V2 als Standarddatenquelle.

## 3. TOKEN & COLLATERAL

- Collateral: USDC (6 decimals). Intern 1e18-Skalierung (SCALE = 1e12).
- Keine Governance-Token im MVP; Fees fließen anteilig an Vault/Insurance/Treasury (config).

## 4. MATCHING & AUKTIONEN

### CONTINUOUS

- Best-Bid/Best-Ask Tick-Listen, FIFO in Level.
- Taker swept gegen beste Gegenseite; Limit-Checks, Stop-Trigger (Keeper) möglich.
- Gebühren: Maker/Taker Bps (konfigurierbar, gedeckelt BPS=10\_000).

### BATCH AUCTION

- Eligible Orders (Queued) ? Clearing-Preis = Preis mit max. matched Volumen; Tiebreak: Nähe zum Oracle-Preis.

- Residual-Imbalance optional vom Vault gefüllt (wenn aktiviert).
- Cooldown 'auctionInterval', Limit 'maxAuctionOrders' pro Run.

#### 5. RISK ENGINE (PERPENGINEV2)

- Cross-Margin, 1 Collateral (USDC), Position pro Market.
- Limits: 'initialMarginBps', 'maintenanceMarginBps', 'maxLeverage', 'maxOpenInterest', 'maxAccountExposure'.
- Funding: 'fundingRatePerSecond', 'cumulativeFundingRate'; Keeper setzt Rate (Imbalance-basiert).
- Liquidation: Partial close, Penalty (feeBps) an Liquidator; Defizit ? Insurance ? BadDebt/ADL-Flag.
- Fees: Maker/Taker-Anteil ? Vault/Insurance/Treasury (konfigurierbare Shares, Summe = 100%).

#### 6. ORACLE-DESIGN

- Router aggregiert mehrere Quellen (Pull-Modell); MaxAge/Deviation je Market; TWAP optional.
- Keeper ruft 'updatePrice' periodisch auf Basis externer Feeds.
- Engine validiert Freshness via 'maxPriceAge'.

#### 7. VAULT & INSURANCE

- Vault: ERC20 Shares, 'totalAssets' = Wallet + Engine-Allocation (scaled). Share-Mint/Burn ~ ERC4626-ähnlich, aber minimal gehalten.
- Insurance Fund: Custodies Fees/Top-ups, kann in Engine allokieren.
- Residual-Auction-Fill: Vault kann gegensätzliche Seite bei Clearing füllen (optional, konservativ kapseln).

#### 8. UPGRADEABILITY & GOVERNANCE

- UUPS + Ownable (MVP); Timelock/Gnosis Safe empfohlen vor Mainnet.
- Upgrade-Scope: Router/Orderbook/Engine/Vault/Insurance separat.
- Kill/Pause Hooks: Pausable in kritischen Contracts (Owner).

#### 9. SICHERHEITSPRINZIPIEN

- ReentrancyGuard, Input-Validierungen (tickSize, min/max size, caps).
- Price Staleness/Deviation Checks über OracleRouter + Engine.
- Fee-Caps (BPS), Exposure-/OI-Caps, Maintenance/Initial Margin Checks.
- BadDebt surfacing + ADL-Flag (Implementierung ausstehend).
- Tests: Unit + Invariants (Margin Safety, Fee Conservation, OI Caps, No Stuck Orders), Fuzz auf Matching/Funding/Liq.

#### 10. BACKEND/INDEXER/KEEPERS (ZIELBILD)

- DB-Schema: V2-Events (OrderPlaced/Matched/AuctionExecuted, FundingRateUpdated, LiquidationExecuted, Fees/Vault).
- Indexer: Base RPC, Restart-Safety, Batch-Logs, Dual-Stack V1/V2.
- Keepers:
  - Auction Executor (batch),
  - Stop Trigger (preisbasierte Aktivierung),
  - Funding Updater (Imbalance-Berechnung + setFundingRate),
  - Liquidation,
  - Oracle Updater (Router sources).
- API/WS: '/v2/\*' als Standard; '/v1/\*' für Legacy. Live-Streams aus V2-Events.

#### 11. FRONTEND (ZIELBILD)

- Netzwerk: Base (8453) für V2; Legacy-Tab für V1 (Sepolia).
- Order Entry: Market/Limit/Stop, Continuous/Auction Toggle, Slippage, Fees (Maker/Taker) sichtbar, Operator-Approvals falls nötig.
- Orderbook/Trades: Live WS, Spread/Tick-Bands, Auction-Countdown/Result.
- Positions: Margin/Funding/PnL aus V2, Liq-Preis auf Basis Contract-Params, Health-Bar.
- Vault UI: echte USDC-Ein/Auszahlung, Shares, Fee-Yield, ggf. Claim.
- Settings: Switch V1/V2, Netzwerk-Hinweise, Sprache (DE/EN).

#### 12. DEPLOYMENT & OPS (BASE)

- Hardhat: Base-Netzwerk, Proxy-Deploy (UUPS) für Router/Orderbook/Engine/Vault/Insurance.
- Konfig: Market-Setup (tickSize, margins, caps), Fee-Shares, Auction-Interval, Router-Sources.
- ENV: V2-spezifische Adressen, Keeper Keys, RPC, CORS.
- Rollout: Deploy ? Wire Router ? Seed Price ? Create Market ? Set Fees ? Start Keepers ? Smoke Tests.

#### 13. OFFENE ARBEITSSCHRITTE (UMSETZUNG)

- PerpEngineV2: ADL/Defizit-Flow finalisieren, Funding-Calc (Imbalance), Events, Operator/Access.
- OrderbookV2: Level-Bookkeeping, Vault-Residual-Guards, Fee-Caps, Trigger-Keeper.
- OracleRouter: Source-Adapter + Admin-Flow; TWAP/Deviation Tests.
- Vault/Insurance: Rounding/Pause/Limit-Guards.
- Tests: Unit + Invariant + Integration (Keeper/Indexer).
- Backend/Frontend: Dual-Stack, Mocks raus, Base-Wiring.
- Deploy-Skripte & Docs (Base), ENV-Beispiele.

#### 14. LEGACY (V1)

- V1 bleibt deploybar (Sepolia), im Frontend als Legacy/Untermenü.
- Backend darf V1 weiterhin bedienen, V2 ist Default-Ansicht und -API.