Soundstream

TPI 2017 N’hairi Ilias

Table des matières

**Aucune entrée de table des matières n'a été trouvée.**

# Analyse préliminaire

## Introduction

Dans le cadre de mon travail pratique individuel je dois réaliser une application servant à écouter de la musique en streaming. Je dois aussi créer un site internet servant à gérer la base de données de cette application.

Objectifs de ce document :

* Présenter le processus d’élaboration du site web
* Documenter le code du site web
* Recenser les informations nécessaires à la reprise du projet

## Objectifs

Les objectifs du site web de gestion sont les suivants :

* Page de login ou création de compte
* Utilisateur
  + L’utilisateur peut uniquement consulter ses propres playlists
* Admin
  + Effacer un ou des utilisateurs
  + Supprimer les playlists d’un ou de plusieurs utilisateurs
  + Uploader des titres au format .mp3

# Analyse

## Analyse concurrentielle

Je n’ai pas trouvé grand-chose de ressemblant à ce site web. C’est selon moi juste un petit outil de gestion je n’ai pas besoin de beaucoup m’inspirer pour le réaliser.

## Cahier des charges détaillé

### Définition de l’audience

Le site web sera utilisé par les utilisateur de mon application ainsi que par les administrateurs qui pourrons gérer l’application depuis ce site.

### Définition du contenu et des fonctionnalités

* Les utilisateur peuvent créer un compte.
* Les utilisateurs peuvent se connecter et choisir quelle playlist consulter
* Les administrateurs peuvent se connecter et gérer les utilisateurs, leurs playlists ainsi que les titres sur la base de données.

### Maquette préliminaire

Pour la réalisation des maquettes j’ai utilisé « Balsamiq online » disponible à cette adresse :

<https://webdemo.balsamiq.com/>

Voici les maquettes que j’ai réalisé avant de me lancer dans le projet :

Fenêtre de connexion :

L’utilisateur peut se connecter ou cliquer sur le lien « Inscription » pour ouvrir la page d’inscription.

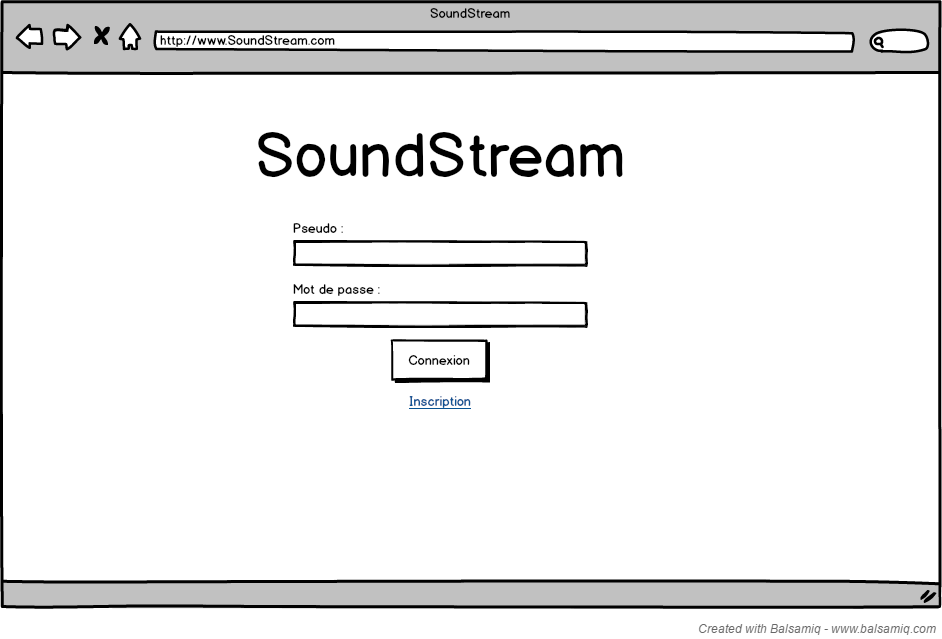


Figure 1 Fenêtre connexion

Fenêtre inscription :

L’utilisateur arrive depuis la fenêtre de connexion. Il peut soit s’inscrire soit retourner sur la fenêtre de connexion.

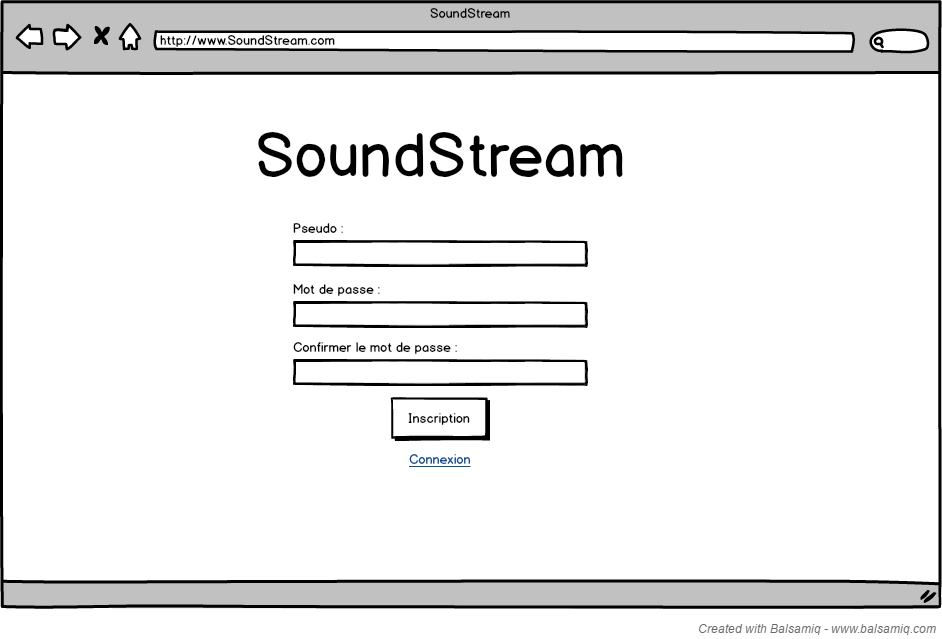


Figure 2 Fenêtre inscription

Fenêtre liste des playlists :

Une fois l’utilisateur connecté il arrive sur une page avec son pseudo et une liste de playlists sur lesquelles il peut cliquer pour les consulter.

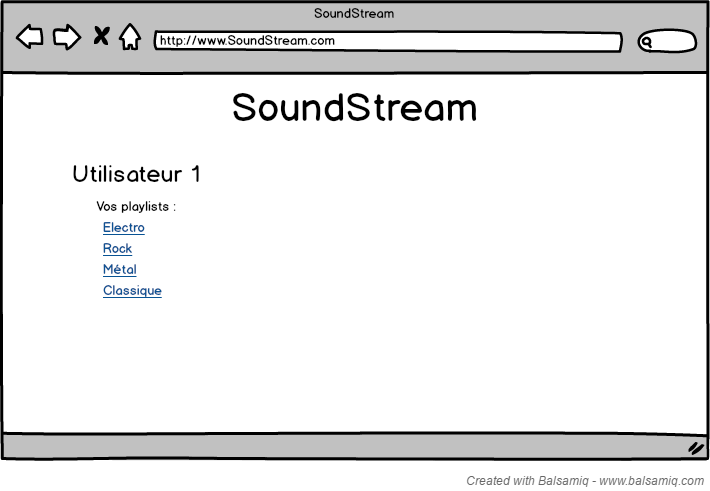


Figure 3 Fenêtre liste playlists

Fenêtre consultation d’une playlist :

Une fois que l’utilisateur a cliqué sur une playlist il arrive sur cette page contenant les informations sur la playlist qu’il à choisi.

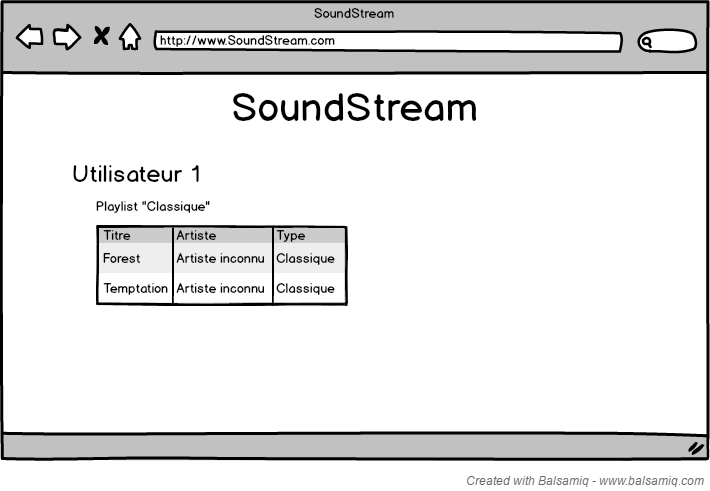


Figure 4 Fenêtre playlist

Fenêtre administration :

Une fois qu’un administrateur s’est connecté il arrive sur une page ou il a deux liens, un pour gérer les utilisateurs et l’autre pour l’ajout de fichiers dans la base de données.

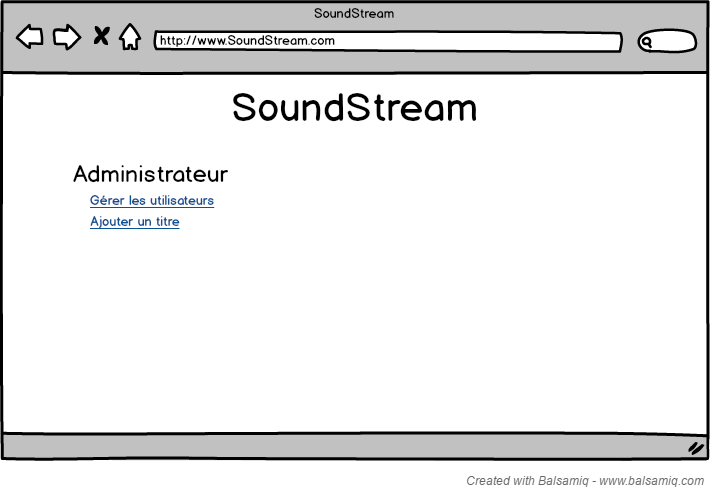


Figure 5 Fenêtre administration

Fenêtre gestion utilisateurs :

Une fois que l’administrateur a cliqué sur le lien « Gérer les utilisateurs » il arrive sur une page avec la liste des utilisateurs ou il peut soit consulter leurs playlists ou cocher les utilisateurs qu’il veut supprimer.

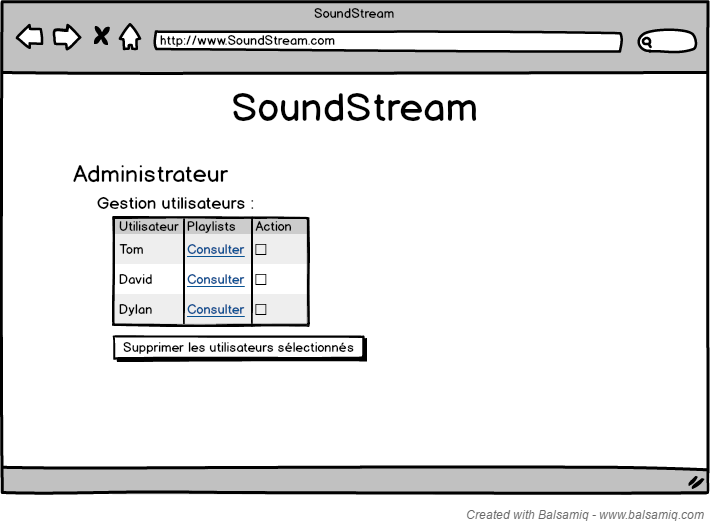


Figure Fenêtre gestion utilisateurs

Fenêtre gestion d’un utilisateur spécifique :

Quand l’administrateur clique sur « Consulter » il arrive sur la page de gestion dédié à l’utilisateur sélectionné. Depuis cette fenêtre l’administrateur peut supprimer les playlists de l’utilisateur sélectionné ou le supprimer en personne.

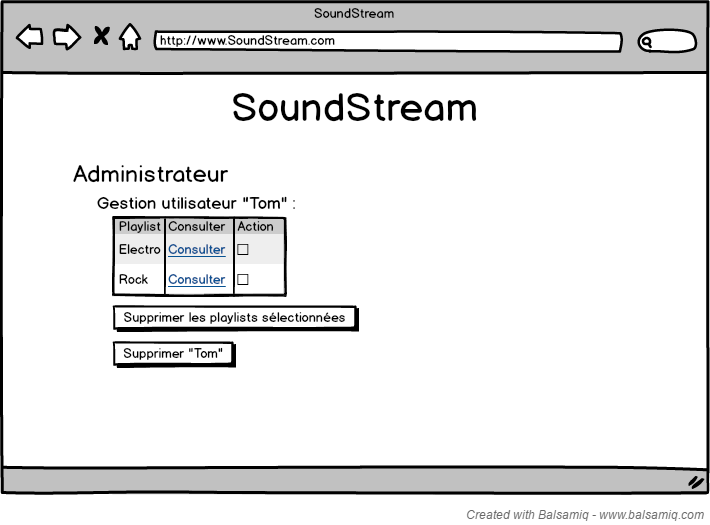


Figure Gestion dédié utilisateur

Depuis la page d’administration l’administrateur peut cliquer sur « Ajouter un titre » pour arriver sur la page contenant le formulaire d’ajout de titre. Depuis cette page il lui suffit de remplir le formulaire et d’upload le fichier .mp3 correspondant.

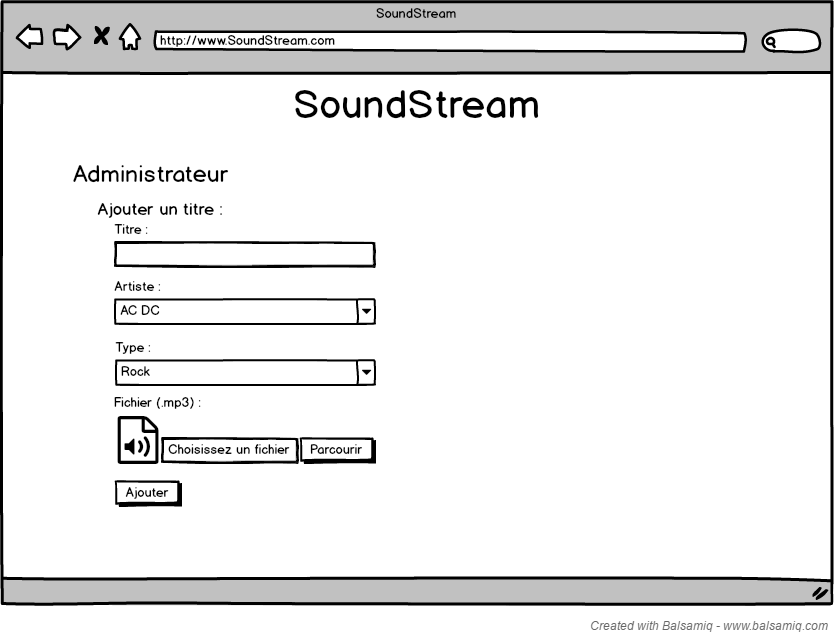


Figure Fenêtre ajout de titre

### Détermination des points faibles

Pour ce projet j’ai choisi d’utiliser la librairie Bootstrap, je la connais déjà légèrement mais je devrai me rafraichir les idées.

## Stratégie de tests

Pour tester mon application je vais réaliser des plans de tests ainsi que des rapports. Je donnerai aussi un plan à un de mes collègues pour qu’il le réalise vu qu’il n’a pas connaissance de mon application.

# Conception

## Analyse de l’environnement

Pour ce projet le matériel informatique est le suivant :

* Un ordinateur
* Un SSD 330 Go
* Clavier + souris
* Deux écrans

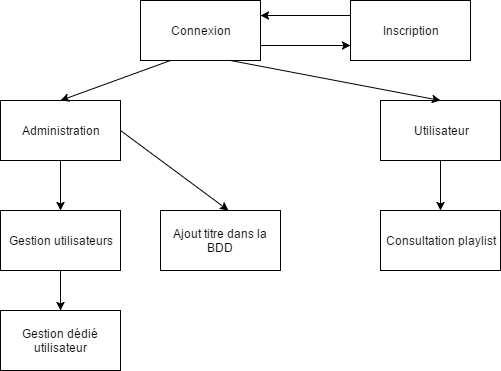
Logiciels utilisés pour ce projet :

* Atom
* MS Word 2016
* MS Excel 2016
* Notepad ++
* GitKraken
* Google Chrome
* EasyPhp
* PhpMyAdmin

## Détermination de l’arborescence du site et des rubriques

L’arborescence est séparée en trois parties :

1. La partie connexion/inscription qui est le départ du site
2. La partie utilisateur ou il pourra consulter ses playlists
3. Et la partie administration qui sert à gérer les morceaux ainsi que les utilisateurs



## Définition de la charte graphique

Concernant la charte graphique j’ai décidé de faire quelque chose de sobre tout comme l’application. De plus j’ai choisi d’utiliser la librairie Bootstrap car elle est très facile à utiliser.

## Maquette graphique

Voici les maquettes graphiques que j’ai réalisé en HTML dans le but de les rendre dynamiques avec du PHP.

## Interrogation

Voici les requêtes SQL que j’ai utilisé dans le site web avec une petite explication.

* SELECT \* FROM artists  
  Récupère les artistes dans le but de créer une liste déroulante.
* SELECT \* FROM types  
  Récupère les types dans le but de créer une liste déroulante.
* DELETE FROM playlists WHERE idPlaylist = :id  
  Cette requête me permet de supprimer un playlist et toutes les entrées correspondantes dans la table «Contain». Cependant pour que cela fonctionne en une requête il faut changer les contraintes en « ON DELETE CASCADE ».
* DELETE FROM users WHERE idUser = :id  
  Cette requête me permet de supprimer un playlist et toutes les entrées correspondantes dans la table « playlists » et «Contain». Cependant pour que cela fonctionne en une requête il faut changer les contraintes en « ON DELETE CASCADE ».
* SELECT \* FROM users WHERE pseudoUser = :pseudo AND passUser = :pass  
  Vérifie les informations de connexion fournies par l’utilisateur dans le but de la connexion.
* INSERT INTO users (pseudoUser,passUser,privilegesUser) VALUES (:pseudo,:pass,0)  
  Ajoute un utilisateur dans la base de données en fonctions des données fournies par lui-même.
* SELECT \* FROM users WHERE idUser = :id  
  Récupère les informations d’un utilisateur dans un but d’administration.
* SELECT \* FROM playlists WHERE idUser = :id  
  Récupère les playlists d’un utilisateur dans un but d’administration.
* SELECT \* FROM users WHERE privilegesUser != 1  
  Récupère tous les utilisateurs qui ne sont pas administrateur dans un but d’administration.
* INSERT INTO musics (titleMusic,idType,idArtist,fileName) VALUES (:title,:idType,:idArtist,:file)  
  Ajoute une musique dans la base de données.
* SELECT \* FROM playlists WHERE idPlaylist = :idPlaylist AND idUser = :idUser  
  Vérifie qu’une playlist appartient à un utilisateur. Cette requête est utilisée pour vérifier qu’un utilisateur non admin consulte une de ses playlists.
* SELECT namePlaylist,pseudoUser FROM playlists, users WHERE users.idUser = playlists.idUser AND playlists.idPlaylist = :idPlaylist  
  Récupère les information d’une playlist dans le but d’y afficher dans la page.
* SELECT \* FROM musics,playlists,types,contain,artists WHERE musics.idType = types.idType AND contain.idPlaylist = playlists.idPlaylist AND
* contain.idMusic = musics.idMusic AND playlists.idPlaylist = :idPlaylist AND artists.idArtist = musics.idArtist  
  Récupère toutes les musiques d’une playlist afin d’y afficher dans un tableau.
* SELECT \* FROM playlists WHERE idUser = :id  
  Récupère les playlists d’un utilisateur.

## Conception du code

Dans ce chapitre je vais détailler le code du site web. Tout est rédigé en HTML + PHP + SQL.

Pour se connecter l’utilisateur envoie ses informations puis elles sont vérifiés.  
Pour vérifier les informations j’exécute une requête avec celle-ci et si une seule ligne est retournée alors je stock les informations dans la session puis je redirige l’utilisateur en fonction de ses privilèges. La suite pour un utilisateur normal est simple j’affiche ses playlists dans une liste de lien.

Pour la page « playlist.php » c’est un peu plus compliqué. Premièrement je vérifie si la playlist appartient à l’utilisateur s’il n’est pas administrateur. De manière à ce que les administrateurs peuvent voir toutes les playlists et les utilisateurs que les leurs. Une fois cette vérification faite je récupère les informations ainsi que les musiques de la playlist.

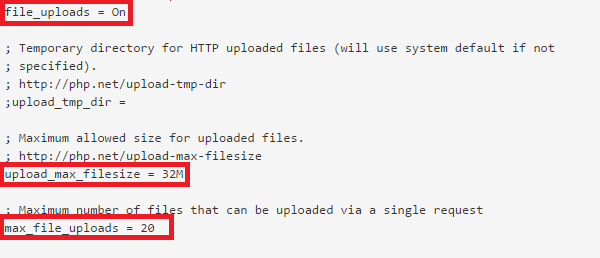


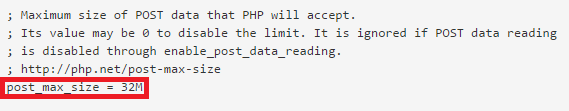
Pour la suppression d’utilisateurs ou de playlist c’est le même fonctionnement.



Je récupère un tableau contenant les id des playlists ou utilisateurs à supprimer puis je les supprime à l’aide du requête SQL. J’ai eu des soucis pour faire ceci à cause des contraintes de clés étrangers j’ai donc modifié les contraintes en « ON DELETE CASCADE » de manière à supprimer toutes les clés étrangères liées à la première.

Concernant l’ajout de titre je récupère les informations sur le nouveau morceau. Une fois cela fait je vérifie que le fichier est un .mp3 et lui génère un nom unique. Pour terminer j’ajoute tout dans la base de données après avoir sauvegarder le fichier dans le dossier « Music ». J’ai aussi eu besoin de modifier quelques lignes dans le fichier « php.ini ».







Mon fichier « functions.php » contient quelques fonctions utiles. La première « Database » sert à se connecter à la base de données et à créer une instance PDO.



Une fonction qui m’a été très utile durant mon travail est la fonction « Debug » que j’ai créé dans le but de pouvoir consulter le contenu de mes variables.



C’est une petite fonction mais elle est très utile.

J’ai aussi créer deux fonctions qui ont pour but de créer des listes déroulantes (pour les artistes et types).



## Ressources

Pour ce projet je n’ai pas eu besoin de beaucoup de ressources externe hormis la librairie Bootstrap.

<http://getbootstrap.com/>

J’ai aussi trouvé le bout de code servant à créer un nom unique sur un forum à cette adresse :

<http://stackoverflow.com/questions/36030621/create-a-unique-name-for-the-uploaded-image>

# Réalisation

## Dossier de réalisation

* http
  + css
    - Contient les fichiers .css de Bootstrap
  + Fonts
    - Contient les fichiers de police de Bootstrap
  + Js
    - Contient les fichiers javascript de Bootstrap
  + Music
    - Contient toutes les musiques que l’application utilisera
  + Php
    - Functions.php
      * Contient mes fonctions
  + addTitle.php
    - Contient le formulaire pour l’ajout de titres
  + Admin.php
    - Contient les liens des pages d’ajout de titre ou de gestion d’utilisateurs
  + Deconnexion.php
    - Contient quelques lignes servant à déconnecter l’utilisateur
  + delPlaylists.php
    - Contient le code servant à supprimer les playlists
  + delUsers.php
    - Contient le code servant à supprimer les utilisateurs
  + Index.php
    - Contient la page de connexion et un bout de code d’authentification
  + Inscription.php
    - Contient le formulaire d’inscription ainsi que le code d’ajout à la base de données
  + manageUser.php
    - Contient les formulaires de gestion de l’utilisateur et de ses playlists
  + manageUsers.php
    - Contient le formulaire de gestion des utilisateurs.
  + newMusic.php
    - Contient le code d’ajout de morceau à la base ainsi que la sauvegarde du fichier
  + Playlist.php
    - Contient l’affichage des playlists en fonction de l’id
  + Template.php
    - C’est la maquette que j’ai réalisée en premier pour me baser dessus pour chaque page.
  + User.php
    - La page affichant les playlists d’un utilisateur en fonction de son id dans la session.

## Description des tests

Pour tester mon site je vais réaliser un plan de test, les résultats seront dans le rapport de test que vous trouverez en annexe.

## Erreurs restantes

Il n’y a pas d’erreurs restantes.

# Conclusion

# Annexes

## Sources – bibliographie

* <http://getbootstrap.com/>
* <http://php.net>

## Procédure d’installation