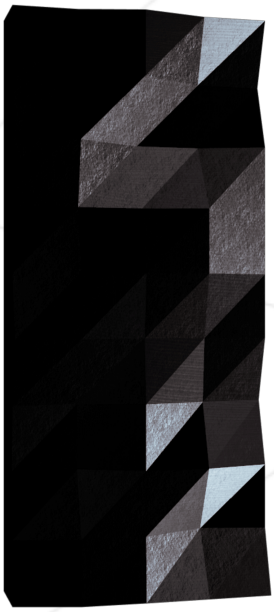

Music Room

Musique, Collaboration et Mobilité avec Deezer

42 Staff pedago@staff.42.fr
Deezer school@deezer.com

Résumé: Création d'une solution mobile complète autour de la musique et de l'expérience utilisateur. Ce sujet a été réalisé en collaboration avec Deezer.

Deezer for Android / Deezer for iOS

Table des matières

Chapitre I

Préambule

I.1 L'expérience musicale

La "consommation" de musiques est une expérience personnelle que chacun vit à sa manière :

- En s'entourant de musique au quotidien,
- En profitant de la musique lors de moments partagés avec ses amis,
- En utilisant la musique pour s'isoler ou se concentrer,
- En ne jurant que par le live, les concerts ou les festivals.

I.2 L'aventure Deezer

Deezer est une société Française créée en 2007 à partir d'une passion pour la musique et d'une problématique du fondateur pour partager cette musique avec ses amis. C'est à partir des constats précédents que la société oriente aujourd'hui ses offres et les fonctionnalités de son produit pour s'addresser au plus grand nombre. En quelques chiffres, **Deezer** en 2016 c'est plus de :

- 100 millions de playlists,
- 40 millions de titres,
- 6 millions d'utilisateurs payants,
- 700 serveurs,
- 180 pays,
- 2.5 peta-octets de donnée.

Deezer a également une très forte activité Business (B2B) au travers de contrats internationaux avec de nombreux opérateurs et partenaires tels que **Sonos**, **Orange**, **Vodafone**, **Bose**, **Bang & Olufsen**, **Apple**, **Google**. Chacun de ces accords est spécifique et permet de garantir la croissance du nombre d'utilisateurs qui peuvent profiter du produit. L'API et les SDK **Deezer** sont des mécanismes clés qui permettent de réaliser les intégrations avec ces partenaires.

Chapitre II

Objectifs

L'objectif de ce projet est d'aborder l'ensemble des concepts nécessaires à la création d'une application mobile, connectée et collaborative en prenant en compte les contraintes d'un produit réel.

Vous allez devoir penser à l'architecture client/serveur de votre solution, faire des choix pour le stockage de vos données, définir une API qui va servir de canal de communication entre votre serveur et vos clients, anticiper les problématiques de montée en charge et de sécurisation, et apprendre à travailler avec des techno tierce "third-parties" en intégrant des SDK externes.

Les sujets sont nombreux, il est donc conseillé de se répartir les tâches au sein de votre équipe projet.

Les services suivants vont devoir être implémentés :

- **Music Track Vote** - Génération live d'enchainement de musiques par vote,
- **Music Control Delegation** - Délégation du contrôle de l'écoute.
- **Music Playlist Editor** - Édition de playlists à plusieurs en temps réel,

Pour s'inscrire à ce projet, valider le `First Internship` est obligatoire. Toutefois, selon la partie qui vous intéresse, nous vous recommandons de valider aussi ces projets :

- **Partie mobile** - `ft_hangout` et/ou `Swifty Companion`
- **Partie Backend** - `Hypertube` et/ou `Friends with Benefits`

Chapitre III

Consignes générales

III.1 Consignes liées à l'architecture logicielle

Il n'y a pas de contrainte sur la stratégie ou le langage à utiliser pour le back-end : c'est à vous de mesurer les avantages des différentes technologies disponibles et d'identifier les solutions qui vous semblent avantageuses par rapport aux attentes du projet. Il faudra pouvoir défendre ces choix.

Vous ne devez pas committer de librairie que vous n'avez pas écrite par vous même dans votre repository. Les dépendances de votre projet doivent pouvoir être téléchargées automatiquement à partir d'un clone de votre repository à l'aide de Makefile ou de mécanismes similaires.

III.2 Consignes liées à l'expérience mobile

L'application mobile doit être implémentée au choix pour **Android** et/ou pour **iOS**. Vous pouvez très bien développer votre application en natif (Android SDK, UIKit, Xamarin) ou alors utiliser un framework hybride (React Native, Ionic...).
L'utilisateur doit pouvoir réaliser toutes les actions depuis l'application mobile.

Chapitre IV

Partie obligatoire

IV.1 Utilisateur

Quand l'utilisateur lance l'application pour la première fois, il doit se créer un compte depuis l'application. Il doit avoir le choix entre créer un compte avec mail/password ou avec un compte social (**Facebook** ou **Google**).

Une fois que l'utilisateur a créé son compte sur votre plateforme, l'application doit lui permettre d'attacher son compte **Deezer** et également au moins un autre compte social additionnel (**Facebook** ou **Google**)

Dans son compte, l'utilisateur doit pouvoir indiquer et maintenir à jour :

- ses informations publiques,
- ses informations restreintes à ses amis,
- ses informations privées,
- ses préférences musicales.

`Si l'utilisateur crée son compte avec mail/password, l'application doit imposer la validation du mail et proposer à l'utilisateur de réinitialiser son mot de passe en cas de perte du celui-ci.`

IV.2 Services

Depuis l'application l'utilisateur doit avoir accès à au moins 2 des 3 fonctionnalités suivantes, ie. **Music Track Vote**, **Music Playlist Editor** et **Music Control Delegation**.

IV.2.1 Music Track Vote

`Génération live d'enchainement de musiques par vote.`

Des personnes sont réunies dans un lieu avec de la musique (soirée, évènement...). Votre service doit permettre à chacun de suggérer ou de voter pour une des prochaines tracks qui sera intégrée dans la playlist actuelle. Si une track reçoit beaucoup de vote, elle peut remonter dans la liste pour être jouée plus tôt.

Une gestion de la visibilité (Public/Privé) doit être intégrée au service :

- Par défaut, votre évènement est publique.
- Tous les utilisateurs peuvent trouver votre évenement et voter, si votre évènement est publique.
- Seuls les utilisateurs invités peuvent trouver l'évènement et voter, si votre évènement est privé.

Une gestion des droits doit être intégrée au service :

- Par défaut, tout le monde peut voter.
- Seules les personnes invitées pourront voter, si les bons droits sont mis.
- Seules les personnes localisées à un endroit particulier durant une tranche horaire spécifique (par exemple entre 16h et 18h) pourront voter, si les bons droits sont mis.

`Attention à porter un soin particulier à la gestion des problématiques de concurrence : par exemple si plusieurs personnes votent pour différentes tracks ou pour la même track au sein d'une playlist.`

IV.2.2 Music Control Delegation

Délégation du contrôle de l'écoute.

Une gestion des droits doit être intégrée au service. Elle doit être spécifique à chaque appareil que l'utilisateur attache à son compte. L'utilisateur peut décider de déléguer le contrôle à un ou à plusieurs amis.

IV.2.3 Music Playlist Editor

Édition de playlists à plusieurs en temps réel.

Collaborer avec vos amis ou avec des personnes partageant un même centre d'intérêt musical sur la composition de playlists en temps réel. Les utilisateurs ont ainsi à la possibilité de créer des radios originales.

Une gestion de la visibilité (Public/Privé) doit être intégrée au service :

- Par défaut, une playlist est publique.
- Tous les utilisateurs peuvent la trouver et la consulter, si la playlist est référencée comme public.
- Seuls les utilisateurs invités peuvent trouver et consulter la playlist, si la playlist est référencée comme privée.

Une gestion des droits doit être intégrée au service :

- Par défaut, tout le monde peut éditer la playlist.
- Seuls les utilisateurs invités peuvent éditer la playlist, si les bons droits sont mis.

Attention à porter un soin particulier à la gestion des problématiques de concurrence : par exemple si plusieurs personnes déplacent différentes tracks ou la même track au sein d'une playlist.

IV.3 Serveur

Toutes les données du service doivent être stockées côté back-end. Le back-end est le référentiel, c'est lui qui possède et représente "la vérité vraie". Vous avez la possibilité d'utiliser la technologie et les frameworks que vous souhaitez (php, nodejs, golang, firebase, etc.).

IV.4 API

L'API va être le point d'accès pour toutes les applications à votre back-end. Des développeurs vont s'appuyer sur votre API pour l'intégrer dans divers contextes, la documentation de cette API est donc primordiale et les premiers développeurs à l'utiliser, c'est vous. Cette documentation de référence de l'API doit présenter les méthodes, inputs et outputs. Elle peut être auto-générée, par exemple avec Swagger.

Nous vous conseillons de réaliser une API qui s'appuie sur les principes REST mais d'autres principes existent (de nouvelles tendances émergent chaque jour). Vous devez dans tous les cas pouvoir justifier votre choix et expliquer ses caractéristiques.

Nous vous conseillons de vous appuyer sur JSON comme format d'échange avec l'API mais d'autres formats existent. Vous devez dans tous les cas pouvoir justifier votre choix et expliquer ses caractéristiques.

IV.5 Application mobile

Les applications ne doivent être que des "télécommandes" du back-end et l'adresse du back-end doit pouvoir être configurée au niveau de l'application à des fins de tests.

Le support de l'authentification via un service social (**Facebook** ou **Google**) doit être intégré dans l'application mobile. Des ressources sont disponibles sur : developers.facebook.com et developers.google.com.

Un SDK **Deezer** doit être intégré dans l'application mobile. Les versions **Android** et **iOS** du SDK **Deezer** s'appuient sur l'API **Deezer** et intègrent le player audio adapté à la plateforme. Ils sont disponibles sur : developers.deezer.com

En **mode non-authentifié**, l'API et le SDK **Deezer** permettent d'accéder sans limitation aux metadata du catalogue, artistes, albums et pistes musicales. En **mode authentifié**, ils permettent de manipuler les données, playlists et préférences de l'utilisateur connecté.

IV.6 Sécurisation

Un utilisateur authentifié sur votre solution doit avoir accès à ses données mais pas aux données des autres utilisateurs. Vous devez prévoir les comportements malicieux (brut force de votre API, vol de session, etc.) mais on ne s'attend pas que votre API soit impénétrable :

- vous devez intégrer des mécanismes pour protéger vos utilisateurs,
- vous devez identifier d'autres risques et pouvoir expliquer les protections adaptées.

Des logs doivent être générées au niveau du back-end pour toute actions réalisées par les applications mobiles. Les applications mobiles doivent pouvoir être identifiées par le serveur :

- Plateforme (Android, iOS, etc.),
- Device (iPhone 6G, iPad Air, Samsung Edge, etc.),
- Application Version.

IV.7 Montée en charge

Vous devez être en mesure d'évaluer la charge que peut supporter votre API et votre back-end, c'est à dire justifier et mesurer le nombre d'utilisateurs qui peuvent utiliser simultanément vos 3 services. Vous pouvez par exemple utiliser AB (Apache Benchmark), Gatling, Siege, Tsung, JMeter.

N'oubliez pas de préciser les caractéristiques serveurs (CPU, RAM, Cloud ou Premise, etc.). On s'attend à ce que le nombre d'utilisateurs maximums restent cohérent avec le choix de la plateforme ie. quelques dizaines pour un Raspberry et quelques milliers pour un serveur d'entrée de gamme.

IV.8 Agilité, qualité et intégration continue

Les couches et fonctionnalités à implémenter dans le cadre de ce projet sont nombreuses et nécessitent de travailler en équipe. Nous vous conseillons de vous organiser pour travailler en mode agile, de savoir remettre en cause vos choix et priorité et de mettre en place des tests unitaires spécifiques à chaque couche.

Chapitre V

Partie bonus

Les bonus que l'on vous suggère ici correspondent à des problématiques réelles et business à prendre en compte de le cas d'applications mobiles.

V.1 Support du multi-plateforme

Une fois que vous avez un serveur et une API fonctionnelle vous avez la possibilité de rendre disponible vos services au travers de différentes platformes. Vous supportez déjà maintenant normalement une plateforme mobile (Android ou iOS) dans le cadre de ce bonus vous pouvez rendre également disponible votre service sur le web en mode "responsive" pour s'adapter à toutes les tailles d'écran.

En fonction de vos choix technologiques, le support du web peut nécessiter de ré-écrire plus au moins complètement votre code client.

V.2 Réflexion autour de l'IoT

Vous pouvez intégrer un mécanisme comme des beacons sur votre event. Par exemple quand des personnes arrivent à proximiter d'un évènement public enregistré sur votre service, ils reçoivent automatiquement des informations sur l'événement, les accès, types de musique, etc.

V.3 Offre Gratuite vs. Offre Payante

Les solutions web et les applications mobiles proposées de nos jours permettent très souvent à l'utilisateur de choisir entre une offre gratuite de qualité mais limitée et une ou plusieurs offres payantes plus complètes. C'est le cas en particulier du produit **Deezer** qui a une offre **Freemium**, une offre **Premium** et une offre **Elite**.

L'objectif de ce bonus est d'intégrer ce type de logique dans votre application. Il s'agira alors de permettre à vos utilisateurs de pouvoir basculer entre 2 offres que vous aurez définies au préalable. Certaines fonctionnalités de votre application (par exemple **Music Playlist Editor**) devront alors n'être accessibles qu'aux utilisateurs ayant souscrit à votre offre payante.

V.4 Offline Mode

L'objectif de ce bonus est de mettre en place un mécanisme permettant à l'utilisateur de profiter de l'application dans des moments où il n'a pas de réseau, comme par exemple dans les transports - on parle alors de "mode déconnecté". Dans ce cas, la qualité de l'expérience et les possibilités offertes par les services peuvent être différentes.

Dans le cas d'une utilisation offline de l'application, une synchronisation est à prévoir. Attention aux problèmes qui viennent avec les mécanismes de synchronisation :

- Gestion des conflits et des concurrences,
- Gestion des données obsolètes côté application mobile.

`Le support du mode offline est intéressant en 2016 car la connectivité n'est pas optimale partout, mais celà peut changer avec l'évolution des couvertures mobiles et des offres haut-débit.`

Chapitre VI

Rendu et peer-évaluation

VI.1 Rendu

Rendez-votre travail sur votre dépot GIT comme d'habitude : seul le travail présent sur votre dépot sera évalué en soutenance.

VI.2 Évaluation

La qualité de la réflexion menée et l'originalité de l'expérience proposée doivent être considérées positivement.