Hướng dẫn dùng docker để nộp bài cho Zalo AI Challenge 2022

1. Giới thiệu về Docker

Docker là một nền tảng giúp người dùng đóng gói và chạy chương trình của mình trên các môi trường khác nhau một cách nhanh nhất dựa trên các *container*.

Docker Image là một dạng tập hợp các tệp của ứng dụng, được tạo ra bởi Docker engine. Nội dung của các Docker image sẽ không bị thay đổi khi di chuyển.
Docker image được dùng để chạy các Docker container.

Docker Container là một dạng runtime của các Docker image, dùng để làm môi trường chạy ứng dụng.

Hướng dẫn chi tiết các bạn có thể tham khảo tại: https://docs.docker.com/get-started/

2. Cài đặt Docker trên Ubuntu

Đối với các hệ điều hành khác, tham khảo cách cài đặt tại đây https://docs.docker.com/install/overview/

1. Cài đặt docker

   > sudo apt-get install docker.io

2. Kiểm tra phiên bản docker

   > sudo docker --version

   ⚠️ Docker version 20.10.14, build a224086

3. Chạy thử docker hello world

   > sudo docker run hello-world

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

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

4. Một số câu lệnh phổ biến
    - Liệt kê các images hiện có

      > sudo docker images

    - Liệt kê các container hiện có

      > sudo docker ps -a

3. Cài đặt Nvidia Docker

Để sử dụng đươc GPU trên docker, bạn cần cài đặt Nvidia docker.

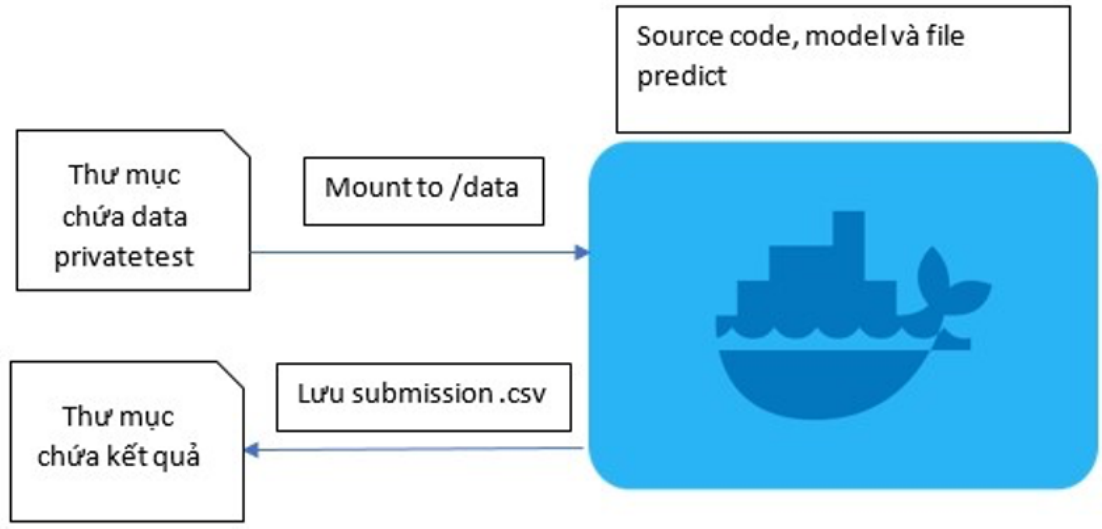
Hướng dẫn cài đặt Nvidia docker tại đây: https://github.com/NVIDIA/nvidia-docker

4. Nộp bài cho Zalo AI Challange

Các bước cần thiết để tạo được một Docker Image

1. Tạo một docker container mới (hoặc sử dụng docker có sẵn).
2. Đưa model và source code đang có vào docker.
3. Install các packages và libraries cần thiết mà solution của bạn sử dụng để chạy.

4. Viết một file script tên predict.sh. File này chứa command các bước để chạy test. Nhận input từ **/data** và output ra **/result/submission.csv** (tuỳ theo format quy định của đề bài). Ngoài ra, xuất ra các thông số thời gian load model, thời gian predict ở stdout.
5. Nộp file predict **predict_notebook.ipynb** để đo thời gian inference
6. Commit các thay đổi trong docker container
7. Save docker container thành 1 file image và nộp lên website cuộc thi.



5. Cấu trúc thư mục code

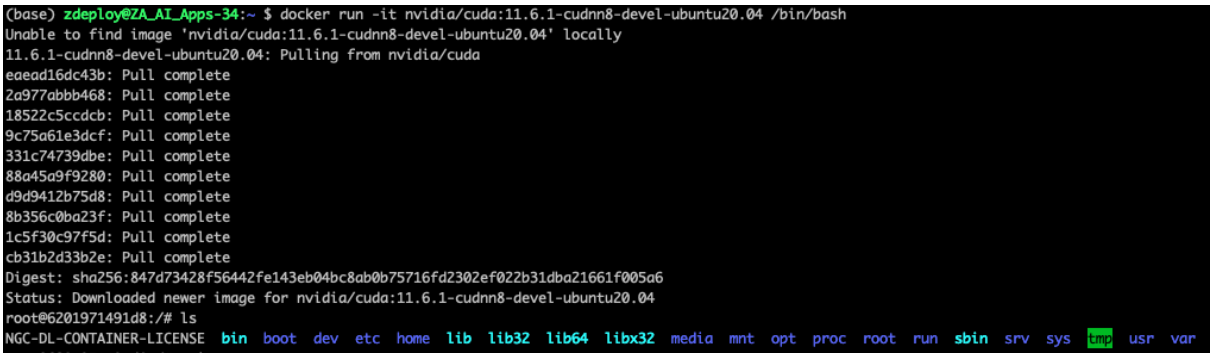
Ví dụ source code của bạn ở folder **/home/zdeploy/AILab/zac2022** với cấu trúc như sau.

```
|---- predict.py
|---- preprocessing.py
|---- saved_models
|       |---- models.h5
|---- train.py
|---- requirements.txt
|---- predict.sh
|---- start_jupyter.sh
|---- predict_notebook.ipynb # sử dụng để đánh giá thời gian, xem hướng dẫn viết notebook ở cuối
```

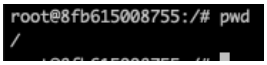
1. Khởi động docker container và đặt tên container với name **zac2022**.

> docker run --gpus '"device=0"' --network host -it --name zac2022 nvidia/cuda:11.6.1-cudnn8-devel-ubuntu20.04 /bin/bash

- Bạn **bắt buộc** phải sử dụng image base có cuda 11 và cudnn8-devel để tương tích với server chấm điểm của ban tổ chức.
- Cần phải có cờ **--network host** để thuận tiện run jupyter đo thời gian inference ở bước sau

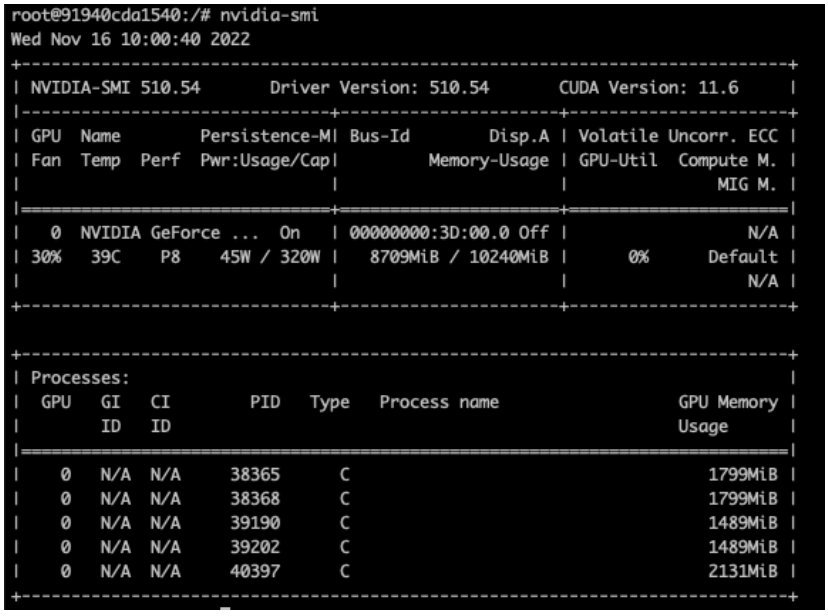


Lúc này trong container đang ở vị trí /



2. Kiểm tra container có sử dụng được GPU hay không bằng lệnh nvidia-smi

> nvidia-smi

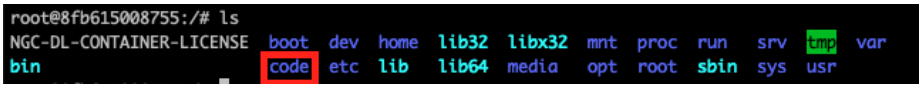


3. **Mở một terminal mới**, copy source code ở folder hiện tại vào trong container với cú pháp.

   sudo docker cp [source_path] [container_name]:[destination_path]

> sudo docker cp /home/zdeploy/AILab/zac2022/. zac2022:/code/

Lúc này toàn bộ source code ở ngoài đã được copy vào trong container ở vị trí /code trong **container**



3. Cài python và các package cần thiết

> apt update; apt-get -y install libgl1-mesa-glx libglib2.0-0 vim; apt -y install python3-pip

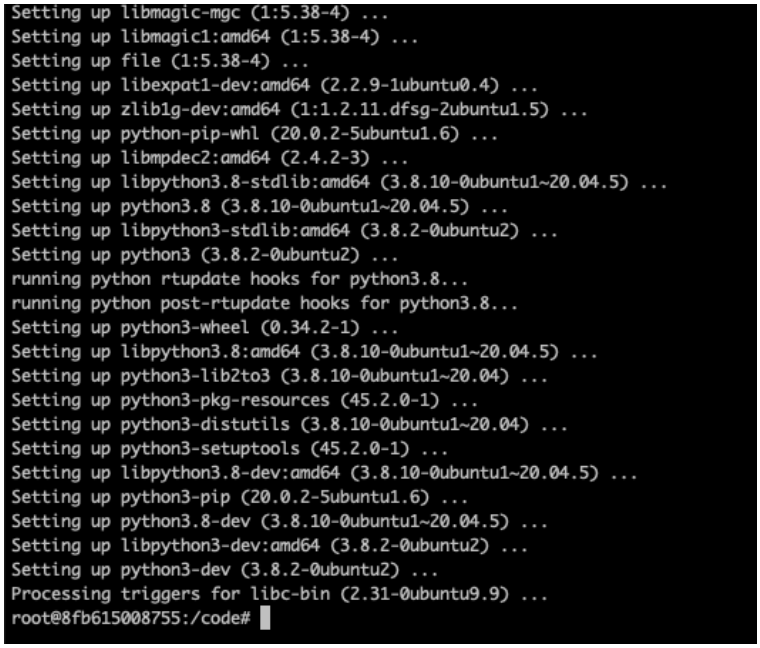
```
Please select the geographic area in which you live. Subsequent configurati

  1. Africa  2. America  3. Antarctica  4. Australia  5. Arctic  6. Asia  7
Geographic area: 6

Please select the city or region corresponding to your time zone.

  1. Aden      9. Baghdad   17. Chita       25. Dushanbe     33. Irkutsk
  2. Almaty    10. Bahrain  18. Choibalsan  26. Famagusta    34. Istanbul
  3. Amman     11. Baku     19. Chongqing   27. Gaza         35. Jakarta
  4. Anadyr    12. Bangkok  20. Colombo     28. Harbin       36. Jayapura
  5. Aqtau     13. Barnaul  21. Damascus    29. Hebron       37. Jerusalem
  6. Aqtobe    14. Beirut   22. Dhaka       30. Ho_Chi_Minh  38. Kabul
  7. Ashgabat  15. Bishkek  23. Dili        31. Hong_Kong    39. Kamchatka
  8. Atyrau    16. Brunei   24. Dubai       32. Hovd         40. Karachi
Time zone: 35
```

Nhập câu trả lời cần thiết nếu được hỏi



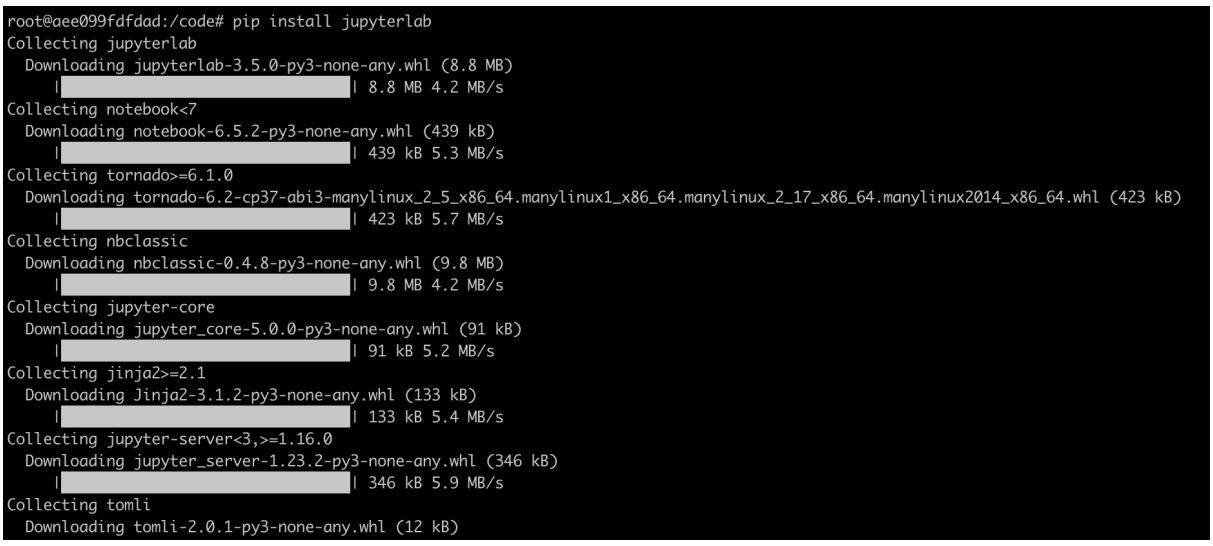
3. Cài đặt các thư viện cần thiết từ file requirements.txt hoặc tự install **trong container**

   Trước hết chuyển tới thư mục code bằng lệnh

   > cd /code

   > pip install jupyterlab
   > pip install -r requirements.txt
   > pip install numpy

   Lưu ý môi trường bắt buộc phải cài đặt **jupyterlab** để chấm thời gian inference



```
root@8fb615008755:/code# pip install -r requirements.txt

Collecting Pillow
\  Downloading Pillow-9.3.0-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (3.2 MB)
     |                                | 3.2 MB 2.1 MB/s
Installing collected packages: Pillow
Successfully installed Pillow-9.3.0
```

```
root@8fb615008755:/code# pip install numpy
python3Collecting numpy
  Downloading numpy-1.23.4-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (17.1 MB)
     |                                | 17.1 MB 5.2 MB/s
Installing collected packages: numpy
Successfully installed numpy-1.23.4
```

4. Chỉnh sửa file **/code/predict.sh**

   Ví dụ có 2 bước để chạy bộ test ra kết quả.

   python3 /code/preprocessing.py
   python3 /code/predict.py

   File predict.py phải ghi kết quả ở **/result/submission.csv hoặc .json (trong container, code phải tự tạo folder /result nếu folder chưa tồn tại)**

   > sh /code/predict.sh

```
root@8fb615008755:/code# sh /code/predict.sh
preprocessing...
output will be saved in /result/submission.csv
```

5. Chỉnh sửa file /code/start_jupyter.sh

```
jupyter lab --port 9777 --ip 0.0.0.0 --NotebookApp.password='zac2022' --NotebookApp.token='zac2022'  --allow-root --no-browser
```

6. Chỉnh sửa file **predict_notebook.ipynb: Xem phụ lục ở cuối**
7. Sau khi hoàn thành predict tiến hành lưu lại các thay đổi trên container. (**thao tác ở terminal**)

   sudo docker commit [CONTAINER ID] [image name]:[tag name]

   > sudo docker commit zac2022 zac2022:v1

```
(base) zdeploy@ZA_AI_Apps-34:~ $ docker commit zac2022 zac2022:v1
sha256:bfb23d6c81b027a6fc2333be60deaacc3d098141f4b61964eb7bb54d4d3c5c98
```

10. Kiểm tra docker lần cuối

    10.1 Kiểm tra hàm predict để xuất submission.csv

    Chạy predict.sh (các tham số về hyperparameters phải phải được set default trong file **predict.py**, BTC không chịu trách nhiệm nếu các bạn muốn thay đổi tham số này sau khi nộp bài)

    sudo docker run –v [path to test data]:/data –v [current dir]:/result [docker name]

    > sudo docker run --gpus '"device=0"' -v /data:/data -v /home/zdeploy/AILab/zac2022/:/result zac2022:v1 /bin/bash /code/predict.sh

```
(base) zdeploy@ZA_AI_Apps-34:~/AILab/zac2022 $ docker run -v /data:/data -v /home/zdeploy/AILab/zac2022/:/result zac2022:v1 /bin/bash /code/predict.sh
preprocessing...
output will be saved in /result/submission.csv
(base) zdeploy@ZA_AI_Apps-34:~/AILab/zac2022 $ ls
predict.py  predict.sh  preprocessing.py  requirements.txt  submission.csv
```

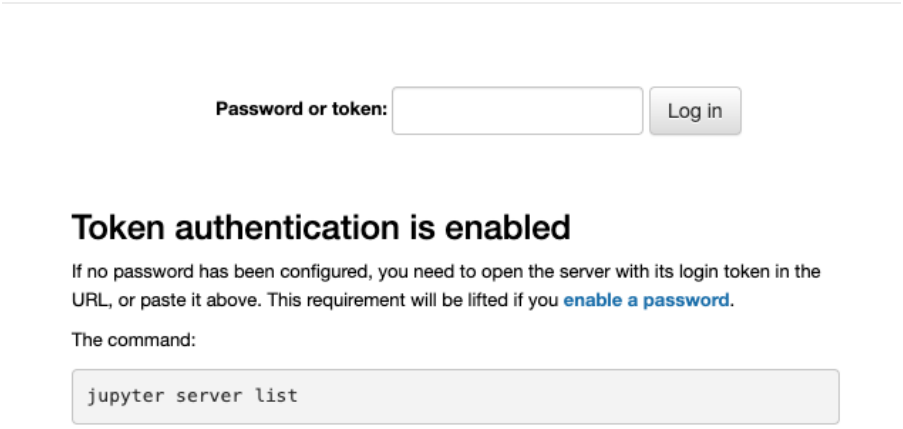
Kết quả của file submission.csv hoặc **.json** sẽ được trả về ở thư mục **/home/zdeploy/AILab/zac2022**

10.2 Kiểm tra jupyter có chạy được không.

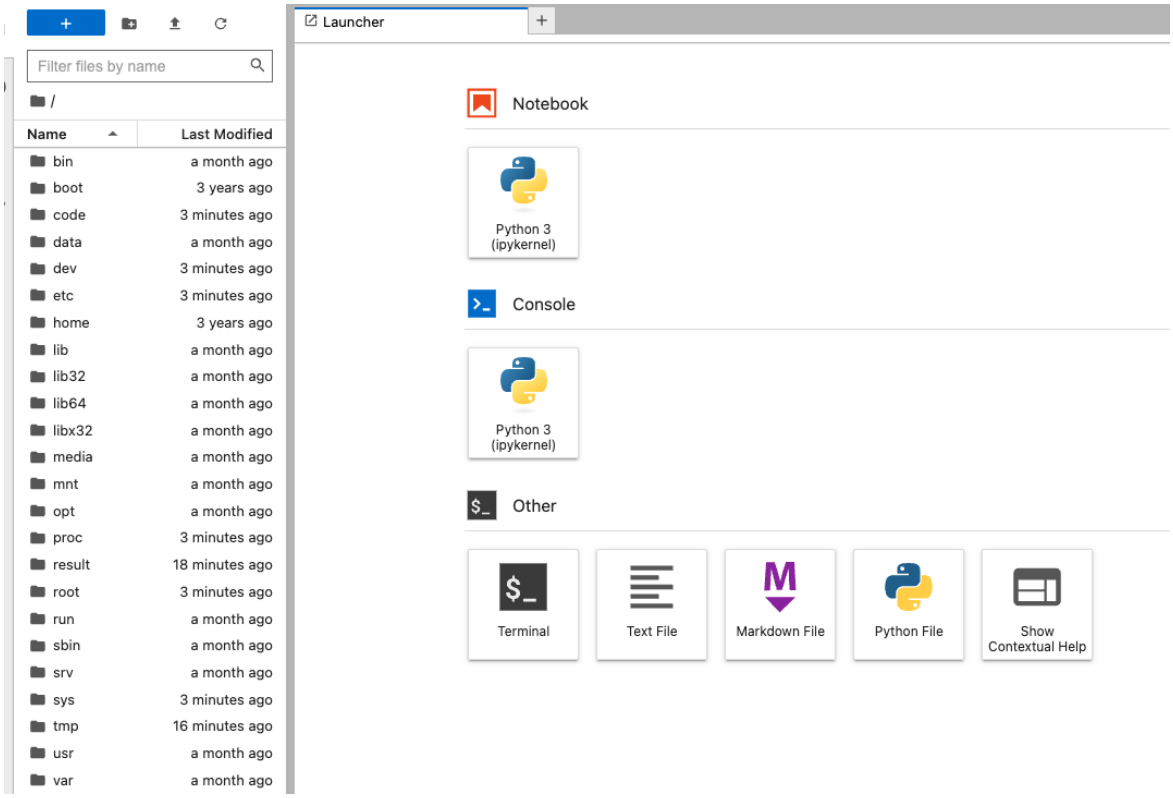
- Chạy start_jupyter.sh

> sudo docker run -it --gpus '"device=0"' -p9777:9777 -v /data:/data -v /home/zdeploy/AILab/zac2022/:/result zac2022:v1 /bin/bash /code/start_jupyter.sh

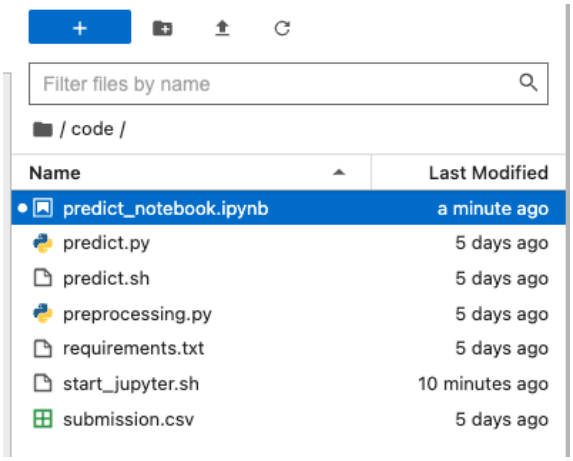
- Truy cập local host ở port 9777
- Nhập mật khẩu **zac2022**



- Giao diện cuối cùng sau khi cài đặt môi trường thành công



- Trong folder code phải có file predict_notebook.ipynb với nội dung như phần **phụ lục**



8. Xuất image ra và nộp bài.

   sudo docker save –o [output file] [image id]

   > sudo docker save -o zac2022.tar.gz zac2022:v1

   File **zac2022.tar.gz** là file bạn cần nộp.

6. Phụ lục: Cấu trúc viết Jupyter notebook để đo lường thời gian

Bên cạnh các tài liệu hướng dẫn thực thi các mã nguồn huấn luyện và dự đoán. Mỗi team cần chuẩn bị một Jupyter notebook để thuận tiện cho việc đánh giá thời gian. BTC sẽ dùng notebook này để đo lường thời gian chạy của mỗi test case.

Nội dung của file notebook này sẽ tương tự như file **predict.py,** BTC sẽ tính điểm thời gian từ bước load file cho tới bước inference ra kết quả cho mỗi testcase.

Notebook này sẽ thể hiện các bước của việc dự đoán được thực hiện tách rời nhau như: (1) Nạp mô hình và tài nguyên; (2) đọc dữ liệu test cases (3) thực hiện dự đoán. Do vậy notebook này phải gồm ít nhất 3 ô (cell) tương ứng với 3 bước chính:

- Bước 1: Nạp mô hình và các tài nguyên cần thiết

```
#
model = ...
```

- Bước 2: Đọc nội dung các test case từ thư mục đầu vào

```
# read all test cases
test_cases = ...
```

- Bước 3: Thực hiện dự đoán và in ra kết quả cũng như thời gian.

```
#
all_predicted_time = []
all_result = []
for file_name in test_cases: # hoặc for item in test_cases tuỳ theo từng task
  t1 = time()
  input_ = preprocess(file_name) # convert file_path into expected input
  forward = model.predict(input_) # model forward
  result = postprocess(forward) # postprocess to expected format result
  t2 = time()
  predicted_time = int(t2*1000 - t1*1000)
  all_predicted_time.append((file_name, predicted_time))
  all_result.append(result)

write_predict_file(all_result) # output with the same format as predict.py
write_time_file(all_predicted_time) # write to time_submission.csv
```

(Mã nguồn mang tính chất tham khảo tuỳ theo từng task)

Lưu ý: BTC chỉ mở file notebook và run all cells, nhiệm vụ của các bạn phải đảm bảo các cells khi chạy không bị lỗi ở bước nào, và cuối cùng phải xuất ra được 2 files

1. **time_submission.csv** với format gồm 2 cột là **fname** và **time (millisecond)**
2. **jupyter_submission.csv / .json / .zip** với nội dung tương tự file **submission.csv / .json / .zip**

   Để tránh bị overwrite kết quả khi chạy bước predict.py, đối với các kết quả được xuất ra khi chạy jupyter, các bạn phải thêm prefix là "**jupyter**_"

Các đội phải đảm bảo khi BTC chấm điểm thì điểm của file jupyter_submission.csv / .json / .zip sẽ phải giống với submission.csv / .json / .zip thì mới bắt đầu chấm điểm qua phần time_submission.csv

7. Upload docker

1. Lấy checksum MD5 của file docker đã nén ở bước 8

   Tham khảo thêm cách check MD5 ở Windows và MacOS:

   https://portal.nutanix.com/page/documents/kbs/details?targetId=kA07V000000LWYqS

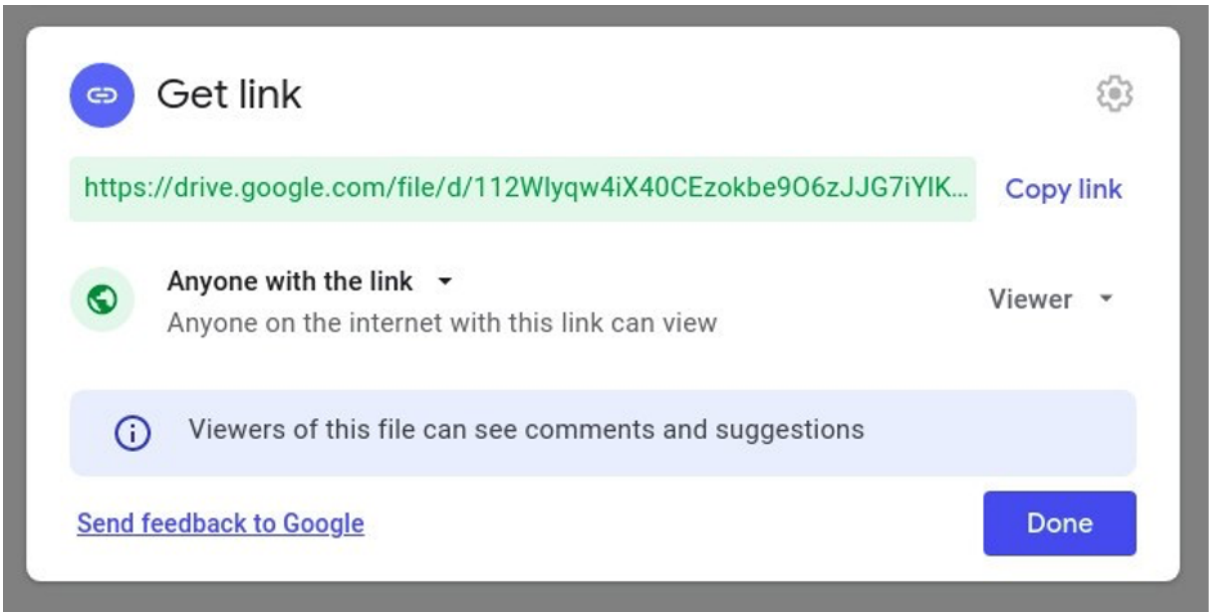
   Linux: https://www.geeksforgeeks.org/md5sum-linux-command/

2. Upload docker

   BTC sẽ download docker của các bạn về máy chủ để tiến hành bước kiểm tra kết quả cuối cùng.

   Upload file docker đã nén lên dịch vụ storage của Google drive Chỉnh sang chế độ share “Anyone with the link”

   Gửi URL download cho BTC và checksum MD5 ở bước 3.



Chúc các bạn thành công.