**ORAL SAE**

Introduction :

Aujourd’hui, lorsque l’on souhaite reconnaitre un air musical, il suffit de sortir son smartphone et en quelques gestes, nous avons accès à ce que nous souhaitions grâce à l’application de reconnaissance musicale mondialement reconnu : Shazam. Le projet que nous allons vous présenter fut alors de confectionner une modélisation logiciel visant à offrir les mêmes fonctionnalités que Shazam à travers une nouvelle application web qu’on nommera Shazoom dans la suite de cette présentation.

Cas d’utilisation :

Nous avons entamé notre modélisation par l’identification des cas d’utilisation. Comme le montre notre diagramme, nous nous sommes limités à une application très simple afin de privilégier au maximum l’expérience utilisateur lors de l’implémentation de l’application. Ainsi, nous avons identifié deux acteurs principaux :

➢ L’administrateur qui aura les droits de supprimer ou d’ajouter des musiques, voir la liste de toutes les musiques enregistrées dans la base de données et modifier les informations d’une musique

➢ Tout utilisateur qui aura les droits d’identifier un son

Maquette :

Comme énoncé lors de l’introduction, nous avons décidé d’implémenter notre application en tant qu’application WEB. Nous vous présentons alors les pages que nous avons imaginé avec l’aide de Benjamin : la page d’accueil, la page qui s’affiche lorsqu’un son est trouvé, la page qui s’affiche lorsqu’aucun son n’a été trouvé, la page de connexion pour l’administrateur, la page administrateur avec toutes les musiques présentes en BDD, la page d’ajout d’une musique et enfin la page qui détaille toutes les musiques.

Séquence et Activité :

Dans cette seconde partie plus technique, afin de comprendre quelles vont être les interactions entre les différents objets et composants de notre application, nous avons réalisé plusieurs diagrammes de séquence et d’activité. En exemple, nous allons alors vous présenter et vous expliquer le diagramme de séquence et d’activité correspondant à la partie identification d’un son.

Tout d’abord, nous pouvons observer dans le diagramme de séquence qu’il y a plusieurs acteurs : l’utilisateur (simple ou administrateur), l’application et le SGBD qui vont interagir ensemble. Ceci se retrouve sur tous nos diagrammes. Dans le cas de l’identification d’un son, nous demandons donc à l’utilisateur d’enregistrer un extrait de 20 secondes et on demande au SGBD la liste de tous les hashs connus.

La partie qui cherche des correspondances se fait ensuite en plusieurs étapes. Premièrement, nous devons créer une constellation de points. Cette dernière s’effectue d’abord par le calcul du spectrogramme de plusieurs extrait de chanson. Un spectrogramme est une représentation d’un signal avec le temps en abscise, la fréquence en ordonnée et l’intensité de chaque fréquence représentée avec des couleurs. Une fois ce spectrogramme réalisé, on réalise la constellation de points en gardant seulement les fréquences les plus importantes et leur temps d’apparition. Ceci permet de simplifier la recherche et de limiter les perturbations liées au bruit.

Une fois, cette constellation trouvée, on crée des jetons aussi appelé hash, en prenant un point d’ancrage venant de la constellation puis un ensemble de points dans la zone voisine du point d’ancrage. Pour chaque point d’ancrage, on crée alors des pairs avec les points de la zone voisine puis on garde la fréquence du point d’ancrage, la fréquence du point voisin, la différence de temps d’apparition entre les deux points, ce qui crée le hash et enfin on garde également le temps d’apparition du point d’ancrage. Nous venons de créer une table de hashage pour notre extrait de son. On associe ensuite chaque hash à une musique grâce à un identifiant et on place cela dans une base de données.

Pour la partie qui cherche les correspondances, on a juste alors à effectuer ce travail sur l’extrait de l’utilisateur et comparer la table de hashage à toutes les tables présentes en base de données. Cependant, en comparant, les tables de hashage peuvent ne pas être exactement les mêmes en raison de parasite, ainsi pour régler ce problème, on crée un histogramme en utilisant la différence des temps d’apparition des points d’ancrage en abscise et le nombre de correspondance en ordonnée comme indiqué sur le brevet.

DAO :

Afin de vous présenter le cœur de notre modélisation nous avons jugé important de vous expliquer comment nous imaginions l’implémentation futur de notre application.

Vous pouvez observer ici notre diagramme de classe, ce dernier est basé sur le fait que nous voulions réaliser une implémentation tournée MVC (Modèle-Vue-Controller), ainsi on comprend que nos classes sont principalement des modèles simples qui vont permettre de structurer nos informations sous formes d’objets. Ces différents objets seront utilisés eux au cœur des contrôleur afin de réaliser les calculs et la logique nécessaire qui sera ensuite mis en forme à travers les vues de nos différentes pages.

Cependant certains le remarquerons déjà, nous avons l’intégralité de nos classes métiers qui sont dupliquées en classe DAO (Data Access Object). Effectivement, nous imaginons lors de l’implémentation l’utilisation d’un DAO qui va s’assurer de récupérer les informations des différentes tables de notre base de données puis les classes DAO vont se charger elle de faire la réalisation du mapping entre les informations de la base de données et les objets métiers. Ainsi tout le traitement en lien avec la base de données se feront au cœur de ces différentes classes afin de rendre l’utilisation des données au cœur de notre programme beaucoup plus facile pour les développeurs.

Une telle structuration semble nécessaire si on souhaite réaliser un code se voulant le plus propre et organisé possible

Algorithme :

Enfin, nous tenons à vous informer que pour comprendre plus en détail, l’implémentation que nous allons faire au second semestre surtout pour la partie identification d’un son, nous avons réalisé un pseudo code très détaillé que nous vous laisserons consulter lorsque vous aurez choisi notre modélisation 😉. Dans ce dernier, nous nous sommes beaucoup inspirés des fonctions offertes par les bibliothèques numpy, scipy en python et nous avons pris comme exemple de structure de données, des tableaux afin de stocker les différentes informations.

Conclusion :

Nous pouvons conclure, qu’avec une telle implémentation nous espérons obtenir un code qui se voudra lisible, organisé, simple à maintenir et à faire évoluer lors du second semestre.