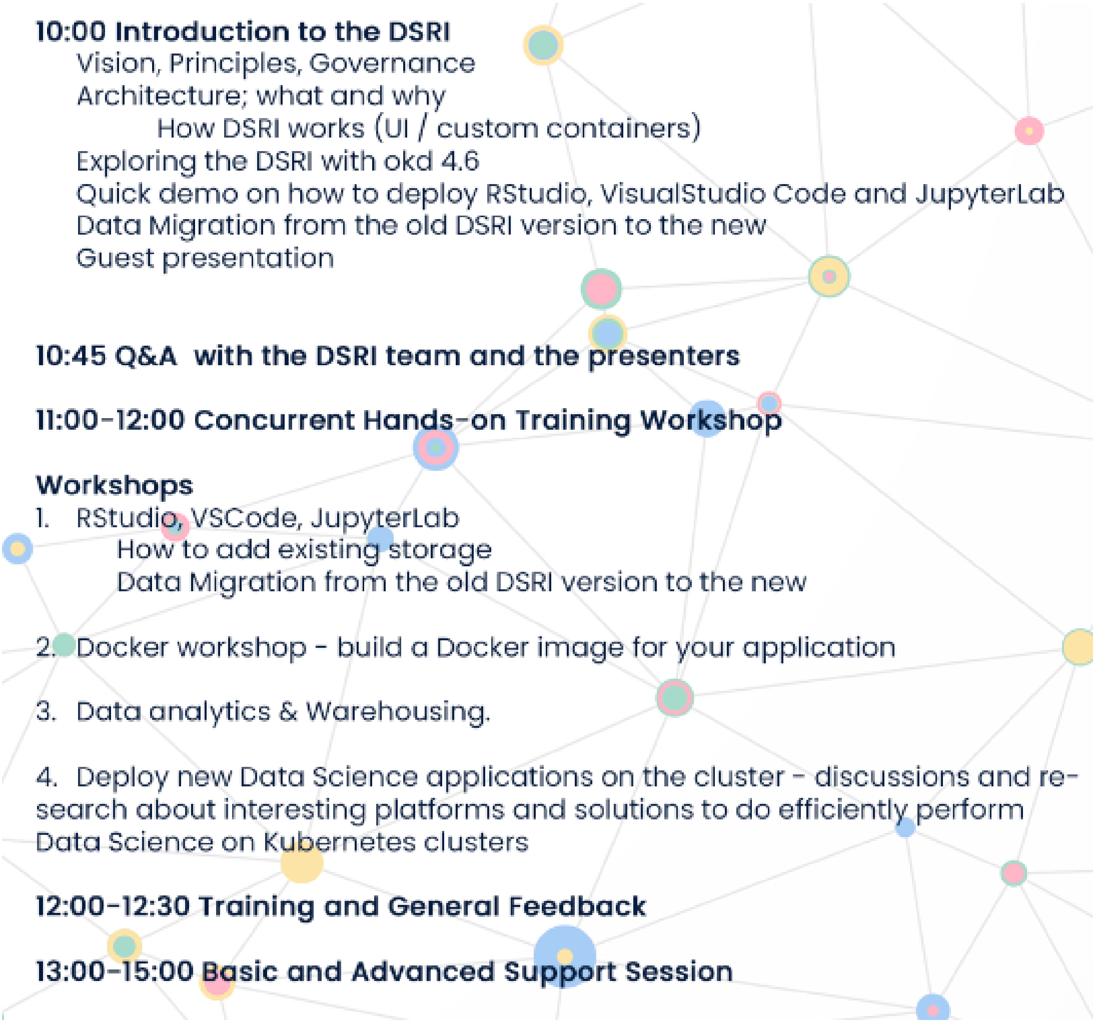
Maastricht University

Institute of Data Science

DSRI Community Event 2021

7th April 2021

Agenda



10:00 Introduction to the DSRI

Vision, Principles, Governance

Architecture; what and why

How DSRI works (UI / custom containers)

Exploring the DSRI with okd 4.6

Quick demo on how to deploy RStudio, VisualStudio Code and JupyterLab

Data Migration from the old DSRI version to the new

Guest presentation

10:45 Q&A with the DSRI team and the presenters

11:00-12:00 Concurrent Hands-on Training Workshop

Workshops

1. RStudio, VSCode, JupyterLab
   How to add existing storage
   Data Migration from the old DSRI version to the new

2. Docker workshop - build a Docker image for your application

3. Data analytics & Warehousing.

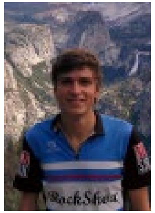
4. Deploy new Data Science applications on the cluster - discussions and research about interesting platforms and solutions to do efficiently perform Data Science on Kubernetes clusters

12:00-12:30 Training and General Feedback

13:00-15:00 Basic and Advanced Support Session

DSRI Team

Michel Dumontier
Institute of Data Science
Project Lead



Vincent Emonet
Institute of Data Science
Support

Binosha Weerarathna
Institute of Data Science
User outreach and training

Rob Schlooz
Institute of Data Science
Project Manager

Chris Kuipers
ICTS
Linux System Engineer

Marcel Brouwers
ICTS
Linux System Engineer

Jordy Frijns
ICTS
Linux System Engineer

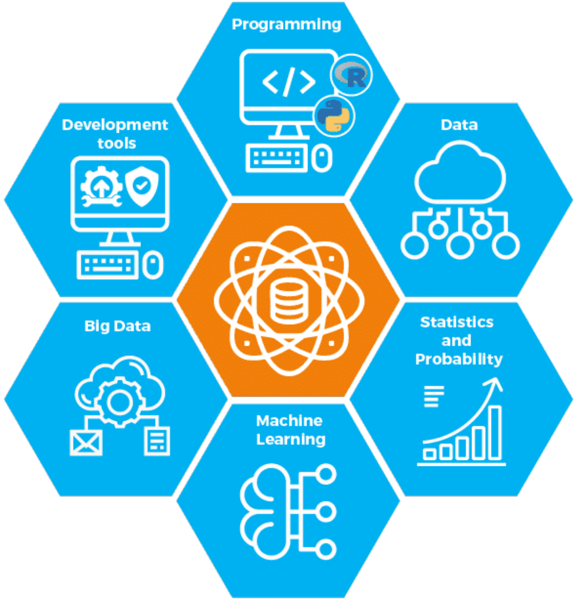
Armand Habets
ICTS
Product Manager

Emiel Kremers
Fourco
Consultant

Arjen van Wijngaarden
Fourco
Consultant

Vision

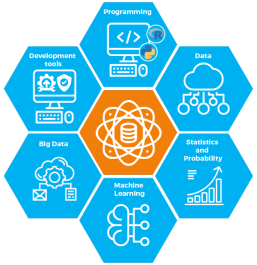
An effective, scalable, and sustainable data science computing infrastructure at Maastricht University



initiated in 2018 as a collaboration between the Institute of Data Science and ICTS

Vision

An effective, scalable, and sustainable data science computing infrastructure at Maastricht University



Effective in that DSRI helps you get data science work done quickly and with less effort

Scalable in both that you can use more resources for your problem, and that we can grow the cluster when needed

Sustainable in that it is an infrastructure that is maintained by UM and its community of users

Why is DSRI needed?

1. **Lack of a shared research computing infrastructure** has resulted in *multiple isolated, incompatible, and independently managed infrastructures* that have differing policies and patchy compliance to organizational, national and international regulations, that cannot be combined.
2. **Researchers should focus on their research**, instead of being burdened with administrating computational infrastructure
3. UM wants to make research results **FAIR** - Findable, Accessible, Interoperable, Reusable - a shared infrastructure would foster best practices to help researchers achieve **FAIR and reproducible research and workflows**.
4. A shared infrastructure will enhance the position of the UM and help **attract and retain data science talent**

Design Objectives

An infrastructure that

- **Facilitates large scale data analysis** using big data technologies using both CPU + GPU computing
- Enables **component deployment via containers** (Docker)
- Enables **data sharing** via a flexible and shared storage solution
- Reduces administrative overhead with **self-administrative user interfaces**
- Is **scalable and fault-tolerant** by combining global monitoring with auto-migration

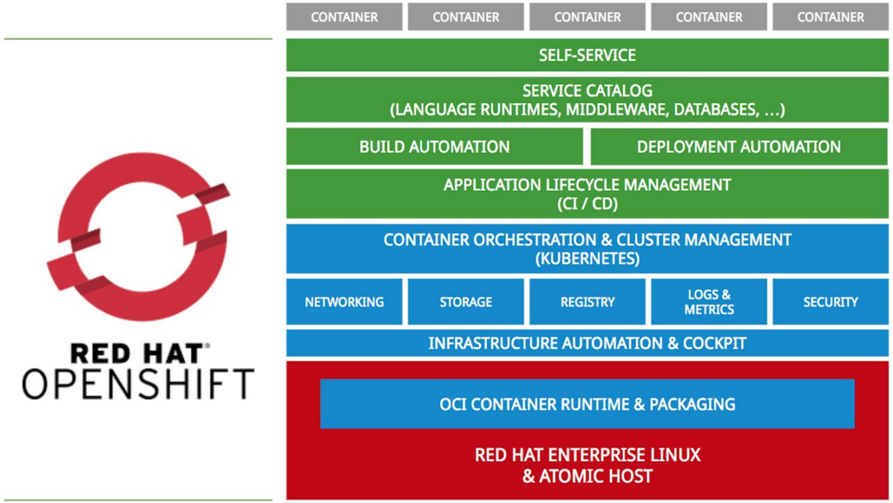
An Orchestrated Solution

Automated configuration, coordination, and management of DSRI

Orchestration using OpenShift and Kubernetes

Ceph storage

Runs containers → Open Containers Initiative



RED HAT OPENSHIFT

CONTAINER | CONTAINER | CONTAINER | CONTAINER | CONTAINER

SELF-SERVICE

SERVICE CATALOG
(LANGUAGE RUNTIMES, MIDDLEWARE, DATABASES, ...)

BUILD AUTOMATION | DEPLOYMENT AUTOMATION

APPLICATION LIFECYCLE MANAGEMENT
(CI / CD)

CONTAINER ORCHESTRATION & CLUSTER MANAGEMENT
(KUBERNETES)

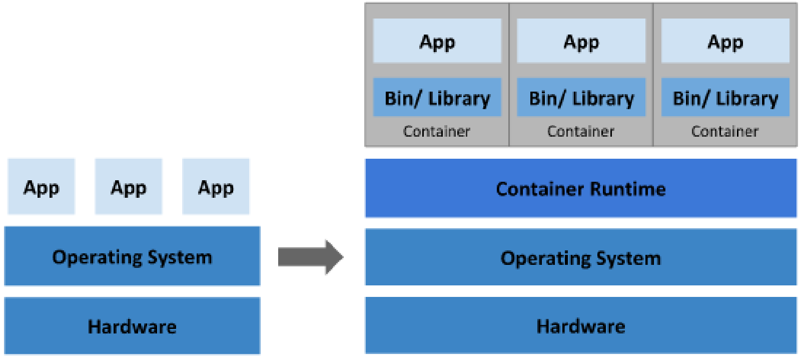
NETWORKING | STORAGE | REGISTRY | LOGS & METRICS | SECURITY

INFRASTRUCTURE AUTOMATION & COCKPIT

OCI CONTAINER RUNTIME & PACKAGING

RED HAT ENTERPRISE LINUX
& ATOMIC HOST

Containers have exactly what is needed to deploy an application



App App App

Operating System

Hardware

Traditional Deployment

App | App | App

Bin/ Library | Bin/ Library | Bin/ Library

Container | Container | Container

Container Runtime

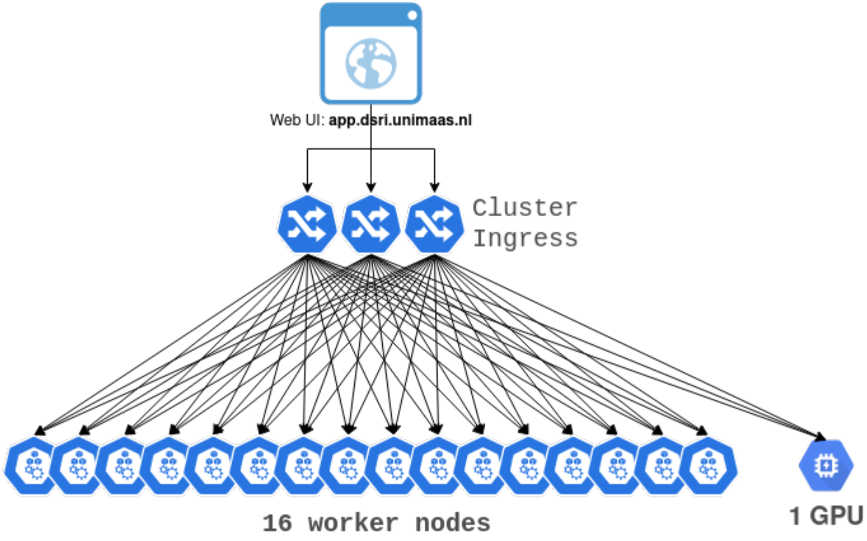
Operating System

Hardware

Container Deployment

- Applications are prepared with everything that is required to successfully deploy them *elsewhere*
- Cloud and OS portability: runs on Ubuntu, RHEL, on-premises, and in major public clouds
- Higher efficiency in using underlying compute resources through load balancing and scaleout
- Protect underlying systems from application specific exploits
- Easy for users to find and redeploy specific apps for their own use

DSRI configuration



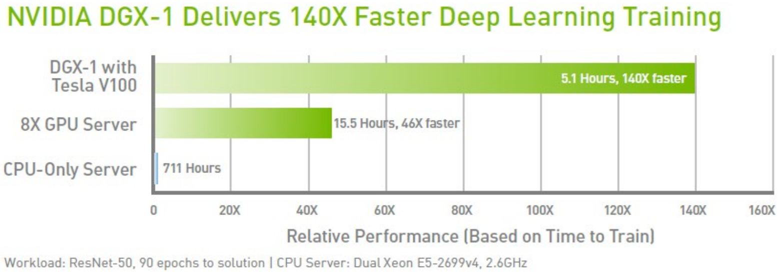
16x CPU nodes

- 2x AMD EPYC 7551
- 512 GB Memory
- 120TB (1920TB total)

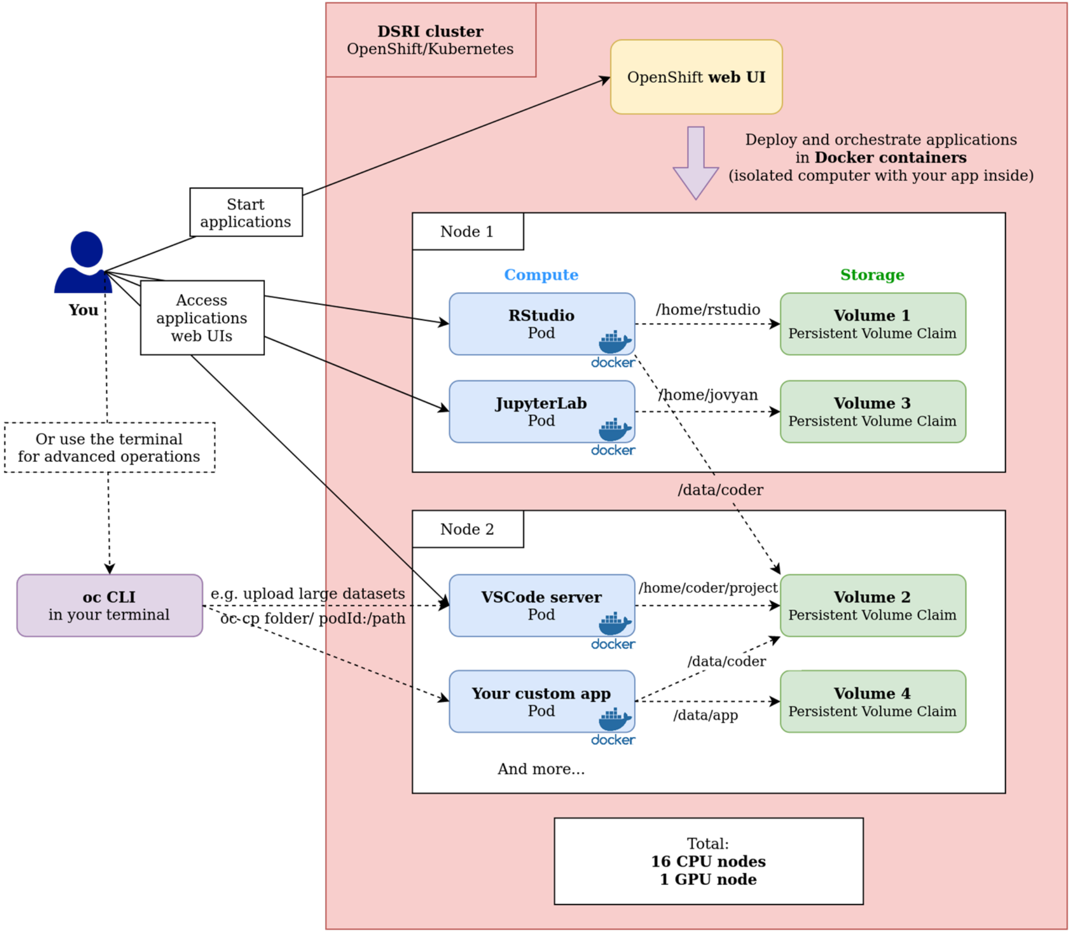
1x GPU node (Nvidia DGX-1)

- 8x NVIDIA Tesla V100 32 GB/GPU
- 40,960 Nvidia CUDA cores
- 5,120 Tensor Cores

40 Gb/s interconnects



DSRI in a nutshell (or any other shell)



What can be done on the DSRI

- Run **Data Science applications** in Docker container on the UM network
  - JupyterLab (scipy, tensorflow, all-spark, and more)
  - RStudio, with a complementary Shiny server
  - VisualStudio Code server
  - Tensorflow or PyTorch on Nvidia GPU
  - SQL, NoSQL and Graph databases (PostgreSQL, MongoDB, Blazegrapl
  - Apache Flink cluster for Streaming applications
  - Apache Spark cluster for parallel computing
  - JupyterHub with GitHub authentication
- You can also deploy **any customized container image** (Docker)

The caveats

You can **deploy any application you want** from a Docker image (usually existing), with more resources (CPU, memory, storage) than your laptop. **But…**

- You are deploying an **application accessible from the web**
  - Within UM net, but security is not to be underestimated!
  - Use good passwords
  - Avoid applications that are exposing web console without login (anyone could run anything in your app), or add a proxy/gateway
- With great security comes **extra config**
  - OpenShift comes with additional security
  - Most Docker images run using the root user, this requires to edit the application deployment to use the *anyuid* service account
  - More tuning might be required for complex apps which require advanced permissions, e.g. related to storage or network
- Everything is on the DSRI servers, **not your laptop**
  - Need to upload your data to the DSRI storage (more sustainable and safe on the long run)
  - Desktop UI might be harder to expose and access from your web browser
  - The distributed storage can make reading/writing files or objects slower than on a centralized, let us know if you are experiencing any issues

Exploring the DSRI with okd 4.6

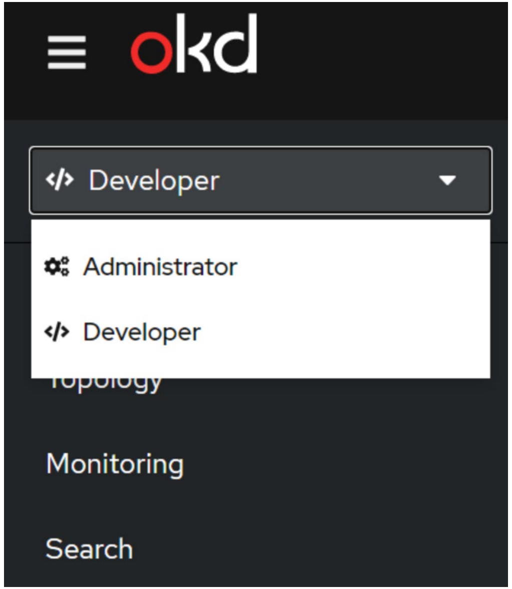
Developers can use the web console to visualize, browse, and manage the contents of projects in the new version of OKD4.

The OpenShift Container Platform web console provides two perspectives;

- the **Administrator** perspective
- the **Developer** perspective.

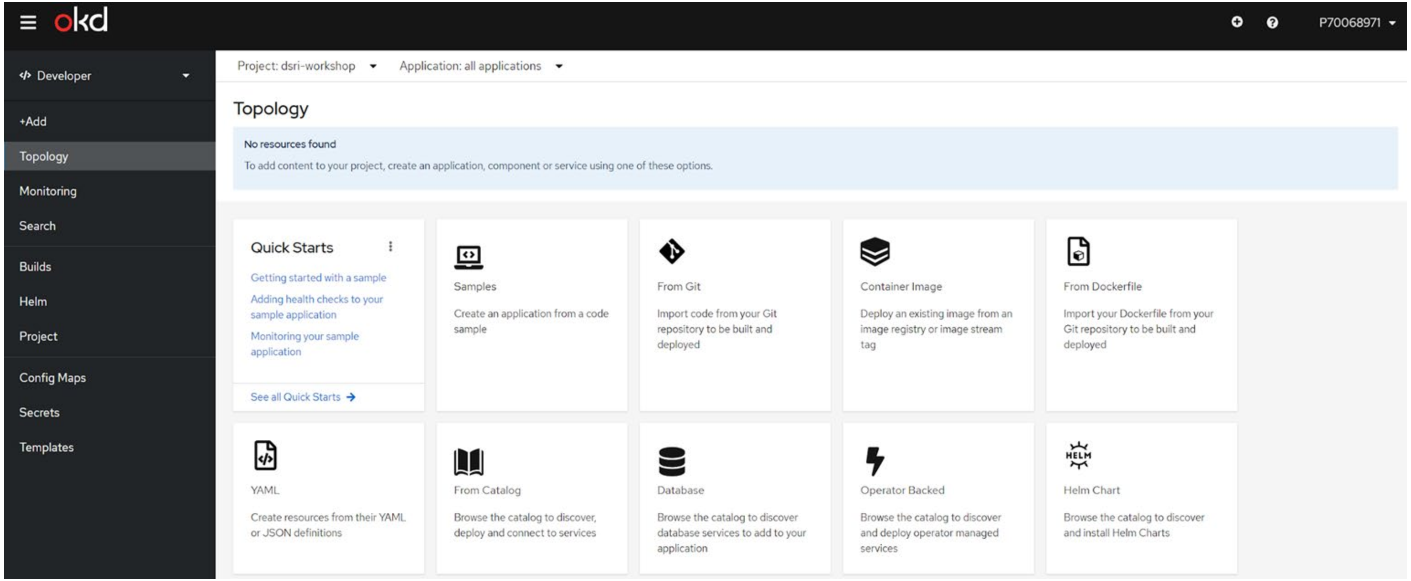
The Developer perspective provides workflows specific to developer use cases, such as the ability to create and deploy applications.

The Administrator perspective provides workflows specific to user admin use cases, such as the ability to create persistent storage, network information etc.



Manage your applications

- Through the OpenShift Web UI (behind the VPN)



Topology

No resources found

To add content to your project, create an application, component or service using one of these options.

Quick Starts

Getting started with a sample

Adding health checks to your sample application

Monitoring your sample application

See all Quick Starts →

Samples

Create an application from a code sample

From Git

Import code from your Git repository to be built and deployed

Container Image

Deploy an existing image from an image registry or image stream tag

From Dockerfile

Import your Dockerfile from your Git repository to be built and deployed

YAML

Create resources from their YAML or JSON definitions

From Catalog

Browse the catalog to discover, deploy and connect to services

Database

Browse the catalog to discover database services to add to your application

Operator Backed

Browse the catalog to discover and deploy operator managed services

Helm Chart

Browse the catalog to discover and install Helm Charts

- Or through the terminal using the **oc** command line interface
  - Which is better for some operations, such as loading large datasets

Easily Deploy Applications using templates

Find a template to deploy your data science application

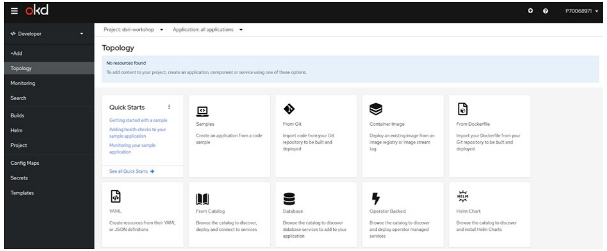
Provide a few parameters to start the application

Access your application through its web UI

Ask for new templates if needed!

Such as name, password, storage location

Using a URL created by the DSRI
Or connect via the terminal



Topology

No resources found

To add content to your project, create an application, component or service using one of these options.

Samples – Create an application from a code sample

From Git – Import code from your Git repository to be built and deployed

Container Image – Deploy an existing image from an image registry or image stream tag

From Dockerfile – Import your Dockerfile from your Git repository to be built and deployed

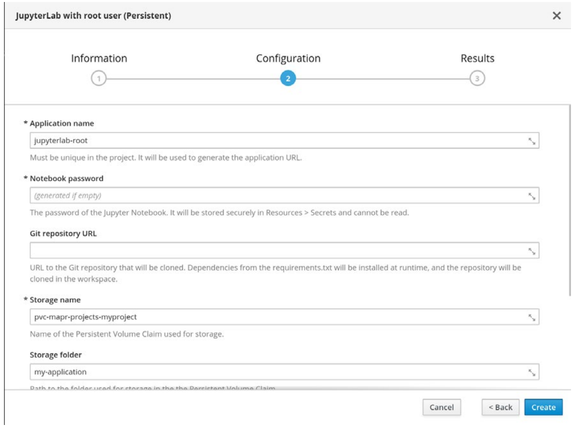
YAML – Create resources from their YAML or JSON definitions

From Catalog – Browse the catalog to discover, deploy and connect to services

Database – Browse the catalog to discover database services to add to your application

Operator Backed – Browse the catalog to discover and deploy operator managed services

Helm Chart – Browse the catalog to discover and install Helm Charts



JupyterLab with root user (Persistent)

* Application name

jupyterlab-root

Must be unique in the project. It will be used to generate the application URL.

* Notebook password

(generated if empty)

The password of the Jupyter Notebook. It will be stored securely in Resources > Secrets and cannot be read.

Git repository URL

URL to the Git repository that will be cloned. Dependencies from the requirements.txt will be installed at runtime, and the repository will be cloned in the workspace.

* Storage name

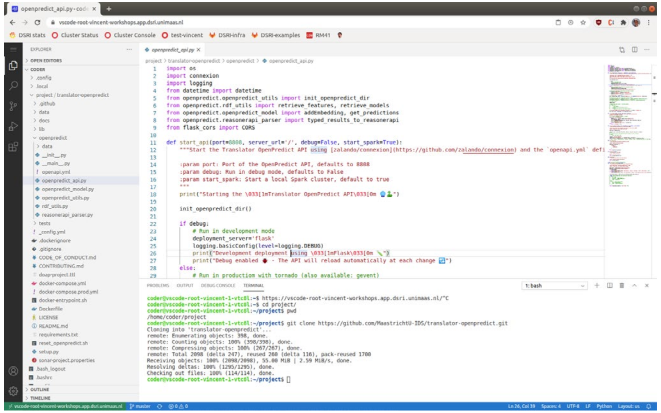
pvc-mapr-projects-myproject

Name of the Persistent Volume Claim used for storage.

Storage folder

my-application

Path to the folder used for storage in the the Persistent Volume Claim



```python
import os
import connexion
import logging
from datetime import datetime
from openpredict.openpredict_utils import init_openpredict_dir
from openpredict.rdf_utils import retrieve_features, retrieve_models
from openpredict.openpredict_model import addEmbedding, get_predictions
from openpredict.reasonerapi_parser import typed_results_to_reasonerapi
from flask_cors import CORS

def start_api(port=8808, server_url='/', debug=False, start_spark=True):
    """Start the Translator OpenPredict API using [zalando/connexion](https://github.com/zalando/connexion) and the `openapi.yml` defi

    :param port: Port of the OpenPredict API, defaults to 8808
    :param debug: Run in debug mode, defaults to False
    :param start_spark: Start a local Spark cluster, default to true
    """
    print("Starting the \033[1mTranslator OpenPredict API\033[0m ")

    init_openpredict_dir()

    if debug:
        # Run in development mode
        deployment_server='flask'
        logging.basicConfig(level=logging.DEBUG)
        print("Development deployment using \033[1mFlask\033[0m ")
        print("Debug enabled  - The API will reload automatically at each change ")
    else:
        # Run in production with tornado (also available: gevent)
```

```
coder@vscode-root-vincent-1-vtc8l:~$ https://vscode-root-vincent-workshops.app.dsri.unimaas.nl/^C
coder@vscode-root-vincent-1-vtc8l:~$ cd project/
coder@vscode-root-vincent-1-vtc8l:~/project$ pwd
/home/coder/project
coder@vscode-root-vincent-1-vtc8l:~/project$ git clone https://github.com/MaastrichtU-IDS/translator-openpredict.git
Cloning into 'translator-openpredict'...
remote: Enumerating objects: 398, done.
remote: Counting objects: 100% (398/398), done.
remote: Compressing objects: 100% (267/267), done.
remote: Total 2098 (delta 247), reused 260 (delta 116), pack-reused 1700
Receiving objects: 100% (2098/2098), 55.00 MiB | 2.59 MiB/s, done.
Resolving deltas: 100% (1295/1295), done.
Checking out files: 100% (114/114), done.
coder@vscode-root-vincent-1-vtc8l:~/project$ 
```

Or define your application deployment!

- Any Docker image can be deployed on the DSRI with a “bit” of configuration
  - You will need to write some YAML files to define how to deploy your app (port, storage, resources limitations, etc)
- The DSRI supports **Helm**, the package manager for Kubernetes
  - To deploy existing deployments
  - Or create new deployments with multiple services easily
- DSRI also supports the use of Operators from the Operator Framework

DSRI Storage Solutions

The DSRI use the RedHat Ceph storage: an open source, massively scalable, simplified storage solution for modern data pipelines

Ephemeral Storage

- Storage is bound to the pod
- Data will be lost when the pod is deleted
- We do not propose this solution anymore, feel free to ask us if you need it

Persistent storage

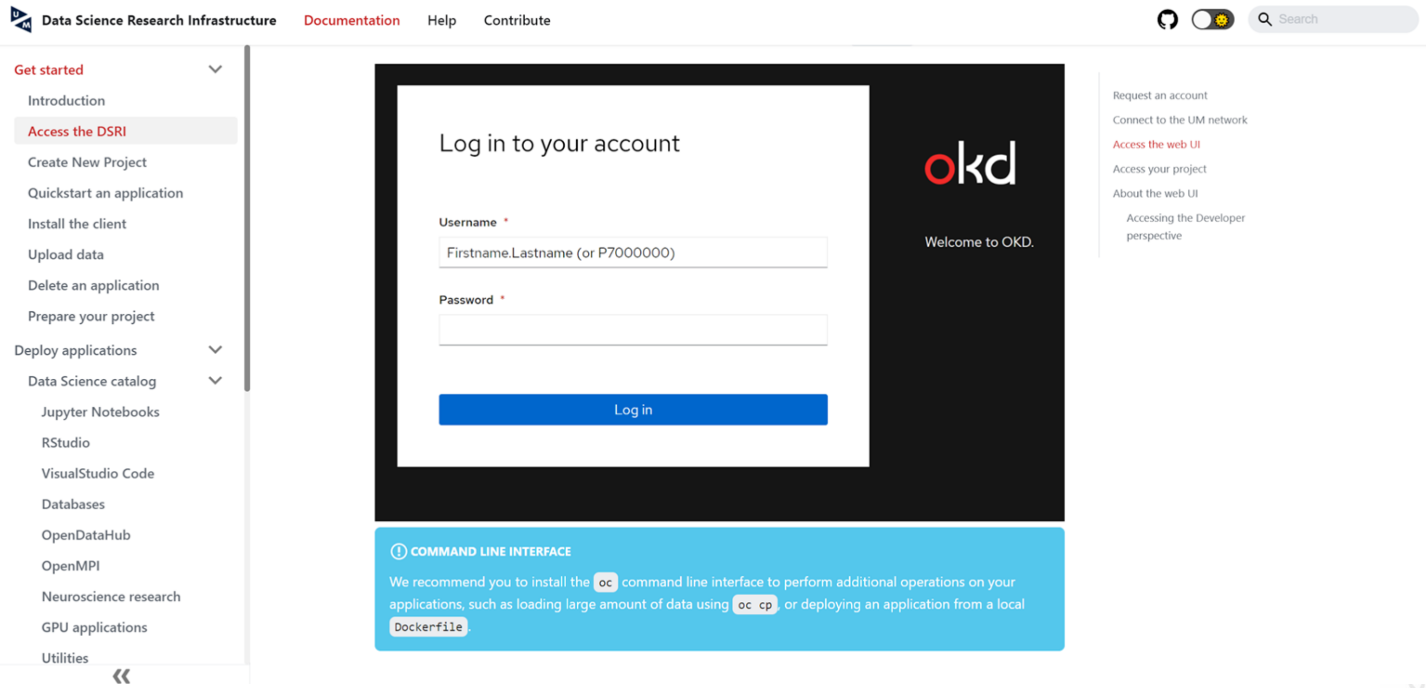
- Data will not be lost when pod get restarted.
- Automatically create when starting most templates
- Can also be created in the OpenShift web UI

Reasons to use the DSRI

- **Run** your work on a **remote server** at UM through popular web UI (Jupyter notebooks, RStudio, VisualStudio Code) instead of your computer
- **Get faster results** with 120 cores to parallelize tasks, or the 500GB memory to run large workloads
- Make use of **best practices** (using git to version and share code) and provide shared environments (containers) to improve project FAIRness
- **Develop and share** these results with your (UM) collaborators

Collaborative documentation website

https://maastrichtu -ids.github.io/dsri-documentation



Data Science Research Infrastructure | Documentation | Help | Contribute | Search

- Get started
  - Introduction
  - Access the DSRI
  - Create New Project
  - Quickstart an application
  - Install the client
  - Upload data
  - Delete an application
  - Prepare your project
- Deploy applications
  - Data Science catalog
    - Jupyter Notebooks
    - RStudio
    - VisualStudio Code
    - Databases
    - OpenDataHub
    - OpenMPI
    - Neuroscience research
    - GPU applications
    - Utilities

Log in to your account

Username *

Firstname.Lastname (or P7000000)

Password *

Log in

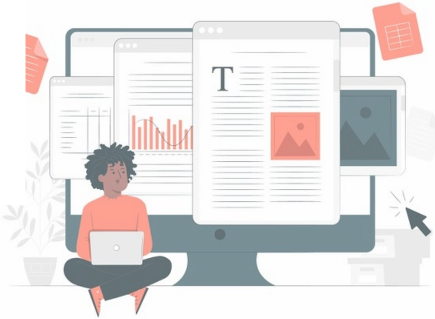
okd

Welcome to OKD.

COMMAND LINE INTERFACE

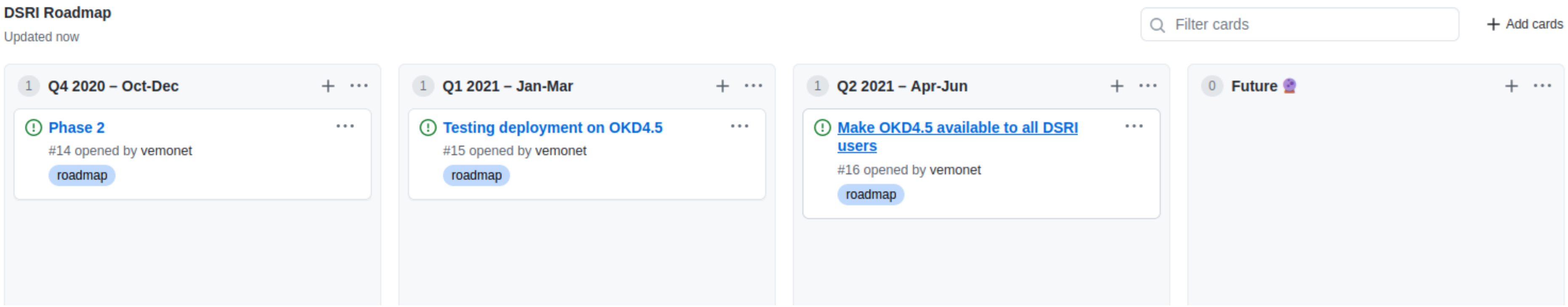
We recommend you to install the `oc` command line interface to perform additional operations on your applications, such as loading large amount of data using `oc cp`, or deploying an application from a local `Dockerfile`.

- Request an account
- Connect to the UM network
- Access the web UI
- Access your project
- About the web UI
  - Accessing the Developer perspective



User Community

- We use slack as instant messaging platform for DSRI communications
  - Get the invitation to Slack after registering to the DSRI
  - **#helpdesk** channel
- Issues tracker on GitHub
  - https://github.com/MaastrichtU-IDS/dsri-documentation/issues
- A public roadmap for the DSRI
  - https://github.com/MaastrichtU-IDS/dsri-documentation/projects/1


DSRI Roadmap

Updated now

1 Q4 2020 – Oct-Dec	1 Q1 2021 – Jan-Mar	1 Q2 2021 – Apr-Jun	0 Future
Phase 2 #14 opened by vemonet roadmap	Testing deployment on OKD4.5 #15 opened by vemonet roadmap	Make OKD4.5 available to all DSRI users #16 opened by vemonet roadmap	

117 registered users and 62+ documented projects

bigcat	Department of Bioinformatics	FHML
fhml	Faculty of Health, Medicine and Life Sciences	FHML
hsr	Department of Health Services Research	FHML
maastro	Maastro Clinic	FHML
NUTRIM	School of Nutrition and Translational Research in Metabolism	FHML
phartox	Department of Pharmacology & Toxicology	FHML
pn	Department of Psychiatry and Neuropsychology	FHML
tgx	Department of Toxicogenomics	FHML
Tech Lab	Law and Tech Lab	FL
dke	Department of Data Science and Knowledge Engineering	FSE
fse	Faculty of Science and Engineering	FSE
gwfp	Gravitational Waves and Fundamental Physics	FSE
ids	Institute of Data Science	FSE
lofse	LO-FSE	FSE
macsbio	MACSBIO System Biology	FSE
msp	Maastricht Science Program	FSE
MSCM		SBE
sbe	School of Business and Economics	SBE
icts	ICT services	UM
um	Maastricht University	UM

Wordcloud from project descriptions

DEMO

Migrate from okd 3.11 to 4.6

If you currently have a project on the previous version of the DSRI (OKD 3.11), you will need to migrate your project to the new version of the DSRI (OKD 4.6)

- Automated persistent storage
- Faster storage more adapted to Data Science workloads
- Better monitoring
- More developer oriented (you don’t need to be a sysadmin to start and manage an app)

It can be done following those instructions:

https://maastrichtu-ids.github.io/dsri-documentation/docs/openshift-migrate-project

Project presentation

Luc De Meyer (BiGCaT)

BiGCaT and I meet DSRI

Our goal

BiGCaT wants to use the DSRI as the standard in-house computing platform for performance BioInformatics related programming and analysis.

BiGCaT and I meet DSRI

Requirements

- To succeed, the users must be able to use the system to build and deploy their application fast and easy
- Deployment must be reliable, repeatable, secure

cont.

Getting started in the v3 era

- Using version 3 of the DSRI at the time I joined BiGCaT
- Documentation scattered
- Creating an app first time is not easy (and I failed)
- Needs a lot of system-level command knowledge
- App-catalog with standard apps is not usable
- Creating from scratch works better if you have disk-space

cont.

Getting RE-started in the v4 era

- User interface is MUCH more usable and user-friendly
- Automatic disk-space allocator works out-of-the-box
- App store now has applications at the click of a button
- Created 2 MySQL apps (and succeeded this time!)
- Need to use OC tunnel to get access from workstation
- A lot of work has gone into improving the docs

cont.

Current

- BiGCaT users are trying it out and first responses are very good
- A lot of BioInformatics related programs will be deployed on the DSRI v4 in the very near future
- BiGCaT has invested in a DSRI node to achieve the goals
- DSRI team has invested a lot of time and effort in improving the user-experience, usability, productivity and documentation

cont.

Thanks !

Thanks to the entire DSRI team

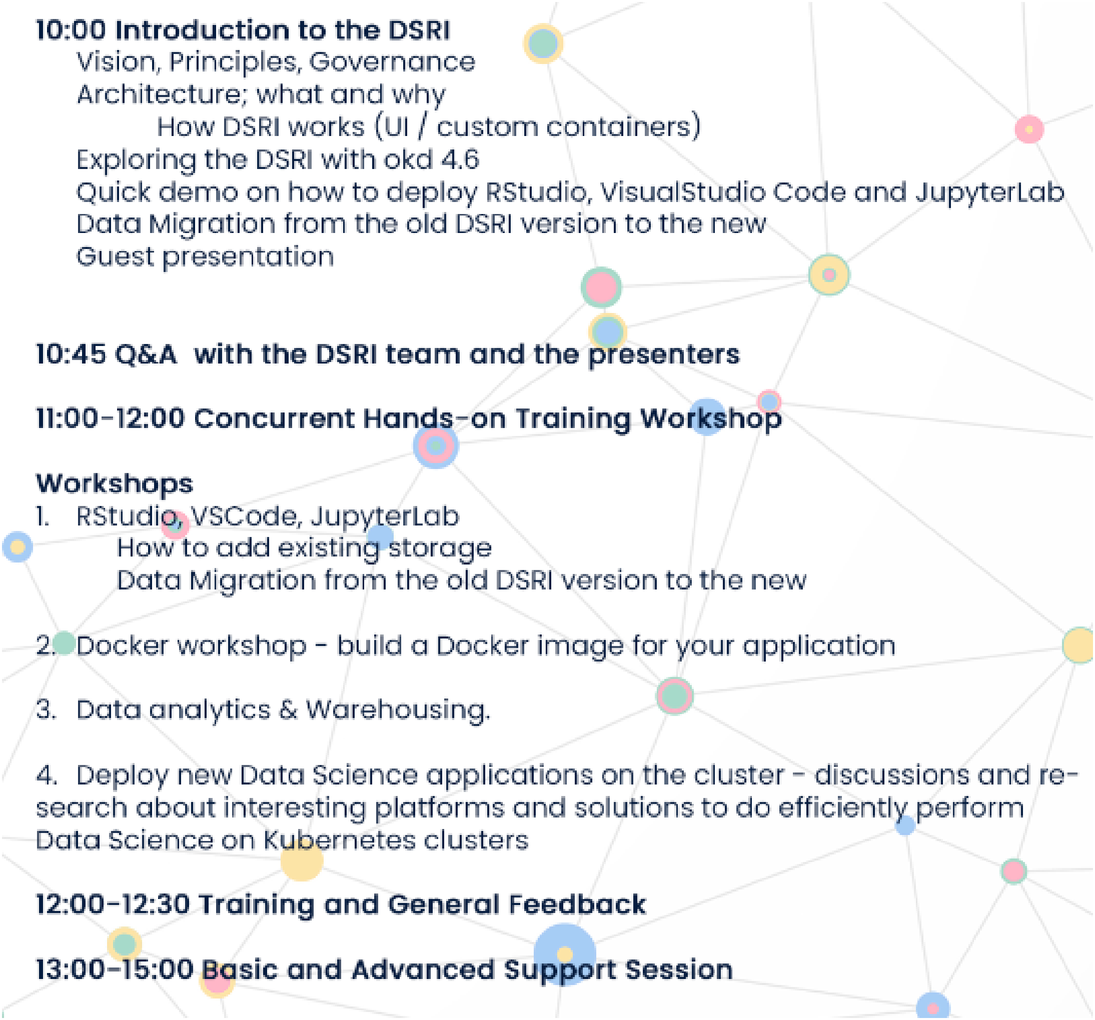
What are our future plans?

- **A vibrant community-supported infrastructure**
  - Weekly technical meetings and monthly planning meetings
  - Advice and feedback from advisory board
  - Regular (2-3x annual) community meetings and training workshops
  - Improved user experience and multi-media documentation
  - Mon-Fri user support
- **Infrastructure improvements**
  - testing OKD 4.5 on a subset of the cluster + CEPH storage (ongoing)
  - resource scheduling and quota management (GPU, CPU)
  - security, data protection, and disaster recovery policies
- **Deploy new Data Science and Machine Learning platforms**
  - Apache Spark, OpenDataHub, KubeFlow, FAIRscape
  - Public-facing applications by the UM research community
- Develop **community-based governance and policies;** invite new investors, secure long term financing, and gain external funding.
  - GDPR certification in progress

Questions?

- ▶ **What DSRI is and how it works?**
- ▶ **On deploying new applications for data science?**
- ▶ **On using the DSRI for complex research?**

Agenda



10:00 Introduction to the DSRI

Vision, Principles, Governance

Architecture; what and why

How DSRI works (UI / custom containers)

Exploring the DSRI with okd 4.6

Quick demo on how to deploy RStudio, VisualStudio Code and JupyterLab

Data Migration from the old DSRI version to the new

Guest presentation

10:45 Q&A with the DSRI team and the presenters

11:00-12:00 Concurrent Hands-on Training Workshop

Workshops

1. RStudio, VSCode, JupyterLab
   How to add existing storage
   Data Migration from the old DSRI version to the new

2. Docker workshop - build a Docker image for your application

3. Data analytics & Warehousing.

4. Deploy new Data Science applications on the cluster - discussions and research about interesting platforms and solutions to do efficiently perform Data Science on Kubernetes clusters

12:00-12:30 Training and General Feedback

13:00-15:00 Basic and Advanced Support Session

Workshops options

1. **Docker** workshop

Build and deploy an application from a custom Dockerfile on the DSRI

1. **Start an application** on the DSRI

You will be guided through deploying a popular Data Science application (RStudio, JupyterLab, VSCode) from a template

1. Use the **DSRI for data analytics**

You will be presented an example of how the DSRI can be used to perform data analytics: deploy JupyterLab, a postgreSQL database, and a MongoDB

1. Explore **potential platforms for Data Science**

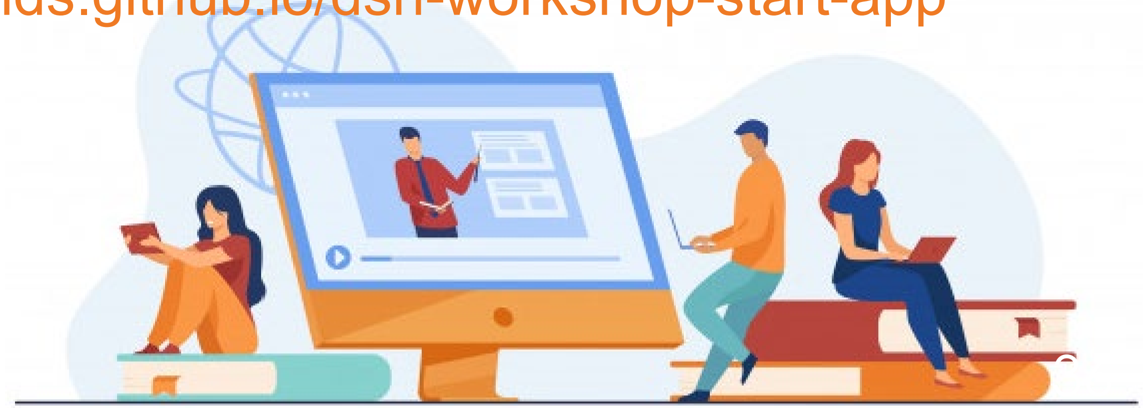
More of a discussion than a guided workshop, we will look into existing solutions to deploy complex but well integrated platforms for Data Science (workflows, visualization, metrics...)

Workshop

Join to your preferred workshop breakout room session

And follow the workshop instructions at

https://maastrichtu-ids.github.io/dsri-workshop-start-app

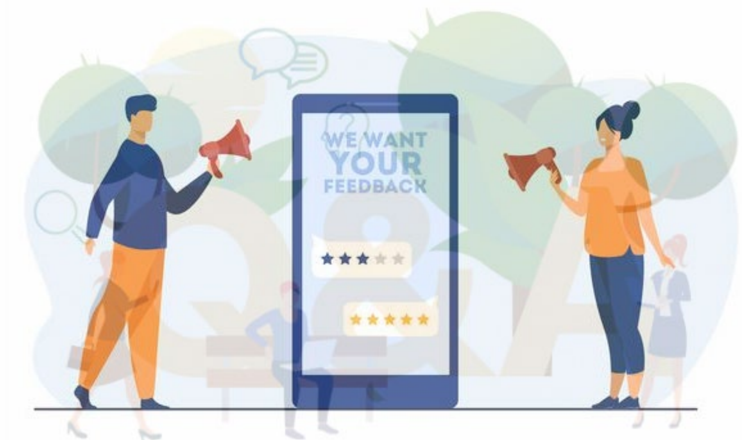


Feedback

Share your thoughts on your experience on the DSRI community day

1. What did you think about DSRI getting started and setup procedure?
2. What other applications would you like to see on the DSRI?
3. What would take it to get you starting to use DSRI (more?)

Feedback form



Questions?

Contact the DSRI Team: https://maastrichtu-ids.github.io/dsri-documentation/help