Operate First Data Science Community Meetup!

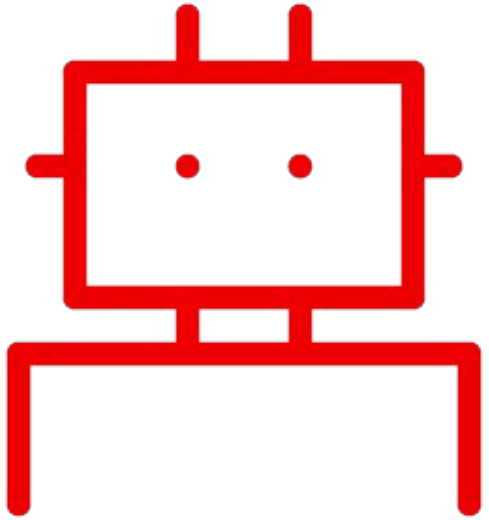
Michael Clifford

Red Hat


OPERATE FIRST
DATA SCIENCE COMMUNITY
a concept to incorporate operational experience into software projects
Ready to get started?

Data Science + Operate First

Operate First



Data Science

Links!

Website:
https://www.operate-first.cloud/data-science/

JupyterHub:
https://Jupyterhub-opf-jupyterhub.apps.smaug.na.operate-first.cloud

Meetup info:
https://www.operate-first.cloud/data-science/operate-first-data-science-community/docs/meetup-landing-page.md

Curren Projects:
https://www.operate-first.cloud/data-science/projectsoverview.md

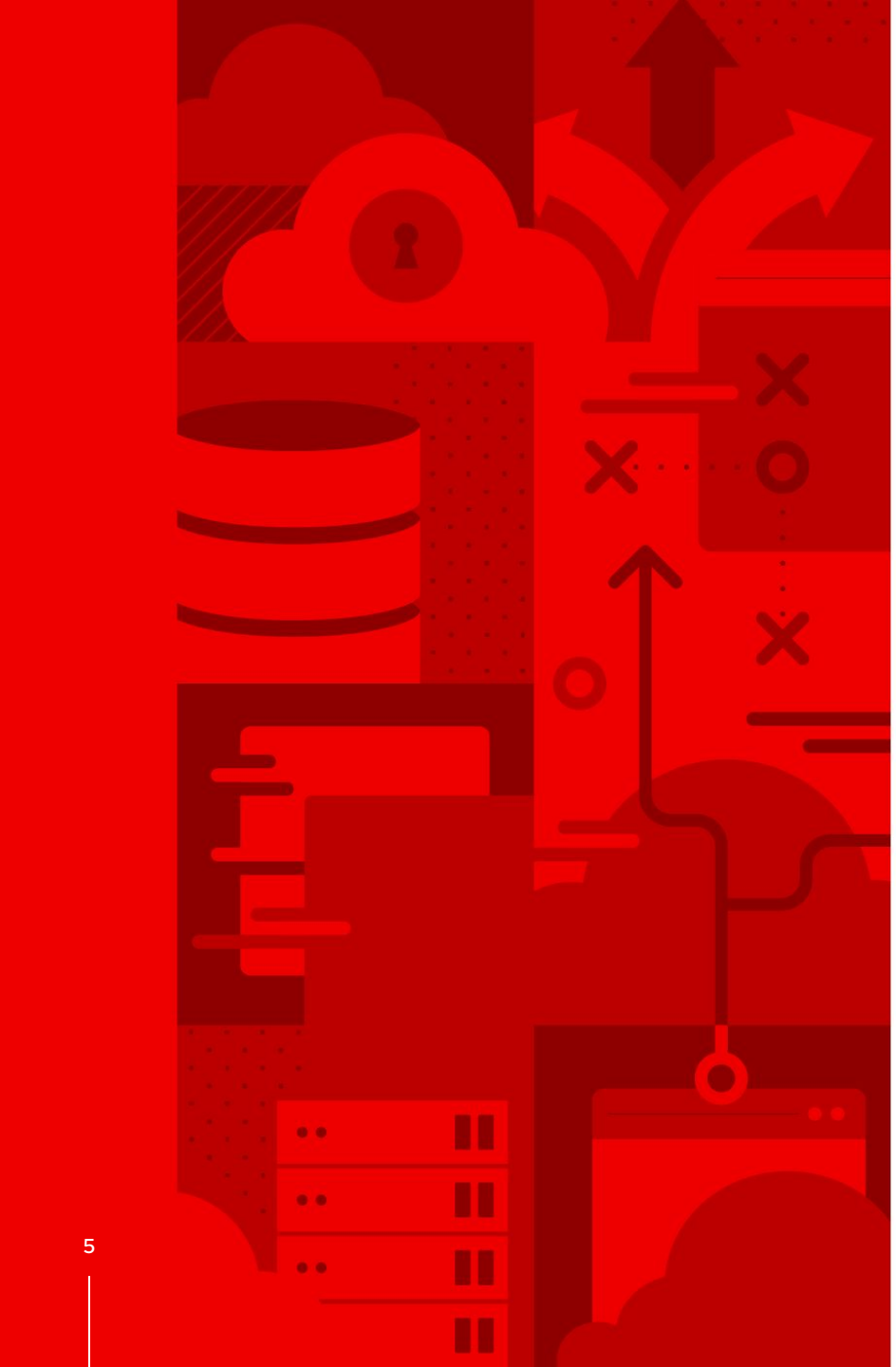
Slack:
https://join.slack.com/t/operatefirst/shared_invite/zt-o2gn4wn8-O39g7sthTAuPCvaCNRnLww

Youtube:
https://youtube.com/playlist?list=PL8VBRDTElCWqC5WcZUUNJxGn9WsIK6Ilb

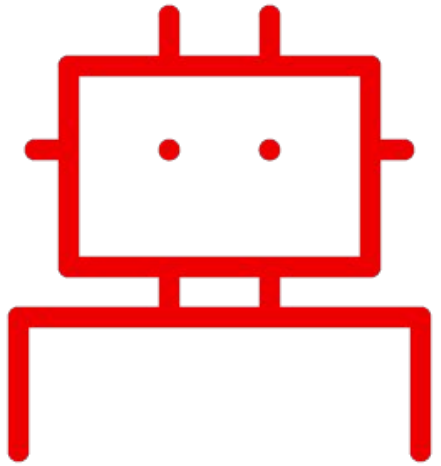
GitHub:
https://github.com/aicoe-aiops/cloud-first-data-science-community/issues/new/choose

Operate First

Red Hat



Thanks!



Red Hat

Reproducible Data Science with Operate First

Karan Chauhan
Data Scientist, Red Hat

Schedule

> Reproducibility and Shareability Challenges

> Solution with Operate-First + Thoth + AICoE-CI

> How-To with Examples

> Demo

> Q & A

How Reproducible and Shareable is Your Data Science Content?

- How are Python dependencies specified?
- Are there system-level requirements or initial setup?
- Does it need specific compute resources e.g. GPUs?

How to Share Your Data Science Content in a Reproducible Way

- How are Python dependencies specified?

  Pinned dependencies in Pipfile using Thoth recs

- Are there system-level requirements or initial setup?

  Project container image via AICoE-CI

- Does it need specific compute resources e.g. GPUs?

  Cloud environment via Operate-First

Python Dependencies

- Pinned requirements in a Thoth-advised Pipfile

```
(river-demo) ----------------------------------------------
~/Documents/playpen/river-demo » thamos add numpy pandas onnxruntime tensorflow
```

Pipfile

```
[[source]]
url = "https://pypi.org/simple"
verify_ssl = true
name = "pypi-org-simple"

[packages]
numpy = "*"
pandas = "*"
onnxruntime = "*"
tensorflow = "*"

[dev-packages]

[requires]
python_version = "3.8"

[thoth]
disable_index_adjustment = false

[thoth.allow_prereleases]
```

Python Dependencies

```
(river-demo)
~/Documents/playpen/river-demo » thamos advise
```

Application stack guidance

Link	Message	Type
https://thoth-station.ninja/j/cve_timestamp	CVE database of known vulnerabilities for Python packages was updated at '2021-12-11T00:37:34.592440'	✔INFO
https://thoth-station.ninja/j/thoth_s2i	It is recommended to use Thoth's s2i to have recommendations specific to runtime environment	✔INFO
https://thoth-station.ninja/j/rhel_ubi	Using observations for RHEL instead of UBI, RHEL is ABI compatible with UBI	⚠WARNING

Recommended stack report

Link	Message	Package name	Type
https://github.com/thoth-station/s2i-thoth/	See S2I Thoth base images and their docs for more info	-	✔INFO
https://stackoverflow.com/questions/tagged/python-3.8	See StackOverflow tags specific for Python 3.8	-	✔INFO
https://thoth-station.ninja/recommendation-types/	The recommended software stack was computed based on the 'latest' recommendation type	-	✔INFO
https://libraries.io/pypi/flatbuffers/	Information about 'flatbuffers' on libraries.io	flatbuffers	✔INFO
https://thoth-station.ninja/j/no_cve	No known CVE known for 'flatbuffers' in version '2.0'	flatbuffers	✔INFO
https://pypi.org/project/flatbuffers/2.0/	Package 'flatbuffers' in version '2.0' is released on PyPI	flatbuffers	✔INFO

Python Dependencies

```
(river-demo)
~/Documents/playpen/river-demo » ls
demo.ipynb  Pipfile  Pipfile.lock
```

Pipfile.lock

```
        "onnxruntime": {
            "hashes": [
                "sha256:2d8eb89d4d62ba956f5b3392b0a02e50dfa3ba6e255561c16e3fd10c
                "sha256:aa7346af63eade9a041b79363b2be04a60e4a5d4fe82bce4de2b1cdc
                "sha256:2201b12c736c7c1ce683289ff012a768dc6ea0eeb81c9276fde31c86
                "sha256:579a38f5abee3e89684ee223701d62409be20742767356d982e92cf2
                "sha256:417d09475d3928b224225e4bd1a6aa8ce061de7458ccda2b86dfe125
                "sha256:7395d86c1c18c5e191f6d7cfe79ecc590fe8ed30f4b01a65e1da6c6c
                "sha256:c140299b0f94f4a3cdf872308b21941622b973868f491a37e7f3148f
                "sha256:1c2cf7d5fe89e74f363596d90b952c367653763f16655a8b5060c7de
                "sha256:2df2999fdc0f1f5bcc87365b93fd951adbcdb615d312b1eb69aa3ffa
                "sha256:923721385a33d681d20878a090ae9e9e653c40eaa57665a4a87ddbcc
                "sha256:695e9badeb538bae87e25ee17c330c50a32114ebe6eae9bf2da03d66
                "sha256:8fd9200c327042359ff733739ca4d0e2491d52f540b31a3fa29e2805
                "sha256:89fa6e392637c94c17e693772294ed9afe2403832f78c5efa1ccd79f
                "sha256:a7a98e820b6a2d9a23908efad28ce4b0c7181e897273e70daa22b611
                "sha256:affda6e92c4de3a3d7b35c8da35dedf8fc618b17638f719bb5398acc
                "sha256:cac8e35f953ff9898f0b5fdca0b76f492ae508e99e2eeb02ea70339e
            ],
            "index": "pypi-org-simple",
            "version": "==1.8.1"
        },
        "pandas": {
            "hashes": [
                "sha256:68408a39a54ebadb9014ee5a4fae27b2fe524317bc80adf56c9ac59e
```

How to Share Your Data Science Content in a Reproducible Way

- How are Python dependencies managed?

  Pinned dependencies in Pipfile using Thoth recs

- Are there system-level requirements or initial setup?

  Project container image via AICoE-CI

- Does it need specific compute resources e.g. GPUs?

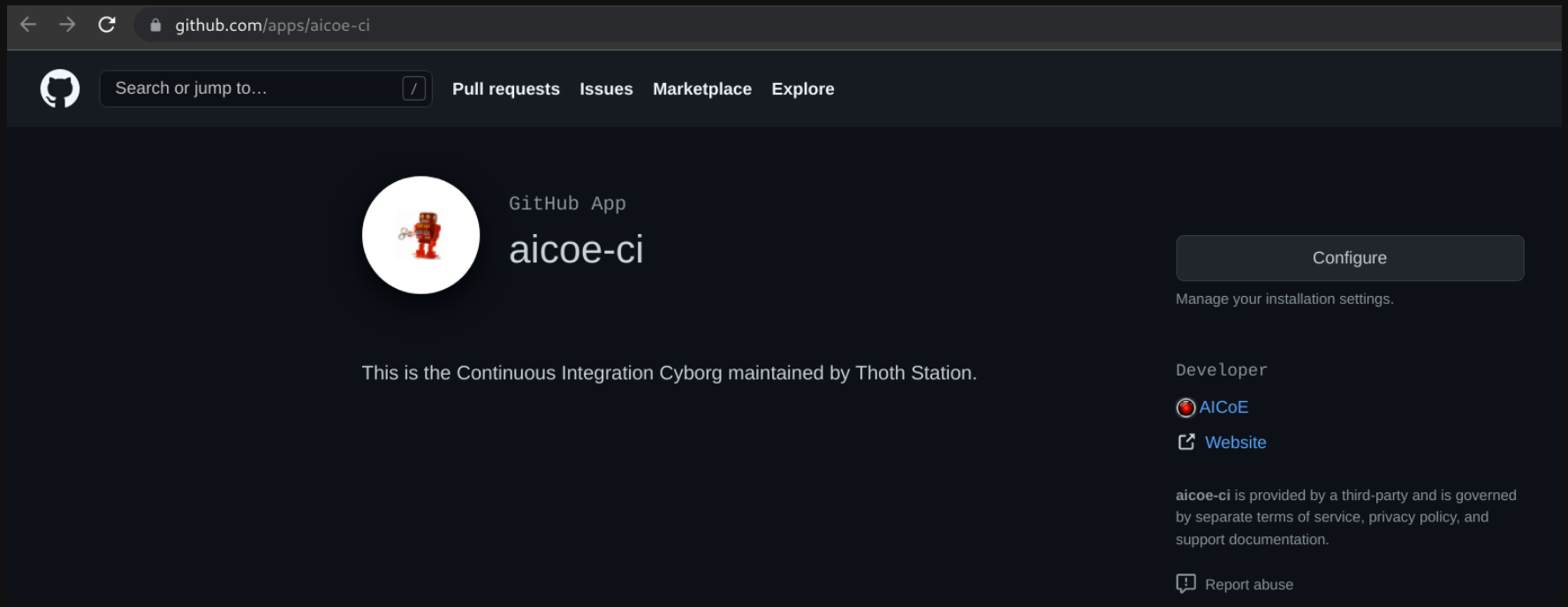
  Cloud environment via Operate-First

Project Container Image via AICoE-CI

- Package system libraries, initial setup, etc. into container image described via s2i scripts or Dockerfile
- Trigger image build automatically upon tag release or by creating issue
- If repo is within operate-first / aicoe orgs, can use already set up instance, else can host your own
- Set up in [less than] ***five*** easy steps

Container Image using aicoe-ci

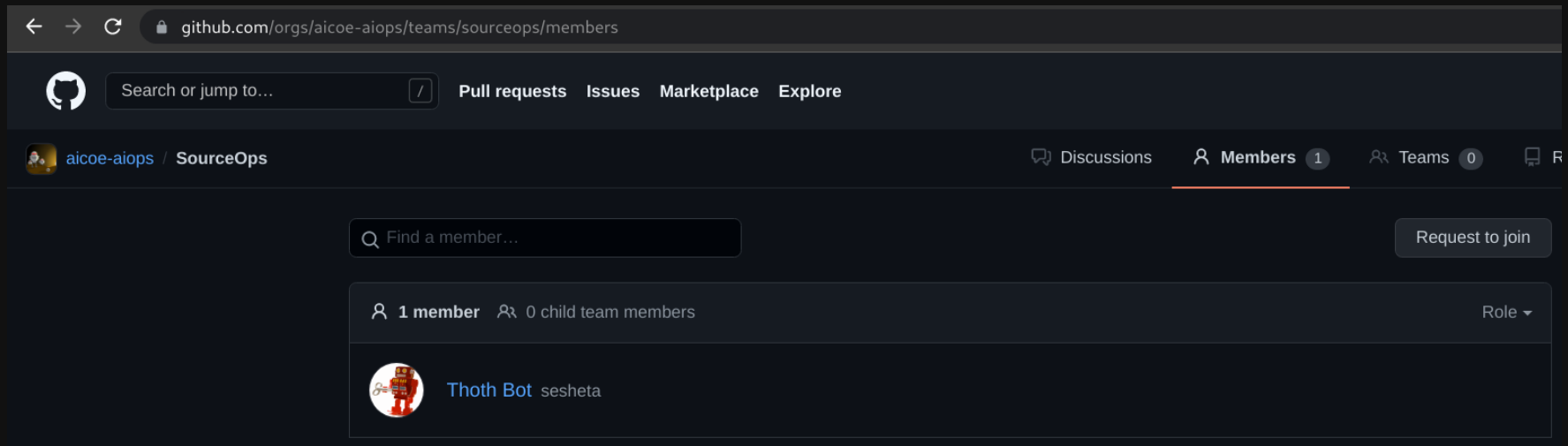
Step 1: Install aicoe-ci github app in your org/repo



Note: already done for repos within operate-first / aicoe orgs

Container Image using aicoe-ci

Step 2: Add sesheta as a collaborator to your org/repo



Note: already done for repos within operate-first / aicoe orgs

Container Image using aicoe-ci

Step 3: Add .aicoe-ci.yaml config file to your repo

```yaml
check:
  - thoth-build

build:
  custom-tag: latest
  registry: quay.io
  registry-org: aicoe
  registry-project: ocp-ci-analysis
  registry-secret: aicoe-pusher-secret
  build-stratergy: Source
  build-source-script: "image:///opt/app-root/builder"
  base-image: quay.io/thoth-station/s2i-elyra-custom-notebook:latest
```

Example config for s2i build

Container Image using aicoe-ci

Step 3: Add .aicoe-ci.yaml config file to your repo

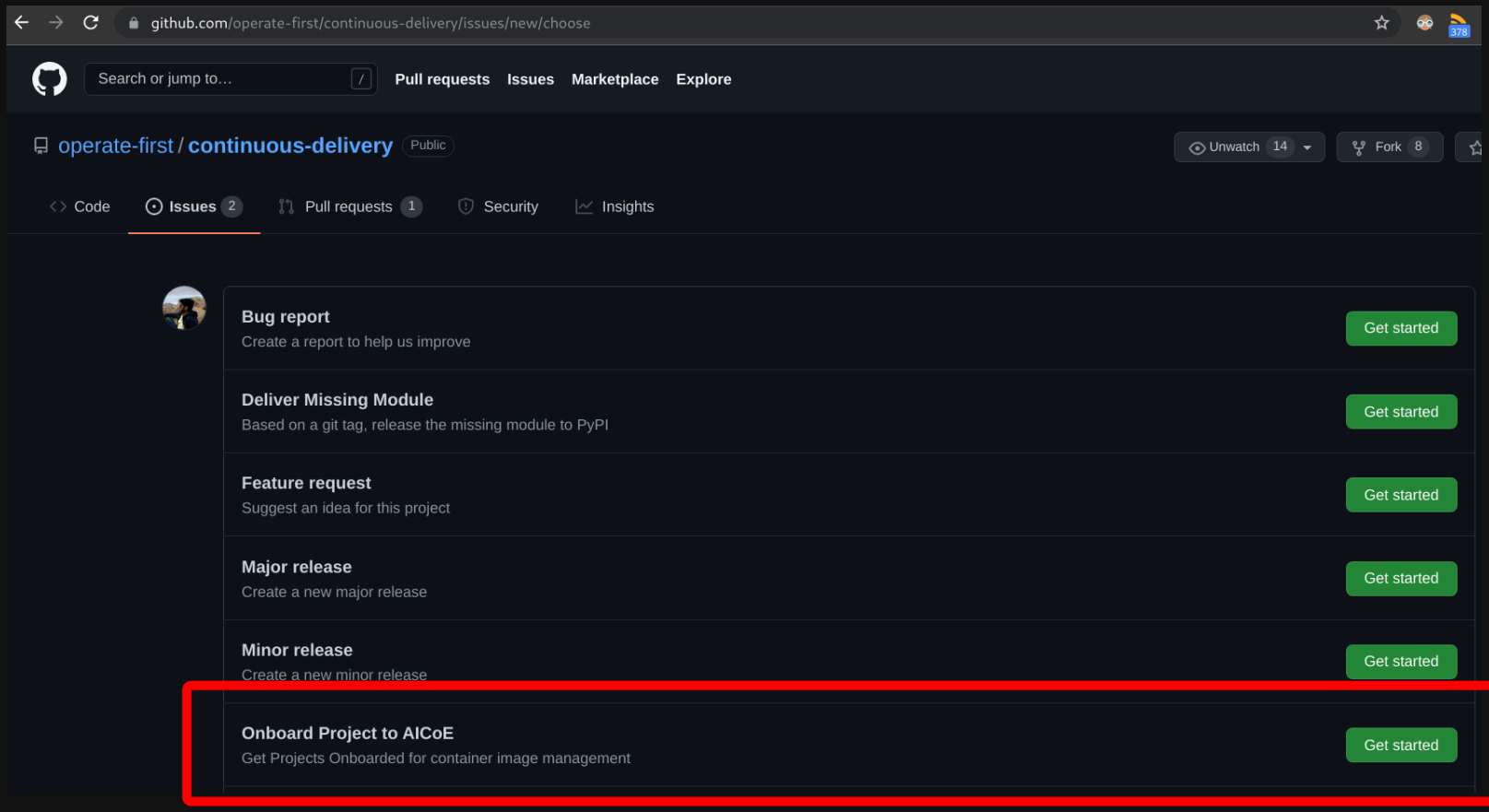
```yaml
check:
  - thoth-build

build:
  custom-tag: latest
  registry: quay.io
  registry-org: os-climate
  registry-project: aicoe-osc-demo
  registry-secret: os-climate-pusher-secret
  build-stratergy: Dockerfile
  dockerfile-path: Dockerfile
```

Example config for Dockerfile build

Container Image using aicoe-ci

Step 4: Open issue on operate-first/continuous-delivery

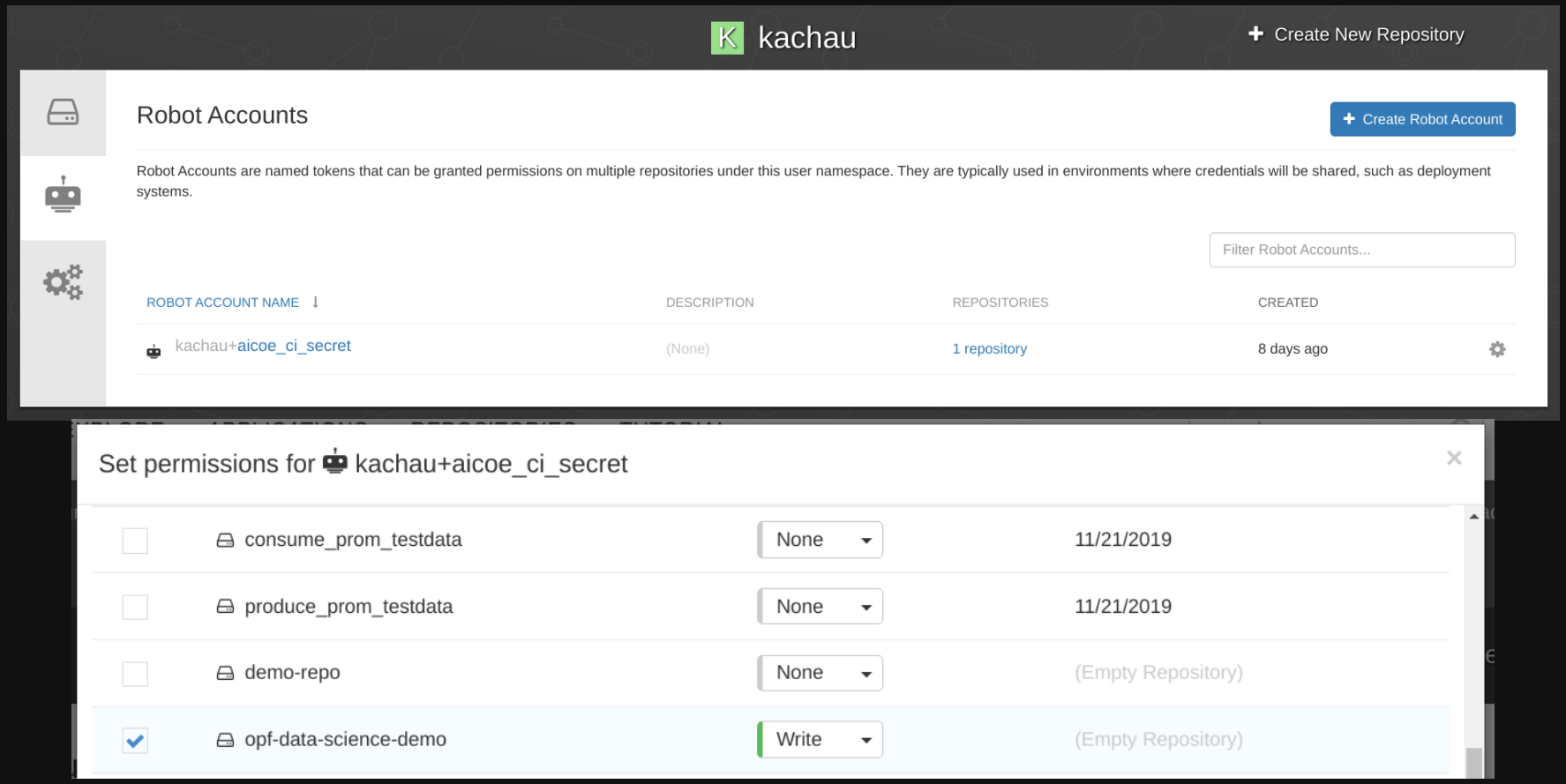
(in case of aicoe-ci instance on operate-first / aicoe orgs)



Container Image using aicoe-ci

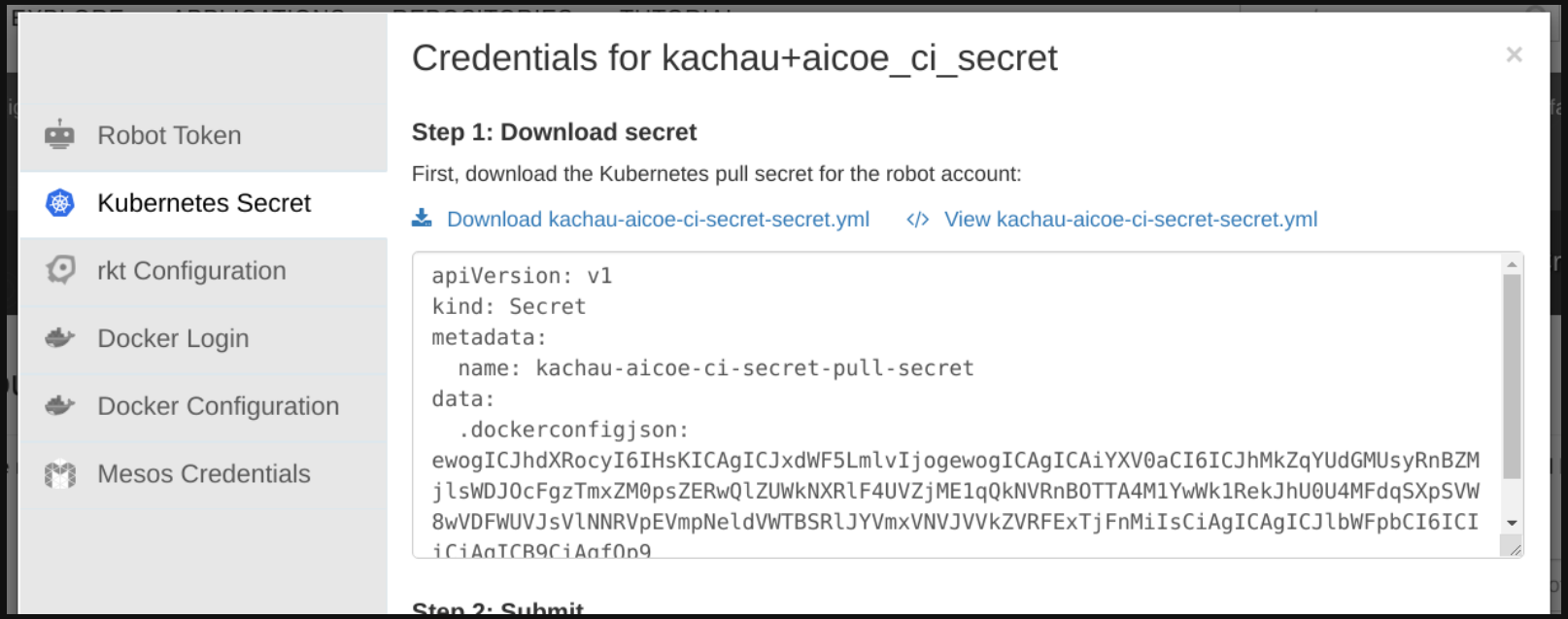
Step 4: Create quay repo + robot account with write access

(in case of your own aicoe-ci instance)



Container Image using aicoe-ci

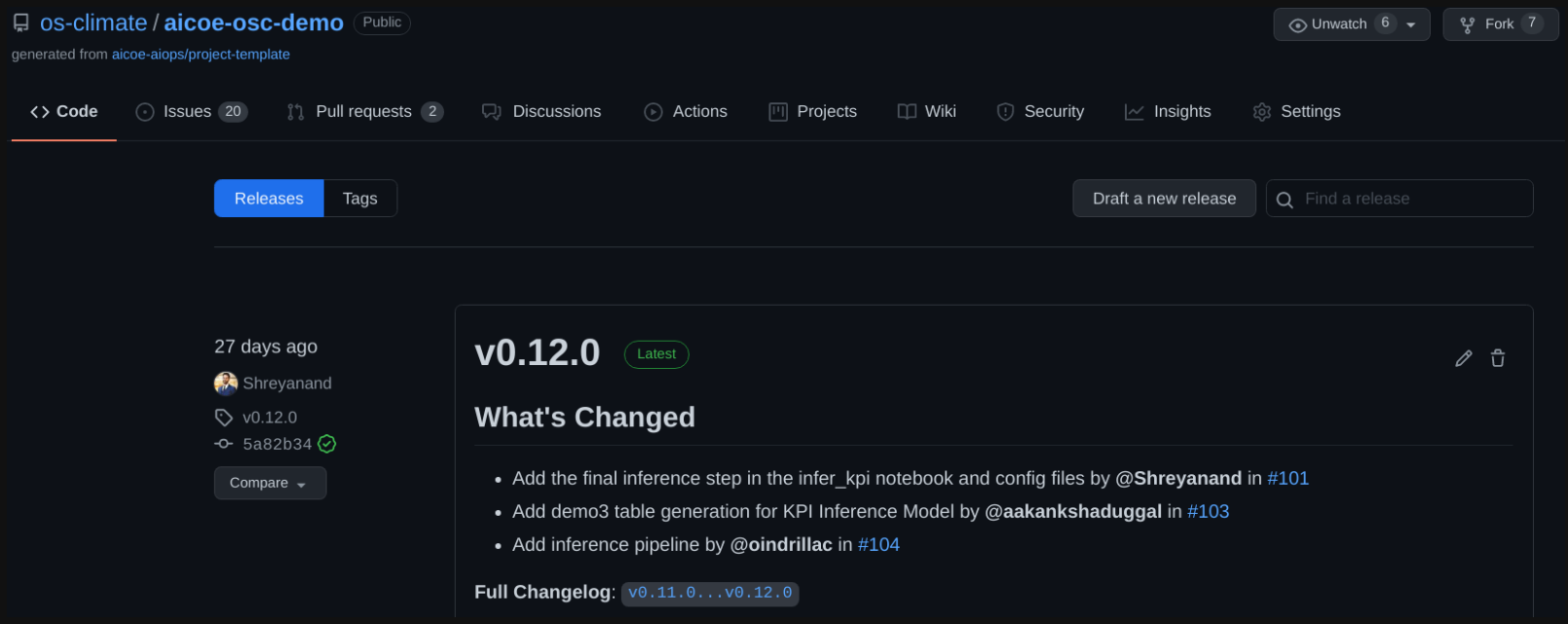
Step 5: Add the k8s secret to aicoe-ci instance

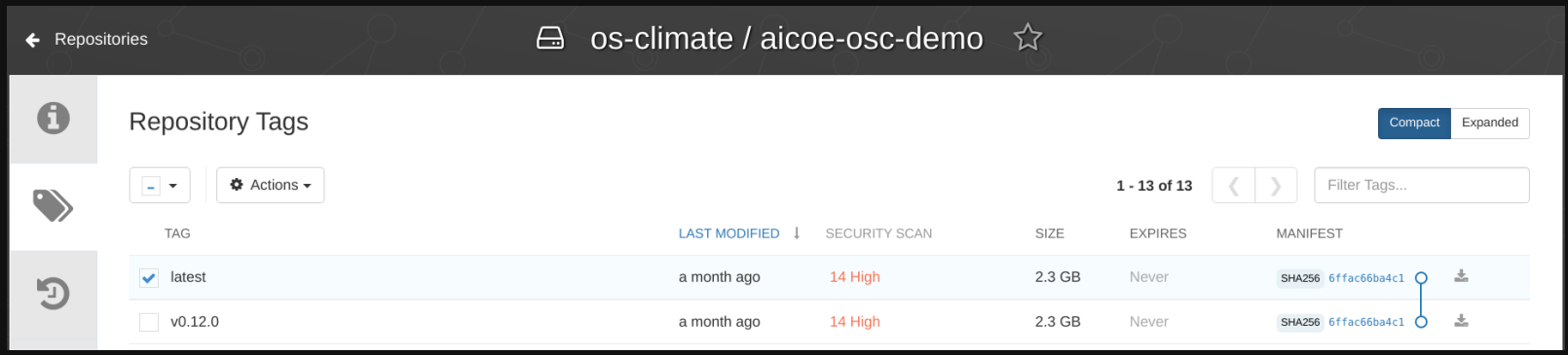


Note: already done by issue opened in previous step, for repos within operate-first / aicoe orgs

Container Image using aicoe-ci

Result: Image created on every release





How to Share Your Data Science Content in a Reproducible Way

- How are Python dependencies managed?

  Pinned dependencies in Pipfile using Thoth recs

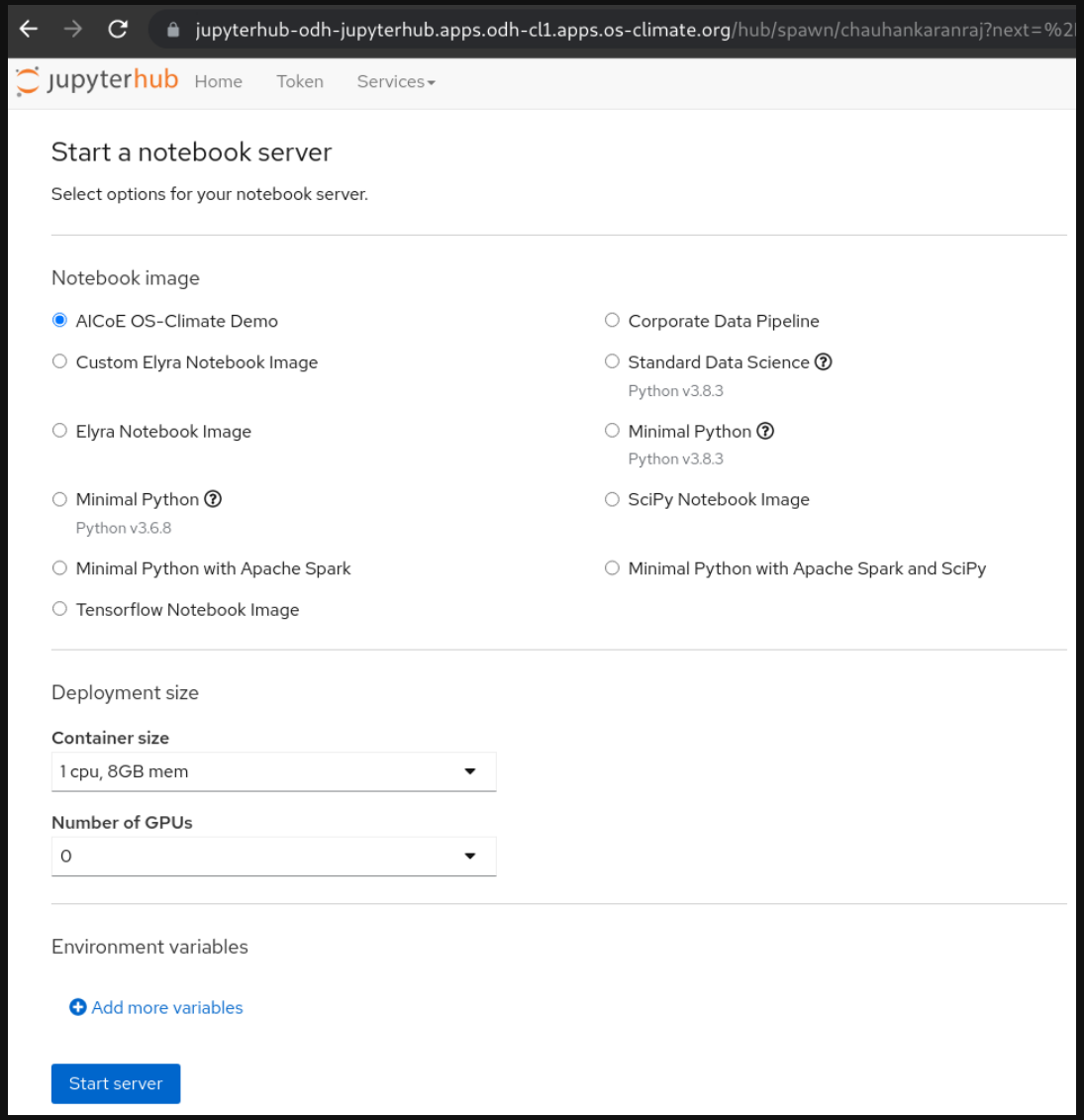
- What about system-level requirements?

  Project container image via AICoE-CI

- Does it need specific compute resources e.g. GPUs?

  Cloud environment via Operate-First

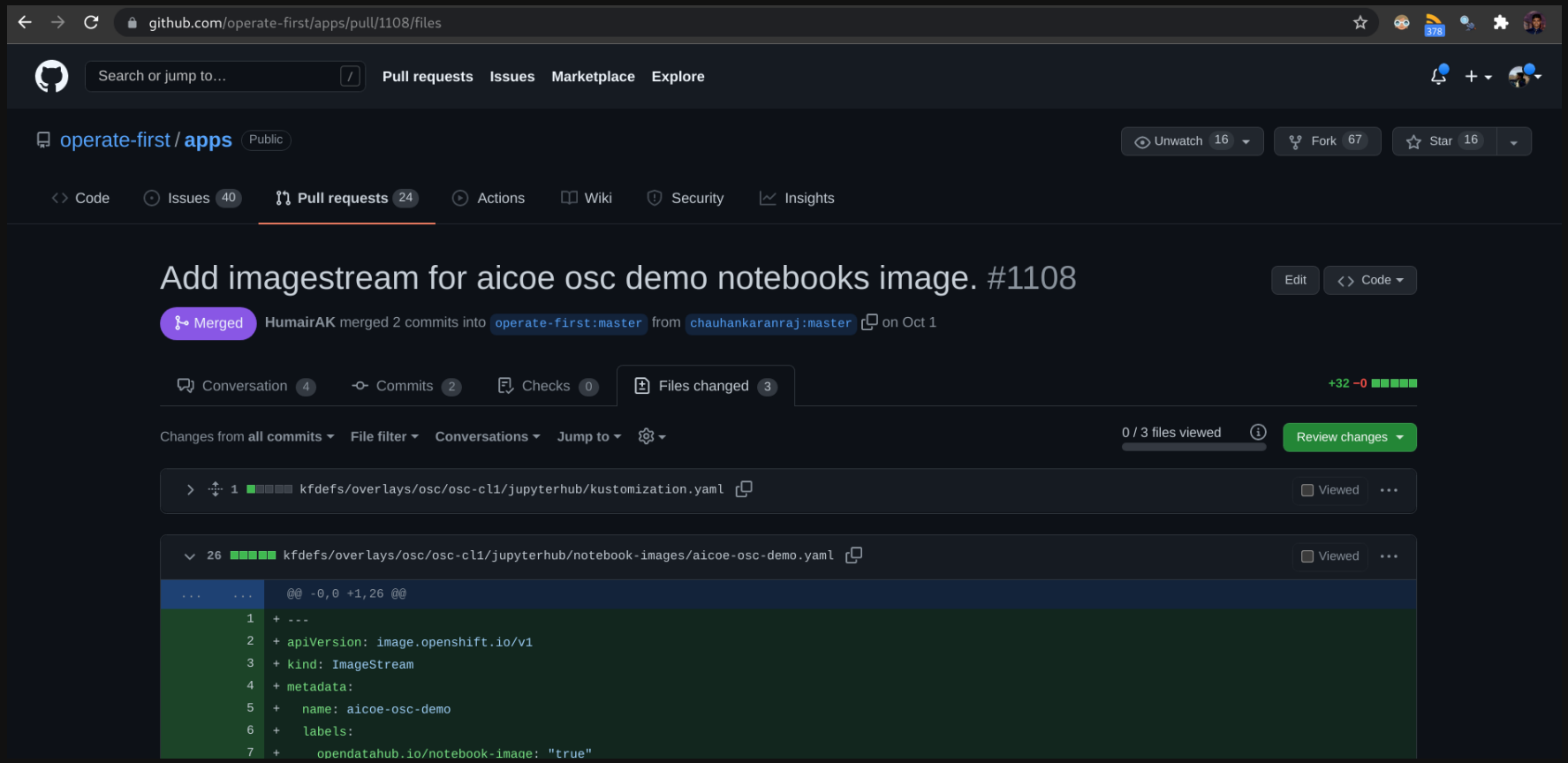
Share via JupyterHub



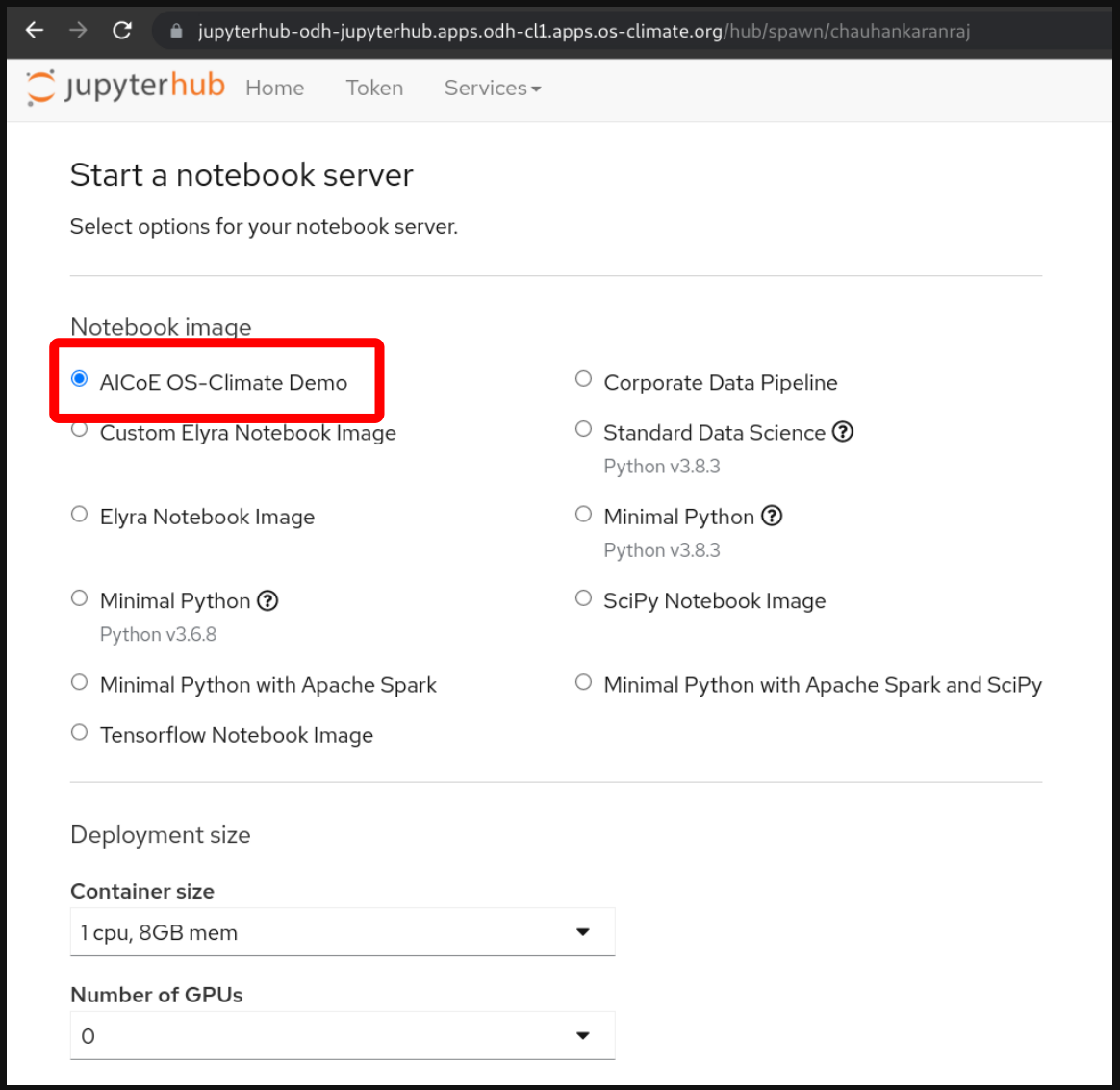
Add to JupyterHub

```yaml
---
apiVersion: image.openshift.io/v1
kind: ImageStream
metadata:
  name: aicoe-osc-demo
  labels:
    opendatahub.io/notebook-image: "true"
  annotations:
    opendatahub.io/notebook-image-url:
      "https://github.com/os-climate/aicoe-osc-demo"
    opendatahub.io/notebook-image-name:
      "AICoE OS-Climate Demo"
    opendatahub.io/notebook-image-desc:
      "Jupyter notebooks for the Nimbus and the Cirrus demos for OS-Climate"
spec:
  lookupPolicy:
    local: true
  tags:
    - name: "latest"
      from:
        kind: DockerImage
        name: quay.io/os-climate/aicoe-osc-demo:latest
      annotations:
        openshift.io/imported-from: quay.io/os-climate/aicoe-osc-demo
      importPolicy:
        scheduled: true
```

Add to JupyterHub



Add to JupyterHub



How to Share Your Data Science Content in a Reproducible Way

- How are Python dependencies managed?

  Pinned dependencies in Pipfile using Thoth recs

- What about system-level requirements?

  Project container image via AICoE-CI

- Does it need specific compute resources e.g. GPUs?

  Cloud environment via Operate-First

Recap

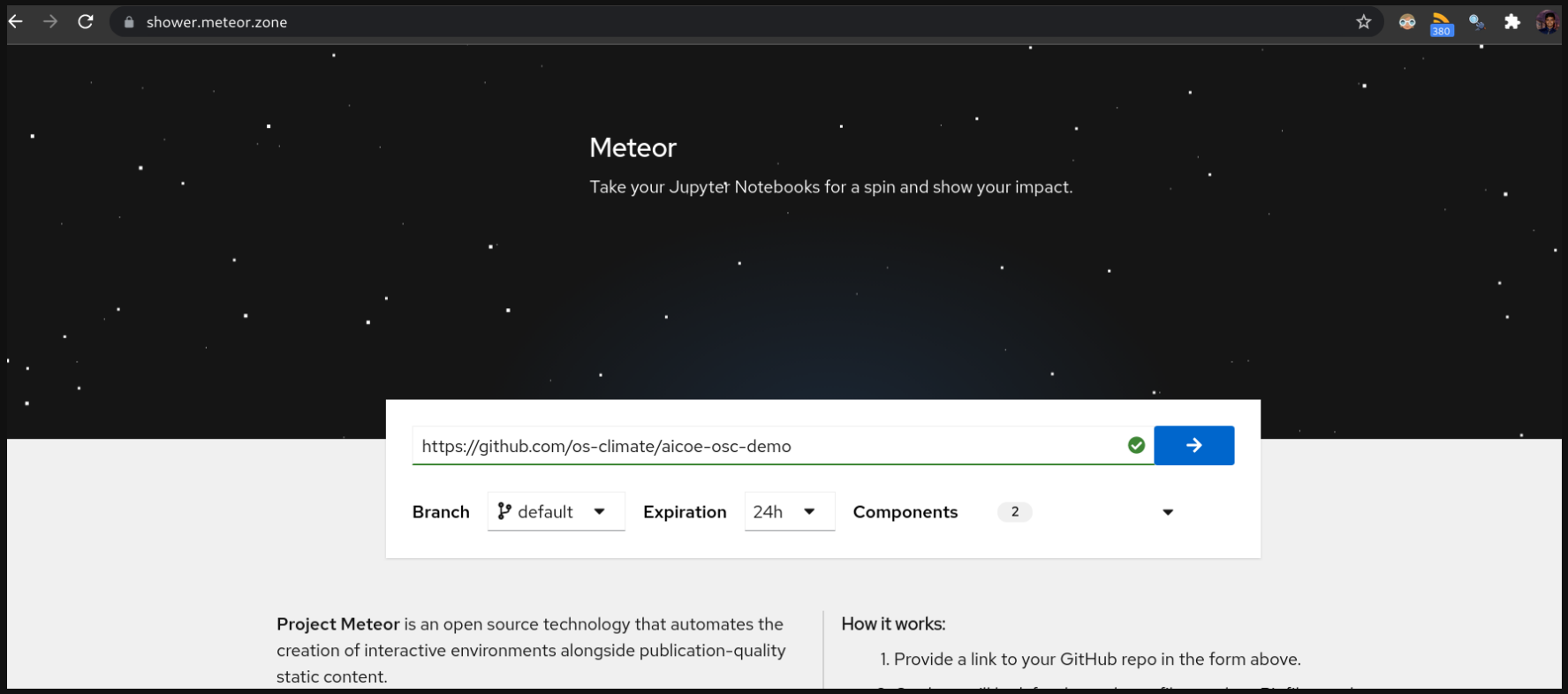
- Thoth recommended, pinned dependencies in Pipfile
- Set up AICoE-CI
    - Add .aicoe-ci.yaml config file to repo
    - Open issue to onboard project
- Create release or open an issue to build image
- Add image to JupyterHub through a PR

Demo

- Example repo with aicoe-ci set up
- Create release and trigger build

Bonus: Meteor

shower.meteor.zone



Q & A