



IMT Lille Douai
École Mines-Télécom
IMT-Université de Lille

LAVÉRAT Benjamin

Projet mobile et vision

Documentation technique

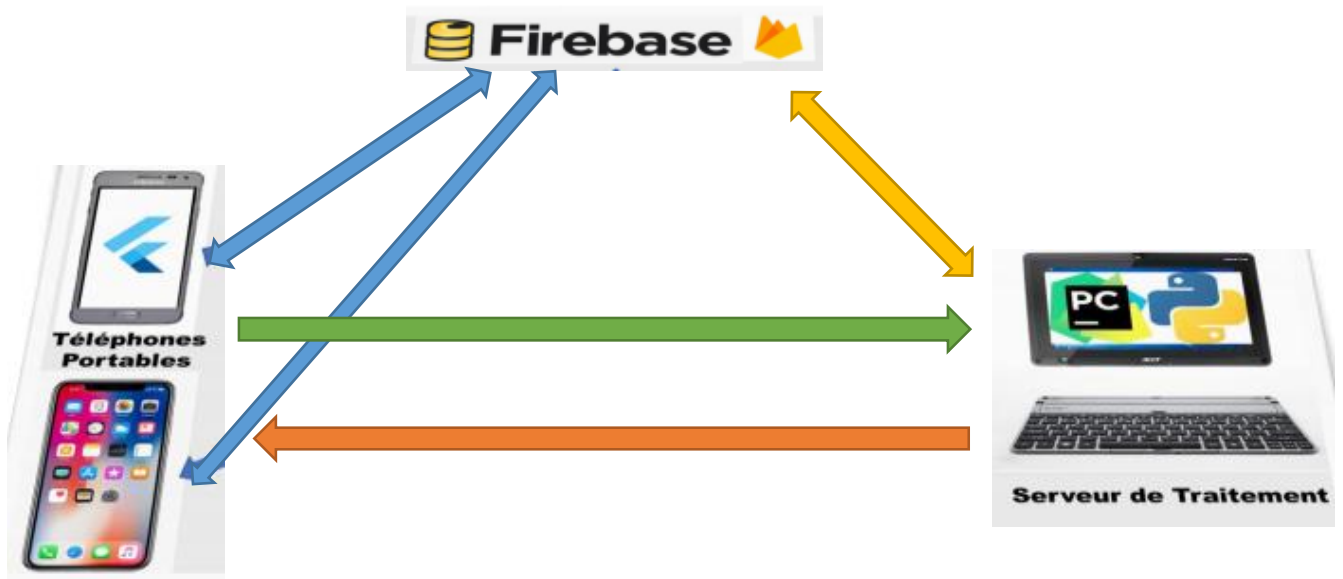
Application et Serveur Coin Counter







Table des matières

I.	Interopérabilité entre l'application mobile et le serveur	3
II.	Application mobile	4
A.	La structure de l'application	4
B.	Les packages indispensables au bon fonctionnement de l'application	4
C.	Description du fonctionnement de chacune des pages de l'application	5
D.	La base de données	7
E.	L'authentification	8
F.	Les requêtes web.....	8
G.	Les améliorations possibles.....	8
III.	Serveur web.....	9
A.	La structure du serveur	9
B.	Les packages indispensables au bon fonctionnement du serveur	9
C.	Description du fonctionnement du serveur	9
D.	Les améliorations possibles.....	11

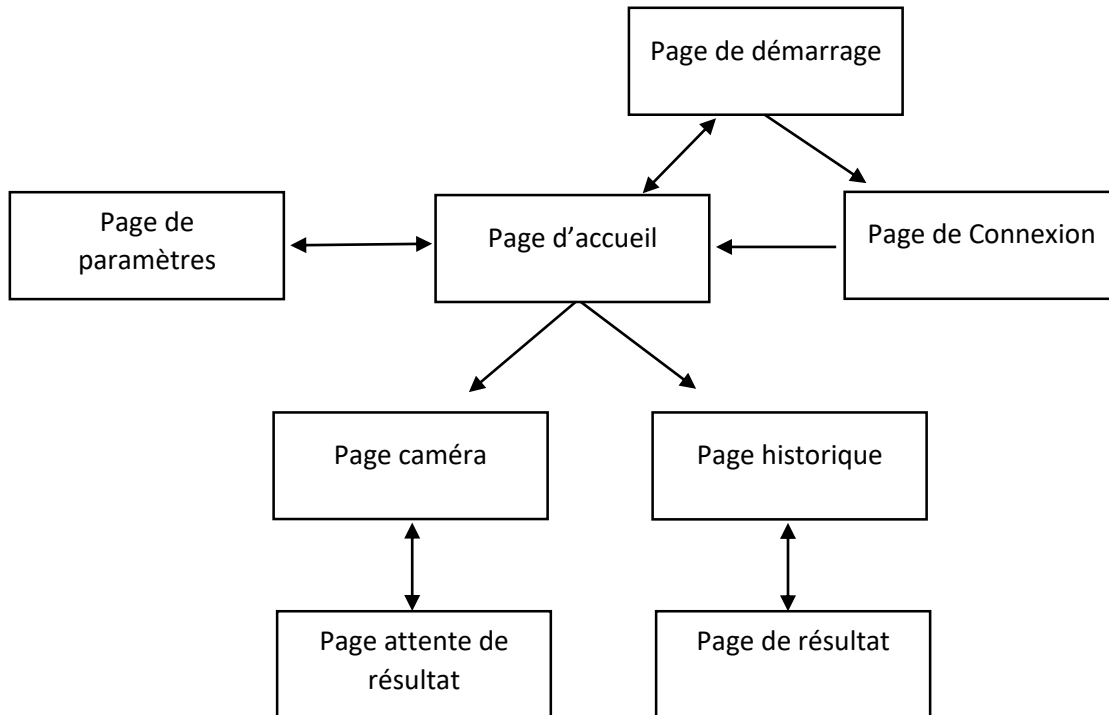
I. Interopérabilité entre l'application mobile et le serveur



-  Authentification de l'utilisateur
-  Requête web post, contenant l'image, et le token de l'utilisateur au format json.
-  Réponse à la requête web, renvoyant un fichier json contenant les informations identifiées dans l'image.
-  Vérification de l'authenticité du token reçu.

II. Application mobile

A. La structure de l'application



B. Les packages indispensables au bon fonctionnement de l'application

L'application mobile Coin Counter est développée via l'outil de développement flutter, en langage dart.

Afin que l'application mobile fonctionne les imports suivants sont nécessaires :

```
camera: ^0.7.0
path_provider: ^1.6.24
http: ^0.12.2
firebase_core: ^0.5.0
firebase_auth: ^0.18.3
cloud_firestore: ^0.14.1
sqflite: ^1.3.2
json_annotation: ^3.0.0
shared_preferences: ^0.5.12+4
flutter_slidable: ^0.5.7
flutter_rating_bar: ^3.2.0+1
```

Vous retrouvez ces imports dans le fichier « pubspec.yaml »

C. Description du fonctionnement de chacune des pages de l'application

Page de démarrage :

La page de démarrage est invisible pour l'utilisateur, elle permet de détecter si l'utilisateur a déjà utilisé l'application et ne s'est pas déconnecté, dans ce cas elle envoie l'utilisateur sur la page d'accueil de l'application.

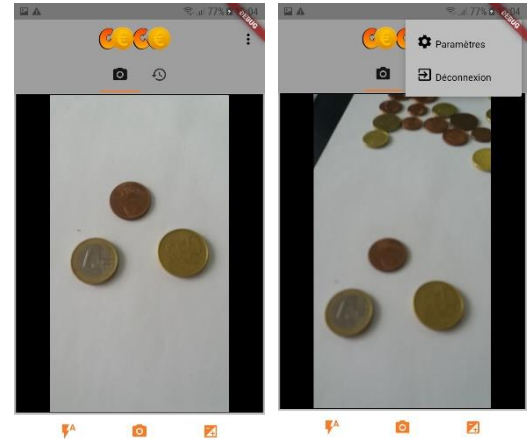
Dans le cas inverse la page démarrage envoie l'utilisateur vers la page de connexion afin que celui-ci s'identifie ou créer son login/mot de passe.

Page d'accueil :

La page d'accueil est la page principale de l'application, elle permet d'avoir accès à la page caméra, à la page d'historique, la page de paramètre mais aussi de se déconnecter de l'application.

Les pages caméra et historique sont joignables grâce au widget TabBar qui permet de passer d'une page à l'autre grâce à un mouvement de swipe de droit à gauche ou inversement.

La page paramètre et l'option de déconnexion sont atteignables grâce au popupMenu présent dans le coin supérieur droit de la page.



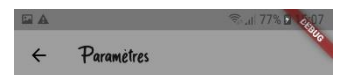
Page de Connexion :

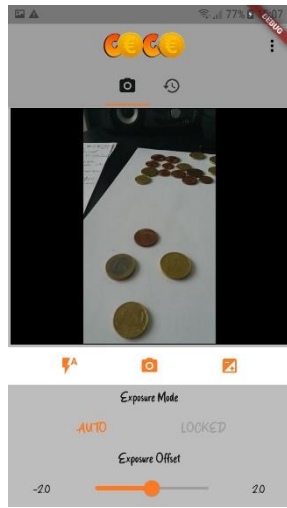
La page de connexion est une page très basique comprenant deux champs textes et deux boutons. Les champs textes permettent à l'utilisateur de rentrer ses identifiants ainsi que son mot de passe. Le premier bouton permet de valider la connexion tant dit que le deuxième bouton permet à l'utilisateur de créer son compte pour l'application.

Page de paramètres :

La page de paramètres permet à l'utilisateur 3 actions :

- Changer l'adresse du serveur
- Tester la connexion
- Supprimer tous les enregistrements de l'utilisateur de la base de données.





Page caméra :

La page caméra permet de prendre en photo les pièces dont l'on veut reconnaître et compte la somme totale.

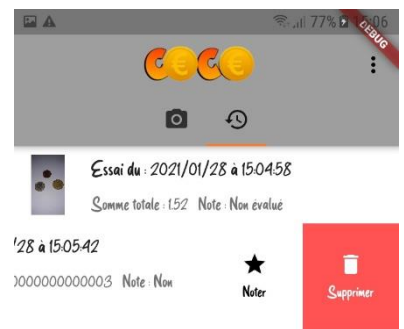
La photo est prise lorsque l'utilisateur appuie sur le bouton central. Les boutons annexes permettent de régler le flash ou les paramètres d'expositions de l'appareil photo.

Page historique :

La page historique permet à l'utilisateur de visualiser tous les essais qu'il a effectués depuis l'application.

Il a également la possibilité de swiper vers la gauche sur l'essai afin de faire apparaître deux options pour la tentative, soit de la noter, soit de la supprimer.

Si l'utilisateur clique sur un essai il sera envoyé vers la page de résultat de l'essai avec tous les détails.



Page attente de résultat :

La page d'attente de résultat est la page qui apparaît dès lors que l'utilisateur a pris la photo.

C'est cette page qui envoie la requête vers le serveur avec l'image prise par l'utilisateur.

La page affiche un indicateur circulaire tournant le temps que le serveur traite la requête et répond à l'application.

Une fois le résultat obtenu la page affiche l'image prise par l'utilisateur et l'identification effectuée par le serveur sur l'image.

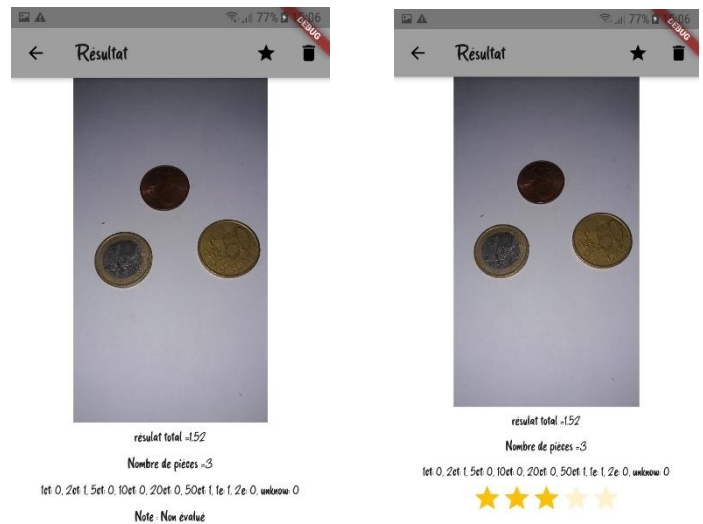
Cette également cette page qui enregistre l'essai dans la base de données présente sur le téléphone.

Page de résultat :

La page de résultat est la page que l'utilisateur obtient après avoir cliqué sur un essai de la page historique.

Elle affiche tous les détails obtenus lors de l'identification par le serveur ainsi que la notation réalisée ou non par l'utilisateur.

Celui-ci à la possibilité une nouvelle fois de noter ou supprimer l'essai à l'aide des boutons dans le coin supérieur droit.



D. La base de données

J'ai fait le choix de positionner la base de données sur le téléphone de l'utilisateur ainsi il est le seul propriétaire de ses données puisque seul l'image et le token sont envoyés vers le serveur.

La base de données est faite à l'aide du package SQLite, elle comporte uniquement une seule table, composer des champs suivants :

TABLE historique	
identative	INTEGER PRIMARY KEY
userID	TEXT
Image	TEXT
pieces	TEXT
Date	TEXT
Note	REAL

La base de données est gérée grâce au package gestion_bdd que j'ai créé, ce package permet d'avoir accès aux méthodes :

- D'ouverture de la base de données
- De création de la base de données
- D'ajout de nouveaux éléments dans la base
- La lecture des éléments présents
- La suppression d'un ou de plusieurs éléments

E. L'authentification

L'authentification est administrée par le service firebase de Google. Afin d'avoir toutes les méthodes nécessaires au bon fonctionnement de la page de connexion, le projet comporte un package authentification dans lequel est défini :

- La méthode de connexion
- La méthode d'inscription de l'utilisateur
- La méthode de déconnexion
- La méthode d'obtention de l'uid

F. Les requêtes web

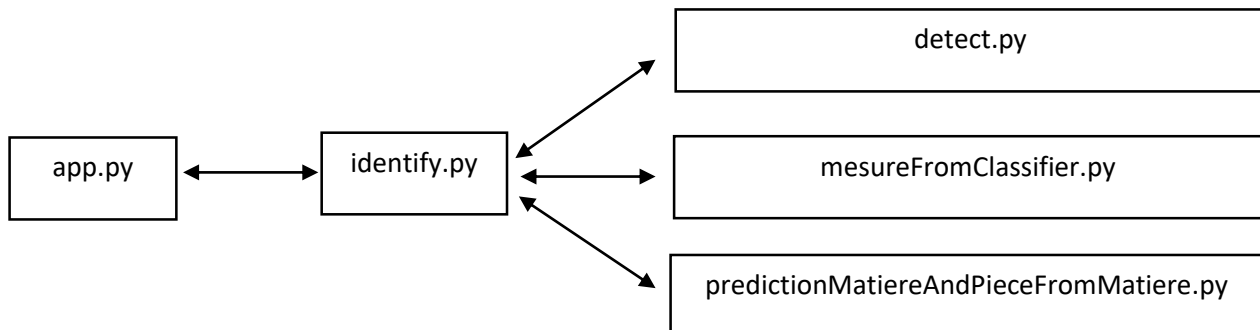
Les requêtes web envoyées vers le serveur de l'application sont gérées à l'aide du package `http_helper`. Ce package permet à l'application d'avoir accès à la modification de l'adresse du serveur, de créer une requête afin de connaître le statut du serveur et si celui-ci est joignable par l'application. Enfin il permet également de créer la requête pour identifier le contenu de l'image.

G. Les améliorations possibles

L'application tourne plutôt bien à l'heure actuelle, cependant il reste des améliorations possibles à mettre en place. Tout d'abord des optimisations du code sont certainement possibles étant donné que c'est le premier projet que je réalise sur flutter. Ensuite des petits problèmes de rafraîchissement des pages lors de la modification des données dans la base de données existent, de plus la prise en compte des timeouts lors des envois de requêtes vers le serveur permettrait de ne pas laisser les pages tournées indéfiniment. Enfin l'ajout de nouvelles méthodes d'authentification dans l'application sont envisageables.

III. Serveur web

A. La structure du serveur



B. Les packages indispensables au bon fonctionnement du serveur

Le serveur Coin Counter est développé via l'outil de développement Pycharm, en langage python.

Afin que l'application mobile fonctionne les dépendances suivantes sont nécessaires :

- Opencv-contrib-python
- Flask
- Tensorflow
- Firebase-admin
- Keras
- Imutils
- Sklearn

C. Description du fonctionnement du serveur

app.py :

C'est le script principal du serveur qui est lancé via le service flask, ce script comporte les différentes routes que comporte le serveur.

La route statut qui renvoie uniquement ok à toutes les requêtes qu'il reçoit.

Ainsi que la route principale `todetect`, qui permet d'identifier le contenu de l'image.

Le fonctionnement de cette route est le suivant : quand le serveur reçoit la requête, il vérifie que la requête est de type post, ensuite il décrypte le contenu de la requête et enregistre l'image et en local et décrypte le token reçu.

Si le token est valide alors le serveur lance le processus d'identification avec l'appel de la fonction identify définie dans la page homonyme. Autrement le serveur renvoie un message d'erreur à l'application.

identify.py :

C'est le fichier python central dans l'identification des formes détectées. Il reçoit en paramètre l'image à traiter, puis il coordonne tous les actions à réaliser sur celle-ci :

- Appelle à la fonction detect afin d'obtenir toutes les formes circulaires de l'image.
- Première classification des formes grâce au réseau de neurones entraîné
- Appelle à la fonction mesureFromClassifier afin de mesurer toutes les formes depuis les échelles trouvées grâce à la première classification.
- Appelle de la fonction predictionMatiereAndPieceFromMatiere afin d'obtenir une autre classification des formes mais cette fois-ci par rapport à l'histogramme moyen de la forme. A partir de la classification par histogramme le script réalise à nouveau une mesure des formes avec l'échelle de la plus grosse pièce détectée.

Une fois toutes les actions réalisées la fonction pondère tous les résultats obtenus pour ne renvoyer que ceux qui ont eu la plus grande occurrence lors des identifications.

detect.py :

Va permettre de remonter toutes les formes circulaires présentes dans l'image, pour cela on applique plusieurs filtres à l'image afin de ne ressortir que les contours des objets présents dans l'image et ensuite on ne récupère que les objets ayant une forme elliptique, je ne me suis pas restreint au cercle car avec la déformation des formes dû à l'angle de caméra les résultats n'étaient pas bons.

Une fois toutes les ellipses trouvées dans l'image, le script les redessine toutes afin d'éliminer les éventuelles ellipses dans une ellipse (tel que le tour intérieur d'une pièce de 1 ou 2 euros). L'élimination faite, le script renvoie une liste avec les coordonnées de 4 points formant un carré qui contient l'ellipse, ainsi que tous les points composant cette ellipse et les propriétés de celle-ci.

mesureFromClassifier.py :

La fonction permet à partir des pièces déjà identifiées de mesurer toutes les formes présentes afin d'identifier les autres pièces présentes sur l'image par rapport à leur taille.

La fonction effectue autant de fois les mesures sur toutes les pièces que le nombre de pièces qui ont pu être identifiées par le classifieur.

La fonction renvoie donc un tableau avec toutes les mesures des formes selon le référentiel identifié.

predictionMatiereAndPieceFromMatiere.py :

Ce script utilise un autre modèle de classification basé sur sklearn et va composer un modèle de prédiction de la matière des formes détectées pour identifier leur « famille » (cuivre, doré, 1euro, 2euro). Une fois ce modèle composé et les formes de l'image analysées, on se sert de la classification par famille pour effectuer une mesure des pièces pour tenter de les identifier par rapport à leur famille et à leur taille. Une fois cette opération réalisée on renvoie un tableau avec la famille et la pièce identifiées pour chaque forme.

D. Les améliorations possibles

Le serveur tourne correctement actuellement mais je pense que beaucoup d'optimisations peuvent être apportées, mais pour mon premier projet réalisé en python je suis assez fière de ce que j'ai accompli. Des améliorations peuvent être apportées en fournissant des datasets plus fournis en images et de meilleur qualité mes image provienne d'un téléphone Android d'entrée de gamme d'il y a 6 ans donc la qualité photo n'est pas la meilleur. Mes options d'entraînement de mon modèle peuvent être surement perfectionnées même si j'ai déjà mené plusieurs tests afin d'obtenir quelque chose de convenable. Je pense également qu'il faudrait créer une détection spéciale pour la face des pièces, actuellement seul le côté pile est traité. De plus lors de la détection afin d'être indépendant du fond de l'image il faudrait mettre en place une couleur de remplacement du fond ou un masque sur l'image.