Week 10 Pilot Project

**Deploying Machine Translation Model**

Submitted to: Fouad Trad



Done By:

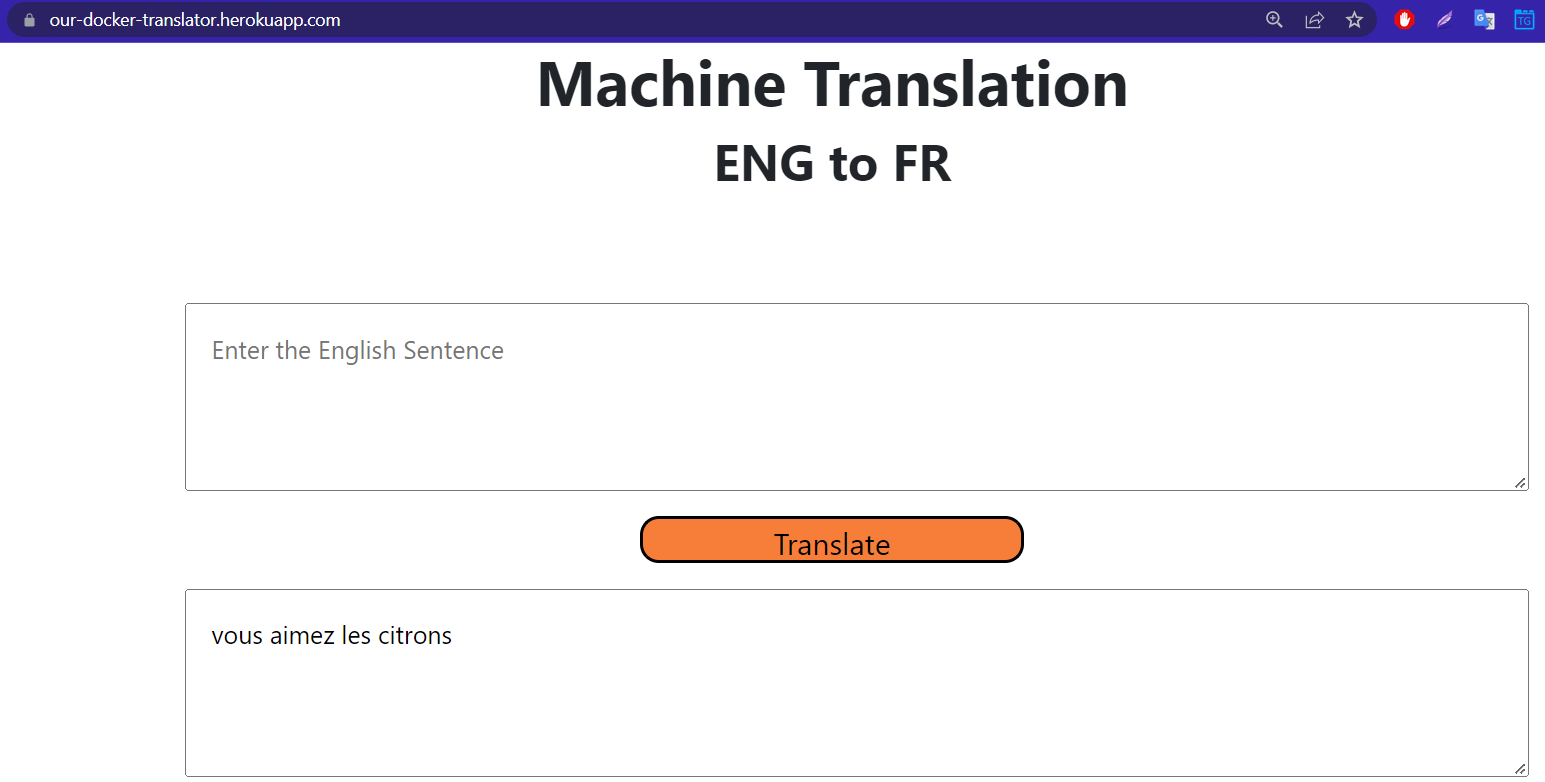
Lara El Ousman

Nabil Miri

Abdulrahim El Mohamad

English to French Translator model Deployment using Flask, Docker and Heroku. Our goal is to deploy our translator model on Heroku using Docker.

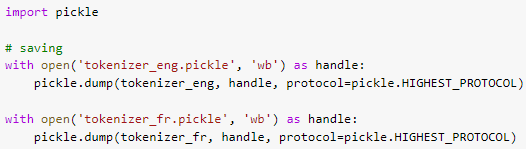
[Our Heroku Web App](https://our-docker-translator.herokuapp.com/) Link: <https://our-docker-translator.herokuapp.com/>

[](https://user-images.githubusercontent.com/75530842/194746157-926dda4c-77d1-4f22-86ea-aabca7696852.png)

Below steps were done to reach the final deployment app:

**1: Downloading the final model and tokenizers**

We downloaded the translator model that we created in Colab. Moreover, we also downloaded the english and french tokenizers using pickle library to use them in our deployment.

[](https://user-images.githubusercontent.com/75530842/194746400-96c44223-1b26-4307-aef4-962117378395.png)

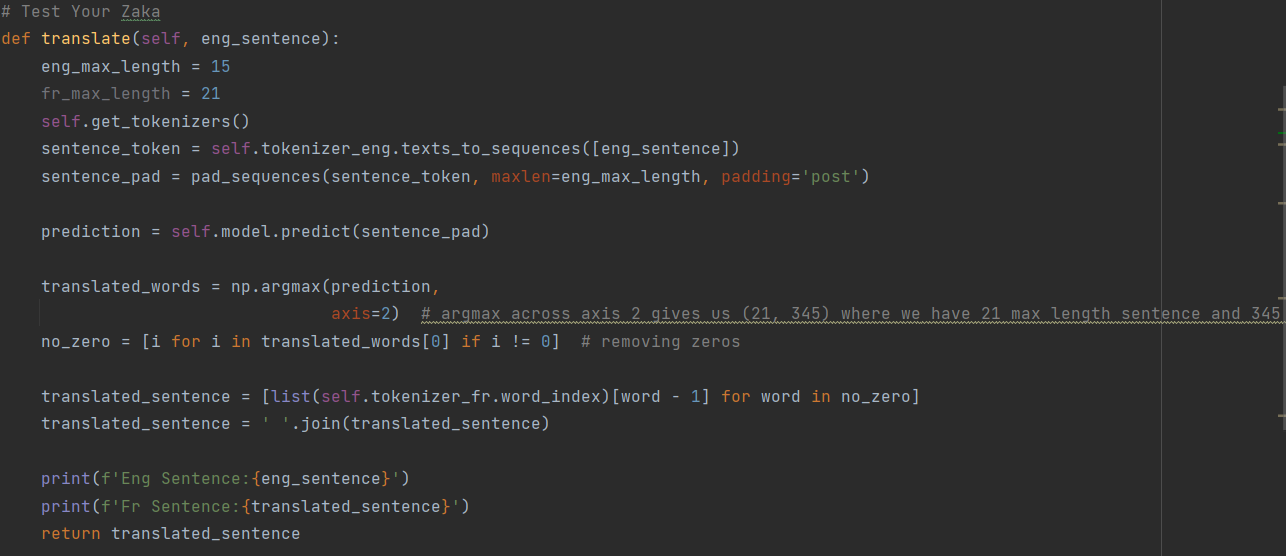
**2: Creating HTML page**

We have created a simple UI page using HTML to let the user interact with our model. [HTML File](https://github.com/Nabil-Miri/Translator-Heroku/blob/main/templates/HTML.html)

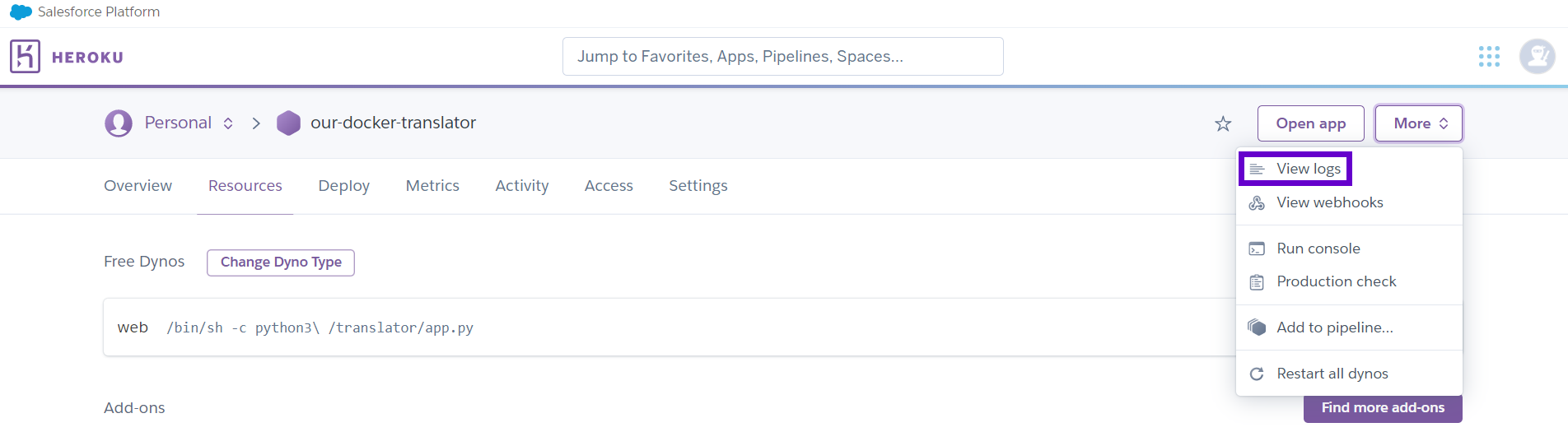
**3: Using Flask as framework**

We created [app.py](https://github.com/Nabil-Miri/Translator-Heroku/blob/main/app.py) which contains the flask code. This code also imports the EnFrTranslator class from [model.py](https://github.com/Nabil-Miri/Translator-Heroku/blob/main/model.py) file. app.py makes the Flask framework returns the HTML page and based on the user interaction it does specific tasks such as pushing the translate button will print the translated text in the french text box. When the user clicks the tanslate button, the input text is fed to the model after which they are tokenized.

Translation Code:

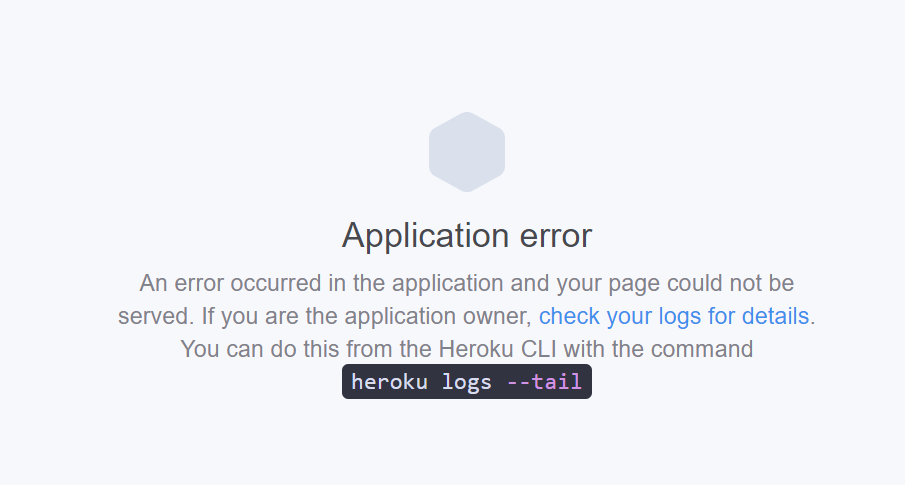
[](https://user-images.githubusercontent.com/75530842/194747173-4ee0c033-ddec-4c63-bff8-089b69585574.png)

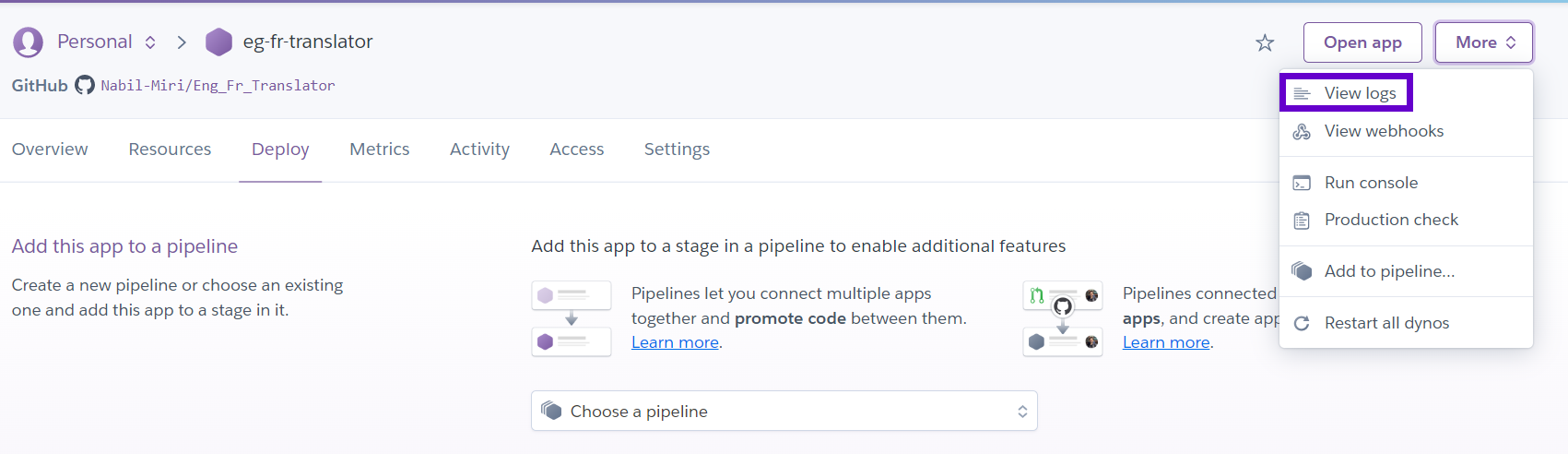
**4: Deploying model using Heroku (No Docker)**

There are 2 ways: #1) Connect directly the github repo to heroku using the website [](https://user-images.githubusercontent.com/75530842/194746016-3d48d800-8e9c-4172-b4a3-28ff94edc82f.png)

We first tried to connect to the github repo and it worked but we had some errors when deploying which couldn't be visualized this way. Thus, we used Heroku CLI and runned the commands using git bash.

*Note: Later it was discovered that log can be visualized on the website: :)*

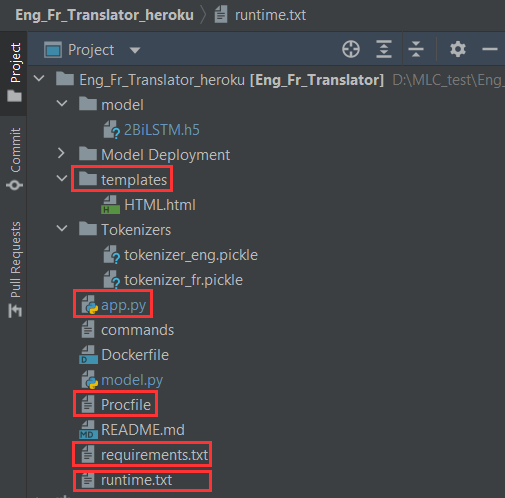
[](https://user-images.githubusercontent.com/75530842/194747319-23633e06-f001-4efb-b6df-b337ade7cf93.png)

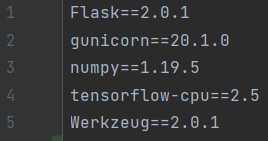
[](https://user-images.githubusercontent.com/75530842/194747343-6474991d-2207-493a-8b20-f2f9d947e3fc.png)

#2) Downloading Heroku CLI and using it to deploy the model

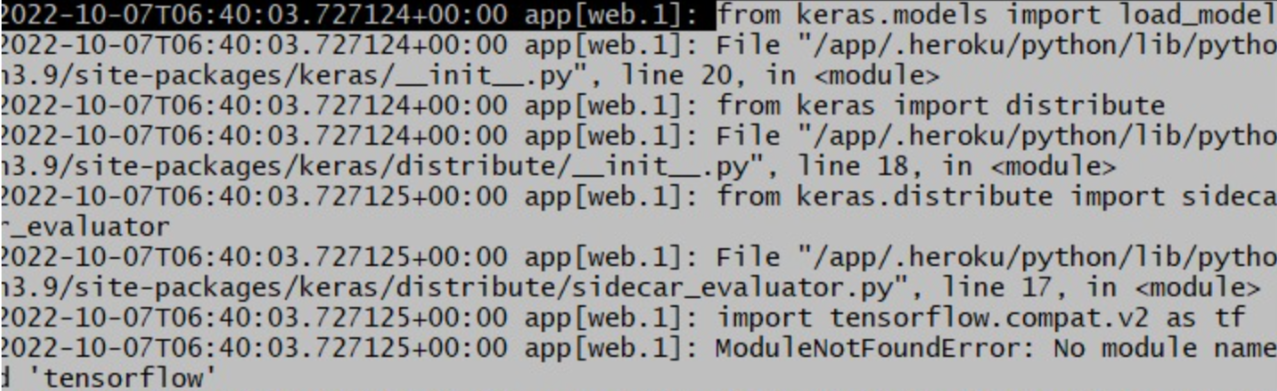
We used 'heroku logs --tail' to know what the errors were. Some of them are:

A) The file structure was not right, and we fixed it. The structure should be:

[](https://user-images.githubusercontent.com/75530842/194516888-235c3f91-201d-4993-b224-15e5a71ee705.png)

* runtime.txt: Used by Heroku to know which python version we want it to use and the list of supported versions are found on the website. [image](https://user-images.githubusercontent.com/75530842/194517148-0c4c57c8-3dca-4662-a871-a7effaa9afed.png)
* requirments.txt: Used by Heroku to know the needed libraries and their versions (also needed by Heroku) [](https://user-images.githubusercontent.com/75530842/194517474-6d59c34a-1cd6-43a9-8416-080170107910.png)
* app.py: our main code. it should be in the main root of the file so that heroku could detect that it is a python build (it can detect this also from the requirments,txt)
* Procfile: a file without extension for heroku to know the main file [image](https://user-images.githubusercontent.com/75530842/194517915-834bf512-e745-462a-8c6f-24a5ef7919e5.png) Note: if your main py file is not named as app, you could change it in the procfile also.

B) Tensorflow was not imported:

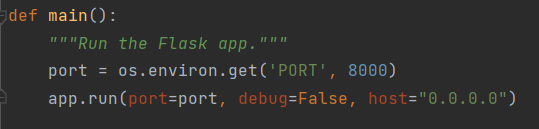
[](https://user-images.githubusercontent.com/75530842/194773405-e3fffbfd-6dba-4eeb-ae6a-7eb9c6d8223d.png)

Although we had no tensorflow importing, we got this error as it seems that something in the Keras module needs tensorflow. Thus, we imported it. But we got the following error (C).

C) Size of the Heroku Slug is > 500 MB

This happened when 'tensorflow' was in the requirments as it is around 450MB and more. After a lot of testing, we found 'tensorflow-cpu', a much smaller and compact version which solved the problem.

*Notes: We edited the flask code as follows:*

[](https://user-images.githubusercontent.com/75530842/194748805-224f377d-5b0a-45df-a9ab-443d343cb328.png)

It was recommended to do so for heroku.

When we do any change in the repo we run git push heroku main and after it finishes, we type heroku open which will open the link in the browser.

Code Steps:

heroku login

heroku create app\_name

git push heroku main # main: name of the branch

heroku open

Finally, it worked and we were able to deploy the model on heroku without Docker

**5: Deploying model using Heroku (Docker)**

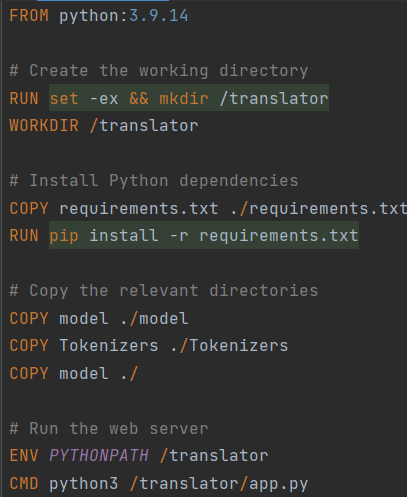
**Creating Docker Container**

First, the Docker file is prepared and built with all the required dependencies.

Steps:

docker login

then we prepare the Dockerfile:

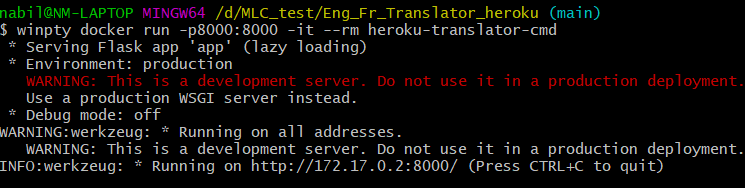
[](https://user-images.githubusercontent.com/75530842/194549461-19f449df-841c-43b1-a388-b71119b53a43.png)

*Note: We removed the 'EXPOSE 8000' from code. In the tutorials it was not written so we removed it just that it wont create any unnecessary error*

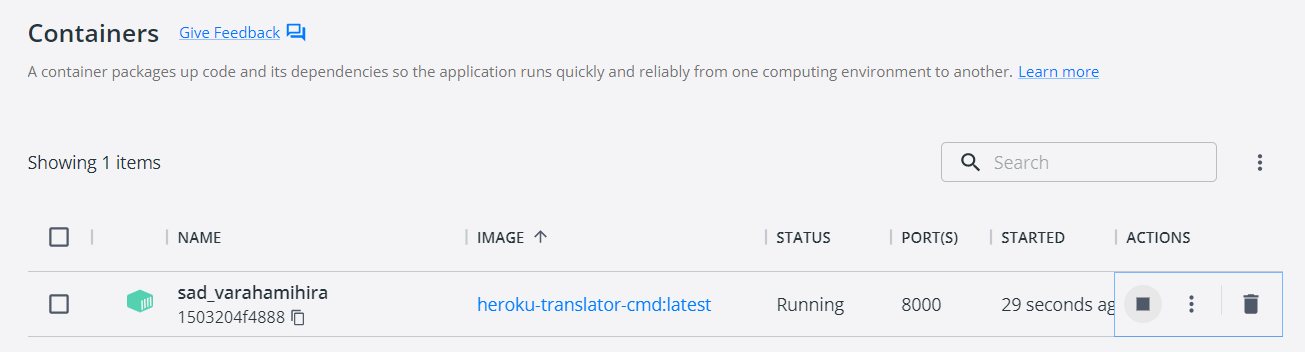
We build an image either using the docker plugin in PyCharm, which is connected to Docker Desktop or by using the docker command in the git bash. The command for building a docker image is:

docker build -t heroku-translator-cmd -f Dockerfile .

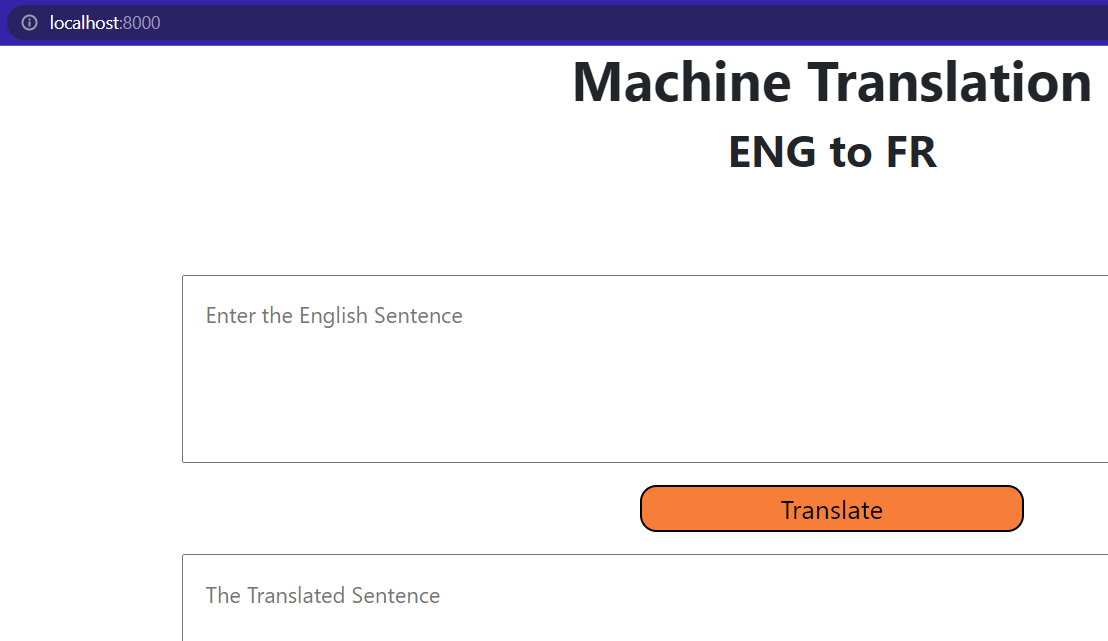
For running we use: winpty docker run -8000:8000 -it --rm our-heroku-translator

[](https://user-images.githubusercontent.com/75530842/194709746-a7615080-ee00-488b-af26-9c7b6fd7694a.png)

We can also run it from the Docker Desktop:

[](https://user-images.githubusercontent.com/75530842/194709756-1d39d1d8-78ac-4687-ad05-88f78b483c03.png)

Docker app running on the local host:

[](https://user-images.githubusercontent.com/75530842/194709775-d3faf236-8bf0-4cf4-85b3-d8f133afd658.png)

**Heroku and Docker**

After creating the container and checking that everything is alright, we used heroku.

The used set of codes:

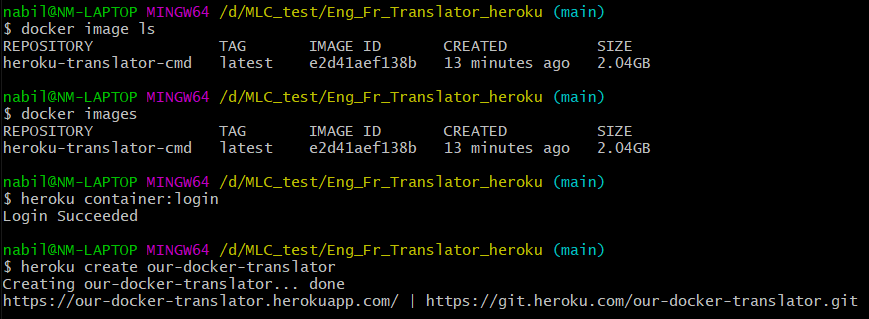
heroku container:login

heroku create app\_name

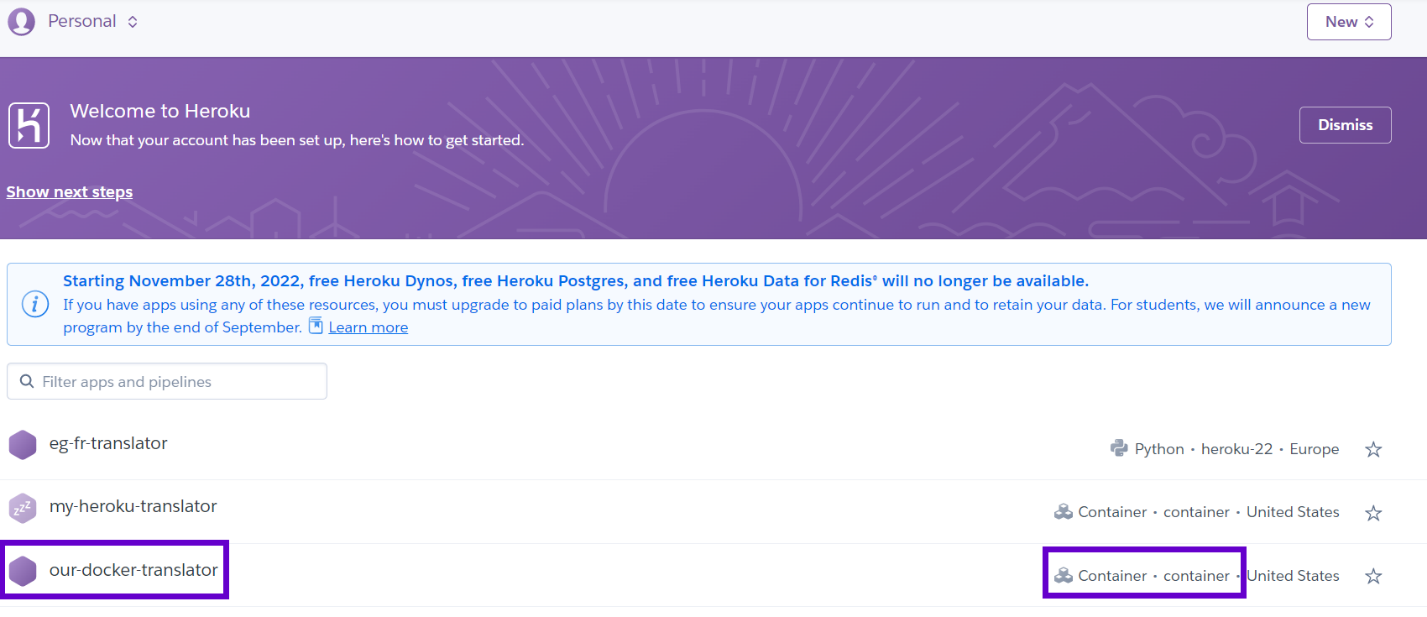
heroku container:push web

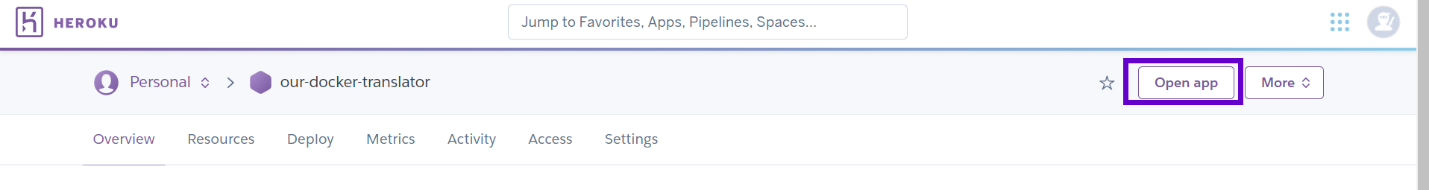
heroku container:release web

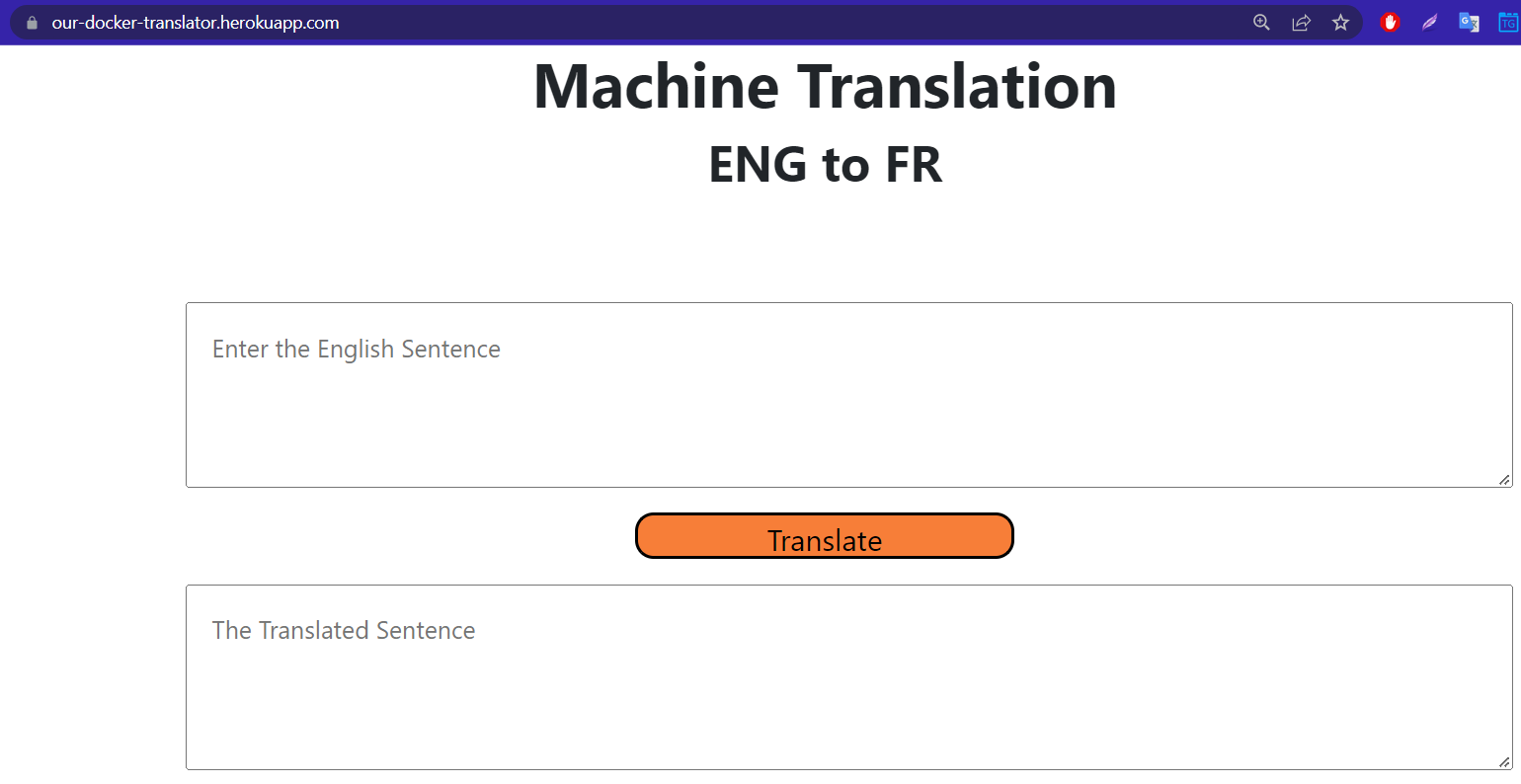
heroku open

[](https://user-images.githubusercontent.com/75530842/194710296-6fd951c2-b84e-4d57-ba0a-bdc98845b7da.png)

[image](https://user-images.githubusercontent.com/75530842/194710406-26a990fa-b3f3-404b-8bce-b0cf68c7daa8.png)

[](https://user-images.githubusercontent.com/75530842/194748729-cb6fb412-2ca3-4035-b3c6-b7d3b680bef1.png)

[](https://user-images.githubusercontent.com/75530842/194748748-25b69eb4-b25b-4773-b613-4c29ee6c2d8c.png)

[](https://user-images.githubusercontent.com/75530842/194748757-ad1f1459-37ff-4595-b0e3-6904a80558a5.png)

[The Heroku App Link](https://our-docker-translator.herokuapp.com/)