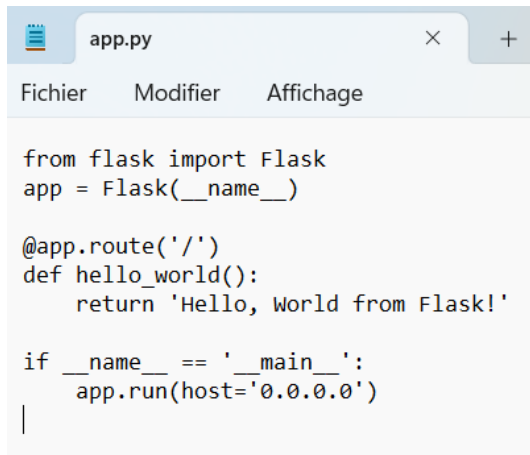


TP-3

Gaspard Devoivre

1- Python Script

We start by creating a simple python script, as asked, that we will then be able to package into a container.

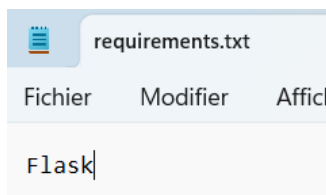


```
from flask import Flask
app = Flask(__name__)

@app.route('/')
def hello_world():
    return 'Hello, World from Flask!'

if __name__ == '__main__':
    app.run(host='0.0.0.0')
```

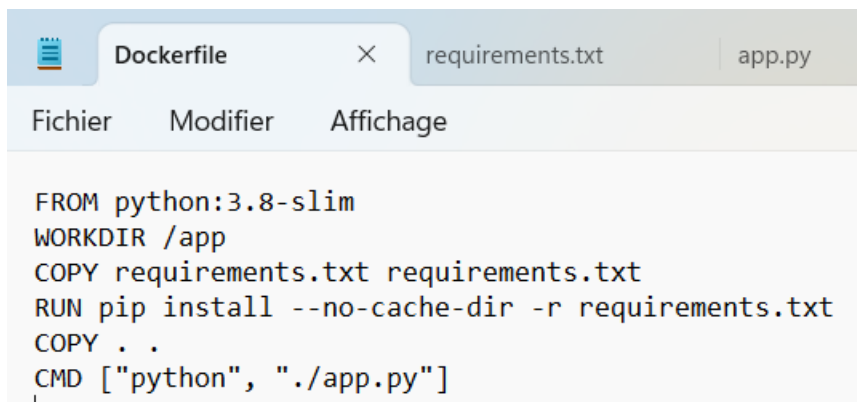
Alongside this we also need a text file that specifies the necessary Python packages that we require (In our case we only need Flask).



```
Flask
```

2- Dockerfile

Now that the python script is done, we'll move on to creating a dockerfile containing the instructions necessary to build the Docker image.



```
FROM python:3.8-slim
WORKDIR /app
COPY requirements.txt requirements.txt
RUN pip install --no-cache-dir -r requirements.txt
COPY . .
CMD ["python", "./app.py"]
```

Description:

- 'FROM python:3.8-slim': Uses a slim Python 3.8 image (smaller size).
- 'WORKDIR /app': Sets the working directory.
- 'COPY requirements.txt requirements.txt': Copies requirements.txt to the container.
- 'RUN pip install': Installs dependencies.
- 'COPY . .': Copies all files in the current directory to the container.
- 'CMD': Command to run the Python script.

3- Building and Pushing the Image

Build:

```
C:\tmp\ESILV-A4\Contenerisation Technologies\CT_TP3>docker build -t gdevoivre/python_app:ct_td3 .
[+] Building 1.9s (11/11) FINISHED                                docker:default
=> [internal] load build definition from Dockerfile                0.0s
=> => transferring dockerfile: 205B                                0.0s
=> [internal] load .dockerignore                                  0.0s
=> => transferring context: 2B                                       0.0s
=> [internal] load metadata for docker.io/library/python:3.8-slim 1.8s
=> [auth] library/python:pull token for registry-1.docker.io      0.0s
=> [1/5] FROM docker.io/library/python:3.8-slim@sha256:a92a4fd9129160a074f537482b2db341c8e9d6e04a07bfb0867228ae9 0.0s
=> [internal] load build context                                  0.0s
=> => transferring context: 93B                                       0.0s
=> CACHED [2/5] WORKDIR /app                                       0.0s
=> CACHED [3/5] COPY requirements.txt requirements.txt             0.0s
=> CACHED [4/5] RUN pip install --no-cache-dir -r requirements.txt 0.0s
=> CACHED [5/5] COPY . .                                           0.0s
=> exporting to image                                              0.0s
=> => exporting layers                                              0.0s
=> => writing image sha256:ca5d60bf3d9c10452b8a88e439406cf858a2d9b7b8be7bb801abff084fe2eb52 0.0s
=> => naming to docker.io/gdevoivre/python_app:ct_td3            0.0s

View build details: docker-desktop://dashboard/build/default/default/pxaf9wb70sbhgzf9y11d8vbe6
```

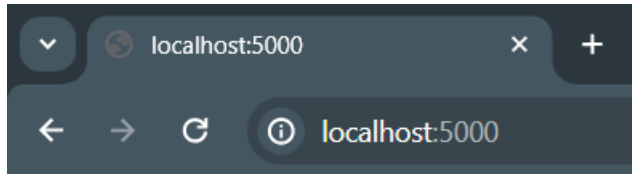
Push:

```
C:\tmp\ESILV-A4\Contenerisation Technologies\CT_TP3>docker push gdevoivre/python_app:ct_td3
The push refers to repository [docker.io/gdevoivre/python_app]
983d7dcf06a7: Pushed
deca561dc782: Pushed
ff872091cc61: Pushed
d9af246391d0: Pushed
8ffe8fa4c406: Pushed
681a1fc3389e: Pushed
47cdb7a27fca: Pushed
e96fe707bd25: Pushed
571ade696b26: Pushed
ct_td3: digest: sha256:195876ce8730f4ccb3745879cdae5bcfe8ff93c65ff72d1db0d19b84385ed498 size: 2201

C:\tmp\ESILV-A4\Contenerisation Technologies\CT_TP3>_
```

4- Running the Container

```
C:\tmp\ESILV-A4\Contenerisation Technologies\CT_TP3>docker run -p 5000:5000 gdevoivre/python_app:ct_td3
* Serving Flask app 'app'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://172.17.0.2:5000
Press CTRL+C to quit
```



Hello, World from Flask!

Success!

*** BONUS ***

- 1) Smallest Possible Image Size: I used 'python:3.8-slim'
- 2) Difference Between ADD and COPY:
 - a. COPY is for copying files and directories from the build context.
 - b. ADD can do that plus handle URLs and unpack compressed files.
- 3) Make the container run without sudo rights: 'sudo usermod -aG docker \$gdevoivre', this will add my user to the Docker group (has root privileges), bypassing sudo rights.
- 4) Link to GitHub repository: <https://github.com/gdevoivre/Containerization-Technologies>