

Jupyter Notebook

Jupyter Notebooks

- To effectively build and deploy Machine Learning models, you would need the appropriate environment hence the use of Jupyter notebooks. [Jupyter Notebook](#) is an open-source web application that allows you share documents that contain codes, visualizations, equations, etc.
- In Kubeflow the jupyter notebooks are built on a notebook server. Integrating Jupyter notebooks in Kubeflow as an enterprise environment helps teams share notebooks easily amongst themselves. Users can create notebook containers or pods directly in the cluster, rather than locally. Admins can also provide standard/custom notebook images for their organization, and set up role-based access control (RBAC), secrets and credentials to manage which teams and individuals can access the notebooks.
- Kubeflow allows set up of multiple notebook servers per kubflow deployments, each having a single namespace that corresponds to a team or project name. Servers can also contain multiple notebooks.



Jupyter Notebooks

The screenshot displays the Kubeflow dashboard interface. On the left is a dark blue sidebar with navigation links: Home, Pipelines, **Notebook Servers** (highlighted with a red rectangle), Katib, Artifact Store, Manage Contributors, and GitHub. At the bottom of the sidebar are links for Privacy and Usage Reporting, and the build version 0.7.0. The main content area has a top bar with the Kubeflow logo, the namespace 'kubeflow-sarahmaddox', and tabs for 'Dashboard' (selected) and 'Activity'. The dashboard is divided into three columns. The first column, 'Quick shortcuts', lists actions like 'Upload a pipeline', 'View all pipeline runs', 'Create a new Notebook server', 'View Katib Studies', and 'View Metadata Artifacts'. The second column contains 'Recent Notebooks' (showing 'No Notebooks in namespace kubeflow-sarahmaddox') and 'Recent Pipelines' (listing sample pipelines for Basic - Exit Handler, Basic - Conditional execution, Basic - Parallel execution, Basic - Sequential execution, and ML - XGBoost - Training with ...). The third column, 'Documentation', lists links for 'Getting Started with Kubeflow', 'MiniKF', 'Microk8s for Kubeflow', 'Minikube for Kubeflow', 'Kubeflow on GCP', 'Kubeflow on AWS', and 'Requirements for Kubeflow'.

Kubeflow

kubeflow-sarahmaddox (...)

Home

Pipelines

Notebook Servers

Katib

Artifact Store

Manage Contributors

GitHub

Privacy • Usage Reporting
build version 0.7.0

Dashboard Activity

Quick shortcuts

- ⚡ Upload a pipeline
Pipelines
- ⚡ View all pipeline runs
Pipelines
- ⚡ Create a new Notebook server
Notebook Servers
- ⚡ View Katib Studies
Katib
- ⚡ View Metadata Artifacts
Artifact Store

Recent Notebooks

No Notebooks in namespace kubeflow-sarahmaddox

Recent Pipelines

- [Sample] Basic - Exit Handler
Created 22/12/2019, 06:50:18
- [Sample] Basic - Conditional execution
Created 22/12/2019, 06:50:17
- [Sample] Basic - Parallel execution
Created 22/12/2019, 06:50:16
- [Sample] Basic - Sequential execution
Created 22/12/2019, 06:50:15
- [Sample] ML - XGBoost - Training with ...
Created 22/12/2019, 06:50:14

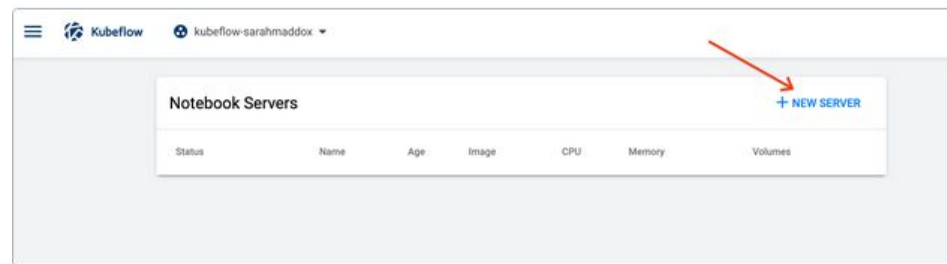
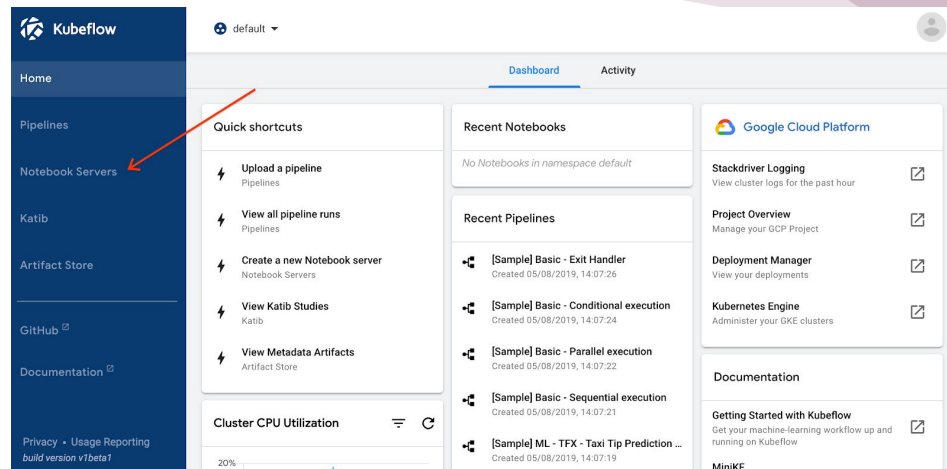
Documentation

- Getting Started with Kubeflow**
Get your machine-learning workflow up and running on Kubeflow
- MiniKF**
A fast and easy way to deploy Kubeflow locally
- Microk8s for Kubeflow**
Quickly get Kubeflow running locally on native hypervisors
- Minikube for Kubeflow**
Quickly get Kubeflow running locally
- Kubeflow on GCP**
Running Kubeflow on Kubernetes Engine and Google Cloud Platform
- Kubeflow on AWS**
Running Kubeflow on Elastic Container Service and Amazon Web Services
- Requirements for Kubeflow**

Set up your Notebook

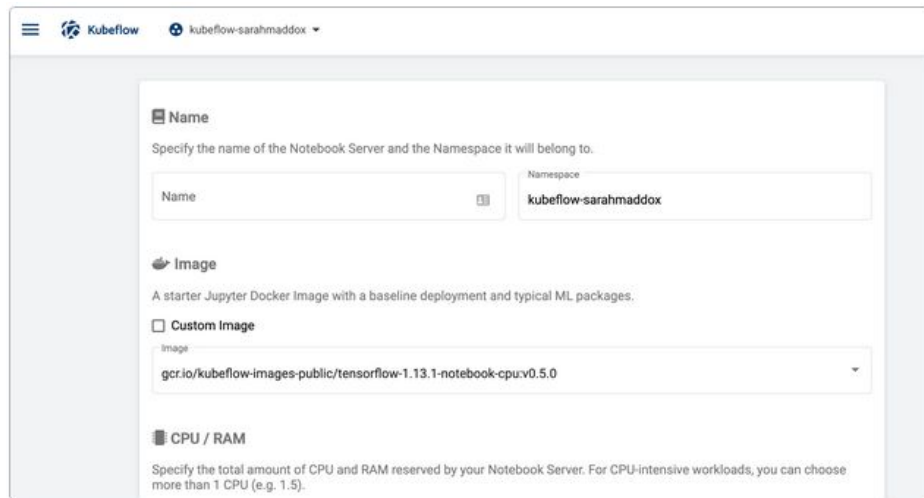
Start by setting up a jupyter notebook through the Notebook Servers tab following the steps below:

1. Click **Notebook Servers** in the left-hand panel of the Kubeflow UI.
2. Click the **namespace** dropdown and choose the one that corresponds to your Kubeflow profile.
3. Click **new server** at the top right corner of the Notebook Servers page to create a notebook server.



Set up your Notebook

4. Enter the details of your new server on the next page:
 - a. Give a **name** of your choice to the notebook server, which must be in lowercase
 - b. The **namespace** is automatically updated by Kubeflow
 - c. Select a Docker image. You can either use a custom image you created or one of the standard images. If you select a custom image you have to specify the docker image in the form: `registry/image:tag`



The screenshot shows the Kubeflow Notebook Server configuration interface. At the top, there is a header with the Kubeflow logo and the namespace 'kubeflow-sarahmaddox'. The main content area is divided into sections for configuring the notebook server. The 'Name' section has a text input field for the server name and a dropdown menu for the namespace, which is currently set to 'kubeflow-sarahmaddox'. The 'Image' section has a description of the starter Jupyter Docker image and a checkbox for 'Custom Image'. Below this, there is a text input field for the image name, which is currently set to 'gcr.io/kubeflow-images-public/tensorflow-1.13.1-notebook-cpu:v0.5.0'. The 'CPU / RAM' section has a description of the resources and a text input field for specifying the total amount of CPU and RAM reserved by the notebook server.

Set up your Notebook – creating custom images

5. While setting up your Jupyter notebook you could either use a standard docker image or a custom image you created. The custom image created must meet the requirements of the Kubeflow notebook controller which manages the life cycle of notebooks.

Follow these steps to configure the launch command in your Docker image:

- Set the working directory :
`--notebook-dir=/home/jovyan`
- Allow Jupyter to listen on all IP addresses: `--ip=0.0.0.0`
- Allow the user to run the notebook as root:
`--allow-root`
- Set the port: `--port=8888`
- Allow passwordless access to your Jupyter notebook servers: `--NotebookApp.token=''`
`--NotebookApp.password=''`
- Allow any origin to access your Jupyter notebook server: `--NotebookApp.allow_origin='*'`
- Set the base URL: `--NotebookApp.base_url=NB_PREFIX`

```
ENV NB_PREFIX /
```

```
CMD ["sh", "-c", "jupyter notebook --  
notebook-dir=/home/jovyan --ip=0.0.0.0 --  
no-browser --allow-root --port=8888 --  
NotebookApp.token='' --  
NotebookApp.password='' --  
NotebookApp.allow_origin='*' --  
NotebookApp.base_url=${NB_PREFIX}"]
```

Set up your Notebook

6. Specify the total amount of **CPU** that your notebook server should reserve.

7. Specify the total amount of memory your notebook server should reserve.

8. Specify a **workspace volume** to hold your personal workspace for this notebook server. Kubeflow provisions a Kubernetes persistent volume (PV) for your workspace volume. The PV ensures that you can retain data even if you destroy your notebook server.

9. Click **LAUNCH** and you should see a new Notebook server entry like below.

Notebook Servers							+ NEW SERVER
Status	Name	Age	Image	CPU	Memory	Volumes	
✓	my-first-notebook	4 mins ago	tensorflow-1.13.1-notebook-cpu:v0.5.0	0.5	1.0Gi		CONNECT 