Ethereum Exit Queue Monitor (Post-Pectra) - PRD

Purpose

To provide a real-time, user-friendly web interface that tracks and visualizes key Ethereum validator exit queue metrics post-Pectra using JSON-RPC via QuickNode and stores the data in a SQL database for historical analysis.

Target Audience

- Staking operators
- Institutional Ethereum validators
- Protocol researchers
- Power users and solo stakers

Key Features

1. Validator Status Tracking
- Count of all validators with status:
  - 0x01 (Active, pending exit)
  - 0x02 (Exited or pending withdrawal)

2. Exit Request Analysis
- Track validators in each status who have submitted full exit requests:
  - Full exits from 0x01 set
  - Full exits from 0x02 set

3. Churn Limit Consumption
- Per block, calculate how much of the exit churn limit is consumed.

4. Current Exit Epoch
- Track the current exit_epoch assigned to new exit requests.

5. Partial Exit Request Tracking
- Count of partial exit requests per block.

6. Excess Fee Calculation

- Display the current 'excess' value which impacts fees for partial exits.

Backend Architecture

Technologies:

- Python 3.11+
- FastAPI for APIs and background workers
- SQLAlchemy for ORM
- PostgreSQL for storage
- APScheduler / Celery for scheduled data collection jobs

Ethereum Interaction:

- QuickNode JSON-RPC APIs (via web3.py)
- Use relevant Beacon API endpoints for validator status and exit queue data

Frontend Architecture

Technologies:

- React (with Vite)
- TailwindCSS for styling
- Chart.js or Recharts for metrics visualization
- Axios for API calls

Core UI Components:

- Dashboard (aggregate metrics, epoch exit, churn graph)
- Validators Table (filterable by status, exit intent)
- Block Metrics View (churn usage, partial exits)
- Settings & Node Info Panel

Data Schema (PostgreSQL Tables)

validators:

- validator_index: INTEGER
- status: VARCHAR
- has_exit: BOOLEAN
- is_partial_exit: BOOLEAN
- exit_epoch: INTEGER
- updated_at: TIMESTAMP

block_metrics:

- block_number: BIGINT
- churn_used: INTEGER
- partial_exit_count: INTEGER
- excess_value: NUMERIC
- timestamp: TIMESTAMP

Non-Functional Requirements

- System should handle historical block replay (e.g. backfilling data)
- Near real-time updates (<30s latency for most metrics)
- Responsive UI across desktop and tablet

Stretch Goals

- Historical graph of churn consumption over time
- Alert system for churn saturation or spikes in exit requests
- Export data as CSV