Guidance

Access to Azure Lab

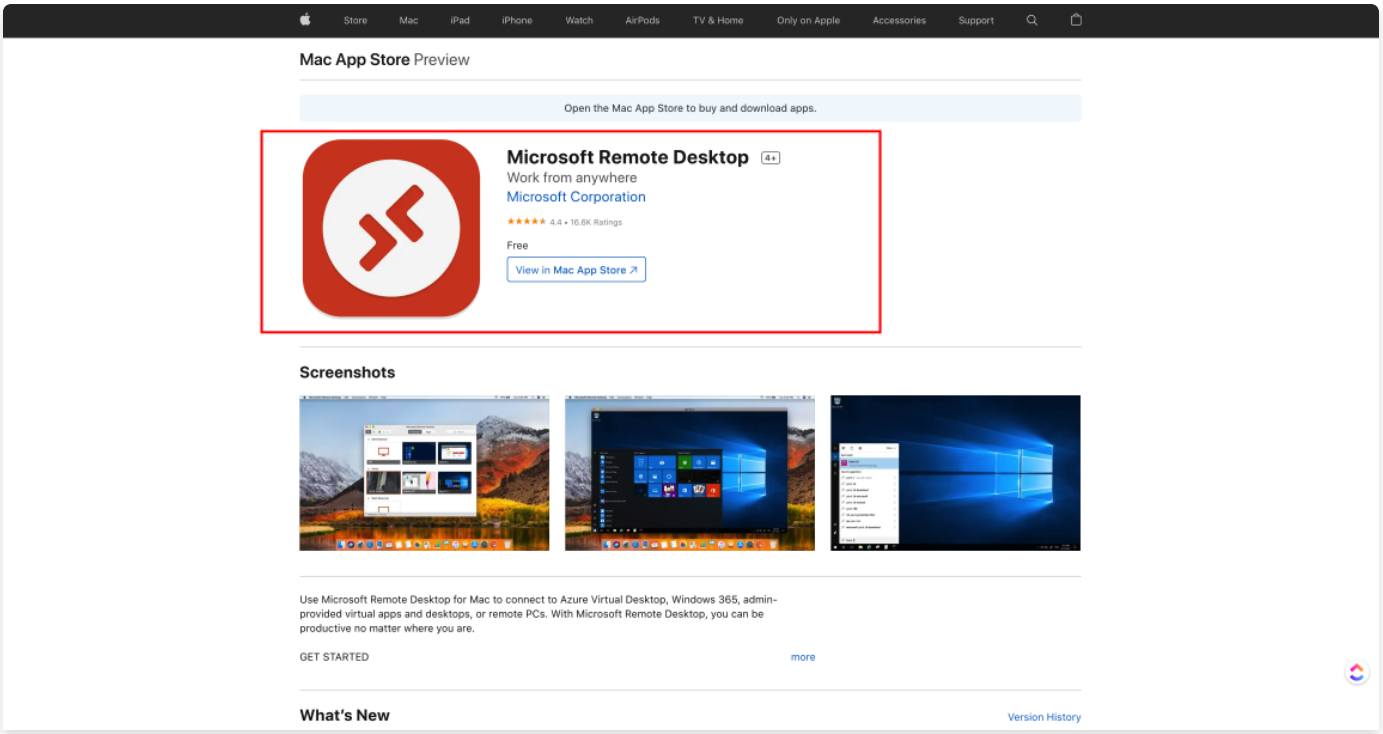
You can access to Azure Lab both in campus or out of campus.

Every student has admin privileges for Windows and Ubuntu, which means you can install any software you need.

Download Remote Desktop App to connect your Azure VM

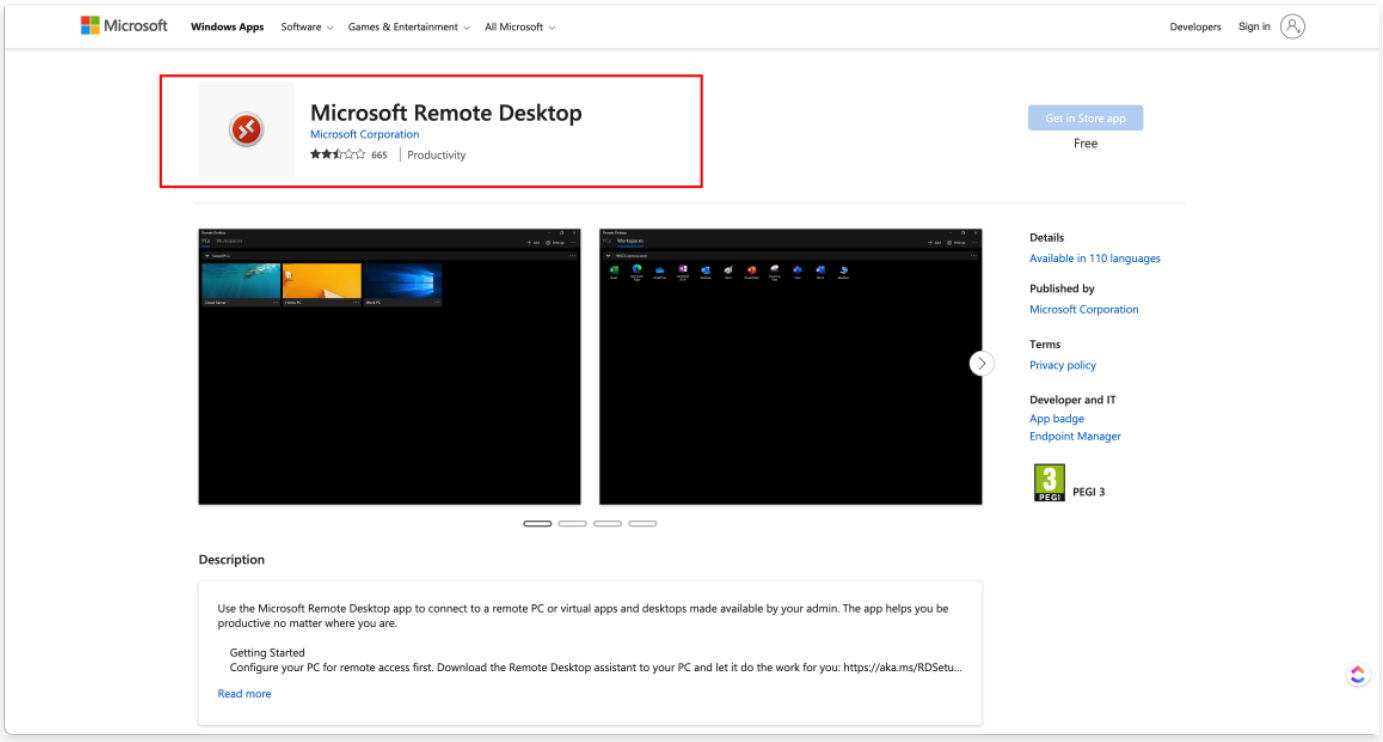
Getting it for Mac

App Store: Microsoft Remote Desktop



Getting it for Windows

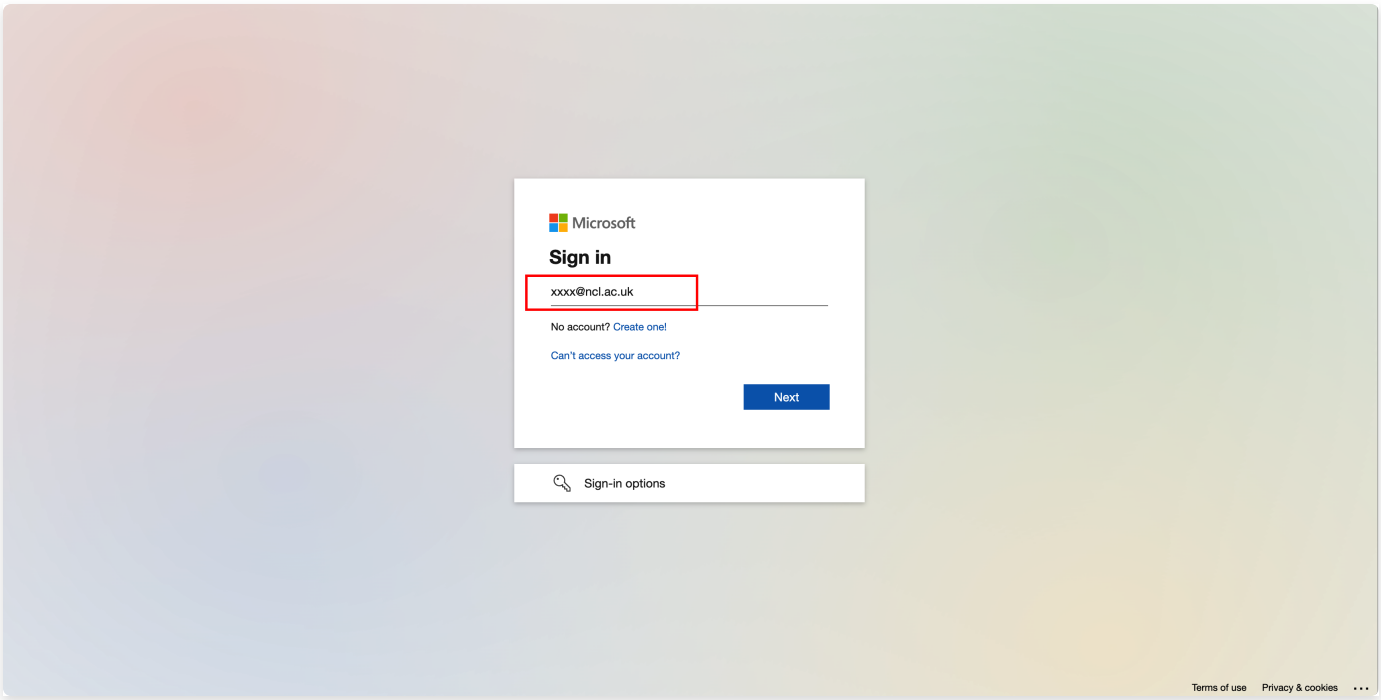
Microsoft Store: Get $Microsoft Remote Desktop from the Microsoft Store



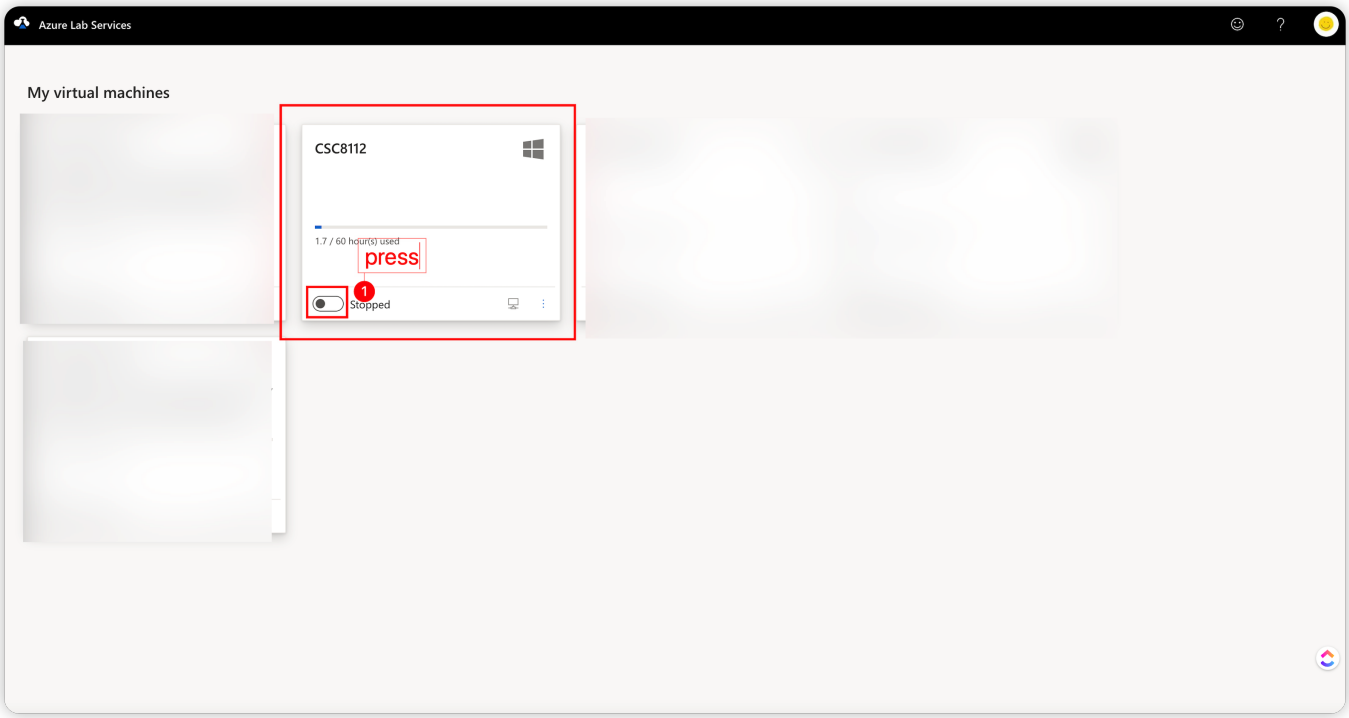
Start Azure VM

1. Go to: Azure Lab Services

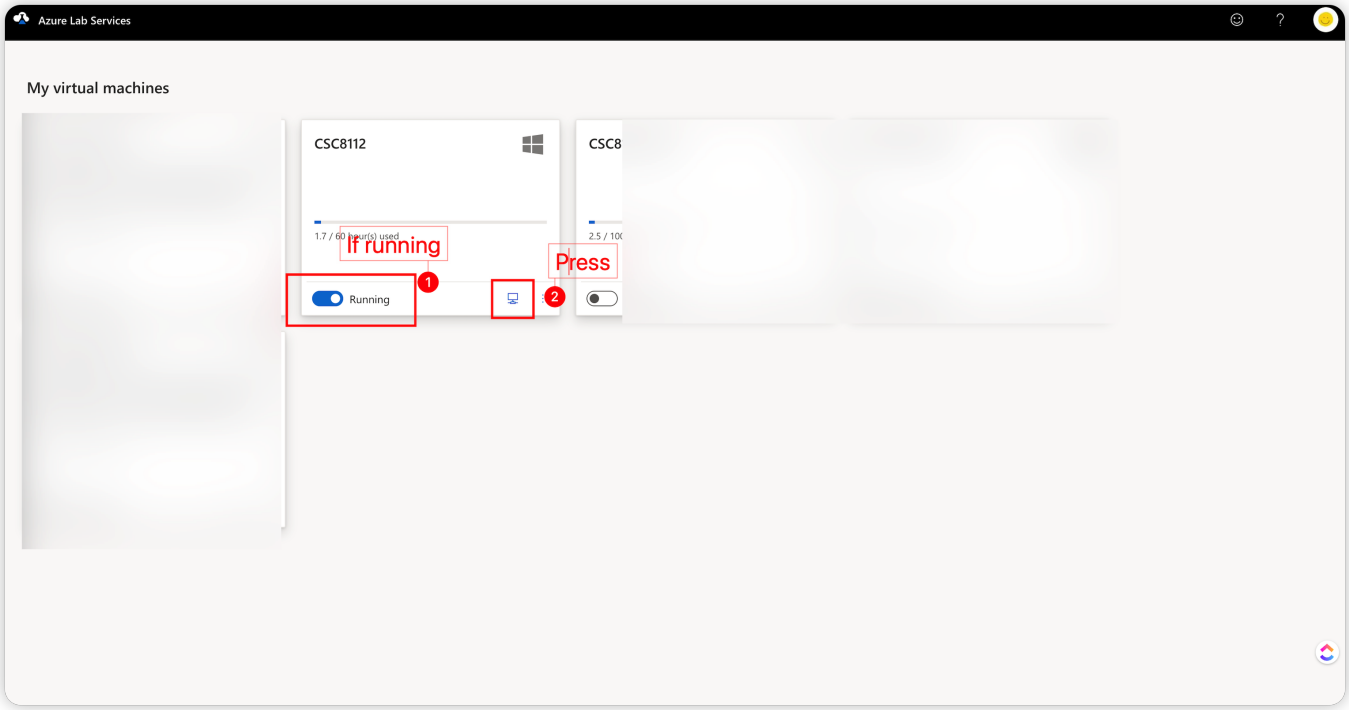
2. Login with your University account

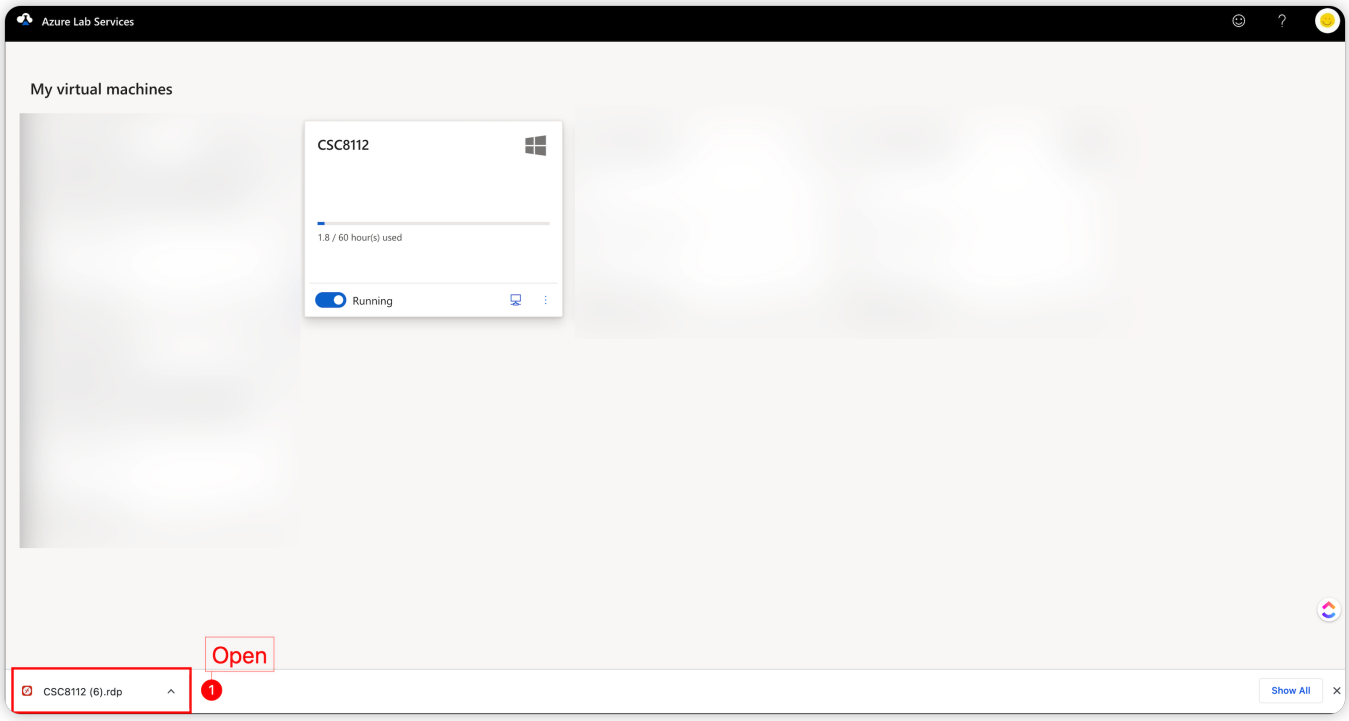


3. Start your VM



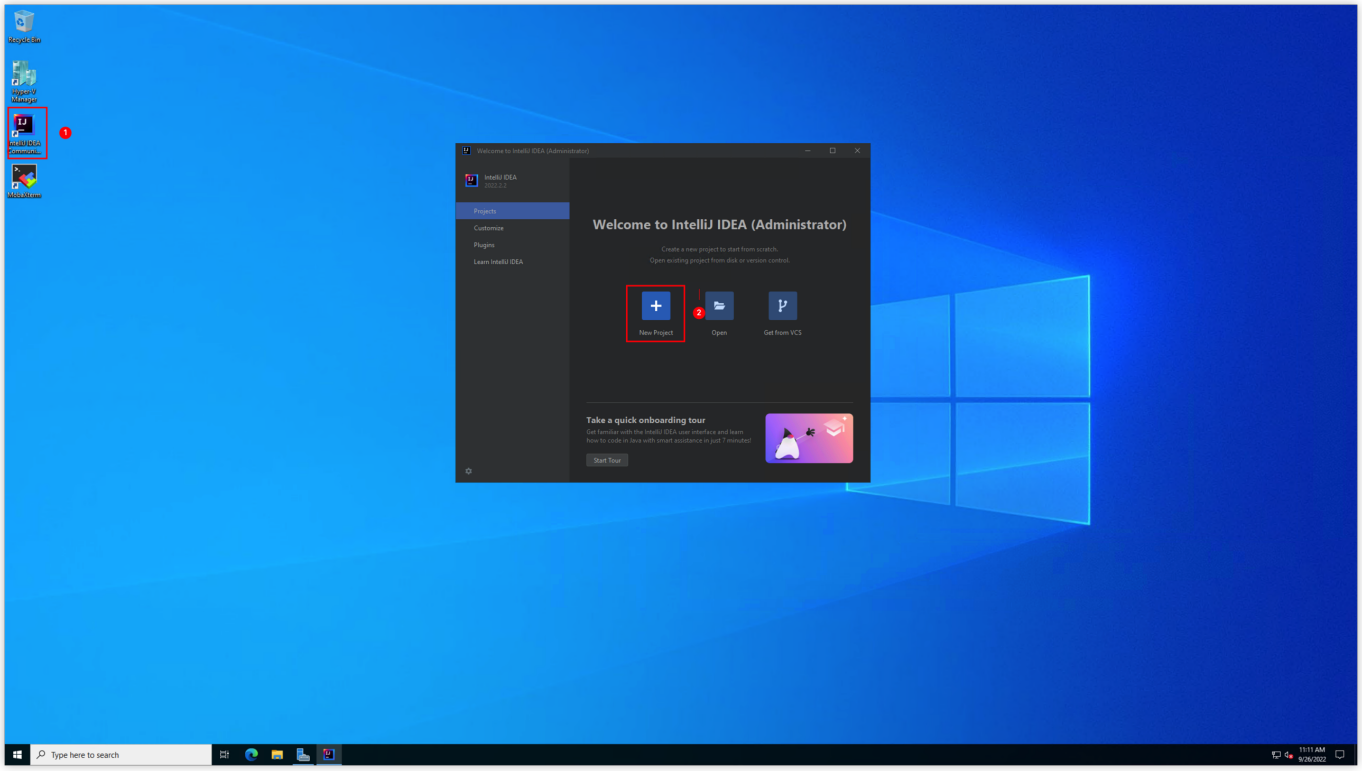
4. Connect your VM with password (Please check it in Teams channel)

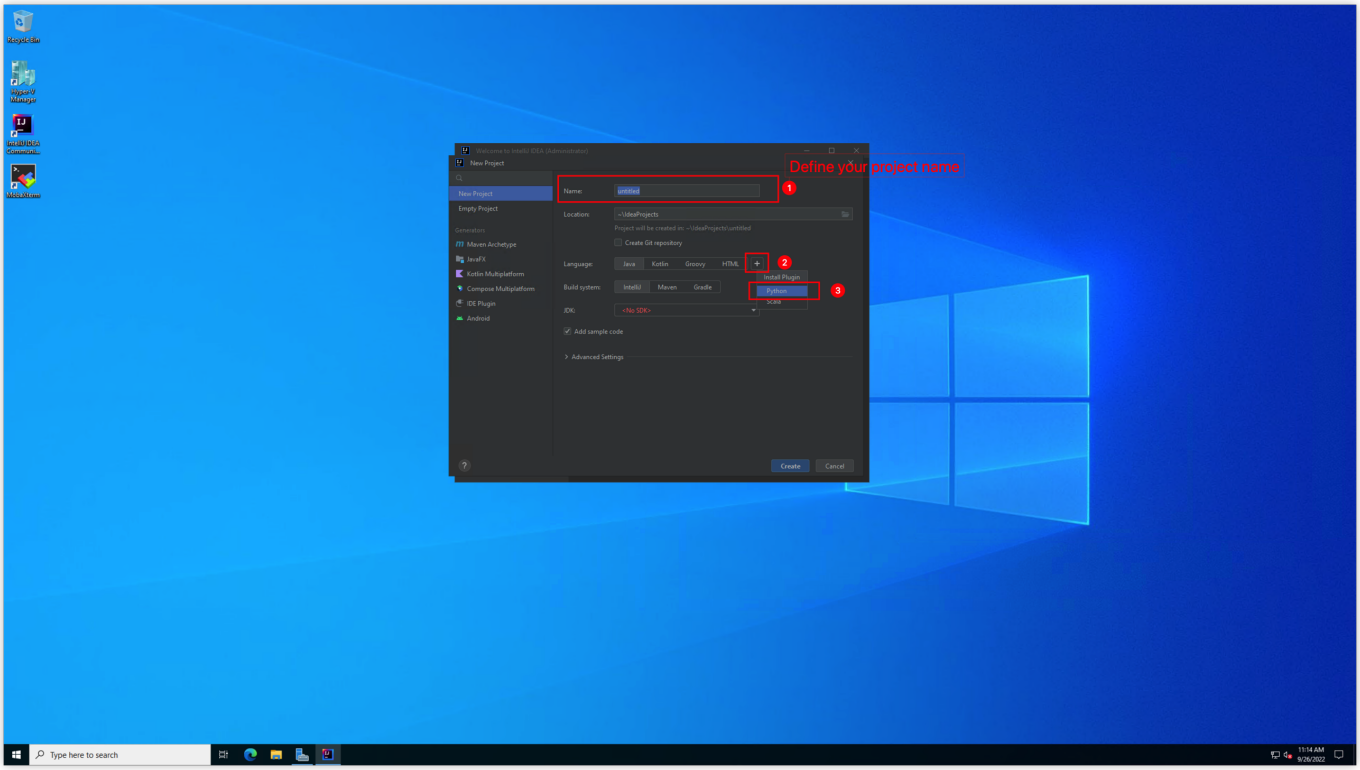


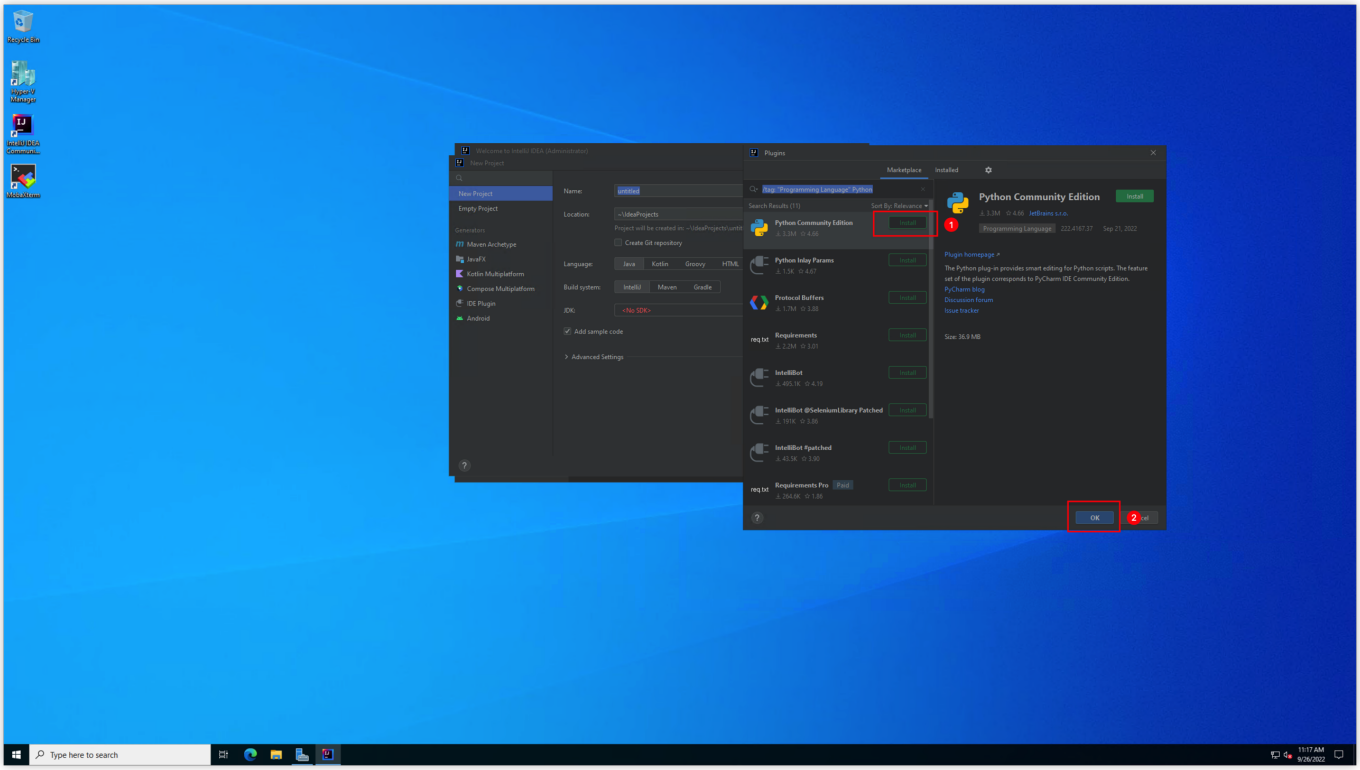


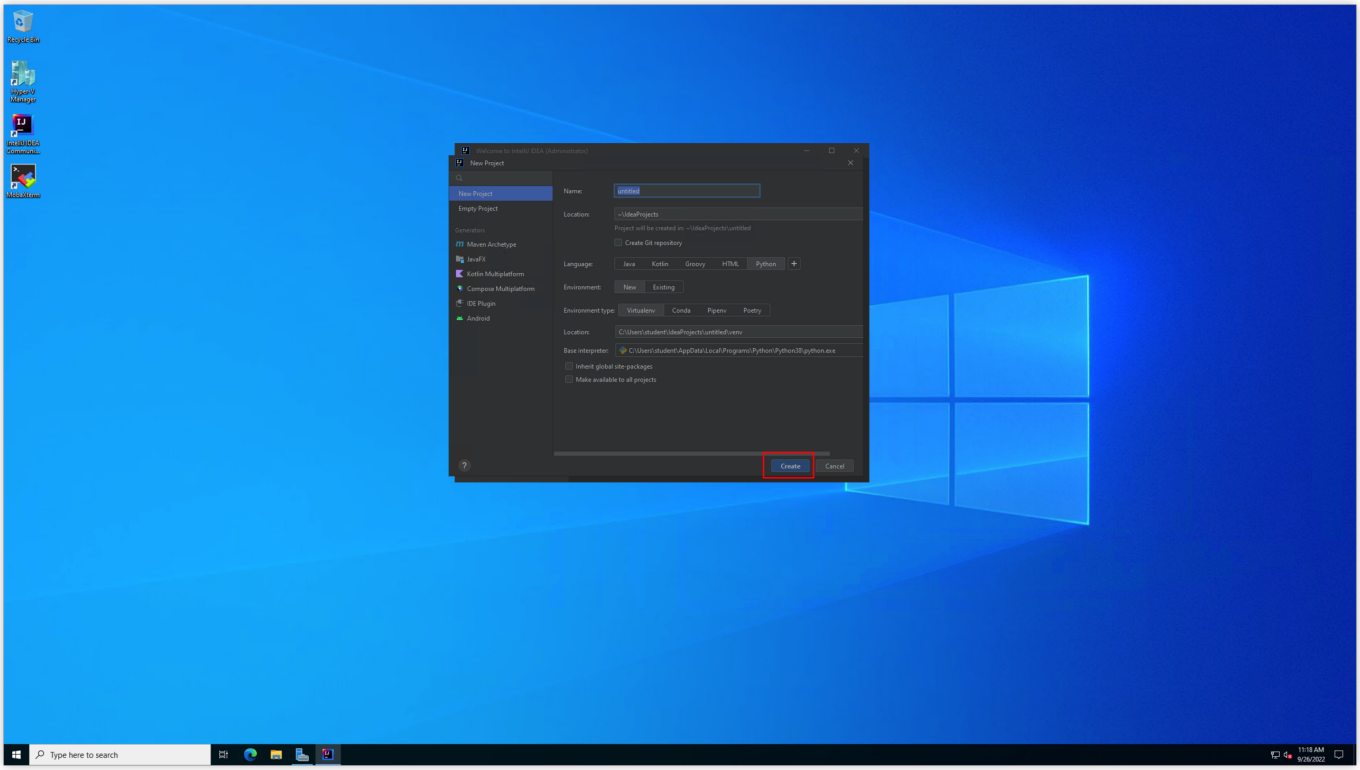
IntelliJ IDE

Start an new Python project

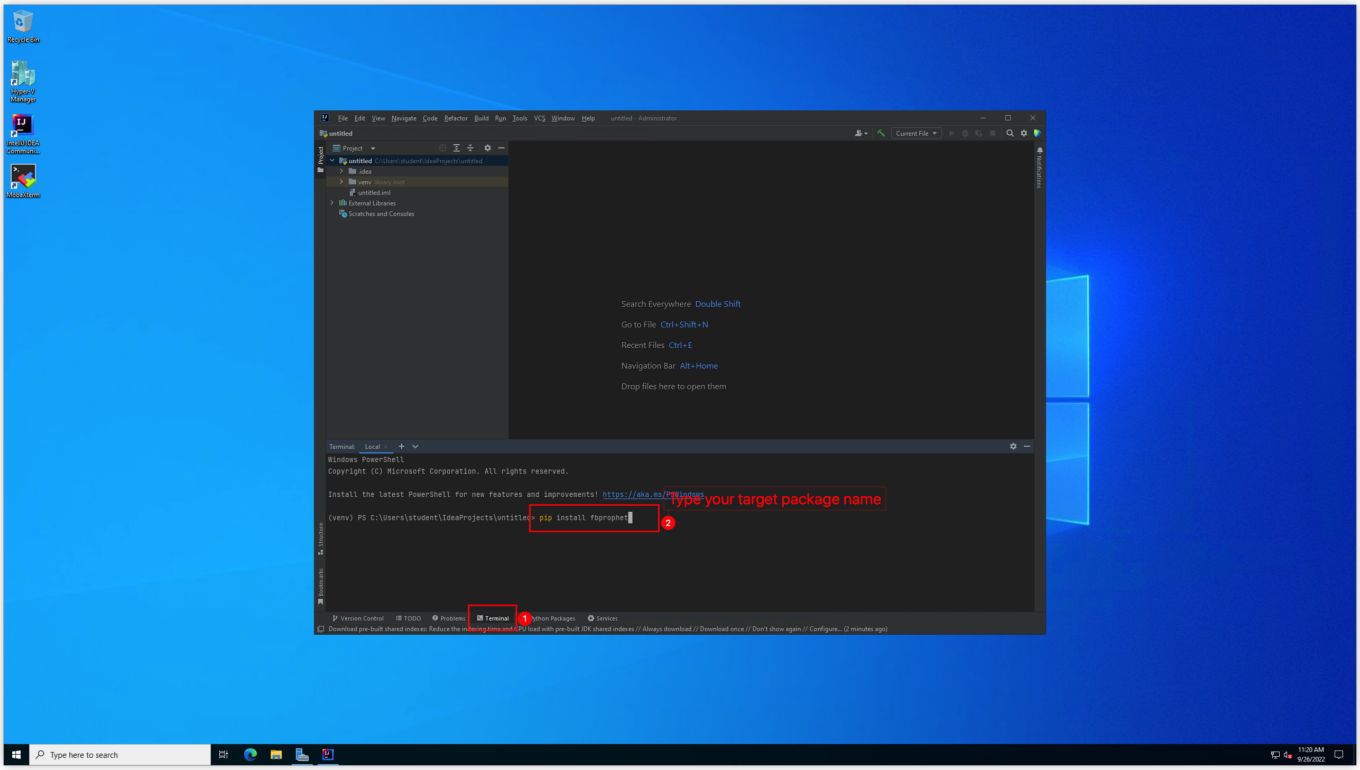

Recycle Bin
Hyper-V Manager
IntelliJ IDEA Community...
MicroMiner
1
Welcome to IntelliJ IDEA (Administrator)
IntelliJ IDEA
2022.2.2
Projects
Customize
Plugins
Learn IntelliJ IDEA
Welcome to IntelliJ IDEA (Administrator)
Create a new project to start from scratch.
Open existing project from disk or version control.
New Project
2
Open
Get from VCS
Take a quick onboarding tour
Get familiar with the IntelliJ IDEA user interface and learn how to code in Java with smart assistance in just 7 minutes!
Start Tour
Type here to search
11:11 AM
9/26/2022


Define your project name
1
2
3
Type here to search


Plugins
Marketplace
Installed
Python Community Edition
Install
1
2
OK
Type here to search



To install a Python denpendency package



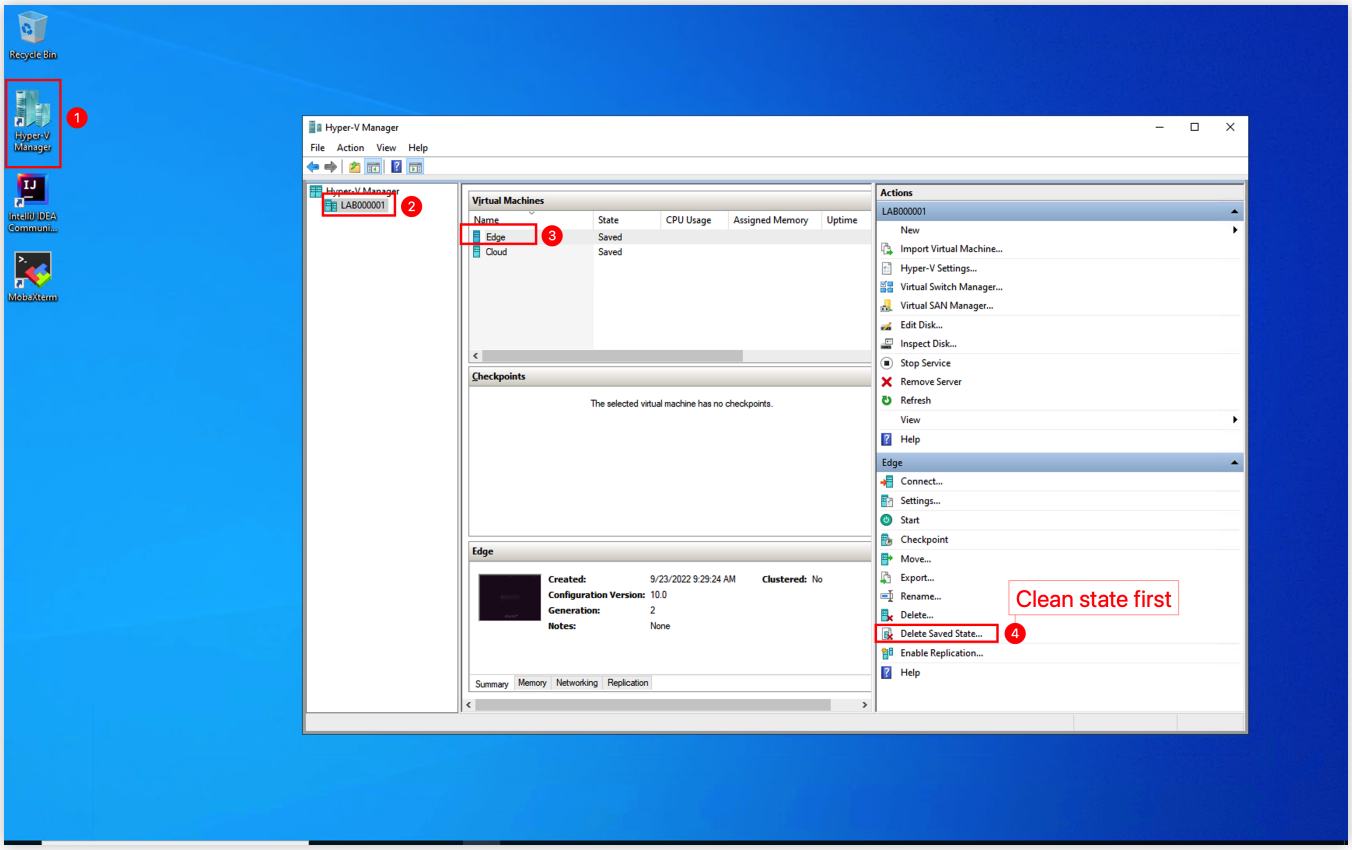
Dependencies Needed in Course

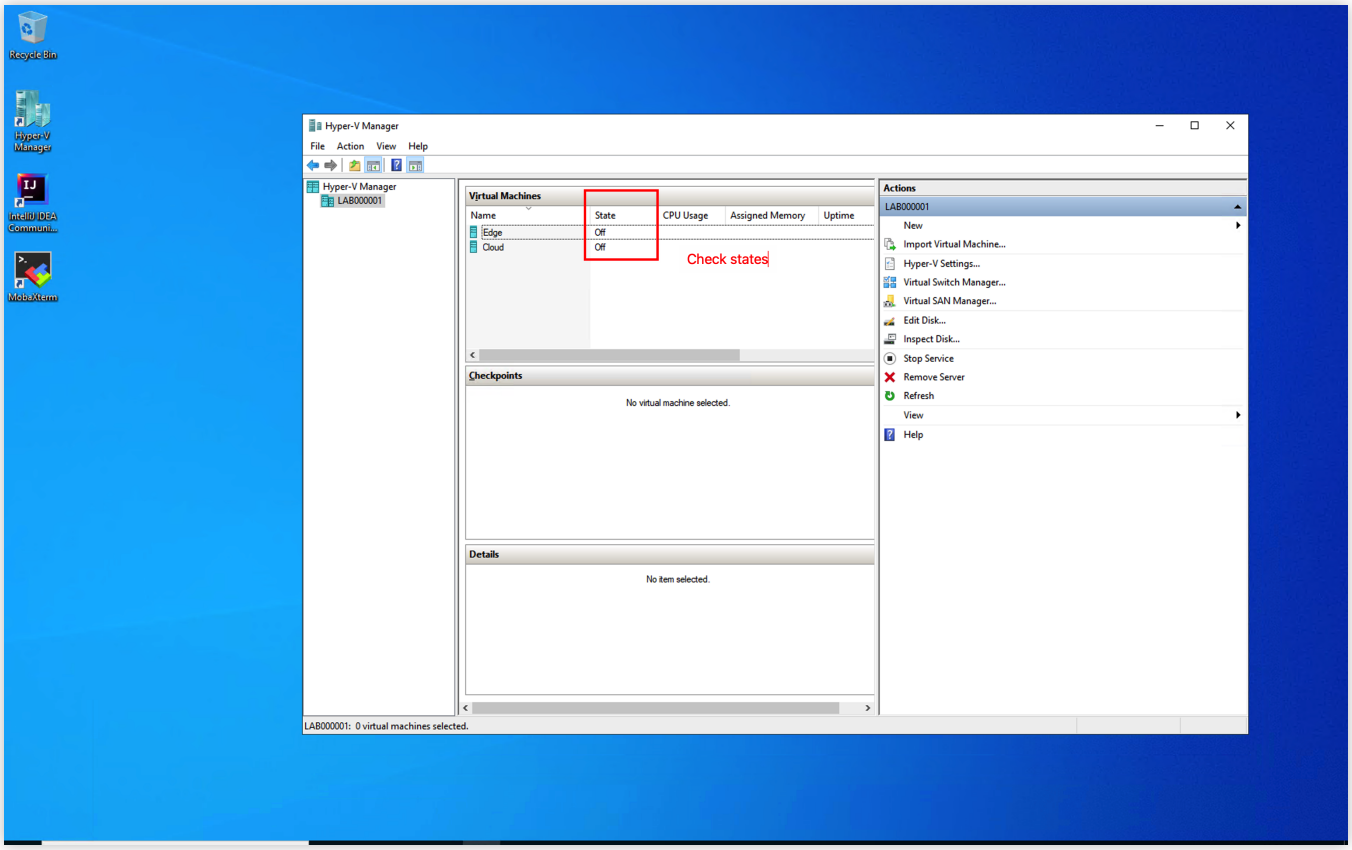
Package Name	Version (== or latest)	Install Command	Task
requests	2.28.1	pip install requests	1
paho.mqtt	1.6.1	pip install paho–mqtt	1 & 2
pika	1.3.0	python –m pip install pika –– upgrade	2 & 3
fbprophet	0.7.1	python –m pip install prophet	3
matplotlib	3.6.0	pip install matplotlib	3

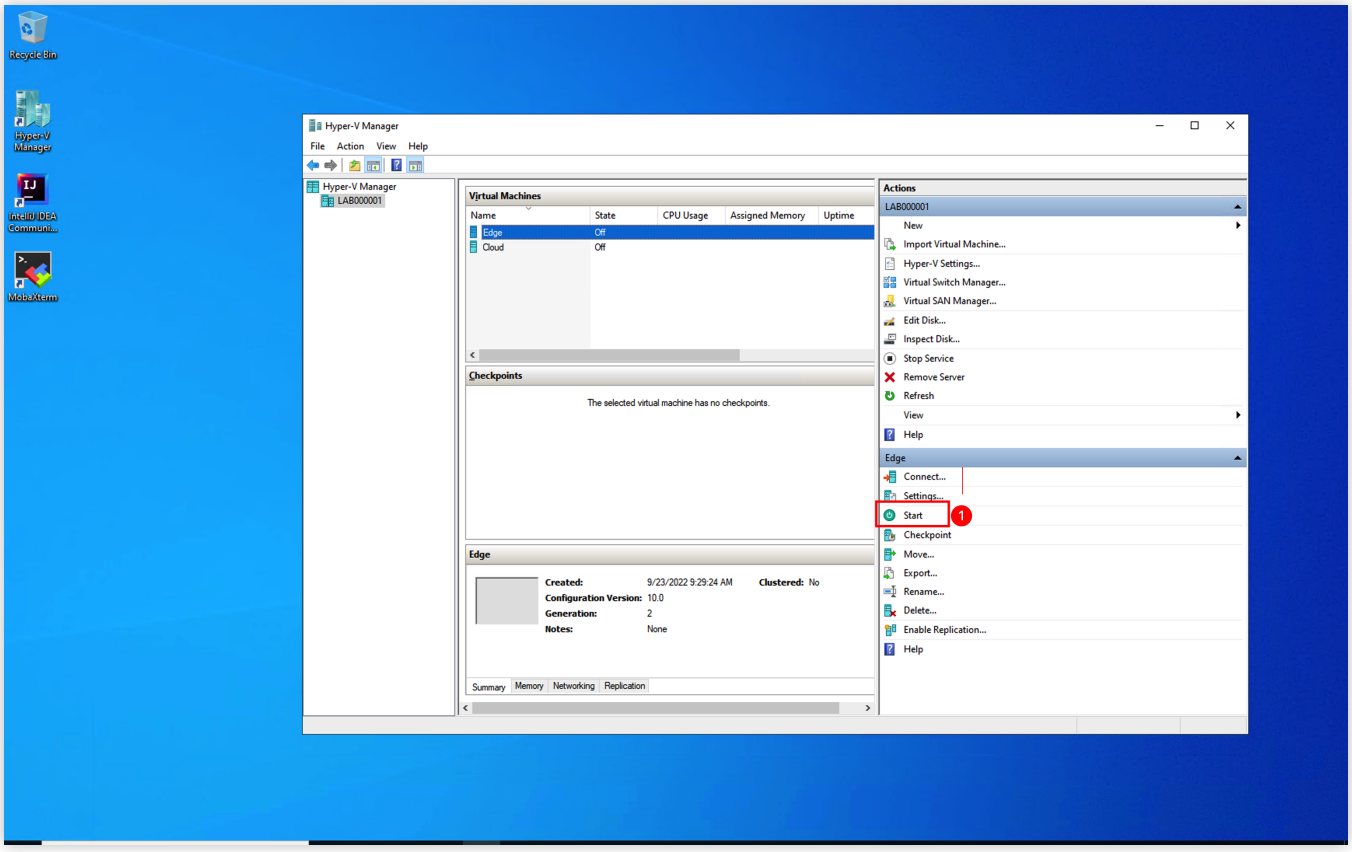
Ubuntu VMs

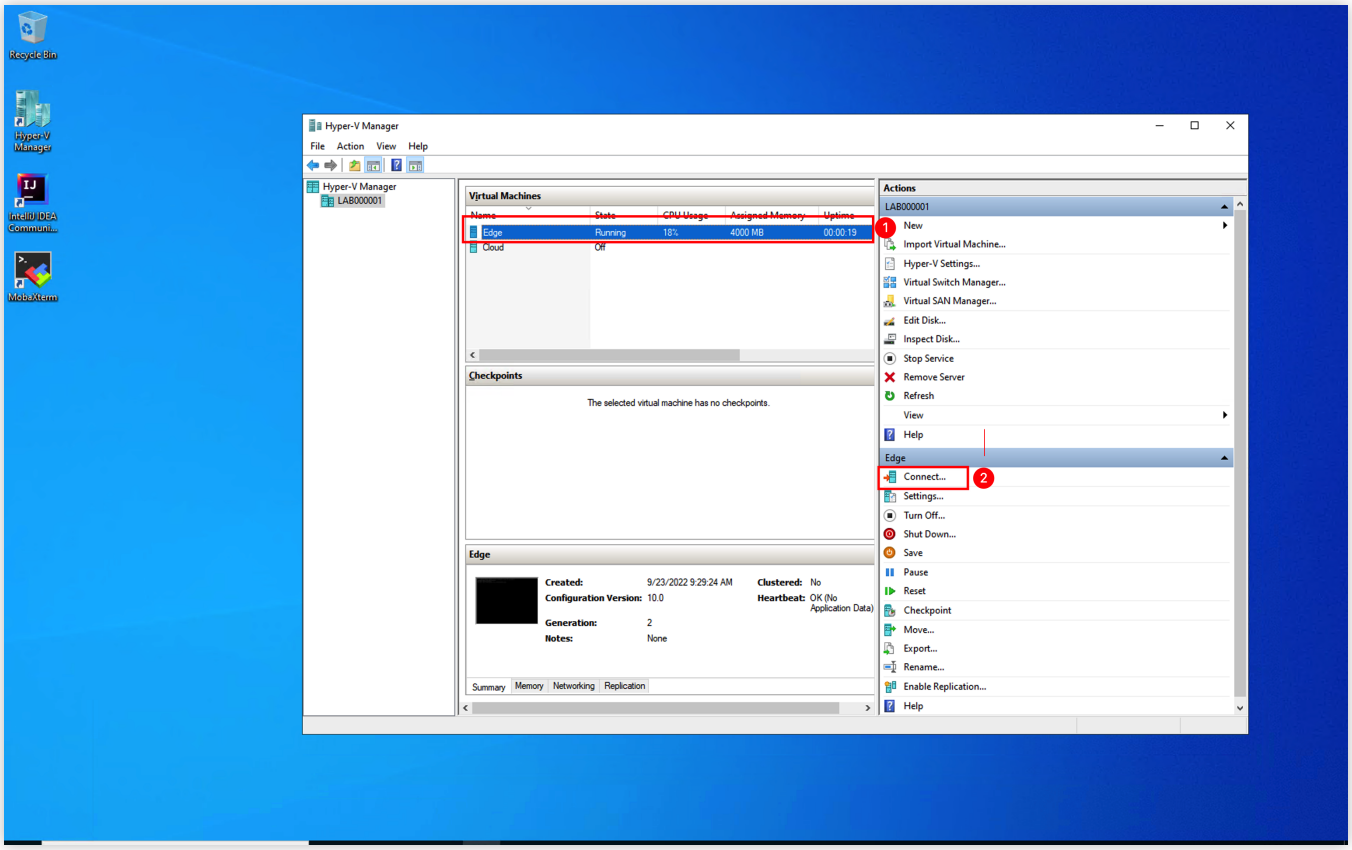
To start Ubuntu VMs

For the first time using, please clean saved state first (both Edge and Cloud), according to the following screenshot !

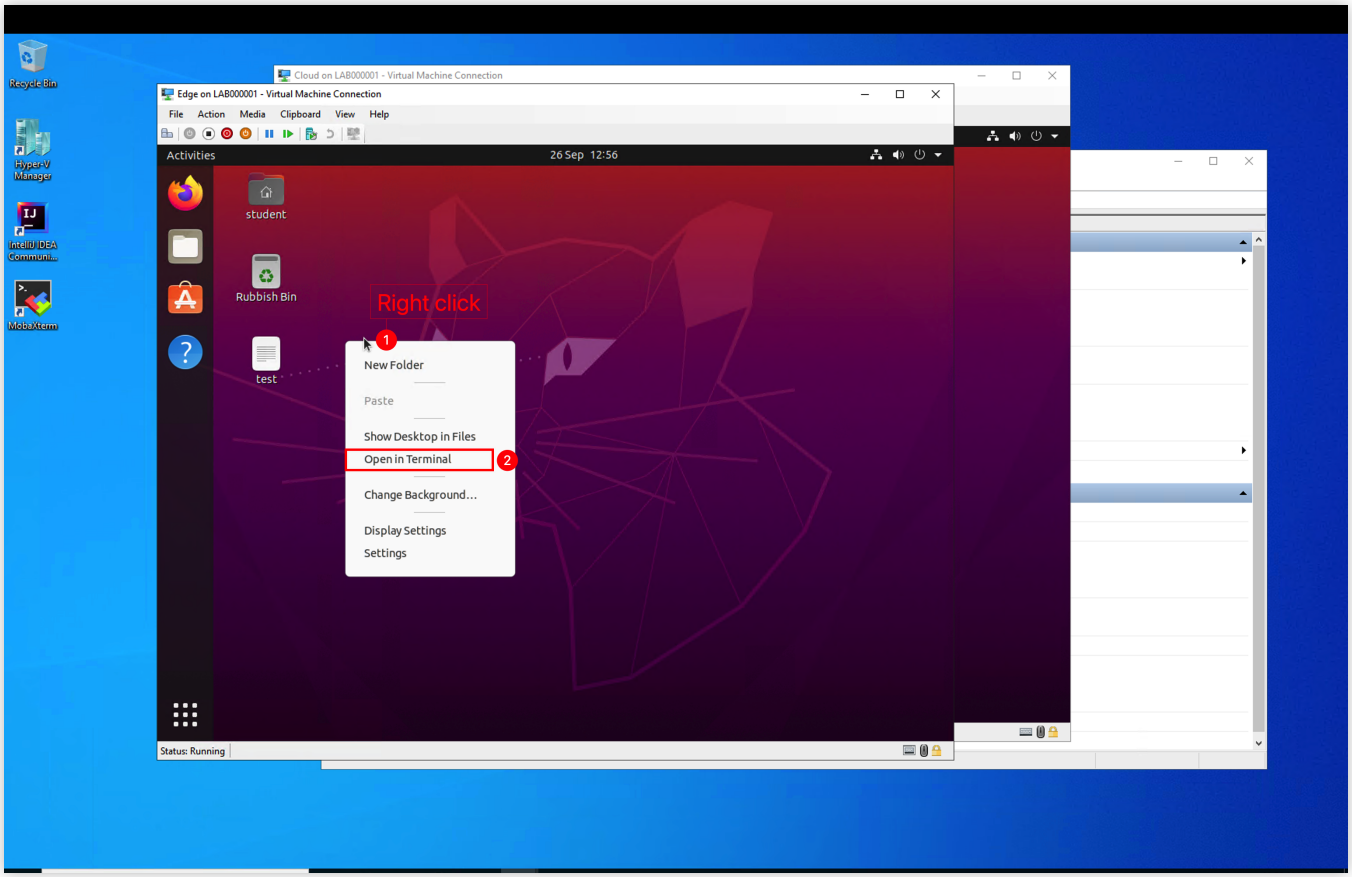

Recycle Bin
Hyper-V Manager
IntelliJ IDEA Communi...
MobaXterm
1
Hyper-V Manager
File Action View Help
Hyper-V Manager
LAB000001
2
Virtual Machines
Name State CPU Usage Assigned Memory Uptime
Edge Saved
Cloud Saved
3
Checkpoints
The selected virtual machine has no checkpoints.
Edge
Created: 9/23/2022 9:29:24 AM Clustered: No
Configuration Version: 10.0
Generation: 2
Notes: None
Summary Memory Networking Replication
Actions
LAB000001
New
Import Virtual Machine...
Hyper-V Settings...
Virtual Switch Manager...
Virtual SAN Manager...
Edit Disk...
Inspect Disk...
Stop Service
Remove Server
Refresh
View
Help
Edge
Connect...
Settings...
Start
Checkpoint
Move...
Export...
Rename...
Delete...
Delete Saved State...
4
Enable Replication...
Help
Clean state first


Recycle Bin
Hyper-V Manager
IntelliJ IDEA Communi...
MobaXterm
Hyper-V Manager
File Action View Help
Hyper-V Manager
LAB000001
Virtual Machines
Name State CPU Usage Assigned Memory Uptime
Edge Off
Cloud Off
Check states
Checkpoints
No virtual machine selected.
Details
No item selected.
Actions
LAB000001
New
Import Virtual Machine...
Hyper-V Settings...
Virtual Switch Manager...
Virtual SAN Manager...
Edit Disk...
Inspect Disk...
Stop Service
Remove Server
Refresh
View
Help
LAB000001: 0 virtual machines selected.





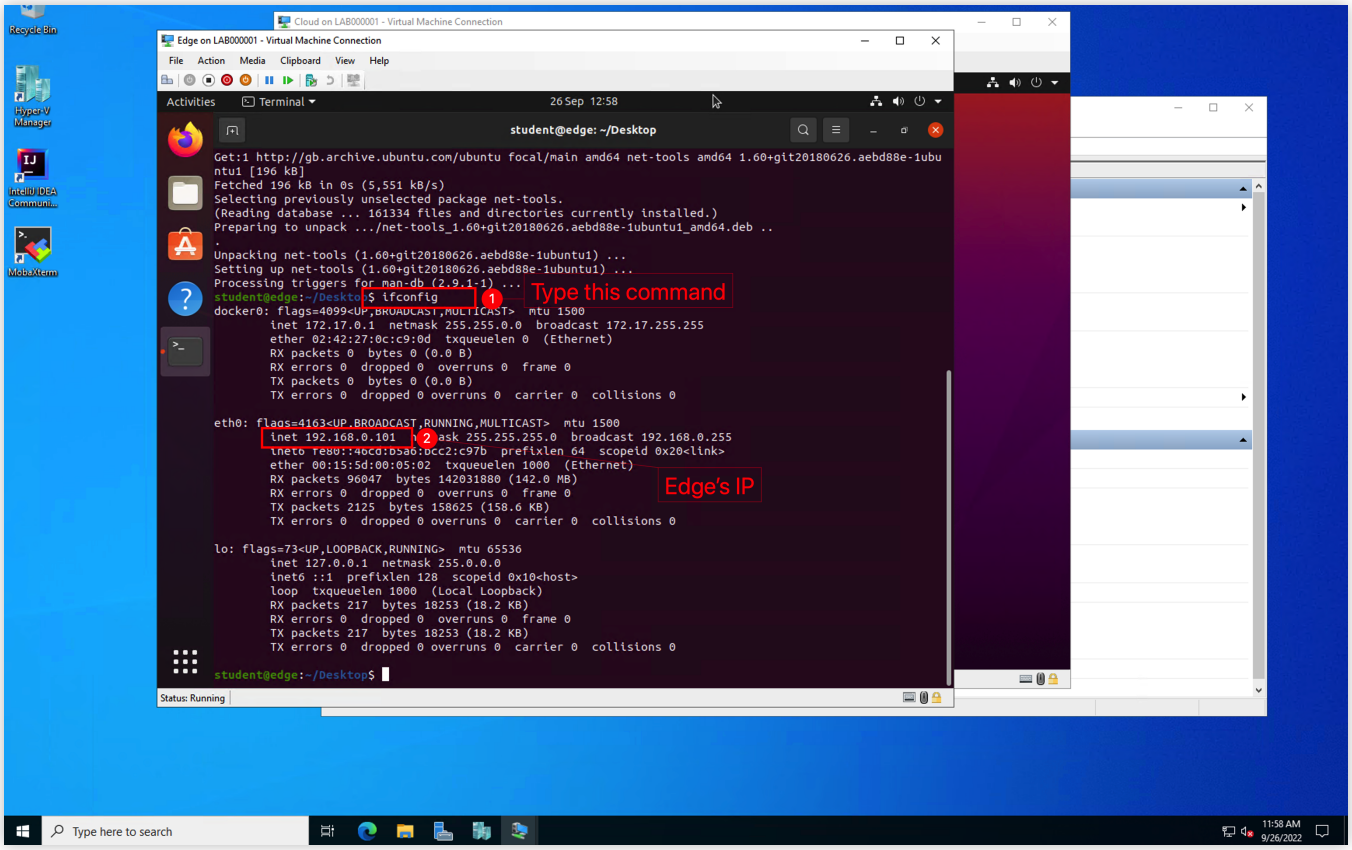
To get Ubuntu VM's IP address



Install net-tools to system

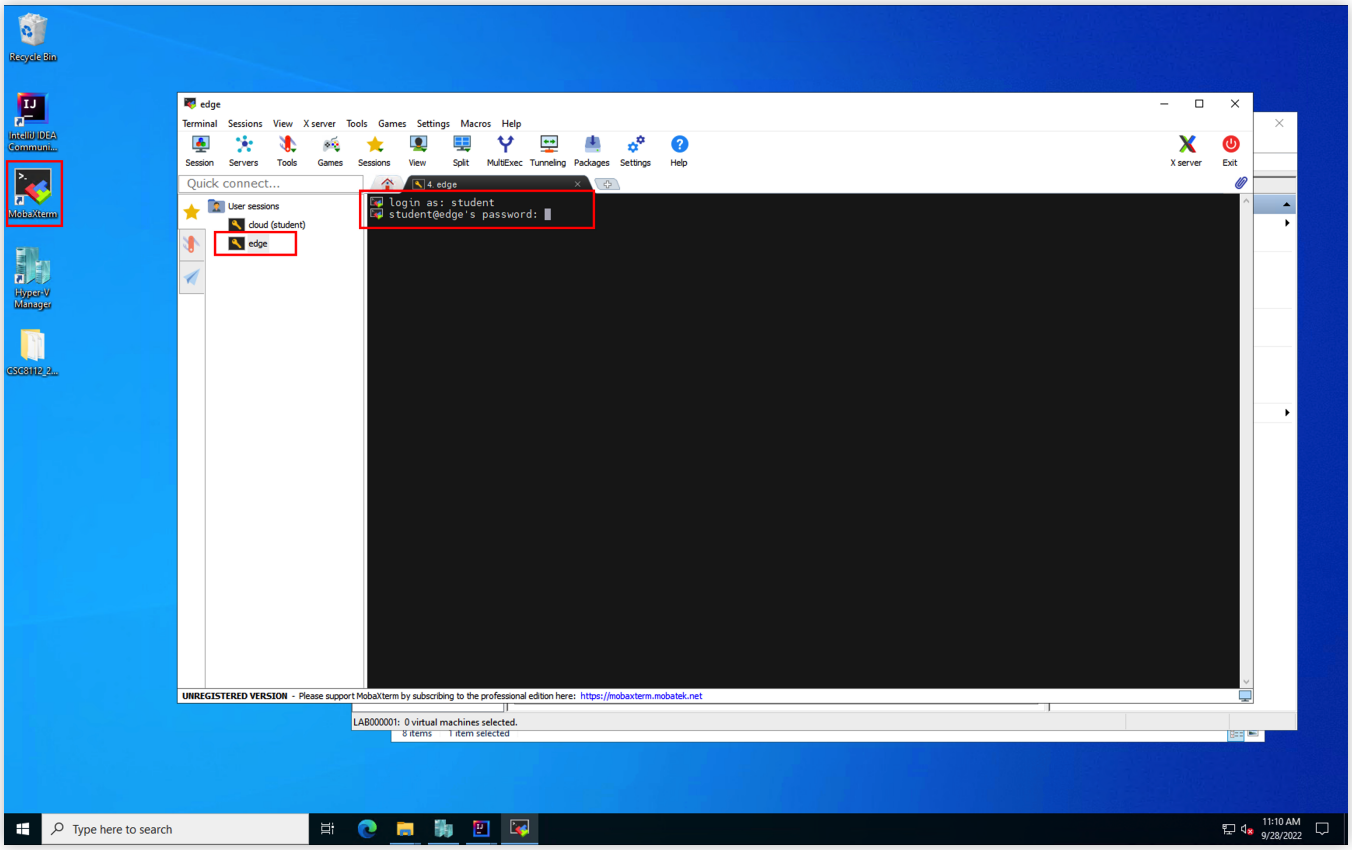
Install net-tools Bash 复制代码

```
sudo apt install net-tools
```



SSH to Ubuntu VMs

The Azure VM provided "MobaXterm" to let you easily connect Ubuntu VMs by terminal way.



Prepare necessary environment for Ubuntu VMs

Python 3

Please manually install Python3.8 and pip in both Ubuntu VMs.

Install Python 3.8 and Pip — Shell | 复制代码

```
sudo apt install python3

sudo apt install python3-pip
```

Docker-compose tool

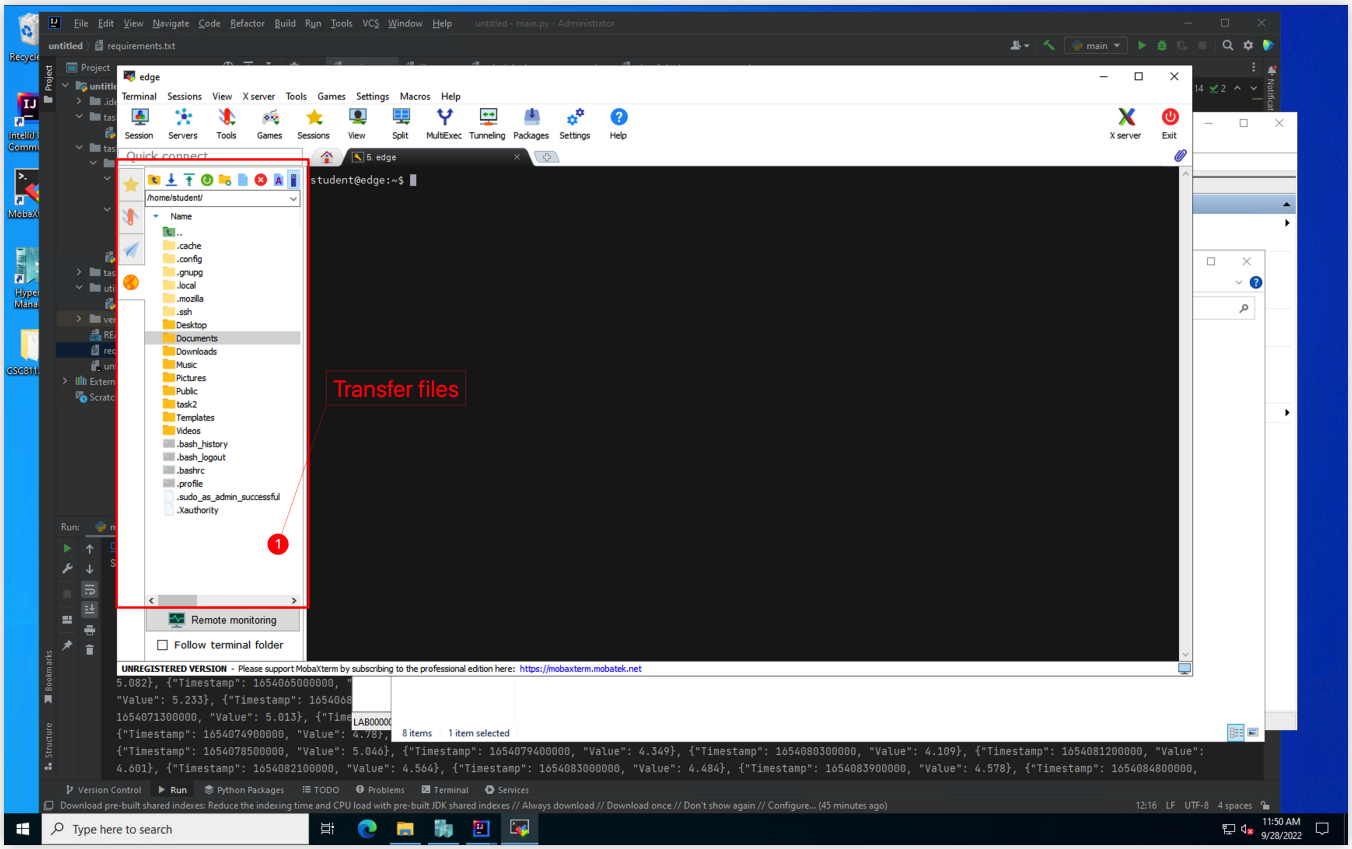
Please manually install docker-compose tool in both Ubuntu VMs.

Install docker-compose tool — Shell | 复制代码

```
sudo apt install docker-compose
```

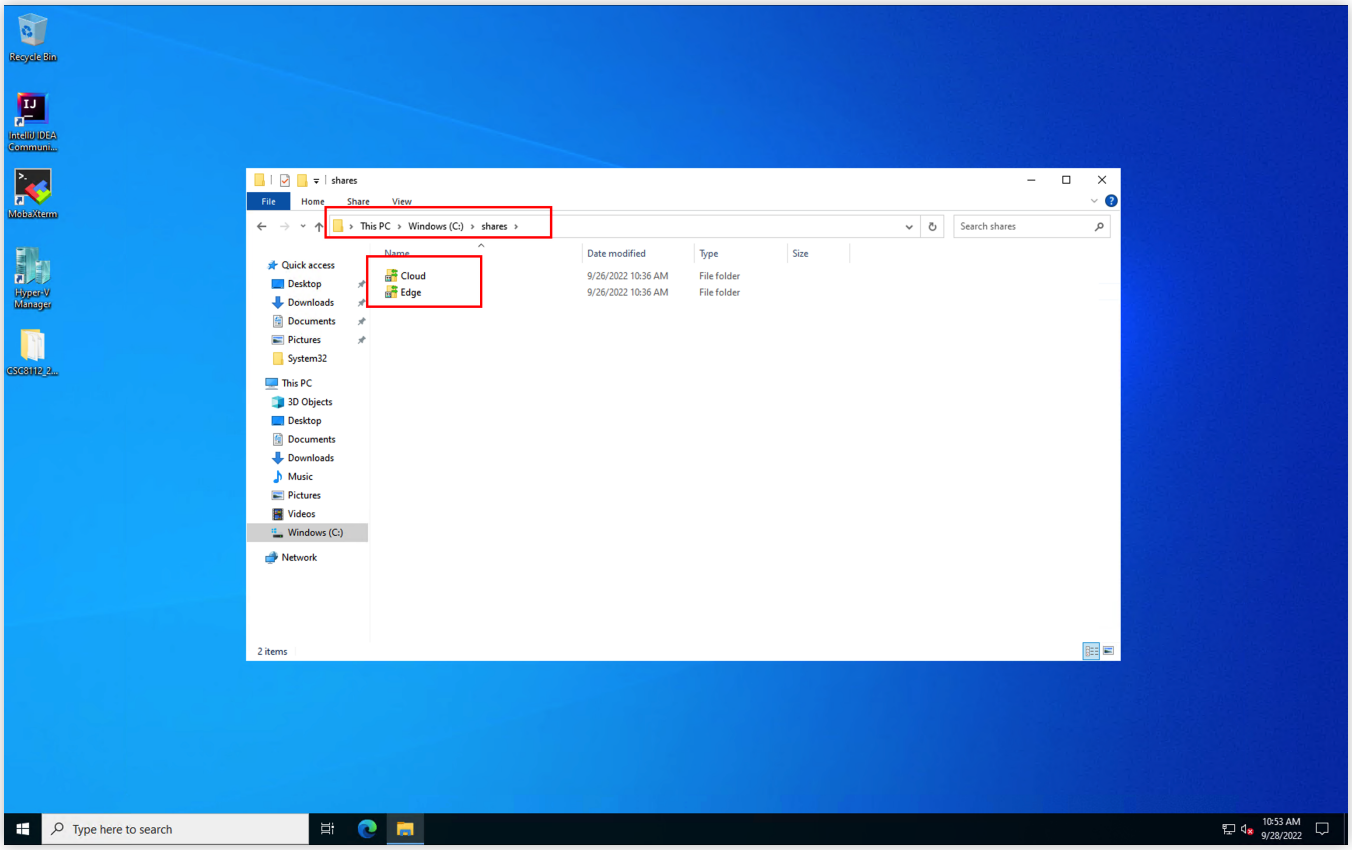
Transfer files with VMs

By MobaXterm (Recommend)



By shared folder

We provide two shared directories for Cloud and Edge respectively.



How to write a docker–compose configuration file

Hint: Refer the docker–compose.yml file in Use Docker Compose

Example:

docker–compose file example — YAML — 复制代码

```yaml
version: "3"
services:
  mongo:
    image: mongo
    deploy:
      replicas: 1
    ports:
      - '27017:27017'
```

How to build your code into docker image

Prepare a Dockerfile

Example:

Dockerfile — Dockerfile | 复制代码

```
# Base on image_full_name (e.g., ubuntu:18.04) docker image
FROM python:3.8.12

# Switch to root
USER root

# Copy all sources files to workdir
ADD <your project directory> /usr/local/source

# Change working dir
WORKDIR /usr/local/source

# Prepare project required running system environments
#   requirements.txt is a document that pre-define any
#   python dependencies with versions required of your code
RUN pip3 install -r requirements.txt

# Start task
CMD python3 <your main .py file>
```

Prepare a docker-compose configuration file

docker-compose configuration file — YAML | 复制代码

```
version: "3"
services:
  data_injector:
    image: data_injector:latest
```

Execution flow

1. Run your Dockerfile file first with command:

Build code — Shell | 复制代码

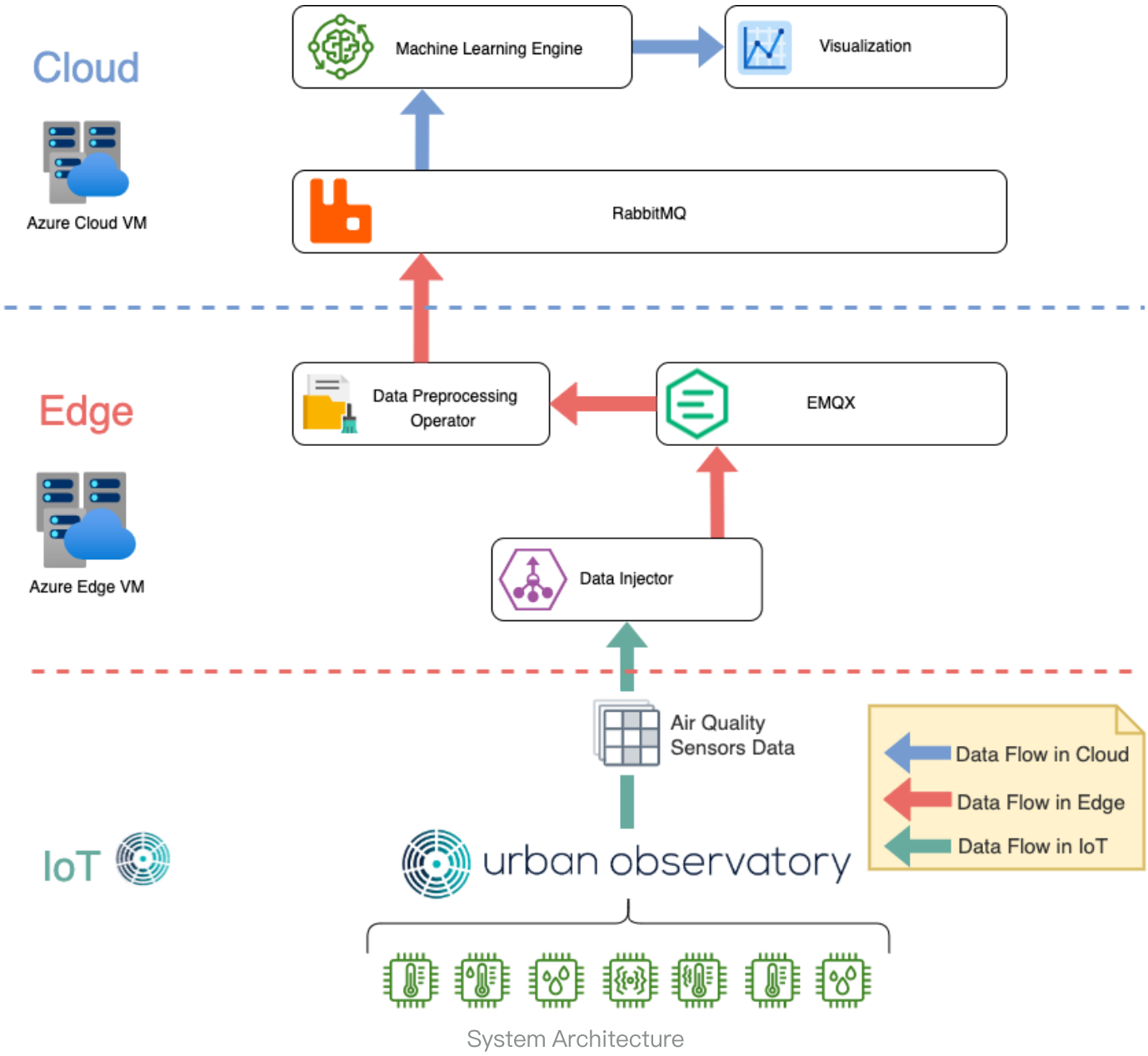
```
sudo docker build -t <name:tag> <source directory (relative)>
```

2. Run your docker-compose file with command:
    a. Please make sure the docker-compose tool already installed, if not, please refer to chapter Docker-compose tool

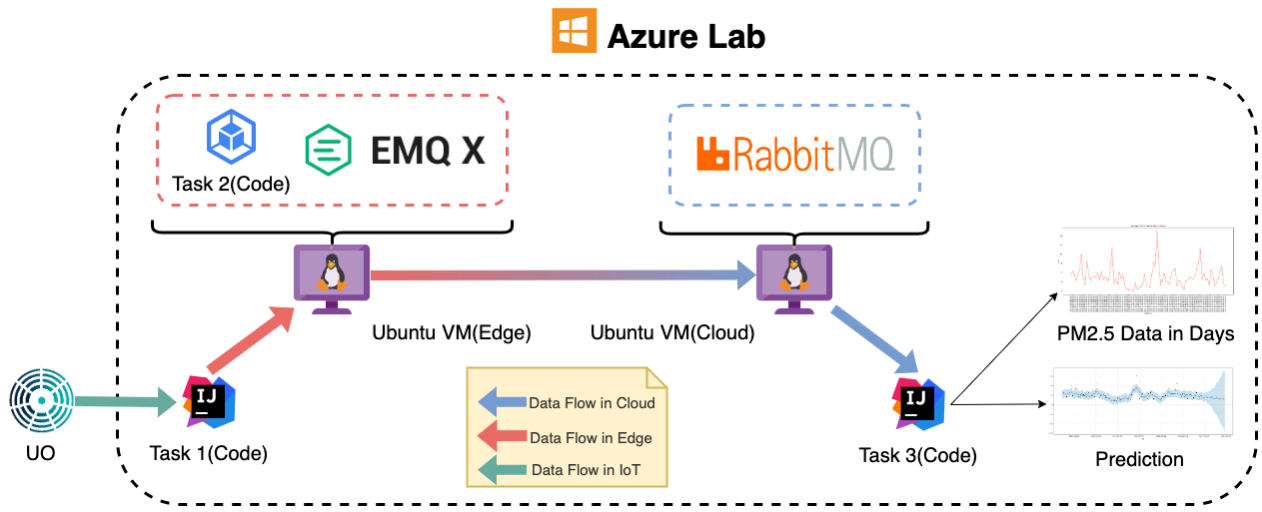
```
Start a image                                    Shell | 复制代码
1  sudo docker-compose up
```

Coursework supplements

Overview structures

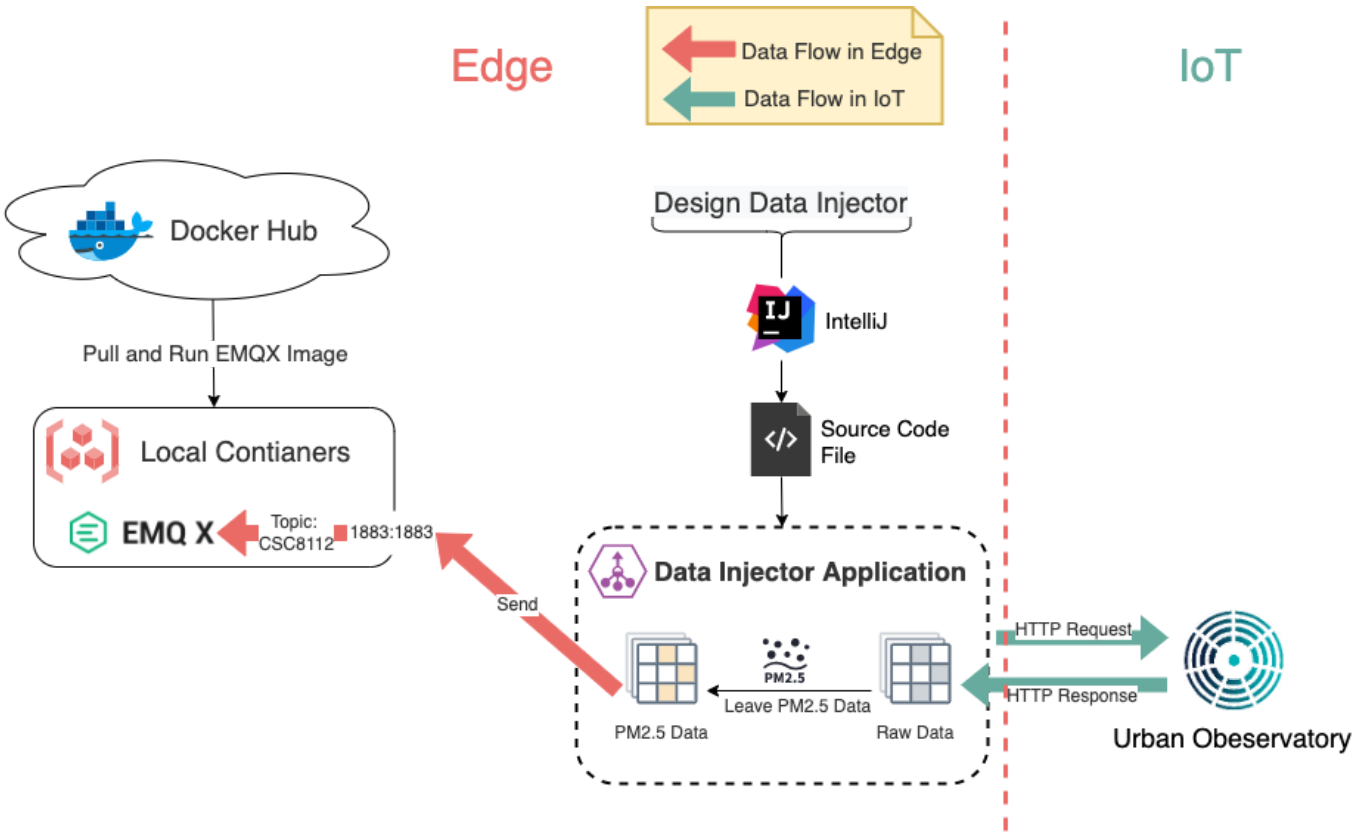


System Architecture



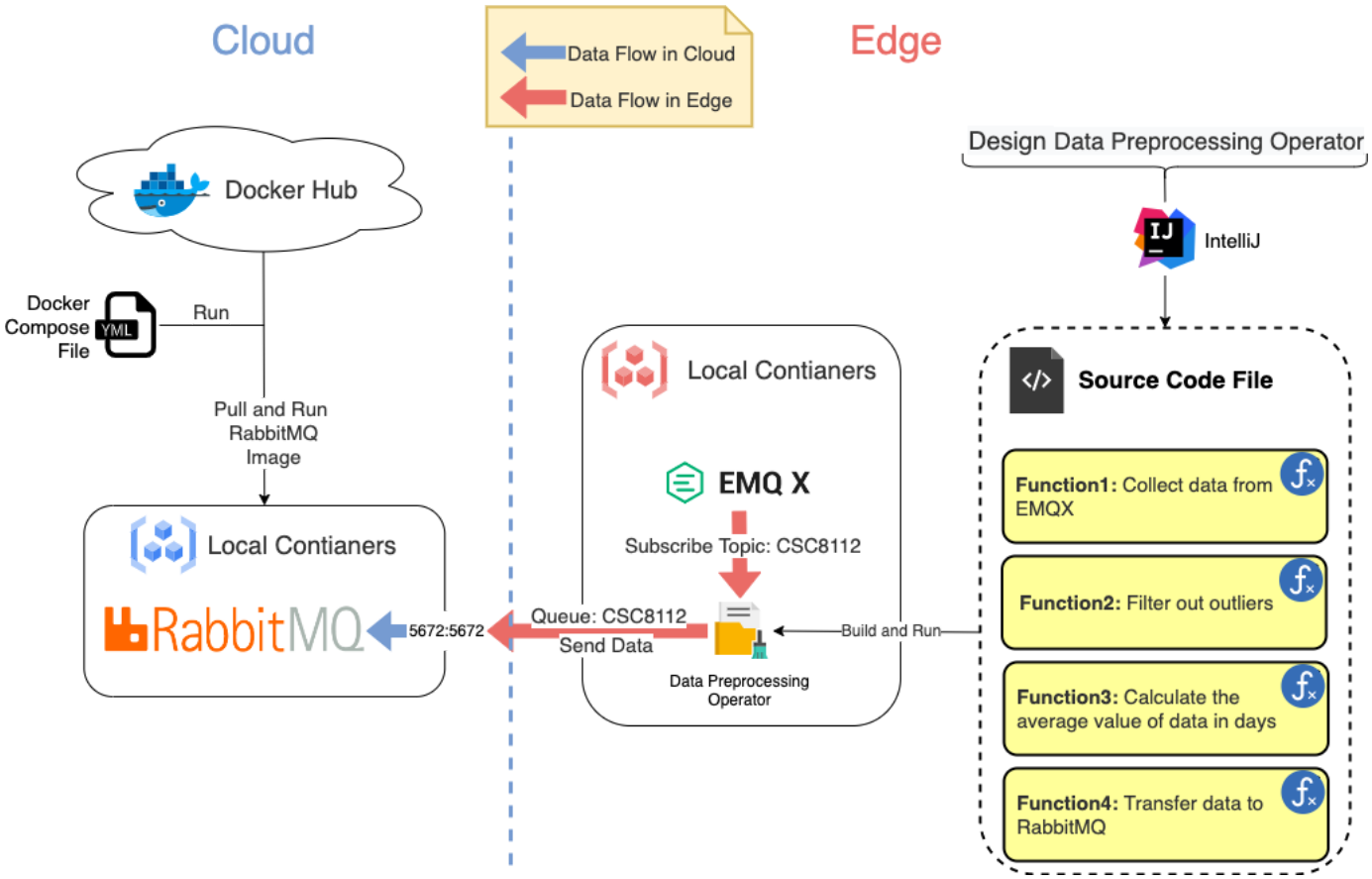
The Final Workflow Structure

Task 1



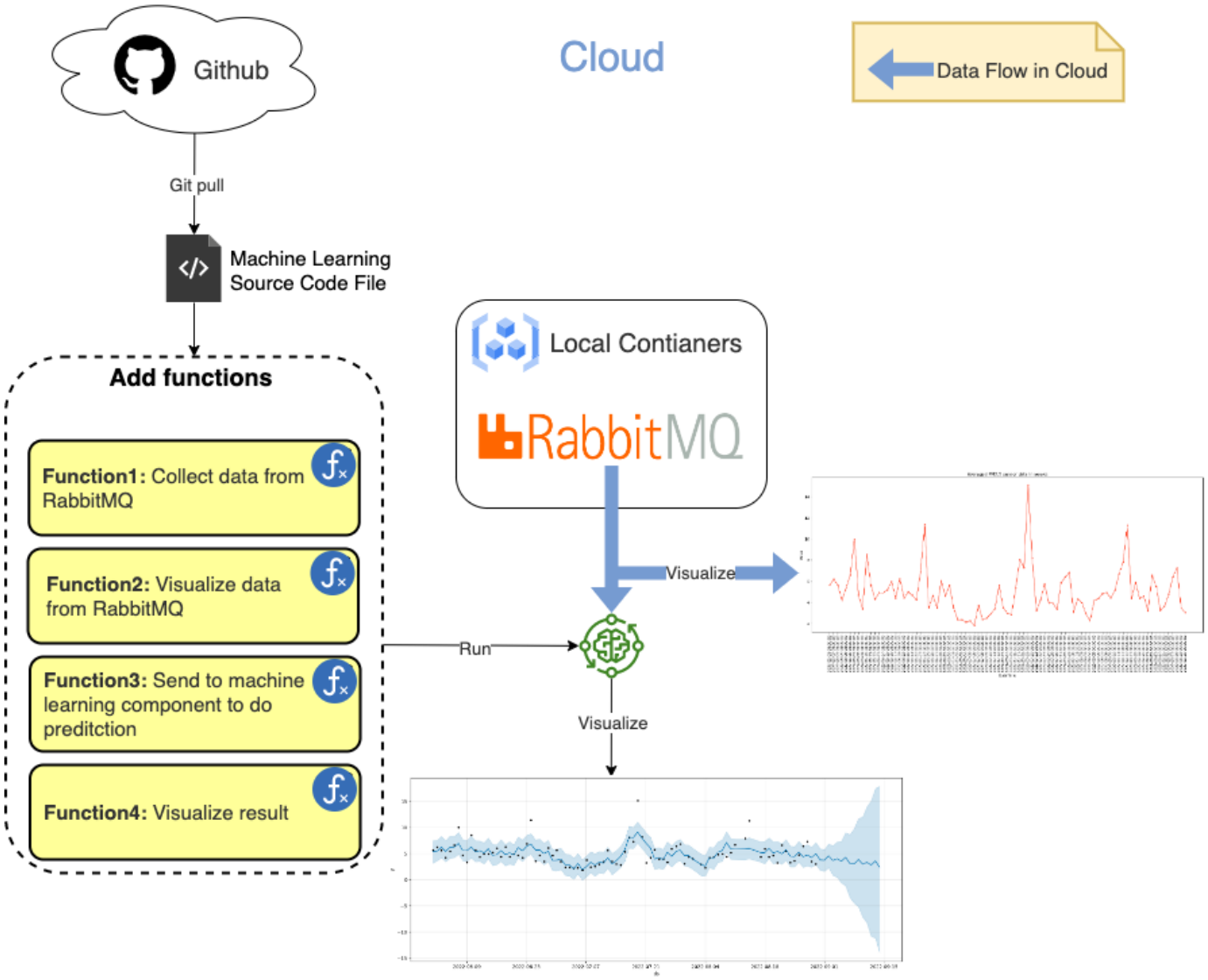
Task1 System Architecture

Task 2



Task2 System Architecture

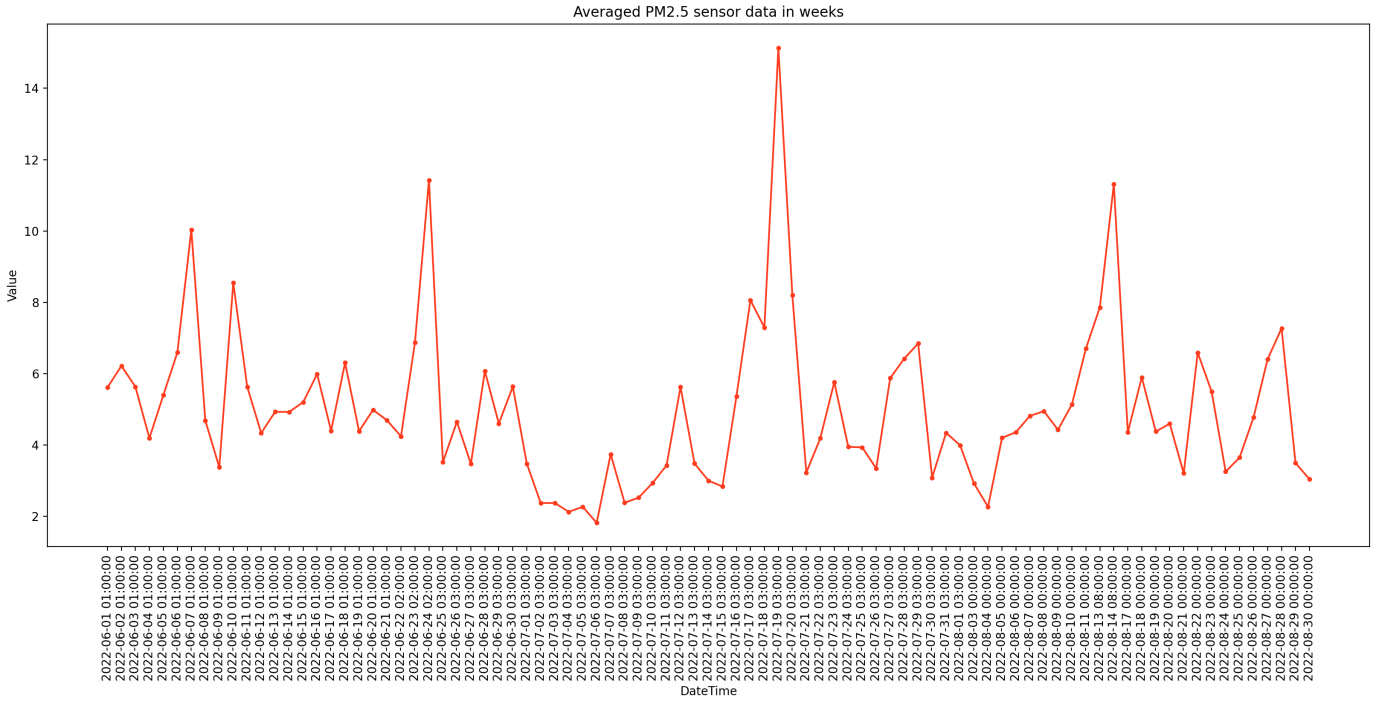
Task 3



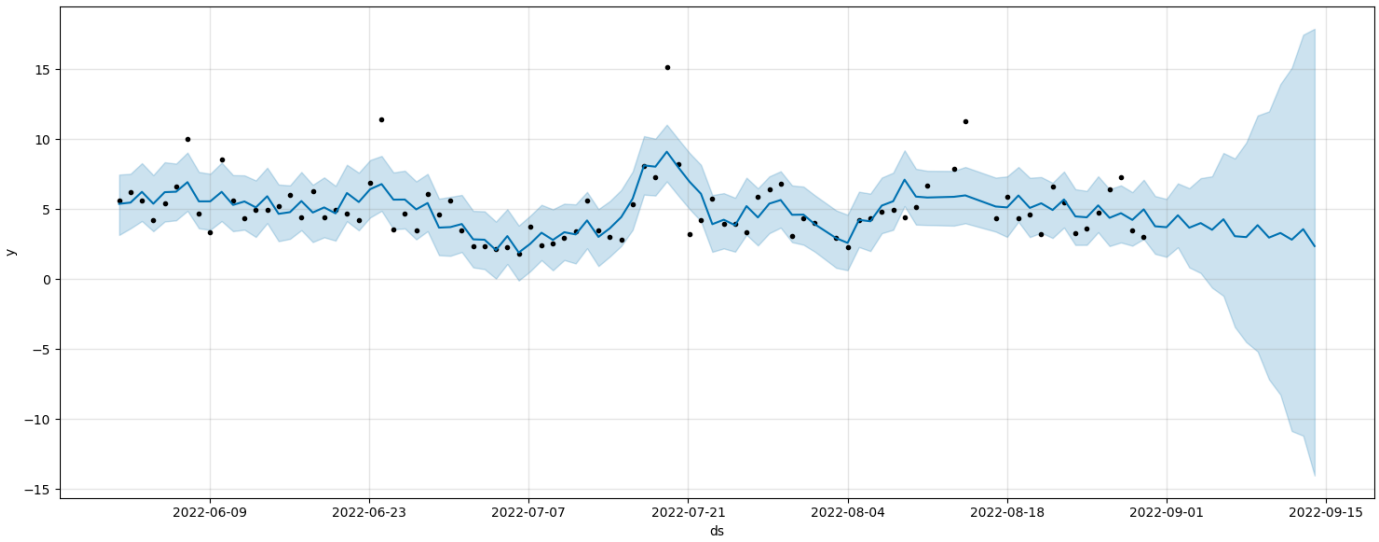
Task3 Solution Architecture

Download Machine Learning engine code: GitHub – ncl-iot-team/CSC8112_MLEngine

Final results in visualization looks like:



Averaged PM2.5 data in days



PM2.5 Prediction for next 15 days

Implementation Examples

How to send HTTP request to get sensor data

HTTP Get Python 复制代码

```python
import requests

if __name__ == '__main__':
    # Target URL
    url = "http://uoweb3.ncl.ac.uk/api/v1.1/sensors/PER_NE_CAJT_NCA189_SJB1_SJB2" \
          "/data/json/?starttime=20220901&endtime=20221001 "

    # Request data from Urban Observatory Platform
    resp = requests.get(url)

    # Convert response(Json) to dictionary format
    raw_data_dict = resp.json()

    print(raw_data_dict)

```

How to define MQTT subscriber in Python

MQTT subscriber Python 复制代码

```python
import json

from paho.mqtt import client as mqtt_client

if __name__ == '__main__':
    mqtt_ip = "localhost"
    mqtt_port = 1883
    topic = "CSC8112"

    # Create a mqtt client object
    client = mqtt_client.Client()


    # Callback function for MQTT connection
    def on_connect(client, userdata, flags, rc):
        if rc == 0:
            print("Connected to MQTT OK!")
        else:
            print("Failed to connect, return code %d\n", rc)

    # Connect to MQTT service
    client.on_connect = on_connect
    client.connect(mqtt_ip, mqtt_port)

    # Callback function will be triggered
    def on_message(client, userdata, msg):
        print(f"Get message from publisher {json.loads(msg.payload)}")

    # Subscribe MQTT topic
    client.subscribe(topic)
    client.on_message = on_message

    # Start a thread to monitor message from publisher
    client.loop_forever()
```

How to define MQTT publisher in Python

MQTT publisher Python 复制代码

```python
import json
from paho.mqtt import client as mqtt_client

if __name__ == '__main__':
    mqtt_ip = "localhost"
    mqtt_port = 1883
    topic = "CSC8112"
    msg = "Hello!"

    # Create a mqtt client object
    client = mqtt_client.Client()

    # Callback function for MQTT connection
    def on_connect(client, userdata, flags, rc):
        if rc == 0:
            print("Connected to MQTT OK!")
        else:
            print("Failed to connect, return code %d\n", rc)

    # Connect to MQTT service
    client.on_connect = on_connect
    client.connect(mqtt_ip, mqtt_port)

    # Publish message to MQTT
    # Note: MQTT payload must be a string, bytearray, int, float or None
    msg = json.dumps(msg)
    client.publish(topic, msg)
```

How to define RabbitMQ consumer inPython

RabbitMQ consumer Python 复制代码

```python
import json
import pika

if __name__ == '__main__':

    rabbitmq_ip = "localhost"
    rabbitmq_port = 5672
    # Queue name
    rabbitmq_queque = "CSC8112"

    def callback(ch, method, properties, body):
        print(f"Got message from producer msg: {json.loads(body)}")

    # Connect to RabbitMQ service with timeout 1min
    connection = pika.BlockingConnection(
        pika.ConnectionParameters(host=rabbitmq_ip, port=rabbitmq_port, so
cket_timeout=60))
    channel = connection.channel()
    # Declare a queue
    channel.queue_declare(queue=rabbitmq_queque)

    channel.basic_consume(queue=rabbitmq_queque,
                          auto_ack=True,
                          on_message_callback=callback)

    channel.start_consuming()
```

How to define RabbitMQ producer inPython

RabbitMQ producer Python 复制代码

```python
import pika
import json

if __name__ == '__main__':
    rabbitmq_ip = "localhost"
    rabbitmq_port = 5672
    # Queue name
    rabbitmq_queque = "CSC8112"
    msg = "Hello!"
    # Connect to RabbitMQ service
    connection = pika.BlockingConnection(pika.ConnectionParameters(host=rabbitmq_ip, port=rabbitmq_port))
    channel = connection.channel()

    # Declare a queue
    channel.queue_declare(queue=rabbitmq_queque)

    # Produce message
    channel.basic_publish(exchange='',
                          routing_key=rabbitmq_queque,
                          body=json.dumps(msg))

    connection.close()
```

How to use Matplotlib to visualize a line chart

Matplotlib line chart Python 复制代码

```python
import matplotlib.pyplot as plt
import pandas as pd

if __name__ == '__main__':
    # Prepare data
    data = {
        'Timestamp': ['01/09','02/09','03/09','04/09','05/09'],
        'Value': [1,2,1,10,5]
    }

    data_df = pd.DataFrame(data)

    # Initialize a canvas
    plt.figure(figsize=(8, 4), dpi=200)
    # Plot data into canvas
    plt.plot(data_df["Timestamp"], data_df["Value"], color="#FF3B1D", marker='.', linestyle="-")
    plt.title("Example data for demonstration")
    plt.xlabel("DateTime")
    plt.ylabel("Value")

    # Save as file
    plt.savefig("figure1.png")
    # Directly display
    plt.show()
```

How to use Machine Learning Engine

Using Machine Learning Engine — Python — 复制代码

```python
from ml_engine import MLPredictor

if __name__ == '__main__':
    # Prepare data
    data = {
        'Timestamp': ['2020-09-01', '2020-09-02', '2020-09-03',
                      '2020-09-04', '2020-09-05'],
        'Value': [1, 2, 1, 10, 5]
    }
    data_df = pd.DataFrame(data)

    # Create ML engine predictor object
    predictor = MLPredictor(data_df)
    # Train ML model
    predictor.train()
    # Do prediction
    forecast = predictor.predict()

    # Get canvas
    fig = predictor.plot_result(forecast)
    fig.savefig("prediction.png")
    fig.show()
```

Problems

Docker logs cannot print output

1. Please add (flush=True in your print code)

print — Python — 复制代码

```python
print("hello",flush=True)
```

Appendix

About target sensors information

Useful links:

1. Urban Observatory : Urban Observatory
2. City Data API v1.1 doc : https://newcastle.urbanobservatory.ac.uk/api_docs/doc/sensors-csv/
3. Sensors search: https://newcastle.urbanobservatory.ac.uk/#sensor_info

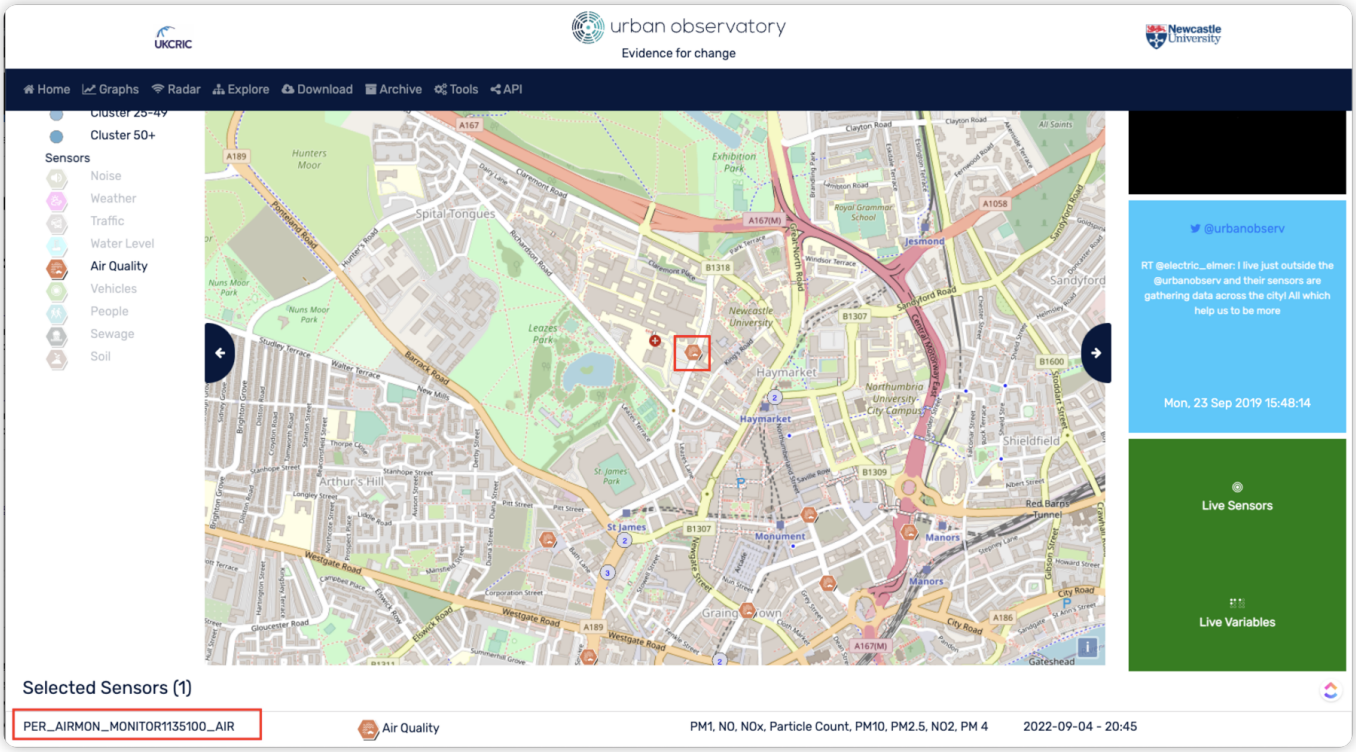
Target sensor's information:

Sensor Info JSON 复制代码


```
{"Third Party": false, "Location (WKT)": "POINT(-1.617676 54.979118)", "Sensor Centroid Longitude": -1.617676,
  "Sensor Centroid Latitude": 54.979118, "Broker Name": "aq_mesh_api", "Sensor Height Above Ground": 2.0,
  "Raw ID": "79525", "Ground Height Above Sea Level": 57.6699981689, "Sensor Name":"PER_AIRMON_MONITOR1135100"}
```


About Raw Data:

- **Date range:** From 01/06/2022 to 31/08/2022.
- **Data format:** Json
- **Size:** around 16.85MB(total)
    - PM2.5 data (total data points: 8057 with timestamp in millisecond, collecting rate: every 15min (900000ms))



Geo–location of target sensor