# L1Beat Internship Take-Home Task

## Live Coqnet Explorer (Fullstack Version)

### Objective

Build a mini fullstack web app that fetches and stores live block data from Coqnet, exposes an API, and displays the data in a frontend dashboard.

### Task Breakdown

**1. Backend (Data Collector + API)**

- Create a background job (cron or interval-based) that:
- Fetches the latest block data every 30 seconds (from the provided API)
- Extracts the following fields:
- blockNumber
- timestamp
- transactionCount
- gasUsed
- Saves each block into a database (SQLite or MongoDB recommended)
- Build an API with the following endpoints:
- GET /api/blocks → Returns the latest N blocks
- GET /api/blocks/stats → Returns:
- Average block time
- Total transactions in the last 100 blocks
- Average gas used
- Latest block timestamp

**2. Frontend (Dashboard UI)**

Build a dashboard at /explorer that:

- Shows the latest block number and timestamp

Displays:

- Total transactions in the last 100 blocks
- Average gas used
- Average block time (in seconds)
- Includes a simple chart showing:
- X-axis: block number
- Y-axis: transaction count
- Optional: Add a “Refresh” button to manually re-fetch data from your backend

### Provided Resources

You can use this endpoint to get the latest block from Coqnet:
https://23aqu1537ysecjmxw11xhjpra6bptbsps5d4xxupt8hn2queag.idx2.solokhin.com/api/blocks/latest

### Submission Checklist

- GitHub repo with complete code (frontend + backend)
- README.md with:
- Setup instructions
- API documentation for your endpoints
- Screenshots or a GIF of the working app
- (Optional) Live demo link

**Deadline**

3–4 days from receiving the task

**Evaluation Criteria**

- Clear and modular code (backend + frontend)
- Accurate metrics and data handling
- Clean, responsive UI
- Good practices (error handling, loading states, API design)
- Bonus: Deployment, refresh UX, polished charts