



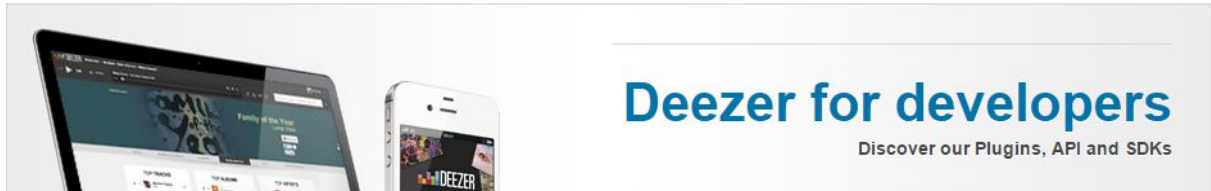
Projet 2 - Deez Web

Développement Front-end - Brief

Objectif

Vous devez réaliser une application web permettant de rechercher, explorer et organiser des **musiques**, des **albums** et des **artistes** en vous aidant de l'API développeurs gratuite fournie par le service Deezer.

Le service (API) Deezer



L'API que vous utiliserez sera celle de Deezer : <https://developers.deezer.com>



Un compte Deezer gratuit est nécessaire pour accéder aux données.
Une fois votre compte créé, logguez-vous via le bouton **Login**.

Une fois loggué, vous accèderez au panneau suivant :

Introduction
Global parameters
API errors
AUTHENTICATION
OAuth
Permissions
TOOL
API Explorer
oEmbed
API OBJECTS
Album
Artist
Chart
Comment
Editorial
Episode
Genre
Infos
Options
Playlist
Podcast
Radio
Search
Track
User
ACTIONS
Create/Edit (POST)
Delete (DELETE)

Introduction

Unlimited Access, without stress, without identification. Deezer Simple API provides a nice set of services to build up web applications allowing the discovery of Deezer's music catalogue.

Requests

You may use ordinary HTTP GET messages. The base URL for each API method looks like the following

```
https://api.deezer.com/version/service/id/method/?parameters
```

Query quota

The number of requests per second is limited to 50 requests / 5 seconds.

Requests format examples

```
https://api.deezer.com/user/2529
```

```
https://api.deezer.com/user/2529/playlists
```

```
https://api.deezer.com/album/302127
```

Response format

In order to search the specified artist name 'eminem', and get xml back, you can do :

```
GET /search/artist/?q=eminem&index=0&limit=2&output=xml
```

The available formats are those following :

DESCRIPTION	ACCEPT HEADER	EXTENSION
JSON	application/json	.json
JSONP	application/json	.json
XML	application/xml, text/xml	.xml
PHP	application/xml, application/json	.php

L'API de Deezer est très riche et permet de nombreuses choses. Néanmoins, les éléments qui vont nous intéresser seront ceux encadrés en vert sur la capture, à savoir :

- Album
- Artist
- Search
- Track

Dans un premier temps, vous prendrez quelques instants pour faire le tour de ces 4 "endpoints" afin de comprendre comment les utiliser.

Vous êtes encouragés à effectuer ce premier travail d'approche en groupe durant la journée de lancement.

Pages

Le projet que vous devez réaliser est une **application web utilisant les informations de l'API Deezer**. Votre application doit être constituée de 5 pages au total :

- La page de recherche (accueil)
- La page détaillant un titre
- La page détaillant un album
- La page détaillant un artiste
- La page regroupant les titres favoris de l'utilisateur de l'application Deez'Web

Détails et instructions pour chaque page

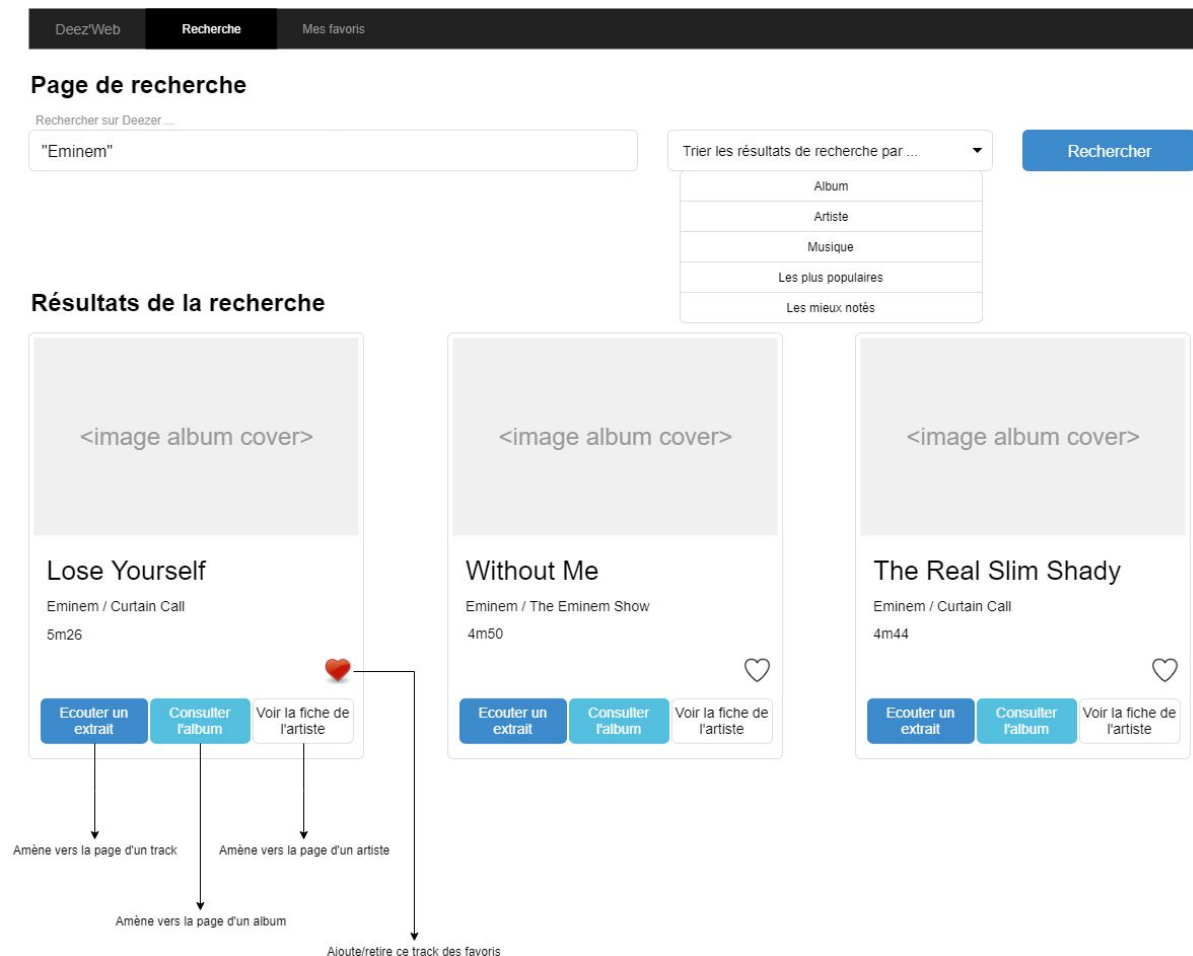
Les visuels qui suivent sont présents à des fins d'illustration. **Il ne s'agit pas de maquettes à intégrer**. Vous restez maître de l'aspect visuel, de l'agencement des informations et de l'UX (expérience utilisateur) de votre interface finale !

Vous devez faire preuve de bon sens et de créativité pour présenter au mieux les informations sur chaque page.

NB : Ces visuels sont également disponible au format PNG, et joints avec ce brief

1. Page d'accueil

Cette page constituant la home de l'application donne la possibilité aux utilisateurs de chercher des titres sur Deezer et d'afficher des résultats de recherche en dessous du formulaire.



La page doit être composée d'un formulaire avec au minimum 2 champs :

- Un champs pour rechercher un titre ("q")
- Un champs pour préciser le mode de tri ("order") et permettant de trier sur 5 critères :
 - par Album (croissant)
 - par Artiste (croissant)
 - par Titre (croissant)
 - par popularité (croissant)
 - par rang (croissant)

Ces deux paramètres peuvent être passés à l'API de Deezer de la façon suivante :

https://api.deezer.com/search?q=eminem&order=RATING_ASC

Plus de détails sur le endpoint "Search" de l'API Deezer :

<https://developers.deezer.com/api/search>

Les résultats de la recherche doivent être présentés sous forme de liste de musiques (tracks) et afficher pour chaque résultat les informations suivantes :

- L'image de l'album associé
- Le titre du morceau
- L'artiste et l'album
- La durée du titre

Chaque résultat doit en plus permettre de naviguer vers la page de l'album concerné, de l'artiste concerné ou du titre (track) en lui-même.

Un bouton permettant d'ajouter/retirer des favoris doit être présent.

2. Page Album

Cette page doit afficher les caractéristiques complètes d'un album grâce aux informations de l'API Deezer.

Lien vers le endpoint "Album" de l'API Deezer :

<https://developers.deezer.com/api/album>

On arrive sur cette page en ayant cliqué préalablement sur le lien d'un album n'importe où depuis l'application. Il faudra par conséquent toujours avoir un identifiant d'album présent dans l'URL, par exemple :

[album.html?id=15542154](#) (renseignez-vous en JavaScript comment récupérer les paramètres de la chaîne de requête)

Dans le cas contraire, l'accès à cette page n'est pas possible.

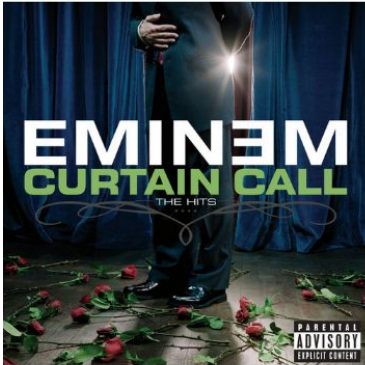
DeezWeb

Recherche

Mes favoris

Album : Curtain Call

Artiste : [Eminem](#) → Amène vers la page d'un artiste



Liste des tracks de cet album

Intro (Curtain Call) (0m34)
Fack (3m26)
The Way I Am (4m50)
My Name Is (4m27)
Stan (6m46)
Lose Yourself (5m26)

Voir l'album sur Deezer

★★★★☆

Le click sur un titre amène vers la page détaillant ce titre (track)

Cette page doit au minimum afficher les informations suivantes :

- Titre de l'album
- Photo de l'album
- Artiste (auteur) de l'album
- Liste des titres (tracks) de cet album
- Le rang (ranking) de cet album sur Deezer
- Un lien menant à l'album sur Deezer

3. Page Artiste

Cette page doit afficher les caractéristiques complètes d'un artiste grâce aux informations de l'API Deezer.

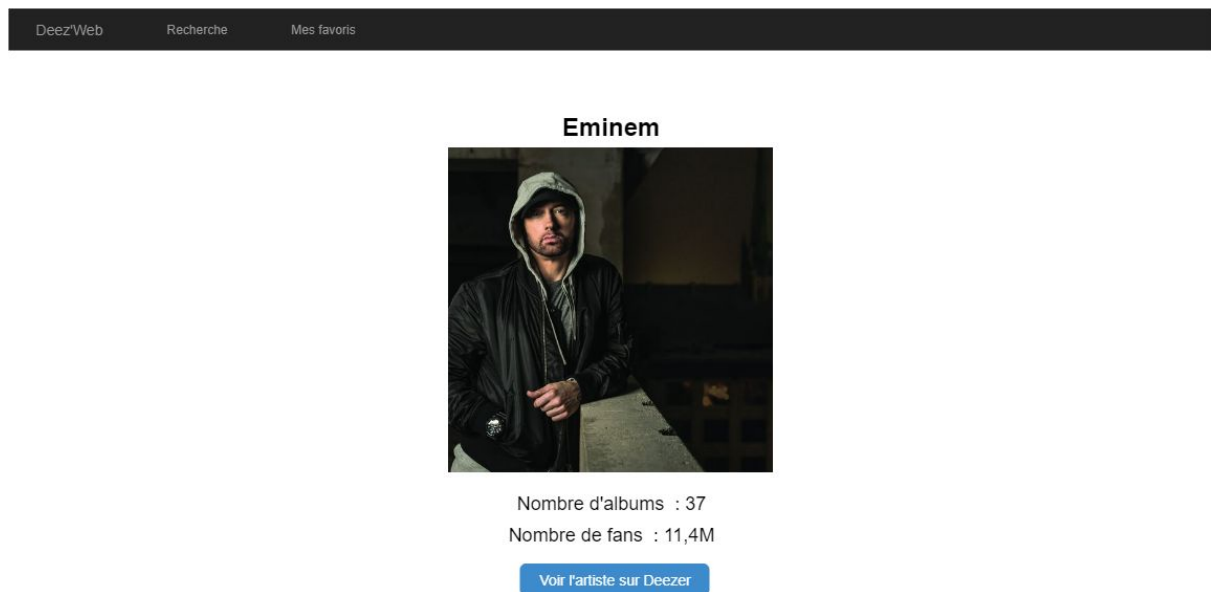
Lien vers le endpoint "Artist" de l'API Deezer :

<https://developers.deezer.com/api/artist>

On arrive sur cette page en ayant cliqué préalablement sur le lien d'un artiste n'importe où depuis l'application. Il faudra par conséquent toujours avoir un identifiant d'artiste présent dans l'URL, par exemple :

[artist.html?id=15542154](#) (renseignez-vous en JavaScript comment récupérer les paramètres de la chaîne de requête)

Dans le cas contraire, l'accès à cette page n'est pas possible.



Cette page doit au minimum afficher les informations suivantes :

- Nom de l'artiste
- Photo de l'artiste
- Nombre d'albums
- Nombre de fans
- Un lien menant à la fiche de l'artiste sur Deezer

4. Page d'un Titre (track)

De la même manière, cette page doit afficher les caractéristiques complètes d'un titre (track) grâce aux informations de l'API Deezer.

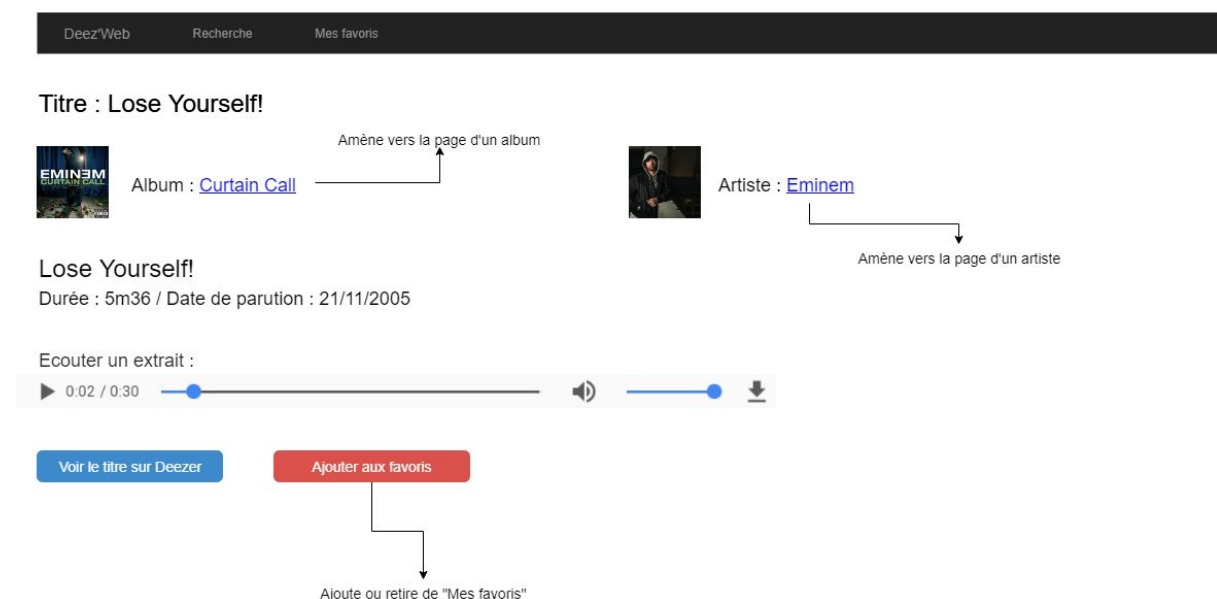
Lien vers le endpoint "Track" de l'API Deezer :

<https://developers.deezer.com/api/track>

On arrive sur cette page en ayant cliqué préalablement sur le lien d'un track n'importe où depuis l'application. Il faudra par conséquent toujours avoir un identifiant de track présent dans l'URL, par exemple :

<track.html?id=15542154> (renseignez-vous en JavaScript comment récupérer les paramètres de la chaîne de requête)

Dans le cas contraire, l'accès à cette page n'est pas possible.



Cette page doit au minimum afficher les informations suivantes :

- Titre du morceau
- Photo et titre de l'album
- Photo et nom de l'artiste
- Durée totale du morceau
- Date de parution
- Un player audio permettant d'écouter un extrait (preview)
- Un lien menant à ce titre sur Deezer
- Un bouton pour ajouter/retirer des favoris

5. Page “Mes Favoris”

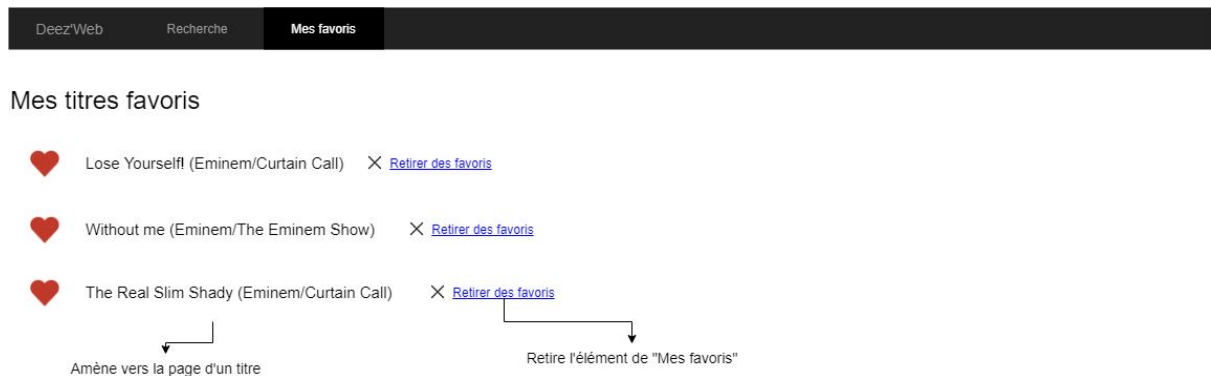
Cette page est accessible depuis le menu principal et regroupe simplement les favoris ajoutés par l'utilisateur pendant sa navigation sur l'application.

Les informations de favoris doivent être sauvegardées de façon permanente dans le **localStorage** du navigateur web.

Un rappel vers le lien d'Alsacrérations expliquant en détails le fonctionnement de l'API Web Storage :

<https://www.alsacreations.com/article/lire/1402-web-storage-localstorage-sessionstorage.html>

(Souvenez-vous aussi qu'il est possible de sauvegarder des objets littéraux JavaScript complets dans le localStorage grâce à [JSON.stringify](#))



Cette page doit être composée au minimum d'une liste des tracks ajoutés en favori par l'utilisateur. Chaque élément de la liste doit afficher :

- Le titre du track
- L'auteur
- L'album

... et doit pouvoir donner la possibilité de supprimer cet élément des favoris (donc simplement du localStorage)

Modalités du projet

Ce projet est à réaliser sur une période d'une semaine, en centre de formation, en présence d'un intervenant assurant le suivi technique du projet.

- Vous devez **obligatoirement être présent** durant les heures de suivi.
- La communication et l'échange entre les étudiants est encouragé, cependant le travail à fournir reste **individuel**.

Technologies

Vous devez utiliser vos connaissances en HTML, CSS et JavaScript pour réaliser ce projet.

- L'utilisation d'un framework JavaScript (e.g. Vue.js) **n'est pas obligatoire**.
- L'utilisation d'une bibliothèque JavaScript (e.g. jQuery) **n'est pas obligatoire**.

Cela étant dit, il est très fortement conseillé d'en choisir un parmi les deux ! Choisissez en fonction de vos capacités, n'utilisez pas un outil que vous ne vous sentez pas de maîtriser !

Rendu final

Le rendu s'effectue le Vendredi en fin de journée, sous la forme d'un fichier (archive compressée) à remettre sur un disque dur à l'intervenant.

Pour des raisons pratiques, aucun envoi par email n'est possible !

Votre archive doit respecter le format suivant : **DeezWeb_NOM_Prenom.zip**

Critères de notation

Seront pris en compte les critères suivant pour la note du projet :

- Application fonctionnelle (respect des fonctionnalités demandés sur chacune des 5 pages)
(10 points)
- Ergonomie et qualité de l'interface
(2 points)

- Qualité du code HTML, CSS et JavaScript (respect des normes W3C)
(2 points)
- Maîtrise de la bibliothèque (ou du framework) JavaScript utilisé
(4 points)
- Utilisation conforme d'AJAX et de l'API Deezer
(2 points)

Total sur **20 points**.

Points bonus :

Des points bonus pourront être attribués à la discrétion du formateur sur les critères suivants:

- Prise d'initiative sur l'amélioration des fonctionnalités demandées.
- Ajout d'animations améliorant l'ergonomie et l'interface.
- Gestion des erreurs :
 - Dans le cas où l'application n'a plus de réseau
 - Dans le cas où un identifiant d'artiste n'existe pas
 - ...

De la même manière, des points pourront être retirés en cas de constat de triche évidente, de non-présence durant les heures de cours, ou de mauvais comportement.