

## Multimedia Mining and Indexing

MST IASD/S3 2024-2025

# Développement d'une application Web pour l'indexation d'images basée sur le contenu

Encadrer par : M'hamed AIT KBIR

Réaliser Par :

- Liefriid Chihab Eddine
- Idriss Khattabi
- Boufarhi Ayman

# Introduction :

La recherche d'images par le contenu (CBIR - *Content-Based Image Retrieval*) est une technologie qui permet de rechercher des images dans une base de données en analysant leurs caractéristiques visuelles, telles que la couleur, la texture et la forme, plutôt que par des métadonnées textuelles. Avec l'explosion des contenus visuels sur le web et les bases de données, il devient essentiel de disposer de systèmes performants pour organiser, indexer et rechercher efficacement ces images.

Dans ce projet, nous visons à développer une application web permettant de mettre en œuvre un système d'indexation et de recherche d'images par le contenu. L'application intégrera deux méthodes de recherche : une recherche simple basée sur des descripteurs visuels tels que les histogrammes de couleurs, les couleurs dominantes, les descripteurs de texture, les moments de Hu, la couleur moyenne et les histogrammes des contours ; ainsi qu'une méthode avancée avec retour de pertinence, en particulier la méthode de déplacement du point de requête (*Query-Point Movement*), permettant d'affiner les résultats des recherches en fonction des préférences de l'utilisateur.

L'application sera conçue à l'aide de MEAN pour la partie frontend, et Flask pour la partie backend. Elle permettra de gérer les images (téléchargement, organisation, transformation) et de rechercher des images similaires dans une base prédéfinie, notamment la base RSSCN7, tout en fournissant des services RESTful pour les calculs des descripteurs. Ce projet constitue une approche pratique et innovante pour explorer les capacités des systèmes CBIR, en intégrant des fonctionnalités avancées et une expérience utilisateur enrichie.

## Revue de la littérature

Dans le cadre de ce projet, il est essentiel de comprendre les principes fondamentaux des systèmes de recherche d'images par le contenu (CBIR) et leurs fonctionnalités pour offrir une expérience utilisateur optimisée. Les systèmes CBIR permettent d'effectuer des recherches d'images basées sur les caractéristiques visuelles des fichiers, et ce projet intègre plusieurs approches pour maximiser leur efficacité et leur convivialité.

### Fonctionnalités générales de l'application

L'application développée offre deux principales méthodes de recherche d'images pour répondre aux besoins variés des utilisateurs :

#### 1. Recherche simple

Cette fonctionnalité compare une image requête à celles de la base en utilisant des descripteurs visuels, notamment :

- o **Histogrammes de couleurs** : analysent la répartition des couleurs.
- o **Couleurs dominantes** : identifient les teintes principales.
- o **Descripteurs de texture** : détectent les motifs à l'aide de filtres comme ceux de Gabor.
- o **Moments de Hu** : extraient des informations sur la forme et les contours.
- o **Couleur moyenne** : calcule la couleur moyenne de l'image.
- o **Histogrammes des contours** : analysent la distribution des contours présents dans l'image.

## 2. Recherche avancée avec retour de pertinence (Relevance Feedback)

Cette fonctionnalité améliore les résultats en tenant compte des préférences exprimées par l'utilisateur, grâce à une méthode interactive où les images jugées pertinentes influencent les futures recherches.

### Présentation du dataset RSSCN7

L'application s'appuie sur le dataset RSSCN7, composé de 2800 images réparties en sept catégories (Résidentiel, Forêt, Industrie, etc.). Cette base de données, riche et diversifiée, permet de tester et valider les performances du système tout en offrant des résultats adaptés à divers scénarios d'utilisation.

### Tâches réalisées pour le développement de l'application

L'application Web a été développée en utilisant le stack MEAN (MongoDB, Express.js, Angular, Node.js) et intègre plusieurs fonctionnalités interactives :

- **Gestion des images** : possibilité de charger (upload), télécharger (download), et supprimer des images ou un ensemble d'images.
- **Organisation des images** : classement des images selon les catégories définies dans la base d'exemples.
- **Création de nouvelles images** : application de transformations (crop, resize, flip, rotate) sur des images existantes.
- **Consommation de services RESTful** : intégration de services développés avec Flask et Flask-RESTful pour calculer les descripteurs visuels d'une image ou d'un ensemble d'images.
- **Visualisation des descripteurs** : affichage des descripteurs calculés dans un format clair et structuré.
- **Recherche d'images similaires** : sélection d'une image requête pour identifier les images les plus proches, à l'aide des deux méthodes proposées : recherche simple et recherche avec retour de pertinence.

Ces fonctionnalités permettent de proposer une application robuste, interactive et alignée sur les besoins des utilisateurs, tout en exploitant des approches avancées pour le traitement et la recherche d'images.

## Descripteurs d'Image

Les descripteurs d'image permettent de représenter les caractéristiques visuelles des images sous forme numérique. Voici les descripteurs utilisés et les raisons de leur choix :

### 1. Histogramme de couleurs

Représente la distribution des couleurs dans une image. Ce descripteur est choisi pour sa capacité à capturer les informations globales de couleur, utiles pour différencier des images ayant des palettes distinctes.

### 2. Couleurs dominantes

Identifie les couleurs prédominantes dans une image. Ce descripteur simplifie la comparaison en réduisant la complexité des informations sur les couleurs.

### 3. Descripteurs de texture (Filtres de Gabor)

Capturent les variations de texture dans l'image, permettant de différencier des surfaces lisses

et rugueuses. Ces descripteurs sont utiles pour des recherches dans des images avec des détails fins ou des motifs.

#### 4. **Moments de Hu**

Calculés à partir des contours d'une image, ces moments sont invariants aux transformations géométriques (translation, rotation, échelle). Ils aident à identifier des formes globales.

#### 5. **Couleur moyenne**

Fournit une représentation simplifiée des couleurs en calculant leur moyenne. Ce descripteur est efficace pour une recherche rapide et approximative.

#### 6. **Histogramme des contours (Edge Histogram)**

Décrit la fréquence et la direction des contours dans une image. Ce descripteur est pertinent pour capturer les structures principales et les détails d'une image.

## Algorithme de Recherche Simple

Cet algorithme calcule un score global en combinant les distances des descripteurs.

### Fonctionnement

L'algorithme procède comme suit :

- Pour chaque image de la base, il calcule trois distances principales :
  - o Une distance pour les contours, basée sur les moments de Hu et l'histogramme des contours.
  - o Une distance pour les couleurs, utilisant l'histogramme des couleurs, la couleur moyenne et les couleurs dominantes.
  - o Une distance pour les textures.
- Chaque groupe de distances est pondéré selon des poids définis, permettant de privilégier certaines caractéristiques visuelles.
- Le score total pour une image est la somme pondérée des distances des trois groupes.
- Les images sont triées en fonction de leur score, et les plus proches de l'image requête sont renvoyées.

## Algorithme de Retour de Pertinence : Méthode Query-Point Movement

Cet algorithme ajuste dynamiquement les poids des groupes de descripteurs en fonction des retours d'information de l'utilisateur.

### Fonctionnement

- Les poids des groupes et des descripteurs individuels sont mis à jour en utilisant trois éléments :
  - o Une augmentation proportionnelle aux scores des images jugées pertinentes.
  - o Une réduction proportionnelle aux scores des images jugées non pertinentes.

- o Un facteur de normalisation pour maintenir une somme cohérente des poids.
- La mise à jour des poids suit les formules :

$$Q' = \alpha Q + \beta \left( \frac{1}{N_{R'}} \sum_{i \in D_R} D_i \right) - \gamma \left( \frac{1}{N_{N'}} \sum_{i \in D_N} D_i \right)$$

- o Les poids des descripteurs individuels sont recalculés de manière similaire pour chaque groupe.
- Cette méthode permet d'affiner la recherche en fonction des préférences utilisateur.

## Architecture de l'Application

L'application repose sur une architecture hybride utilisant les technologies MEAN (MongoDB, Express.js, Angular, Node.js) et Flask.

### 1. Back-end Flask

- o Gère le traitement des images.
- o Fournit une API RESTful pour calculer les descripteurs de contenu.
- o Permet des transformations d'image telles que le recadrage, le redimensionnement, la rotation et le retournement.
- o Permet de recherche d'une image par la recherche simple or l'algorithme de Retour de Pertinence (Méthode Query-Point Movement)

### 2. API Express.js

- o Sert d'interface entre le front-end Angular et le back-end Flask.
- o Traite les requêtes utilisateur pour le téléchargement, la suppression et la recherche d'images.

### 3. Front-end Angular

- o Offre une interface utilisateur intuitive.
- o Permet de charger des images, d'afficher les descripteurs et de visualiser les résultats de recherche.

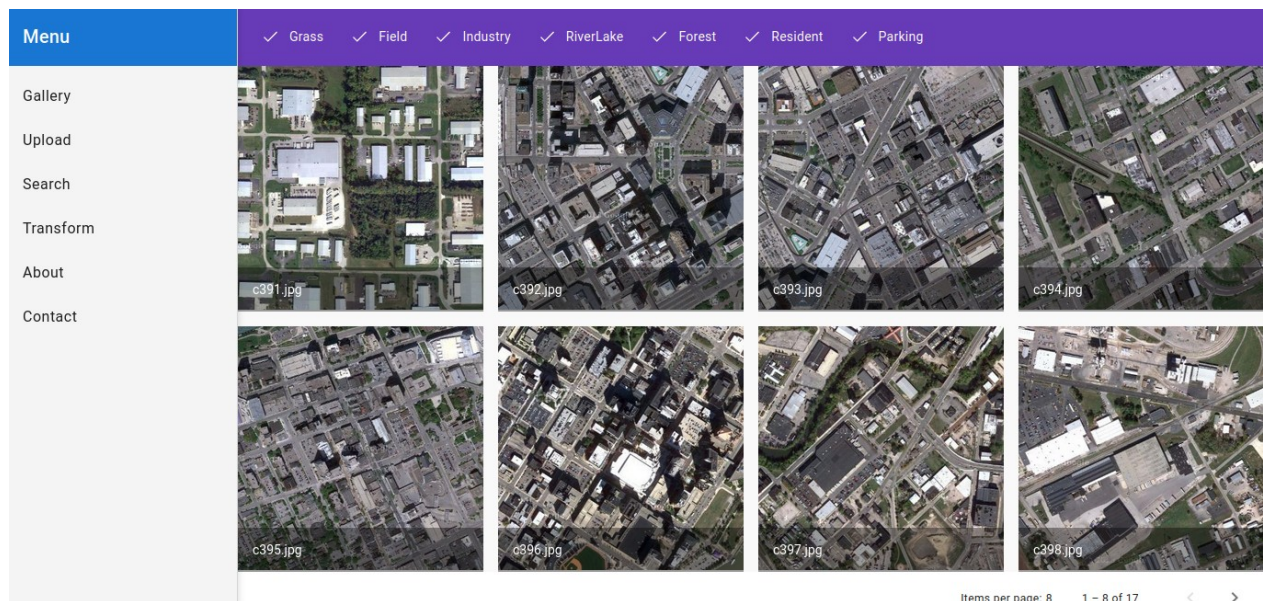
### 4. Base de données MongoDB

- o Stocke les descripteurs et les métadonnées des images.
- o Facilite la gestion des catégories d'images.

## Interface de l'Application :

### Gallery :

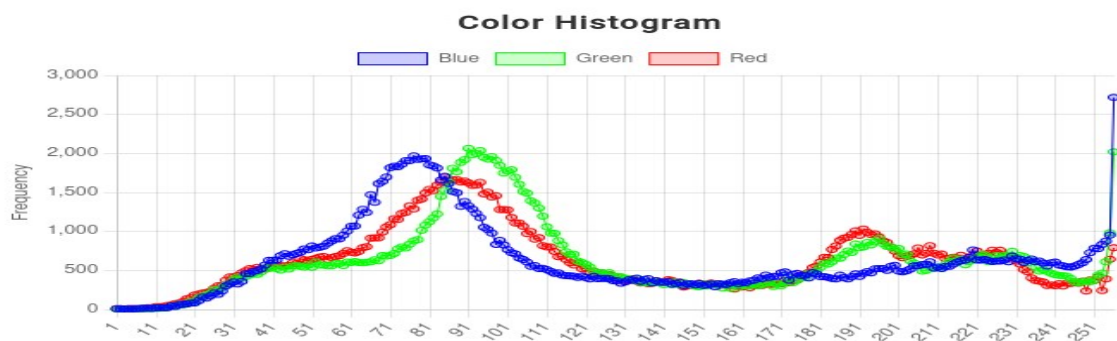
Cette page présente les images contenu dans la base de données , avec un filtre pour filtrer par catégorie :

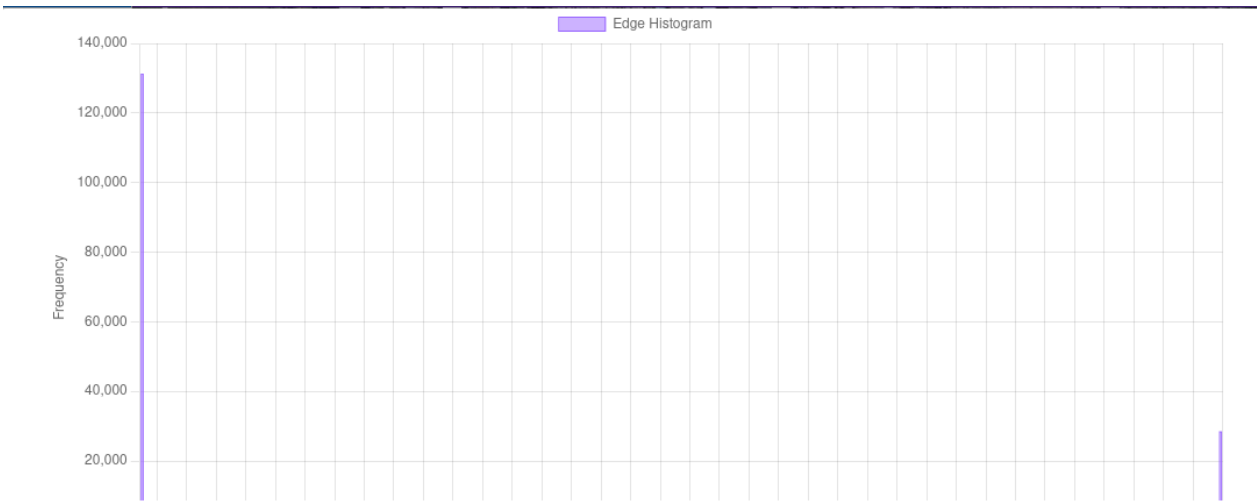
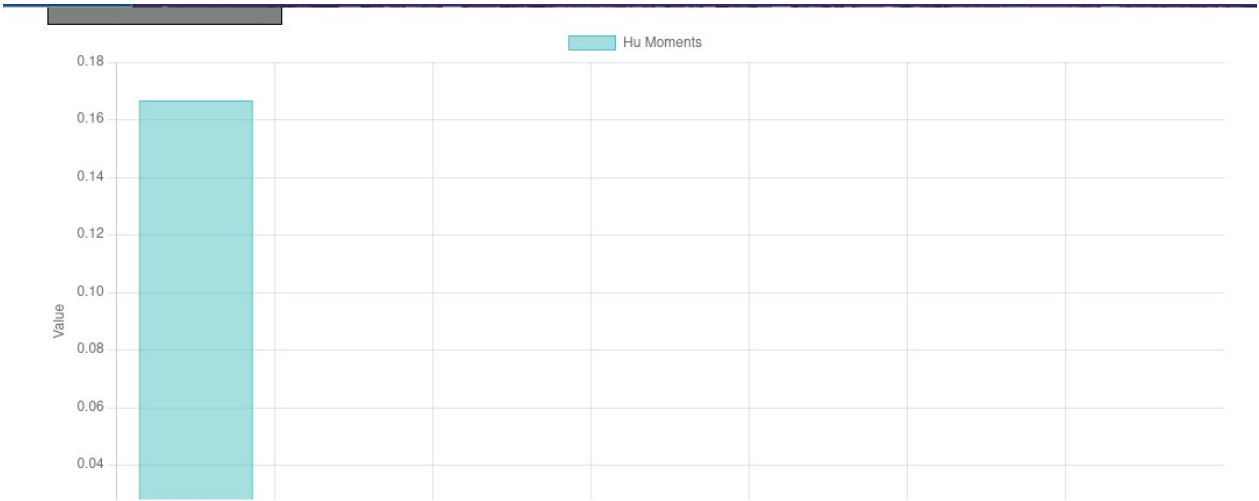
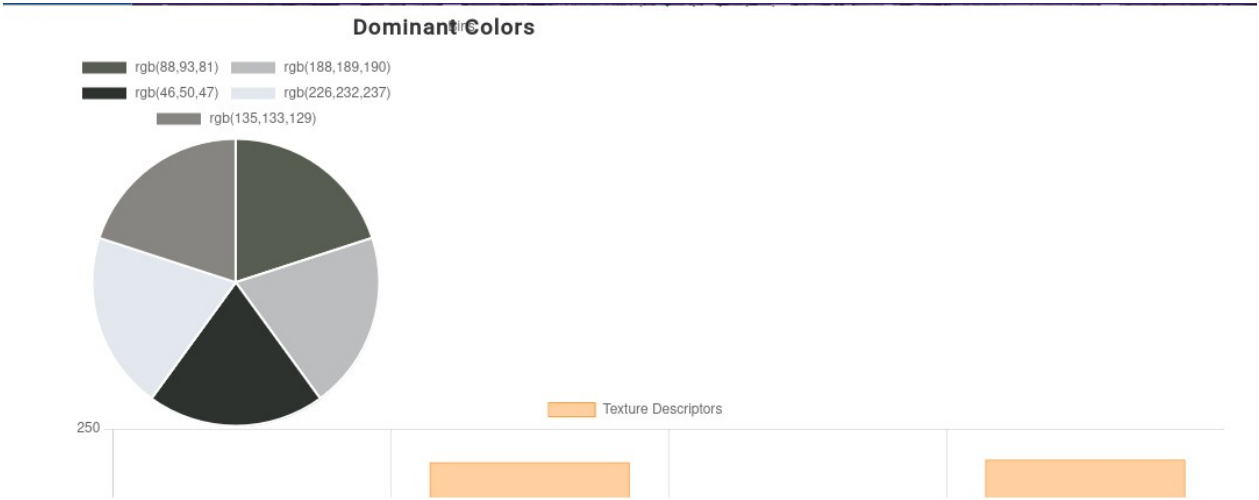


### Caractéristiques:

On cliquant sur les images dans l'application il devient possible de visualiser les différents caractéristiques de chaque image :

### Image Characteristics





Page Upload:



Ici il est possible d'ajouter les images dans la gallery

### Image Upload

Drag & drop your images here or click to upload






Industry

Save to Gallery

### Page Search :

On utilisant une image il devient possible d'effectuer une recherche simple sur la base de données par similarités .




**c394.jpg**  
Score: -13.896137069799616  
Category:

**c400.jpg**  
Score: 3196389.4762980994  
Category:


**g392.jpg**  
Score: 3263747.7425526897  
Category:

Il devient possible les images pertinent par le bouton like pour améloirer les résultat de la recherche.






**c394.jpg**  
Score: -13.896137069799616  
Category:




[Download](#) [Like](#)

**c400.jpg**  
Score: 3196389.4762980994  
Category:



[Download](#) [Like](#)


**g392.jpg**  
Score: 3263747.7425526897  
Category:




[Download](#) [Like](#)

Après avoir sélectionné les images pertinentes le bouton pour améliorer la recherche s'allume :

**Recherche Pertinence**




**c394.jpg**  
Score: -13.896137069799616  
Category:



[Download](#) [Liked](#)

**c400.jpg**  
Score: 3196389.4762980994  
Category:



[Download](#) [Like](#)

### Page Transform:

Celle-ci est la page de transformation il est possible d'effectuer des transformations sur l'image :

## Live Image Transformation

Upload Image:

Browse... c392.jpg

### Transformations

Crop Coords (x, y, w, h):

e.g., 10,10,100,100

Resize Dimensions (width, height):

e.g., 200,200

Flip:

None

Rotate Angle:

e.g., 90

Apply Transform

Preview

90

Apply Transform

Preview



## Conclusion :

Ce projet a permis de développer un système innovant de recherche d'images basé sur des descripteurs visuels et des mécanismes avancés de feedback de pertinence. En combinant les descripteurs de couleur, de texture et de forme, le système offre une recherche précise et intuitive. Grâce à l'intégration des techniques de

feedback de pertinence, l'expérience utilisateur est continuellement améliorée en alignant les résultats avec les préférences exprimées.

Le frontend, conçu avec MEAN, propose une interface moderne et intuitive pour interagir avec le système. Le backend, développé avec Flask et Flask-RESTful, assure une gestion efficace des API REST et des calculs complexes, notamment pour l'extraction et l'optimisation des descripteurs visuels.

En conclusion, ce projet démontre la puissance des technologies modernes combinées à des approches algorithmiques avancées pour répondre aux besoins croissants en matière de gestion et de recherche d'images. Les futures améliorations pourraient inclure l'intégration de modèles d'apprentissage profond pour une extraction et une classification encore plus précise.

## Références :

1. <https://github.com/palewithout/RSSCN7>
2. Dongping Tian : A Review on Relevance Feedback for Content-based Image Retrieval
1. OpenCV Documentation : <https://docs.opencv.org>
2. Flask Documentation : <https://flask.palletsprojects.com>
3. Angular Material Documentation : <https://material.angular.io>