Data Archiving and Networked Services
DANS

# Dataverse technical workshop
# Tromso, Norway, 24.01.2020


contributors:
Gustavo Durand (Harvard IQSS, USA)
Oliver Bertuch (FZ Jülich, Germany)
Stefan Kasberger (AUSSDA, Austria)
James Myers (GDCC, USA)

chaired by Slava Tykhonov (DANS-KNAW, Netherlands)

# Dataverse as a FOSS product: good news

- Dataverse is Open Source software
- Great community with more than 100 contributors
- Contributions are coming from all continents
- Maintenance cost reduces as all community members are using the same software and helping each other
- Governance models can be reused by different countries
- Innovation goes fast

# Dataverse as a FOSS product: bad news

- Open Source doesn’t mean Free!
- Consider all required resources: both hardware and human
- Building a service is difficult, maintenance is expensive
- Integration with other services requires the management of changes and sometimes even not possible
- technical development is fast, the expertise isn’t up-to-date
- requires continuous training and very good communication between all partners

# Slides of Gustavo

- Communication channels in the Dataverse community, Global Dataverse Community Consortium (GDCC)
- The development process, issues and challenges
- Dataverse forks and clones, customization of templates
- Maintenance of the infrastructure, capacity, team management, SLA
- Dataverse architecture overview, glassfish tips
- Support of the federated authentication, version deaccessioning and embargo

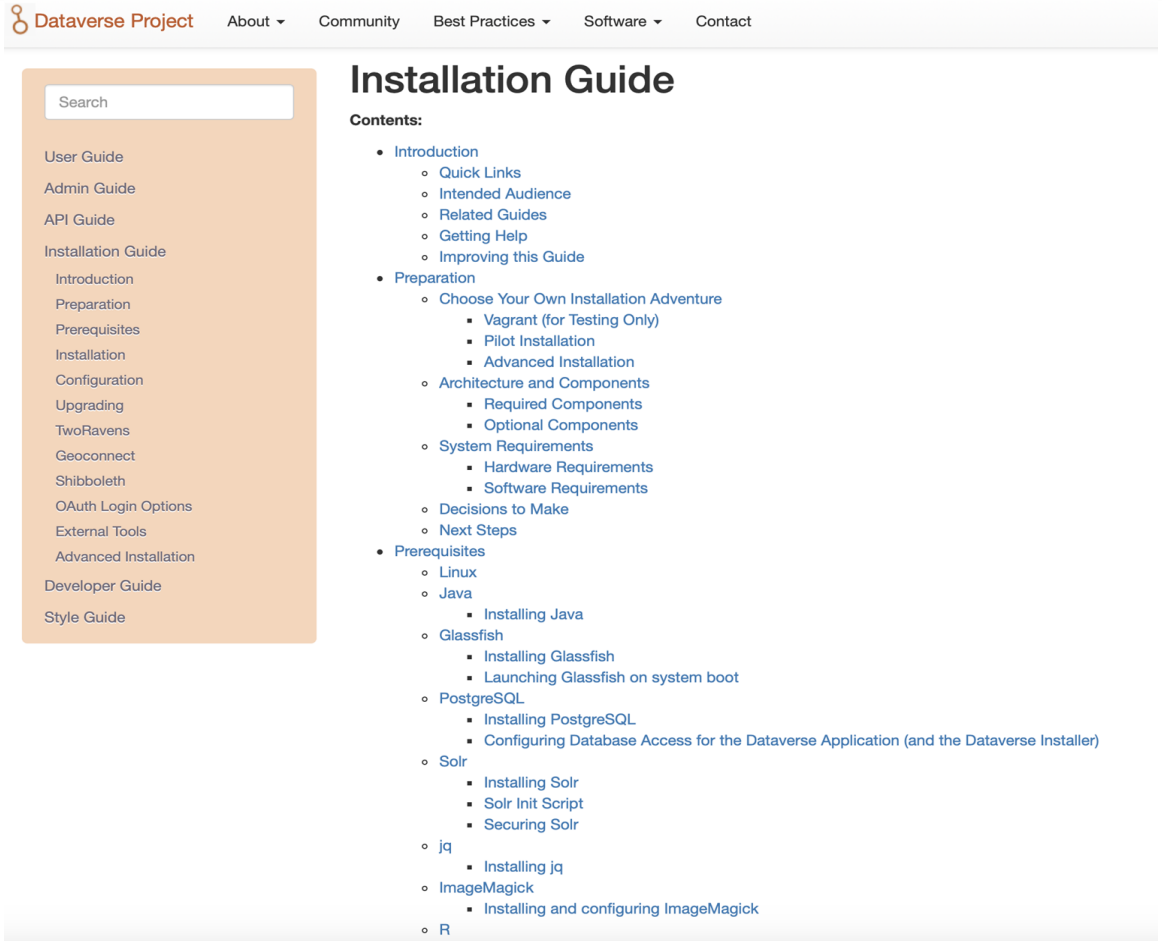
Slides

# Dataverse installation guide

- Manual setup according to the Dataverse Guide
- Dataverse Docker module introduction, share the instructions and slides
- Deploying Dataverse on a Cloud, how to choose Cloud provider

Practical task: participants will install Dataverse locally on their laptops using Docker Module and will investigate its infrastructure by accessing running Docker containers (10 min)

# Dataverse Installation Guide




[Instructions](http://guides.dataverse.org/en/latest/installation/)
http://guides.dataverse.org/en/latest/installation/

Before you start: installation requires preparation!

# Dataverse Docker module

This module was developed in one-year CESSDA DataverseEU project and aimed for CESSDA Service Providers who have limited technical resources. DANS led this project.

The goal was to deploy Dataverse software on CESSDA Technical Infrastructure (Google Cloud). Project was funded by the CESSDA 2018 workplan.

DataverseEU partners:  ADP (Slovenia), AUSSDA (Austria), GESIS (Germany), SND (Sweden), TARKI (Hungary), SiencePro (France), UKDA (UK), UniData (Italy), SODA (Belgium), LSZDA (Latvia), DANS (Netherlands)

# Docker introduction

- Extremely powerful configuration tool
- Allows to install software on any platform (Linux, Mac, Windows)
- Any software can be installed from Docker as standalone container or container delivering Microservices (database, search engine, core service)
- Docker allows to host unlimited amount of the same software tools on different ports
- Docker can be used to organise multilingual interfaces, for example

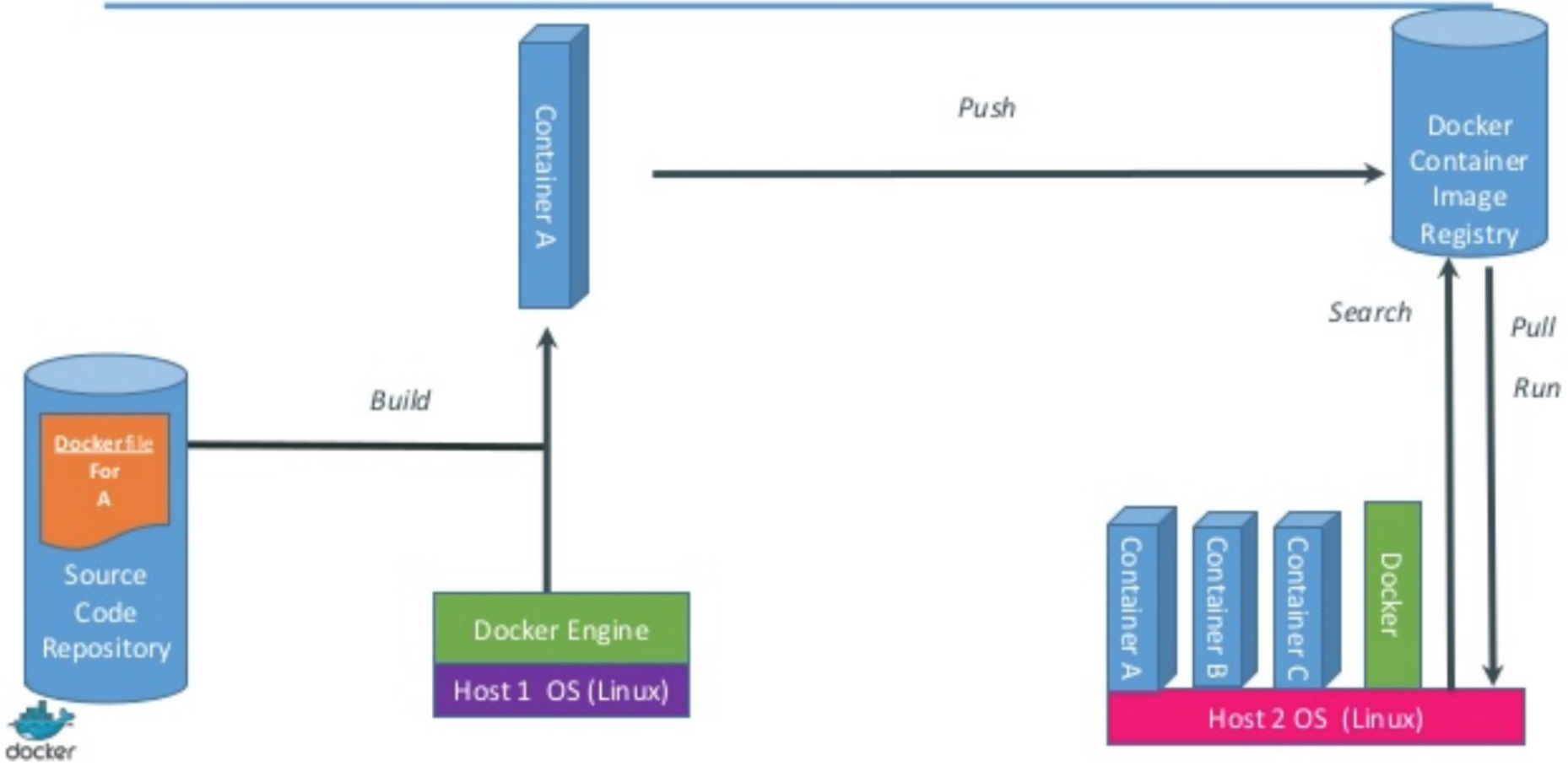
# Docker advantages

- Faster development and deployments
- Isolation of running containers allows to scale up apps
- Portability saves time to run the same image on the local computer or in the cloud
- Snapshotting allows to archive Docker images state
- Resource limitation can be adjusted

# Docker deployment with k8s in Clouds

- Google Cloud (policy for CESSDA SaW)
- Microsoft Azure
- Amazon Cloud
- OpenShift Cloud
- local Docker installation (minikube)

# What are the basics of the Docker system?



# Example: Dataverse as set of Docker microservices

```
docker run --name=postgres -d  postgres:9.3
```

Start a Solr 4.6 container for Dataverse:

```
docker run --name=solr -d ndslabs/dataverse-solr:latest
```

Start optional Rserve container:

```
docker run --name=rserve -d ndslabs/dataverse-rserve:latest
```

Start optional TwoRavens container:

```
docker run --name=tworavens -d ndslabs/dataverse-tworavens:latest
```

Start the preservation iRods server w/ federation listener:

```
docker run -e RODS_ZONE=fedZone  -p 1247:1247 -d  --name=icat-preservation ndslabs/irods-icat:latest
```

Start the Dataverse-local iRods server with custom rules:

```
docker run -e RODS_ZONE=dvnZone -e PRESERVATION_USER=dataverse -e PRESERVATION_ZONE=fedZone -e PRESERVAT
```

Without iRods containers: Start dataverse using the "link" flag to specify the other containers. Environment variables are used to setup the DVN database, user, and password

# Docker Desktop (Community Edition)

Ideal for developers and small teams looking to get started with Docker https://www.docker.com/community-edition

Features:

- docker-for-desktop
- docker-compose support
- integrated kubernetes (minikube)
- kitematic: Visual Docker Container Management

# Docker Hub

Docker Hub is registry containing images

Example: https://hub.docker.com/_/httpd/

$ docker pull httpd

Push images to Docker Hub: https://docs.docker.com/docker-cloud/builds/push-images/

$ docker login

$ docker tag my_image $DOCKER_ID_USER/my_image

$ docker push $DOCKER_ID_USER/my_image

# Docker concepts

- Containers are runnable artefacts
- Images are copies of containers with filesystems
- Containers can be archived as images and executed in different clouds
- Images can preserved in repositories https://act.dataverse.nl/dataset.xhtml?persistentId=hdl:10695/9VCRBR
- data folders can be hosted outside of containers on persistent volumes.

# Hello world app (Flask application)

Dockerfile https://github.com/DANS-KNAW/parthenos-widget/blob/master/Dockerfile

```
FROM python:2.7

MAINTAINER Vyacheslav Tykhonov

COPY . /widget

WORKDIR /widget

RUN pip install -r requirements.txt

ENTRYPOINT ["python"]
CMD ["app.py"]
```

# Docker command line usage

Command line allows to manage containers and images and execute Docker commands

*$ docker help run*

*$ docker ps*

*$ docker login*

*$ docker pull, push, commit*

*$ docker build, run*

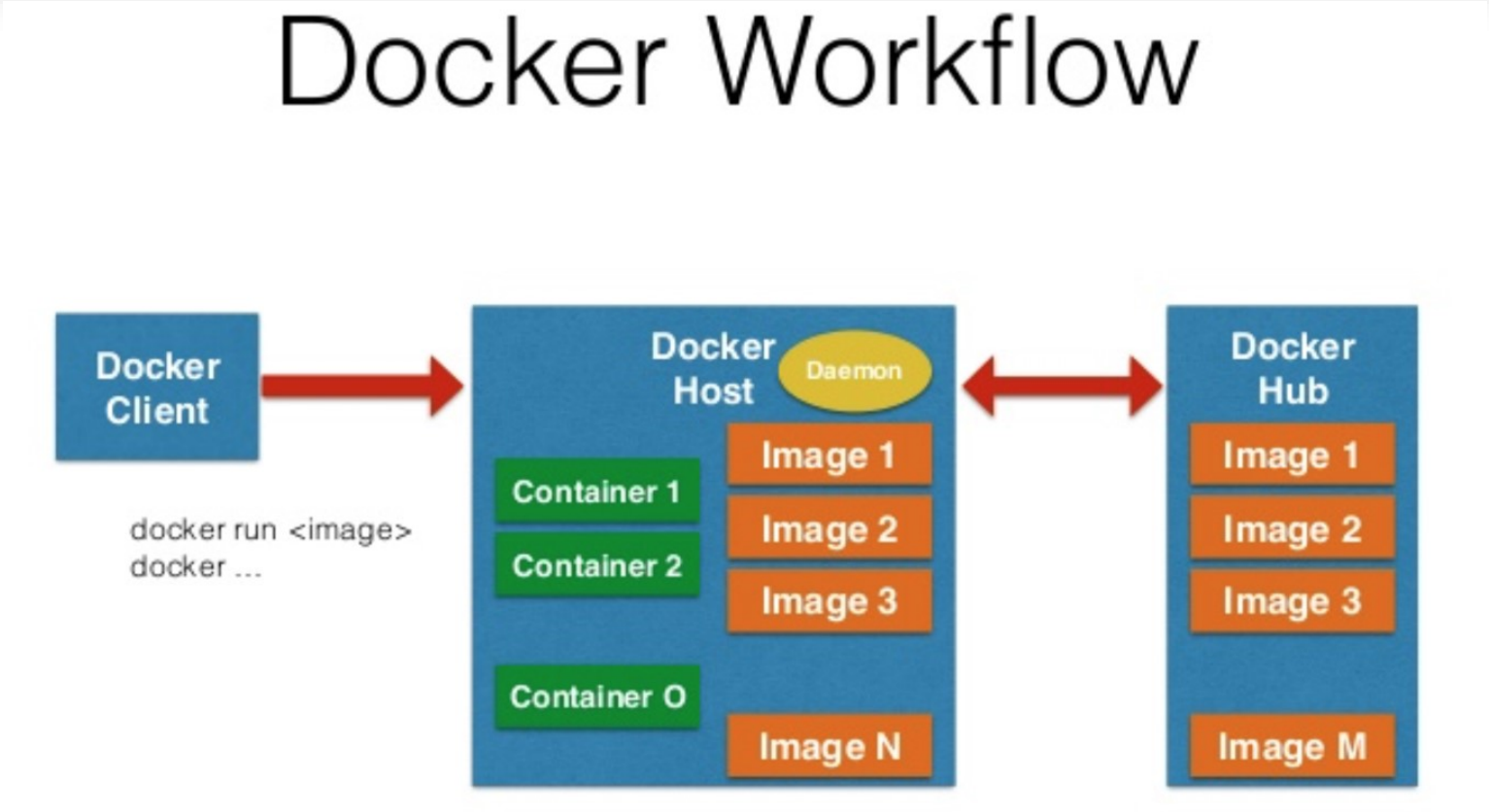
*$ docker exec*

*$ docker stop, rm, rmi*

# Typical Docker pipeline

Install all dependencies and build tool from scratch:
*$ docker build -t parthenos:latest .*
Run image from command line
*$ docker run -p 8081:8081 -name parthenos parthenos*
Check if container is running
*$ docker ps|grep parthenos*
Login inside of the container
*$ docker exec -it [CONTAINER_ID] /bin/bash*
Copy configuration inside of the container
*$ docker cp ./parthenos.config [CONTAINER_ID]:/widget*
Copy from container to local folder
*$ docker [CONTAINER_ID]:/widget/* ./*
*Ship "dockerized" app to the world (Docker Hub or another registry)*
*$ docker push [IMAGE_ID]*

# Pipeline explanation




Credits: Arun Gupta, [Package your Java EE Application using Docker and Kubernetes]

# Docker archiving process

Easy process to archive running software, metadata and data separately

https://docs.docker.com/engine/reference/commandline/save/

- postgresql database with metadata and users information
- datasets files in separate folder
- software image with some individual settings

*$ docker save -o archive.tar [CONTAINER_ID]*

Easy to restore complete system with data and metadata by Docker composer.

*$ docker load archive.tar*

# Docker Compose

Management tool for Docker configuration for multicontainer solutions

All connections, networks, containers, port specifications stored in one file (YML specification)

Example (DataverseEU):

http://github.com/IQSS/dataverse-docker

Tool to turn Docker Compose to Kubernetes config called Kompose:
https://github.com/kubernetes/kompose

Usage:

$ docker-compose [something]

Docker Compose is perfect tool to keep the PROVenance of software (versions control, etc)

# Practical task: Dataverse installation

Install Dataverse locally on your laptop using Docker Module and investigate its infrastructure by accessing running Docker containers.

Software to be installed:

- Git https://git-scm.com/book/en/v2/Getting-Started-Installing-Git
- Docker Desktop https://www.docker.com/products/docker-desktop
- Minikube (optional) https://kubernetes.io/docs/tasks/tools/install-minikube/

Open terminal, run installations, wait and go to http://0.0.0.0:8085 to see Dataverse:

```
git clone http://github.com/IQSS/dataverse-docker
git checkout workshop
./initial.bash
docker-compose build
docker-compose up
```

# When Docker is building images...

```
version: '2'

services:
    postgres:
        build:  postgresql/
        container_name: db
        ports:
        - "5435:5432"
        volumes:
        - ./data/db:/var/lib/postgresql/data
        - ./postgresql/init.sql:/docker-entrypoint-initdb.d/init.sql

    solr:
        build: solr7/
        container_name: solr
        ports:
        - "8985:8983"
        environment:
          - "SOLR_HOST=solr"
          - "SOLR_PORT=8983"
        volumes:
        - ./data/solr:/usr/local/solr-7.3.0/example/solr/collection1/data

    dataverse:
        build: dataversedock/
        container_name: dataverse
        ports:
        - "443:443"
        - "8085:8080"
        environment:
          - "HOST_DNS_ADDRESS=0.0.0.0"
          - "LANG=en"
          - "BUNDLEPROPERTIES=Bundle.properties"
          - "GLASSFISH_DIRECTORY=/opt/glassfish4"
          - "ADMIN_EMAIL=admin@localhost"
          - "MAIL_SERVER=mailrelay"
          - "POSTGRES_ADMIN_PASSWORD=secret"
          - "POSTGRES_SERVER=postgres"
          - "POSTGRES_PORT=5432"
          - "POSTGRES_DATABASE=dvndb"
          - "POSTGRES_USER=dvnapp"
          - "POSTGRES_PASSWORD=secret"
          - "SOLR_LOCATION=solr:8983"
          - "TWORAVENS_LOCATION=NOT INSTALLED"
          - "RSERVE_HOST=localhost"
          - "RSERVE_PORT=6311"
          - "RSERVE_USER=rserve"
          - "RSERVE_PASSWORD=rserve"
        depends_on:
          - postgres
          - solr
        volumes:
```

- Open file docker-compose.yml in any editor (vim by default)
- Inspect the infrastructure to see all Dataverse components and environmental variables
- Check that folders in volumes section are persistent

# Dataverse Docker containers exploration

```
# Show Docker images
docker images
# Show all running containers
docker ps
# Remove Docker image by container_id (don’t execute)
docker rmi container_id
# Delete old images (don’t execute)
docker rmi `docker images -aq`
# To access Dataverse container, type exit to quit
docker exec -it dataverse /bin/bash
# PostgreSQL container, exit to quit
docker exec -it postgres /bin/bash
# Solr container, exit to quit
docker exec -it solr /bin/bash
# Copy files and folders to the running container
docker cp ./testfile dataverse:/tmp/
# Copy files and folders from the running container to your disk space
docker cp dataverse:/opt/dv/dvinstall.zip /tmp/
# Stop Dataverse container
docker stop dataverse
# Run Dataverse container
docker start dataverse
```

# Dataverse maintenance with Docker

```
# Open the page with latest Dataverse release https://github.com/IQSS/dataverse/releases
# Follow the upgrade instruction containing war and zip, optionally .tsv or .xml schema
docker exec -it dataverse /bin/bash
wget https://github.com/IQSS/dataverse/releases/download/v4.18.1/dataverse-4.18.1.war -O dataverse.war
asadmin undeploy dataverse
rm -rf glassfish4/glassfish/domains/domain1/generated
asadmin deploy ./dataverse.war
asadmin restart
# After Glassfish will restart go to 0.0.0.0:8085 and check the version of Dataverse
# Remember: you'll lose all changes in your Docker container after restart!
```

# Maintenance of Docker infrastructure

```
# Go to hub.docker.com and create an account.
# Login with your credentials, remember your_docker_name
docker login
# Let's create image out of the running Dataverse container
docker commit dataverse
# New image will be available on top
docker images
# Let's put a tag on image and update internal Docker registry, replace your_docker_name
docker tag new_dataverse_image_id [your_docker_name]/dataverse:4.18.1
# Push new image to Docker Hub
docker push [your_docker_name]/dataverse:4.18.1
# Go to Docker Hub to check if the repo was updated:
https://hub.docker.com/r/[your_docker_name]/dataverse
# Visit the page https://docs.docker.com/docker-hub/repos/#pushing-a-docker-container-
image-to-docker-hub if your need more information about the update of Docker images
```

# Postgres database management

```
# Change dataverseAdmin password to default

docker exec -i -t postgres /bin/bash

su - postgres

psql dvndb


# Run command to update password to ‘admin’ and quit with \q:
update builtinuser set
encryptedpassword='$2a$10$x5qqRctMqRhehNZf/uesM.ll9pYn7hv9EEGLuODvDMnk93H4xkGTm';
```

Note: all changes in database are permanent, never do something like that in production!

# Reverse engineering based on postgres dump

```
# You can also do reverse engineering of database
pg_dump dvndb > original_db.sql
# Do some actions in Dataverse interface and get new postgres dump
pg_dump dvndb > changed_db.sql
diff  original_db.sql changed_db.sql
161241a161242,161243
> 6a03133d-68d1-4b43-a5ab-43321507fa6b    OK
                edu.harvard.iq.dataverse.engine.command.impl.AssignRoleCommand                Command
                2020-01-19 15:32:19.735                :[8508 New dataset revision 3]       2020-01-19 15:32:19.734
                @dataverseAdmin
> abc5af9a-133c-4d3c-8b75-5cf764f60ed8      OK
                edu.harvard.iq.dataverse.engine.command.impl.CreatePrivateUrlCommand        Command
                2020-01-19 15:32:19.739                :[8508 New dataset revision 3]       2020-01-19 15:32:19.725
```

# pyDataverse practical guide (Stefan)

- How to install and setup pyDataverse, features
- use case: migration of metadata from Excel to Dataverse

(To do: add link to slides)

Practical task: participants should import test datasets to Dataverse and explore the functionality of Dataverse using Jupyter Notebook

https://github.com/AUSSDA/pyDataverse_workshop_tromso

# The development of external applications and other pipelines (James Myers slides)

- Using bagIt pipeline to archive datasets, OAIS model
- external previewers, step-by-step manual how to extend Dataverse with new previewers
- How DVUploader works, with some practical examples
- MakeDataCount contribution
- Support of multiple storages inside of the same Dataverse

Slides

Practical task: install DVUploader and try to move some folder with files to Dataverse

# Deploying Dataverse as a service on Cloud

- Why do you need Cloud?
  - Community service, CESSDA Technical Infrastructure
- How to choose Cloud provider?
  - Deploying Dataverse on Google Cloud, Amazon AWS, Microsoft Azure and OpenShift Container Platform
- How to maintain a Cloud service?
- Is it expensive?

# CESSDA Technical Infrastructure

Basic “requirements”

» restricted BitBucket Git
» Docker containers
» Kubernetes (K8S) @ Google cloud (Docker orchestrator)
» Use descriptor files (YAML) to define K8S Services and Deployments
» Jenkins; Continuous Deployment and Integration pipeline
» Infrastructure as code (IaC)

Benefits:

» Scalable cluster (horizontal and vertical)
» Docker containers are managed by K8S cluster manager
» Portable applications (IaC)

# Google Cloud Platform

Resources needed to deploy Dataverse:

- GCP VM-instance(s)
- Kubernetes (K8S) cluster
- Docker images
- K8S Services
- K8S Deployments (workloads)
- Load Balancer or Ingress (SSL certificates)
- Mail relay (default port 25 is blocked)
- Persistent volumes (storage) for the Solr, Postgres, Dataverse and SSL services

# Integration with Google Cloud

Google Cloud Platform

» GCloud

a command-line (CLI) tool for running commands against Google Cloud Platform. GCloud is part of the Google Cloud SDK

Kubernetes cluster

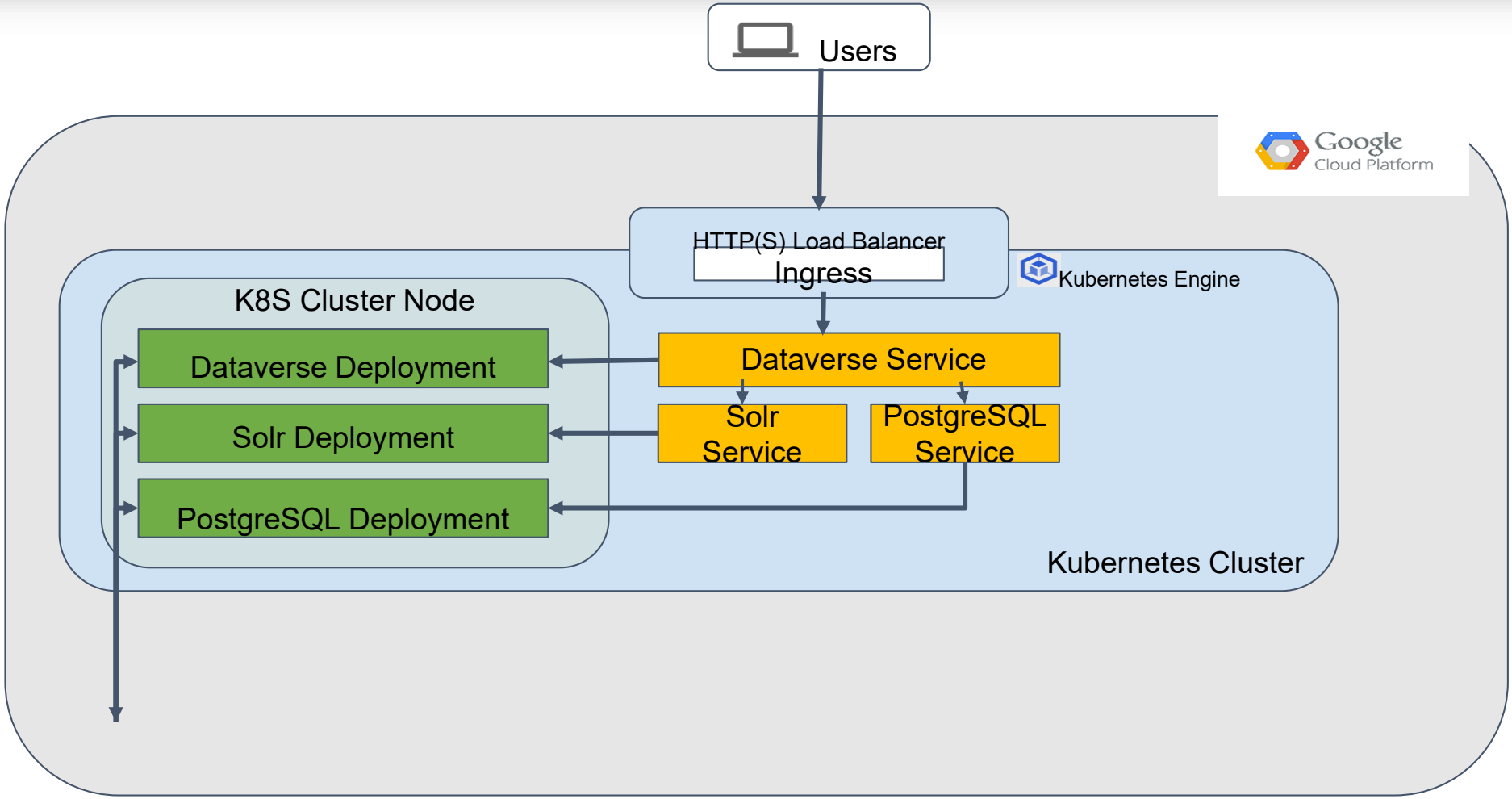
» Kubectl

A command line interface (CLI) for running commands against Kubernetes clusters

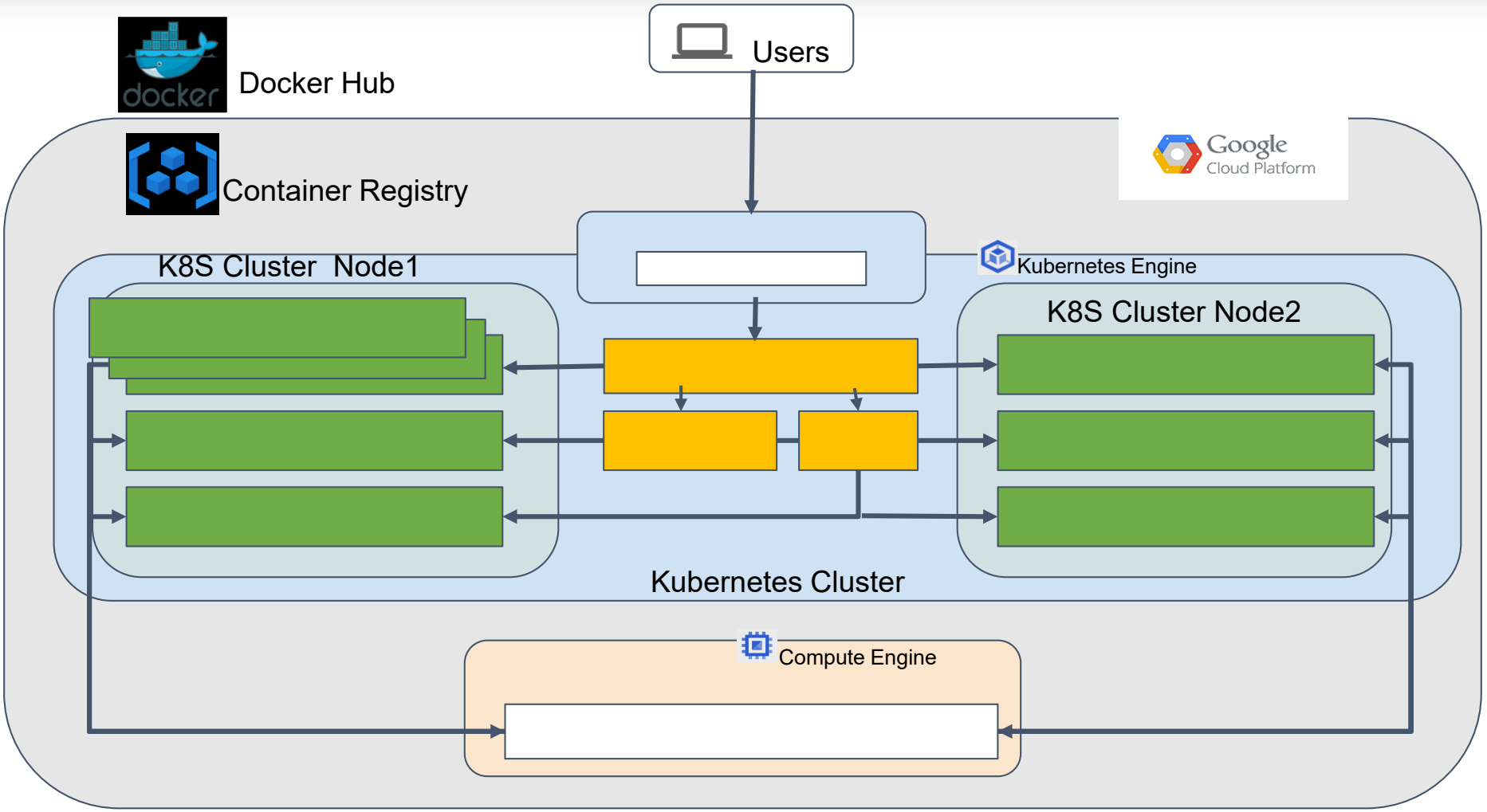
» Use of .yaml configuration files to describe the desired resources

Note: Kubernetes software is used by all major Cloud providers, the different is the storage specification!

# Dataverse Cloud architecture

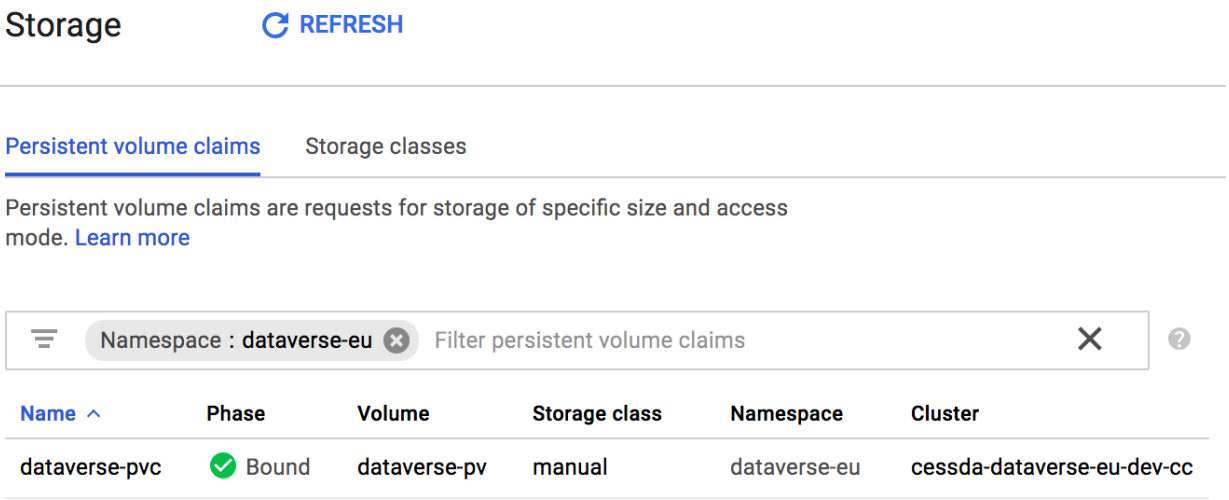


# How to scale up Kubernetes horizontally



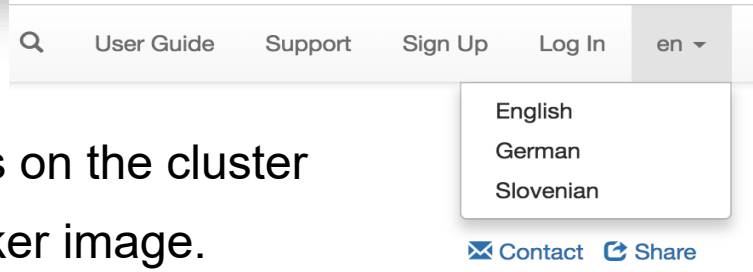
# The importance of Persistent Storage

Docker containers write files to disk (I/O) for state or storage, both in /data and /docroot folders. If a Docker container is restarted for some reason, all data will be **lost**.




Solution: mount Persistent storage into the container on external disk hosted in the Cloud.

# Internalization (i18n)



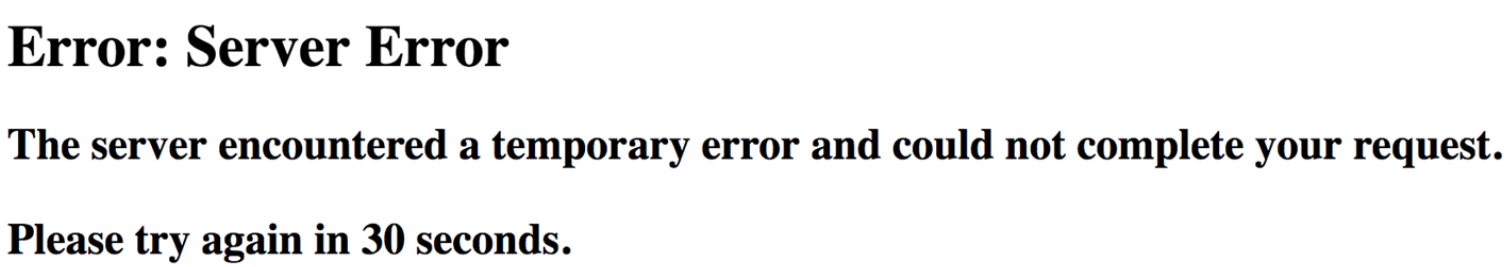

Run multiple localized Dataverse docker containers on the cluster

Switch redirects to domain running a localized docker image.

| | | | | | | |
|---|---|---|---|---|---|---|
| dataverse-sl | Ok | Node Port | 10.31.244.217:8080 TCP | 1 / 1 | dataverse-eu | cessda-dataverse-eu-dev-cc |
| dataverse-en | Ok | Node Port | 10.31.246.207:8080 TCP | 1 / 1 | dataverse-eu | cessda-dataverse-eu-dev-cc |
| dataverse-dev-ingress-service | Ok | Ingress | dataverse-dev.cessda.eu<br>dataverse-dev.cessda.eu/.well-known/acme-challenge/*<br>uk.dataverse-dev.cessda.eu<br>uk.dataverse-dev.cessda.eu/.well-known/acme-challenge/*<br>de.dataverse-dev.cessda.eu<br>de.dataverse-dev.cessda.eu/.well-known/acme-challenge/*<br>at.dataverse-dev.cessda.eu<br>at.dataverse-dev.cessda.eu/.well-known/acme-challenge/*<br>sl.dataverse-dev.cessda.eu<br>sl.dataverse-dev.cessda.eu/.well-known/acme-challenge/*<br>si.dataverse-dev.cessda.eu<br>si.dataverse-dev.cessda.eu/.well-known/acme-challenge/*<br>Less | 0 / 0 | dataverse-eu | cessda-dataverse-eu-dev-cc |
| dataverse-de | Ok | Node Port | 10.31.252.225:8080 TCP | 1 / 1 | dataverse-eu | cessda-dataverse-eu-dev-cc |

# Health checks

Creation of Health Checks and ReadinessProbes on your container will allow to detect problems with deployment:




K8S will create a default Health Check for you on the Ingress/Load balancer if you do not supply one

*Solution*:

It will check for HTTP 200 OK at your applications root ‘/’ or the path you supplied to your custom readinessprobe

# Setting up E-mail functionality

*Problem*:
SMTP outgoing port 25 (smtp) is blocked on GCP

*Solution*:

Deploy a mail-relay container on the cluster, that listens (internally) on port 25, that relays your email to G Suite server over HTTPS, that can send the email

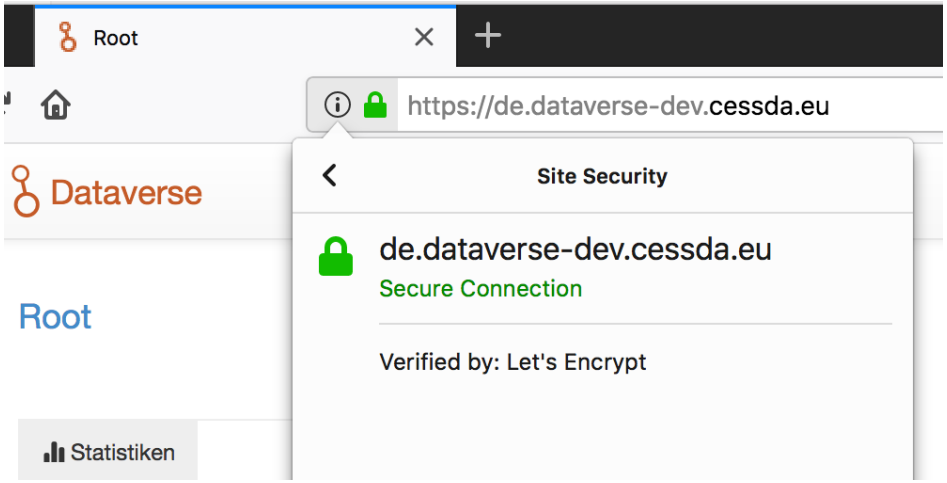
From Dataverse container — TCP/25 —> Mail Relay container — Transport Layer Security (TLS) —> G Suite server —> Receiver inbox

The mail-relay container is available for CESSDA projects from the CESSDA docker Container Registry

# HTTPS support, SSL Certification

*Problem:*

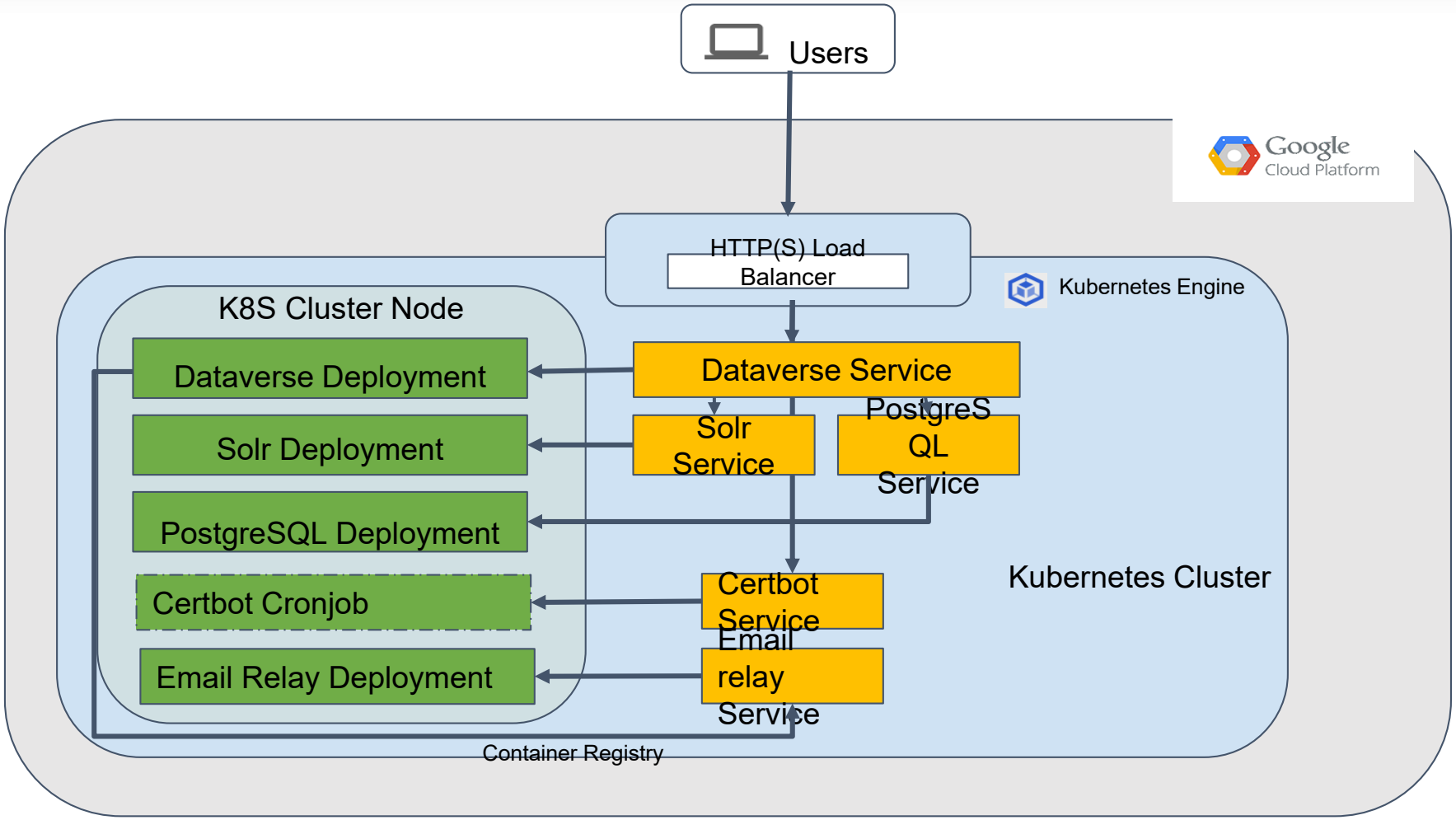
Let’s Encrypt SSL certificates expire (90 days)

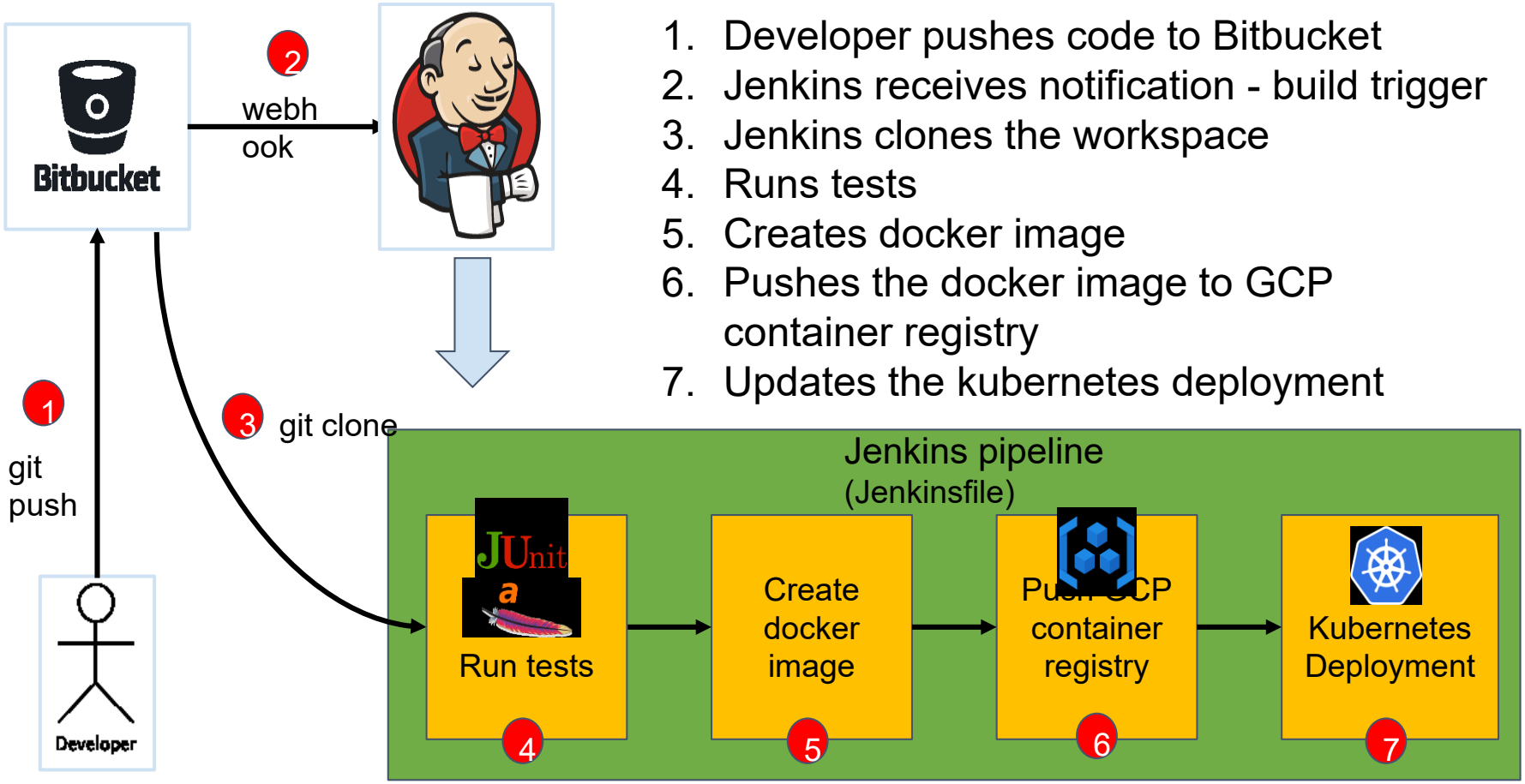

*Solution*:

Automatic Let's Encrypt certification renewal with ‘Certbot’ docker image in a Kubernetes cronjob.

# Running Dataverse in production

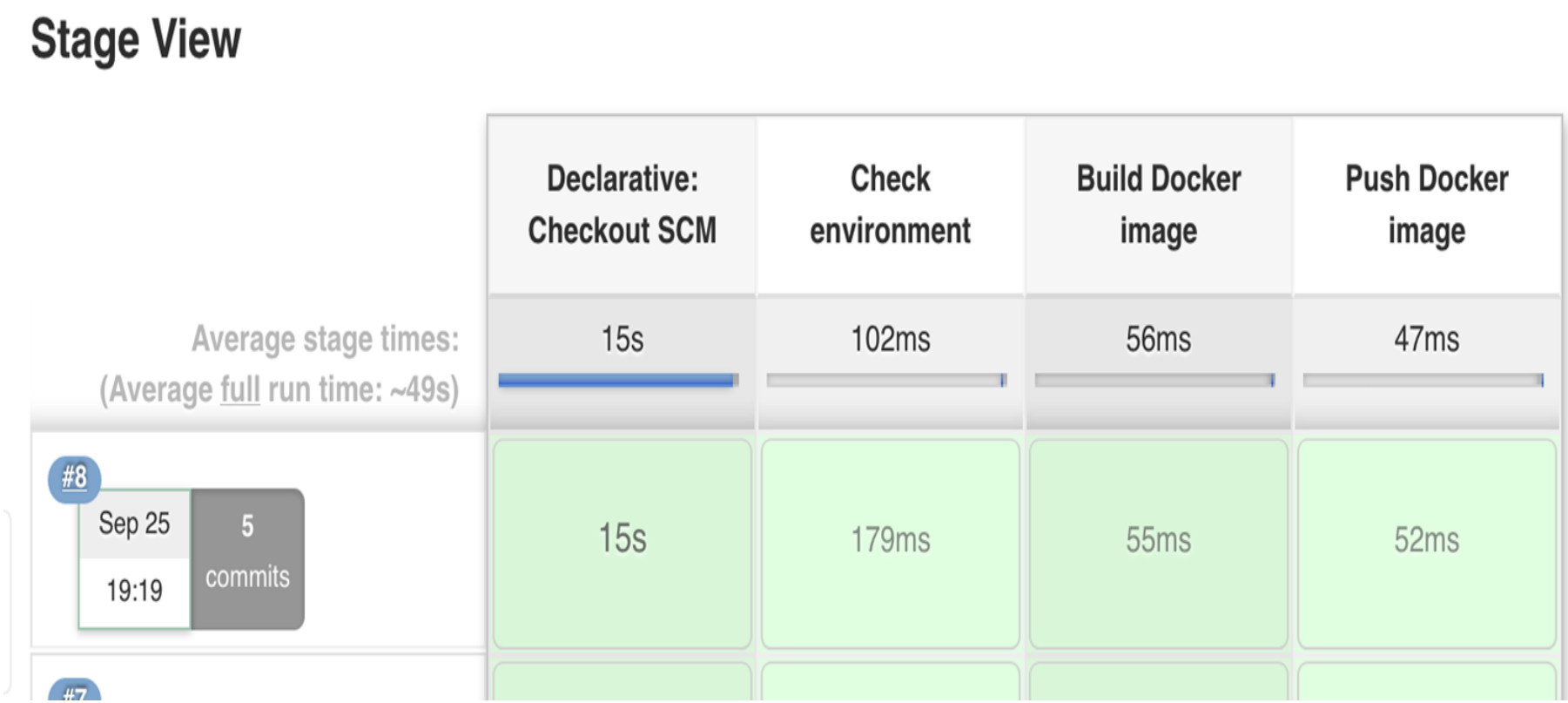


# Continuous deployment pipeline




1. Developer pushes code to Bitbucket
2. Jenkins receives notification - build trigger
3. Jenkins clones the workspace
4. Runs tests
5. Creates docker image
6. Pushes the docker image to GCP container registry
7. Updates the kubernetes deployment

# Continuous delivery



# Running Dataverse on Kubernetes (Oliver)

- Kubernetes infrastructure: secrets, storage, Dataverse upgrades, Cloud compatibility and kustomize module
- K8s bootstrap jobs
- Maintenance of Docker images with Jenkins pipeline, assigning tags strategy
- Dynamic update of metadata fields in K8s (.tsv and Solr)
- Tips and suggestions to engage developers to be involved in Dataverse Kubernetes development, attract new users
- Migration progress from Glassfish to Payara 5
- OpenID Connect support in Dataverse

Practical task: deploy Dataverse on Google Cloud using dataverse-kubernetes module.

Slides

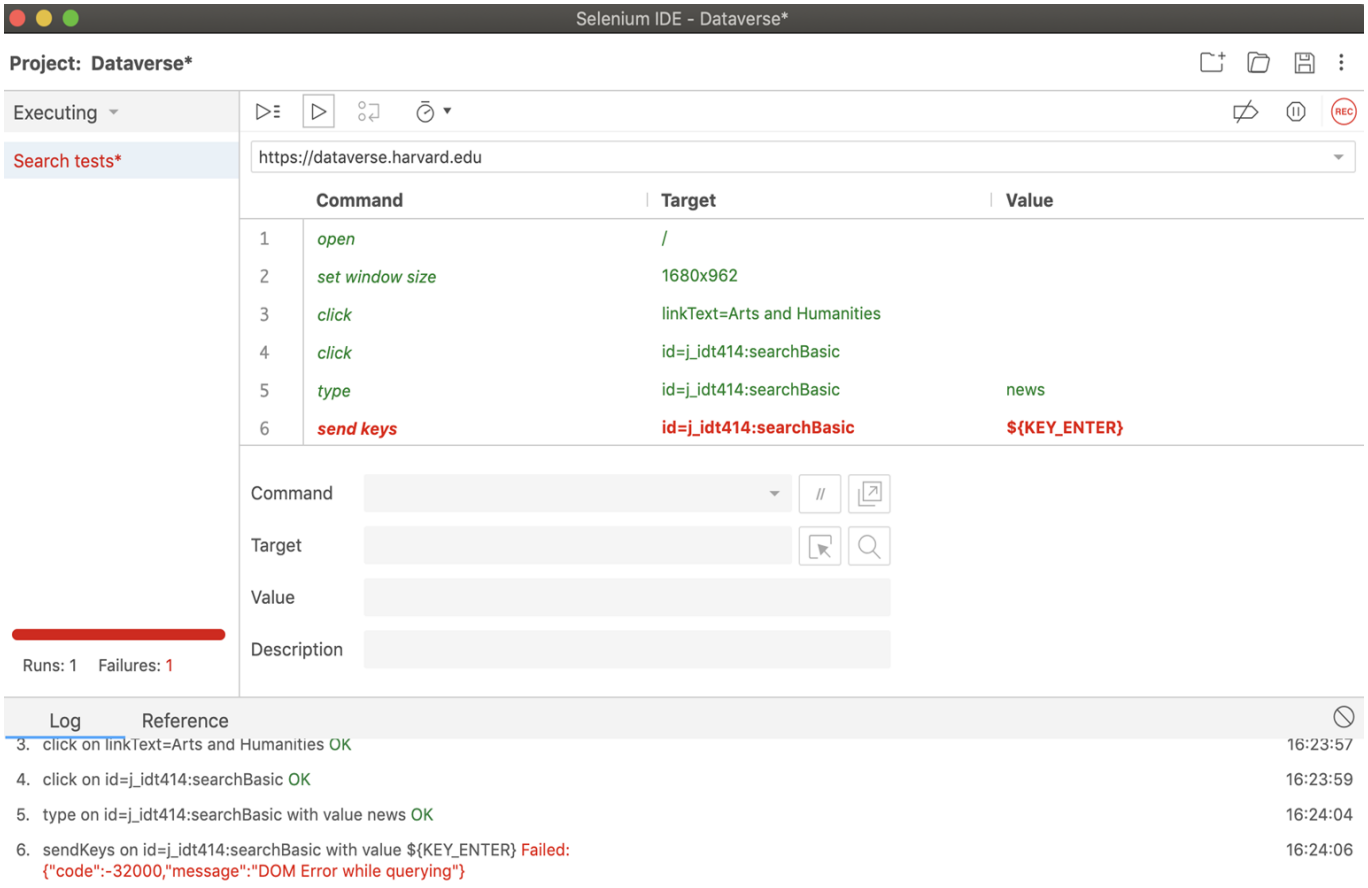
# Distributed Dataverse infra on Kubernetes

- Network of Dataverses with central portal to host metadata and multiple Dataverse nodes
- Testing strategies with Selenium and Cypress
- Unit tests, integration tests and Jenkins CI/CD pipeline
- Running external applications on Kubernetes infrastructure, OpenAIRE Amnesia tool
- Multiple languages support and maintenance, Weblate as a service
- Using iRODS to support multiple storages for different datasets

# Before we'll start...

- The maintenance of the distributed applications is very difficult and expensive
- requires the highest level of service maturity
- increasing the **code** coverage does not necessarily lead to more **functionality** coverage
- writing integration tests even more important than adding more unit tests
- it's almost not possible to run distributed services without the help from community

# Quality Assurance (QA) as a community service




Selenium IDE allows to create and replay all UI tests in your browser

Shared tests can be reused by Dataverse CI/CD pipeline

Let's work together on it!

# Example of Selenium .side file


```
"id": "db43851a-926b-480e-972d-4596903c1857",
"version": "2.0",
"name": "Dataverse",
"url": "https://dataverse.harvard.edu",
"tests": [{
  "id": "801cee96-729c-4a9a-9798-1020db0a74ce",
  "name": "test1",
  "commands": [{
    "id": "53f18a71-fb98-48d0-b674-4898cf317ac5",
    "comment": "",
    "command": "open",
    "target": "/",
    "targets": [],
    "value": ""
  }, {
    "id": "08334d34-8a27-44ff-b029-e3b253531753",
    "comment": "",
    "command": "setWindowSize",
    "target": "1680x962",
    "targets": [],
    "value": ""
  }, {
    "id": "595aedf8-797d-4901-8c00-e39e5033e4a9",
    "comment": "",
    "command": "type",
    "target": "name=q",
    "targets": [
      ["name=q", "name"],
      ["css=.gLFyf", "css:finder"],
      ["xpath=//input[@name='q']", "xpath:attributes"],
      ["xpath=//form[@id='tsf']/div[2]/div/div/div/div[2]/input", "xpath:idRelative"],
      ["xpath=//div[2]/input", "xpath:position"]
    ],
    "value": "https://harvard.dataverse.edu/"
  }, {
    "id": "d30c7da0-1dd8-49d5-90d7-abc8f500f793",
    "comment": "",
    "command": "sendKeys",
    "target": "name=q",
    "targets": [
      ["name=q", "name"],
      ["css=.gLFyf", "css:finder"],
      ["xpath=//input[@name='q']", "xpath:attributes"],
      ["xpath=//form[@id='tsf']/div[2]/div/div/div/div[2]/input", "xpath:idRelative"],
      ["xpath=//div[2]/input", "xpath:position"]
    ],
    "value": "${KEY_ENTER}"
  }, {
    "id": "8eb2d2f8-4f8d-4fc3-a760-f3f4962311a8",
    "comment": "",
    "command": "click",
```


- .side is the extension for the new selenium ide tests
- json format, every section describes some action
- template rules can be used by Selenium webdriver
- can be easily integrated in Continuous deployment pipeline with Jenkins jobs
- running SIDE Runner with the given parameters can even test the different components!

# Community-driven QA plan

- Download test-suite with all tests provided by community
- Install Selenium IDE in Google Chrome
- Create new project in Selenium and upload tests
- Run all tests and provide feedback back to the community
- Everybody can create a new tests specific to your requirements, record and share with others
- more contributors will bring even more maturity of Dataverse services with CI/CD pipeline
- it means Dataverse maintenance cost will start to drop down

maturity of the community = maturity of services

# Community-driven multilingual support

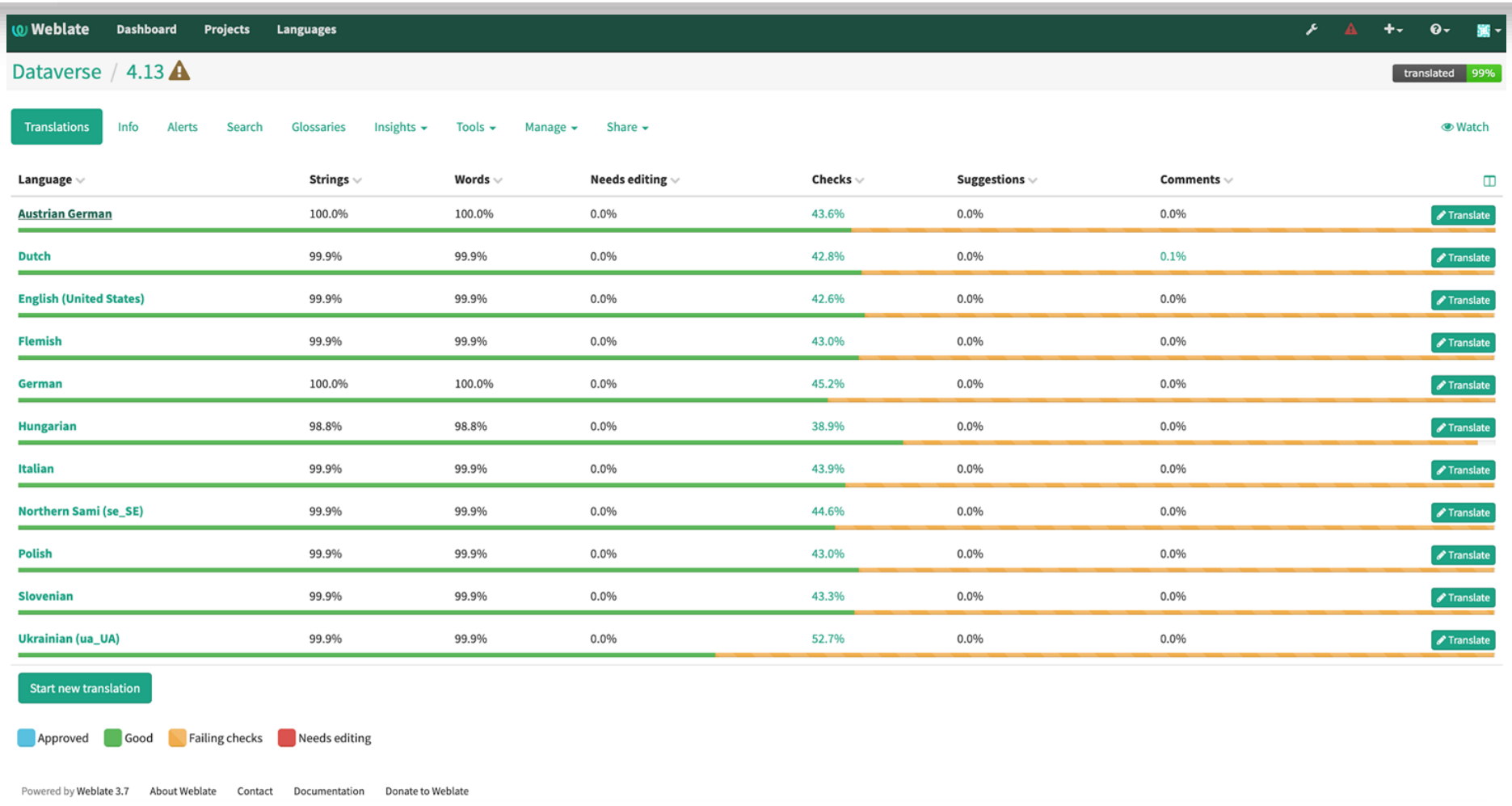
Dataverse language packages developed by Scholars Portal and CESSDA Dataverse community and preserved by GDCC:

https://github.com/GlobalDataverseCommunityConsortium/dataverse-language-packs

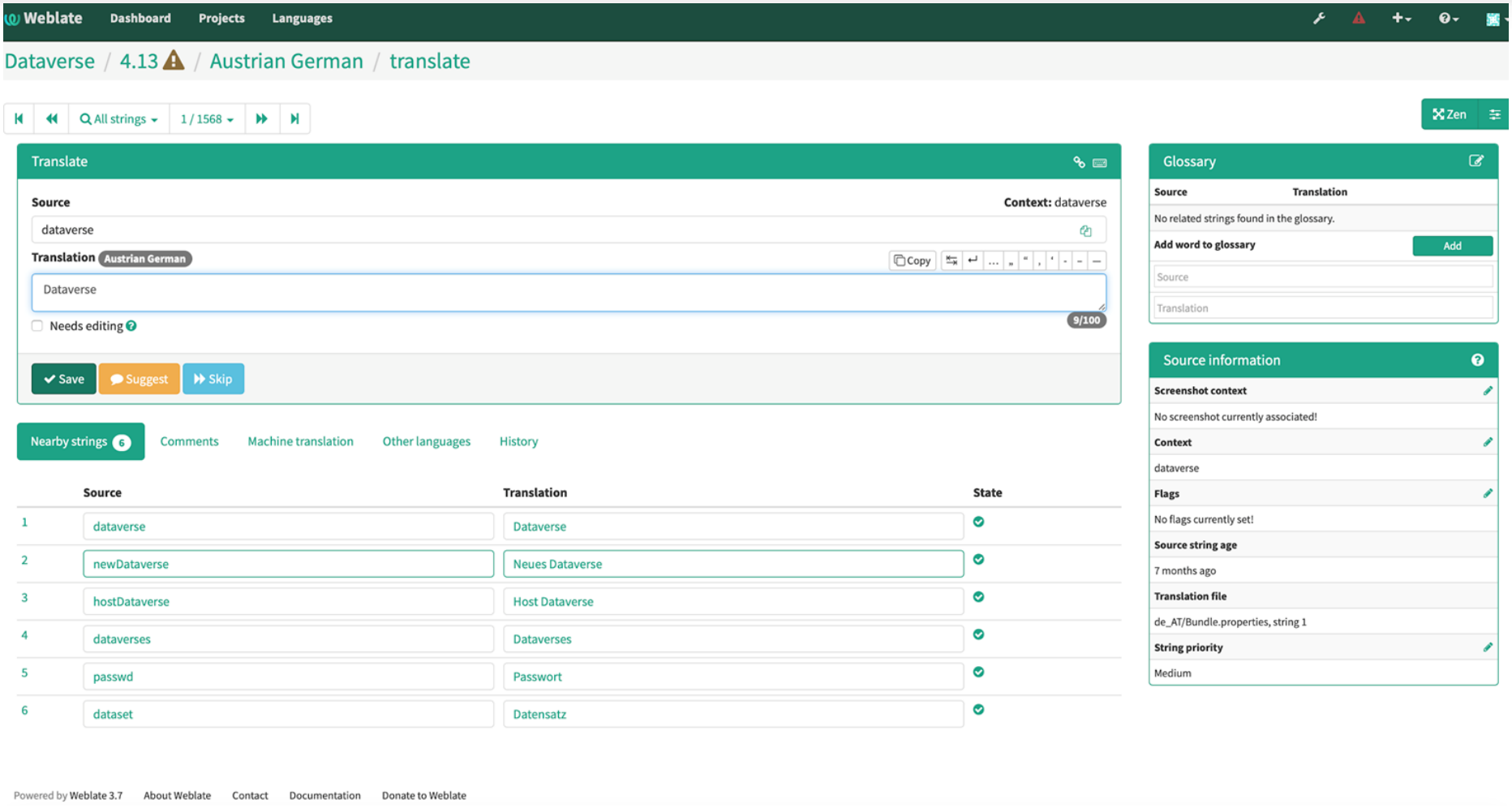
However, it’s difficult to synchronize and maintain all translations without using a tool.

CESSDA Dataverse team started to build a translation community service out of Weblate tool.

# Weblate as a multilingual support service



# Managing translations with Weblate



# Integration of translated properties in CI/CD

- Weblate will be deployed as a service running on Kubernetes
- every Dataverse partner can get an account to manage translation in own language
- translators will get notifications about new Dataverse properties delivered by webhooks after new release
- contributed translations will be moderated and pushed back to GDCC language package (github)
- Jenkins jobs should get language updates, rebuild images and run test packages
- we need community-driven Selenium tests to run Weblate as a service!

# Practical task: create test(s) and run a network of Dataverses

Practical task: Install Selenium IDE as Google Chrome plug-in and create test case for your own Dataverse.

Run this test and save as .side file.

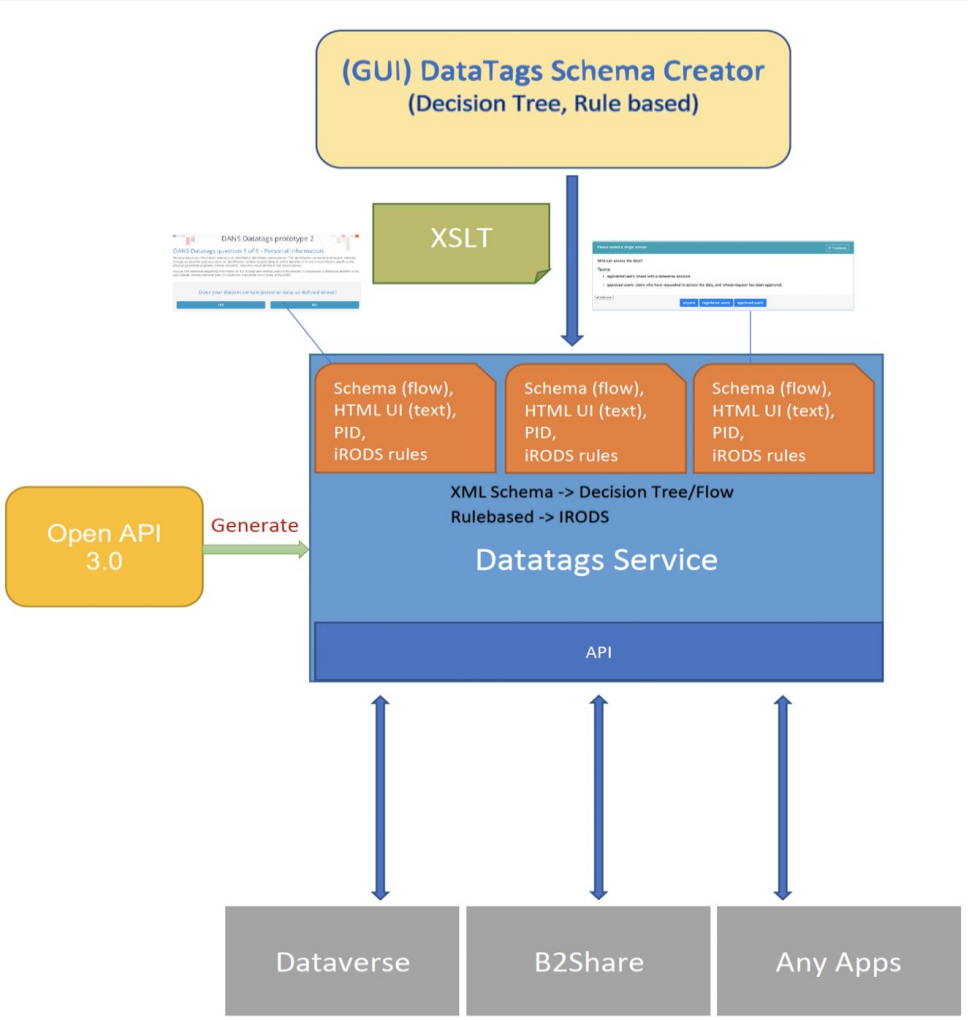
# Sensitive data support and roadmap (Gustavo)

- DataTags concept and integration with Dataverse
- Future Dataverse integrations (ORCID, etc) and new workflows, Whole Tale features
- Challenges in the management of the growing community

# Dataverse in Europe: vision and challenges

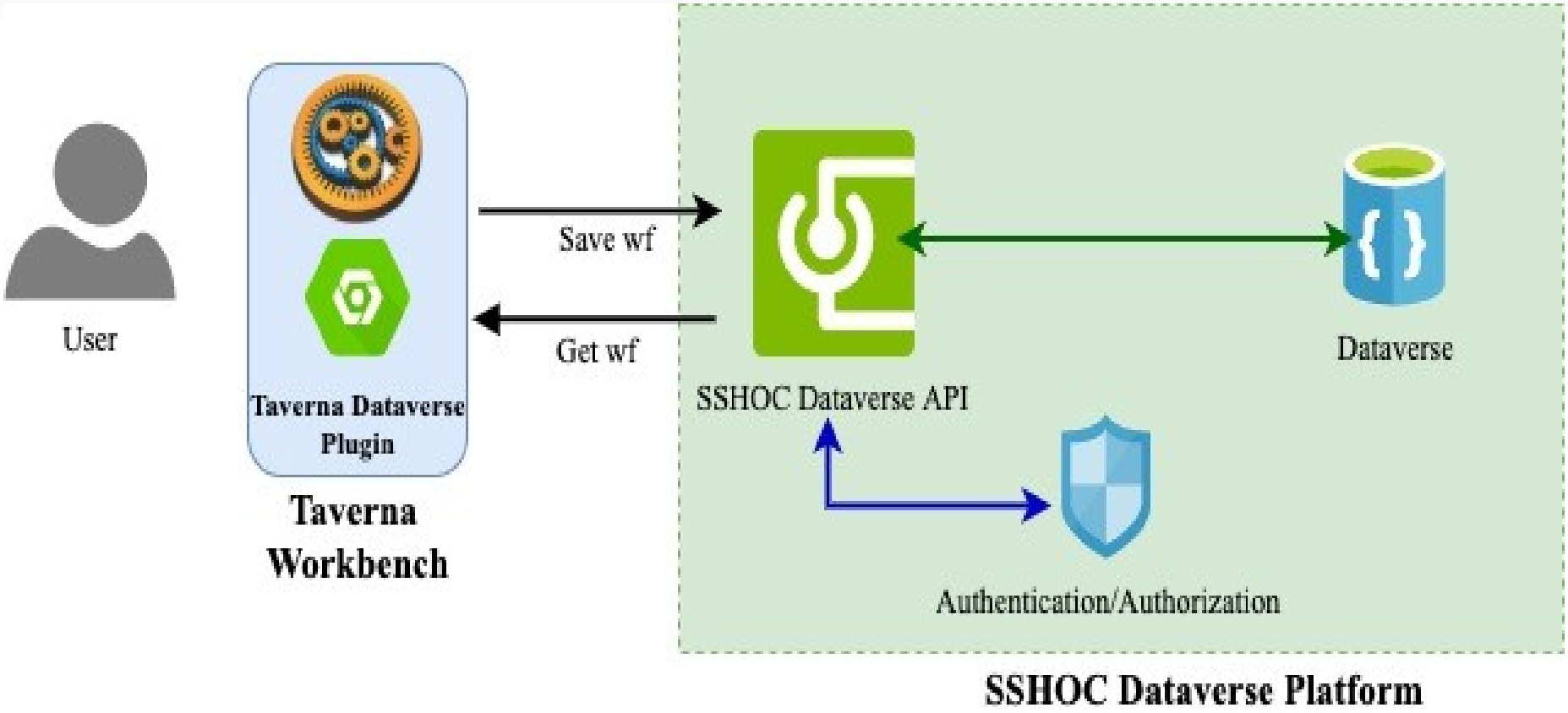
- Support of the Large Scale projects like Time Machine and Digging Into Data
- GDPR compliance
- FAIR Dataverse
- Archiving Docker containers and Kubernetes infrastructure in Dataverse, making a move to Reproducible Science
- Apache Taverna integration with Dataverse to design, execute, preserve and replay research workflows

# GDPR Compliance and DataTags in EU




- DataTags implemented as “yet” another external application
- Deployed on EOSC Cloud as a service
- can be connected to any data repository, not only to Dataverse (B2Share)
- all policies are handled as XML schemas
- Swagger API support
- Flexible configuration to connect the same DataTags instance to all apps

# Apache Taverna integration with Dataverse




Taverna-Dataverse plugin, designed and developed in SSHOC by ISTI-CNR (Italy)

# Questions

Thank you!

We’re more than happy to answer all of your questions!

Gustavo Durand (Harvard IQSS, USA)
Oliver Bertuch (FZ Jülich, Germany)
Stefan Kasberger (AUSSDA, Austria)
James Myers (GDCC, USA)
Slava Tykhonov (DANS-KNAW, Netherlands)