

TP3 DEVOPS :

Objectifs :

l'Objectif du TP 3 devops est de configurer un workflow Github Action, de le transférer en API à l'aide des fichiers du deuxième TP. Ensuite, le mettre à disposition son image (format API) sur Azure Container Registry (ACR) à l'aide de Github Actions.

Contenu du code :

Fichiers :

- [Dockerfile](#) : Les Dockerfiles sont des fichiers qui permettent de construire une image Docker adaptée à nos besoins, c'est pour cela qu'on l'utilise pour créer notre environnement.

[Code](#) :

```
🐳 docker.dockerfile > ...  
1 FROM python:3.8-alpine  
2 RUN mkdir /app  
3 ADD . /app  
4 WORKDIR /app  
5 RUN pip install -r requirements.txt  
6 CMD ["python", "tp_devops_api.py"]
```

- [tp_devops_api](#) : Le wrapper est nécessaire pour faire l'appel à notre API, pour cela on utilise le os.environ pour récupérer les variables depuis la commande docker -env. A l'aide du request on peut récupérer les données des villes qu'on veut et les afficher au format souhaité (JSON). L'api est basé sur le wrapper, on récupère la latitude et longitude avec des request flask pour permettre de les ajouter dans l'url. On lance l'api sur le localhost port:8081 comme demandé sur le TP.

[Code](#) :

```

import requests
import json
from flask import Flask, request, jsonify
import os
app = Flask(__name__)
api_key = os.environ['API_KEY']

with app.app_context():
    @app.route("/")
    def meteo():
        lat = request.args.get('lat')
        lon = request.args.get('lon')
        url = "https://api.openweathermap.org/data/2.5/weather?lat=%s&lon=%s&appid=%s&units=metric" % (
            lat, lon, api_key)
        response = requests.get(url)
        data = json.loads(response.text)
        if response.status_code != 200:
            return jsonify({
                'status': 'error',
                'message': 'La requête à l\'API météo n\'a pas fonctionné. Voici le message renvoyé par l\'API : {}'.format(data)
            }, 500)

        return jsonify({
            'status': 'ok',
            'data': data
        })

if __name__ == "__main__":
    port=80
    app.run(host="0.0.0.0", port=port)

```

- [requirements.txt](#) : Il s'agit de la liste des paquets Python dont l'installation est requise dans un environnement virtuel pour que l'application s'exécute correctement.
- le work FLOW : Automatisation du push sur Azure

name: Publish Docker image

on: push

jobs:

push_to_registry-linux:

name: Push Docker image to Docker Hub

runs-on: ubuntu-latest

steps:

- name: Check out the repo
uses: actions/checkout@v3

- name: 'Login via Azure CLI'
uses: azure/login@v1
with:
 creds: \${{ secrets.AZURE_CREDENTIALS }}

- name: hadolint
 uses: reviewdog/action-hadolint@v1
 with:
 reporter: github-pr-review

- name: 'Build and push image'
 uses: azure/docker-login@v1
 with:
 login-server: \${{ secrets.REGISTRY_LOGIN_SERVER }}
 username: \${{ secrets.REGISTRY_USERNAME }}
 password: \${{ secrets.REGISTRY_PASSWORD }}

- run: |
 docker build . -t \${{ secrets.REGISTRY_LOGIN_SERVER }}/20210325:v1
 docker push \${{ secrets.REGISTRY_LOGIN_SERVER }}/20210325:v1

```
- name: 'Deploy to Azure Container Instances'
  uses: 'azure/aci-deploy@v1'
  with:
    resource-group: ${ secrets.RESOURCE_GROUP }}
    dns-name-label: devops-20210325
    image: ${ secrets.REGISTRY_LOGIN_SERVER }}/20210325:v1
    registry-login-server: ${ secrets.REGISTRY_LOGIN_SERVER }}
    registry-username: ${ secrets.REGISTRY_USERNAME }}
    registry-password: ${ secrets.REGISTRY_PASSWORD }}
    environment-variables: API_KEY=${ secrets.API_KEY }}
    name: 20210325
    location: 'france central'
```

- **Secrets** : on peut cacher l'API key, identifiants efrei ainsi que openweather dans la partie secrets de GITHUB.

```
creds: ${ secrets.AZURE_CREDENTIALS }}
login-server: ${ secrets.REGISTRY_LOGIN_SERVER }}
username: ${ secrets.REGISTRY_USERNAME }}
password: ${ secrets.REGISTRY_PASSWORD }}
secure-environment-variables: API_KEY=${ secrets.API_KEY }}
name: ${ secrets.IDENTIFIANT_EFREI }}
```


Container instance :

Accueil > Abonnements > Efrei - Apprentis BDML >

 **20210325** ✕ ☆ ...
Instances de conteneur

▶ Démarrer ◀ Redémarrer □ Arrêter 🗑 Supprimer ↻ Actualiser

 Vue d'ensemble

 Journal d'activité

 Contrôle d'accès (IAM)

 Étiquettes

Paramètres

 Conteneurs

 Identité

 Propriétés

 Verrous

Supervision

 Métriques

 Alertes

^ Bases

[Vue JSON](#)

Groupe de ressources ([déplacer](#))

[devops-TP2](#)

Statut

En cours d'exécution

Emplacement

France Central

Abonnement ([déplacer](#))

[Efrei - Apprentis BDML](#)

ID d'abonnement

765266c6-9a23-4638-af32-dd1e32613047

Étiquettes ([modifier](#))

[Cliquez ici pour ajouter des étiquettes](#)

Type de système d'exploitation

Linux

Adresse IP (Public)

20.74.98.216

FQDN

devops-20210325.francecentral.azurecontainer.io

Nombre de conteneurs

1

Processeur

100

∞

Mémoire

100

∞

Résultat de l'appel de l'API :

```
anas@DESKTOP-IDM2KEV:/mnt/c/WINDOWS/system32$ curl "http://devops-20200828.francecentral.azurecontainer.io/?lat=5.902785&lon=102.754175"
{"data":{"base":"stations","clouds":{"all":98},"cod":200,"coord":{"lat":5.9028,"lon":102.7542},"dt":1655470985,"id":1736405,"main":{"feels_like":29.65,"grnd_level":983,"humidity":73,"pressure":1010,"sea_level":1010,"temp":27.29,"temp_max":27.29,"temp_min":27.29},"name":"Jertih","sys":{"country":"MY","sunrise":1655420157,"sunset":1655465023},"timezone":28800,"visibility":10000,"weather":[{"description":"overcast clouds","icon":"04n","id":804,"main":"Clouds"}],"wind":{"deg":136,"gust":4.55,"speed":4.06},"status":"ok"}}
```

Bonus :

- Données sensibles :

Aucune données sensibles est stockées dans l'image ou le code source.

- handolint :

name: Publish Docker image

on: push

jobs:

push_to_registry-linux:

name: Push Docker image to Docker Hub

runs-on: ubuntu-latest

steps:

- name: Check out the repo
uses: actions/checkout@v3

- name: 'Login via Azure CLI'
uses: azure/login@v1
with:
 creds: \${{ secrets.AZURE_CREDENTIALS }}

- name: hadolint
 uses: reviewdog/action-hadolint@v1
 with:
 reporter: github-pr-review

- name: 'Build and push image'
 uses: azure/docker-login@v1
 with:
 login-server: \${{ secrets.REGISTRY_LOGIN_SERVER }}
 username: \${{ secrets.REGISTRY_USERNAME }}
 password: \${{ secrets.REGISTRY_PASSWORD }}

- run: |
 docker build . -t \${{ secrets.REGISTRY_LOGIN_SERVER }}/20210325:v1
 docker push \${{ secrets.REGISTRY_LOGIN_SERVER }}/20210325:v1

```
- name: 'Deploy to Azure Container Instances'
  uses: 'azure/aci-deploy@v1'
  with:
    resource-group: ${ secrets.RESOURCE_GROUP }
    dns-name-label: devops-20210325
    image: ${ secrets.REGISTRY_LOGIN_SERVER }/20210325:v1
    registry-login-server: ${ secrets.REGISTRY_LOGIN_SERVER }
    registry-username: ${ secrets.REGISTRY_USERNAME }
    registry-password: ${ secrets.REGISTRY_PASSWORD }
    environment-variables: API_KEY=${ secrets.API_KEY }
    name: 20210325
    location: 'france central'
```