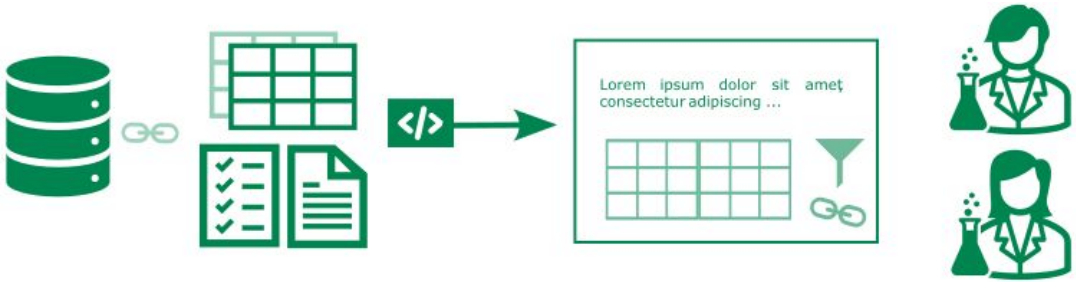
Where are my data?

The AFFORD workflow to create your own data index with Git, R and Quarto

Gorka Fraga González



https://osf.io/preprints/metaarxiv/64fch_v1

🎯 Goals

- Show researchers ways of improving data management with simple tools
- Enhance data reusability
- Promote workflows to improve computational reproducibility

👥Target audience

- Researchers with an interest in data management
- Data stewards
- Requires: basic IT affinity and basic Git and R or Python

👌 Main Features

Affordable, simple, flexible, uses generalist and open source tools

Background

A Framework for Avoiding Open Research Data Dump (AFFORD)

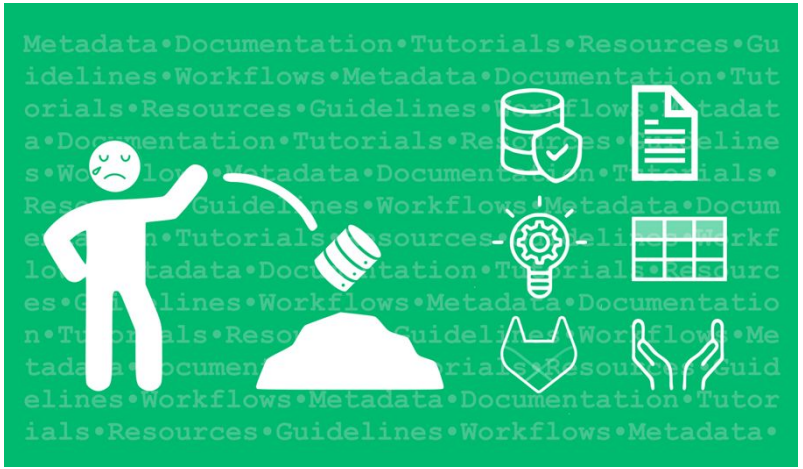
We supported data curation in a project:

- Data-intensive (biomedical data)
- Interdisciplinary and multicenter
- Ongoing data collection waves

Background

A Framework for Avoiding Open Research Data Dump (AFFORD)

We supported data curation in a project:



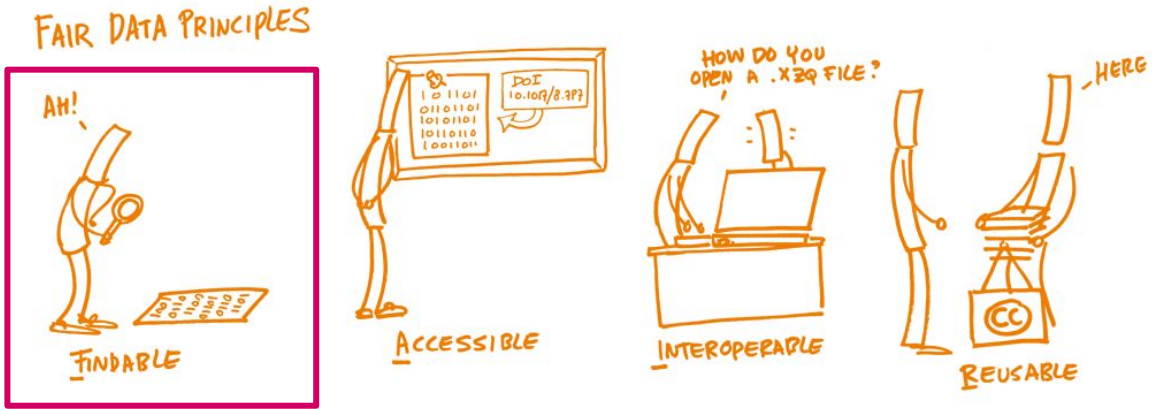
- We focused on one data type (synchrotron) and team
- Focused on sharing within project
- Limited time frame: need to minimize dependencies in the medium/long term

Background

Findable, Accessible, Interoperable and Reusable (FAIR)

Guiding principles for managing research data. Focus on :

- Machine-readable **metadata**
- **Documentation**
- **Persistent identifiers**
- **Interoperable** formats and vocabularies
- **Licensing**
- **Publishing** in accessible repositories



https://www.ub.uzh.ch/de/wissenschaftlich-arbeiten/mit-daten-arbeiten/FAIR-und-Open-Data.html

Background

FAIR data is **challenging**

- Interdisciplinary differences in technical skills and research culture
- Data-intensive research: complex pipelines, heterogeneous file types
- Heterogeneity of resources for data curation (e.g., stewards, IT)
- Fragmented landscape of tools
- Time-consuming decision making and coordination

A data index

“To share your data you first need to find it” - Anonymous

Data index: catalogue that allows browsing through data by using the available metadata.

A data index in the AFFORD project

in AFFORD our **target** data index should:

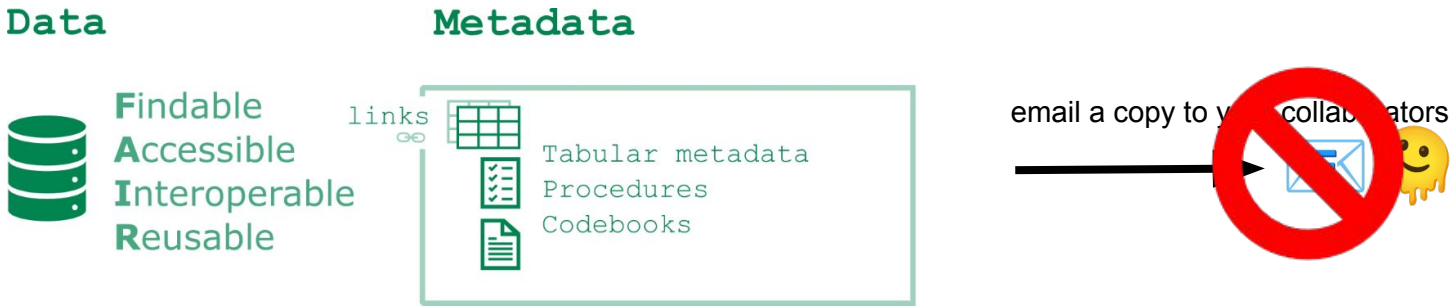
- Provide enough context (metadata) to the data
- Allow to find where the source data files
- Privacy: first, it should be possible to share only internally
- Eventually, it should be public and useful for all
- **Maintainable** by researchers, **scalable** and **portable**

A data index in the AFFORD project

Data

- Large volume stored externally
- Different storage locations
- Subsets expected to be published online
- Ongoing data collection
- Different facilities -> different constraints (e.g., filenaming, folder structure)
- Different data types

A data index in the AFFORD project



- Large volume stored externally
- Different storage locations
- Subsets expected to be published online
- Ongoing data collection
- Different facilities -> different constraints (e.g., filenaming, folder structure)
- Different data types

Minimum FAIRNESS requires

- Machine-readable metadata
- Procedures
- Codebooks
- Documentation

A data index in the AFFORD project

- Large volume stored externally
- Different storage locations
- Subsets expected to be published online
- Ongoing data collection
- Different facilities -> different constraints (e.g., filenaming, folder structure)
- Different data types

Minimum FAIRNESS requires

- Machine-readable metadata
- Procedures
- Codebooks
- Documentation

Metadata + simple code to enable SHARING:

- Navigate through metadata
- Filters
- Clickable links to data location
- OPTIONALLY: representative data previews

A data index in the AFFORD project

- Large volume stored externally
- Different storage locations
- Subsets expected to be published online

Minimum FAIRNESS requires

- Machine-readable metadata
- Procedures

Metadata + simple code to:

- Navigate through metadata
- Filters
- Clickable links to data location
- OPTIONALLY: links to tative data previews

- **FAIR** is much more: full machine-actionability, controlled vocabularies, ontologies, integration in larger data frameworks, etc.
  But we start with an unambitious, basic level FAIRness that would enable data to be found and understood by other researchers.

The workflow

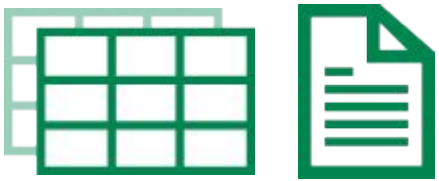
It uses 3 components important for computational reproducibility:

1. **Dynamic document generation**
2. Repositories with **Git version control**
3. Software **containerization**

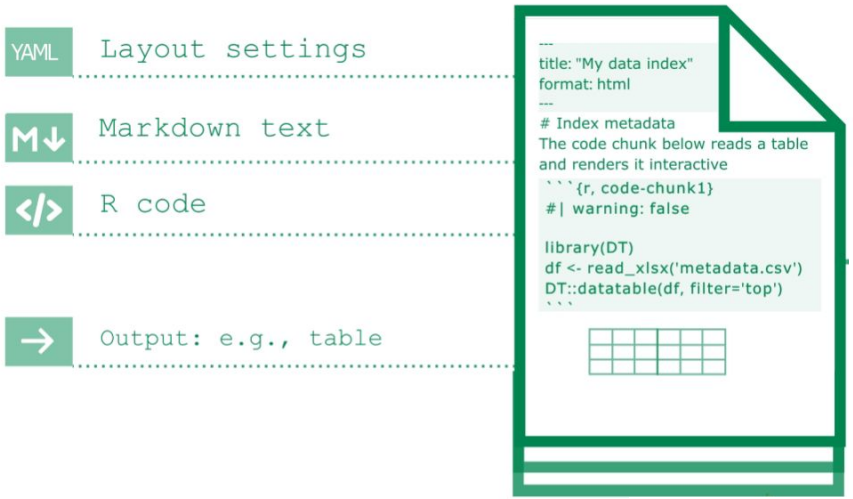
The workflow

It requires researchers to create first:

- **Metadata tables** in CSV format (comma separated value, favoured for interoperability). Adequately formatted so that is machine readable
- **Codebooks** describing the variables



The workflow



1. Dynamic document generation

We use Quarto, integrated in Rstudio.
It can read a file with:

- Header settings/metadata (YAML)
- Markdown text
- Executable Code (R, Python,...)

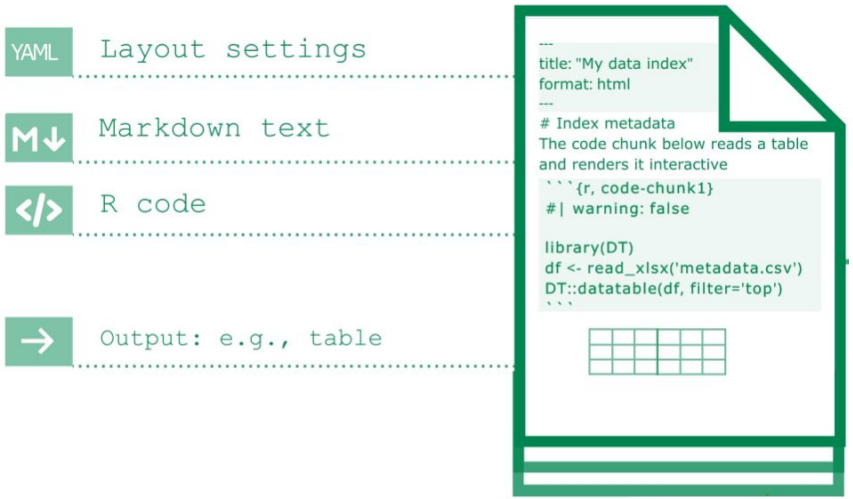
And print it into multiple formats:

- HTML, pdf, word, etc.

Group docs with different layouts:

- Websites, book, reports, etc

The workflow



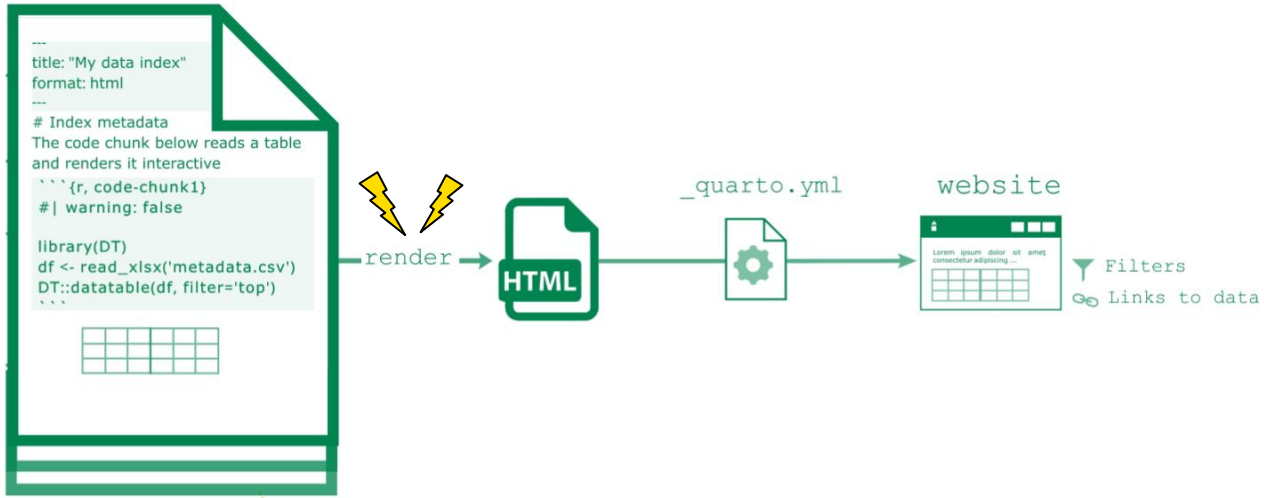
YAML header. Title, author information, type of output, document , etc.

Markdown text. Includes instructions, introductory descriptions, etc.

R code chunk. Reads .CSV tables with metadata and *generates (render)* an interactive table with filters options

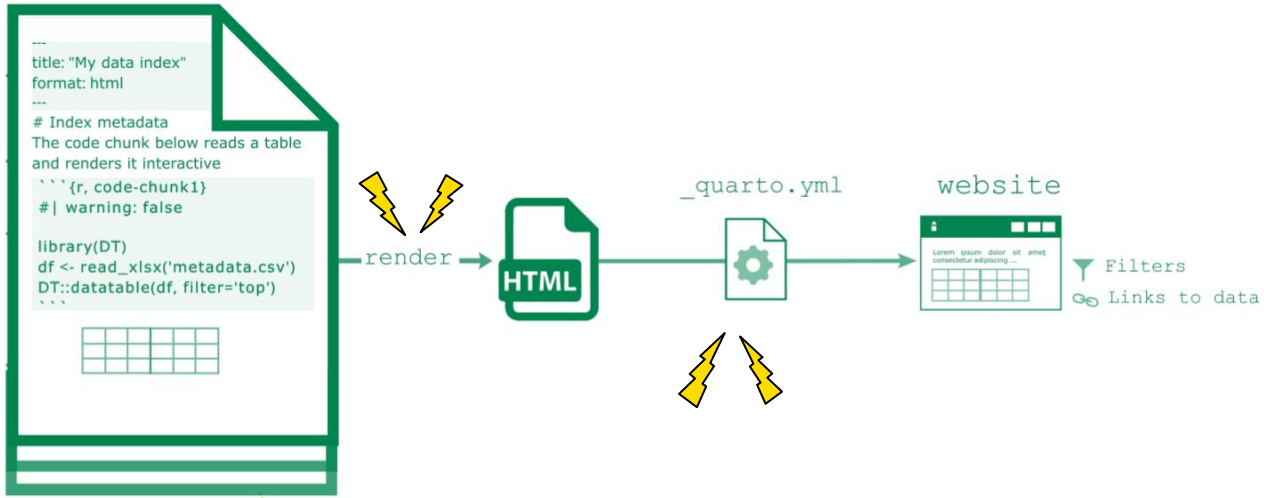
Note: these files have .qmd extension. They can be opened and edited with any text editor (e.g., notepad) as they are plain text. But to execute you need Quarto and R (e.g., in R studio)

The workflow



When we run the command *quarto render*, the code is executed and the file is saved into a nicely-formatted HTML with an interactive version of the table (with filter panels or any additional panel we want to add in the code)

The workflow



(RECOMMENDED) configuration file *_quarto.yml (quarto project file)* is also a plain text file with a series of options offered by quarto to define the project type and layout. In this case we have a 'website' that combines multiple HTML files as there are several tables.

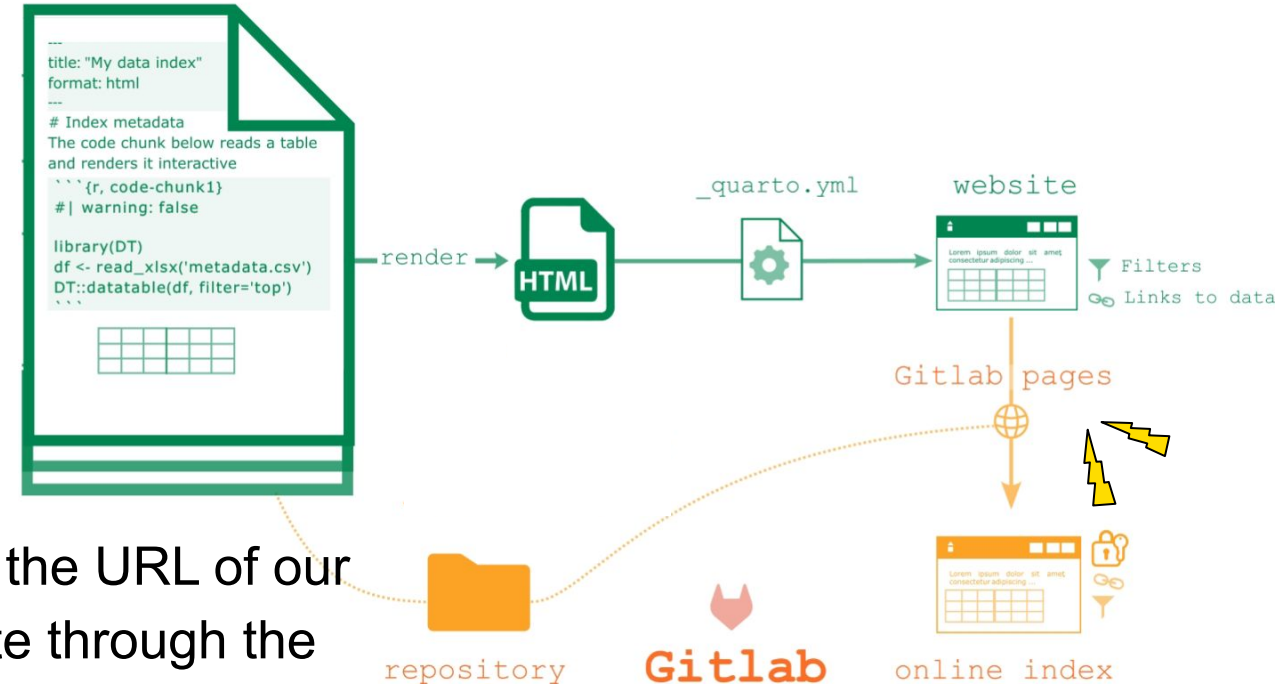
The workflow

2. Version control with Git.

Metadata tables (.CSV) and source Quarto files to generate reports (.QMD) are stored in a Git repository. Git is a decentralized system for tracking changes in files

- Our repository is hosted by Gitlab (instance managed by UZH)
- Gitlab offer Gitlab pages a services to make HTML documents accessible from an internet browser
- Gitlab pages can be **private**
- They are **static** pages (all users see only whatever is in the HTML)

The workflow



Collaborators can go to the URL of our Gitlab page and navigate through the metadata tables

The workflow

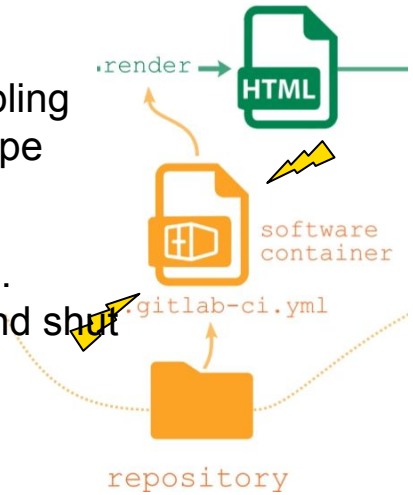
3. Software containerization

The code to make the interactive table and website should not break after a package is updated. We run it always through an ‘encapsulated software environment’

Recipe. Human and machine-readable text file. A blueprint of the environment

Image. Machine-readable file assembling and storing all components in the recipe

Container. A working copy of the environment assembled in the image. Meant to be temporary: it is started and shut down after use.



The workflow

Gitlab Continuous Integration (Gitlab) file

Every time we change the repository (e.g., new entry on the table). This file executes the code from the software container

```
image: rocker/verse:4.5.0

pages:
  stage: deploy
  before_script:
  # Install additional packages using dependency version control by rocker
  - install2.r kableExtra openxlsx DT here fs stringr

  script:
  # Create site folders
  - mkdir ./public
  # Call python to convert inputs from the CMS into .qmd and add code chunks if needed
  - python3 ./admin/convert_md_to_qmd.py ./webpage_contents/ ./site_qmds/
  # Go to folder with QMDs and the quarto project file, render the site from there
  - cp -r ./webpage_contents/* ./site_qmds/
  - cd ./site_qmds
  - quarto render
  - cp -r ../_site/* ../public/
  - cp -r ../admin/ ../public/
  - cp -r . ../public/

  artifacts:
    paths:
    - public

  only:
  - main
  interruptible: true
```

A container image is downloaded the Rocker , a website with open images of R software

Extend rocker image with additional packages

in-house code to convert entries as .md into .qmd (see CMS extension in next slides)

Create website's HTML

The workflow

Gitlab Continuous Integration (Gitlab) file

Users don’t need to executed from our computer. The application that executing the code using the container is the Gitlab runner .

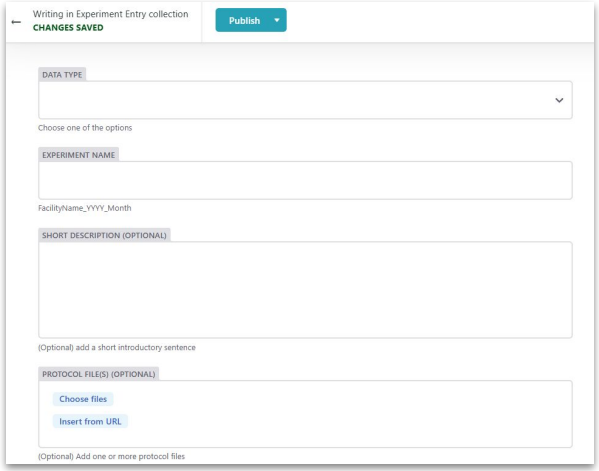
This code is very simple and not computational demanding so we will not need additional resources to run it.

Variations and extensions

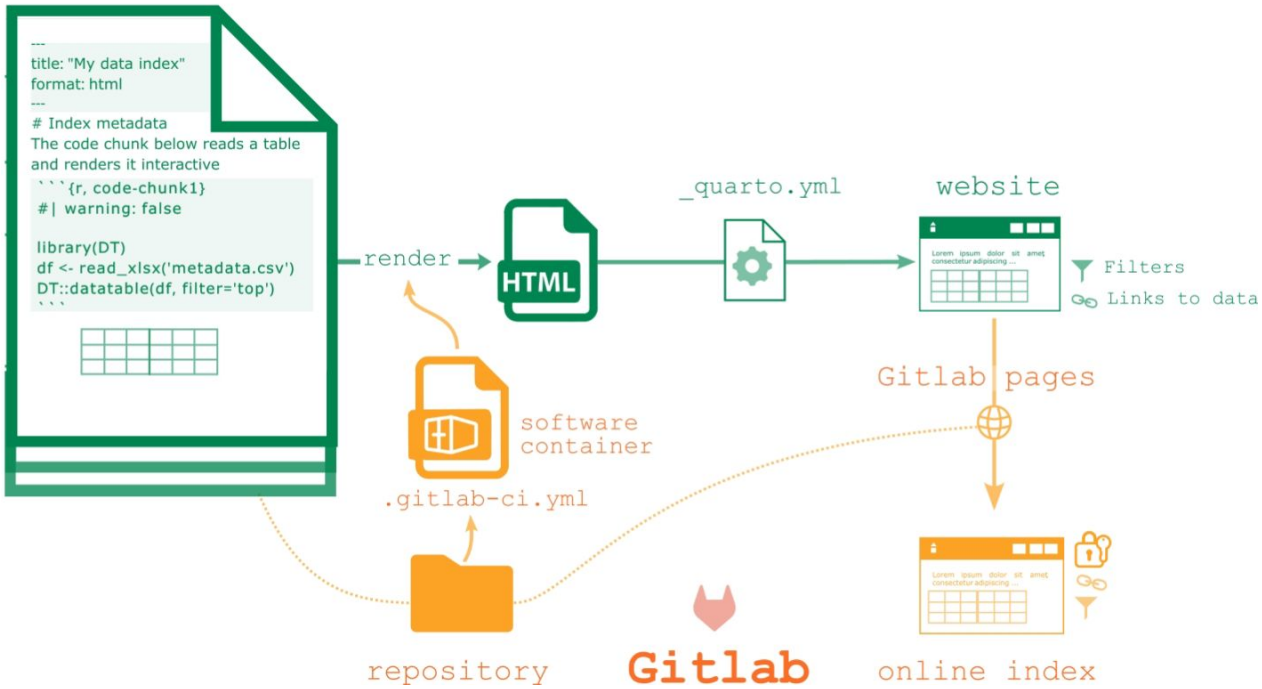
- Example fabric4.ch from the actual research project supported in AFFORD
- The table can also display **pictures** stored in the Repository. Example: https://crsuzh.pages.uzh.ch/AFFORD_website/ORD_index/
- A **simpler** use case: e.g., crsuzh.pages.uzh.ch/datastew/

A slightly more complicated workflow:

- To facilitate **data entry** for dummies, the application DECAP CMS can be use to have a simple Content Management System (CMS). A demo repository with this CMS is work in progress



The workflow



This code is just for illustration. Ignore the error reading using read_xlsx for a csv table ¯_(ツ)_/¯

Expected actions from researchers / data stewards

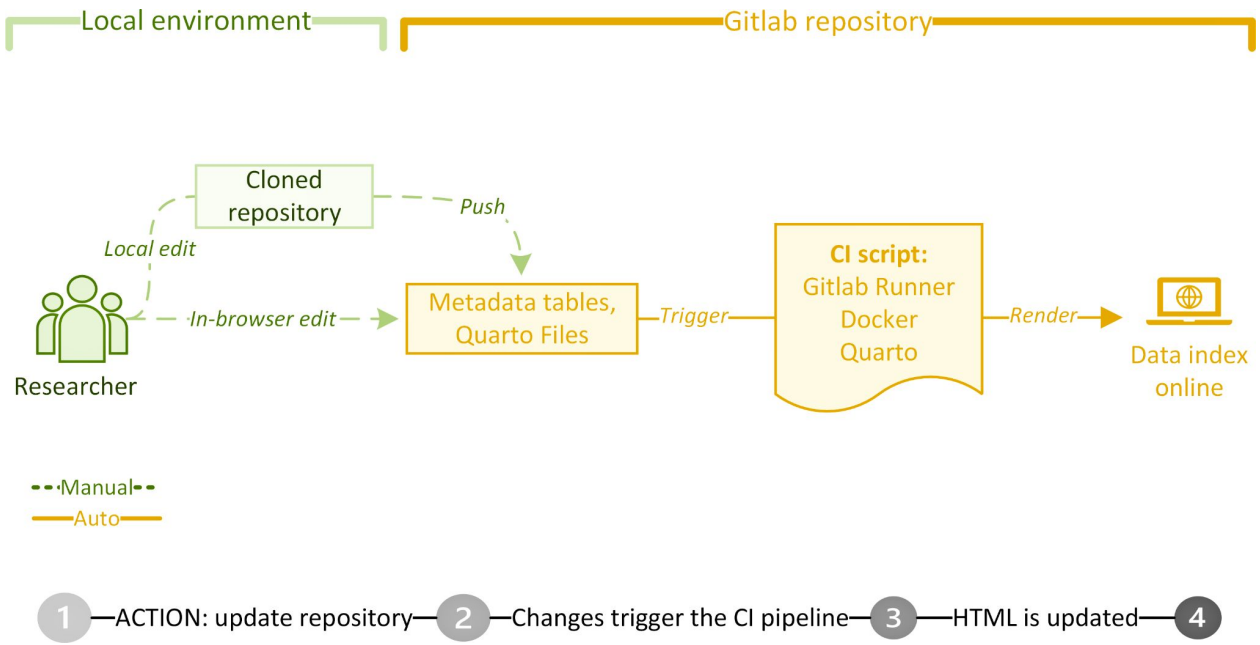
To **set up** this workflow requires familiarity with Git, R and Quarto is needed. You can start by copying this **demo repository** and edit it: https://crsuzh.pages.uzh.ch/afford_repos/affordable_data_index/

Recommendations for data stewards

- Explore customizing the workflow
- Keep it simple and easy to maintain in the future
- Consider your collaborators expertise
- Don’t make it entirely dependable of the one person that set it up
- Other coding languages and Git platforms enable similar actions
- Similar workflow can be applied to analytic code, visualizations, etc

The actions from researchers

Collaborators can edit the metadata through the browser (or via a local copy of the repository). The website updates automatically after changing the table



Making the Index Citable

- Citing the URL to a Gitlab page is not recommended
- Make an entry in a platform enabling a Digital Object Identifier (DOI) (e.g., Zenodo, Open Science Framework) and link it to the URL
- If the content is stable, upload a copy of the code repository (‘release’) including the source metadata tables
- If not stable, users will still access the website through the URL at the DOI entry. If the URL changes , the URL in the DOI entry can be changed without requiring a new DOI

Challenges

- well-formatted metadata tables cannot be taken for granted
- PIs need to agree on minimum actions to integrate all groups
- future maintainers will need additional training (e.g., Git commits)
- if made too easy for editing (e.g., CMS) it won't be as easy to maintain code for a researcher-steward

Summary and discussion

This workflow

- needs machine-readable metadata tables, encouraging FAIR efforts
- uses 3 important elements for computational reproducibility (dynamic reporting, Git, containers)
- It is very flexible, adaptable to other tools, environments and needs

Considerations

- Find a balance between features and maintenance efforts
- The metadata CSV tables are the most important. The workflow just facilitates sharing in a central, public (or private) platform, version controlled and avoiding duplicates

Additional resources

Preprint detailing the workflow: https://osf.io/preprints/metaarxiv/64fch

High-level recommendations and ‘reality checks’ from AFFORD

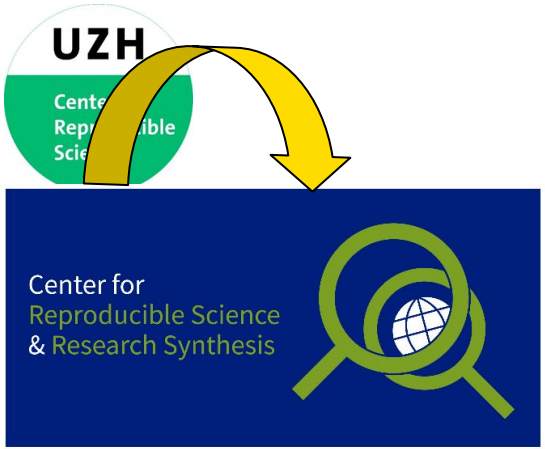
Fraga-González, G., van de Wiel, H., Garassino, F. *et al.* Affording reusable data: recommendations for researchers from a data-intensive project. *Sci Data* 12, 258 (2025). https://doi.org/10.1038/s41597-025-04565-0

Easy-reads for PIs: the CRS primers

- on software containers: https://doi.org/10.5281/zenodo.13757230
- on digital collaboration: https://doi.org/10.5281/zenodo.8354375

Thank you for your attention!

- Requests? Improvements? Write me! gorka.fragagonzalez@uzh.ch
- Follow us on LinkedIn for regular updates
- Check out the blog of SwissRN working group Open Research Data



The Center for Reproducible Science is now being transformed to a new **Center for Reproducibility and Research Synthesis** and made permanent
www.crs.uzh.ch