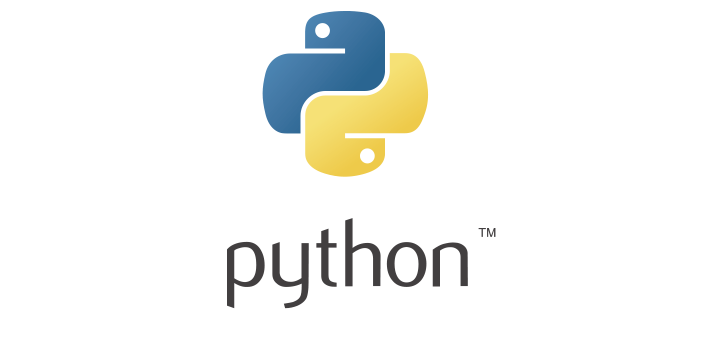
Crampon Louise FA20

Decanter Maxence

**Projet Informatique**

**Création d’une application de recherche d’image**





# Introduction

Ce rapport vient conclure et résumer le travail effectué sur le projet de création d’un moteur de recherche d’images via Android.

Pour ce faire, nous avons utilisé plusieurs technologies liées entre elles comme Android Studio pour la partie applicative et Postman ainsi que Pycharm (et donc Python !) pour la partie Serveur.

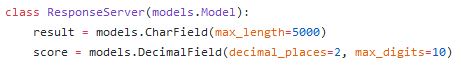
Le projet a occupé une grande partie de notre année puisqu’il nous a été premièrement présenté en Septembre, bien que le sujet n’était pas tout à fait certain et a ensuite été programmé à la suite des cours de mise à niveau que nous avons eus sur les mois suivants. Nous avons réellement commencé la programmation en début d’année 2019 pour un rendu au mois d’avril.

# Partie Serveur

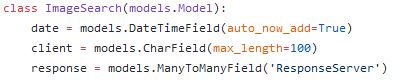
Le fonctionnement du serveur Django Rest est le suivant :

* Tout d’abord, l’application Android lui envoie une requête POST avec dans le corps de la requête, l’image à analyser en base64.
* Ensuite le serveur va décoder cette base64 pour recréer l’image dans un fichier temporaire.
* Une fois cette image enregistrée, le serveur lance l’analyse de l’image. Suite à des soucis pour faire fonctionner TensorFlow, nous avons choisi d’utiliser la technique des Bags of Words.
* Le serveur lance donc l’indexing puis lance le traitement. Ce dernier retourne un JSON contenant pour les 3 images ayant le score le plus élevé, l’URL de l’image et son score. Nous avons défini dans model.py, deux modèles pour enregistrer le résultat du traitement.

Le premier modèle « ResponseServer » enregistre une URL couplée à un score.

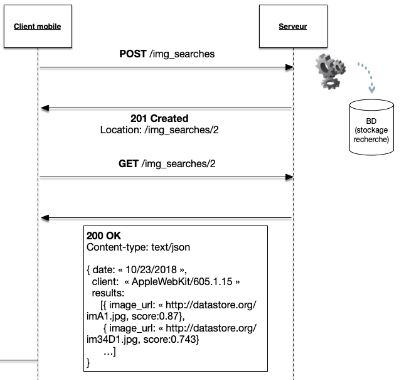


Le second model enregistre la date de l’analyse et le client qui a réalisé l’analyse ainsi qu’un JSON de « ResponseServer » des 3 images correspondantes.



Une fois ces étapes réalisées, le serveur renvoie à l’application l’id de l’objet ImageSearch sauvegardé dans la base de données correspondant à l’analyse réalisée.

Une fois que l’application reçoit un statut 201\_CREATED avec l’id, l’application Android fait un GET avec cet id sur le serveur qui va lui renvoyer pour terminer un JSON de l’objet ImageSearch. Les échanges entre le serveur et l’application sont décrits dans l’image ci-dessous.



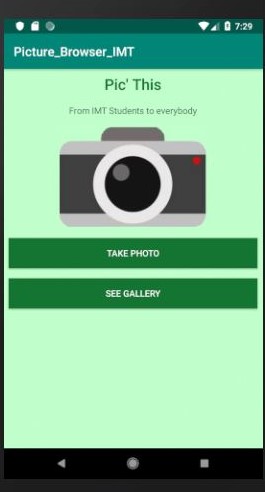
# Partie Application

La partie applicative est générée grâce au logiciel de développement Android Studio. Il ne peut être testé que sur des appareils possédant le système d’exploitation Android. Ainsi, pour permettre à tout le monde de l’utiliser, le logiciel propose différents émulateurs, en plus de la possibilité de tester son travail sur son propre téléphone.

A la création de notre application, nous avons également la possibilité de choisir un APK de référence dont la version peut être inférieure à la version « actuelle », ce qui permet de tester notre travail même si l’appareil de test est ancien.

Voici un aperçu de l’application telle qu’elle est livrée :

* La première page correspond à la page d’accueil de l’application. On y retrouve le nom de l’application, un crédit sur les créateurs, un logo ainsi que eux boutons : le premier permet d’accéder à un appareil photo pour créer une nouvelle image à rechercher dans notre moteur. La seconde donne accès à la galerie de photos à partir de laquelle on peut choisir une image à traiter.
* Quel que soit le premier choix que nous ayons effectué, nous arrivons ensuite sur la deuxième page à partir de laquelle on récupère notre photo et lance l’analyse qui nous donnera par la suite le nom de l’entreprise à laquelle appartient notre logo ainsi que les trois images de notre base de données qui ressemblent lui ressemblent le plus.



En termes de traitement, l’application envoie une requête au serveur dans laquelle elle lui donne l’image à analyser convertie en base 64.

Le serveur réalise son algorithme puis renvoie un fichier JSON à l’application, qu’il faut ensuite parser pour récupérer le tableau « response » et trier les trois résultats « results : ; score : » dans lesquels se trouvent les URL vers les images les plus proches de notre photo suivies du score correspondant à la probabilité de ressemblance.

Une fois ces résultats récupérés et séparés, l’application les renvoie dans des ImageView pour les afficher sous la photo analysée dans l’application.

# Conclusion

Pour conclure, ce projet était intéressant car il fallait que chaque membre du binôme communique avec l’autre pour lui donner son avancement et tester nos parties respectives via les interfaces de l’autre.

Il n’était au départ par évident de déterminer exactement quel allait être le travail de chacun, où allaient se dérouler les différents traitements, …

Concernant les problèmes rencontrés, le fait de tester l’application en wifi avec le serveur d’un côté et l’application de l’autre nous a causé quelques problèmes de latence. De plus les adresses de récupération du serveur changeaient régulièrement lorsque l’on changeait de point de connexion, il fallait donc toujours s’adapter et penser à se mettre à jour par rapport au deuxième membre du binôme. Au niveau de la partie Android, la connexion à l’appareil photo a demandé plusieurs essais plus ou moins fructueux, tout comme les tests des méthodes GET et POST.

Pour terminer, nous pourrions améliorer notre projet en modifiant le code Android pour essayer de le faire réagir plus rapidement avec le serveur, lors du traitement et de l’affichage des images « correspondantes » car pour l’instant ce dernier est trop long et les images n’apparaissent pas ailleurs que sur le navigateur (sur lequel elles mettent un certain temps à apparaître). Nous pourrions également penser à prendre en compte un feedback pour les images apparues à l’écran et qui n’appartiennent pas à la marque recherchée.