



High-Performance Data-Intensive
Computing Systems Laboratory

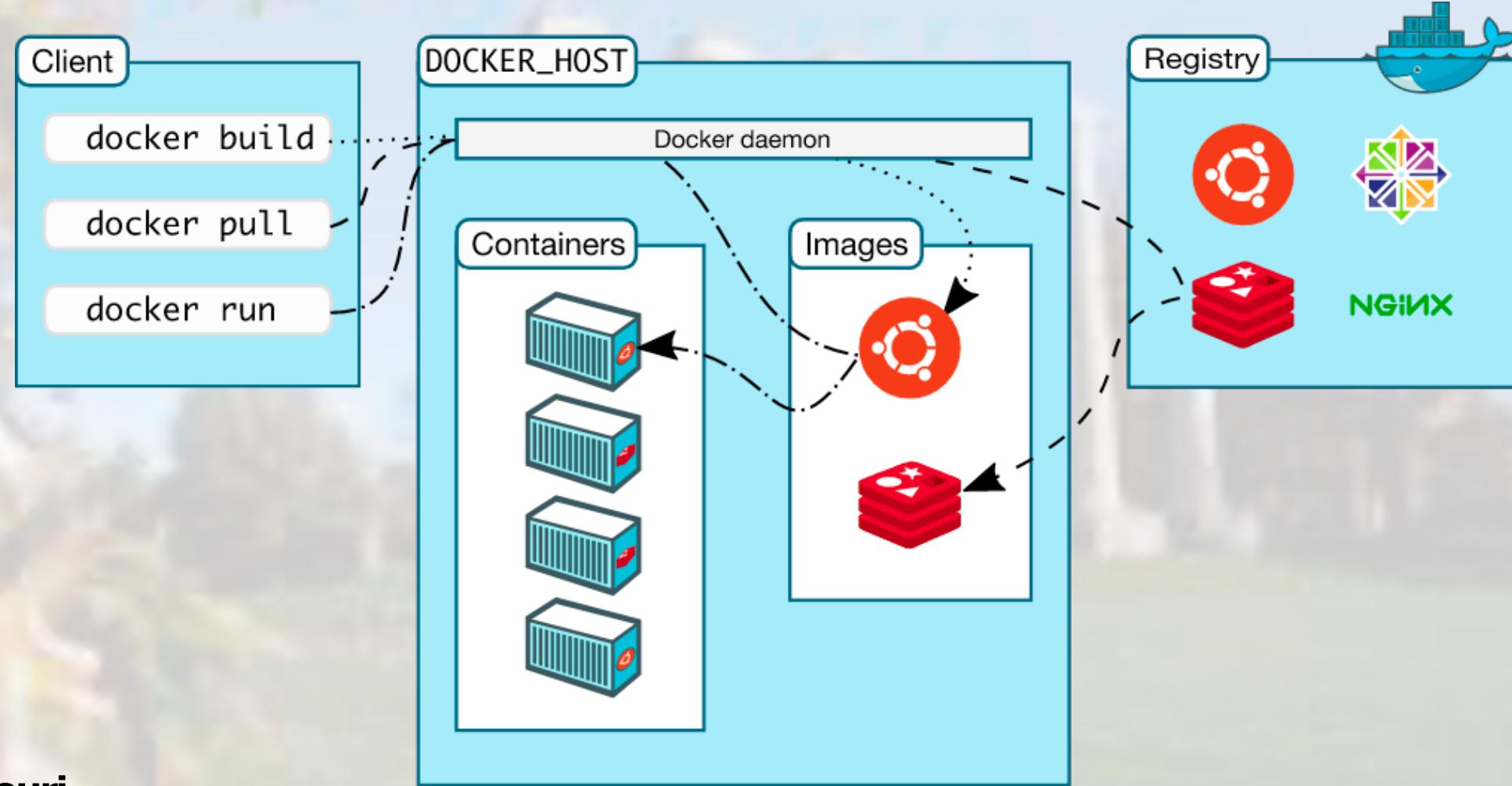
Bring Your Own Container Image: GitLab CI/CD and Automating Image Creation for Kubernetes



University of Missouri

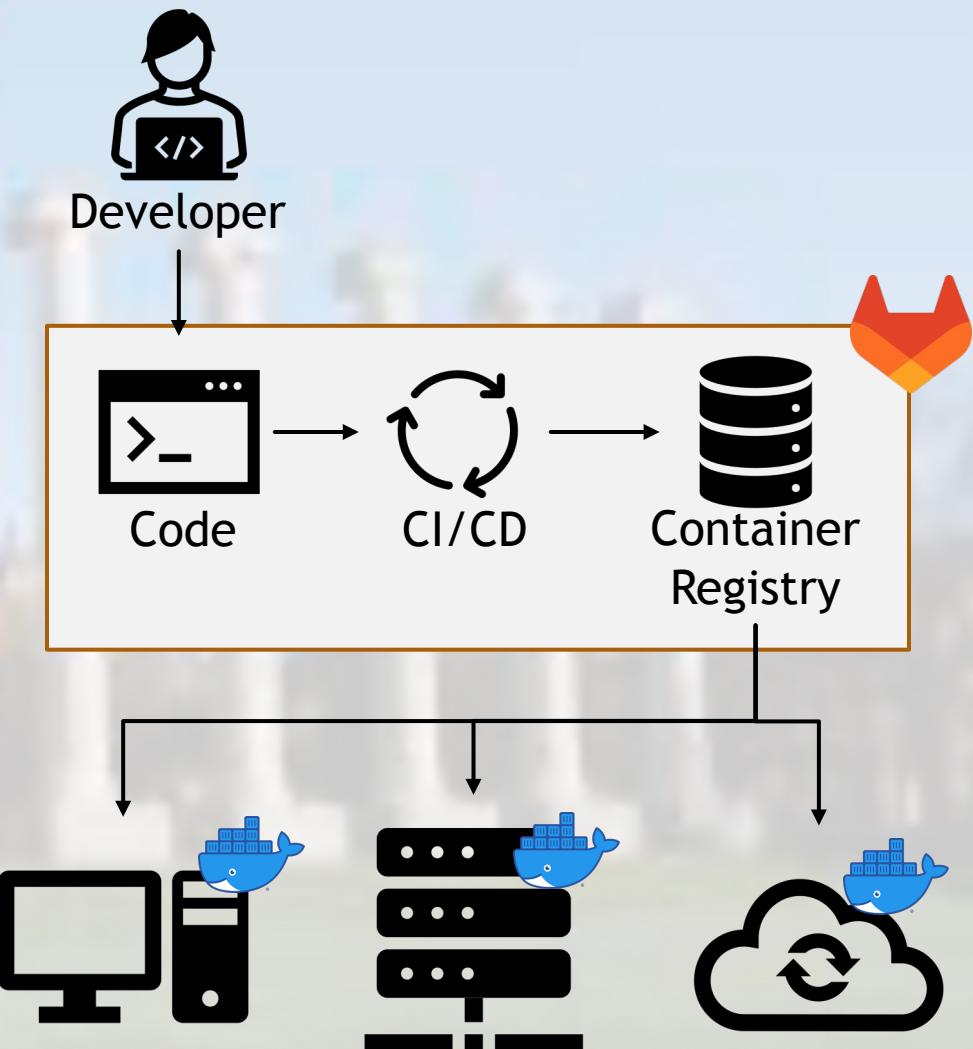
Containerization & Kubernetes

- ▶ Kubernetes requires a publicly available image, and we need the flexibility to use own our images
- ▶ Nautilus Gitlab
 - ▶ VCS
 - ▶ Create easily maintained and developed custom images using GitLab CI



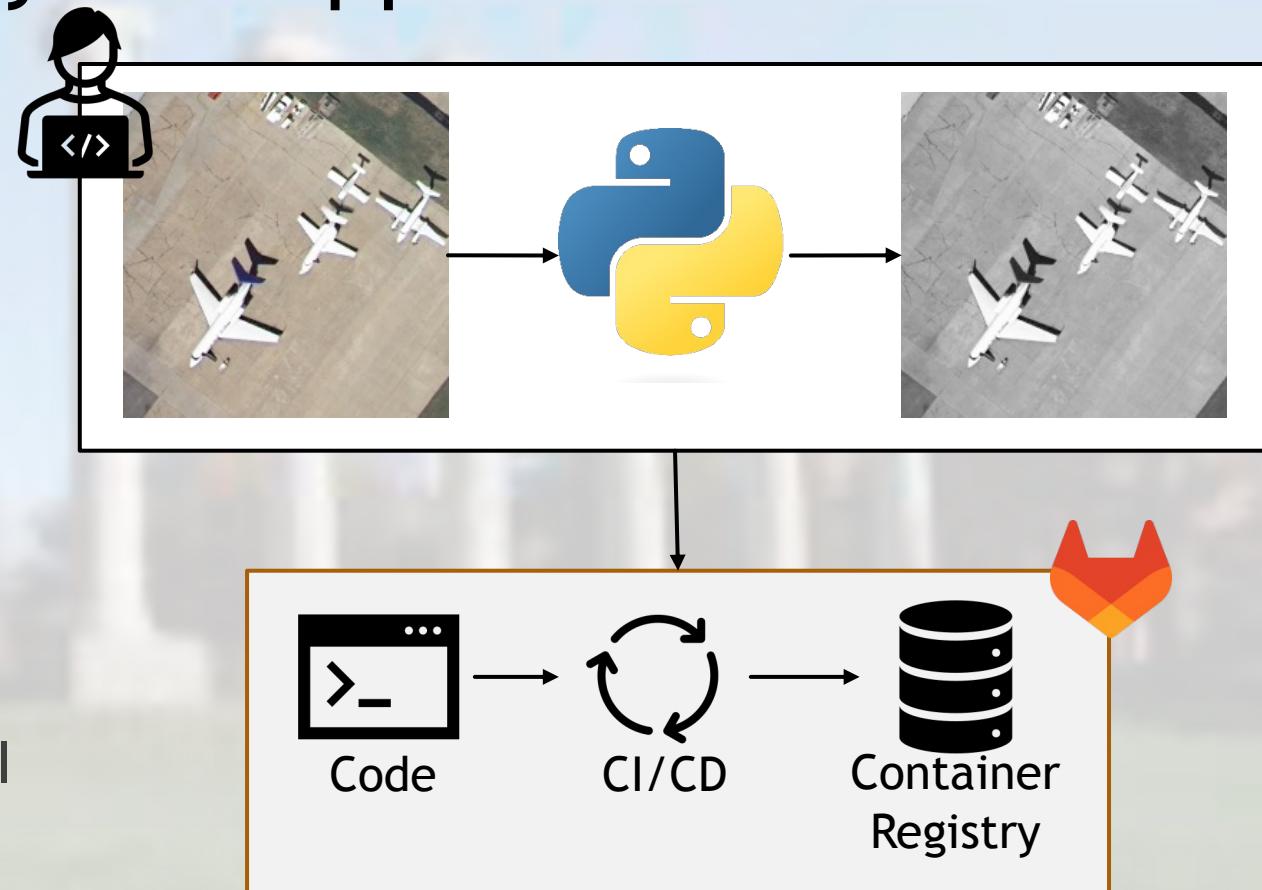
Automation with GitLab

- ▶ GitLab is a common tool for CI/CD with Docker because it contains VCS, CI/CD and a built-in image registry with access restrictions
 - ▶ Only those added to the repository can access the code, CI/CD build instructions, and the container registry
- ▶ In our previous example, the VCS, CI/CD, and Registry providers were all independent, but GitLab contains all three of these on a per-repository or per-group basis



Automation in Action: GitLab CI + Registry for Python Application

- ▶ We configure GitLab to utilize CI/CD and the Container Registry to automatically build and push a container each time we push a new commit to the repository
- ▶ Each commit has its own tag, and latest is automatically updated to point to the most recent commit
- ▶ Let's view the repository, the Gitlab CI config, and the container registry!



Containerization

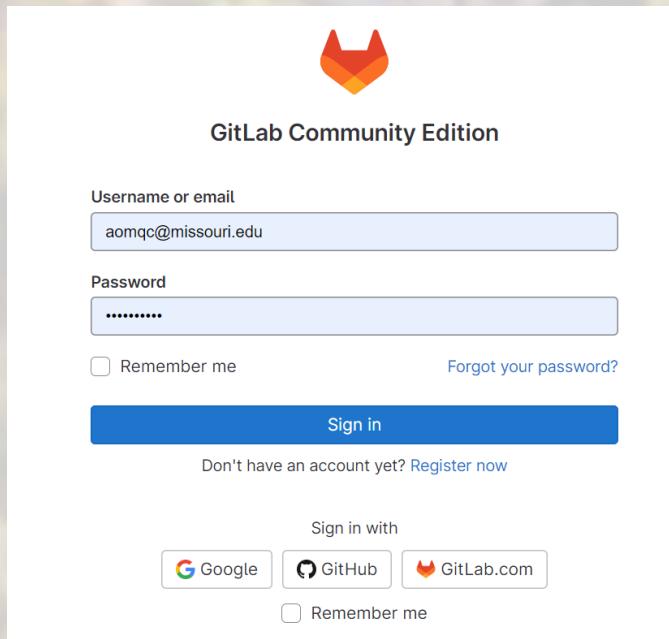
Nautilus GitLab

<https://gitlab.nrp-nautilus.io/>

- ▶ Git Version Control System
- ▶ CI/CD services for building and storing container images to use on Nautilus



Creating an NRP GitLab Account



GitLab Community Edition

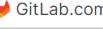
Username or email
aomqc@missouri.edu

Password
.....

Remember me [Forgot your password?](#)

Sign in

Don't have an account yet? [Register now](#)

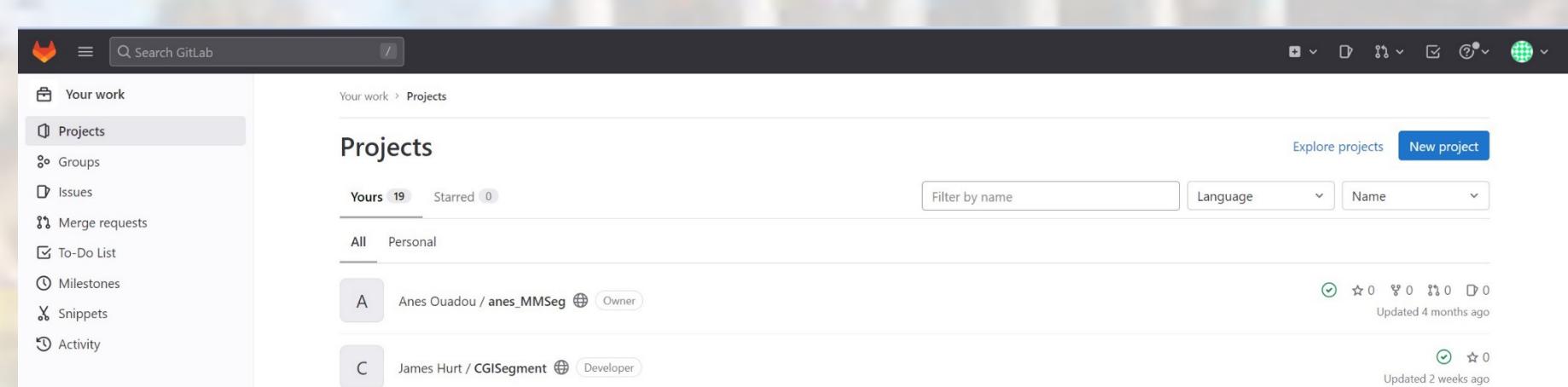
Sign in with
 Google  GitHub  GitLab.com

Remember me



Create an account on Nautilus GitLab

Note: You may need to have your university system added by NRP Admins to use NRP GitLab



Your work > Projects

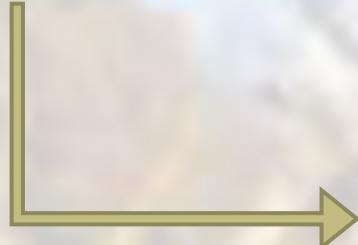
Projects

Yours 19 Starred 0

All Personal

A Anes Ouadou / anes_MMSeq Owner

C James Hurt / CGISeegment Developer



Creating a Custom Container Image

Create new repository

- Code & other files can be private, but the repository and the container registry **MUST** be public
- Open the repository in the Web IDE or clone it locally

Create Gitlab CI YAML File

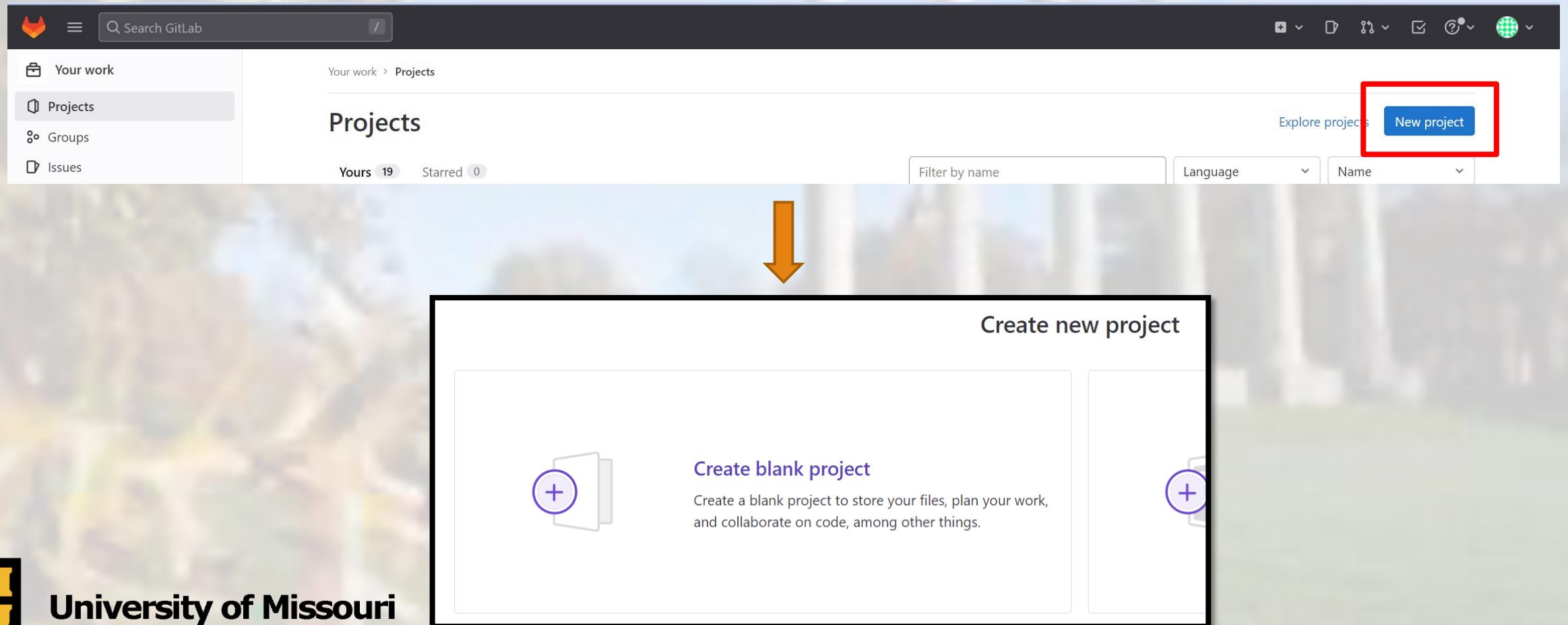
- Create a file called `.gitlab-ci.yml`
- Copy its content from this link: [.gitlab-ci.yml](#)
- Commit and push to your repository

Create Dockerfile

- Write the appropriate commands to build the image you need
- Use a community container as your base
- Commit and push to your repository

NRP GitLab

Creating a New Repository



The screenshot shows the GitLab web interface. On the left, there's a sidebar with options: 'Your work' (selected), 'Projects' (highlighted with a red box), 'Groups', and 'Issues'. The main area is titled 'Projects' and shows 'Your work > Projects'. It displays 'Yours 19' and 'Starred 0'. On the right, there are search and filter fields ('Filter by name', 'Language', 'Name') and a blue 'New project' button, which is also highlighted with a red box. A large orange arrow points from the 'New project' button down to a callout box labeled 'Create new project'. This callout box contains two sections: 'Create blank project' (with a description: 'Create a blank project to store your files, plan your work, and collaborate on code, among other things.') and another 'Create blank project' button.

NRP GitLab

Creating a New Repository

Projects

Groups

Issues

Merge requests

To-Do List

Milestones

Snippets

Activity

Create blank project

Create a blank project to store your files, plan your work, and collaborate on code, among other things.

Project name
My awesome project

Must start with a lowercase or uppercase letter, digit, emoji, or underscore. Can also contain dots, pluses, dashes, or spaces.

Project URL
<https://gitlab.nrp-nautilus.io/aomqc/>

Project slug
/ my-awesome-project

Want to organize several dependent projects under the same namespace? [Create a group](#).

Visibility Level [?](#)

 Private
Project access must be granted explicitly to each user. If this project is part of a group, access is granted to members of the group.

 Internal
The project can be accessed by any logged in user except external users.

 Public
The project can be accessed without any authentication.

Project Configuration

Initialize repository with a README
Allows you to immediately clone this project's repository. Skip this if you plan to push up an existing repository.

Enable Static Application Security Testing (SAST)
Analyze your source code for known security vulnerabilities. [Learn more](#).

Create project **Cancel**

CGISegment

Project ID: 2316 [Leave project](#)

568 Commits 1 Branch 0 Tags 1.1 MB Project Storage

Fix Docker image e98e742e
 Alex Hurt authored 6 months ago

master / cgisegment / +

[Find file](#) [Web IDE](#) [Clone](#)

[README](#) [CI/CD configuration](#) [Wiki](#)

Name	Last commit	Last update
cgisegment	Wrong indentation for final test set aggregation	9 months ago
sample_configs	Updating the sample configs	1 year ago
tests	Getting Tests Up-To-Date	1 year ago
.gitignore	*ipynb	1 year ago
.gitlab-ci.yml	Add gitlab CI YML	7 months ago
Dockerfile	Fix Docker image	6 months ago
README.md	Remove ref	7 months ago
main.py	SegFormer and Lawin	10 months ago

► We need to create:

- Gitlab CI YAML
- Dockerfile



Creating the GitLab CI YAML

<https://ucsd-prp.gitlab.io/userdocs/development/gitlab/>

```
image: gcr.io/kaniko-project/executor:debug

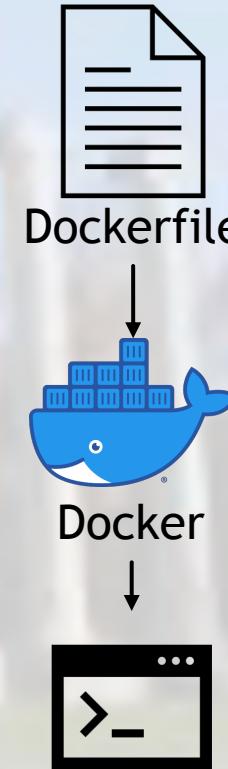
stages:
- build-and-push

build-and-push-job:
  stage: build-and-push
  variables:
    GODEBUG: "http2client=0"
  script:
    - echo "{\"auths\":{\"$CI_REGISTRY\":{\"username\":\"$CI_REGISTRY_USER\",\"password\":\"$CI_REGISTRY_PASSWORD\"}}}"
    - /kaniko/executor --cache=true --push-retry=10 --context $CI_PROJECT_DIR --docke
```



Dockerfiles

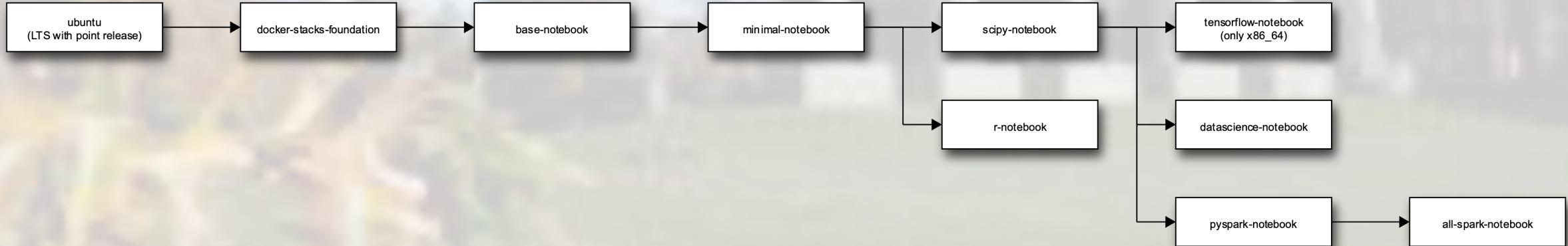
- ▶ Recall: Containers and container images enable reliable portability of software
- ▶ We need a way to build container images in a distributed and standardized way, so that anyone with a container runtime can build and run our software package (container image)
- ▶ Dockerfiles are the template, or recipe, for how a container image will be built
- ▶ Dockerfiles use a custom format and set of commands to describe how a container image will be built



Runnable Container Image

Using Community Published Container Images

- ▶ Container images are extensible!
 - ▶ Containers can be built from other images
- ▶ We can then build hierarchical docker containers, where each container has a specific set of dependencies and packages installed
- ▶ Base images enable rapid and reproducible building of even complex docker containers by leveraging the open source, published docker images



Example: Jupyter Docker Images¹



University of Missouri

1. <https://jupyter-docker-stacks.readthedocs.io/en/latest/using/selecting.html>

Dockerfile: Common Commands

- ▶ Dockerfiles function via a set of directives and their respective arguments:

DIRECTIVE arg1 arg2 ...

- ▶ FROM

- ▶ The FROM command defines the starting point for the Docker build process. If you are building custom software from the ground up, you may be setting this to a linux operating system, such as ubuntu.
- ▶ If you are working in a framework or programming language, this may also be a certain version of that framework, such as python:3.8, node:8, pytorch:1.8, etc.
- ▶ The docker image specified in the from command must be present either on the public docker hub repository, or be visible to the build context via a full URL.

- ▶ ENV & ARG

- ▶ The ENV and ARG commands define environment variables during the build process. There are 2 key differences between them:
- ▶ ENV commands persist to the final container, so ENV key value will persist to the launched container, while ARG key value will not
- ▶ ARG commands can be overridden at build time, which allows for templating of Dockerfiles
- ▶ The ENV command is very useful for tasks like ensuring your executable is present on the PATH of the container, while ARG is useful for things like ensuring you are building the correct version of the software.



Dockerfile: Common Commands

► CMD and ENTRYPOINT

- ▶ The CMD and ENTRYPOINT commands both set the command to be run when the container starts via a docker run command.
- ▶ There is 1 key difference between them: a CMD can be overridden via either a downstream Dockerfile (i.e., someone uses your container in their FROM command) or via the command line during container startup. An ENTRYPOINT cannot be easily overridden, and requires a special flag to be overridden.
- ▶ Best practice is to set a CMD unless you have reasoning behind using ENTRYPOINT

► COPY

- ▶ The COPY command is how local files are copied into the container. Recall that Docker containers are mini virtual machines, meaning that it has its own filesystem. In order to build and run software in docker, we often need to copy our code and scripts into the container. To do this, we use the COPY command

► RUN

- ▶ The RUN command tells the docker build process to run a command inside the container. This could be everything from installing a package with apt (RUN apt install) to creating and removing files in the container.
- ▶ The commands available for the RUN command are determined by the base container, i.e. apt will only be available in debian based containers.



Sample Dockerfile

```
ARG PYVERSION=3.8
FROM python:${PYVERSION}
RUN mkdir -p /workspace
WORKDIR /workspace
COPY /requirements.txt /workspace
RUN pip install -r ./requirements.txt
COPY /*.py /workspace/
CMD /bin/bash
```

Create a build argument for the python version that defaults to 3.8

Start FROM an existing image in a Docker *registry*. In this case, python

Create a directory named /workspace and set it as the working directory

Copy a file named “requirements.txt” from the build directory to /workspace in the container

Use pip to install the requirements defined in requirements.txt

Copy all python files in the build directory to the /workspace directory in the container

Set the default command to run when the container starts to /bin/bash

Writing Dockerfiles: Multiple Paths to the Same Goal

Two different styles of writing Docker file

main ▾ deeplearning_pytorch / Dockerfile

Dockerfile 1.24 KiB

```

1 FROM nvidia/cuda:11.2.2-cudnn8-runtime-ubuntu20.04
2
3 RUN chmod 1777 /tmp && chmod 1777 /var/tmp
4 RUN apt-get update
5 RUN apt-get upgrade -y
6 RUN apt-get install -y vim curl ca-certificates amqp-tools wget graphviz
7 RUN apt-get -y install git
8
9 RUN apt-get install -y python3
10 RUN apt-get install -y python3-pip
11
12 RUN pip install --upgrade protobuf==3.20.1
13
14 RUN pip install numpy scipy
15 RUN pip install matplotlib
16 RUN pip install pyyaml
17 RUN pip install pytest-shutil
18 RUN pip install rasterio
19 RUN pip install pandas
20 RUN pip install geopandas
21 RUN pip install geojson
22 RUN pip install rtree
23 RUN pip install pillow
24 RUN pip install jinja2
25 RUN pip install bs4
26 RUN pip install lxml
27 RUN pip install pydot
28 RUN pip install pydotplus
29
30 RUN apt-get install -y binutils libproj-dev gdal-bin
31

```

Find file Blame History Permalink

Open in Web IDE Replace Delete

master ▾ cgisegment / Dockerfile

 Fix Docker image Alex Hurt authored 6 months ago

Dockerfile 1.29 KiB

```

1 FROM pytorch/pytorch:1.12.1-cuda11.3-cudnn8-devel
2
3 #####
4 # env
5 #####
6 RUN mkdir -p /workspace
7 ENV CGISEGMENT /workspace/cgisegment/
8 ENV DEBIAN_FRONTEND noninteractive
9 ENV DISTRO ubuntu2004
10 ENV ARCH x86_64
11 ENV TORCH_HOME /pytorch_data
12
13 #####
14 # Install dependencies
15 #####
16 RUN apt-key adv --fetch-keys https://developer.download.nvidia.com/compute/cuda/repos/$DISTRO/$ARCH/3bf863cc.pub
17 RUN apt update -y & apt install -y \
18     git \
19     build-essential \
20     libsm6 \
21     libxext6 \
22     libxrender-dev \
23     libgl1-mesa-glx \
24     libglib2.0-0
25
26 #####
27 # Copy files
28 #####
29 RUN mkdir -p ${CGISEGMENT}
30 COPY ./main.py ${CGISEGMENT}
31 COPY ./cgisegment ${CGISEGMENT}/cgisegment
32 COPY ./requirements.txt /tmp/requirements.txt
33

```

Find file Blame History Permalink

Open in Web IDE Replace Delete



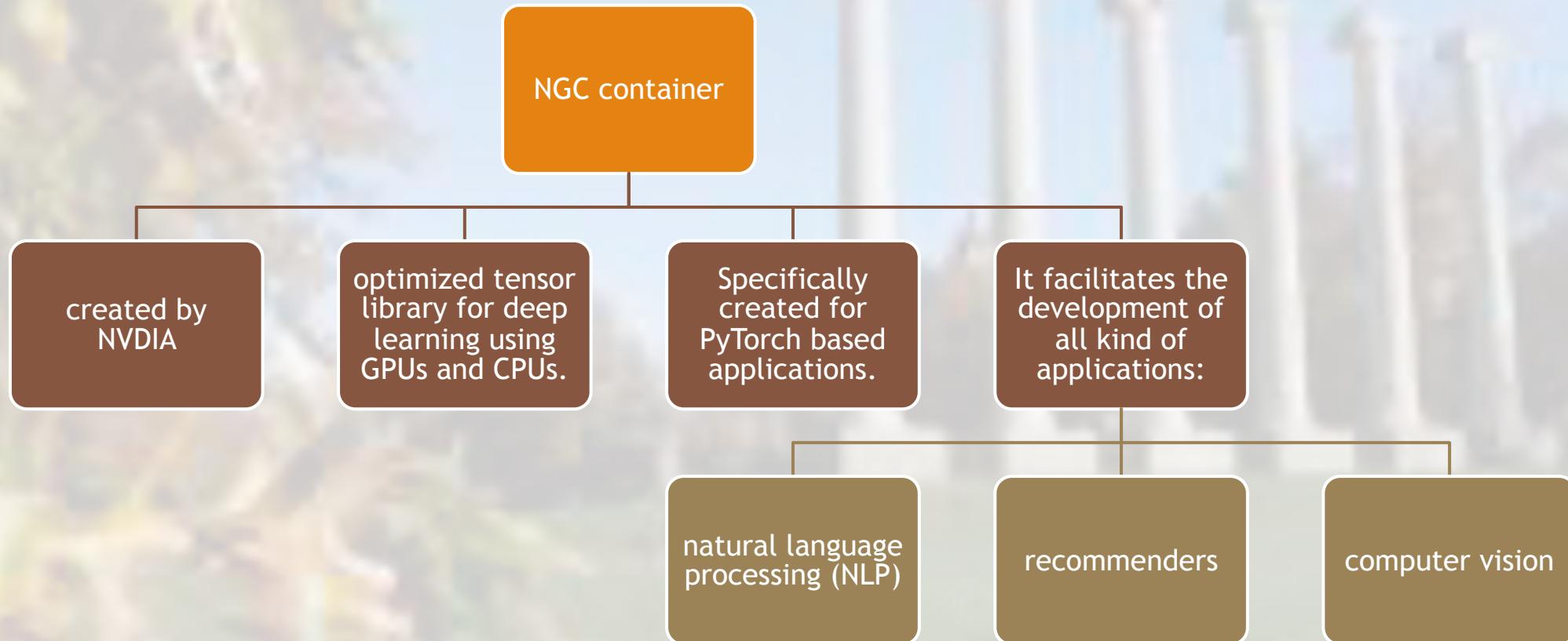
Nautilus Gitlab: Building GPU-enabled Containers

- ▶ To utilize the GPUs in a cluster, **GPU-enabled image is a MUST**
- ▶ GPU containers can be built from community developed and published base containers, such as nvidia/cuda and the NVIDIA NGC
- ▶ CUDA GPU base images comes in three variations:
 - ▶ base
 - ▶ runtime
 - ▶ devel

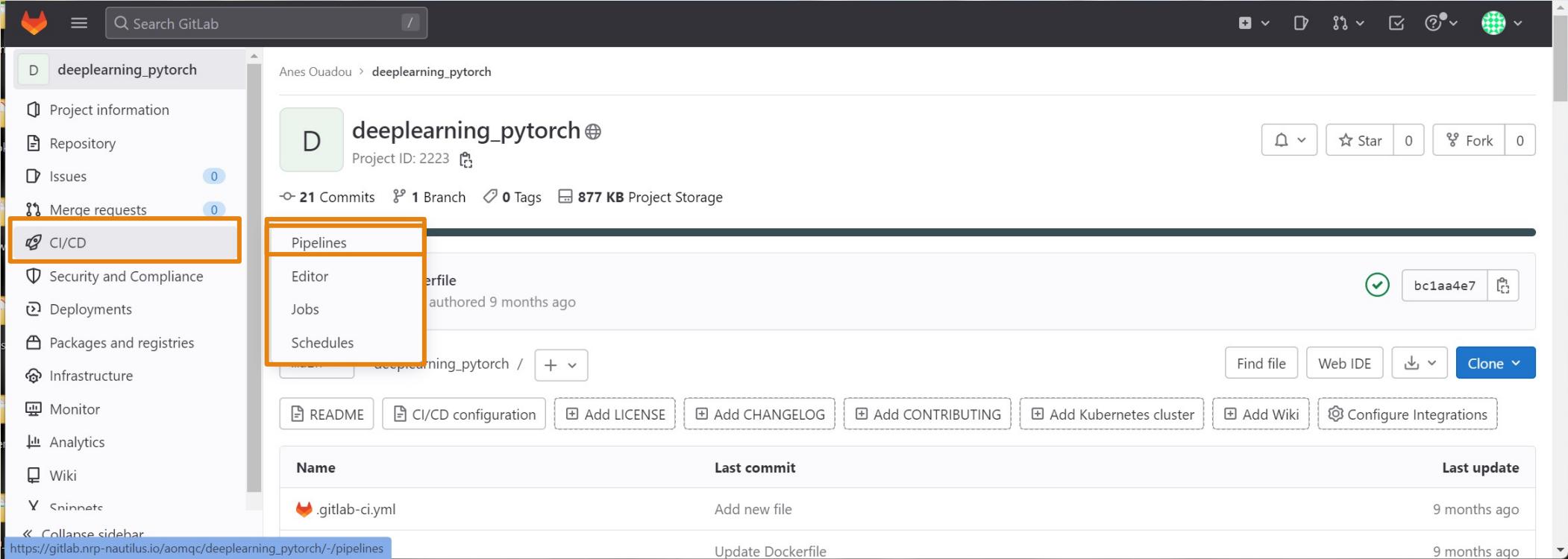
NVIDIA CUDA Containers: Three Different Types



NVIDIA NGC Containers: Optimized But Inflexible



Nautilus GitLab: Viewing CI/CD Pipelines



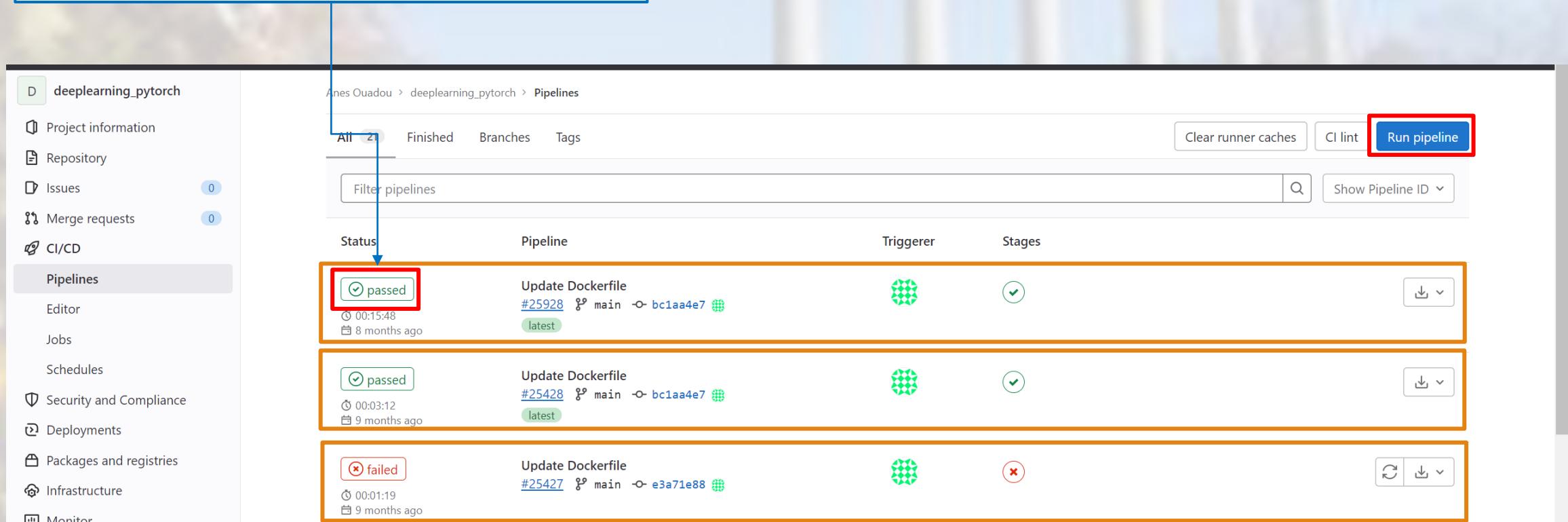
The screenshot shows the GitLab interface for the project "deeplearning_pytorch". The sidebar on the left has a menu with various options, and the "CI/CD" option is highlighted with an orange box. The main content area displays the project details and the "Pipelines" section, which is also highlighted with an orange box. The pipelines listed are "Editor", "Jobs", and "Schedules". Below the pipelines, there is a table showing a single commit for ".gitlab-ci.yml". The commit was authored 9 months ago and updated 9 months ago. The commit message is "Update Dockerfile".

Name	Last commit	Last update
.gitlab-ci.yml	Add new file	9 months ago
	Update Dockerfile	9 months ago

Nautilus Gitlab: Viewing Pipeline Status

The image is being built the status is:

Running



Anes Ouadou > deeplearning_pytorch > Pipelines

All Finished Branches Tags

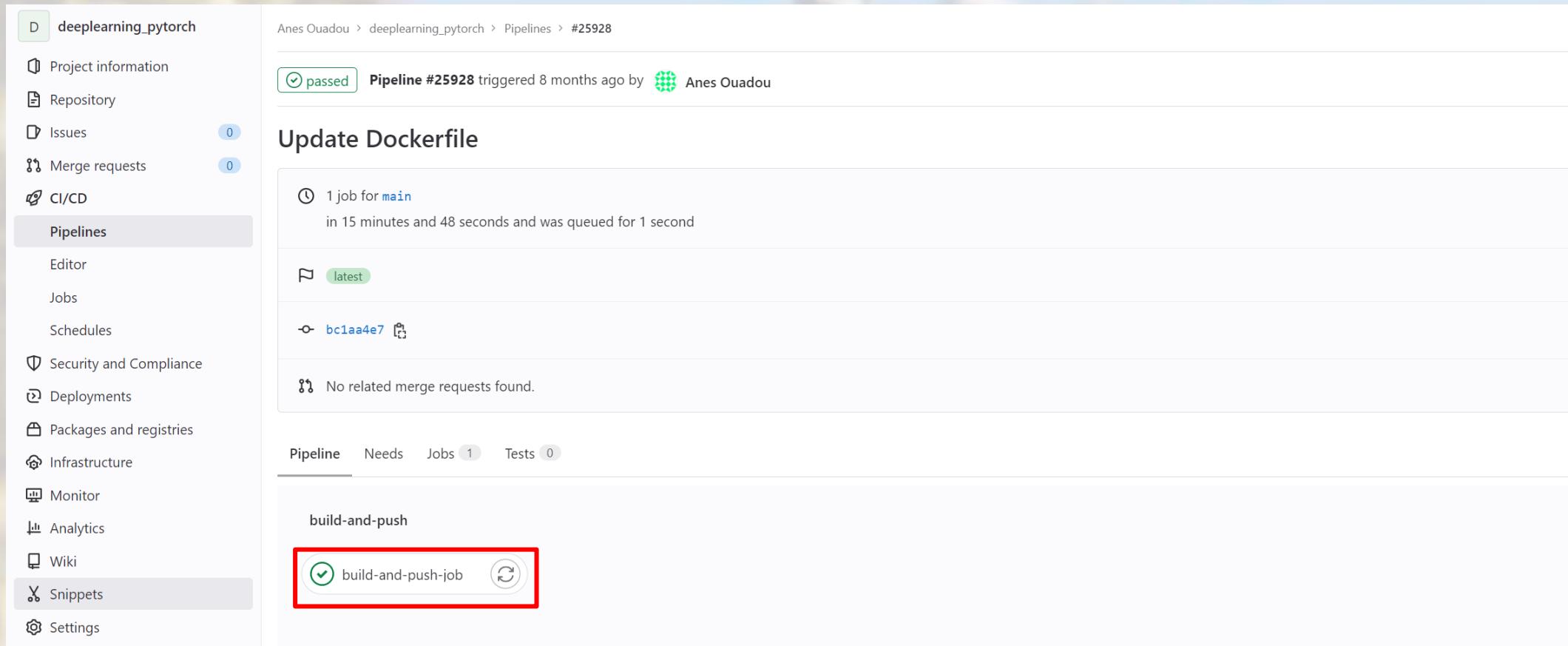
Clear runner caches CI lint Run pipeline

Filter pipelines

Status	Pipeline	Triggerer	Stages
passed	Update Dockerfile #25928 ➝ main -o- bc1aa4e7		✓
passed	Update Dockerfile #25428 ➝ main -o- bc1aa4e7		✓
failed	Update Dockerfile #25427 ➝ main -o- e3a71e88		✗



Nautilus GitLab: Viewing Job Status



The screenshot shows the GitLab interface for the project `deeplearning_pytorch`. The left sidebar has a navigation menu with the following items:

- D deeplearning_pytorch
- Project information
- Repository
- Issues 0
- Merge requests 0
- CI/CD
- Pipelines** (selected)
- Editor
- Jobs
- Schedules
- Security and Compliance
- Deployments
- Packages and registries
- Infrastructure
- Monitor
- Analytics
- Wiki
- Snippets
- Settings

The main content area shows the pipeline for commit `#25928` triggered by Anes Ouadou. The pipeline step `Update Dockerfile` is shown with the following details:

- 1 job for `main` (passed) in 15 minutes and 48 seconds and was queued for 1 second.
- Latest build: `bc1aa4e7`
- No related merge requests found.

Below this, the pipeline summary shows:

- Pipeline
- Needs
- Jobs 1
- Tests 0

The job `build-and-push` is listed with status `build-and-push-job` (passed), indicated by a green checkmark icon. This icon is highlighted with a red border.

Nautilus GitLab: Reviewing Job Output

d **deelearning_pytorch**

- Project information
- Repository
- Issues 0
- Merge requests 0
- CI/CD
- Pipelines
- Editor
- Jobs**
- Schedules
- Security and Compliance
- Deployments
- Packages and registries
- Infrastructure
- Monitor
- Analytics
- Wiki
- X Snippets
- Settings

build-and-push... trash refresh

Duration: 15 minutes 48 seconds
Finished: 8 months ago
Queued: 0 seconds
Timeout: 1h (from project) refresh

Commit [bc1aa4e7](#) diff
Update Dockerfile

Pipeline #25928 for main refresh

build-and-push refresh

→ **build-and-push-job**

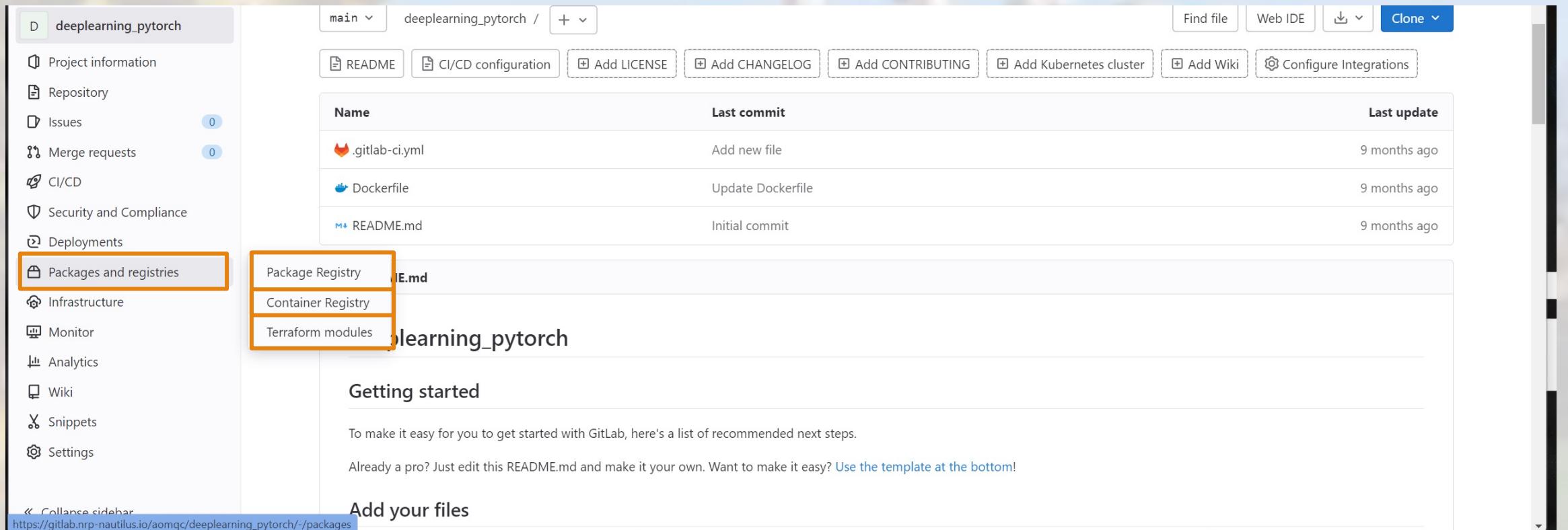
Search job log 🔍 ? 📄 ⬆️ ⬇️

```

2349 Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.8/dist-packages (from matplotlib->mmsegmentation==0.28.0) (1.0.5)
2350 Requirement already satisfied: pyparsing>=2.2.1 in /usr/local/lib/python3.8/dist-packages (from matplotlib->mmsegmentation==0.28.0) (3.0.9)
2351 Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.8/dist-packages (from matplotlib->mmsegmentation==0.28.0) (4.37.3)
2352 Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.8/dist-packages (from matplotlib->mmsegmentation==0.28.0) (0.11.0)
2353 Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.8/dist-packages (from matplotlib->mmsegmentation==0.28.0) (2.8.2)
2354 Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.8/dist-packages (from matplotlib->mmsegmentation==0.28.0) (9.2.0)
2355 Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.8/dist-packages (from matplotlib->mmsegmentation==0.28.0) (1.4.4)
2356 Collecting wcwidth
2357   Downloading wcwidth-0.2.5-py2.py3-none-any.whl (30 kB)
2358 Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.8/dist-packages (from python-dateutil>=2.7->matplotlib->mmsegmentation==0.28.0) (1.16.0)
2359 Installing collected packages: mmcls, wcwidth, prettytable, mmsegmentation
2360   Running setup.py develop for mmsegmentation
2361 Successfully installed mmcls-0.23.2 mmsegmentation prettytable-3.4.1 wcwidth-0.2.5
2362 INFO[0431] Taking snapshot of full filesystem...
2363 INFO[0433] Pushed gitlab-registry.nrp-nautilus.io/aomqc/deeplearning_pytorch/cache@sha256:a4c9c94b28c76135095545240c83671bdeb446bcc18fc87002db017aa4758d6e
2364 INFO[0435] Pushing layer gitlab-registry.nrp-nautilus.io/aomqc/deeplearning_pytorch/cache:22beaa3e535c0e58c905455b9fc7c01cc06e31172da3b68b5fe46e4eb982f9b7 to cache now
2365 INFO[0435] Pushing image to gitlab-registry.nrp-nautilus.io/aomqc/deeplearning_pytorch/cache:22beaa3e535c0e58c905455b9fc7c01cc06e31172da3b68b5fe46e4eb982f9b7
2366 INFO[0436] Pushed gitlab-registry.nrp-nautilus.io/aomqc/deeplearning_pytorch/cache@sha256:0503959bebdaefca73555094b1d364581f26ec4ad0caf5cd3f914db1f09bf13ac
2367 INFO[0438] Pushed gitlab-registry.nrp-nautilus.io/aomqc/deeplearning_pytorch/cache@sha256:2c1d0df479dd67f880213aecf3ac4324bbd2cf908a801027dc596e105d4a2ba5
2368 INFO[0616] Pushed gitlab-registry.nrp-nautilus.io/aomqc/deeplearning_pytorch/cache@sha256:d60f4cc2d0e845a47edbdbbbc5d60ed5ed7fb3b70403d149c98b3cefa6bdc33e
2369 INFO[0616] Pushing image to gitlab-registry.nrp-nautilus.io/aomqc/deeplearning_pytorch.bc1aa4e7
2370 INFO[0931] Pushed gitlab-registry.nrp-nautilus.io/aomqc/deeplearning_pytorch@sha256:916d181c81af70e1e0db2ef9cc777845ec3e89c3d97a1dc82f40bf6cfb436859

```

Nautilus GitLab: Viewing the Container Registry



The screenshot shows the GitLab interface for the repository `deeplearning_pytorch`. The sidebar on the left contains various project management links, with `Packages and registries` highlighted by an orange box. Below it, under the heading `Container Registry`, another orange box highlights the `Container Registry` link. The main content area displays a table of files with their last commit details:

Name	Last commit	Last update
.gitlab-ci.yml	Add new file	9 months ago
Dockerfile	Update Dockerfile	9 months ago
README.md	Initial commit	9 months ago

Below the table, there's a section titled `Getting started` with instructions and a link to a template. At the bottom, there's a section for adding files.

Project navigation bar: main / deeplearning_pytorch / +

File actions: Find file, Web IDE, Download, Clone

File list:

- README
- CI/CD configuration
- Add LICENSE
- Add CHANGELOG
- Add CONTRIBUTING
- Add Kubernetes cluster
- Add Wiki
- Configure Integrations

Name	Last commit	Last update
.gitlab-ci.yml	Add new file	9 months ago
Dockerfile	Update Dockerfile	9 months ago
README.md	Initial commit	9 months ago

Getting started

To make it easy for you to get started with GitLab, here's a list of recommended next steps.

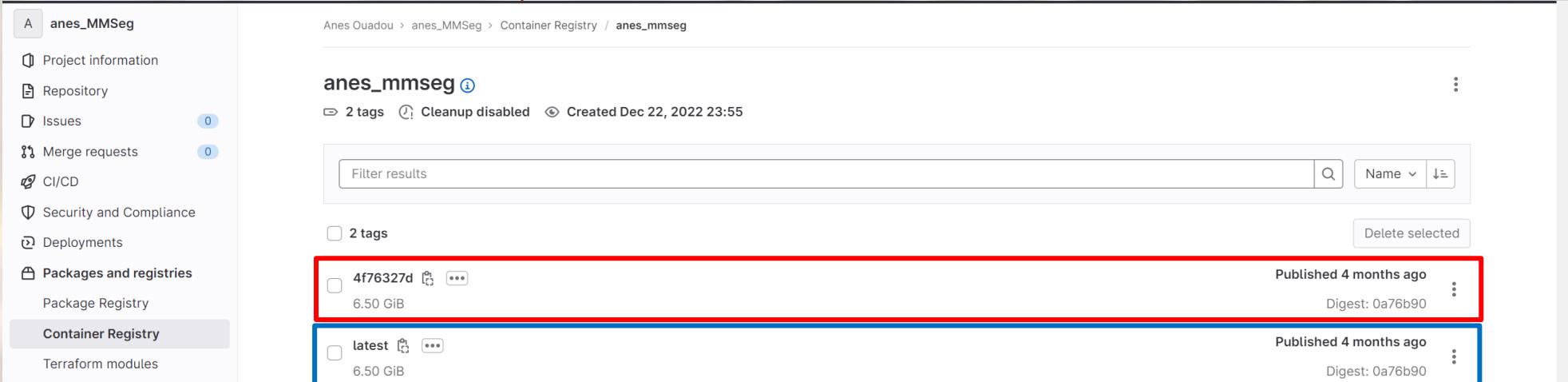
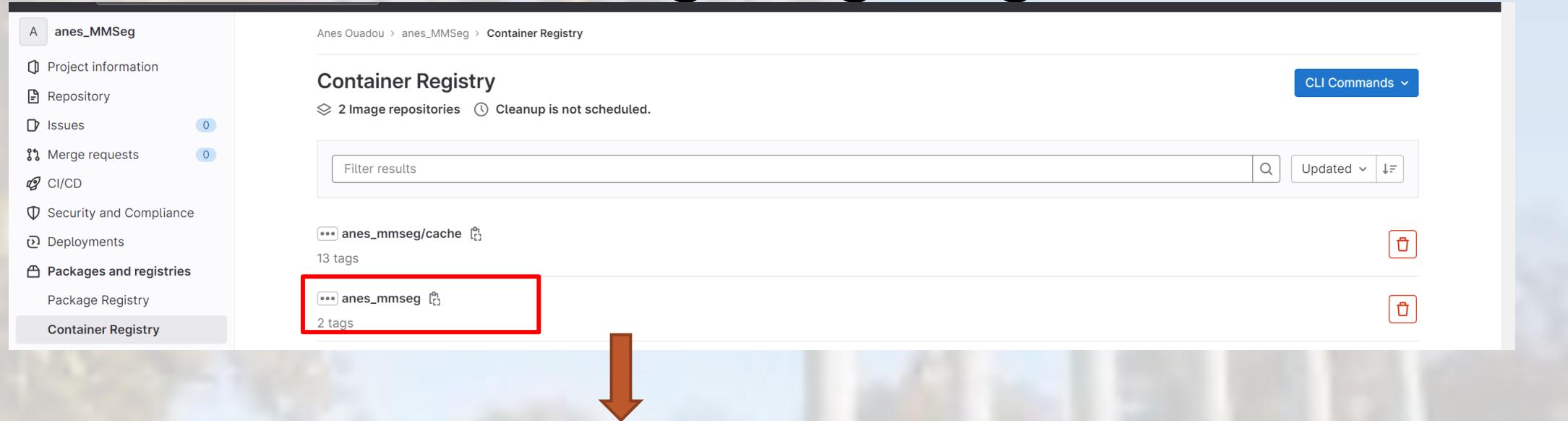
Already a pro? Just edit this README.md and make it your own. Want to make it easy? [Use the template at the bottom!](#)

Add your files

« Collapse sidebar

https://gitlab.nrp-nautilus.io/aomqc/deeplearning_pytorch/-/packages

Nautilus GitLab: Viewing Image Tags



Anes Ouadou > anes_MMSege > Container Registry

Container Registry

2 Image repositories Cleanup is not scheduled.

Filter results Updated

anes_mmseg/cache 13 tags

anes_mmseg 2 tags

2 tags

Anes Ouadou > anes_MMSege > Container Registry / anes_mmseg

anes_mmseg

2 tags Cleanup disabled Created Dec 22, 2022 23:55

Filter results Name

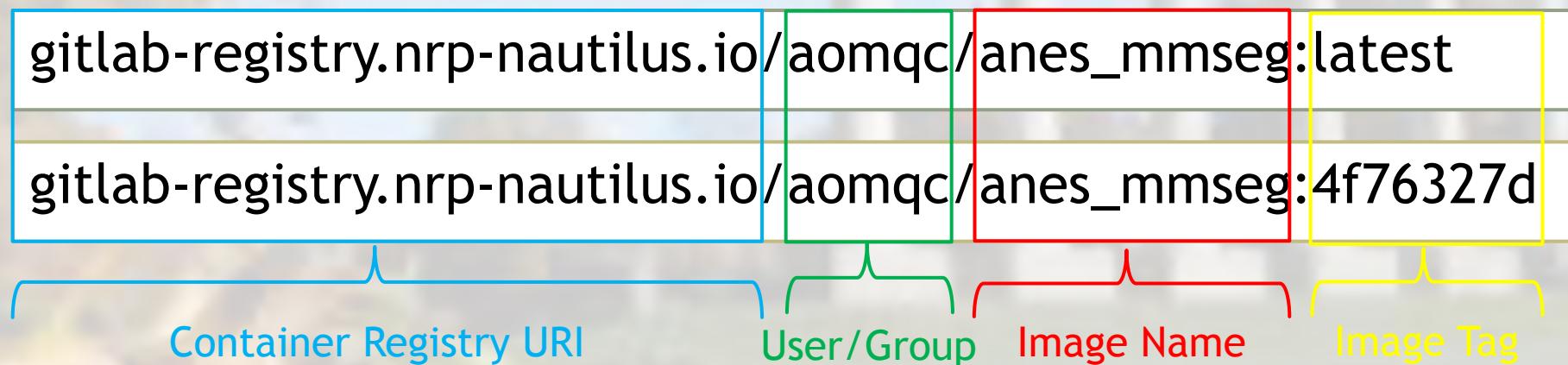
2 tags Delete selected

4f76327d 6.50 GiB Published 4 months ago Digest: 0a76b90

latest 6.50 GiB Published 4 months ago Digest: 0a76b90

Nautilus GitLab: Understanding Container Image URIs

- The list of images will always have one image labeled latest and at least one image with a randomly generated tag



Note: Never use the latest tag in your YAML files, as it causes collision issues on nodes in the cluster

Nautilus GitLab: Using Your Container Image on Nautilus



University of Missouri

```
apiVersion: batch/v1
kind: Job

metadata:
  name: convert-xview-train

spec:
  backoffLimit: 0
  template:
    spec:
      restartPolicy: Never
      containers:
        - name: dsc
          image: gitlab-registry.nrp-nautilus.io/jhurt/dsc:72de1d25
          imagePullPolicy: IfNotPresent
          command: ["python3", "/output/scripts/myscript.py"]
      resources:
        limits:
          memory: 16Gi
          cpu: 16
        requests:
          memory: 16Gi
          cpu: 16
      volumeMounts:
        - mountPath: /data
          name: jhurt-data
      volumes:
        - name: jhurt-data
          persistentVolumeClaim:
            claimName: jhurt-data
```

Hands-On Example: Image Processing (CPU)

- ▶ Register for a Gitlab Account on NRP
- ▶ Login to your Account
- ▶ Create a new public repository
- ▶ Add your Python code
- ▶ Add your Dockerfile
- ▶ Add your Gitlab CI YAML
- ▶ Commit & Push to trigger the CI build and push of your container image
- ▶ Use your container image in a pod on Nautilus



Hands-On Example: Matrix Multiplication (GPU)

- ▶ Create a new public repository
- ▶ Add your **GPU-enabled** Python code
- ▶ Add your **CUDA-based** Dockerfile
- ▶ Add your Gitlab CI YAML
- ▶ Commit & Push to trigger the CI build and push of your container image
- ▶ Use your container image in a pod on Nautilus