# **Documentation Technique du Système de Recherche d'Images Similaires**

## **1. Architecture du Système**

Le système est divisé en deux phases principales : la phase de prétraitement hors ligne (offline) et l'application Web interactive.

### **A. Phase Hors Ligne (Offline)**

Cette phase permet de construire une base de données vectorielle à partir des images sources.

* **Chargement des images** : les images sont récupérées depuis le répertoire Datasets/Datasets.
* **Prétraitement & extraction de caractéristiques** : chaque image est redimensionnée selon les spécifications des modèles (VGG16, ResNet50, InceptionV3), puis traitée pour générer un vecteur de caractéristiques.
* **Clustering avec K-Means** : les vecteurs sont regroupés en clusters à l'aide de l'algorithme K-Means. Le nombre de clusters est défini manuellement ou déterminé à l'aide de la méthode du coude (Elbow method).
* **Sauvegarde des résultats** : les vecteurs, labels de clusters et chemins des images sont enregistrés dans des fichiers .npy dans le dossier data/.

### **B. Application Web (Streamlit)**

Cette phase constitue l’interface utilisateur permettant d’interagir avec le système.

* **Téléversement d'image requête** : l'utilisateur soumet une image à analyser.
* **Choix du modèle et de la métrique** : l'utilisateur sélectionne un modèle de descripteur (VGG16, ResNet50, InceptionV3) et une métrique de distance (Euclidienne, Manhattan, Chebyshev, Canberra).
* **Extraction des caractéristiques de l’image requête** : un vecteur de caractéristiques est extrait pour l’image soumise.
* **Calcul de similarité** : les distances entre le vecteur de l’image requête et ceux de la base sont calculées pour identifier les images les plus proches.
* **Affichage des résultats** : les images similaires sont présentées avec leur label et un histogramme montre la distribution des labels.

## **2. Choix Technologiques**

* **Langage de programmation** : Python 3.10+
* **Framework Web** : Streamlit
* **Modèles de réseaux de neurones** : VGG16, ResNet50, InceptionV3 via tensorflow.keras.applications
* **Clustering** : KMeans (scikit-learn)
* **Traitement d’images** : Pillow, NumPy
* **Visualisation** : Matplotlib

## **3. Instructions d’Installation**

### **Prérequis**

* Python ≥ 3.10
* Git (facultatif)

### **Installation**

python3 -m venv venv

source venv/bin/activate

pip install -r requirements.txt

### **Fichier requirements.txt**

streamlit

numpy

scipy

matplotlib

scikit-learn

tensorflow

Pillow

tqdm

## **4. Guide d’Utilisation**

### **Phase Offline**

Lancer le script de prétraitement pour générer la base de données vectorielle :

python offline.py

Ce script génère les fichiers suivants dans data/ :

* VGG16\_vectors.npy
* VGG16\_labels.npy
* VGG16\_paths.npy
* (similairement pour ResNet50 et InceptionV3)

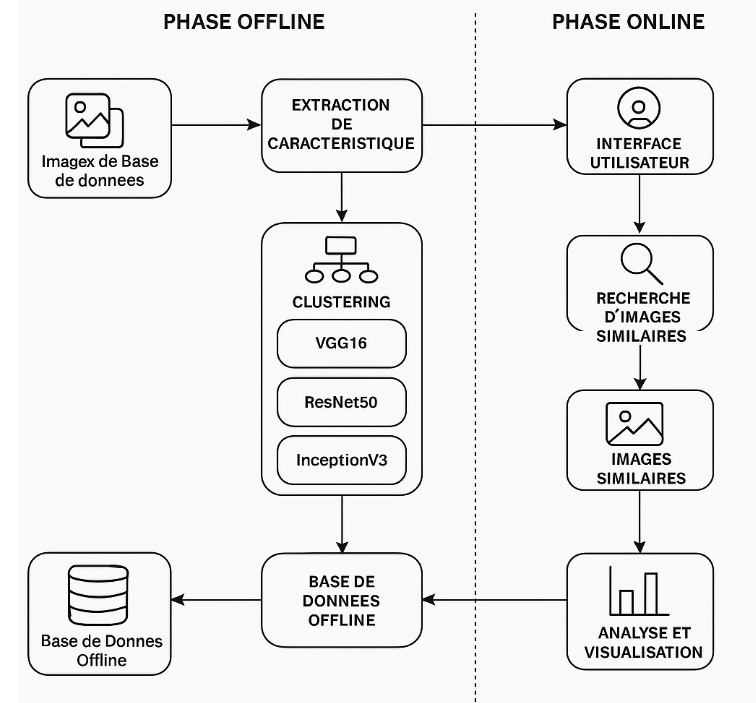
### **Phase Online**

Lancer l'application web avec Streamlit :

streamlit run app.py

L’interface s’ouvrira automatiquement dans le navigateur à l’adresse http://localhost:8501.

## **5. Schéma de l’Architecture Fonctionnelle**



## 

## 

## 

## **6. Organisation du Répertoire**

projet/

├── app.py # Interface Streamlit

├── offline.py # Script d’extraction et de clustering

├── data/ # Données vectorielles et labels

│ ├── VGG16\_vectors.npy

│ ├── VGG16\_labels.npy

│ ├── VGG16\_paths.npy

│ └── ... # Équivalents pour ResNet50 et InceptionV3

├── Datasets/ # Répertoire contenant les images originales

│ └── Datasets/

│ ├── image1.jpg

│ └── ...

├── venv/ # Environnement virtuel

├── requirements.txt # Dépendances Python

└── README.md # Guide utilisateur et installation