# Application Mobile de Recherche d'Images

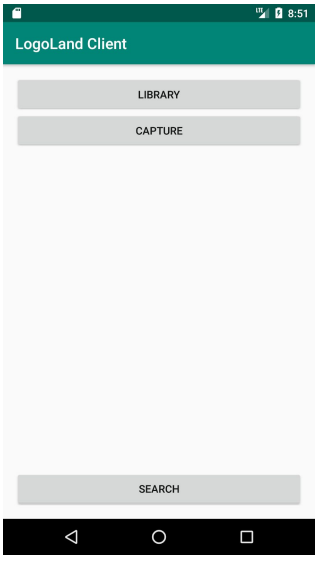
Rapport

Projet : **LogoLand**

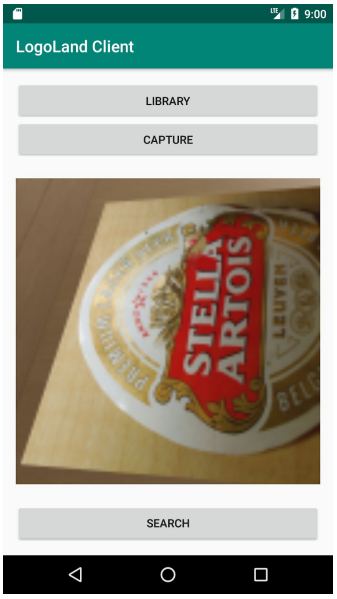
Participants :
**William Jedrzejak**
**Louis Lalleau**

# I. Interface utilisateur

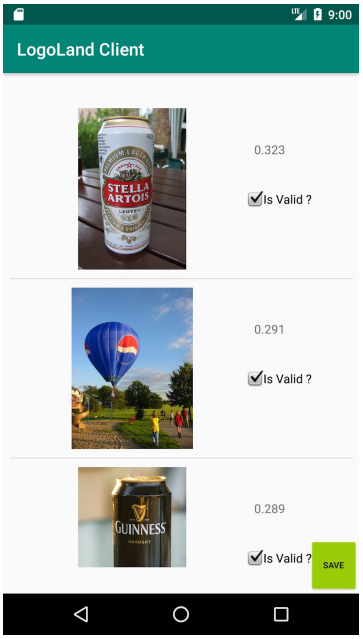
## A. La recherche d'images




Lors du lancement de l'application, l'utilisateur a le choix de sélectionner une photo présente dans la galerie de l'appareil ou de prendre une photo.

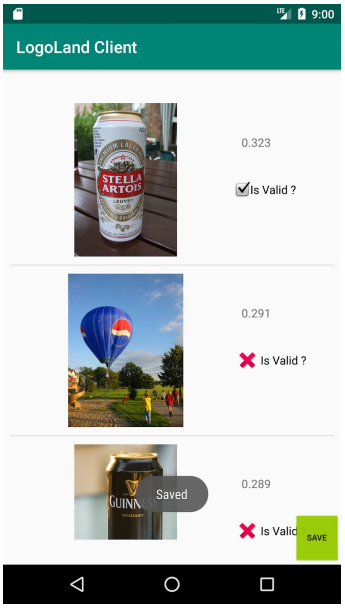



La photo sélectionnée s'affiche, l'utilisateur peut ensuite lancer la recherche en cliquant sur le bouton "Search".




Une liste d'images similaires suivi de leur score de similitude est alors affichée.

## B. Le feed-back




L'utilisateur a la possibilité de signaller si les images ne correspondent pas à sa recherche, dans l'optique d'améliorer le service.

# II. Architecture logicielle

## A. L'application mobile

La première partie de l'application est un client android. Il permet à l'utilisateur d'accéder au service de recherche d'images depuis un appareil Android disposant d'une connection à internet.
Le client est développé en java en utilisant l'API Android 23, ce qui permet une compatibilité de l'application avec les appareils utilisant Android 6 et ou plus.
Le code est séparé en différents fichiers :

- Les fichiers Java :
    - MainActivity.java : Contient les méthodes permetant de récuperer une image depuis la mémoire ou l'appareil photo, d'envoyer l'image au serveur en utilisant la librairie Volley et d'appeler l'activité Result.
    - ResultActivity.java : Contient les méthodes permetant de récuperer la liste des résultats avec la librairie Volley, et d'envoyer le feedback.
    - Logo.java : Une classe définissant un logo de resultat retourné par le serveur. contient une url pour récupérer l'image, un score et un booléen pour le feedback.
    - ImageUtils.java : Classe utilitaire permetant de convertir un objet Bitmap en chaîne base64 et vice versa, ainsi que récupérer une image à partir d'une URL.
- Les fichiers XML de layout :
    - activity_main.xml : Définit l'activité principale, celle qui contient les boutons de choix entre importer une photo de la librairie ou prendre une photo en utilisant l'appareil photo.
    - activity_result.xml : Définit l'activité affichant les résultats de la recherche sous forme de liste ainsi que le bouton pour sauvegarder le feedback.
    - result_layout.xml : Définit chaque item de la liste de résultats : l'image, le score et le checkbox de feedback.
- Le fichier de composant personnalisé :
    - cross_check_box.xml contient la définition du checkbox personnalisé pour le feedback

## B. Le serveur

La deuxième partie de l'application repose sur un serveur proposant une API REST. Pour construire cette API, nous utilisons les frameworks Django et Django REST.

Le serveur été développé selon le principe de modularité. Chaque composant est un module indépendant pouvant, à la convenance des développeurs, être réutilisé pour d'autres projets. Les modules sont :

- **image_retrieval** regroupe toute la logique relative à l'analyse d'image.
- **imgsearch** définit les deux modèles utilisés par l'application
- **imgsearch_rest** expose et gère les différentes vues.

- **utils** propose une unique fonction de traitement d'images

De cette manière le code de l'analyse d'image ou les modèles peuvent être directement implémenté dans un autre projet.

En plus de ces modules, nous retrouvons deux autres dossiers :
- **dataset** contient, comme son nom l'indique subtilement, le dataset
- **logoland_server** décrit l'application pour le framework Django

# III. Analyse d'images

L'application LogoLand permet de retrouver les images d'une dataset ressemblant le plus à une image donnée. En d'autres termes, l'application utilise un système d'image retrieval.

Pour ce faire, deux choix de l'algorithme s'offraient à nous :
- Le sac de mots
- Le réseaux de neurones convolutionnel (CNN)

Notre choix s'est fait par rapport aux librairies nécessaires pour faire tourner ces algorithmes sur nos ordinateurs limités par la politique de sécurité de notre entreprise (Nous faisons notre apprentissage dans la même entreprise).

Après plusieurs jours à essayer de faire tourner TensorFlow sur nos ordinateurs d'entreprise, nous avons écarté cette idée et avons choisi d'utiliser Bag-of-words. Bag-of-words est communément implémenté avec la librairie OpenCV. Cette librairie est compatible avec notre environnement de travail professionnel.

Le système image retrieval se compose de deux phases :
- L'indexation : à partir du dataset, un dictionnaire est produit
- La recherche : à partir des mots extraits de l'image fournie, les images du dataset partageant des mots sont sélectionnées.

## A. Indexation

L'indexation suit les étapes suivantes :
- Toutes les images du dataset sont lues
- Pour toutes les images, les descripteurs vont être extraits
- Ces descripteurs vont être regroupés selon l'algorithme de k-means
- Les histogrammes de ces points d'intérêts sont calculés
- Ces calculs sont normalisés puis stockés dans un dictionnaire

Ce processus doit être réalisé à chaque fois que le dataset évolue

## B. Recherche

La recherche suit les étapes suivantes :

- On charge le dictionnaire
- On détecte les descripteurs de l'image fournie
- L'histogramme est calculé
- Cet histogramme est comparé à chaque histogramme du dictionnaire
- Les meilleurs scores sont gardés

### C. Dataset

Le dataset utilisé dans le projet est un subset du dataset créé et maintenu par Multimedia Computing and Computer Vision Lab de l'université d'Augsburg. Le dataset propose plus de 3G de photos de 32 marques internationales. Les images sont directement prises sur le site Flickr. Le dataset est conçu pour l'image retrieval.

- page du projet
- dataset

## III. Bilan

Ce projet fût très instructif car il permettait de mettre en application deux notions vues en cours : le développement mobile et l'analyse d'images. Obtenir une application fonctionnelle est très gratifiant et nous avons aussi beaucoup appris sur les fonctionnalités avancés de git telles que les différentes branches et les pull request. Aussi la découverte de Django afin de créer une API REST en Python nous apporte des bases dans ce domaine.