

# RAPPORT DE SOUTENANCE

## *EpiReader*



Groupe De Projet :  
CAZALE Lylian | HASS Léo | RAILLARD Victor | TAUR Bastien

Juin 2022

## **Sommaire :**

### **1) Introduction**

- a) Répartition des tâches**
- b) Fonctionnement d'un QR Code**

### **2) Avancement du Projet**

- a) Interface**
- b) Création**
- c) Lecture**
- d) Site Web**

### **3) Objectifs pour la dernière soutenance**

### **4) Conclusion**

- a) Impressions**
- b) Retard**

## **1) Introduction**

Nous allons vous expliquer en quelques lignes en quoi consiste le projet Epi Reader. Dans le cadre de notre projet de S4#; nous avons dû réfléchir à un sujet de travail dans le domaine de l'informatique. Après de multiples réflexions, nous avons eu l'idée d'axer notre projet sur la manipulation de QR codes.

Les Qr codes étant de plus en plus présents dans notre quotidien nous avons jugé intéressant d'y porter attention. C'est donc pour cela que notre projet sera une application du nom d'Epi Reader qui permettra à son utilisateur de créer des Qr codes qui mèneront par exemple à des liens internet juste en les scannant avec votre téléphone.

L'utilisateur aura aussi la possibilité de déchiffrer un QR code c'est-à-dire les informations que celui-ci contient. Par exemple, si l'utilisateur choisi de déchiffrer un QR code contenant un lien internet, l'application lui retournera ce lien. Nous espérons avoir été clairs et nous vous souhaitons de passer du bon temps sur Epi Reader !!

### **a) Répartition des Tâches**

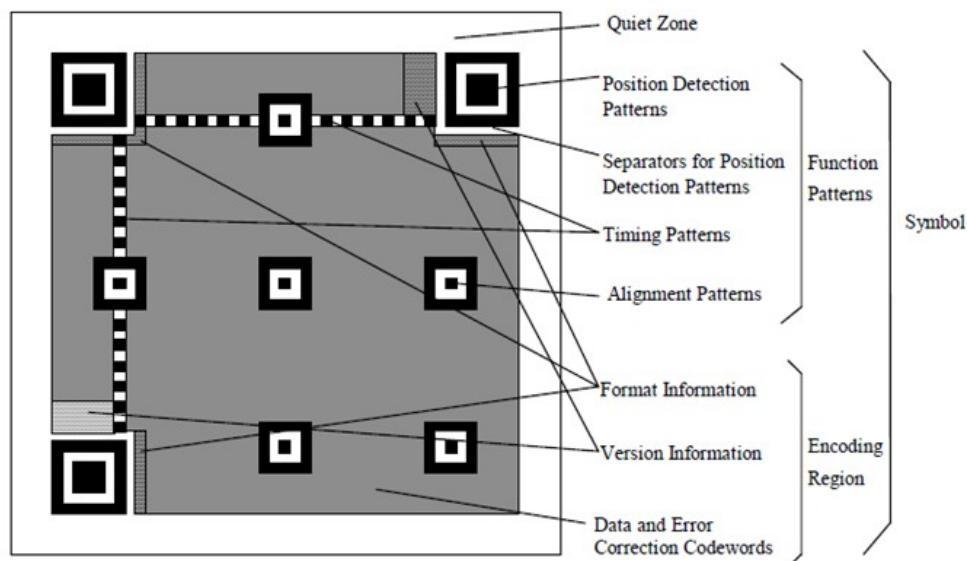
	Lylian Cazale	Léo Hass	Victor Raillard	Bastien Taur
Site Web				
Lecture				
Création				
Application				

Acteur Principal	Acteur Secondaire

## b) Fonctionnement d'un QR code

Les QR codes sont donc des manières de stocker des informations de manière visuelle en deux dimensions. Chaque alternance de carré blanc et noir forme un code binaire qui permet de stocker des informations.

Il existe plusieurs tailles de QR code, par exemple, pour un QR code de 2cm de côté, on va pouvoir stocker jusqu'à 7089 caractères numériques ou 4253 caractères en ASCII, ce qui donne 2.953 Ko.



Les 3 gros carrés dans les coins sont des repères qui gèrent l'alignement pour faciliter la lecture et les six autres petits carrés visibles sont des régulateurs pour faire en sorte que le code puisse être lu sans problème. La grande zone du centre sert à stocker les informations. Alors que les zones en haut et à gauche servent à éviter les erreurs de lecture et donner des informations sur le type d'information que nous avons ici, comme la manière dont sont organisées les informations ou encore comment elles sont codées. Suite à cela, la lecture est assez simple, Blanc = 0 et Noir = 1.



Il est aussi possible d'ajouter une image au milieu d'un QR code. Ce que nous allons faire pour un rendu graphique meilleur

## **2) Avancement du projet**

### **a) Interface :**

Il est maintenant l'heure de nous attaquer à la création de notre interface ou plutôt de notre application. Avant tout un rapide rappel sur l'objectif de cette dernière, notre projet a pour but la création et la lecture de QR code, notre application aura donc pour objectif d'être simple d'utilisation et efficace pour permettre une utilisation suffisamment rapide pour que l'utilisation de QR code soit viable.

Quels sont donc les différents éléments attendus dans notre application :

- Un espace pour entrer le lien internet vers lequel l'utilisateur souhaite créer son QR code
- Un moyen de charger des fichiers, car notre application étant sur un ordinateur il nous sera impossible de lire des images, nous allons donc créer un emplacement dans lequel nous pouvons charger des fichiers puis ensuite une fois le fichier chargé nous rediriger vers l'utilisateur.

Ce sont globalement les deux axes principaux de notre application, la contrainte principale restant tout de même de créer quelque chose de simple d'accès et d'utilisation pour ainsi donner une impression de fluidité à l'utilisateur et rendre notre application utile.

L'esthétique joue donc un rôle plutôt important dans sa création, malheureusement nous ne sommes pas des experts dans l'utilisation de la librairie GTK, c'est donc un problème sur lequel nous nous pencherons une fois l'application terminée ou presque pour au moins avoir une interface utilisable. Si nous devons donc résumer les besoins de notre application, il nous faut une application épurée avec un nombre de fonctionnalités qui n'est pas trop important. C'est pourquoi nous avons eu le temps d'effectuer un grand nombre de tests et nous sommes donnés le luxe de vraiment prendre le temps de prendre la librairie en main.

A l'heure actuelle nous n'avons pas d'application réellement fonctionnelle, seulement des ébauches des différentes fonctionnalités qui devront être présentes dans notre application finale telle que :

- la possibilité de taper du texte, pour l'instant le seul effet étant de recopier ce texte dans une partie inférieure de la fenêtre de l'application. Mais à l'avenir nous pourrons, grâce à un procédé similaire, créer un QR code depuis le texte entré.
- La possibilité de charger des fichiers, pour l'instant les boutons sont inactifs mais nous intégrerons plus tard le chargement du fichier ainsi que l'appel du programme permettant sa lecture et la redirection de l'utilisateur sur le site internet.

Je développerai ces deux fonctionnalités plus tard ainsi que leur mise en place.

Avant d'arriver à ce résultat qui n'est certes pas suffisant, loin de là si l'on considère nos objectifs, nous avons dû en apprendre beaucoup sur cette nouvelle librairie qu'est GTK. La librairie GTK permet la création d'interfaces graphiques avec une grande liberté, son utilisation peut se faire dans différents langages, nous nous sommes donc contentés de l'utiliser en C. Ses utilisations sont diverses et variées et elle est très prisée dans la création d'interface de par ses possibilités couplées à la liberté qu'elle offre.

Il s'agissait de la première fois que nous nous apprêtons à utiliser GTK de façon aussi poussée, la prise en main a donc dû se faire petit à petit. Heureusement grâce aux nombreuses informations présentes sur cette librairie nous avons pu apprendre bien des choses en lisant de la documentation ou bien en nous aidant de tutoriels expliquant les techniques de base. La première étape étant la création de fenêtres.

Petit point vocabulaire :

Il faut savoir que lorsque nous développons des applications dans GTK les objets que nous allons placer dans nos fenêtres sont appelés WIDGET

mot qui n'a pas de traduction littérale et les traductions françaises de ce mot n'étant pas pertinentes nous utiliserons ce mot pour en parler.

Concernant maintenant le fonctionnement des différentes fonctions que nous allons appeler dans notre code, les noms des fonctions ont une syntaxe particulière, ils sont de la forme **gtk\_widget\_action(...)**. C'est cela dans la plupart des cas avec GTK.

On remplace :

- o widget par le type de widget sur lequel on travaille (ex : window pour une fenêtre).
- o action parce que la fonction est censée effectuer (ex : set\_title pour définir un titre)

Voilà quelques informations qui simplifieront la compréhension des différentes explications qui vont suivre. Voici tout de même un exemple :

```

#include <stdlib.h>
#include <gtk/gtk.h>

int main(int argc, char **argv)
{
    GtkWidget * MainWindow = NULL;

    gtk_init(&argc, &argv);

    MainWindow = gtk_window_new(GTK_WINDOW_TOPLEVEL);
    g_signal_connect(G_OBJECT(MainWindow), "delete-event", G_CALLBACK(gtk_main_quit), NULL);

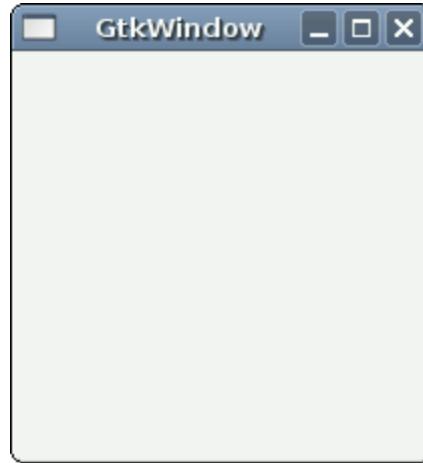
    gtk_widget_show(MainWindow);
    gtk_main();

    return EXIT_SUCCESS;
}

```

Sur cette image nous pouvons voir une fonction qui va tout simplement créer une fenêtre, fenêtre sur laquelle nous allons ajouter nos différents widgets et ainsi pouvoir créer notre interface.

Nous n'allons pas revenir en détail sur la totalité des fonctions présentent dans ces lignes de code, mais l'idée principale est de créer une fenêtre et de l'afficher ce qui est la première étape avant de commencer tout travail. Voici le résultat obtenu



Nous avons ici un exemple du concept des fonctions GTK expliqué plus haut avec la fonction `gtk_window_new(GTK_WINDOW_TOPLEVEL)`, cette fonction ayant pour but de créer une fenêtre ( le « `GTK_WINDOW_TOPLEVEL` » )

signifiant simplement qu'il s'agit d'une fenêtre normale. Le but de ces explications étant simplement de montrer un petit exemple de ce qu'il est possible de faire , nous ne nous pencherons pas sur la totalité des fonctions que nous allons utiliser, tout d'abord car cela nous prendrait beaucoup trop de temps mais cela serait également redondant et inutile car chaque fonction ayant la même syntaxe.

Nous allons directement passer à la première fonctionnalité dont nous avons parlé plus haut :

### La possibilité de taper du texte



Actuellement tout ce que nous pouvons faire c'est faire apparaître du texte, mais à l'avenir il nous sera assez simple de combiner tout cela avec notre programme créant les QR codes pour que ce texte plutôt que d'être affiché soit envoyé directement vers notre programme et qu'à la place du retour de texte nous ayant un PNG avec notre QR code nouvellement créé.

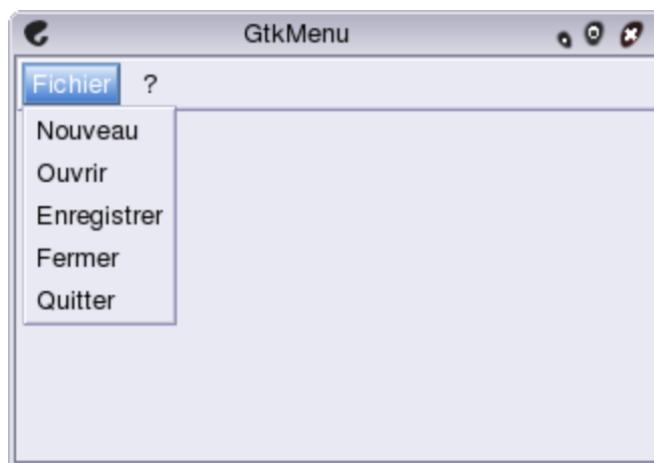
Quelques rapides explications :

Notre code n'est pas des plus complexes ici, il est composé d'une fonction main s'occupant de la création des différents widgets intervenant dans le processus, d'une fonction qui s'activera lorsque l'on rentre le texte pour récupérer ce dernier et enfin une dernière qui aura pour but d'afficher le texte. Globalement le code est tout simplement composé d'un enchaînement de commandes ayant pour but d'effectuer les actions voulues, nous n'avons pas affaire ici à des manipulations de compteur difficile ou quoi que ce soit

demandant une intense réflexion. Le plus difficile dans la mise en place de cette partie de l'interface à été le fait de devoir appeler des fonctions ayant des effets sur l'interface et modifiant ce dernier, il a donc fallu utiliser des fonctions appelées « fonction de rappel »(ou callback en anglais) effectuant les différentes actions, les deux fonctions hors main répondant à ces critères.

Ce travail peut sembler anodin mais il s'agit déjà d'une grande avancée vers notre objectif car il ne nous reste plus qu'à faire en sorte que plutôt que de renvoyer du texte notre interface fasse appel à notre programme puis renvoie le PNG pour que la partie création de QR code soit complète.

*La possibilité de charger un fichier :*



Voici la dernière fonctionnalité que nous avons mise en place, ici elle n'est pas totalement fonctionnelle, son seul intérêt est de nous donner accès au menu déroulant où nous pourrons plus tard charger votre QR code pour être redirigé vers le lien internet.

```
pMenuItem = gtk_menu_item_new_with_label("Ouvrir");
gtk_menu_shell_append(GTK_MENU_SHELL(pMenu), pMenuItem);
```

Voici un exemple de comment nous allons créer les différents éléments présents dans notre application, le bouton « ouvrir » servira à l'avenir à charger notre image qui sera envoyée vers notre programme. Le bouton « quitter » est déjà fonctionnel et permet de sortir et de fermer la fenêtre. Pour ce faire nous avons utilisé une fonction « Quit » qui est appelée lors de la

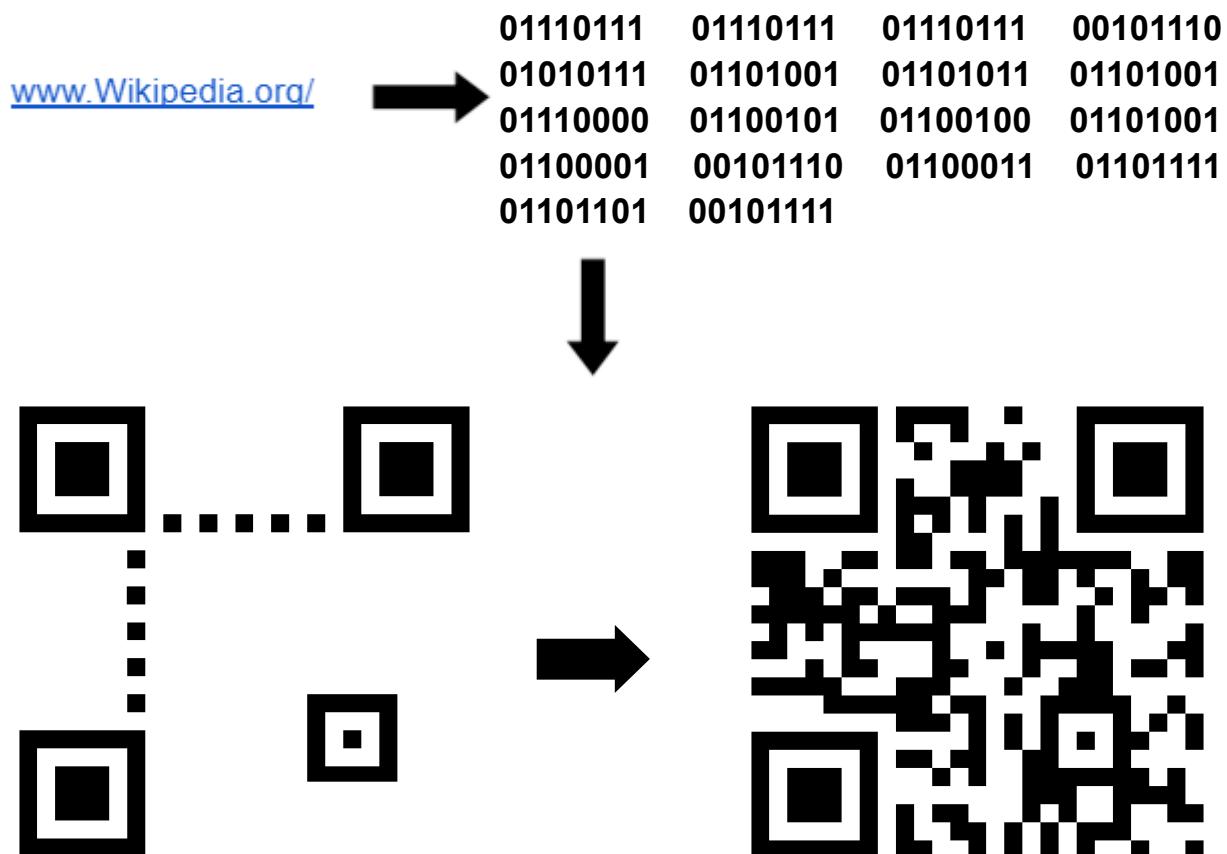
pression du bouton « quitter » et demandant à l'utilisateur si ce dernier est sûr de vouloir quitter en affichant à nouveau des boutons « yes/no ». Nous ne rentrerons pas dans les détails du fonctionnement des différentes fonctions car faisant appel à grand nombre de fonctions présentes dans GTK, il serait très long de revenir sur chacune d'elles. Mais dans l'idée lorsque nous voudrons charger notre image nous utiliserons le même fonctionnement que pour le bouton « quitter », nous devrons créer une fonction qui effectue la tâche de charger le fichier et de l'envoyer vers le programme que nous aurons créé et qui nous redirigera vers le site internet.

Voici globalement l'état d'avancement de notre application avant cette première soutenance, nous détaillerons nos objectifs concernant cette dernière pour la dernière soutenance plus bas.

## b)Création du QR code :

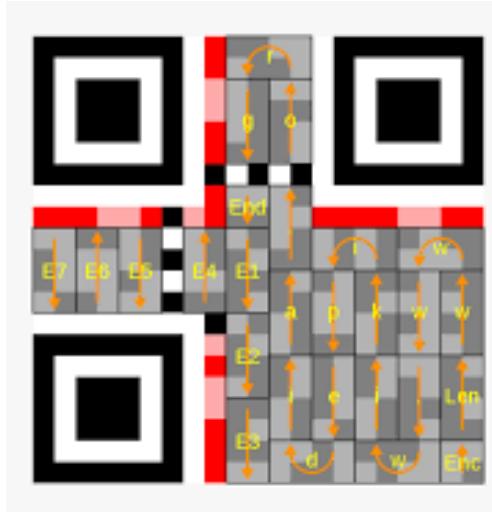
Pour la Création, nous avons dû commencer par comprendre comment était construit un QR code, et surtout comment est-ce qu'on devait lire un QR code pour pouvoir s'adapter et pouvoir en construire un. Pour cela certains marqueurs sont présents dans le QR code lui-même : des zones noires sont présentes à certains endroits pour indiquer certaines informations, comme la nature de l'information(site, .txt, code Wi-Fi ...) la quantité d'informations, le taux d'erreur...

La manière que nous avons choisie et qui nous semble la plus simple est de partir d'un QR code vierge, avec simplement les marqueurs de position et de taille, pour ensuite venir y inscrire les informations que l'utilisateur demande. Pour cela il nous a fallu encoder le (par exemple) Site Web demandé par l'utilisateur en le transformant en code binaire. Ce qui nous permet de passer de ça à ça :



### c) Lecture du QR code :

Pour la lecture du Qr Code nous avons pris une matrice représentant un Qr Code avec un pour chaque case, un 1 pour une cellule noire et un 0 pour une cellule blanche. Nous en avons ensuite extrait les éléments importants tels que le masque utilisé, le niveau et le format de correction d'erreurs. Après avoir isolé l'information contenant le masque utilisé, nous pouvons désormais appliquer ce masque pour faire apparaître les informations contenues dans le Qr Code. Ensuite, on transforme toute l'information contenue dans le QR code en liste en suivant le parcours décrit dans l'image ci-dessous:



```
{1,1,1,1,1,1,0,0,0,0,0,0,1,1,1,1,1,1},  
{1,0,0,0,0,0,0,0,1,1,1,0,1,0,0,0,0,1},  
{1,0,1,1,1,0,0,0,0,0,0,0,1,0,1,1,1,0,1},  
{1,0,1,1,1,0,0,0,0,0,0,1,0,1,0,1,1,1,0,1},  
{1,0,1,1,1,0,0,0,1,0,0,1,0,1,0,1,1,1,0,1},  
{1,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,1},  
{0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0},  
{0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},  
{1,0,1,1,1,1,1,0,0,0,0,1,0,1,1,1,0,1},  
{0,0,1,0,1,0,0,0,0,1,1,0,1,0,1,1,1,0},  
{1,0,1,0,1,0,0,0,1,1,0,1,0,0,0,1,0,0,1},  
{1,0,1,0,1,0,0,1,1,0,1,0,0,0,1,0,0,0,1},  
{1,1,1,0,1,1,1,0,1,1,0,1,1,0,0,1,0,0,0},  
{0,0,0,0,0,0,1,0,1,1,0,0,0,1,0,0,1,0},  
{1,1,1,1,1,1,0,0,0,1,0,0,1,1,1,0,0,1},  
{1,0,0,0,0,0,0,0,1,1,0,1,0,1,0,1,0,0,1},  
{1,0,1,1,1,0,0,0,1,0,0,0,0,0,1,1,1,0,0},  
{1,0,1,1,1,0,0,1,1,1,1,1,1,0,1,1,0,0,0},  
{1,0,0,0,0,0,0,0,0,1,0,0,1,0,1,0,0,0,0},  
{1,1,1,1,1,1,0,0,1,1,1,0,1,0,1,1,0,1,0}
```

```
0100000010100101001001101111011000100110111101101101100001011101000110100101100011011  
1001100001110110000010001111011000001111010001100001001100000111010101010011100  
11110000111110111010101110100011
```

Enc: 4

Length: 9

Message : Epireader

Après avoir fait cette action, il est maintenant facile d'extraire le type d'encodage (code ASCII, alphanumérique, binaire ou des kanjis) sur les quatre premiers bits de la liste obtenue précédemment, sur l'exemple ci-dessus, le type d'encodage est le 4 donc en code ASCII. Puis la longueur du message sur les huit bits suivants (9 caractères dans l'exemple), et enfin le message lui-même (“Epireader” dans l'exemple). Les bits restants représentent la fin du message puis les bits de correction d'erreurs.

## Site Web:

Dans cette partie de notre rapport nous allons parler du site internet de notre projet. Celui-ci étant obligatoire pour toutes les soutenances. Nous avons choisi de le faire complètement pour la première soutenance car nous pourrons, par la suite plus nous concentrer sur des choses plus difficiles du projet.

Afin de créer le site web de notre projet, nous avons utilisé l'éditeur de texte Visual Studio Code et nous avons dû avoir recours à différents tutoriels pour apprendre à coder en HTML et en CSS. Ces tutoriels nous auront été très utiles et bénéfiques pour créer le site.



Nous avons donc commencé par créer la structure de notre site via le langage html.

```
1  <!DOCTYPE html>
2  <html lang="fr">
3      <head>
4          <meta charset="UTF-8">
5
6          <title>Epi Reader</title>
7          | <!--<link rel="stylesheet" href="style.css"> -->
8          <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.css">
9
10     </head>
11     <body>
12         <header>
13             <nav>
14                 <ul>
15                     <!--Pour faire un lien-->
16                     <li id = "logo"><a href="#">Epi Reader</a></li>
17                     <li><a href="#archives">Archives et Ressources</a></li>
18                     <li><a href="#Le projet">Présentation du projet</a></li>
19                     <li><a href="#téléchargement du projet">Téléchargement</a></li>
20
21             </ul>
22         </nav>
```

Nous avons commencé par créer le titre de notre site (ligne 6) et nous avons par suite créer le squelette de la barre de navigation (ligne 13 à 21) notamment via la ligne de code “a href = #....”. Cette ligne permet la création d'un lien qui nous permettra donc de naviguer selon nos envies sur le site.

- [Epi Reader](#)
- [Archives et Ressources](#)
- [Présentation du projet](#)
- [Téléchargement](#)

## Epi Reader

Nous avons ici le titre du site ainsi que le squelette de la barre de navigation.

Nous nous sommes ensuite attaqués à la partie présentation de notre projet accessible soit en scrollant sur le site, soit en cliquant sur “Présentation du projet”.

Nous avons commencé par créer un peu de texte pour expliquer notre création. Tout cela grâce aux balises de section “`<h>`” (titre) et des balises “`<p>`” (paragraphe).

A noter qu'il existe 6 types de balise “`<h>`” numérotés de 1 à 6: h1,h2,h3... Plus la balise a une numération élevée, plus le texte va être petit. Nous avons donc fait plusieurs tests jusqu'à arriver à ce qu'il nous convenait.

### **Epi Reader qu'est-ce que c'est ?**

Lorem ipsum dolor sit amet consectetur adipisicing elit. Error at similique tenetur atque blanditiis odit! Numquam rerum maiores laborum, error eveniet corrupti quidem officiis sed suscipit iure ipsam excepturi consequatur quis non similique esse, quod reprehenderit consectetur. Vel reiciendis quidem dicta laudantium consequatur ipsum quos, impedit repellendus iste similique, explicabo alias distinctio aut molestiae beatae? Suscipit, ratione. Voluptas iusto quos iste odio. Harum maiores molestias ipsa itaque esse cupiditate amet quasi, asperiores minima dolor quos quas odit numquam, laborum eum!

### **Chronologie de notre projet**

Lorem ipsum dolor sit amet consectetur, adipisicing elit. Cupiditate, minima molestiae. Fuga facilis eius eum, tenetur autem quam assumenda culpa dolorum expedita minima mollitia ipsa quae. Inventore aperiam cumque nihil? Cum velit nulla in, aliquam id possimus libero, similique repudiandae deserunt nemo, magnam sequi earum soluta repellendus. Sit ipsam nam numquam. Optio, quis amet perspiciat eveniet accusamus expedita nulla et soluta repellendus, in blanditiis commodi quos, consectetur dolorem itaque quia voluptate. Omnis, cupiditate? Alias aliquid possimus qui suscipit corrupti perspiciat.

### **Problèmes rencontrés**

Lorem ipsum dolor sit amet consectetur, adipisicing elit. Cupiditate, minima molestiae. Fuga facilis eius eum, tenetur autem quam assumenda culpa dolorum expedita minima mollitia ipsa quae. Inventore aperiam cumque nihil? Cum velit nulla in, aliquam id possimus libero, similique repudiandae deserunt nemo, magnam sequi earum soluta repellendus. Sit ipsam nam numquam. Optio, quis amet perspiciat eveniet accusamus expedita nulla et soluta repellendus, in blanditiis commodi quos, consectetur dolorem itaque quia voluptate. Omnis, cupiditate? Alias aliquid possimus qui suscipit corrupti perspiciat.

Voici le résultat. Par ailleurs, les lignes en latin que vous observerez seront bien entendues remplacées par les explications adéquates.

Nous avons continué avec la partie photographiée des membres.

```
</div>
<div id="quinousommes">
    <div class = "imagesdesmembres">
        <h4>Victor Raillard</h4>
        <a href="#"></a>
    </div>
    <div class = "imagesdesmembres">
        <h4>Bastien Taur</h4>
        <a href="#"></a>
    </div>
    <div class = "imagesdesmembres">
        <h4>Lylian Cazale</h4>
        <a href="#"></a>
    </div>
    <div class = "imagesdesmembres">
        <h4>Léo Hass</h4>
        <a href="#"></a>
    </div>
</div>
```

Nous avons ici utilisé <a href = "#> <img src="" alt= "">. le "img src" permet de faire référence à une image choisie et le "alt" permet une solution alternative dans le cas où le "img src" est absente.

## Les membres du groupe

### Victor Raillard



(ici nous avons indiqué victor.PNG dans le “img src” donc l’image s’affiche bien)

```
<div class = "imagesdesmembres">
    <h4>Victor Raillard</h4>
    <a href="#"><img src="" alt="Victor Raillard"></a>
</div>
```

Ici le “img src” n’est pas renseignée donc c’est le “alt” qui prend le relais (voir image ci-dessous).

## Les membres du groupe

### Victor Raillard



Nous avons ensuite continué avec la partie téléchargement de notre projet. Nous nous sommes servis de la ligne de code “`<button><button>`” associée à un lien, ainsi quand l’utilisateur cliquera sur le bouton “Télécharger le projet” il télécharge le fichier que nous aurons inséré au préalable. Nous avons eu aussi l’idée de créer une sorte de tuto en image pour télécharger notre projet. Nous avons inséré les images via le code Css que nous verrons plus tard. (Pour les images des membres nous avions inséré les images directement via le langage html mais nous pouvons aussi le faire via le langage Css).

```
<section id = "telechargement">
    <h2 id="téléchargement du projet">Cliquer ci-dessous pour télécharger</h2>
    <a href="saucisse.txt"><button> Télécharger le projet</button></a>
    <ul>
        <li id = "tuto1"><p>1ère etape</p></li>
        <li id = "tuto2"><p>2ème etape</p></li>
        <li id = "tuto3"><p>3ème etape</p></li>
    </ul>
</section>
```

# Cliquer ci-dessous pour télécharger

[Télécharger le projet](#)

- 1ère etape
- 2ème etape
- 3ème etape

(voici le résultat)

Nous avons continué notre site en nous penchant sur la partie archives et ressources. La partie archive contient les différents rapports de soutenances et la partie ressources contient tous les outils utilisés pour mener à bien notre projet.

Pour la partie archive, comme la partie téléchargement du projet nous avons utilisé la ligne de code “<button><button>” en lien vers nos rapports de soutenances.

Pour la partie ressources nous avons encore utilisé des balises “<h><h>” et des balises “<p><p>”.

## Archives et Ressources

[Rapport de première soutenance](#) [Rapport de deuxième soutenance](#)

### Concernant le site

Nous avons créé notre site à l'aide du langage HTML pour la structure et du langage CSS pour le côté artistique ainsi que du support Visual Studio Code. Notre site est hébergé sur la plate-forme "Git Hub". Celui-ci permettant d'héberger notre site facilement et gratuitement. Nous nous également servi de tutos Youtube et d'article sur google.

### Concernant l'application

### Concernant le créateur de QR code

### Concernant l'interpréteur

La partie “Ressources” sera bien entendue compléter à notre guise.

Nous allons à présent attaquer la dernière partie de notre site internet, la partie copyright et réseaux sociaux.

```
<div id = "deuxiemeTrait"></div>
<div id="copyrightEtIcons">
    <div id = "copyright">
        <span>© fromEpiReader(); 2022</span>
    </div>
    <div id = "icons">
        <a href="http://www.instagram.com"><i class="fa fa-instagram"></i></a>
        <a href="http://www.facebook.fr"><i class="fa fa-facebook"></i></a>
        <a href="http://www.twitter.fr"><i class="fa fa-twitter"></i></a>
    </div>
</div>
```

Nous avons utilisé la ligne de code “`<span><span>`” pour créer notre copyright. A noter que nous avons créé ce copyright pour rendre le site un peu “officiel”.

Concernant les réseaux sociaux nous avons utilisé deux lignes de code bien spécifiques.

“`<a href=><a>`” et le lien du réseau social.

“`<i class=><i>`” accompagné du texte faisant référence au logo du réseau social associé.

Par exemple “`<i class="fa fa-twitter"></i>`” fera référence au logo de Twitter.

Ainsi voici le résultat après ces quelques lignes de code.

© fromEpiReader(); 2022  
  

Nous avons fini la partie HTML de notre site web nous allons pouvoir parler de la partie CSS qui va venir se greffer au langage HTML et ajouté un côté plus artistique à notre site.

Pour lier la partie HTML et CSS nous avons commencé par faire une référence dans le code HTML au code CSS.

```
<link rel="stylesheet" href="style.css">
```

Ainsi toutes nos lignes de code CSS viendront simplement s'ajouter aux lignes de code HTML.

Attaquons à présent le CSS de notre site.

Nous avons commencé par importer une police d'écriture qui nous convenait: le Dancing Script. Nous avons importé cette police via la ligne de code "@import url".

```
@import url('https://fonts.googleapis.com/css?family=Advent+Pro|Dancing+Script&display=swap');
```

Nous avons ensuite ajouté un effet "smooth" à notre site. Cela permet d'avoir un effet agréable quand on scroll avec la souris.

```
html {  
    scroll-behavior: smooth;  
}
```

Nous avons ensuite implémenté différents paramètres graphiques comme la taille de la police, la disposition des textes, des images, l'opacité, la couleur du fond et du texte, largeur, la hauteur, la marge...

```
nav {  
    overflow: hidden;  
    background-color: #FFF8E1;  
    position: fixed;  
    width: 100%;  
    opacity: 0.9;  
}
```

Voici un exemple du code CSS pour la barre de navigation, nous avons mis le paramètre “position” à “fixed” pour que la barre reste en haut de la page et s’affiche même si l’on est tout en bas du site. Nous avons mis la “width” à 100 % pour que la barre prenne toute la largeur de la page. Nous avons choisi une couleur qui nous convenait et de même pour l’opacité.

Nous avons donc modifié ces différents paramètres pour chaque partie du code html créé pour avoir le résultat attendu. Cela ne servirait à rien de vous montrer tout le code car ce sont des paramètres qui reviennent tout le temps et pour chaque partie du code HTML.

(Voici le site après les lignes de code CSS)



## EPI READER QU'EST-CE QUE C'EST ?

Lorem ipsum dolor sit amet consectetur adipisicing elit. Error et similique tenetur atque blanditiis odit! Numquam rerum maiores laborum, error eveniet corrupti quidem officiis sed suscipit iure ipsam excepturi consequatur quis non similique esse, quod reprehenderit consectetur. Vel reiciendis quidem dicta laudantium consequatur ipsum quos, impedit repellendus iste similique, explicabo alias distinctio aut molestiae beatae? Suscipit, ratione. Voluptas iusto quo iste odio Harum maiores molestiae ipsa itaque esse cupiditate amet quasi, asperiores minima dolor quo quis odit numquam, laborum eum!

## CHRONOLOGIE DE NOTRE PROJET

Lorem ipsum dolor sit amet consectetur adipisicing elit. Cupiditate, minima molestiae. Fuga facilis eius eum, tenetur autem quam assumenda culpa dolorum expedita minima mollitia ipsa quae. Inventore operam cumque nihil? Cum velit nulla in, aliquam id possimus libero, similique repudiandae deserunt nemo, magnam sequi eorum soluta repellendus. Sit ipsam nam numquam. Optio, quis amet perspicacis eveniet accusamus expedita nulla et soluta repellendus, in blanditiis commodi quos, consectetur dolorem itaque quia voluptate. Omnis, cupiditate? Alias aliquid possimus qui suscipit corrupti perspicacis.

## PROBLÈMES RENCONTRÉS

Lorem ipsum dolor sit amet consectetur adipisicing elit. Cupiditate, minima molestiae. Fuga facilis eius eum, tenetur autem quam assumenda culpa dolorum expedita minima mollitia ipsa quae. Inventore operam cumque nihil? Cum velit nulla in, aliquam id possimus libero, similique repudiandae deserunt nemo, magnam sequi eorum soluta repellendus. Sit ipsam nam numquam. Optio, quis amet perspicacis eveniet accusamus expedita nulla et soluta repellendus, in blanditiis commodi quos, consectetur dolorem itaque quia voluptate. Omnis, cupiditate? Alias aliquid possimus qui suscipit corrupti perspicacis.

## LES MEMBRES DU GROUPE

Victor Raillard



Bastien Taur



Lylian Cazale

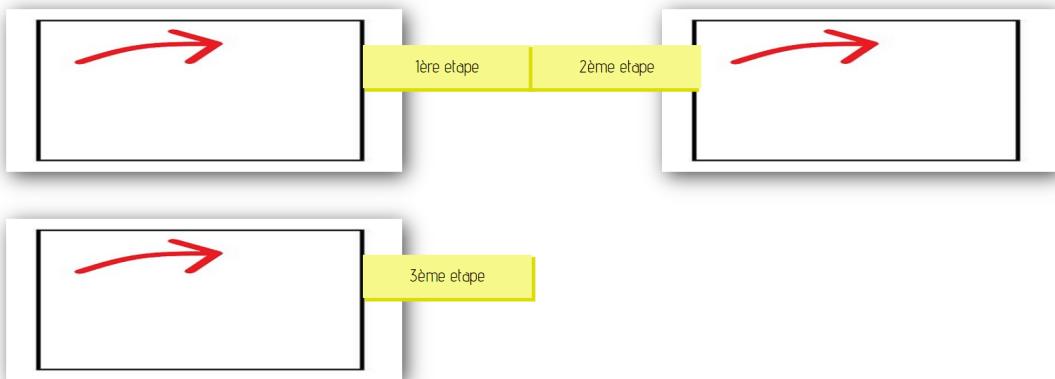


Léo Hass



CLIQUEZ CI-DESSOUS POUR TÉLÉCHARGER

Télécharger le projet



## ARCHIVES ET RESSOURCES

Rapport de première soutenance

Rapport de deuxième soutenance

## CONCERNANT LE SITE

Nous avons créé notre site à l'aide du langage HTML pour la structure et du langage CSS pour le côté esthétique ainsi que du support Visual Studio Code. Notre site est hébergé sur la plate-forme "Git Hub". Celui-ci permettant d'héberger notre site facilement et gratuitement. Nous nous sommes également servi de tutos YouTube et d'article sur google.

## CONCERNANT L'APPLICATION

### CONCERNANT LE CRÉATEUR DE QR CODE

### CONCERNANT L'INTERPRÉTEUR

© fromEPiReader[], 2022



Nous avons aussi rajouté des transformations visuelles lorsque l'on passe son curseur sur des boutons ou sur des images. (Voir ci-dessous)

## ARCHIVES ET RESSOURCES

Rapport de première soutenance

Rapport de deuxième soutenance

Ici nous avons mis notre curseur sur le bouton pour télécharger le premier rapport.

Victor Raillard



Bastien Taur



Lylian Cazale



Léo Hass



Ici la souris est sur la photo de Bastien Taur.

Nous avons terminé notre page web, il ne nous reste plus qu'à l'héberger.  
Pour cela nous nous sommes servis de la plate-forme Git Hub pour la simplicité et la gratuité de l'opération.

De plus, Hub est un outil qui nous est familier grâce aux nombreux projets effectués durant les années précédentes.

Voici le lien de notre site:

<https://petrolas.github.io/index.html>

### **3) Objectifs pour la dernière soutenance**

Tout d'abord nos objectifs vis-à-vis de notre interface, il faudra que cette dernière soit totalement fonctionnelle et que la totalité des fonctionnalités que nous avions prévu de mettre en place soient accessibles directement depuis cette application. Ce qui veut dire qu'il nous faudra finir de compléter les ébauches de fonctionnalités que nous avons déjà réussi à mettre en place jusqu'ici et également rassembler ces dernières dans une seule et même application pour faciliter l'utilisation et la rendre plus ergonomique. Dans un dernier temps nous allons tenter de rendre cette dernière plus agréable esthétiquement car actuellement nous n'avons fait aucune modification sur le plan esthétique, nous sommes restés sur l'apparence initiale fournie par GTK.

Du côté du lecteur de QR Code, il nous faut encore développer les outils permettant d'exploiter les bits de correction d'erreurs dans le cas où une partie du QR Code serait illisible. Il nous faudra aussi développer la reconnaissance d'un QR Code dans une photo et la transformation de ce QR Code en matrice utilisable par l'ordinateur.

Pour ce qui est de la création, il nous est à bien définir le changement de taille en fonction du message. Ensuite nous pourrons gérer la finalité en mettant en place une possibilité de changement de couleur, ainsi que la possibilité de mettre une image au centre du QR code. Cela signifie aussi que nous devrons trouver un moyen d'ignorer ces changements graphiques avec notre lecteur.

## **Conclusion:**

Au cours de cette première étape dans notre projet nous avons pris conscience des difficultés à travailler dans le thème des QR codes. Pour y pallier nous avons mobilisé les compétences de chacun des membres du groupe pour s'adapter aux aspects techniques du projet.

Il a été impossible de se voir de manière physique compte tenu des lieux de vie de chacun lié au semestre au distanciel mais nous avons réussi à communiquer via des applications telles que Discord ou Teams.

Toutefois les objectifs fixés pour cette première soutenance sont globalement atteints, nous sommes satisfaits du premier rendu et nous comptons remplir les objectifs fixés pour la soutenance finale car nous serons très heureux d'avoir un projet final fonctionnant comme nous l'avons voulu.

