**Génie Logiciel**

Projet

Membres du groupe :

Samy CHETOUANE

Paco RAHON-CLOS

Kavitha KICHENASSAMY



3A - 33

2020 -2021

**Sommaire**

**I - Liste des étudiants et leurs tâches effectuées…………………….….p3**

**II - Liste des exigences fonctionnelles…………………………………...p3**

**III - Diagramme UML**

**1 - Diagramme UML des cas d’utilisations……………….……..p4**

**2 - Diagramme UML de séquence……………………………….p5**

**3 - Diagramme UML de classes………………………..………...p6**

**IV - Analyse de la conception selon les principes SOLID……………p7**

**V - Lien vers projet GitHub………………………………………...….p7**

**VI - Liste des tests d’acceptances et des bugs trouvés………………..p7**

**I - Liste des étudiants et leurs tâches effectuées**

|  |  |
| --- | --- |
| Membre | Rôles, tâches et responsabilités |
| Samy CHETOUANE | * Architecture Serveur-Client * User Stories et Constraint Story * Code et Design Pattern |
| Kavitha KICHENASSAMY | * Diagrammes UML * Code et Design Pattern * [Audio] InputStream * Script Maven |
| Paco RAHON-CLOS | * Code et Design Pattern * Test Unitaire |

**II - Liste des exigences fonctionnelles**

User story :

En tant qu’utilisateur, on pourra créer une chanson, un album, une playlist coté serveur. Puis récupérer et afficher notre création coté client.

(Critère d’acceptance) :

* Il faut d’abord ajouter notre élément musical coté serveur pour l’avoir coté client
* L’application et réalisé avec une architecture Serveur-Client
* On pourra écouter les morceaux de musique sur le client

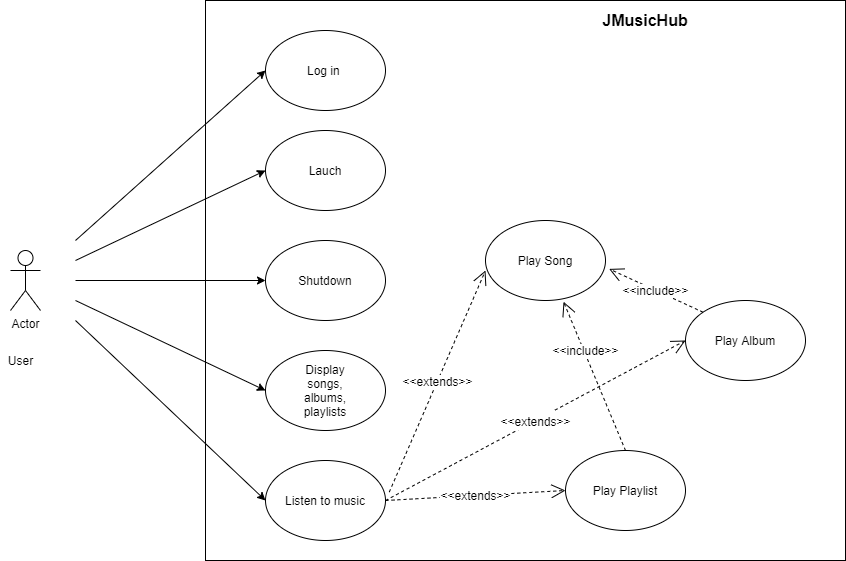
Constraint Story :

* Le système sera écrit en Java
* Le test unit report sera sous la forme d’un document texte
* Les données de configuration seront stockées dans des fichiers XML
* Le système ne fera pas usage de programmation web

Lien Trello : <https://trello.com/b/mi7li9FA>

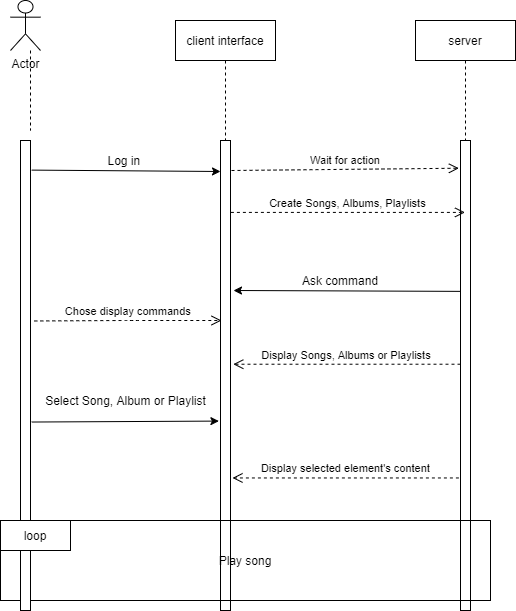
**III - Diagramme UML**

**1 - Diagramme UML des cas d’utilisations**



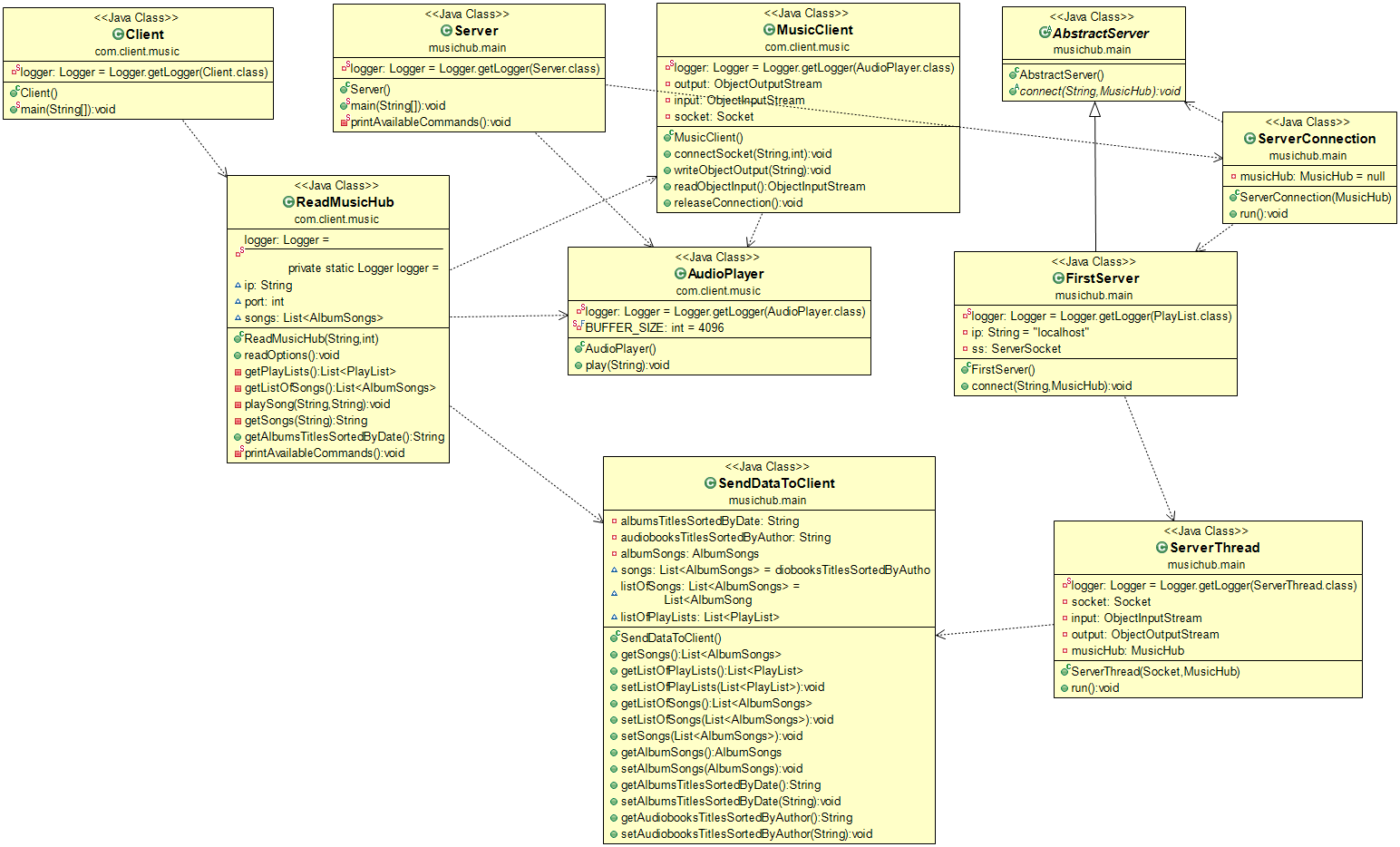
Voir le fichier « diagramme de cas d’utilisation.png »

**2 - Diagramme UML de séquence**



Voir fichier « Diagramme de séquence.png »

**3 - Diagramme UML de classe**



Voir le fichier « UML Diagram.png »

**IV - Analyse de la conception selon les principes SOLID**

Nous avions commencé le code ainsi que le diagramme UML trop tôt, on s’est rendu compte après que nous n’avions pas une connaissance suffisante des principes SOLID. Au cours du projet, Paco Rahon-Clos a décroché et abandonné l’école. Sachant qu’il a contribué au code et qu’il était plus à l’aise que nous en informatique, il fut difficile à deux de tout modifier.

Nous avons utilisé la méthode du patron singleton pour répondre au besoin d’unicité et d’accessibilité.

Le projet est tout de même ouvert à l’extension et fermé à la modification.

Une classe n’implémente jamais une interface qu’elle n’utilise pas ou une méthode qui n’a pas de sens pour elle.

Cependant, il est difficile de substituer les classes sans faire bugger le projet.

Les entités ne dépendent pas toujours d’abstraction.

En conclusion les principes SOLID ont été respecté qu’en partie. Il est possible de faire mieux en utilisant davantage de classes abstraites et d’interfaces.

**V - Lien vers projet GitHub**

https://github.com/kavitha9412/Projet\_GPLOO\_jMusicHub

**VI - Liste des tests d’acceptances et des bugs trouvés**

Le test unitaire s’affiche sous format texte dans le target/surefire-reports. Nous avions trouvé des tutoriels pour le faire sous format html, cependant on s’est rendu compte plus tard que le fait de ne pas utiliser de la programmation web était une exigence. Cette partie a été réalisé par Paco. Elle n’a pas été fini lors de son départ cependant il a laissé une bonne base, avec le ServerTest.java et le ClientTest.java ainsi que les 2 fichiers textes qui vont avec dans le target/surefire-reports.

L’élément musical crée coté server, se retrouve coté client. Le client peut jouer l’élément musical.

Le projet maven s’importe depuis eclipse.