Kubeflow Workshop

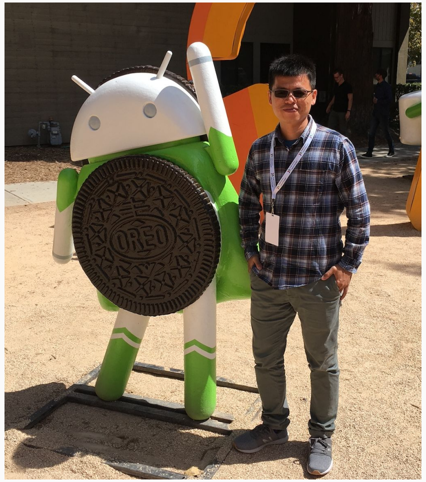
葉信和 / Hsin-Ho Yeh
Software Engineer / Founder / CEO @ 信誠金融科技
hsinho.yeh@footprint-ai.com

信誠金融科技股份有限公司
XINCHEN FINTECH CO., LTD.

2022/01/08

About me

- 2020 - Present at 信誠金融科技
  - Shrimping: A data-sharing platform
    - https://get-shrimping.footprint-ai.com
  - Tintin: a machine learning platform for everyone
    - https://get-tintin.footprint-ai.com
- 2016 - 2020 at IglooInsure (16M+ in series A+ 2020)
  - Provide digital insurance for e-conomic world
  - Funded in KUL, Headquartered in Singapore
  - First employee/ Engineering Lead / Regional Head/ Chief Engineer
- 2013 - 2016 at Studio Engineering @ hTC
  - Principal Engineer on Cloud Infrastructure Team
- 2009 - 2012 at IIS @ Academia Sinica
  - Computer vision, pattern recognition, and data mining
- CS@CCU, CS@NCKU alumni



課程綱要

- 課前知識
- 概念簡介
- 環境介紹
- 詞彙定義
- 範例練習
- 問與答

課前知識

- 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 export with `Google`
  - https://letmegooglethat.com/?q=you+can+google+it

- It is totally OK if you don’t know what is Container and Kubernetes

孩子，您多久沒唸中文了？

荀子《儒效篇》

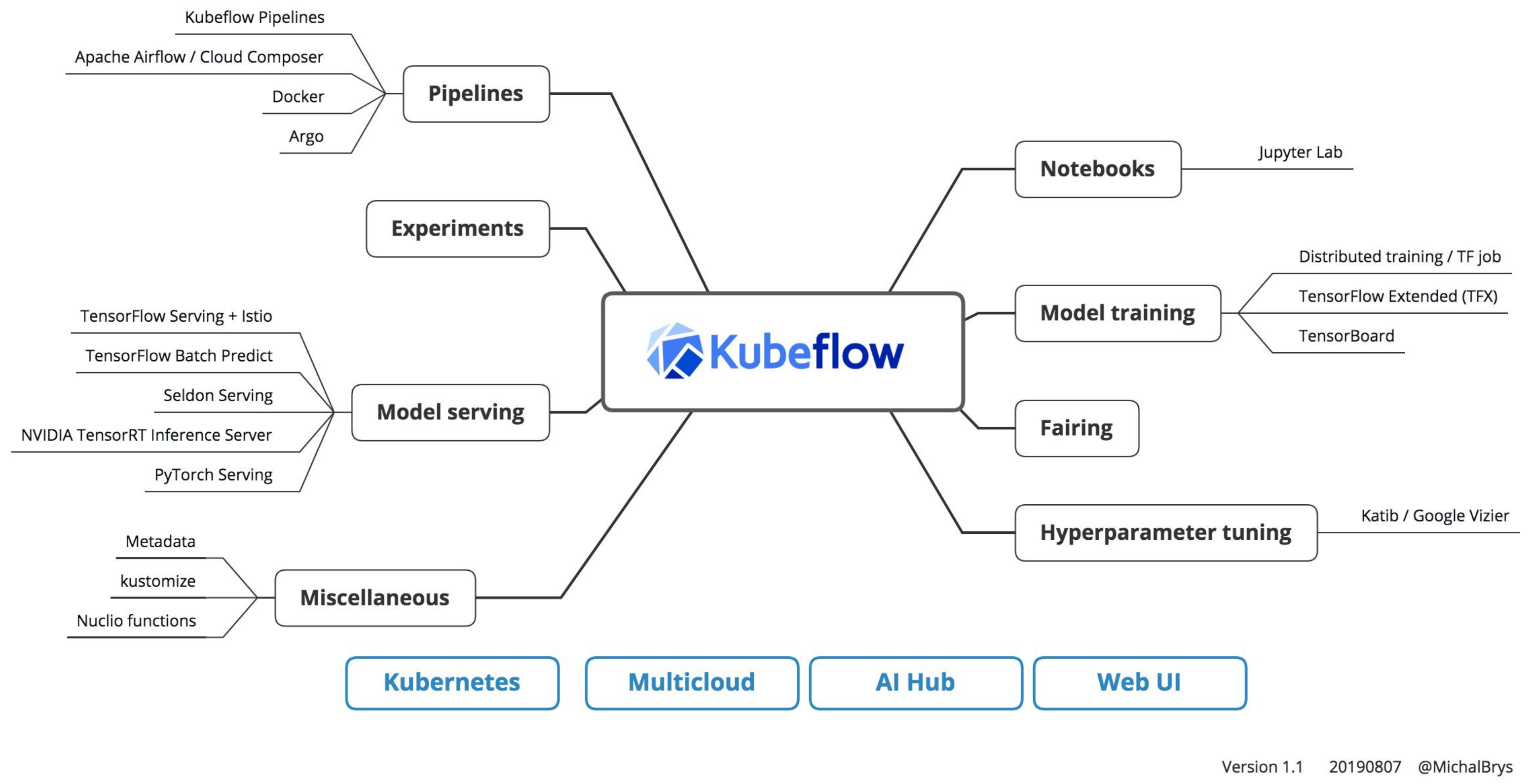
「不聞不若聞之，聞之不若見之，見之不若知之，知之不若行之；學至于行之而止矣。」

範例資源

```
git clone https://github.com/FootprintAI/kubeflow-workshop
```

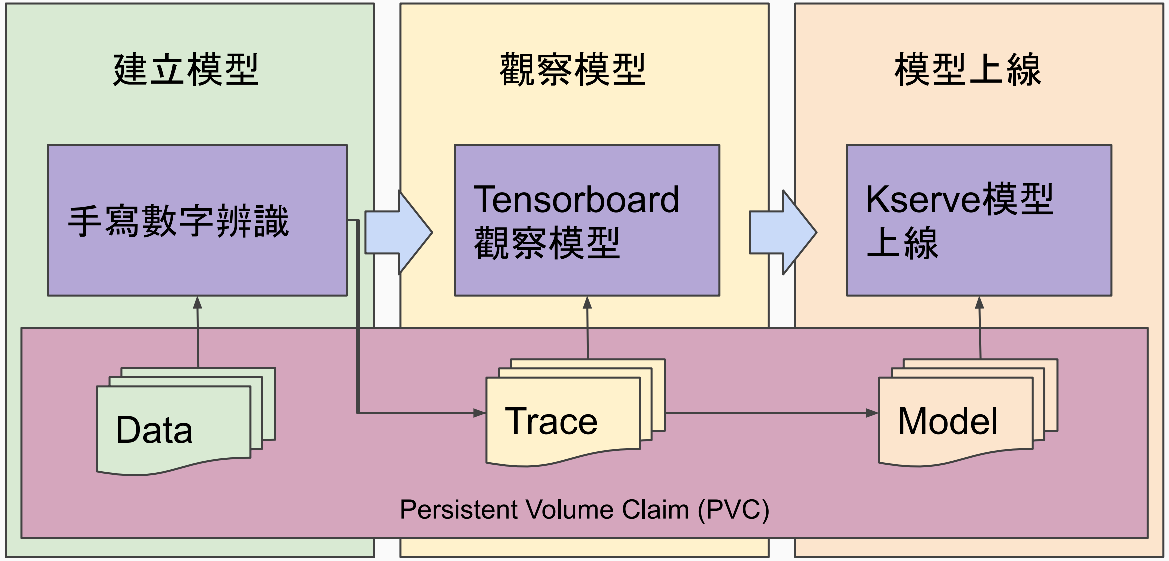
Or Click Me

Kubeflow架構


Kubeflow Pipelines
Apache Airflow / Cloud Composer
Docker
Argo
Pipelines
Experiments
TensorFlow Serving + Istio
TensorFlow Batch Predict
Seldon Serving
NVIDIA TensorRT Inference Server
PyTorch Serving
Model serving
Metadata
kustomize
Nuclio functions
Miscellaneous
Kubeflow
Notebooks
Jupyter Lab
Distributed training / TF job
TensorFlow Extended (TFX)
TensorBoard
Model training
Fairing
Hyperparameter tuning
Katib / Google Vizier
Kubernetes
Multicloud
AI Hub
Web UI
Version 1.1
20190807
@MichalBrys

概念簡介

e2e machine learning flow

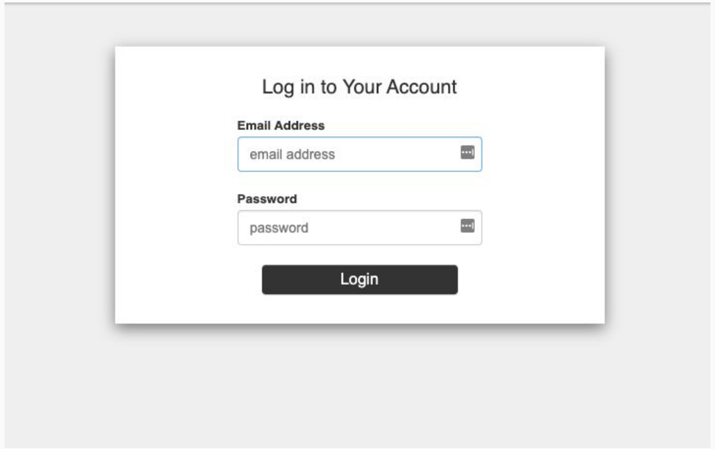


環境介紹

虛擬機環境

1.kubectl port-forward svc/istio-ingressgateway -n istio-system 8080:80 --address 0.0.0.0

2.Open http://localhost:8080


Log in to Your Account

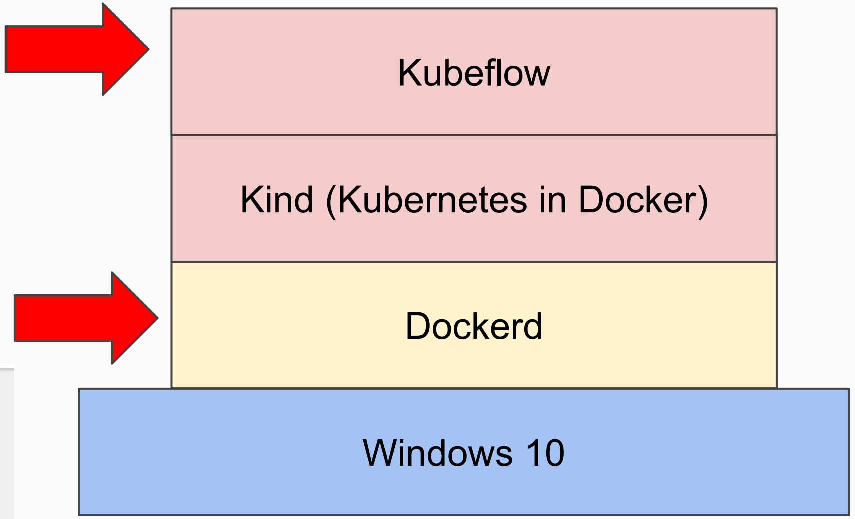
Email Address

email address

Password

password

Login



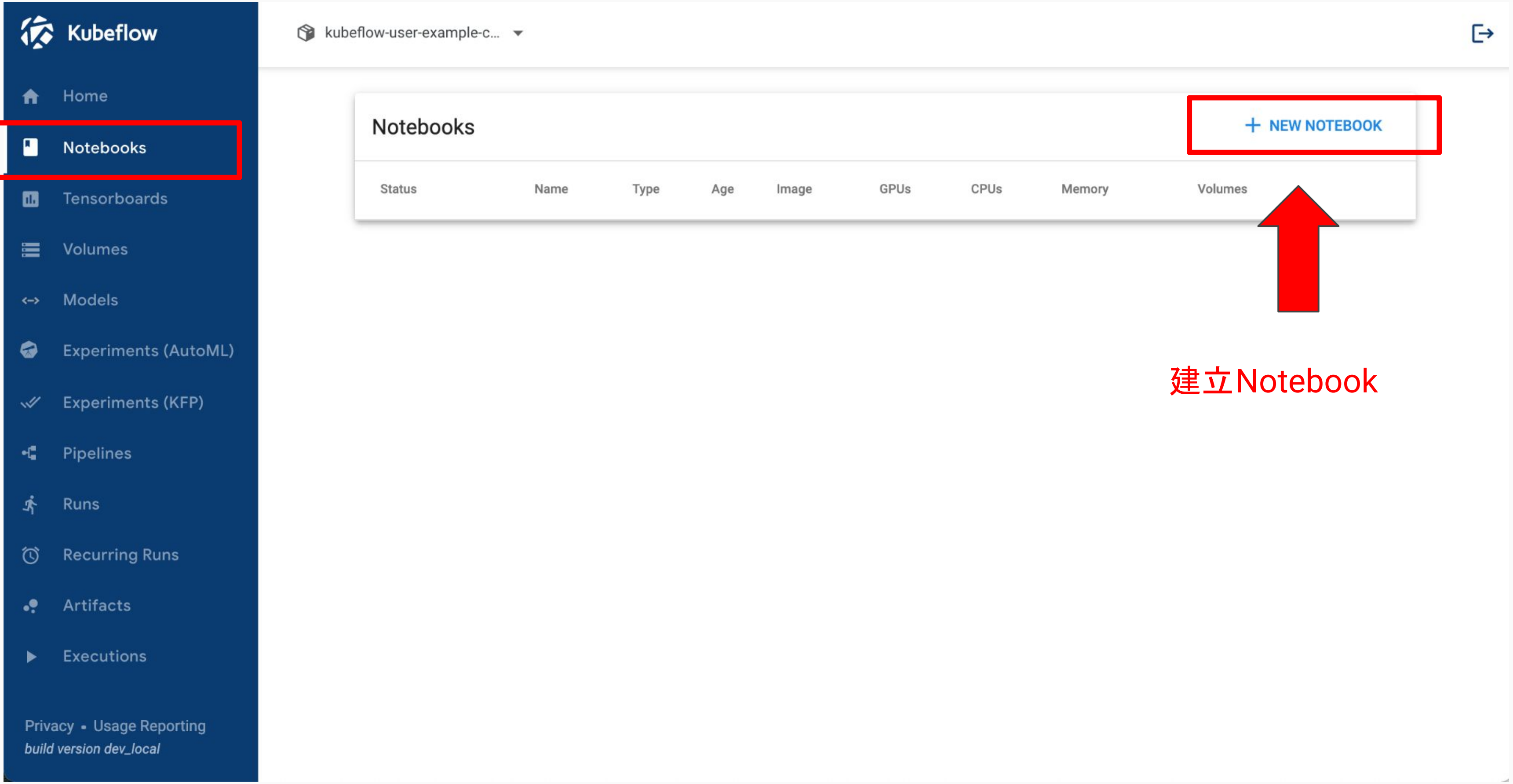
Wait! 所以我說那個帳號密碼呢?

Account: user@example.com
Password: 12341234

Kubectl Cli

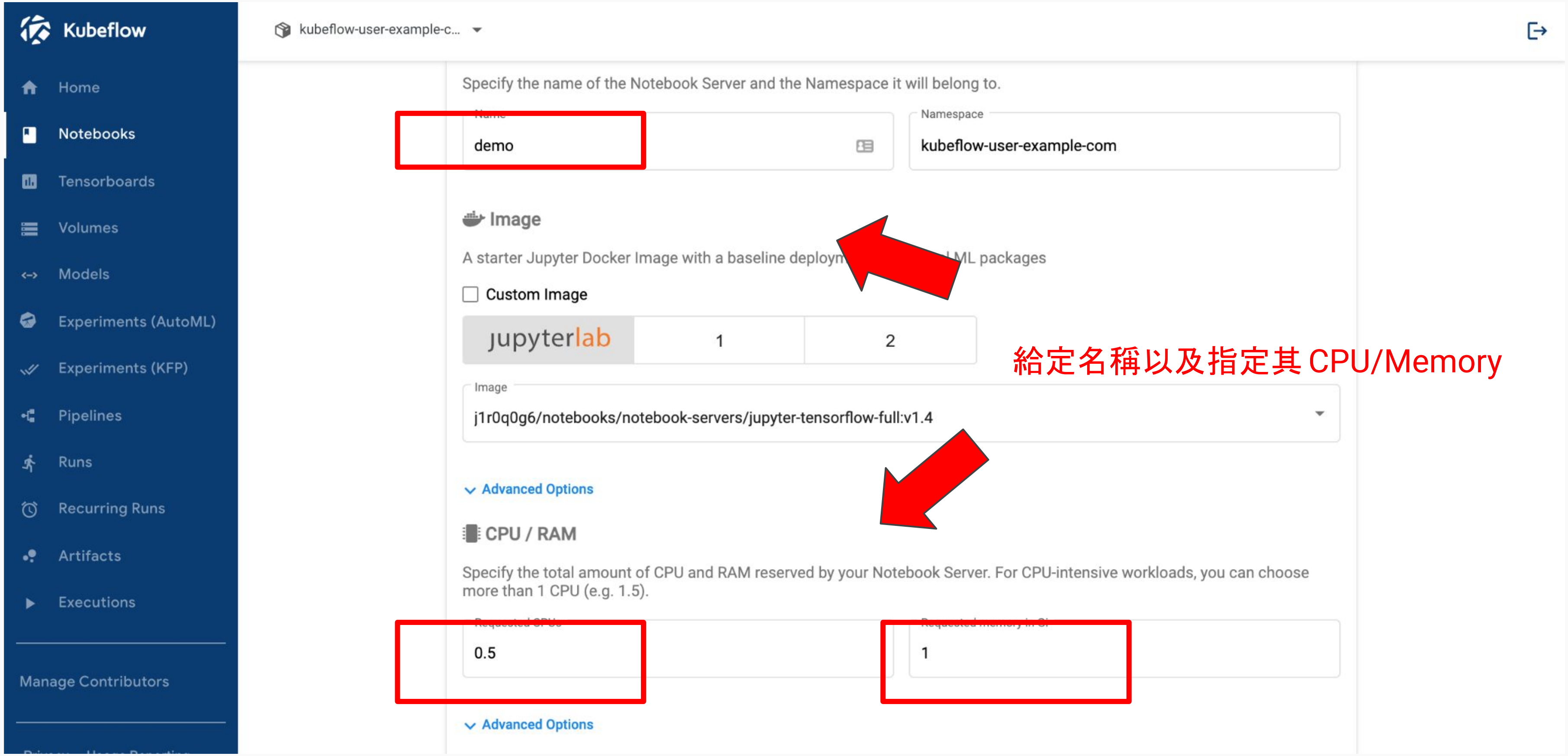
```
// 查看所有namespace
Kubectl get namespaces


// 查看kubeflow中所有運行的Pod
kubectl get pods -n kubeflow


// 查看目前使用者運行的Pod
kubectl get pods -n kubeflow-user-example-com
```

Step1: 開啟Notebook作為開發環境 (1/3)



建立Notebook

Step1: 開啟Notebook作為開發環境 (2/3)



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 deploym... ML packages

Custom Image

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

CPU / RAM

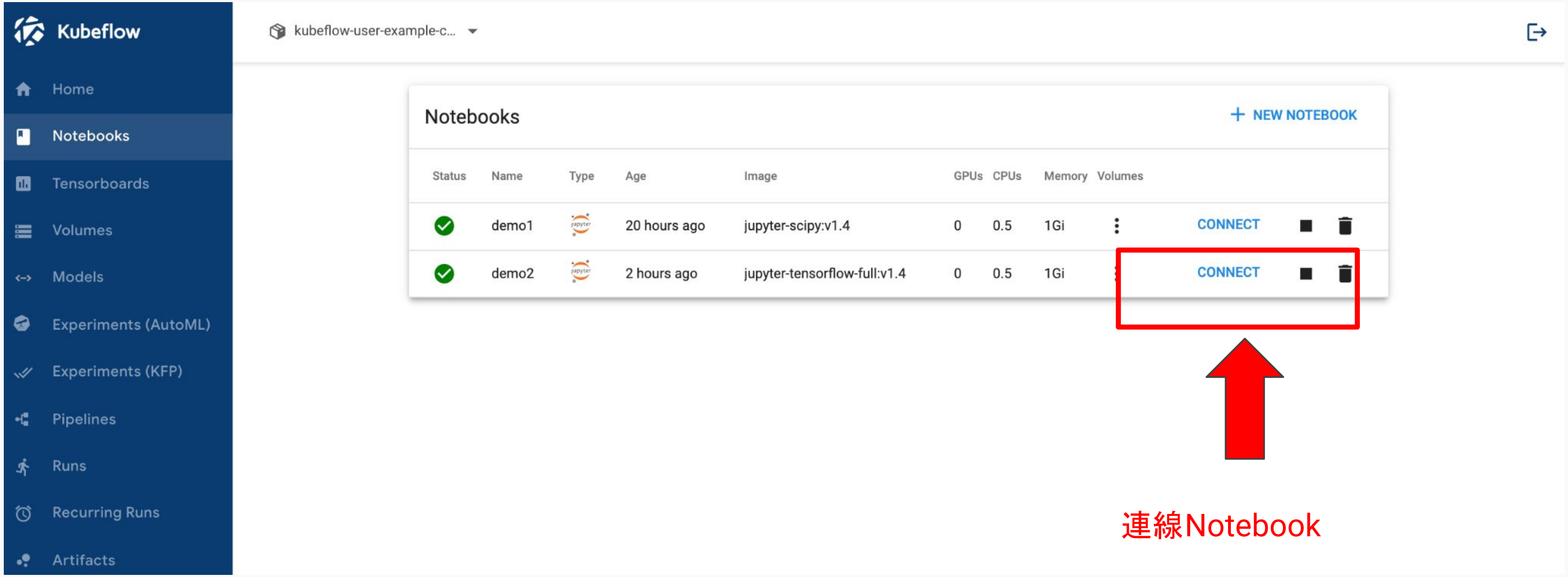
Specify the total amount of CPU and RAM reserved by your Notebook Server. For CPU-intensive workloads, you can choose more than 1 CPU (e.g. 1.5).

Requested CPUs: 0.5

Requested memory in Gi: 1

給定名稱以及指定其 CPU/Memory

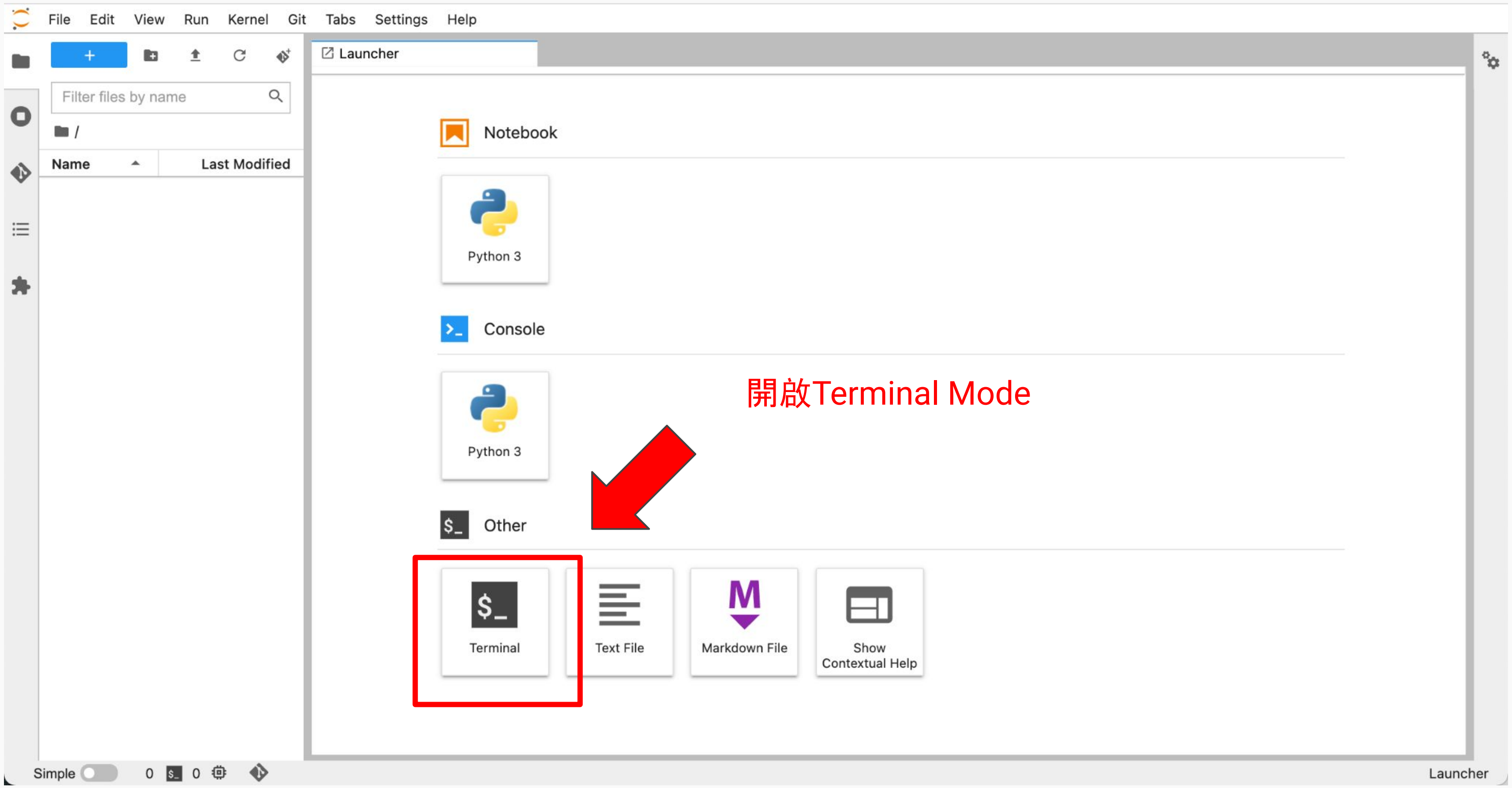
Step1: 開啟Notebook作為開發環境 (3/3)



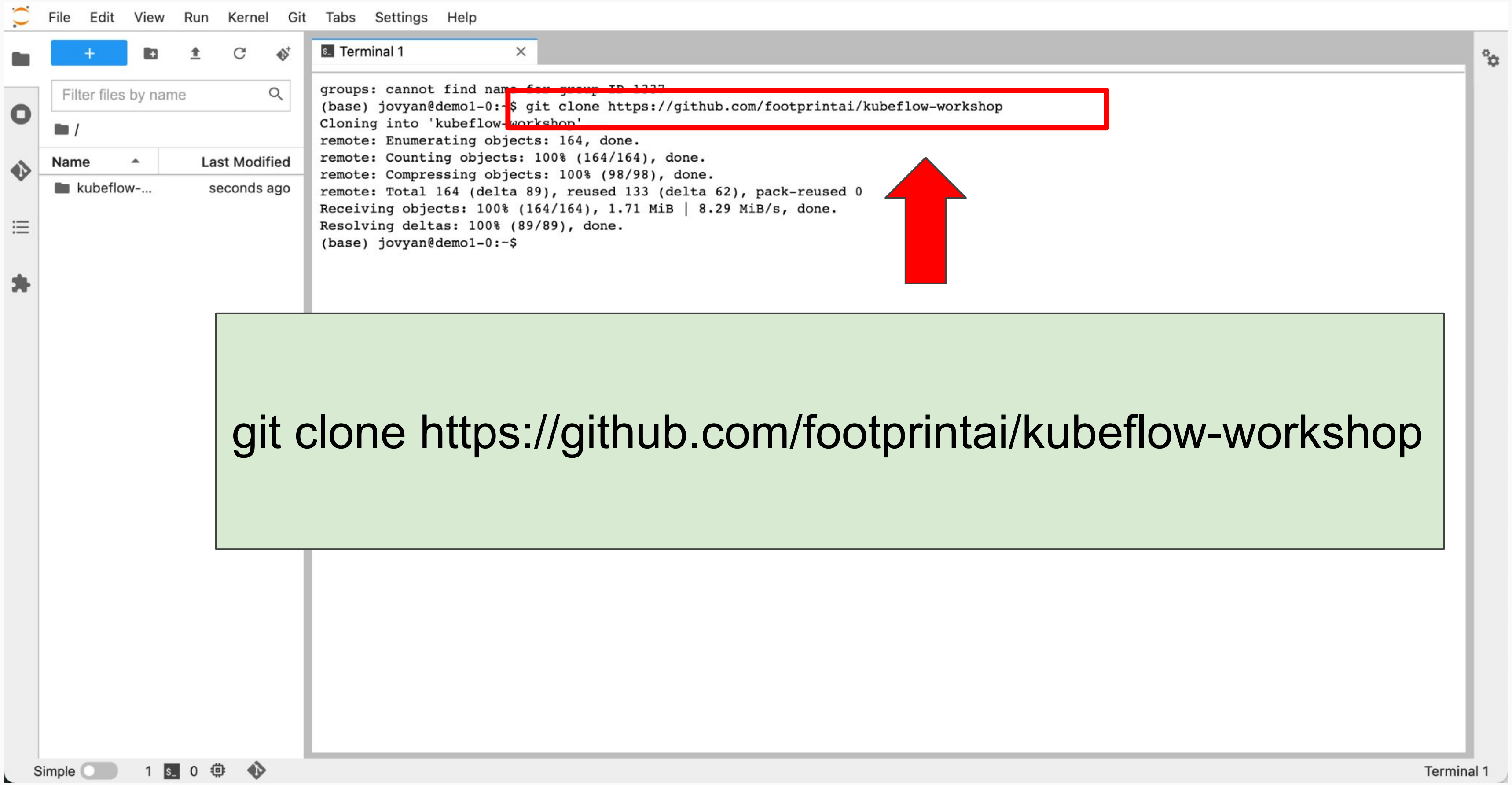
Notebooks

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

Step2: 開啟Terminal下載範例程式(1/3)



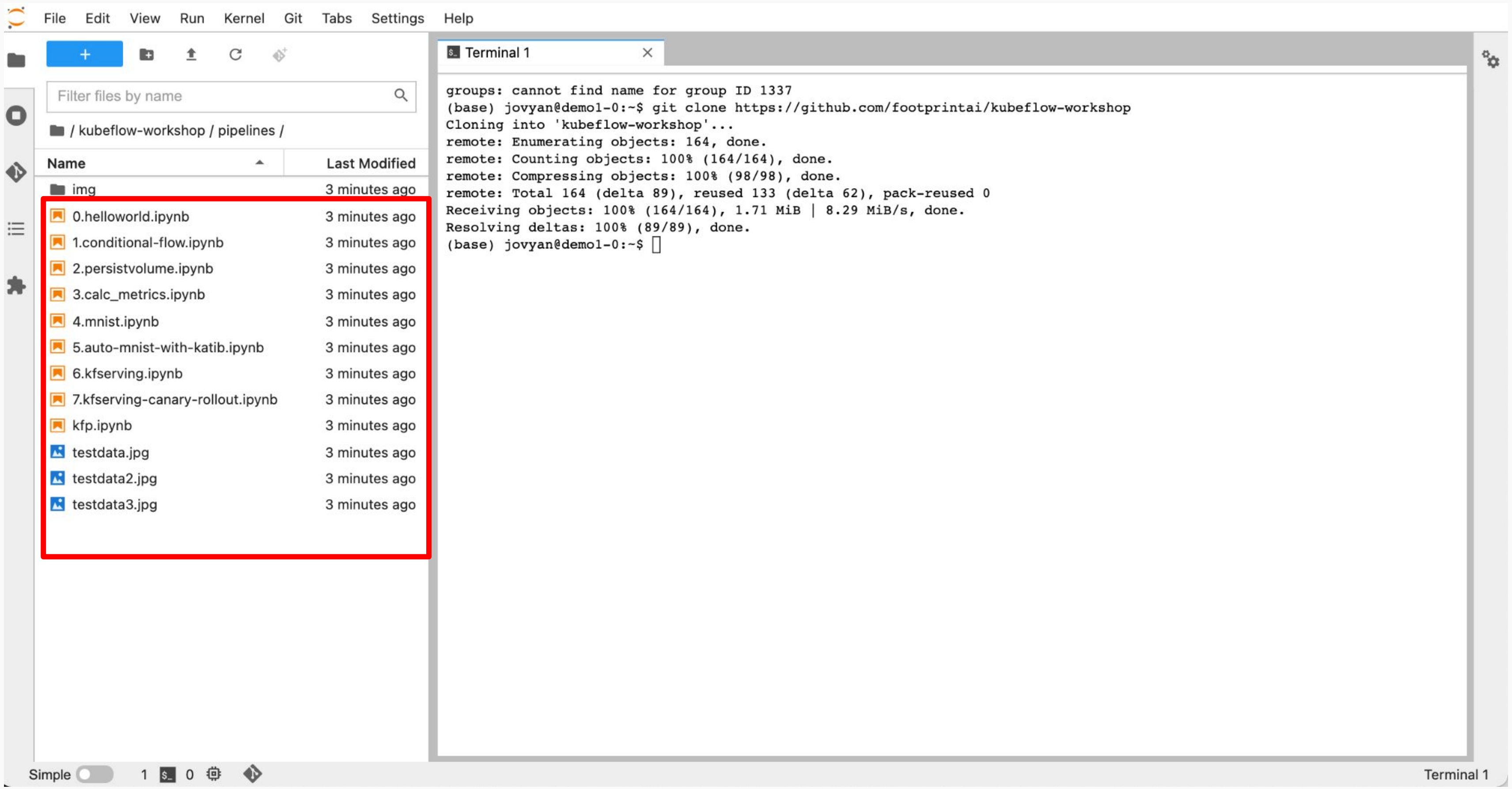
Step2: 開啟Terminal下載範例程式(2/3)



```
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

Step2: 開啟Terminal下載範例程式(3/3)

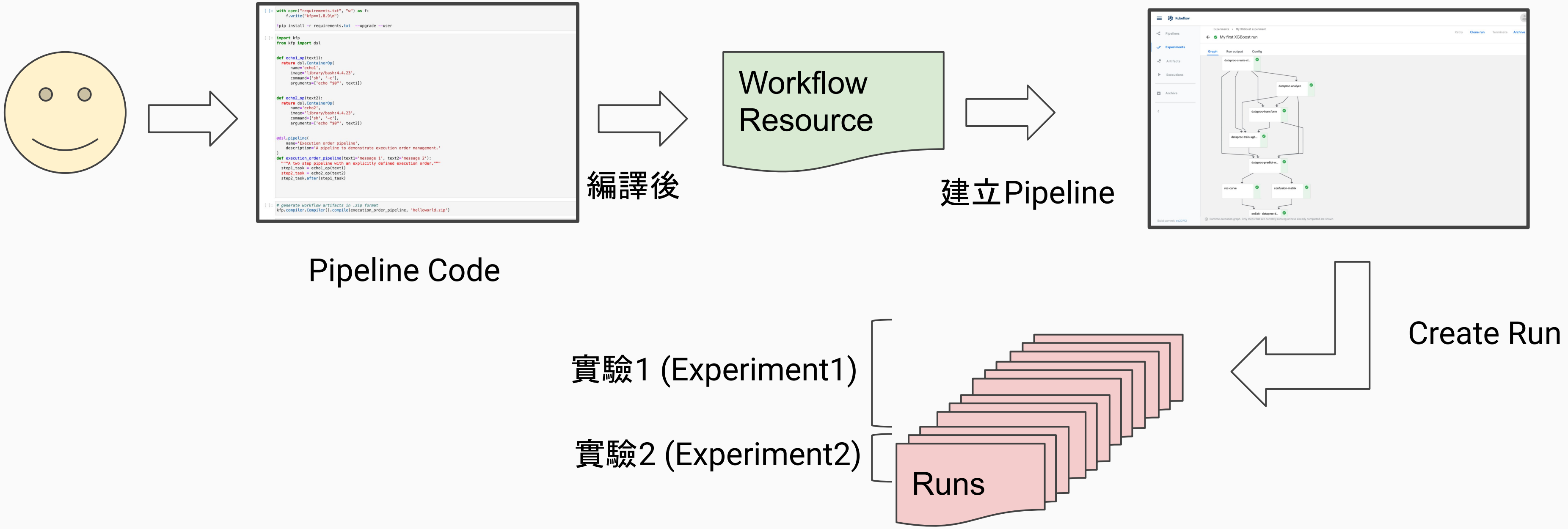


Name	Last Modified
img	3 minutes ago
0.helloworld.ipynb	3 minutes ago
1.conditional-flow.ipynb	3 minutes ago
2.persistvolume.ipynb	3 minutes ago
3.calc_metrics.ipynb	3 minutes ago
4.mnist.ipynb	3 minutes ago
5.auto-mnist-with-katib.ipynb	3 minutes ago
6.kfserving.ipynb	3 minutes ago
7.kfserving-canary-rollout.ipynb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

```
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:~$
```

詞彙說明

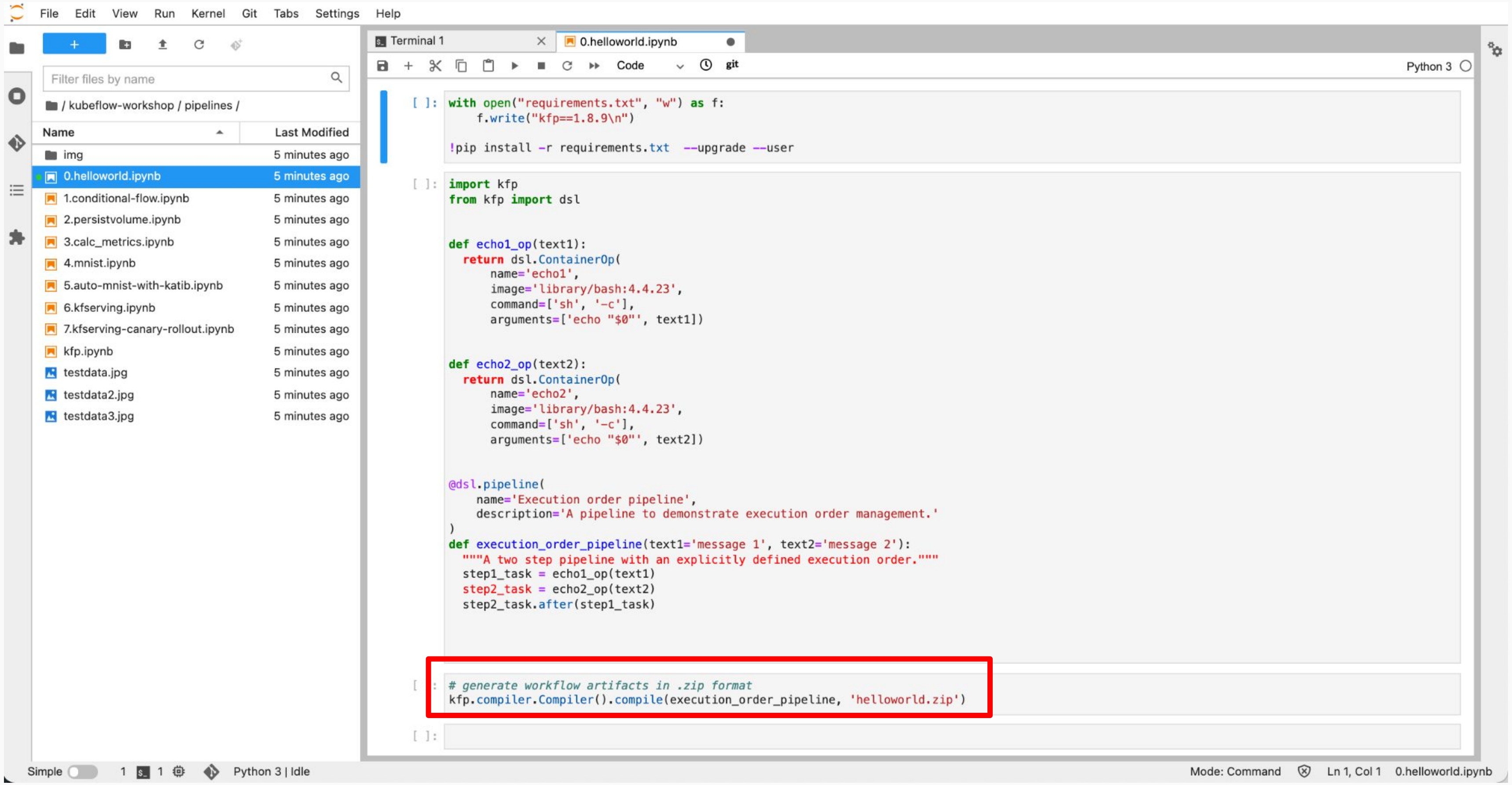
詞彙說明



```
[ ]: 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')
```

範例1 Hello World!

Step3: 編譯helloworld.ipynb (1/2)


File Edit View Run Kernel Git Tabs Settings Help
Filter files by name
/ kubeflow-workshop / pipelines /
Terminal 1
0.helloworld.ipynb
Code
git
Python 3

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

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

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

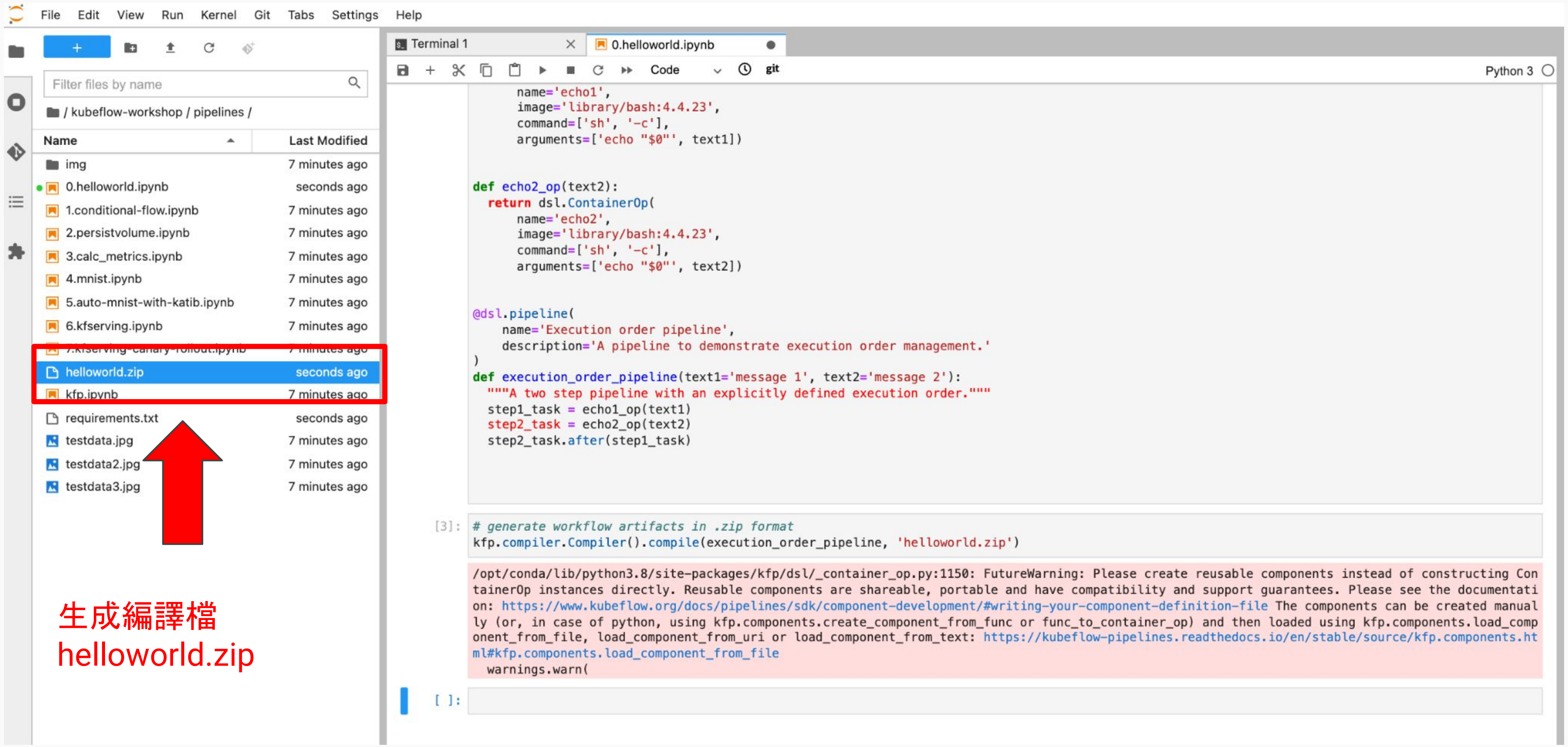
```
[ ]: import kfp
     from kfp import dsl


     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')
```

```
[ ]:
```

Simple 1 1 Python 3 | Idle
Mode: Command Ln 1, Col 1 0.helloworld.ipynb

Step3: 編譯helloworld.ipynb (2/2)



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

```
      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)
```

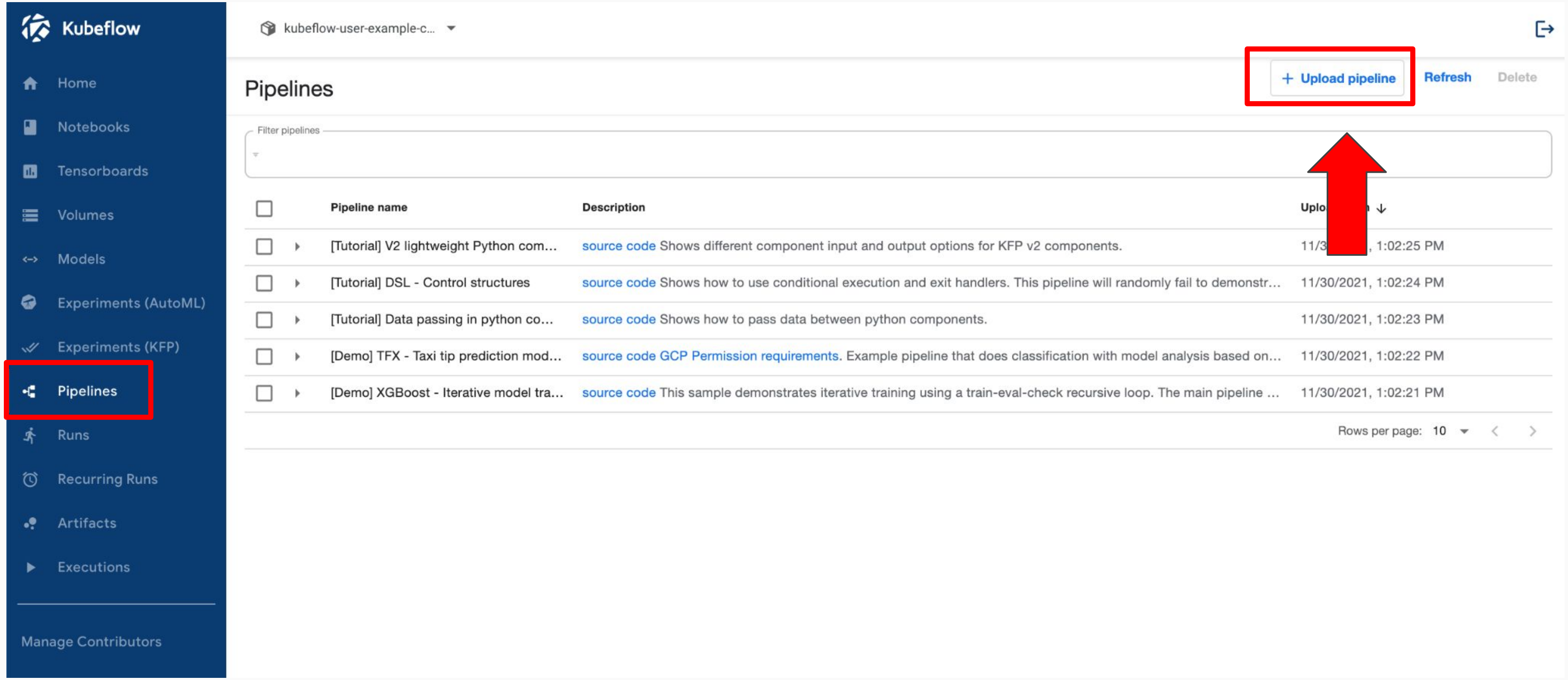
```
[3]: # generate workflow artifacts in .zip format
     kfp.compiler.Compiler().compile(execution_order_pipeline, 'helloworld.zip')
```

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

```
[ ]:
```

生成編譯檔
helloworld.zip

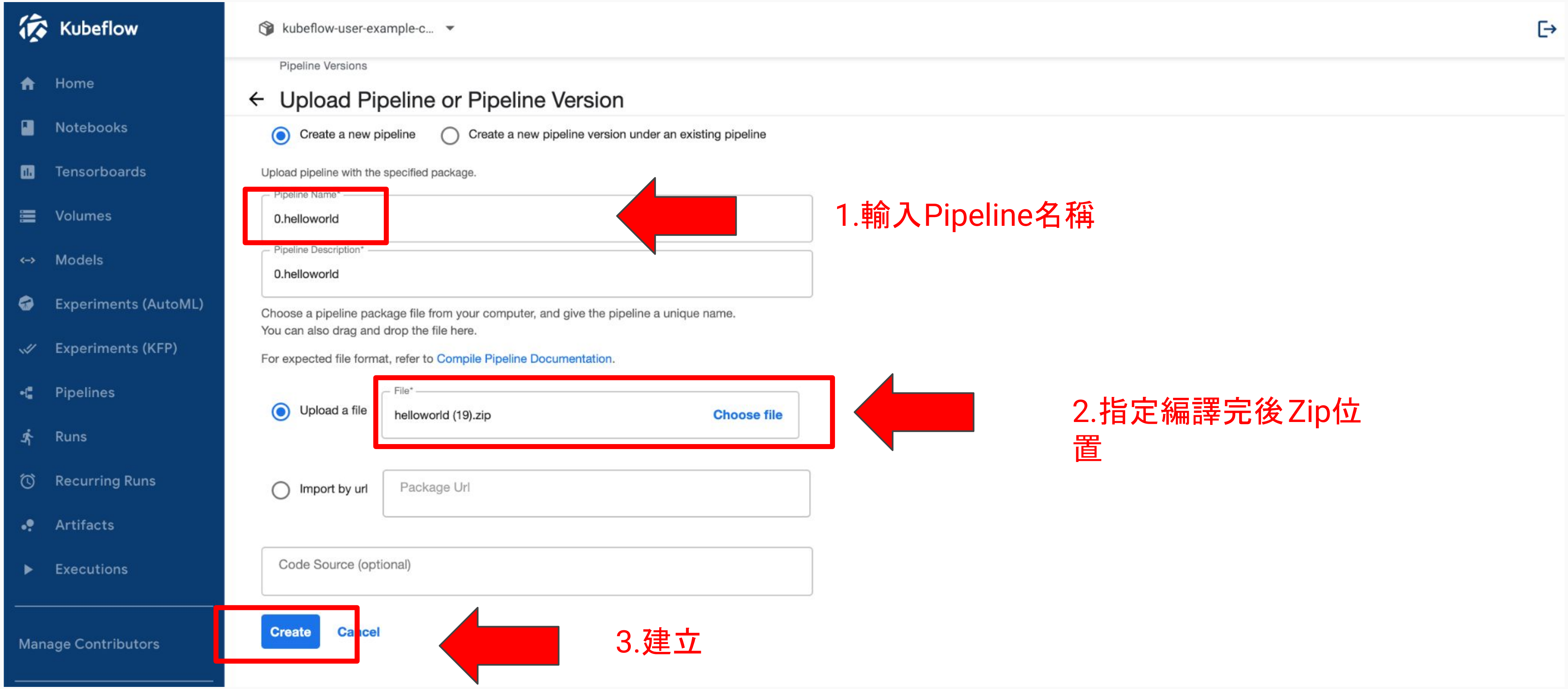
Step4: 建立Pipeline (1/7)



Pipelines

Pipeline name	Description	Uplo...
[Tutorial] V2 lightweight Python com...	source code Shows different component input and output options for KFP v2 components.	11/3..., 1:02:25 PM
[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
[Tutorial] Data passing in python co...	source code Shows how to pass data between python components.	11/30/2021, 1:02:23 PM
[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
[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

Step4: 建立Pipeline (2/7)



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] File*: helloworld (19).zip Choose file

Import by url: Package Url

Code Source (optional)

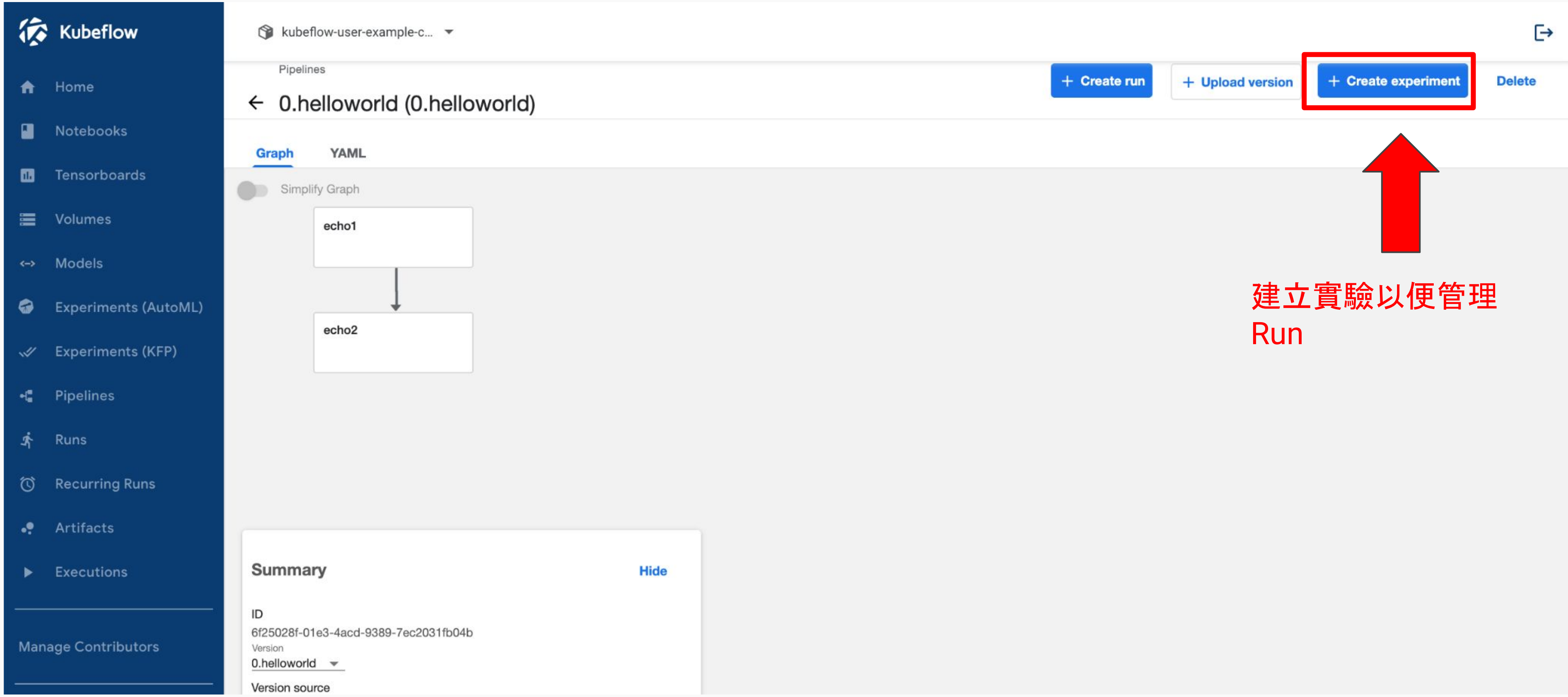
Create Cancel

1.輸入Pipeline名稱

2.指定編譯完後Zip位置

3.建立

Step4: 建立Pipeline (3/7)



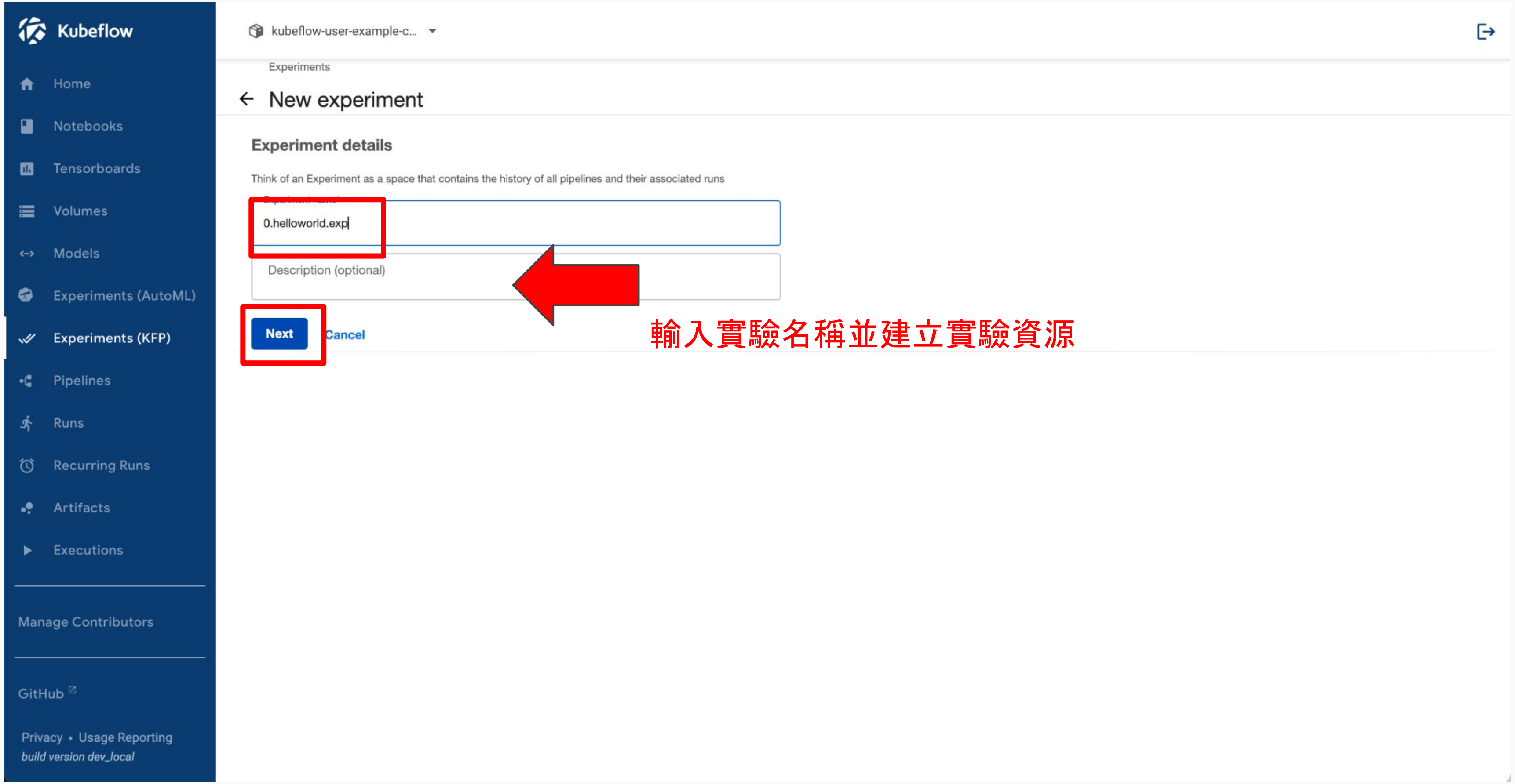
0.helloworld (0.helloworld)

Summary

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

建立實驗以便管理Run

Step4: 建立Pipeline (4/7)



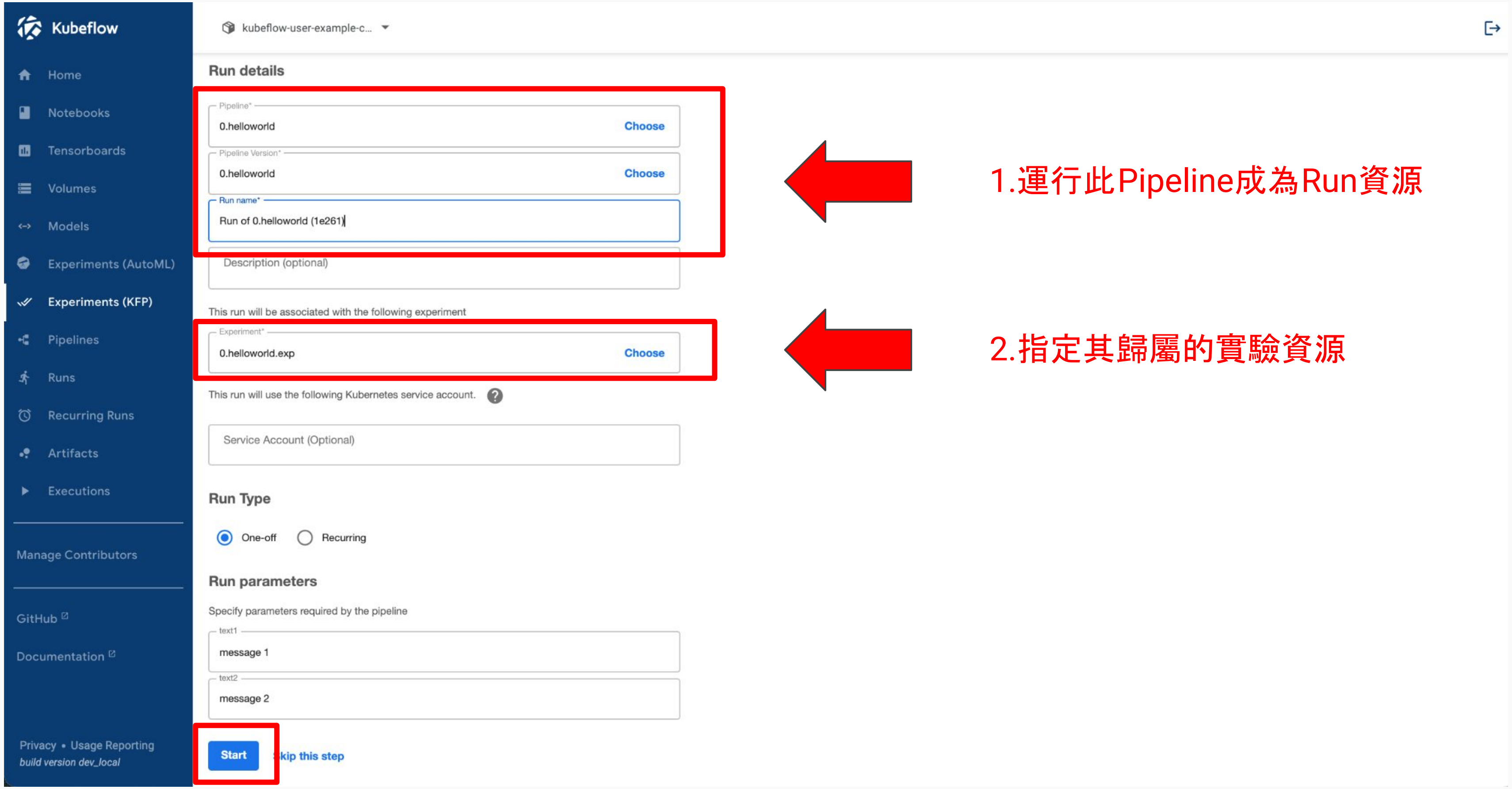
Think of an Experiment as a space that contains the history of all pipelines and their associated runs

Experiment name: 0.helloworld.exp

Description (optional)

輸入實驗名稱並建立實驗資源

Step4: 建立Pipeline (5/7)



Run details

Pipeline*: 0.helloworld

Pipeline Version*: 0.helloworld

Run name*: Run of 0.helloworld (1e261)

Description (optional)

This run will be associated with the following experiment

Experiment*: 0.helloworld.exp

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

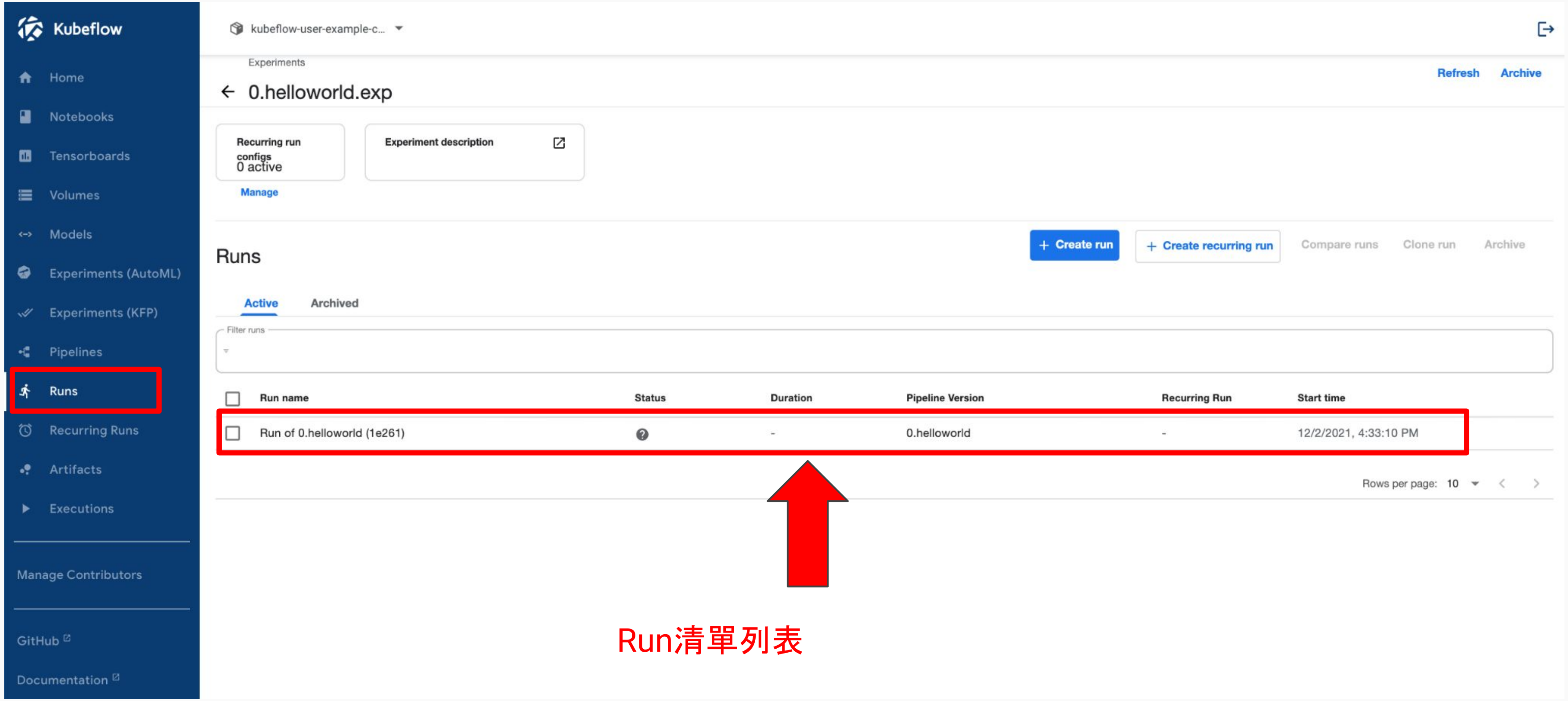
text1: message 1

text2: message 2

1.運行此Pipeline成為Run資源

2.指定其歸屬的實驗資源

Step4: 建立Pipeline (6/7)



Experiments

0.helloworld.exp

Recurring run configs
0 active

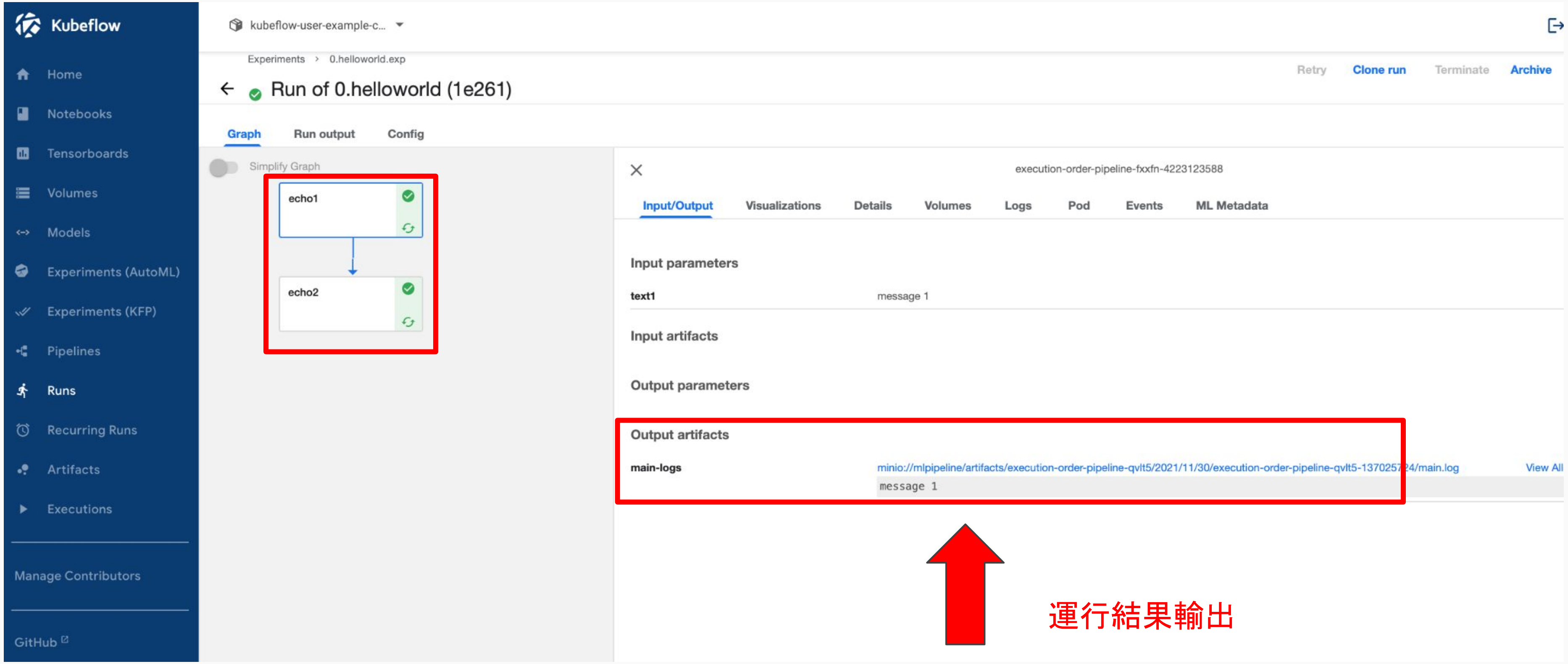
Experiment description

Runs

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

Run清單列表

Step4: 建立Pipeline (7/7)



Run of 0.helloworld (1e261)

execution-order-pipeline-fxxfn-4223123588

Input parameters

text1	message 1

Input artifacts

Output parameters

Output artifacts

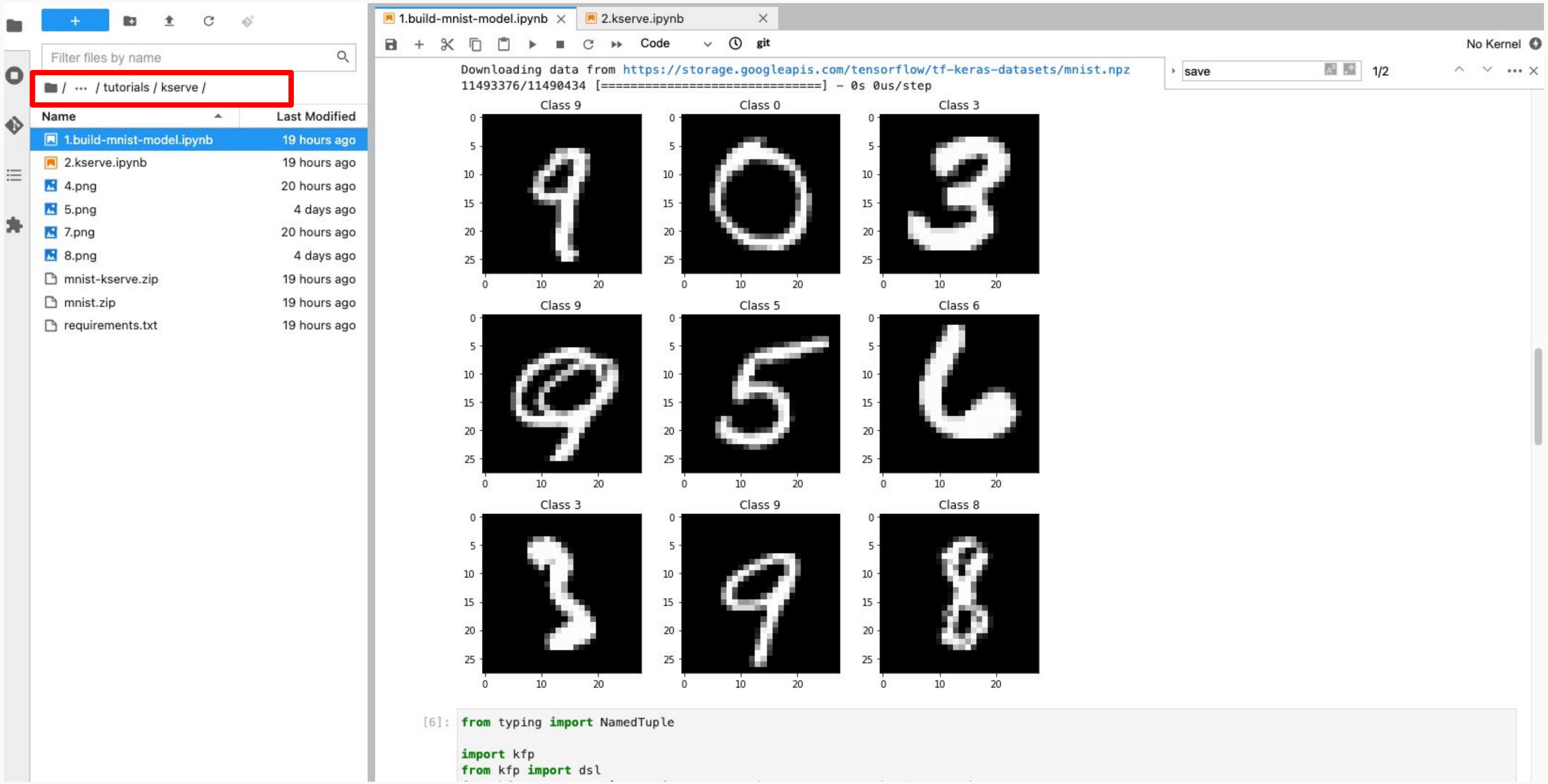
main-logs	minio://mlpipeline/artifacts/execution-order-pipeline-qvlt5/2021/11/30/execution-order-pipeline-qvlt5-1370257?4/main.log

```
message 1
```

運行結果輸出

範例2 e2e 模型建立/觀察/上線

Step1: 建立Mnist模型 (1/3)

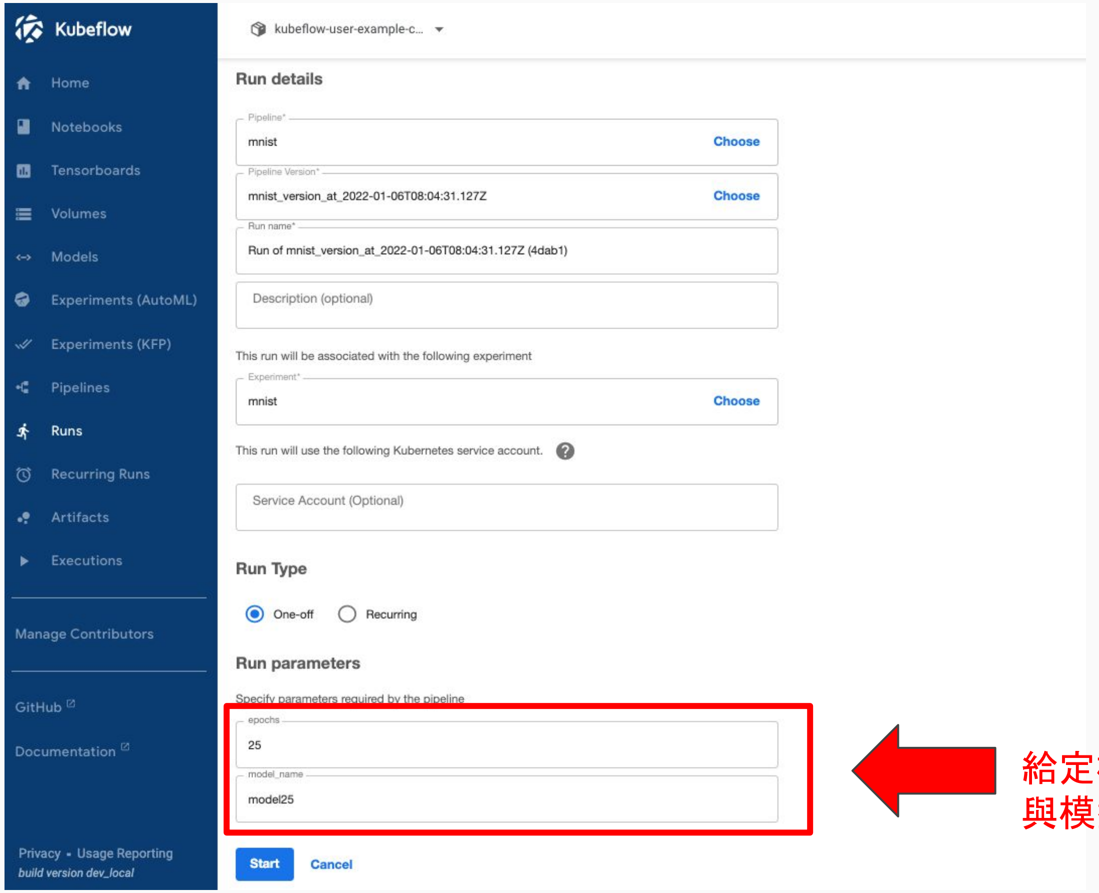


Name	Last Modified
1.build-mnist-model.ipynb	19 hours ago
2.kserve.ipynb	19 hours ago
4.png	20 hours ago
5.png	4 days ago
7.png	20 hours ago
8.png	4 days ago
mnist-kserve.zip	19 hours ago
mnist.zip	19 hours ago
requirements.txt	19 hours ago

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11493376/11490434 [==============================] - 0s 0us/step
```

```
[6]: from typing import NamedTuple

     import kfp
     from kfp import dsl
```

Step1: 建立Mnist模型 (2/3)



Run details

Pipeline*: mnist

Pipeline Version*: mnist_version_at_2022-01-06T08:04:31.127Z

Run name*: Run of mnist_version_at_2022-01-06T08:04:31.127Z (4dab1)

Description (optional)

This run will be associated with the following experiment

Experiment*: mnist

This run will use the following Kubernetes service account.

Service Account (Optional)

Run Type

[✓ One-off] Recurring

Run parameters

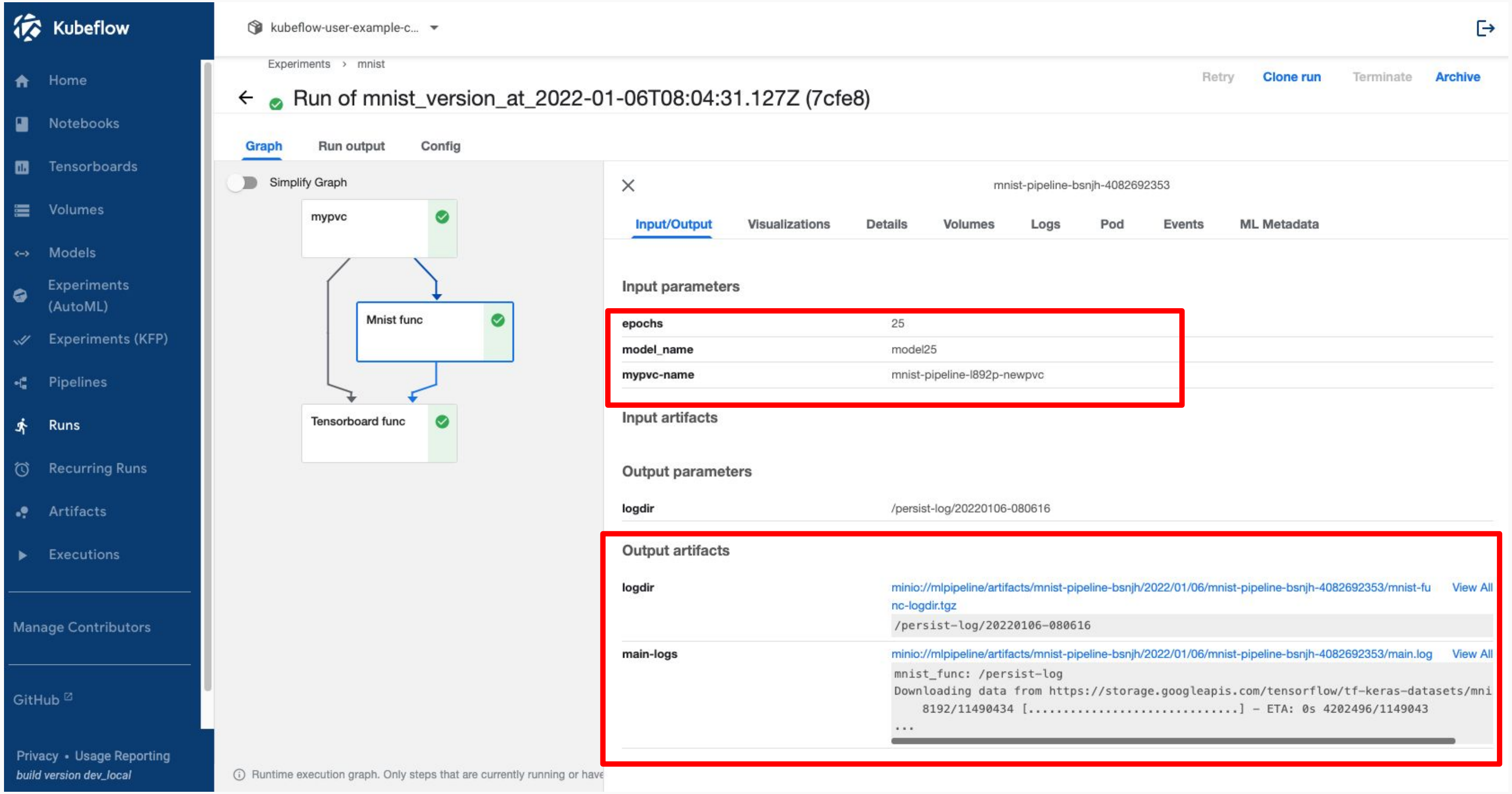
Specify parameters required by the pipeline

epochs: 25

model_name: model25

給定模型參數
與模型名稱

Step1: 建立Mnist模型 (3/3)



Run of mnist_version_at_2022-01-06T08:04:31.127Z (7cfe8)

mnist-pipeline-bsnjh-4082692353

Input parameters

epochs	25
model_name	model25
mypvc-name	mnist-pipeline-l892p-newpvc

Input artifacts

Output parameters

logdir	/persist-log/20220106-080616

Output artifacts

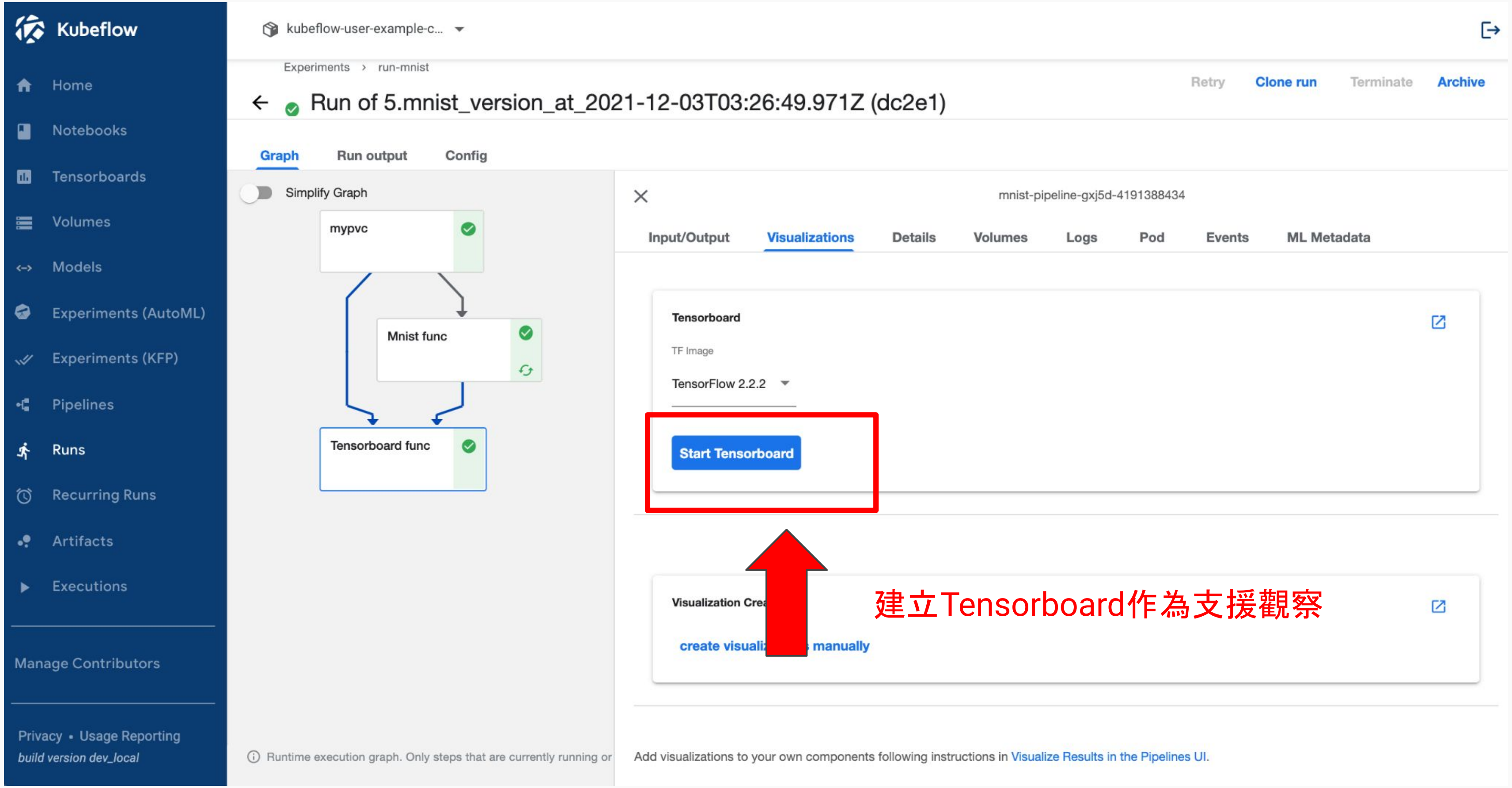
logdir: minio://mlpipeline/artifacts/mnist-pipeline-bsnjh/2022/01/06/mnist-pipeline-bsnjh-4082692353/mnist-func-logdir.tgz

```
/persist-log/20220106-080616
```

main-logs: minio://mlpipeline/artifacts/mnist-pipeline-bsnjh/2022/01/06/mnist-pipeline-bsnjh-4082692353/main.log

```
mnist_func: /persist-log
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mni
    8192/11490434 [..............................] - ETA: 0s 4202496/1149043
...
```

Step2: 分析Mnist模型-Tensorboard (1/5) - Optional



Run of 5.mnist_version_at_2021-12-03T03:26:49.971Z (dc2e1)

Runtime execution graph. Only steps that are currently running or

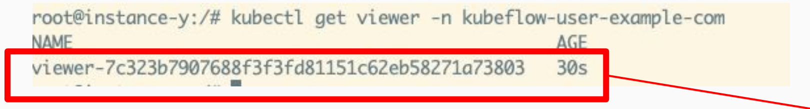
Add visualizations to your own components following instructions in Visualize Results in the Pipelines UI.

Step2: 分析Mnist模型-Tensorboard (2/5) - Optional

(此步為進階功能，若無法實作也無妨)

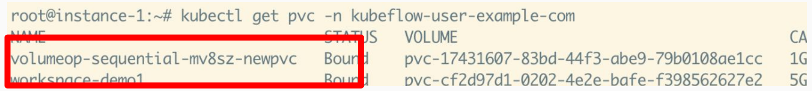
為了讓tensorboard可以查看本地端的資料，我們得將其對應資源建立起：

1. 先獲得viewer的id



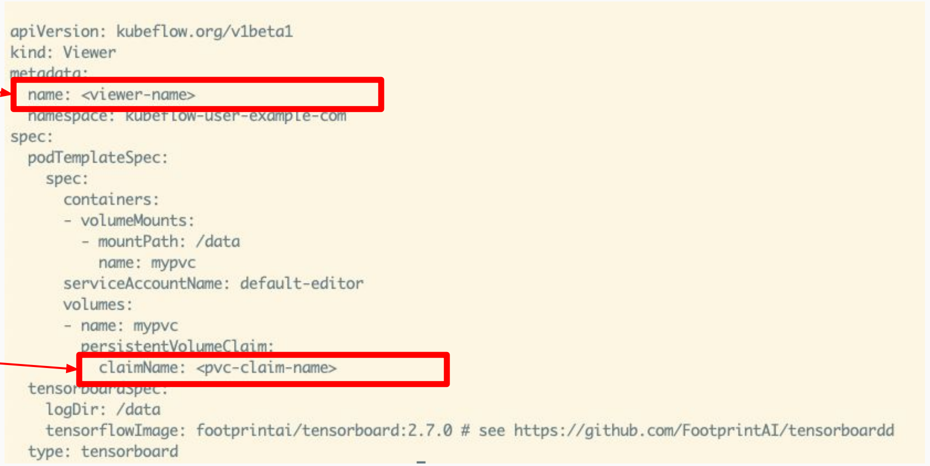
```
root@instance-y:/# kubectl get viewer -n kubeflow-user-example-com
NAME                                                AGE
viewer-7c323b7907688f3f3fd81151c62eb58271a73803   30s
```

2. 先尋找pvc的位置



```
root@instance-1:~# kubectl get pvc -n kubeflow-user-example-com
NAME                             STATUS   VOLUME                                     CA
volumeop-sequential-mv8sz-newpvc Bound    pvc-17431607-83bd-44f3-abe9-79b0108ae1cc   1G
workspace-demo1                  Bound    pvc-cf2d97d1-0202-4e2e-bafe-f398562627e2   5G
```

3. 修改至hack/tensorboard-use-local-volume.yaml底下



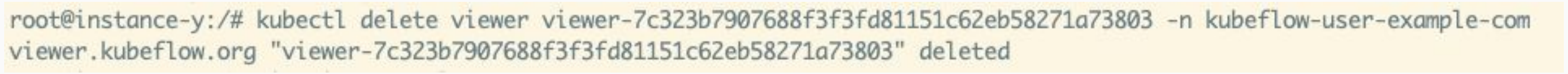
```
apiVersion: kubeflow.org/v1beta1
kind: Viewer
metadata:
  name: <viewer-name>
  namespace: kubeflow-user-example-com
spec:
  podTemplateSpec:
    spec:
      containers:
      - volumeMounts:
        - mountPath: /data
          name: mypvc
      serviceAccountName: default-editor
      volumes:
      - name: mypvc
        persistentVolumeClaim:
          claimName: <pvc-claim-name>
  tensorboardSpec:
    logDir: /data
    tensorflowImage: footprintai/tensorboard:2.7.0 # see https://github.com/FootprintAI/tensorboardd
  type: tensorboard
```

Step2: 分析Mnist模型-Tensorboard (3/5) - Optional

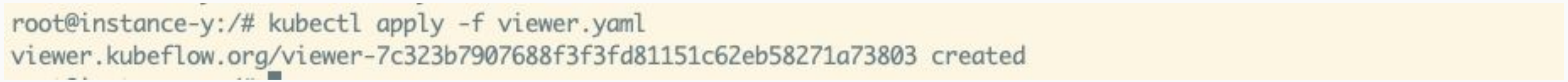
(此步為進階功能，若無法實作也無仿)

為了讓tensorboard可以查看本地端的資料，我們得將其對應資源建立起：

4. 刪掉舊有Viewer並先增新的Viewer (or
kubectl delete viewer -n kubeflow-user-example-com –all)

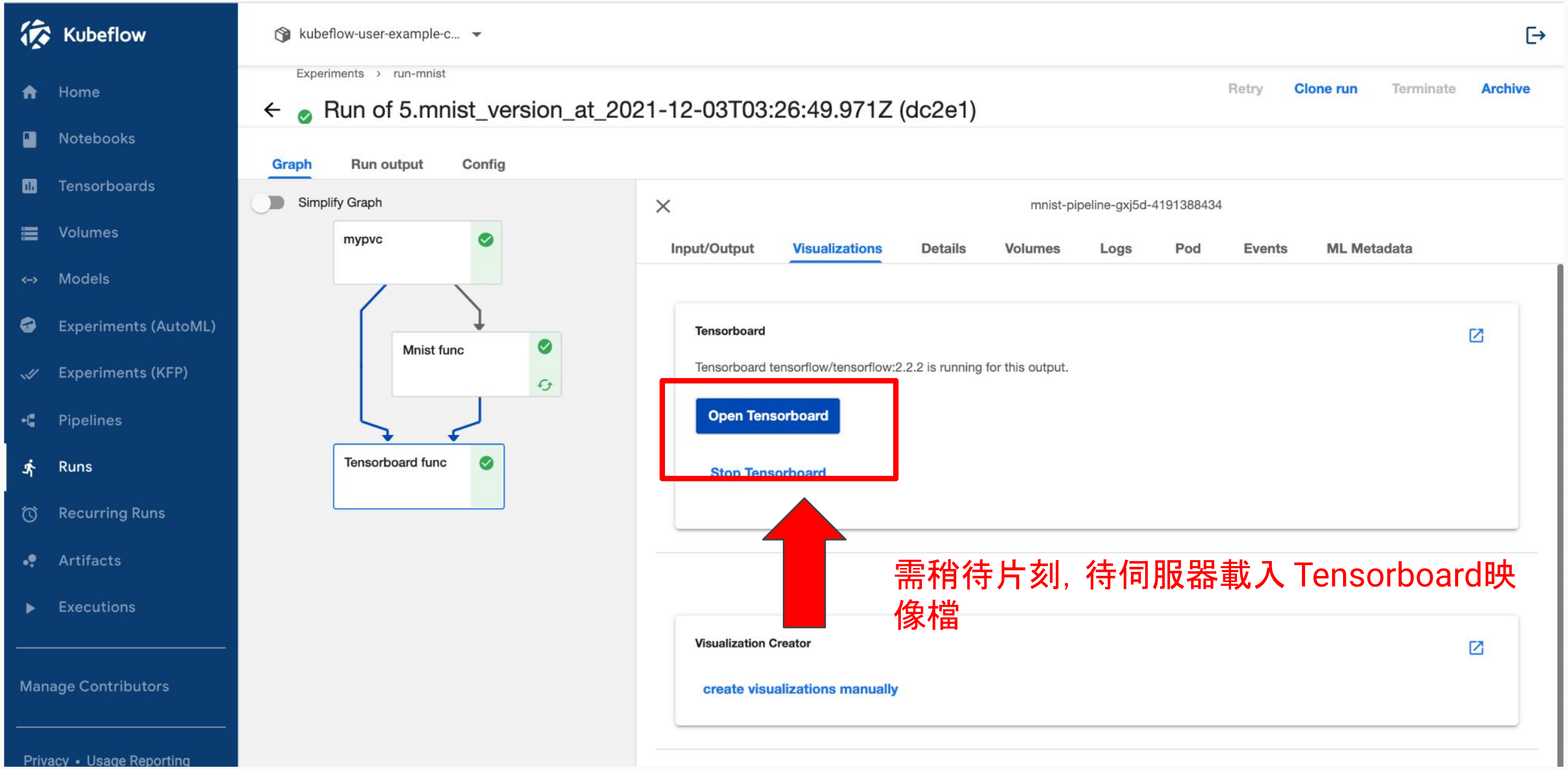


```
root@instance-y:/# kubectl delete viewer viewer-7c323b7907688f3f3fd81151c62eb58271a73803 -n kubeflow-user-example-com
viewer.kubeflow.org "viewer-7c323b7907688f3f3fd81151c62eb58271a73803" deleted
```



```
root@instance-y:/# kubectl apply -f viewer.yaml
viewer.kubeflow.org/viewer-7c323b7907688f3f3fd81151c62eb58271a73803 created
```

Step2: 分析Mnist模型-Tensorboard (4/5) - Optional



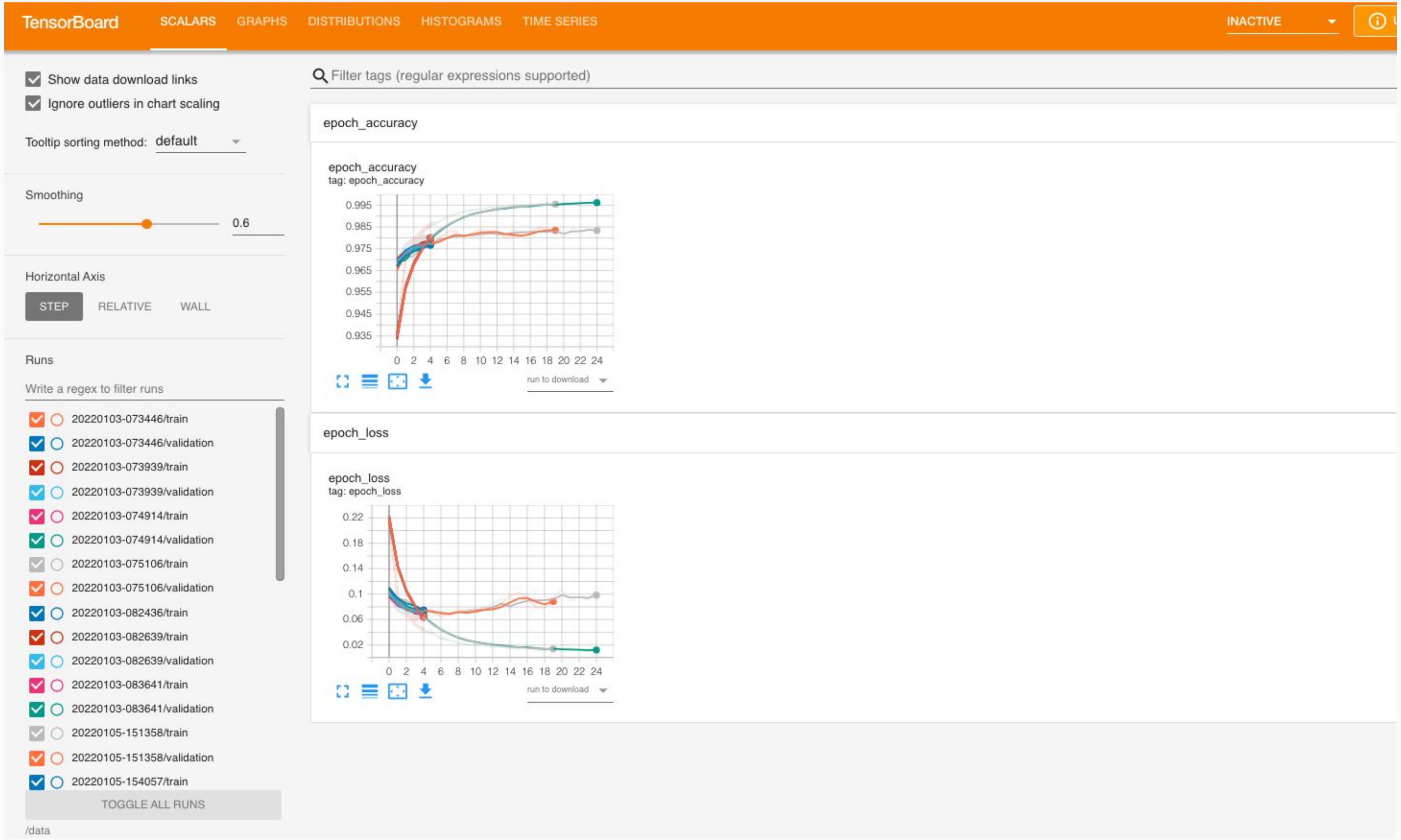
Run of 5.mnist_version_at_2021-12-03T03:26:49.971Z (dc2e1)

mnist-pipeline-gxj5d-4191388434

Tensorboard tensorflow/tensorflow:2.2.2 is running for this output.

需稍待片刻，待伺服器載入 Tensorboard映像檔

Step2: 分析Mnist模型-Tensorboard (5/5) - Optional



☑ Show data download links
☑ Ignore outliers in chart scaling

Tooltip sorting method: default

Smoothing: 0.6

Horizontal Axis: [✓ STEP] RELATIVE WALL

Runs

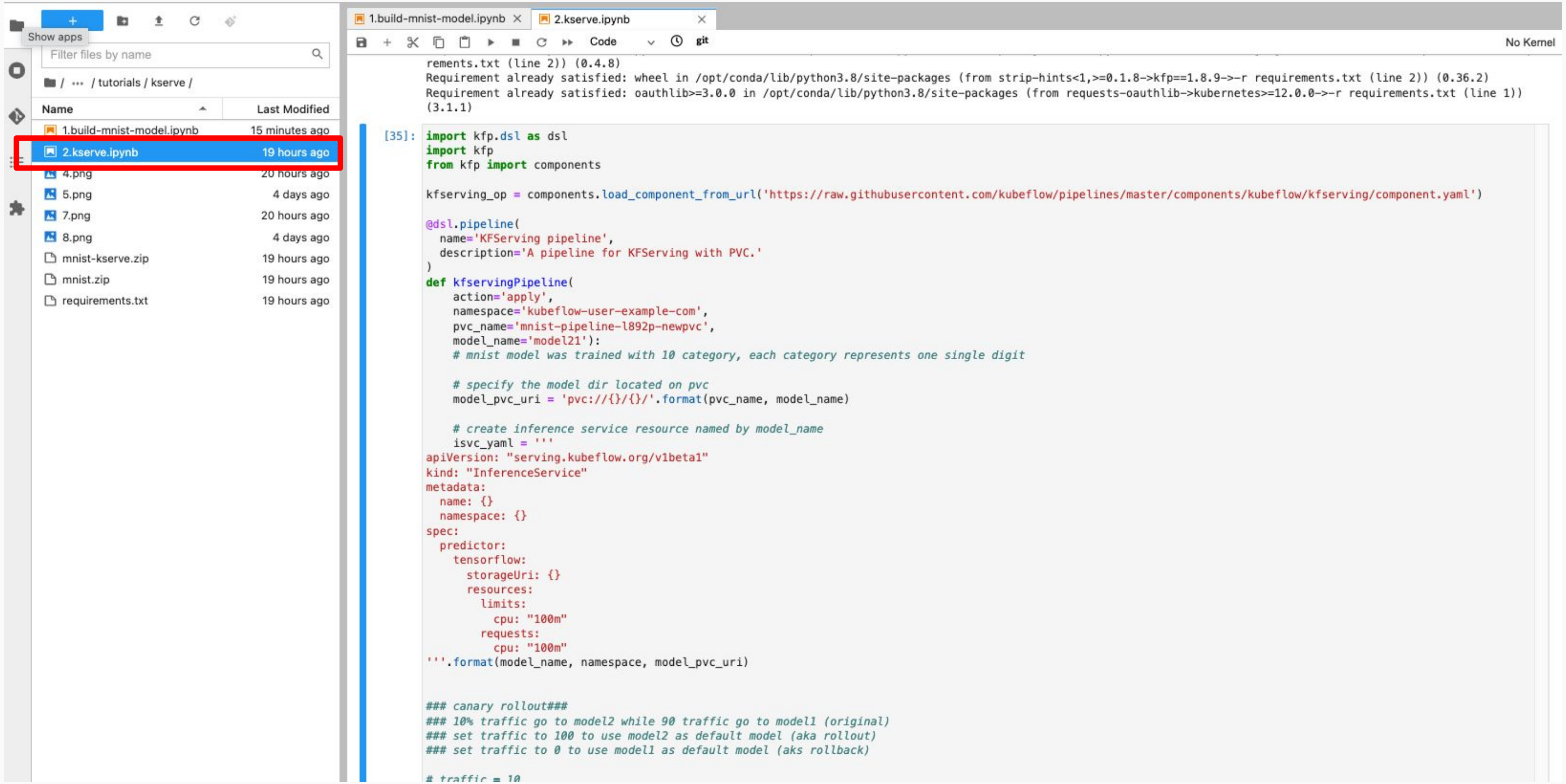
Write a regex to filter runs

- ☑ 20220103-073446/train
- ☑ 20220103-073446/validation
- ☑ 20220103-073939/train
- ☑ 20220103-073939/validation
- ☑ 20220103-074914/train
- ☑ 20220103-074914/validation
- ☑ 20220103-075106/train
- ☑ 20220103-075106/validation
- ☑ 20220103-082436/train
- ☑ 20220103-082639/train
- ☑ 20220103-082639/validation
- ☑ 20220103-083641/train
- ☑ 20220103-083641/validation
- ☑ 20220105-151358/train
- ☑ 20220105-151358/validation
- ☑ 20220105-154057/train

epoch_accuracy
tag: epoch_accuracy

epoch_loss
tag: epoch_loss

Step3: KServe with tensorflow model (1/4)

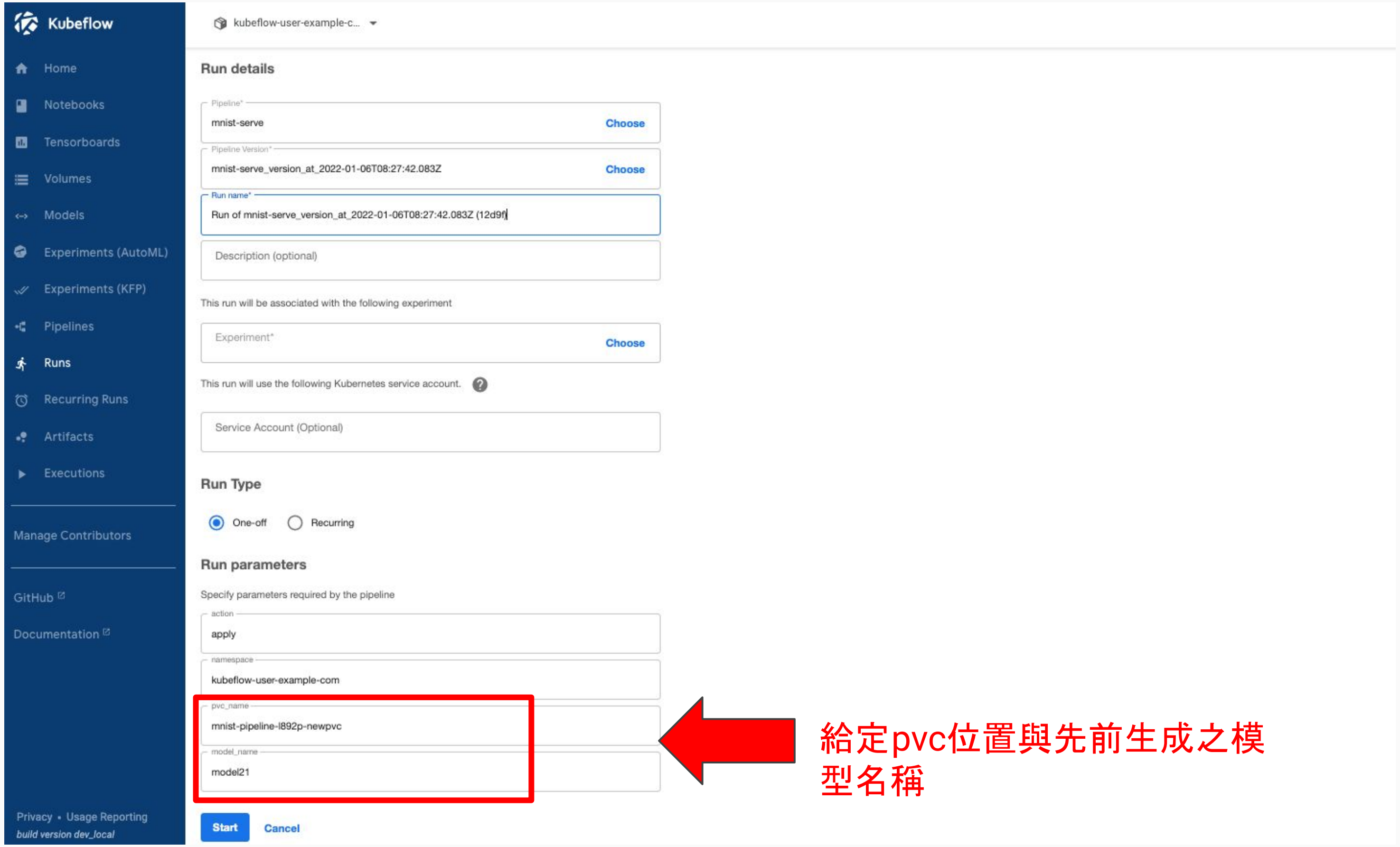


Name	Last Modified
1.build-mnist-model.ipynb	15 minutes ago
2.kserve.ipynb	19 hours ago
4.png	20 hours ago
5.png	4 days ago
7.png	20 hours ago
8.png	4 days ago
mnist-kserve.zip	19 hours ago
mnist.zip	19 hours ago
requirements.txt	19 hours ago

```
rements.txt (line 2)) (0.4.8)
Requirement already satisfied: wheel in /opt/conda/lib/python3.8/site-packages (from strip-hints<1,>=0.1.8->kfp==1.8.9->-r requirements.txt (line 2)) (0.36.2)
Requirement already satisfied: oauthlib>=3.0.0 in /opt/conda/lib/python3.8/site-packages (from requests-oauthlib->kubernetes>=12.0.0->-r requirements.txt (line 1))
(3.1.1)
```

```
[35]: import kfp.dsl as dsl
import kfp
from kfp import components

kfserving_op = components.load_component_from_url('https://raw.githubusercontent.com/kubeflow/pipelines/master/components/kubeflow/kfserving/component.yaml')

@dsl.pipeline(
  name='KFServing pipeline',
  description='A pipeline for KFServing with PVC.'
)
def kfservingPipeline(
    action='apply',
    namespace='kubeflow-user-example-com',
    pvc_name='mnist-pipeline-l892p-newpvc',
    model_name='model21'):
    # mnist model was trained with 10 category, each category represents one single digit

    # specify the model dir located on pvc
    model_pvc_uri = 'pvc://{}/{}/'.format(pvc_name, model_name)

    # create inference service resource named by model_name
    isvc_yaml = '''
apiVersion: "serving.kubeflow.org/v1beta1"
kind: "InferenceService"
metadata:
  name: {}
  namespace: {}
spec:
  predictor:
    tensorflow:
      storageUri: {}
      resources:
        limits:
          cpu: "100m"
        requests:
          cpu: "100m"
'''.format(model_name, namespace, model_pvc_uri)


### canary rollout###
### 10% traffic go to model2 while 90 traffic go to model1 (original)
### set traffic to 100 to use model2 as default model (aka rollout)
### set traffic to 0 to use model1 as default model (aks rollback)

# traffic = 10
```

Step3: KServe with tensorflow model (2/4)



Run details

Pipeline*: mnist-serve (Choose)

Pipeline Version*: mnist-serve_version_at_2022-01-06T08:27:42.083Z (Choose)

Run name*: Run of mnist-serve_version_at_2022-01-06T08:27:42.083Z (12d9f)

Description (optional)

This run will be associated with the following experiment

Experiment* (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

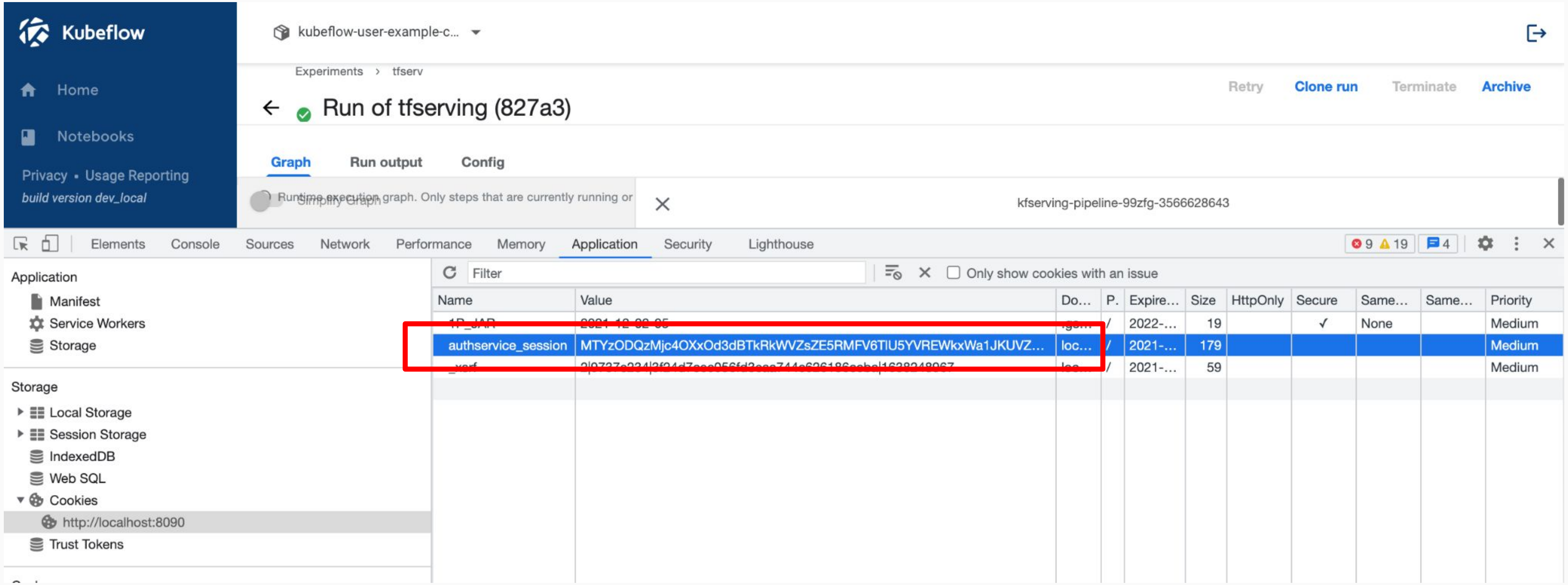
action	apply
namespace	kubeflow-user-example-com
pvc_name	mnist-pipeline-l892p-newpvc
model_name	model21

Start Cancel

給定pvc位置與先前生成之模型名稱

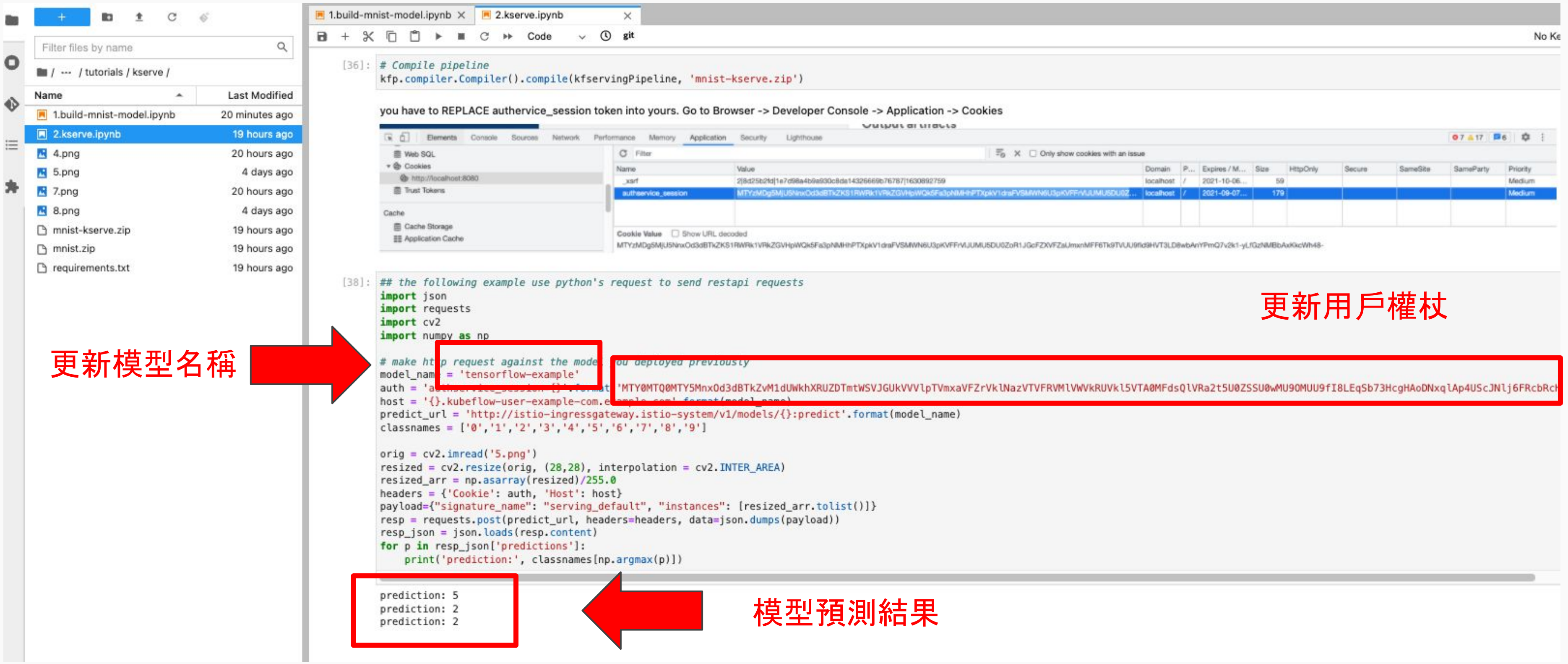
Step3: KServe with tensorflow model (3/4)

獲得用戶權杖:
View -> Developer -> Developer Tools -> Application



Name	Value	Do…	P.	Expire…	Size	HttpOnly	Secure	Same…	Same…	Priority
1P_JAR	2021-12-02-05	.go…	/	2022-…	19		✓	None		Medium
authservice_session	MTYzODQzMjc4OXxOd3dBTkRkWVZsZE5RMFV6TlU5YVREWkxWa1JKUVZ…	loc…	/	2021-…	179					Medium
_xsrf	2|9737e234|3f24d7eee056fd3eaa744c626186eebe|1638248967	loc…	/	2021-…	59					Medium

Step3: KServe with tensorflow model (4/4)



Name	Last Modified
1.build-mnist-model.ipynb	20 minutes ago
2.kserve.ipynb	19 hours ago
4.png	20 hours ago
5.png	4 days ago
7.png	20 hours ago
8.png	4 days ago
mnist-kserve.zip	19 hours ago
mnist.zip	19 hours ago
requirements.txt	19 hours ago

```
[36]: # Compile pipeline
      kfp.compiler.Compiler().compile(kfservingPipeline, 'mnist-kserve.zip')
```

you have to REPLACE authervice_session token into yours. Go to Browser -> Developer Console -> Application -> Cookies

Name	Value	Domain	P...	Expires / M...	Size	HttpOnly	Secure	SameSite	SameParty	Priority
_xsrf	2|8d25b2fd|1e7d98a4b9a930c8da14326669b76787|1630892759	localhost	/	2021-10-06...	59					Medium
authservice_session	MTYzMDg5MjU5NnxOd3dBTkZKS1RWRk1VRkZGVHpWQk5Fa3pNMHhPTXpkV1draFVSMWN6U3pKVFFrVlJUMU5DU0Z...	localhost	/	2021-09-07...	179					Medium

Cookie Value ☐ Show URL decoded
MTYzMDg5MjU5NnxOd3dBTkZKS1RWRk1VRkZGVHpWQk5Fa3pNMHhPTXpkV1draFVSMWN6U3pKVFFrVlJUMU5DU0ZoR1JGcFZXVFZaUmxnMFF6Tk9TVUU9fId9HVT3LD8wbAnYPmQ7v2k1-yLfGzNMBbAxKkcWh48-

```
[38]: ## the following example use python's request to send restapi requests
      import json
      import requests
      import cv2
      import numpy as np

      # make http request against the model you deployed previously
      model_name = 'tensorflow-example'
      auth = 'authservice_session={}'.format('MTY0MTQ0MTY5Mnx0d3dBTkZvM1dUWkhXRUZDTmtWSVJGUkVVVlpTVmxaVFZrVklNazVTVFRVMlVWVkRUVkl5VTA0MFdsQlVRa2t5U0ZSSU0wMU9OMUU9fI8LEqSb73HcgHAoDNxqlAp4UScJNlj6FRcbRcH
      host = '{}.kubeflow-user-example-com.example.com'.format(model_name)
      predict_url = 'http://istio-ingressgateway.istio-system/v1/models/{}:predict'.format(model_name)
      classnames = ['0','1','2','3','4','5','6','7','8','9']

      orig = cv2.imread('5.png')
      resized = cv2.resize(orig, (28,28), interpolation = cv2.INTER_AREA)
      resized_arr = np.asarray(resized)/255.0
      headers = {'Cookie': auth, 'Host': host}
      payload={"signature_name": "serving_default", "instances": [resized_arr.tolist()]}
      resp = requests.post(predict_url, headers=headers, data=json.dumps(payload))
      resp_json = json.loads(resp.content)
      for p in resp_json['predictions']:
          print('prediction:', classnames[np.argmax(p)])
```

```
prediction: 5
prediction: 2
prediction: 2
```

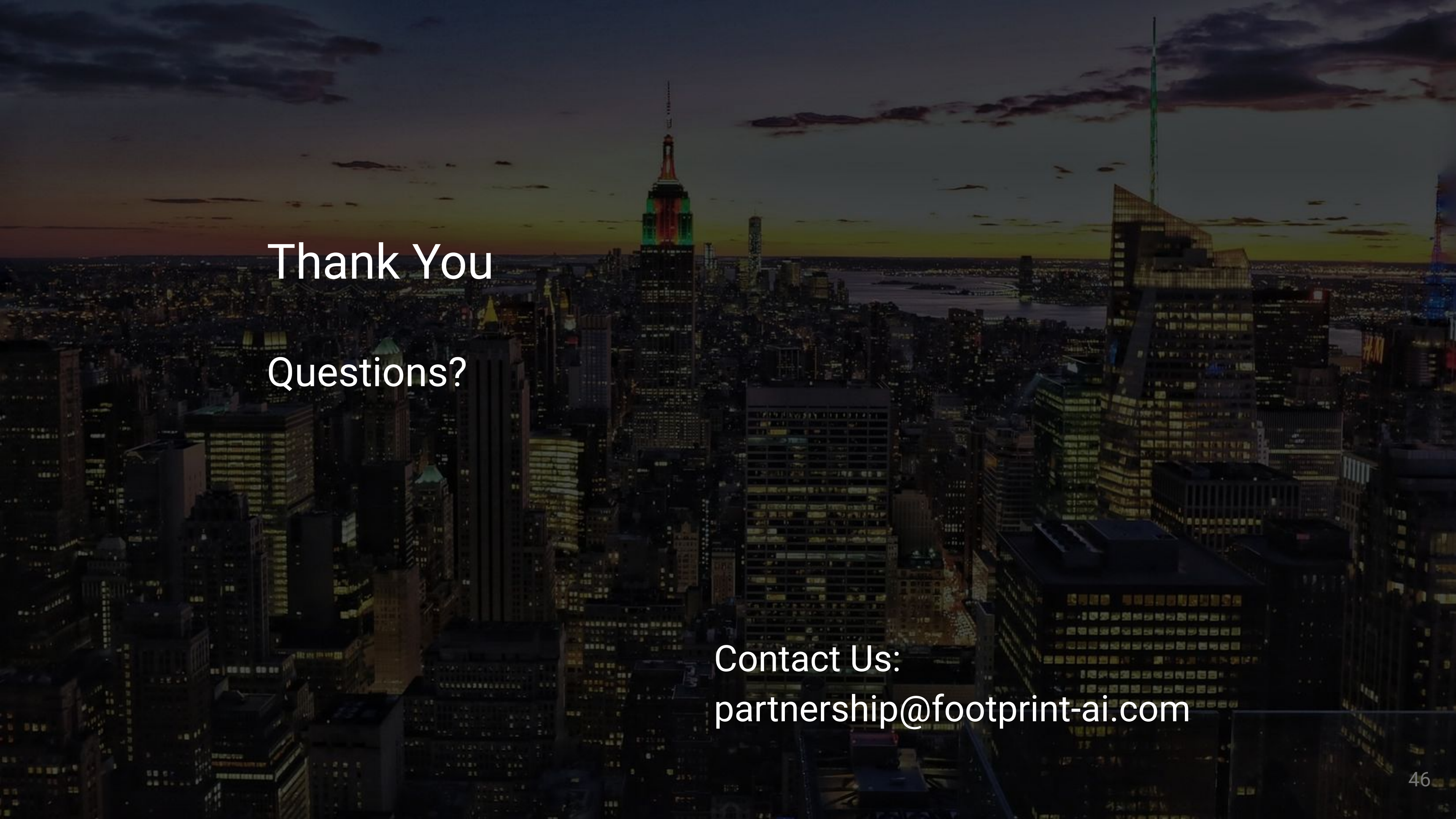
更新模型名稱

更新用戶權杖

模型預測結果

我們剛做了...

- 實現遠端開發，訓練，以及部署之能力
- 流水線模型開發模式
- 分散式訓練
- 模型檢測與分析
- 模型部署



Thank You

Questions?

Contact Us:
partnership@footprint-ai.com

常用資源

- Documentations
  - https://www.kubeflow.org/
- Kubectl cheatsheet
  - https://kubernetes.io/docs/reference/kubectl/cheatsheet/