



**Preparations for the CMS 2017 Data Release**  
CMS Data Preservation and Open Access  
CERN Summer Student Programme 2024 – Project Report

Dana Alsharif, Mohammad Yahya Hamed  
Supervisor: Kati Lassila-Perini

August 2024  
Geneva, Switzerland

# Introduction

The Compact Muon Solenoid (CMS) is a general-purpose detector at the Large Hadron Collider (LHC) aimed at studying the Standard Model and searching for new physics like dark matter [1]. The CMS experiment is one of the largest international scientific collaborations in history, involving more than 6000 particle physicists, engineers, technicians, students, and support staff from 258 institutes in 59 countries [2].

The CMS Data Preservation and Open Access policy, approved by the CMS Collaboration Board in 2020, outlines the principles for preserving and re-using the vast and unique data generated by the CMS experiment. Recognizing the significant scientific value of this data, CMS is committed to preserving it at various levels of complexity and making it available for reuse by CMS collaborators, external researchers, educators, and scientists [3].

The CERN Open Data initiative provides the CERN Open Data Portal for the LHC experiments (eg. CMS), which allows them to make their data publicly available through the portal, along with the necessary documentation to understand and analyze it. This initiative aims to adopt a consistent approach toward the openness and preservation of experimental data, which promotes reproducibility and collaboration within the global scientific community [4].

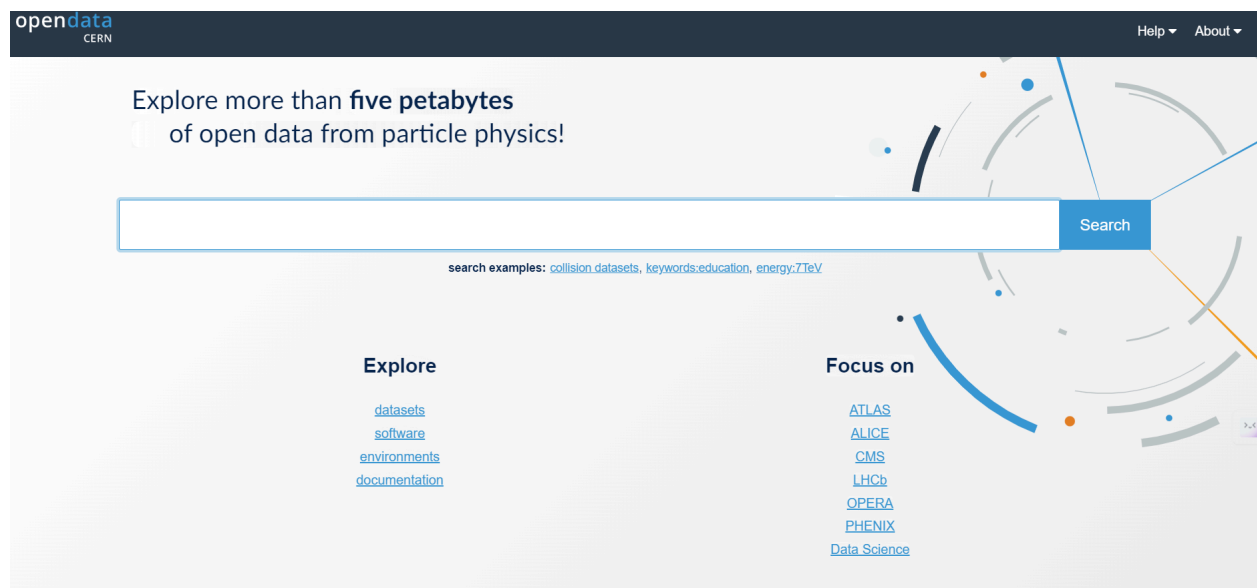


Fig 1: The CERN Open Data Portal

The Data Curation scripts are responsible for preparing the datasets' metadata, software, and testing guides to make new releases on the CERN Open Data Portal. The repository contains scripts for multiple experiments at CERN for different years, and each has its own directory with its accompanying material and scripts [5].

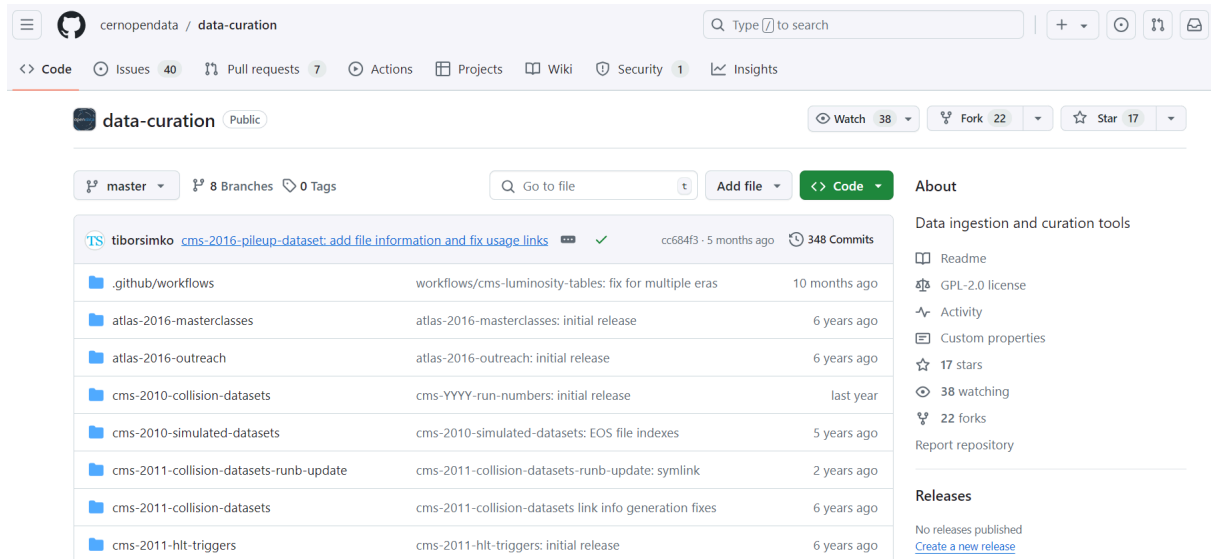


Fig 2: The Data Curation repository on GitHub

## Project Details

In our 8 weeks working at CERN, our tasks were mainly focused on the preparations for the CMS 2017 Data Release, which entailed modifying, refining, and optimizing scripts in the Data Curation repository under *cms-2017-simulated-datasets*, which includes the scripts necessary to generate simulated dataset records for the year 2017.

We also improved metadata accuracy for existing datasets and worked on enhancements for the CMS Open Data Release Guide, as well as the documentation of the 2017 Data Curation scripts.

Additionally, we assisted in the recovery of some lost datasets, particularly for Heavy Ion (HI) collision data and HI Monte Carlo (MC) data. This included querying and gathering information to facilitate the reprocessing and regeneration of the lost data.

## 1. CMS 2017 Data Release Preparations

The data curation scripts for the 2017 CMS simulated datasets prepare the records for the CMS 2017 open data release in a number of steps. The scripts handle tasks such as indexing files, creating configuration files, and the LHE generators step, which finds the generator parameters when the first step in the production chain produces an intermediate file in Les Houches event file format [6]. After setting up the necessary environment and configurations, the scripts process the data and generate the final JSON records, which are stored in the outputs directory.

### 1.1 Configurable Threading

We undertook the task of making threading configurable in the data processing scripts. This addition was implemented through a Command Line Interface (CLI) option `--threads`, enabling users to specify the number of threads to be utilized during execution. This is essential to provide flexibility in resource management and performance optimization based on available hardware. We established a default configuration of 32 threads for most data processing steps while setting a maximum limit of 100 threads per execution step. However, during the LHE generator step, we determined an optimal thread count of 20 and implemented a warning mechanism to notify users when they exceed the recommended threshold.

### 1.2 Efficiency Improvements

Two key efficiency enhancements were implemented:

#### a. Dataset Relationship Retrieval

The process of retrieving dataset relationships was streamlined by replacing the existing caching methodology for the *MiniAODSIM-NanoAODSIM* relationship map. A new mapping system was created at the start of the script chain, which significantly improved performance by providing quick access to frequently queried parent-child relationships, reducing the time required for each query.

#### b. LHE Generator Optimization

The runtime efficiency of the *LHE generator* script was improved by adding an error-handling method to the code. The updated code now interrupts execution immediately upon encountering a threading timeout or an error in accessing the *.tar* files rather than continuing to run unnecessarily, thereby reducing processing time.

### 1.3 Interface Integration via CLI Options

We integrated data processing scripts with the main interface through a unified CLI framework. This integration simplifies the execution process by enabling users to execute specific processing tasks directly from the interface script using the appropriate *--option* flags. This ensures that users can seamlessly follow the prescribed workflow and allows for enhanced usability and smoother testing and execution.

```
Options:
--create-eos-indexes / --no-create-eos-indexes
                                Create EOS rich index files [default: no-
                                create-eos-indexes]
--eos-dir TEXT
                                Output directory for the EOS file indexes
                                [default: ./inputs/eos-file-indexes/]
--ignore-eos-store / --no-ignore-eos-store
                                Presence of EOS file indexes [default: no-
                                ignore-eos-store]
--create-das-json-store / --no-create-das-json-store
                                Get DAS json information [default: no-
                                create-das-json-store]
--das-dir TEXT
                                Output directory for the DAS metadata
                                [default: ./inputs/das-json-store]
--create-mcm-store / --no-create-mcm-store
                                Get McM json information [default: no-
                                create-mcm-store]
--mcm-dir TEXT
                                Output directory for the DAS metadata
                                [default: ./inputs/mcm-store]
--get-conf-files / --no-get-conf-files
                                Get configuration files for the datasets
                                [default: no-get-conf-files]
--conf-dir TEXT
                                Output directory for the configuration files
                                [default: ./inputs/config-store]
--print-categorisation
                                Print results of categorisation
--print-results
                                Print results of categorisation with gen
                                info
--create-records
                                Create json file for records
--create-conffile-records
                                Create json file for conffiles
--recid-file PATH
                                File with record IDs [default:
                                ./inputs/recid.info.py]
--doi-file PATH
                                File with DOI information [default:
                                ./inputs/doi-sim.txt]
--threads INTEGER
                                Number of threads to use [default: 20]
--lhe-generators
                                Create LHE generators.
--create-parent-dicts
                                Create parent and child dictionary for nano
                                and mini datasets.
--parent-file PATH
                                File with parents and children datasets
                                [default: ./inputs/parent_dicts.py]
--help
                                Show this message and exit.
```

Fig 3: CLI options for the interface script

## 2. Documentation Enhancement and Ensuring Data Quality

### 2.1 Ensuring Metadata Quality

Adjustments were made to the metadata of CMS simulated datasets for the years 2015 and 2016. The cross-section field of several datasets showed a *total value* of zero, which was identified as unnecessary information. The datasets containing zero values were identified, and the cross-section field was filtered out from their metadata.

### 2.2 Refining Documentation

Significant improvements were made to the documentation associated with the CMS 2017 simulated datasets, the *cernopendata-client*, and the *CERN Open Data Release Guide*. Updates included adding examples to clarify concepts, thoroughly documenting newly added scripts and features, and refining content to ensure comprehensive descriptions. These enhancements focused on making the documentation clearer, more usable, and accessible.

Another example of using `--filter` would be retrieving the container image details. For instance, to get the container images registered in Docker Hub for a specific record, you can use:

```
$ cernopendata-client get-metadata --recid 22234 --output-value system_details.container_images.name  
docker.io/cmsopendata/cmssw_7_6_7-slc6_amd64_gcc493:latest  
gitlab-registry.cern.ch/cms-cloud/cmssw-docker-opendata/cmssw_7_6_7-slc6_amd64_gcc493:latest  
$ cernopendata-client get-metadata --recid 22234 --output-value system_details.container_images.name --filter registry=dockerhul  
docker.io/cmsopendata/cmssw_7_6_7-slc6_amd64_gcc493:latest
```

Fig 4: cernopendata-client documentation

### 2.3 Improving Dataset Directory Management

We implemented enhanced management practices for organizing dataset directories to improve efficiency. Previously, dataset records were stored individually in directories with numerous entries, leading to inefficiencies. The new method organizes RECIDs (Record IDs) into subdirectories based on a range of 1,000 records. For example, datasets with RECIDs ranging from 32,000 to 32,999 are now stored within a single directory labeled "32000". This restructuring reduces the number of entries per directory, leading to better organization and easier management.

### 3. Lost Datasets Recovery

To support the recovery of lost datasets, we provided critical information for the reprocessing of both HI collision data and HI MC data.

#### 3.1 HI Collision Data

We queried the parent datasets and recorded the runs for multiple HI collision datasets. Additionally, we retrieved the RAW data configurations and identified differences in configuration files. We then passed this information to the data processing team, enabling them to initiate reprocessing.

#### 3.2 HI MC Data

For HI MC data, we retrieved and validated *prepids* from the Data Aggregation System (DAS). In cases of lost *prepids*, we used queries from both DAS and ReqMgr to locate parent datasets and associated request names, facilitating the recovery process.

These efforts were crucial in restoring the datasets and ensuring their availability for future use.

## Acknowledgments

We would like to express our deepest gratitude for the continuous guidance of our supervisor, Kati Lassila-Perini, whose expertise, insightful feedback, and unwavering support were essential in completing this project. We extend our gratitude to CERN for giving us the opportunity to participate in the Summer Student program. Additionally, we would like to thank our university, Princess Sumaya University for Technology, and the Data Science department for their encouragement and support to be a part of this enriching experience.

## References

- [1] CERN, <https://home.cern/science/experiments/cms>, accessed 2024-08-08
- [2] The CERN Experimental Programme, <https://greybook.cern.ch/experiment/detail?id=CMS>, accessed 2024-08-08
- [3] CERN Open Data Portal, <https://opendata.cern.ch/record/415>, accessed 2024-08-08
- [4] CERN Open Science, <https://openscience.cern/open-data>, accessed 2024-08-08
- [5] CERN Open Data, Data Curation, <https://github.com/cernopendata/data-curation>, accessed 2024-08-08
- [6] Les Houches Event Files, <https://pythia.org/latest-manual/LHEF.html>, accessed 2024-08-15