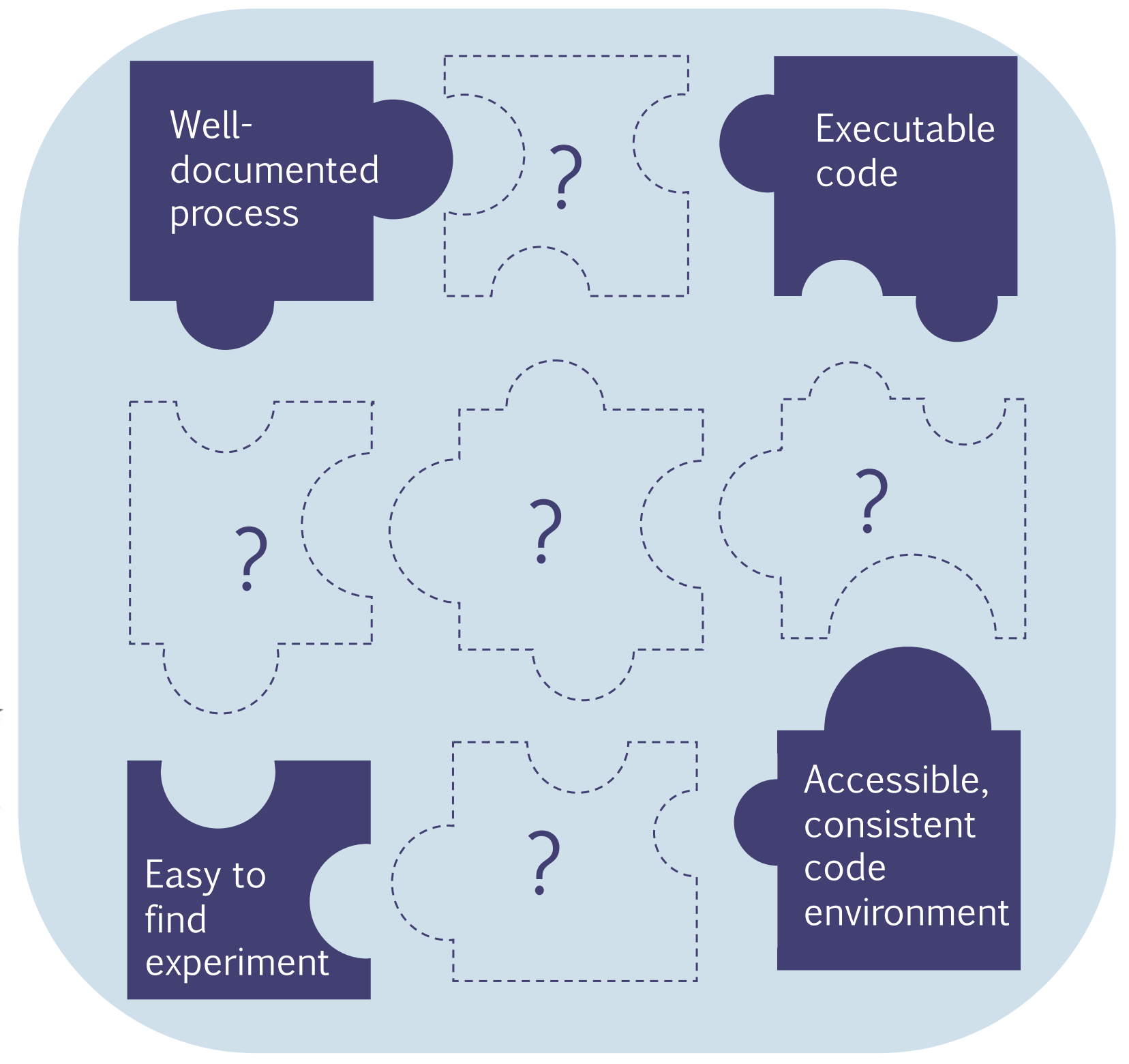
Sharing and Replicability of Notebook-Based Research on Open Testbeds

Maxine King[1], Jason Anderson[1], Kate Keahey[1,2]
[1]University of Chicago [2]Argonne National Laboratory

We seek to facilitate replicability by creating a way to share experiments easily in and out of notebook-based, open testbed environments and a sharing platform for such experiments in order to allow researchers to combine shareability, consistency of code environment, and well-documented process.

The Problem

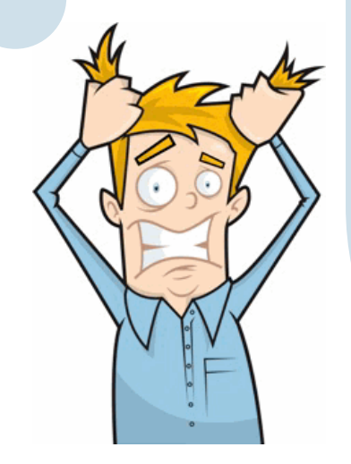
Ensuring that research can be reproduced, replicated, and repeated is the shield that prevents the scientific world from relying on falsehoods. However, as research has gotten more advanced, creating experiments that can be easily replicated has gotten to be quite complicated.

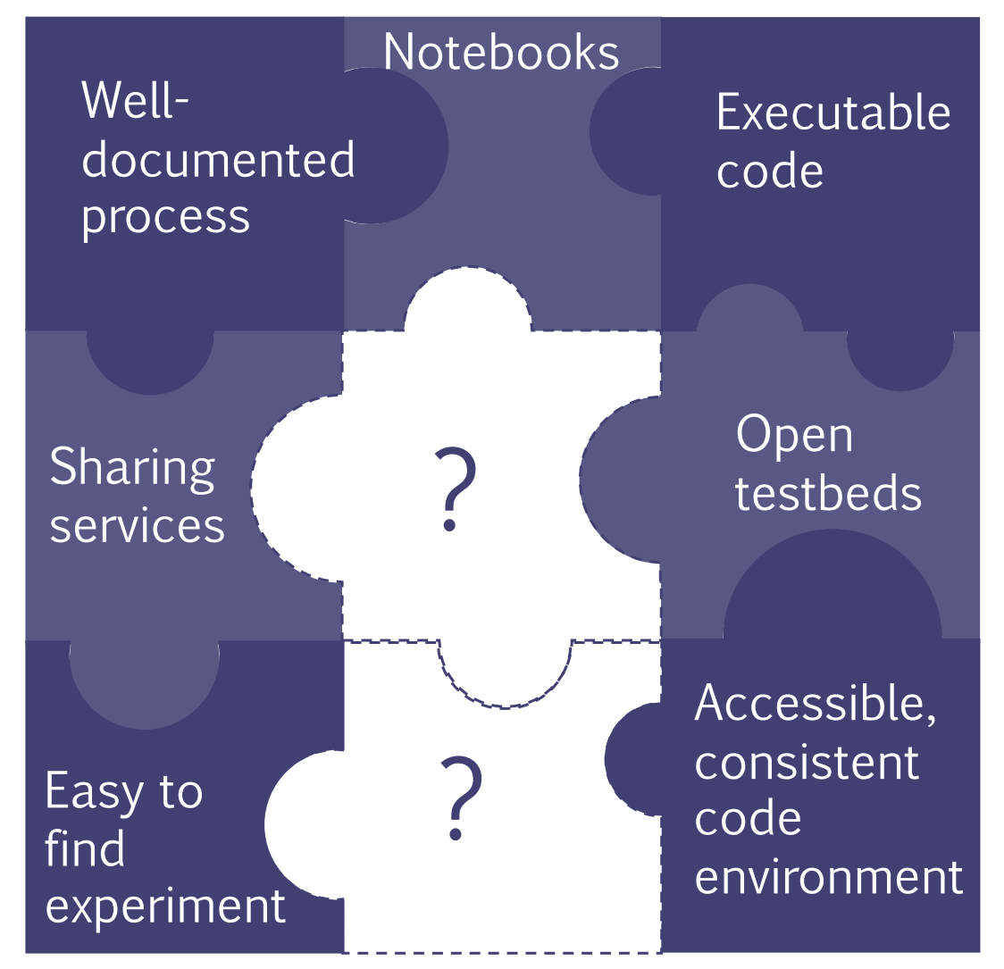


Researchers need to simultaneously…

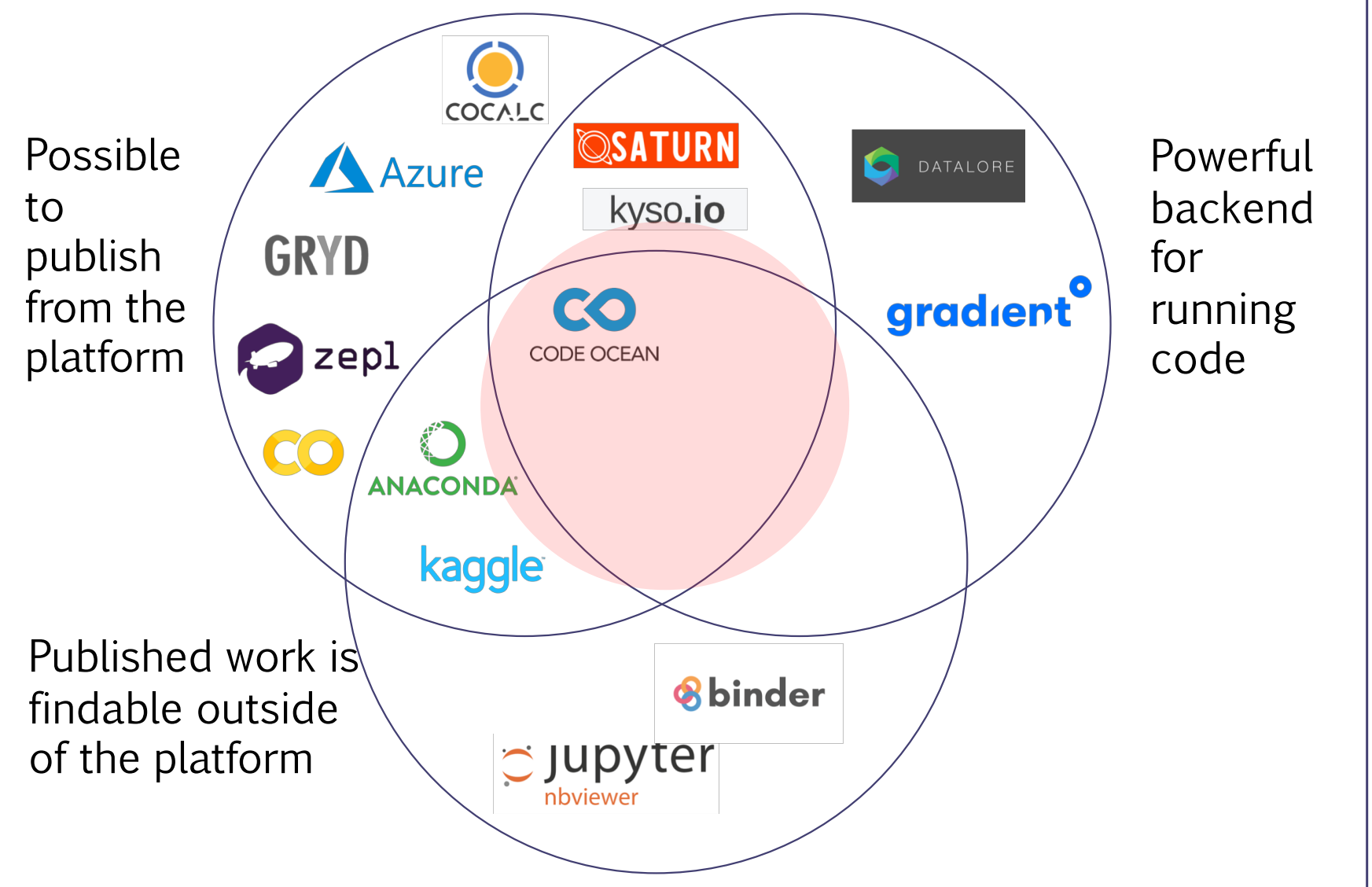
1. Run code in a consistent, customizable, powerful environment
2. Combine executable code with process documentation
3. Use a sharing platform where multiple people can put experiments

Up until now, we have had ways to connect two of these at a time, but not to make a full picture





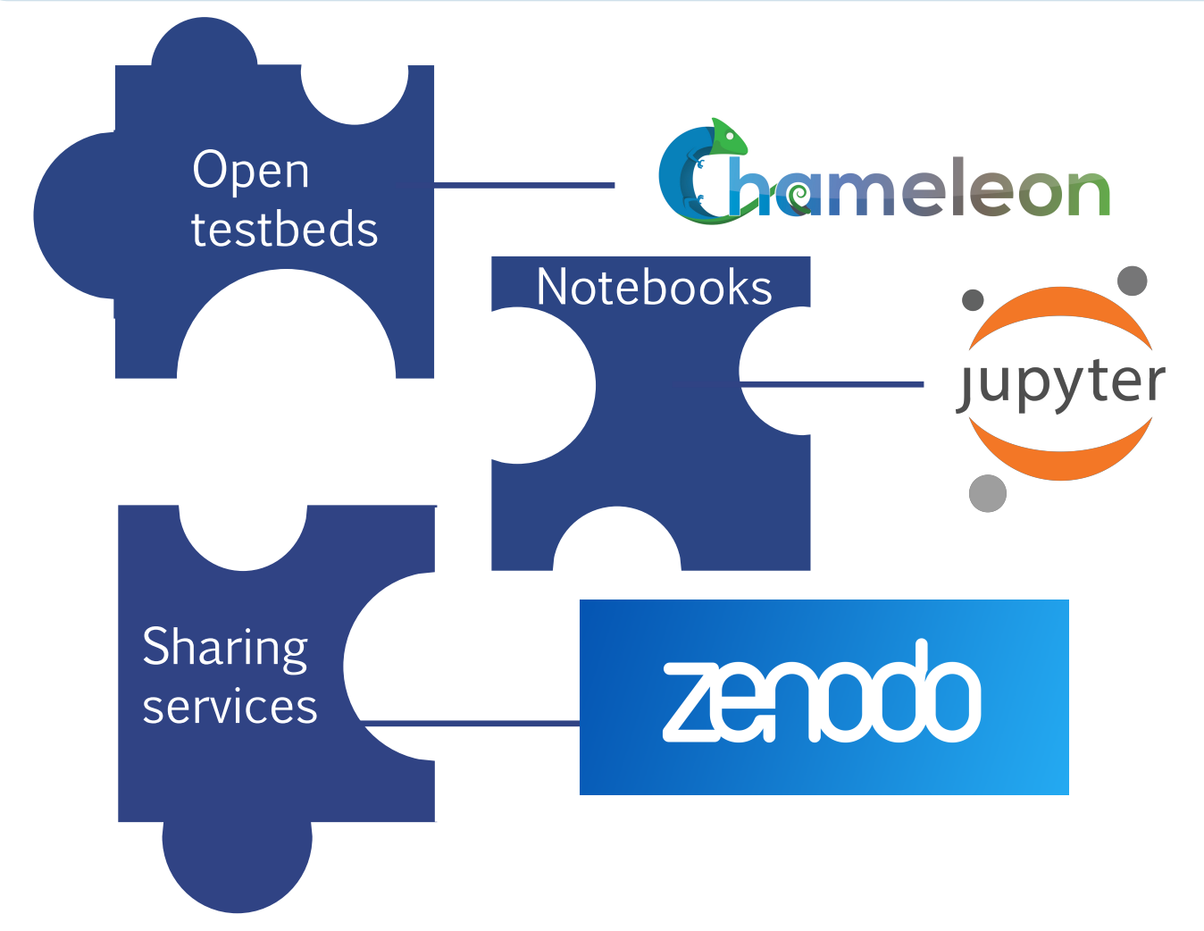
There are some existing solutions that aim to fill this gap, but very few have all the features a researcher would want


Possible to publish from the platform
Powerful backend for running code
Published work is findable outside of the platform
COCALC
Azure
SATURN
kyso.io
DATALORE
GRYD
CODE OCEAN
gradient
zepl
ANACONDA
kaggle
binder
jupyter nbviewer

The closest existing solution is CodeOcean, but it publishes only to its own research library and its export/import features cannot be added to other JupyterLab setups

We needed a place that would be widely accessible, easy to search, provides trustworthy storage, and would make it easy to share notebooks between an open testbed and a popular sharing site.

Our Pieces



Large-scale, bare-metal reconfigurable testbed, has significant hardware diversity that makes it capable of supporting a broad set of experiments

Uses markdown and Python, already integrated with the Chameleon testbed

Storage is backed by CERN, assigns each deposition a DOI, has no file size limit, already widely used

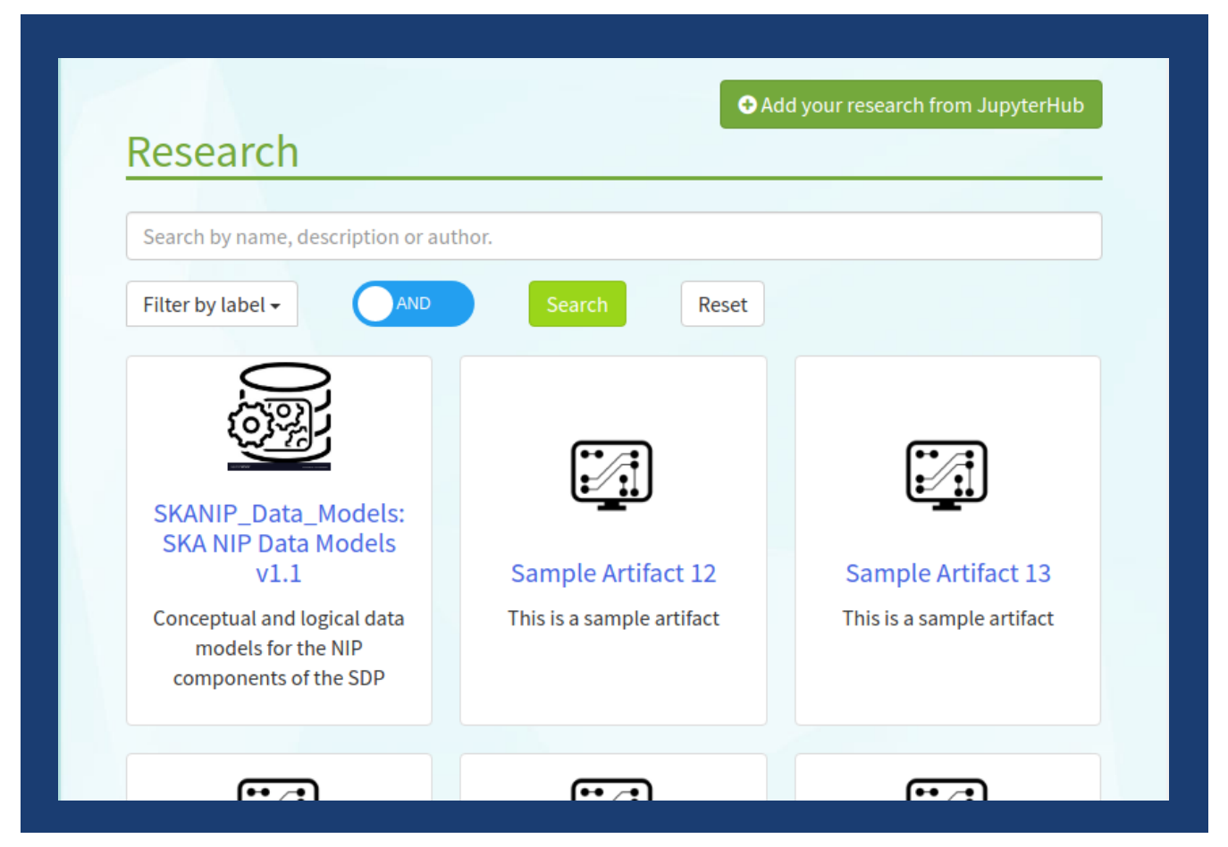
Our Solution

Based on these pieces, we created a three-part connection

1 A Sharing Platform

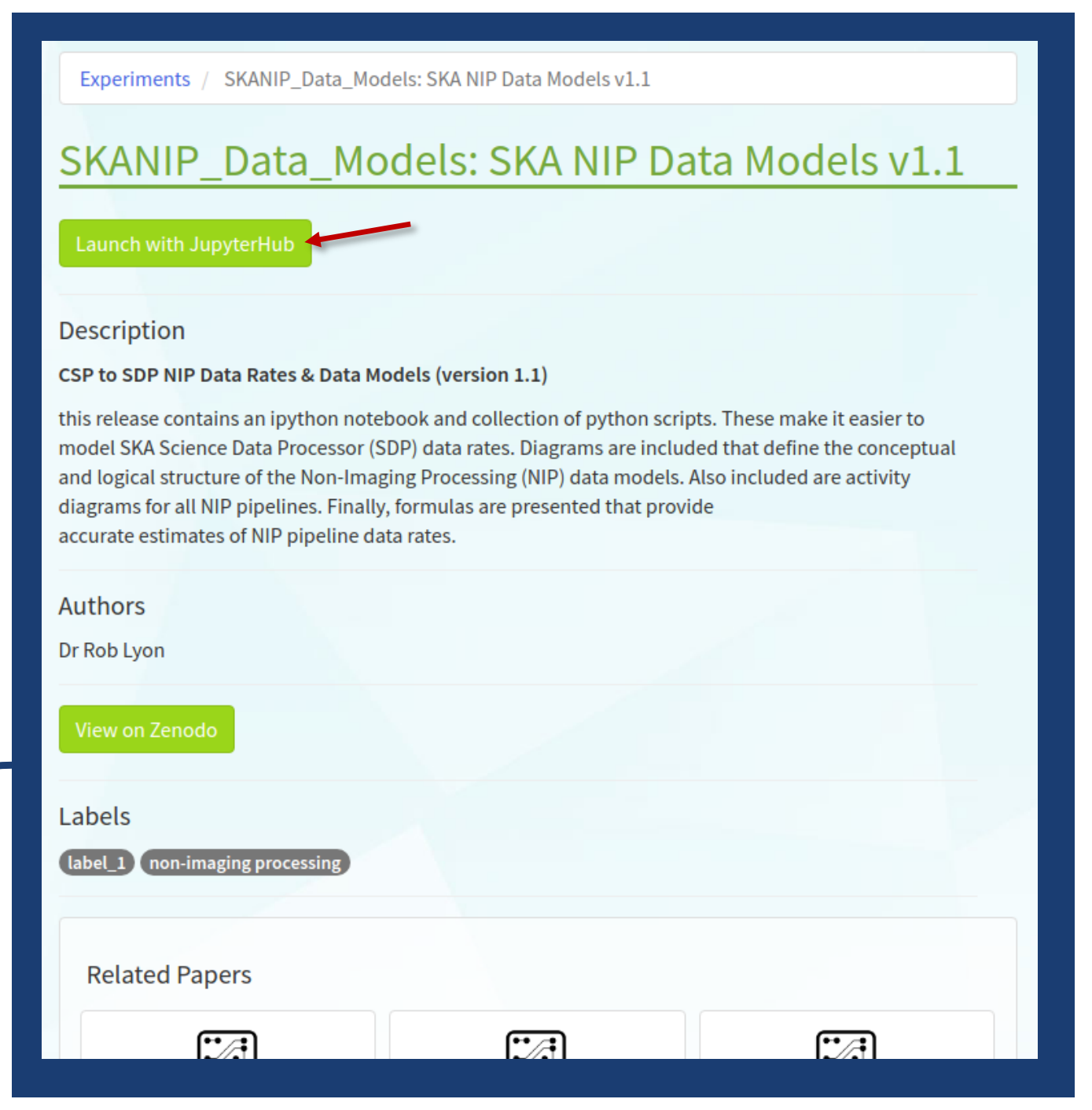
Chameleon-hosted web app that lets its users browse through others' research

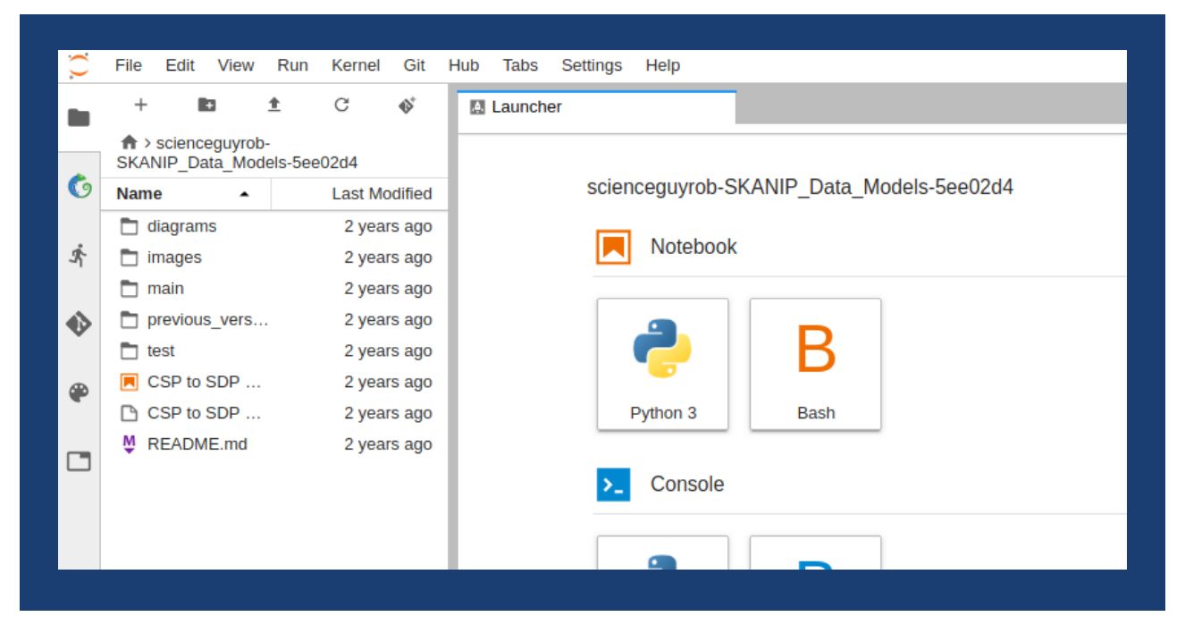
- Allows filtering by labels
- Searchable by keyword, author, or description
- Provides links to view items in Zenodo
- Acts as a one-stop shop for all Chameleon/Jupyter compatible research


Research
Add your research from JupyterHub
Search by name, description or author
Filter by label
Search
Reset
SKANIP_Data_Models: SKA NIP Data Models v1.1
Conceptual and logical data models for the NIP components of the SDP
Sample Artifact 12
This is a sample artifact
Sample Artifact 13
This is a sample artifact

2 An Import Plugin

- A simple button press pulls all files and setup info from Zenodo and lands the user in Chameleon's JupyterHub environment
- All files are pre-loaded, python requirements are pre-installed
- Server has a separate file system from researcher's main experiment server
- Live on Chameleon


Experiments / SKANIP_Data_Models: SKA NIP Data Models v1.1
SKANIP_Data_Models: SKA NIP Data Models v1.1
Launch with JupyterHub
Description
CSP to SDP NIP Data Rates & Data Models (version 1.1)
Authors
View on Zenodo
Labels
non-imaging processing
Related Papers



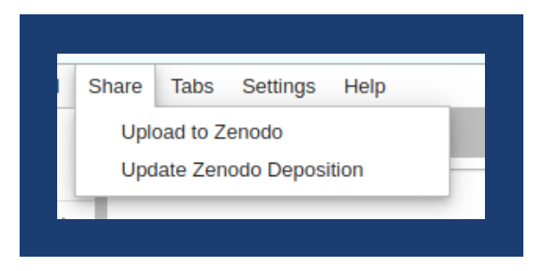
3 An Export Plugin

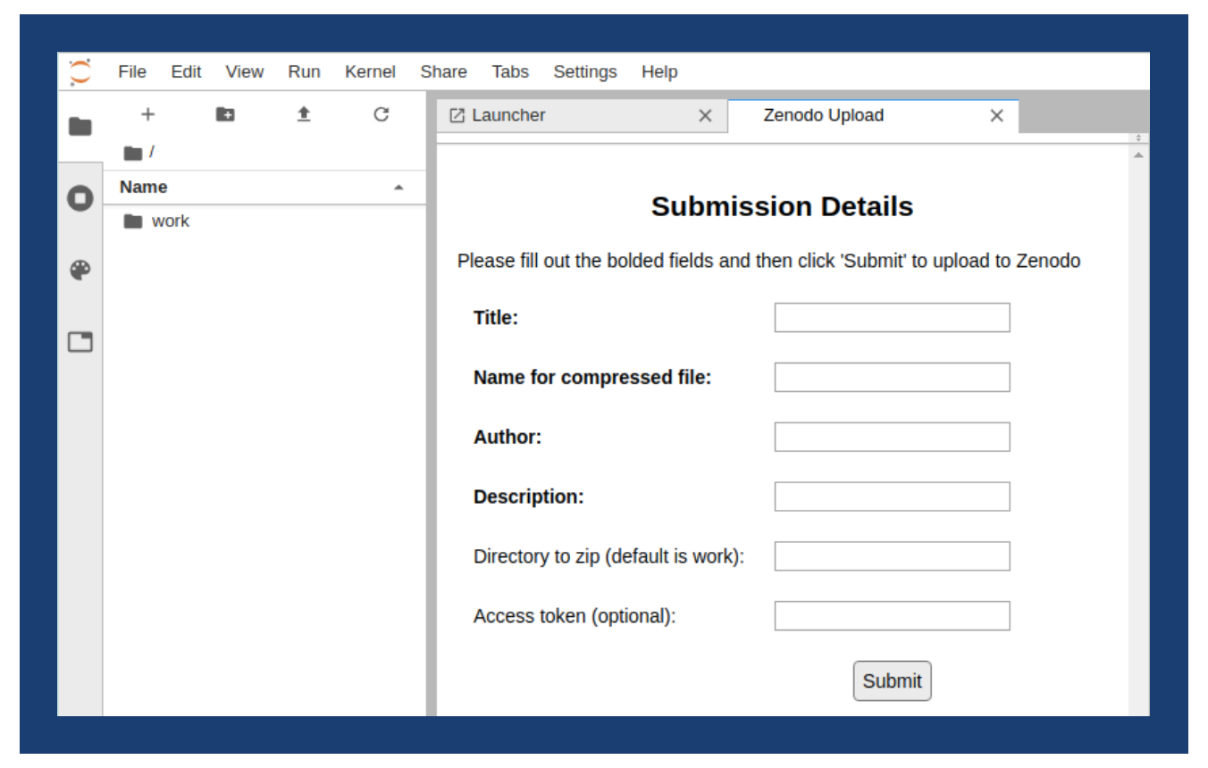
A Zenodo extension for JupyterHub allows users to publish their work directly

- No need to download files locally
- Users don't need to leave the JupyterHub environment to share their work

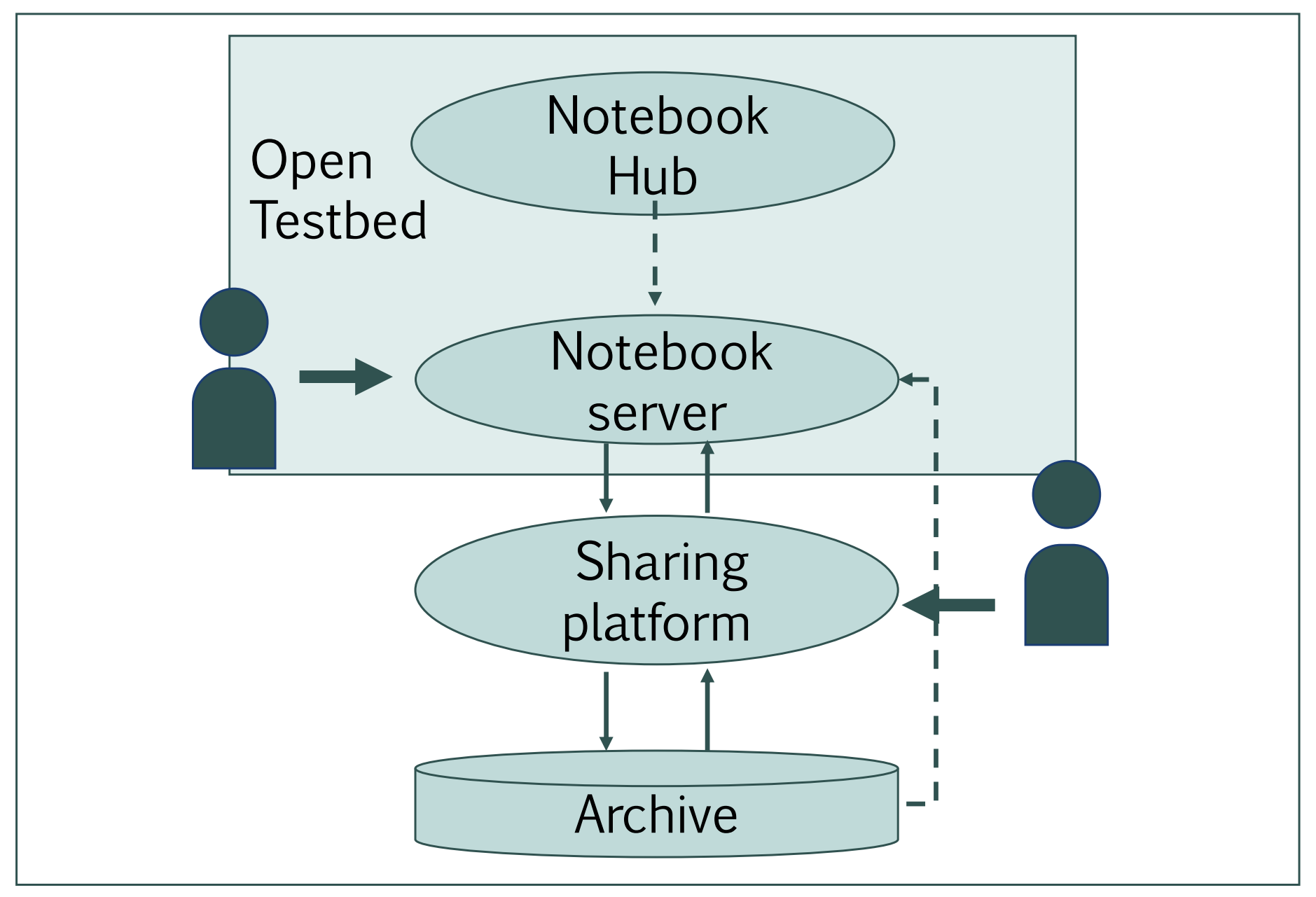
Artifacts are uploaded to Chameleon's sharing portal and archived on Zenodo in one click.

- Plugin can be installed via PyPi and NPM (https://github.com/ChameleonCloud/jupyterlab-zenodo) and is live on Chameleon




Submission Details
Please fill out the bolded fields and then click 'Submit' to upload to Zenodo
Title:
Name for compressed file:
Author:
Description:
Directory to zip (default is work):
Access token (optional):
Submit

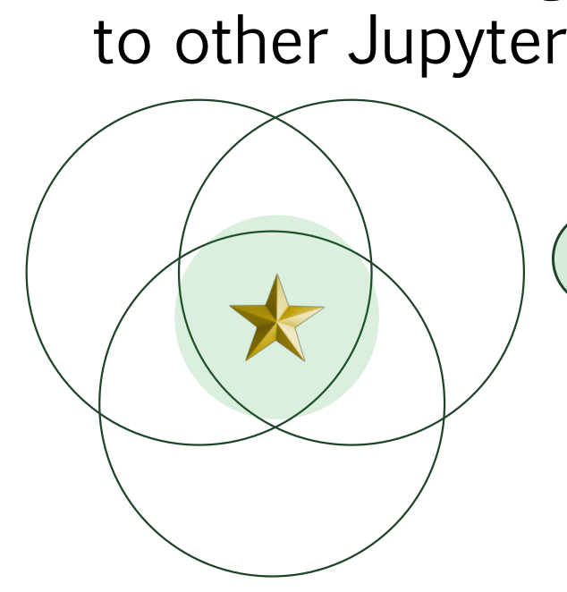
Architecture

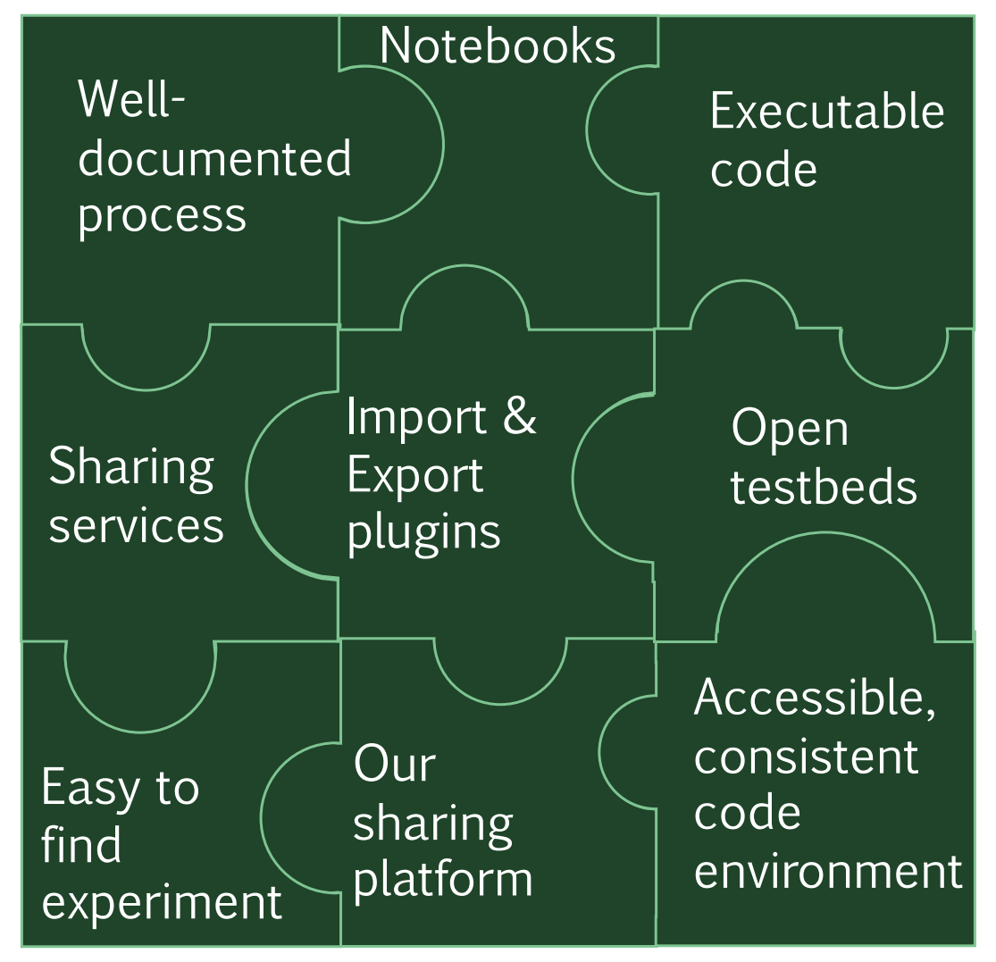


Evaluation

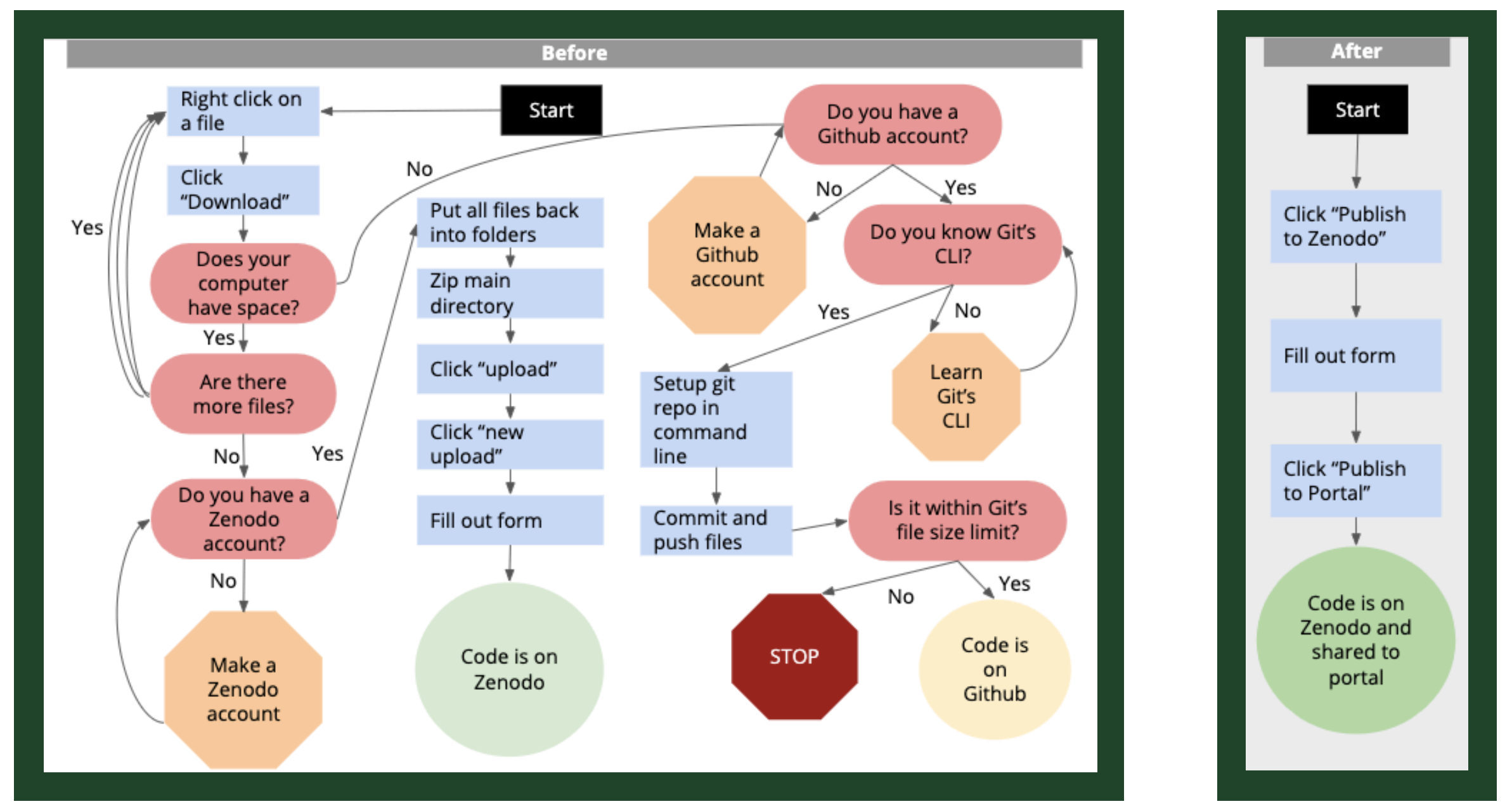
This solution…

- Combines guaranteed storage and wide popularity of Zenodo with searchability and having all relevant work in one place
- Makes getting code into the environment **easy**
- Makes sharing code out of this environment **easy**
- Includes sharing features that can be easily added to other JupyterLab environments

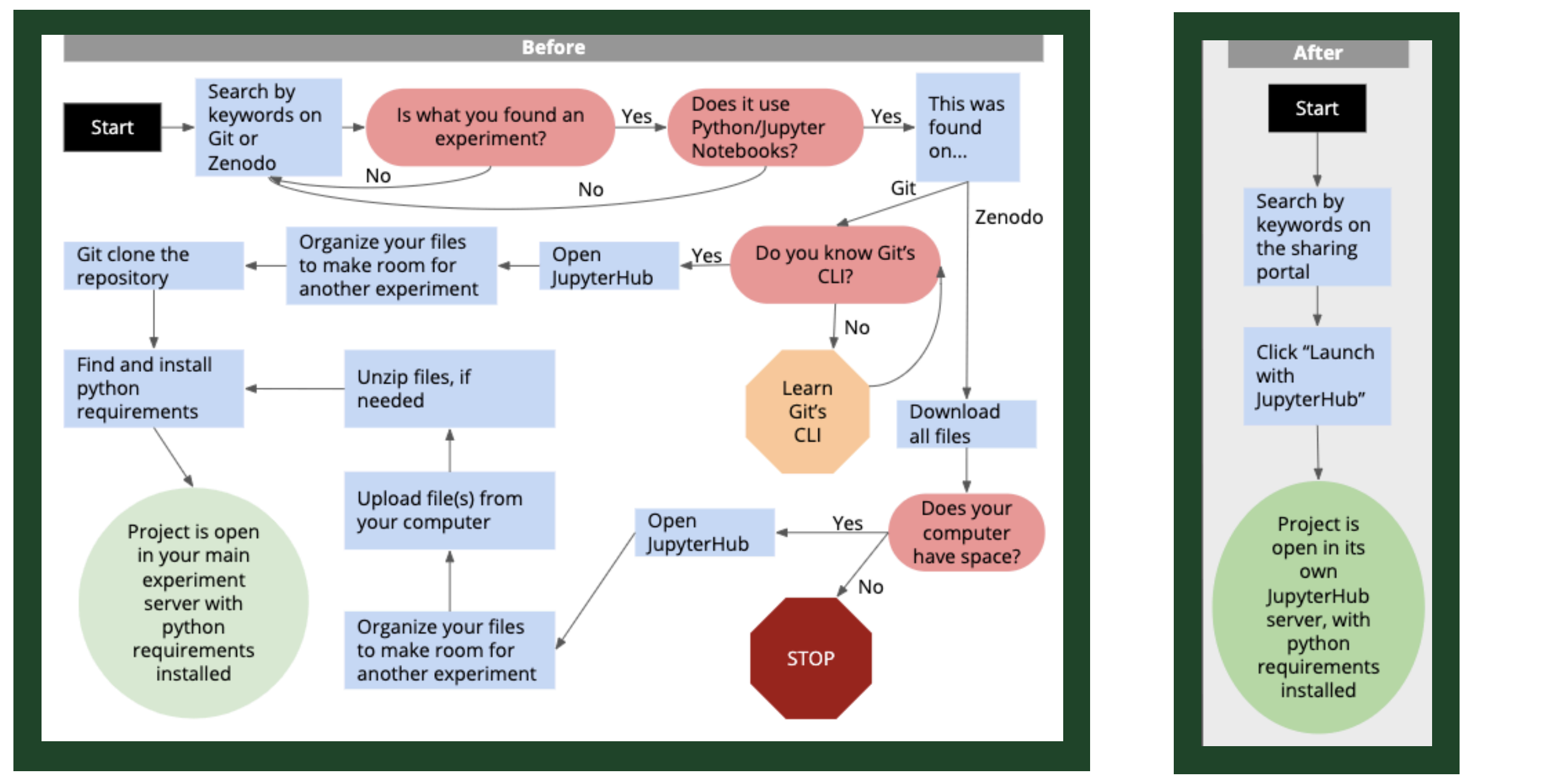




As a researcher, how do I share my code from a notebook in an open testbed?



As a researcher, how do I import code into an open testbed?


Before
Start
Search by keywords on Git or Zenodo
Is what you found an experiment?
No
Yes
Does it use Python/Jupyter Notebooks?
No
Yes
This was found on…
Git
Zenodo
Do you know Git's CLI?
Yes
No
Learn Git's CLI
Open JupyterHub
Organize your files to make room for another experiment
Git clone the repository
Find and install python requirements
Project is open in your main experiment server with python requirements installed
Download all files
Does your computer have space?
Yes
No
STOP
Open JupyterHub
Organize your files to make room for another experiment
Upload file(s) from your computer
Unzip files, if needed
After
Start
Search by keywords on the sharing portal
Click "Launch with JupyterHub"
Project is open in its own JupyterHub server, with python requirements installed

Conclusions

Was our hypothesis correct?

Yes: We were able to create a sharing platform that keeps relevant research in one place, is easy to search, and makes it easy to get code in and out in order to replicate computer science experiments using notebooks on an open testbed.

Current limitations

As of right now, this software is limited in several ways. It only supports one type of compressed file, it looks for requirements in only one location, it can only install pip requirements, and it requires users to manually add labels on the sharing platform.

Possible expansions

That said, these limitations exist only because of the relatively early stage of this project. With more time spent, all of these could easily be overcome. Additionally, it would be relatively easy to expand this solution to work with other notebooks, testbeds, and artifact storage solutions.