



雲林科技大學AI雲平台-系統使用說明

Outline

- 淺談虛擬化技術：虛擬機與容器
 - AI雲平台系統示意圖
 - AI雲平台名詞解說
 - AI雲平台實際操作說明
-

淺談虛擬化技術：虛擬機與容器

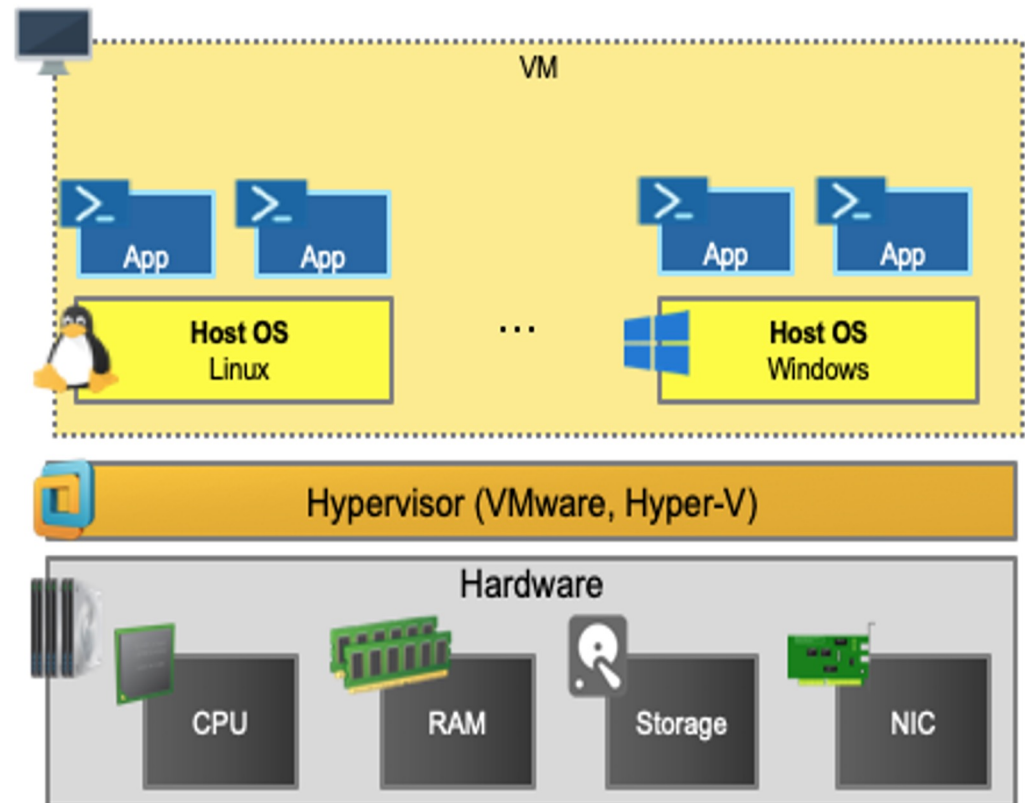
虛擬機(VM)與容器(Container)

作業系統(OS)虛擬化技術，
處理如何分享硬體資源的技術

作業系統虛擬化技術，
通常會有一個Hypervisor的角色，負責管理與
分配硬體資源

雲級別的Hypervisor：AWS、GCP、Azure..

作業系統級別的Hypervisor：VirtualBox、
VMWare、Hyper-V...

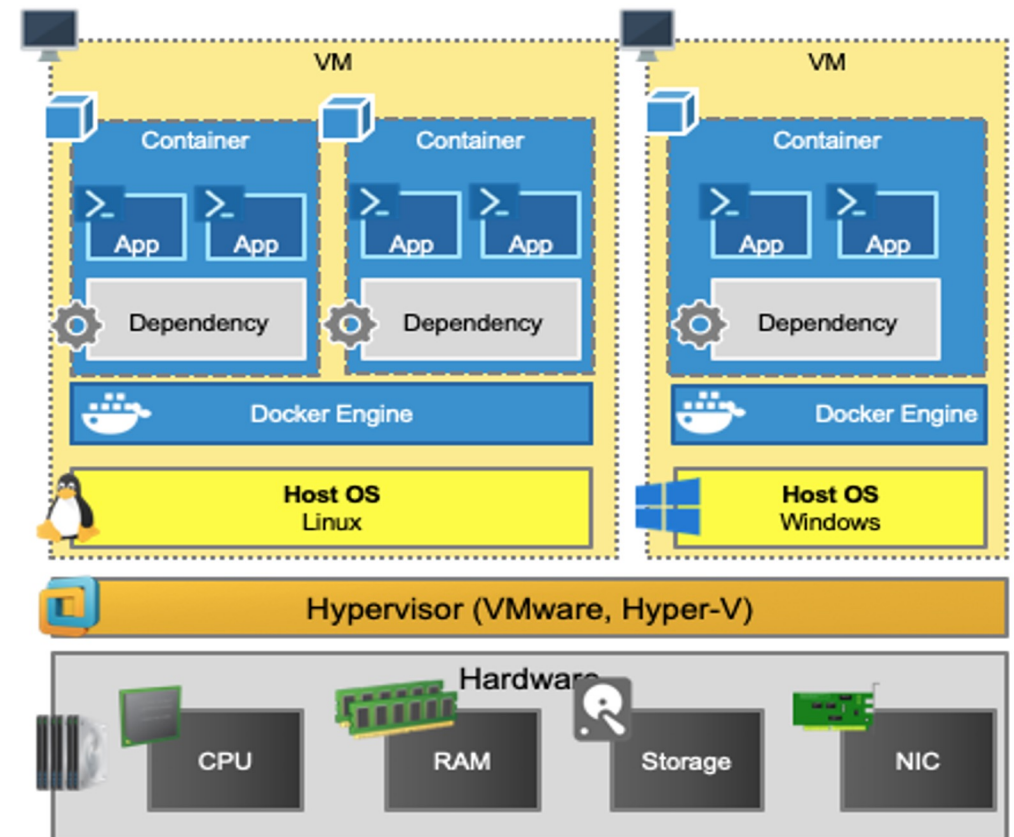


虛擬機(VM)與容器(Container)

容器(Container)虛擬化技術，
處理軟體系統的環境建置

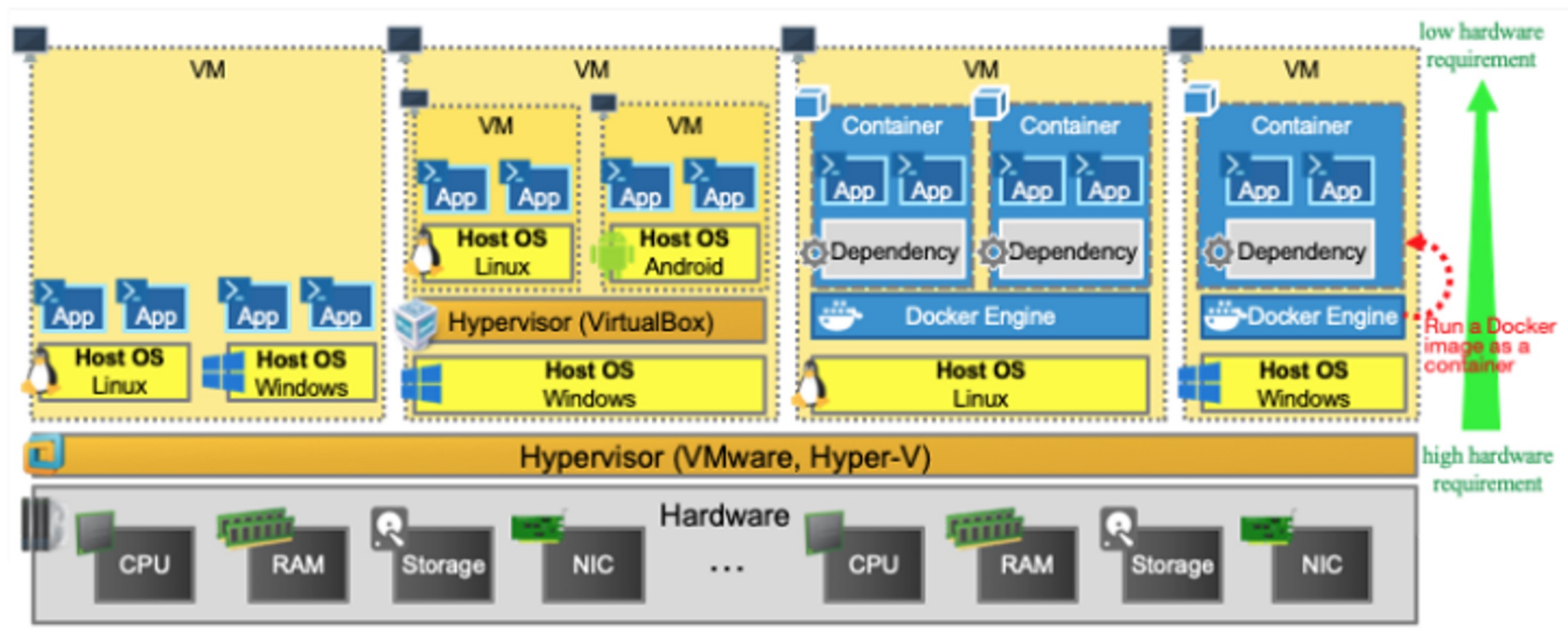
讓整個軟體系統的執行環境可以搬遷，
「與其為每一個軟體系統建置執行環境，倒不如讓整個軟體系統的執行環境是可以搬遷與重複利用的」

容器(Container)就是一種可以搬遷與重複利用的執行環境



虛擬機(VM)與容器(Container)

容器化技術並不是用來取代傳統作業系統虛擬化技術

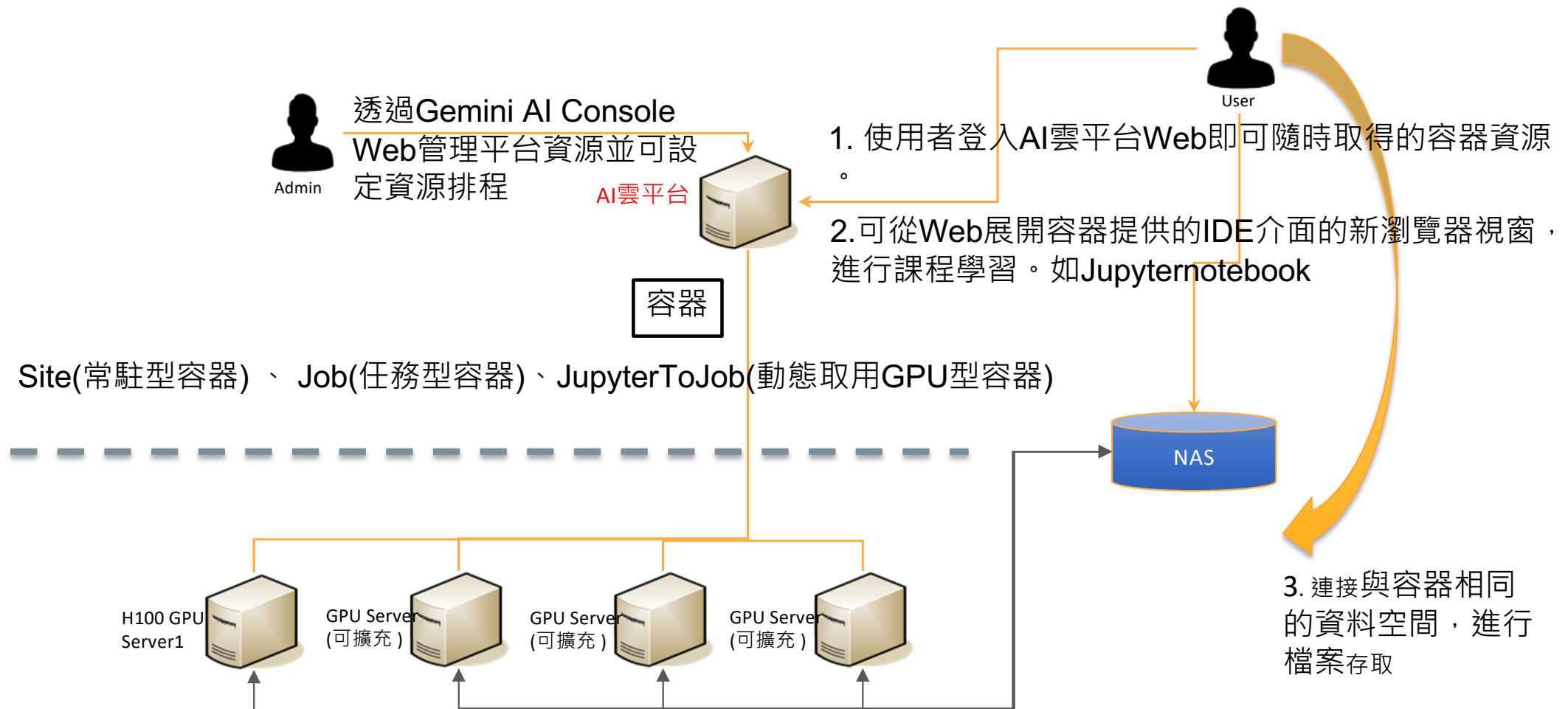


虛擬機與容器-兩種虛擬化技術的差異

- 以代表性服務為例：**Docker**是以**App**為單位的虛擬化技術；**VMWare**是以作業系統為單位的虛擬化技術。
- 容器虛擬化技術不是用來取代作業系統虛擬化技術的新技術
- 容器並不是**VM**
- 容器虛擬化技術以共享**Host OS Kernel**的方式來執行**App**
- 容器(**container**)是執行環境的實例(**instance**)；**VM**是作業系統的實例(**instance**)
- 容器化技術更面向**DevOps**的需求
- 硬體資源的分配
- 應用程式的獨立性

AI雲平台-系統示意圖

AI雲平台-系統示意圖

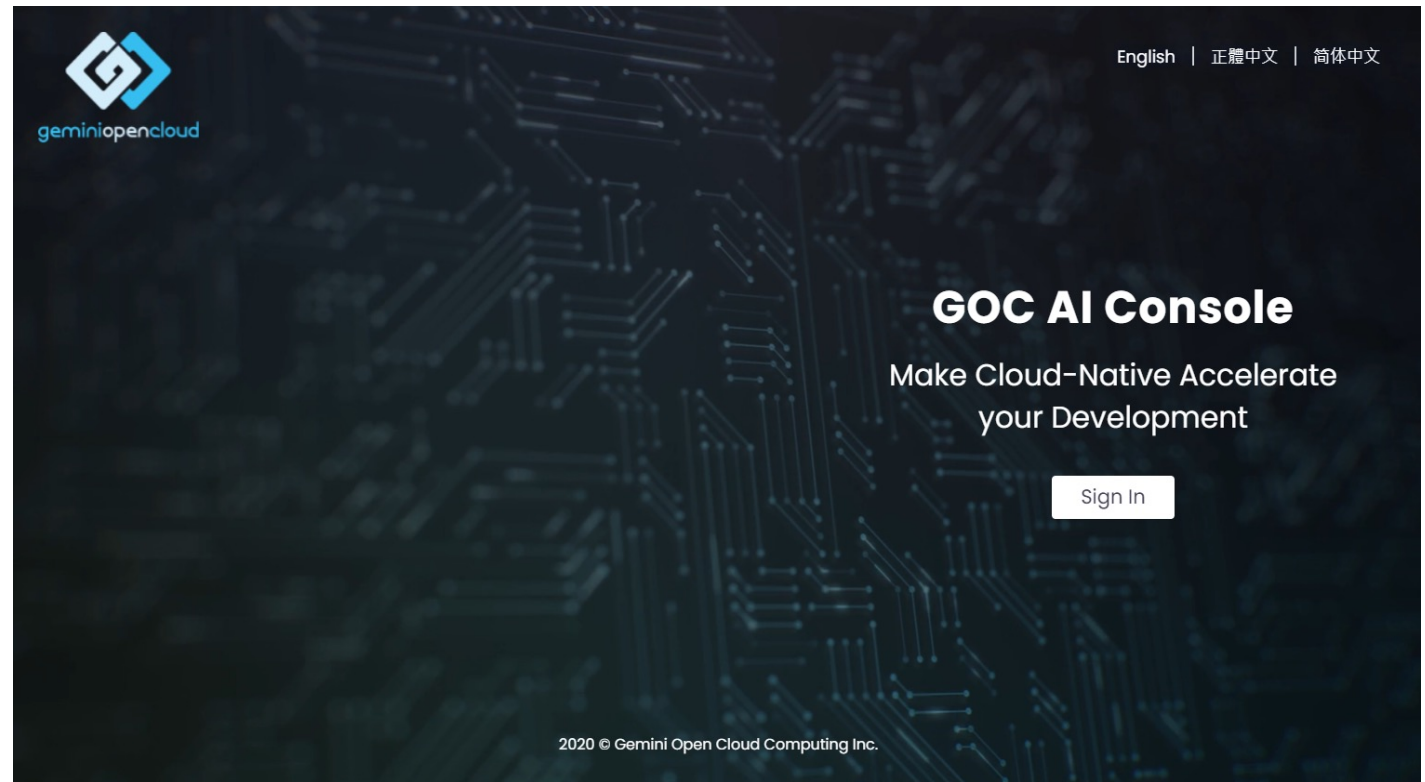


AI雲平台名詞解說

AI雲平台-Portal

提供管理員資源管理

提供使用者容器資源取得與使用



AI雲平台-Application(常駐型容器)

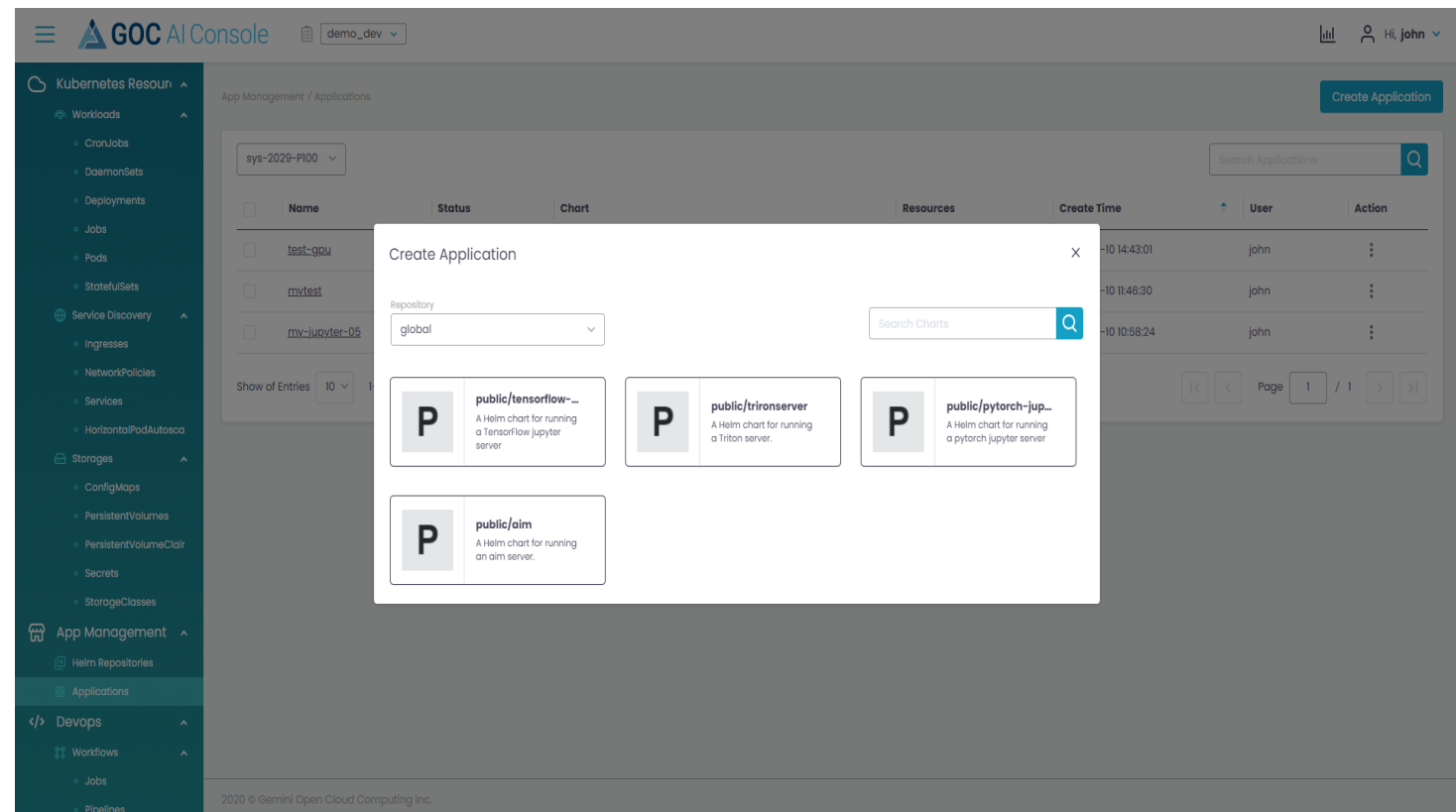
可依需求自行建立容器

依容器服務特性提供連線資訊

提供網頁式的編程開發工具介面，方便使用者快速取得工具

網頁式的編程開發工具，方便程式開發人員即時運行程式，即時除錯

可將程式碼儲存至個人空間，利用Job取得更多運算資源執行程式



AI雲平台-JupyterToJob(動態取用GPU型容器)

以常駐型容器為基礎

建立時無須占用GPU資源

程式執行時，可選擇執行環境
資源規模，動態取用GPU

AI雲平台-Job(任務型容器)

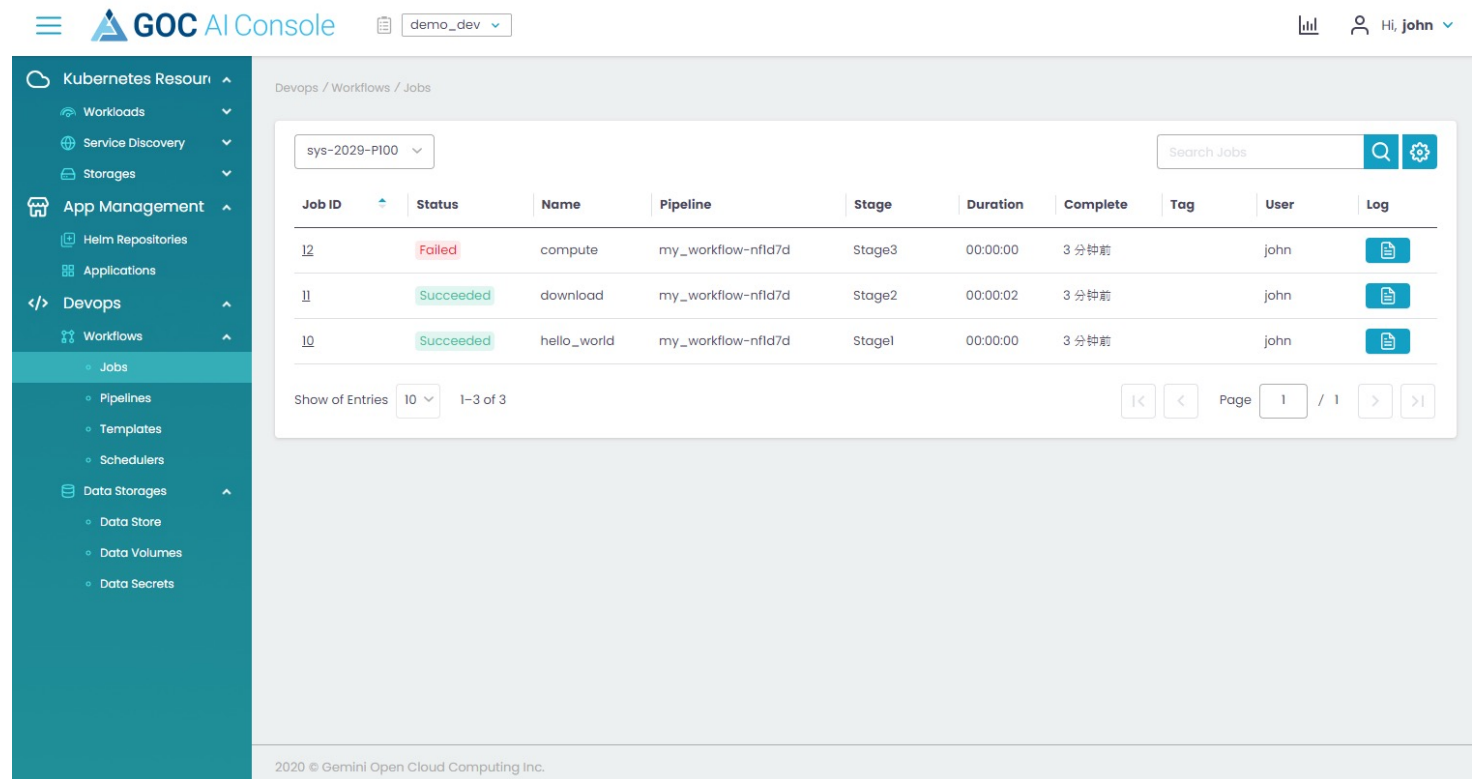
運用GPU共享區資源

透過linux command方式執行程式




多個Command可以分號『;』分隔

提供多種GPU數量資源格式運行容器

任務排程方式執行訓練程式



The screenshot displays the GOC AI Console interface. On the left is a sidebar menu with categories like Kubernetes Resources, App Management, Devops, and Data Storages. The 'Jobs' option under 'Devops' is selected. The main panel shows a table of jobs with columns for Job ID, Status, Name, Pipeline, Stage, Duration, Complete, Tag, User, and Log. Three jobs are listed: Job 12 (Failed), Job 11 (Succeeded), and Job 10 (Succeeded). A search bar and pagination controls are also visible.

Job ID	Status	Name	Pipeline	Stage	Duration	Complete	Tag	User	Log
12	Failed	compute	my_workflow-nfld7d	Stage3	00:00:00	3 分钟前		john	
11	Succeeded	download	my_workflow-nfld7d	Stage2	00:00:02	3 分钟前		john	
10	Succeeded	hello_world	my_workflow-nfld7d	Stage1	00:00:00	3 分钟前		john	

Footer: 2020 © Gemini Open Cloud Computing Inc.

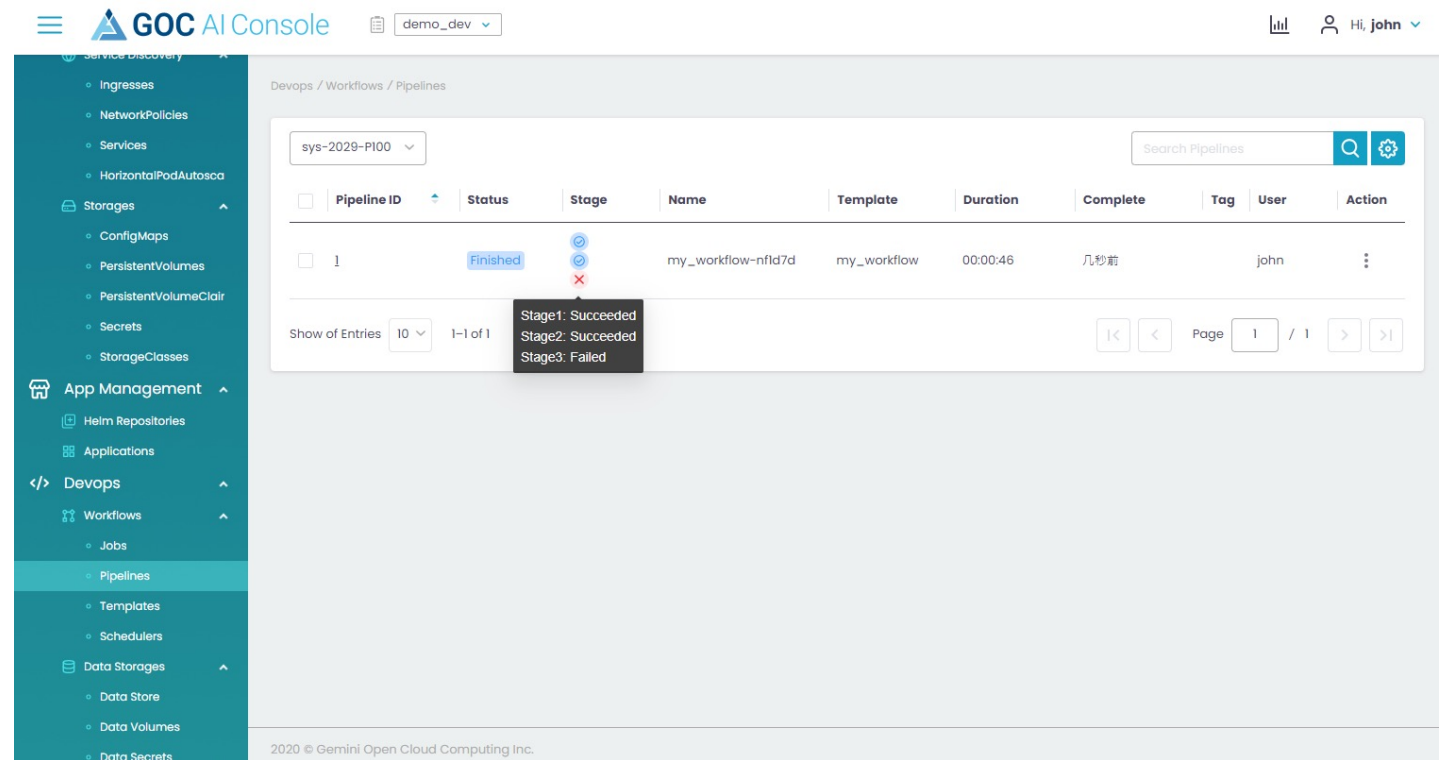
Job延伸-Pipeline(指令多工)

可將多個AI運算工作分成不同階段去執行，當一個階段完成後才會換下一階段去執行，等於將AI運算處理流程化

簡單概念：

單一Job是將單一指令下去運行
Pipeline則是將多個指令分為不同階段下去運行

使用情境：若一個AI訓練透過Tensorflow完成，而我需要用Pytorch或其他工具去驗證結果時，便可使用Pipeline完成。

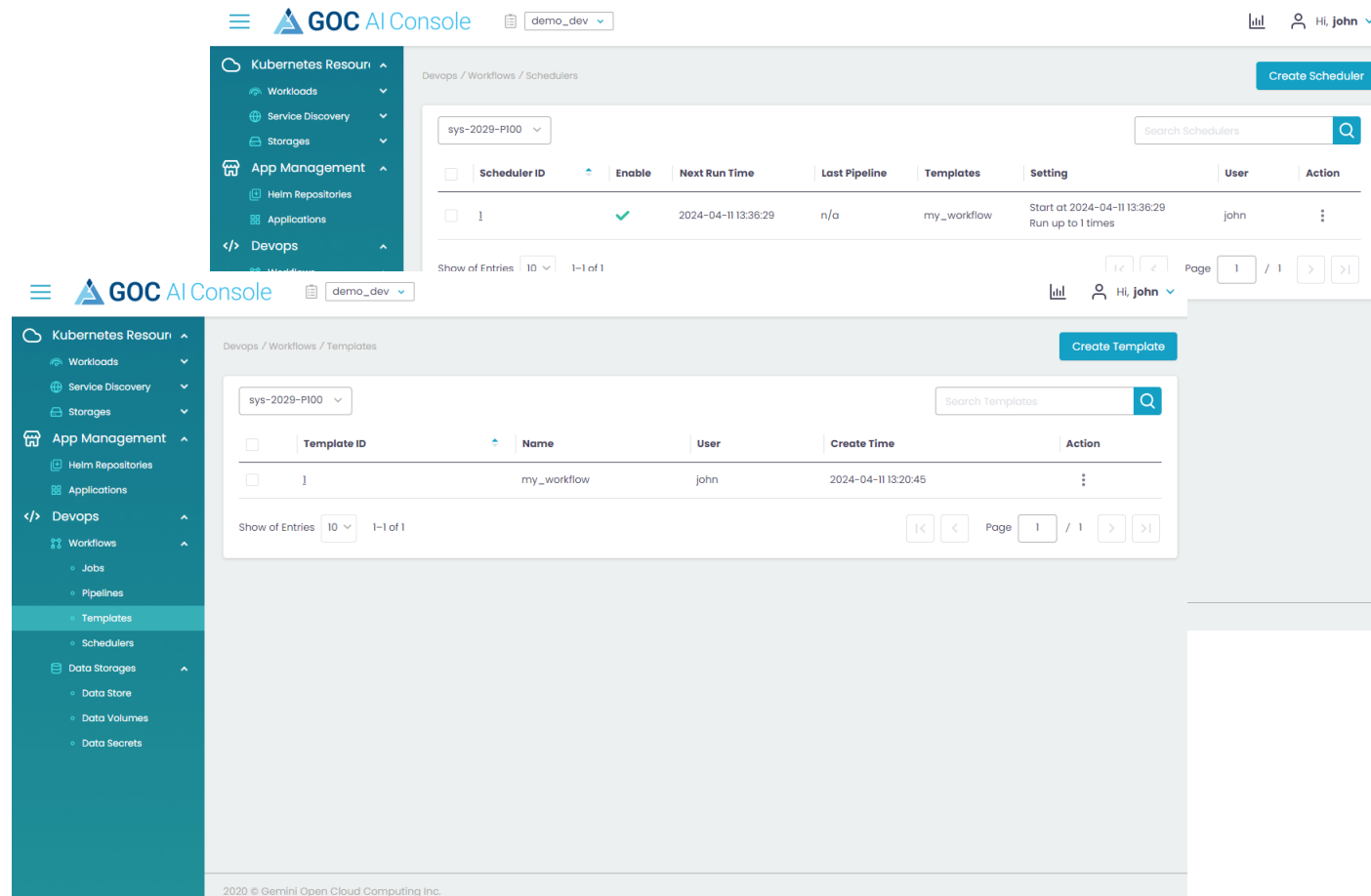


The screenshot displays the GOC AI Console interface. On the left is a sidebar menu with categories like Service Discovery, Storages, App Management, Devops, and Data Storages. The 'Devops' section is expanded, showing 'Workflows' and 'Pipelines'. The main panel shows the 'Pipelines' view for a specific system (sys-2029-P100). It features a table with columns: Pipeline ID, Status, Stage, Name, Template, Duration, Complete, Tag, User, and Action. A single pipeline entry is shown with ID '1', status 'Finished', and a duration of 00:00:46. A tooltip for the 'Stage' column indicates: Stage1: Succeeded, Stage2: Succeeded, Stage3: Failed. The bottom of the interface shows a footer with the copyright notice: 2020 © Gemini Open Cloud Computing Inc.

Pipeline延伸-Template & Scheduler

Template：可將Pipeline儲存成一個固定範本，當有新的資料需要用同樣的運算流程去跑時，只要確保程式碼檔案與資料路徑都正確，就能使用此功能直接運行。

Scheduler：若資料是固定時間會自動放入，則可將現有的**Template**設定排程，讓AI雲平台能夠在指定的時間自動執行運算。



The image displays two screenshots of the GOC AI Console interface, illustrating the management of Pipelines, Templates, and Schedulers.

Top Screenshot (Schedulers): The interface shows the 'Devops / Workflows / Schedulers' section. A table lists existing schedulers. The first entry is '1', which is enabled (indicated by a green checkmark), with a next run time of '2024-04-11 13:36:29'. It is associated with the 'my_workflow' template and is set to run up to 1 time. The user 'john' is listed as the owner.

Scheduler ID	Enable	Next Run Time	Last Pipeline	Templates	Setting	User	Action
1	✓	2024-04-11 13:36:29	n/a	my_workflow	Start at 2024-04-11 13:36:29 Run up to 1 times	john	⋮

Bottom Screenshot (Templates): The interface shows the 'Devops / Workflows / Templates' section. A table lists existing templates. The first entry is '1', named 'my_workflow', owned by 'john', and created at '2024-04-11 13:20:45'.

Template ID	Name	User	Create Time	Action
1	my_workflow	john	2024-04-11 13:20:45	⋮

Hands On

AI雲平台實際操作說明

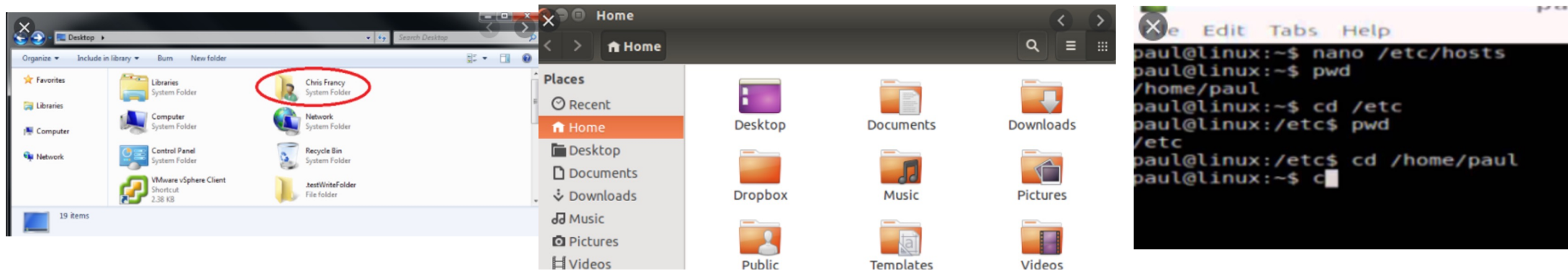
- 1.基本概念：家目錄(home folder)
 - 2.Job, Site, User如何存取家目錄
 - 3.AI雲平台提供的網頁式編程工具
 - 4.如何透過AI雲平台建立Application
 - 5.如何透過AI雲平台建立Job
 - 6.如何使用沒有GPU的Jupyternotebook環境，動態取用GPU來執行運算
-

在AI Console專案內，個人資料夾如何使用

1. admin或project admin需要為kubernetes cluster建立StorageClass，讓使用者得知Storage資訊(server, share)
2. user需要使用該cluster，建立第一個PersistentVolumeClaim，以便取得Storage資料夾路徑 (example: "volumeName":"pvc-dc5aedef7-14b5-4bdb-ad90-91ac988340ef")
3. user需要使用該cluster，在**Devops**建立DataStore與Data Volumes，以便使用相同路徑存取相同資料。
 1. Add Data Store:
 1. NFS Server IP輸入步驟一取得的server ip。
 2. NFS Path輸入share path以及步驟二取得的pvc-xxx合併為同一個NFS path，ex: /nfs/path/pvc-dc5aedef7-14b5-4bdb-ad90-91ac988340ef。
 3. 勾選Using existing NFS data for volume creation
 2. Create Data Volume
 1. Data Volume Setting的地方，Default Container Mount Directory請輸入/home
 2. 勾選Default Data Volume

基本概念：家目錄

- 這是AI雲平台提供給使用者的個人專屬空間。
- 類似Google drive、OneDrive的雲端空間
- AI雲平台的Application, Job透過家目錄，將資料儲存位置串連起來
(類似網路磁碟機的概念)



Job, Site, User如何存取家目錄：1) 透過FTP or SFTP參考管理員提供之方式

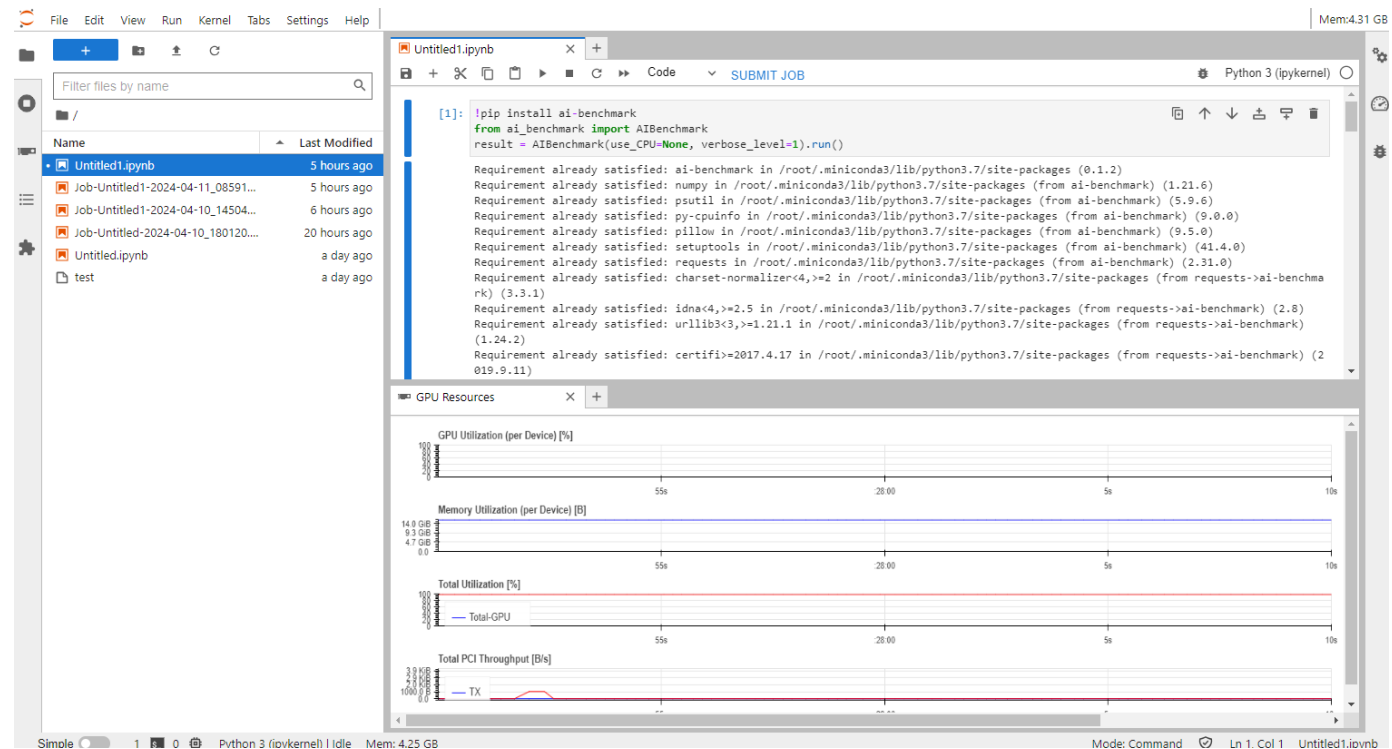
儲存空間伺服器IP:
請參考App Detail

可透過winscp
或
FileZilla等工具連線

<input type="checkbox"/>	torch-site	Deployed	global/public/pytorch-jupyter:3.0.2	2	2024-09-11 15:48:44	test	⋮
Show of Entries 10 1-1 of 1							
Resources Values Detail Helm Status							
Name		torch-site		Endpoints		torch-site-pytorch-jupyter	
Status Reason						140.125.248.15:32297 (web)	
Project		id	3			140.125.248.15:30492 (tensorboard)	
		name	test				

Job, Site, User如何存取家目錄：2) 透過Application(常駐型容器)

Application(常駐型容器)裡，
JupyterLab檔案總管預設路徑
(/home)就是使用者資料夾



The screenshot displays the JupyterLab interface. On the left, the file manager shows a directory structure with files like 'Untitled1.ipynb' and 'test'. The main area shows a code editor with a Python script that installs and runs 'ai-benchmark'. The output of the script is visible, showing various requirements already satisfied. Below the code editor, there are four graphs: 'GPU Utilization (per Device) [%]', 'Memory Utilization (per Device) [B]', 'Total Utilization [%]', and 'Total PCI Throughput [B/s]'. The status bar at the bottom indicates 'Python 3 (ipykernel) | Idle | Mem: 4.25 GB'.

AI雲平台提供的網頁式編程工具

- Jupyter notebook + tensorflow
- Jupyter notebook + pytorch



如何透過AI雲平台建立Application(常駐型容器)

建立Application (常駐型容器)可以快速的取得網頁式的編程工具

- 1.使用帳號登入Portal
- 2.左側功能清單App Management -> Applications，點擊Create Application
- 3.選擇工具類型->輸入資訊->建立
- 4.建立完成後點選該項目名稱->下方顯示的Detail項目
- 5.Endpoints項目清單中，複製(web)該項的IP:Port
- 6.開啟新的瀏覽器，貼上IP:Port
- 6.登入編程工具網頁服務(例如：jupyterlab)

如何透過AI雲平台建立Job(任務型容器)

建立Job(任務型容器)可以取得更多GPU的運算資源來執行程式

使用帳號登入 Portal

左側功能清單Devops -> Workflows -> Templates , 點擊Create Template

建立任務Template

輸入command

啟動排程(可關閉後再開)

Log查詢執行過程

如何使用沒有GPU的Jupyternotebook環境，動態取用GPU來執行運算



Run notebook on AIConsole job

Select one type of gpu to use

Kubeshare gpu limit

50

Allocate nvidia gpu

Nvidia GPU

0

CANCEL SEND

```
[21]: import torch
import torchvision
import torchvision.transforms as transforms

[22]: transform
[transfo
transform

[23]: trainset = datasets.MNIST(root='./data', train=True,
download=True, transform=transforms.Compose([
transform.ToTensor(),
transform.Normalize([0.1307], [0.3081])]),
trainloader = data.DataLoader(trainset, batch_size=64,
shuffle=True, num_workers=0)
testset = datasets.MNIST(root='./data', train=False,
download=True, transform=transforms.Compose([
transform.ToTensor(),
transform.Normalize([0.1307], [0.3081])]),
testloader = data.DataLoader(testset, batch_size=64,
shuffle=False, num_workers=0)

[24]: import torch.nn.functional as F
import torch.nn as nn
class SimpleCNN(nn.Module):
def __init__(self):
super(SimpleCNN, self).__init__()
self.conv1 = nn.Conv2d(1, 16, 5, 5)
self.pool = nn.MaxPool2d(2, 2)
self.conv2 = nn.Conv2d(16, 16, 5, 5)
self.fc1 = nn.Linear(16 * 5 * 5, 120)
```

Q&A

