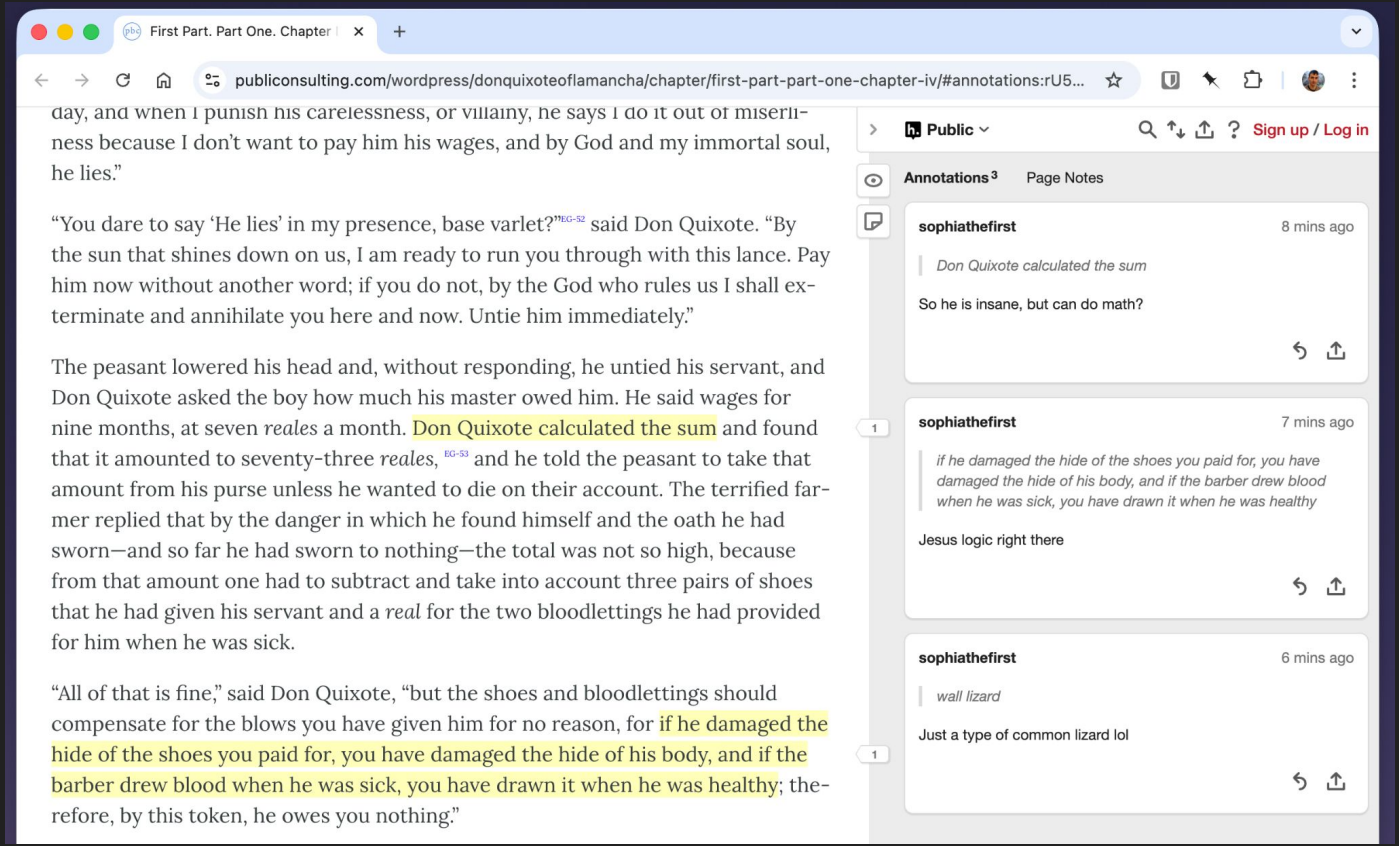
Transactional Job Queues and the Two Generals' Problem

Our worst outage in ten years of running Hypothesis, and how we fixed it (with Postgres)

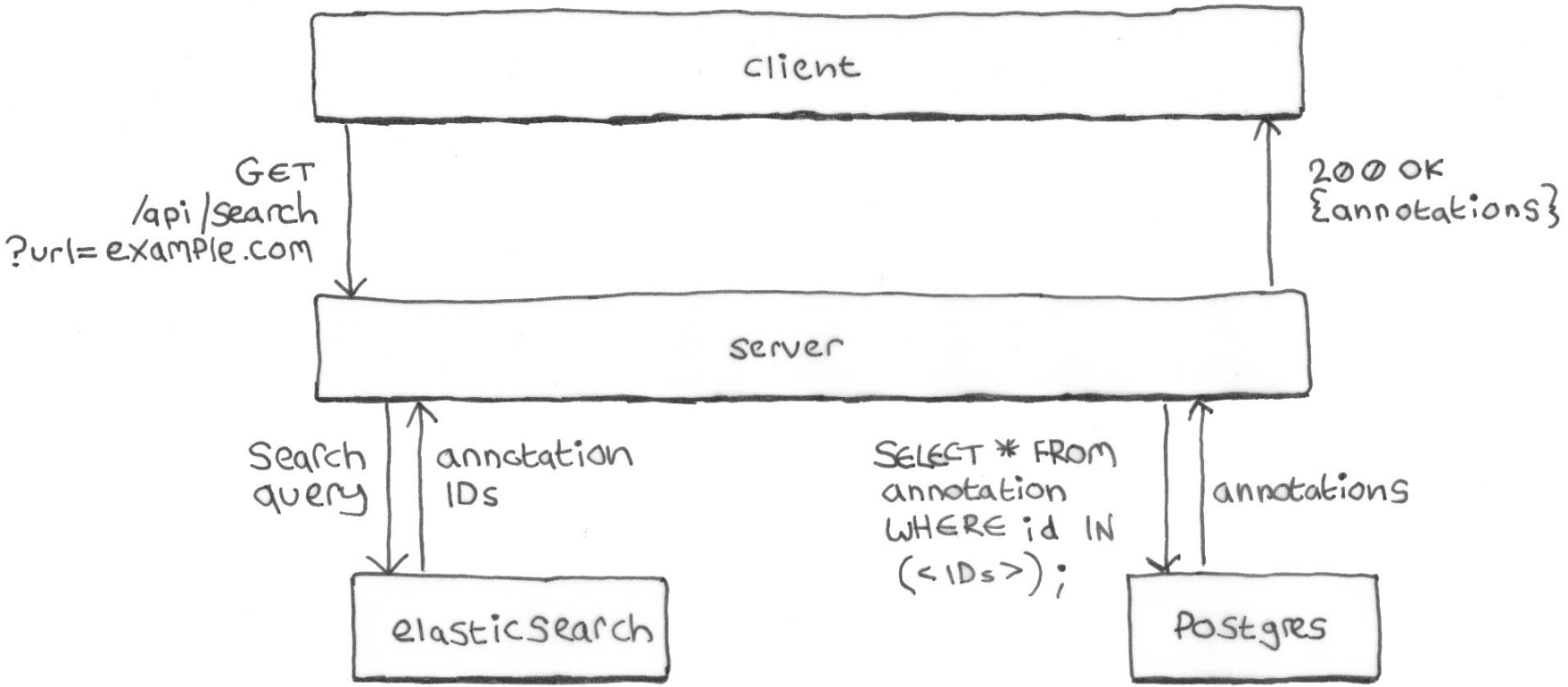
https://seanh.cc/ ← Currently available for software engineering work!

Hypothesis


First Part. Part One. Chapter
publiconsulting.com/wordpress/donquixoteoflamancha/chapter/first-part-part-one-chapter-iv/#annotations:rU5...
day, and when I punish his carelessness, or villainy, he says I do it out of miserli-
ness because I don't want to pay him his wages, and by God and my immortal soul, he lies."
"You dare to say 'He lies' in my presence, base varlet?" said Don Quixote. "By the sun that shines down on us, I am ready to run you through with this lance. Pay him now without another word; if you do not, by the God who rules us I shall exterminate and annihilate you here and now. Untie him immediately."
The peasant lowered his head and, without responding, he untied his servant, and Don Quixote asked the boy how much his master owed him. He said wages for nine months, at seven *reales* a month. Don Quixote calculated the sum and found that it amounted to seventy-three *reales*, and he told the peasant to take that amount from his purse unless he wanted to die on their account. The terrified farmer replied that by the danger in which he found himself and the oath he had sworn—and so far he had sworn to nothing—the total was not so high, because from that amount one had to subtract and take into account three pairs of shoes that he had given his servant and a *real* for the two bloodlettings he had provided for him when he was sick.
"All of that is fine," said Don Quixote, "but the shoes and bloodlettings should compensate for the blows you have given him for no reason, for if he damaged the hide of the shoes you paid for, you have damaged the hide of his body, and if the barber drew blood when he was sick, you have drawn it when he was healthy; therefore, by this token, he owes you nothing."
Public
Sign up / Log in
Annotations 3
Page Notes
sophiathefirst
8 mins ago
Don Quixote calculated the sum
So he is insane, but can do math?
sophiathefirst
7 mins ago
if he damaged the hide of the shoes you paid for, you have damaged the hide of his body, and if the barber drew blood when he was sick, you have drawn it when he was healthy
Jesus logic right there
sophiathefirst
6 mins ago
wall lizard
Just a type of common lizard lol

How Hypothesis fetches annotations

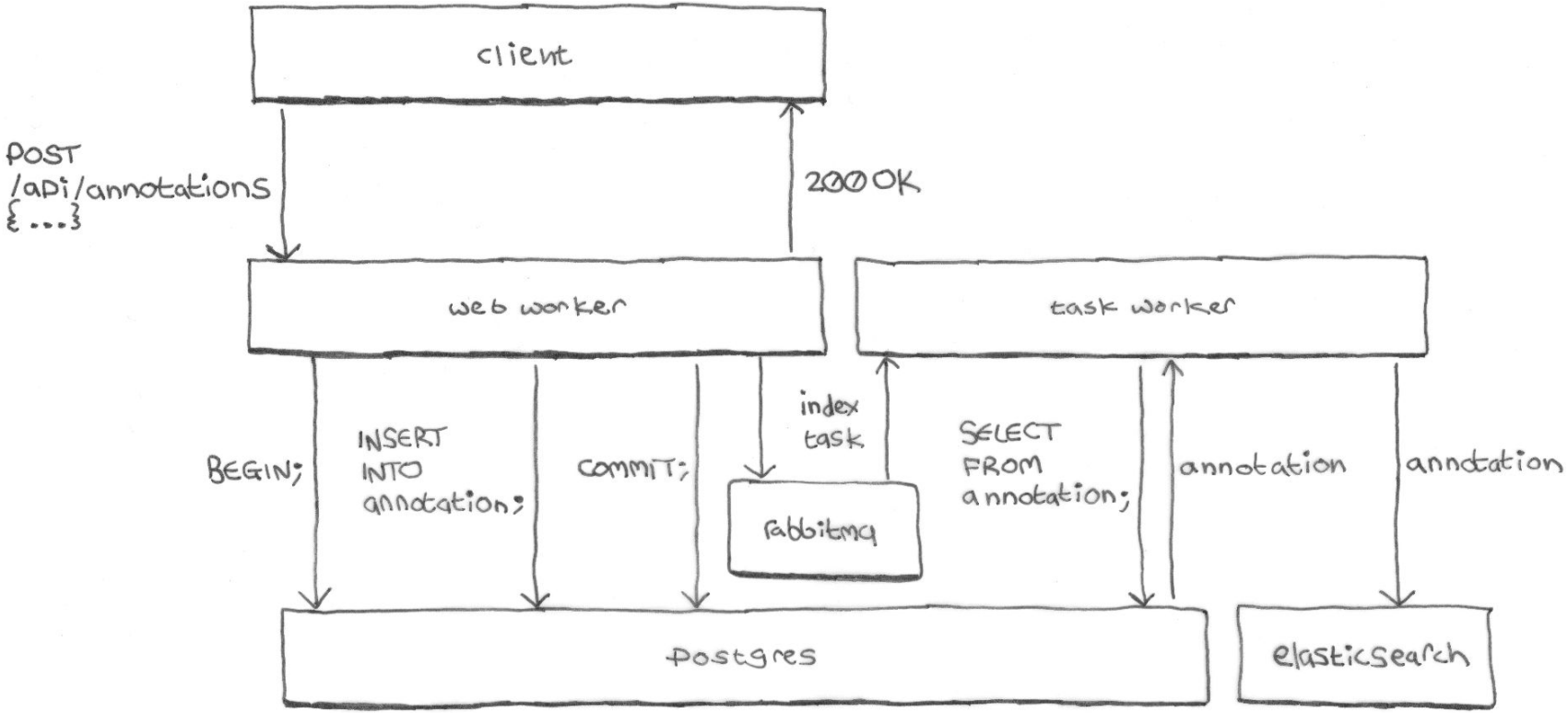
/search?url=https://example.com/foo



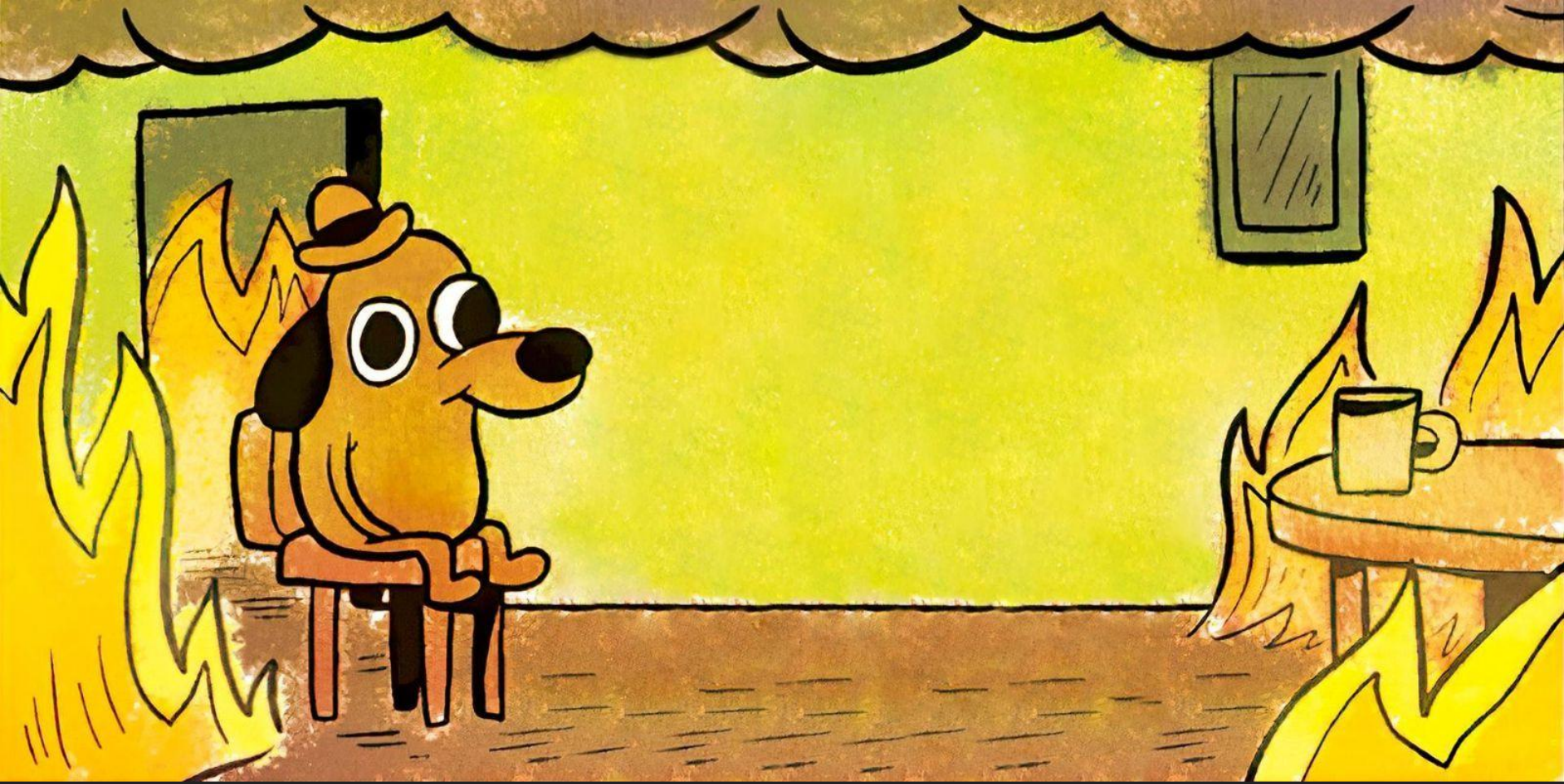
How Hypothesis fetches annotations

Annotation missing from Elasticsearch = effectively missing from the app.

So getting annotations into Elasticsearch is really important, then!



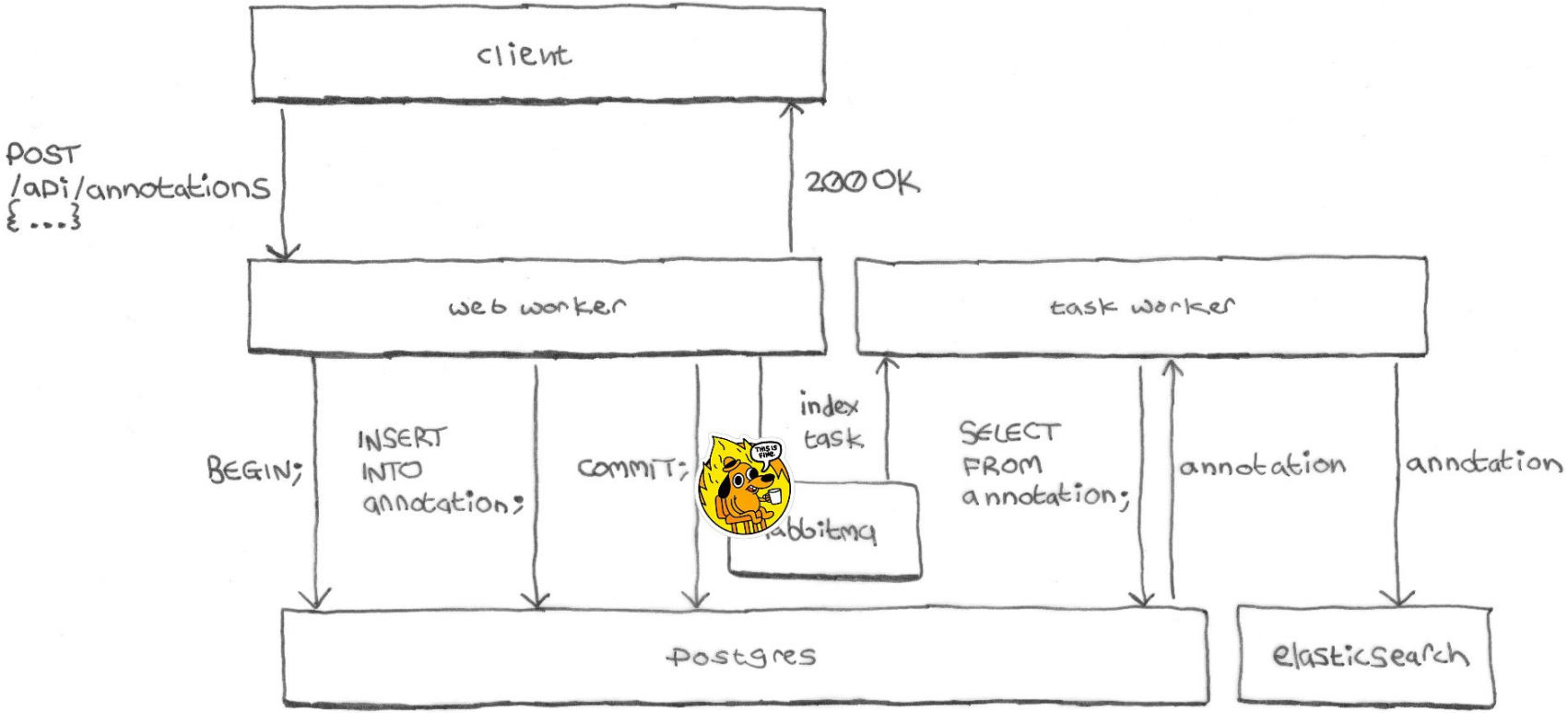
How Hypothesis *used to* save annotations



It worked, until it didn't.

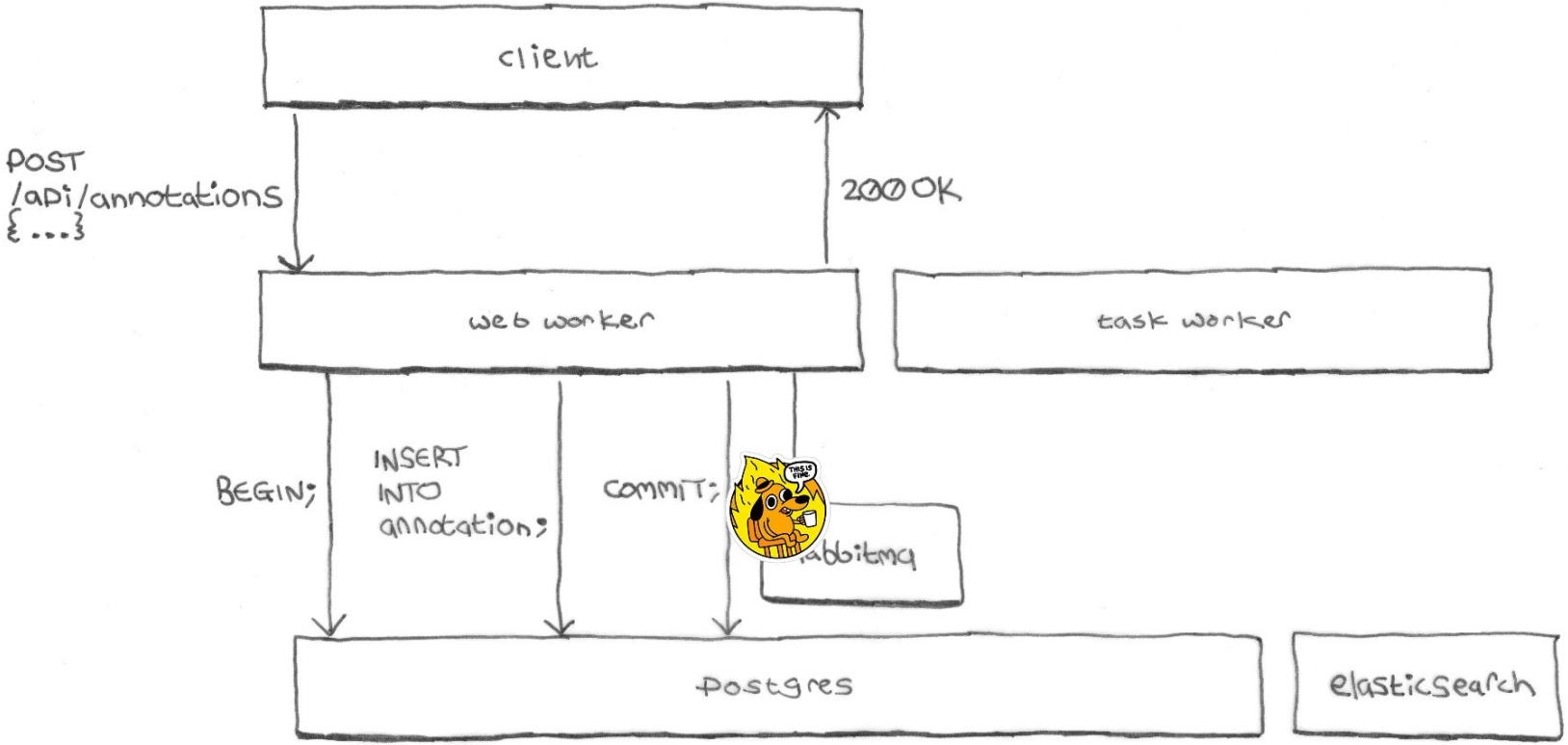
Any guesses?

- All sorts of alarms firing
- Logs flooded with seemingly unrelated error messages: everything was failing
- CPU usage spiked and pinned to the ceiling
- General service degradation:
  - Requests across all endpoints erroring, slow, or timing out
  - But all of this only *intermittently*
- User *sometimes* seeing errors when saving annotations
- Some users reported **annotations appearing to save successfully, then disappearing**

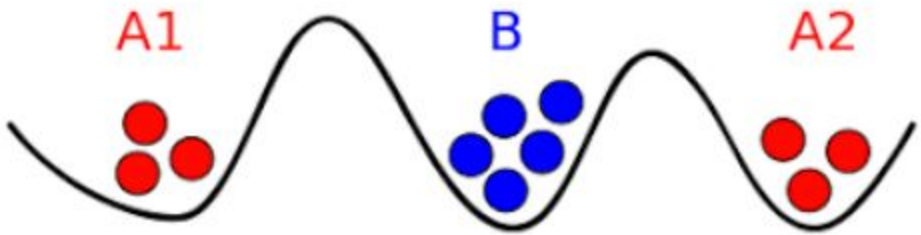


RabbitMQ was down!

Two hours to restore service

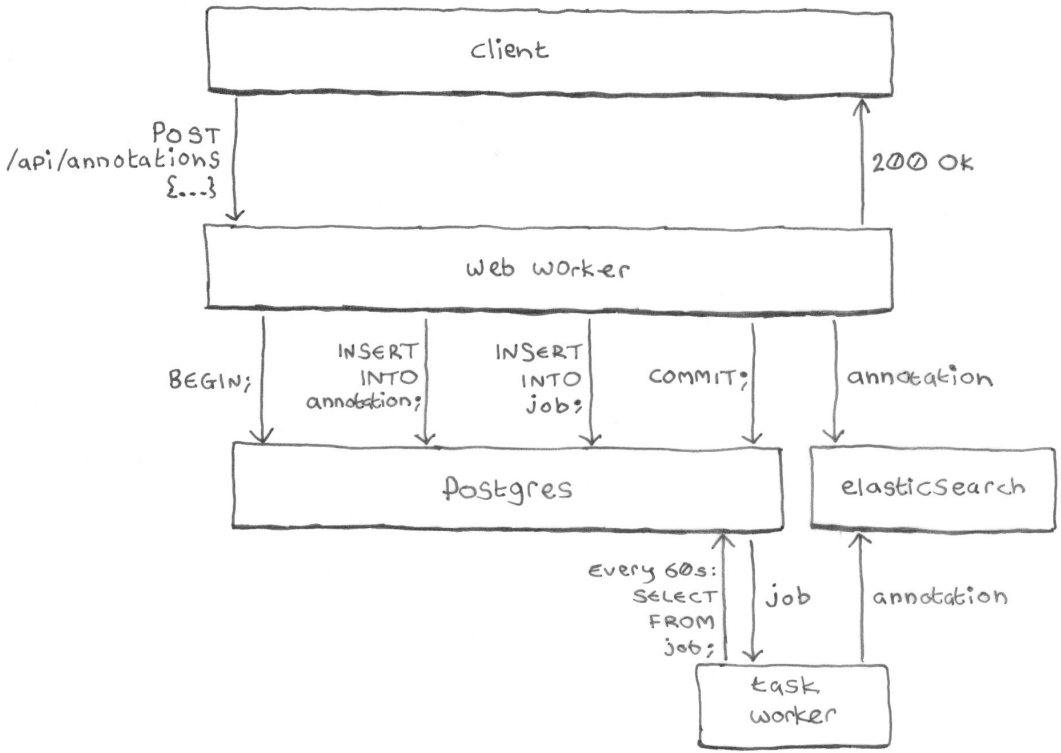


21.5K annotations saved to Postgres but not to Elasticsearch



It's the Two Generals Problem!

How we fixed it (with Postgres)



How Hypothesis now saves annotations

(separation of concerns)

Direct indexing = fast

Periodic indexing = reliable

`job` table in Postgres

```
CREATE TABLE job (
  id INTEGER PRIMARY KEY GENERATED ALWAYS AS IDENTITY (CYCLE),
  name TEXT NOT NULL, -- Example: 'index'
  enqueued_at TIMESTAMP WITHOUT TIME ZONE DEFAULT now() NOT NULL, -- For sorting
  scheduled_at TIMESTAMP WITHOUT TIME ZONE DEFAULT now() NOT NULL,
  expires_at TIMESTAMP WITHOUT TIME ZONE DEFAULT now() + interval '30 days' NOT NULL,
  priority INTEGER NOT NULL,
  tag TEXT NOT NULL, -- Example: 'storage.create_annotation'
  kwargs JSONB DEFAULT '{}'::jsonb NOT NULL -- Example: {annotation_id: 'xyz'}
);
```

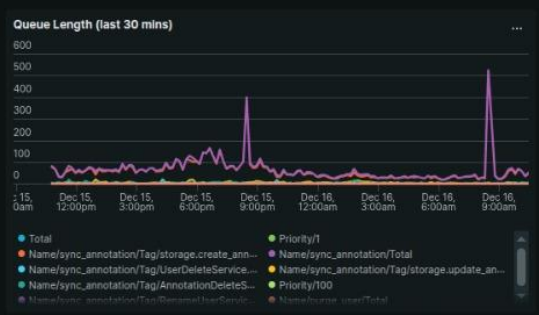
`job` table in Postgres

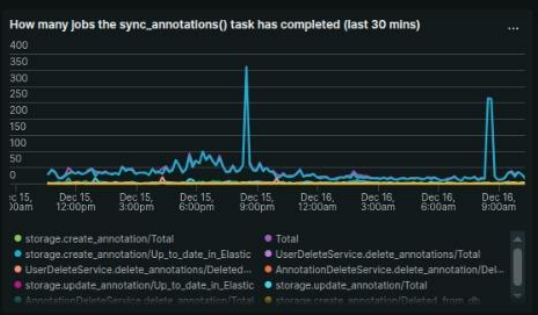
```
INSERT INTO job (name, scheduled_at, priority, tag, kwargs)
VALUES (
  'index',
  now() + interval '60 seconds',
  1,
  'storage.create_annotation',
  '{"annotation_id": "xyz"}'
);
```

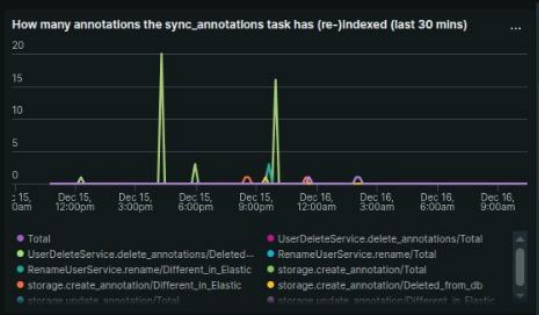
```
def sync_annotations():   # Gets called once a minute.
  fetch up to 2500 rows from job table   #  Each job contains an annotation_id (primary key).
  fetch the corresponding annotations from Postgres          # Fetch by primary key (id).
                                ...and from Elasticsearch    # Also by id.
  for each job:
    if annotation not in Postgres: delete job
    else if annotation already in Elasticsearch: delete job
    else: write annotation to Elasticsearch   # And don't delete the job.
```

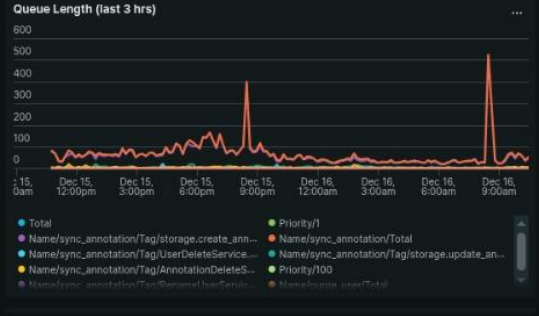
```
SELECT      id, kwargs
FROM        job
WHERE       job.name = 'index'
AND         job.scheduled_at < now()
AND         job.expires_at > now()
ORDER BY    job.priority, job.enqueued_at
LIMIT       2500
FOR UPDATE SKIP LOCKED;
```

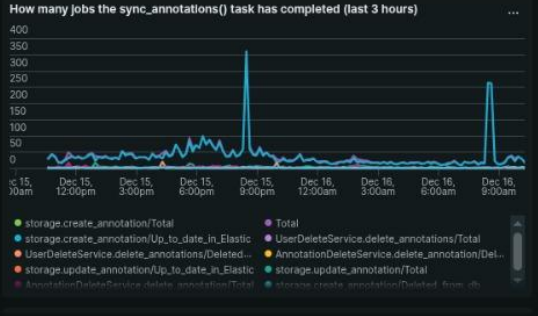
Job queue monitoring

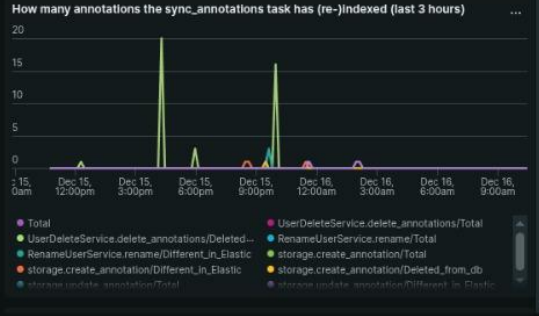

Queue Length (last 30 mins)
600
500
400
300
200
100
0
Dec 15, 12:00pm
Dec 15, 3:00pm
Dec 15, 6:00pm
Dec 15, 9:00pm
Dec 16, 12:00am
Dec 16, 3:00am
Dec 16, 6:00am
Dec 16, 9:00am
Total
Priority/1
Name/sync_annotation/Tag/storage.create_ann...
Name/sync_annotation/Total
Name/sync_annotation/Tag/UserDeleteService....
Name/sync_annotation/Tag/storage.update_an...
Name/sync_annotation/Tag/AnnotationDeleteS...
Priority/100


How many jobs the sync_annotations() task has completed (last 30 mins)
400
350
300
250
200
150
100
50
0
Dec 15, 12:00pm
Dec 15, 3:00pm
Dec 15, 6:00pm
Dec 15, 9:00pm
Dec 16, 12:00am
Dec 16, 3:00am
Dec 16, 6:00am
Dec 16, 9:00am
storage.create_annotation/Total
Total
storage.create_annotation/Up_to_date_in_Elastic
UserDeleteService.delete_annotations/Total
UserDeleteService.delete_annotations/Deleted...
AnnotationDeleteService.delete_annotation/Del...
storage.update_annotation/Up_to_date_in_Elastic
storage.update_annotation/Total


How many annotations the sync_annotations task has (re-)indexed (last 30 mins)
20
15
10
5
0
Dec 15, 12:00pm
Dec 15, 3:00pm
Dec 15, 6:00pm
Dec 15, 9:00pm
Dec 16, 12:00am
Dec 16, 3:00am
Dec 16, 6:00am
Dec 16, 9:00am
Total
UserDeleteService.delete_annotations/Total
UserDeleteService.delete_annotations/Deleted...
RenameUserService.rename/Total
RenameUserService.rename/Different_in_Elastic
storage.create_annotation/Total
storage.create_annotation/Different_in_Elastic
storage.create_annotation/Deleted_from_db


Queue Length (last 3 hrs)
600
500
400
300
200
100
0
Dec 15, 12:00pm
Dec 15, 3:00pm
Dec 15, 6:00pm
Dec 15, 9:00pm
Dec 16, 12:00am
Dec 16, 3:00am
Dec 16, 6:00am
Dec 16, 9:00am
Total
Priority/1
Name/sync_annotation/Tag/storage.create_ann...
Name/sync_annotation/Total
Name/sync_annotation/Tag/UserDeleteService....
Name/sync_annotation/Tag/storage.update_an...
Name/sync_annotation/Tag/AnnotationDeleteS...
Priority/100


How many jobs the sync_annotations() task has completed (last 3 hours)
400
350
300
250
200
150
100
50
0
Dec 15, 12:00pm
Dec 15, 3:00pm
Dec 15, 6:00pm
Dec 15, 9:00pm
Dec 16, 12:00am
Dec 16, 3:00am
Dec 16, 6:00am
Dec 16, 9:00am
storage.create_annotation/Total
Total
storage.create_annotation/Up_to_date_in_Elastic
UserDeleteService.delete_annotations/Total
UserDeleteService.delete_annotations/Deleted...
AnnotationDeleteService.delete_annotation/Del...
storage.update_annotation/Up_to_date_in_Elastic
storage.update_annotation/Total


How many annotations the sync_annotations task has (re-)indexed (last 3 hours)
20
15
10
5
0
Dec 15, 12:00pm
Dec 15, 3:00pm
Dec 15, 6:00pm
Dec 15, 9:00pm
Dec 16, 12:00am
Dec 16, 3:00am
Dec 16, 6:00am
Dec 16, 9:00am
Total
UserDeleteService.delete_annotations/Total
UserDeleteService.delete_annotations/Deleted...
RenameUserService.rename/Total
RenameUserService.rename/Different_in_Elastic
storage.create_annotation/Total
storage.create_annotation/Different_in_Elastic
storage.create_annotation/Deleted_from_db

Job queue alarms

Consumer stops	Job completion rate low-watermark alarm
Producer stops	Job completion rate low-watermark alarm
Consumer and producer both stop	Job completion rate low-watermark alarm
Producer outpaces consumer	Queue length high-watermark alarm
Stuck jobs	Queue length high-watermark alarm (eventually) Expired job alarm (eventually, even if just a single job sticks)

Re-indexing was now also possible

```
INSERT INTO job
  (name, scheduled_at, priority, ...)
SELECT
  'index',
  '2026-01-30 12:38:15.951648',
  100,
  'reindex_user',
  jsonb_build_object(
    'annotation_id', annotation.id
  )
FROM annotation
WHERE annotation.userid = :userid;
```

Re-index all annotations:

- From a given time period
- From a given user
- From a given group
- Of a given document
- ...

This was also used when renaming or deleting users, etc.

“Transactionally Staged Job Drain”

brandur.org/job-drain