UNIVERSITÉ CÔTE D'AZUR

POLYTECH NICE-SOPHIA

Informatique pour la robotique 2

TROISIEME PARTIE – EXPLOITER LA JETSON NANO

Enseignant : Maëva LECAVELIER – maeva.lecavelier@gmail.com

Plan

- Objectifs et travail à faire
- Lien Arduino - Jetson
- Hello World IA-Arduino

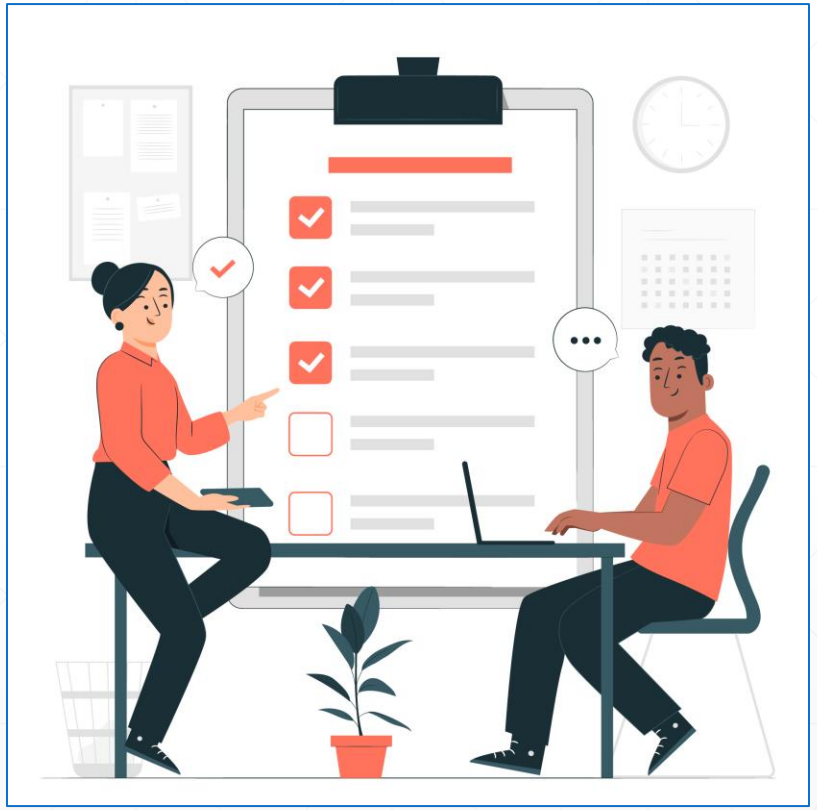
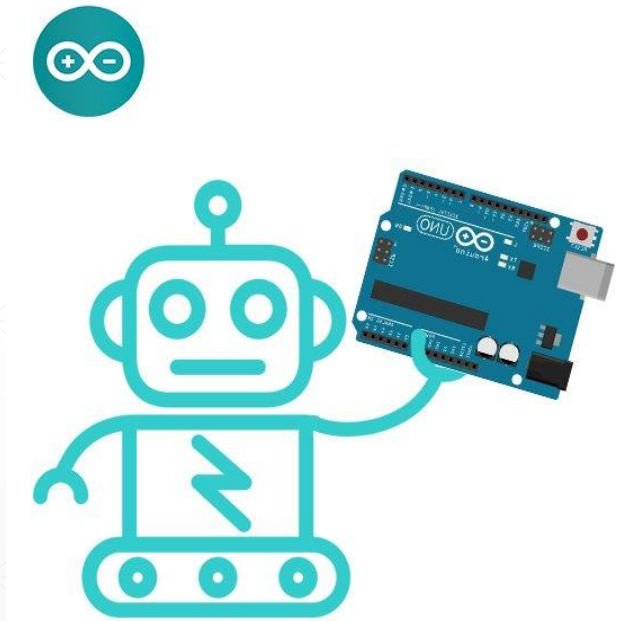


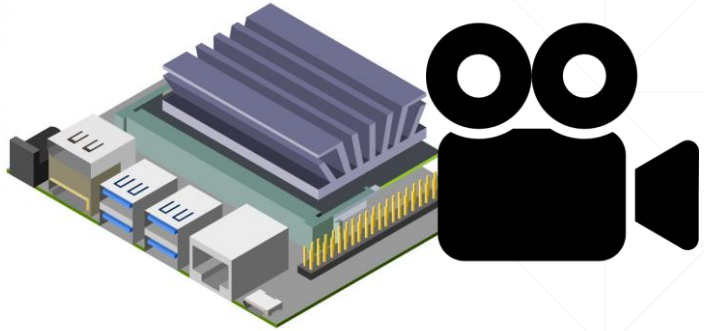
Image de storyset sur Freepik

Objectifs et devoirs

Objectifs



Chien détecté



Chien détecté ! Coordonnées GPS transmises

Objectifs

- En d'autres termes : ajouter de la reconnaissance d'image à votre robot

Dans un premier temps… Ensuite, si vous souhaitez aller plus loin et rajouter de l'IA sur les données de vos capteurs, ou utiliser la Jetson Nano à d'autres fins, vous le pouvez.

Travail à faire (03/04)

- **Pour mercredi 3 avril** : envoyer à M. Chatoev et Mme Lecavelier une présentation de quelques slides de votre robot :
  - nom des étudiants
  - nom du robot
  - objectif du robot
  - quelques éléments techniques : synoptique, machine à état, architecture de votre robot et de votre code, les risques identifiés s'il y en a
  - **Et comment vous allez utiliser la Jetson Nano dans votre robot** : si vous avez plusieurs idées, notez les

Pas de notes prévues pour ce rendu, c'est surtout pour que nous connaissions vos projets et puissions vous aider.

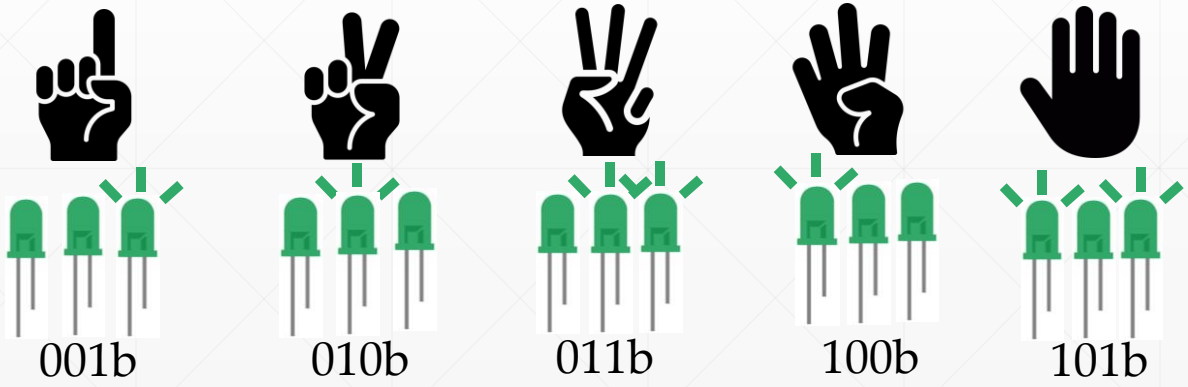
Travail à faire noté (17/04)

- **Pour le mercredi 17 avril** : envoyer à M. Chatoev et Mme Lecavelier un Jupyter Notebook faisant le compte rendu du travail suivant :

Vous devez développer (ou réadapter…) un code permettant de compter les doigts d'une main et afficher sur l'Arduino le nombre de doigts comptés.

Vous joindrez également **le code Arduino** (que nous exécuterons sur une Arduino Uno), le **schéma de câblage** et **les données utilisées pour entraîner votre modèle**.

Suggestion : vous pouvez utiliser trois LEDs pour compter en binaire. Exemple :



Travail à faire noté

Nous voulons connaître votre démarche allant de la connexion I2C entre les cartes, à l'affichage sur l'Arduino.

Nous exécuterons le code sur notre Jetson pour vérifier qu'il soit fonctionnel et que le Jupyter Notebook contient toutes les commandes nécessaires. Nous ne noterons pas l'efficacité de votre IA

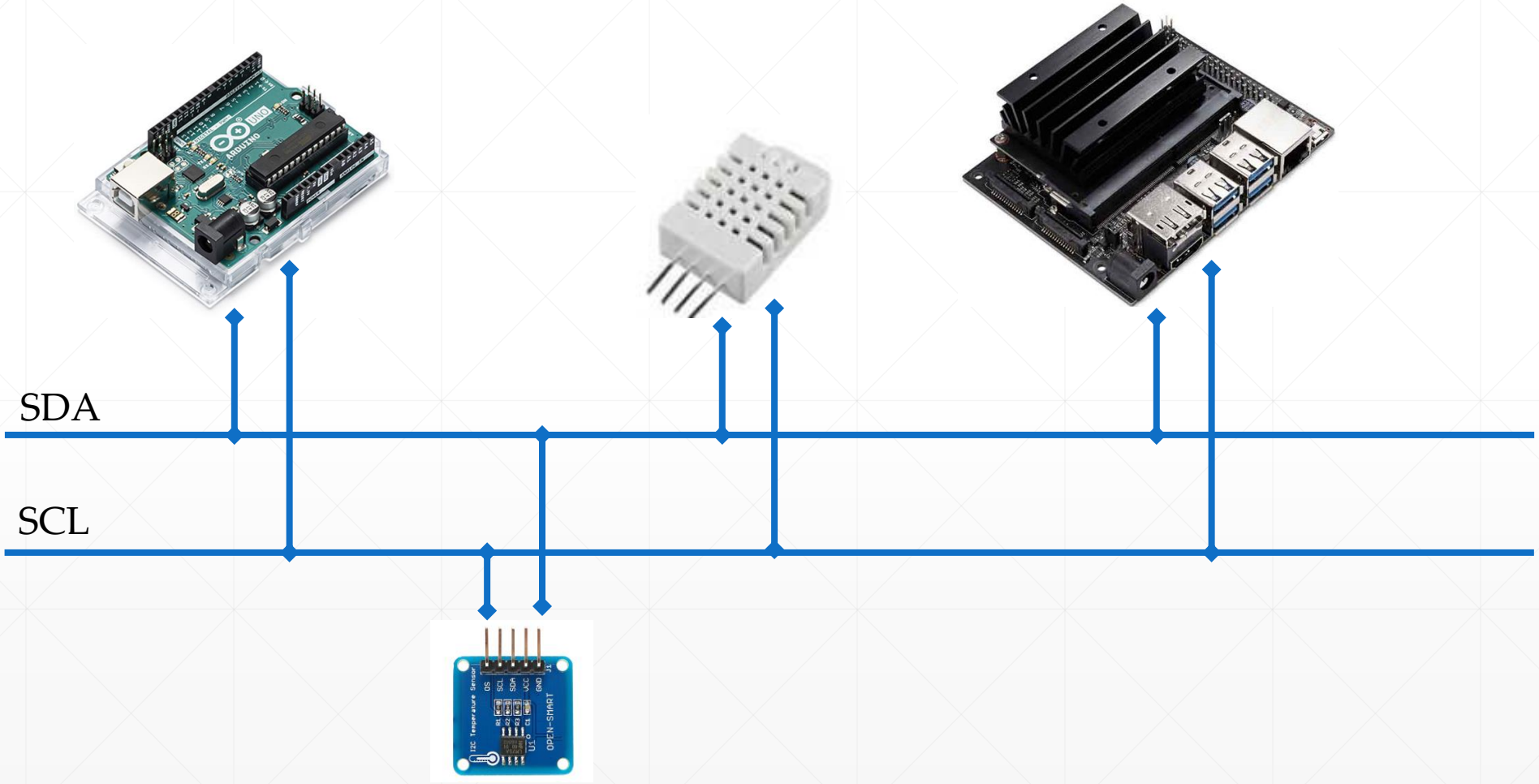
Critères d'évaluation :

- Qualité de la rédaction de la documentation/rapport (introduction et contexte, problématique, difficultés rencontrées, code commenté et/ou sourcé)
- Réalisation de l'objectif initial
- Réflexion sur la solution
- Qualité du code

Toute trace de travail sera valorisée et prise en compte.

Objectifs aujourd'hui

Communication I2C : contexte



Lien Arduino - Jetson

Envoyer des informations depuis la Jetson vers l'Arduino

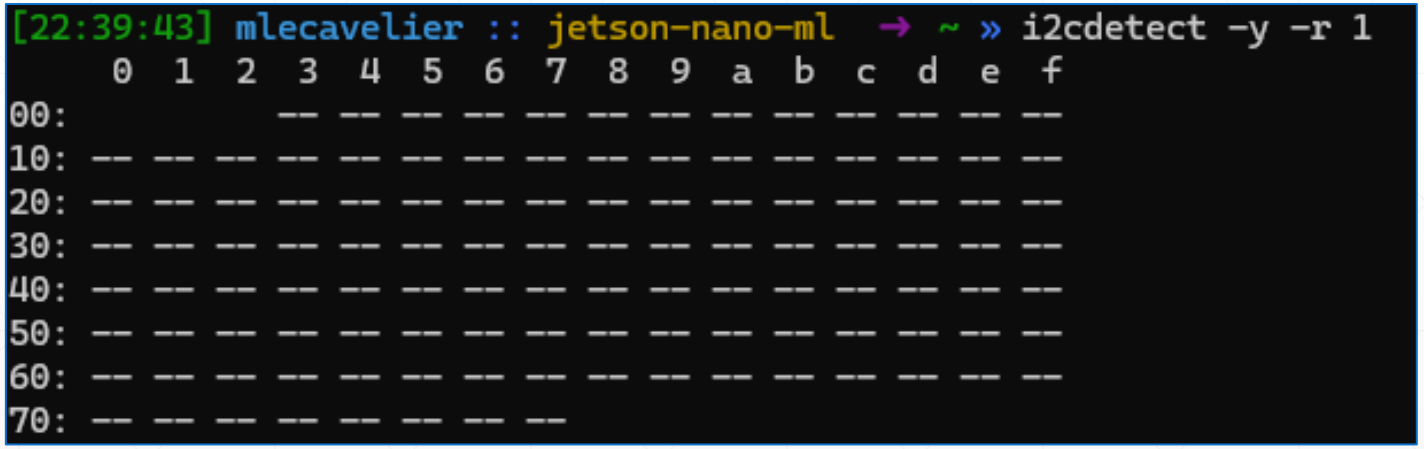
Communication I2C : première étape

Permettre la communication I2C sur la Jetson

Il va falloir faire un peu de configuration Linux… mais maintenant vous connaissez ☺

Ressources : Internet

Vous pouvez passer à l'étape suivante quand `i2cdetect -y (0, 1 ou 2)` s'exécute correctement. Exemple :



Communication I2C : deuxième étape

Connecter l'Arduino à la Jetson Nano, et vérifier que la Jetson détecte l'Arduino.

Votre Arduino devra faire tourner un code « slave » pour que la Jetson la détecte

- Vous pouvez passer à l'étape suivante quand `i2cdetect -y (0, 1 ou 2)` renvoie une adresse valide. Exemple :

```
[22:43:45] mlecavelier :: jetson-nano-ml  → ~ » i2cdetect -y -r 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- 14 -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
[22:43:47] mlecavelier :: jetson-nano-ml  → ~ »
```

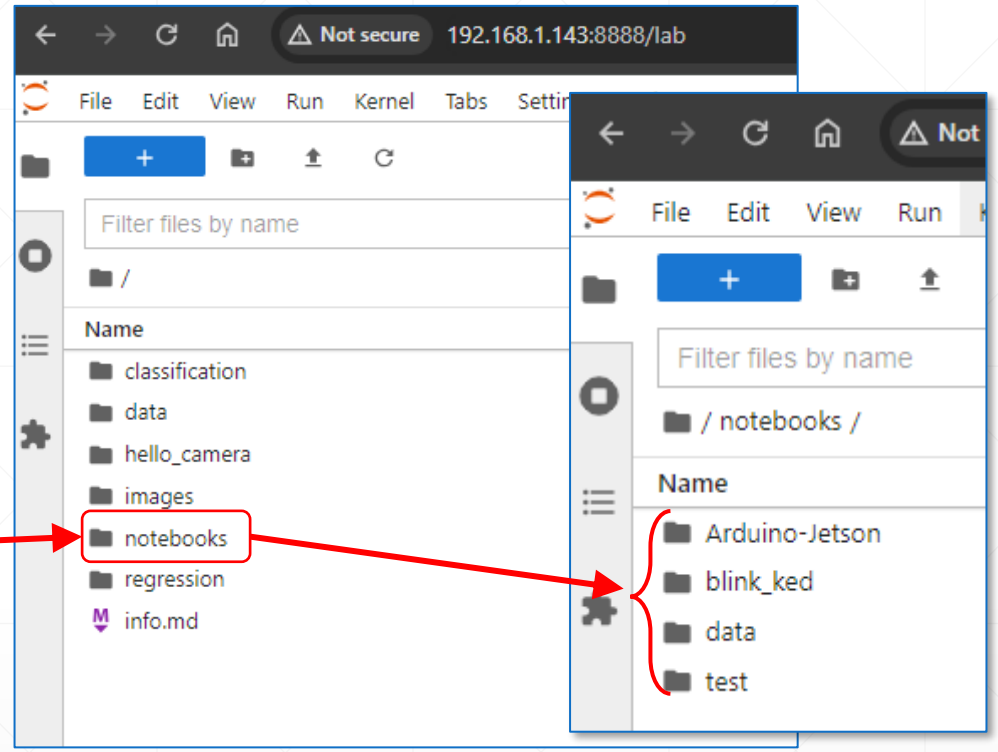
Aparté : sauvegarde Docker

!! Recommandation importante !!

Pour sauvegarder vos notebooks personnels, je vous conseille de rajouter cette ligne également :

```
--volume ~/notebooks:/nvdli-nano/notebooks
```

Cela signifie que vos Jupyter Notebooks créés dans le dossier /nvdli-nano/notebooks dans votre docker, seront sauvegardés dans le dossier notebooks de votre Jetson. En image :




```
[22:04:31] mlecavelier :: jetson-nano-ml  → ~ » hostname
jetson-nano-ml
[22:04:37] mlecavelier :: jetson-nano-ml  → ~ » pwd
/home/mlecavelier
[22:04:41] mlecavelier :: jetson-nano-ml  → ~ » ls notebooks
Arduino-Jetson  blink_ked  data  test
[22:04:43] mlecavelier :: jetson-nano-ml  → ~ »
```

Communication I2C : troisième étape

- Vérifier la communication I2C entre la Jetson Nano et l'Arduino, via le Jupyter Notebook.

Vous allez devoir modifier votre commande docker en rajoutant quelques flags :

```
sudo docker run --runtime nvidia -it --network host \
--volume ~/nvdli-data:/nvdli-nano/data \
--device /dev/video0 --device /dev/gpiochip0 \
--privileged \
nvcr.io/nvidia/dli/dli-nano-ai:v2.0.2-r32.7.1
```

Utiliser Jupyter Notebook pour développer sur la Jetson

- **Faites une première connexion simple entre les deux. Vérifier le résultat dans la console de l'Arduino**

Pour cela, créer votre propre Jupyter Notebook, et utiliser Python pour communiquer en I2C depuis la Jetson.

Librairie python conseillée : smbus.

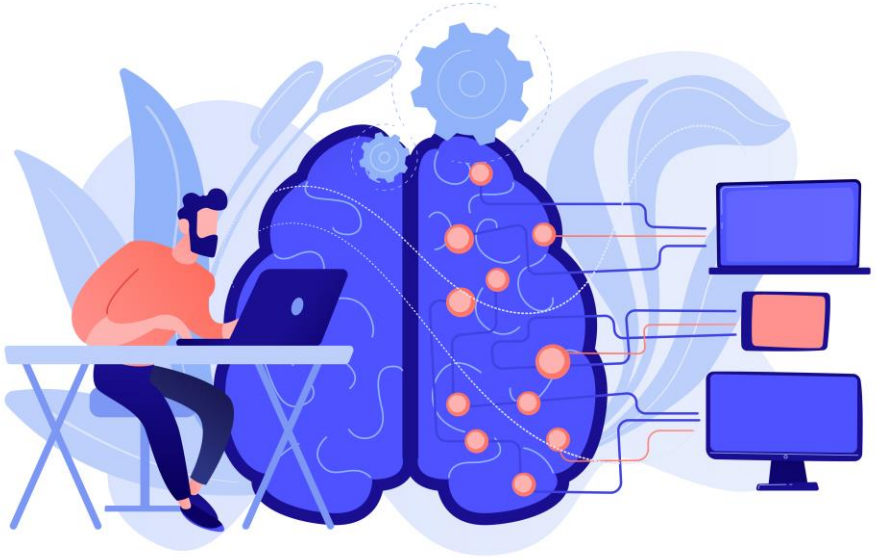
N'oubliez pas de l'installer directement depuis votre Jupyter Notebook (utile pour le rendu du 17 avril ☺)

Hello World IA-Arduino

La Jetson traite une image, et envoie le résultat à l'Arduino, qui réagit

L'IA sur la Jetson

- Les modèles présents sur la Jetson, se basent sur du Machine Learning. Cela veut dire que la machine s'entraîne sur des données, et grâce à son entraînement, elle est capable de prédire des résultats sur des données similaires.
- Le secret d'une bonne « IA » sur la Jetson, va principalement se baser sur **l'entraînement**. Plus un modèle aura de référence d'images, plus il sera performant. Donc, dans le cadre de ce projet, vous allez devoir prendre beaucoup de photos de votre cible pour que votre IA soit performante.



Thumb up – thumb down : LED verte, LED rouge

- Reprenez le code NVIDIA du projet thumb-up/thumb-down et rajoutez-y l'envoi du résultat vers l'Arduino.
  Vous êtes libre de choisir le format du résultat pour l'interprétation de l'Arduino (1 = thumb-up, 0 = thumb-down; UP = thumb-up, DOWN = thumb-down…, tout est possible)

- Ajouter un indicateur visuel sur l'Arduino pour visualiser le résultat : la LED s'allume vert si le pouce est en l'air, et en rouge si le pouce est vers le bas.

Si vous le souhaitez, vous pouvez optimiser votre IA pour la rendre plus efficace.

Place à l'exercice noté !

- **Pareil pour que le thumb-up et thumb-down, sauf que cette fois-ci c'est avec le nombre de doigts levés (voir slide 7) .**

N'oubliez pas d'expliquer votre démarche, et de mettre les commandes systèmes (à exécuter dans le terminal : `pip3 install` par exemple). Commentez votre code et mettez des photos.

Vous pouvez (voire, vous êtes encouragés) à réutiliser le code donné dans les tutoriels pour vous les approprier. Cela vous aidera pour votre projet

Des questions ?