

Docker Flask App

Cycle de Projet :

1ere étape :

- Configurer le DockerFile
- Générer le pyc du python
- Faciliter le logging
- Installation du bibliothèques
- Changer en tant qu'user ce qui n'est pas le root

```
Dockerfile M X
frontapp > Dockerfile > ...
1 FROM python:3.8-slim-buster
2
3 EXPOSE 8080
4
5 ENV PYTHONDONTWRITEBYTECODE=1
6
7 ENV PYTHONUNBUFFERED=1
8
9 COPY requirements.txt .
10 RUN python -m pip install -r requirements.txt
11
12 WORKDIR /app
13 COPY . /app
14
15 RUN useradd appuser && chown -R appuser /app
16 USER appuser
17
18 CMD ["python", "app.py"]
19
```

- Configurer .dockerignore pour ignorer les fichiers unitiles.

```
.dockerignore X
app > .dockerignore
1  **/__pycache__
2  **/.classpath
3  **/.dockerignore
4  **/.env
5  **/.git
6  **/.gitignore
7  **/.project
8  **/.settings
9  **/.toolstarget
10 **/.vs
11 **/.vscode
12 **/*.proj.user
13 **/*.dbmdl
14 **/*.jfm
15 **/azds.yaml
16 **/bin
17 **/charts
18 **/docker-compose*
19 **/Dockerfile*
20 **/node_modules
21 **/npm-debug.log
22 **/obj
23 **/secrets.dev.yaml
24 **/values.dev.yaml
25 README.md
```

2eme étape :

Configurer le docker compose pour générer les commandes du dockerFile.

Dans lequel j'ai instancier le backend dans un conteneur et le frontend dans un autre l'un contient le port 5000 et l'autre 8080.

```
🐛 docker-compose.yml X
🐛 docker-compose.yml
1  version: '3'
2  services:
3    backend:
4      build:
5        context: ./app
6        dockerfile: ./Dockerfile
7      restart: always
8      ports:
9        - "5000:5000"
10
11    frontend:
12      build:
13        context: ./frontapp
14        dockerfile: ./Dockerfile
15      ports:
16        - "8080:8080"
```

Configurer le dock-compose.debug pour debugger les commandes.

```
🐛 docker-compose.debug.yml X
app > 🐛 docker-compose.debug.yml
1  version: '3.4'
2
3  services:
4    flask:
5      image: flask
6      build:
7        context: .
8        dockerfile: ./Dockerfile
9      command: ["sh", "-c", "pip install debugpy -t /tmp && python /tmp/debugpy --wait-f
10     ports:
11       - 5000:5000
12       - 5678:5678
13     environment:
14       - FLASK_APP=app\app.py
```

- Configurer le fichier requirements.txt

Pour vous assurer que les dépendances des applications sont transférées de votre environnement virtuel/machine hôte vers votre conteneur

```
≡ requirements.txt M X
app > ≡ requirements.txt
1  flask==1.1.2
2  librosa==0.6.0
3  numba==0.48
4  numpy
5  keras
6  tensorflow
7  scikit-learn
8  pillow
9  matplotlib
10
```

- Développer le unittests.py :

C'est une partie du Machine Learning

```
unittests.py 1 X
app > unittests.py > ...
1 import unittest
2 import app
3
4 BASE_URL = 'http://127.0.0.1:5000/'
5
6
7
8 class TestFlaskApi(unittest.TestCase):
9
10     def setUp(self):
11         self.app = app.app.test_client()
12         self.app.testing = True
13
14     def test_svm(self):
15         rv = self.app.get(BASE_URL+'uploader')
16         self.assertEqual(rv.status, '200 OK')
17
18     def test_vgg(self):
19
20         rv = self.app.get(BASE_URL+'uploadervgg')
21         self.assertEqual(rv.status, '200 OK')
22
23
24 if __name__ == '__main__':
25     import xmlrunner
26     runner = xmlrunner.XMLTestRunner(output='test-reports')
27     unittest.main(testRunner=runner)
```

3eme étape :

Développer le backend :

```
app.py 2 X
frontapp > app.py > ...
1  from flask import Flask
2  from flask import render_template
3  app=Flask(__name__, template_folder='templates')
4
5
6  @app.route('/upload')
7  def upload_files():
8      return render_template("form.html")
9
10
11  if __name__ == "__main__":
12      app.run(host="0.0.0.0", port=8080)
```

4eme étape :

Développer le frontend :

J'ai utilisé une requête 'POST' dans le chemin est <http://localhost:5000/uploadervgg> et une autre dans le chemin est <http://localhost:5000/uploader>

```
41 <div class="box">
42   <h1>Test VGG</h1>
43   <form action = "http://localhost:5000/uploadervgg" method = "POST"
44     enctype = "multipart/form-data">
45     <input type = "file" name = "file" />
46     <input class = "submit" type = "submit" />
47   </form>
48
49   <h1>Test SVM</h1>
50   <form action = "http://localhost:5000/uploader" method = "POST"
51     enctype = "multipart/form-data">
52     <input type = "file" name = "file" />
53     <input class = "submit" type = "submit" />
54   </form>
55 </div>
56 </body>
57 </html>
```

