The Dataverse® Project

HARVARD Dataverse

Intermediate Curation on the Dataverse Supported Repository

The Dataverse® Project

IQSS
The Institute for Quantitative Social Science

HARVARD LIBRARY

HARVARD UNIVERSITY
Information Technology

Sonia Maria Barbosa
Associate Director,
Dataverse support,
Data curation

Code of Conduct

The Dataverse® Project

Our Pledge

We as members, contributors, and leaders pledge to make participation in our community a harassment-free experience for everyone, regardless of age, body size, visible or invisible disability, ethnicity, sex characteristics, gender identity and expression, level of experience, education, socio-economic status, nationality, personal appearance, race, caste, color, religion, or sexual identity and orientation.

Our Standards

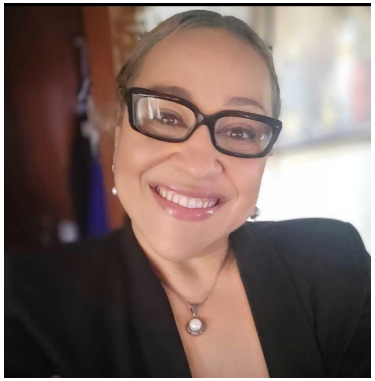
Examples of behavior that contributes to creating a positive environment include:

- Demonstrating empathy and kindness toward other people
- Being respectful of differing viewpoints and experiences
- Being respectful of differing opinions, viewpoints, and experiences
- Giving and gracefully accepting constructive feedback
- Accepting responsibility and apologizing to those affected by our mistakes, and learning from the experience
- Focusing on what is best not just for us as individuals, but for the overall community

Examples of unacceptable behavior by participants include:

- The use of sexualized language or imagery, and sexual attention or advances of any kind
- Trolling, insulting or derogatory comments, and personal or political attacks
- Public or private harassment
- Publishing others' private information, such as a physical or email address, without their explicit permission
- Other conduct which could reasonably be considered inappropriate in a professional setting

The Dataverse® Project



Sonia Barbosa

Associate Director
of Dataverse Support,
Data Curation, and
The Murray Archive

Agenda

Intermediate Data Curation

Review of Fundamentals:

- Recap of Data Curation Fundamentals

Curation Features and Tools on the Dataverse Platform

- Features/Tools: Collections, Datasets, Files
- Use Cases and Examples
- Q&A Session

The Dataverse® Project

Review of Fundamentals in Data Curation

The Dataverse® Project

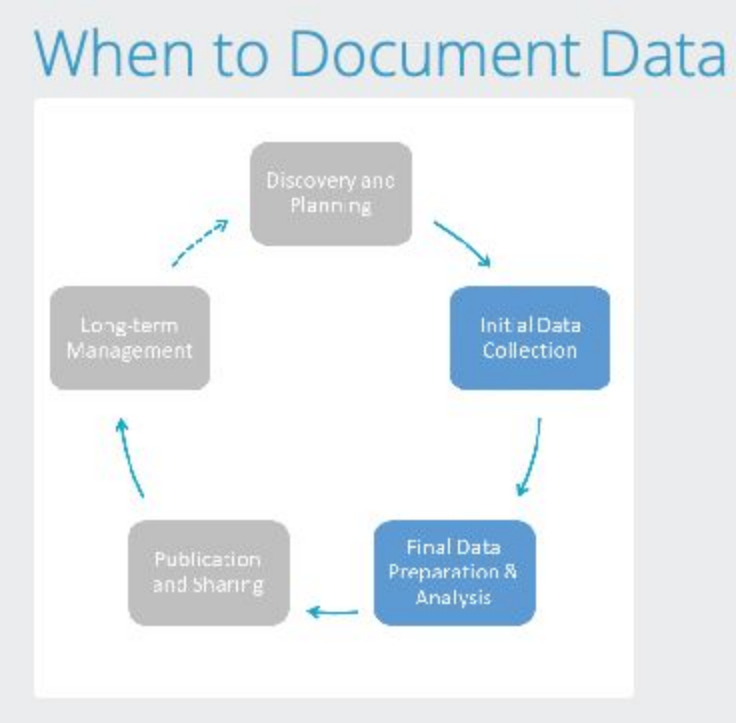
Introduction to Data Curation

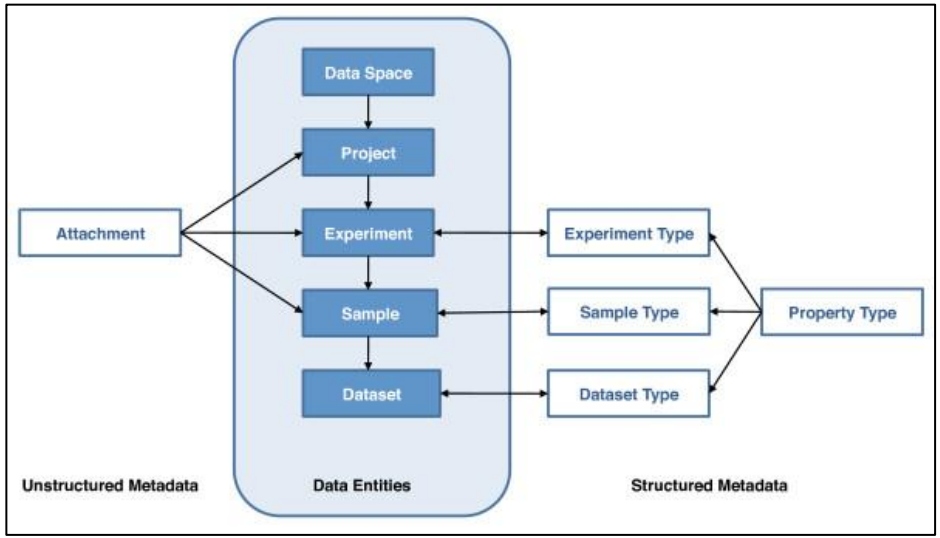
The Importance of Curation:

- Enhance Data Quality
- Efficient Organization
- Advanced Analysis
- Collaborative Workflows
- Ethical Compliance
- Future-proofing Data

Basic Data Curation Principles:

- Data Identification
- Data Collection
- Data Cleaning
- Data Organization
- Data Storage
- Metadata Management
- Data Preservation
- Data Sharing
- Documentation
- Compliance and Ethics





OpenBIS: A flexible framework for managing and analyzing complex data in biology research - Scientific Figure on ResearchGate. Available from: https://www.researchgate.net/figure/Data-organization-and-metadata-Data-are-organized-using-entities-and-relations-that-are_fig7_51861855 [accessed 3 Dec, 2023]

The Dataverse® Project

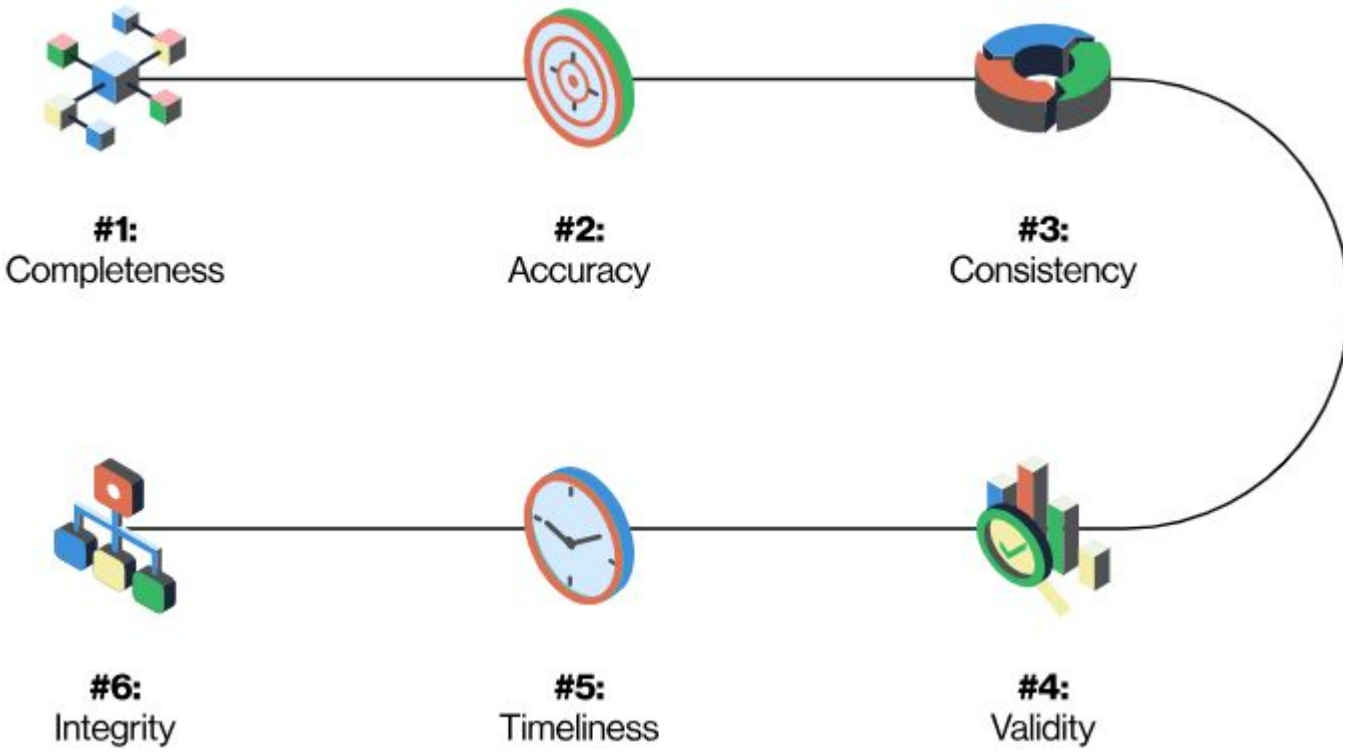
Understanding Data Quality:

Assessing Data Quality, Metrics:

- Data Accuracy
- Data Completeness
- Data Consistency
- Data Validity
- Timeliness
- Relevance
- Usability
- Duplication
- Privacy/Security

Data Cleaning Techniques:

- Handle Missing Values
- Detect outliers and Correct
- Normalization and Standardization
- Deduplication of Data
- Parsing and Formatting
- Error Correction
- Handle Inconsistencies



https://www.cloverdx.com/blog/6-data-quality-metrics-you-cant-ignore

a. Employees

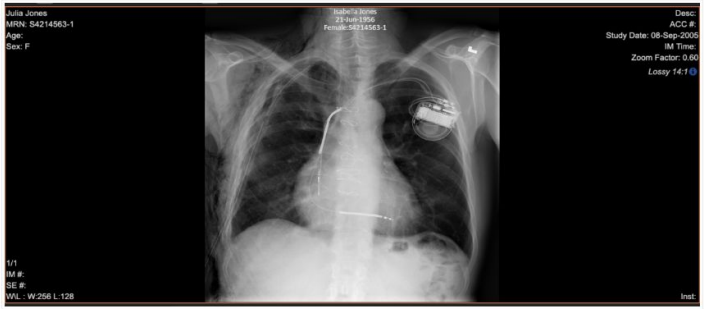
name	*age*	*salary*
Paul	1978	NULL
Paul	NULL	29,000
Paul	1979	NULL
Melanie	1990	NULL
Bob	NULL	37,000
Bob	1977	NULL
Charlie	1978	32,000

b. Employees

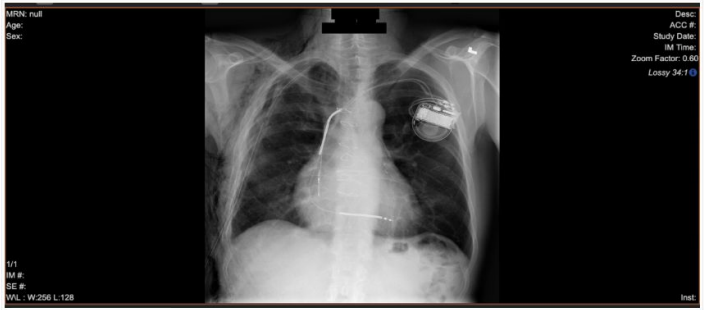
name	*age*	*salary*
Paul	?	29,000
Melanie	1990	NULL
Bob	1977	37,000
Charlie	1978	32,000

Paul.age = 1978 OR 1979 ?

[Figure 3](#) from "++Spicy: an Open-Source Tool for Second-Generation Schema Mapping and Data Exchange".



The second image shows the same x-ray image with all sample PII and PHI metadata removed or obscured.



“After de-identification, all image metadata is removed, and all text burned into the image is obscured with an opaque rectangle. This configuration of de-identification is useful for when you need only the image pixel data for further analysis, machine learning (ML) model training, or inference.”

https://cloud.google.com/architecture/de-identification-of-medical-images-through-the-cloud-healthcare-api

Data Organization:

Taxonomies and Ontologies:

Taxonomies

Helpful in organizing data (organisms) into hierarchical structures, easing navigation and retrieval of data

Example:
Field - Biology
- Species Taxonomy:
  - Kingdoms
  - Phyla
  - Classes
  - Orders
  - Families
  - Genera
  - Species

Ontologies

Enable a complex representation of knowledge. Allows inferences and reasoning with a domain

Example:
Field: Medical Field

Diabetes Mellitus Diagnosis Ontology (DDO) is an ontology for diagnosis of diabetes, containing diabetes-related complications, symptoms, drugs, lab tests, etc. The Adverse Event Reporting Ontology (AERO) is aimed at increasing quality and accuracy of reported adverse clinical events

Taxonomies and Ontologies

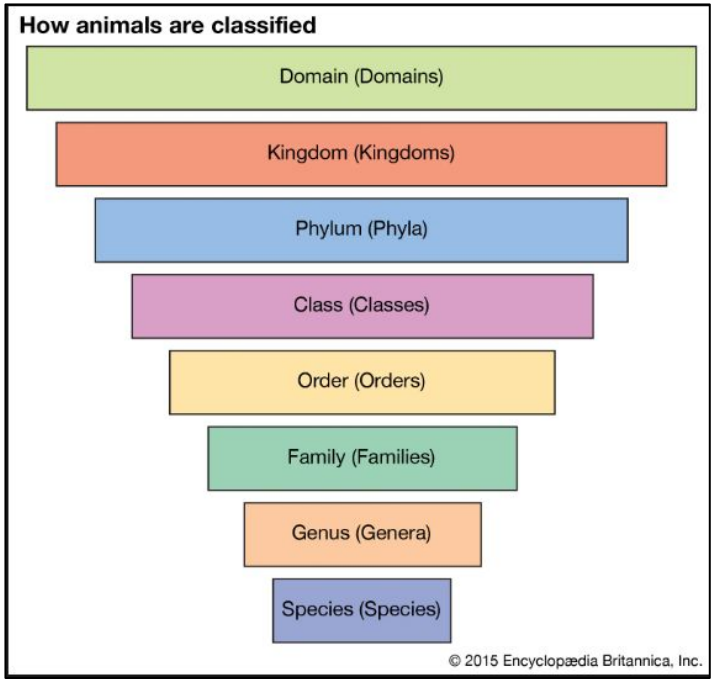


Table 1 | **Some principal biological ontologies and other web sites**

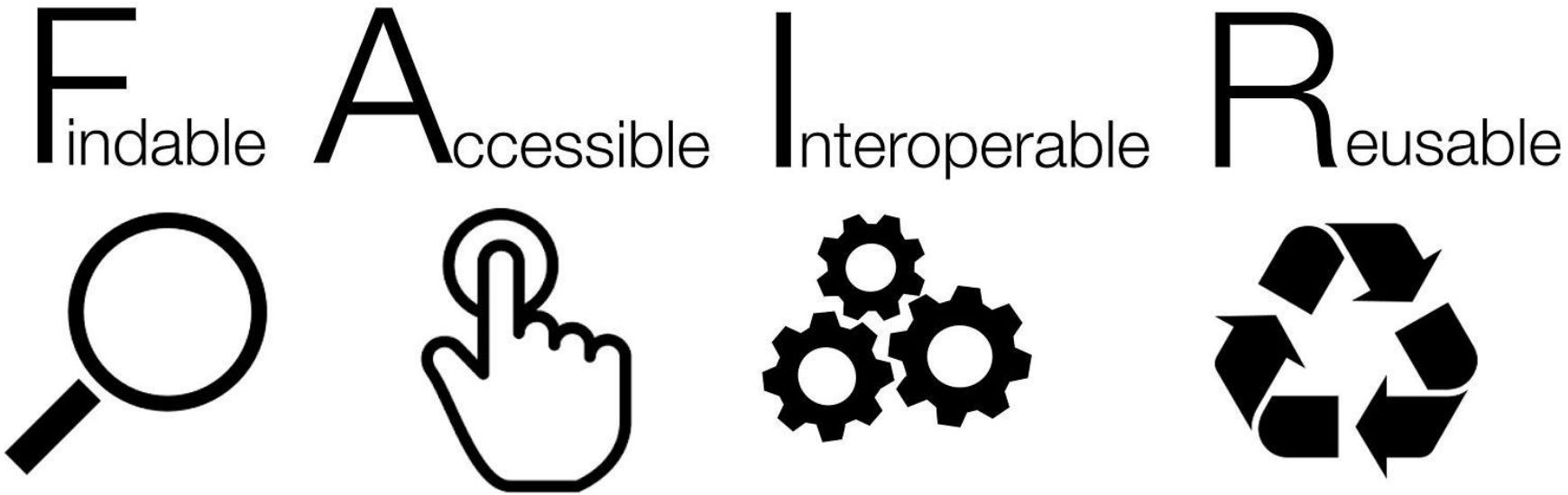
Web site name	URL	Function
AmiGO	www.godatabase.org/cgi-bin/go.cgi	Web application for browsing and searching gene ontology and gene associations
Cell Ontology (OBO)	obo.sourceforge.net/list.shtml	β-version containing >600 cell types
Directed Acyclic Graph (DAG)	www.nist.gov/dads/HTML/directAcycGraph.html	Provides definitions for *DAG*
DAG-edit	sourceforge.net/project/showfiles.php?group_id=36855	An ontology editor: a program for creating, editing and visualizing ontologies
Common Ontology Browser for Anatomy — CObrA (XSPAN)	www.xspan.org/applications/cobra/	Ontology editor that allows links to be made between ontologies
A Cross-Species Anatomy Network (XSPAN)	www.xspan.org	Project for linking anatomies of model organisms
Digital Anatomist[32]	depts.washington.edu/ventures/pfolio/fma.htm	Complex and rich ontology of human anatomy
Edinburgh Mouse Atlas Project (EMAP)	genex.hgu.mrc.ac.uk	Graphical database of mouse gene expression
Galen[31]	www.opengalen.org	A management architecture for clinical information that includes an ontology for human anatomy
Genecensus[17]	bioinfo.mbb.yale.edu/genome	Provides access to pathway-analysis tools
Gene Ontology (GO) Consortium[30]	www.geneontology.org	Resource for molecular function, biological process and cellular component ontologies
Gene Expression Database (GXD)[33]	www.informatics.jax.org/searches/expression_form.shtml	Mouse gene-expression database
GO-TermFinder	search.cpan.org/~sherlock/GO-TermFinder-0.5	Programs to facilitate analysis of GO annotations
Human Developmental Anatomy	genex.hgu.mrc.ac.uk/Databases/HumanAnatomy	Human anatomy of Carnegie stages* 1–20
Interoperable Informatics Infrastructure Consortium (I3C)	www.i3c.org	Provides standards for interoperability in bioinformatics
MetaCyc[13]	metacyc.org	Database of metabolic pathways
Mouse Developmental Anatomy	genex.hgu.mrc.ac.uk/Databases/Anatomy	Anatomy of Theiler stages‡ 1–26
Mouse Phenotype Ontology (MGI)	www.informatics.jax.org/searches/Phat.cgi?id=MP:0000001	Terms for describing mutant mice
Open Biological Ontologies (OBO)	obo.sourceforge.net	Umbrella web site for open bio-ontology projects
Ontology Markup Language (OML)	xml.coverpages.org/oml9808.html	XML§ markup language for ontologies
OntoExpress[18]	vortex.cs.wayne.edu/projects.htm	Tools for exploring microarray data
Ontology Web Language (OWL)	www.w3.org/TR/2003/PR-owl-ref-20031215	Specification of a semantic markup language to formalize ontologies on the web
Pathbase[12]	www.pathbase.net	Database of mouse pathology mages
Pharmacogenetics and Pharmacogenomics Knowledge Base (PharmGKB)[14]	pharmgkb.org/index.jsp	Resource on how genetic variability links to variability in drug responses
Plant Ontology (PO) Consortium	plantontology.org	Resource for anatomy and developmental stages ontologies for flowering plants
Protégé[34]	protege.stanford.edu/index.html	Program for making ontology frames
PubMed	www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=PubMed	Database of the biomedical literature
Renamed ABox and Concept Expression Reasoner (RACER)	www.sts.tu-harburg.de/~r.f.moeller/racer	An ontology reasoner‖
Resource Description Framework Specification (RDFS)	www.w3.org/RDF	Ontology language for the exchange of knowledge on the web using XML and URI technologies¶
Semantic Web	www.w3.org/2001/sw	RDF-based representation of data on the web
Unified Medical Language System (UMLS)	www.nlm.nih.gov/research/umls	Project for handling medical concepts
WordNet	www.cogsci.princeton.edu/~wn	Lexical database of the English language

https://www.arabidopsis.org/about/nrg1295.pdf

The Dataverse® Project

Curation Features and Tools on the Dataverse Platform

FAIR Principles and Dataverse



https://en.wikipedia.org/wiki/File:FAIR_data_principles.jpg

Collection-level

Current Features

Dataverse Collections

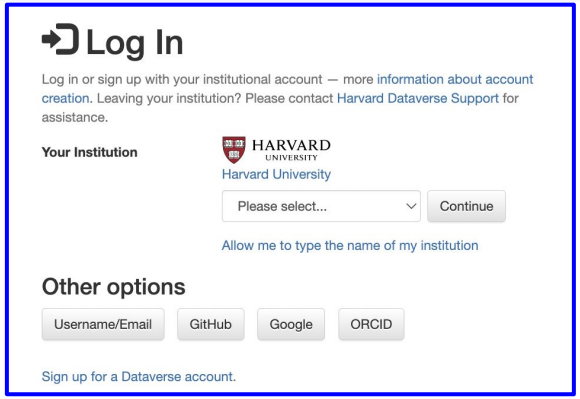
- Own administration
- Own branding (and can be embedded anywhere)

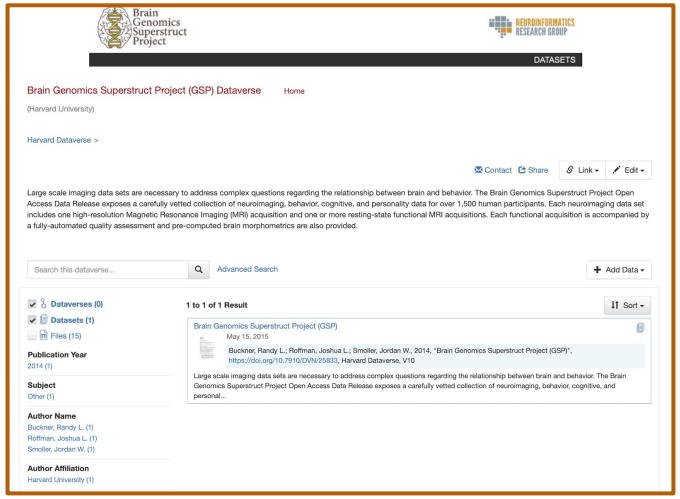
Datasets

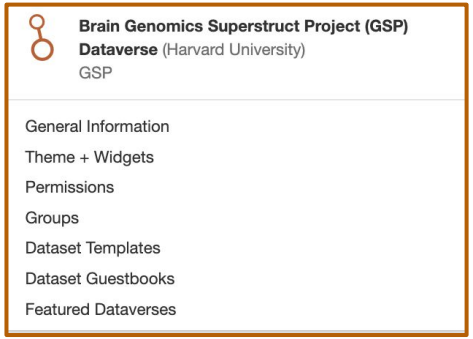
- Citation
- Metadata
- Versioning
- Private URL
- Custom Terms/Multiple License/Permissions
- Guestbooks
- Publishing Workflows

Files

- Citation
- Ingest
- Preview/Explore
- Metadata
- Versioning
- Permissions/Embargo/Restrictions







Dataset-level

Current Features

Dataverse Collections

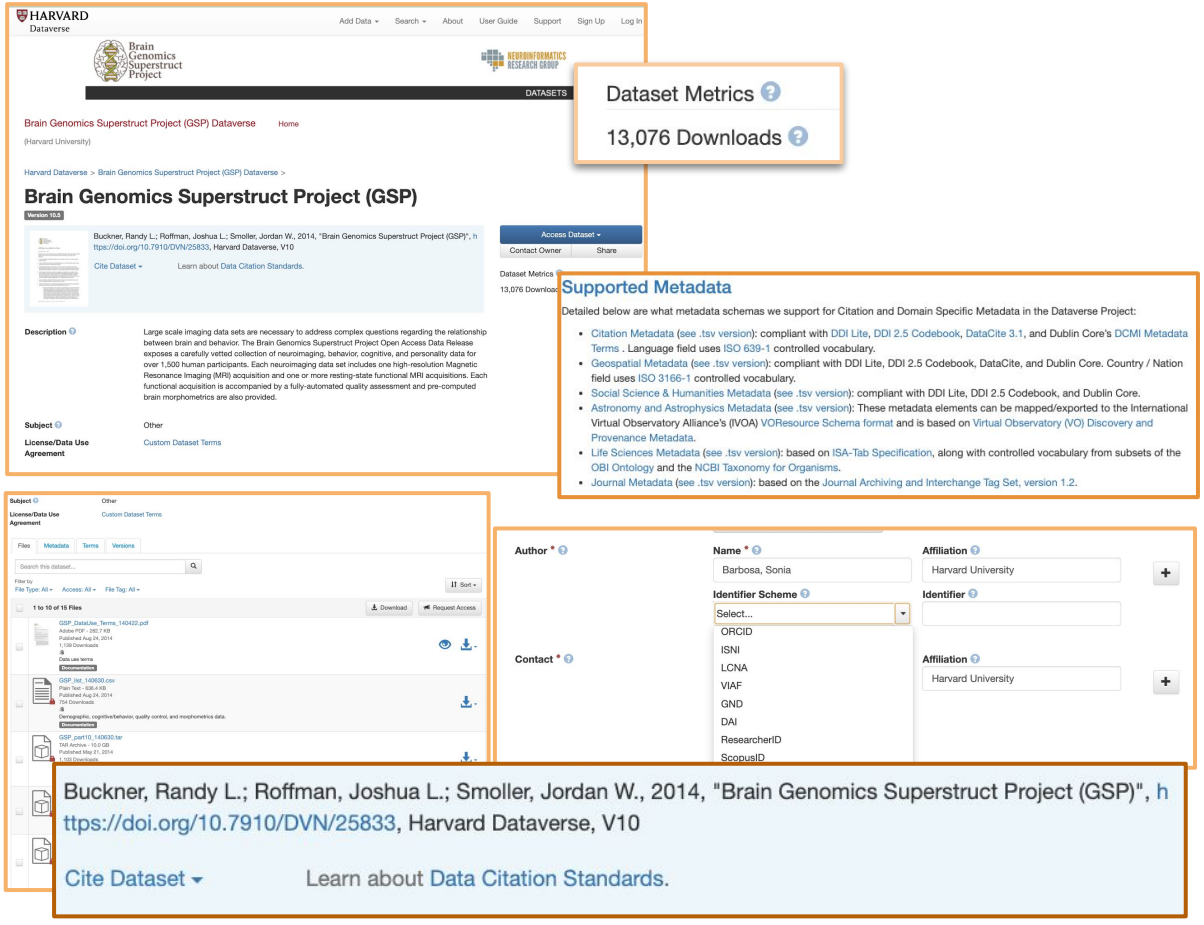
- Own administration
- Own branding (and can be embedded anywhere)

Datasets

- Citation
- Metadata
- Versioning
- Private URL
- Custom Terms/Multiple License/Permissions
- Guestbooks
- Publishing Workflows

Files

- Citation
- Ingest
- Preview/Explore
- Metadata
- Versioning
- Permissions/Embargo/Restrictions



File-level

Current Features

Dataverse Collections

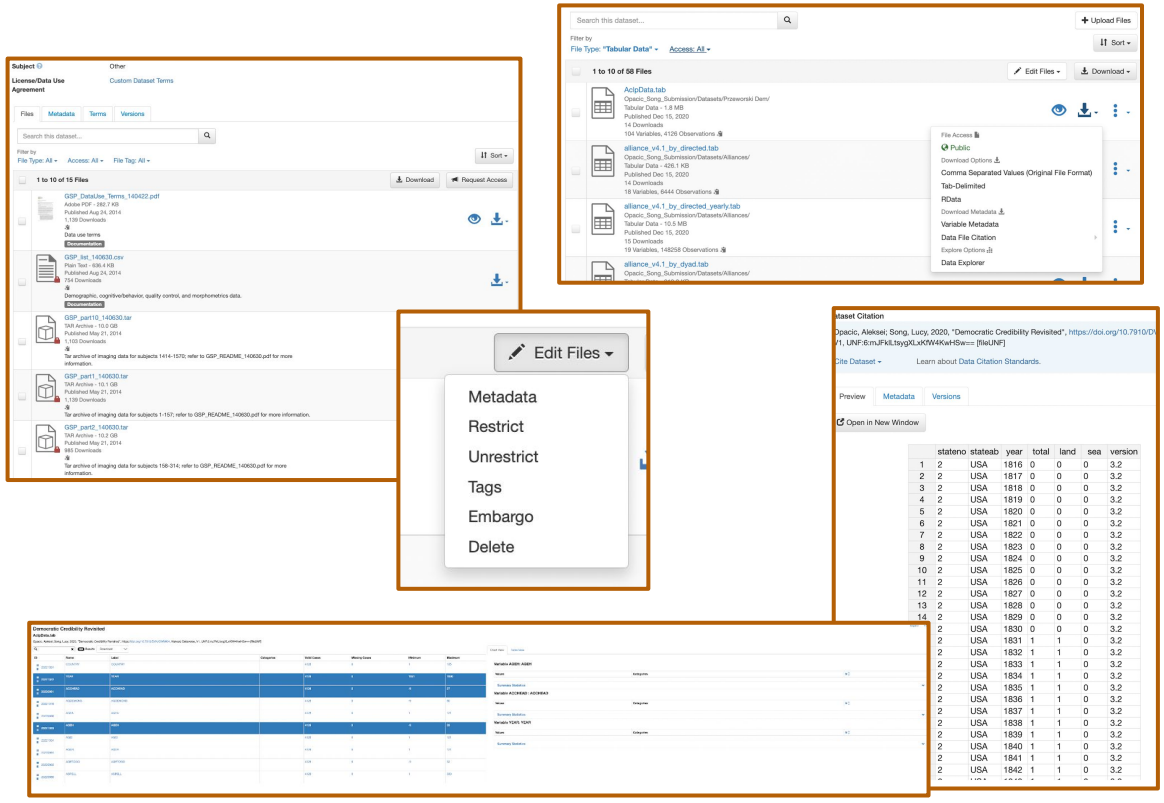
- Own administration
- Own branding (and can be embedded anywhere)

Datasets

- Citation
- Metadata
- Versioning
- Private URL/*Anonymous Peer Review
- Custom Terms/*Multiple Licenses/Permissions
- Guestbooks
- Publishing Workflows

Files

- Citation
- Ingest
- Preview/Explore
- Metadata/Provenance
- Versioning
- Permissions/Embargo/Restrictions

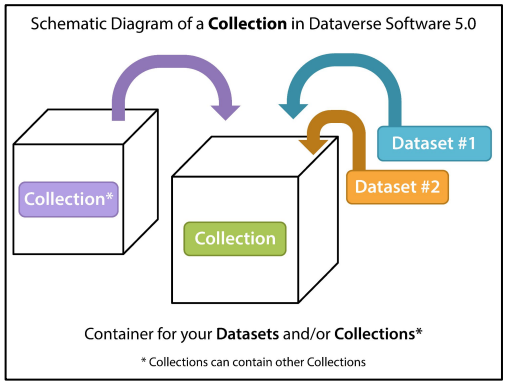


Review of Basics in Dataverse Curation

Best practices:

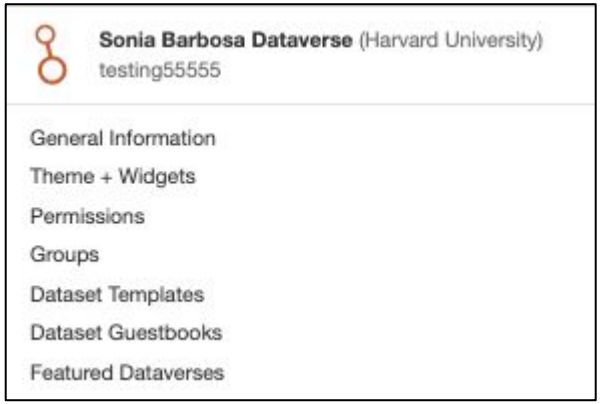
- Request a COLLECTION
- Deposit DATASETS within COLLECTION
- Upload FILES (data, code, documentation) to DATASETS

The Dataverse® Project



Best practices for “COLLECTION” creation:

- Collection name
- Identifier
- Category
- Collection level contact
- Collection Description
- Metadata selection
  - Custom metadata?
- Facet selections

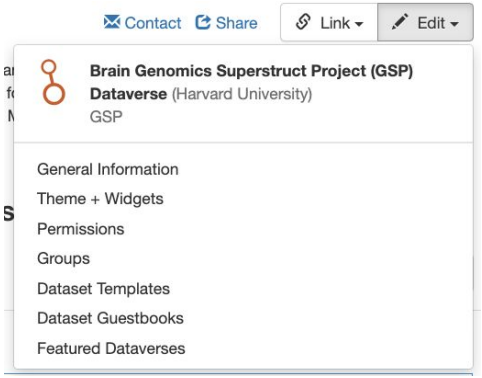


Intermediate options in:

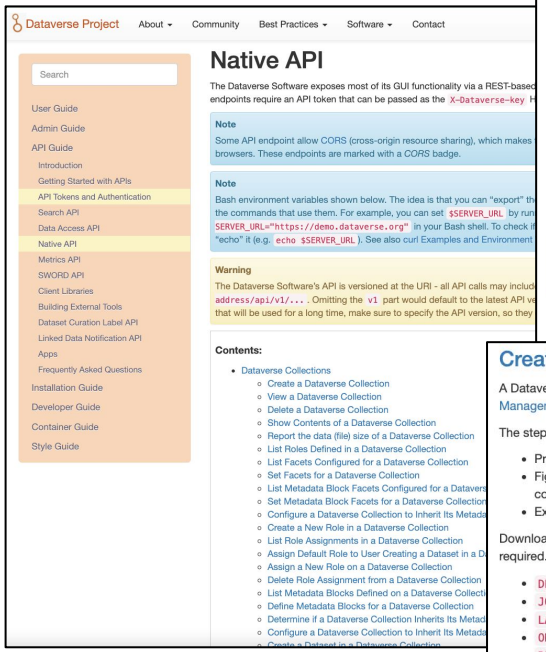
[Collections]

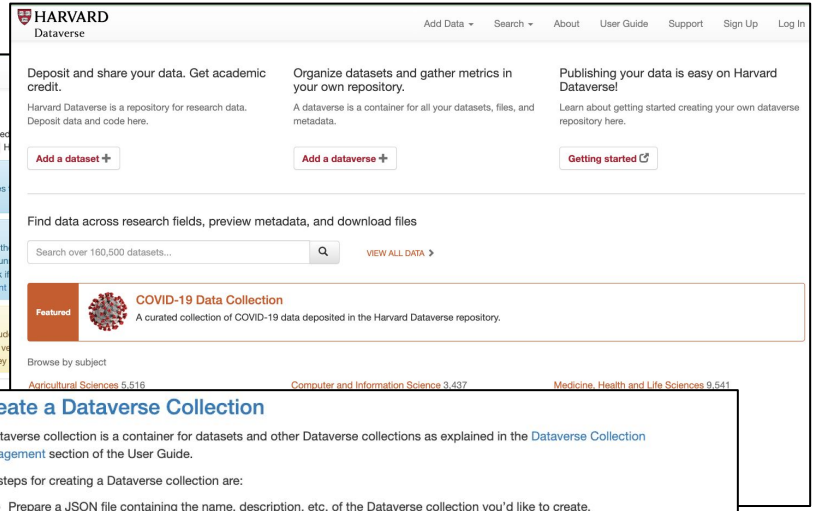
While some of this is required and can’t be skipped, not enabling or using these other options results in underutilization of the features available to you

The Dataverse® Project



Example collection edit options from [Brain Genomics](#) Collection, Harvard Dataverse





Create a Dataverse Collection

A Dataverse collection is a container for datasets and other Dataverse collections as explained in the Dataverse Collection Management section of the User Guide.

The steps for creating a Dataverse collection are:

- Prepare a JSON file containing the name, description, etc, of the Dataverse collection you'd like to create.
- Figure out the alias or database id of the "parent" Dataverse collection into which you will be creating your new Dataverse collection.
- Execute a curl command or equivalent.

Download `dataverse-complete.json` file and modify it to suit your needs. The fields `name`, `alias`, and `dataverseContacts` are required. The controlled vocabulary for `dataverseType` is the following:

- `DEPARTMENT`
- `JOURNALS`
- `LABORATORY`
- `ORGANIZATIONS_INSTITUTIONS`
- `RESEARCHERS`
- `RESEARCH_GROUP`
- `RESEARCH_PROJECTS`
- `TEACHING_COURSES`
- `UNCATEGORIZED`

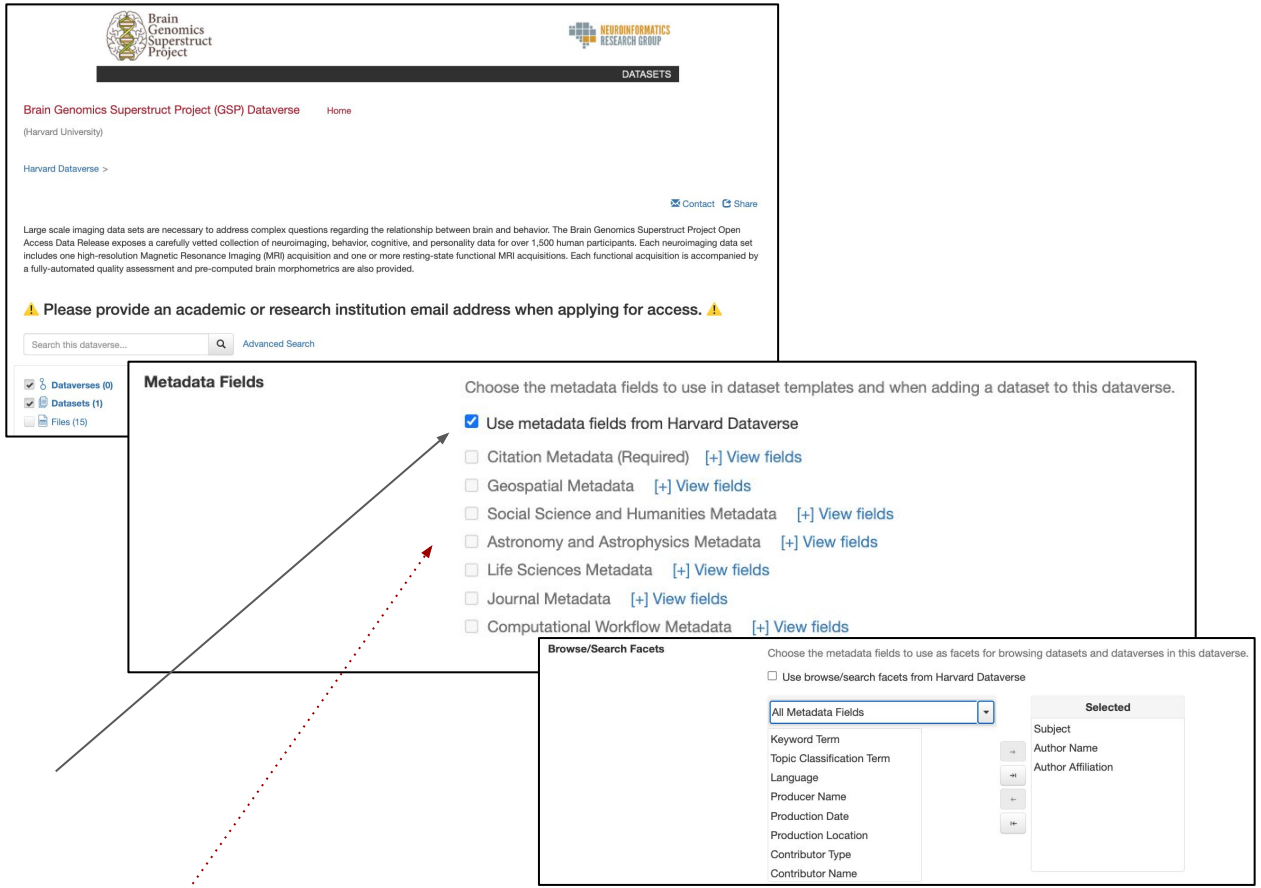
```
{
  "name": "Scientific Research",
  "alias": "science",
  "dataverseContacts": [
    {
      "contactEmail": "pi@example.edu"
    },
    {
      "contactEmail": "student@example.edu"
    }
  ],
  "affiliation": "Scientific Research University",
  "description": "We do all the science.",
  "dataverseType": "LABORATORY"
}
```

Intermediate Options in:

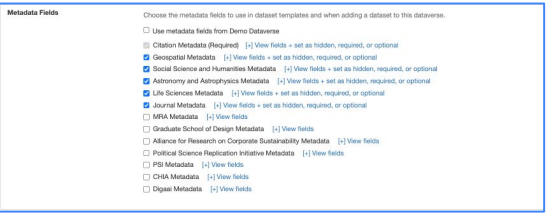
Collections

Intermediate Options in:

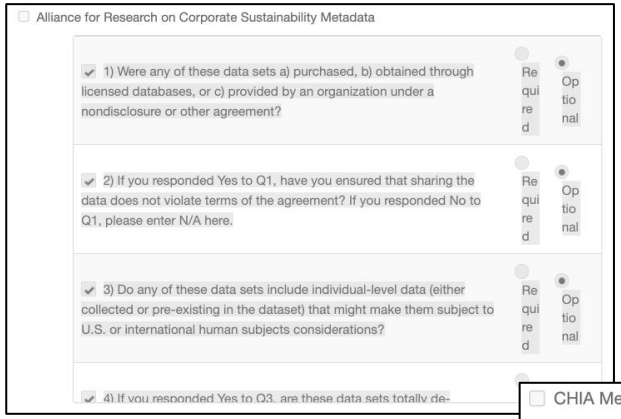
[Collections]

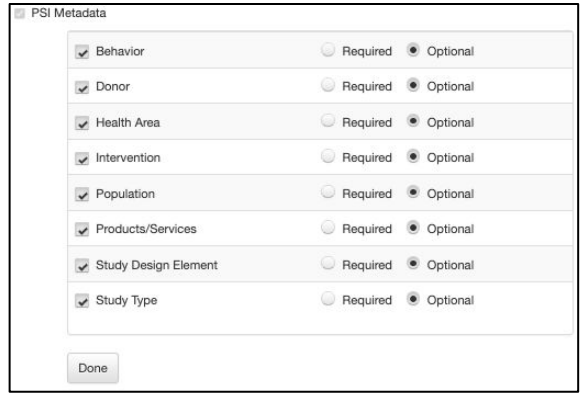


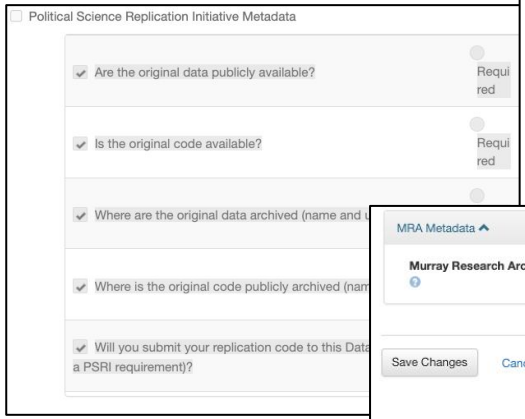
The Dataverse® Project

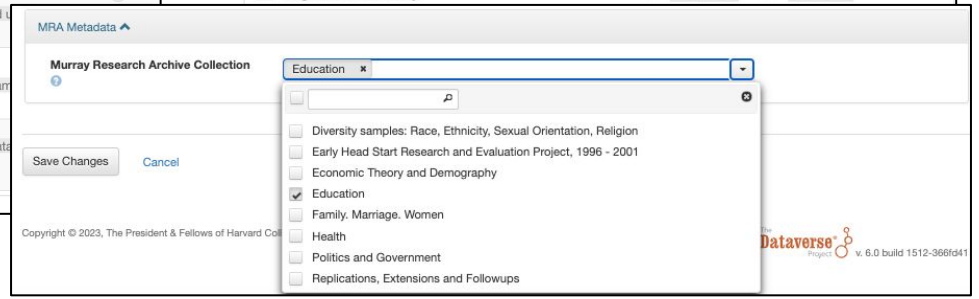


(As seen in repository)







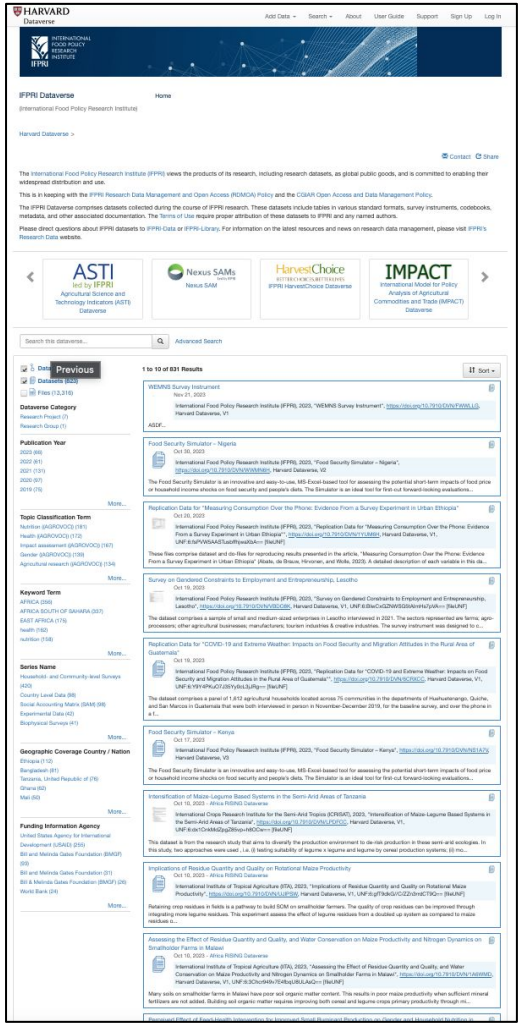


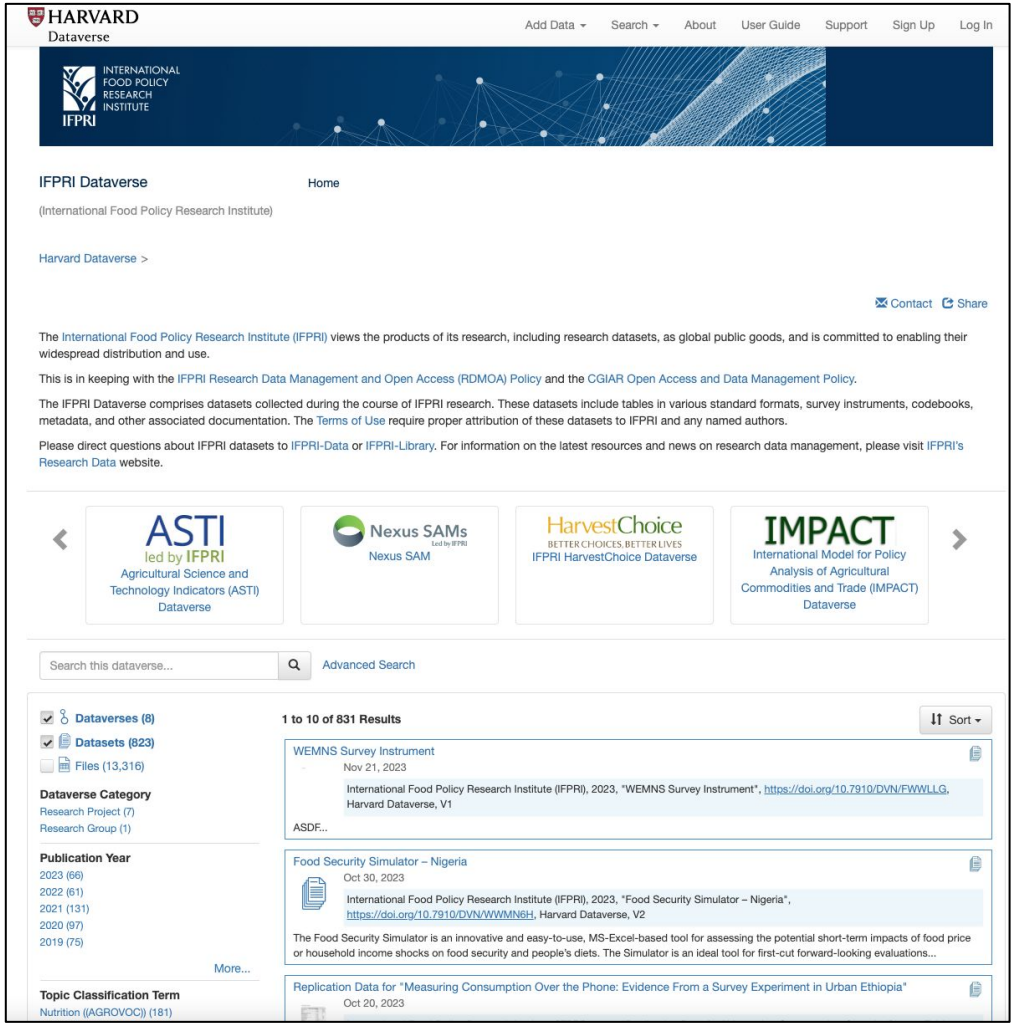
Intermediate Options in:

[Collections]

Intermediate Options in:

[Collections]





Intermediate Options in:

Datasets

Best practices for “Dataset” creation:

- Gather all data and documentation to be shared
- Know what license you will use
- **Data are deidentified**
- All pre sharing requirements have been fulfilled (see data quality above)
- Determine what metadata you will populate to describe the dataset, some are required by default

While some of this is required and can’t be skipped, not enabling or using these other options results in underutilization of the features available to you

The Dataverse® Project

https://guides.dataverse.org/en/latest/api/native-api.html#create-a-dataset-in-a-dataverse-collection

Create a Dataset in a Dataverse Collection

A dataset is a container for files as explained in the Dataset + File Management section of the User Guide.

To create a dataset, you must supply a JSON file that contains at least the following required metadata fields:

- Title
- Author Name
- Point of Contact Email
- Description Text
- Subject

Submit Incomplete Dataset

Note: This feature requires dataverse.api.allow-incomplete-metadata to be enabled and your S
`datasetValid` field.

Providing a `.../datasets?doNotValidate=true` query parameter turns off the validation of
Name" is required. For example, a minimal JSON file would look like this:

```
{
  "datasetVersion": {
    "metadataBlocks": {
      "citation": {
        "fields": [
          {
            "value": [
              {
                "authorName": {
                  "value": "Finch, Fiona",
                  "typeClass": "primitive",
                  "multiple": false,
                  "typeName": "authorName"
                }
              }
            ],
            "typeClass": "compound",
            "multiple": true,
            "typeName": "author"
          }
        ],
        "displayName": "Citation Metadata"
      }
    }
  }
}
```

The following is an example HTTP call with deactivated validation:

The following is an example HTTP call with deactivated validation:

```
export API_TOKEN=xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
export PARENT=root
export SERVER_URL=https://demo.dataverse.org

curl -H "X-Dataverse-key:$API_TOKEN" -X POST "$SERVER_URL/api/dataverses/$PARENT/datasets?doNotValidate=true" --upload-file dataset-incomplete.json -H 'Content-type:application/json'
```

Note: You may learn about an instance's support for deposition of incomplete datasets via Show Support Of Incomplete Metadata Deposition.

Submit Dataset

As a starting point, you can download `dataset-finch1.json` and modify it to meet your needs. (`dataset-finch1_fr.json` is a variant of this file that includes setting the metadata language (see :MetadataLanguages) to French (fr). In addition to this minimal example, you can download `dataset-create-new-all-default-fields.json` which populates all of the metadata fields that ship with a Dataverse installation.)

The curl command below assumes you have kept the name "dataset-finch1.json" and that this file is in your current working directory.

Next you need to figure out the alias or database id of the "parent" Dataverse collection into which you will be creating your new dataset. Out of the box the top level Dataverse collection has an alias of "root" and a database id of "1" but your installation may vary. The easiest way to determine the alias of your root Dataverse collection is to click "Advanced Search" and look at the URL. You may also choose a parent Dataverse collection under the root Dataverse collection.

Note

See curl Examples and Environment Variables if you are unfamiliar with the use of `export` below.

```
export API_TOKEN=xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
export PARENT=root
export SERVER_URL=https://demo.dataverse.org

curl -H "X-Dataverse-key:$API_TOKEN" -X POST "$SERVER_URL/api/dataverses/$PARENT/datasets" --upload-file dataset-finch1.json -H 'Content-type:application/json'
```

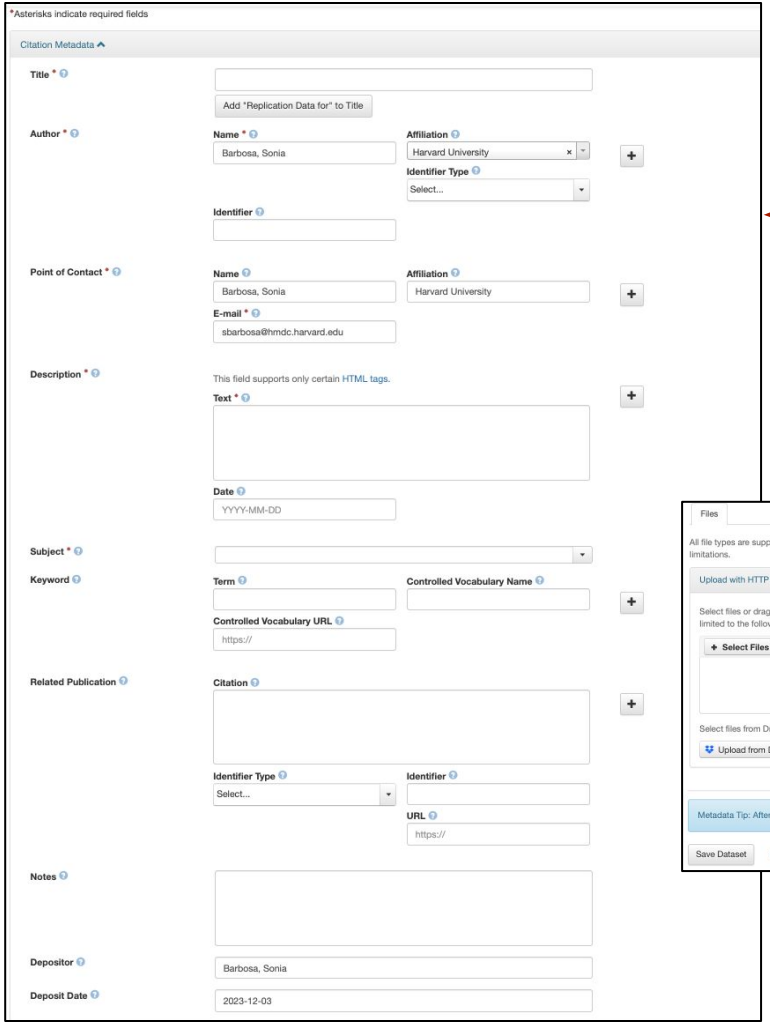
Intermediate Options in:

Datasets

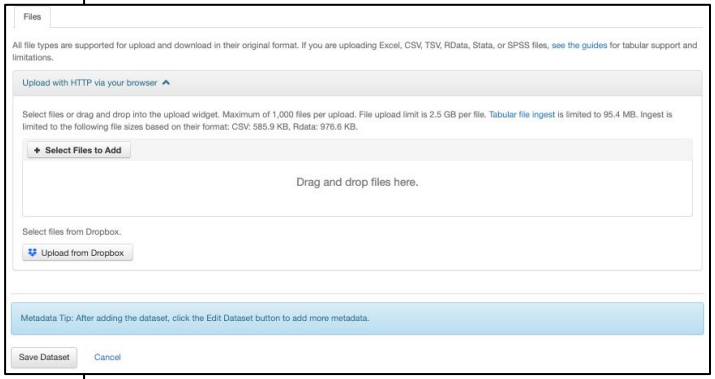
The Dataverse® Project

Intermediate Options in:

[Datasets]



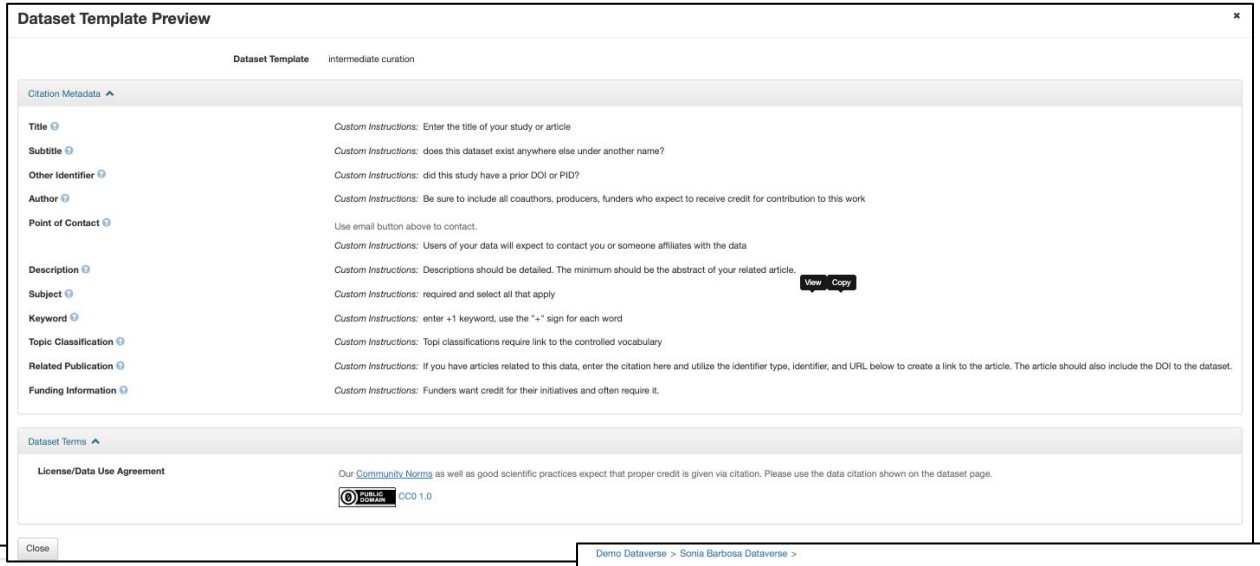
Building a proper dataset citation requires you complete this information

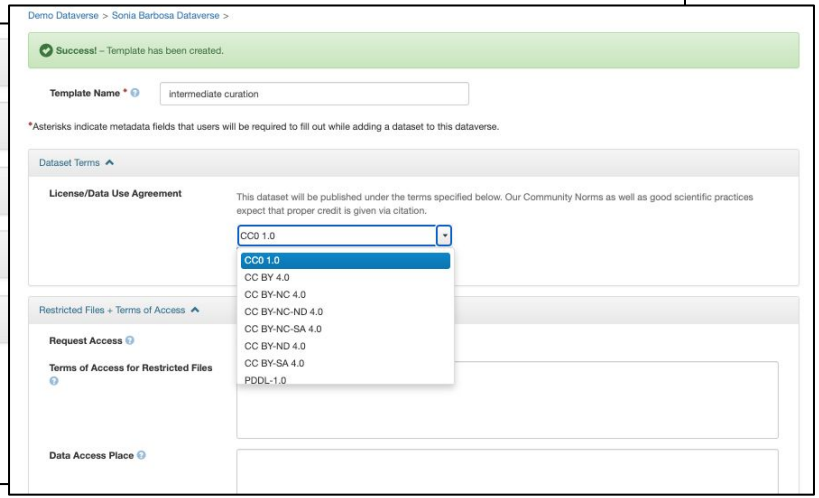


The Dataverse® Project

Intermediate Options in:

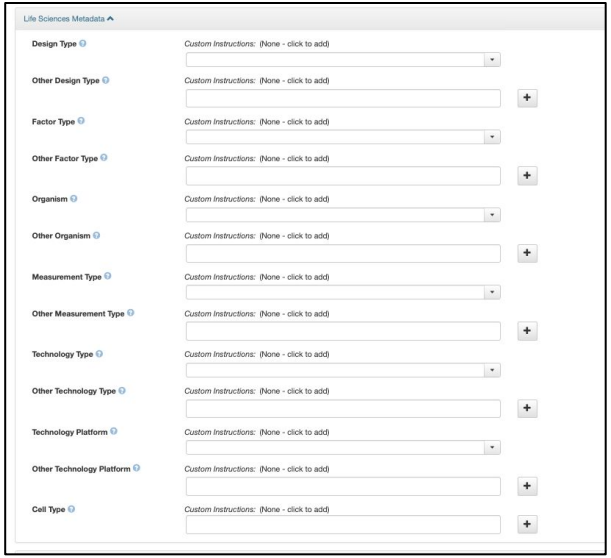
Datasets

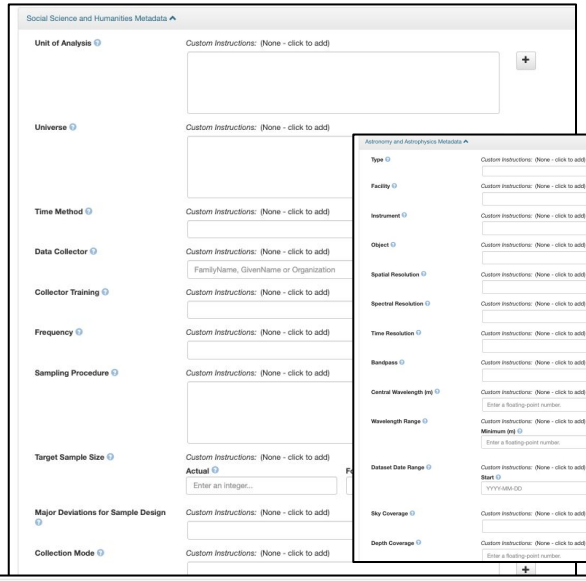


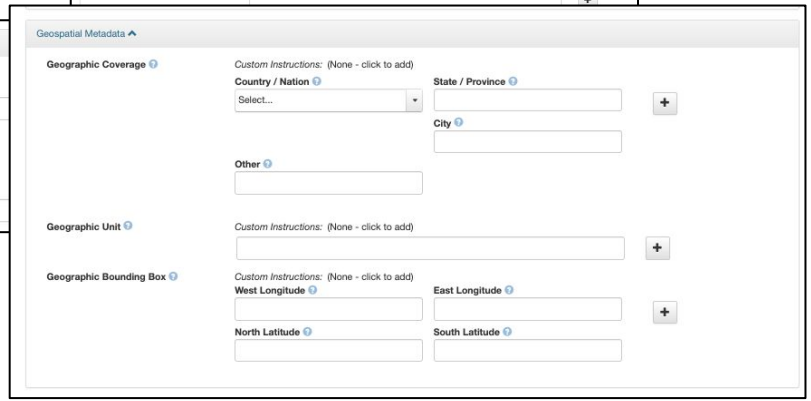


Intermediate Options in:

[Datasets]

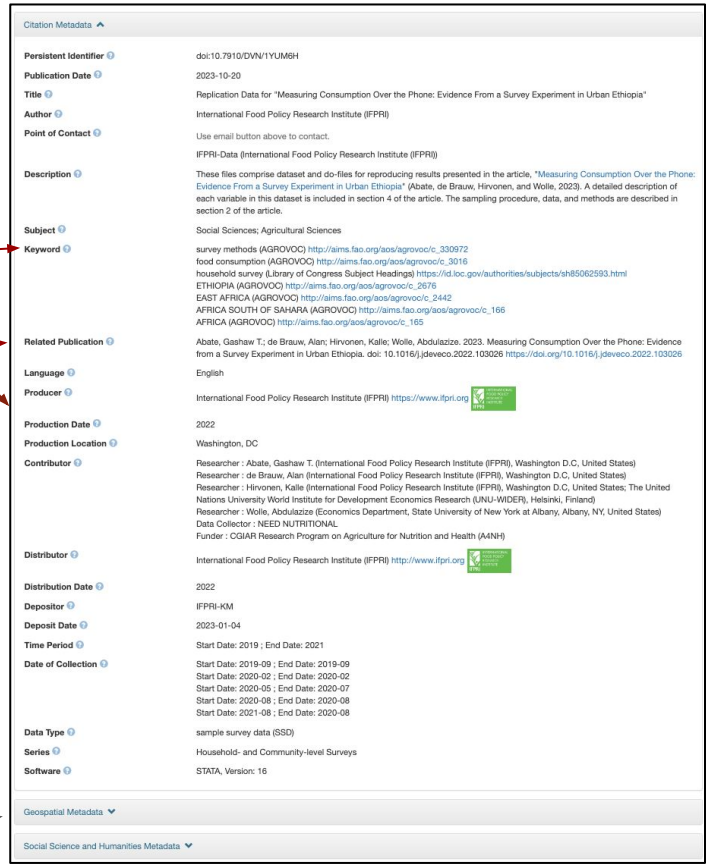






Intermediate Options in:

[Datasets]

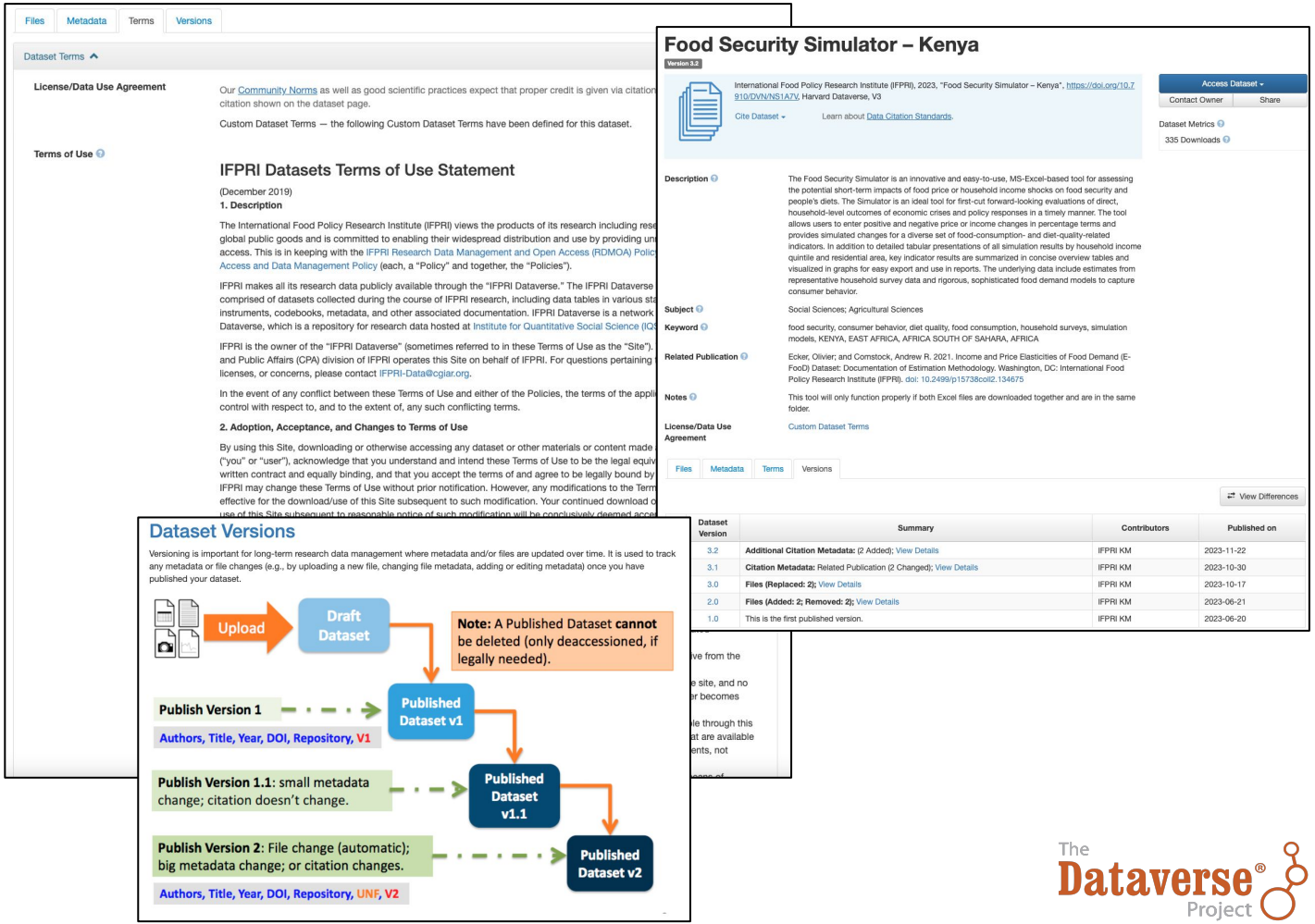


Multiple metadata blocks

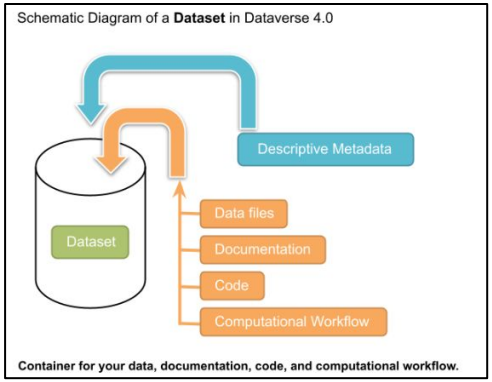
The Dataverse® Project

Intermediate Options in:

[Datasets]



The Dataverse® Project



Best practices for “Files” creation:

- File types and sizes
  - Compressed
  - Tabular
- Know what license to use
- Data are deidentified
- All pre sharing requirements have been fulfilled (see data quality above)
- File level descriptions
- Know what additional features are available to support multi file uploads, folders

While some of this is required and can’t be skipped, not enabling or using these other options results in underutilization of the features available to you

Intermediate Options in:

Files

- File Handling
  - File Previews
  - Tabular Data Files
  - Research Code
  - Computational Workflow
    - Computational Workflow Definition
    - FAIR Computational Workflow
    - How to Create a Computational Workflow
    - How to Upload Your Computational Workflow
    - How to Describe Your Computational Workflow
    - How to Search for Computational Workflows
  - Astronomy (FITS)
  - GeoJSON
  - GeoTIFF
  - Shapefile
  - NetCDF and HDF5
    - H5Web Previewer
    - NcML
    - Geospatial Bounding Box
  - Compressed Files
  - Other File Types

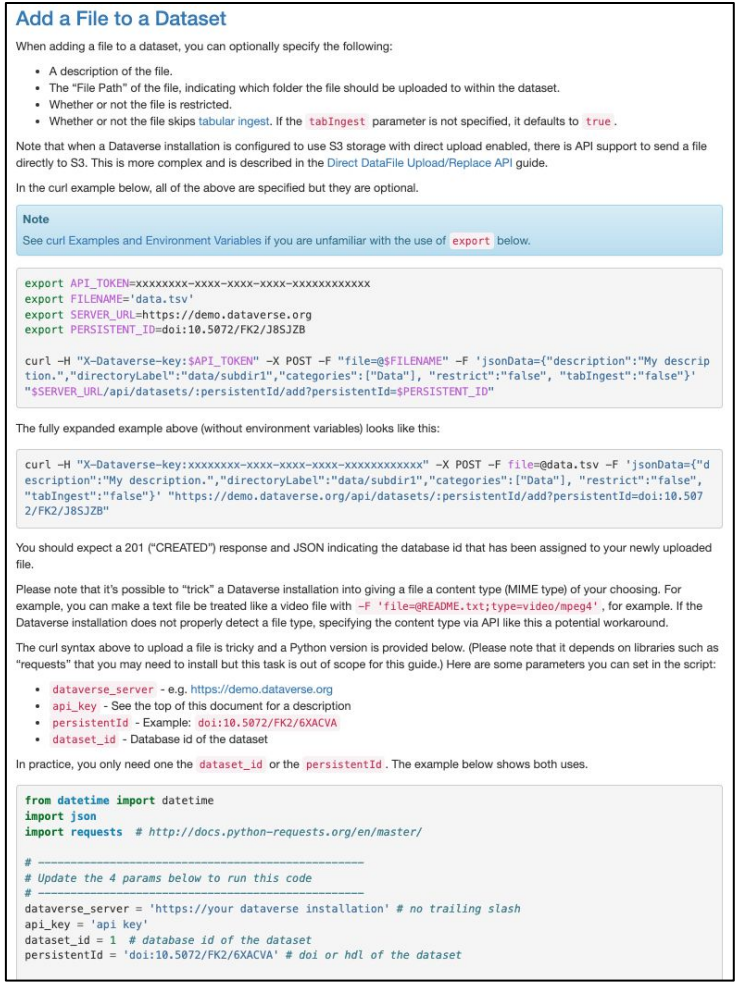
When adding a file to a dataset, you can **optionally specify the following**:

- A **description** of the file.
- The "**File Path**" of the file, indicating which folder the file should be uploaded to within the dataset.
- Whether or not the file is **restricted**.
- Whether or not the file skips tabular ingest. If the `tabIngest` parameter is not specified, it defaults to `true`.

Note that when a Dataverse installation is configured to use S3 storage with direct upload enabled, there is API support to send a file directly to S3. This is more complex and is described in the Direct DataFile Upload/Replace API guide.

Intermediate Options in:

Files



Add a File to a Dataset

When adding a file to a dataset, you can optionally specify the following:

- A description of the file.
- The "File Path" of the file, indicating which folder the file should be uploaded to within the dataset.
- Whether or not the file is restricted.
- Whether or not the file skips tabular ingest. If the `tabIngest` parameter is not specified, it defaults to `true`.

Note that when a Dataverse installation is configured to use S3 storage with direct upload enabled, there is API support to send a file directly to S3. This is more complex and is described in the Direct DataFile Upload/Replace API guide.

In the curl example below, all of the above are specified but they are optional.

Note
See curl Examples and Environment Variables if you are unfamiliar with the use of `export` below.

```
export API_TOKEN=xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
export FILENAME='data.tsv'
export SERVER_URL=https://demo.dataverse.org
export PERSISTENT_ID=doi:10.5072/FK2/J8SJZB

curl -H "X-Dataverse-key:$API_TOKEN" -X POST -F "file=@$FILENAME" -F 'jsonData={"description":"My descrip
tion.","directoryLabel":"data/subdir1","categories":["Data"], "restrict":"false", "tabIngest":"false"}'
"$SERVER_URL/api/datasets/:persistentId/add?persistentId=$PERSISTENT_ID"
```

The fully expanded example above (without environment variables) looks like this:

```
curl -H "X-Dataverse-key:xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx" -X POST -F file=@data.tsv -F 'jsonData={"d
escription":"My description.","directoryLabel":"data/subdir1","categories":["Data"], "restrict":"false",
"tabIngest":"false"}' "https://demo.dataverse.org/api/datasets/:persistentId/add?persistentId=doi:10.507
2/FK2/J8SJZB"
```

You should expect a 201 ("CREATED") response and JSON indicating the database id that has been assigned to your newly uploaded file.

Please note that it's possible to "trick" a Dataverse installation into giving a file a content type (MIME type) of your choosing. For example, you can make a text file be treated like a video file with `-F 'file=@README.txt;type=video/mpeg4'`, for example. If the Dataverse installation does not properly detect a file type, specifying the content type via API like this a potential workaround.

The curl syntax above to upload a file is tricky and a Python version is provided below. (Please note that it depends on libraries such as "requests" that you may need to install but this task is out of scope for this guide.) Here are some parameters you can set in the script:

- `dataverse_server` - e.g. https://demo.dataverse.org
- `api_key` - See the top of this document for a description
- `persistentId` - Example: `doi:10.5072/FK2/6XACVA`
- `dataset_id` - Database id of the dataset

In practice, you only need one the `dataset_id` or the `persistentId`. The example below shows both uses.

```
from datetime import datetime
import json
import requests  # http://docs.python-requests.org/en/master/

# ----------------------------------------------------
# Update the 4 params below to run this code
# ----------------------------------------------------
dataverse_server = 'https://your dataverse installation' # no trailing slash
api_key = 'api key'
dataset_id = 1  # database id of the dataset
persistentId = 'doi:10.5072/FK2/6XACVA' # doi or hdl of the dataset
```

View Dataset Files and Folders as a Directory Index

Provides a *crawlable* view of files and folders within the given dataset and version:

CORS

```
curl "$SERVER_URL/api/datasets/${ID}/dirindex/"
# or
curl "${SERVER_URL}/api/datasets/:persistentId/dirindex?persistentId=doi:${PERSISTENT_ID}"
```

Optional parameters:

- `folder` - A subfolder within the dataset (default: top-level view of the dataset)
- `version` - Specifies the version (default: latest published version)
- `original=true` - Download original versions of ingested tabular files.

This API outputs a simple html listing, based on the standard Apache directory index, with Access API download links for individual files, and recursive calls to the API above for sub-folders.

Using this API, `wget --recursive` (or a similar crawling client) can be used to download all the files in a dataset, preserving the file names and folder structure; without having to use the download-as-zip API. In addition to being faster (zipping is a relatively resource-intensive operation on the server side), this process can be restarted if interrupted (with `wget --continue` or equivalent) - unlike zipped multi-file downloads that always have to start from the beginning.

On a system that uses S3 with download redirects, the individual file downloads will be handled by S3 directly, without having to be proxied through the Dataverse application.

For example, if you have a dataset version with 2 files, one with the folder named "subfolder":

1 to 2 of 2 Files — Edit Files — Download

50by1000.tab
subfolder/
Tabular Data - 102.5 KB - Jan 11, 2021 - 3 Downloads
50 Variables, 1000 Observations - UNF:6:x10r+Q9 EK6aF/BMi+eKzGw==

testfile.txt
Plain Text - 19 B - Jan 13, 2021 - 0 Downloads
SHA-1: 44be2930968c0159f17ef895807150236fdc9d72

or, as viewed as a tree on the dataset page:

Change View — Table | Tree

- subfolder
  - 50by1000.tab (102.5 KB)
  - testfile.txt (19 B)

The output of the API for the top-level folder (`/api/datasets/{dataset}/dirindex/`) will be as follows:

Index of folder / in dataset doi:XXX/YY/ZZZZ (v. MM)

Name	Last Modified	Size	Description
subfolder/		-	-
testfile.txt	13-January-2021 22:35	19 B	

The Dataverse® Project

Intermediate Options in:

[Files]

Uploading Files

To prepare, log in to Dataverse and:

- find the DOI for the dataset you wish to add files to, and
- find or generate an API key for yourself in the Dataverse instance you are using (from the popup menu under your profile).

The simplest way to run the DVUploader is to place the jar file into the directory containing a subdirectory with the files intended for upload. (The DVUploader can be placed anywhere on disk and can upload files from any directory, but this requires adding these paths to the command line and/or configuration of Java's classpath.)

REQUIRED: Run the jar with the following command line:

```
java -jar DVUploader-v1.2.0beta3.jar -key=<api key> -did=<dataset doi> -server=<server URL> <dir or file names>
```

where

<apikey> is replaced with the API Key generated by the user in Dataverse

<dataset doi> is replaced with the DOI of the target Dataset

<serverURL> is replaced by the URL of the Dataverse server being used (with no trailing '/' and do not include any path to a specific Dataverse on the server), and

<dir or file names> is replaced by the name of a directory and/or a list of individual files to upload.

These four arguments are always required. There are additional options listed below. **Note: **For a first test, adding -listonly is useful - it will make the DVUploader list what it would do, but will not perform any uploads.

For example, `java -jar DVUploader-v1.1.0.jar -key=8599b802-659e-49ef-823c-20abd8efc05c -did=doi:10.5072/FK2/TUNNVE -server=https://dataverse.tdl.org testdir` would upload all of the files in the 'testdir' directory (relative to the current directory where the java command is run) to the Dataset at https://dataverse.tdl.org/dataset.xhtml?persistentId=10.5072/FK2/TUNNVE (if it existed: the dataset in this example is not real).

The output from the DVUploader looks like:

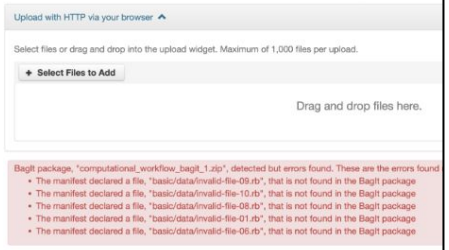
```
Dataverse Mode: Uploading files to a Dataverse instance  Using apiKey: 8599b802-659e-49ef-823c-20abd8efc05c
Adding content to: doi:10.5072/FK2/TUNNVE  Using server: https://dataverse.tdl.org  Request to upload: testdir

PROCESSING(C): testdir  Found as: doi:10.5072/FK2/TUNNVE
```

- File Upload
  - HTTP Upload
  - Dropbox Upload
  - Folder Upload
  - rsync + SSH Upload
    - File Upload Script
  - Command-line DVUploader
    - Usage
  - Integrations Dashboard Uploader
  - Duplicate Files
  - BagIt Support

BagIt Support

BagIt is a set of hierarchical file system conventions designed to support disk-based storage and network transfer of arbitrary digital content. It offers several benefits such as integration with digital libraries, easy implementation, and transfer validation. See the Wikipedia article for more information.

If the Dataverse installation you are using has enabled BagIt file handling, when uplo
checksum values listed in each BagIt's manifest file against the uploaded files and g
repository will identify a certain number of errors, such as the first five errors in each



Supported File Formats

Tabular Data ingest supports the following file formats:

File format	Versions supported
SPSS (POR and SAV formats)	7 to 22
STATA	4 to 15
R	up to 3
Excel	XLSX only (XLS is NOT supported)
CSV (comma-separated values)	(limited support)

See the subsections in the left sidebar for more information on each of these supported formats.

Intermediate Options in:

Files

File Previews

Dataverse installations can add previewers for common file types uploaded by their research communities. The previews appear on the file page. If a preview tool for a specific file type is available, the preview will be created and will display automatically, after terms have been agreed to or a guestbook entry has been made, if necessary. File previews are not available for restricted files unless they are being accessed using a Private URL. See also Private URL to Review Unpublished Dataset.

Previewers are available for the following file types:

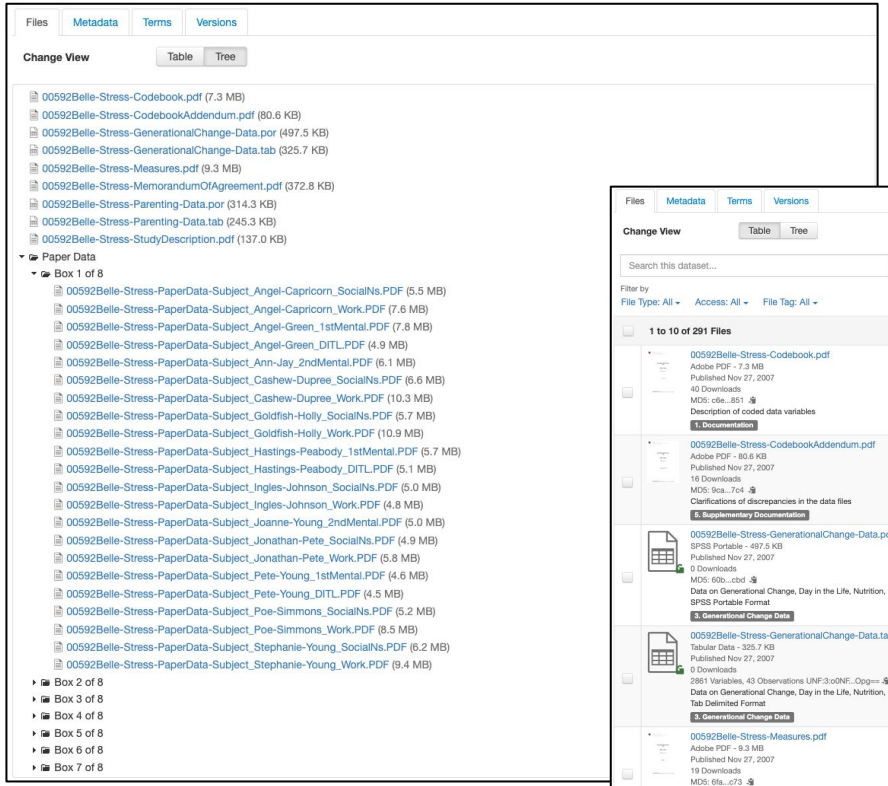
- Text
- PDF
- Tabular (CSV, Excel, etc., see Tabular Data File Ingest)
- Code (R, etc.)
- Images (PNG, GIF, JPG)
- Audio (MP3, MPEG, WAV, OGG, M4A)
- Video (MP4, OGG, Quicktime)
- Zip (preview and extract/download)
- HTML
- GeoJSON
- GeoTIFF
- Shapefile
- NetCDF/HDF5
- Hypothes.is

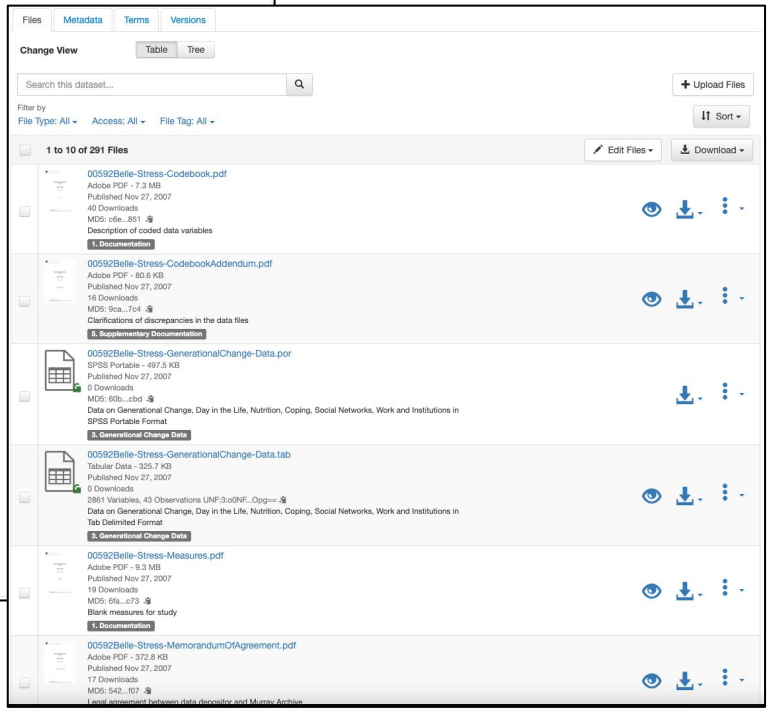
Additional file types will be added to the dataverse-previewers repo before they are listed above so please check there for the latest information or to request (or contribute!) an additional file previewer.

Installation of previewers is explained in the External Tools section of in the Admin Guide.

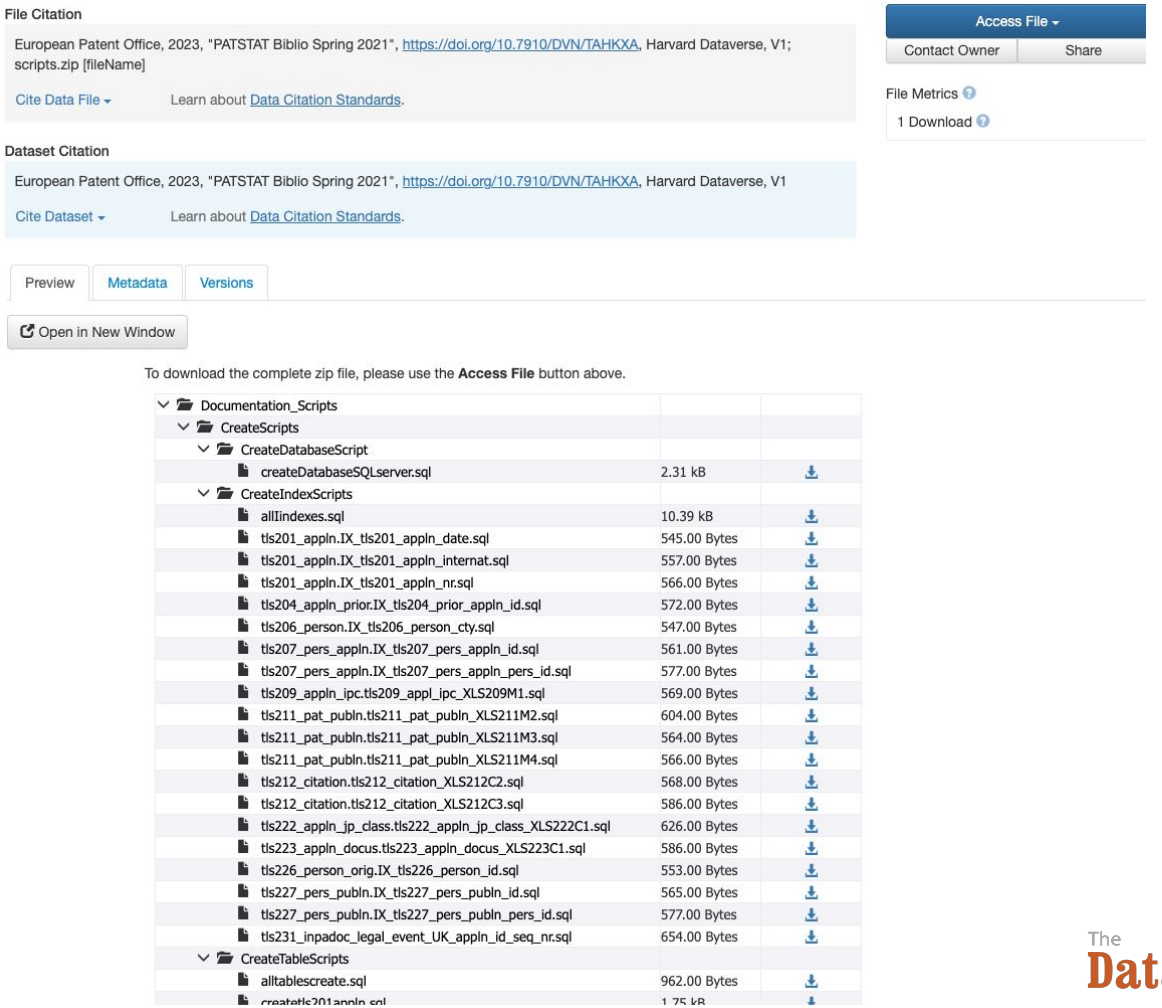
Intermediate Options in:

Files





The Dataverse® Project



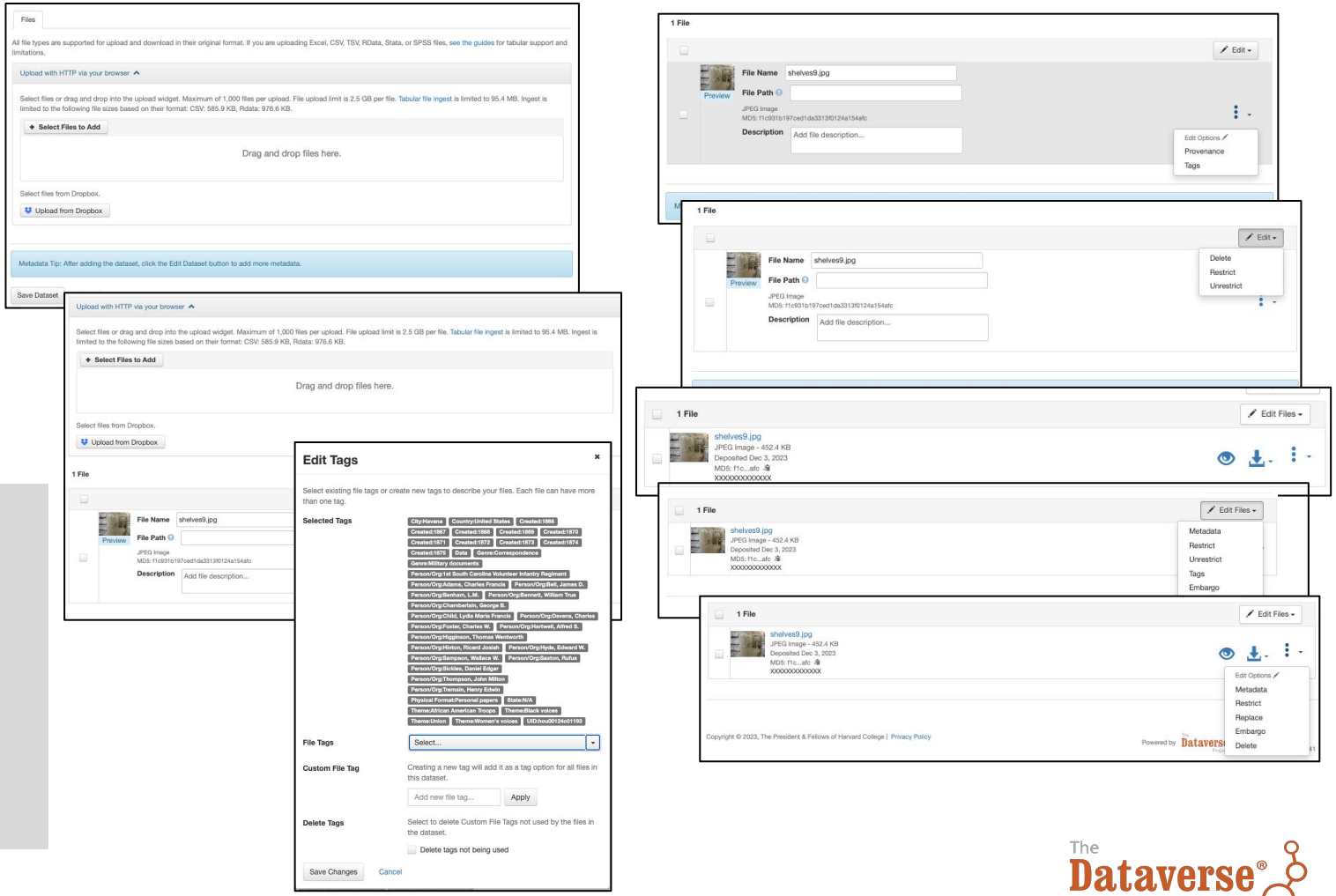
Intermediate Options in:

[Files]

The Dataverse® Project

Intermediate Options in:

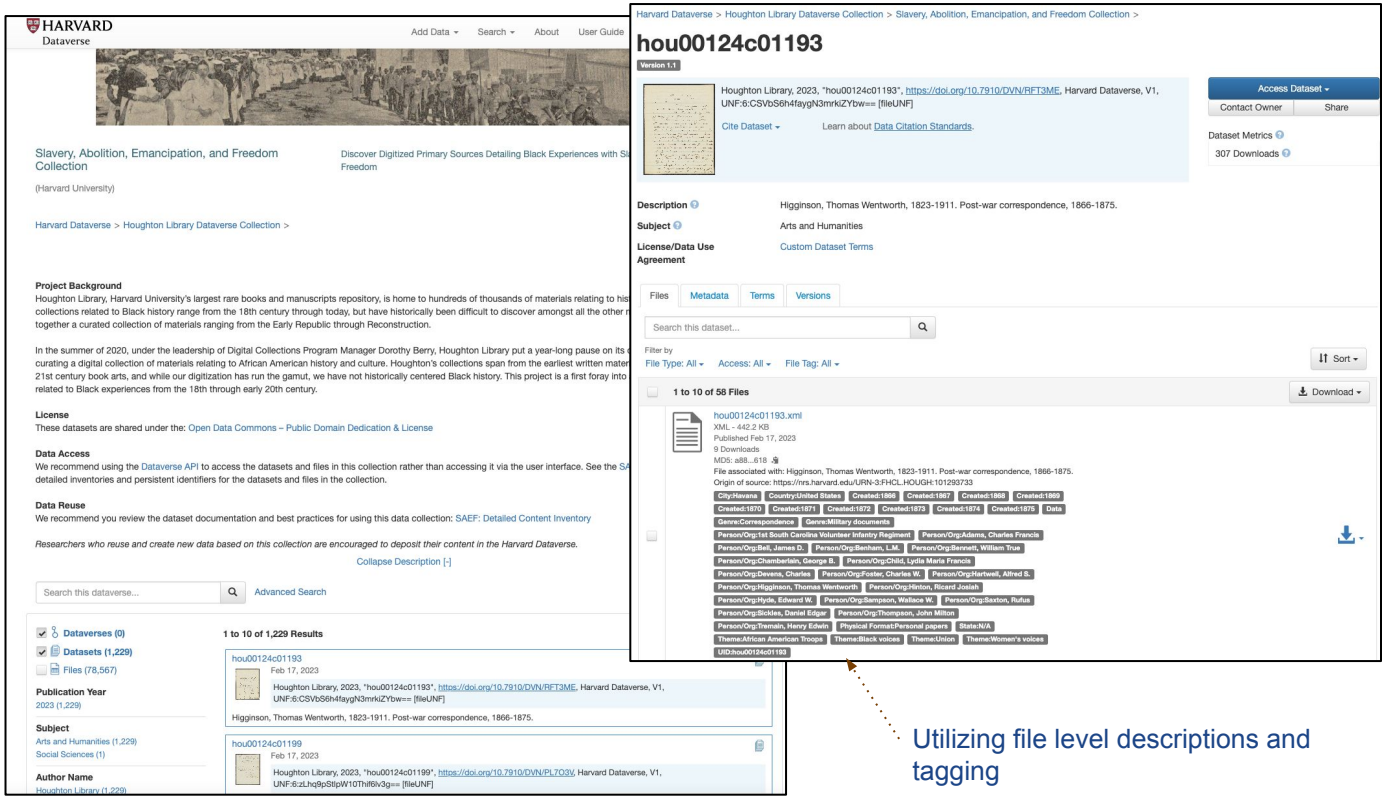
[Files]



SAEF Collection

SAEF on GitHub

Utilizing a “collection” with detailed descriptions and links to related materials as well as information on utilizing APIs for downloading this many-files dataset



Utilizing file level descriptions and tagging

Intermediate & Advanced Settings in:

Files

The Dataverse® Project

SAEF Collection

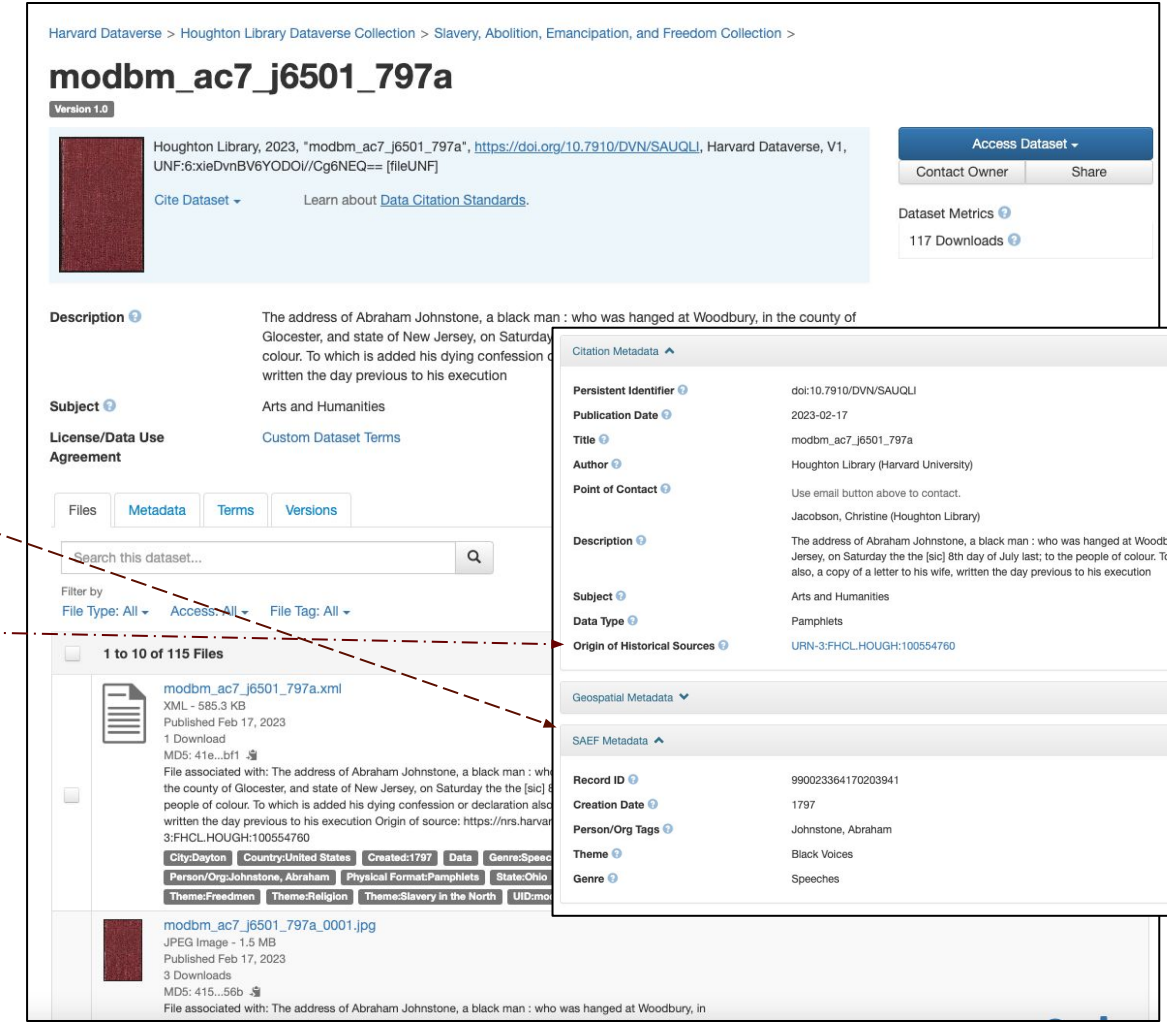
SAEF on GitHub

Utilizes a "Custom metadata block" as well as multiple metadata blocks

Utilizes linking to external data sources

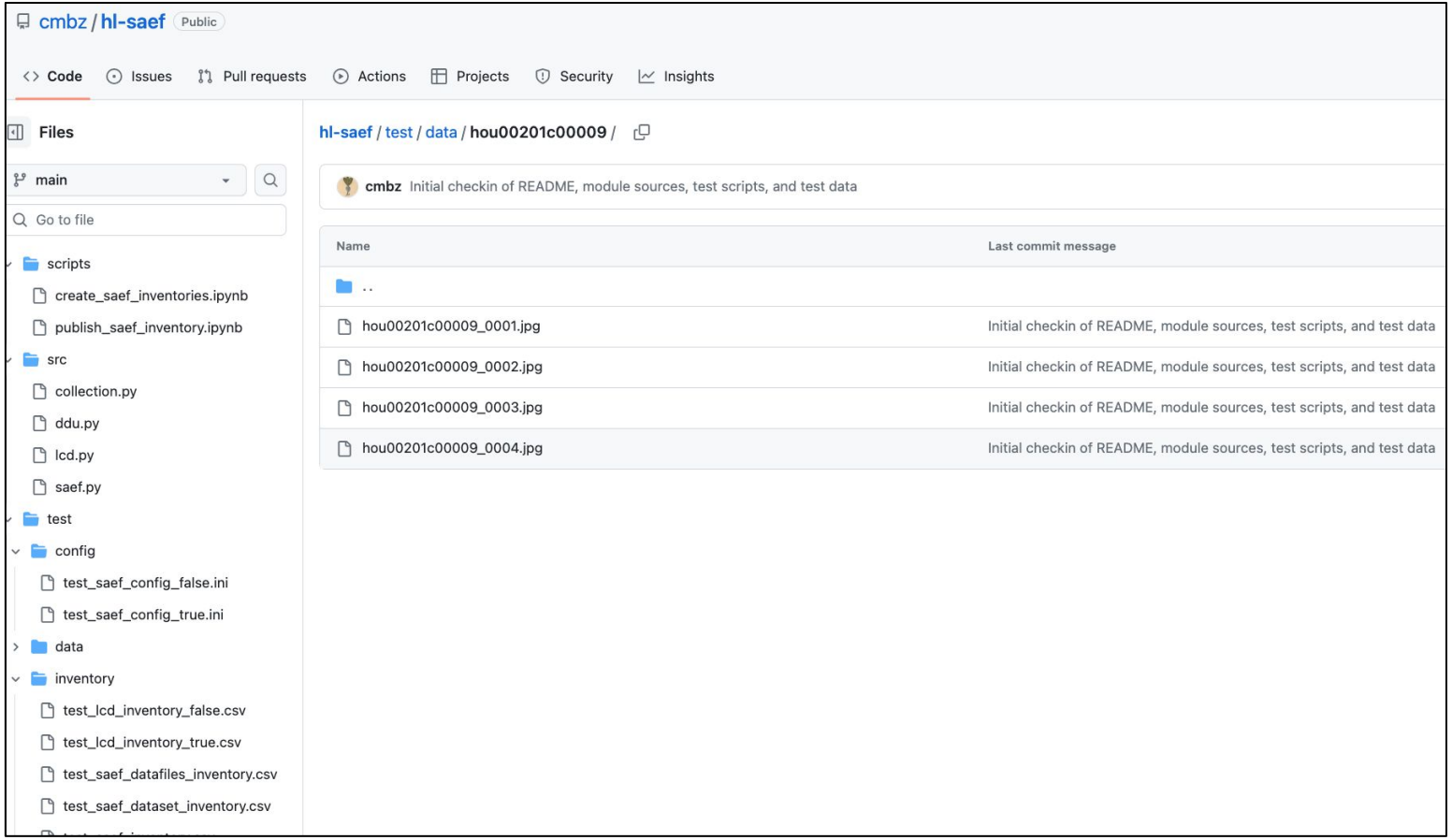
Intermediate & Advanced Settings in:

Files



SAEF Collection

SAEF on GitHub



Intermediate & Advanced Settings in:

Files

The Dataverse® Project

SAEF Collection

SAEF on GitHub

```
# create the pyDataverse datafile
datafile = Datafile()
# set the metadata on the datafile
datafile.set({'pid': pid, 'filename': filepath, 'restrict':restrict,
              'description':description, 'categories':categories})

# upload the datafile via the api
response = api.upload_datafile(pid, filepath, datafile.json(), is_pid=True)
status = int(response.status_code)
if (not ((status >= 200) and
   (status < 300))):
    msg = 'Upload failed: {}'.format(response.status_code)
    print('SAEFDataset::upload_datafiles: Error - failed to upload datafile {}. {}'.format(filepath, msg))
    # log the event
    message = '{} - filename: {} - {}'.format(self._object_osn, filepath, msg)
    self.log_api_message('SAEF::upload_datafiles', 'api.upload_datafile: {}'.format(filepath), status, message)
else:
    # log the successful event
    msg = '{} - {}'.format(self._object_osn, filepath)
    self.log_api_message('SAEF::upload_datafiles', 'api.upload_datafile', status, msg)
```

Intermediate & Advanced Settings in:

Files

The Dataverse® Project

The Dataverse® Project

Summary

Before data sharing:

- Explore data cleaning and validation options and techniques for handling missing data, outliers, and anomalies
- Ensure data consistency and integrity
- Employ data normalization and transformation techniques
- Examine strategies for data integration and metadata management
- Know the importance of versioning and data provenance

When sharing data (Dataverse):

- Utilize APIs and tools for “many” files deposits
- Utilize “collection” creation
  - Select additional metadata blocks outside of the default block
  - Select searchable facets to enhance discoverability of your datasets
- When creating the “dataset,” complete as many metadata fields as relevant to your dataset, and include useful detailed information for title, description, keywords, and related articles to increase the Findability, Accessibility, and ReUsability of your dataset
- Organize Files in supported formats (tabular, compressed) to take advantage of the preview tools and archival functionality that supports Interoperability and ReUse

Thank you!