**Rapport technique**



**Projet SoundHub**

**Auteurs**

Leroy, Maxime – 111 244 596

Longle, Henri – 111 244 597

**Équipe**

13

**Réalisé dans le cadre du cours**

GLO-2005 – Modèles et langages des bases de données pour ingénieurs

**Rapport présenté à**

L’Enseignant, Richard Khoury

**Remis le**

14 avril 2019

## Enonciation du problème et de ses exigences (2 points)

Aujourd’hui, YouTube est la plateforme dominante sur le marché du partage de contenus audiovisuel, et malgré sa croissance incessante, elle s’éloigne de plus en plus de sa communauté en axant davantage ses plans marketing sur des méthodes qui ont pour but de faire fructifier économiquement leur entreprise.

La communauté est donc mise de côté et c’est dans ce cadre qu’intervient notre application SoundHub, qui se veut être une plateforme novatrice utilisant les dernières technologies du web afin de combler les nouveaux besoins des utilisateurs de YouTube : un endroit ou l’information mise en avant est de qualité.

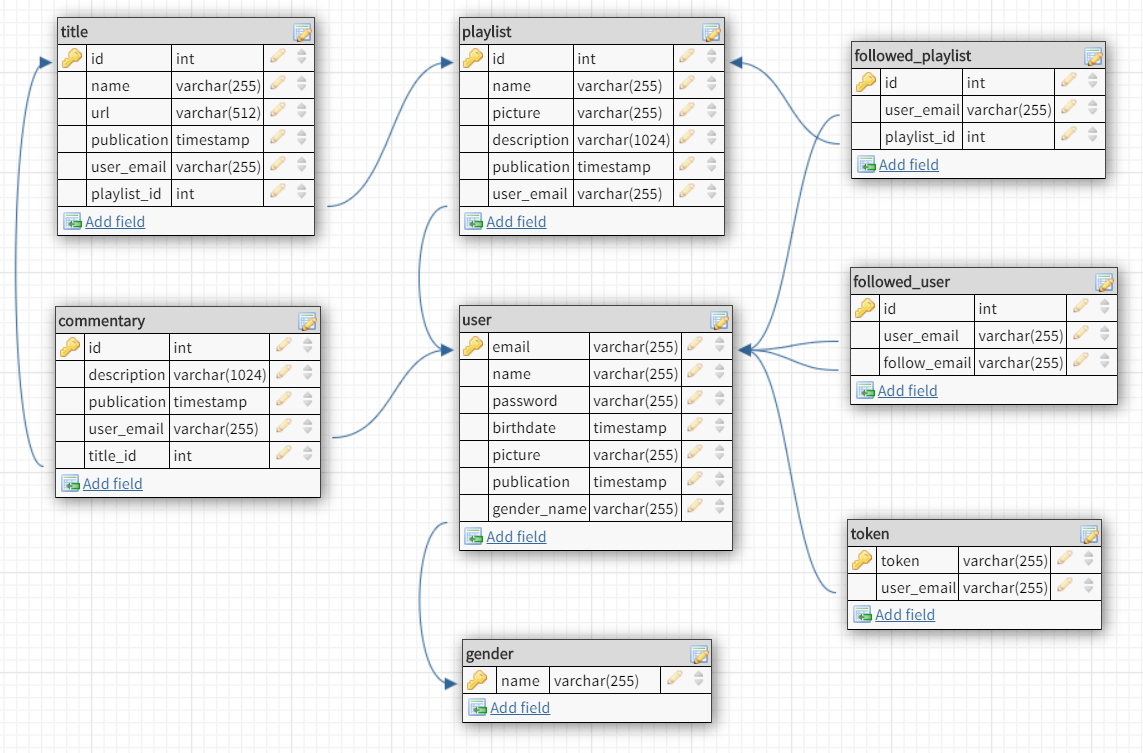
SoundHub est donc une application de partage de contenu musical de qualité, destiné à un cercle privé d’utilisateurs : il est impossible de consulter le contenu de la plateforme sans y être membre.

Le projet répond aux exigences suivantes :

* Créer un compte
* Se connecter
* Se déconnecter
* Afficher une liste de playlists
* Afficher les informations d’une playlist
* Afficher les titres associés à une playlist
* Afficher les informations d’un titre
* Afficher les commentaires associés à un titre
* Afficher sa liste d’abonnement aux utilisateurs
* Afficher les playlists que l’on suit
* Lire un titre via YouTube
* Lire un titre via un lien direct vers un mp4
* Editer une playlist
* Editer un titre
* Editer un profil
* Supprimer une playlist
* Supprimer un titre
* S’abonner à une playlist
* S’abonner à un utilisateur
* Se désabonner d’une playlist
* Se désabonner d’un utilisateur

## Modèle entité-relation du système (3 points)

## Modèle relationnel du système (3 points)



Légende :

*  : clé primaire
*  : clé étrangère

## Implémentation et fonctionnalités du niveau serveur de BD (4 points)

## Indexation des données, normalisation des relations, et optimisation des requêtes (4 points)

## Implémentation et fonctionnalités de la logique d’affaire (3 points)

## Implémentation et fonctionnalités de l’interface utilisateur (3 points)

## Sécurité du système (2 points)

La gestion des utilisateurs se fait via un système d’authentification dit : OAuth (<https://fr.wikipedia.org/wiki/OAuth>).

En effet, quand un utilisateur se connecte, il échange sa combinaison email/mot de passe contre un token qui lui permet d’accéder aux ressources du site, c’est ce qui l’identifie auprès du serveur python comme un utilisateur connecté.

On retrouve ce même système par exemple, sur Facebook, Twitter, YouTube…

Le mot de passe est bien évidemment hashé, et une fois insérée dans la base de données, il n’est jamais renvoyé au client.

Les injections sql ne sont pas possibles, car toutes les recherches se font sur les clés primaires via l’ORM python sqlalchemy.

En ce qui concerne l’insertion de mauvaises valeurs, des triggers sont mis en place pour éviter cela sur la base de données, mais également des vérifications dans les routes python sur le contenu du JSON reçu.

Aussi des formulaires restrictifs pour le client ont été implémentés, afin d’empêcher tout envoi de valeurs incorrectes, qui seront rejetées par l’API dans une optique de protection des données.

## Organisation, gestion de l’équipe, et division des tâches (1 points)

L’organisation a été simple puisque tout le travail réalisé a été fait en la présence de la totalité des membres sur toute la durée du projet : du premier au dernier commit inclus.

La gestion de l’équipe s’est faite entièrement via un système de branches et de pull requests sur un github public créé avec le compte laval github de Maxime (le lien est précisé dans la section revue de code).

La division des tâches s’est faite de manière naturelle en rapport avec les facilités et points faibles de chacun, à savoir :

* Maxime: logique back end API (Python, Flask)
* Henri : configuration et installation de docker (Docker)
* Maxime et Henri : logique front end (HTML, CSS, JavaScript), logique back end SQL (MySQL)

## Revue du code (5 points)

## **Lien du code source**

<https://github.com/maximeleroylaval/bdd/tree/master/projet>