**Mutation registry using lamindb**

Somewhere in my PhD, I needed to find all the genes that can be mutated in a pathway I was working with? The big, glittery public databases—COSMIC, ClinVar, dbSNP, each had more than 5 million mutations. Great for comprehensive coverage. Terrible when you need ONE specific gene and you're not a patient person (which I'm not).

But even if I managed to find all the mutations in my target gene, which was highly unlikely because querying them would take forever. I'd still need to figure out which ones actually affect the DNA repair pathway I care about.

And there it started: the never-ending literature search loop.

"Maybe I'll try Funcotator," I thought. But wait, first I'd need to set it up. Or maybe I could manually annotate genes from Reactome, KEGG, GO, each requiring their own workflow, their own format, their own special way of breaking my spirit.

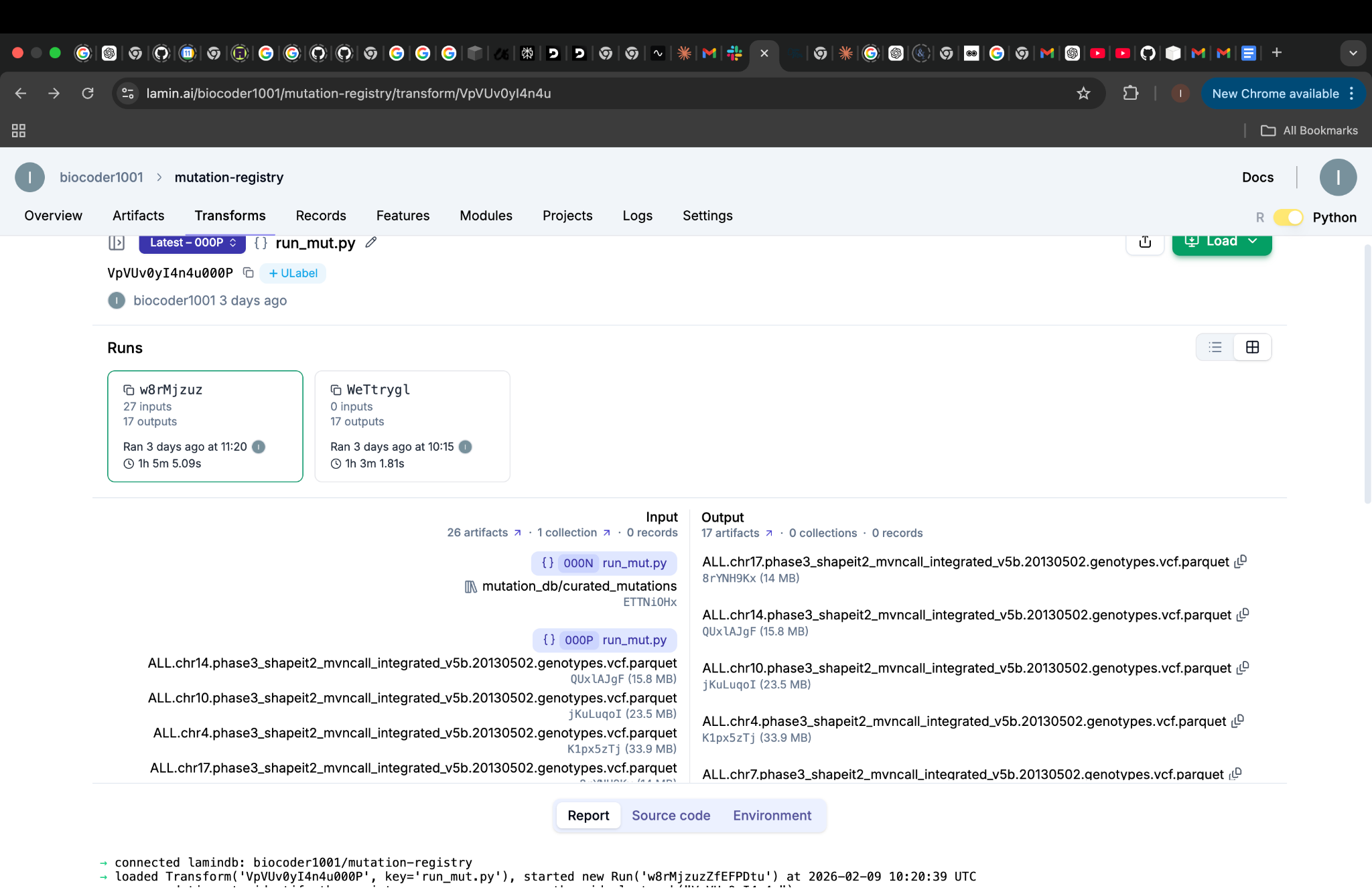
I sat there, staring at my terminal, and wondered: What if there's a simpler way? So I'm the kind of person who likes everything organized in one place. You know, something convenient. Something that doesn't make me choose between three annotation systems and five different file formats.

That's when I discovered LaminDB, a database management system specifically designed for biological data.

Think of it like this: instead of having notebooks scattered across your desk, your car, and three different coffee shops, you have one lab notebook that not only organizes everything but also remembers which experiment led to which result, which protocol you used, and even alerts you if something doesn't look right.

LaminDB does that for biological data. From genes to organisms to pathways to machine learning—everything tracked, traced, and queryable. So here was the idea of making a comprehensive database with all the information one needs about the genes and the mutants

Sidenote: Working with LaminDB is incredibly easy. My whole project became tracked and traced automatically. .Here as you can see in the image, we can track the source code, along with the inputs and the outputs for the script, that in my opinion makes it easier when we are dealing with the external scripts as well. I could instantly see which script made what change, and if that script imported functions from another script, that dependency was recorded too. It's like having a research assistant who actually pays attention.



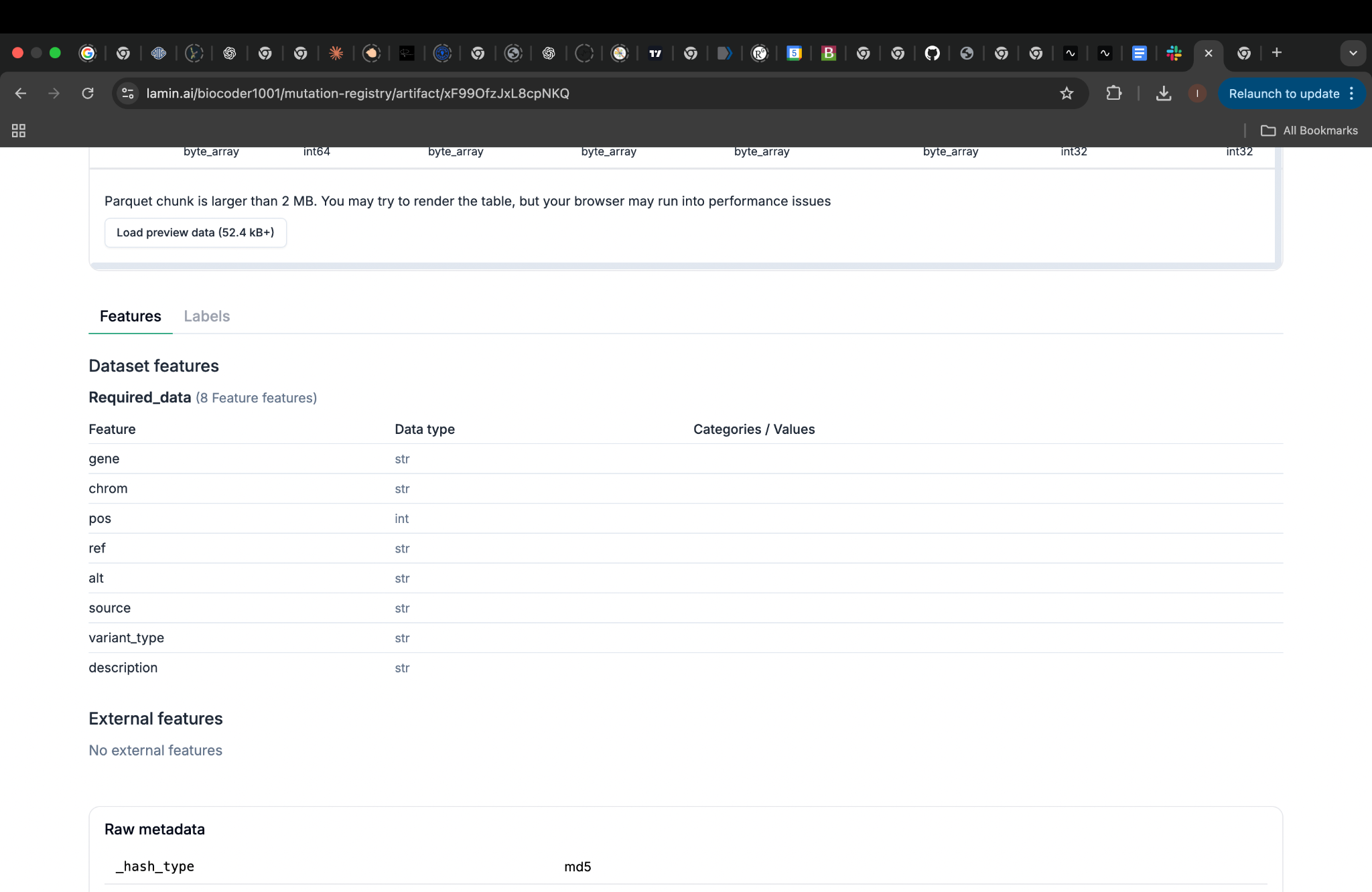
Here's what I did, step by step.

Link to the lamindb instance for accessing the files used and the collections: https://lamin.ai/biocoder1001/mutation-registry

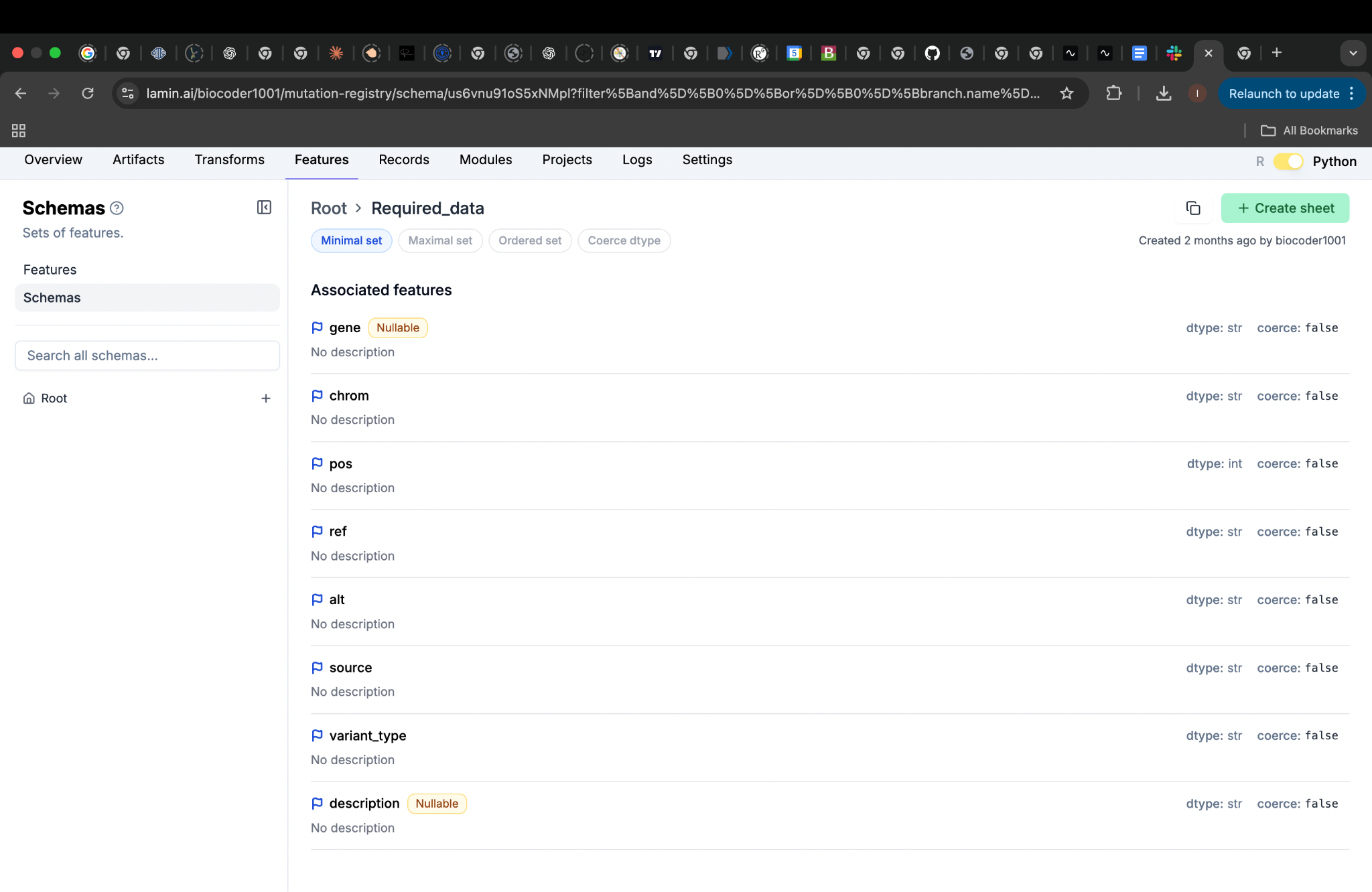
### **Step 1: Register the Public Databases**

I started by registering all the publicly available mutation databases with LaminDB—COSMIC, ClinVar, and dbSNP.

we can also look at the data type of the features of the artifact and the metadata information is also available, which includes the information like the key, and the description of the dataset we want to use or access from the collections.



One of my favorite features? Schema validation. I designed a schema that defined the bare minimum each mutation record should have. For me, that was: gene name and mutation type. If a record didn't have those, it wouldn't get in. Quality control from day one. As we can see here for the public vcfs, collected from various resources, they all had different fields, but some of the mandatory fields were used as schema.



### **Step 2: Create Multiple Collections**

After registering the databases, I built different collections based on what I'd need to query:

**Collection 1**: KEGG-annotated mutations Every mutation linked to KEGG pathway information. Want to know which mutations affect "DNA repair"? This collection has your answer.

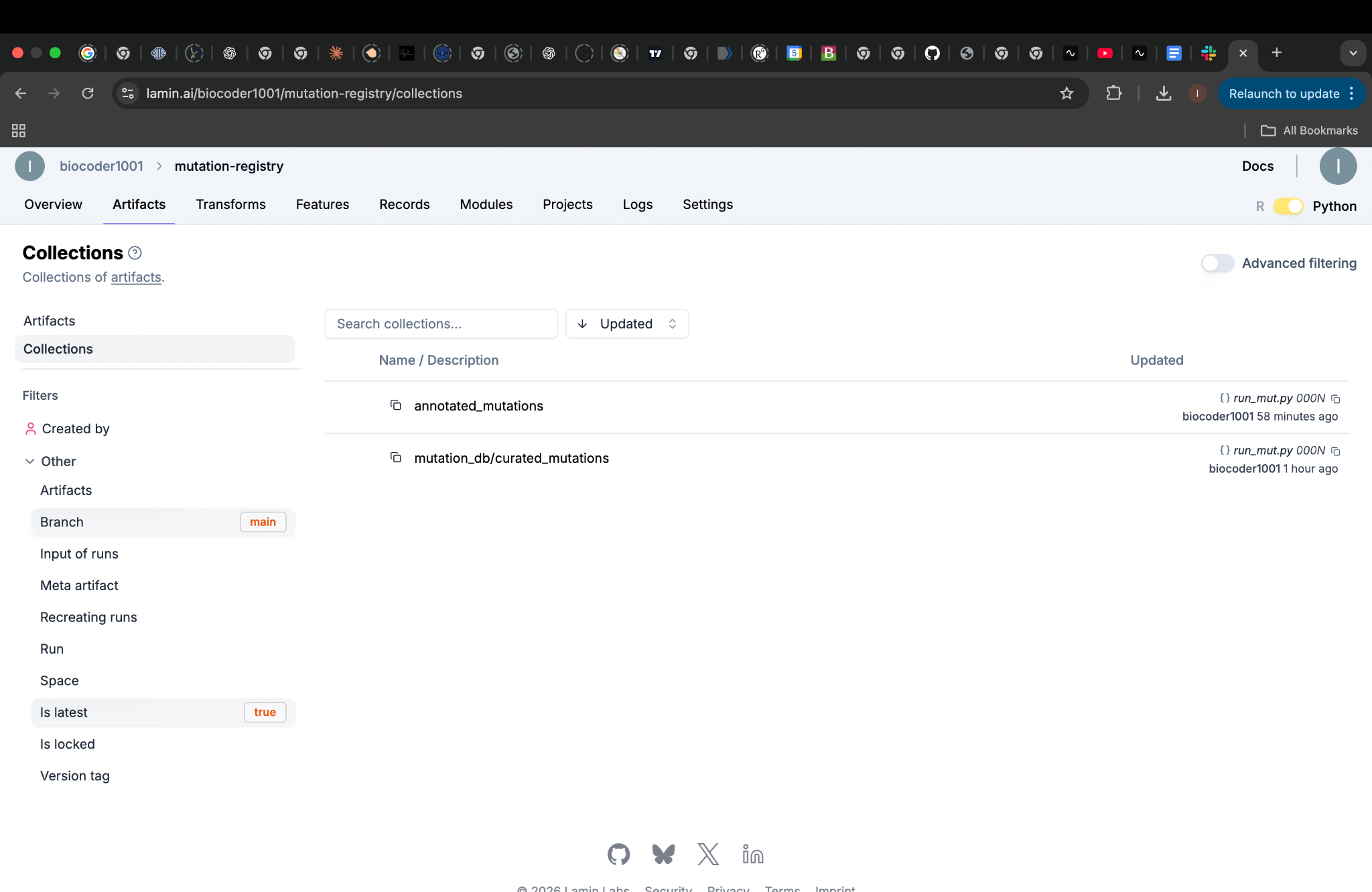
**Collection 2**: GO-annotated mutations Same mutations, but annotated with Gene Ontology biological processes. Different perspective, same genes.

**Collection 3**: Mutations by type Sorted by whether they're indels, SNPs, or point mutations. Different mutation types have different functional impacts, so being able to filter by type is crucial.

**Collection 4**: Artifact-filtered mutations This one filters out mutations from regions with high probability of sequencing artifacts. Why analyze noise when you can focus on real variants?

The beauty of this approach? I can query the same underlying data in multiple ways, depending on what I need at that moment.

Here is an example of how I have the curated mutations from the public vcfs and and annotated mutations in separate collections.

Add description

### **Step 3: Querying**

But what's the point of all this organization if querying still takes hours?

Here's where LaminDB really shines: Leveraging LaminDB’s data infrastructure and Python API, users can query and stream large-scale Parquet datasets.We don't have to worry about the different collection we just need what we want to query with like the gene, or the annotation(any particular pathway you are looking at), and there you go we will have the results out as a file.

Here we have four options, normally I would use the gene and the annotation I want together as shown in the screenshot, if we just use the gene name without mentioning the type of annotation by default we get go annotations.

