RAPPORT
S1.01 IMPLENTATION D'UN BESOIN CLIENT

Travail réalisée du 18 Octobre au 30 Novembre 2022

Germain DOGLIOLI-RUPPERT
Clara TORRI
Jeremy MONCADA
Anice KHENICHE

Table des matières

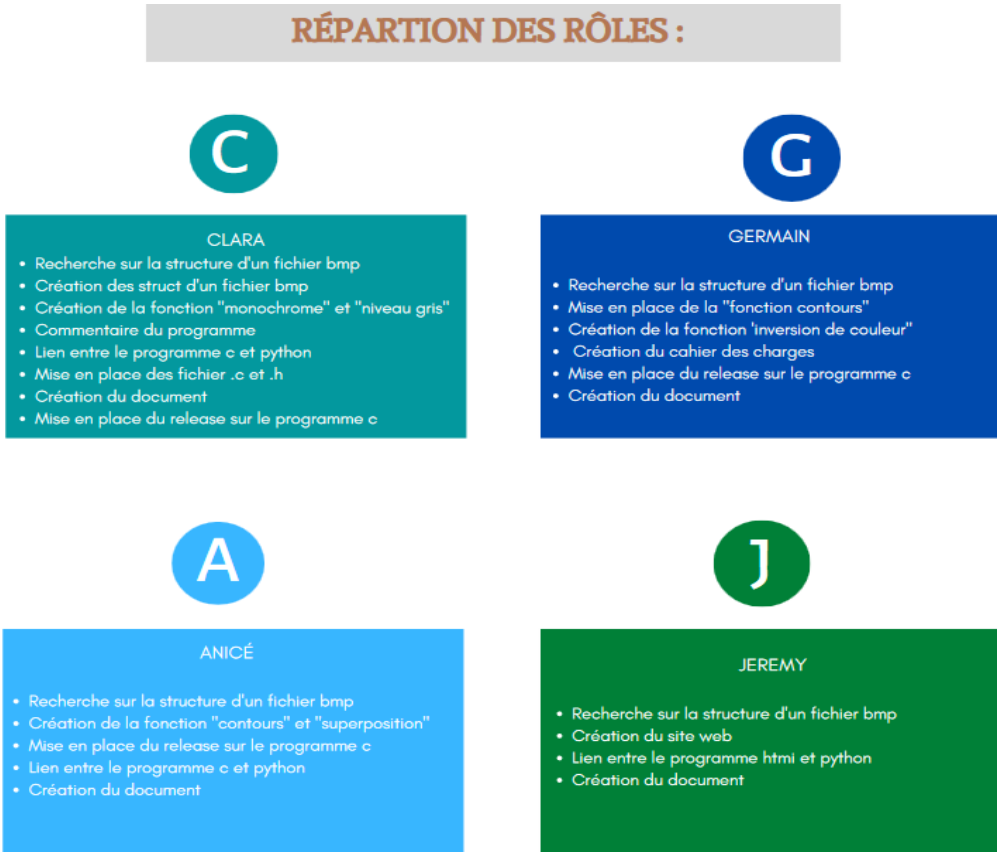
I. Introduction :

Dans ce nouveau projet de traitement d'images qui consiste à l'aide de programme en langage c et HTML de créer un site éditeur d'image. C'est-à-dire un site qui peut modifie une image qui a été sélectionnée par l'utilisateur de 5 façons différentes possibles qui seront énoncé et détaillé plus tard dans le document.

Nous allons vous présenter à l'aide de ce document, la répartition des rôles qui ont été établit à chaque membre du groupe, le découpage des nombreuses tâches à réaliser, ont contribués à la mise en place de notre planning prévisionnel de l'ensemble des séances qui nous ont étaient proposée tout au long du projet. On explique aussi, les principes de fonctionnement du projet ainsi que les parties les plus importantes du programme. Et enfin qu'elle choix ont été pris pour réaliser le code et faire marcher correctement notre programme à notre façon.

II. Répartitions des rôles :

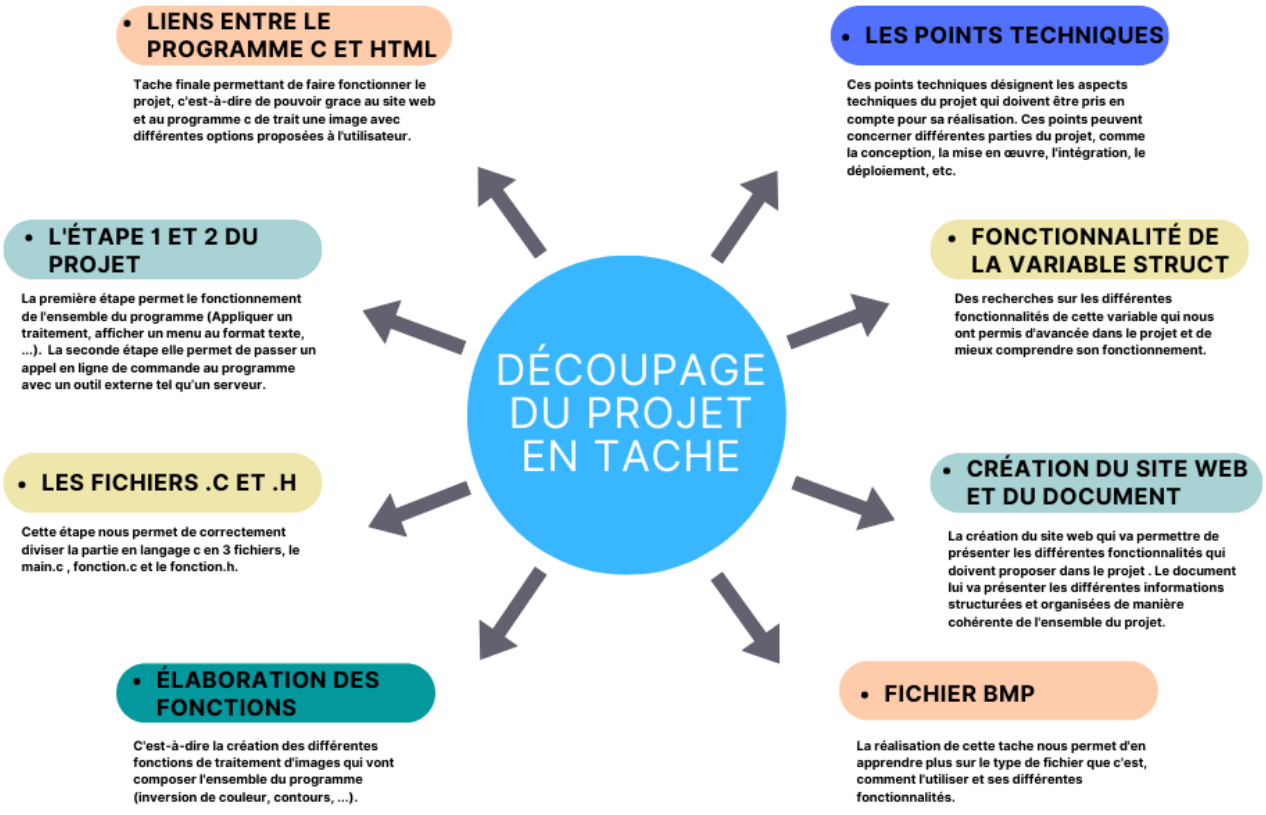
La répartition des rôles dans un projet est un aspect crucial pour assurer sa réussite. Chaque membre d'une équipe de projet a des responsabilités et des tâches spécifiques à accomplir pour contribuer à atteindre les objectifs du projet. Les membres de l'équipe sont responsables de l'exécution des tâches assignées. Il est important de bien répartir les rôles pour assurer l'efficacité et l'efficience de l'équipe. De plus, une bonne répartition des rôles peut aider à éviter les conflits et à maintenir une bonne dynamique au sein de l'équipe.



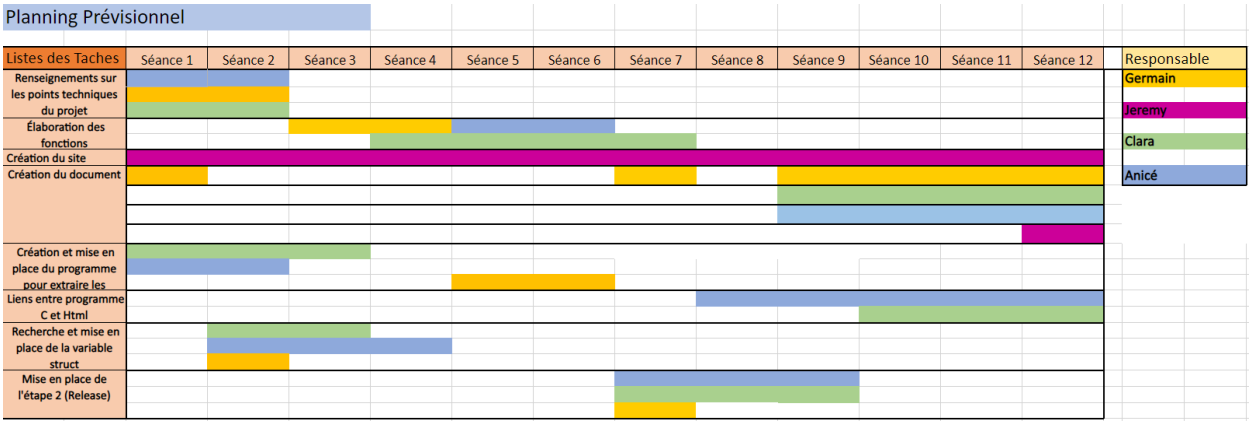
III. Découpage du projet en taches :

Le découpage d'un projet en tâches est une étape cruciale pour assurer sa réussite. Cela consiste à décomposer le projet en différentes étapes et tâches plus petites et gérables, qui peuvent être facilement planifiées et assignées à des membres de l'équipe. Cela permet de rendre le projet plus facile à gérer et de suivre l'avancement des différentes tâches. Le découpage en tâches permet également d'identifier les dépendances entre les différentes tâches, ainsi que les ressources nécessaires pour les accomplir. Cette approche permet de planifier de manière plus précise et d'éviter les retards ou les problèmes qui pourraient survenir au cours de la réalisation du projet.

Voici donc les différentes taches réaliser sur l'ensemble du projet :



IV. Planning Prévisionnel :



V. Explication sur l'architecture des solutions :

1. Le fichier Bitmap :

Également connu sous le nom de fichier bitmap, un fichier BMP est un format de fichier image pixelisé qui remonte au début de l'infographie, et est conçu pour afficher des images indépendamment des périphériques. Ce format de fichier contient beaucoup de renseignements et aura donc tendance à être plutôt lourd.

Avantages des fichiers BMP

• Chaque fichier BMP ne dépend pas du périphérique. Il peut donc être stocké et affiché sur différents périphériques et écrans sans aucune perte de qualité.

• Polyvalent, le format BMP peut prendre en charge de multiples profondeurs de couleur, profils de couleur et couches alpha. Il soutient également la compression des données.

• Enfin, le logiciel est compatible avec de nombreux navigateurs et applications Web. Photoshop est l'une des nombreuses applications de modification d'image qui soutiennent ce format de fichier.

Inconvénients des fichiers BMP

• Certaines personnes croient que le format de fichier BMP est désuet parce qu'il a été conçu pour les anciennes applications Windows, avant l'apparition des appareils mobiles Android et Apple.

• Les fichiers BMP non compressés peuvent être bien plus grands que les fichiers JPEG et PNG, ce qui rend leur partage difficile. Ils peuvent également être trop volumineux pour être utilisés sur des sites Web ou stockés sur des disques durs de faible capacité.

• Ils ne peuvent contenir que des images RVB.

Pour ce projet le fichier *bmp* a été utilisée pour sa simplicité et sa facilité d'accès au sein des pixels.

2. Le main.c :

La fonction main permet d'exécuter les différentes actions lors des appels en ligne du code et des différentes fonctions de traitement d'images.

Pour ce faire, nous avons initialisé l'attribut n qui permet de vérifier si les fichiers existent, l'attribut argument qui prend comme valeur le nombre d'arguments passés dans la main grâce à la variable argc, l'attribut valeur qui est le numéro du traitement appelé. Ainsi que quatre pointeurs sur des chars qui nous permettent de contenir l'adresse mémoire des chemins pour aller jusqu'aux images ainsi que leurs noms. A l'aide de la variable switch, qui prend comme argument *argument*, nous avons programmé les différentes actions demandées :

- Le premier *case 1* est l'action sans argument, 1 car le premier argument est automatique mis, dans un *default,* il ouvre un menu texte permettant de saisir le chemin du répertoire de travail, le numéro du traitement attendu, une boucle *if* vérifie s'il existe pas ou s'il s'est bien déroulé et retourne -1 en cas d'erreur , le nom de l'image à traiter et une 2ème image, pour le traitement 5 seulement, la saisi se fait à l'aide de *scanf* et de *strcat* qui concatène les contenus de deux chaînes de caractères et stock le résultat dans la première chaîne passé en paramètre. Nous avons établi des lignes de commande qui rajoute en cas d'oubli le « \ » lors de la saisi du chemin.

- Le deuxième est celui à 4 arguments, *case 5,* l'exécutable n'ouvre pas de menu texte, il récupère en argument 1 : le chemin vers le répertoire de travail, en a 2 : le numéro du traitement, en 3 : le nom du 1er fichier bmp et en 4 : le nom du 2eme fichier bmp. Il ne contient pas de ligne de commande étant donné qu'il fait à peu près la même chose que celui à 3 arguments et dans le *case 1*, il y a une boucle *if* qui s'occupe de demander à l'utilisateur de donner le nom de la seconde image et la variable *strcat* concatène le chemin de l'image avec le nom de l'image.

- Le dernier est celui à 3 arguments, *case 4*, l'exécutable n'ouvre pas de menu texte, il récupère en argument 1 : le chemin vers le répertoire de travail, en 2 : le numéro du traitement, en 3 : le nom du fichier bmp à traiter. Comme dans le *case 1*, on rajoute la

« \ » en cas d'oubli, on utilise la variable *strcy* qui permet de copier le chemin jusqu'à la première image dans *argv[1]* , qui est donc le premier argument, avec la variable *strcat*, on concatène le contenus de chemin avec le contenue de *argv[3],* qui est le troisième argument donc le nom de l'image, et on stock le résultat dans chemin. Avec une boucle *if,* on dit que si le quatrième argument est différent de nul, alors, on fait la même chose que pour le chemin et le nom de la première image. Pour finir, on transforme l'attribut *valeur,* grâce à la variable atoi, qui permet de transformer une chaîne de caractères, représentant une valeur entière, en une valeur numérique afin de choisir le type de conversion, le *case 4* se termine par un *break*.

Nous avons utilisé un deuxième *switch* qui permet de choisir le type de traitement que l'utilisateur veux effectuer, il faut entrer un chiffre entre 1 et 5, un numéro et relié à une fonction, par exemple 1 et relier à la fonction nuance de gris. Le lien est fait grâce au *#include "fonction.h"* mis au début du programme qui permet de faire le lien avec l'endroit où se situe toutes les fonctions.

- <u>L'étape 1, le fonctionnement du programme :</u>

L'étape 1 est la première étape de fonctionnement de notre programme, dans cette étape nous avons fait afficher une image à l'aide de programmes et par la suite nous avons codé les fonctions qui nous ont permis de traiter les images attendues et de sauvegarder le résultat dans un nouveau fichier « résultat ». Le code n'utilise aucune librairie extérieure qui permettrait d'améliorer le fonctionnement des programmes.

- <u>L'étape 2, le release :</u>

La seconde étape est assez différente de la première car elle consiste non pas à faire fonctionner le programme depuis un IDE mais à partir d'un terminal grâce à des lignes de commande. Chaque argument et passer en ligne de commande, c'est pour cela qu'il faut renseigner le chemin pour accéder au programme ou aux images.

Cette seconde étape, si elle fonctionne, nous permet de faire fonctionner la liaison entre le programme c et HTML. La release est donc l'option à activer dans l'IDE pour faire passer notre programme en ligne de commande.

3. Le fichier fonction.c :

Le fichier contient toutes les fonctions demander :

- Nuances de Gris :

Cette fonction va appliquer un filtre gris à l'image sélectionné. Pour cela, on fait la moyenne de chaque pixel, la moyenne sera l'addition des pixels rouge, vert et bleu (RGB) qu'on divise par 3. Par la suite, tous les pixels vont prendre la valeur de la moyenne des pixels.

- L'inversion de couleurs :

Cette fonction consiste à inverser la couleur de chaque pixel composant le fichier bitmap (par exemple si la couleur d'origine est bleu quand on inverse cela donnera de l'orange). Ainsi, on soustrait les différents pixels par 255, qui est la couleur blanche.

- Monochrome :

La fonction monochrome est une fonction qui met en noir et blanc n'importe qu'elle image. Pour ce faire, on donne une valeur moyenne, un pixel va de 0 à 255 donc la moyenne entre ces deux valeurs est 127, puis on cherche tous les pixels possédant un code couleur RGB inférieur à 127 qu'on transforme en noir. Tous les pixels supérieurs à 127 seront transformés en blanc.

- Contours :

Cette fonction permet de faire ressortir les pixels les plus foncé d'une image. On fait la moyenne d'un pixel et de celui qui se situe avant lui, puis on fait la différence entre ces deux moyennes et si la différence est trop grande alors le pixel ayant la plus grosse moyenne devient noir. Et cela pour tous les pixels tant que les derniers pixels n'est pas atteint

- Superposition :

La fonction superposition permet tout simplement de mettre une première image sur une seconde et de faire en sorte qu'on puisse percevoir à la fois l'image 1 et l'image 2. Chaque pixel prend comme valeur la moyenne de sa couleur de l'image 1 et l'image 2, exemple le pixel rouge prend comme valeur le résultat de la moyenne entre le pixel rouge de l'image 1 et celui de l'image 2.



Toutes les fonctions ont des instructions en commun :

Elles commencent toutes par :

- Ouverture du fichier en mode lecture
- Boucle *if* qui vérifie que le fichier existe et peut être ouvert sinon renvoie -2
- Création du fichier resultat.bmp en mode écriture
- Boucle *if* qui vérifie que resultat.bmp existe et peut être écrit sinon renvoie -3
- Initialisation des structures se trouvant dans fonction.h en leurs donnant un nom précis
- Lecture de l'entête de l'image et du fichier
- Ecrit les informations de l'entête dans le fichier résultat
- Initialisation d'un compteur

Dans la boucle *while* :

- La condition est vraie tant que le pixel lu est différent de 0
- Rajoute 1 au compteur
- Ecrit les valeurs des pixels rouge, vert et bleu dans le fichier résultat
-

Elles finissent toutes par :

- Fermeture des fichiers image et résultat
- Lignes de code qui permettent de modifier le nom du fichier *resultat* en *result*
- Retourner n en cas d'erreur

4. Le fichier fonction.h :

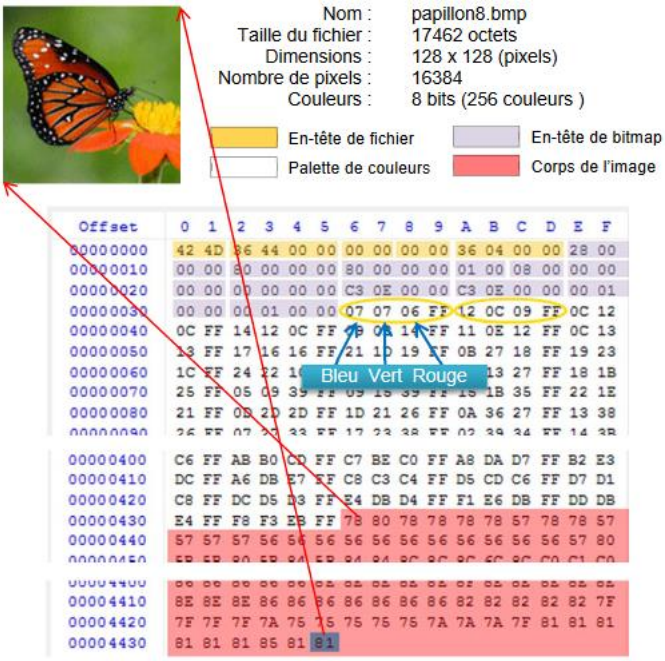
Dans ce fichier, nous avons défini toutes les fonctions qui sont dans la fonction.c ainsi que des structures permettant de décomposer un fichier bmp.

La structure header_image est une structure qui permet de définir l'entête du fichier ainsi ça nous permet d'avoir plus d'informations sur le type du fichier, sa taille et ça indique aussi ou les informations de l'image commencent. Avec des attributs comme la signature qui indique que c'est un fichier bmp, size qui donne la taille totale, un champ réservé et l'offset de l'image soient l'adresse où se situent les informations de l'image.

La structure info_image permet de définir l'entête de l'image grâce à sa taille, ses dimensions avec width et height, ses paramètres dont son nombre de plan, sa profondeur ou encore sa méthode de compression, la définition du nombre de pixels avec sa taille totale et sa résolution horizontale et verticale ainsi que sa palette comme le nombre de couleur ou encore les couleurs importantes.

La structure palette_pixel sert à séparer le pixel qui est composé des trois couleurs primaire le rouge, le vert et le bleu.

Exemple de la décomposition d'un fichier BMP :



VI. Le site web

Le fonctionnement du site est assez simple. Dans un premier temps l'utilisateur doit choisir un fichier à traiter (.bmp uniquement). Par la suite il va devoir valider le ou les types de conversions qu'il souhaite appliquer à son image. Et pour finir il n'a plus qu'a cliquer sur traiter l'image.

Le site web est constitué d'une structure principale centrée sur la page. En haut de la structure nous retrouvons un input de type file correspondant au fichier 1 permettant de sélectionner une image à traiter, il est uniquement possible de choisir une image .bmp, cette indication s'affiche en survolant la zone de sélection de l'image grâce à un hover la feuille de style css nommée « waring » lui est associée. La structure est ensuite composée de différents boutons (un bouton par transformation), une fois le bouton cliqué, s'affiche un contenu étant composé d'un texte indiquant les effets que la transformation effectuera à l'image et un bouton « Valider » apparait aussi permettant de sélectionner le type de conversion. Spécialement pour l'option de fusion des deux images, un nouvel input de type file correspondant au fichier 2 apparait sous le texte explicatif de la conversion permettant de sélectionner un second fichier. Sous chaque boutons (contenu affiché ou non compris) se trouve un aperçu de la transformation en question. Pour finir à la fin de la structure est positionné grâce à un input de type submit, un bouton de traitement qui lance le traitement de l'image une fois celle-ci sélectionnée.

Il y a trois feuilles de style css situées dans un dossier css. Ces trois feuilles sont liées à la partie html du site. Il y a une feuille nommée « main » qui correspond aux éléments et classes principales du site. Une seconde feuille nommée « button » qui correspond à tout ce qui est liée aux différents boutons du site. Et une troisième feuille nommée « warning » qui correspond à la zone de prévention indiquant le type de fichier accepté, ici les fichier de types bitmap uniquement sont accepté.

VII. Choix dans les solutions techniques et Problèmes :

1. Pour le code c :

Pour manipuler les fichiers BMP, on voulait, au tout début, utiliser de l'allocation dynamique avec des malloc afin de mettre en mémoire les pixels des images. Mais nous n'étions pas très confiants de ce que nous faisions. C'est pour cela que nous avons abandonné cette idée et que nous nous sommes rabattus sur les structures, qui étaient plus faciles à comprendre et à expliquer.

Au niveau des problèmes que nous avons rencontrés, nous allons citer en citer quelques-uns :

- Les chemins de travail : nous avons eu du mal à comprendre comment on devait créer l'exécutable qui devait effectuer les appels en lignes et les différentes actions mais après avoir fait plusieurs tests on s'en approchait. A un moment, nous n'avons pas

compris pourquoi, lorsque nous testions une première fois le programme fonctionnait mais quand nous retestions ça ne marcher plus, puis nous avons compris notre erreur, nous avions oublié de mettre « \ », de ce fait nous avons créer une boucle *if* qui le rajoute à la fin du chemin en cas d'oubli. Puis on a continué nos tests mais cela ne fonctionnait toujours pas et après avoir relu l'énoncer du travail nous avons remarqué que nous n'avions pas départagé le programme comme il le fallait, sans argument, avec trois arguments et avec quatre arguments. Donc quand tout était bien départagé, dans un *switch*, nous avons remarqué que case 5, donc 4 arguments, qui contient les lignes de code pour le chemin et le nom de la deuxième image et case 4, 3 arguments, qui contient les lignes de code du chemin et nom de la première image ne fonctionnaient pas donc les lignes de code dans case 5 et allé dans case 4, étant donné que case 4 et 5 sont relié, ils font la même chose mais on rajoute un argument à case 5, grâce à une boucle *if*. Après ce changement, tout a été mis en place et a fonctionné à la perfection.

- Le fichier resultat.bmp : dans notre programme nous avons appelé le fichier résultat resultat.bmp mais dans le python, donné par le professeur de html, le fichier résultat s'appelait result.bmp. Lorsqu'on lance le traitement sur le site on pouvait voir que deux fichiers résultat apparaissaient, celui que nous avons créés avec la modification demandée et celui du professeur avec que l'image demandé sans aucun changement, donc nous nous sommes dit qu'en modifiant le nom du fichiers *resultat* en *result* ça devrait potentiellement marcher. En relançant le site et le traitement nous avons remarqué que cela ne fonctionnait plus et après avec effectué plusieurs tests nous avons réussis à modifier le nom de notre resultat.bmp grâce à la fonction *remove* qui supprime le fichier result.bmp que le fichier python créer et à la fonction *rename* qui renomme notre fichier *resultat.bmp* en *result.bmp*.

VIII. Conclusion :

En définitive, ce projet de traitement d'images à partir d'un site web a permis tous les membres du groupe d'approfondir leurs connaissances en langage c et HTML et d'avoir réalisé un projet de programmation en groupe pour la 1re fois.

Chacun de nous a alors eu des rôles à remplir au sein du projet (création du site web, du document, les fonctions …), au qu'elle nous a découpé ce projet en plusieurs taches distinctes pour non seulement mieux s'organiser mais aussi pour que chacun y trouve un repaire pour travailler.

Nous avons aussi accompagné ce projet d'un planning prévisionnel (voir page 6) qui permet de bien visualiser pour chaque membre du groupe qu'elle tache il a réalisé et sur quelle séance à t-il était fait. Ce planning permet aussi de mieux comprendre l'étendue du projet.

Puis dans une autre partie nous vous avons alors expliqué le fonctionnement du fichier bit map et son utilité pour notre projet. Par la suite nous avons expliqué qu'elles ont étaient l'ensemble des solutions des fonctions sur les programmes c et HTML, que ce soit l'explication des fonctions main.c ou header.h ou encore la clarification des fonctions de traitement d'image (inversion de couleurs, nuance de gris...).

Nous avons aussi expliqué qu’elles étaient nos choix dans la réalisation du site web que ce soit sur le fichier HTML ou encore le fichier CSS.

À la suite de cette partie nous avons alors aussi montré les choix dans nos solutions techniques c'est-à-dire le pourquoi avons-nous décidé par exemple de choisir les structures sur nos fichiers c ou encore pourquoi avons choisi certaines choses dans le développement du site web....

Ce travail nous a alors permis d’obtenir notre premier site web fonctionnable, qui peut parvenir à traiter des images et aussi à afficher le résultat.