English Report – 1st Lesson

José MASSE

LP A2SR - Communication en langue anglaise

Mathieu Perochon

2023

1st lesson

GIT, Docker

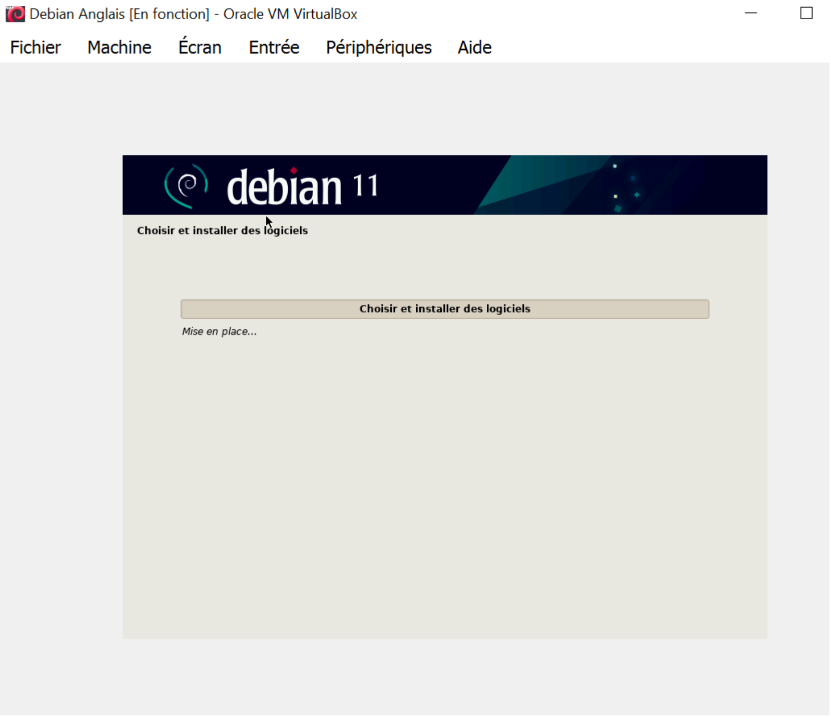
- In this course, we have to speak english only. The purpose of this is to get used to speak english for a technical on technicals subjets in order to be able to work in an international company with foreign co-worker.

- We introduced ourselves to the class and then we started to work on our practical works.

Mission 1 :

First task : Set Up a Debian 11

- In order to use Docker, we need to set up a Linux virtual machine using VirtualBox. I choosed to install a Debian 11.



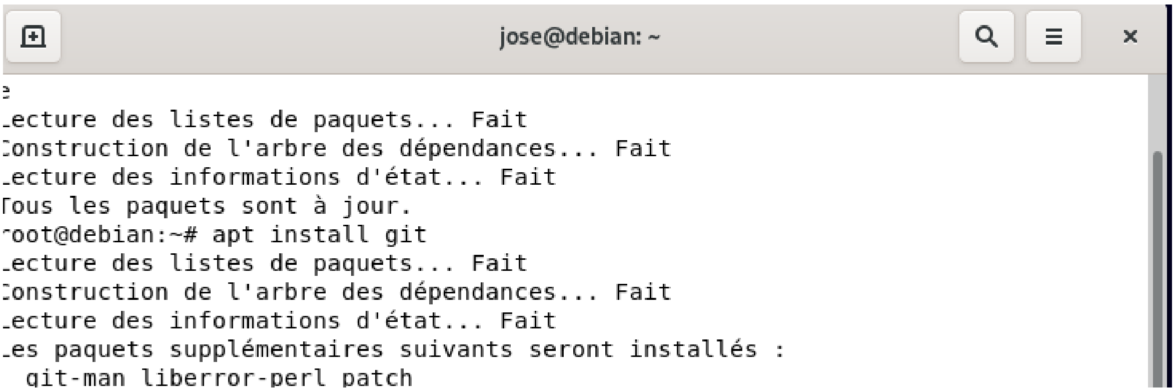
Choisir et installer des logiciels

Choisir et installer des logiciels

Mise en place...

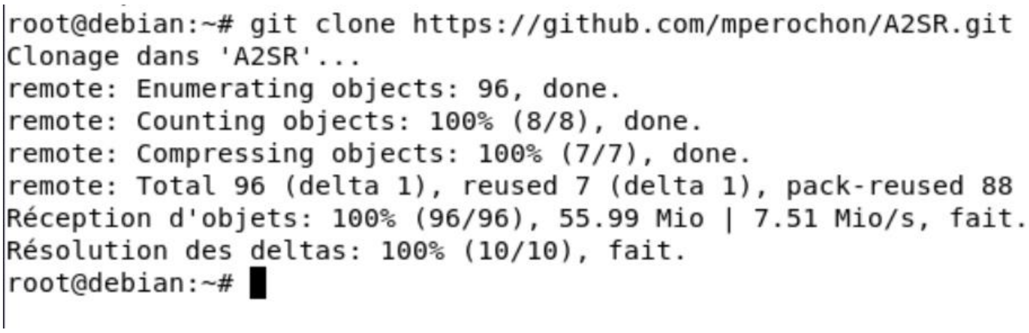
Second step : Install Docker and GIT

Here's a screenshot of the GIT install.



```
e
.ecture des listes de paquets... Fait
:onstruction de l'arbre des dépendances... Fait
.ecture des informations d'état... Fait
Tous les paquets sont à jour.
root@debian:~# apt install git
.ecture des listes de paquets... Fait
:onstruction de l'arbre des dépendances... Fait
.ecture des informations d'état... Fait
.es paquets supplémentaires suivants seront installés :
  git-man liberror-perl patch
```

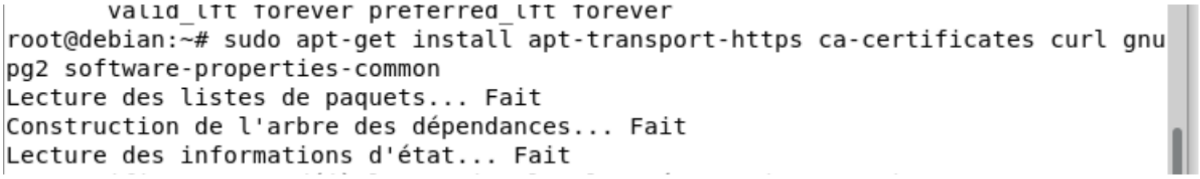
Then, I cloned Mathieu Perochon GitHub Repository :



```
root@debian:~# git clone https://github.com/mperochon/A2SR.git
Clonage dans 'A2SR'...
remote: Enumerating objects: 96, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 96 (delta 1), reused 7 (delta 1), pack-reused 88
Réception d'objets: 100% (96/96), 55.99 Mio | 7.51 Mio/s, fait.
Résolution des deltas: 100% (10/10), fait.
root@debian:~# 
```

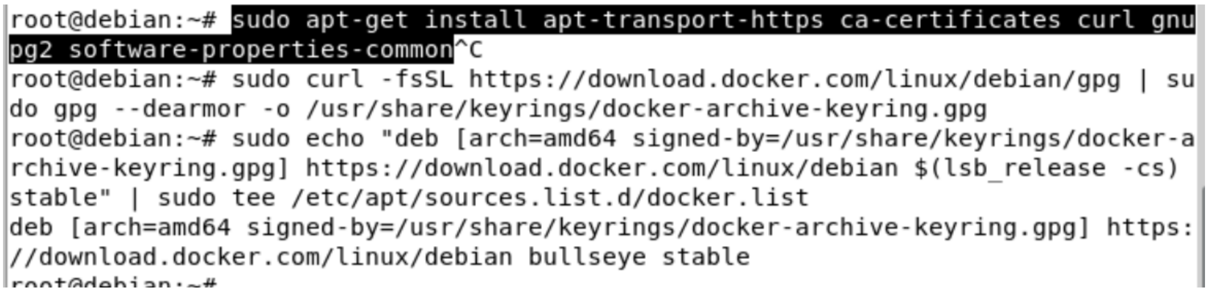
Set up Docker's package repository :

We first installed certificates tools, curl command, etc



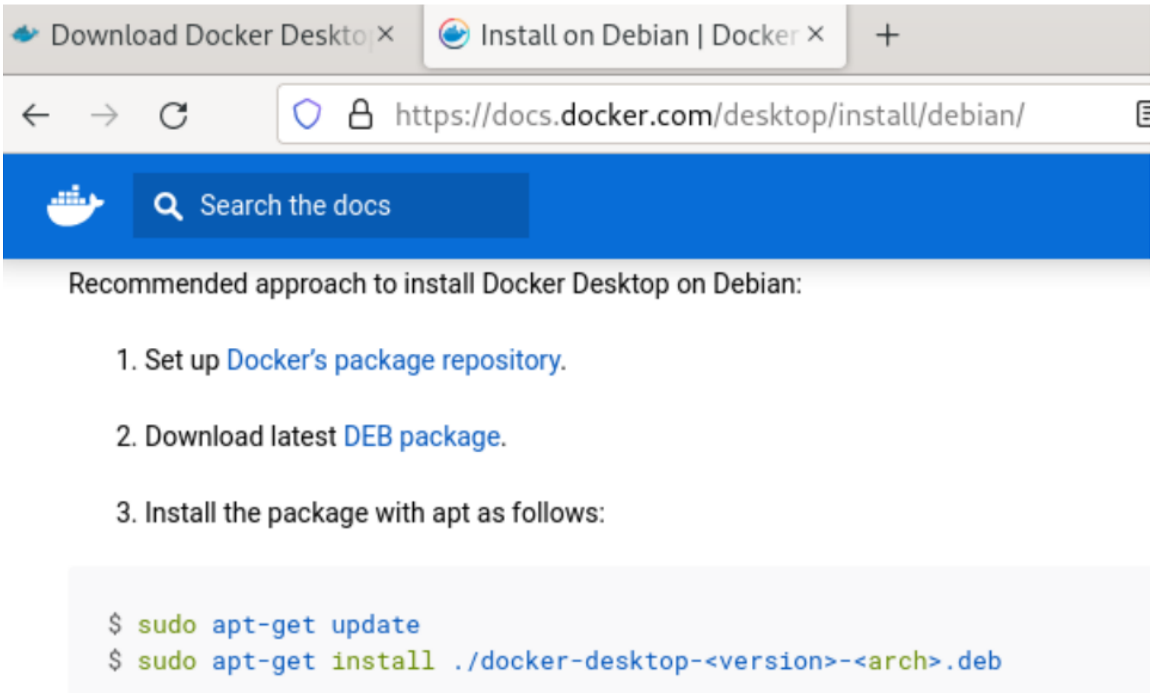
```
       valid_lft forever preferred_lft forever
root@debian:~# sudo apt-get install apt-transport-https ca-certificates curl gnu
pg2 software-properties-common
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
```

Then we downloaded the certificates with a curl command.



```
root@debian:~# sudo apt-get install apt-transport-https ca-certificates curl gnu
pg2 software-properties-common^C
root@debian:~# sudo curl -fsSL https://download.docker.com/linux/debian/gpg | su
do gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
root@debian:~# sudo echo "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-a
rchive-keyring.gpg] https://download.docker.com/linux/debian $(lsb_release -cs)
stable" | sudo tee /etc/apt/sources.list.d/docker.list
deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https:
//download.docker.com/linux/debian bullseye stable
root@debian:~#
```

I downloaded the latest Docker DEB package on the official website and installed it.

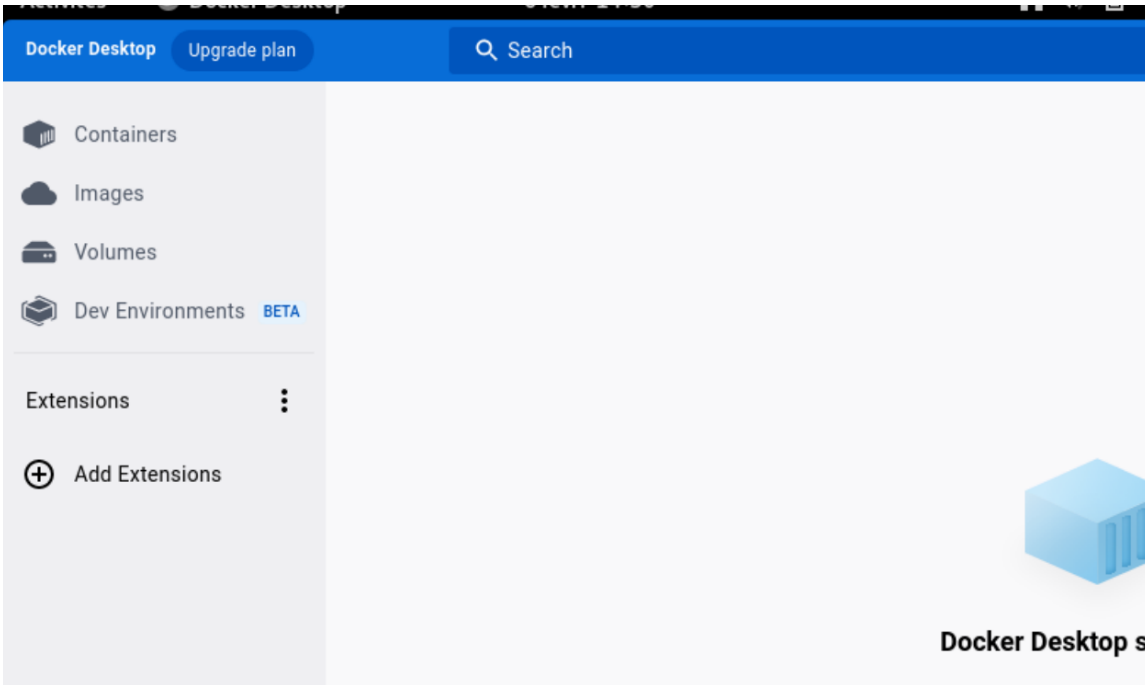

Download Docker Deskto
Install on Debian | Docker
https://docs.docker.com/desktop/install/debian/
Search the docs

Recommended approach to install Docker Desktop on Debian:

1. Set up Docker's package repository.
2. Download latest DEB package.
3. Install the package with apt as follows:

```
$ sudo apt-get update
$ sudo apt-get install ./docker-desktop-<version>-<arch>.deb
```

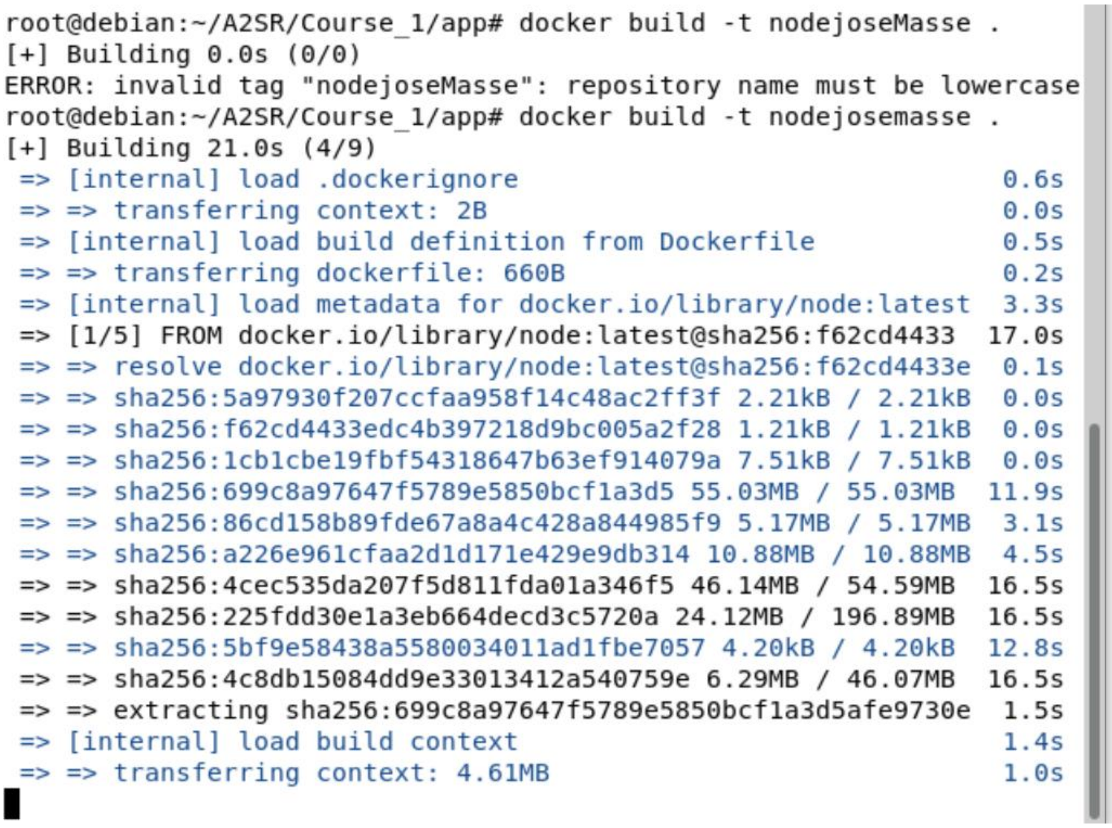
Docker Desktop is ready to use !



After that, we had to complete the DockerFile :

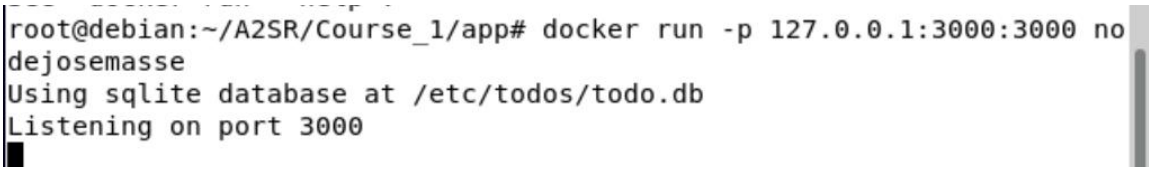
```
# Complete the file :)

# Description : specifies the parent image. Here it’s the latest version
of node
FROM node:latest

# Explain : run a Linux command. This command install python, g++ and
make. This command didn’t work for me so I commented it .

#RUN apk add --no-cache python2 g++ make

# Explain : sets the working directory for the instructions that follow
WORKDIR /app

# Description : copy files and directories to the container. Here ‘.’
specifies the current directory
COPY . .

# Explain : This command install dependencies for the project
RUN yarn install --production

# Explain : This command run the node from index.js
CMD ["node", "src/index.js"]
```

Then, we need to build the image with the commmand docker build –t (image name) (dockerFile directory)



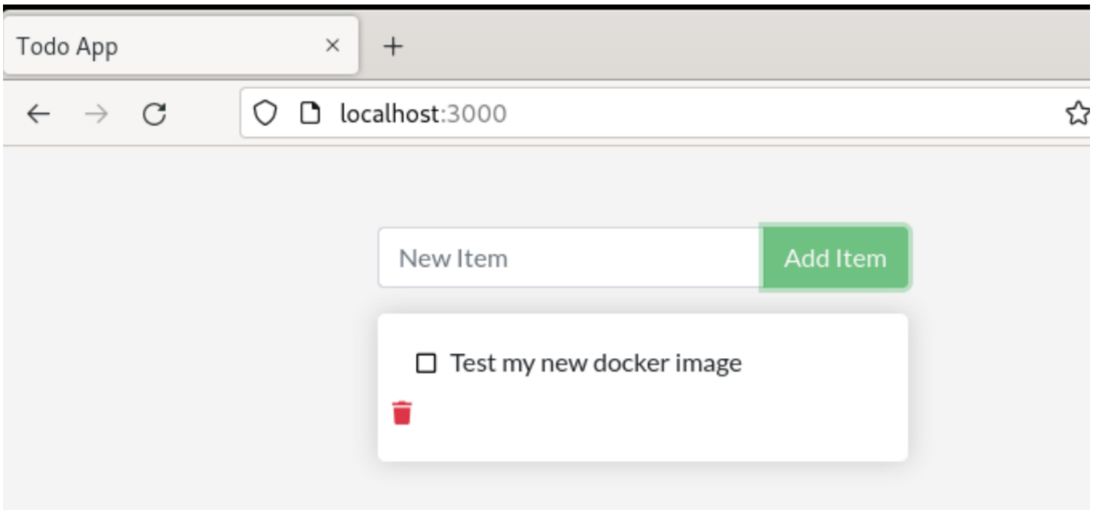
```
root@debian:~/A2SR/Course_1/app# docker build -t nodejoseMasse .
[+] Building 0.0s (0/0)
ERROR: invalid tag "nodejoseMasse": repository name must be lowercase
root@debian:~/A2SR/Course_1/app# docker build -t nodejosemasse .
[+] Building 21.0s (4/9)
 => [internal] load .dockerignore                                    0.6s
 => => transferring context: 2B                                      0.0s
 => [internal] load build definition from Dockerfile                 0.5s
 => => transferring dockerfile: 660B                                 0.2s
 => [internal] load metadata for docker.io/library/node:latest       3.3s
 => [1/5] FROM docker.io/library/node:latest@sha256:f62cd4433       17.0s
 => => resolve docker.io/library/node:latest@sha256:f62cd4433e       0.1s
 => => sha256:5a97930f207ccfaa958f14c48ac2ff3f 2.21kB / 2.21kB       0.0s
 => => sha256:f62cd4433edc4b397218d9bc005a2f28 1.21kB / 1.21kB       0.0s
 => => sha256:1cb1cbe19fbf54318647b63ef914079a 7.51kB / 7.51kB       0.0s
 => => sha256:699c8a97647f5789e5850bcf1a3d5 55.03MB / 55.03MB       11.9s
 => => sha256:86cd158b89fde67a8a4c428a844985f9 5.17MB / 5.17MB       3.1s
 => => sha256:a226e961cfaa2d1d171e429e9db314 10.88MB / 10.88MB       4.5s
 => => sha256:4cec535da207f5d811fda01a346f5 46.14MB / 54.59MB       16.5s
 => => sha256:225fdd30e1a3eb664decd3c5720a 24.12MB / 196.89MB       16.5s
 => => sha256:5bf9e58438a5580034011ad1fbe7057 4.20kB / 4.20kB       12.8s
 => => sha256:4c8db15084dd9e33013412a540759e 6.29MB / 46.07MB       16.5s
 => => extracting sha256:699c8a97647f5789e5850bcf1a3d5afe9730e       1.5s
 => [internal] load build context                                    1.4s
 => => transferring context: 4.61MB                                  1.0s
```

Then I launched my docker image on localhost, port 3000 :



```
root@debian:~/A2SR/Course_1/app# docker run -p 127.0.0.1:3000:3000 no
dejosemasse
Using sqlite database at /etc/todos/todo.db
Listening on port 3000
```

My new docker image works fine.

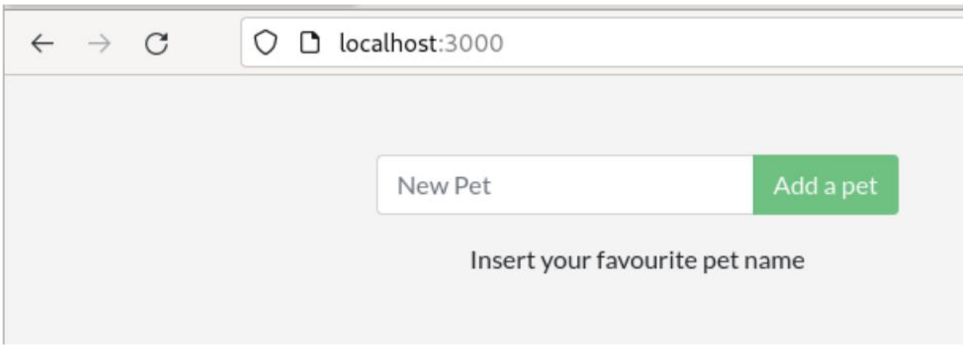


□ Test my new docker image

Mission 2 :

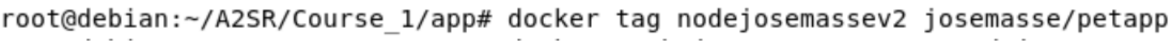
I modified the file app.js :

You can now use this app to list your favourite pets !



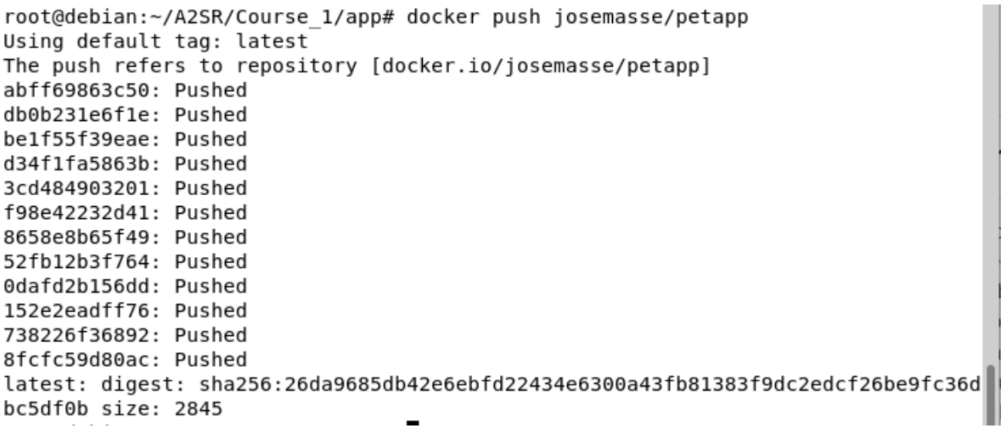
I built again my docker image, called nodejosemassev2.

I logged in my docker account and then I tagged my new app :



```
root@debian:~/A2SR/Course_1/app# docker tag nodejosemassev2 josemasse/petapp
```

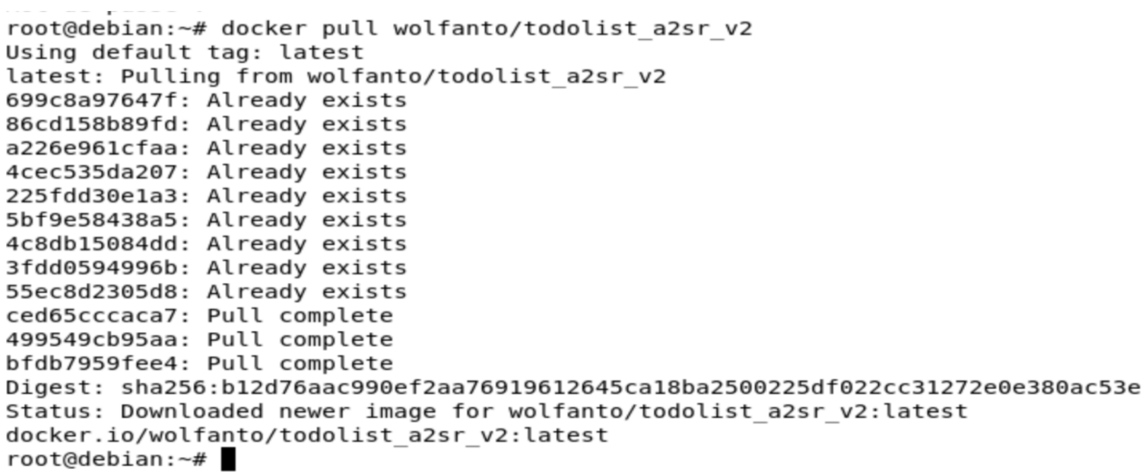
I finally pushed my new node on my Docker repository.



```
root@debian:~/A2SR/Course_1/app# docker push josemasse/petapp
Using default tag: latest
The push refers to repository [docker.io/josemasse/petapp]
abff69863c50: Pushed
db0b231e6f1e: Pushed
be1f55f39eae: Pushed
d34f1fa5863b: Pushed
3cd484903201: Pushed
f98e42232d41: Pushed
8658e8b65f49: Pushed
52fb12b3f764: Pushed
0dafd2b156dd: Pushed
152e2eadff76: Pushed
738226f36892: Pushed
8fcfc59d80ac: Pushed
latest: digest: sha256:26da9685db42e6ebfd22434e6300a43fb81383f9dc2edcf26be9fc36d
bc5df0b size: 2845
```

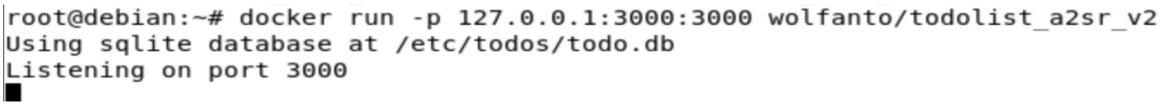
I sent Antonin my docker repository link so he could pull my docker image.

Then I pulled his image, as we can see in this screenshot :



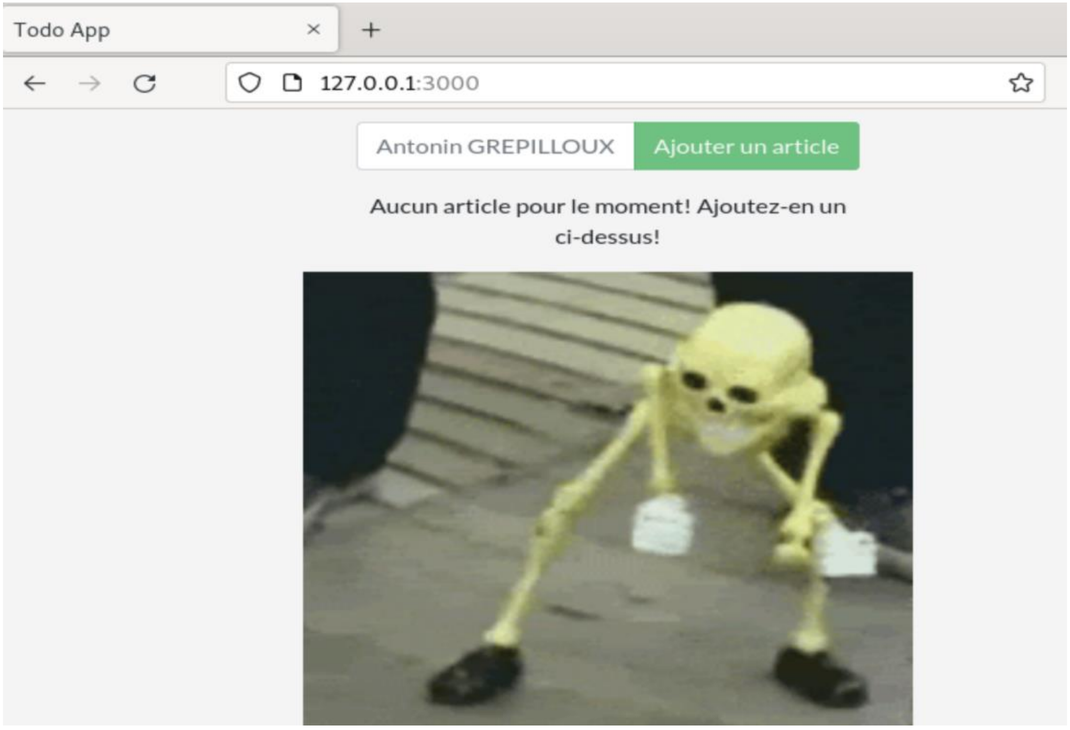
```
root@debian:~# docker pull wolfanto/todolist_a2sr_v2
Using default tag: latest
latest: Pulling from wolfanto/todolist_a2sr_v2
699c8a97647f: Already exists
86cd158b89fd: Already exists
a226e961cfaa: Already exists
4cec535da207: Already exists
225fdd30e1a3: Already exists
5bf9e58438a5: Already exists
4c8db15084dd: Already exists
3fdd0594996b: Already exists
55ec8d2305d8: Already exists
ced65cccaca7: Pull complete
499549cb95aa: Pull complete
bfdb7959fee4: Pull complete
Digest: sha256:b12d76aac990ef2aa76919612645ca18ba2500225df022cc31272e0e380ac53e
Status: Downloaded newer image for wolfanto/todolist_a2sr_v2:latest
docker.io/wolfanto/todolist_a2sr_v2:latest
root@debian:~# 
```

After that I launched this new docker image I received from Antonin’s repository.



```
root@debian:~# docker run -p 127.0.0.1:3000:3000 wolfanto/todolist_a2sr_v2
Using sqlite database at /etc/todos/todo.db
Listening on port 3000
```

We can now check this new app in our browser :



Antonin GREPILLOUX

Aucun article pour le moment! Ajoutez-en un ci-dessus!

Mission 3 :

This mission is about create a Docker from scratch. We had to follow the Docker get started tutorial and then share it like we did in mission 2.

By default, I we drop the container, we lose every datas on it, that’s why we have to find a solution to have persistant data on container.

The solution is using Docker Compose.

Let’s intall Docker Compose :

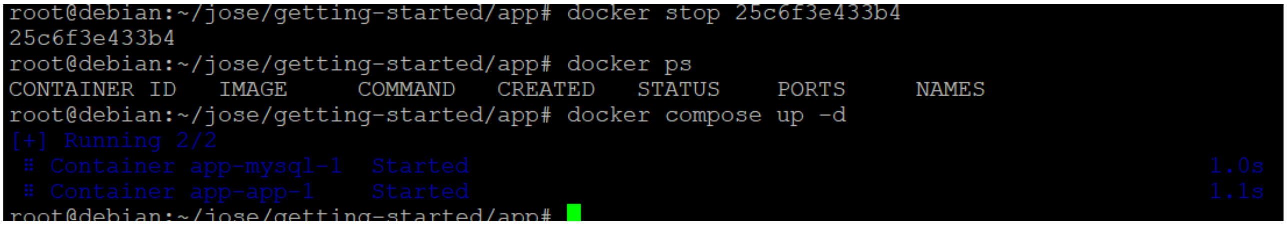
Checking version :

Docker Compose version v2.15.1

I completed the docker-compose.yml.

I first had issue because of the indentation of the .yml file.

I stopped the previous Docker process and started it again with compose :



```
root@debian:~/jose/getting-started/app# docker stop 25c6f3e433b4
25c6f3e433b4
root@debian:~/jose/getting-started/app# docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
root@debian:~/jose/getting-started/app# docker compose up -d
[+] Running 2/2
 ⠿ Container app-mysql-1  Started                                                    1.0s
 ⠿ Container app-app-1    Started                                                    1.1s
root@debian:~/jose/getting-started/app#
```

I finally pushed my repository on my GitHub repository with these steps :

git config --global user.name "josemasse"

git config --global user.email josemasse@hotmail.fr

Git init app/ (to create the git repo in the app directory)

git add . (to add all the files in the current directory in the .git)

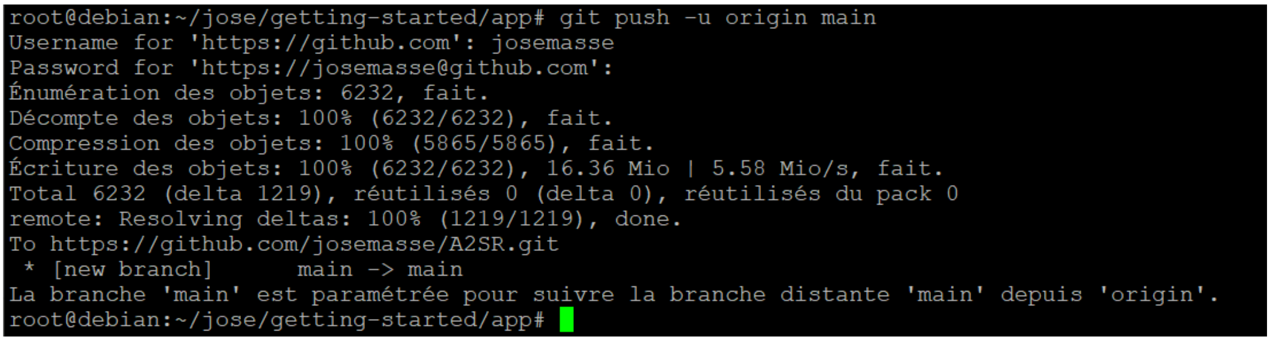
git commit -m "Mon premier commit" (to commit modification)

git remote add origin https://github.com/josemasse/A2SR.git (to specify where is my github repository)

git push -u origin main (to push the main branch)

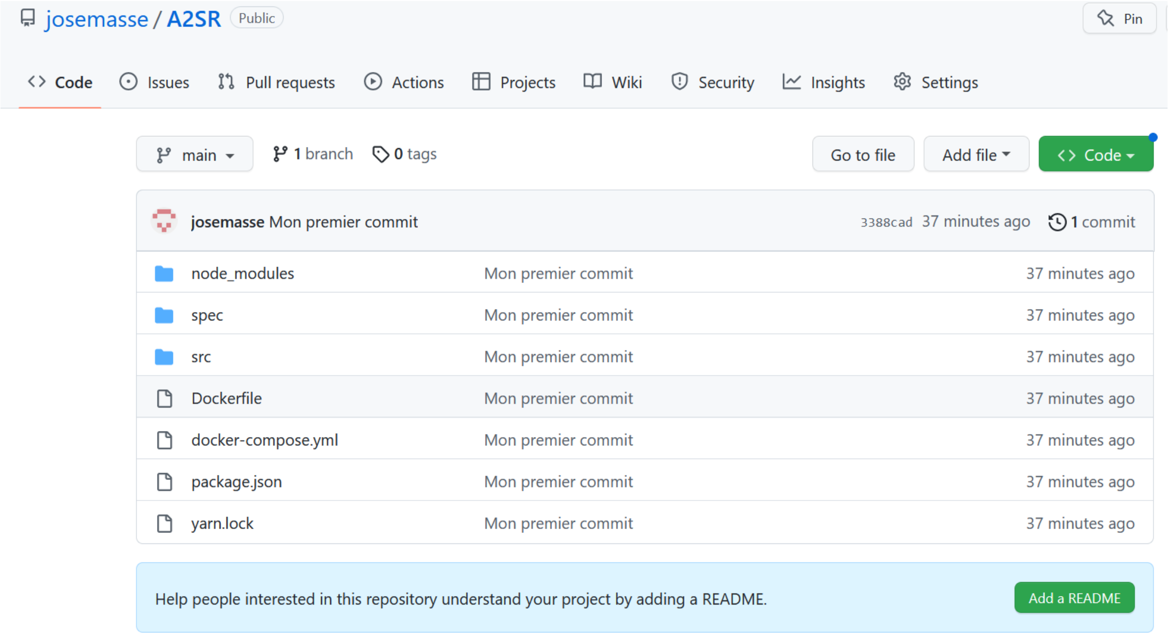
I had an issue here because authentification with a password doesn’t work anymore (since August 2021). Now we need to ask a connection token on GitHub.

It finally worked !



```
root@debian:~/jose/getting-started/app# git push -u origin main
Username for 'https://github.com': josemasse
Password for 'https://josemasse@github.com':
Énumération des objets: 6232, fait.
Décompte des objets: 100% (6232/6232), fait.
Compression des objets: 100% (5865/5865), fait.
Écriture des objets: 100% (6232/6232), 16.36 Mio | 5.58 Mio/s, fait.
Total 6232 (delta 1219), réutilisés 0 (delta 0), réutilisés du pack 0
remote: Resolving deltas: 100% (1219/1219), done.
To https://github.com/josemasse/A2SR.git
 * [new branch]      main -> main
La branche 'main' est paramétrée pour suivre la branche distante 'main' depuis 'origin'.
root@debian:~/jose/getting-started/app# 
```

This is my Github repository :



josemasse Mon premier commit 3388cad 37 minutes ago 1 commit

node_modules	Mon premier commit	37 minutes ago
spec	Mon premier commit	37 minutes ago
src	Mon premier commit	37 minutes ago
Dockerfile	Mon premier commit	37 minutes ago
docker-compose.yml	Mon premier commit	37 minutes ago
package.json	Mon premier commit	37 minutes ago
yarn.lock	Mon premier commit	37 minutes ago

Help people interested in this repository understand your project by adding a README.

Conclusion

Today we covered the use of docker and git. We learned how to install them. We have seen how to clone a GIT directory in order to get the Mathieu Perochon’s A2SR repository on GitHub.

What is GIT ? Git permits to share code with someone else, to handle versioning and to collaborate.

What is Docker ? It is a platform designed build, share and launch certain applications in software containers. Docker’s image are ready to use.

In addition, we discovered the use of the Dockerfile, we described the different options and how to configure it.

Finally we have seen how to launch, pull and push a docker image.

We also created a GitHub repository and commited our repository.