

# Set up Elyra with Data-Science Pipeline

Note: As Elyra inclusion in the notebook image is still a **work in progress**  
And ODH-Dashboard support is yet to be worked on, a lot of the setup below would be automated, and users wouldn't go over the same steps.  
This is major just to showcase a connection of notebooks with the data-science pipeline.

[OBJ]

Pre-requisite:

- Openshift cluster with openshift-pipeline installed

Steps:

1. Setup data-science-pipeline in the cluster.
  - a. (Only if DSPO is not installed via RHODS already)  
As DS pipeline operator is yet to be part of RHODS.  
One can install the DSPO with the help of kfdef:

```
apiVersion: kfdef.apps.kubeflow.org/v1
kind: KfDef
metadata:
  name: odh-core
spec:
  applications:
  - kustomizeConfig:
    repoRef:
      name: manifests
      path: odh-common
    name: odh-common
  - kustomizeConfig:
    parameters:
    - name: namespace
      value: openshift-operators
    repoRef:
      name: manifests
      path: openshift-pipelines/cluster
    name: openshift-pipelines
  - kustomizeConfig:
    repoRef:
      name: app
```

```

  path: config
  name: data-science-pipelines-operator
  repos:
  - name: manifests
    uri: https://github.com/openshift-io/odh-manifests/tarball/master
  - name: app
    uri: https://github.com/openshift-io/data-science-pipelines-operator/tarball/main
  version: master

```

- b. Once data-science pipeline operator pods are up. An instance of a data-science pipeline can be set up.(all this would be later done through odh-dashboard in future)

For now:

Create a new namespace: `oc new-project dspa`

Apply the CR:

```

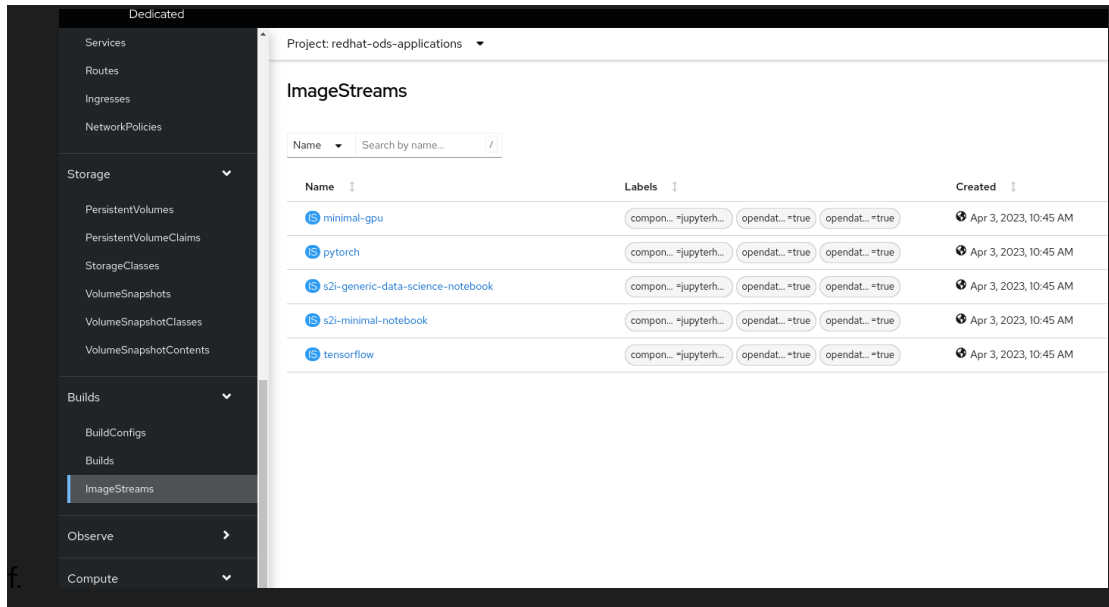
apiVersion: datasciencepipelinesapplications.openshift.io/v1alpha1
kind: DataSciencePipelinesApplication
metadata:
  name: sample
spec:
  objectStorage:
    minio:
      image:
        'quay.io/openshift-io/minio:RELEASE.2019-08-14T20-37-41Z-license-compliance'
  mlpipelineUI:
    image: 'quay.io/openshift-io/odh-ml-pipelines-frontend-container:beta-ui'

```

- c. Once everything is set up, we require following information for elyra pipelines
- i. DS pipeline serve api : check for route on dspa namespace
  - ii. Minio storage details: check for secret named

### **ml-pipeline-minio-artifact**

- d. Now that ds-pipeline is setup, time to setup elyra notebook
- e. On the redhat-ods-application namespace,  
Edit the imagestream `s2i-generic-data-science-notebook`



### Add additional tag:

```

- name: py3.8-v2
  annotations:
    opendatahub.io/notebook-python-dependencies: >-
    [{"name": "Boto3", "version": "1.17"}, {"name": "Kafka-Python", "version": "2.0"}, {"name": "Matplotlib", "version": "3.4"}, {"name": "Numpy", "version": "1.19"}, {"name": "Pandas", "version": "1.2"}, {"name": "Scikit-learn", "version": "0.24"}, {"name": "Scipy", "version": "1.6"}]
    opendatahub.io/notebook-software: [{"name": "Python", "version": "v3.8"}]
    openshift.io/imported-from: quay.io/openshift-workbench-images
  from:
    kind: DockerImage
    name: >-
    quay.io/openshift-workbench-images:jupyter-datascience-ubi8-python-3.8-20230407-e95157b
  generation: 8
  importPolicy: {}
  referencePolicy:
    type: Source
  
```

```

10 opendatahub.io/notebook-image: name: standard-data-science
11 opendatahub.io/notebook-image-order: '20'
12 opendatahub.io/notebook-image-uri: >-
13   https://github.com/red-hat-data-services/notebooks/tree/main/jupyter/datascience/ubi8-python-3.8
14 openshift.io/image.dockerRepositoryCheck: '2023-04-03T16:54:32Z'
15 resourceVersion: '165884'
16 name: s2i-generic-data-science-notebook
17 uid: des6795d-7a18-4888-91d8-e72338ce09fe
18 creationTimestamp: '2023-04-03T14:45:45Z'
19 generation: 6
20 > managedFields: ...
74 namespace: redhat-ods-applications
75 labels:
76   component.opendatahub.io/name: jupyterhub
77   opendatahub.io/component: 'true'
78   opendatahub.io/notebook-image: 'true'
79 spec:
80   lookupPolicy:
81     local: true
82   tags:
83     - name: py3.8-v1
84     annotations:
85       opendatahub.io/notebook-python-dependencies: >-
86         [{"name": "Boto3", "version": "1.17"}, {"name": "Kafka-Python", "version": "2.0"}, {"name": "Matplotlib", "version": "3.4"}, {"name": "Numpy", "v
87       opendatahub.io/notebook-software: [{"name": "Python", "version": "v3.8"}]
88       openshift.io/imported-from: quay.io/modh/odh-generic-data-science-notebook
89     from:
90       kind: DockerImage
91       name: >-
92         quay.io/modh/odh-generic-data-science-notebook@sha256:ebb5613e6b53dc4e8efcfe3878b4cd10ccb77c67d12c00d2b8c9d41aeffd7df5
93     generation: 2
94     importPolicy: {}
95     referencePolicy:
96       type: Source
97     - name: py3.8-v2
98     annotations:
99       opendatahub.io/notebook-python-dependencies: >-
100         [{"name": "Boto3", "version": "1.17"}, {"name": "Kafka-Python", "version": "2.0"}, {"name": "Matplotlib", "version": "3.4"}, {"name": "Numpy", "v
101       opendatahub.io/notebook-software: [{"name": "Python", "version": "v3.8"}]
102       openshift.io/imported-from: quay.io/opendatahub/workbench-images
103     from:
104       kind: DockerImage
105       name: >-
106         quay.io/opendatahub/workbench-images:jupyter-datascience-ubi8-python-3.8-pr-58
107     generation: 6
108     importPolicy: {}
109     referencePolicy:
110       type: Source

```

Once the changes are saved, Launch the notebook server selecting “Standard Data Science py3.8-v2”

- Once the server is started, we need to configure data -science-pipeline with notebook manually (Note: this would be automated by odh-dashboard)

Head to runtimes configure:

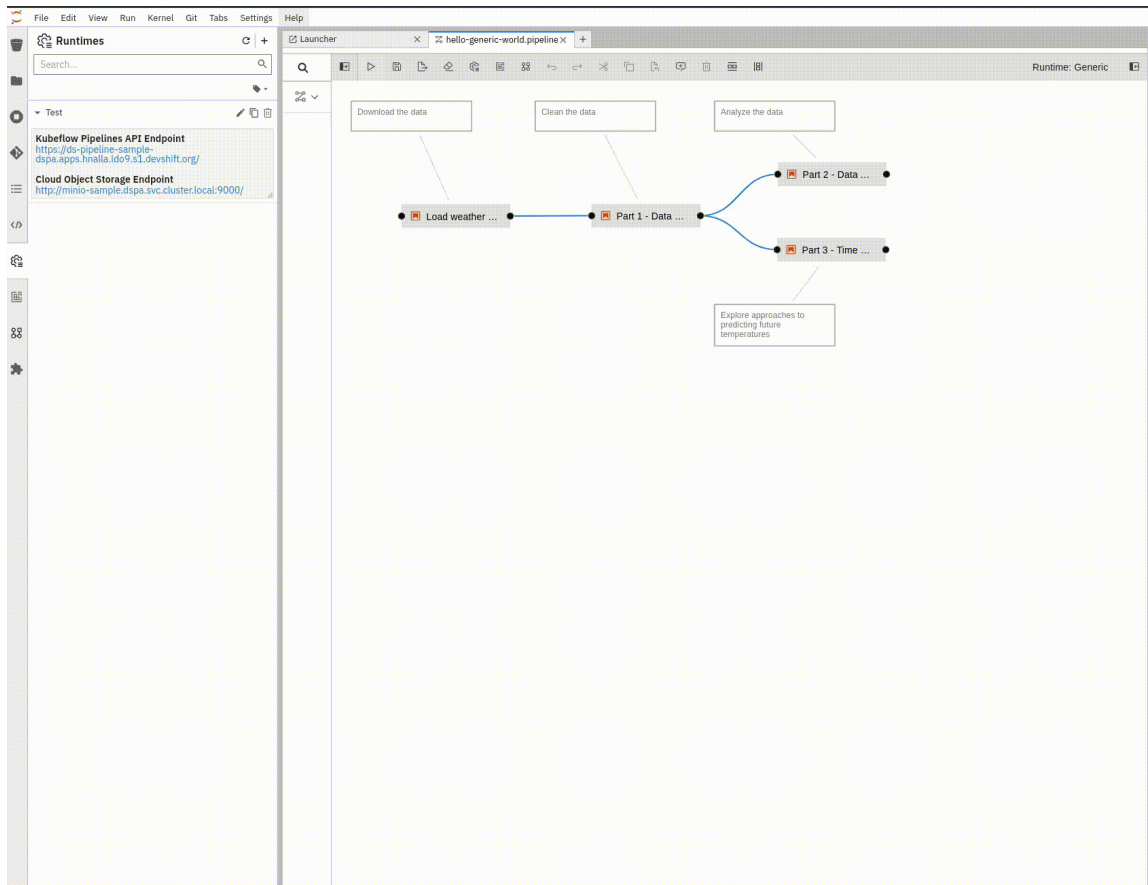
Fill the runtime configuration details:

1. kubeflow pipeline API endpoint: ds pipeline server api route
2. Authentication type: Existing\_bearer\_token
3. Kubeflow Pipelines API Endpoint Password Or Token: user bearer token of openshift

Leave all the other options black

1. For cloud object storage:
  - a. Cloud Object Storage Endpoint: <http://minio-sample.dspa.svc.cluster.local:9000>
  - b. Cloud Object Storage Bucket Name: test
  - c. Cloud Object Storage Credentials Secret: <access-token>

d. Cloud Object Storage Password: <secret-access-token>



Now we are set up.

Elyra has examples to run through the setup.

Follow the tutorial:

<https://github.com/elyra-ai/examples/tree/main/pipelines/run-generic-pipelines-on-kubeflow-pipelines>