Constellations: a real-time indexer for Stargaze

Fabien Penso

@fabienpenso

https://pen.so

- 1998: Created linuxfr.org
- 2011: Bought some BTC
- 2021: Seed investor in Stargaze
- 2022: Started blockchain-related coding

What is Stargaze?

Cosmoverse 2021: "Most decentralised chain"

Now Minting

Stargazer: The Game

Created by: berny.stars

Tap to play* and earn the highest score! Stargazer: The game, A NFT Mini Game wrapped in a Stargaze theme. Will you be the best? *You can only play if you own the NFT.

Open edition mint ends in **10h:5m:57s**

Quantity	Royalties	Minted	Airdropped
Created on demand	3%	654	23

Open edition sale — Live

Price 200 STARS • Max 25 tokens

to Mint

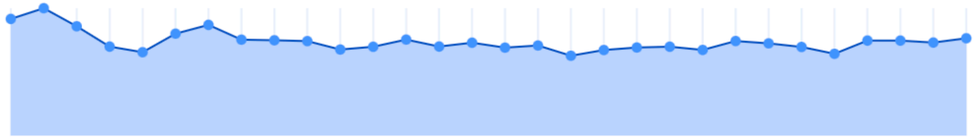
1 Mint

Traffic

Overview

24 Hours 7 Days **30 Days** 7 JUNE — 7 JULY

Unique Visitors
369.49k



Total Requests
3.91B

Percent Cached
85.85%

Total Data Served
37 TB

Data Cached
31 TB

View more analytics Download data

Indexers are mandatory

Users expect web3 products
with web2 performance

Blockchain launch

Smart Contracts

Infrastructure

Indexer

Frontend

A lot of blockchain launches are all and only about smart contracts, but that's a tiny 10% of the whole shebang. The remaining 90% comes from the frontend, the backend, and the infrastructure.

The significance of these components is often overlooked, yet they're crucial for developing a high-quality product. It requires approximately two years to reach the level of Stargaze, accounting for all these elements.

Ipfs gateway slow

novaknole Apr '21

I know This question might have been asked many times, but i'd appreciate the answer in 2021 too.

Basically, I use `ipfs-desktop` on mac. Then , I add it as `ipfs add filename.txt`

Then i check it on `myowngateway.com/ipfs/cid` and also on `ipfs.io/ipfs/cid`

As it turns out, `myowngateway.com` is so slow that 98% of the time, it never resolves to a file and just returns 504 timed out. As for the `ipfs.io` , 85-90% of the time, it resolves to a file somehow quickly(sometimes takes 5 min too), but sometimes it just never resolves and hangs indefinatelly.

After I do `ipfs add filename.txt`, I also check on another computer this command: `ipfs cat cid` and it turns out this is super fast , immediatelly returns the file...

Question 1: How can I make sure that my gateway is fast and not this slow (as in never resolving to a file). Note that if `ipfs.io` gateway loads the file, then my gateway also starts loading it instantly.

Question 2: If I don't use gateway, is there any chance that i can do ipfs fetch from javascript ? I know that `ipfs cat` is a terminal command, but anything like this from javascript and still not using gateway ?

Very slow IPFS

Very slow RPC

IPFS: The (Very Slow) Distributed Permanent Web

2018-05-08 / Blockchain

IPFS stands for InterPlanetary File System, but you could simply consider it as a distributed, permanent, but ridiculously slow, not properly functioning version of web. You could upload any static file and static website to IPFS. And the whole swarm would probably distribute your files to the moon, that might be why IPFS is so fucking slow.

ref:
https://ipfs.io/

What is Constellations?

All CosmWasm contracts deployed on Stargaze automatically get indexed by Constellations. As a developer you get a free real-time API, enabling development of sites as fast as stargaze.zone and pixelwizards.art.

@shan3v

Thank you @fabienpenso for Constellations❤️ @IBCMuffins

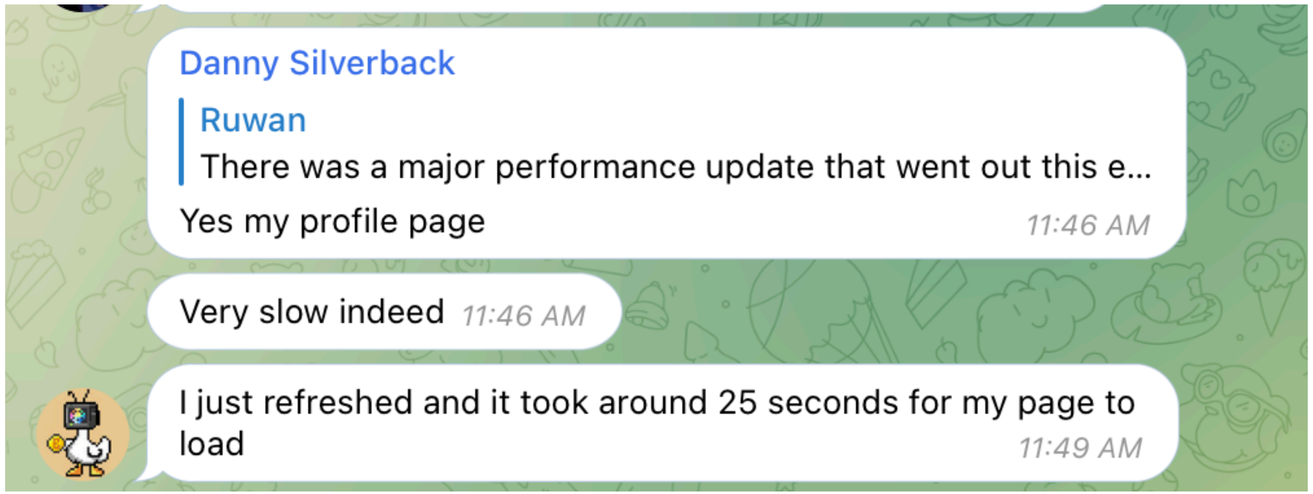
Constellations indexer [...] singlehandedly sped up development [...] threefold. @josefleventon

... and many using the public API

Stargaze: Benefits from no indexer to indexer?

No Indexer

- Front-end fetch **lots** of data directly from the chain
- Some pages took > 30sec. Our launchpad failed when too much content was loaded. Not sustainable.



With Indexer

Page loaded < 500ms

How to build an indexer?

Make it work
Make it robust
Make it fast

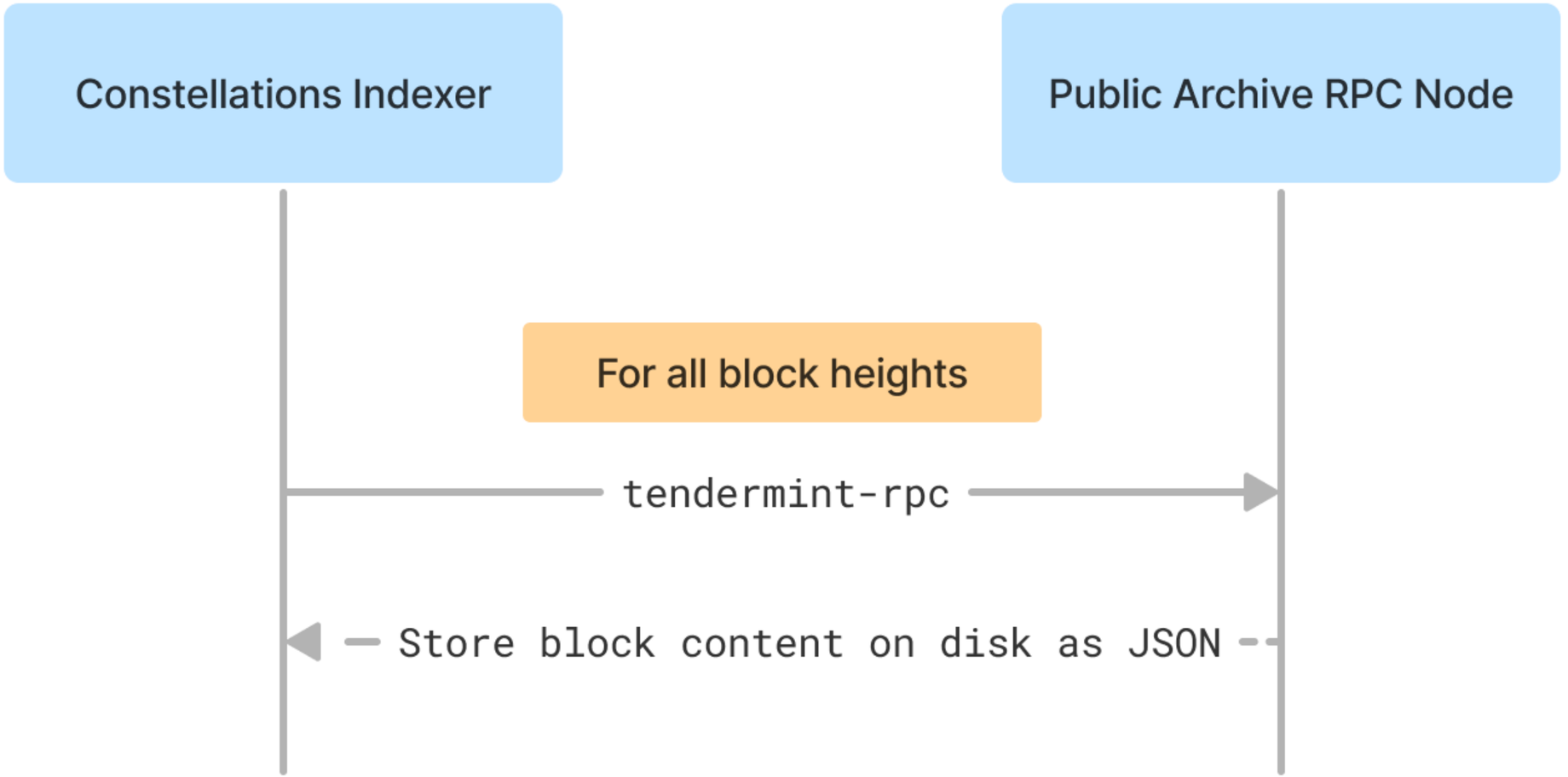
Rust

- ✓ Memory safety (multi-thread)
- ✓ Most loved language 8 years in a row (Stack Overflow)
- ✓ Performance
- ✓ Functional programming

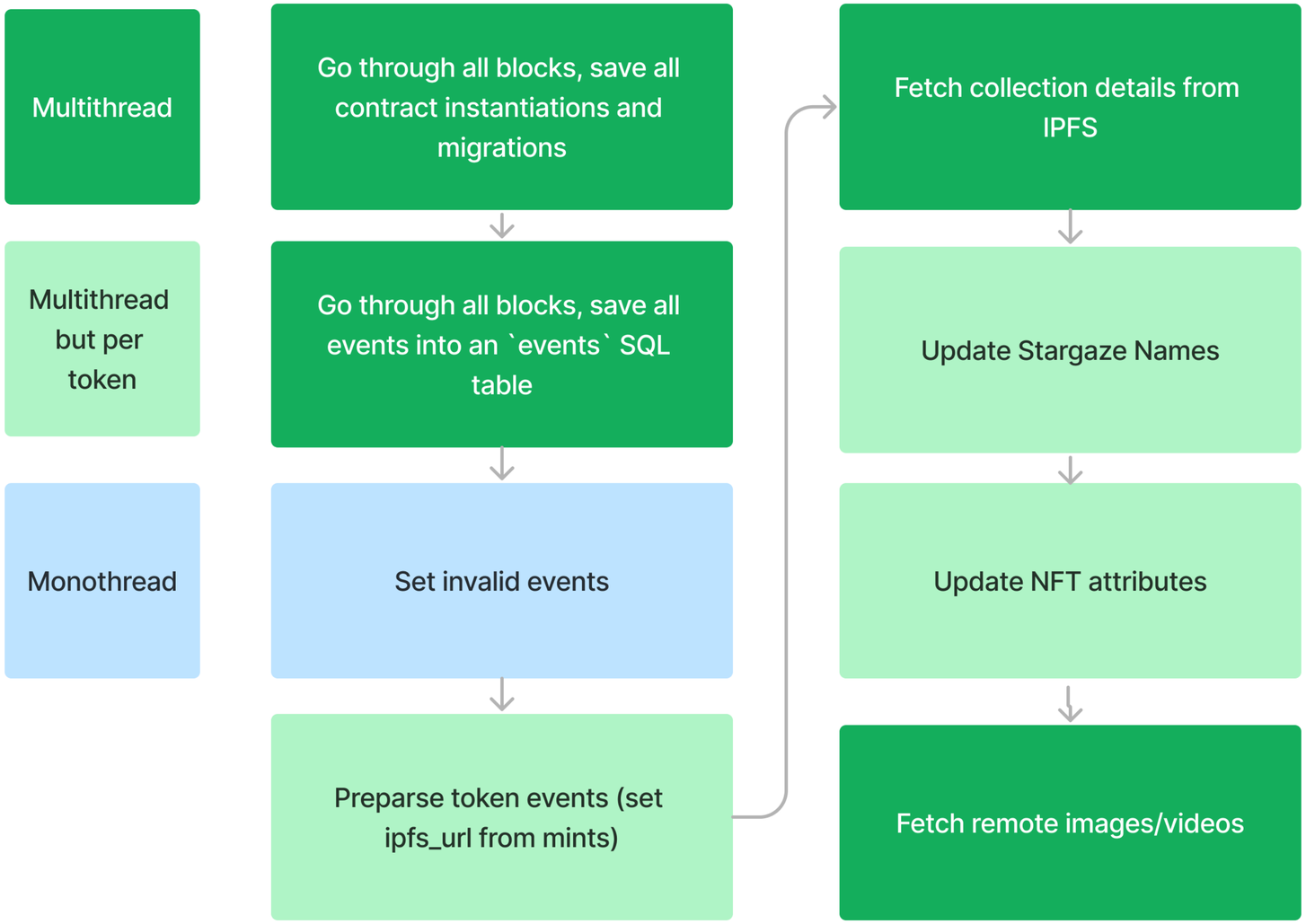
But not enough, lots of time spent tweaking performance, and making things faster, or adding caching layers

1. Make it work

Fetching blocks



Parsing locally saved blocks



Add a GraphQL API

And a BI tool (Metabase)

2. Make it robust

You're dealing with on-chain and off-chain data, you can't trust any of it. Bogus utf8, changing smart contracts events. Must build for errors and resilience.

```
loop {
    let _contract_info = client.contract_info(&args.addr).await?;
    // ...
}
```

```
loop {
    let _contract_info = match client.contract_info(&args.addr).await {
        Ok(contract_info) => contract_info,
        Err(e) => {
            println!("Error: {:#?}", e);
            continue;
        }
    };
    // ...
}
```

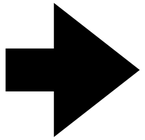
```rust
#[stargaze::test]
async fn parse_settle_auctions() -> Result<(), anyhow::Error> {
    // Parse settle auction
    let block = 5533383;
    index_blocks(test_ctx.ctx(), &vec![block]).await?;

    let events = entity::event::Entity::find()
        .settle_auctions()
        .all(test_ctx.db())
        .await?;

    assert_that!(events).has_length(1);

    Ok(())
}
```

```rust
#[stargaze::test]
async fn parse_create_auction() -> Result<(), anyhow::Error> {
    let collection_addr = "stars10hm2p3ll26zkzwmm202mfdmqy0x0qaxjtqcu6y9cl45razea84hs62p5zn";
    let token_id = 4995;

    // Create collection
    create_collection(ctx.clone(), collection_addr, token_id, vec![1611797]).await?;

    // Parse create-auction
    let block = 5326713;
    index_blocks(ctx.clone(), &vec![block]).await?;
    update_tokens(&ctx, &[block]).await?;

    let token = entity::token::Entity::find_by_collection_addr_and_token_id(
        collection_addr.to_string(),
        token_id.to_string(),
    )
    .one(db)
    .await?
    .expect("Can't find token");

    assert!(token.for_sale);
    let expected_price = entity::price::Price::from_fields("100000000", "2.07", "0.02072")?;
    assert_that!(token.ask_price).is_equal_to(Some(expected_price));
    assert_that!(token.ask_sale_type).is_equal_to(Some("live_auction".to_string()));

    Ok(())
}
```

```
let output = quote! {
    #headers
    #ignore
    #vis #sig {
        let mut test_ctx = TestContext::config()
            .#network
            .test_ctx()
            .await?;
        test_ctx.app_ctx.archive_mode = #archive_mode;
        let ctx = test_ctx.ctx();
        let db = test_ctx.db();
        #fn_block
    }
};
```

file:///Users/penso/stargaze/constellations-code-coverage/target/coverage/html/i

top_level



LINES	FUNCTIONS	BRANCHES
43.2 %	21.96 %	100 %

Directory	Line Coverage		Functions		Branches	
entity/src	52.01%	543 / 1044	32.03%	738 / 2304	100%	0 / 0
entity/src/contract	100%	12 / 12	55.56%	25 / 45	100%	0 / 0
entity/src/event	67.42%	360 / 534	34.59%	92 / 266	100%	0 / 0
migration/src	0%	0 / 3497	0%	0 / 483	100%	0 / 0
src	63.3%	407 / 643	26.41%	1307 / 4949	100%	0 / 0
src/bin	2.14%	52 / 2428	2.63%	52 / 1974	100%	0 / 0
src/blockchain	63.18%	508 / 804	35.85%	346 / 965	100%	0 / 0
src/controllers	55.38%	684 / 1235	20.66%	586 / 2836	100%	0 / 0
src/controllers/event_controller	82.22%	259 / 315	15.64%	43 / 275	100%	0 / 0
src/database	37.25%	19 / 51	4.12%	4 / 97	100%	0 / 0
src/decorators	0%	0 / 778	0%	0 / 200	100%	0 / 0
src/errors	0%	0 / 1	0%	0 / 8	100%	0 / 0
src/fetchers	67.54%	1130 / 1673	22.16%	452 / 2040	100%	0 / 0
src/graphql	97.87%	92 / 94	25%	1 / 4	100%	0 / 0

How robust?

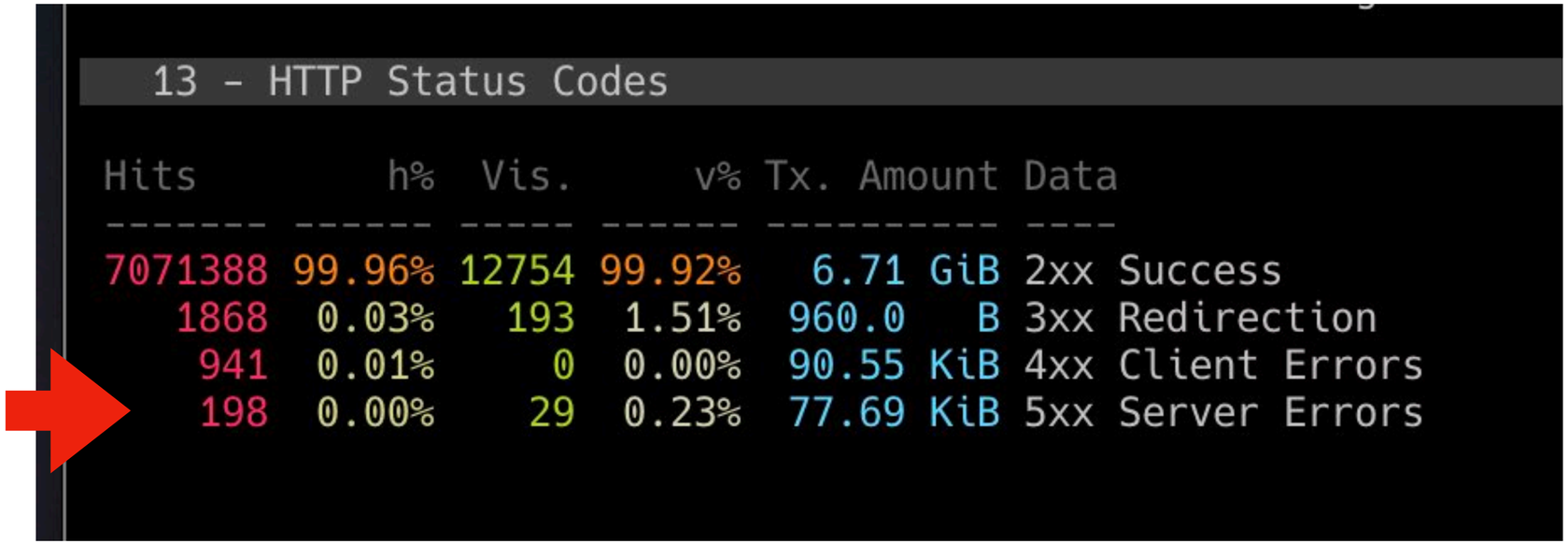
constellations | All Envs | 14D | All Categories | Custom Search | Search for events, users, tags, and more

Resolve | Ignore | Mark Reviewed | Merge | ··· | Last Seen

	GRAPH: 24h 14d	EVENTS	USERS	ASSIGNEE
Error fetching last block height Regression CONSTELLATIONS-8 22hr ago | 1mo old live_blocks		84	0	
Error fetching fetch_ipfs_urls_from_base_token_uri: utf8Error: invalid utf-8 sequence of 1 bytes ... Regression CONSTELLATIONS-14 2d ago | 2wk old stargaze_idx::fetchers::collection_fetcher		9	0	
Error encountered while processing the incoming HTTP request: SeaOrm(ConnectionAcquire) Regression CONSTELLATIONS-M 2d ago | 1mo old tracing_actix_web::middleware		1	0	
Calling https://rest.stargaze-1.publicawesome.dev/wasm/contract/stars12gx3qjgmd65tluttrkxpf5mglkn... CONSTELLATIONS-1E 7d ago | 7d old stargaze_idx::stargaze_http_client		1	0	
Fetching contract failed: StatusCode: 500 Internal Server Error CONSTELLATIONS-1D 1wk ago | 1wk old stargaze_idx::controllers::contract_controller		1	0	
Error parsing body from ipfs://bafybeihkdwehdxmed6afnrnlhygyr7wvg24lca6yqpd5rjibpfn34vzj44/310.js... CONSTELLATIONS-10 1wk ago | 3wk old stargaze_idx::stargaze_nft_client		275	0	
Error while fetching collection: token_uri isn't a String, Marking this contract as error CONSTELLATIONS-1C 1wk ago | 1wk old stargaze_idx::fetchers::collection_fetcher		1	0	
Query Error: error returned from database: duplicate key value violates unique constraint "idx-ev... CONSTELLATIONS-19 1wk ago | 1wk old stargaze_idx::recorders::event_recorder		5.5k	0	
Error while storing event Block: 5504198, Tx: 0, Msg: 0, Event: 13: Query Error: error returned f... CONSTELLATIONS-17 1wk ago | 1wk old stargaze_idx::controllers::event_controller		5.5k	0	
Future Error: task 127771791 panicked CONSTELLATIONS-1A 1wk ago | 1wk old stargaze_idx::blockchain::block_parser		77	0	
panic- ~~rust_begin_unwind~~ in panicking.rs Can't get share CONSTELLATIONS-18 Unhandled 1wk ago | 1wk old		162	0	
Error while parsing 5502325: Can't get collection info CONSTELLATIONS-1B 1wk ago | 1wk old stargaze_idx::blockchain::block_parser		64	0	
Fetching contract failed: Request error: error sending request for url (https://rest.stargaze-1.p... CONSTELLATIONS-16 2wk ago | 2wk old stargaze_idx::controllers::contract_controller		1	0	
Can't set caching: Is a directory (os error 21) CONSTELLATIONS-15 2wk ago | 2wk old stargaze_idx::fetchers::ipfs_fetcher		1	0	

1-14 of 14

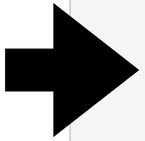
7M requests, 198 5xx errors

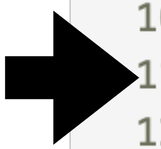


3. Make it fast

- Async: Heavy use of Tokio and spawned tasks with semaphore
- Caching: IPFS caching, blockchain caching, remote file caching, fixtures for tests
- Has to keep improving, as dataset grows over time
- The following trick improved some code from 90 minutes to less than 3 minutes

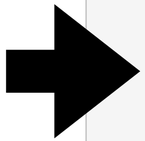
```rust
/// Elapsed: 1.206s
fn slow() {
    for i in 0..100 {
        sleep(std::time::Duration::from_millis(10));
    }
}
```



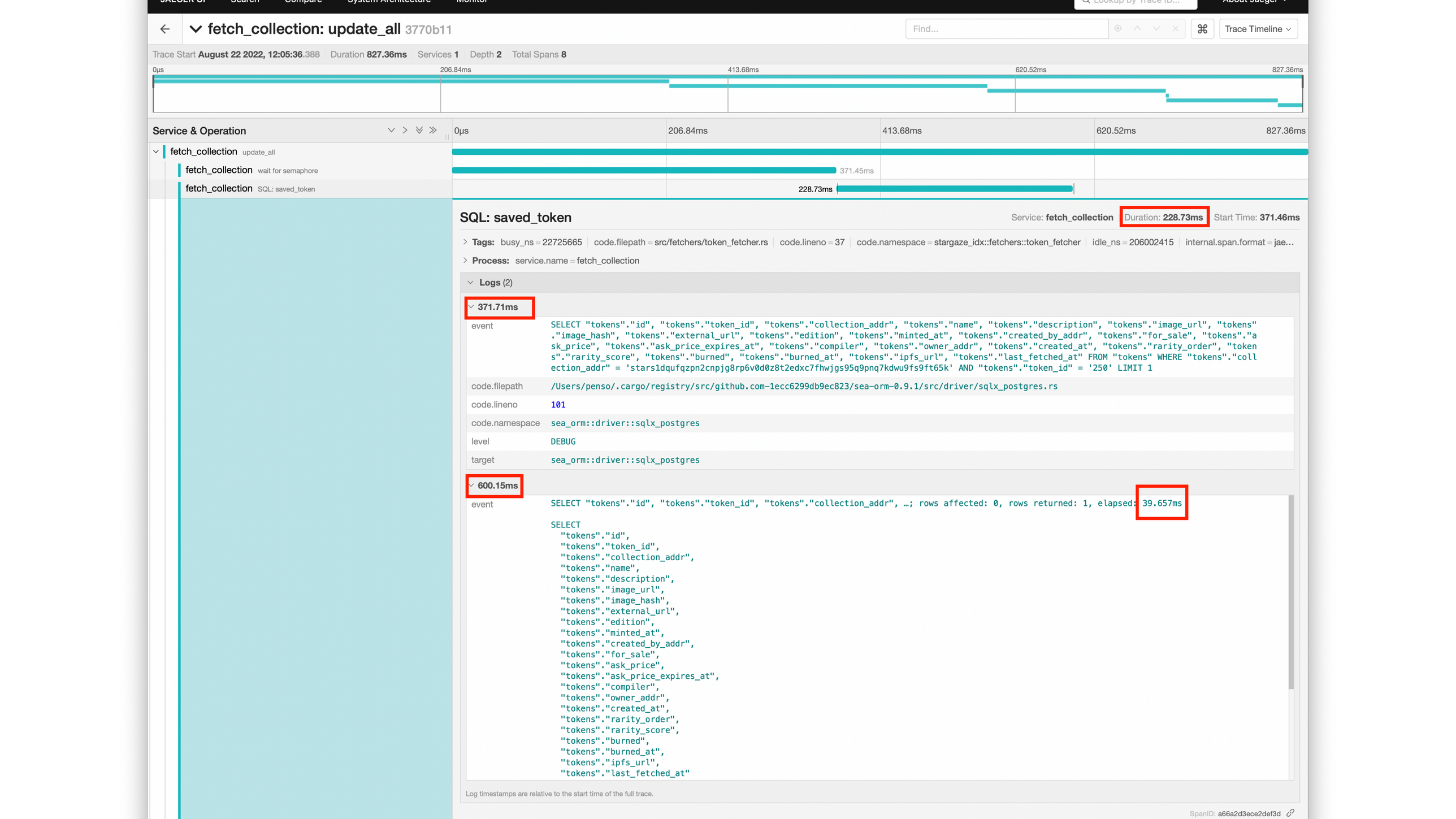


```rust
/// Elapsed: 63.132ms
async fn fast() -> Result<(), anyhow::Error> {
    // How many futures run concurrently?
    let sem = Arc::new(Semaphore::new(30));

    let mut tasks: Vec<JoinHandle<()>> = vec![];

    for i in 0..100 {
        let permit = Arc::clone(&sem).acquire_owned().await?;

        let task = tokio::spawn(async move {
            let _permit = permit;
            sleep(std::time::Duration::from_millis(10));
        });
        tasks.push(task);
    }

    // ...
}
```

Instrumentation

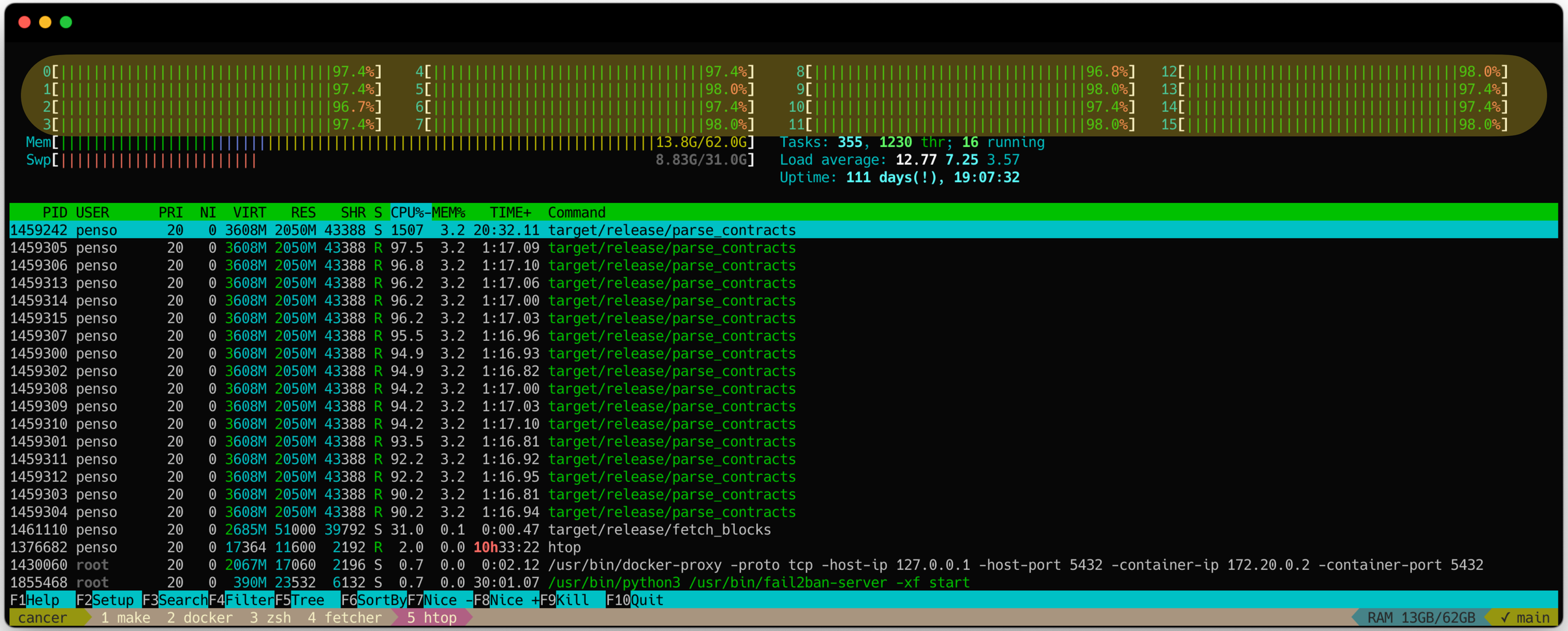


```rust
#[ComplexObject]
impl Model {
    #[tracing::instrument(name = "graphql:contract:contract_info", skip_all)]
    pub async fn contract_info(&self) -> HashMap<String, serde_json::Value> {
        (&self.contract_info).into()
    }
}
```

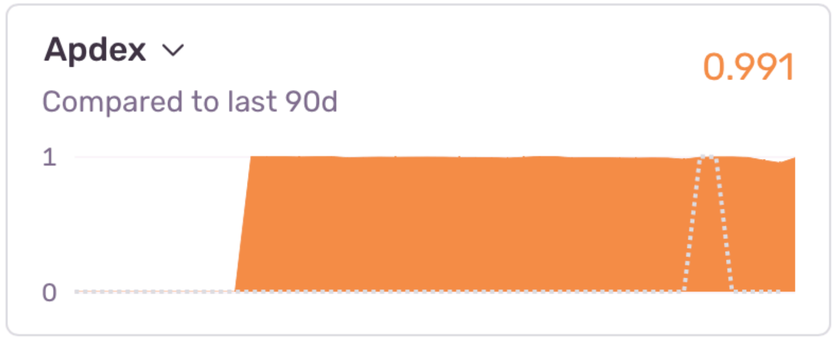

fetch_collection: update_all 3770b11
Find...
Trace Timeline
Trace Start August 22 2022, 12:05:36.388 Duration 827.36ms Services 1 Depth 2 Total Spans 8
0µs
206.84ms
413.68ms
620.52ms
827.36ms
Service & Operation
fetch_collection update_all
fetch_collection wait for semaphore
371.45ms
fetch_collection SQL: saved_token
228.73ms
SQL: saved_token
Service: fetch_collection
Duration: 228.73ms
Start Time: 371.46ms
Tags: busy_ns = 22725665 | code.filepath = src/fetchers/token_fetcher.rs | code.lineno = 37 | code.namespace = stargaze_idx::fetchers::token_fetcher | idle_ns = 206002415 | internal.span.format = jae…
Process: service.name = fetch_collection
Logs (2)
371.71ms
event
SELECT "tokens"."id", "tokens"."token_id", "tokens"."collection_addr", "tokens"."name", "tokens"."description", "tokens"."image_url", "tokens"."image_hash", "tokens"."external_url", "tokens"."edition", "tokens"."minted_at", "tokens"."created_by_addr", "tokens"."for_sale", "tokens"."ask_price", "tokens"."ask_price_expires_at", "tokens"."compiler", "tokens"."owner_addr", "tokens"."created_at", "tokens"."rarity_order", "tokens"."rarity_score", "tokens"."burned", "tokens"."burned_at", "tokens"."ipfs_url", "tokens"."last_fetched_at" FROM "tokens" WHERE "tokens"."collection_addr" = 'stars1dqufqzpn2cnpjg8rp6v0d0z8t2edxc7fhwjgs95q9pnq7kdwu9fs9ft65k' AND "tokens"."token_id" = '250' LIMIT 1
code.filepath
/Users/penso/.cargo/registry/src/github.com-1ecc6299db9ec823/sea-orm-0.9.1/src/driver/sqlx_postgres.rs
code.lineno
101
code.namespace
sea_orm::driver::sqlx_postgres
level
DEBUG
target
sea_orm::driver::sqlx_postgres
600.15ms
event
SELECT "tokens"."id", "tokens"."token_id", "tokens"."collection_addr", …; rows affected: 0, rows returned: 1, elapsed: 39.657ms
SELECT
"tokens"."id",
"tokens"."token_id",
"tokens"."collection_addr",
"tokens"."name",
"tokens"."description",
"tokens"."image_url",
"tokens"."image_hash",
"tokens"."external_url",
"tokens"."edition",
"tokens"."minted_at",
"tokens"."created_by_addr",
"tokens"."for_sale",
"tokens"."ask_price",
"tokens"."ask_price_expires_at",
"tokens"."compiler",
"tokens"."owner_addr",
"tokens"."created_at",
"tokens"."rarity_order",
"tokens"."rarity_score",
"tokens"."burned",
"tokens"."burned_at",
"tokens"."ipfs_url",
"tokens"."last_fetched_at"
Log timestamps are relative to the start time of the full trace.
SpanID: a66a2d3ece2def3d

How fast?

At indexing time

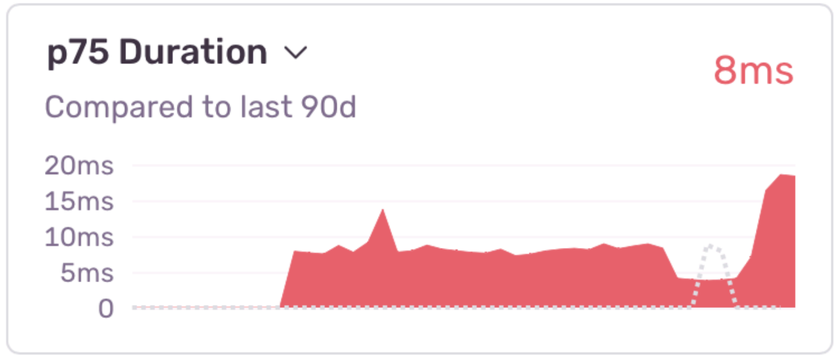


Sentry dashboard

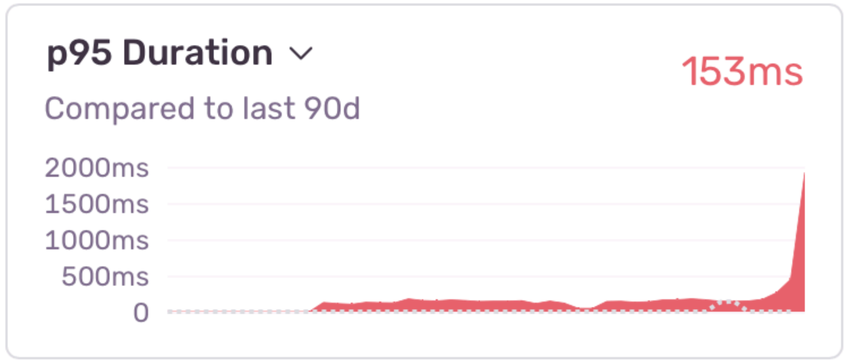


An excellent score falls in 1.00-0.94, a good score ranks from 0.93-0.85, a fair score hits 0.84-0.70 and a poor one between 0.69 and 0.49. Any lower number is unacceptable.

Source: TechTarget



The p75 threshold is the value at which 25% of transaction durations are greater than the threshold



Public GraphQL API

PRETTIFY HISTORY https://api.mainnet.stargaze.constellations.zone/graphql

```
query BidsForGivenBidder {
  events(
    dataFilters: [
      {
        name: "bidder"
        value: "stars1rmxl4fps24pe8s9uv8an3nqpng3ggyf8sfavr0"
        operator: EQUAL
      }
      {
        name: "collection"
        value: "stars19jq6mj84cnt9p7sagjxqf8hxtczwc8wlpuwe4sh62w45aheseues57n420"
        operator: EQUAL
      }
    ]
    contractFilters: [
      {
        contractType: "crates.io:sg-marketplace"
        events: [{ name: "wasm-set-bid", action: null }]
      }
    ]
    sortBy: BLOCK_HEIGHT_DESC
  ) {
    edges {
      node {
        txHash
        action
        contractInfo
        createdAt
        isValid
```


```
{
  "data": {
    "events": {
      "edges": [
        {
          "node": {
            "txHash":
"4f7d399928616b4db8fdd3e6bc1981b33aea3f0271e60d4f3e54d73
16",
            "action": null,
            "contractInfo": {},
            "createdAt": "2023-02-09T18:33:55.540607+00:
            "isValid": false,
            "eventName": "wasm-set-bid",
            "data": {
              "starsUsdRatio": "0.02929",
              "expires": "1676572436.187000000",
              "bidPrice": "5100000000",
              "collection":
"stars19jq6mj84cnt9p7sagjxqf8hxtczwc8wlpuwe4sh62w45ahese
20",
              "saleType": "auction",
              "tokenId": "7740",
              "bidPriceUsd": "149.38",
              "bidder":
"stars1rmxl4fps24pe8s9uv8an3nqpng3ggyf8sfavr0"
            }
          }
        },
```

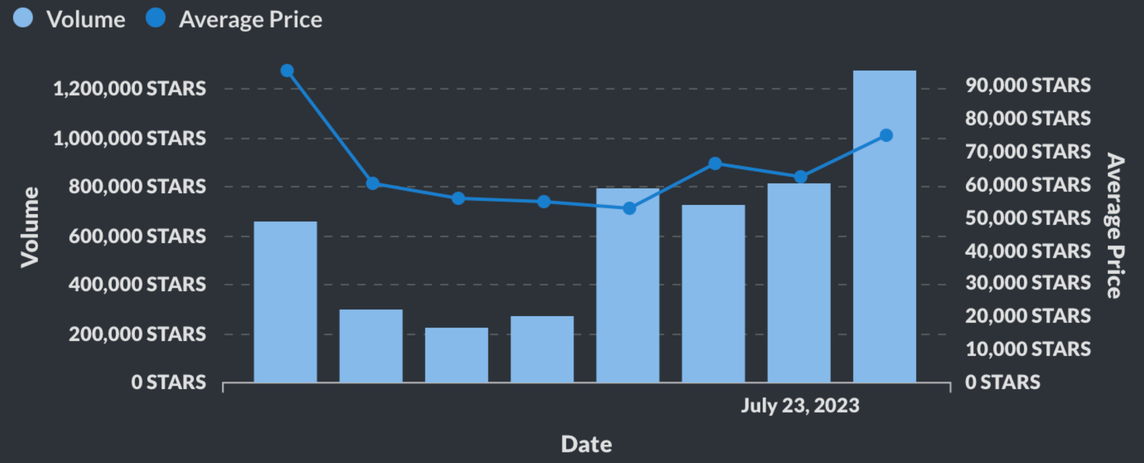

QUERY VARIABLES HTTP HEADERS

Metabase

@cosmic_calvin

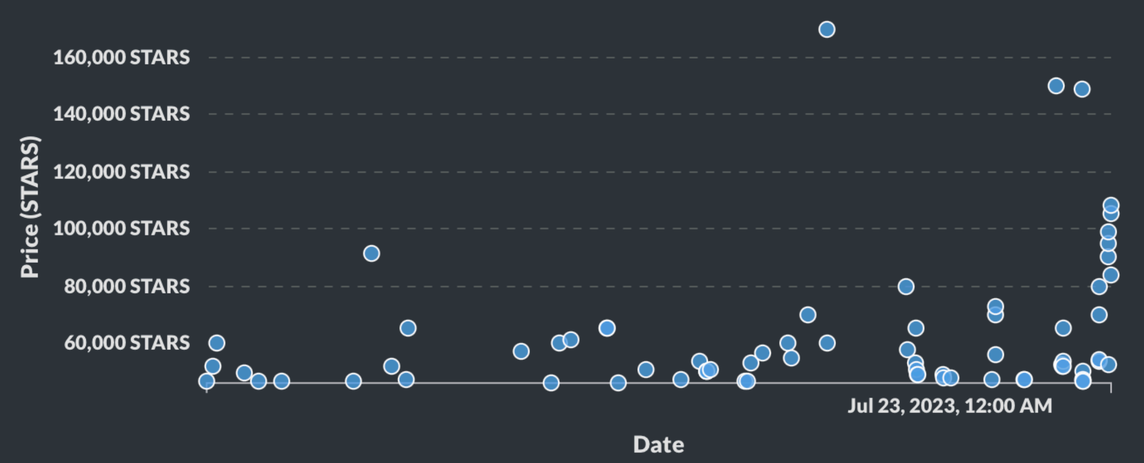
Marketplace

Price and Volume

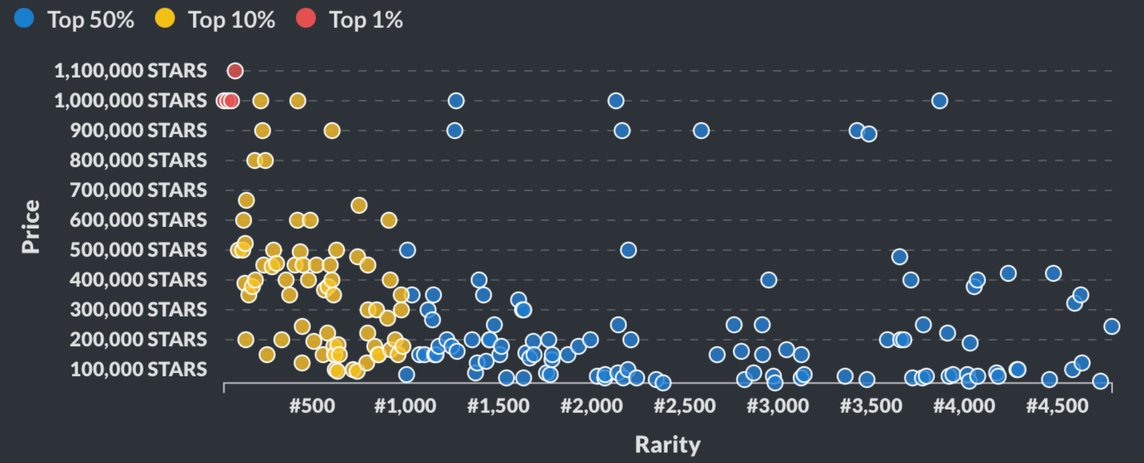


Floor Price

Sales



Price vs. Rarity



Sales

Image	STARS Price	USD Price	Rarity	Time
	108,000 STARS	$1,211	#1,055	Jul 24, 2023, 6:30 AM

Active Listings

Image	Item	Price	Seller	Rarity	Listing Date
	5084	50,793 STARS	stars1u...ax6uh	#6,091	Jul 24, 2023, 4:10 AM

Deployment

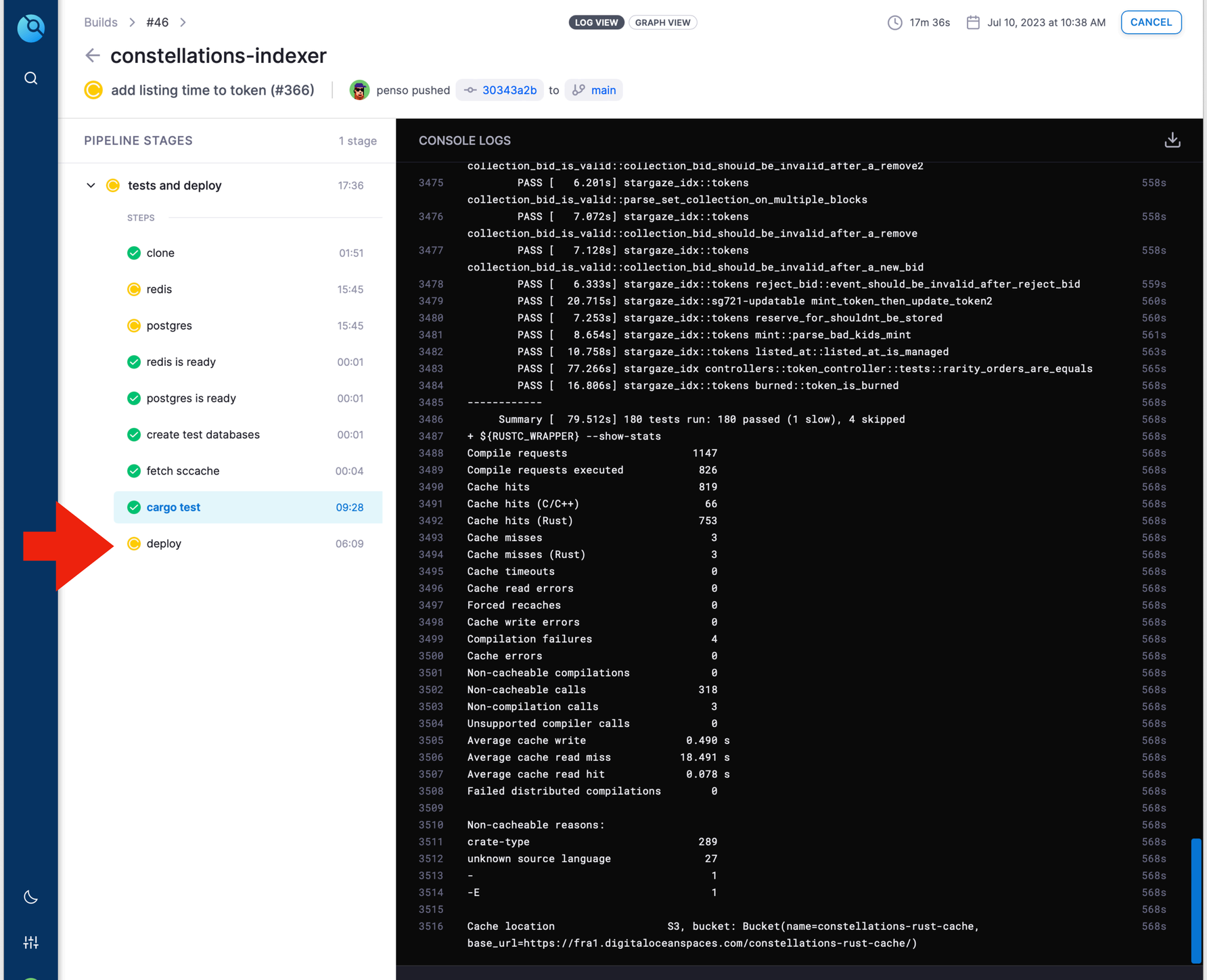
> Ansible based
> 2 base-metal servers (Hetzner + Interserver) for redundancy
> Cloudflare load balancer
> Drone CI + our own runners (Github CI was too slow)

```
ansible-playbook -e @secrets.enc \
  --vault-password-file password_file \
  -i production copy_db_and_deploy.yml
```

1. Dump local databases (async)
2. Copy them over to production servers (async)
3. Create DB + inject data
4. Update metabase DBs
5. Deploy code

```
ansible-playbook -e @secrets.enc \
  --vault-password-file password_file \
  -i production deploy.yml
```

1. detect new nginx ports
2. git checkout
3. cargo build
4. set ENV variables
5. run DB migration
6. rotate logs
7. wait for new ports to be up + indexed
8. reload nginx config with new proxy ports


Builds
#46
LOG VIEW
GRAPH VIEW
17m 36s
Jul 10, 2023 at 10:38 AM
CANCEL
constellations-indexer
add listing time to token (#366)
penso pushed
30343a2b
to
main
PIPELINE STAGES
1 stage
tests and deploy
17:36
STEPS
clone
01:51
redis
15:45
postgres
15:45
redis is ready
00:01
postgres is ready
00:01
create test databases
00:01
fetch sccache
00:04
cargo test
09:28
deploy
06:09
CONSOLE LOGS
collection_bid_is_valid::collection_bid_should_be_invalid_after_a_remove2
3475 PASS [6.201s] stargaze_idx::tokens 558s
collection_bid_is_valid::parse_set_collection_on_multiple_blocks
3476 PASS [7.072s] stargaze_idx::tokens 558s
collection_bid_is_valid::collection_bid_should_be_invalid_after_a_remove
3477 PASS [7.128s] stargaze_idx::tokens 558s
collection_bid_is_valid::collection_bid_should_be_invalid_after_a_new_bid
3478 PASS [6.333s] stargaze_idx::tokens reject_bid::event_should_be_invalid_after_reject_bid 559s
3479 PASS [20.715s] stargaze_idx::sg721-updatable mint_token_then_update_token2 560s
3480 PASS [7.253s] stargaze_idx::tokens reserve_for_shouldnt_be_stored 560s
3481 PASS [8.654s] stargaze_idx::tokens mint::parse_bad_kids_mint 561s
3482 PASS [10.758s] stargaze_idx::tokens listed_at::listed_at_is_managed 563s
3483 PASS [77.266s] stargaze_idx controllers::token_controller::tests::rarity_orders_are_equals 565s
3484 PASS [16.806s] stargaze_idx::tokens burned::token_is_burned 568s
3485 ------------ 568s
3486 Summary [79.512s] 180 tests run: 180 passed (1 slow), 4 skipped 568s
3487 + ${RUSTC_WRAPPER} --show-stats 568s
3488 Compile requests 1147 568s
3489 Compile requests executed 826 568s
3490 Cache hits 819 568s
3491 Cache hits (C/C++) 66 568s
3492 Cache hits (Rust) 753 568s
3493 Cache misses 3 568s
3494 Cache misses (Rust) 3 568s
3495 Cache timeouts 0 568s
3496 Cache read errors 0 568s
3497 Forced recaches 0 568s
3498 Cache write errors 0 568s
3499 Compilation failures 4 568s
3500 Cache errors 0 568s
3501 Non-cacheable compilations 0 568s
3502 Non-cacheable calls 318 568s
3503 Non-compilation calls 3 568s
3504 Unsupported compiler calls 0 568s
3505 Average cache write 0.490 s 568s
3506 Average cache read miss 18.491 s 568s
3507 Average cache read hit 0.078 s 568s
3508 Failed distributed compilations 0 568s
3509 568s
3510 Non-cacheable reasons: 568s
3511 crate-type 289 568s
3512 unknown source language 27 568s
3513 - 1 568s
3514 -E 1 568s
3515 568s
3516 Cache location S3, bucket: Bucket(name=constellations-rust-cache, 568s
base_url=https://fra1.digitaloceanspaces.com/constellations-rust-cache/)

token bids

```
query Bids($filters: [ContractFilter!], $dataFilters: [DataFilter!]) {
  events(contractFilters: $filters, dataFilters: $dataFilters) {
    edges {
      node {
        data
      }
    }
  }
}
```

```
{
  "filters": [
    {
      "contractType": "crates.io:sg-marketplace",
      "events": [
        {
          "name": "wasm-set-bid",
          "action": null
        }
      ]
    }
  ],
  "dataFilters": []
}
```

Try it out live on explorer

What's next?

- Add parsing verification based on previous successful run
- Index messages
- Index state changes instead of events? R&D
- Support IBC transfers
- Index more non cosmwasm events
- Index other blockchains?


It’s done.