AHMED-KHALIFA Aminetou

Projet Web

L3 MIAGE

Groupe 4 Miage

04 Janvier 2015

PAPELIER Romain

PIERSON Guillaume

VERHOOF Tom

Projet de Programmation Web

Rapport

Groupe 4 Miage

Site de musique

SOMMAIRE

[Introduction 3](#_Toc408145713)

[Choix techniques 4](#_Toc408145714)

[Architecture du site 4](#_Toc408145715)

[Actions possibles 6](#_Toc408145716)

[Fonctionnement « Back-end » 6](#_Toc408145717)

[Technologies utilisées 7](#_Toc408145718)

[Guide d’utilisation 8](#_Toc408145719)

[Jeu d’essai 9](#_Toc408145720)

[Limites de l’application 13](#_Toc408145721)

[Difficultés rencontrées 13](#_Toc408145722)

[Niveaux et attentes differentes 13](#_Toc408145723)

[Compréhension du code 14](#_Toc408145724)

[Conclusion 14](#_Toc408145725)

# Introduction

Le sujet du projet consiste en un site internet d’écoute de musiques, à l’instar de Deezer par exemple. Ce site internet devait respecter trois niveaux de priorités :

* Le site permet de rechercher des titres et de consulter les informations sur les artistes
* Le site permet de construire des playlists et de les écouter
* Le site permet de créer des comptes utilisateur et de sauvegarder les playlists

L’architecture de la base de données du site était imposée, et était composée des tables suivantes :

* Tracks(track\_id, *artist\_id*, title, mp3\_url) qui représente une musique associée à un artiste,
* Artists(artist\_id, name, image\_url, info) qui représente un artiste,
* Playlists(*user\_id*, playlist\_id, playlist\_name) qui représente une playlist associée à un utilisateur,
* Users(user\_id, username, password) qui représente un utilisateur,
* Playlists\_tracks(playlist\_id, position, track\_id) qui représente l’association d’une musique à une playlist à telle position dans la playlist. Il est important de noter que la clé est composée de playlist\_id et position, ce qui signifie que l’on peut avoir des doublons dans une même playlist, et que la suppression d’une musique dans une playlist se fait en fonction de la position de cette musique dans la playlist.

Etant donné que le site devra permettre l’écoute de musiques et de playlists de musiques, il est nécessaire de ne pas interrompre cette lecture par des rafraichissements de pages intempestifs qu’engendreraient l’utilisation de pages PHP uniquement, ce qui nous amène à deux possibilités :

* Un site internet composé de pages PHP, nécessitant un rafraichissement permanant pour tout changement dynamique dans la page, mais où le lecteur serait ouvert dans une autre fenêtre, comme Pulsradio.com
* Un site internet composé d’une seule page PHP, et utilisant des scripts en JavaScript pour modifier la page dynamiquement sans besoin de rafraichissement, et utilisant des requêtes AJAX vers des scripts PHP pour interagir avec la base de données, ce qui permet d’avoir le lecteur intégré dans cette même page, comme Deezer.

Nous avons retenu la deuxième solution car plus confortable pour l’utilisateur et plus en accord avec nos cours de Programmation Web.

NOTE : Il est possible de tester l’application par vous-même aux adresses suivantes :

<http://romainpapelier.fr/L3Miage/projet/src/>

<http://10h.guillaumepierson.fr/>

Il est également fortement déconseillé d’utiliser Internet Explorer dans une version inférieure à la 11

# Choix techniques

## Architecture du site

Dans cette optique de conception d’un site entièrement dynamique à base de JS-AJAX-PHP, nous avons structuré le site autour de 4 éléments principaux :

* Le **Header**, qui contient l’entête de la page, avec son nom. Il fait office de bannière.
* La barre d’outils **Nav**, qui contient le moteur de recherche à gauche, et à droite des boutons pour accéder à la liste des playlists de l’utilisateur, et pour se connecter, s’inscrire, ou se déconnecter.
* La div principale **MainDiv**, qui affiche les résultats de recherche, la page d’un artiste, ou l’interface de connexion et d’inscription.
* Le **Footer**, collé en bas de la page, qui contient le lecteur audio et le volet déroulant pour la playlist en cours. Ce volet se déroule jusqu’à la barre de Nav, recouvrant ainsi la div principale.

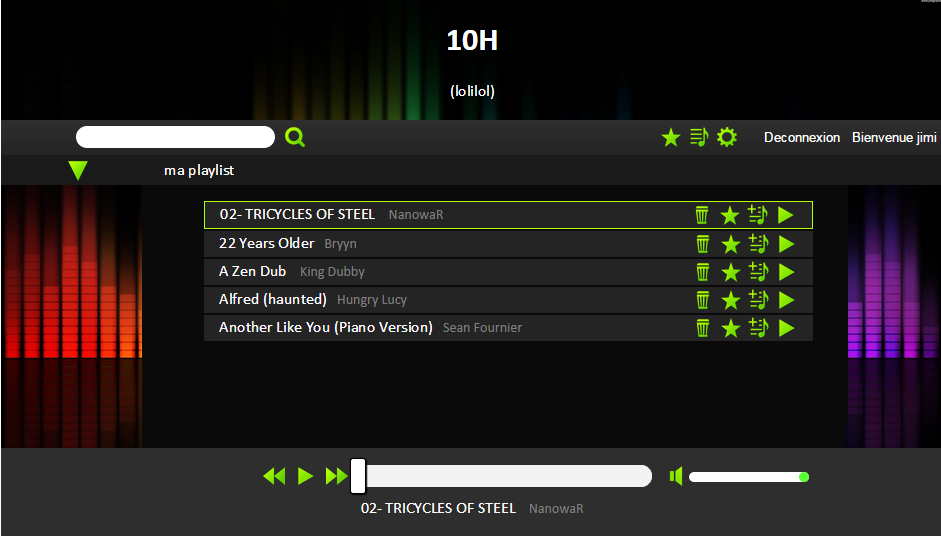
**Nav**

**Header**

Écran lors d’une recherche vide

**Footer**

**MainDiv**

Ecran après déroulement du volet de playlist

Pourquoi cet agencement ?

La première contrainte fut celle du côté « responsive design » que le site devait avoir, dans le cas où l’utilisateur ne l’affiche que sur une seule moitié de l’écran, afin d’écouter la musique en faisant autre chose à côté. C’est pour cela que les éléments ont une disposition très horizontale, et s’enchainent de haut en bas de la page. Ils occupent toute la largeur disponible et se redimensionnent en donc dans ce sens.

Nous avons également décidé de séparer l’interface en deux zones distinctes conformément à l’énoncé du projet, qui demandait une page concernant la recherche et la consultation d’un artiste, et une page permettant la navigation dans une playlist et sa lecture. Ainsi, la div centrale concerne la recherche et la consultation, et le footer concerne la navigation et la lecture d’une playlist.

Enfin nous nous sommes quelques peu inspirés de Spotify pour l’agencement et le design du site.

## Actions possibles

## Fonctionnement « Back-end »

Le site fonctionne dynamiquement en utilisant JQuery pour la manipulation du DOM. Des requêtes AJAX sont envoyées vers des scripts PHP, qui eux-mêmes interagissent avec la base de données.

Nous avons structuré l’architecture des scripts PHP selon le Pattern MVC (Modèle, Vue, Contrôleur).

Les Modèles regroupent les différentes entités présentes en base de données, codé selon le pattern Active Record : une table est représentée par une classe, une ligne de cette table est représentée par une instance de cette classe, et une colonne par un attribut de cette classe. Nous avons donc programmé une classe par table en base : Artist, Track, Playlist, PlaylistTrack et User. L’interaction avec la base de données se fait essentiellement dans ces Modèles, via des requêtes filtrées afin d’éviter toute injection SQL.

L’unique Vue (defaultView.php) affiche la page unique du site internet (HTML, Javascript)

Les Contrôleurs eux gèrent les actions relatives aux scripts PHP, et interagissent avec les Modèles.

Afin de garantir une meilleure sécurité, seul le fichier index.php est accessible pour le navigateur, et c’est ensuite ce fichier qui appel le contrôleur frontal situé dans le dossier Protected.

L’appel à un contrôleur se fait via un Contrôleur frontal (FrontController) qui fait office de routeur, et redirige la requête vers le contrôleur désiré (BaseController, PlaylistController, etc).

JavaScript

Vue (page)

Contrôleur frontal (PHP)

Contrôleur de Playlist (PHP)

Base de données

Modèle Playlist

## Technologies utilisées

Les langages utilisés pour la programmation de cite sont HTML5, CSS3, PHP, JavaScript, TypeScript et SQL.

Le TypeScript est un langage qui permet de programmer des classes d’objet destinés à être compilés et utilisé en JavaScript. Nous avons utilisé TypeScript pour coder le lecteur audio (Player.ts), qui une fois compilé donne Player.js

Nous avons utilisé les bibliothèques JQuery pour la manipulation du DOM et l’envoi de requêtes AJAX, ainsi que Notify.js pour afficher des petites popup de notification.

Pour interagir avec la base de données coté PHP, nous avons utilisé la bibliothèque PDO, semblable à mysqli, mais offrant plus de possibilités et une meilleure sécurité dans les requêtes SQL, notamment en préparant les requêtes et en filtrant les arguments transmis par l’utilisateur afin d’éviter toute injection SQL.

# Guide d’utilisation

A l’arrivée sur le site, un utilisateur peut se connecter, s’inscrire, ou bien directement utiliser les fonctionnalités du site sans être connecté, mais ses playlists ne seront pas sauvegardées dans la base de données.

Pour écouter une musique, l’utilisateur peut soit lancer une playlist depuis l’icône des playlits  s’il est connecté, en cliquant sur le nom de la playlist dans la liste de ses playlists, ou bien lancer une recherche dans la barre de recherche pour rechercher un titre ou un artiste particulier.

Ensuite, depuis le résultat de cette recherche, ou depuis n’importe quelle autre liste de musiques affichée sur le site, l’utilisateur peut lancer une musique seule via le bouton de lecture  , l’ajouter à une playlist via le bouton d’ajout à une playlist  ou la retirer de la playlist en cours via le bouton de suppression .

Lorsque l’utilisateur clique sur le bouton d’ajout à une playlist, une popup s’ouvre, proposant les playlists existantes actuellement pour cet utilisateur. L’utilisateur peut soit choisir la playlist à laquelle ajouter la musique en cliquant sur le nom de la playlist, soit créer une nouvelle playlist via le premier bouton possédant l’icône puis cliquer sur le nom de la playlist créée pour ajouter la musique à celle-ci, soit supprimer une playlist depuis le bouton .

L’utilisateur peut également supprimer une playlist via ce même bouton en affichant la liste de ses playlists depuis l’icône de la barre d’outils.

NOTE : l’utilisateur ne peut pas supprimer une playlist en cours de lecture, c’est-à-dire une playlist dont une de ses musiques est actuellement en cours de lecture.

L’utilisateur ne peut pas retirer une musique d’une playlist si cette musique est en cours de lecture.

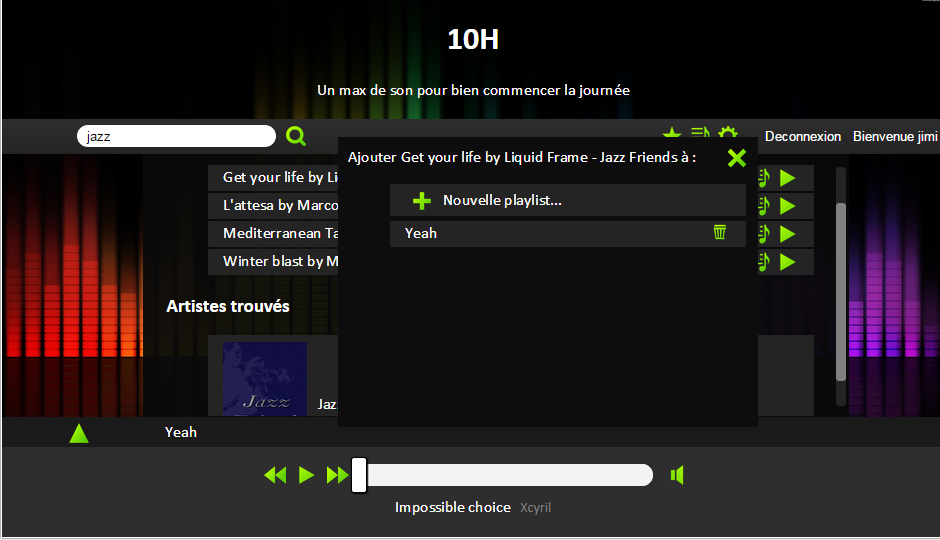
L’utilisateur ne peut pas créer deux playlists avec un nom identique, bien que rien ne l’empêche du point de vue du code.

Si l’utilisateur créé des playlists en tant qu’invité (non connecté donc), puis qu’il s’inscrit ou se connecte, les playlists créées en tant qu’invité son sauvegardées dans la base de donnée sur le compte avec lequel l’utilisateur vient de se connecter.

# Jeu d’essai

Illustration de l’auto complétion du moteur de recherche

Le moteur de recherche trouve les artistes selon leur nom, et les musiques selon leur titre ou le nom de leur artiste.



Affichage de la popup d’ajout à une playlist

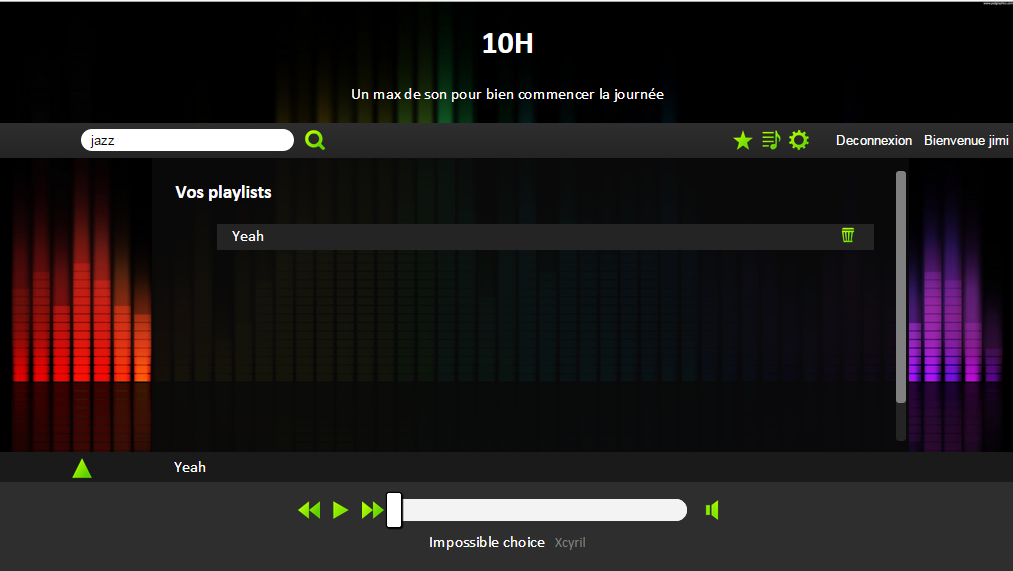
L’ajout d’une musique à une playlist en cours de lecture mettra celle-ci à jour dans le volet de playlist :



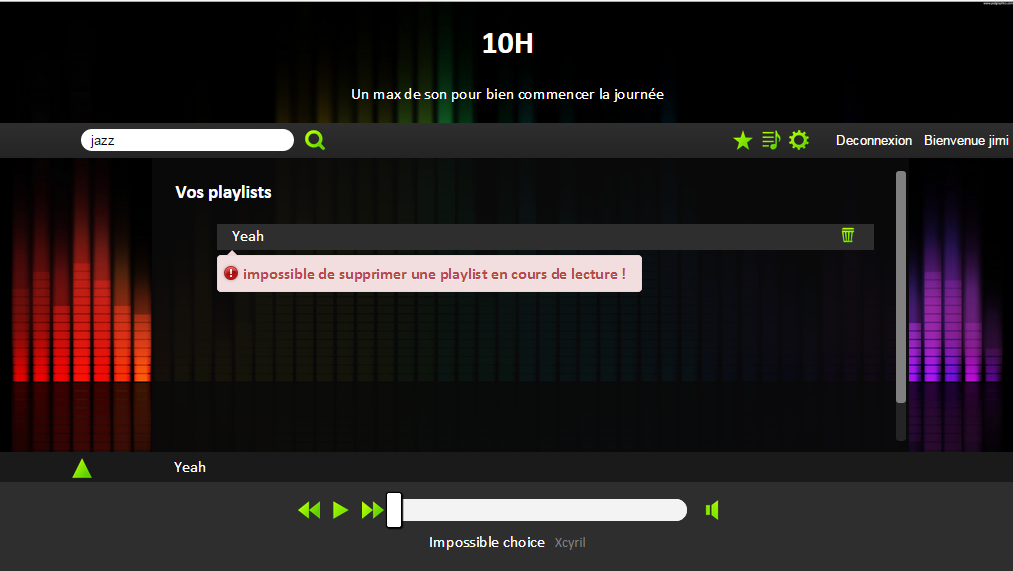
tentative de suppression d’une musique en cours de lecture dans la playlist

affichage de la page d’un artiste

(ici, depuis le nom de l’artiste de la musique en cours de lecture)



Affichage des playlists de l’utilisateur



Tentative de suppression d’une playlist en cours de lecture

# Limites de l’application

Notre application remplit les critères qui lui étaient imposé, mais peut tout de même être améliorée sur certains points :

Il serait plus confortable pour l’utilisateur, lors de la lecture d’une playlist, d’afficher la position en cours de lecture sur le total dans l’entête du volet de playlist, à côté du nom de la playlist.

Il serait également plus pratique de pouvoir bouger la popup d’ajout d’une musique à une playlist à la souris.

# Difficultés rencontrées

## Niveaux et attentes differentes

Le niveau des membres du groupe étant différents, ce ne fut pas évident de construire un site au niveau et au goût de tout le monde tout en faisant en sorte que tout le monde puisse participer de manière égale et significative. Nous avons pour cela distribué les tâches de manière adaptée aux membres du projet, et fait en sorte de mettre un membre qui aurait des lacunes avec un membre plus aisé ensembles sur une tâche jugée assez poussée.

## Compréhension du code

La compréhension du code afin de permettre à n’importe quel membre de le reprendre est un enjeu important du travail en groupe. Chacun ayant un niveau et une manière de coder différente, il a fallu construire un code clair et commenté.

# Conclusion

Ce projet fut enrichissant du point de vue du travail de groupe dans le sens où nous avons dû distribuer les tâches équitablement en fonction des capacités et des préférences de chaque membre, mais aussi instructif dans l’échange de connaissances entre ancien L2 MIASHS, et ancien DUT Informatique.

Nous nous sommes efforcés de mettre en œuvre toutes les notions vues en cours de web pendant ce semestre de L3, mais également les notions apprises en DUT Informatique, comme la protection des données (injections SQL), les patterns MVC et Active Record, le responsive design, etc…

Nous avons également fait en sorte que l’application ressemble à une application client du point de vue de l’interface, de l’ergonomie et du design, tout en restant une application web dans son fonctionnement.