

Projet Container Avec Docker Compose

PROJET_DOCKERCOMPOSE/

```
|
|
├── model_api/
|   ├── model/
|   |   └── model_wdbc.pkl # Le modèle pré-entraîné
|   ├── Dockerfile
|   ├── app.py           # Script Flask pour l'API
|   └── requirements.txt # Dépendances Python
|
├── ui/
|   ├── src/
|   |   ├── index.html # Page HTML pour l'UI
|   |   ├── script.js  # Script JavaScript pour interagir avec l'API
|   |   └── style.css   # Feuille de style CSS (optionnelle)
|   └── Dockerfile      # Dockerfile pour l'UI, utilisant un serveur web comme Nginx
|
└── docker-compose.yml  # Docker Compose pour orchestrer les conteneurs
```

Ce projet vise à développer et déployer une application Web permettant la prédiction du type de cancer (basé sur le jeu de données du cancer du sein de Wisconsin) à l'aide d'un modèle de Machine Learning.

L'application est conçue autour d'une architecture de microservices utilisant des conteneurs Docker pour isoler les composants, facilitant ainsi leur déploiement, leur scalabilité et leur maintenance.

L'architecture se compose de deux services principaux : un service d'API Flask fournissant l'interface de prédiction du modèle et **un service d'interface utilisateur (UI)** servant une page web statique pour interagir avec l'API.

Architecture du projet

Conteneurs Docker

Conteneur API Flask (model_api) : Exécute une application Flask qui charge un modèle pré-entraîné de Machine Learning pour prédire le type de cancer basé sur des entrées utilisateur. L'API accepte les requêtes POST contenant les données de prédiction et renvoie le résultat de la prédiction.

Conteneur UI (ui) : Héberge une page web statique accessible via un navigateur web. Cette interface permet aux utilisateurs de saisir les caractéristiques nécessaires à la prédiction du cancer et d'afficher les résultats renvoyés par l'API Flask.

Réseau Docker

Les deux conteneurs sont connectés via un réseau Docker personnalisé (mon-reseau), permettant ainsi une communication interne sécurisée entre l'interface utilisateur et l'API Flask sans exposer l'API au réseau externe.

```
C:\Users\tarak>docker network ls
```

NETWORK ID	NAME	DRIVER	SCOPE
f6b6660c9353	bridge	bridge	local
8bf9425a9eda	docker_compose_wdbc_default	bridge	local
f4366a0870df	host	host	local
cf302b748753	none	null	local
6353e210ad92	projet_default	bridge	local

Détails des fichiers et fonctionnalités

API Flask (model_api)

Dockerfile : Définit l'image Docker de l'API Flask, spécifiant l'environnement Python, les dépendances (requirements.txt), et la commande pour exécuter l'application Flask (app.py).

requirements.txt : Liste toutes les bibliothèques Python nécessaires à l'exécution de l'application Flask, y compris flask, flask_cors, joblib, et scikit-learn.

app.py : Le script principal de l'API Flask. Charge le modèle de Machine Learning et définit un endpoint /predict qui traite les requêtes POST, effectue des prédictions à l'aide du modèle, et renvoie les résultats.

Interface Utilisateur (ui)

Dockerfile : Construit une image Docker basée sur Nginx pour servir les fichiers statiques de l'interface utilisateur, copiant les fichiers depuis le répertoire local vers le conteneur.

index.html : La page principale de l'interface utilisateur, contenant le formulaire pour saisir les caractéristiques nécessaires à la prédiction et afficher les résultats.

script.js : Script JavaScript gérant l'envoi des données du formulaire à l'API Flask via une requête AJAX et l'affichage des résultats de prédiction.

style.css : Feuille de style CSS pour améliorer l'apparence de l'interface utilisateur.

Communication et déploiement

`docker-compose.yml` : Fichier de configuration Docker Compose décrivant les services, les configurations réseau, et les volumes nécessaires pour déployer l'application. Facilite le démarrage simultané des conteneurs API et UI avec la commande `docker-compose up`.

Résumé

Ce projet illustre l'application de concepts avancés d'informatique et de génie logiciel, y compris le développement d'applications basées sur des microservices, l'utilisation de conteneurs pour l'isolation et la portabilité des services, et l'intégration de modèles de Machine Learning dans des applications web. En utilisant Docker et Docker Compose, ce projet démontre comment déployer efficacement des applications complexes, en assurant une communication sécurisée entre les services internes et en simplifiant le processus de déploiement et de scalabilité. Ce système permet une approche modulaire du développement de logiciels, où chaque service peut être développé, testé, et déployé indépendamment, tout en contribuant à la fonctionnalité globale de l'application.

Projet Container Avec Vagrant

Pour intégrer mes conteneurs Docker existants (l'API Flask et l'interface utilisateur) dans un environnement Vagrant :

1.Préparation de l'environnement

Installation de Vagrant . : vagrant --version

Installation VirtualBox .

2. Préparation du Vagrantfile

Dans le répertoire racine de mon projet, où se trouvent les répertoires model_api et ui, J'ai créée **un Vagrantfile** pour configurer mon environnement de développement virtuel avec Vagrant.

projet/

|

| **└─ api/**

| | **└─ Dockerfile**

| | **└─ app.py**

| | **└─ requirements.txt**

|

| **└─ ui/**

| | **└─ Dockerfile**

| | **└─ index.html**

| | **└─ script.js**

| | **└─ style.css**

|

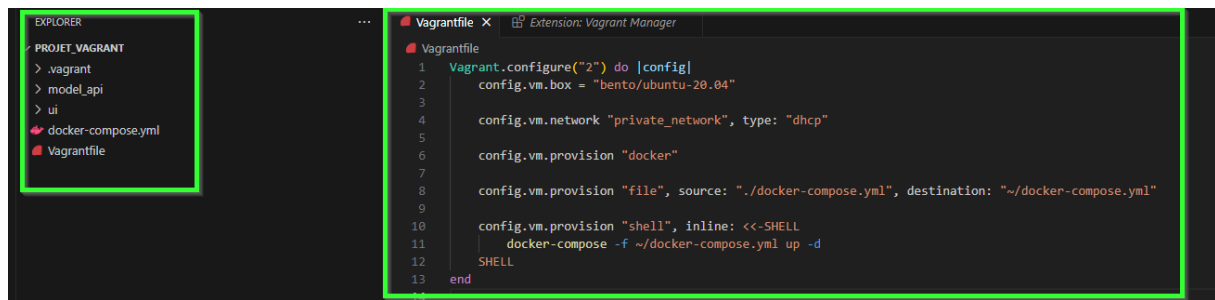
| **└─ docker-compose.yml**

|

| **└─ Vagrantfile**

3.Configuration du projet

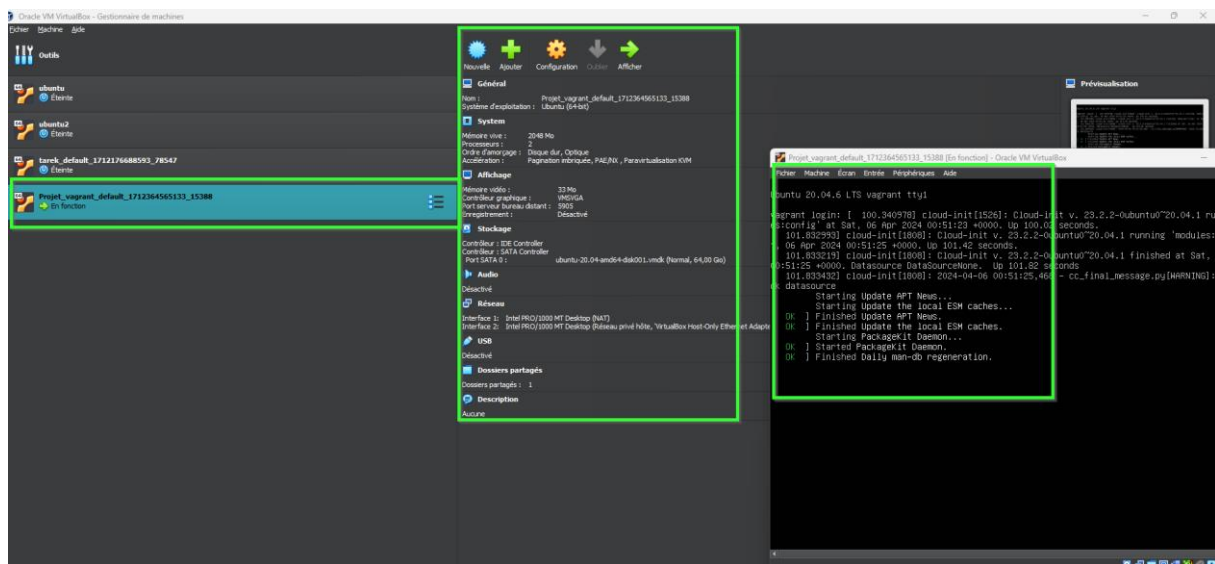
Placer mon docker-compose.yml, Vagrantfile et les dossiers api/ et ui/ dans un répertoire de projet avec Visual Studio Code.



4. Exécution de Vagrant

vagrant up

Cette commande initialisera et configurera la machine virtuelle spécifiée dans le Vagrantfile, y compris l'installation de Docker et le démarrage des conteneurs Docker via docker-compose.



5. Accès à la machine virtuelle

vagrant ssh

Cette commande me connectera à la machine virtuelle où Docker est en cours d'exécution avec mes conteneurs.

6. Ajustement des paramètres de Dockerfile

les conteneurs n'ont pas été démarrés automatiquement, je les démarre manuellement.

docker-compose.yml doit être présent dans la machine Vagrant, puis démarre les conteneurs en utilisant Docker Compose :

docker-compose up -d

Vérifier les logs de Docker
docker-compose logs

Changer la disposition du clavier de façon permanente :sur dockerfile

```
config.vm.provision "shell", inline: <<-SHELL
  echo "setxkbmap fr" >> /home/vagrant/.profile
SHELL
```

Installez Docker Compose

```
sudo curl -L "https://github.com/docker/compose/releases/download/1.29.2/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose
```

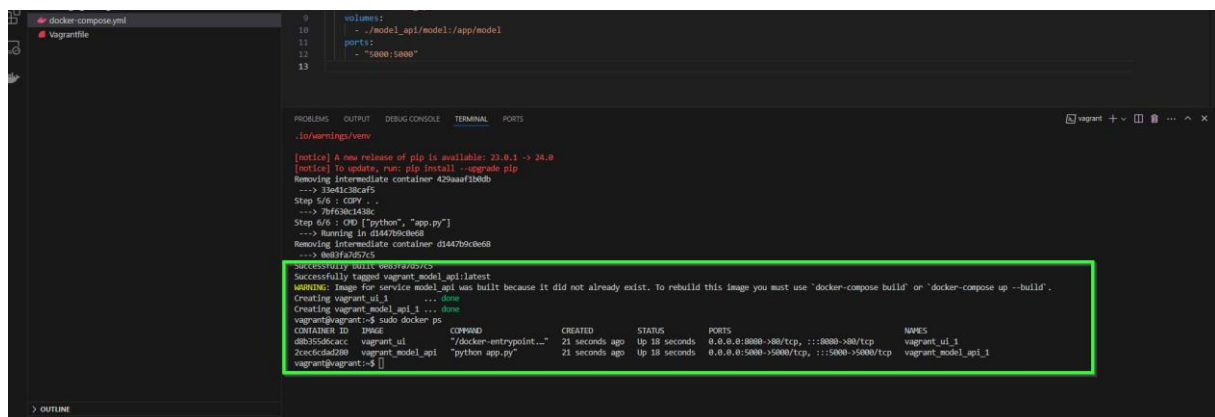
docker-compose --version

docker-compose up -d

pas d'accès aux pages servies par mes conteneurs Docker via

http://localhost:5000

<http://localhost:8080>



```
docker-compose.yml
Vagrantfile

volumes:
  - /model_api/model:/app/model
ports:
  - "5000:5000"

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
./warnings/vowm
[notice] A new release of pip is available: 23.0.1 -> 24.0
[notice] To update, run: pip install --upgrade pip
Removing intermediate container 429aaf2bdeb
--> 3be4130c9f5
Step 5/6 : COPY . .
--> 7bf30c1438c
Step 6/6 : CMD ["python", "app.py"]
--> Running in d14476c8ede8
Removing intermediate container d14476c8ede8
--> d63f48251c5
Successfully built vagrant_model_api:latest
WARNING: Image for service vagrant_model_api was built because it did not already exist. To rebuild this image you must use 'docker-compose build' or 'docker-compose up --build'.
Creating vagrant_ui_1 ... done
Creating vagrant_model_api_1 ... done
vagrant@vagrant:~$ sudo docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
d8b355d5cacc   vagrant_ui     "/docker-entrypoint..." 21 seconds ago Up 18 seconds 0.0.0.0:8080->80/tcp             vagrant_ui_1
20c4d4d4d0b   vagrant_model_api  "python app.py"          21 seconds ago Up 18 seconds 0.0.0.0:5000->5000/tcp           vagrant_model_api_1
vagrant@vagrant:~$
```

config.vm.network "forwarded_port", guest: 5000, host: 5000

config.vm.network "forwarded_port", guest: 8080, host: 8080

PS C:\Users\tarak\Paris-8\Projet_container\Projet_vagrant> vagrant ssh

vagrant@127.0.0.1's password:

Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-162-generic x86_64)

- * Documentation: <https://help.ubuntu.com>
- * Management: <https://landscape.canonical.com>
- * Support: <https://ubuntu.com/advantage>

System information as of Sat 06 Apr 2024 02:53:34 AM UTC

System load: 0.16 Users logged in: 1

Usage of /: 13.2% of 30.34GB IPv4 address for docker0: 172.17.0.1

Memory usage: 13% IPv4 address for eth0: 10.0.2.15
Swap usage: 0% IPv4 address for eth1: 192.168.56.15
Processes: 159

This system is built by the Bento project by Chef Software
More information can be found at <https://github.com/chef/bento>
Last login: Sat Apr 6 02:51:24 2024

vagrant@vagrant:~\$ ls

docker-compose.yml model_api ui Vagrantfile

vagrant@vagrant:~\$ docker ps

permission denied while trying to connect to the Docker daemon socket at
unix:///var/run/docker.sock: Get "http://%2Fvar%2Frun%2Fdocker.sock/v1.24/containers/json":
dial unix /var/run/docker.sock: connect: permission denied

vagrant@vagrant:~\$ sudo docker ps

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
--------------	-------	---------	---------	--------	-------	-------

vagrant@vagrant:~\$ docker-compose up -d

Traceback (most recent call last):

File "urllib3/connectionpool.py", line 677, in urlopen

File "urllib3/connectionpool.py", line 392, in _make_request

File "http/client.py", line 1277, in request

File "http/client.py", line 1323, in _send_request

File "http/client.py", line 1272, in endheaders

File "http/client.py", line 1032, in _send_output

File "http/client.py", line 972, in send

File "docker/transport/unixconn.py", line 43, in connect

PermissionError: [Errno 13] Permission denied

During handling of the above exception, another exception occurred:

Traceback (most recent call last):

File "requests/adapters.py", line 449, in send

File "urllib3/connectionpool.py", line 727, in urlopen

File "urllib3/util/retry.py", line 410, in increment

File "urllib3/packages/six.py", line 734, in reraise

File "urllib3/connectionpool.py", line 677, in urlopen

File "urllib3/connectionpool.py", line 392, in _make_request

File "http/client.py", line 1277, in request

File "http/client.py", line 1323, in _send_request

File "http/client.py", line 1272, in endheaders

File "http/client.py", line 1032, in _send_output

File "http/client.py", line 972, in send

File "docker/transport/unixconn.py", line 43, in connect

urllib3.exceptions.ProtocolError: ('Connection aborted.', PermissionError(13, 'Permission denied'))

During handling of the above exception, another exception occurred:

Traceback (most recent call last):

File "docker/api/client.py", line 214, in _retrieve_server_version

File "docker/api/daemon.py", line 181, in version

File "docker/utils/decorators.py", line 46, in inner

```
File "docker/api/client.py", line 237, in _get
File "requests/sessions.py", line 543, in get
File "requests/sessions.py", line 530, in request
File "requests/sessions.py", line 643, in send
File "requests/adapters.py", line 498, in send
requests.exceptions.ConnectionError: ('Connection aborted.', PermissionError(13, 'Permission
denied'))
```

During handling of the above exception, another exception occurred:

Traceback (most recent call last):

```
File "docker-compose", line 3, in <module>
File "compose/cli/main.py", line 81, in main
File "compose/cli/main.py", line 200, in perform_command
File "compose/cli/command.py", line 70, in project_from_options
File "compose/cli/command.py", line 153, in get_project
File "compose/cli/docker_client.py", line 43, in get_client
File "compose/cli/docker_client.py", line 170, in docker_client
File "docker/api/client.py", line 197, in __init__
File "docker/api/client.py", line 222, in _retrieve_server_version
docker.errors.DockerException: Error while fetching server API version: ('Connection aborted.',
PermissionError(13, 'Permission denied'))
[4430] Failed to execute script docker-compose
vagrant@vagrant:~$ sudo docker-compose up -d
Creating network "vagrant_default" with the default driver
Building ui
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/
```

```
Sending build context to Docker daemon 10.24kB
Step 1/2 : FROM nginx:stable-alpine
stable-alpine: Pulling from library/nginx
3c854c8cbf46: Pull complete
de5d475193dd: Pull complete
b407bcc80638: Pull complete
da33b1ad0ac4: Pull complete
a0fbd691d7c1: Pull complete
16eaaaf5f1c0: Pull complete
5e845cc16269: Pull complete
Digest: sha256:8f62e8ffc22a112ab3aeb56f56b9ea3e2561248dee1d8cb72c5d6462a7789b5e
Status: Downloaded newer image for nginx:stable-alpine
---> 249f59e1dec7
Step 2/2 : COPY src /usr/share/nginx/html
---> 8639a7262577
Successfully built 8639a7262577
Successfully tagged vagrant_ui:latest
WARNING: Image for service ui was built because it did not already exist. To rebuild this image
you must use `docker-compose build` or `docker-compose up --build`.
Building model_api
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
```


<https://docs.docker.com/go/buildx/>

Sending build context to Docker daemon 519.7kB

Step 1/6 : FROM python:3.8-slim

3.8-slim: Pulling from library/python

8a1e25ce7c4f: Pull complete

1103112ebfc4: Pull complete

93d3f6d14ae5: Pull complete

46996c1c5ef3: Pull complete

18dacb59e6d3: Pull complete

Digest: sha256:72ae14e80c21f274f31111debd505d8fa64536fdf41b57f03930b3baf84d8b8d

Status: Downloaded newer image for python:3.8-slim

---> 04977f08feb1

Step 2/6 : WORKDIR /app

---> Running in 7a03c53aad8f

Removing intermediate container 7a03c53aad8f

---> c7da8979e090

Step 3/6 : COPY requirements.txt .

---> 3908a4b03e48

Step 4/6 : RUN pip install -r requirements.txt

---> Running in 429aaaf1b0db

Collecting flask

Downloading flask-3.0.2-py3-none-any.whl (101 kB)

101.3/101.3 kB 4.2 MB/s eta 0:00:00

Collecting flask_cors

Downloading Flask_Cors-4.0.0-py2.py3-none-any.whl (14 kB)

Collecting joblib

Downloading joblib-1.3.2-py3-none-any.whl (302 kB)

302.2/302.2 kB 9.6 MB/s eta 0:00:00

Collecting scikit-learn==1.3.0

Downloading scikit_learn-1.3.0-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (11.1 MB)

11.1/11.1 MB 25.7 MB/s eta 0:00:00

Collecting threadpoolctl>=2.0.0

Downloading threadpoolctl-3.4.0-py3-none-any.whl (17 kB)

Collecting numpy>=1.17.3

Downloading numpy-1.24.4-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (17.3 MB)

17.3/17.3 MB 19.9 MB/s eta 0:00:00

Collecting scipy>=1.5.0

Downloading scipy-1.10.1-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (34.5 MB)

34.5/34.5 MB 11.4 MB/s eta 0:00:00

Collecting Jinja2>=3.1.2

Downloading Jinja2-3.1.3-py3-none-any.whl (133 kB)

133.2/133.2 kB 19.3 MB/s eta 0:00:00

Collecting click>=8.1.3

Downloading click-8.1.7-py3-none-any.whl (97 kB)

97.9/97.9 kB 11.8 MB/s eta 0:00:00

Collecting Werkzeug>=3.0.0

Downloading werkzeug-3.0.2-py3-none-any.whl (226 kB)

226.8/226.8 kB 20.8 MB/s eta 0:00:00

Collecting blinker>=1.6.2

Downloading blinker-1.7.0-py3-none-any.whl (13 kB)

Collecting itsdangerous>=2.1.2

Downloading itsdangerous-2.1.2-py3-none-any.whl (15 kB)

Collecting importlib-metadata>=3.6.0

Downloading importlib_metadata-7.1.0-py3-none-any.whl (24 kB)

Collecting zipp>=0.5

Downloading zipp-3.18.1-py3-none-any.whl (8.2 kB)

Collecting MarkupSafe>=2.0

Downloading MarkupSafe-2.1.5-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (26 kB)

Installing collected packages: zipp, threadpoolctl, numpy, MarkupSafe, joblib, itsdangerous, click, blinker, Werkzeug, scipy, Jinja2, importlib-metadata, scikit-learn, flask, flask_cors

Successfully installed Jinja2-3.1.3 MarkupSafe-2.1.5 Werkzeug-3.0.2 blinker-1.7.0 click-8.1.7 flask-3.0.2 flask_cors-4.0.0 importlib-metadata-7.1.0 itsdangerous-2.1.2 joblib-1.3.2 numpy-1.24.4 scikit-learn-1.3.0 scipy-1.10.1 threadpoolctl-3.4.0 zipp-3.18.1

WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager. It is recommended to use a virtual environment instead: <https://pip.pypa.io/warnings/venv>

[notice] A new release of pip is available: 23.0.1 -> 24.0

[notice] To update, run: pip install --upgrade pip

Removing intermediate container 429aaaf1b0db

---> 33e41c38caf5

Step 5/6 : COPY . .

---> 7bf630c1438c

Step 6/6 : CMD ["python", "app.py"]

---> Running in d1447b9c0e68

Removing intermediate container d1447b9c0e68

---> 0e83fa7d57c5

Successfully built 0e83fa7d57c5

Successfully tagged vagrant_model_api:latest

WARNING: Image for service model_api was built because it did not already exist. To rebuild this image you must use `docker-compose build` or `docker-compose up --build`.

Creating vagrant_ui_1 ... done

Creating vagrant_model_api_1 ... done

vagrant@vagrant:~\$ sudo docker ps

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
d8b355d6cacc	vagrant_ui	"/docker-entrypoint...."	21 seconds ago	Up	18 seconds
0.0.0.0:8080->80/tcp	:::8080->80/tcp	vagrant_ui_1			

```
2cec6cdad280 vagrant_model_api "python app.py"    21 seconds ago Up 18 seconds
0.0.0.0:5000->5000/tcp, :::5000->5000/tcp vagrant_model_api_1
vagrant@vagrant:~$
```

7.Test

```
docker exec -it vagrant_model_api_1 /bin/sh
```

```
curl http://localhost:5000
```

```
docker exec -it vagrant_ui_1 /bin/sh
```

À l'intérieur du conteneur, je vérifie les processus et utilisez curl pour vous-même :

```
curl http://localhost:80
```

```
vagrant@vagrant:~$ sudo docker exec -it vagrant_ui_1 /bin/sh
```

```
/ # ps
```

```
PID USER TIME COMMAND
```

```
1 root 0:00 nginx: master process nginx -g daemon off;
```

```
29 nginx 0:00 nginx: worker process
```

```
30 nginx 0:00 nginx: worker process
```

```
31 root 0:00 /bin/sh
```

```
36 root 0:00 ps
```

```
/ #
```

```
/ # curl http://localhost:80
```

```
<!DOCTYPE html>
```

```
<html lang="en">
```

```
<head>
```

```
<meta charset="UTF-8">
```

```
<title>Cancer Prediction after PCA</title>
```

```
<link rel="stylesheet" href="style.css">
```

```
</head>
```

```
<body>
```

```
<div class="container">
```

```
<h1>Cancer Prediction</h1>
```

```
<p>Enter the PCA features values to predict the cancer type.</p>
```

```
<form id="prediction-form">
```

```
<div class="form-group">
```

```
<label for="feature1">Feature 1:</label>
```

```
<input type="number" id="feature1" class="feature-input" step="0.01" required>
```

```
</div>
```

```
<div class="form-group">
```

```
<label for="feature2">Feature 2:</label>
```

```
<input type="number" id="feature2" class="feature-input" step="0.01" required>
```

```
</div>
```

```
<div class="form-group">
```

```
<label for="feature3">Feature 3:</label>
```

```
<input type="number" id="feature3" class="feature-input" step="0.01" required>
```

```
</div>
```

```
<div class="form-group">
```

```
<label for="feature4">Feature 4:</label>
```

```
<input type="number" id="feature4" class="feature-input" step="0.01" required>
```

```
</div>
```

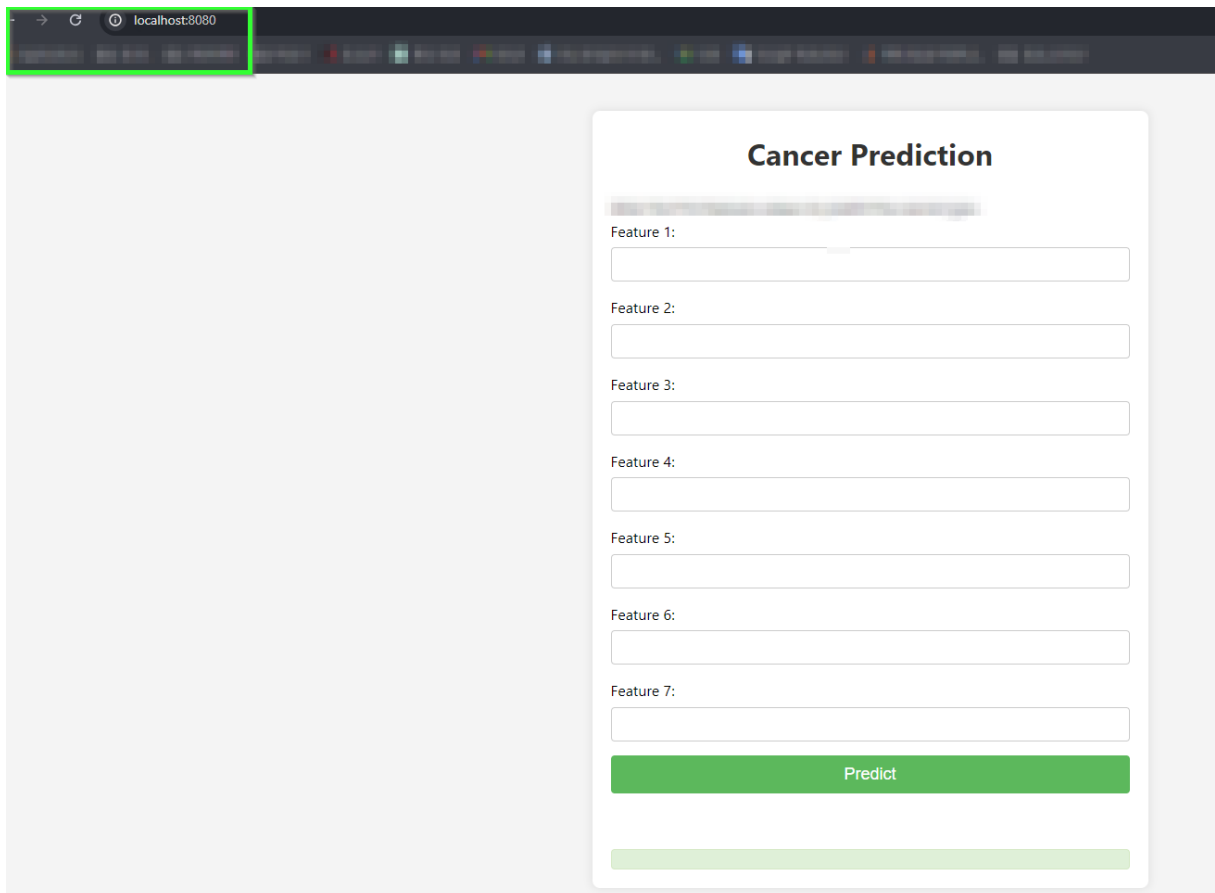
```

<div class="form-group">
  <label for="feature5">Feature 5:</label>
  <input type="number" id="feature5" class="feature-input" step="0.01" required>
</div>
<div class="form-group">
  <label for="feature6">Feature 6:</label>
  <input type="number" id="feature6" class="feature-input" step="0.01" required>
</div>
<div class="form-group">
  <label for="feature7">Feature 7:</label>
  <input type="number" id="feature7" class="feature-input" step="0.01" required>
</div>
<button type="submit" id="predict-button">Predict</button>
</form>

<div id="status-message" class="status-message"></div>
<div id="result" class="result"></div>
</div>

<script src="script.js"></script>
</body>
</html>
/ #

```



The screenshot shows a web browser window with the address bar displaying 'localhost:8080'. The main content area features a 'Cancer Prediction' form. The form is titled 'Cancer Prediction' in bold black text. Below the title, there are seven input fields, each preceded by a label 'Feature 1:' through 'Feature 7:'. The input fields are empty. At the bottom of the form, there is a green button labeled 'Predict'. Below the button, there is a light green rectangular box, likely intended for the prediction result.

Le but du projet est de créer un environnement isolé et reproductible pour le développement et le test d'une application web divisée en deux principaux composants :

Interface Utilisateur (UI) : Un front-end web pour interagir avec l'utilisateur.

API (model_api) : Un back-end Flask fournissant une API, pour traiter des données et retourner des résultats, de prédictions de modèles.

Architecture du Projet

ui/ : Contient les fichiers nécessaires à l'interface utilisateur, comme index.html, style.css, et script.js.

model_api/ : Contient les fichiers du serveur Flask, y compris app.py et les dépendances Python dans requirements.txt.

docker-compose.yml : Un fichier qui définit comment les services Docker (UI et API) doivent être construits et lancés, spécifiant les images, les ports, les volumes, etc.

Vagrantfile : Un fichier de configuration pour Vagrant qui définit comment la machine virtuelle est créée, configurée et comment elle interagit avec les conteneurs Docker.

Environnement de Développement et Outils

Vagrant : Utilisé pour créer et gérer une machine virtuelle (VM) qui fournit un environnement isolé et contrôlé pour le développement. La VM est basée sur Ubuntu 20.04.

Docker : Utilisé à l'intérieur de la VM pour contenir et exécuter l'application en deux parties : l'interface utilisateur et l'API.

Docker Compose : Utilisé pour orchestrer le lancement des conteneurs Docker basés sur les configurations définies dans docker-compose.yml.

Processus de Configuration et d'Exécution

Configuration initiale :

Le Vagrantfile est configuré pour provisionner la VM avec Docker et Docker Compose, en plus de configurer les ports et les dossiers partagés.

docker-compose.yml est configuré pour définir comment les conteneurs Docker pour l'UI et l'API doivent être construits et exécutés.

Lancement de l'environnement :

La commande vagrant up est utilisée pour démarrer la VM. Vagrant lit le Vagrantfile pour créer et configurer la VM selon les spécifications.

Des scripts dans le Vagrantfile installent Docker et Docker Compose dans la VM, et préparent l'environnement Linux (comme la configuration du clavier en AZERTY).

Déploiement des applications :

Une fois dans la VM (via vagrant ssh), docker-compose up -d est utilisé pour construire et lancer les conteneurs Docker pour l'UI et l'API basés sur docker-compose.yml.

Accès et Test :

Les services sont accessibles via les ports mappés spécifiés dans le Vagrantfile et docker-compose.yml, permettant l'accès à l'UI et à l'API depuis le navigateur de la machine hôte.

Conclusion

Ce projet utilise une combinaison de Vagrant, Docker, et Docker Compose pour créer un environnement de développement isolé et facilement reproductible, assurant que l'application fonctionne de manière cohérente sur toutes les machines et facilitant le partage entre les développeurs.

version-image

le répertoire nommé version-image qui contient un fichier docker-compose.yml pour gérer vos conteneurs comme des images prédéfinies (sans processus de build)

```
mkdir version-image
cd version-image
```

Préparation du fichier docker-compose.yml

Dans le répertoire version-image, je veux créer un fichier docker-compose.yml qui spécifie les images à utiliser pour mes conteneurs.

Mais en avant je déjà fait un push de mes images sur docker hub

```
PS C:\Users\tarak\Paris-8\Projet_container\docker_compose_wdbc\ui> docker build -t darza/ui_image:V1 .
```

```
[+] Building 1.2s (8/8) FINISHED
=> [internal] load build definition from Dockerfile                                0.0s
=> => transferring dockerfile: 95B                                                0.0s
=> [internal] load metadata for docker.io/library/nginx:stable-alpine            1.1s
=> [auth] library/nginx:pull token for registry-1.docker.io                     0.0s
=> [internal] load .dockerignore                                                  0.0s
=> => transferring context: 2B                                                    0.0s
=> [internal] load build context                                                  0.0s
=> => transferring context: 2.26kB                                                0.0s
=> CACHED [1/2] FROM docker.io/library/nginx:stable-
alpine@sha256:8f62e8ffc22a112ab3aeb56f56b9ea3e2561248dee1d8c 0.0s
=> [2/2] COPY src /usr/share/nginx/html                                          0.0s
=> exporting to image                                                            0.0s
=> => exporting layers                                                            0.0s
=> => writing image
sha256:6ce00be4443f868a2dbbd5ccd0254f5cfbdfc9d12d40fb4933f31d02a21acaf7
0.0s
=> => naming to docker.io/darza/ui_image:V1                                     0.0s
```

View build details: [docker-
desktop://dashboard/build/default/default/f7amygkbmigj3ojq68535w4i1](https://desktop.docker.com/win/en/desktop://dashboard/build/default/default/f7amygkbmigj3ojq68535w4i1)

What's Next?

View a summary of image vulnerabilities and recommendations → [docker scout quickview](#)

```
PS C:\Users\tarak\Paris-8\Projet_container\docker_compose_wdbc\ui> cd ..
```

```
PS C:\Users\tarak\Paris-8\Projet_container\docker_compose_wdbc> dir
```

Répertoire : C:\Users\tarak\Paris-8\Projet_container\docker_compose_wdbc

Mode	LastWriteTime	Length	Name
----	-----	-----	

```
d----- 06/04/2024 01:22      model_api
d----- 06/04/2024 01:19      ui
-a----- 06/04/2024 01:29      203 docker-compose.yml
```

```
PS C:\Users\tarak\Paris-8\Projet_container\docker_compose_wdbc> cd .\model_api\
PS C:\Users\tarak\Paris-8\Projet_container\docker_compose_wdbc\model_api> docker
build -t darza/api_image:V1 .
```

```
[+] Building 1.2s (11/11) FINISHED                                docker:default
=> [internal] load build definition from Dockerfile                0.0s
=> => transferring dockerfile: 173B                                0.0s
=> [internal] load metadata for docker.io/library/python:3.8-slim 1.1s
=> [auth] library/python:pull token for registry-1.docker.io      0.0s
=> [internal] load .dockerignore                                  0.0s
=> => transferring context: 2B                                     0.0s
=> [1/5] FROM docker.io/library/python:3.8-
slim@sha256:72ae14e80c21f274f31111debd505d8fa64536fdf41b57f03930b3baf 0.0s
=> [internal] load build context                                  0.0s
=> => transferring context: 157B                                   0.0s
=> CACHED [2/5] WORKDIR /app                                     0.0s
=> CACHED [3/5] COPY requirements.txt .                           0.0s
=> CACHED [4/5] RUN pip install -r requirements.txt              0.0s
=> CACHED [5/5] COPY . .                                         0.0s
=> exporting to image                                           0.0s
=> => exporting layers                                           0.0s
=> => writing image
sha256:20898654954f98940e02fd7527d101bcd78d94f9543c2158a6eab5feff5beb6e
0.0s
=> => naming to docker.io/darza/api_image:V1                    0.0s
```

View build details: [docker-
desktop://dashboard/build/default/default/9bojy85ayhdaxhqv7r6zzhfqb](https://desktop.docker.com/win/en/desktop://dashboard/build/default/default/9bojy85ayhdaxhqv7r6zzhfqb)

What's Next?

View a summary of image vulnerabilities and recommendations → [docker scout quickview](#)

```
PS C:\Users\tarak\Paris-8\Projet_container\docker_compose_wdbc\model_api> docker
images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
darza/ui_image	V1	6ce00be4443f	4 minutes ago	41.1MB
docker_compose_wdbc-ui	latest	7969afe21337	4 hours ago	41.1MB
darza/api_image	V1	20898654954f	4 hours ago	436MB
docker_compose_wdbc-model_api	latest	4378c38fcd93	4 hours ago	436MB

```
PS C:\Users\tarak\Paris-8\Projet_container\docker_compose_wdbc\model_api> cd ..
PS C:\Users\tarak\Paris-8\Projet_container\docker_compose_wdbc> cd ..
```


darza

Search by repository name

All Content

Create repository

darza / api_image

Contains: Image • Last pushed: 4 minutes ago

Security unknown

0

2

Public

darza / ui_image

Contains: Image • Last pushed: 8 minutes ago

Security unknown

0

2

Public

docker-compose.yml

version: '3'
services:
 ui:
 image: darza/ui_image:V1
 ports:
 - "8080:80"
 api:
 image: darza/api_image:V1
 ports:
 - "5000:5000"

PS C:\Users\tarak\Paris-8\Projet_container\version-image> docker-compose up -d

[+] Running 2/3
- Network version-image_default Created 0.6s
✓ Container version-image-ui-1 Started 0.6s
✓ Container version-image-api-1 Started 0.5s
PS C:\Users\tarak\Paris-8\Projet_container\version-image> docker ps

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
8c2cd9d6c867	darza/api_image:V1	"python app.py"	8 seconds ago	Up 7 seconds		version-image-api-1
0.0.0.0:5000->5000/tcp						
7aa94ab3e302	darza/ui_image:V1	"/docker-entrypoint...."	8 seconds ago	Up 7 seconds		version-image-ui-1
0.0.0.0:8080->80/tcp						

PS C:\Users\tarak\Paris-8\Projet_container\version-image>

```
PS C:\Users\tarak\Paris-8\Projet_container\version-image> docker pull darza/api_image:V1
V1: Pulling from darza/api_image
Digest: sha256:4f5188743a5e2513d0fb4d6a9281877a3ff7af19b2492a489705af00f3b86c08
Status: Image is up to date for darza/api_image:V1
docker.io/darza/api_image:V1

What's Next?
View a summary of image vulnerabilities and recommendations → docker scout quickview darza/api_image:V1
PS C:\Users\tarak\Paris-8\Projet_container\version-image> docker pull darza/ui_image:V1
V1: Pulling from darza/ui_image
Digest: sha256:ad9e7c8f72bd00d0bbdbec89a574668c75f2fac54d5f51a0203977fc6a50cfa
Status: Image is up to date for darza/ui_image:V1
docker.io/darza/ui_image:V1

What's Next?
View a summary of image vulnerabilities and recommendations → docker scout quickview darza/ui_image:V1
PS C:\Users\tarak\Paris-8\Projet_container\version-image> docker-compose up -d
[+] Running 2/3
- Network version-image_default Created 0.6s
✓ Container version-image-ui-1 Started 0.6s
✓ Container version-image-api-1 Started 0.5s
PS C:\Users\tarak\Paris-8\Projet_container\version-image> docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
8c2cd9d6c867   darza/api_image:V1   "python app.py"         8 seconds ago Up 7 seconds   0.0.0.0:5000->5000/tcp             version-image-api-1
7aa94ab3e302   darza/ui_image:V1    "/docker-entrypoint..." 8 seconds ago Up 7 seconds   0.0.0.0:8080->80/tcp              version-image-ui-1
PS C:\Users\tarak\Paris-8\Projet_container\version-image>
```

version-swarm

Pour utiliser Docker Swarm d'abord initialiser votre essaim (Swarm), ce qui transforme mon Docker en un environnement Swarm, permettant de déployer des services de manière distribuée et managée à travers plusieurs nœuds Docker, qui sont des instances de Docker Engine dans Swarm mode.

Pour déployer mon application en utilisant Docker Swarm:

1. Initialiser Docker Swarm:

```
C:\Users\tarak>docker swarm init
Swarm initialized: current node (ikd8iwq9zuva1bx2bfjl8qgf6) is now a manager.

To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-34ctdwzhi86g36jpaw8b5e9un1r4o2vwlzu1jialhexqq5tyvj-a7y8da2w4pcu5a8y9rns6y8 192.168.65.3:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.
```

C:\Users\tarak>docker swarm init

Swarm initialized: current node (ikd8iwq9zuva1bx2bfjl8qgf6) is now a manager.

To add a worker to this swarm, run the following command:

```
docker swarm join --token SWMTKN-1-
34ctdwzhi86g36jpaw8b5e9un1r4o2vwlzu1jialhexqq5tyvj-a7y8da2w4pcu5a8y9rns6y8
192.168.65.3:2377
```

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.

C:\Users\tarak>cd C:\Users\tarak\Paris-8\Projet_container\version-image

C:\Users\tarak\Paris-8\Projet_container\version-image>cd ..

C:\Users\tarak\Paris-8\Projet_container>dir

Le volume dans le lecteur C s'appelle OSDisk
Le numéro de série du volume est 8A86-7C3E

Répertoire de C:\Users\tarak\Paris-8\Projet_container

```
06/04/2024 06:34 <DIR>      .
06/04/2024 02:19 <DIR>      ..
06/04/2024 01:23 <DIR>      docker_compose_wdbc
06/04/2024 05:37      241 658 Projet Container.docx
06/04/2024 05:24 <DIR>      Projet_vagrant
06/04/2024 06:05 <DIR>      version-image
06/04/2024 06:34 <DIR>      version_swarm
          1 fichier(s)      241 658 octets
          6 Rép(s) 24 892 612 608 octets libres
```

C:\Users\tarak\Paris-8\Projet_container>cd version_swarm

C:\Users\tarak\Paris-8\Projet_container\version_swarm>docker stack deploy -c docker-compose.yml mystack

```
C:\Users\tarak\Paris-8\Projet_container\version_swarm>docker stack deploy -c docker-compose.yml mystack
```

Creating network mystack_default

Creating service mystack_api

Creating service mystack_ui

```
C:\Users\tarak\Paris-8\Projet_container\version_swarm>docker service ls
```

ID	NAME	MODE	REPLICAS	IMAGE	PORTS
slqzyotq5avt	mystack_api	replicated	2/2	darza/api_image:V1	*:5000->5000/tcp
w1vwexmtmjd8	mystack_ui	replicated	2/2	darza/ui_image:V1	*:8080->80/tcp

```
C:\Users\tarak\Paris-8\Projet_container\version_swarm>docker service ps mystack_ui
```

ID	NAME	IMAGE	NODE	DESIRED STATE	CURRENT STATE	ERROR
wllpu3fkdvrrp	mystack_ui.1	darza/ui_image:V1	docker-desktop	Running	Running 53 seconds ago	
74m3vijzs05g	mystack_ui.2	darza/ui_image:V1	docker-desktop	Running	Running 53 seconds ago	

```
C:\Users\tarak\Paris-8\Projet_container\version_swarm>docker service scale
```

mystack_ui=3

mystack_ui scaled to 3

overall progress: 3 out of 3 tasks

1/3: running [=====>]

2/3: running [=====>]

3/3: running [=====>]

verify: Service converged

```
C:\Users\tarak\Paris-8\Projet_container\version_swarm>docker service ls
```

ID	NAME	MODE	REPLICAS	IMAGE	PORTS
slqzyotq5avt	mystack_api	replicated	2/2	darza/api_image:V1	*:5000->5000/tcp
w1vwexmtmjd8	mystack_ui	replicated	3/3	darza/ui_image:V1	*:8080->80/tcp

```
C:\Users\tarak\Paris-8\Projet_container\version_swarm>docker service ps mystack_ui
```

ID	NAME	IMAGE	NODE	DESIRED STATE	CURRENT STATE	ERROR
wllpu3fkdvrrp	mystack_ui.1	darza/ui_image:V1	docker-desktop	Running	Running 14 minutes ago	
74m3vijzs05g	mystack_ui.2	darza/ui_image:V1	docker-desktop	Running	Running 14 minutes ago	
ac8k288sero3	mystack_ui.3	darza/ui_image:V1	docker-desktop	Running	Running 44 seconds ago	

```
C:\Users\tarak\Paris-8\Projet_container\version_swarm>
```