ENSEIRB-MATMECA

RAPPORT DU PROJET C++

19 mai 2016

Bibliothèque Numérique

Bastien SÉGOT
Anas ALAOUI M'HAMDI

ENCADRÉS PAR : MR. JÉRÉMIE CRENNE

Table des matières

Introduction **3**

1 Cahier des charges **3**

2 Diagramme de classes **3**

3 Attributs des classes - Commandes **6**
3.1 Fonctionnement du main 6
3.2 BYE : 6
3.3 L'ajout : la fonction ADD {type} 7
3.4 LOAD filename : 7
3.5 SAVE filename 8
3.6 SEARCH chaîne 8
3.7 CLEAR 8
3.8 LIST 9
3.9 SHOW {id} 9
3.10 DELETE{id} 10
3.11 RESET 11
3.12 Louer 11
3.13 Retour 12

Conclusion **13**

Introduction

La bibliothèque numérique gère différente type de ressource telles les livres, les magazines, les Vhs, les Dvd, les Cd et les ressources numériques digitales. Le programme est destiné à deux types d'utilisateurs. Le client et l'administrateur. Le client peut juste consulter les diverses informations au sujet des ressources. L'administrateur a en plus le droit de supprimer et d'ajouter des ressources.

1 Cahier des charges

Cette application doit permettre l'ajout ou la suppression de médias dans la médiathèque, permettre une recherche (incrémentale) d'un média selon une chaîne de caractères entrée, l'affichage ou la remise à zéro de cette recherche. On doit aussi pouvoir afficher les informations relatives à un média si l'on donne son numéro de référence.
Pour une utilisation plus pratique encore, on doit pouvoir sauvegarder ou charger une médiathèque à partir d'un fichier.
Un module d'emprunt et de retour est aussi utiliser pour simuler le vrai fonctionnement d'une médiathèque.
Pour créer ce programme, nous avons utilisés différents types de fonction dont voici la liste non détaillée (les fonctions avec une étoile sont disponibles que pour l'administrateur) :

La fonction	Son rôle
*ADD {type}	Ajoute une nouvelle ressource de type type dans la médiathèque
BYE	Quitte le programme
CHANGE	Change l'utilisateur connecté
CLEAR	Réinitialise la médiathèque après une recherche
*DELETE {id}	Supprime la ressource de la médiathèque d'identifiant {id}
LIST	Affiche les ressources présentes dans la médiathèque
HELP	Accéder au module d'aide
*LOAD {fichier}	Charge une médiathèque à partir du fichier {fichier}
*RESET	Supprime toutes les ressources de la médiathèque
*SAVE {fichier}	Enregistre la médiathèque dans le fichier {fichier}
SEARCH {chaine}	Recherche les ressources contenant la chaine de caractères en paramètre
*SCHOW {id}	Affiche les informations relatives à le ressource d'identifiant {id}

2 Diagramme de classes

Toutes les fonctions doivent pouvoir manipuler les différents types de média.
Le diagramme des classes est le suivant :

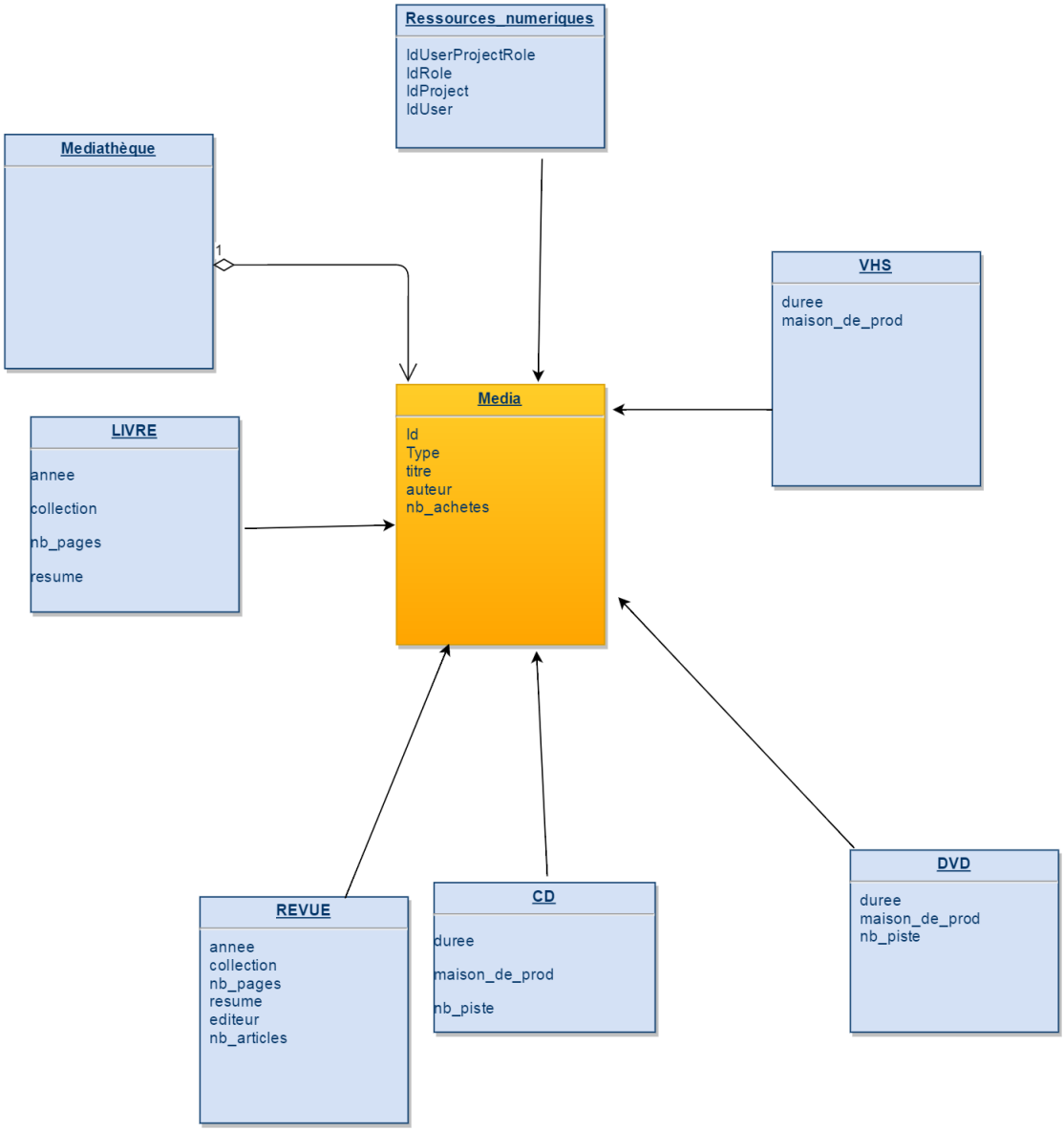

FIGURE 1 – diagramme des classes

1. Media

La classe “Media” est une classe abstraite, cela signifie que l’on ne peut pas instancier un objet de type media. Elle sert de classe de base pour toutes les classes qui en héritent. Elle contient plusieurs attributs communs qui sont le titre, l’identifiant,le titre, le type, le nombre acheté et l’auteur.

La méthode “Lister” est uniquement associée à la classe Media, cette méthode permet d’afficher les informations d’une ressource. Pour créer notre objet on passe par la fonction add qui rentre une à une les informations.

Cette classe contient des méthodes virtuelles, en effet on veut utiliser le principe de polymorphisme. Cela permet de définir dans la classe mère une méthode et de la compléter dans chaque classe fille. Il n’est pas nécessaire de mettre “virtual” dans les classes filles car elles le sont automatiquement par l’héritage.

2. Livre

La classe “Book” est une classe fille de la classe “Media”, ainsi elle contient les attributs qui lui sont hérités. Toutefois cette classe à ses propres attributs, qui correspondent à “l’année de publication”, “le résumé”, “la collection”, “le nombre de page”

Cette sous classe de média possèdent les méthodes héritées de média mais aussi ses propres méthodes qui servent au fonctionnement de l’élément livre.

3. Ressource numérique

La classe ressource numérique est une classe fille de “Media“. Elle contient donc les attributs de sa classe mère média mais aussi ses propres attributs qui sont : “le chemin d’accès”, “le format”, et la “taille”.

4. revue

La classe revue hérite de la classe Média, elle contient tout les attributs de la classe mère mais aussi ses propres attributs qui sont “l’année”, “la collection”, “le résumé”, ansi que “le nombre de page”.

5. Vhs

La classe revue hérite de la classe Média, elle contient tout les attributs de la classe mère mais aussi ses propres attributs qui sont “la maison de production” et “la durée de la vhs”.

6. Dvd

La classe revue hérite de la classe Média, elle contient tout les attributs de la classe mère mais aussi ses propres attributs qui sont “la maison de production”, “la durée de la vhs” et “le nombre de pistes”.

7. Cd

La classe revue hérite de la classe Média, elle contient tout les attributs de la classe mère mais aussi ses propres attributs qui sont “la maison de production”, “la durée de la vhs” et “le nombre de pistes”.

8. Mediatheque

Cette classe contient l’ensemble des méthodes qui servent lors de notre projet. Les fonctions sont définies en public pour pouvoir être accessibles depuis tout les endroits du projet.

3 Attributs des classes - Commandes

On a définit les différentes classes, reste maintenant l'étape la plus importante du projet : définir les méthodes que l'on va devoir associer à celles-ci. Par évidence, on trouve dans chaque méthode la lecture et l'écriture pour chaque attribut présent.

3.1 Fonctionnement du main

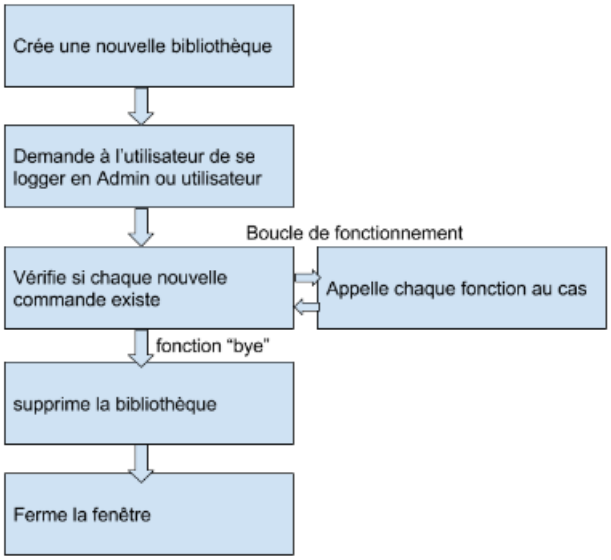

Figure 2 – Fonctionnement du Main

3.2 BYE :

La fonction Bye a pour but de tout fermer au sein de l'ordinateur. Pour quitter l'utilisation de notre médiathèque. Cette fonction utilise "delete" mais sur la médiathèque principale et la médiathèque temporaire.

Ainsi on supprime tout les médias appelés. Ainsi, on voit l'utilité d'avoir le C++, car en appellant tout les types de média "Bibliotheque", en appellant "delete" sur Bibliotheque on supprime d'un coup tout les médias rattachés à notre projet.

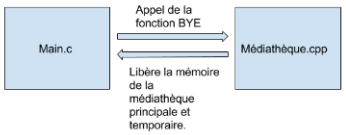

FIGURE 3 – Diagramme de séquence de la commande Bye

3.3 L'ajout : la fonction ADD {type}

On crée une ressource de type type dans un premier temps. On accède alors à la fonction add() qui correspond à ce type. Le role de cette fonction est de récolter les informations que va rentrer l'utilisateur. Une fois ces informations récupérées on les ajoutent à la médiathèque principale. On récupère ces informations en demandant à l'utilisateur de les rentrer grâce à une fonction.

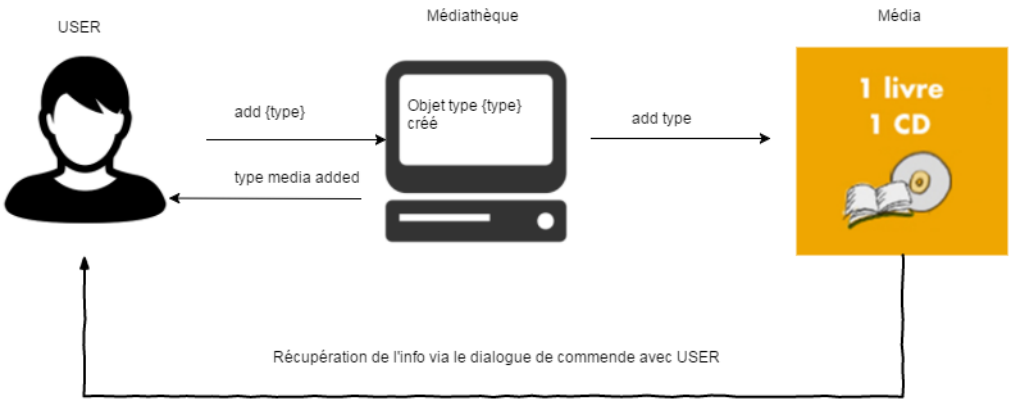

FIGURE 4 – Diagramme de séquence de la commande ADD type

3.4 LOAD filename :

Le principe de cette fonction est d'abord d'ouvrir un fichier nommé filename s'il existe si la bibliothèque à déjà été utilisée. Grace à ce fichier on

lit ligne à ligne jusqu'à la fin du fichier. Pour chacune des lignes, on récupère le type de média. Et on en créer l'objet correspondant. On utilise alors la fonction load(ligne) qui permet de récolter tous les renseignements de la ressource. Cette ressource est ajoutée dans la médiathèque principale.

3.5 SAVE filename

On commence par ouvrir le fichier correspondant à filename. Pour chaque ressource de la médiathèque, il faut écrire ses coordonnées selon une trame définie.

il faut notamment qu'à une seule ligne correspondant à une seule ressource. Chaque type de média aura donc la fonction save( )

Remarque : Si le fichier que l'on veut sauver existe déjà les nouvelles informations vont réécrire les anciennes.

3.6 SEARCH chaîne

Pour rechercher un élément, nous nous plaçons dans la médiathèque temporaire. Pour toutes les ressources présentes dans cette médiathèque, nous recherchons si la chaine tapée en paramètre se retrouve dans l'un des renseignements concernant la ressource. Pour chaque type de média, on a alors besoin d'une fonction search(chaine).

Enfin, nous devons remplacer la médiathèque temporaire par cette nouvelle médiathèque et l'afficher.

3.7 CLEAR

On remplace la médiathèque temporaire par la médiathèque principale : on appelle cette opération clone( ).

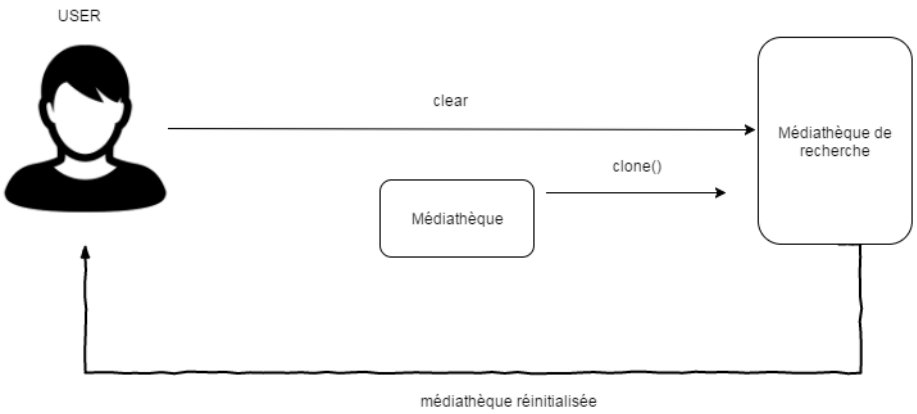

FIGURE 5 – Diagramme de séquence de la commande LIST

3.8 LIST

On affiche toutes les ressources présentes dans la médiathèque temporaire, la fonction d'affichage étant la fonction show( ), quel que soit le média.

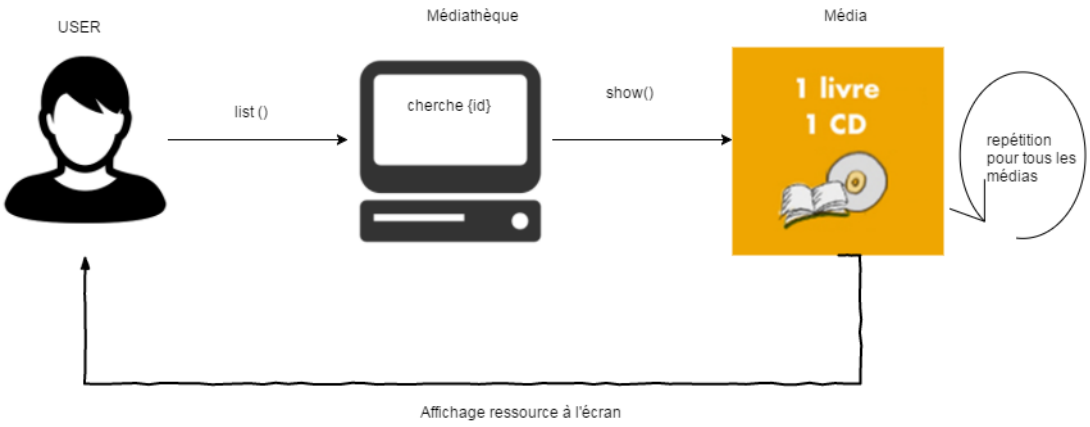

FIGURE 6 – Diagramme de séquence de la commande LIST

3.9 SHOW {id}

On cherche la place dans la médiathèque principale de la ressource correspondant au bon identifiant ; c'est la fonction show(id). Si cet identifiant existe, on affiche la ressource correspondante.

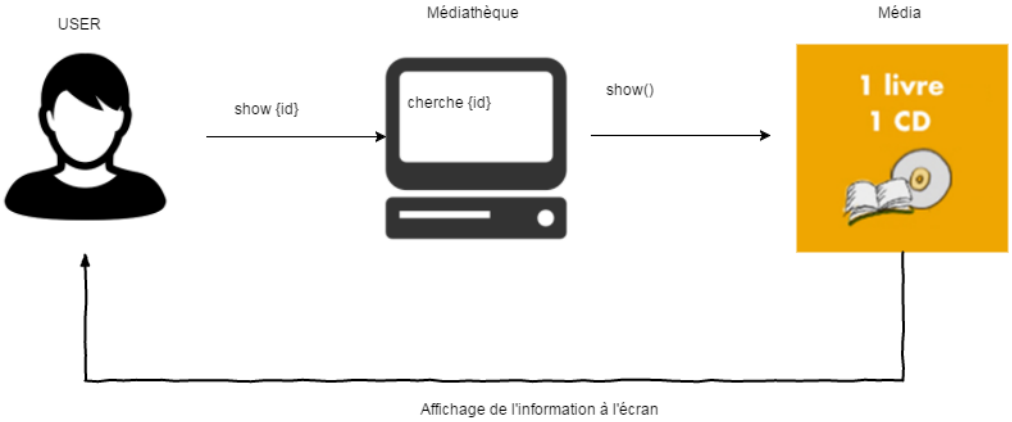

FIGURE 7 – Diagramme de séquence de la commande SHOW id

3.10 DELETE{id}

La fonction show(id) sert à trouver la place dans la médiathèque principale. On supprime la ressource correspondante de la médiathèque principale. On fait la meme chose pour la médiathèque temporaire.

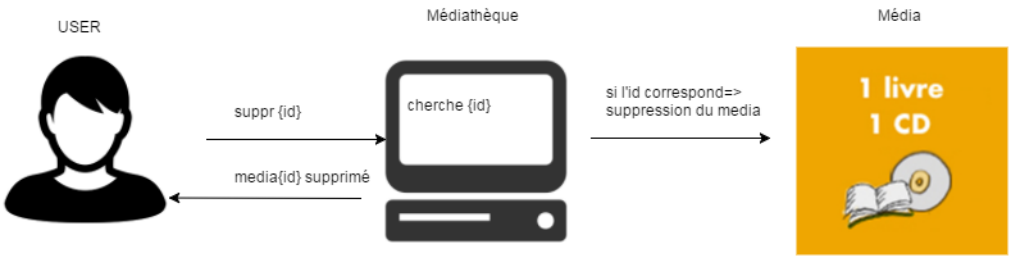

FIGURE 8 – Diagramme de séquence de la commande DELETE id

“delete” est déja une commande en C++, il faut donc l’appeler différemment, nous choisissons suppr(int id) dans la classe Médiathèque.

3.11 RESET

La fonction reset à un fonctionnement assez basique. Il s’agit d’appeler le tableau qui contient les données du média que nous avons chargé.

=> contrairement à la fonction bye par exemple elle ne supprime pas la bibliothèque, elle ne fait que supprimer le média chargé.

On utilise la fonction de base C++ “delete” pour libérer le bloc mémoire.

FIGURE 9 – Diagramme de séquence de la commande DELETE id

3.12 Louer

Pour louer on recherche d’abord de la place dans la médiathèque principale. Ensuite, on cherche la place dans la médiathèque principale pour la ressource correspondante au bon identifiant.

Finalement, on fait la réservation à l’aide de la fonction location() qui décrémente la variable nb_restant. Ce qui permet de louer un élément.

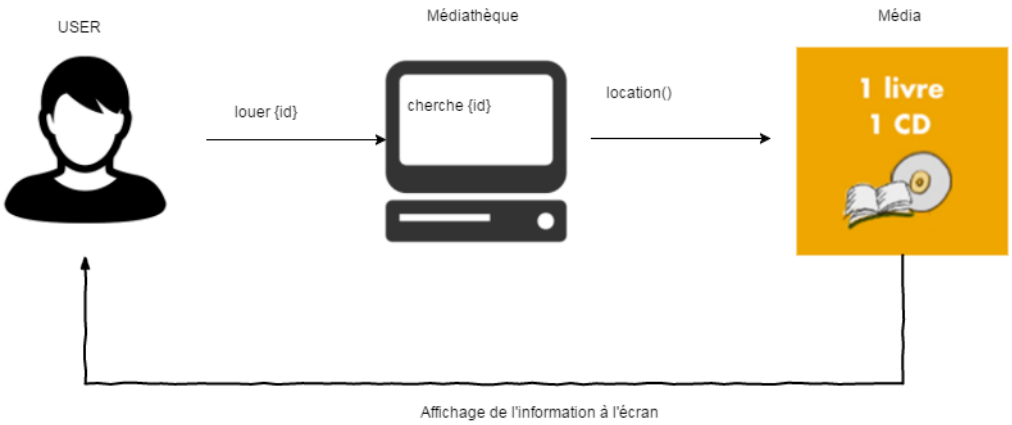

FIGURE 10 – Diagramme de séquence de la commande LOUER

3.13 Retour

Pour retourner on cherche d'abord de la place dans la médiathèque principale. Ensuite, on fait la réservation à l'aide de la fonction retourner() qui incrémente la variable nb_restant.

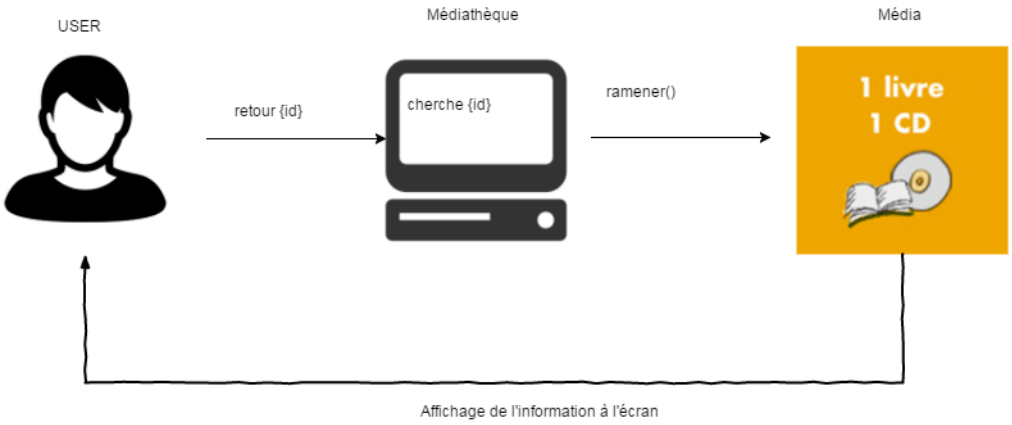

FIGURE 11 – Diagramme de séquence de la commande RETOUR

Conclusion

Notre projet répond au cahier des charges qui nous était fixé. D'autres choses ont été envisagées comme par exemple l'ajout d'une interface avec Qt Design. Mais faute de temps et des problèmes de compilations sur Qt créator à cause de windows ont fait que nous avons du nous concentrer sur le projet dans un terminal.

Nous avons pu découvrir la programmation en C++ en appliquant ce qui nous avait été montré en cours. Avec les différents principes d'héritages et de polymorphismes.

Nous avons conscience que la médiathèque que nous avons réalisée pourrait être optimisée, par exemple nous pourrions créer une répartition des classes plus complexe avec des sous classes de sous classes (en rangeant les revues comme sous classe de livre). D'autres optimisations que nous pourrions envisager mais nous ne sommes pas expert en C++. D'un point de vue algorithmie il doit exister différentes optimisations.