esiea
ECOLE D'INGENIEURS
DU MONDE NUMERIQUE

Génie logiciel et Projet GLPOO

cahier des charges

MAKON Maniym ma
NAJMI Mehdi
MVE MEYE BEKOUROU Sankara

IOINASCU Félicia

Table des matières

I. CADRE DU PROJET

1. Presentation de l'équipe

Notre équipe est composée de trois étudiants de classe 31 qui sont
Mehdi NAJMI.
MAKON MANYIM MA
Sankara MVE MEYE BEKOUROU

2. Enjeux et objectifs

L'objectif du projet génie logiciel est de nous faire travailler de manière collaborative sur un sujet donné, en mettant à l'œuvre les notions du cours, en améliorant nos compétences en POO et Java, et en produisant tous les artefacts d'un processus de développement de projet.
Toutes les notions vues en cours sont mises en pratique dans ce projet afin de comprendre leur utilité.

3. Livrables

Comme livrable, il est attendu une application fonctionnelle. Celle-ci sera basée sur l'application fournie par le professeur et notre but est de l'améliorer.

En implémentons tous les différents éléments de modification que nous avons présenté dans ce cahier de charge, nous faisons évoluer l'application.

4. Planning prévisionnel

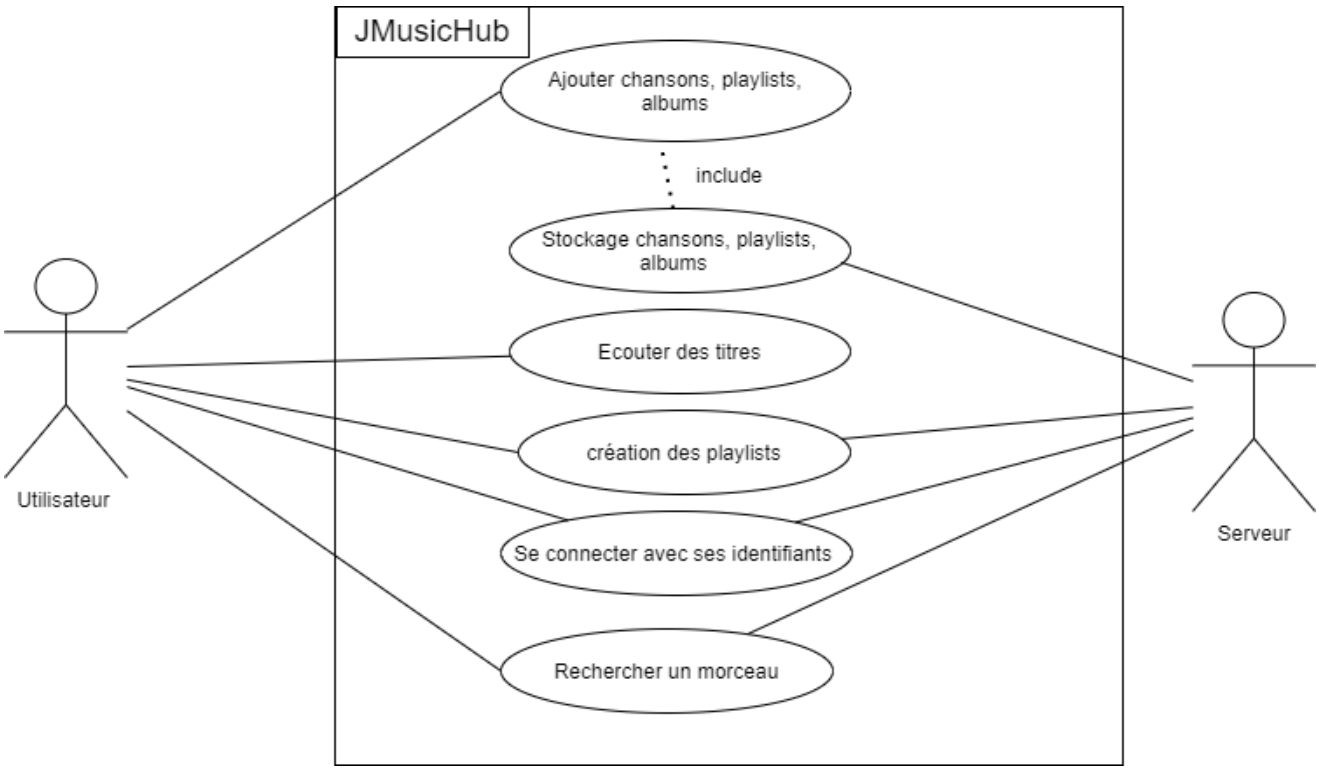
Toutes les modifications logicielles que nous devrons appliquées au logiciel seront opérationnelles au 3 mai 2021.
Nous avons déterminé toutes les exigences et spécifications nécessaires qui nous aideraient à la modification de l'application Music.
Nous avons également conçu tous les diagrammes utiles à la modification des logiciels.

II. SPECIFICATIONS

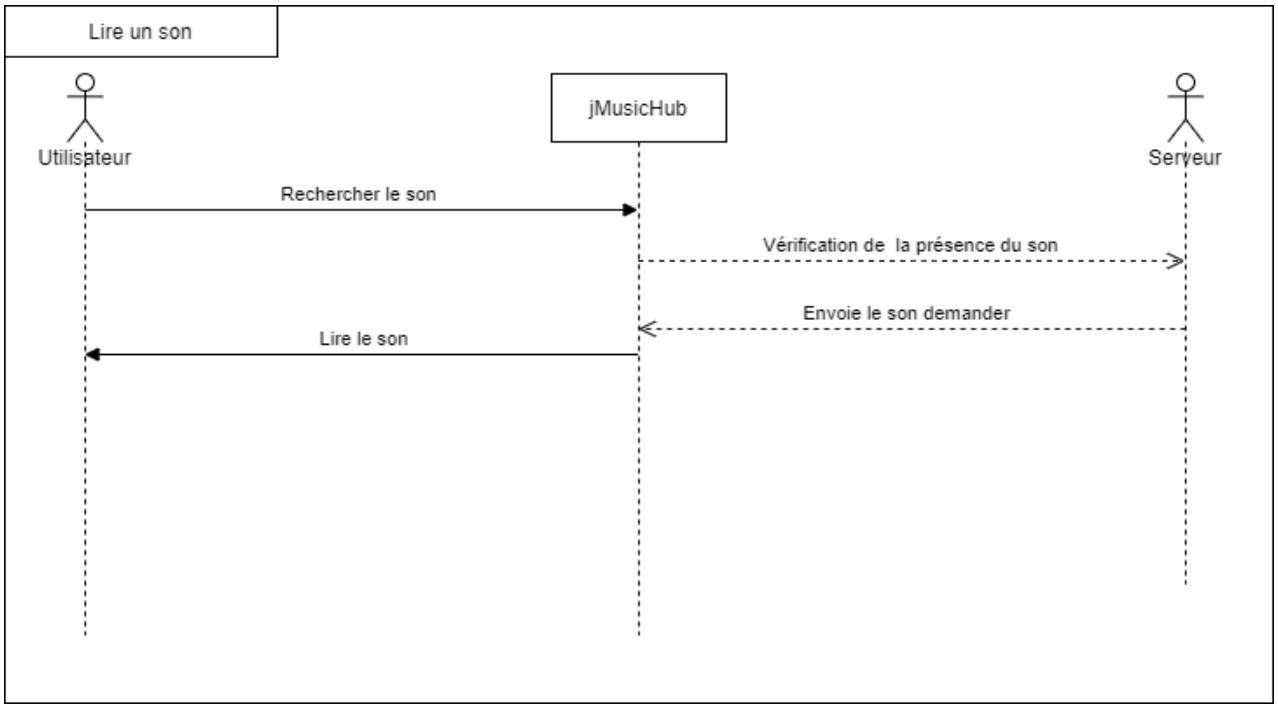
1. Exigences fonctionnelles et non-fonctionnelles

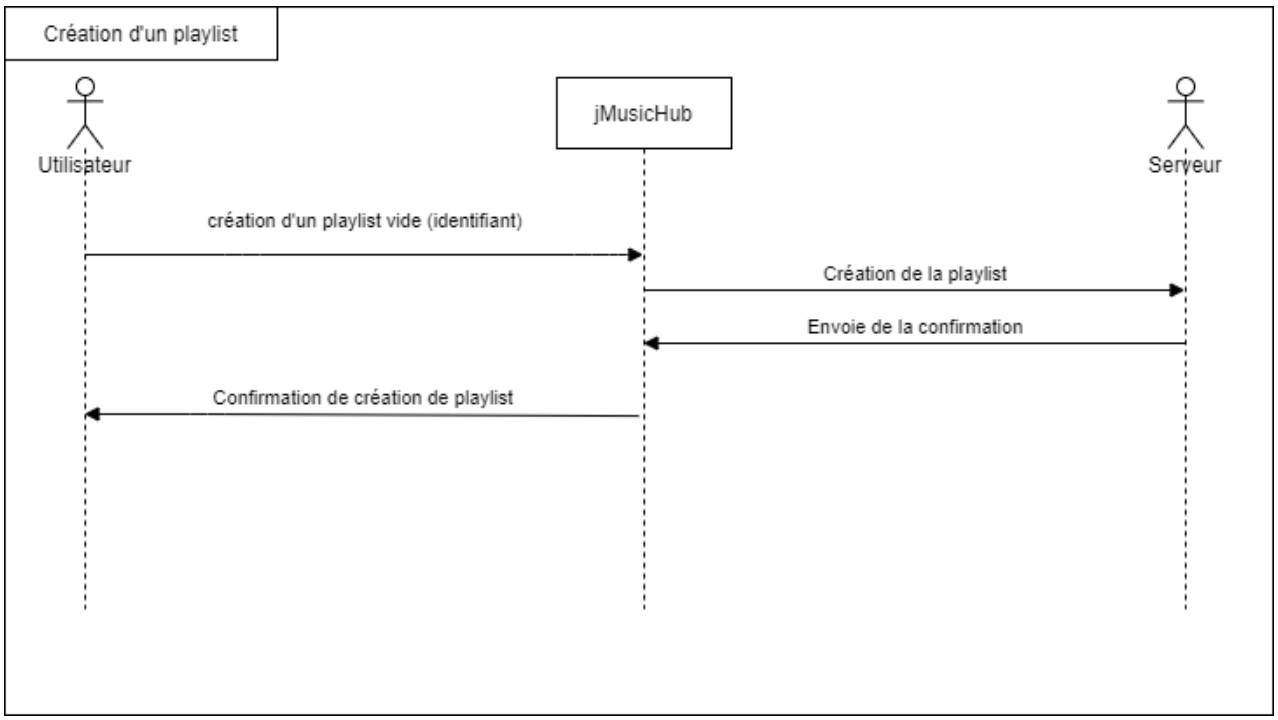
Exigences Fonctionnelles	Exigences Non-Fonctionnelles
Ajouter des morceaux à un album.	Travail uniquement en Java.
Lire aléatoirement des morceaux.	Utilisation de l'application sur les OS Windows, Linux, et IOS.
Sélection des chansons d'un album.	L'application ne fonctionne que dans la console.
Sélection des chansons d'un album.	
Affichage des chansons rangés par date de sortie, par genre, par playlist, par auteur (livre).	
Rajout d'une nouvelle chanson, album.	
Rajout d'une chanson existante à un album.	
Rajout d'un livre audio.	
Création d'une nouvelle Playlist à partir de chansons et livres audio existants.	
Sauvegarde des Playlist, des albums, des chansons et des livres audios dans les fichiers XML respectifs.	

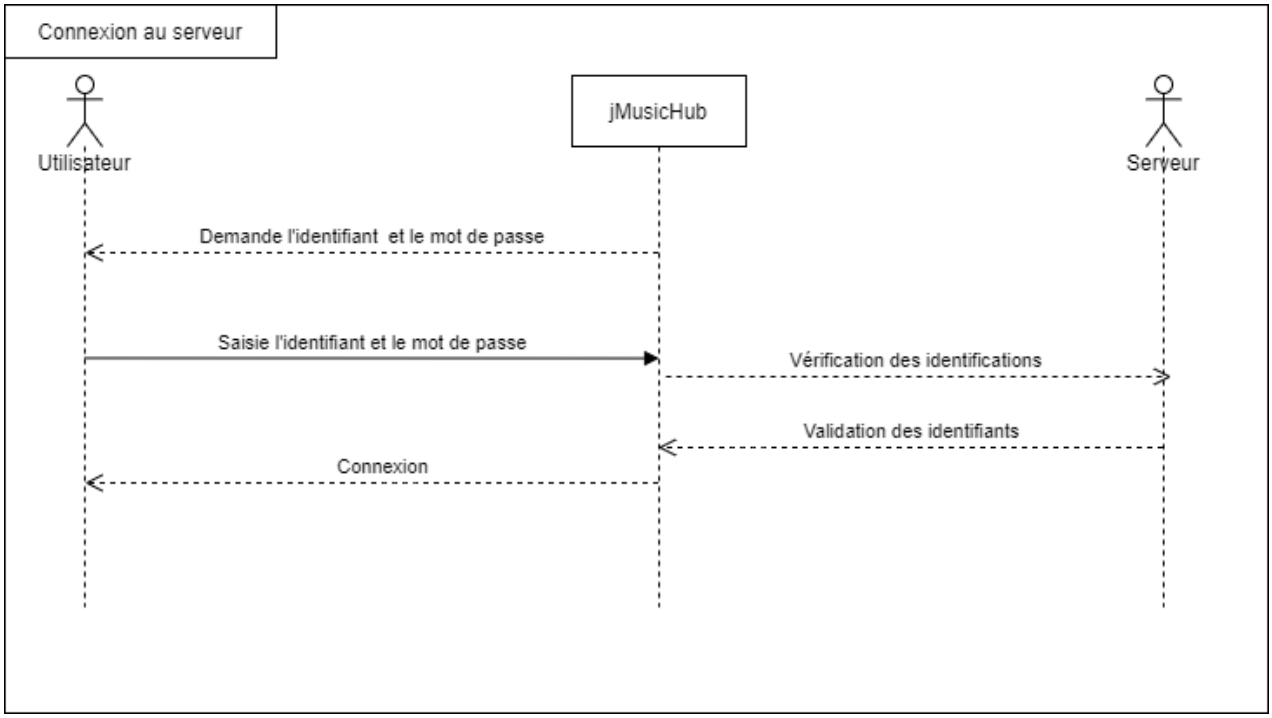
2. Diagramme de cas d'utilisation

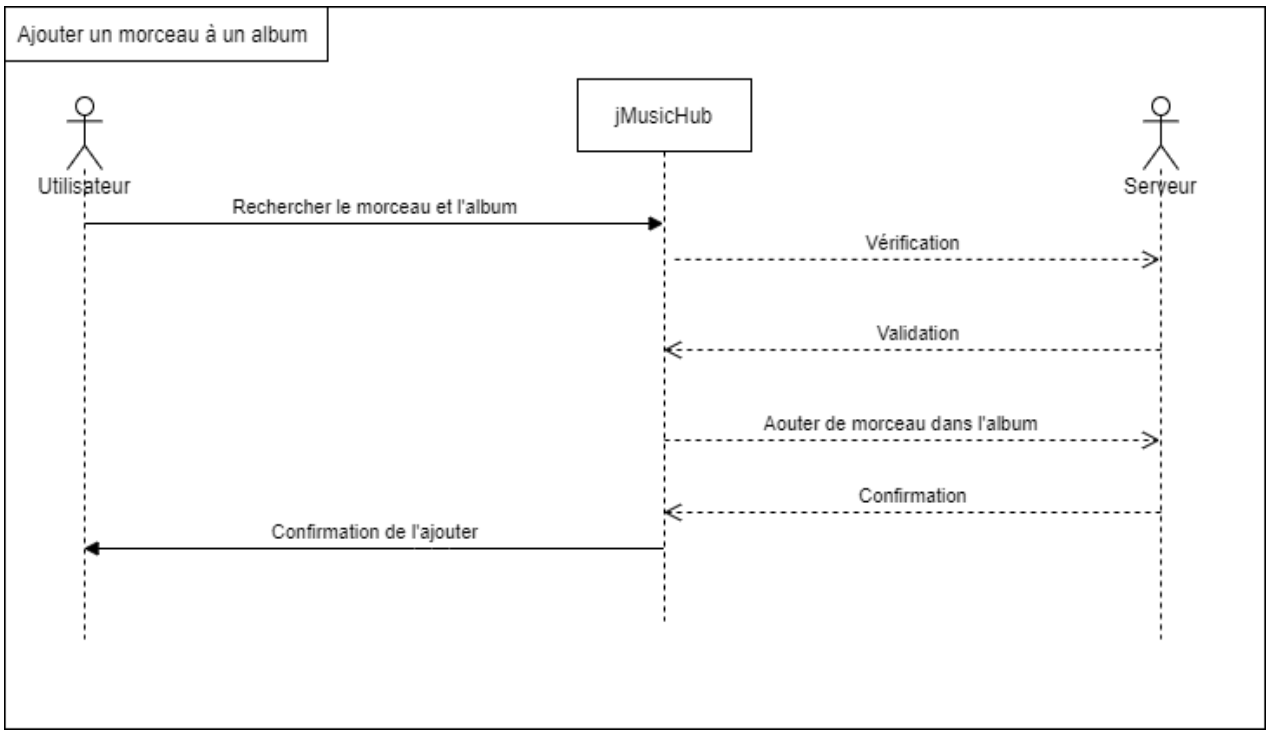


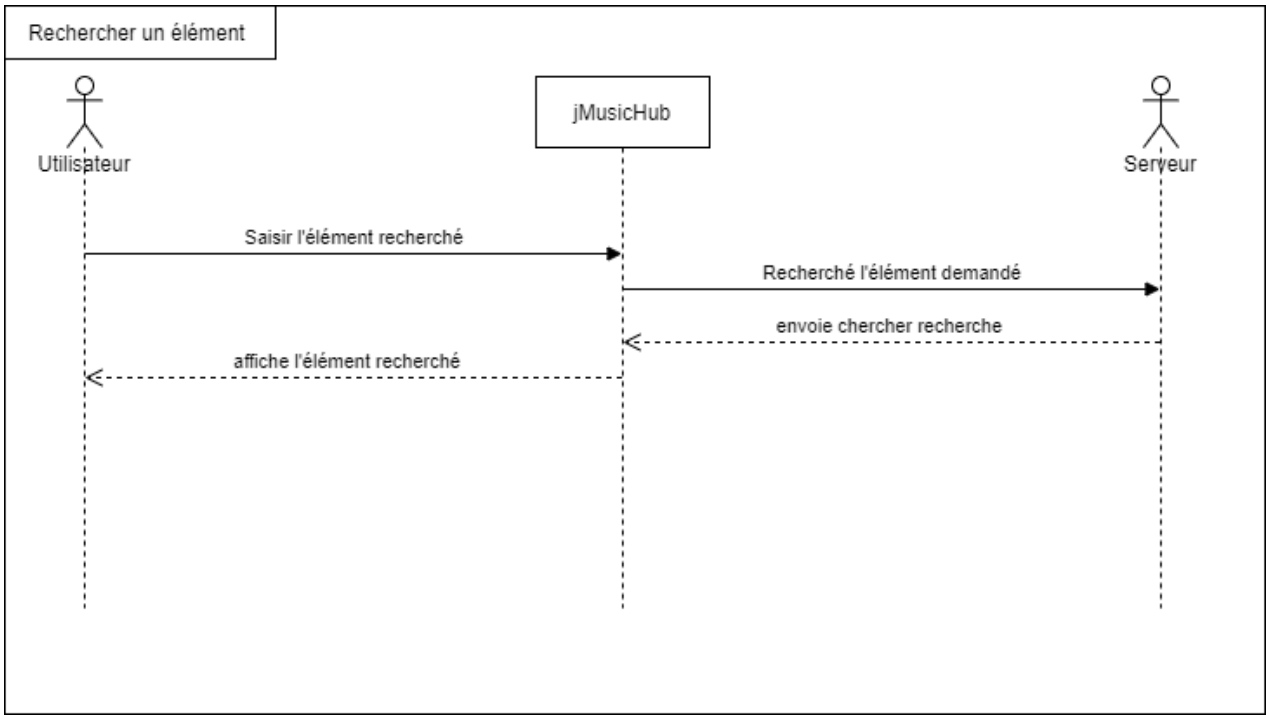
3. Diagrammes de sequence




Création d'un playlist
Utilisateur
jMusicHub
Serveur
création d'un playlist vide (identifiant)
Création de la playlist
Envoie de la confirmation
Confirmation de création de playlist

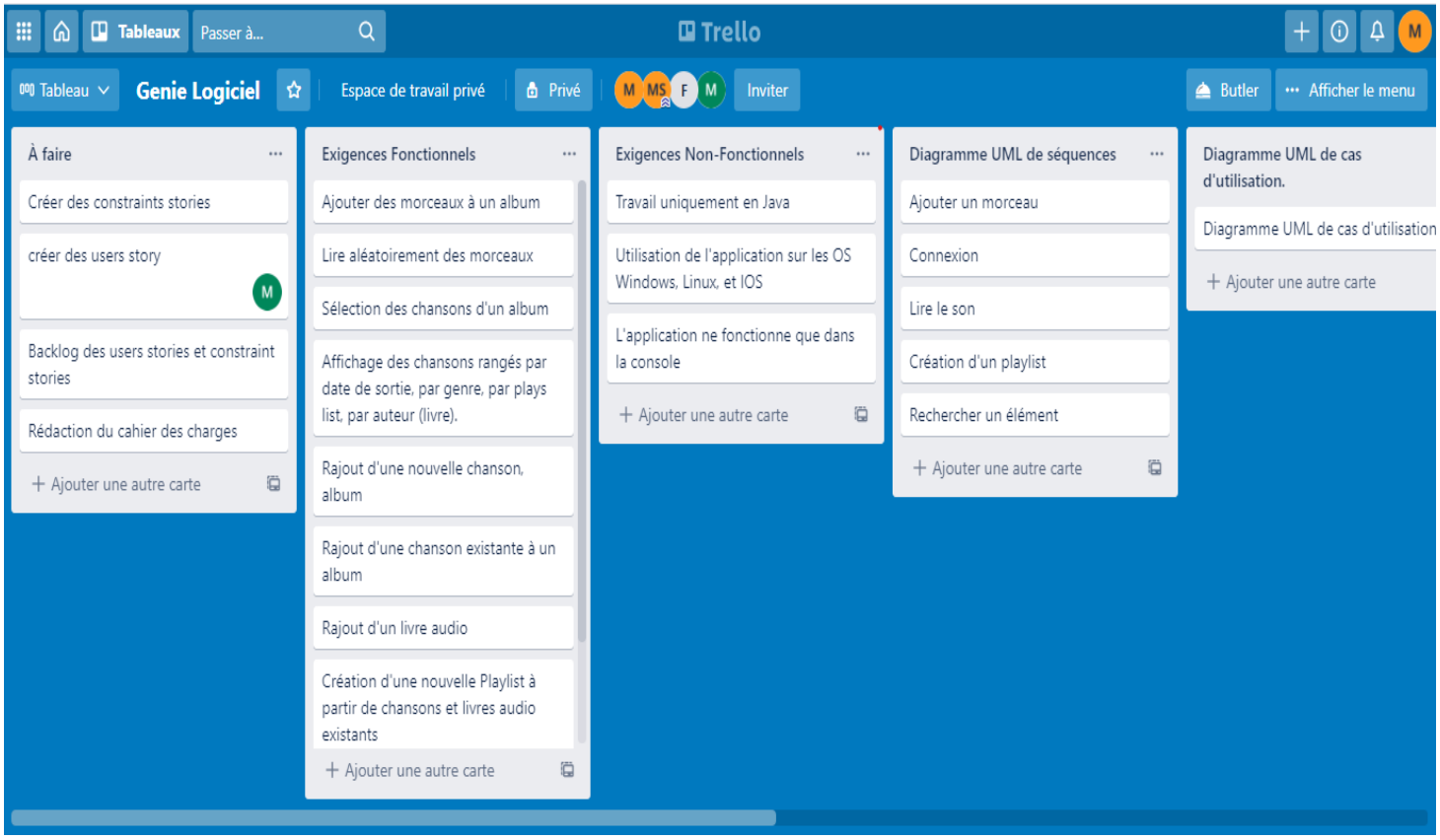

Connexion au serveur
Utilisateur
jMusicHub
Serveur
Demande l'identifiant et le mot de passe
Saisie l'identifiant et le mot de passe
Vérification des identifications
Validation des identifiants
Connexion


Ajouter un morceau à un album
jMusicHub
Utilisateur
Serveur
Rechercher le morceau et l'album
Vérification
Validation
Aouter de morceau dans l'album
Confirmation
Confirmation de l'ajouter


Rechercher un élément
jMusicHub
Utilisateur
Serveur
Saisir l'élément recherché
Recherché l'élément demandé
envoie chercher recherche
affiche l'élément recherché

4. Backlog

On a utilisé Trello comment outil de gestion de projet inspiré par la méthode Kanban.



Le lien vers le projet Trello <https://trello.com/b/DJ8JyETG/genie-logiciel>

User Stories(Client)

US 1.1 En tant qu'utilisateur normal, je veux me logger sur le serveur afin de s'identifier

US 1.2 En tant qu'utilisateur, je veux pouvoir supprimer un élément musical parce que ça ne m'intéresse plus

US 1.3 En tant qu'utilisateur, je veux rechercher un élément musical pour en obtenir les détails

US 1.4 En tant qu'utilisateur normal, Je veux ajouter des éléments musicaux dans une playlist afin de d'accéder plus facilement à mes sons favoris

US 1.5 En tant qu'utilisateur lambda, je peux obtenir une liste aléatoire d' éléments musicaux provenant d'un album afin de les lire

US 1.6 En tant qu'utilisateur, je veux pouvoir supprimer un élément musical parce que ça ne m'intéresse plus

US 1.7 En tant qu'utilisateur normal, Je veux ajouter des éléments musicaux dans une playlist afin de d'accéder plus facilement à mes sons favoris

US 1.8 En tant qu'utilisateur lambda, je peux obtenir une liste aléatoire d' éléments musicaux provenant d'un album afin de les lire

US 1.9 En tant qu'utilisateur, Je veux lire un élément musical afin de l'écouter

US 1.10 En tant qu'utilisateur normal, Je veux créer une playlist, afin de réécouter les éléments musicaux qui me plaisent

User Stories(Artiste)

US 1.1 En tant qu' artiste, Je peux créer un album afin de le rendre accessible

US 1.2 En tant qu'artiste, je peux supprimer u album pour ne plus le faire écouter

US 1.3 En tant qu'artiste, je peux ajouter un morceau dans un album afin de pouvoir l'améliorer

III. CONCEPTION

1. Diagramme UML de package, de composant, de déploiement et de classe

Diagramme UML de classe

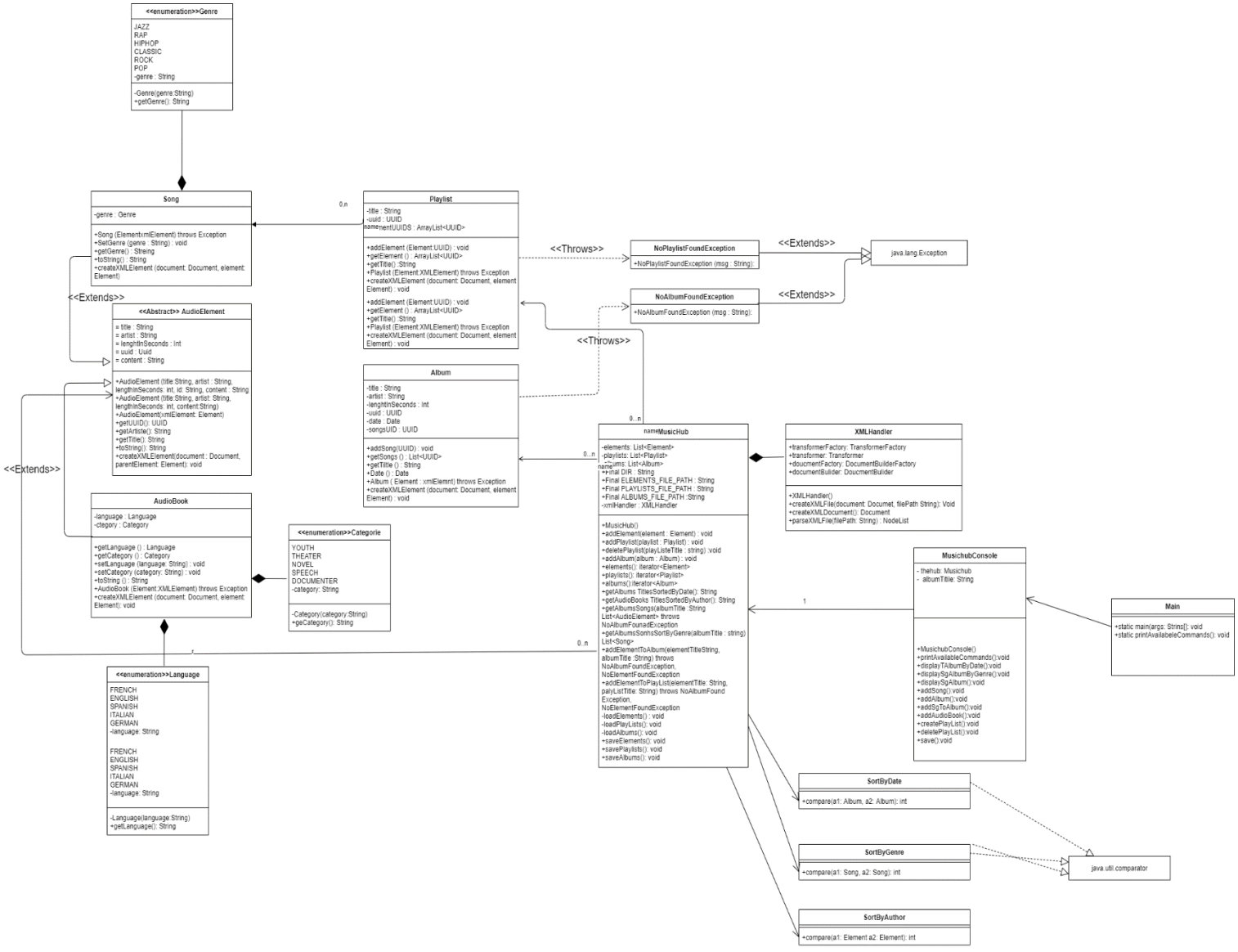


Diagramme UML de composant

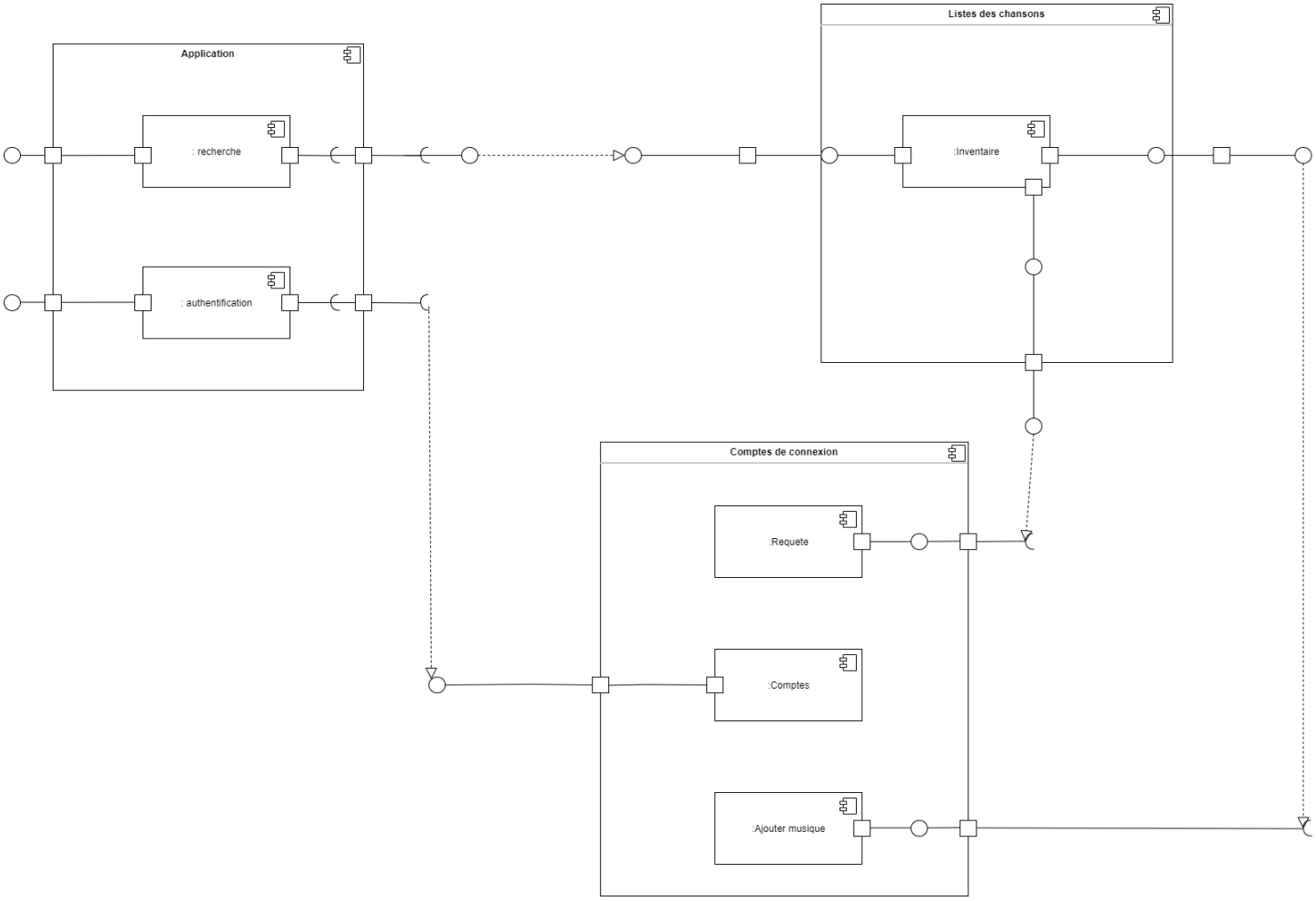


Diagramme UML de déploiement

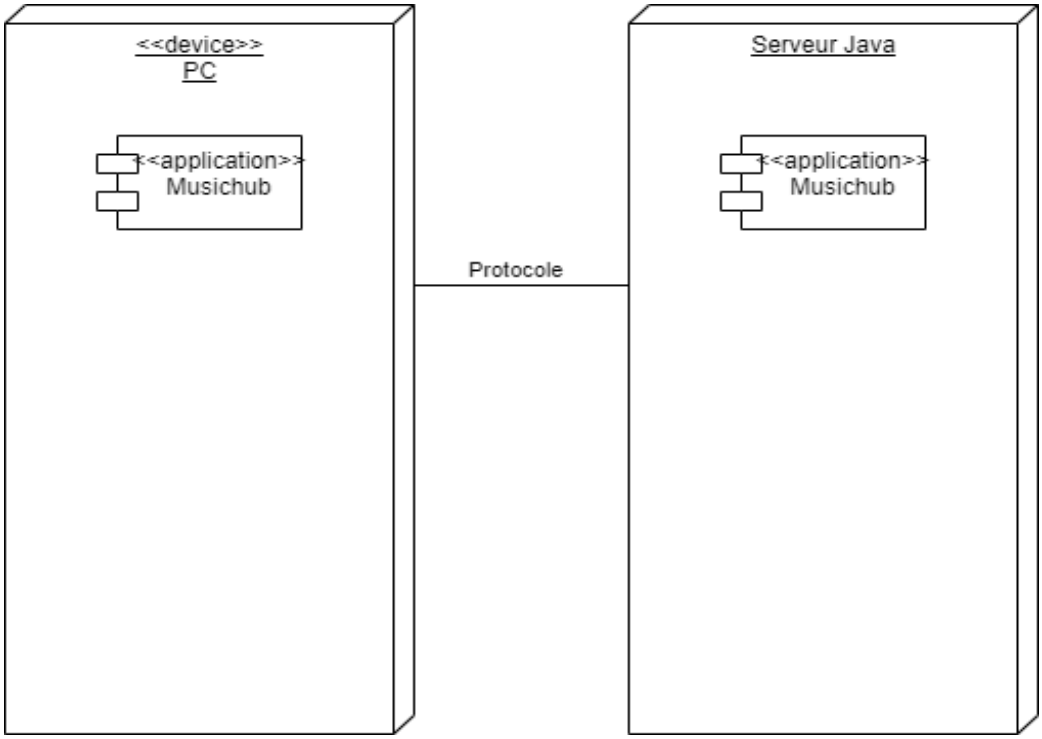
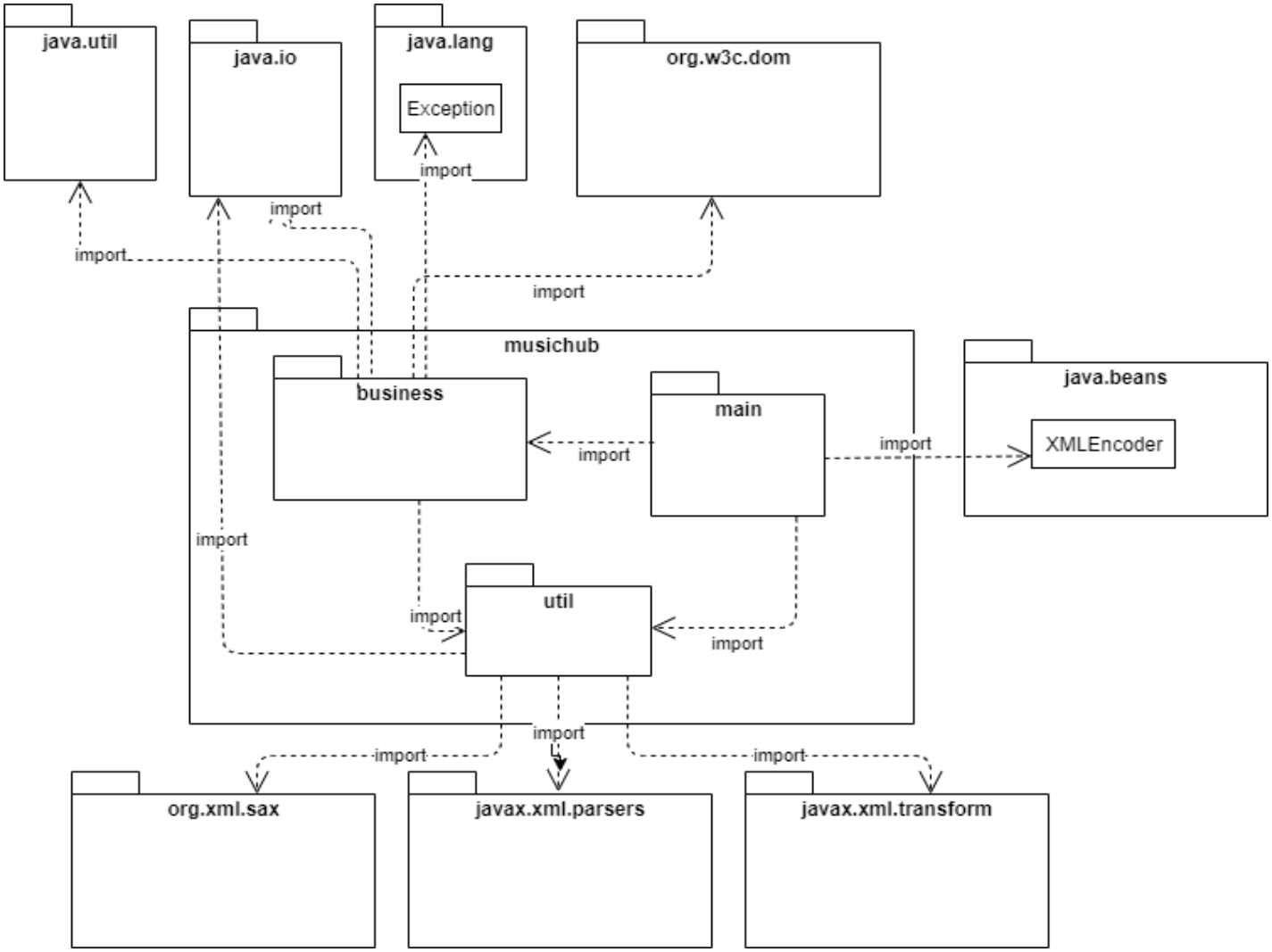


Diagramme de package



2. Analyse de la conception selon les principes SOLID avec argumentaire

a. Ouvert/Fermé

Nous avons déterminé des classes et méthodes publiques qui peuvent hériter d'autres fonctionnalités, mais les symboles sont privés pour ne pas perturber le fonctionnement de notre logiciel.

« Toutes les entités logicielles doivent être ouvertes à l'extension, mais fermées à la modification »

b. Responsabilité unique

Nous pensons que la caractéristique de responsabilité unique est bien honorée puisque nous avons découpé le projet en fonctionnalité que nous avons adapté par classe. Nous avons un tas d'éléments musicaux : albums, playlist, morceaux. Ils sont chacun représentés par une classe séparée lorsqu'on ajoute des morceaux, playlists sou albums au serveur.
De même pour les relations entre le client et le serveur, les méthodes sont encapsulées par un Request manager dans le but de permettre l'envoi des commandes du client au serveur.

« Chaque module, classe, ou méthode doit être responsable d'une seule partie de la fonctionnalité que le logiciel fournit, et cette responsabilité devrait être entièrement encapsulée par la classe, le module, ou la méthode. »

c. Ségrégation des interfaces

Pour ce caractère, le logiciel IntelliJ gouverne cela en nous proposant de supprimer les interfaces qui ne sont pas utilisés. Par ailleurs nous n'avons eu en recours que très rarement à l'implémentation d'une interface.

« Une classe ne doit jamais être forcée à implémenter une interface qu'elle ne l'utilise pas ou une méthode qui n'a pas de sens pour elle »

d. Inversion des dépendances

Le classe Song comme peut le témoigner le diagramme UML de classes ne Dépend que de l'abstraction AudioElement.

« Les entités doivent dépendre uniquement des abstractions »

3. : Un lien vers le projet sous GitHub et les identifiants des étudiant

Le lien : https://github.com/EdMkn/Projet_glpoo_musichub/tree/main

Les identifiants : NAJMI MEHDI → Najmi19
MAKON MANYIM MA → EdMkn
MVE MEYE BEKOUROU SANKARA → mvesankara

Descriptions des Rôles en Equipe		
Tâche	**Prénom**	**Description**
La liste des exigences fonctionnelles et non fonctionnelles	Makon	Décrit les nécessités et les attentes auquelles doit répondre le produit. Lors d'une réunion du groupe, elles sont validées, redéfinies et certaines mêmes sont supprimées ou rajoutées.
Le diagramme UML des cas d'utilisation	Mehdi	Diagramme concu pour représenter le comportement fonctionnel du logiciel, puis modification, appéciation et validation par l'ensemble du groupe lors d'une réunion
Les diagramme UML de séquence	Mehdi	Permet de montrer les interactions d'objets dans le cadre d'un scénarion de diagramme d'utlisation.
Trello: Le backlog des User Stories et Constraint Stories, chacune ayant	Sankara-Mehdi	Trello est la plateforme permettant de suvre les différentes du projet. On peut y distinguer les tâches déjà effectués, et celles qu'il y a encore à faire.
Diagrammes UML de package	Makon	Représentation graphique des paqutages composant le système qu'est notre application JMusicHub.
Diagrammes UML composants	Sankara	Description de l'organisation de l'application du point de vue des éléments logiciels qui le constituent.
Diagrammes UML déploiement	Makon	Vue statique qui sert à représenter l'utilisation de l'infrastructure physique par le système et la manière dont les composants du système sont répartis ainsi que leurs relations entre eux.
Diagrammes UML classes	Mehdi	Schéma utilisé représentant les classes et les interfaces des systèmes ainsi que leurs relations.
les principes SOLID	Mehdi	Ces principes de programmation sont la base de tout code qui se veut clair, propre, facilement maintenable et facile à faire évoluer. je me suis basse sur les 5 principes pour voir si notre code les respecte.
design patterns	Makon	J'ai créé une façade MusichubConsole pour la classe Musichub: cette façade contient toutes les méthodes utilisées pour faire office de services aux utilisateurs
javaDoc	Sankara-Mehdi	une documentation d'API depuis les commentaires présents dans un code source en Java.
des classes de tests unitaires		
Un script Maven		