

Kubeflow Workshop

葉信和 / Hsin-Ho Yeh
Software Engineer / CEO @ 信誠金融科技
hsinho.yeh@footprint-ai.com

Pre-requirement

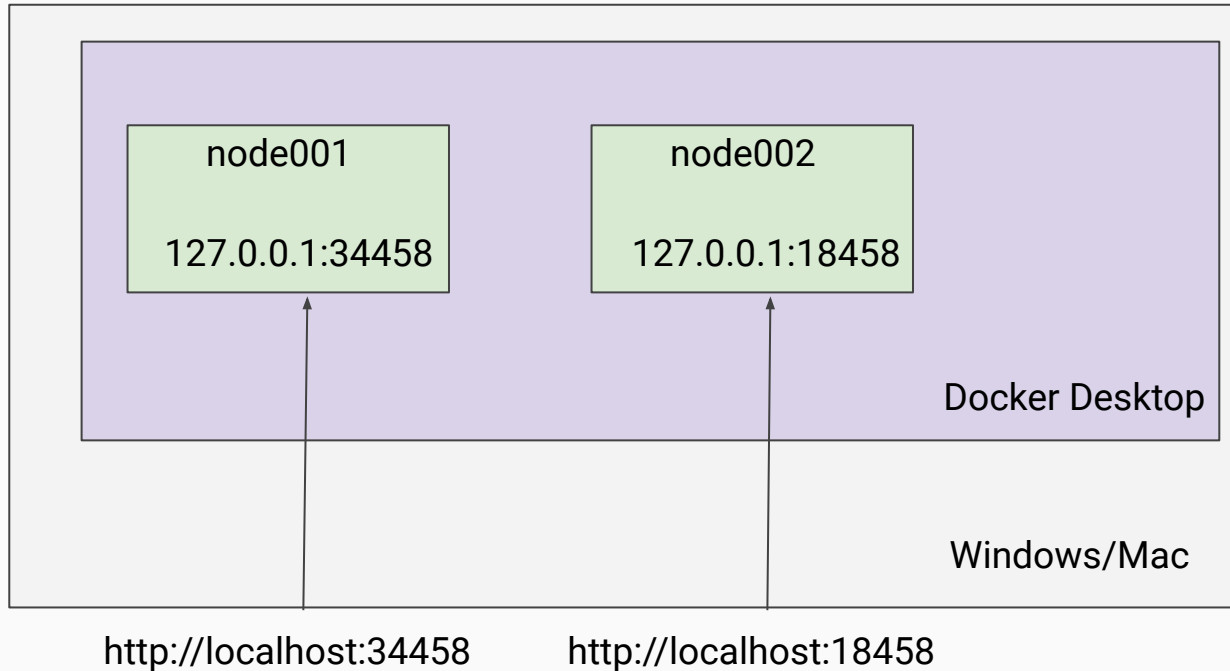
- Be comfortable with UNIX command line
 - Navigating directories with ``cd`` or ``tree``
 - Editing files, like ``vim``, ``nano``
 - Bash scripting, like env or looping
- Be an expert with ``Google``
 - <https://letmegoogletthat.com/?q=you+can+google+it>
- It is totally OK if you don't know what is Container and Kubernetes

荀子《儒效篇》

「不聞不若聞之，聞之不若見之，見之不若知之，知之不若行之；學至于行之而止矣。」

multikf: One-click Installation

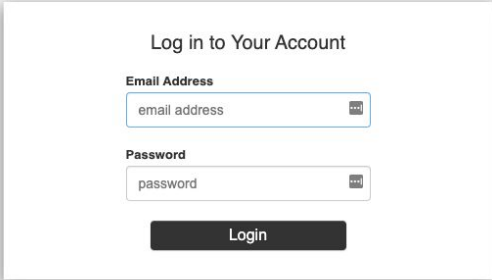
- Multikf: <https://github.com/footprintai/multikf>



Wait! 所以我說那個帳號密碼呢?

Account: [user@example.com](#)

Password: 12341234



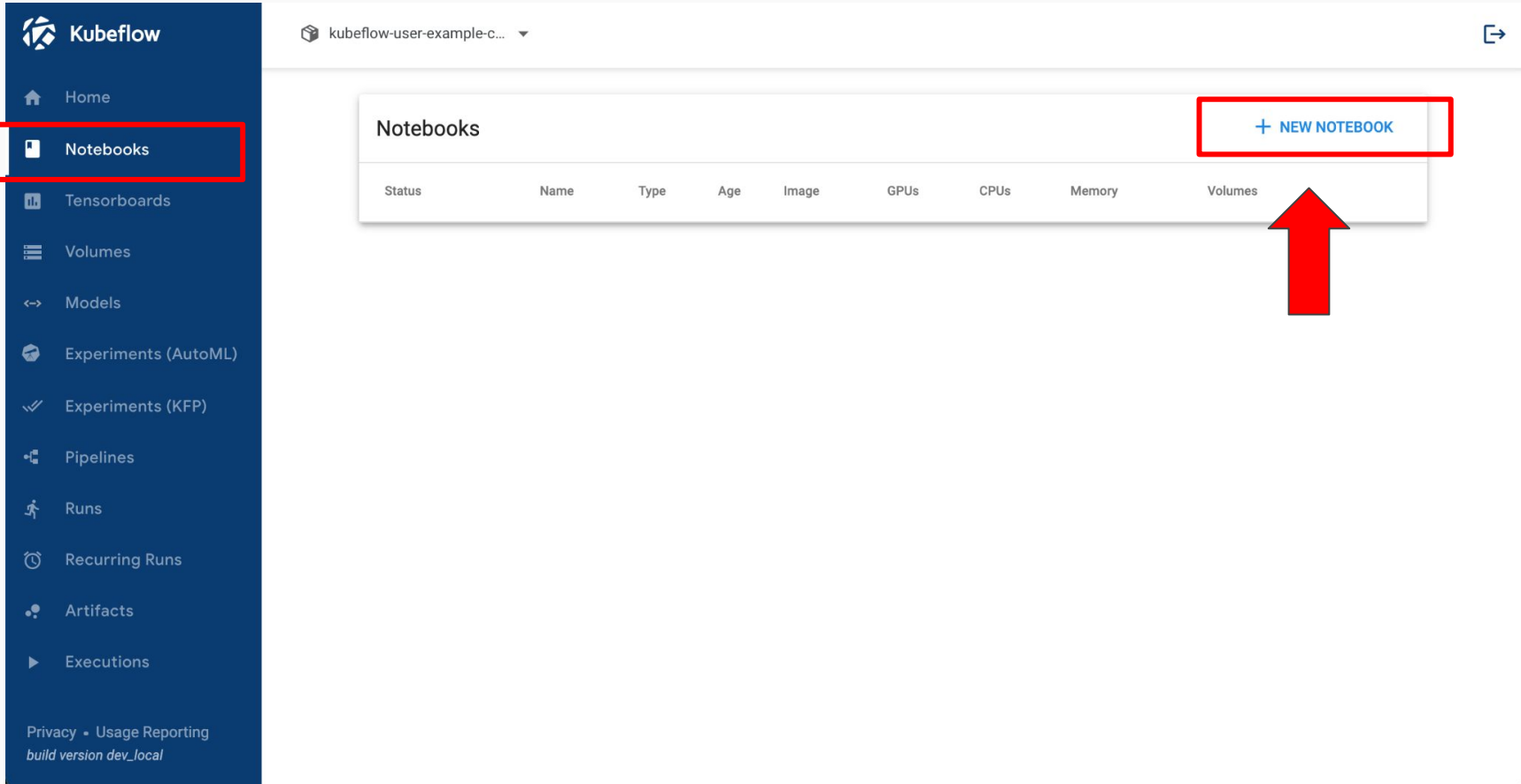
Log in to Your Account

Email Address

Password

Login

Step1: Use Notebook as Online IDE (1/3)



The screenshot displays the Kubeflow dashboard interface. On the left, a dark blue sidebar contains navigation links: Home, Notebooks, Tensorboards, Volumes, Models, Experiments (AutoML), Experiments (KFP), Pipelines, Runs, Recurring Runs, Artifacts, and Executions. The 'Notebooks' link is highlighted with a red rectangle. The main content area shows a 'Notebooks' section with a table header and a '+ NEW NOTEBOOK' button. The button is also highlighted with a red rectangle, and a large red arrow points to it from below. The table header includes columns for Status, Name, Type, Age, Image, GPUs, CPUs, Memory, and Volumes. The top right of the dashboard shows a user profile dropdown and a share icon.

Kubeflow

kubeflow-user-example-c...

Home

Notebooks

Tensorboards

Volumes

Models

Experiments (AutoML)

Experiments (KFP)

Pipelines

Runs

Recurring Runs

Artifacts

Executions


Privacy • Usage Reporting
build version dev_local

Notebooks

+ NEW NOTEBOOK

Status	Name	Type	Age	Image	GPUs	CPUs	Memory	Volumes
--------	------	------	-----	-------	------	------	--------	---------

Step1: Use Notebook as Online IDE (2/3)

 Kubeflow

Home

Notebooks

Tensorboards

Volumes

Models

Experiments (AutoML)

Experiments (KFP)

Pipelines

Runs


Recurring Runs

Artifacts

Executions

Manage Contributors

kubeflow-user-example-c...




Specify the name of the Notebook Server and the Namespace it will belong to.

Name

demo

Namespace

kubeflow-user-example-com

 Image

A starter Jupyter Docker Image with a baseline deployment of TensorFlow and PyTorch ML packages

☐ Custom Image

jupyterlab

1

2

Image

j1r0q0g6/notebooks/notebook-servers/jupyter-tensorflow-full:v1.4

Advanced Options

CPU / RAM

Specify the total amount of CPU and RAM reserved by your Notebook Server. For CPU-intensive workloads, you can choose more than 1 CPU (e.g. 1.5).

Requested CPUs

0.5


Requested memory in Gi

1


Advanced Options

Specify a name and resources like CPU and Memory









Step1: Use Notebook as Online IDE (3/3)


 **Kubeflow**

[Home](#)
[Notebooks](#)
[Tensorboards](#)
[Volumes](#)
[Models](#)
[Experiments \(AutoML\)](#)
[Experiments \(KFP\)](#)
[Pipelines](#)
[Runs](#)
[Recurring Runs](#)
[Artifacts](#)

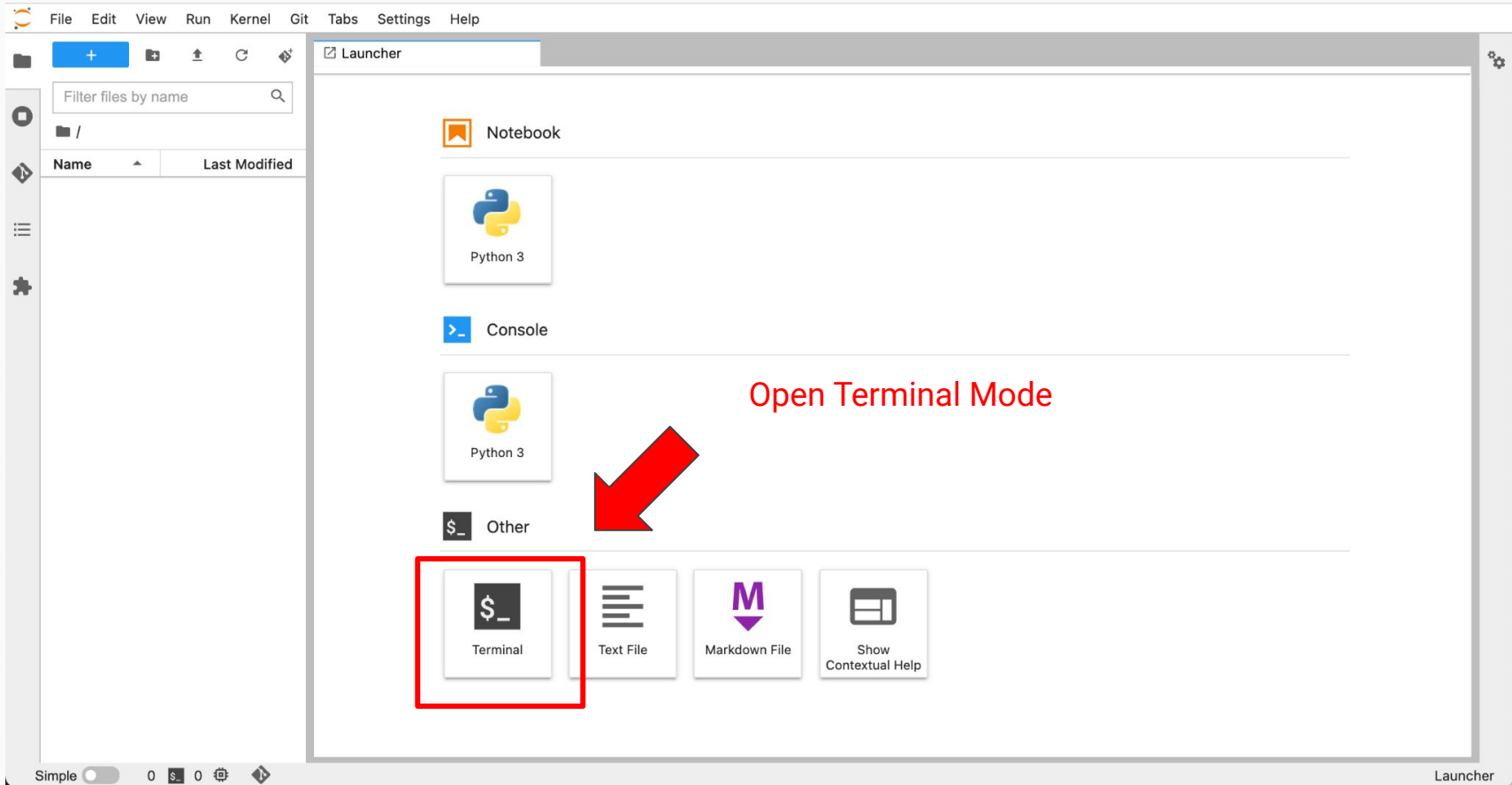
kubeflow-user-example-c... 

Notebooks [+ NEW NOTEBOOK](#)

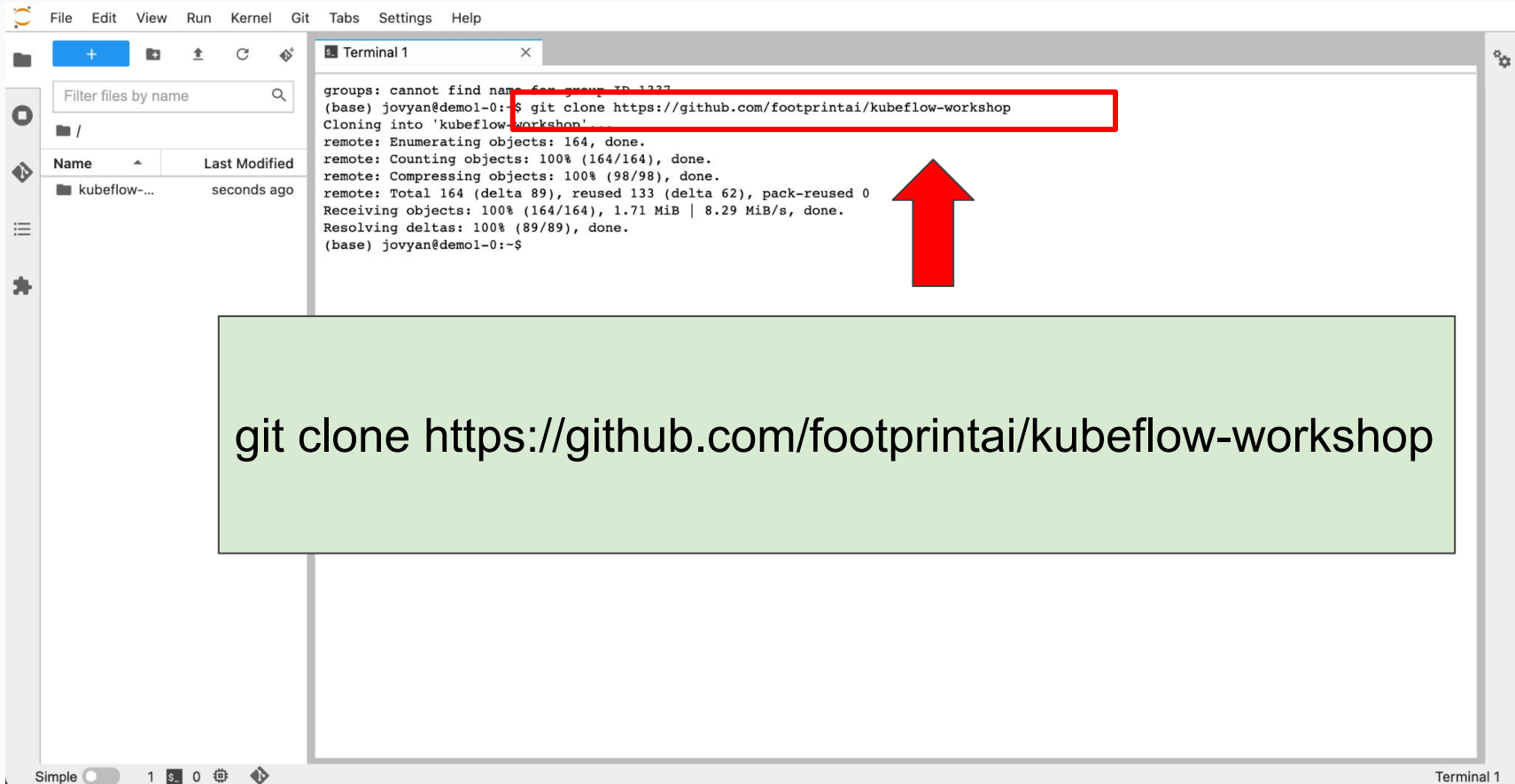
Status	Name	Type	Age	Image	GPUs	CPUs	Memory	Volumes	
✓	demo1		20 hours ago	jupyter-scipy:v1.4	0	0.5	1Gi		CONNECT  
✓	demo2		2 hours ago	jupyter-tensorflow-full:v1.4	0	0.5	1Gi		CONNECT  



Step2: Use terminal to download the materials (1/3)



Step2: Use terminal to download the materials (2/3)



The screenshot shows a JupyterLab environment with a terminal window titled "Terminal 1". The terminal displays the output of a `git clone` command. A red rectangular box highlights the command `(base) jovyan@demo1-0:~$ git clone https://github.com/footprintai/kubeflow-workshop`. A large red arrow points upwards from a green box containing the same command text. The terminal output shows the progress of cloning the repository, including object counting and compression.

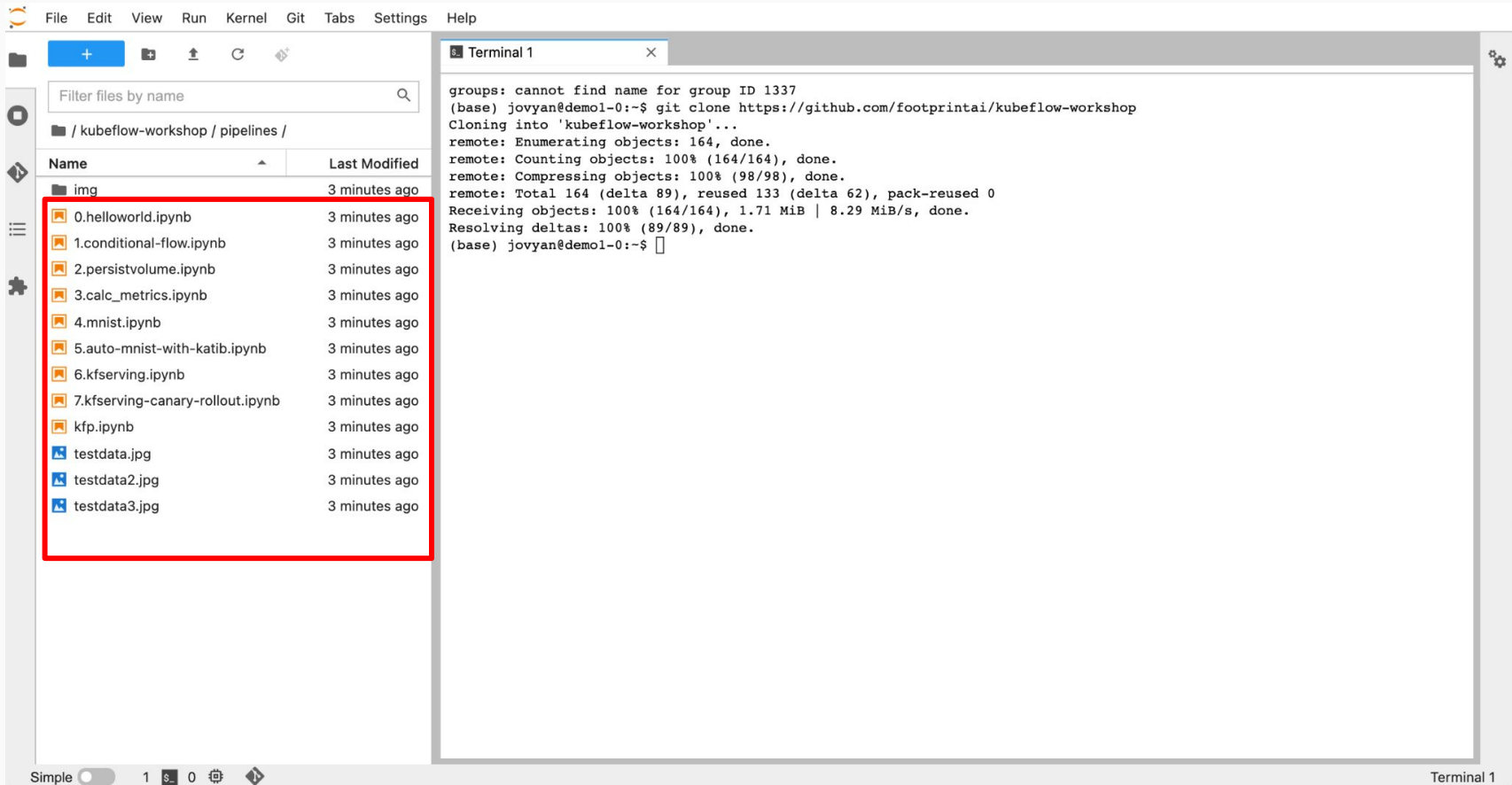
```
groups: cannot find name for group ID 1337
(base) jovyan@demo1-0:~$ git clone https://github.com/footprintai/kubeflow-workshop
Cloning into 'kubeflow-workshop'...
remote: Enumerating objects: 164, done.
remote: Counting objects: 100% (164/164), done.
remote: Compressing objects: 100% (98/98), done.
remote: Total 164 (delta 89), reused 133 (delta 62), pack-reused 0
Receiving objects: 100% (164/164), 1.71 MiB | 8.29 MiB/s, done.
Resolving deltas: 100% (89/89), done.
(base) jovyan@demo1-0:~$
```

git clone https://github.com/footprintai/kubeflow-workshop

Simple 1 0

Terminal 1

Step2: Use terminal to download the materials (3/3)



The screenshot shows a JupyterLab interface. On the left is a file browser with a search bar and a list of files. A red box highlights the first 10 files in the 'img' directory. On the right is a terminal window showing the output of a git clone command.

File Browser:

Name	Last Modified
img	3 minutes ago
0.helloworld.ipynb	3 minutes ago
1.conditional-flow.ipynb	3 minutes ago
2.persistvolume.ipynb	3 minutes ago
3.calc_metrics.ipynb	3 minutes ago
4.mnist.ipynb	3 minutes ago
5.auto-mnist-with-katib.ipynb	3 minutes ago
6.kfserving.ipynb	3 minutes ago
7.kfserving-canary-rollout.ipynb	3 minutes ago
kfp.ipynb	3 minutes ago
testdata.jpg	3 minutes ago
testdata2.jpg	3 minutes ago
testdata3.jpg	3 minutes ago

Terminal:

```
groups: cannot find name for group ID 1337
(base) jovyan@demol-0:~$ git clone https://github.com/footprintai/kubeflow-workshop
Cloning into 'kubeflow-workshop'...
remote: Enumerating objects: 164, done.
remote: Counting objects: 100% (164/164), done.
remote: Compressing objects: 100% (98/98), done.
remote: Total 164 (delta 89), reused 133 (delta 62), pack-reused 0
Receiving objects: 100% (164/164), 1.71 MiB | 8.29 MiB/s, done.
Resolving deltas: 100% (89/89), done.
(base) jovyan@demol-0:~$
```

Kubeflow Terms



```

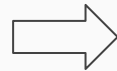
1 | with session$connection[get("c"), "c" %>%
2 |   for(fort["fwd", 8, 7, 9])]
3 |
4 | [for initial / or deployment -> deploy -> deploy]
5 |
6 | # Export kty
7 | Fort Kty Export
8 |
9 | set.seed(1000000000)
10 |
11 | # Create 80 Connections
12 |
13 | # Create 80 Connections
14 |
15 | # Create 80 Connections
16 |
17 | # Create 80 Connections
18 |
19 | # Create 80 Connections
20 |
21 | # Create 80 Connections
22 |
23 | # Create 80 Connections
24 |
25 | # Create 80 Connections
26 |
27 | # Create 80 Connections
28 |
29 | # Create 80 Connections
30 |
31 | # Create 80 Connections
32 |
33 | # Create 80 Connections
34 |
35 | # Create 80 Connections
36 |
37 | # Create 80 Connections
38 |
39 | # Create 80 Connections
40 |
41 | # Create 80 Connections
42 |
43 | # Create 80 Connections
44 |
45 | # Create 80 Connections
46 |
47 | # Create 80 Connections
48 |
49 | # Create 80 Connections
50 |
51 | # Create 80 Connections
52 |
53 | # Create 80 Connections
54 |
55 | # Create 80 Connections
56 |
57 | # Create 80 Connections
58 |
59 | # Create 80 Connections
60 |
61 | # Create 80 Connections
62 |
63 | # Create 80 Connections
64 |
65 | # Create 80 Connections
66 |
67 | # Create 80 Connections
68 |
69 | # Create 80 Connections
70 |
71 | # Create 80 Connections
72 |
73 | # Create 80 Connections
74 |
75 | # Create 80 Connections
76 |
77 | # Create 80 Connections
78 |
79 | # Create 80 Connections
80 |
81 | # Create 80 Connections
82 |
83 | # Create 80 Connections
84 |
85 | # Create 80 Connections
86 |
87 | # Create 80 Connections
88 |
89 | # Create 80 Connections
90 |
91 | # Create 80 Connections
92 |
93 | # Create 80 Connections
94 |
95 | # Create 80 Connections
96 |
97 | # Create 80 Connections
98 |
99 | # Create 80 Connections
100 |

```

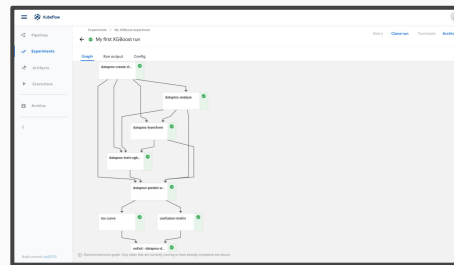


Compiled

Workflow Resource



Create a Pipeline



Experiment1

Experiment2

Runs

Hello World Example

Step4: Compile helloworld.ipynb (1/2)

The screenshot shows the JupyterLab interface with the following components:

- Left Sidebar:** A file browser showing the directory structure. The file `0.helloworld.ipynb` is selected. Other files listed include `1.conditional-flow.ipynb`, `2.persistvolume.ipynb`, `3.calc_metrics.ipynb`, `4.mnist.ipynb`, `5.auto-mnist-with-katib.ipynb`, `6.kfserving.ipynb`, `7.kfserving-canary-rollout.ipynb`, `kfp.ipynb`, `testdata.jpg`, `testdata2.jpg`, and `testdata3.jpg`.
- Terminal 1:** A terminal window titled `0.helloworld.ipynb` showing the execution of the code. The output is `Python 3`.
- Code Editor:** The main editor area contains the following code:

```
[ ]: with open("requirements.txt", "w") as f:
    f.write("kfp==1.8.9\n")

!pip install -r requirements.txt --upgrade --user

[ ]: import kfp
from kfp import dsl

def echo1_op(text1):
    return dsl.ContainerOp(
        name='echo1',
        image='library/bash:4.4.23',
        command=['sh', '-c'],
        arguments=['echo "$0"', text1])

def echo2_op(text2):
    return dsl.ContainerOp(
        name='echo2',
        image='library/bash:4.4.23',
        command=['sh', '-c'],
        arguments=['echo "$0"', text2])

@dsl.pipeline(
    name='Execution order pipeline',
    description='A pipeline to demonstrate execution order management.'
)
def execution_order_pipeline(text1='message 1', text2='message 2'):
    """A two step pipeline with an explicitly defined execution order."""
    step1_task = echo1_op(text1)
    step2_task = echo2_op(text2)
    step2_task.after(step1_task)

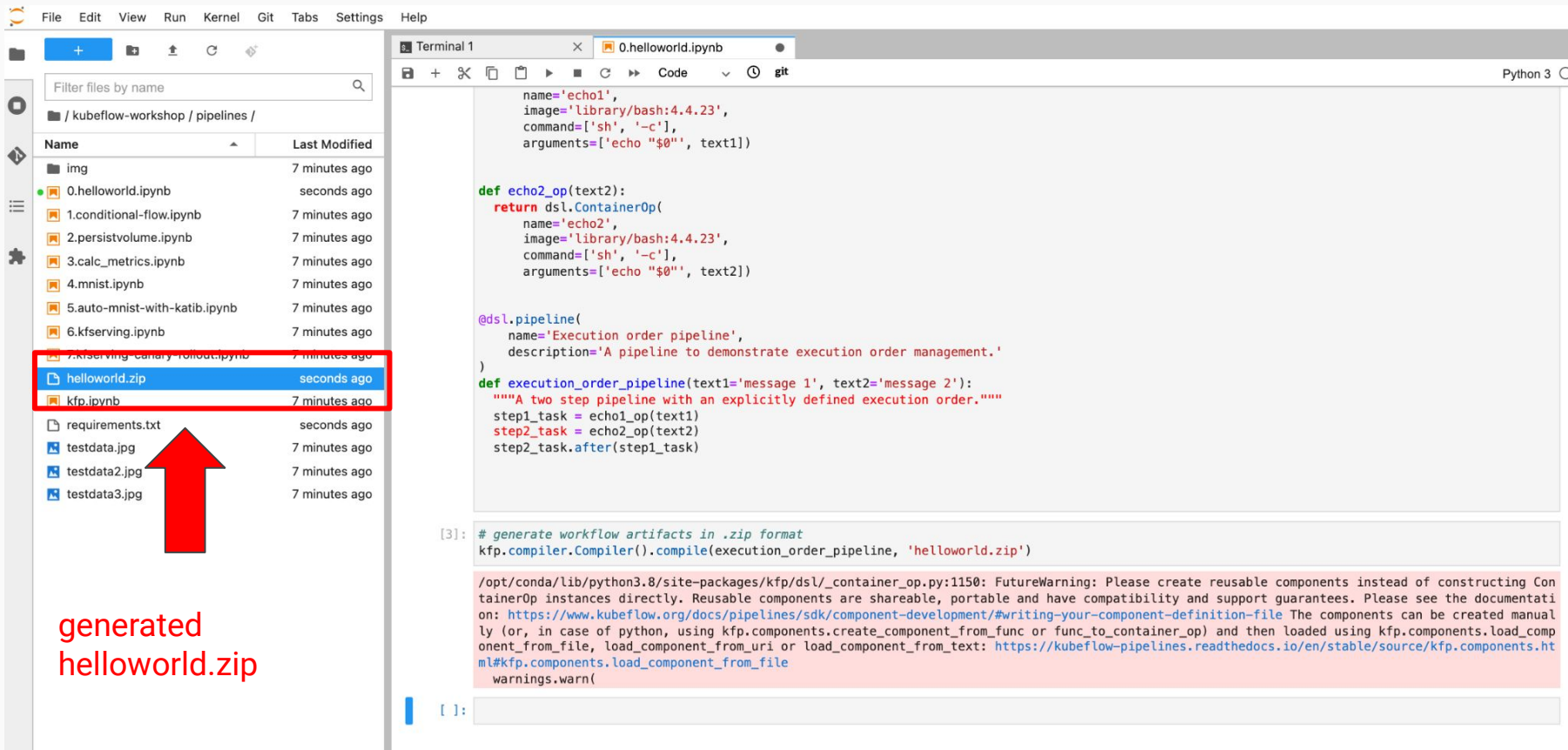
[ ]: # generate workflow artifacts in .zip format
kfp.compiler.Compiler().compile(execution_order_pipeline, 'helloworld.zip')

[ ]:
```

A red box highlights the compilation command at the bottom of the code editor:

```
[ ]: # generate workflow artifacts in .zip format
kfp.compiler.Compiler().compile(execution_order_pipeline, 'helloworld.zip')
```

Step4: Compile helloworld.ipynb (2/2)



File Edit View Run Kernel Git Tabs Settings Help

Filter files by name

/ kubeflow-workshop / pipelines /

Name	Last Modified
img	7 minutes ago
0.helloworld.ipynb	seconds ago
1.conditional-flow.ipynb	7 minutes ago
2.persistvolume.ipynb	7 minutes ago
3.calc_metrics.ipynb	7 minutes ago
4.mnist.ipynb	7 minutes ago
5.auto-mnist-with-katib.ipynb	7 minutes ago
6.kfserving.ipynb	7 minutes ago
7.kfserving-canary-rollout.ipynb	7 minutes ago
helloworld.zip	seconds ago
kfp.ipynb	7 minutes ago
requirements.txt	seconds ago
testdata.jpg	7 minutes ago
testdata2.jpg	7 minutes ago
testdata3.jpg	7 minutes ago

generated helloworld.zip

Terminal 1 0.helloworld.ipynb

```
name='echo1',
image='library/bash:4.4.23',
command=['sh', '-c'],
arguments=['echo "$0"', text1])

def echo2_op(text2):
    return dsl.ContainerOp(
        name='echo2',
        image='library/bash:4.4.23',
        command=['sh', '-c'],
        arguments=['echo "$0"', text2])


@dsl.pipeline(
    name='Execution order pipeline',
    description='A pipeline to demonstrate execution order management.'
)

def execution_order_pipeline(text1='message 1', text2='message 2'):
    """A two step pipeline with an explicitly defined execution order."""
    step1_task = echo1_op(text1)
    step2_task = echo2_op(text2)
    step2_task.after(step1_task)

[3]: # generate workflow artifacts in .zip format
kfp.compiler.Compiler().compile(execution_order_pipeline, 'helloworld.zip')

/opt/conda/lib/python3.8/site-packages/kfp/dsl/_container_op.py:1150: FutureWarning: Please create reusable components instead of constructing ContainerOp instances directly. Reusable components are shareable, portable and have compatibility and support guarantees. Please see the documentation on: https://www.kubeflow.org/docs/pipelines/sdk/component-development/#writing-your-component-definition-file The components can be created manually (or, in case of python, using kfp.components.create_component_from_func or func_to_container_op) and then loaded using kfp.components.load_component_from_file, load_component_from_uri or load_component_from_text: https://kubeflow-pipelines.readthedocs.io/en/stable/source/kfp.components.html#kfp.components.load\_component\_from\_file
warnings.warn()
```


Step4: Create a Pipeline (1/7)

 **Kubeflow**

Home

Notebooks

Tensorboards

Volumes

Models

Experiments (AutoML)

Experiments (KFP)

Pipelines

Runs

Recurring Runs

Artifacts

Executions

Manage Contributors

kubeflow-user-example-c... ▾

↗

+ Upload pipeline

Refresh

Delete

Filter pipelines

▾

<input type="checkbox"/>	Pipeline name	Description	Uploaded
<input type="checkbox"/>	▶ [Tutorial] V2 lightweight Python com...	source code Shows different component input and output options for KFP v2 components.	11/30/2021, 1:02:25 PM
<input type="checkbox"/>	▶ [Tutorial] DSL - Control structures	source code Shows how to use conditional execution and exit handlers. This pipeline will randomly fail to demonstr...	11/30/2021, 1:02:24 PM
<input type="checkbox"/>	▶ [Tutorial] Data passing in python co...	source code Shows how to pass data between python components.	11/30/2021, 1:02:23 PM
<input type="checkbox"/>	▶ [Demo] TFX - Taxi tip prediction mod...	source code GCP Permission requirements . Example pipeline that does classification with model analysis based on...	11/30/2021, 1:02:22 PM
<input type="checkbox"/>	▶ [Demo] XGBoost - Iterative model tra...	source code This sample demonstrates iterative training using a train-eval-check recursive loop. The main pipeline ...	11/30/2021, 1:02:21 PM

Rows per page: 10 ▾ < >

Step4: Create a Pipeline (2/7)

Kubeflow

kubeflow-user-example-c...

Pipeline Versions

← Upload Pipeline or Pipeline Version

☒ Create a new pipeline ☐ Create a new pipeline version under an existing pipeline

Upload pipeline with the specified package.

Pipeline Name
0.helloworld

Pipeline Description
0.helloworld

Choose a pipeline package file from your computer, and give the pipeline a unique name.
You can also drag and drop the file here.

For expected file format, refer to [Compile Pipeline Documentation](#).

☒ Upload a file ☐ Import by url

File
helloworld (19).zip [Choose file](#)

Package Url

Code Source (optional)

[Create](#) [Cancel](#)

1. Pipeline Name

2. specify zip file location

3. Create

Step4: Create a Pipeline (3/7)

The screenshot displays the Kubeflow Pipelines web interface. On the left is a dark blue sidebar with navigation links: Home, Notebooks, Tensorboards, Volumes, Models, Experiments (AutoML), Experiments (KFP), Pipelines, Runs, Recurring Runs, Artifacts, and Executions. At the bottom of the sidebar is a link for 'Manage Contributors'. The main content area has a header with the user 'kubeflow-user-example-c...' and a dropdown arrow. Below this, the title '0.helloworld (0.helloworld)' is shown with a back arrow. To the right of the title are three buttons: '+ Create run', '+ Upload version', and '+ Create experiment'. The '+ Create experiment' button is highlighted with a red rectangular box. A large red arrow points upwards from the text 'Create an experiment' towards this button. Below the title, there are tabs for 'Graph' and 'YAML'. The 'Graph' tab is active, showing a pipeline graph with two steps: 'echo1' and 'echo2', connected by a downward arrow. A 'Simplify Graph' toggle is visible. At the bottom, a 'Summary' panel is partially visible, showing the ID '6f25028f-01e3-4acd-9389-7ec2031fb04b', the version '0.helloworld', and a dropdown arrow next to it.

Kubeflow

kubeflow-user-example-c...

Pipelines

← 0.helloworld (0.helloworld)

+ Create run + Upload version + Create experiment Delete

Graph YAML

Simplify Graph

echo1

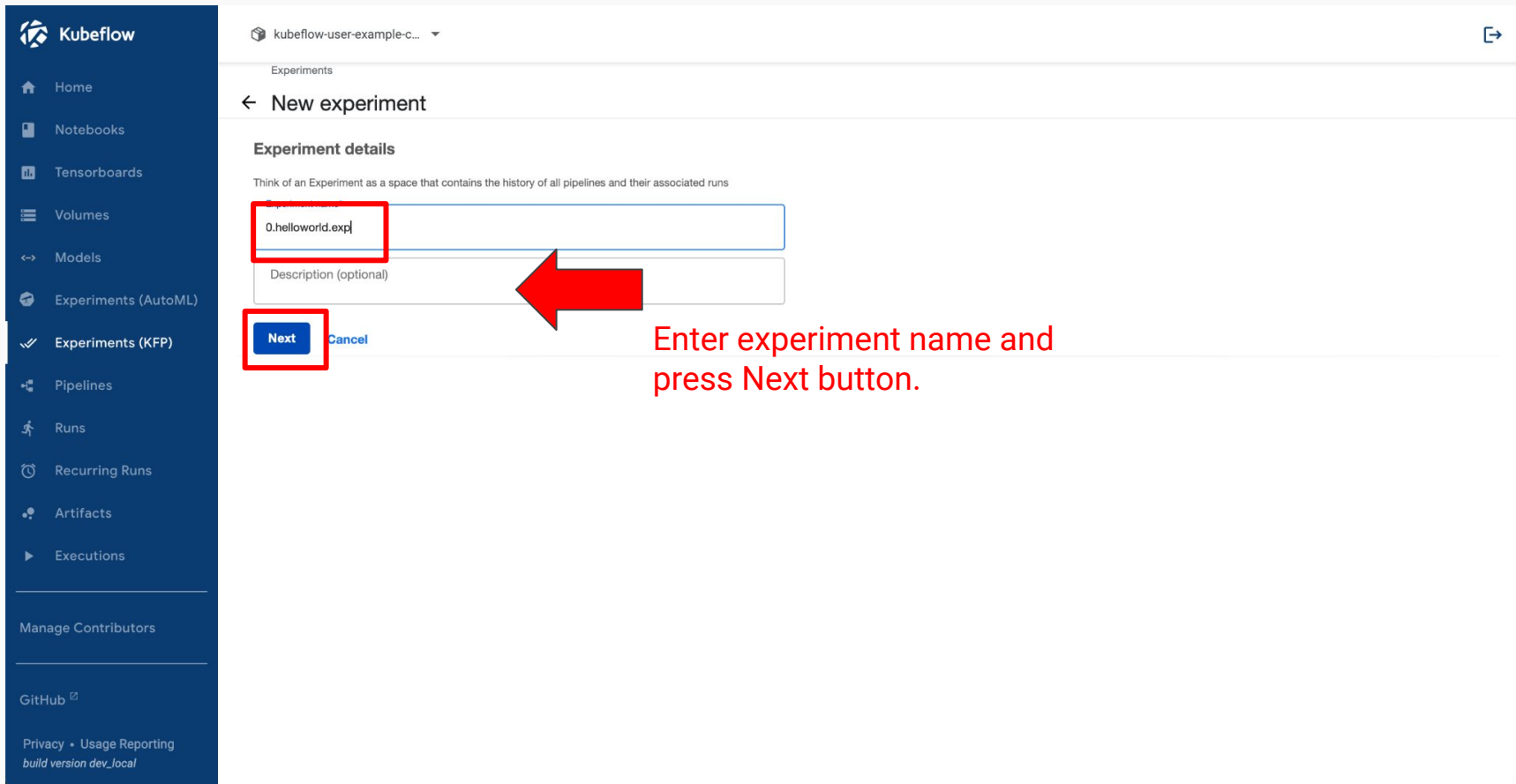
echo2

Summary Hide

ID
6f25028f-01e3-4acd-9389-7ec2031fb04b
Version
0.helloworld
Version source

Create an experiment

Step4: Create a Pipeline (4/7)



Kubeflow

kubeflow-user-example-c...

Experiments

← New experiment

Experiment details

Think of an Experiment as a space that contains the history of all pipelines and their associated runs

0.helloworld.exp

Description (optional)

Next Cancel

Enter experiment name and press Next button.

Home

Notebooks

Tensorboards

Volumes

Models

Experiments (AutoML)

Experiments (KFP)

Pipelines

Runs

Recurring Runs

Artifacts


Executions

Manage Contributors

GitHub

Privacy • Usage Reporting
build version dev_local

Step4: Create a Pipeline (5/7)

 Kubeflow

Home

Notebooks

Tensorboards

Volumes

Models

Experiments (AutoML)

Experiments (KFP)

Pipelines

Runs

Recurring Runs

Artifacts

Executions

Manage Contributors

GitHub

Documentation

Privacy • Usage Reporting
build version dev_local

kubeflow-user-example-c...

Run details

Pipeline*

0.helloworld

Choose

Pipeline Version*

0.helloworld

Choose

Run name*

Run of 0.helloworld [1e261]

Description (optional)

This run will be associated with the following experiment

Experiment*

0.helloworld.exp

Choose

This run will use the following Kubernetes service account.

Service Account (Optional)

Run Type

☒ One-off

☐ Recurring

Run parameters

Specify parameters required by the pipeline

text1

message 1

text2

message 2

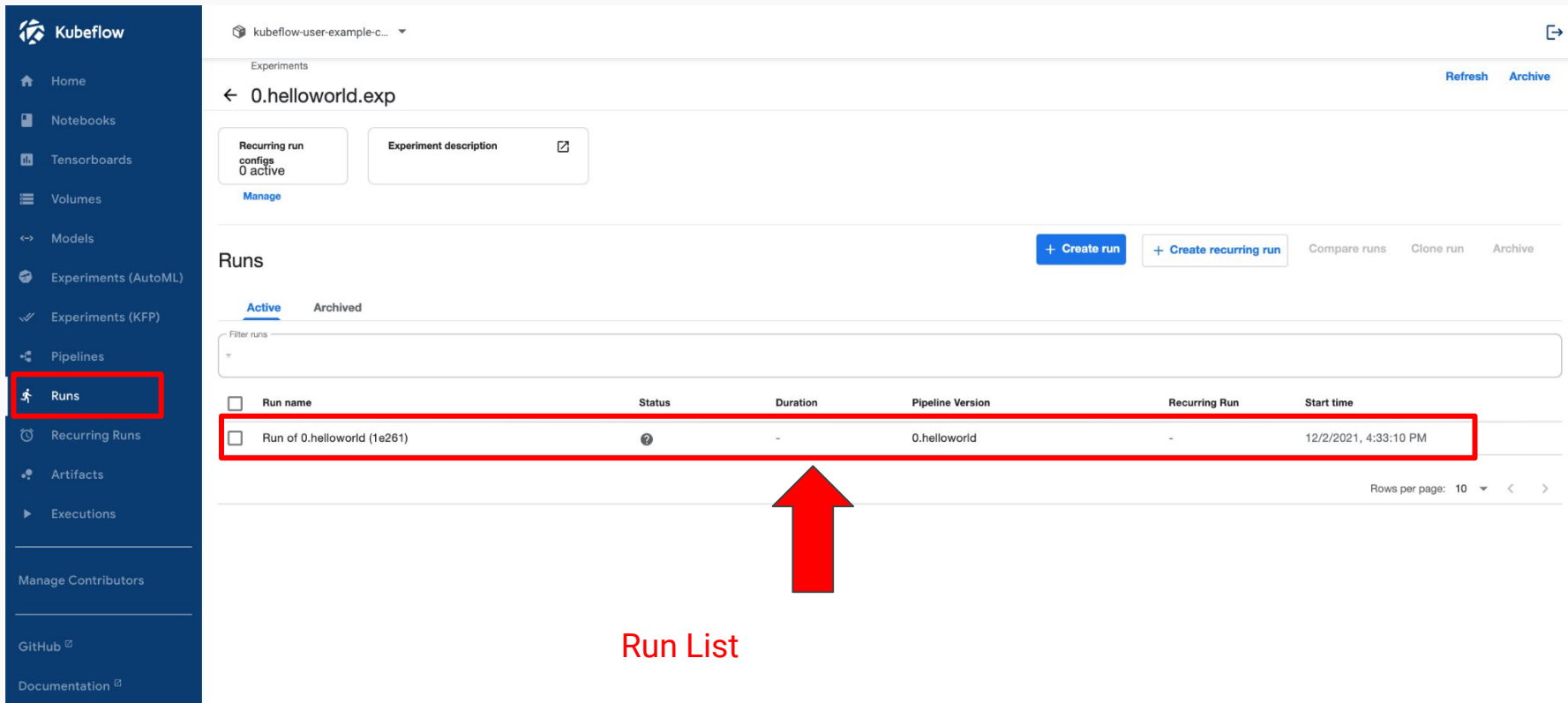
Start

Skip this step

1. Run a pipeline and specify its version

2. Add its experiment name

Step4: Create a Pipeline (6/7)



Kubeflow

Home

Notebooks

Tensorboards

Volumes

Models

Experiments (AutoML)

Experiments (KFP)

Pipelines

Runs

Recurring Runs

Artifacts

Executions

Manage Contributors

GitHub

Documentation

kubeflow-user-example-c...

Experiments

Refresh Archive

← 0.helloworld.exp

Recurring run configs 0 active Manage

Experiment description

Runs

+ Create run + Create recurring run Compare runs Clone run Archive

Active Archived

Filter runs

<input type="checkbox"/>	Run name	Status	Duration	Pipeline Version	Recurring Run	Start time
<input type="checkbox"/>	Run of 0.helloworld (1e261)	?	-	0.helloworld	-	12/2/2021, 4:33:10 PM

Rows per page: 10 < >

Run List

Step4: Create a Pipeline (7/7)

The screenshot displays the Kubeflow dashboard interface. On the left is a dark blue sidebar with navigation links: Home, Notebooks, Tensorboards, Volumes, Models, Experiments (AutoML), Experiments (KFP), Pipelines, Runs, Recurring Runs, Artifacts, Executions, Manage Contributors, and GitHub. The main content area shows the 'Run of 0.helloworld (1e261)' under the 'Experiments > 0.helloworld.exp' path. It includes tabs for 'Graph', 'Run output', and 'Config'. The 'Graph' tab shows a pipeline graph with two steps, 'echo1' and 'echo2', connected by a downward arrow. Both steps have green checkmarks and refresh icons. A red rectangle highlights this graph. To the right, a modal window titled 'execution-order-pipeline-fxxfn-4223123588' is open, showing the 'Input/Output' tab. This tab lists 'Input parameters' (text1: message 1), 'Input artifacts', 'Output parameters', and 'Output artifacts'. The 'Output artifacts' section is highlighted with a red rectangle and shows 'main-logs' with a link to 'minio://mlpipeline/artifacts/execution-order-pipeline-qvlt5/2021/11/30/execution-order-pipeline-qvlt5-137025384/main.log' and the text 'message 1'. A large red arrow points from the 'Output artifacts' section towards the bottom right, where the text '運行結果輸出' (Output of execution results) is written in red.

Kubeflow

kubeflow-user-example-c...

Experiments > 0.helloworld.exp

← Run of 0.helloworld (1e261)

Graph Run output Config

Simplify Graph

echo1

echo2

execution-order-pipeline-fxxfn-4223123588

Input/Output Visualizations Details Volumes Logs Pod Events ML Metadata

Input parameters

text1 message 1

Input artifacts

Output parameters

Output artifacts

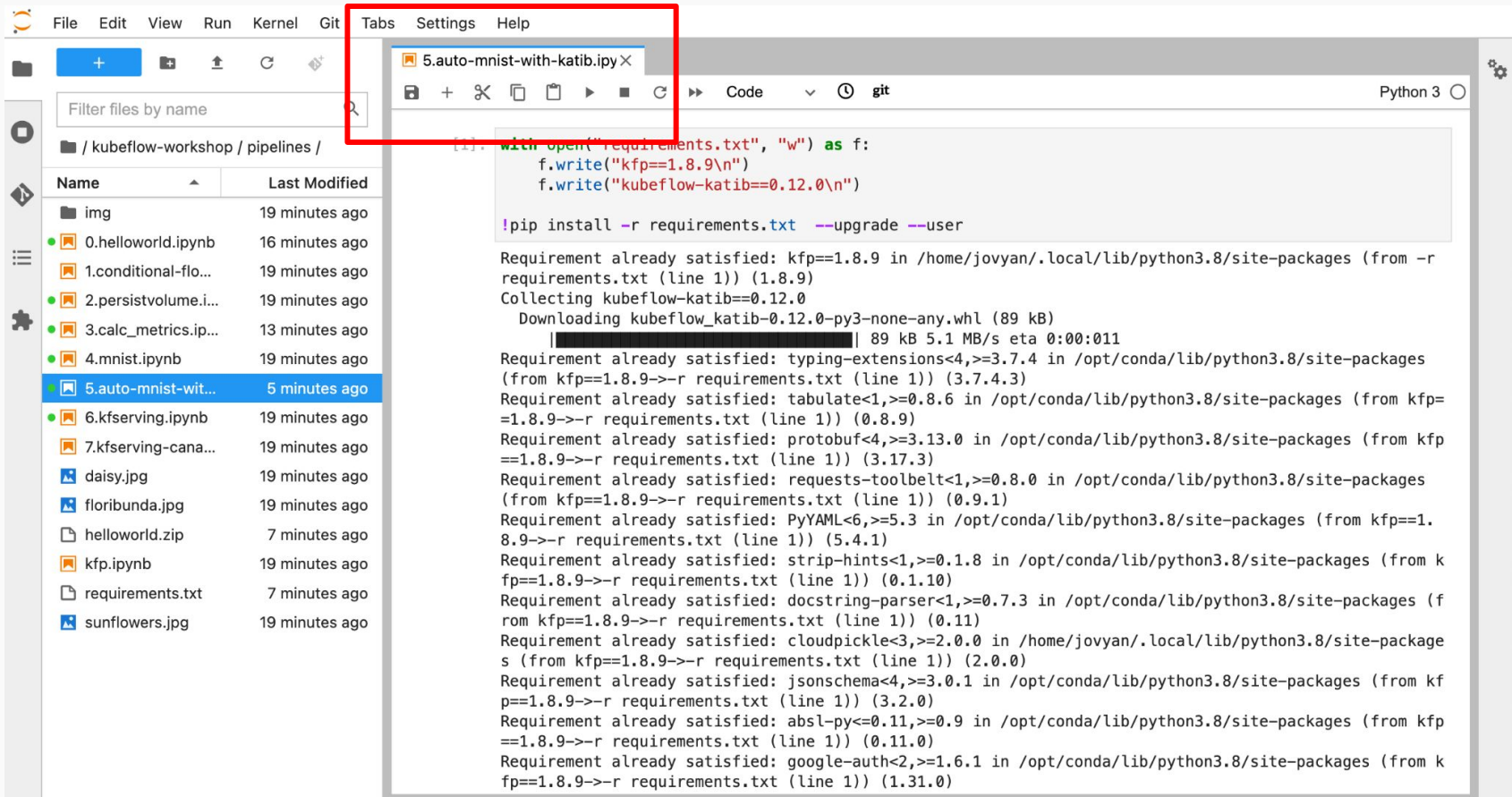
main-logs minio://mlpipeline/artifacts/execution-order-pipeline-qvlt5/2021/11/30/execution-order-pipeline-qvlt5-137025384/main.log View All

message 1

運行結果輸出

Hyperparameter Example

Step5: Hyperparameter tuning with katib (1/4)



The screenshot displays the JupyterLab interface. On the left, the file browser shows a list of files and folders, with '5.auto-mnist-with-katib.ipynb' selected. The right pane shows the code editor for this file. The code defines a requirements.txt file and installs dependencies using pip. The output shows that all requirements are already satisfied.

File Browser:

Name	Last Modified
img	19 minutes ago
0.helloworld.ipynb	16 minutes ago
1.conditional-flo...	19 minutes ago
2.persistvolume.i...	19 minutes ago
3.calc_metrics.ip...	13 minutes ago
4.mnist.ipynb	19 minutes ago
5.auto-mnist-wit...	5 minutes ago
6.kfserving.ipynb	19 minutes ago
7.kfserving-cana...	19 minutes ago
daisy.jpg	19 minutes ago
floribunda.jpg	19 minutes ago
helloworld.zip	7 minutes ago
kfp.ipynb	19 minutes ago
requirements.txt	7 minutes ago
sunflowers.jpg	19 minutes ago


Code Editor:

```
[1]: with open("requirements.txt", "w") as f:
      f.write("kfp==1.8.9\n")
      f.write("kubeflow-katib==0.12.0\n")


!pip install -r requirements.txt --upgrade --user
```

Requirement already satisfied: kfp==1.8.9 in /home/jovyan/.local/lib/python3.8/site-packages (from -r requirements.txt (line 1)) (1.8.9)
Collecting kubeflow-katib==0.12.0
 Downloading kubeflow_katib-0.12.0-py3-none-any.whl (89 kB)
 [REDACTED] 89 kB 5.1 MB/s eta 0:00:011
Requirement already satisfied: typing-extensions<4,>=3.7.4 in /opt/conda/lib/python3.8/site-packages (from kfp==1.8.9->-r requirements.txt (line 1)) (3.7.4.3)
Requirement already satisfied: tabulate<1,>=0.8.6 in /opt/conda/lib/python3.8/site-packages (from kfp==1.8.9->-r requirements.txt (line 1)) (0.8.9)
Requirement already satisfied: protobuf<4,>=3.13.0 in /opt/conda/lib/python3.8/site-packages (from kfp==1.8.9->-r requirements.txt (line 1)) (3.17.3)
Requirement already satisfied: requests-toolbelt<1,>=0.8.0 in /opt/conda/lib/python3.8/site-packages (from kfp==1.8.9->-r requirements.txt (line 1)) (0.9.1)
Requirement already satisfied: PyYAML<6,>=5.3 in /opt/conda/lib/python3.8/site-packages (from kfp==1.8.9->-r requirements.txt (line 1)) (5.4.1)
Requirement already satisfied: strip-hints<1,>=0.1.8 in /opt/conda/lib/python3.8/site-packages (from kfp==1.8.9->-r requirements.txt (line 1)) (0.1.10)
Requirement already satisfied: docstring-parser<1,>=0.7.3 in /opt/conda/lib/python3.8/site-packages (from kfp==1.8.9->-r requirements.txt (line 1)) (0.11)
Requirement already satisfied: cloudpickle<3,>=2.0.0 in /home/jovyan/.local/lib/python3.8/site-packages (from kfp==1.8.9->-r requirements.txt (line 1)) (2.0.0)
Requirement already satisfied: jsonschema<4,>=3.0.1 in /opt/conda/lib/python3.8/site-packages (from kfp==1.8.9->-r requirements.txt (line 1)) (3.2.0)
Requirement already satisfied: absl-py<=0.11,>=0.9 in /opt/conda/lib/python3.8/site-packages (from kfp==1.8.9->-r requirements.txt (line 1)) (0.11.0)
Requirement already satisfied: google-auth<2,>=1.6.1 in /opt/conda/lib/python3.8/site-packages (from kfp==1.8.9->-r requirements.txt (line 1)) (1.31.0)

Step5: Hyperparameter tuning with katib (2/4)

 **Kubeflow**


[Home](#)
[Notebooks](#)
[Tensorboards](#)
[Volumes](#)
[Models](#)
[Experiments \(AutoML\)](#)
[Experiments \(KFP\)](#)
[Pipelines](#)
[Runs](#)
[Recurring Runs](#)
[Artifacts](#)

kubeflow-user-example-c... 

Experiments [Refresh](#) [Archive](#)

[<](#) hello1

Recurring run configs
0 active
[Manage](#)

Experiment description 

[+ Create run](#) [+ Create recurring run](#) [Compare runs](#) [Clone run](#) [Archive](#)

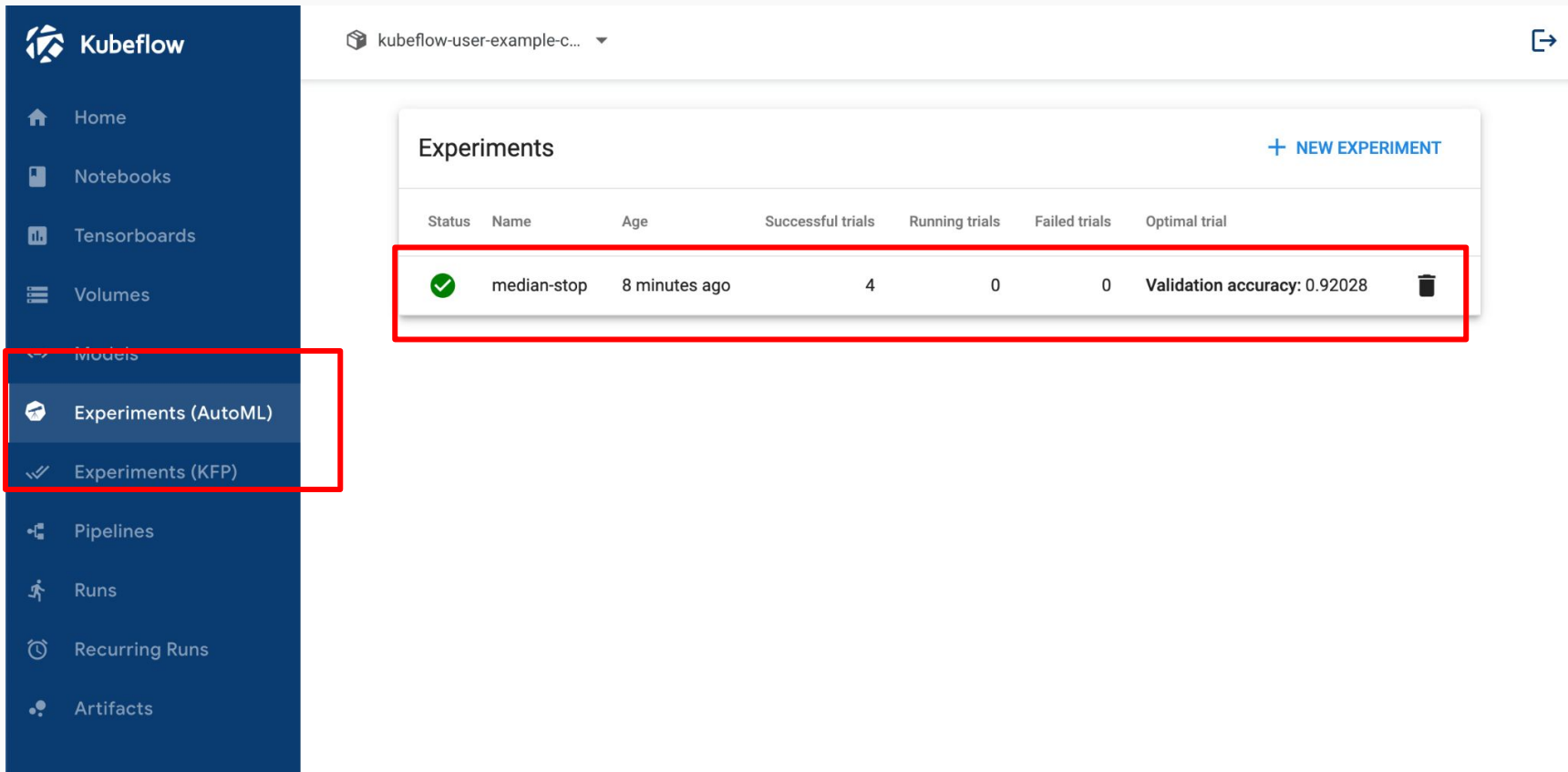
Runs

[Active](#) [Archived](#)

Filter runs

<input type="checkbox"/>	Run name	Status	Duration	Pipeline Version	Recurri...	Start time	quotient	remainder
<input type="checkbox"/>	Run of hello-world_versio...	?	-	hello-world_vers...	-	5/17/2022, 9:42:...		
<input type="checkbox"/>	Run of hello-world_versio...	✓	0:06:19	hello-world_vers...	-	5/17/2022, 9:30:...		
<input type="checkbox"/>	Run of hello-world_versio...	✓	0:01:24	hello-world_vers...	-	5/17/2022, 9:23:...	0.000	6.000

Step5: Hyperparameter tuning with katib (3/4)



The image shows the Kubeflow Experiments (AutoML) interface. On the left is a dark blue sidebar with navigation links: Home, Notebooks, Tensorboards, Volumes, Models, Experiments (AutoML), Experiments (KFP), Pipelines, Runs, Recurring Runs, and Artifacts. The 'Experiments (AutoML)' link is highlighted with a red box. The main content area has a header 'Experiments' with a '+ NEW EXPERIMENT' button. Below is a table with columns: Status, Name, Age, Successful trials, Running trials, Failed trials, and Optimal trial. A single row is visible, representing an experiment named 'median-stop' with a green checkmark status, an age of '8 minutes ago', 4 successful trials, 0 running trials, 0 failed trials, and a validation accuracy of 0.92028. This row is also highlighted with a red box. A trash icon is visible at the end of the row.

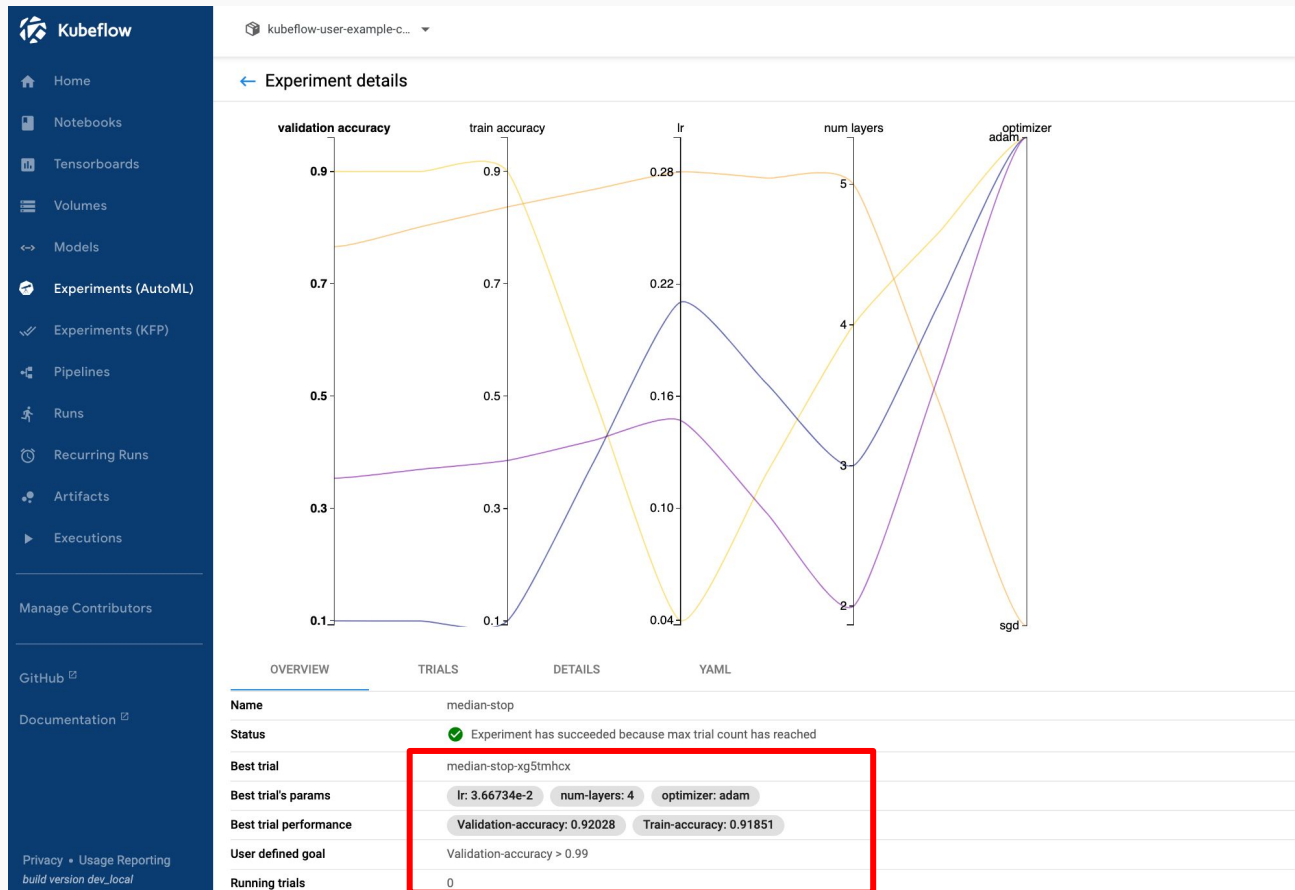
Kubeflow

kubeflow-user-example-c...

Experiments + NEW EXPERIMENT

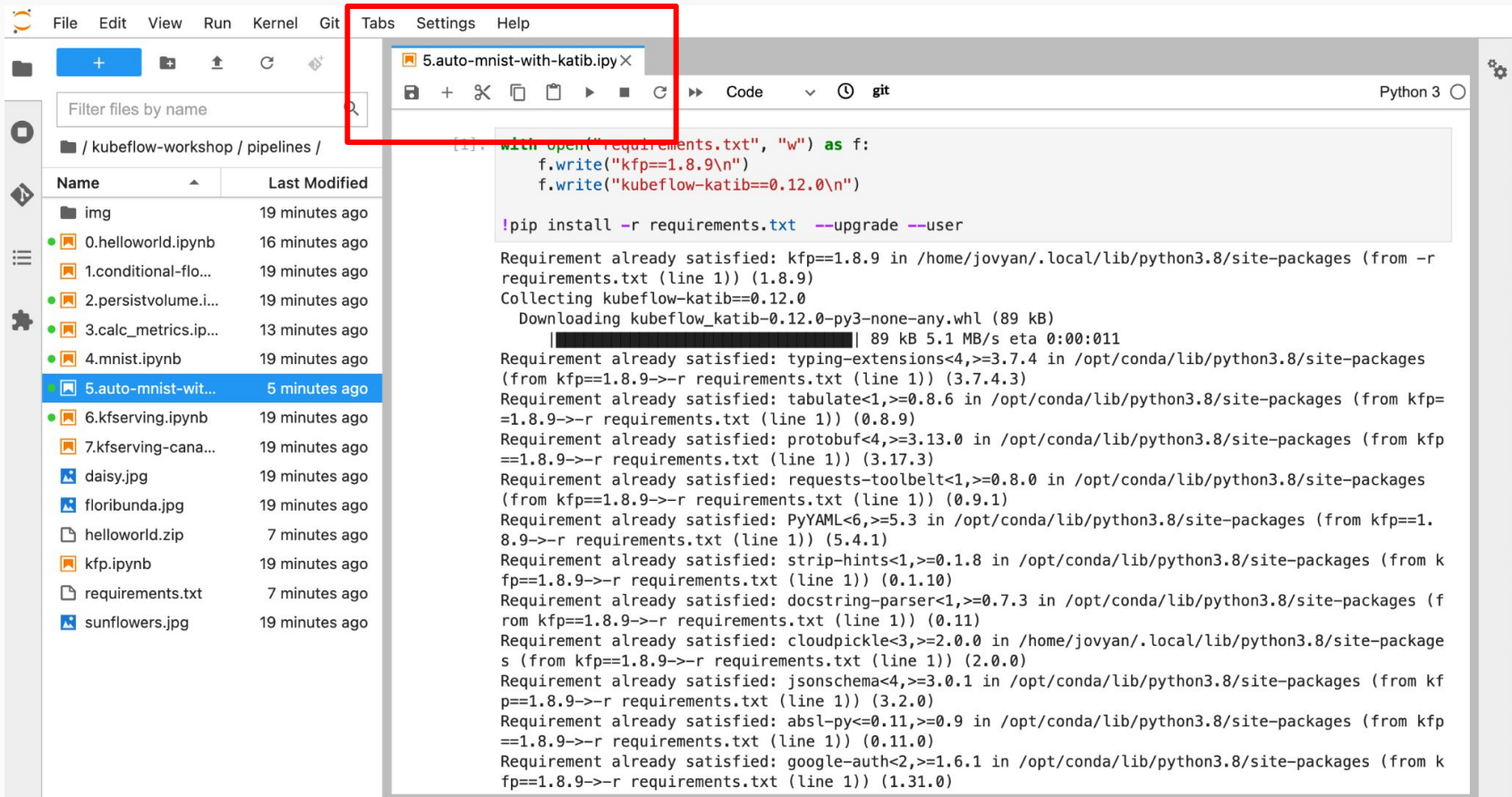
Status	Name	Age	Successful trials	Running trials	Failed trials	Optimal trial
✓	median-stop	8 minutes ago	4	0	0	Validation accuracy: 0.92028

Step5: Hyperparameter tuning with katib (4/4)



Kserve Example

Step6: Serving model with Kserve (1/4)



The screenshot displays a JupyterLab environment. On the left, a file browser shows a directory structure under '/ kubeflow-workshop / pipelines /'. The file '5.auto-mnist-wit...' is selected, showing it was modified 5 minutes ago. The main area is a code editor for the file '5.auto-mnist-with-katib.ipynb', which is currently in 'Code' view. The code editor shows a Python script that writes to 'requirements.txt' and runs 'pip install'.


```
[1]: with open("requirements.txt", "w") as f:
    f.write("kfp==1.8.9\n")
    f.write("kubeflow-katib==0.12.0\n")

!pip install -r requirements.txt --upgrade --user
```


The output of the command shows the following:

```
Requirement already satisfied: kfp==1.8.9 in /home/jovyan/.local/lib/python3.8/site-packages (from -r requirements.txt (line 1)) (1.8.9)
Collecting kubeflow-katib==0.12.0
  Downloading kubeflow_katib-0.12.0-py3-none-any.whl (89 kB)
    [REDACTED] 89 kB 5.1 MB/s eta 0:00:011
Requirement already satisfied: typing-extensions<4,>=3.7.4 in /opt/conda/lib/python3.8/site-packages (from kfp==1.8.9->-r requirements.txt (line 1)) (3.7.4.3)
Requirement already satisfied: tabulate<1,>=0.8.6 in /opt/conda/lib/python3.8/site-packages (from kfp==1.8.9->-r requirements.txt (line 1)) (0.8.9)
Requirement already satisfied: protobuf<4,>=3.13.0 in /opt/conda/lib/python3.8/site-packages (from kfp==1.8.9->-r requirements.txt (line 1)) (3.17.3)
Requirement already satisfied: requests-toolbelt<1,>=0.8.0 in /opt/conda/lib/python3.8/site-packages (from kfp==1.8.9->-r requirements.txt (line 1)) (0.9.1)
Requirement already satisfied: PyYAML<6,>=5.3 in /opt/conda/lib/python3.8/site-packages (from kfp==1.8.9->-r requirements.txt (line 1)) (5.4.1)
Requirement already satisfied: strip-hints<1,>=0.1.8 in /opt/conda/lib/python3.8/site-packages (from kfp==1.8.9->-r requirements.txt (line 1)) (0.1.10)
Requirement already satisfied: docstring-parser<1,>=0.7.3 in /opt/conda/lib/python3.8/site-packages (from kfp==1.8.9->-r requirements.txt (line 1)) (0.11)
Requirement already satisfied: cloudpickle<3,>=2.0.0 in /home/jovyan/.local/lib/python3.8/site-packages (from kfp==1.8.9->-r requirements.txt (line 1)) (2.0.0)
Requirement already satisfied: jsonschema<4,>=3.0.1 in /opt/conda/lib/python3.8/site-packages (from kfp==1.8.9->-r requirements.txt (line 1)) (3.2.0)
Requirement already satisfied: absl-py<=0.11,>=0.9 in /opt/conda/lib/python3.8/site-packages (from kfp==1.8.9->-r requirements.txt (line 1)) (0.11.0)
Requirement already satisfied: google-auth<2,>=1.6.1 in /opt/conda/lib/python3.8/site-packages (from kfp==1.8.9->-r requirements.txt (line 1)) (1.31.0)
```

Step6: Serving model with Kserve (2/4)

 **Kubeflow**


[Home](#)
[Notebooks](#)
[Tensorboards](#)
[Volumes](#)
[Models](#)
[Experiments \(AutoML\)](#)
[Experiments \(KFP\)](#)
[Pipelines](#)
[Runs](#)
[Recurring Runs](#)
[Artifacts](#)

kubeflow-user-example-c... 

Experiments [Refresh](#) [Archive](#)

[<](#) hello1

Recurring run
configs
0 active
[Manage](#)

Experiment description 

[+ Create run](#) [+ Create recurring run](#) [Compare runs](#) [Clone run](#) [Archive](#)

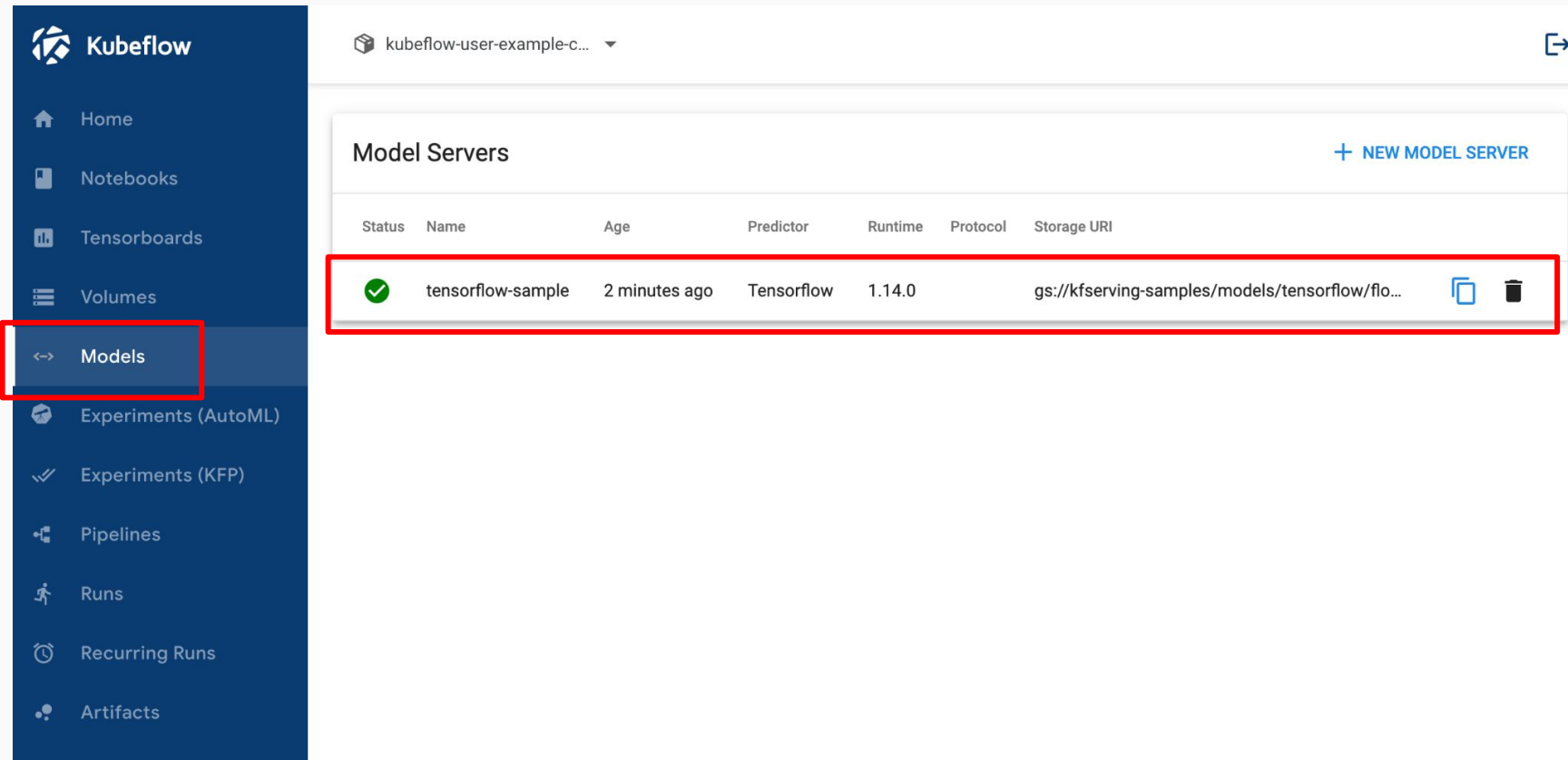
Runs

[Active](#) [Archived](#)

Filter runs

<input type="checkbox"/>	Run name	Status	Duration	Pipeline Version	Recurri...	Start time	quotient	remainder
<input type="checkbox"/>	Run of hello-world_versio...	?	-	hello-world_vers...	-	5/17/2022, 9:42:...		
<input type="checkbox"/>	Run of hello-world_versio...	✓	0:06:19	hello-world_vers...	-	5/17/2022, 9:30:...		
<input type="checkbox"/>	Run of hello-world_versio...	✓	0:01:24	hello-world_vers...	-	5/17/2022, 9:23:...	0.000	6.000

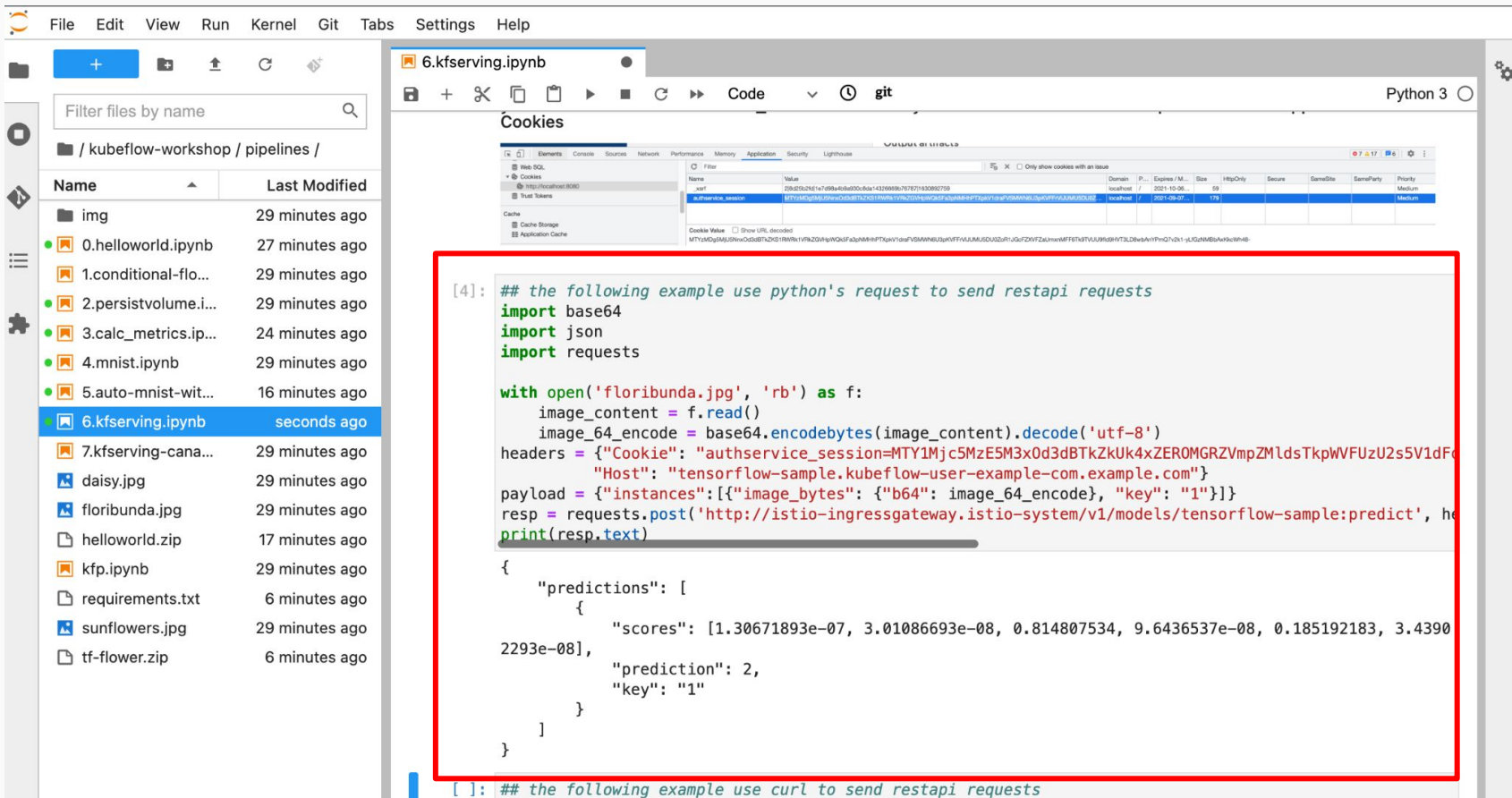
Step6: Serving model with Kserve (3/4)



The screenshot displays the Kubeflow dashboard interface. On the left, a dark blue sidebar contains navigation links: Home, Notebooks, Tensorboards, Volumes, **Models** (highlighted with a red box), Experiments (AutoML), Experiments (KFP), Pipelines, Runs, Recurring Runs, and Artifacts. The main content area is titled 'Model Servers' and includes a '+ NEW MODEL SERVER' button. Below the title is a table with the following columns: Status, Name, Age, Predictor, Runtime, Protocol, and Storage URI. A single row is visible, representing a model server named 'tensorflow-sample'. This row is highlighted with a red box and contains a green checkmark in the Status column, indicating it is ready. The Storage URI is truncated as 'gs://kfserving-samples/models/tensorflow/flo...'. The user 'kubeflow-user-example-c...' is logged in, as shown in the top right corner.

Status	Name	Age	Predictor	Runtime	Protocol	Storage URI
✓	tensorflow-sample	2 minutes ago	Tensorflow	1.14.0		gs://kfserving-samples/models/tensorflow/flo...

Step6: Serving model with Kserve (4/4)



File Edit View Run Kernel Git Tabs Settings Help

Filter files by name

/ kubeflow-workshop / pipelines /

Name	Last Modified
img	29 minutes ago
0.helloworld.ipynb	27 minutes ago
1.conditional-flo...	29 minutes ago
2.persistvolume.i...	29 minutes ago
3.calc_metrics.ip...	24 minutes ago
4.mnist.ipynb	29 minutes ago
5.auto-mnist-wit...	16 minutes ago
6.kfserving.ipynb	seconds ago
7.kfserving-cana...	29 minutes ago
daisy.jpg	29 minutes ago
floribunda.jpg	29 minutes ago
helloworld.zip	17 minutes ago
kfp.ipynb	29 minutes ago
requirements.txt	6 minutes ago
sunflowers.jpg	29 minutes ago
tf-flower.zip	6 minutes ago

6.kfserving.ipynb

Python 3

Cookies

Name	Value	Domain	Expires / M...	Size	HttpOnly	Secure	SameSite	SameParty	Priority
authservice_session	MTY1Mjc5MzE5M3x0d3dBTkZkUk4xZEROMGRZVmpZlDsTkpwVWFuZU2s5V1dF...	localhost	2021-10-26	99			lax		Medium

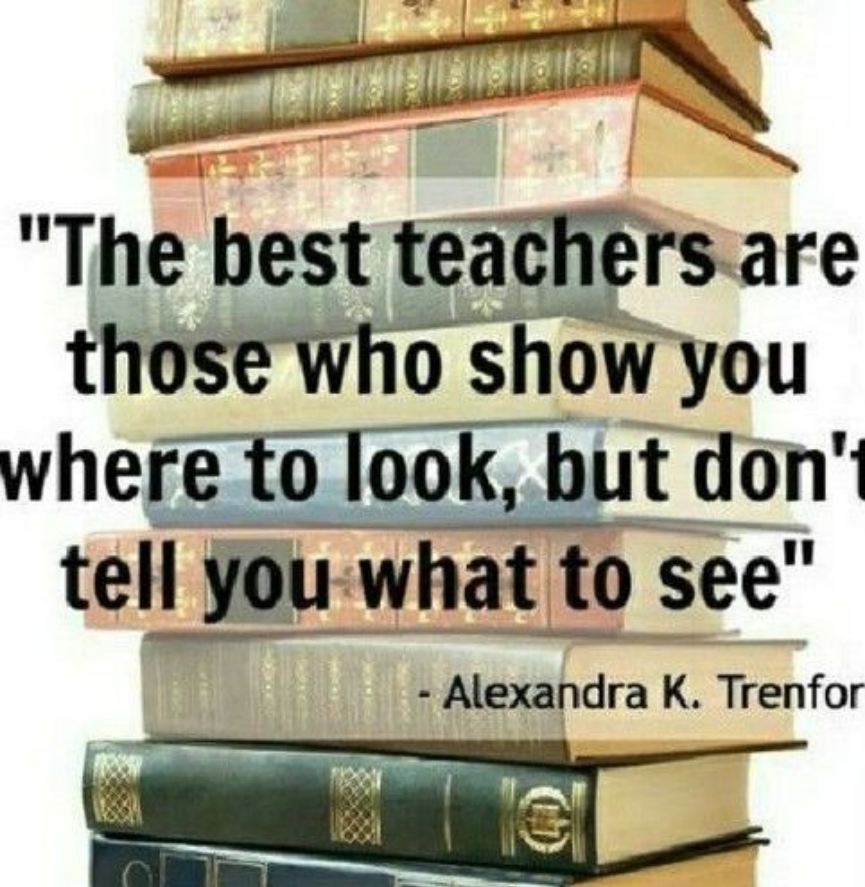
```
[4]: ## the following example use python's request to send restapi requests
import base64
import json
import requests

with open('floribunda.jpg', 'rb') as f:
    image_content = f.read()
    image_64_encode = base64.encodebytes(image_content).decode('utf-8')
headers = {"Cookie": "authservice_session=MTY1Mjc5MzE5M3x0d3dBTkZkUk4xZEROMGRZVmpZlDsTkpwVWFuZU2s5V1dF...",
           "Host": "tensorflow-sample.kubeflow-user-example-com.example.com"}
payload = {"instances": [{"image_bytes": {"b64": image_64_encode}, "key": "1"}]}
resp = requests.post('http://istio-ingressgateway.istio-system/v1/models/tensorflow-sample:predict', headers=headers, data=payload)
print(resp.text)

{
  "predictions": [
    {
      "scores": [1.30671893e-07, 3.01086693e-08, 0.814807534, 9.6436537e-08, 0.185192183, 3.4390
2293e-08],
      "prediction": 2,
      "key": "1"
    }
  ]
}
```

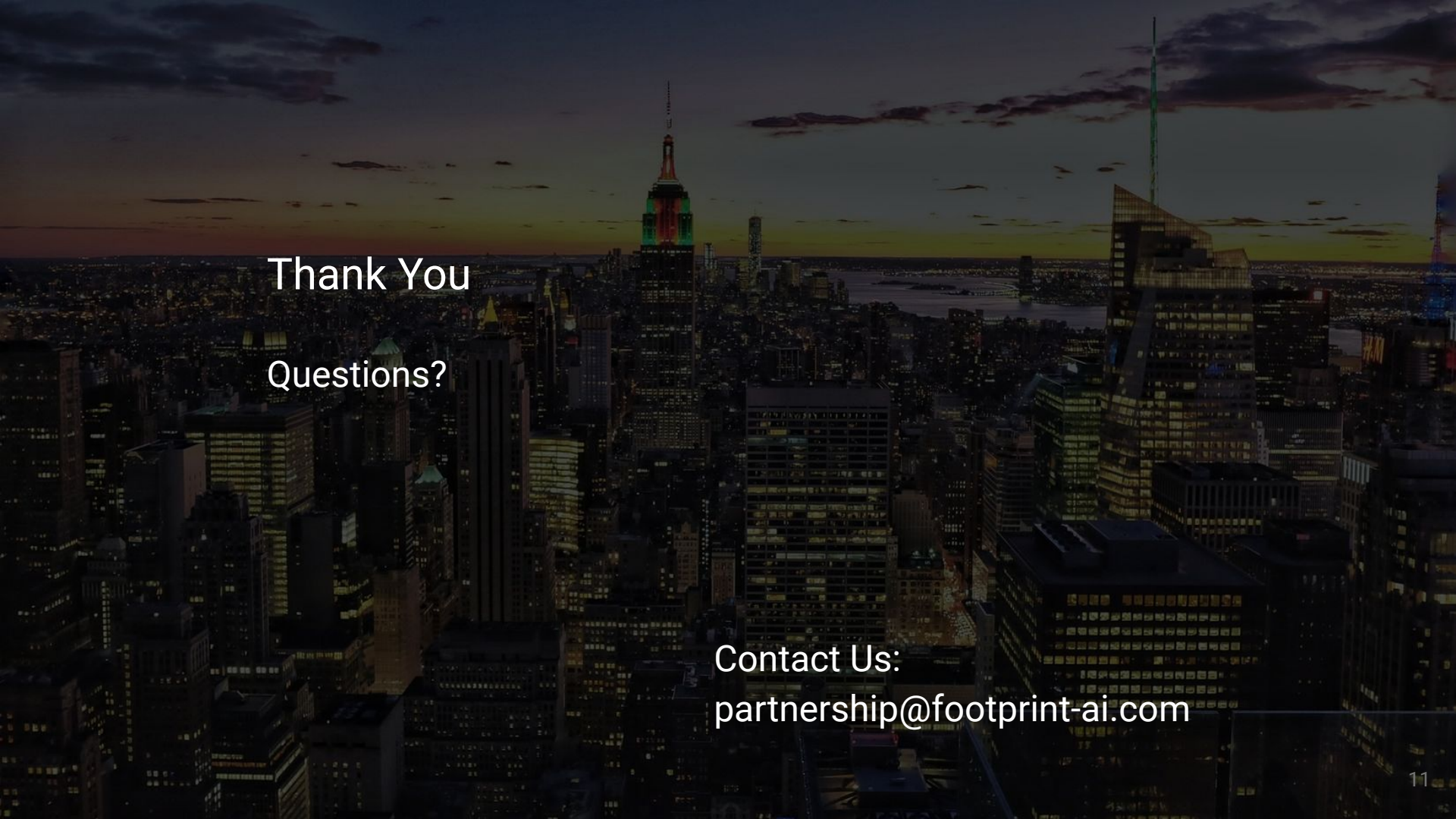
```
[ ]: ## the following example use curl to send restapi requests
```

Quote: The best teachers are those who show you where to look, but don't tell you what to see.



**"The best teachers are
those who show you
where to look, but don't
tell you what to see"**

- Alexandra K. Trenfor

An aerial photograph of the New York City skyline at dusk. The Empire State Building is prominently featured in the center, illuminated with red and green lights. The city is densely packed with skyscrapers, and the Hudson River is visible in the background under a twilight sky with scattered clouds.

Thank You
Questions?

Contact Us:
partnership@footprint-ai.com