Cahier des charges

adam.mahraoui
emil.toulouse
samy.achache

14 février 2022

Sommaire

1 Introduction

1.1 Présentation du projet

Notre projet de S4, est un logiciel dans lequel l'algorithmique a une part très importante, il s'agira d'étudier le QR-code très en vogue en ce moment dû à la crise sanitaire. Ce projet est divisé en plusieurs sous-programmes. Le premier sera l'encodage d'un texte ou lien hypertexte sous forme de QR-code et d'avoir la possibilité de récupérer ce résultat sous forme d'image au format classique.

Par la suite nous avons jugé intéressant que notre logiciel propose également le décodage de ces QR-code et avec la possibilité de voir sous forme textuelle le contenu d'un QR-code. Nous avons donc besoin d'un algorithme de décodage de QR-code ainsi que de regrouper tout cela dans une GUI afin que l'expérience avec notre logiciel n'en soit que meilleur. Avant d'aller plus loin et sans plus attendre voici notre groupe.

1.2 Qui sont les membres du groupe ?

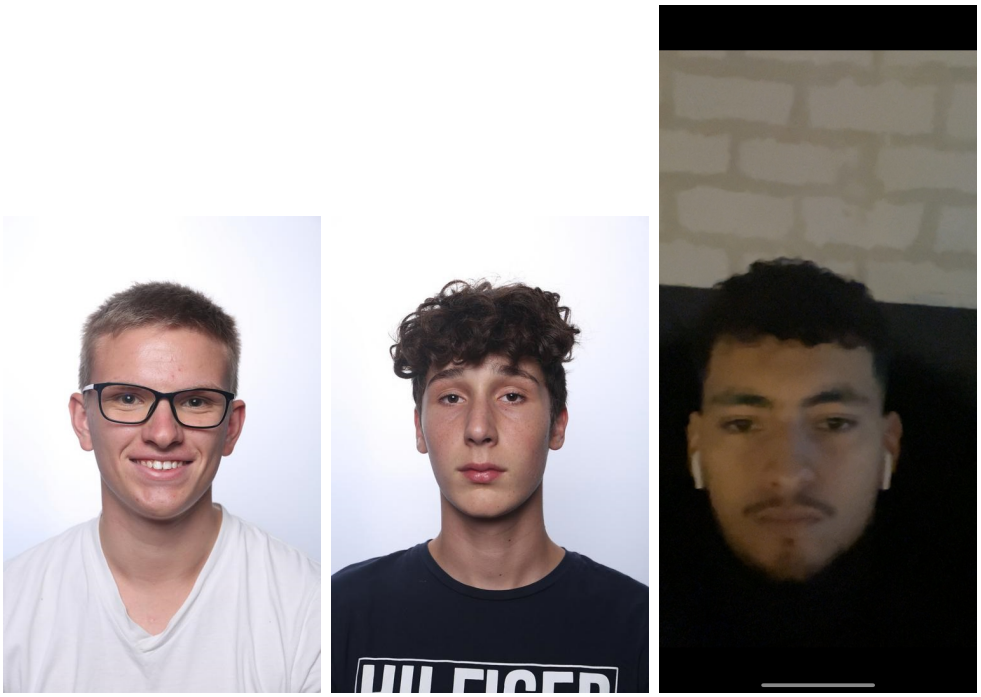
Notre groupe est composé de quatres personnes actuellement, Emil Toulouse, Adam Mahraoui, Icham Benabbas et Samy Achache. Avec Icham Bennabas pour chef de groupe désigné.

- Emil Toulouse : Le QR-code pour moi est un outil qui a un fort potentiel pour transmettre une information et simplifier des actions de manière rapide. Dans cette optique la proposition de faire notre version du QR-code m'as bien plu. C'est pour cela que j'ai accepté de rejoindre l'équipe. Ayant l'ambition de bien comprendre cette technique qui semble magique, je compte me positionner sur la partie encodage d'un QR-code.

- Adam Mahraoui : La curiosité est un trait de personnalité qui me caractérise bien et c'est pour cela que cette idée projet m'a plu. J'ai toujours utilisé les QR-codes en me demandant comment cela fonctionnait, je n'ai jamais vraiment eu le temps de chercher la réponse à cette question mais ce projet va me permettre d'y répondre. Ce thème me permet de revoir le traitement de l'image et mêler ça avec de l'algorithmie sur un sujet intéressant et concret.

- Samy Achache : Je baigne dans le monde de l'informatique depuis l'âge de quatre ans, à travers les jeux videos, console, Pc. Mon expérience informatique avant Epitas resume plus à l'utilisation a la création. Plus tard je souhaite idealementme spécialiser dans une branche particulière de l'informatique qui est la cybersécurité. L'avantage de ce projet est

le choix de notre logiciel. Effectivement faire un projet autour des QR-Codes sachant le contexte actuelle (pandemie) ou l'utilisation des QR-Codes est devenu repandue connaitre leur fonctionnement me semble trezs important pour moi.



Membres de l'équipe

2 Pourquoi ce projet ?

La technique des QR-codes n'est pas récente, en effet elle existe depuis les années 2000. Elle permet de stocker une information sous forme d'image (du texte, une connexion internet, envoyer un courriel, une vidéo en ligne...).

C'est une technique simple à utiliser et comme son nom l'indique "Quick response Code" qui permet d'obtenir une réponse rapide et simple : il suffit juste de scanner cette image qui ressemble à un damier pour que quelque chose se passe, cela semble presque magique.

Nous voulons comprendre les rouages derrière cette méthode de partage de donné. Comment une simple image peut-elle permettre de faire autant de chose ? Et comment est-il possible de transformer une information en un "damier corrompu" lisible par tous les appareils électroniques.

2.1 Les objectifs du projet

Pour ce projet qui a pour but final de rendre un logiciel capable de lire un QR code ou de transformer un lien en celui-ci, nous nous fixons plusieurs objectifs intermédiaires pour rendre un travail complet, fonctionnel et qui répondra aux exigences demandées. Voici la liste de nos principaux objectifs intermédiaires qui devront être atteints pour la soutenance finale :

- Chargement d'une image contenant un QR Code ;
- Détection du QR code sur l'image ;
- Implémentation de l'algorithme permettant de lire le QR Code;
- Implémentation de l'algorithme permettant de creer son QR Code;
- Réaliser une interface pour le logiciel ;

3 Encodage de l'information

Nous avons decider de nous pencher au plus vite sur la transformation de la donnée utilisateur en une nouvelle chaine binaire correspondant à ce qui va completer le QR-code. Le principe est simple sur le papier et plus compliqué dans sa mise en place. En effet la creation d'un QR-code (les données contenues) doivent respecter un certain pattern et des paramétrages bien précis pour que le résultat final soient utilisable.

3.1 Choix du mode et de du niveau de correction

Les QR-codes peuvent stocker un certain nombre d'informations et de type différents. Le choix d'un mode correspond à la palette de caractère accepter pour l'encryption du message initial.

Numeric mode is for decimal digits 0 through 9.

Alphanumeric mode is for the decimal digits 0 through 9, as well as uppercase letters (not lowercase!), and the symbols $, %, *, +, -, ., /, and : as well as a space. All of the supported characters for alphanumeric mode are listed in the left column of this alphanumeric table.

Byte mode, by default, is for characters from the ISO-8859-1 character set. However, some QR code scanners can automatically detect if UTF-8 is used in byte mode instead.

Tableau des modes QR code

Parmis ces 3 choix nous avons retenu le plus complet (Byte mode) qui prendra en compte tout les caractères du clavier AZERTY classique. Cependant nous avons pris le choix de ne pas prendre en compte un quatrième mode le Kanji Mode puisqu'il inclut des caratères supplémentaires que nous avons juger peux utile pour la charge de travail apporté.

Passons au niveau de correction. Ce niveau de correction est un paramétrage simple ayant pour objectif de spécifier la complexité du QR-code afin que,meme apres degradation partiel, il reste opérationnel.Nous avions le choix entre plusiseurs paramétrage et nous avons retenu le mode M qui une fois mise en place dans sa totalité pourra être utlisable même après 15 pourcent de dégradation.

3.2 Prémice de l'encodage

Nous avons opté pour réaliser une structure contenant tout les possibles paramètres de l'encodage du QR-code avec notament sa taille, sa version, la donnée traité etc... La structure est disponible dans le projet.

```c
struct encdata
{
    // Version of the QR code.
    int version;

    // Mode indicator.
    // 1 -> byte mode
    char mi;

    // Length of the input string.
    size_t size;

    // Error correction level.
    char correction_level;

    // Length of the encoded data.
    size_t len;

    // Length necessary for the encoded data.
    size_t nlen;

    // Encoded data (binary).
    char *data;

    // Error correction Codewords Per Block
    size_t ec;

    // Number of blocks in group 1
    size_t block1;

    // Number of Data Codewords in Each of Group 1's Blocks
    size_t group1;

    // Number of Blocks in Group 2
    size_t block2;

    // Number of Data Codewords in Each of Group 2's Blocks
    size_t group2;
};
```

structure d'encodage QR code

Une fois créer il nous faut donc la remplir.Cela passe par le choix de la version (la plus petite possible). Le mode ainsi que l'erreur de correction étant choisi nous devons passez à la taille/version qu'aura le QR-code.Pour cela rien de plus simple chaque version du QR-code peut contenir un certain nombre d'octect et que donc en connaissant ce tableau des version nous pouvons facielement trouver le version adapté.

Nous allons enfin pouvoir commencer a remplir le champ de la data. Tous d'abord nous ecrivons "0100" qui correspond au choix du mode, ici Byte, puis nous ecrivons la taille de la donnée en binaire par exemple :

"Hello World" contient 11 charactères donc $11(10) = 1011(2)$.

Pour respecter les nomenclatures nous devons impérativement faire en sortes que cette information soient représenté sur 9 bits. Nous ajoutons donc des 0 pour que cela correspondent aux demandes.

Pour le mode alphanumérique, chaque caractère alphanumérique est représen -té par un nombre .Pour chaque paire de caractères, on récupère la représentation numérique du premier caractère et multipliez-la par 45. Ajoutez ensuite ce nombre à la représentation numérique du deuxième caractère.On convertis maintenant ce nombre en une chaîne binaire de 11 bits, en remplissant à gauche avec des 0 si nécessaire.Si la chaine est de longeur impaire le dernier charactère sera représenté sur 6 bits.

Pour avoir la bonne représentation de l'information la correction d'erreur impose une certaine taille pour la data encodée. Ainsi avec le mode choisis nous pouvons determiner cette taille. Un tableau est fourni avec le nombres de bloc erreur nécessaire. On multiplie par 8 ce nombre pour obtenir la longeur nécessaire de la chaine data.Une fois cette longeur de chaine connue nous devons donc remplir les espaces manquants. On commence par ajouter au maximum quatres zéro pour tomber remplir la taille nécessaire. Si ce n'est pas le cas on mets quatre 0.

Maintenant nous devons finir la chaine d'une manière bien spécifique. Il faut tout d'abord ajouter des 0 jusqu'a tomber sur une longeur de chaine divisible par 8 une fois réaliser. Nous terminons le processus par completer la chaine jusque la taille nécessaire avec 11101100 00010001 chacun leur tour.

3.3 codage de correction d'erreurs

Il est maintenant enfin temps de commencer à générer des mots de code de correction d'erreurs. Nous créerons des mots avec des valeur numériques qui seront utilisés comme coefficients du polynôme du message.

L'étape de codage des données a abouti aux mots de code de données suivants pour HELLO WORLD sous la forme d'un code 1-M.

00100000 01011011 00001011 01111000 11010001 01110010 11011100 01001101 01000011 01000000 11101100 00010001 11101100 00010001 11101100 00010001

On convertis ces nombres binaires en décimal : 32, 91, 11, 120, 209, 114, 220, 77, 67, 64, 236, 17, 236, 17, 236, 17

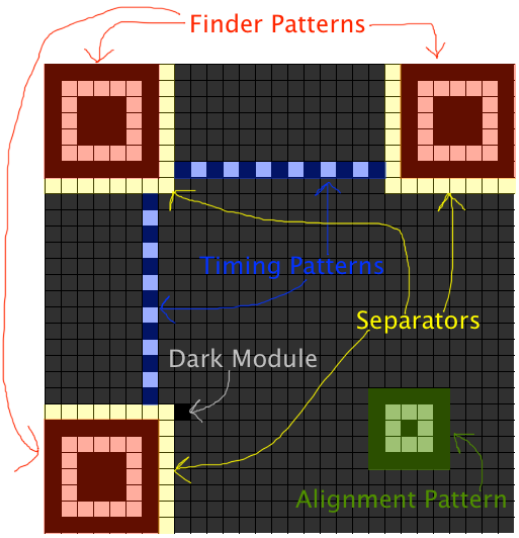
Ces nombres sont les coefficients du polynôme du message. En d'autres termes: 32x 15 + 91x 14 + 11x 13 ... et ainsi de suite

Nous n'avons pas continuer nos recherches sur ces algorithmes de creation de code erreur. Mais il restent a generer un polynome generateur et de les multiplier les puissances par le nombre de message erreurs nécessaires puis de diviser avec une porte logique XOR ces 2 polynomes afin de retomber sur des valeurs numériques comprises entre 0 et 255 que nous remettrons en binaire pour avoir le résultat fin.

4 Encodage du QR-Code

Un QR-Code peut être représenté sous la forme d'une matrice carrée contenant des 1 pour les cases noires et des 0 pour les blanches pour représenter la donnée stockée. Pour générer cette matrice nous avons besoin de la version V du QR-Code pour pouvoir calculer sa taille, en effet sa largeur et sa longueur seront égales à (((V-1)*4)+21).

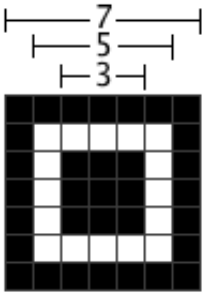
Tous les QR-Codes possèdent les mêmes éléments, seuls leurs emplacements diffèrent en fonction de taille et donc de leur version. On peut retrouver alors 4 paternes : les paternes de détection de position, les séparateurs, les "timings patterns" et les paternes d'alignements.



Structure d'un QR code

4.1 Paternes de détection de position et les séparateurs

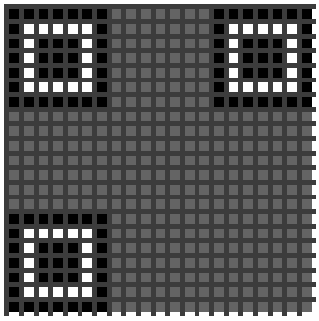
Les paternes de détection de position sont les 3 gros carrés visibles très facilement sur un QR-Code et ils sont au nombre de 3. Un paterne se compose d'un carré noir extérieur de 7 pixels sur 7 pixels, d'un carré blanc intérieur de 5 pixels sur 5 pixels et d'un carré noir uni au centre de 3 pixels sur 3 pixels.



Dimension d'un paterne de détection de position

Leurs positions sont calculées en fonction de la longueur de la matrice nommée "size" : Pour le premier, le coin supérieur gauche se situe en (0,0),

pour le deuxième en (size-7,0) et pour le dernier en (0, size-7). Ils permettent au lecteur de reconnaître le code avec précision et de lire avec rapidité les informations contenues dans le QR-Code. Ils indiquent aussi l'orientation de la structure.

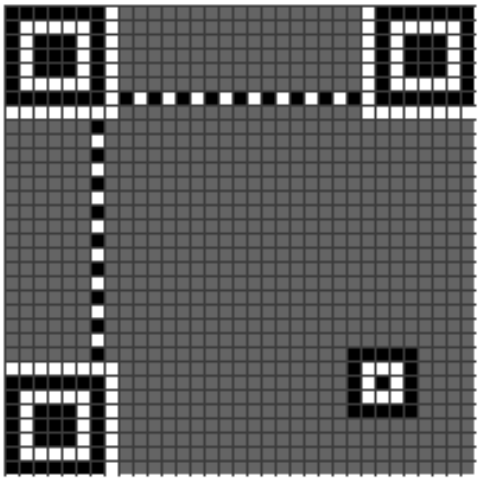


Localisations des paternes de détection de position

Pour ce qui est des séparateurs, se sont des lignes blanches qui se situent sur les bords exterieurs des paternes précedents, ils permettent comme leur nom l'indique de séparer la donnée que contient l'image aux paternes.

4.2 Les "timing Patterns"

Ils sont au nombre de deux, ils se composent de deux lignes avec une alternance de couleur partant du coin inférieur droit du paterne de détection de position haut gauche, jusqu'aux deux autres paternes. L'une est positionnée sur la 6-ème ligne et l'autre sur la 6-ème colonne de la matrice, de plus elles commencent et finissent toujours par un pixel noir. Ces lignes permettant au lecteur de déterminer facilement la taille du QR-code et donc d'obtenir la version de celui-ci.



Localisation des "timing patterns"

4.3 Paternes d'alignements

Tout comme les paternes de détection de position, ce sont des carrés qui permettent de bien cadrer l'image du QR de faciliter sa lecture même quand l'image est positionnée sur une surface qui n'est pas plate. Un paterne d'alignement se compose d'un carré noir de 5 pixels sur 5 pixels, d'un carré blanc intérieur de 3 pixels sur 3 pixels et d'un seul pixel noir au centre.

Ils sont présents uniquement sur les QR-Codes de versions supérieures à 2 et leurs positions sont déterminées par une table qui pour chaque version attribue une liste de valeur possible pour les coordonnées (x, y) du centre de chaque paterne. Une fois que nous avons la liste de ces valeurs il faut les tester une par une pour voir s'il est possible de placer le carré dans la matrice sans empiéter sur un paterne de détection.

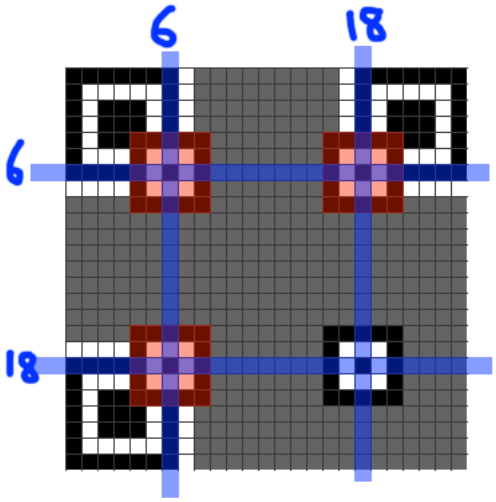


Illustration des coordonées possible des paternes d'alignement

Pour ce faire, j'ai créé un tableau à 2 dimensions qui pour un indexe V donné (correspondant la version du QR-Code) la liste des coordonnées (a,b) possibles. Via une boucle "for" je teste alors si chaque combinaison est possible avec cette condition :

```
if( mat[size_p*(a+2)+(b+2)] =='2' && mat[size_p*(a+2)+(b-2)] == '2' &&

mat[size_p*(a-2)+(b-2)] =='2' && mat[size_p*(a-2)+(b+2)] == '2')
    _align(a, b, mat, size_p);
```

Condition pour placer un paterne avec comme centre (a,b)

Ici, à partir des coordonnés du centre du carré, je vérifie si les 4 coins du carré sont bien disponibles et donc si je peux appeler la fonction "align" pour placer le paterne.

5 Chargement de l'image

Pour réaliser la lecture de QR code, il faut d'abord charger cette image que l'algorithme puisse charger l'image du QR code , et donc pouvoir après tout un process de traitement de l'image de pouvoir lire ce dernier. En effet, afin de charger une image plusieurs moyens sont possibles différent programme peuvent être réalise avec différent moyen, nous allons utiliser une méthode déjà vu à EPITA en utilisant la Bibliothèque SDL. Premièrement, nous avons une image dans le format que l'on souhaite nous n'avons plus qu'à appliquer le programme sur cette image. Voici l'image de base que nous souhaitons afficher via le programme que nous créons.

Photo Initial

Pour ce faire, nous initialisations les differentes variables pour afficher l'image tel que la texture, le rendu, la surface et la fenetre dont on precise la dimension sur cette fenetre lors de l'affichage de l'image.

```
SDL_Window *window = NULL;
SDL_Renderer *renderer = NULL;
SDL_Surface *picture = NULL;
SDL_Texture *texture = NULL;
SDL_Rect dest_rect = {0, 0, 640, 480};
```

Les différentes Variables

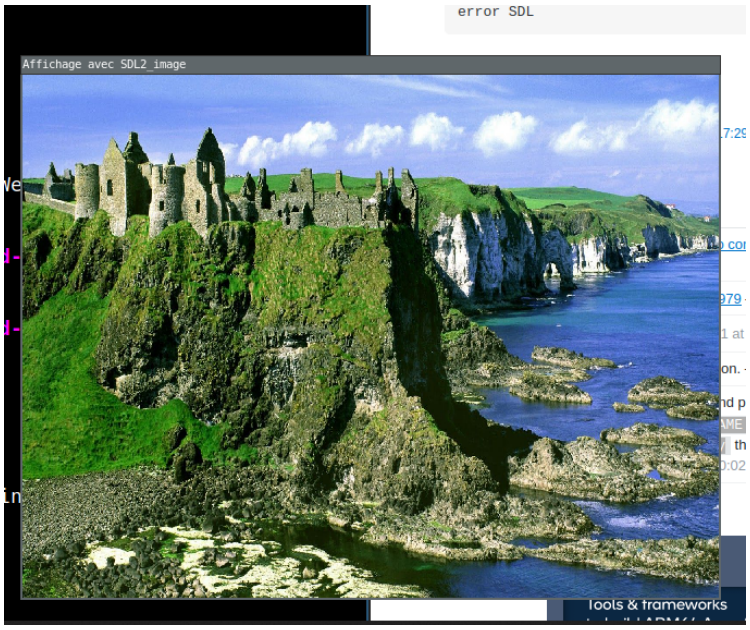
Après cela tout est simple, il suffit de traiter chaques variables et de traiter les possibles cas d'erreur et quand tous ces cas sont traités et donc que le code peut fonctionne, il suffit de copier le rendu puis de l'afficher. Pour rendre cela plus intéressant, on met un délai de 5 secondes lors de l'affichage de l'image.

```
SDL_RenderPresent(renderer);
SDL_Delay(5000);

clean_resources(window, renderer, texture);
return EXIT_SUCCESS;
```

La derniere étape

Et voici ce que donne le résultat final, après exécution du code de l'image initial.



Affichage de L'image

6 Site Web

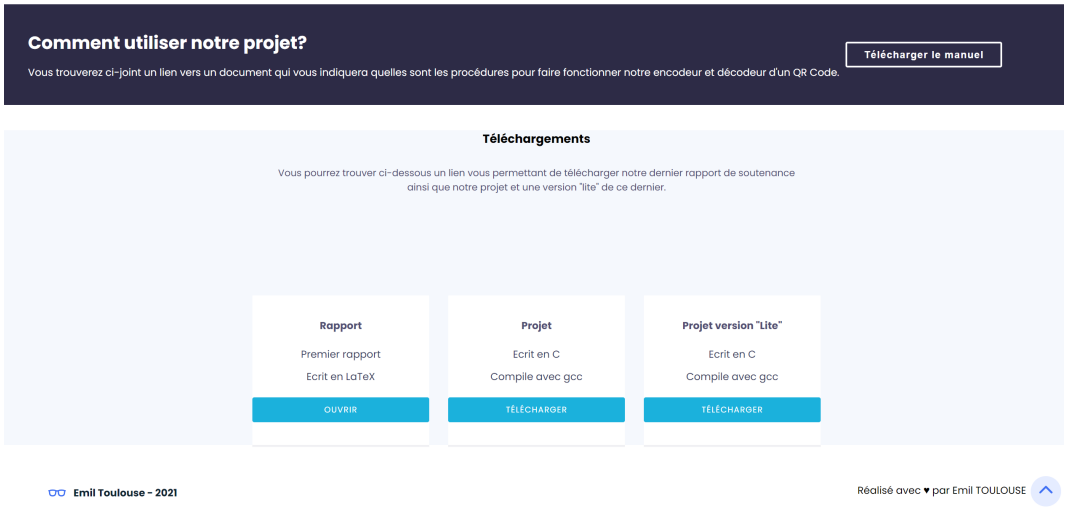
Pour le site web nous avons créer un site avec HTML CSS et JS.

Le HTML forment la structure de la page, et CSS ainsi que JS on pour objectif de rendre la page jolie et dynamique. Le site actuellement n'est pas en ligne mais dans une version locale. Il aura pour but a terme d'être en ligne avec tout les lien pour le téléchargment documentation et support potentiel.

Le but est de faire en sorte que des gens puissent utiliser notre code pour de vrai. Voici des apercu du site acutellement.

Page d'accueil

La chronologie du projet



Possibilité de télécharger

7 Organisation du projet

7.1 Tableau de répartition des tâches

	Emil	Adam	Samy
Chargement d'image			⊕
Pretraitement			⊕
Decodage du QR Code		⊕	
Encodage du QR Code	⊕		
Création d'une image résultat	⊕	⊕	
Sauvegarde et chargement d'image		⊕	
Interface graphique	⊕	⊕	⊕
Site Web	⊕		

7.2 Planning prévisionnel

	Soutenance 1	Soutenance 2	Soutenance 3
Chargement d'image	100%	100%	100%
Prétraitement	50%	100%	100%
Décodage du QR Code	0%	50%	100%
Encodage du QR Code	75%	100%	100%
Création de l'image résultat	25%	100%	100%
Sauvegarde et chargement d'une image	0%	50%	100%
Interface graphique	0%	25%	100%
Site Web	50%	75%	100%

7.3 Ressenti personnel

7.3.1 Adam Mahraoui

Pour ma part j'ai trouvé cette première partie très intéressante. J'ai beaucoup aimé la manipulation de matrice pour pouvoir créer la base d'un QR-Code qui peut s'adapter à toutes les versions possibles. De base ce n'était pas du tout ma partie, mais avec le départ d'un membre du groupe j'ai dû arrêter mes recherches sur le décodage et tout ce qui touche à l'image pour pouvoir avancer au maximum sur l'objectif qu'on s'était fixé par rapport à l'encodage du QR-Code. Mon objectif pour la prochaine soutenance est de réussir à remplir une matrice avec toutes les données possibles d'un QR-code pour qu'elle puisse être transformée en une image.

7.3.2 Emil Toulouse

J'ai trouvé ce début de projet très intéressant. Entre la creation du projet, les idées qui en sont sorties, les avancements compliqués et l'evolution du groupe, j'ai beaucoup aimé me pencher sur ce qu'etait un QR-code. Avec le départ d'un membre du groupe nous avons dut remanier les roles et voir à la baisse certains objectif. Mon objectif pour la prochaine soutenance est de réussir à finir l'encodage des données et creer un réel résultat (image fonctionnel). J'avancerais sur le site dans ce meme lapse de temps.

7.3.3 Samy Achache

Ce projet pour ma part est très intéressant, en effet l'avantage de ce dernier est de choisir son sujet on s'est mis d'accord sur un sujet qui nous intéressait beaucoup la lecture de QR code qui depuis peu nous entoure quotidiennement dans notre vie. Le fait qu'un membre du groupe parte vu qu'il effectue son S4 à l'étranger, a été certes un inconvénient et une difficulté en plus mais cela nous a permis de mieux nous souder et de nous répartir les nouvelles taches. Pour cette première soutenance nous somme complètements opérationnels et j'espère que nous allons réussir à remplir tous nos différents objectif fixe pour les soutenances futures. Mon objectif pour la prochaine soutenance est de réussir à finir le traitement de l'image du QR code.

8 Conclusion

Bien que nous nous ne connaissions pas entre nous et que nous travaillons pour la première fois en groupe une cohésion et forme d'entraide entre chaque membre a été vite réalise. Une bonne entente règne dans ce groupe et nos objectifs ont été réalisés. Tout au long de ce premier rapport de soutenance, nous vous avons déroulé les avancées de notre projet, point par point, de manière à expliciter chaque partie et faire un point objectif sur ce qu'il nous reste à faire, ce que nous avons déjà réussi à faire et les premiers rendus de ce défi que nous nous sommes posé. Il nous reste du chemin à parcourir, mais nous avons réussi à avancer comme nous le souhaitions sans nous mettre en retard, et nous sommes sur la bonne voie pour réaliser le meilleur projet possible avec les échéances que nous avons à respecter et celles que nous nous sommes imposées. La communication au sein du groupe est fluide, claire et efficace, de manière à ce que chacun puisse avancer au mieux sur son travail tout en ayant les dernières informations sur l'avancée des autres. Le travail paye toujours !