Document technique

API

- FastAPI :
  - pred() : sert à déterminer ce qui est pesé

    renvoi liste : [classe, libellé, prix total, codebarre, type_vente)

    type_vente : au poids ou à l'unité
  - update_item() : mise à jour d'un produit
  - get_image() : renvoi l'image correspondant au No demandé

Web :

- Flask
-

Interface WEB

- /static : css, img
- /static/images : images produits afficher dans l’appli de test
- /templates : html

Mix FastAPI / Flask

- La même application va prendre en charge les appels FastAPI et Flask
- Les appels API sont vues en préfixant le point d'entrée par « /fast » (prévoir un nom moins explicite comme v1 en prod)
- ref : https://fastapi.tiangolo.com/advanced/wsgi/

Methods Py (1/2)

- ml.load_models() : précharge les models locaux
- update_model() : vérifie si un nouveau model est dispo et le charge
- ml.get_predict() : lance une prediction sur une image, et retourne une liste de données
- db.get_product_info() : récupère depuis une bdd les infos d'une classe (un produit)
- db.init_bdd() : Initialise la bdd (csv or bdd)

Methods (2/2)

- db.update_item() : met à jour la bdd
- main.init_app() : at startup, init des différents éléments (présence model, init bdd, verif updates, xxx)
- Main.check_updates() :vérifie présence mise à jour model, bdd...
- db.updt_stats() : mise à jour stats d'utilisations
- save__bdd() : sauvegarde la bdd (si nécéssaire)

Docker

- Hébergement de l’application pour faciliter son déploiement
  - ??? Modalités ???