

## COMPTE RENDU TP3

### Conteneurisation 1<sup>ère</sup> partie

Etudiant

Abdelhadi Mouzafir



Professeure

Mme Oumayma Banouar

Q1) Hello world to docker

```
PS C:\Users\Abdelhadi\ABDELHADI_MOUZAFIR> docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:e18f0a777aefabe047a671ab3ec3eed05414477c951ab1a6f352a06974245fe7
Status: Downloaded newer image for hello-world:latest
```

```
Hello from Docker!
This message shows that your installation appears to be working correctly.
```

Q2) list docker commands

```
PS C:\Users\Abdelhadi\ABDELHADI_MOUZAFIR> docker
```

Usage: docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

Options:

--config string	Location of client config files (default "C:\Users\Abdelhadi\.docker")
-c, --context string	Name of the context to use to connect to the daemon (overrides DOCKER_HOST env var and default context set with "docker context use")
-D, --debug	Enable debug mode
-H, --host list	Daemon socket(s) to connect to
-l, --log-level string	Set the logging level ("debug" "info" "warn" "error" "fatal") (default "info")
--tls	Use TLS; implied by --tlsverify
--tlscacert string	Trust certs signed only by this CA (default "C:\Users\Abdelhadi\.docker\ca.pem")
--tlscert string	Path to TLS certificate file (default "C:\Users\Abdelhadi\.docker\cert.pem")
--tlskey string	Path to TLS key file (default "C:\Users\Abdelhadi\.docker\key.pem")
--tlsverify	Use TLS and verify the remote
-v, --version	Print version information and quit

Management Commands:

builder      Manage builds

### Q3) docker help

```
Usage: docker build [OPTIONS] PATH | URL | -
Build an image from a Dockerfile

Options:
  --add-host list      Add a custom host-to-IP mapping (host:ip)
  --build-arg list     Set build-time variables
  --cache-from strings Images to consider as cache sources
  --disable-content-trust Skip image verification (default true)
  -f, --file string    Name of the Dockerfile (Default is 'PATH/Dockerfile')
  --iidfile string     Write the image ID to the file
  --isolation string   Container isolation technology
  --label list        Set metadata for an image
  --network string     Set the networking mode for the RUN instructions during build (default "default")
  --no-cache           Do not use cache when building the image
  -o, --output stringArray Output destination (format: type=local,dest-path)
  --platform string   Set platform if server is multi-platform capable
  --progress string    Set type of progress output (auto, plain, tty). Use plain to show container output (default "auto")
  --pull              Always attempt to pull a newer version of the image
  -q, --quiet          Suppress the build output and print image ID on success
  --secret stringArray Secret file to expose to the build (only
```

## Q4) hello-world dockerfile

```
FROM mcr.microsoft.com/windows/nanoserver:XXX
COPY hello.txt C:
CMD ["cmd", "/C", "type C:\\hello.txt"]
```

## Q5) docker ps , list current active processes

```
PS C:\Users\Abdelhadi\ABDELHADI_MOUZAFIR> docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                    NAMES
34a209d16256   postgres:14-alpine  "docker-entrypoint.s..." 6 days ago    Up 14 minutes  5432/tcp, 0.0.0.0:5433->5433/tcp  my-db
```

## Q6) docker ps -a , list all of the containers

```
PS C:\Users\Abdelhadi\ABDELHADI_MOUZAFIR> docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                    NAMES
4a27657f882f   hello-world    "/hello"                11 minutes ago Exited (0) 11 minutes ago           magical_leakey
e7e9681a7262   hello-world    "/hello"                12 minutes ago Exited (0) 12 minutes ago           nostalgic_bhabha
23abfd8ef071   node-plant-backend_service  "docker-entrypoint.s..." 6 days ago    Exited (137) 6 days ago           my-service
34a209d16256   postgres:14-alpine  "docker-entrypoint.s..." 6 days ago    Exited (0) About a minute ago       my-db
26efdda8d769   c6f0ed328db8   "docker-entrypoint.s..." 6 days ago    Exited (1) 6 days ago           angry_feistel
19243e34f1a0   node-plant-api-test:latest  "docker-entrypoint.s..." 7 days ago    Exited (1) 7 days ago           optimistic_leakey
5ebcfa243c4d   node-plant-api-test  "docker-entrypoint.s..." 7 days ago    Exited (1) 7 days ago           funny_feistel
aaf243cb708e   neo4j-node:latest  "docker-entrypoint.s..." 2 weeks ago   Exited (137) 2 weeks ago           busy_tharp
848a344771f0   redis:latest     "docker-entrypoint.s..." 5 weeks ago   Exited (255) 5 weeks ago           redis-BMs5
c6bcc3e06eb0   sandbox_indexer   "/tmp/start.sh"         7 weeks ago   Exited (137) 7 weeks ago           algorand-sandbox-indexer
c1ff2bd4e79c7  3e42dd4e79c7    "docker-entrypoint.s..." 2 months ago  Exited (0) 6 days ago           my-redis
fd23f8c06786   ubuntu:latest     "bash"                  2 months ago  Exited (255) 2 months ago           intelligent_neumann
283b6b3c2ddd   ubuntu:latest     "bash"                  2 months ago  Exited (255) 2 months ago           zen_booth
4b681c2801bd   adria-contrats    "java -jar adria-con..." 2 months ago  Exited (143) 2 months ago           dazzling_bassi
63de6cc1077c   testagain         "catalina.sh run"       2 months ago  Exited (143) 2 months ago           upbeat_mclaren
7cbd1d8cb0cf   ubuntu            "bash"                  3 months ago  Exited (0) 3 months ago           vibrant_mccarthy
39a3f1362e57   ubuntu            "bash"                  3 months ago  Exited (0) 3 months ago           sleepy_bose
0529cbc7e5fd   hello-from-abde   "docker-entrypoint.s..." 3 months ago  Exited (0) 3 months ago           frosty_fermat
f66a02984bfd   sandbox_algod:latest  "/opt/start_algod.sh"   4 months ago  Exited (137) 4 months ago           beautiful_heisenberg
49e1e4d8c985   sandbox_algod     "/opt/start_algod.sh"   4 months ago  Exited (137) 7 weeks ago           algorand-sandbox-algod
22bfdd3da71d   postgres:13-alpine  "docker-entrypoint.s..." 4 months ago  Exited (0) 7 weeks ago           algorand-sandbox-postgres
```

## Q7) docker images

```
PS C:\Users\Abdelhadi\ABDELHADI_MOUZAFIR> docker images
REPOSITORY              TAG                IMAGE ID           CREATED            SIZE
node-plant-backend_service   latest            f372c090d61f      6 days ago        126MB
<none>                     <none>            77e164789572      6 days ago        126MB
<none>                     <none>            5adc97322e28      6 days ago        126MB
<none>                     <none>            abd884508af2      6 days ago        126MB
<none>                     <none>            c327ce8c8da3      6 days ago        126MB
<none>                     <none>            c00ba20cfab4      6 days ago        126MB
node-plant-backend_app       latest            7987a9d9c9f7      6 days ago        123MB
node-plant-api-test          latest            bf53e19b1a96      7 days ago        121MB
<none>                     <none>            c6f0ed328db8      7 days ago        121MB
neo4j-node                   latest            d588d7f49c04      2 weeks ago       175MB
postgres                     14-alpine        aac01494762a      4 weeks ago       216MB
redis                        latest            9da089657551      6 weeks ago       117MB
abderoxdocker/thymeleaf-spring-boot-test dev             a0cbdbc2dfb7      2 months ago      427MB
adria-contrats               latest            a0cbdbc2dfb7      2 months ago      427MB
abderoxdocker/tomcat-servlets latest            82ff66c075d9      2 months ago      403MB
testagain                    latest            82ff66c075d9      2 months ago      403MB
hello-from-abde              latest            63d972c4c2a9      3 months ago      166MB
abderoxdocker/test-docker12  latest            63d972c4c2a9      3 months ago      166MB
redis                        <none>            3e42dd4e79c7      3 months ago      117MB
ubuntu                       latest            df5de72bdb3b      3 months ago      77.8MB
sandbox_indexer              latest            c736e7fc24dc      4 months ago      1.69GB
sandbox_algod                 latest            9d5bf279d753      4 months ago      1.33GB
postgres                     13-alpine        2bb8cea1e0bb      4 months ago      213MB
hello-world                   latest            feb5d9fea6a5      13 months ago     13.3kB
```

## Q8) docker search centos

```
PS C:\Users\Abdelhadi\ABDELHADI_MOUZAFIR> docker search centos
NAME                                DESCRIPTION                                STARS     OFFICIAL   AUTOMATED
kasmweb/centos-7-desktop             CentOS 7 desktop for Kasm Workspaces        25
couchbase/centos7-systemd            centos7-systemd images with additional debug... 5          [OK]
dokken/centos-7                     CentOS 7 image for kitchen-dokken           3
continuumio/centos5_gcc5_base        CentOS 5 image for kitchen-dokken           3
dokken/centos-stream-9               CentOS 9 image for kitchen-dokken           2
dokken/centos-stream-8               CentOS 8 image for kitchen-dokken           2
spack/centos6                       CentOS 6 with Spack preinstalled             1
dokken/centos-8                     CentOS 8 image for kitchen-dokken           1
spack/centos7                       CentOS 7 with Spack preinstalled             1
centos                              DEPRECATED; The official build of CentOS.    0          [OK]
```

It finds all the images that include the word "centos" from **Docker** Hub , and let us choose the one that fits our needs .

## Q9) docker search --filter stars=5 centos

NAME	DESCRIPTION	STARS	OFFICIAL	AUTOMATED
kasmweb/centos-7-desktop	CentOS 7 desktop for Kasm Workspaces	25		
couchbase/centos7-systemd	centos7-systemd images with additional debug...	5		[OK]

### Q9) docker pull centos

```
PS C:\Users\Abdelhadi\ABDELHADI_MOUZAFIR> docker pull centos
Using default tag: latest
latest: Pulling from library/centos
a1d0c7532777: Pull complete
Digest: sha256:a27fd8080b517143cbbbab9dfb7c8571c40d67d534bbdee55bd6c473f432b177
Status: Downloaded newer image for centos:latest
docker.io/library/centos:latest
```

### Q10) docker runs the image

```
PS C:\Users\Abdelhadi\ABDELHADI_MOUZAFIR> docker run -it centos
[root@6575603b3e32 /]# ls
bin dev etc home lib lib64 lost+found media mnt opt proc root run sbin srv sys tmp usr var
[root@6575603b3e32 /]#
```

### Q11) docker removes the container

```
PS C:\Users\Abdelhadi\ABDELHADI_MOUZAFIR> docker rm 6575603b3e32
6575603b3e32
PS C:\Users\Abdelhadi\ABDELHADI_MOUZAFIR>
```

### Q12) docker removes image

```
PS C:\Users\Abdelhadi\ABDELHADI_MOUZAFIR> docker image rm 5d0da3dc9764
Untagged: centos:latest
Untagged: centos@sha256:a27fd8080b517143cbbbab9dfb7c8571c40d67d534bbdee55bd6c473f432b177
Deleted: sha256:5d0da3dc976460b72c77d94c8a1ad043720b0416bfc16c52c45d4847e53fadb6
Deleted: sha256:74ddd0ec08fa43d09f32636ba91a0a3053b02cb4627c35051aff89f853606b59
PS C:\Users\Abdelhadi\ABDELHADI_MOUZAFIR>
```

### Q12) docker force stop images and delete :

```
docker rmi -f $(docker images -a -q)
```

Q13) docker delete all images : `docker rmi $(docker images -a -q)`

Q14) docker build image

```
PS C:\Users\Abdelhadi\ABDELHADI_MOUZAFIR\service> docker build -t abderox/test-node .
[+] Building 47.8s (9/11)
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 231B
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load metadata for docker.io/library/node:16-alpine
=> [auth] library/node:pull token for registry-1.docker.io
=> [1/6] FROM docker.io/library/node:16-alpine@sha256:f16544bc93cf1a36d213c8e2efecf682
=> => resolve docker.io/library/node:16-alpine@sha256:f16544bc93cf1a36d213c8e2efecf682
=> [internal] load build context
=> => transferring context: 4.85MB
=> CACHED [2/6] RUN mkdir -p /usr/src/app
=> CACHED [3/6] WORKDIR /usr/src/app
=> [4/6] COPY package.json /usr/src/app/
=> [5/6] RUN npm install
```

Q15) docker run image

```
PS C:\Users\Abdelhadi\ABDELHADI_MOUZAFIR\service> docker run -p 3000:4851 -d abderox/test-node
2decd5ed86510ad8133f25a1e4ccdc952263b974037862e9f7dff490abb60f0
PS C:\Users\Abdelhadi\ABDELHADI_MOUZAFIR\service>
```

```
FROM node:16-alpine
RUN mkdir -p /usr/src/app
WORKDIR /usr/src/app
COPY package.json /usr/src/app/
RUN npm install
COPY . /usr/src/app
EXPOSE 4851
CMD [ "node", "app.js" ]
```

## Q16) docker login

```
PS C:\Users\Abdelhadi\ABDELHADI_MOUZAFIR\service> docker login
Authenticating with existing credentials...
Login Succeeded
```

## Q17) docker tag image\_id abderox/node-test:latest

```
PS C:\Users\Abdelhadi\ABDELHADI_MOUZAFIR\service> docker tag 372288562179 abderox/node-test:latest
PS C:\Users\Abdelhadi\ABDELHADI_MOUZAFIR\service> docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
abderox/node-test	latest	372288562179	9 minutes ago	131MB
abderox/test-node	latest	372288562179	9 minutes ago	131MB

## Q18) docker push abderox/node-test:latest

```
PS C:\Users\Abdelhadi\ABDELHADI_MOUZAFIR\service> docker tag 372288562179 abderoxdocker/node-test:latest
PS C:\Users\Abdelhadi\ABDELHADI_MOUZAFIR\service> docker push abderoxdocker/node-test:latest
The push refers to repository [docker.io/abderoxdocker/node-test]
7103f024e6fa: Pushed
a448a7e710a3: Pushed
acdd93824527: Pushed
5f70bf18a086: Pushed
8632e2be5d33: Pushed
a79681b2645a: Pushed
f506bc026271: Pushed
11816ca779c7: Pushed
994393dc58e7: Pushed
372288562179: Pushed
```

abderoxdocker / <b>node-test</b> Contains: No content   Last pushed: 2 minutes ago	Not Scanned	☆ 0	↓ 0	Public
abderoxdocker / <b>thymeleaf-spring-boot-test</b> Contains: Image   Last pushed: 3 months ago	Not Scanned	☆ 0	↓ 3	Public
abderoxdocker / <b>tomcat-servlets</b> Contains: Image   Last pushed: 3 months ago	Not Scanned	☆ 0	↓ 2	Public
abderoxdocker / <b>test-docker12</b> Contains: Image   Last pushed: 3 months ago	Not Scanned	☆ 0	↓ 1	Public
abderoxdocker / <b>docker101tutorial</b> Contains: Image   Last pushed: 8 months ago	Not Scanned	☆ 0	↓ 7	Public

## Q19) fullstack application with docker

```
version: '3'
services:
  postgres:
    container_name: 'my-dc-op-7'
    image: postgres:14-alpine
    ports:
      - "5432:5432"
    restart: always
    environment:
      - POSTGRES_PASSWORD=root
      - POSTGRES_USER=postgres
      - POSTGRES_DB=book-abdelhadi
    volumes:
      - ./postgresql2/data:/var/lib/postgresql/data
    networks:
      - backend
  service:
    container_name: 'my-service-op'
    build:
      context:
        ./service/
      dockerfile: Dockerfile
    image: 'abderox/my-service13-7'
    ports:
      - "8085:8080"
    depends_on:
      - postgres
    networks:
      - backend
      - frontend
  ui:
    container_name: 'my-ui-oi-7'
    build:
      context: ./ui/
      dockerfile: Dockerfile
    image: abderox/my-ui-30
    ports:
      - "8888:80"
    depends_on:
      - service
    networks:
      - frontend

networks:
  backend:
  frontend:
```

```
> postgresql2
> service
> ui
  docker-compose.yml
```

```
> postgresql2\data
  service
    api
    covers
    node_modules
    .dockerignore
    .env
    app.js
    Dockerfile
    package-lock.json
    package.json
  ui
    node_modules
    public
    src
    .dockerignore
    .gitignore
    Dockerfile
    package-lock.json
    package.json
    README.md
  docker-compose.yml
```

The directory named **service** holds the Backoffice of the application whilst ui folder contains the **frontend**, the other folder renowned **postgresql2** is the volume I specified in docker compose for my database.

My docker compose



```
e > Dockerfile > ...  
FROM node:16-alpine  
  
RUN mkdir -p /usr/src/app/service  
  
WORKDIR /usr/src/app/service  
  
COPY package.json /usr/src/app/service  
  
RUN npm install  
  
COPY . /usr/src/app/service  
  
CMD [ "node", "app.js" ]
```

Dockerfile of my application backend

```
FROM node:16-alpine AS builder  
  
RUN mkdir -p /usr/src/app  
  
WORKDIR /usr/src/app  
  
COPY package.json /usr/src/app  
  
RUN npm install  
  
COPY . /usr/src/app  
  
RUN npm run build  
  
FROM nginx:1.17.0-alpine  
  
COPY --from=builder /usr/src/app/build /usr/share/nginx/html  
  
EXPOSE 80  
  
CMD ["nginx", "-g", "daemon off;"]
```



























Dockerfile of my application frontend

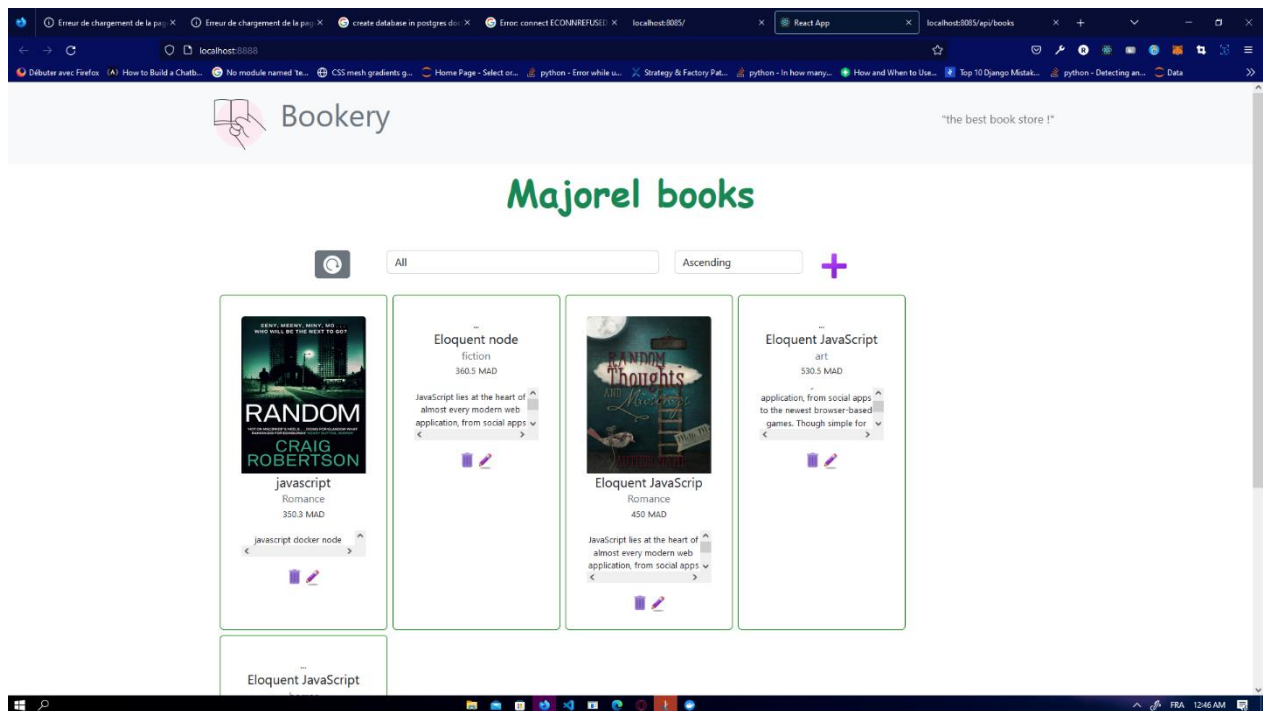
I use the following command to make sure the cache is clean to avoid confusions between ports , and images.

```
PS C:\Users\Abdelhadi\ABDELHADI_MOUZAFIR> docker-compose build --no-cache
postgres uses an image, skipping
Building service
[+] Building 49.5s (11/11) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 32B
=> [internal] load .dockerignore
=> => transferring context: 34B
=> [internal] load metadata for docker.io/library/node:16-alpine
=> [internal] load build context
=> => transferring context: 1.08kB
=> CACHED [1/6] FROM docker.io/library/node:16-alpine@sha256:15dd66f723aab8b367abc7ac6ed25594ca4653f2ce49ad1505bfbe740ad5190e
=> => resolve docker.io/library/node:16-alpine@sha256:15dd66f723aab8b367abc7ac6ed25594ca4653f2ce49ad1505bfbe740ad5190e
=> [2/6] RUN mkdir -p /usr/src/app/service
=> [3/6] WORKDIR /usr/src/app/service
=> [4/6] COPY package.json /usr/src/app/service
=> [5/6] RUN npm install
=> [6/6] COPY . /usr/src/app/service
=> exporting to image
=> => exporting layers
=> => writing image sha256:6a9ce1d74198686213c91de5fff7a0de99fc779f14183434616ba2d5ea62833a
=> => naming to docker.io/abderox/my-service-7
Building ui
[+] Building 351.8s (15/15) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 32B
=> [internal] load .dockerignore
=> => transferring context: 34B
=> [internal] load metadata for docker.io/library/nginx:1.17.0-alpine
=> [internal] load metadata for docker.io/library/node:16-alpine
=> CACHED [builder 1/7] FROM docker.io/library/node:16-alpine@sha256:15dd66f723aab8b367abc7ac6ed25594ca4653f2ce49ad1505bfbe740ad5190e
=> => resolve docker.io/library/node:16-alpine@sha256:15dd66f723aab8b367abc7ac6ed25594ca4653f2ce49ad1505bfbe740ad5190e
=> CACHED [stage-1 1/2] FROM docker.io/library/nginx:1.17.0-alpine@sha256:b126fee6820be927b1e04ae36b3f51aa47d9b73bf6b1826ff19a59d22b2b4c63
=> [internal] load build context
=> [builder 2/7] RUN mkdir -p /usr/src/app
=> [builder 3/7] WORKDIR /usr/src/app
=> [builder 4/7] COPY package.json /usr/src/app
=> [builder 5/7] RUN npm install
=> [builder 6/7] COPY . /usr/src/app
=> [builder 7/7] RUN npm run build
```

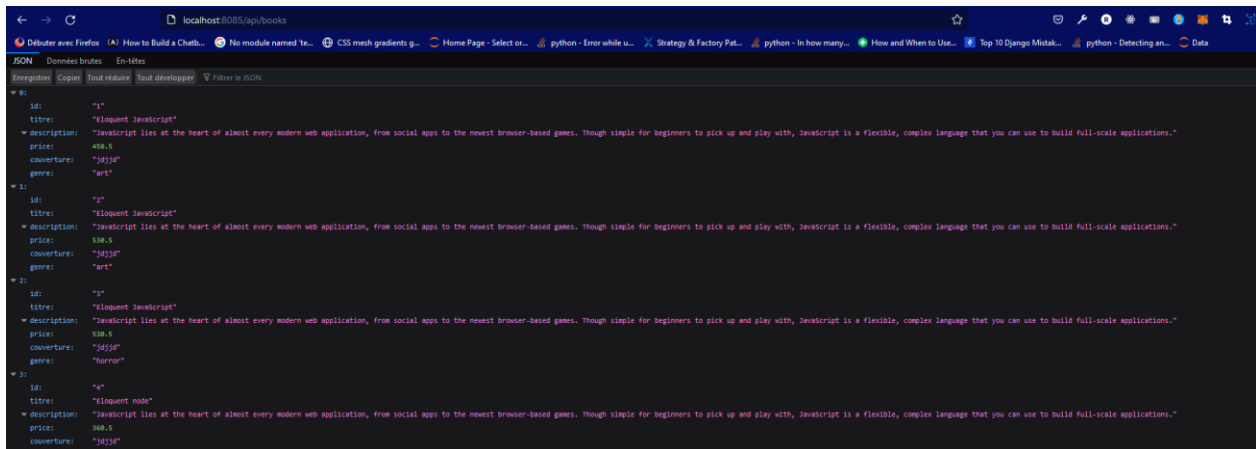
```
PS C:\Users\Abdelhadi\ABDELHADI_MOUZAFIR> docker-compose up -d
my-dc-op-7 is up-to-date
my-service-op is up-to-date
my-ui-oi-7 is up-to-date
```

All of the three containers are up to date and running

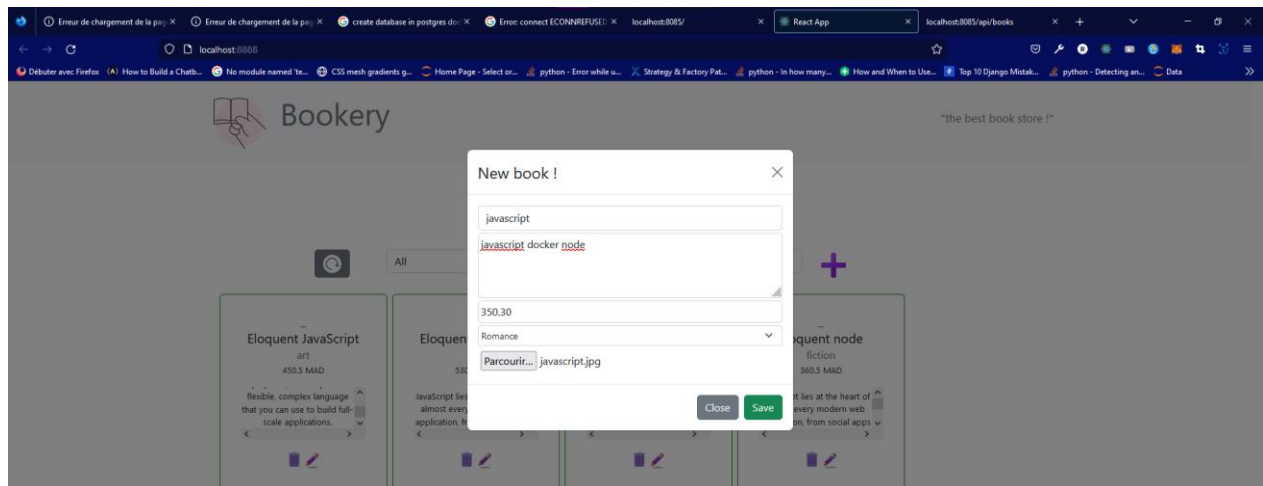
<input type="checkbox"/>	 <b>abdelhadi_mouzafir</b> 3 containers	-	Running (3/3)	-	 Open   
<input type="checkbox"/>	 <b>my-dc-op-7</b> 98bdbbfdb97 	postgres:14-alpine	Running	5432	27 minutes ag     
<input type="checkbox"/>	 <b>my-service-op</b> c15d3e429b04 	abderox/my-service-13:latest	Running	8085	27 minutes ag     
<input type="checkbox"/>	 <b>my-ui-oi-7</b> ffee95b2e2ef 	abderox/my-ui-30:latest	Running	8888	21 minutes ag     



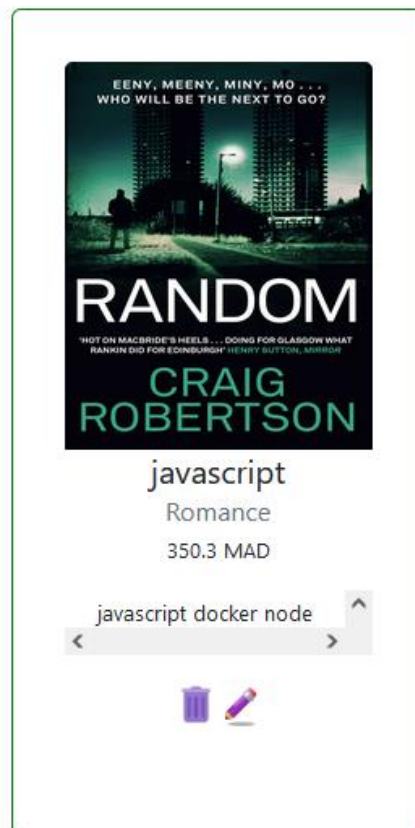
The frontend is running on port 8888



The backend is running on port 8085



Testing communication between the three containers : it worked !!



```

PS C:\Users\Abdelhadi\ABDELHADI_MOUZAFIR> docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                    NAMES
ffee95b2e2ef   abderox/my-ui-30   "nginx -g 'daemon of..." 28 minutes ago Up 28 minutes   0.0.0.0:8888->80/tcp      my-ui-oi-7
c15d36429b04   abderox/my-service13-7   "docker-entrypoint.s..." 34 minutes ago Up 34 minutes   0.0.0.0:8085->8080/tcp    my-service-op
98dbbfbdb97    postgres:14-alpine   "docker-entrypoint.s..." 34 minutes ago Up 34 minutes   0.0.0.0:5432->5432/tcp    my-dc-op-7
PS C:\Users\Abdelhadi\ABDELHADI_MOUZAFIR> docker exec -it 98dbbfbdb97 bash
bash-5.1# select * from book-abdelhadi;
bash: syntax error near unexpected token `from'
bash-5.1# select * from 'book-abdelhadi';
bash: syntax error near unexpected token `from'
bash-5.1# psql -U postgres book-abdelhadi
psql (14.5)
Type "help" for help.

book-abdelhadi=# select * from Books;
 id |      titre      | price | couverture | genre | description
-----+-----+-----+-----+-----+-----
  2 | Eloquent JavaScript | 530.5 | jdjjd      | art   | JavaScript lies at the heart of almost every modern web application, from social apps to the newest browser-based games. Though simple for beginners to pick up and play with, JavaScript is a flexible, complex language that you can use to build full-scale applications.
  3 | Eloquent JavaScript | 530.5 | jdjjd      | horror | JavaScript lies at the heart of almost every modern web application, from social apps to the newest browser-based games. Though simple for beginners to pick up and play with, JavaScript is a flexible, complex language that you can use to build full-scale applications.
  4 | Eloquent node       | 530.5 | jdjjd      | fiction | JavaScript lies at the heart of almost every modern web application, from social apps to the newest browser-based games. Though simple for beginners to pick up and play with, JavaScript is a flexible, complex language that you can use to build full-scale applications.
  1 | Eloquent JavaScript | 450   | node.png   | Romance | JavaScript lies at the heart of almost every modern web application, from social apps to the newest browser-based games. Though simple for beginners to pick up and play with, JavaScript is a flexible, complex language that you can use to build full-scale applications.
  5 | javascript          | 350.3 | javascript.jpg | Romance | javascript docker node

(5 rows)

book-abdelhadi=#

```

For more insurance, I tried to access my database by accessing to the bash of the running postgresql container, and I displayed previously inserted rows .

END