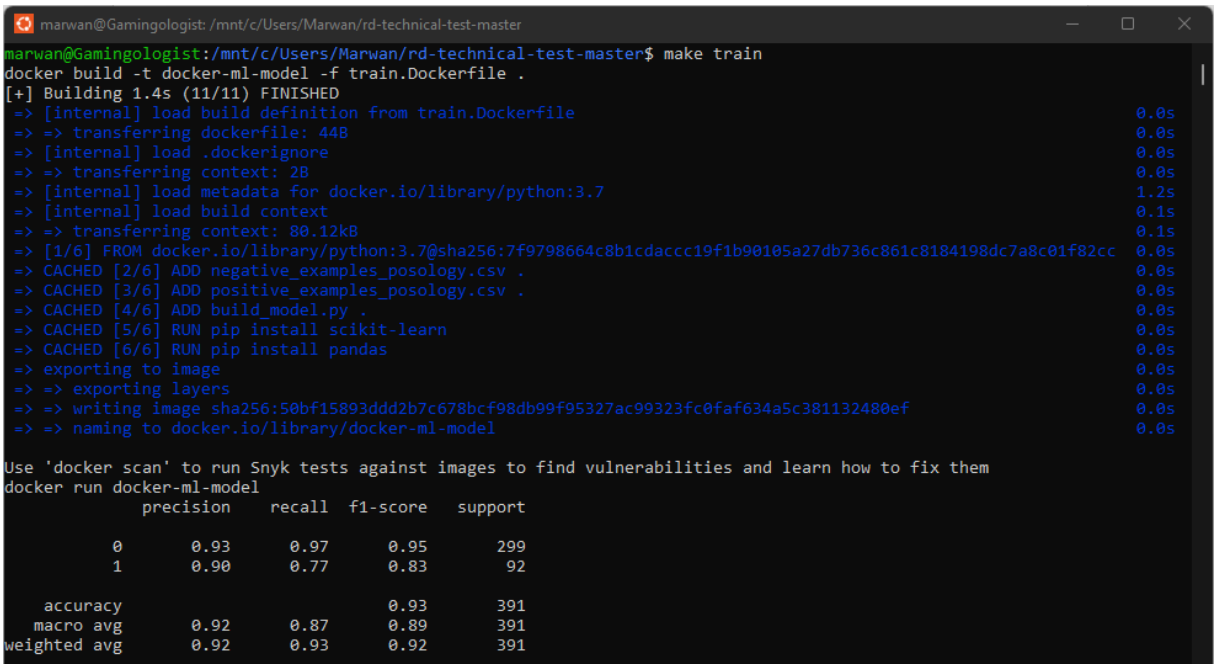
Report of technical test

POSOLOGY CLASSIFICATION

Marwan AL OMARI
PHD APPLICANT | MARWAN.AL.OMARI@ETU.UNIV-POITIERS.FR

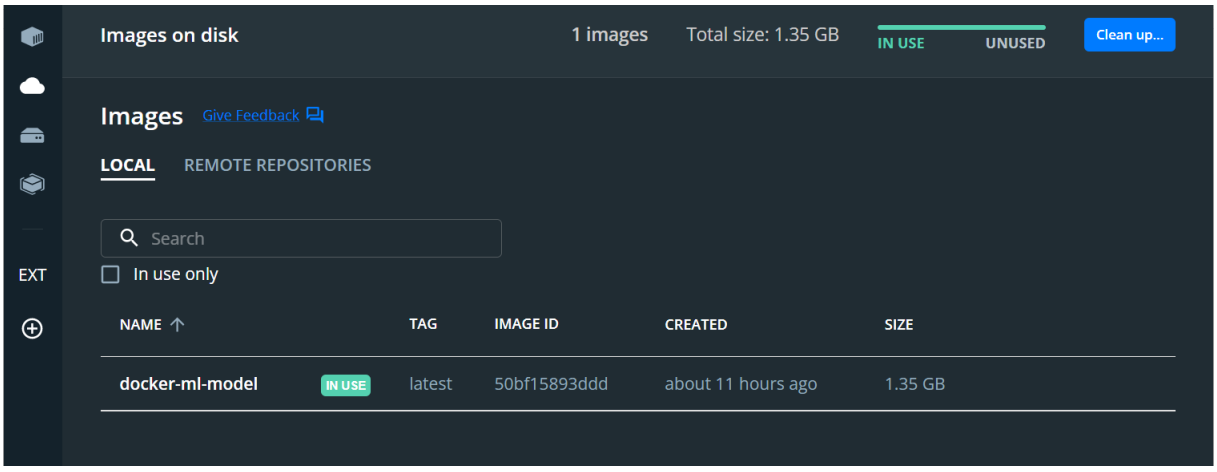
The training and the data preprocessing of the model are called by *make train*, which will initialize the model and create the docker image as well as the container. As we could see in the following figure (1), the docker image and container are created successfully.

Figure 1- make train command: create docker image and container with the trained model



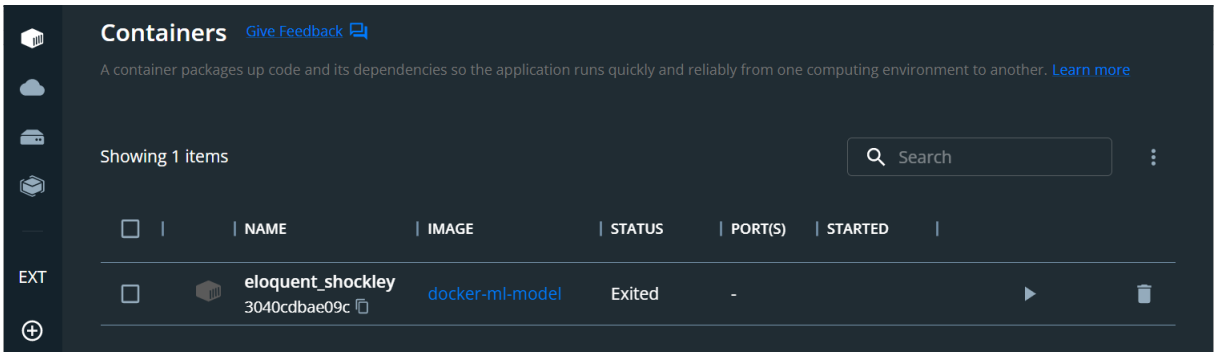
The following figure (2) shows the created image of machine learning model for production.

Figure 2- Docker image of the machine learning model



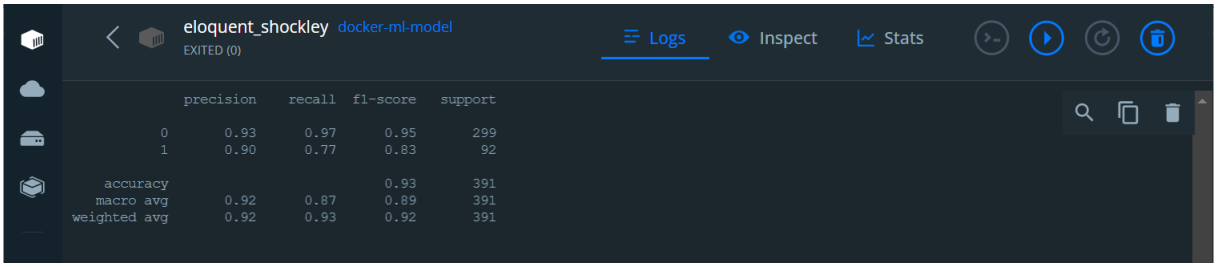
With the successful creation of docker image, the container is well placed for running the model and logging the model's progress. The container is well represented in the following figure (3).

Figure 3- Docker container of the machine learning model



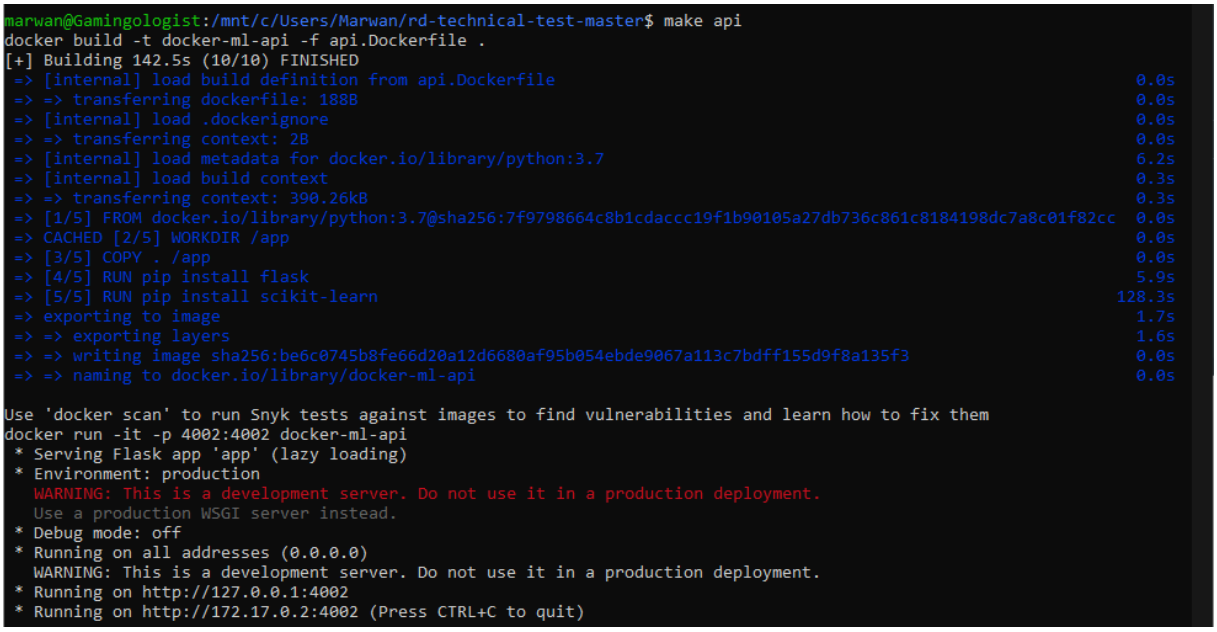
Therefore, in the following figure (4), we could observe the logging of the machine learning image, which is the results of logistic regression model.

Figure 4- The logs of the machine learning model



On the other hand, for putting the model in real production as a microservice, the command *make api* would take the trained model in the first image into production using flask api on 4002 ports. The command and the process are well presented in the following figure (5):

Figure 5- The deployment of the machine learning model



In the following two figure (6) and (7), the docker image and container of the deployed machine learning model are successfully created in docker.

Figure 6- Docker image of the deployed model

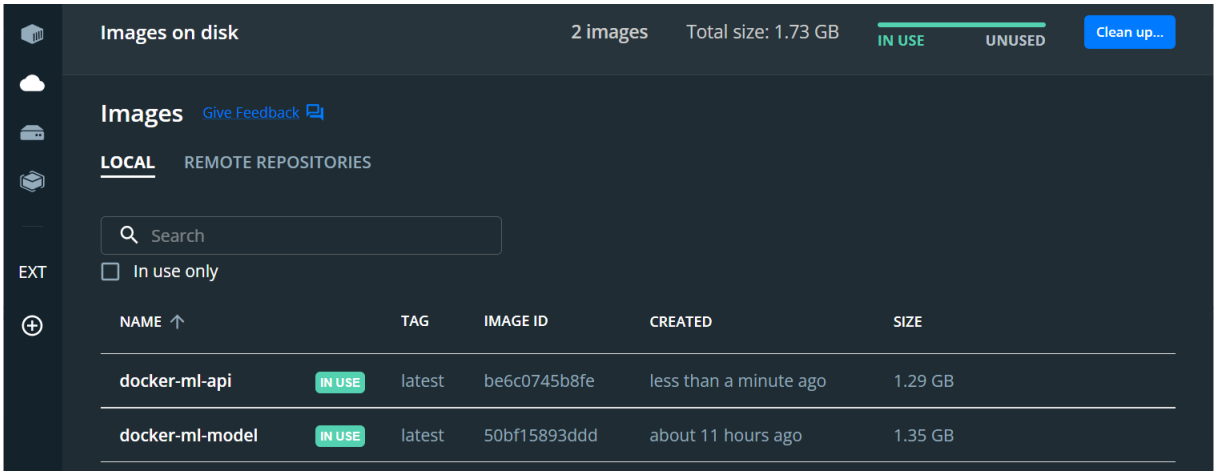
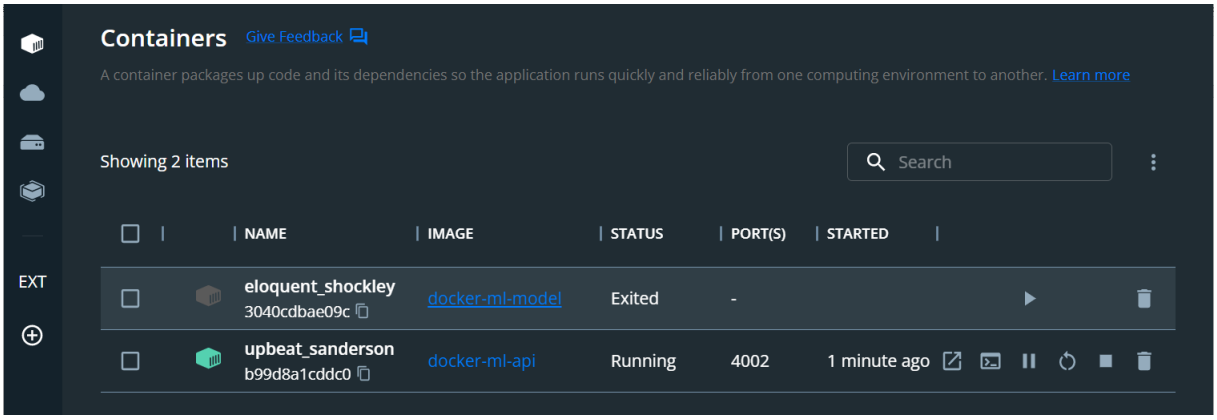
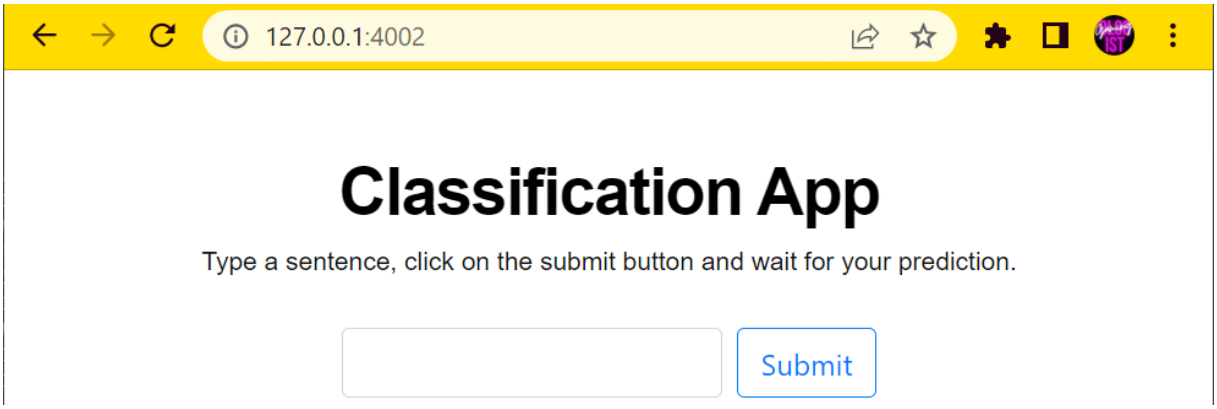


Figure 7- Docker container of the deployed model



Hence, with the deployment of the model, now, it is ready in production, as accessing the local host in port 4002, as it is well represented in the following figure (8):

Figure 8- the model in production



Moreover, the following two figures (9) and (10) represent two examples of questioning the model. An example outputs false posology and the other output true positive.

Figure 9- A negative example of the deployed model

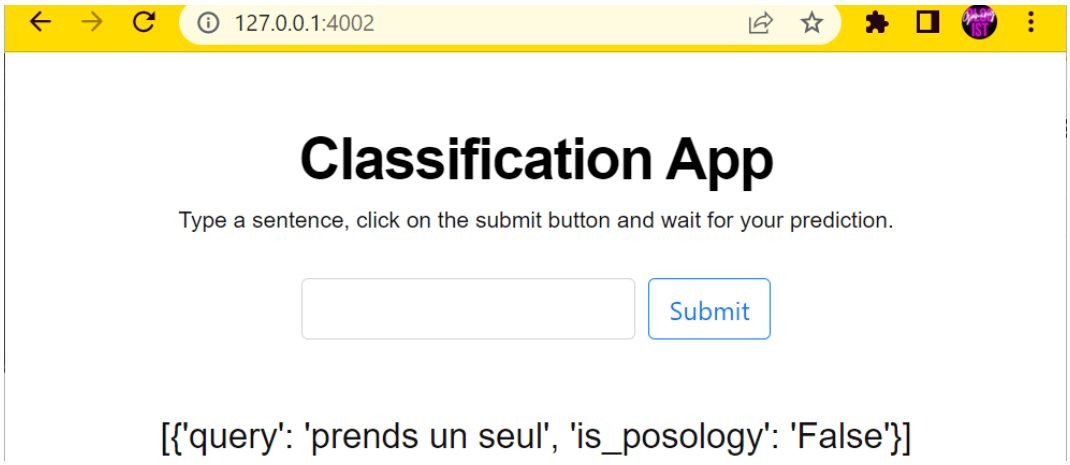
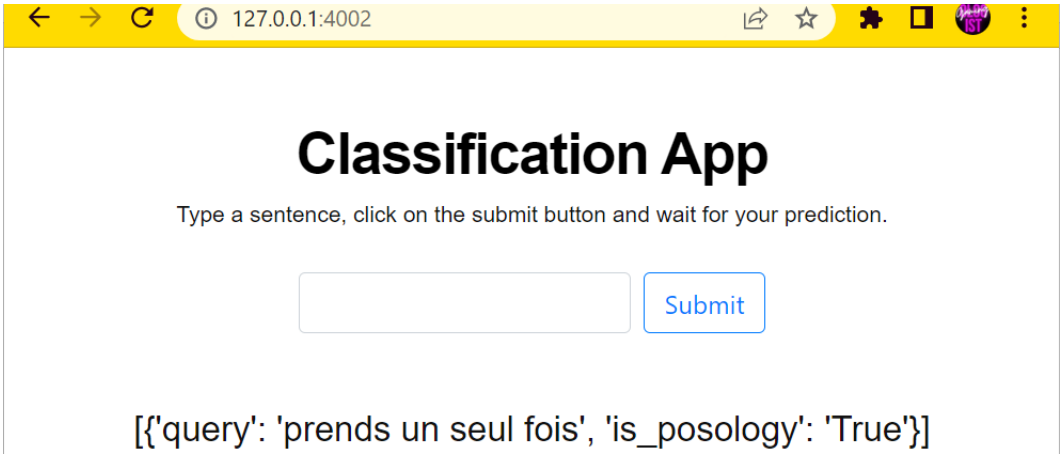


Figure 10- A positive example of the deployed model



In conclusion, as we could see in the following figure (11), the model is well on production and the docker shows the logs in terminal as we have sent the two examples presented before.

Figure 11- The logs of the deployed model