L’application que nous avons souhaité implémenter est un site où chaque personne peut donner son avis sur une musique, expliquer une phrase d’une chanson, un texte en entier. Chaque musique sera récupéré, les paroles de la chanson pourront être écrite par un utilisateur.

Pour faire cela, nous allons utiliser l’API de Deezer afin de collecter les titres, les auteurs, les dates de sortie, toutes les informations dont nous aurons besoin. Il y aura la possibilité de faire une recherche par titre. Pour chaque recherche, nous appellerons utiliserons l’api afin de collecter les résultats correspondant à cette recherche. Chaque résultat de requête se présente sous forme de JSON, ce sera ensuite à nous de récupérer les éléments qui nous intéressent.

Un utilisateur pourra faire une recherche sur une musique et pourra donner son avis, commenter, écrire les paroles de la chanson ou même expliquer le texte. Il faudra donc au préalable avoir créée un compte et être connecté pour pouvoir poster un message. Les utilisateurs pourront aussi répondre à des messages d’autres personnes, mais aussi « aimer » le message.

Lorsque un utilisateur arrive sur la page principale du site, il peut faire une recherche, ou alors simplement visualiser les derniers commentaires. Si il souhaite faire une recherche, il tape l’élément de sa recherche dans une boite de texte, et le résultat s’affichera à l’écran. Il peut donner son avis sur la musique. Pour cela il doit être connecté. S’il ne possède pas de compte, il devra en créer un en précisant son adresse e-mail et le mot de passe qu’il souhaite utiliser ainsi que le pseudonyme sous lequel il souhaite apparaître. Si il possède un compte, il pourra se connecter en utilisant soit son adresse mail soit son pseudonyme, et bien entendu écrire le mot de passe. Lorsqu’il est connecté, il y a une boite de texte où il peut écrire ce qu’il souhaite. Une fois son texte écrit, il clique sur « Valider » et son message est posté. Le pseudonyme et la date sont affichées en plus du texte.

Sous chaque message, les utilisateurs connectés ont la possibilité de répondre à celui-ci. Si il clique sur le bouton « Répondre » une boite de texte s’affiche et il pourra écrire son message. Une fois ceci fait, il peut cliquer sur « Valider » et la réponse sera postée sous le message, avec le pseudonyme de la personne ainsi que la date à laquelle le message a été posté.

Un utilisateur peut aussi « aimer » un commentaire. Pour cela il a la possibilité de cliquer sur « J’aime » et le nombre de « Like » sera incrémenté de « 1 » sous le message.

Par défaut les messages seront triés par pertinence, c’est à dire que ceux ayant reçu le plus de « J’aime » seront affichés en premier. Sinon, il y a la possibilité d’afficher les message par date d’écriture.

L’utilisateur aura la possibilité de se déconnecter de son compte grâce à un bouton « se déconnecter », mais pourra aussi accéder à son profil ou il pourra changer son mot de passe, changer de pseudonyme une fois par mois.

Chaque utilisateur peut accéder au profil d’un autre utilisateur, et peut voir les messages que celui-ci a posté, les « like » qu’il a mis, mais aussi les musiques qu’il a aimé et ses playlists.

Un utilisateur pourra directement écouter une musique depuis le site, écouter une playlist, copier le lien vers la playlist et même être directement redirigé sur le site « Deezer » en fonction d’où il a cliqué. Si c’est une playlist, il sera redirigé sur Deezer au niveau de la playlist, si c’est un artiste, la page Deezer de l’artiste s’affichera, etc.

Pour ces différentes fonctionnalités il y aura plusieurs bases de données :

Tout d’abord, il y aura une table qui stockera les informations relatives à chaque utilisateur tel que le nom d’utilisateur, l’adresse mail, le mot de passe mais aussi un identifiant unique afin de différencier chaque utilisateur. Chaque utilisateur aura aussi les identifiants de tous les messages qu’il aura posté ainsi que les identifiants de tous les commentaires qu’il aura « liké ».

Une table sera crée pour les musiques. Chaque musique aura son propre identifiant afin de les différencier. De plus, le titre, le nom de l’artiste et son âge ainsi que le nom de l’album seront stockés.

De plus, chaque musique possédera une partie commentaire où tous les commentaires relatifs à l'élément seront stockés grâce à son identifiant. Les commentaires seront définis par leur identifiant et auront comme information l'utilisateur qui l'a posté, la date, le contenu du message, ainsi que le nombre de « J'aime » qu'il aura reçu.

Lorsqu'un utilisateur souhaite changer de pseudonyme, cela change ses informations grâce à son identifiant unique, et donc, le pseudonyme qui sera affiché sur chaque commentaire sera mis à jour automatiquement.

Dès qu'un nouveau commentaire est posté, les informations sont ajoutées à la base de donnée Commentaire, l'identifiant du commentaire est ajouté dans la liste des messages postés de l'utilisateur, il est aussi ajouté dans la base de donnée de l'endroit où il a été posté. Si c'est un commentaire sur un artiste, l'identifiant est ajouté dans les commentaires liées à cet artiste, si c'est un commentaire sur un album, l'identifiant est ajouté dans les commentaires liées à cet album, même chose pour les musiques.

Lorsqu'une recherche sera effectuée, si cette recherche a déjà été effectuée alors le serveur recherche les informations liées à cette recherche dans ses tables, sinon, une requête à l'API sera faite, et en fonction du type de la recherche.

Nous avons souhaité faire une approche ressource. En effet, les opérations et requêtes que nous allons effectué, feront soit appel à la base de donnée, soit un appel à l'API. Pour se faire, nous allons avoir une base de donnée comportant une table pour stocker les utilisateurs, une table pour stocker les musiques et une dernière table pour stocker les commentaires. De plus, nous ferons des appels à l'API pour collecter les informations dont nous avons besoin. On a un composant Authentification qui se chargera de la connexion et déconnexion d'un utilisateur, le composant Accueil qui comportera la barre de recherche ainsi que la liste des chansons les plus « Liké », le composant Résultat qui récupère le résultat de la requête faite à l'API ou de la requête faite à la base de donnée et qui va s'occuper ensuite du traitement de ce résultat et pour finir un composant Musique qui va gérer les informations relatives à la musique.

Le site va comporter 2 pages :
- L'accueil qui va afficher les chansons avec le plus de « J'aime », on interroge la base de donnée qui va récolter les 10 titres avec le plus de « J'aime » et les afficher, il y aura aussi les 10 derniers commentaires qui auront été posté en interrogeant aussi la base de donnée, les boutons pour se connecter/déconnecter, pour connecter une personne, on demandera à la base de donnée si les informations saisies sont valides et pour finir la barre de recherche avec un appel à l'API qui sera fait avec ce qui aura été écrit dans la barre de recherche.
- Une page pour les musiques, chaque musique aura son propre URL mais le contenu de la page sera affiché avec les informations qui auront été trouvée par l'API Deezer ainsi que ce qui sera stocké dans la base de donnée. En souhaitant poster un message ou en mettant un « J'aime », des requêtes seront effectuées pour mettre à jour les informations dans la base de donnée.

Description des requêtes :

Utilisateur :
- GET id → récupère un utilisateur grâce à son id
- POST id pseudo mail pwd → ajouter un utilisateur

Commentaire :
- GET id → récupère un commentaire grâce à son id
- GET id_user → récupère tous les commentaires d'un utilisateur
- GET id_music → récupère tous les commentaires d'une musique
- POST id id_user id_music → ajouter d'un commentaires

- POST id like dislike → mettre à jour le nombre de Like et Dislike

Like :
- GET id_user → récupère les like d’un utilisateur
- GET id_music → récupère les like d’une musique
- POST id_user id_music → ajouter un like

Dislike :
- GET id_user → récupère les dislikes d’un utilisateur
- GET id_music → récupère les dislikes d’une musique
- POST id_user id_music → ajouter un dislike