



UNIVERSITÉ
TOULOUSE III
PAUL SABATIER



118 RTE DE NARBONNE, 31062 TOULOUSE

M2 EEA SIA

RAPPORT DE PROJET

Informatique et Projets Scientifiques

Gestion de bibliothèques de fichiers signaux/images et traitements de ces fichiers

Auteurs :

Lyakout TESSADA
Wissem SMATI
Manel TAIABI
Yasmine KARIM
Hugo ANDRE
Daniela TALBA

Tuteurs :

Agnan DE BONNEVAL
Florence PUGEAULT

28 janvier 2023

REMERCIEMENTS

Nous tenons à exprimer notre gratitude à toutes les personnes qui ont contribué de près ou de loin à la réalisation de ce projet.

Nous remercions en premier lieu nos tuteurs monsieur Agnan DE BONNEVAL et madame Florence PUGEAULT pour leur encadrement et accompagnement tout au long de ce projet ainsi que leurs disponibilités et leurs conseils avisés.

Nous remercions les professeurs Nahla TABTI, Ali EL AMINE, Stanislas PEDEBEARN et Pascal BERTHOU pour leur aide précieuse tout au long du déroulement de ce projet. Leur soutien et leur expertise ont été d'une grande importance pour la réussite de ce travail.

Nous remercions également les membres de notre équipe pour leur engagement, leur dévouement et leur travail acharné, sans eux le projet n'aurait pas pu aboutir.

Finalement, nous remercions nos chers amis qui nous ont aidé et soutenu constamment et qui nous ont accordé leur temps et leur patience pour la réussite de ce projet.

Ce projet a été une expérience riche en enseignements et nous espérons qu'il contribuera à l'avancée de notre domaine.

TABLE DES MATIÈRES

Glossaire	0
Introduction	1
1 Présentation du projet	2
I Objectifs du projet	2
II Organisation du projet	2
III Planification du travail	3
IV Répartition des taches	4
V Évaluation des risques	4
VI Matrice SWOT	5
2 Mise en œuvre du projet	7
I Problématique	7
II Architecture globale de l'application et ses principales fonctionnalités	7
II.1 Technologies mises en place	7
II.2 Description des données utilisées	8
II.3 Description des classes utilisées	8
II.3.1 Classe Bibliotheques	9
II.3.2 Classe traitement	10
III Traitements d'images	11
III.1 Histogramme et Égalisation	11
III.2 Filtre médian	12
III.3 Filtrage par convolution	13
III.3.1 Filtre Gaussien	13
III.3.2 Filtre Moyenneur	14
III.4 Filtre Sobel	15
III.5 Seuillage Simple	16
III.6 Rehaussement de contour	16
III.7 Segmentation Couleur	16
3 Présentation des résultats	17
I Identification des utilisateurs	17
II Fonctionnalités d'un utilisateurs de niveau 1	19
II.1 Affichage	19
II.2 Mise à jour	20
II.3 Trier la bibliothèque	21
II.4 Sous-listes des descripteurs des images	23
II.5 Sauvegarder la bibliothèque et récupérer un fichier a partir d'une bibliothèque existante .	24
II.6 Appliquer un traitement a l'image sélectionnée par l'utilisateur	24
III Fonctionnalités accessibles à l'utilisateur de niveau 2	30
Conclusion	31

Bibliographie	32
Annexe	33

TABLE DES FIGURES

1.1	Diagramme de phases	2
1.2	Diagramme de Gantt prévu	3
1.3	Diagramme de Gantt réel	3
1.4	Répartition des tâches	4
1.5	Anticipation des risques	5
1.6	Matrice SWOT du projet	6
2.1	Diagramme de classes	8
2.2	Classe Bibliothèque	9
2.3	Classe traitement	10
2.4	Algorithme d'égalisation de l'histogramme	11
2.5	Algorithme du filtre médian	12
2.6	Algorithme de la fonction de convolution	13
2.7	Gradient horizontal et gradient vertical	15
3.1	Identification des utilisateurs	17
3.2	Accès non autorisé	18
3.3	Menu d'affichage d'un utilisateur de niveau 1	19
3.4	Liste des descripteurs	19
3.5	Coût de l'image sélectionnée	20
3.6	Menu de la mise à jour de la bibliothèque	21
3.7	Affichage des descripteurs après la suppression d'une image	21
3.8	Menu du tri de la Bibliothèque	22
3.9	Tri du coût par ordre croissant	22
3.10	Tri du titre d'une image par ordre alphabétique	23
3.11	Menu des sous-listes	23
3.12	Sous-listes	24
3.13	Sauvegarder la bibliothèque	24
3.14	Affichage du menu de traitement d'images	24
3.15	Histogramme et Égalisation d'histogramme de l'image	25
3.16	Image originale "Fille" et Image "Fille" bruitée	25
3.17	Image filtrée	26
3.18	Image originale "Cérébrale" et image filtrée	26
3.19	Image originale "Loup" et image bruitée	27
3.20	Image filtrée	27
3.21	Image originale "Splash" et image filtrée	28
3.22	Rehaussement de contours sur l'image "Route"	28
3.23	Image originale "Ville" et image seuillée	29
3.24	Image originale "Pots" et image segmentée	29
3.25	Menu et affichage pour l'utilisateur de niveau 2	30
3.26	Affichage du coût d'une image pour l'utilisateur de niveau 2	30

GLOSSAIRE

SWOT : Strengths, Weaknesses, Opportunities, Threats.

IHM : Interface Homme-Machine.

OpenCV : Open Source Computer Vision.

PNG : Portable Network Graphics.

JPG : Joint Photographic Experts Group.

CSV : Comma Separated Values.

IDE : environnement de développement.

YUV : la luminance (Y) et la chrominance (U, V).

INTRODUCTION

Dans le cadre de notre Master 2 **Électronique, Énergie Électrique et Automatique** parcours **Signal, Imagerie et Apprentissage Automatique** et du Master 2 **Ingénierie de Santé** parcours **Imagerie Médicale** de l'Université Toulouse 3 Paul Sabatier, nous avons réaliser un mini-projet de techniques de programmation avec une mise en oeuvre en langage C++, dans le but de gérer et traiter des bibliothèques de fichiers signaux/images.

L'objectif de ce projet est de développer une application qui permet aux utilisateurs d'entrer leurs identifiants pour y accéder, de choisir une des images disponibles dans la base de donnée, de gérer sa fiche technique, puis de lui appliquer un traitement d'images qu'ils doivent sélectionner.

Ce travail nous permettra de développer nos compétences en programmation, en l'occurrence en langage C++, et de mettre en pratique les algorithmes de traitements d'images étudiés pendant notre formation, à l'aide de la bibliothèque **Opencv**. Il nous offrira également une expérience en gestion de projet et en travail en équipe.

Dans ce rapport, nous allons vous exposer le travail accompli tout au long de ces trois derniers mois, en commençant par une présentation du projet sous les différents aspects de sa conception jusqu'à sa réalisation, avant d'aborder la problématique traitée et les outils techniques mis en œuvre. Finalement, après avoir pris connaissance de la méthodologie de travail, nous allons vous dévoiler les résultats obtenus pour les discuter et les analyser afin de conclure sur notre expérience.

CHAPITRE 1

PRÉSENTATION DU PROJET

Dans cette partie, nous allons traiter toutes les étapes et plans de gestion et l'organisation de ce projet, de l'analyse du cahier des charges à la réalisation. Nous allons aussi éclaircir les apports ainsi que les contraintes techniques et humaines du travail en équipe.

I Objectifs du projet

L'objectif de ce projet est parfaire et compléter nos connaissances techniques sur la conception orientée objet et le langage de programmation C++, sans oublier la manipulation de fichiers et de structures dynamiques en appliquant le tout à un sujet concret en rapport avec un des domaines de notre formation.

Le C++ étant un outil potentiel pour la mise en oeuvre de problèmes dans d'autres matières de la formation, il nous permettra, à travers ce projet, de préparer et compléter d'autres matières de notre master en mettant en oeuvre des algorithmes déjà vus et réaliser dans d'autres langages de programmation. Cela nous permettra de montrer le caractère générique d'un algorithme puis de comparer les avantages et inconvénients de différentes mises en oeuvre d'un même algorithme.

De plus, ce projet nous permettra d'apprendre l'utilisation de la bibliothèque **opencv** très utilisée en traitement d'images.

II Organisation du projet

Dans le but de gérer l'organisation de notre projet, nous avons suivi les phases définies dans le diagramme ci dessous :

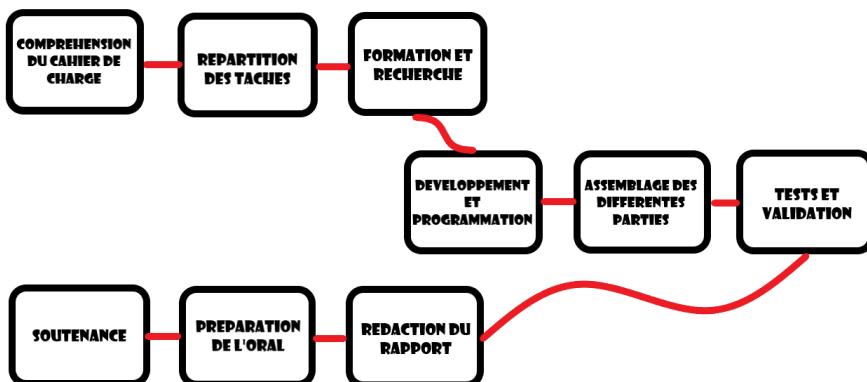


FIGURE 1.1 – Diagramme de phases

- Nous avons défini neuf étapes importantes pour la réalisation du projet ;
- La compréhension du cahier de charge assurant la clarté des consignes exigées.
 - La répartition des tâches permettant de partager le travail de manière équitable entre les différents membres du groupe.
 - La formation et la recherche concernant la partie d'apprentissage de la programmation en C++ et de l'auto-formation éventuelle.
 - Le développement, l'étape où chacun programme la partie du traitement qui lui a été attribuée.
 - Par la suite, on rassemble les différentes parties du code pour n'en former qu'un.
 - Puis, on teste le tout pour assurer le bon fonctionnement de l'ensemble.
 - Et finalement, on entame la rédaction du rapport qui contient tous les détails du projet.
 - Après cela vient la préparation de l'oral et des diapos pour passer la soutenance.

III Planification du travail

Pour veiller sur la bonne coordination et suivre l'avancement du projet tout au long de ces 3 mois de travail, nous avons adopté le planning illustré sur le diagramme à barres de Gantt ci-dessous afin de bien gérer la durée du projet. Il s'agit d'une représentation graphique des tâches du projet, des dates de début et de fin de ces tâches et des dépendances entre elles.

La première figure représente la planning prévu avant le commencement du projet.

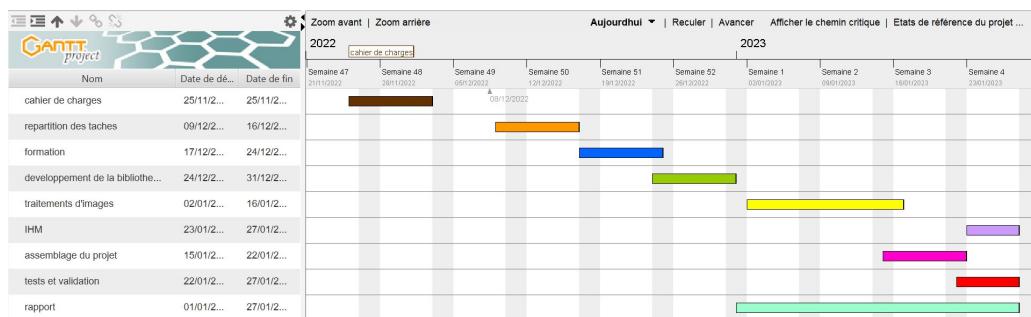


FIGURE 1.2 – Diagramme de Gantt prévu

La deuxième, représente les durées réelles qui ont été consacrées pour chaque tache.

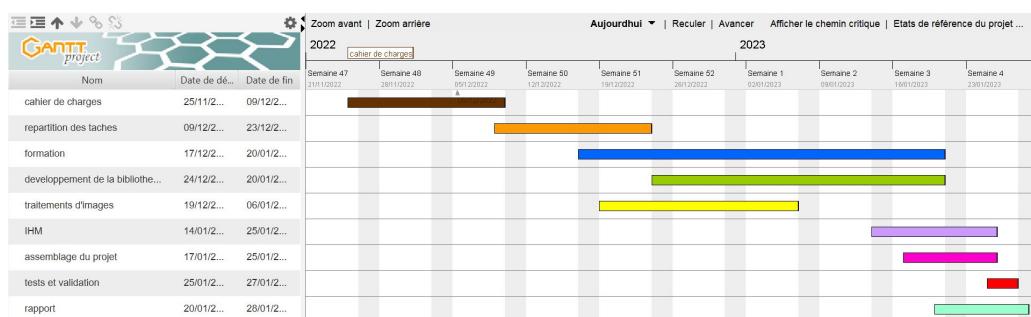


FIGURE 1.3 – Diagramme de Gantt réel

IV Répartition des taches

La répartition des taches est une étape clé de la réussite du projet. Nous avons fait en sorte de distribuer le travail équitablement sur les membres du groupe en tenant compte des compétences et motivations de chacun d'entre nous afin de veiller sur le bon avancement du projet.

	<i>Yasmine</i>	<i>Lyakout</i>	<i>Wissem</i>	<i>Hugo</i>	<i>Manel</i>	<i>Daniela</i>
<i>Gestion de la bibliothèque de fichiers images</i>	. Affichage de la liste complète des descripteurs . Sous listes par coût et par source . Sauvegarde de la bibliothèque dans un fichier	. Affichage du coût . Suppression de l'image . Tri du coût par ordre croissant et par ordre alphabétique . Modifier . Ajouter	. Affichage du coût . Suppression de l'image . Tri du coût par ordre croissant et par ordre alphabétique . Modifier . Ajouter		. Affichage de la liste complète des descripteurs . Sous listes par coût et par source . Sauvegarde de la bibliothèque dans un fichier	
<i>Traitements d'images</i>	. Filtrage de l'image par un filtre gaussien . Rehaussement de contours détecté par le Laplacien	. Segmentation couleur . Seuillage simple	. Filtrage de l'image par un filtre médian . Rehaussement de contours détecté par le Laplacien	- Histogramme et son égalisation	. Filtrage par convolution . Filtre de Sobel	Filtrage de l'image par un filtre moyenieur
<i>IHM</i>		X				
<i>Assemblage</i>	Dxygen		Assemblage des traitements d'images avec la bibliothèque (Héritage)		Assemblage des traitements d'images avec la bibliothèque (Héritage)	
<i>Rapport</i>	Participation	Participation	Participation	Participation	Participation	Participation

FIGURE 1.4 – Répartition des taches

V Évaluation des risques

Un risque est une situation ou une activité qui pourrait entraîner des conséquences négatives. Les risques peuvent être évalués et gérés pour minimiser les conséquences potentielles.

Pour éviter de faire face à ces contraintes, nous avons anticipé leurs évaluations en les identifiant et en essayant de mettre en place des actions pour les contrer.

Action mise en place pour éviter le risque	Identification des risques	Famille	Typologie	Gravité	Apparition	Non détection	Criticité
Validation de la conception par le tuteur	Mauvaise compréhension du cahier de charge	Conception	Endogène	8	5	5	200
Souplesse et organisation	Mauvaise gestion du temps	Management	Exogène	6	9	8	432
Autoformation	Faibles compétences en programmation	Technique	Endogène	7	8	2	112
Commenter le code	Bugs lors de l'assemblage des codes	Technique	Endogène	8	8	2	128
Prendre de la marge	Respect des délais	Management	Endogène	7	6	6	252

FIGURE 1.5 – Anticipation des risques

VI Matrice SWOT

SWOT est un acronyme pour les termes anglais "Strengths", "Weaknesses", "Opportunities" et "Threats". Il s'agit d'une méthode d'analyse utilisée pour évaluer les forces, les faiblesses, les opportunités et les menaces du projet. Cette analyse permet de déterminer les atouts et les points à améliorer d'une entreprise, ainsi que les opportunités et les risques potentiels. Elle peut être utilisée pour élaborer des stratégies d'entreprise et prendre des décisions éclairées.

FORCES / ATOUTS

- Nos différentes compétences
- Motivation

FAIBLESSES / HANDICAPS

- Différentes spécialités
- Différents emplois du temps
- Capacité en programmation
- Charge du travail

OPPORTUNITES

- Acquérir de nouvelles compétences
- Expérimenter le travail en équipe

MENACES / RISQUES

- Ne pas valider le module

FIGURE 1.6 – Matrice SWOT du projet

CHAPITRE 2

MISE EN ŒUVRE DU PROJET

Dans cette partie, nous allons traiter la problématique principale du projet en présentant l'architecture globale de notre application et ses principales fonctionnalités.

I Problématique

Dans le but de gérer une bibliothèque d'images et de réaliser des traitements sur ces dernières, notre projet consiste à créer une application qui permet de réaliser des traitements d'images à l'aide d'une interface homme-machine (IHM) permettant aux utilisateurs de niveau 1 ou 2 de traiter les images stockées dans une bibliothèque. La gestion des fonctions de cette bibliothèque, quant à elle, dépend du niveau de ces utilisateurs. Nous allons éclaircir dans ce chapitre, les différentes classes utilisées, en l'occurrence la classe bibliothèque où nous avons mis en place plusieurs fonctionnalités, ainsi que la classe des traitements effectués sur les images de notre base de données en expliquant quelques algorithmes et les résultats obtenus.

II Architecture globale de l'application et ses principales fonctionnalités

II.1 Technologies mises en place

Pour réaliser ce projet et mettre en œuvre notre application nous avons travailler sous l'environnement **Windows** avec le logiciel **Visual Studio** car son utilisation et son installation était simple et pratique pour tous les membres du groupe. Par ailleurs, un membre du groupe a utilisé un environnement **MacOS** car c'est un système d'exploitation stable et bien documenté qui offre un environnement de développement intégré (IDE) solide, comme Xcode, qui facilite la création, le débogage et le déploiement d'applications, il offre également un système de fichiers efficace et fiable pour la gestion de fichiers, ce qui est important pour notre application de traitement d'images qui gère de nombreux fichiers d'images.

De plus, nous avons utilisé la bibliothèque opencv qui est une bibliothèque de fonctions de programmation open source et multiplateforme possédant une interface C++.

Enfin, nous avons fait usage de Doxygen qui est un outil de génération de documentation pour le code source. Il prend en entrée des fichiers de code source et génère une documentation détaillée en sortie qui peut inclure des informations sur les classes, les fonctions, les variables, les dépendances, etc. Doxygen utilise des commentaires dans le code source pour déterminer ce qui doit être inclus dans la documentation.

II.2 Description des données utilisées

Dans cette partie, nous allons décrire la manière dont les différents composants de l'application interagissent entre eux ainsi que les principales fonctionnalités de celle-ci. Tout d'abord, nous avons créer deux dossiers appelés "**Images**" et "**Bibliotheques**" afin de stocker les images de références et les informations associées aux images respectives. Nous avons choisi le format "PNG" (Portable Network Graphics) du fait qu'il offre une meilleure qualité d'image que le JPG pour les images avec des détails fins, car il utilise une compression de données moins importante que le JPG. Par ailleurs, nous avons choisi des fichiers "CSV" (Comma Separated Values) car ils permettent de stocker des données structurées, de stocker des données sous forme de tableaux à plusieurs colonnes et lignes. Ces fichiers là nous les avons stockés dans le dossier "Bibliotheque". Prenons l'exemple du fichier "Pots.csv", une bibliothèque où chaque ligne correspond aux images et chaque colonne correspond aux descripteurs.

Nous avons choisi d'utiliser des fichiers CSV car ils permettent une gestion efficace des fonctionnalités. En utilisant la bibliothèque **vector** en C++, nous pouvons facilement ajouter, supprimer et accéder aux éléments d'un vecteur, ce qui nous permet de naviguer entre les lignes qui décrivent chaque image.

II.3 Description des classes utilisées

Notre programme comporte deux classes principales : une pour les traitements des images, intitulée "**Traitemet**", et l'autre pour la gestion de la bibliothèque d'images et de leurs descripteurs, intitulée "**Bibliotheque**".

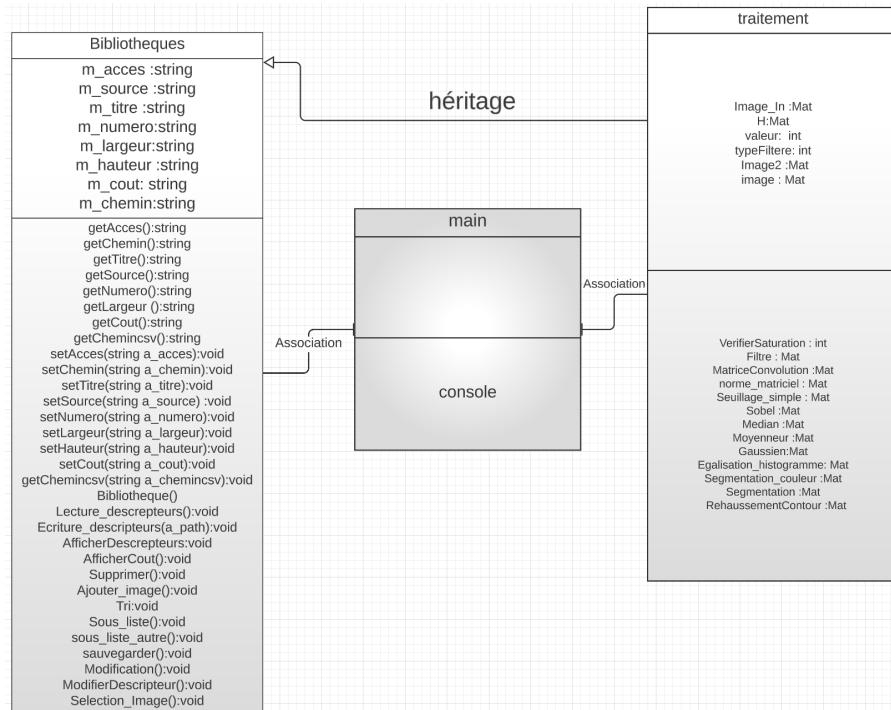


FIGURE 2.1 – Diagramme de classes

II.3.1 Classe Bibliothèques

Notre classe "Bibliothèque" est conçue pour gérer les chemins d'accès et les attributs fondamentaux des images. Nous avons utilisé des mécanismes d'encapsulation pour protéger les données de l'objet contre les erreurs de manipulation et les accès non autorisés. Nous avons utilisé les accesseurs "get" et "set" pour chaque attribut pour faciliter les modifications et permettre une utilisation intuitive des données par les autres parties du programme. Les attributs privés de cette classe sont de type **chaîne de caractères** et représentent les différents types de descripteurs (titre, source, numéro, accès, largeur, hauteur, coût). Nous allons maintenant décrire les fonctionnalités implémentées dans cette classe, telles que :

FONCTIONS	DÉFINITIONS
VERIFIER_EXTENSIONCSV(STRING A_NAME_FILE)	Permet de vérifier si le nom entré par l'utilisateur contient l'extension "CSV" ou pas.
LECTURE_DESCRIPTEURS()	Lis les descripteurs dans le fichier en utilisant la bibliothèque <iostream> de C++
ECRITURE_DESCRIPTEURS(STRING CONST A_PATH)	Permet d'écrire dans une bibliothèque
ECRITURE_DESCRIPTEURS(STRING CONST A_PATH)	Retourne l'affichage des descripteurs selon le numéro de l'image
AFFICHERCOUT ()	Fonction qui afficher le coût selon le numéro d'image rentré par l'utilisateur
SUPPRIMER()	Supprime les descripteurs de l'image selon son numéro
AJOUTER_IMAGE()	Permet d'ajouter des descripteurs associés à une image dans la bibliothèque
MODIFIER_DESCRIPTEUR()	Permet de modifier plusieurs caractéristiques de l'image
TRI()	Tri du cout d'une image par ordre croissant / tri du titre par ordre alphabétique
SOUS_LISTE()	Permet d'appliquer un filtre médian sur une image
SOUS_LISTE_AUTRE()	Permet d'afficher les sous-listes selon le critère de la source de l'image

FIGURE 2.2 – Classe Bibliothèque

II.3.2 Classe traitement

Dans cette classe, nous avons défini les différentes fonctions de traitement d'images que nous avons implémentées. Cette classe est liée à la classe Bibliothèques, et nous avons mis en évidence ce lien en utilisant la notion d'héritage entre ces deux classes. Les principales méthodes utilisées dans cette classe sont représentées dans le tableau suivant :

FONCTIONS	DÉFINITIONS
AFFICHERCONTENUIMAGE(CONST MAT IMAGE_IN)	Permet d'afficher le contenu d'une image stockée dans notre bibliothèque de données
TRAITEMENTIMAGE()	Permet de regrouper tous les traitements effectués dans un 'switch' pour faciliter leur récupération
VERIFIERSATURATION(CONST INT VALEUR)	Permet de vérifier que l'attribut "valeur" de cette fonction appartient à une plage valide pour réaliser des opérations de traitement d'images
FILTRE(CONST INT TYPEFILTRE)	Permet de générer plusieurs types de masque de convolution
MATRICECONVOLUTION(CONST MAT IMAGE_IN, CONST MAT H)	Réalise une opération de convolution dans d'autres traitement d'images tel que le moyenneur
NORME_MATRICIEL (CONST MAT IMAGE, CONST MAT IMAGE2)	Permet le calcul de la norme de gradient
SEUILLAGE_SIMPLE()	Permet d'appliquer une binarisation selon un seuil choisi par l'utilisateur
SOBEL()	Permet la détection des contours des images
MOYENNEUR()	Permet d'appliquer un filtre moyenneur sur l'image
MEDIAN()	Permet d'appliquer un filtre médian sur une image
GAUSSIEN()	Permet d'appliquer un filtre gaussien en utilisant un noyau qui dépend de la variance saisie par l'utilisateur
EGALISATION_HISTOGRAMME()	Permet de calculer et afficher l'histogramme puis d'équilibrer le contraste de l'image
REHAUSSEMENT_CONTOUR()	Permet de mettre en évidence les contours d'une image en accentuant les variations de luminosité ou de couleur entre les pixels adjacents

FIGURE 2.3 – Classe traitement

III Traitements d'images

III.1 Histogramme et Égalisation

L'histogramme d'une image est un tableau qui montre la répartition des différents niveaux de gris présents dans l'image, en indiquant le nombre de pixels ayant chaque niveau de gris. Dans le cas d'une image codée sur 8 bits, il y a 256 niveaux de gris possibles, donc 256 entrées dans l'histogramme. L'égalisation de l'histogramme consiste à rendre la répartition des niveaux de luminosité plus uniforme afin d'augmenter les nuances dans l'image. Pour réaliser cette opération, on calcule d'abord l'histogramme en parcourant tous les pixels de l'image, puis on calcule les probabilités cumulées de chaque valeur de pixel et on les applique à chaque pixel en utilisant une boucle. Ci-dessous l'algorithme utilisée :

```
Entrée :  
    Image Original : Matrice  
Sortie :  
    Histogramme : Matrice  
Variable :  
    TAB[ 256 ] : Tableau  
Maximal ← 0 : Entier  
Nb_ligne, Nb_colonne : Entrée  
Pour i allant 0 jusqu'à NB_ligne  
Faire  
    Pour j allant de 0 jusqu'à Nb_colonne  
    Faire  
        Indice tableau= Image original ( i , j )  
        TAB [ indice tableau ] = TAB1 [ indice tableau ] +1  
    Fin pour  
Fin pour  
Pour i allant de 0 jusqu'à 255  
Faire  
    Si TAB1 [ i ] > maximale  
    Fin si  
Fin pour  
Pour i allant de 0 jusqu'à 255    \\\ tracer les rectangles de l'apparition de chaque pixel
```

FIGURE 2.4 – Algorithme d'égalisation de l'histogramme

III.2 Filtre médian

Le filtrage médian est une technique utilisée pour réduire les bruits impulsionnels dans une image en remplaçant chaque pixel par la valeur médiane de ses voisins proches. Pour réaliser ce filtrage, on convertit d'abord l'image en niveau de gris, puis pour chaque pixel de cette image, on calcule la valeur médiane des pixels de sa fenêtre de voisinage de taille (3x3). On stocke ensuite les valeurs des pixels de la fenêtre dans un vecteur, qu'on trie pour affecter la valeur médiane au pixel en cours, ce qui permet d'obtenir une image filtrée et moins bruitée.

```
Entree :  
    Image originale : matrice  
Sortie :  
    Image filtrée : matrice  
Variable :  
    Nb_ligne, Nb_colonne , i , j : Entier  
TAB[ ] : double  
Pour i allant de 1 jusqu'à Nb_ligne  
Faire  
    Pour j allant de 1 jusqu'à Nb_colonne  
        Faire  
            TAB [ Image original ( i-1, j-1 ) ]  
            TAB [ Image original ( i, j-1 ) ]  
            TAB [ Image original ( i+1, j-1 ) ]  
            TAB [ Image original ( i-1, j ) ]  
            TAB [ Image original ( i, j ) ]  
            TAB [ Image original ( i+1, j ) ]  
            TAB [ Image original ( i-1, j+1 ) ]  
            TAB [ Image original ( i, j+1 ) ]  
            TAB [ Image original ( i+1, j+1 ) ]  
            Sort ( TAB [ i , j ] ) \\\ trier les éléments de tableau de l'ordre croissant ou decroissant  
            Image filtrée ( i-1, j-1 ) = TAB [ 4 ]  
        Fin pour  
    Fin pour
```

FIGURE 2.5 – Algorithme du filtre médian

III.3 Filtrage par convolution

La convolution 2D consiste à appliquer un filtre à une image en utilisant une matrice de convolution (H). Pour cela, on calcule le produit de convolution de l'image originale $I(x,y)$ par la réponse impulsionnelle $H(i,j)$ du filtre, cela s'exprime par :

$$J(x, y) = \sum_{i=0}^2 \sum_{j=0}^2 H(i, j) \times I(x - 1 + j, y - 1 + i) \quad (2.1)$$

L'algorithme de convolution utilise une double boucle afin de parcourir tous les pixels de l'image, puis effectue une multiplication des valeurs des pixels voisins avec les valeurs de la matrice de convolution H , et enfin une addition des résultats de ces multiplications pour obtenir la valeur de chaque pixel de l'image traitée. Cette fonction prend en entrée une image (Mat Image_In) et une matrice de convolution (Mat H) et retourne une image traitée (Mat Image_Out). Ci-dessous l'algorithme utilisée :

```

Entrée
  Image original : Matrice
  Filtre de convolution : Matrice

Sortie
  Image filtrée : Matrice

Variable :
  Valeur, i, j , Nb_ligne, Nb_colonne : Entier
  Valeur =0

Pour i allons de 1 jusqu'à Nb_ligne
  | Faire
    | Pour j allons de 1 jusqu'à Nb_colonne
      |   | Faire
        |   |   Valeur ← Image original (i-1, j-1) * H(0,0) + Image original (i-1, j) * H(0,1) +
        |   |   Image original (i-1, j+1) * H(0,2) + Image original (i, j-1) * H(1,0) +
        |   |   Image original (i, j) * H(1,1) + Image original (i, j+1) * H(1,2) +
        |   |   Image original (i+1, j-1) * H(2,0) + Image original (i+1, j) * H(2,1) +
        |   |   Image original (i+1, j+1) * H(2,2)
        |   |   Image filtrée ( i-1, j-1) ← valeur
      |   | Fin pour
    |   | Fin pour
  | Fin pour

```

FIGURE 2.6 – Algorithme de la fonction de convolution

III.3.1 Filtre Gaussien

La forme du filtre gaussien est dérivée de l'équation d'une gaussienne :

$$G_\sigma(x, y) = \frac{1}{2\pi\sigma^2} e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)} \quad (2.2)$$

où ;

σ est le paramètre de réglage de la largeur de la fonction gaussienne,
et x, y sont les coordonnées dans l'espace.

Il est utilisé pour réduire le bruit et améliorer la netteté d'une image en utilisant une technique de lissage.

D'après l'algorithme montré ci-dessus, la lecture de l'image est effectuée en utilisant la fonction "getImage()". Des matrices intermédiaires telles que "blur" et "Image-Out" sont déclarées pour stocker des images temporaires. Les paramètres du filtre gaussien tels que σ , la dimension, x_0 et y_0 , sont stockés dans un tableau 2D. L'image est convertie en niveaux de gris, puis les coefficients du filtre gaussien sont calculés en utilisant la formule mathématique de la fonction gaussienne en bouclant sur les dimensions du filtre (dim x dim). Ces coefficients sont utilisés pour appliquer le filtre gaussien à l'image en niveaux de gris. Pour chaque pixel, une somme pondérée des pixels voisins est calculée et divisée par la somme totale des poids pour obtenir la valeur moyenne pondérée.

III.3.2 Filtre Moyenneur

En traitement d'images, un filtre moyenneur est souvent utilisé pour atténuer le bruit additif d'une image. Afin de l'appliquer, l'algorithme prend en entrée une image, la convertit en niveau de gris, puis pour chaque pixel de cette image en niveau de gris, il utilise un kernel de taille (3x3) en utilisant le pixel (x, y) comme le pixel central de ce kernel. Ensuite, il effectue une convolution (fonction : **MatriceConvolution**) en multipliant chaque valeur de pixel de l'image d'entrée par la valeur correspondante dans le kernel, puis additionne toutes les valeurs obtenues et les place dans l'image de sortie à la position (x, y). Cette opération est répétée pour chaque pixel de l'image. Le résultat final est une image avec un niveau de bruit atténué.

Le kernel (noyau de convolution) est un petit tableau de nombres, utilisé pour effectuer l'opération de convolution sur l'image. Il détecte les caractéristiques spécifiques dans l'image, comme les contours ou les textures.

III.4 Filtre Sobel

Pour détecter les bords dans une image, nous utilisons le filtre Sobel qui est composé de deux masques de convolution. Le premier masque est utilisé pour détecter les bords horizontaux, tandis que le second masque est utilisé pour détecter les bords verticaux. Les deux masques sont ensuite combinés pour obtenir un résultat final plus précis. Pour appliquer ces masques, nous utilisons une technique de convolution en parcourant tous les pixels de l'image d'entrée et en multipliant chaque pixel par la valeur correspondante dans le masque, puis en additionnant les résultats pour obtenir la valeur du pixel de sortie.

Les masques utilisés pour détecter les bords horizontaux et verticaux sont respectivement :

-1	0	1
-2	0	2
-1	0	1

|

1	2	1
0	0	0
-1	2	-1

FIGURE 2.7 – Gradient horizontal et gradient vertical

Ensuite, nous avons utilisé une fonction de convolution pour combiner l'image d'entrée avec les masques de gradient horizontal et vertical, pour obtenir les gradients correspondants. La seconde étape consiste à normaliser les valeurs de gradient obtenues, pour cela nous avons utilisé la fonction de norme matricielle. Elle prend en entrée les matrices des gradients horizontaux et verticaux et retourne une matrice de sortie. La fonction parcourt chaque pixel des matrices d'entrée, calcule la norme en utilisant la formule $\sqrt{G_x^2 + G_y^2}$, où G_x et G_y sont les gradients horizontaux et verticaux. Enfin, elle affecte la valeur de la norme au pixel correspondant dans la matrice de sortie.

III.5 Seuillage Simple

Dans cette partie, nous avons réaliser le seuillage qui est une technique de traitement d'image utilisée pour transformer une image en noir et blanc en comparant chaque pixel à un seuil prédéfini par l'utilisateur. Tous les pixels ayant une valeur supérieure ou égale au seuil choisi prendra la valeur 255 , et si sa valeur est inférieure, elle prendra la valeur 0 suivant la formule suivante :

$$g(i, j) = \begin{cases} 255 & \text{si } f(i, j) \geq s \\ 0 & \text{sinon} \end{cases}$$

Le résultat du seuillage est une binarisation de l'image contenant des pixels noirs et blancs.

III.6 Rehaussement de contour

Cette technique consiste à accentuer les contours d'une image en utilisant le filtre laplacien. Elle est réalisée en soustrayant une version de l'image traitée avec le filtre laplacien de l'image originale. Nous avons fait appel à la fonction "MatriceConvolution" qui est utilisée pour appliquer le filtre laplacien et détecter les contours, puis ces contours détectés sont soustraits de l'image originale. Le résultat final est une image avec des contours plus marqués et plus nets. L'objectif est d'améliorer la clarté de l'image et de mettre en évidence les détails importants pour faciliter la reconnaissance. Le résultat final est une image avec des contours plus marqués et plus nets, en utilisant la formule suivante :

$$I'(x, y) = I(x, y) - c\nabla^2 I(x, y) \quad (2.3)$$

III.7 Segmentation Couleur

La segmentation de couleur est une technique d'analyse d'images utilisée pour séparer les différentes régions d'une image en fonction de leur couleur. Il est possible de réaliser cette segmentation couleur par région, par pixel ou par histogramme. Dans notre cas, nous avons opté pour la segmentation couleur par région pour grouper les pixels de l'image en fonction de leur couleur similaire.

Nous avons commencé par réaliser un pré-traitement en convertissant l'image en un espace de couleur YUV. C'est un espace de couleur qui sépare la luminance (Y) de la chrominance (U, V). Ensuite, nous avons segmenté en seuillant les composantes de luminance (Y) et de chrominance (U, V) pour chaque pixel, avec des seuils maximum et minimum.

L'utilisation de l'espace YUV et de la luminance, nous permet de séparer les régions de l'image en fonction de leur luminance similaire, plutôt que de leur couleur similaire, ce qui peut être utile dans certaines applications où la luminance est un facteur important. Cependant, cela peut ne pas être approprié pour les applications qui dépendent de la chrominance.

CHAPITRE 3

PRÉSENTATION DES RÉSULTATS

Notre programme a été exécuté dans une console pour afficher les résultats et permettre une interaction avec les utilisateurs. Cette console est une fenêtre qui agit comme une interface entre les applications non graphiques (cachées derrière la console) et l'utilisateur (face à la console), de sorte que l'entrée de l'utilisateur se fait principalement avec le clavier.

I Identification des utilisateurs

Le processus d'identification pour se connecter à l'application sera comme suit :

Pour l'utilisateur de niveau 1 : son identifiant est " LL11LL ", cela lui permet d'avoir accès à toutes les fonctionnalités de l'application. Depuis le menu principal affiché sur l'image ci-dessous, il peut sélectionner une fonctionnalité en entrant le numéro correspondant.

Pour l'utilisateur de niveau 2 : son identifiant est "LL22LL", leurs droits sont limités à un accès en consultation et uniquement sur les images ayant le caractère d'accès "A" définis préalablement. cet utilisateur peut donc affiché la liste de tous les descripteurs dont il saisira le numéro 1 et saisira le numéro 2 afin d'afficher le coût d'une image d'accès "A". Enfin, il peut simplement retourner au menu principal en saisissant le nombre 0.

```
(base) vm@virtual:~/Bureau/GROUPE_5_FINAL$ ./program.exe
*****
*****       Menu d'identification      *****
*****
*****bienvenue cher utilisateur
Veuillez entrer votre identifiant : LL11LL
Connexion réussie!
*****       Choix des bibliotheques      *****
Veuillez donner le nom d'une bibliotheque existante:
Pots
Bienvenue et Félicitation !
Accès autorisé aux fonctionnalités d'un utilisateur de niveau 1
*****
*****       Menu Principal de la gestion de bibliotheque  *****
*****
1. Affichage
2. Mise a jour de la bibliotheque
3. Triage de la bibliotheque
4. Sous liste des descripteurs
5. Sauvegarder la bibliotheque
6. Application d'un traitement d'images
0. Retour au menu d identification
Veuillez saisir votre choix!
```

(a) Utilisateur de niveau 1

```
(base) vm@virtual:~/Bureau/GROUPE_5_FINAL$ ./program.exe
*****
*****       Menu d'identification      *****
*****
*****bienvenue cher utilisateur
Veuillez entrer votre identifiant : LL22LL
Connexion réussie!
*****       Choix des bibliotheques      *****
Veuillez donner le nom d'une bibliotheque existante:
Pots
Bienvenue et Félicitation !
Accès autorisé aux fonctionnalités d'un utilisateur de niveau 2
*****
*****       Menu Principal de la gestion de bibliotheque  *****
*****
1. Affichage de la liste des descripteurs
2. Affichage du cout d'une image particulière
0. Retour au menu precedent
Veuillez saisir votre choix!
```

(b) Utilisateur de niveau 2

FIGURE 3.1 – Identification des utilisateurs

Les utilisateurs ne disposant pas d'un identifiant de niveau 1 ou 2 n'ont pas accès à cette application et n'ont aucun droit. Cela s'affichera alors comme suit :

```
o (base) vm@virtual:~/Bureau/GROUPE_5_FINAL$ ./program.exe
*****
*****          Menu d'identification          *****
*****
bienvenue cher utilisateur
Veuillez entrer votre identifiant : others
Identifiant incorrect
Accès refusé

Tapez sur la touche Entrée pour retourner au menu...
█
```

FIGURE 3.2 – Accès non autorisé

II Fonctionnalités d'un utilisateurs de niveau 1

II.1 Affichage

Nous avons créé un menu principal comprenant toutes les fonctionnalités pour que l'utilisateur puisse choisir une option pour traiter les descripteurs des images. Cet affichage comprend une présentation globale de tous les descripteurs en saisissant le nom de fichier ou en affichant le coût d'une image. L'utilisateur peut également saisir le numéro d'image à l'aide du clavier.

```
Descripteurs
Bienvenue et Félicitation !
Accès autorisé aux fonctionnalités d'un utilisateur de niveau 1
*****
***** Menu Principal de la gestion de bibliothèque *****
*****
1. Affichage
2. Mise à jour de la bibliothèque
3. Triage de la bibliothèque
4. Sous liste des descripteurs
5. Sauvegarder la bibliothèque
6. Application d'un traitement d'images
0. Retour au menu d'identification
Veuillez saisir votre choix!
1
*****
1. AFFICHAGE
*****
1. Affichage de la liste des descripteurs
2. Affichage du coût d'une image particulière
0. Retour au menu précédent
Veuillez saisir votre choix!
```

FIGURE 3.3 – Menu d'affichage d'un utilisateur de niveau 1

1. Afficher la liste complète des descripteurs des images correspondant aux droits de l'utilisateur courant

```
0. Retour au menu précédent
Veuillez saisir votre choix!
1
Accès : N || Chemin :./Data/Image/Route.png || Titre : Route || Cout : 0 || Source : Google image || Numéro : 1 || Largeur : 740 || Hauteur : 410

Accès : A || Chemin :./Data/Image/Cerveau.png || Titre : Cerveau || Cout : 10 || Source : Pinterest || Numéro : 7 || Largeur : 373 || Hauteur : 507

Accès : A || Chemin :./Data/Image/Splash.png || Titre : Splash || Cout : 40 || Source : Google image || Numéro : 11 || Largeur : 842 || Hauteur : 595

Accès : A || Chemin :./Data/Image/Ville.png || Titre : Ville || Cout : 50 || Source : Pinterest || Numéro : 3 || Largeur : 474 || Hauteur : 632

Accès : A || Chemin :./Data/Image/Rayures.png || Titre : Rayures || Cout : 80 || Source : Google image || Numéro : 8 || Largeur : 500 || Hauteur : 749

Accès : N || Chemin :./Data/Image/Formes.png || Titre : Formes || Cout : 100 || Source : Pinterest || Numéro : 10 || Largeur : 366 || Hauteur : 488

Accès : N || Chemin :./Data/Image/Tigre.png || Titre : Tigre || Cout : 130 || Source : Google image || Numéro : 6 || Largeur : 577 || Hauteur : 800

Accès : N || Chemin :./Data/Image/Verre.png || Titre : Verre || Cout : 180 || Source : Pinterest || Numéro : 5 || Largeur : 495 || Hauteur : 742

Accès : N || Chemin :./Data/Image/Fille.png || Titre : Fille || Cout : 200 || Source : Pinterest || Numéro : 9 || Largeur : 563 || Hauteur : 361
```

FIGURE 3.4 – Liste des descripteurs

2. Afficher le coût d'une image particulière dont l'utilisateur saisira le numéro au clavier

```

*****
*****          Menu d'identification          *****
*****
bienvenue cher utilisateur
Veuillez entrer votre identifiant : LL22LL

Connexion réussie!

*****
*****          Choix des bibliotheques          *****
*****
Veuillez donner le nom d'une bibliotheque existante:
Descripteurs

Bienvenue et Félicitation !
Accès autorisé aux fonctionnalités d'un utilisateur de niveau 2

*****
*****          Menu Principal de la gestion de bibliotheque      *****
*****
1. Affichage de la liste des descripteurs
2. Affichage du cout d'une image particulière
0. Retour au menu precedent

Veuillez saisir votre choix!
2
Entrer numero d'image que tu veux afficher son prix : 4
Image 4 : 470

Tapez sur la touche Entrée pour retourner au menu...

```

FIGURE 3.5 – Coût de l'image sélectionnée

On peut constater que lorsque l'image numéro 4 est sélectionnée, son prix est affiché correctement en dessous. Il est donc possible de conclure que la fonction est efficace.

II.2 Mise à jour

La mise à jour de la bibliothèque de données comprend trois opérations :

1. Suppression de descripteurs d'une image en saisissant son numéro, suivie d'une mise à jour automatique des numéros d'images.
2. Ajout d'une image en saisissant les nouveaux descripteurs ou en les saisissant manuellement.
3. Modification des caractéristiques d'une image en saisissant son numéro.

```

PROBLÈMES SORTIE CONSOLE DE DÉBOGAGE TERMINAL
0
Accès autorisé aux fonctionnalités d'un utilisateur de niveau 1
*****
***** Menu Principal de la gestion de bibliothèque *****
*****
1. Affichage
2. Mise à jour de la bibliothèque
3. Triage de la bibliothèque
4. Sous liste des descripteurs
5. Sauvegarder la bibliothèque
6. Application d'un traitement d'images
0. Retour au menu d'identification
Veuillez saisir votre choix!
2
*****
2. MISE A JOUR DE LA BIBLIOTHEQUE *****
1. Supprimer l'image
2. Ajouter une image
3. Modifier une ou des caractéristiques d'une image
0. Retour au menu précédent
Veuillez saisir votre choix!

```

FIGURE 3.6 – Menu de la mise à jour de la bibliothèque

On prend l'exemple de la méthode "supprimer une image" présenté ci-dessous :

1. Supprimer une image dont l'utilisateur donne le numéro

```

PROBLÈMES SORTIE CONSOLE DE DÉBOGAGE TERMINAL
Accès : L || Chemin : ./Data/Image/Ville.png || Titre : Ville || Cout : 50 || Source : Pinterest || Numéro : 3 || Largeur : 474 || Hauteur : 632
Accès : L || Chemin : ./Data/Image/Pots.png || Titre : Pots || Cout : 470 || Source : Google image || Numéro : 4 || Largeur : 876 || Hauteur : 634
Accès : N || Chemin : ./Data/Image/Tigre.png || Titre : Tigre || Cout : 130 || Source : Google image || Numéro : 6 || Largeur : 577 || Hauteur : 800
Accès : L || Chemin : ./Data/Image/Cerveau.png || Titre : Cerveau || Cout : 10 || Source : Pinterest || Numéro : 7 || Largeur : 373 || Hauteur : 507
Accès : L || Chemin : ./Data/Image/Rayures.png || Titre : Rayures || Cout : 80 || Source : Google image || Numéro : 8 || Largeur : 500 || Hauteur : 749
Accès : N || Chemin : ./Data/Image/Fille.png || Titre : Fille || Cout : 200 || Source : Pinterest || Numéro : 9 || Largeur : 563 || Hauteur : 361
Accès : N || Chemin : ./Data/Image/Formes.png || Titre : Formes || Cout : 100 || Source : Pinterest || Numéro : 10 || Largeur : 366 || Hauteur : 488
Accès : L || Chemin : ./Data/Image/Splash.png || Titre : Splash || Cout : 40 || Source : Google image || Numéro : 11 || Largeur : 842 || Hauteur : 595
Taille de la bibliothèque : 10

```

FIGURE 3.7 – Affichage des descripteurs après la suppression d'une image

On constate que la taille de la bibliothèque était initialement égale à 11. Après avoir supprimé une image, la taille de la bibliothèque a diminué à 10. Cela prouve que cette méthode a fonctionné efficacement.

II.3 Trier la bibliothèque

Nous avons organisé les données de notre bibliothèque en utilisant l'encapsulation des membres de données et en recourant à la classe vector de la bibliothèque standard C++. Ces données ont été trié en fonction de leur coût, critère du cahier des charges, et par un autre choix au bon vouloir des développeurs. Par conséquent, ils nous paraissaient intuitifs à implémenter un tri par ordre alphabétique. Ainsi, le menu triage se présente de la manière suivante :

```

Veuillez saisir votre choix!
0 Accès autorisé aux fonctionnalités d'un utilisateur de niveau 1
*****
***** Menu Principal de la gestion de bibliotheque ****
*****
1. Affichage
2. Mise a jour de la bibliotheque
3. Triage de la bibliotheque
4. Sous liste des descripteurs
5. Sauvegarder la bibliotheque
6. Application d'un traitement d'images
0. Retour au menu d identification
Veuillez saisir votre choix!
3
*****
3. TRIAGE DE LA BIBLIOTHEQUE
*****
1. Trier par ordre croissant
2. Trier par ordre alphabetique
0. Retour au menu precedent
Veuillez saisir votre choix!

```

FIGURE 3.8 – Menu du tri de la Bibliothèque

1. Trier le coût d'une image par ordre croissant

Nous avons utilisé le fichier "Descripteurs.csv" qui contient un total de 11 images. Nous avons trié ces images en fonction de leur coût, comme vous pouvez le voir ci-dessous :

```

PROBLEMES SORTIE CONSOLE DE DÉBOGAGE TERMINAL
474 || Hauteur : 632
Acces : L || Chemin :./Data/Image/Rayures.png || Titre : Rayures || Cout : 80 || Source : Google image || Numero : 8 || Largeur : 366 || Hauteur : 749
Acces : N || Chemin :./Data/Image/Formes.png || Titre : Formes || Cout : 100 || Source : Pinterest || Numero : 10 || Largeur : 366 || Hauteur : 488
Acces : N || Chemin :./Data/Image/Tigre.png || Titre : Tigre || Cout : 130 || Source : Google image || Numero : 6 || Largeur : 577 || Hauteur : 800
Acces : N || Chemin :./Data/Image/Verre.png || Titre : Verre || Cout : 180 || Source : Pinterest || Numero : 5 || Largeur : 495 || Hauteur : 742
Acces : N || Chemin :./Data/Image/Fille.png || Titre : Fille || Cout : 200 || Source : Pinterest || Numero : 9 || Largeur : 563 || Hauteur : 361
Acces : N || Chemin :./Data/Image/Loup.png || Titre : Loup || Cout : 350 || Source : Google || Numero : 2 || Largeur : 564 || Hauteur : 641
Acces : L || Chemin :./Data/Image/Pots.png || Titre : Pots || Cout : 470 || Source : Google image || Numero : 4 || Largeur : 876 || Hauteur : 634
Taille de la bibliotheque :11
Tapez sur la touche Entrée pour retourner au menu...

```

FIGURE 3.9 – Tri du coût par ordre croissant

En analysant les résultats ci-dessus, on peut constater que le coût des images a été trié de manière ascendante, allant de 80 euros à 470 euros.

2. Tri du titre d'une image par ordre alphabétique

Examinons maintenant le deuxième critère de tri que nous avons choisi, qui est le tri du titre de l'image par ordre alphabétique, les résultats sont présentés ci-dessous :

```

*****      3. TRIAGE DE LA BIBLIOTHEQUE      *****
1. Trier par ordre croissant
2. Trier par ordre alphabétique
0. Retour au menu précédent
Veuillez saisir votre choix!
2
Accès : A || Chemin :./Data/Image/Cerveau.png || Titre : Cerveau || Cout : 10 || Source : Pinterest || Numéro : 7 || Largeur : 373 || Hauteur : 507

Accès : N || Chemin :./Data/Image/Fille.png || Titre : Fille || Cout : 200 || Source : Pinterest || Numéro : 9 || Largeur : 563 || Hauteur : 361

Accès : N || Chemin :./Data/Image/Formes.png || Titre : Formes || Cout : 100 || Source : Pinterest || Numéro : 10 || Largeur : 366 || Hauteur : 481

Accès : N || Chemin :./Data/Image/Loup.png || Titre : Loup || Cout : 350 || Source : Google || Numéro : 2 || Largeur : 564 || Hauteur : 841

Accès : A || Chemin :./Data/Image/Pots.png || Titre : Pots || Cout : 470 || Source : Google image || Numéro : 4 || Largeur : 876 || Hauteur : 634

Accès : A || Chemin :./Data/Image/Rayures.png || Titre : Rayures || Cout : 80 || Source : Google image || Numéro : 8 || Largeur : 500 || Hauteur : 749

Accès : N || Chemin :./Data/Image/Route.png || Titre : Route || Cout : 0 || Source : Google image || Numéro : 1 || Largeur : 740 || Hauteur : 410

```

FIGURE 3.10 – Tri du titre d'une image par ordre alphabétique

En observant les résultats ci-dessus, nous pouvons constater que le tri des titres des images a été réussi, car on peut voir que les titres sont classés par ordre alphabétique, sans prendre en compte les numéros des images.

II.4 Sous-listes des descripteurs des images

Cette partie consiste à construire des sous-listes des descripteurs des images en fonction d'un critère lié à une caractéristique des images. Ce critère sera saisi au clavier par l'utilisateur. Le menu des sous-listes s'affiche comme suit :

```

*****      Menu Principal de la gestion de bibliothèque      *****
*****
1. Affichage
2. Mise à jour de la bibliothèque
3. Triage de la bibliothèque
4. Sous liste des descripteurs
5. Sauvegarder la bibliothèque
6. Application d'un traitement d'images
0. Retour au menu d identification
Veuillez saisir votre choix!
4
*****          4. SOUS LISTE DES DESCRIPTEURS          *****
1. Gratuit
2. Cout inferieur a 9.99 euros
3. Cout compris entre 10 et 99.99 euros
4. Cout superieur a 100 euros
5. Par source de l image
0. Retour au menu précédent
Veuillez saisir votre choix!

```

FIGURE 3.11 – Menu des sous-listes

Au départ, l'utilisateur peut spécifier un critère de sélection : la plage de coût. Il peut ensuite visualiser les sous-listes d'images correspondant à ce critère de coût pour faciliter la sélection des images qui répondent aux besoins et aux exigences de l'utilisateur. Pour le deuxième critère, nous avons choisi de proposer la source de l'image en tant que critère de choix car cela permet à l'utilisateur de vérifier l'origine et la fiabilité de ces images.

- 1. Construire et afficher une sous liste selon le coût de l'image**
- 2. Construire et afficher une sous liste selon le type de la source de l'image**

Les résultats de l'affichage des sous-listes en fonction des deux critères imposés sont présentés ci-dessous.

```

2. Mise a jour de la bibliothèque
3. Triage de la bibliothèque
4. Sous liste des descripteurs
5. Sauvegarder la bibliothèque
6. Application d'un traitement d'images
0. Retour au menu d'identification
Veuillez saisir votre choix!
*****
        4. SOUS LISTE DES DESCRIPTEURS
*****
1. Gratuit
2. Cout inferieur a 9.99 euros
3. Cout compris entre 10 et 99.99 euros
4. Cout supérieur a 100 euros
5. Par source de l'image
0. Retour au menu précédent
Veuillez saisir votre choix!
2
Access : N || Chemin : ./Data/Image/Route.png || Titre : Route || Cout : 0 || Source : Google image || Numero : 1 || Largeur : 740 || Hauteur : 410
Copiez sur la touche Entrée pour retourner au menu...

```

(a) Affichage d'une sous-liste selon le coût de l'image

```

1. Gratuit
2. Cout inferieur à 9.99 euros
3. Cout compris entre 10 et 99.99 euros
4. Cout supérieur a 100 euros
5. Par source de l image
0. Retour au menu précédent
Veuillez saisir votre choix!
3
Veuillez entrer une source
Pinterest
Access : N || Chemin : ./Data/Image/Cerveau.png || Titre : Cerveau || Cout : 10 || Source : Pinterest || Numero : 7 || Largeur : 563 || Hauteur : 373
Access : N || Chemin : ./Data/Image/Fille.png || Titre : Fille || Cout : 200 || Source : Pinterest || Numero : 9 || Largeur : 366 || Hauteur : 361
Access : N || Chemin : ./Data/Image/Formes.png || Titre : Formes || Cout : 100 || Source : Pinterest || Numero : 10 || Largeur : 488 || Hauteur : 488
Access : A || Chemin : ./Data/Image/Verre.png || Titre : Verre || Cout : 100 || Source : Pinterest || Numero : 5 || Largeur : 458 || Hauteur : 742
Access : A || Chemin : ./Data/Image/Ville.png || Titre : Ville || Cout : 50 || Source : Pinterest || Numero : 3 || Largeur : 474 || Hauteur : 632
Tapez sur la touche Entrée pour retourner au menu...

```

(b) Affichage de la sous-liste selon la source de l'image

FIGURE 3.12 – Sous-listes

II.5 Sauvegarder la bibliothèque et récupérer un fichier a partir d'une bibliothèque existante

La sauvegarde de la bibliothèque est un processus permettant de stocker les informations et les modifications apportées à la bibliothèque originale. Lorsque l'utilisateur est invité à enregistrer son traitement, il peut choisir de le faire en saisissant "Y" pour oui ou "N" pour non. Si l'utilisateur choisit "Y", il est ensuite invité à saisir un nom pour enregistrer l'image. Il est important de noter que seul un format d'image valide est accepté pour la sauvegarde. Enfin, l'utilisateur est invité à choisir un accès pour l'image en saisissant "A" pour accès autorisé à l'utilisateur de niveau 2 ou "N" pour accès autorisé à l'utilisateur de niveau 1.

```

6. Seuillage
7. Segmentation
8. Rehaussement de contours

0. Retour au précédent
Veuillez faire un choix de traitement
1
Vous voulez enregister votre traitement Y ou N
Y
Veuillez saisir un nom pour enregister l'image :
Loup_egal.png
Extension de l image valide
Quel Acces à donner à l'image (A ou N ) : 

```

FIGURE 3.13 – Sauvegarder la bibliothèque

II.6 Appliquer un traitement a l'image sélectionnée par l'utilisateur

La fonction "TRAITEMENTIMAGE()" créée dans la classe "Traitement" permet à l'utilisateur de sélectionner le type de traitement souhaité en saisissant le numéro correspondant. Cette fonction récupère également tous les traitements disponibles pour faciliter leur affichage.

```

Access : N || Chemin : ./Data/Image/Formes.png || Titre : Formes || Cout : 100 || Source : Pinterest || Numero : 10 || Largeur : 366 || Hauteur : 488
Access : L || Chemin : ./Data/Image/Splash.png || Titre : Splash || Cout : 40 || Source : Google image || Numero : 11 || Largeur : 842 || Hauteur : 595
Taille de la bibliothèque :11
Enter le rang de l'image sur laquelle vous voulez appliquer des traitements :
5
*****
        Menu de traitement des Images
*****
1. Histogramme
2. Gaußien
3. Moyenneur
4. Median
5. Sobel
6. Seuillage
7. Segmentation
8. Rehaussement de contours
0. Retour au précédent
Veuillez faire un choix de traitement

```

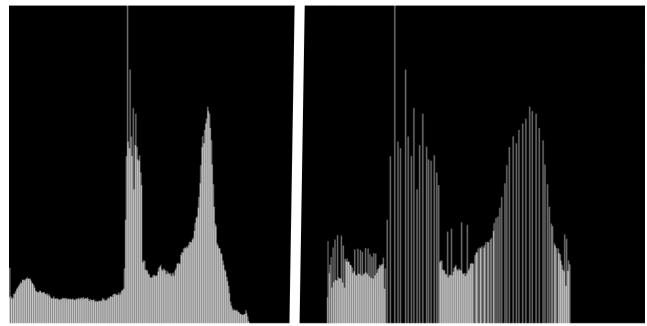
FIGURE 3.14 – Affichage du menu de traitement d'images

— Histogramme et Égalisation

Dans cette étape, nous avons calculé l'histogramme de l'image pour afficher la distribution des niveaux de gris. Ensuite, nous avons transformé cet histogramme pour qu'il soit uniformément réparti, ce qui a pour effet de rehausser le contraste de l'image affichée.



(a) Image "Loup"



(b) Histogramme et son égalisation

FIGURE 3.15 – Histogramme et Égalisation d'histogramme de l'image

— Filtre Médian

On utilise l'image originale "Fille" et on lui ajoute un bruit impulsionnel de type "poivre et sel" en remplaçant certains pixels par des valeurs de 0 ou 255.



FIGURE 3.16 – Image originale "Fille" et Image "Fille" bruitée

Nous utilisons ensuite un filtre médian pour supprimer le bruit présent dans l'image, car il est particulièrement efficace pour réduire ce type de bruit impulsionnel.



FIGURE 3.17 – Image filtrée

Après l'application du filtre médian, le bruit a été supprimé et la qualité visuelle de l'image s'est améliorée, mais cela a également entraîné une perte de détails importants dans l'image.

— Filtrage par convolution

1. Filtre Gaussien

L'utilisateur a la possibilité d'ajuster l'effet du filtre en modifiant la valeur de la variance. Plus cette valeur est élevée, plus l'effet du filtrage est prononcé. Il est à noter que cela se traduit par une suppression des textures et un flou des contours sur l'image filtrée.

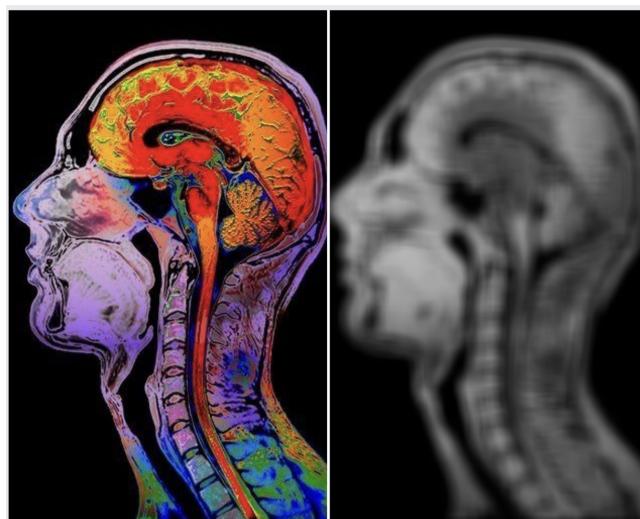


FIGURE 3.18 – Image originale "Cérébrale" et image filtrée

2. Filtre Moyenneur

En appliquant un filtre moyenneur sur l'image bruitée par un bruit gaussien additif, on observe un lissage global de l'image après l'application du filtre.



FIGURE 3.19 – Image originale "Loup" et image bruitée

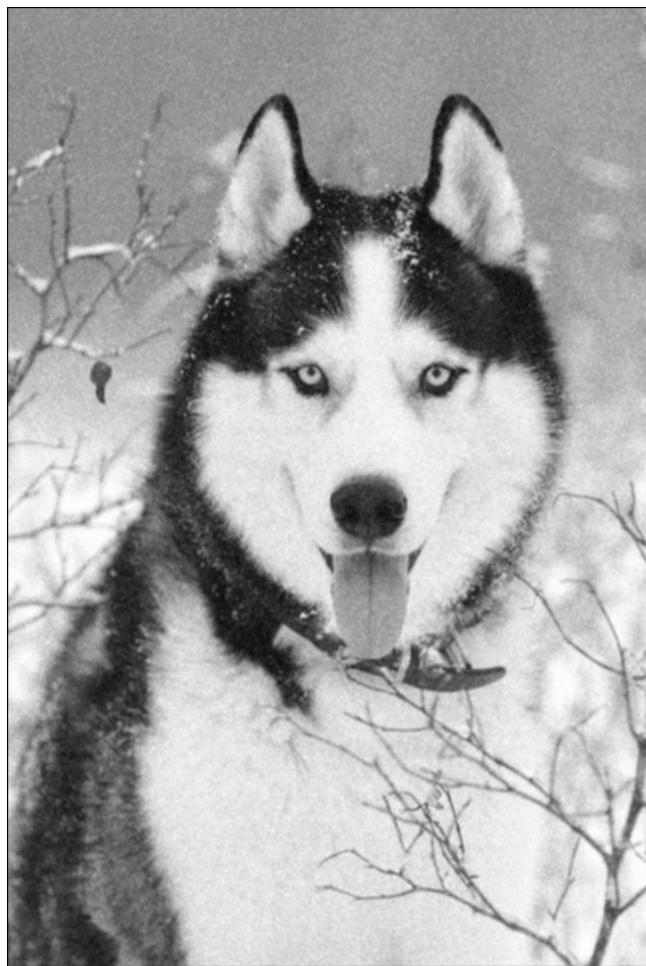


FIGURE 3.20 – Image filtrée

— Filtre Sobel

L'image "Splash" après avoir été filtre par le filtre de Sobel présente des contours nets et bien définis, mettant en évidence les différentes textures et structures présentes dans l'image.

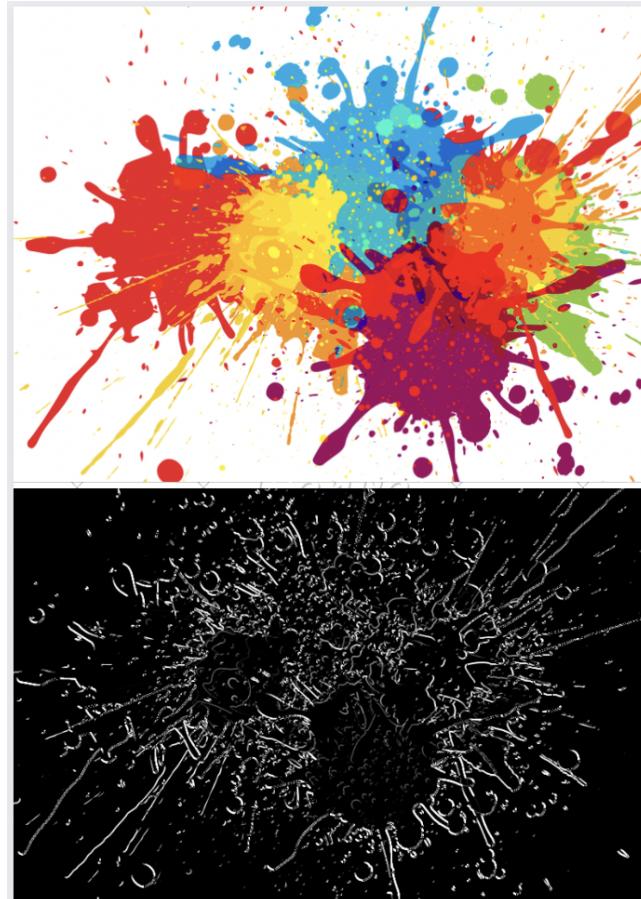


FIGURE 3.21 – Image originale "Splash" et image filtrée

— Rehaussement de contours

En utilisant le masque Laplacien, nous avons pu améliorer les contours de l'image. On peut constater que l'image est devenue plus nette, avec des contours plus définis et des détails plus marqués. Cependant, il est important de noter que l'utilisation excessive de cette technique peut entraîner un effet de sursaturation et un aspect artificiel de l'image. Il est donc important de trouver le bon équilibre pour obtenir des résultats les plus naturels possible. Il est donc important de trouver le bon équilibre pour obtenir des résultats les plus naturels possible.



(a) Image originale "Route"



(b) Image rehaussée

FIGURE 3.22 – Rehaussement de contours sur l'image "Route"

— Segmentation d'une image au niveaux de gris

L'image est binarisée en utilisant un seuil spécifié par l'utilisateur, les pixels ayant une intensité supérieure à ce seuil sont considérés comme appartenant à l'objet de l'image. L'image binarisée contient le résultat du seuillage.



FIGURE 3.23 – Image originale "Ville" et image seuillée

— Segmentation d'images couleurs

La segmentation couleur effectuée a permis d'attribuer chaque pixel à une plage de couleur spécifique, ce qui a conduit à la segmentation de l'image en sélectionnant une plage de couleur particulière.

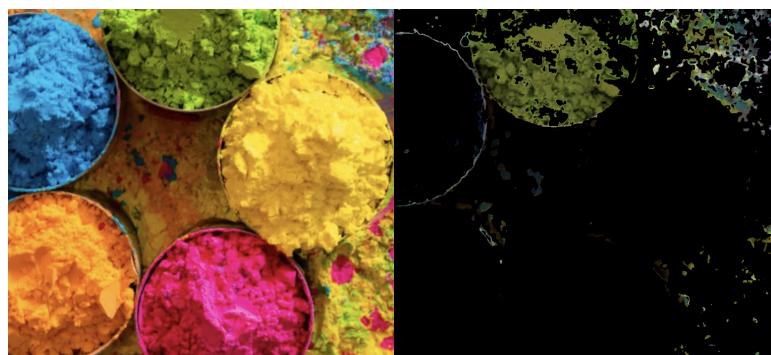


FIGURE 3.24 – Image originale "Pots" et image segmentée

III Fonctionnalités accessibles à l'utilisateur de niveau 2

Comme indiqué précédemment, l'utilisateur de niveau 2 a des droits d'accès limités en consultation, donc son menu d'affichage est comme suit.

```
Bienvenue et Félicitation !
Accès autorisé aux fonctionnalités d'un utilisateur de niveau 2

*****
***** Menu Principal de la gestion de bibliotheque *****
*****

1. Affichage de la liste des descripteurs
2. Affichage du cout d'une image particulière
0. Retour au menu précédent
```

FIGURE 3.25 – Menu et affichage pour l'utilisateur de niveau 2

Si l'utilisateur souhaite afficher le coût d'une image spécifique, son affichage sera :

```
*****
*****           Menu d'identification           *****
*****
bienvenue cher utilisateur
Veuillez entrer votre identifiant : LL22LL

Connexion réussie!

*****
*****           Choix des bibliothèques          *****
*****
Veuillez donner le nom d'une bibliothèque existante:
Descripteurs

Bienvenue et Félicitation !
Accès autorisé aux fonctionnalités d'un utilisateur de niveau 2

*****
*****   Menu Principal de la gestion de bibliotheque   *****
*****

1. Affichage de la liste des descripteurs
2. Affichage du cout d'une image particulière
0. Retour au menu précédent

Veuillez saisir votre choix!
2
Entrer numero d'image que tu veux afficher son prix : 4
Image 4 : 470

Tapez sur la touche Entrée pour retourner au menu...
```

FIGURE 3.26 – Affichage du coût d'une image pour l'utilisateur de niveau 2

CONCLUSION

Dans le but de conclure sur notre expérience, ce projet nous a apporté une amélioration au niveau de nos compétences techniques et au niveau relationnel. Nous avons appris à travailler en équipe sous des conditions différentes et face à plusieurs contraintes, ce qui nous a permis d'avoir un avant goût du monde professionnel. Toutefois, nous avons acquis d'avantage aptitudes dans les différentes unités de nos enseignements.

Notre projet de développement d'une application de gestion de bibliothèque d'images a été une expérience riche en enseignements. Nous avons pu mettre en pratique les différents aspects de la gestion de projet, tels que la gestion des fichiers techniques et l'utilisation des méthodes de traitement d'image en utilisant C++. Cependant, la gestion du temps a été un défi en raison des indisponibilités des membres du groupe en raison de nos différentes spécialités. Malgré cela, nous avons réussi à mener à bien ce projet et à atteindre nos objectifs. En outre, nous avons pu appréhender les différentes techniques liées à la programmation orientée objet et à la réalisation d'une application de traitement d'images. Nous avons pu tester les différents algorithmes de traitement d'images en utilisant OPENCV et les comparer avec d'autres environnements vus précédemment. Ce projet nous a également permis de nous connaître et de travailler ensemble en équipe, ce qui a été un dé expérience très enrichissante pour chacun d'entre nous.

BIBLIOGRAPHIE

- [1] _TSCHUMPERLE David et TILAMANT Christophe et BARRA Vincent_ **Le traitement numérique des images en c++** (www.editions-ellipses.fr), pp. 88-99, Février 2021.
- [2] _VASILIU Marius_ **Le langage C++** (www.pearsoneducation.fr), Juin 2005.
- [3] Mr EL AMINE Ali "cours", **C++ et GIT**, (<https://moodle.univ-tlse3.fr/course/view.php?id=8482>), Novembre 2022.

ANNEXES

```
//main.cpp
/**
 * @file main.cpp
 * @brief main
 * @version 0.1
 * @date 2023-01-28
 * @authors Lyakout TESSADA, Wissem SMATI, Manel TAIABI,
 * @authors Yasmine KARIM, Hugo ANDRE, Daniela TALBA
 * @copyright Copyright (c) 2023
 */
#include "../header/Bibliotheques.hpp"
#include "../header/traitement.hpp"

using namespace std;

using namespace cv; // A SUPPRIMER

const int User_niveau_1 = 1;
const int User_niveau_2 = 2;

void droits_user();
void menu_principale(int droits,Traitement biblio);
bool authentification(string userId);
void PourContinuer();
void menu_traitement();

void droits_user() {
    string userId;

    cout << endl << "*****" << endl
    << endl << "*****Menu d'identification*****" << endl
    << endl << "*****" << endl;

    cout << "bienvenue_cher_utilisateur" << endl
    << "Veuillez_entrer_votre_identifiant:" ;
    cin >> userId;
    if (authentification(userId)) {
        cout << endl << "Connexion_russie!" << endl;
```

```

        }
    else {
        cout << "Identifiant_incorrect" << endl;
        cout << "Accs_refus" << std::endl;
        PourContinuer();
        droits_user();
    }

vector<pair<string, int>> utilisateurs = {"LL11LL", User_niveau_1}, {"LL22LL", User_niveau_2};
for (auto &user : utilisateurs) {
    if (user.first == userId) {
        Traitement l_biblio;
        l_biblio.Lecture_descripteurs();

        menu_principale(user.second,l_biblio);
    }
}
}

bool authentification(string userId) {
    vector<string> autorisation_users = {"LL11LL", "LL22LL"};
    for (auto &autorisation : autorisation_users) {
        if (autorisation == userId) {

            return true;
        }
    }
    return false;
}

void PourContinuer(){
    cout << endl << "Tapez sur la touche Entrée pour retourner au menu..."<<endl;
    cin.ignore(std::numeric_limits<streamsize>::max(), '\n');
    cin.get();
}

void menu_traitement(){

    cout << endl << "*****Menu de traitement des Images*****" << endl;
    cout << endl << "*****1.Histogramme" << endl;
    cout << endl << "2.Gaussien" << endl;
    cout << endl << "3.Moyenneur" << endl;
    cout << endl << "4.Median" << endl;
    cout << endl << "5.Sobel" << endl;
    cout << endl << "6.Seuillage" << endl;
    cout << endl << "7.Segmentations" << endl;
    cout << endl << "8.Rehaussement de contours" << endl;
    cout << endl << "0.Retour au précédent" << endl;
}

void menu_principale(int droits,Traitement biblio) {

    int Choice;

    if (droits == User_niveau_1) {

```

```

cout << "Accès autorisé aux fonctionnalités d'un utilisateur de niveau 1" << endl;

while(1){

    string saisi;
    cout << endl << "*****Menu Principal de la gestion de bibliothèque*****" << endl;
    cout << endl << "*****Menu Principal de la gestion de bibliothèque*****" << endl;
    cout << endl << "*****Affichage" << endl;
    cout << endl << "2.Mise à jour de la bibliothèque" << endl;
    cout << endl << "3.Triage de la bibliothèque" << endl;
    cout << endl << "4.Sous liste des descripteurs" << endl;
    cout << endl << "5.Sauvegarder la bibliothèque" << endl;
    cout << endl << "6.Application d'un traitement d'images" << endl;
    cout << endl << "0.Retour au menu d'identification" << endl;
    cout << endl << "Veuillez saisir votre choix!" << endl;

    while(!(cin>>Choice)) {
        cout << "Veuillez entrer un numéro d'image valide:" ;
        cin.clear();
        cin.ignore(numeric_limits<streamsize>::max(), '\n');
    }
    switch(Choice){
        case 1:
            int Choice_aff;
            while(1){

                cout << endl << "*****1.AFFICHAGE*****" << endl;
                cout << endl << "1.Affichage de la liste des descripteurs" << endl;
                cout << endl << "2.Affichage du cout d'une image particulière" << endl;
                cout << endl << "0.Retour au menu précédent" << endl;
                cout << endl << "Veuillez saisir votre choix!" << endl;
                while(!(cin>>Choice_aff)) {
                    cout << "Veuillez entrer un numéro d'image valide:" ;
                    cin.clear();
                    cin.ignore(numeric_limits<streamsize>::max(), '\n');
                }

                switch (Choice_aff)
                {
                case 1 :
                    biblio.AfficherDescripteurs();
                    break;

                case 2 :
                    biblio.AfficherCout();
                    break;

                default:
                    menu_principale(droits,biblio);
                    break;
                }
                PourContinuer();
            }
            break;

        case 2:
            int Choice_maj;

```

```

while(1){
    cout << endl << "*****_A_JOUR_DE_LA_BIBLIOTHEQUE*****";
    cout << endl << "1._Supprimer_l'image" << endl;
    cout << endl << "2._Ajouter_une_image" << endl;
    cout << endl << "3._Modifier_une_ou_des_caracteristiques_d'une_image" << endl;
    cout << endl << "0._Retour_au_menu_precedent" << endl;
    cout << endl << "Veuillez_saisir_votre_choix!" << endl;
}

while(!(cin>>Choice_maj)) {
    cout << "Veuillez_entrer_un_numero_d'image_valide:_";
    cin.clear();
    cin.ignore(numeric_limits<streamsize>::max(), '\n');
}

switch (Choice_maj)
{

    case 1:
        biblio.Supprimer();
        biblio.AfficherDescripteurs();
        break;

    case 2:
        biblio.Ajouter_image();
        biblio.AfficherDescripteurs();
        break;

    case 3:
        biblio.ModifierDescripteur();
        break;

    default:
        menu_principale(droits,biblio);
        break;
}
PourContinuer();
}

break;

case 3:
int Choice_tri;
while(1){
    cout << endl << "*****_TRIAGE_LA_BIBLIOTHEQUE*****";
    cout << endl << "1._Trier_par_ordre_croissant" << endl;
    cout << endl << "2._Trier_par_ordre_alphaetique" << endl;
    cout << endl << "0._Retour_au_menu_precedent" << endl;
    cout << endl << "Veuillez_saisir_votre_choix!" << endl;

    while(!(cin>>Choice_tri)) {
        cout << "Veuillez_entrer_un_numero_d'image_valide:_";
        cin.clear();
        cin.ignore(numeric_limits<streamsize>::max(), '\n');
    }

    switch (Choice_tri)
    {

        case 1:

```

```

        biblio.Tri(Choice_tri);
        //biblio.AfficherDescripteurs();
        break;

    case 2:
        biblio.Tri(Choice_tri);
        //biblio.AfficherDescripteurs();

        break;

    default:
        menu_principale(droits,biblio);
        break;
    }
    PourContinuer();
}

break;

case 4:
    int Choice_lst;
    while(1){
        cout << endl << "*****4. SOUS_LISTE DES DESCRIPTEURS*****";
        cout << endl << "1. Gratuit" << endl;
        cout << endl << "2. Cout inferieur a 9.99 euros" << endl;
        cout << endl << "3. Cout compris entre 10 et 99.99 euros" << endl;
        cout << endl << "4. Cout superieur a 100 euros" << endl;
        cout << endl << "5. Par source de l'image" << endl;
        cout << endl << "0. Retour au menu precedent" << endl;
        cout << endl << "Veuillez saisir votre choix!" << endl;

        while(!(cin>>Choice_lst)) {
            cout << "Veuillez entrer un numero d'image valide:" ;
            cin.clear();
            cin.ignore(numeric_limits<streamsize>::max(), '\n');
        }

        switch (Choice_lst)
        {
            case 1:
                biblio.Sous_liste(Choice_lst);

                break;

            case 2:
                biblio.Sous_liste(Choice_lst);

                break;

            case 3:
                biblio.Sous_liste(Choice_lst);

                break;

            case 4:
                biblio.Sous_liste(Choice_lst);

                break;

            case 5:

```

```

        biblio.Sous_liste_autre();

        break;

    default:
        menu_principale(droits,biblio);
        break;
    }
    PourContinuer();
}
break;

case 5:
int svg;
while(1){
    cout << endl << "*****5. SAUVEGARDER LA BIBLIOTHEQUE*****";
    cout << endl << "1. Sauvegarder les modifications apportees sur la biblioth";
    cout << endl << "0. Retour au menu precedent" << endl;
    cout << endl << "Veuillez saisir votre choix!" << endl;

    while(!(cin>>svg)) {
        cout << "Veuillez entrer un numero d'image valide:" ;
        cin.clear();
        cin.ignore(numeric_limits<streamsize>::max(), '\n');
    }

    switch (svg)
    {
        case 1:
            biblio.Bibliotheque::Sauvegarder();
            break;

        default:
            menu_principale(droits,biblio);
            break;
    }
    PourContinuer();
}
break;

case 6:
int Choice_app;
while(1){
    cout << endl << "*****6. APPLICATION D'UN TRAITEMENT D'IMAGE*****";
    cout << endl << "1. Application d'un traitement d'images" << endl;
    cout << endl << "0. Retour au menu precedent" << endl;
    cout << endl << "Veuillez saisir votre choix!" << endl;

    while(!(cin>>Choice_app)) {
        cout << "Veuillez entrer un numero d'image valide:" ;
        cin.clear();
        cin.ignore(numeric_limits<streamsize>::max(), '\n');
    }

    switch (Choice_app)
    {
        case 1:

            biblio.AfficherDescripteurs();

```

```

        biblio.Selection_Image(); //Obligatoire
        menu_traitement();
        biblio.TraitemenImage(); //Obligatoire

        break;

    default:
        menu_principale(droits,biblio);
        break;

    }

    cout << endl << "Tapez sur la touche Entrée pour revenir au menu d'appl
    cin.get();

}

break;

default:
    cout << endl << "Retour au menu précédent" << endl;
    droits_user();
    break;

}

PourContinuer();
}

} else if (droits == User_niveau_2) {

    cout << "Accès autorisé aux fonctionnalités d'un utilisateur de niveau 2" << endl<<endl;
    while(1){
        cout << endl << "*****Menu Principal de la gestion de bibliothèque*****" << endl;
        cout << endl << "*****Affichage de la liste des descripteurs" << endl;
        cout << endl << "*****Affichage du cout d'une image particulière" << endl;
        cout << endl << "*****Retour au menu précédent" << endl;
        cout << endl << "Veuillez saisir votre choix!" << endl;
        cin>>Choice;

        switch(Choice){
            case 1:
                biblio.AfficherDescripteurs();

                break;

            case 2:
                biblio.AfficherCout();
                break;

            default :
                droits_user();
                break;

        }
        PourContinuer();
    }
}

```

```

        }

    }

int main (void)
{
    droits_user();
    //menu_principale();

    //Gestion Bibliotheque
    //Traitement biblio;
    //biblio.Lecture_descripteurs(); //Obligatoire
    //biblio.AfficherDescripteurs();
    //biblio.Ajouter_image();
    //biblio.AfficherCout();
    //biblio.Tri();
    //biblio.Supprimer();
    //biblio.ModifierDescripteur();
    //biblio.AfficherDescripteurs();
    //ligne de commabe
    //biblio.Selection_Image(); //Obligatoire
    //biblio.TraitementImage(); //Obligatoire
    //biblio.Sous_liste();
    //biblio.Sous_liste_autre();
    //biblio.Sauvegarder();

    //A SUPPRIMER COMMANDE SUIVANTE
/* Traitement biblio;
   Mat Image_originale;
   Mat Image_originale_gray;
   Mat Image_bruit;
   Mat gaussine_noise;
   Mat Img_rslt;
   Image_originale = imread("./Data/Image/Fille.png",IMREAD_COLOR);
   gaussine_noise = Image_originale.clone();
   randn(gaussine_noise,18,30);
   Image_bruit = gaussine_noise + Image_originale;
   Mat saltpepper_noise = Mat::zeros(Image_originale.rows, Image_originale.cols,CV_8U);
   randu(saltpepper_noise,0,255);

   Mat black = saltpepper_noise < 30;
   Mat white = saltpepper_noise > 225;

   Mat saltpepper_img = Image_originale.clone();
   saltpepper_img.setTo(255,white);
   saltpepper_img.setTo(0,black);
   cvtColor(saltpepper_img,Image_originale_gray, CV_BGR2GRAY);
   cvtColor(Image_bruit,Image_originale_gray, CV_BGR2GRAY);
   imwrite("./Data/Image/Fille_random_noise.png", Image_originale_gray); */

    return 0;
}

```

```
//////////BIBLIOTHEQUE|||||||
```

```
#include "../header/Bibliotheques.hpp"

Bibliotheque::Bibliotheque(){
    string l_file_csv ; // nom du fichier
    bool l_exist; // Verification de l'existence du fichier

    string l_name_file;
    string l_path;
    do{
        // Saisie du nom du nouveau fichier
        cout << endl << "*****Choix_des_bibliotheques*****" << endl<<endl;
        cout << "Veuillez donner le nom d'une bibliotheque existante:" << endl ;
        cin >> l_file_csv ;

        // Vrification de la prsence de l'extension csv
        l_name_file = Verification_Extensioncsv(l_file_csv);
        l_path = "./Data/Bibliotheque/" +l_name_file;

        l_exist = experimental::filesystem::exists(l_path) ;
    }while (l_exist ==false); // Vrification ou non de l'existance du nom

    cout<<endl<<"Bienvenue et Flicitation!" << endl;

    m_chemincsv = l_path;
}

string Bibliotheque::Verification_Extensioncsv(string a_Name_file){
    string l_file_extension = ".csv";
    string l_name_file;
    //Rechercher la position de l'extension ".csv" dans le titre du fichier
    string::size_type idx = a_Name_file.rfind(l_file_extension);

    if (idx != std::string::npos) {
        l_name_file = a_Name_file;
    }
    else {
        //ajout de lextension
        l_name_file = a_Name_file+".csv";
    }
    return l_name_file;
}

void Bibliotheque::Lecture_descripteurs(){

    string ligne;
    ifstream fichier_csv(m_chemincsv);
```

```

if (!fichier_csv.good()) {
    cerr << "Erreur lors de l'ouverture du fichier" << m_chemincsv << endl;
}

//Ignorer l'entete
getline(fichier_csv, ligne);

while (getline(fichier_csv, ligne)) {

    stringstream ss(ligne);
    //Descripteurs
    string a_acces,a_chemin,a_titre,a_source;
    string a_numero, a_largeur,a_hauteur;
    string a_cout;

    getline(ss, a_acces, ',');
    getline(ss, a_chemin, ',');
    getline(ss, a_titre, ',');
    getline(ss, a_cout, ',');
    getline(ss, a_source, ',');
    getline(ss, a_numero, ',');
    getline(ss, a_largeur, ',');
    getline(ss, a_hauteur);

    Descripteurs img;

    img.setAcces(a_acces);
    img.setChemin(a_chemin);
    img.setTitre(a_titre);
    img.setCout(a_cout);
    img.setSource(a_source);
    img.setNumero(a_numero);
    img.setLargeur(a_largeur);
    img.setHauteur(a_hauteur);

    //On le met dans un vecteur
    bibliotheque.push_back(img);
}

void Bibliotheque::AfficherDescripteurs() {
    //cout << "Afficher les descripteurs des images :" << endl;

    for (auto& img : bibliotheque) {
        cout<<endl << "Acces:" <<img.getAcces()
            << endl << "Chemin:" <<img.getChemin()
            << endl << "Titre:" <<img.getTitre()
            << endl << "Cout:" <<img.getCout()
            << endl << "Source:" <<img.getSource()
            << endl << "Numero:" <<img.getNumero()
            << endl << "Largeur:" <<img.getLargeur()
            << endl << "Hauteur:" <<img.getHauteur()
            << endl<<endl;
    }
    cout << "Taille de la bibliotheque:" << bibliotheque.size()<<endl;
}

```

```

void Bibliotheque::AfficherDescripteurs(int const a_num){

    cout << "Acces:" << bibliotheque[a_num-1].getAcces()
        << endl << "Chemin:" << bibliotheque[a_num-1].getChemin()
        << endl << "Titre:" << bibliotheque[a_num-1].getTitre()
        << endl << "Cout:" << bibliotheque[a_num-1].getCout()
        << endl << "Source:" << bibliotheque[a_num-1].getSource()
        << endl << "Numero:" << bibliotheque[a_num-1].getNumero()
        << endl << "Largeur:" << bibliotheque[a_num-1].getLargeur()
        << endl << "Hauteur:" << bibliotheque[a_num-1].getHauteur()
        << endl << endl;

}

void Bibliotheque::Ajouter_image(){

    string a_acces,a_chemin,a_titre,a_source;
    string a_numero;
    string a_cout,a_largeur,a_hauteur;
    int a_cout_tmp,a_largeur_tmp,a_hauteur_tmp;

    string l_path;
    bool isValid_acces = false;
    bool isValid_titre = false;

    string l_last_nombre_img; // numero de la dernier image
    int l_tmp; // numero de l'image en int

    Descripteurs img;
    //Acces
    /* "isValid_acces" qui est initialis false. La boucle while va continuer
       tant que la saisie n'est pas valide (isValid_acces = false).
       Si la saisie est valide, isValid est mis true et la boucle s'arrete.
    */
    while (!isValid_acces) {
        cout << "Quel Acces donner l'image (A ou N):";
        cin >> a_acces;
        cin.clear();
        cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
        if (a_acces == "A" || a_acces == "N") {
            isValid_acces = true;
        } else {
            std::cout << "Acces non valide. Veuillez saisir A ou N." << endl;
        }
    }

    //Chemin
    while (!isValid_titre) {

        cout << "Veuillez saisir le nom de l'image a rajouter" << endl;
        getline(cin,a_titre);

        l_path = "./Data/Image/" + a_titre + ".png";

        isValid_titre = experimental::filesystem::exists(l_path);
    }

    a_chemin = l_path;

    //Cout
}

```

```

cout<<"Entrez la valeur du cout de l'image : "<<endl;
while(!(cin>>a_cout_tmp)) {
    cout << "Veuillez une valeur de cout valide : ";
    cin.clear();
    cin.ignore(numeric_limits<streamsize>::max(), '\n');
}
a_cout = to_string(a_cout_tmp);

//Source
cin.ignore(numeric_limits<streamsize>::max(), '\n');
cout<<"Entrez la source de l'image : "<<endl;
getline(cin,a_source);

//Numero
l_last_nombre_img = bibliotheque.back().getNumero();
l_tmp = atoi(l_last_nombre_img.c_str())+1;
a_numero = to_string(l_tmp);

//Largeur
cout <<"Entrez la largeur de l'image : "<<endl;
while(!(cin>>a_largeur_tmp)) {
    cout << "Veuillez une valeur de largeur valide : ";
    cin.clear();
    cin.ignore(numeric_limits<streamsize>::max(), '\n');
}
a_largeur = to_string(a_largeur_tmp);

//Hauteur
cout <<"Entrez la hauteur de l'image : "<<endl;
while(!(cin>>a_hauteur_tmp)) {
    cout << "Veuillez une valeur de hauteur valide : ";
    cin.clear();
    cin.ignore(numeric_limits<streamsize>::max(), '\n');
}
a_hauteur = to_string(a_hauteur_tmp);

cin.ignore(numeric_limits<streamsize>::max(), '\n');

img.setAcces(a_acces);
img.setChemin(a_chemin);
img.setTitre(a_titre);
img.setCout(a_cout);
img.setSource(a_source);
img.setNumero(a_numero);
img.setLargeur(a_largeur);
img.setHauteur(a_hauteur);
bibliotheque.push_back(img);

}

void Bibliotheque::Tri(int choix) {

int l_size_biblio;
l_size_biblio = int(bibliotheque.size());
switch (choix)
{
case 1 :
    sort(bibliotheque.begin(), bibliotheque.end(), [] (const Descripteurs& a, const Descripteurs&
    return stoi(a.getCout()) < stoi(b.getCout());
});
}

```

```

// rcrire les lignes triees
for (const auto& row : bibliotheque) {
    row.getAcces();row.getTitre();row.getChemin();
    row.getSource();row.getCout();row.getHauteur();
    row.getLargeur();row.getNumero();
}

for(int i = 1; i <=l_size_biblio;i++){
    bibliotheque[i-1].setNumero(to_string(i));
}
break;
case 2:
    // trier les lignes de fichier par Trie
    std::sort(bibliotheque.begin(), bibliotheque.end(), [](const Descripteurs& a, const Descripteurs
        return a.getTitre() < b.getTitre();
    ));

// rcrire les lignes triees
for (const auto& row : bibliotheque) {
    row.getAcces();row.getTitre();row.getChemin();
    row.getSource();row.getCout();row.getHauteur();
    row.getLargeur();row.getNumero();
}

for(int i = 1; i <=l_size_biblio;i++){
    bibliotheque[i-1].setNumero(to_string(i));
}
break;
default:
    break;
}

AfficherDescripteurs();

}

void Bibliotheque:: Sous_liste(int prix){

vector<Descripteurs> l_sub_images;

if(prix == 1 ){
    copy_if(bibliotheque.begin(), bibliotheque.end(), back_inserter(l_sub_images), [](const Descript
    return img.getCout() == "0";});

}

if(prix == 2 ){
    copy_if(bibliotheque.begin(), bibliotheque.end(), back_inserter(l_sub_images), [](const Descript
    return atoi(img.getCout().c_str()) <= 9.99;});

}

if(prix==3){
    copy_if(bibliotheque.begin(), bibliotheque.end(), back_inserter(l_sub_images), [](const Descript
    return atoi(img.getCout().c_str()) <= 99.99 && atoi(img.getCout().c_str()) > 10;});
}

```

```

}

if(prix == 4) {
    copy_if(bibliotheque.begin(), bibliotheque.end(), back_inserter(l_sub_images), [](const Descripteur & img) { return atoi(img.getCout().c_str()) > 100; });
}

for (auto& img : l_sub_images) {
    cout<<endl << "Acces:" << img.getAcces()
        << "Chemin:" << img.getChemin()
        << "Titre:" << img.getTitre()
        << "Cout:" << img.getCout()
        << "Source:" << img.getSource()
        << "Numero:" << img.getNumero()
        << "Largeur:" << img.getLargeur()
        << "Hauteur:" << img.getHauteur()
        << endl<<endl;
}

}

void Bibliotheque:: Sous_liste_autre(){
    string l_source;
    Descripteurs descripteur_sub_images;
    vector<Descripteurs> l_sub_images;

    cin.ignore(numeric_limits<streamsize>::max(), '\n');
    cout<<"Veuillez entrer une source"<<endl;
    getline(cin,l_source);
    for (auto & img : bibliotheque) {
        //cout << "Image " << i + 1 << " : " << bibliotheque[i].getSource() << endl;

        if(img.getSource() == l_source)
        {
            descripteur_sub_images.setAcces(img.getAcces());
            descripteur_sub_images.setChemin(img.getChemin());
            descripteur_sub_images.setTitre(img.getTitre());
            descripteur_sub_images.setCout(img.getCout());
            descripteur_sub_images.setSource(img.getSource());
            descripteur_sub_images.setNumero(img.getNumero());
            descripteur_sub_images.setLargeur(img.getLargeur());
            descripteur_sub_images.setHauteur(img.getHauteur());
            l_sub_images.push_back(descripteur_sub_images);
        }
    }

    for (auto& img : l_sub_images) {
        cout<<endl << "Acces:" << img.getAcces()
            << "Chemin:" << img.getChemin()
            << "Titre:" << img.getTitre()
            << "Cout:" << img.getCout()
            << "Source:" << img.getSource()
            << "Numero:" << img.getNumero()
            << "Largeur:" << img.getLargeur()
            << "Hauteur:" << img.getHauteur()
            << endl<<endl;
    }
}

```

```

}

void Bibliotheque::Sauvegarder(){
    string l_file_csv ; // nom du fichier
    bool l_exist; // Verification de l'existence du fichier

    string l_name_file;
    string l_path;

    cout << "Veuillez saisir un nom pour enregister vos modifications" << endl ;
    cin >> l_file_csv ;

    // Vrification de la prsence de l'extension csv
    l_name_file = Verification_Extensioncsv(l_file_csv);
    l_path = "./Data/Bibliotheque/" +l_name_file;

    l_exist = experimental::filesystem::exists(l_path) ;

    if(l_exist){
        //e

        /* "isValid_reponse" qui est initialis false. La boucle while va continuer
        tant que la saisie n'est pas valide (isValid_reponse = false).
        Si la saisie est valide, isValid_reponse est mis true et la boucle s'arrte.
        */
        bool isValid_reponse = false;
        string l_reponse; //Rponse au question de confirmation Y ou N pos
        while (!isValid_reponse) {
            cout << l_name_file << "existe deja. Voulez-vous le remplacez ? Y ou N" << endl ;
            cin >> l_reponse;
            cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
            if (l_reponse == "Y" || l_reponse == "N") {
                isValid_reponse = true;
            }
            else {
                std::cout << "Acces non valide. Veuillez saisir Y ou N." << endl;
                cin.clear();
            }
        }
        if(l_reponse == "N"){
            //Nous revenous au dbut de la methode et on recommence la procedure
            Sauvegarder();
        }
        else{
            Ecriture_descripteurs(l_path);
            cout <<l_name_file<<"atcras"<<endl;
        }
    }

    else{
        Ecriture_descripteurs(l_path);
        m_chemincsv = l_path;
    }
}

void Bibliotheque::Ecriture_descripteurs(string const a_path){


```

```

ofstream l_file_csv;

//Permet d'écrire dans une bibliothèque existante ou bibliothèque nouvelle
l_file_csv.open(a_path);

//Première ligne du fichier csv qui correspond au descripteur
l_file_csv<<"Acces,Chemin,Titre,Cout,Source,Numero,Largeur,Hauteur";

for(auto& row : bibliotheque){

    l_file_csv << endl << row.getAcces() << "," << row.getChemin() << "," << row.getTitre() << "," 
    << row.getCout() << "," << row.getSource() << "," << row.getNumero() << ","
    << row.getLargeur() << "," << row.getHauteur();
}

l_file_csv.close();

}

void Bibliotheque::AfficherCout(){
    // demander le numero d'image que l'utilisateur il veut afficher son prix
    int l_Numero_ligne;
    int l_size_biblio;

    cout << "Entrer le numero d'image que tu veux afficher son prix : ";

    l_size_biblio = int(bibliotheque.size());
    while(!(cin>>l_Numero_ligne)|| (l_Numero_ligne<=0)|| (l_Numero_ligne>l_size_biblio)) {
        cout << "Veuillez entrer un numero d'image valide : ";
        cin.clear();
        cin.ignore(numeric_limits<streamsize>::max(), '\n');
    }

    cout << "Image " << l_Numero_ligne << " : " << bibliotheque[l_Numero_ligne-1].getCout()
}

void Bibliotheque::Supprimer() {
    // demander à l'utilisateur le numero de ligne qu'il vaut supprimer
    int l_Numero_ligne;
    int l_size_biblio;
    vector <Descripteurs> test ;

    l_size_biblio = int(bibliotheque.size());
    cout << "Veuillez entrer le numero à supprimer" << endl;

    while(!(cin>>l_Numero_ligne)|| (l_Numero_ligne<=0)|| (l_Numero_ligne>l_size_biblio)) {
        cout << "Veuillez entrer un numero d'image valide : ";
        cin.clear();
        cin.ignore(numeric_limits<streamsize>::max(), '\n');
    }

    // supprimer la ligne

    bibliotheque.erase(bibliotheque.begin() + l_Numero_ligne - 1);
    // Mise jour des numero
    for(int i = l_Numero_ligne; i <=l_size_biblio;i++){
        bibliotheque[i-1].setNumero(to_string(i));
    }
}

```

```

void Bibliotheque::ModifierDescripteur() {
    int l_Numero_ligne;
    int l_size_biblio;

    l_size_biblio = bibliotheque.size();

    cout<< endl << "Entrer le numero d'image que tu veux modifier : ";
    while(!(cin>>l_Numero_ligne)||(l_Numero_ligne<=0)|| (l_Numero_ligne>l_size_biblio)) {
        cout << "Veuillez entrer un numero d'image valide : ";
        cin.clear();
        cin.ignore(numeric_limits<streamsize>::max(), '\n');
    }

    bool isValid_reponse;
    string l_reponse; //Rponse au question de confirmation Y ou N pos

    string l_nouveux_titre;
    string l_nouveux_Source;
    string l_nouveux_acces;
    string l_nouveux_cout;
    string l_nouveux_numero;
    string l_nouveux_largeur;
    string l_nouveux_hauteur;

    Descripteurs selected_row = bibliotheque.at(l_Numero_ligne - 1);
    cin.ignore(numeric_limits<streamsize>::max(), '\n');
    do{
        cout << "Enter un titre : ";
        getline(cin,l_nouveux_titre);
        bibliotheque[l_Numero_ligne - 1].setTitre(l_nouveux_titre);

        cout << "Enter une Source : ";
        getline(cin,l_nouveux_Source);
        bibliotheque[l_Numero_ligne - 1].setSource(l_nouveux_Source);

        cout << "Enter un Acces : ";
        getline(cin,l_nouveux_acces);
        bibliotheque[l_Numero_ligne - 1].setAcces(l_nouveux_acces);

        cout << "Enter un cout : ";
        getline(cin,l_nouveux_cout);
        bibliotheque[l_Numero_ligne - 1].setCout(l_nouveux_cout);

        cout << "Enter un Numero : ";
        getline(cin,l_nouveux_numero);
        bibliotheque[l_Numero_ligne - 1].setNumero(l_nouveux_numero);

        cout << "Enter la Largeur d'image : ";
        getline(cin,l_nouveux_largeur);
        bibliotheque[l_Numero_ligne - 1].setLargeur(l_nouveux_largeur);

        cout << "Enter la Hauteur d'image : ";
        getline(cin,l_nouveux_hauteur);
        bibliotheque[l_Numero_ligne - 1].setHauteur(l_nouveux_hauteur);

    AfficherDescripteurs(l_Numero_ligne);
}

```

```

isValid_reponse = false;

while (!isValid_reponse) {
    cout << "Etes-vous satisfait de votre modification? Y ou N?" << endl ;
    cin >> l_reponse;
    cin.ignore(numeric_limits<streamsize>::max(), '\n');
    if (l_reponse == "Y" || l_reponse == "N") {
        isValid_reponse = true;
    }
    else {
        cout << "Acces non valide. Veuillez saisir Y ou N." << endl;
        cin.clear();
    }
}

if(l_reponse == "N"){
    //Nous revenons au dbut de la methode et on recommence la procedure
    isValid_reponse = false;
}
}while(!isValid_reponse);

}

void Bibliotheque::Selection_Image() {
    int l_Numero_ligne;
    int l_size_biblio;
    cout << endl << "Enter le rang de l'image sur laquelle vous voulez appliquer des traitements:" << endl;
    l_size_biblio = int(bibliotheque.size());

    while (!(cin >> l_Numero_ligne) || (l_Numero_ligne <= 0) || (l_Numero_ligne > l_size_biblio)) {
        cout << "Veuillez entrer un numero d'image valide:" ;
        cin.clear();
        cin.ignore(numeric_limits<streamsize>::max(), '\n');
    }
    p_choice_Image = l_Numero_ligne;
}

```



```

    cin.ignore(numeric_limits<streamsize>::max(), '\n');
}

switch (l_choixTraitement){
    case 1:
        /* Histogramme */
        Egalisation_histogramme();
        Sauvegarder();
        break;
    case 2:
        /* Gaussien */
        l_Image_Resultat = Gaussien();
        AfficherContenuImage(l_Image_Resultat);
        Sauvegarder();
        break;

    case 3:
        /* Moyenneur */
        l_Image_Resultat = Moyenneur();
        AfficherContenuImage(l_Image_Resultat);
        Sauvegarder();
        break;
    case 4:
        /* Median */
        l_Image_Resultat = Median();
        AfficherContenuImage(l_Image_Resultat);
        Sauvegarder();
        break;

    case 5:
        /* Sobel */
        l_Image_Resultat = Sobel();
        AfficherContenuImage(l_Image_Resultat);
        Sauvegarder();
        break;
    case 6:
        /* Seuillage simple */
        l_Image_Resultat = Seuillage_simple();
        AfficherContenuImage(l_Image_Resultat);
        Sauvegarder();
        break;
    case 7:
        /* Segmentation */
        l_Image_Resultat = Segmentation();
        AfficherContenuImage(l_Image_Resultat);
        Sauvegarder();
        break;

    case 8:
        /* Rehaussement de Contours */
        l_Image_Resultat = RehaussementContour();
        AfficherContenuImage(l_Image_Resultat);
        Sauvegarder();
        break;

    default:
        cin.get();
        break;
}

```

}

```

// Verifier la saturation
int Traitement::VerifierSaturation(const int valeur){
    if(valeur > 255){
        return 255 ;
    }else if(valeur < 0){
        return 0 ;
    }else{
        return valeur ;
    }
}

// Gnr les filtres Moyenneur; gradient en x et y de Sobel
Mat Traitement::Filtre(const int typeFiltre){
    switch (typeFiltre){
        // Filtre moyenneur
        case 1 :
            return ((Mat_<double>(3,3) << 1, 1, 1, 1, 1, 1, 1, 1, 1)/9) ;
            break ;

        // Filtre gradient en x (Sobel)
        case 2 :
            return (Mat_<double>(3,3) << -1, -2, -1, 0, 0, 0, 1, 2, 1) ;
            break ;
        // Filtre gradient en y (Sobel)
        case 3 :
            return (Mat_<double>(3,3) << -1, 0, 1, -2, 0, 2, -1, 0, 1) ;
            break ;

        //Filtre Laplacien
        case 4 :
            return (Mat_<double>(3,3) << 0, 1, 0, 1, -4, 1, 0, 1, 0) ;
            break ;

        default:
            return (Mat_<double>(3,3) << 0, 0, 0, 0, 1, 0, 0, 0, 0) ;
            break;
    }
}

// Convolution entre deux matrice
Mat Traitement::MatriceConvolution(const Mat Image_In, const Mat H){
    // Declaration des variables
    int ligne, colonne; // Indices
    int L_image, C_image ; // Dimensions de l'image
    int val ; // Variable intermediaire
    Mat Image_Out(Image_In.rows,Image_In.cols, CV_8U,double(0)) ; // Image resultante

    // Dimensions de l'image
    L_image = Image_In.rows ; // Nombre de lignes
    C_image = Image_In.cols ; // Nombre de colonnes

    // Produit de convolution
    for(ligne = 1 ; ligne < L_image; ligne++){ // Pour chaque ligne de l'image
        for(colonne = 1 ; colonne < C_image ; colonne++){ // Pour chaque colonne de l'image
            val = Image_In.at<unsigned char>(ligne-1,colonne-1) * H.at<double>(0,0) + Image_In.at<unsigned char>(ligne-1,colonne+1) * H.at<double>(2,0) + Image_In.at<unsigned char>(ligne,colonne) * H.at<double>(1,1) + Image_In.at<unsigned char>(ligne+1,colonne-1) * H.at<double>(0,2) + Image_In.at<unsigned char>(ligne+1,colonne+1) * H.at<double>(1,2) + Image_In.at<unsigned char>(ligne+2,colonne-1) * H.at<double>(2,1) + Image_In.at<unsigned char>(ligne+2,colonne+1) * H.at<double>(2,2) ;
            Image_Out.at<unsigned char>(ligne,colonne) = val ;
        }
    }
}

```

```

        + Image_In.at<unsigned char>(ligne+1,colonne+1) * H.at<double>(2,2);
        Image_Out.at<unsigned char>(ligne, colonne) = VerifierSaturation(val) ;

    }
}

// Retour
return Image_Out ;
}

//Calcul la norme de la matrice
Mat Traitement::norme_matriciel (const Mat Image, const Mat Image2){
    Mat Img_Out (Image.size(), CV_8U) ; // Image de resultat
    double X2,Y2; //x et Y

    int ligne, colonne ; // indice des lignes et des colonnes
    int L_image, C_image ; // Dimensions de l'image
    // Dimensions de l'image
    L_image = Image.rows ; // Nombre de lignes
    C_image = Image.cols ; // Nombre de colonnes

    for(ligne = 0; ligne < L_image; ligne ++){
        for(colonne = 0; colonne<C_image; colonne++){
            //Calcul du produit de chaque pixel
            X2 = Image.at<unsigned char>(ligne,colonne) * Image.at<unsigned char>(ligne,colonne);
            Y2 = Image2.at<unsigned char>(ligne,colonne) * Image2.at<unsigned char>(ligne,colonne);

            // Calcul de la norme
            Img_Out.at<unsigned char>(ligne,colonne)= sqrt(X2 + Y2);
        }
    }
    return Img_Out;
}

// Seuillage simple
Mat Traitement::Seuillage_simple(){

    const Mat l_Image_In = getImage();
    Mat l_grayMat;

    //Initialisation de la matrice resultat
    Mat l_Image_Out(l_Image_In.size(), CV_8U, double(0));

    int l_ligne,l_colonne;

    int l_seuil;
    cout << "Veuillez entrer la valeur du seuil" << endl;
    cin >> l_seuil;

    // Conversion en niveau de gris de l'image d'origine
    cvtColor(l_Image_In,l_grayMat, CV_BGR2GRAY);

    for (l_ligne = 0; l_ligne< l_grayMat.rows ; l_ligne++) {
        for(l_colonne = 0; l_colonne < l_grayMat.cols; l_colonne++){
            if(l_grayMat.at<unsigned char>(l_ligne,l_colonne) > l_seuil){
                l_Image_Out.at<unsigned char>(l_ligne,l_colonne) = 255;
            }
            else{
                l_Image_Out.at<unsigned char>(l_ligne,l_colonne) = 0;
            }
        }
    }
}

```

```

    }

}

setImageTraite(l_Image_Out);
return l_Image_Out;

}

//Detection de contours Sobel_x et Filtre
Mat Traitement::Sobel(){

    Mat l_Image_In = getImage();
    //Initialisation de la matrice resultat
    Mat l_Image_Out(l_Image_In.size(), CV_8U);
    Mat l_grayMat; //Image en niveau de gris
    Mat l_gradientX; // Gradient en x de Sobel
    Mat l_gradientY; // Gradient en y de Sobel

    // Conversion en niveau de gris de l'image d'origine
    cvtColor(l_Image_In,l_grayMat, CV_BGR2GRAY);

    //
l_gradientX = MatriceConvolution(l_grayMat,Filtre(2));
l_gradientY = MatriceConvolution(l_grayMat,Filtre(3));

    l_Image_Out = norme_matriciel (l_gradientX, l_gradientY);

    setImageTraite(l_Image_Out);
    return l_Image_Out;
}

//Filtrage avec un filtre moyenneur
Mat Traitement::Moyenneur(){
    Mat l_Image_In = getImage();

    //Initialisation de la matrice resultat
    Mat l_Image_Out(l_Image_In.size(), CV_8U);
    Mat l_grayMat; //Image en niveau de gris

    // Conversion en niveau de gris de l'image d'origine
    cvtColor(l_Image_In,l_grayMat, CV_BGR2GRAY);

    // Convolution entre l'image en niveau de gris et le filtre Moyenneur
    l_Image_Out = MatriceConvolution(l_grayMat,Filtre(1));

    setImageTraite(l_Image_Out);
    return l_Image_Out;

}

//Filtre median
Mat Traitement::Median(){

    Mat l_Image_In = getImage();
    Mat l_grayMat;
    int l_ligne, l_colonne;
    //Initialisation de la matrice resultat
    Mat l_Image_Out(l_Image_In.size(), CV_8U,double (0));
}

```

```

    // Conversion en niveau de gris de l'image d'origine
    cvtColor(l_Image_In, l_grayMat, CV_BGR2GRAY);

    l_Image_Out = l_grayMat.clone();

    vector <double> l_fenetre;
    for ( l_ligne = 1; l_ligne < l_grayMat.rows; l_ligne++)
    {

        for ( l_colonne = 1; l_colonne < l_grayMat.cols; l_colonne++)

        {

            l_fenetre.push_back(l_grayMat.at<unsigned char>(l_ligne - 1, l_colonne - 1));
            l_fenetre.push_back(l_grayMat.at<unsigned char>(l_ligne, l_colonne - 1));
            l_fenetre.push_back(l_grayMat.at<unsigned char>(l_ligne + 1, l_colonne - 1));
            l_fenetre.push_back(l_grayMat.at<unsigned char>(l_ligne - 1, l_colonne));
            l_fenetre.push_back(l_grayMat.at<unsigned char>(l_ligne, l_colonne));
            l_fenetre.push_back(l_grayMat.at<unsigned char>(l_ligne + 1, l_colonne));
            l_fenetre.push_back(l_grayMat.at<unsigned char>(l_ligne - 1, l_colonne + 1));
            l_fenetre.push_back(l_grayMat.at<unsigned char>(l_ligne, l_colonne + 1));
            l_fenetre.push_back(l_grayMat.at<unsigned char>(l_ligne + 1, l_colonne + 1));

            //Ranger par ordre croissant
            sort(l_fenetre.begin(), l_fenetre.end());

            l_Image_Out.at<unsigned char>(l_ligne-1, l_colonne-1) = l_fenetre[4];

            // Supprimer le contenu du filtre
            l_fenetre.clear();
        }
    }

    setImageTraite(l_Image_Out);
    return l_Image_Out;
}

```

```

Mat Traitement::Gaussien(){

    //Initialisations
    Mat l_Image_In = getImage();
    Mat blur(l_Image_In.size(), CV_8U,double (0));
    Mat differen(l_Image_In.size(), CV_8U,double (0));
    Mat Image_Out(l_Image_In.size(), CV_8U,double (0));
    Mat grayMat;

    float gauss[25][25] = { 0 };
    int dim = 15;
    int x0 = dim / 2;
    int y0 = dim / 2;
    int sigma;
    float pi = 3.1416;
    float sumFiltr = 0;

    float sum2;
    int promo2;

    cvtColor(l_Image_In,grayMat, CV_BGR2GRAY);
    Image_Out = grayMat.clone();

    //Initialisation par l'utilisateur de sigma
    cout<<"Veuillez un valeur de sigma:" <<endl;
    cin>>sigma;
}

```

```

for (int i = 0; i < dim; i++) {
    for (int j = 0; j < dim; j++) {
        int cX = i - x0;
        int cY = y0 - j;
        float up = (cX * cX) + (cY * cY);
        float down = 2 * (sigma * sigma);
        float exp1 = exp(-(up) / (down));
        float constant = 1.0 / (sigma * sigma * 2 * pi);
        gauss[i][j] = constant * exp1;
        sumFiltr += constant * exp1;
    }
}

for (int i = 0; i < grayMat.rows; i++)
{
    for (int j = 0; j < grayMat.cols; j++)
    {
        sum2 = 0;
        promo2 = 0;
        int y2 = 0;
        for (int a = -(dim / 2); a <= dim / 2; a++)
        {
            int x2 = 0;
            for (int b = -(dim / 2); b <= dim / 2; b++)
            {
                //////// suma
                if ((i + a) >= 0 &&
                    (i + a) < grayMat.rows &&
                    (j + b) >= 0 &&
                    (j + b) < grayMat.cols)
                {
                    sum2 += int(grayMat.at<unsigned char>(i + a, j + b)) * gauss
                }
                x2++;
            }
            y2++;
        }
        promo2 = int(sum2 / sumFiltr);
        Image_Out.at<unsigned char>(i, j) = promo2;
    }
}
setImageTraite(Image_Out);
return Image_Out;
}

void Traitement::Egalisation_histogramme(){
    //initialisation des tableaux pour stockage
    int arr[256] = { 0 };
    float arr2[256] = { 0 };
    float arr3[256] = { 0 };
    Mat l_Image_In = getImage();
    Mat l_Image_Out(l_Image_In.size(), CV_8U,double (0));
    Mat grayMat;

    cvtColor(l_Image_In,grayMat, CV_BGR2GRAY);
    l_Image_Out = grayMat.clone();

    for (int i = 0; i < grayMat.rows; i++) {
        for (int j = 0; j < grayMat.cols; j++) {
            int index = (int)(grayMat.at<unsigned char>(i, j));

```

```

        arr[index]++;
    }

}

int max = 0;

for (int i = 0; i < 255; i++) {
    if (arr[i] > max) {
        max = arr[i];
    }
}

//int height = grayMat.rows;
//int width = grayMat.cols;

Mat hist(1000, 1000, CV_8U, Scalar(0));

int inc = 1000 / 256;

// calculer l'histogramme de l'image originale
for (int i = 0; i < 255; i++) {
    rectangle(hist, Point(inc * i, hist.rows), Point((inc * (i + 1) - 1), hist.rows - ((arr[i] * 1000) / 256)));
}

float total = grayMat.cols * grayMat.rows;
for (int i = 0; i < 255; i++)
{
    arr2[i] = float(arr[i]) / total;
}
arr3[0] = arr2[0];

for (int i = 1; i < 255; i++)
{
    arr3[i] = arr2[i] + arr3[i - 1];
}

for (int i = 0; i < grayMat.rows; i++) {
    for (int j = 0; j < grayMat.cols; j++) {

        l_Image_Out.at<unsigned char>(i, j) = floor((256 - 1) * arr3[grayMat.at<unsigned char>(i, j)]);
    }
}

// egalisation d'image
int h2[256] = { 0 };

for (int i = 0; i < l_Image_Out.rows; i++) {
    for (int j = 0; j < l_Image_Out.cols; j++) {
        int index = (int)(l_Image_Out.at<unsigned char>(i, j));
        h2[index]++;
    }
}

int maxH2 = 0;

for (int i = 0; i < 255; i++) {

```

```

        if (h2[i] > maxH2) {
            maxH2 = h2[i];
        }
    }

    Mat hist2(1000, 1000, CV_8U, Scalar(0));

    inc = 1000 / 256;

    // calculer l'histogramme egaliser
    for (int i = 0; i < 255; i++) {
        rectangle(hist2, Point(inc * i, hist2.rows), Point((inc * (i + 1) - 1), hist2.rows));
    }

    namedWindow("Histogramme\u00e9galise\u00e9", WINDOW_NORMAL);
    resizeWindow("Histogramme\u00e9galise\u00e9", hist.rows/2, hist.cols);
    imshow("Histogramme\u00e9galise\u00e9", hist);

    namedWindow("Histogram\u00e9galise\u00e9", WINDOW_NORMAL);
    resizeWindow("Histogram\u00e9galise\u00e9", hist2.rows/2, hist2.cols);
    imshow("Histogram\u00e9galise\u00e9", hist2);

    namedWindow("Image\u00e9galise\u00e9", WINDOW_AUTOSIZE);
    imshow("Image\u00e9galise\u00e9", l_Image_Out);
    setImageTraite(l_Image_Out);

}

waitKey(0);

destroyWindow("Histogramme\u00e9galise\u00e9");
destroyWindow("Histogram\u00e9galise\u00e9");
destroyWindow("Image\u00e9galise\u00e9");

}

Mat Traitement::Segmentation()
{
    Mat image = getImage();

    // Convertir l'image en YUV
    Mat l_yuvImage;
    cvtColor(image, l_yuvImage, COLOR_BGR2YUV);

    // Dfinir les limites de la plage de couleurs slectionner
    int yMin = 0, yMax = 255;
    int uMin = 90, uMax = 140;
    int vMin = 90, vMax = 140;
    /*
    int yMin, yMax;
    int uMin, uMax;
    int vMin, vMax;

    cout<<"entrez la valeur de yMin :"<<endl;
    cin >> yMin;
}

```

```

cout<<"entrez la valeur de yMax :"<

```

```

}

void Traitement::Sauvegarder(){

    bool l_found = false;
    bool l_exist;
    string l_Image_name;
    string l_path;
    vector <string> extensions = {"jpeg", "jpg", "JPEG", "JPG", "png", "PNG"};

        bool isValid_reponse1 = false;

    string l_reponse1; //Rponse au question de confirmation Y ou N pos
    while (!isValid_reponse1) {
        cout << "Vous voulez enregister votre traitement Y ou N" << endl ;
        cin >> l_reponse1;
        cin.ignore(numeric_limits<streamsize>::max(), '\n');
        if (l_reponse1 == "Y" || l_reponse1 == "N") {
            isValid_reponse1 = true;
        }
        else {
            std::cout << "Acces non valide. Veuillez saisir Y ou N." << endl;
            cin.clear();
        }
    }

    if(l_reponse1== "Y"){

        // Vrification de la prsence d'une extension
        do{
            int i = 0;
            cout << "Veuillez saisir un nom pour enregister l'image:" << endl ;
            cin >> l_Image_name ;
            // Vrification de la prsence d'une extension
            while (i < int(extensions.size()) && !l_found) {

                if (recherche_extension(l_Image_name, extensions[i])) {
                    l_found = true;
                }
                i++;
            }

            if (l_found) {
                cout << "Extension de l'image valide" << endl;
            }
            else {
                cout << "Veuillez saisir une extension d'image valide avec le nom" << endl;
            }
        }while(!l_found);

        l_path = "./Data/Image/" +l_Image_name;

        l_exist = experimental::filesystem::exists(l_path) ;

        if(l_exist){


```

```

/* "isValid_reponse" qui est initialisé false. La boucle while va continuer
tant que la saisie n'est pas valide (isValid_reponse = false).
Si la saisie est valide, isValid_reponse est mis true et la boucle s'arrête.
*/
bool isValid_reponse = false;

string l_reponse; //Réponse au question de confirmation Y ou N posé
while (!isValid_reponse) {
    cout << l_Image_name << "Existe déjà. Voulez-vous le remplacer ? Y ou N" << endl ;
    cin >> l_reponse;
    cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
    if (l_reponse == "Y" || l_reponse == "N") {
        isValid_reponse = true;
    }
    else {
        std::cout << "Accès non valide. Veuillez saisir Y ou N." << endl;
        cin.clear();
    }
}
if(l_reponse == "N"){
    //Nous revenons au début de la méthode et on recommence la procédure
    Sauvegarder();
}
else{
    setImage(getImageTraité());
    imwrite(l_path, getImageTraité());
}
else{
    imwrite(l_path, getImageTraité());
    Ajouter_image();
}

}

else{
    setImageTraité(getImage());
}
cout << "Tapez sur la touche Entrée pour sortir des traitements...";
//cin.ignore(std::numeric_limits<streamsize>::max(), '\n');
cin.get();
}

```