

Année 2020-21

M2 Informatique

Mémoire de stage



Développement d'un site web
de création de playlists :
implémentation et optimisation

par Jordan ARRIGO



1 - Introduction	3
1-1 Le laboratoire LIG	3
1-2 Problématique du stage	4
2 - Contributions apportées durant le stage	6
2-1 GazePlay	6
2-1-1 Qu'est-ce que c'est ?	6
2-1-2 Existant	6
2-1-3 Ma contribution	8
ProgressIndicator :	8
Durée de fixation :	8
2-2 GazeMedia Player	9
2-2-1 Qu'est-ce que c'est ?	9
2-2-2 Existant	9
2-2-3 Ma contribution	10
Page Connexion :	10
Page Playlist :	13
Page Youtube :	26
Page Spotify :	29
Page Deezer :	32
3 - Technologies utilisées	35
3-1 Langages utilisés	35
3-2 Logiciels utilisés	36
4 - Gestion du travail	37
4-1 Organisation générale	37
4-2 Organisation d'une fonctionnalité	37
5 - Bilan et perspectives	37
5-1 Bilan	37
5-2 Perspectives	38
6 - Remerciements	39
7 - Annexes	40
7-1 Fenêtre importation vidéo	40
7-2 Json Validator	40
7-3 Lecteur Spotify	41
7-4 Lecteur Deezer	41
7-5 Lecteur musique importé	42
7-6 Lecteur vidéo importé	42

1 - Introduction

1-1 Le laboratoire LIG

Le Laboratoire d'Informatique de Grenoble (LIG) est un laboratoire de recherche français en informatique, créé le 1^{er} janvier 2007. Celui-ci est sous la tutelle conjointe de l'université Grenoble-Alpes, de l'Institut Polytechnique de Grenoble et du CNRS. Il est partenaire de l'INRIA.



Le LIG rassemble près de 500 chercheurs, enseignants-chercheurs, doctorants et personnels en support à la recherche. Ils relèvent des différents organismes et sont répartis sur trois sites du LIG : le campus, Minatec et Montbonnot.

L'ambition est de s'appuyer sur la complémentarité et la qualité reconnue des 24 équipes de recherche du LIG pour contribuer au développement des aspects fondamentaux de l'informatique (modèles, langages, méthodes, algorithmes) et pour développer une synergie entre les défis conceptuels, technologiques et sociétaux associés à cette discipline.

La diversité et la dynamicité des données, des services, des dispositifs d'interaction et des contextes d'usage imposent l'évolution des systèmes et des logiciels pour en garantir des propriétés essentielles telles que leur fiabilité, performance, autonomie et adaptabilité. Relever ces défis trouve une résonance dans les cinq axes thématiques de recherche explorés au LIG.

Ces 5 axes sont :

- > Génie des logiciels et des systèmes d'information
- > Méthodes formelles, modèles et langages
- > Systèmes intelligents pour les données, les connaissances et les humains
- > Systèmes interactifs et cognitifs
- > Systèmes répartis, calculs parallèles et réseaux

Le LIG se veut être un laboratoire centré sur les fondements et le développement des sciences informatiques, tout en veillant à une ouverture ambitieuse sur la société pour en accompagner les nouveaux défis.

1-2 Problématique du stage

Le projet InterAACTIONBox (<http://interaactionbox.fr>) soutenu par un financement de l'Association Française du Syndrome de Rett vise à aider les personnes en situation de handicap cognitif. Ce handicap les empêche de communiquer par la parole et de se développer intellectuellement via cette unique modalité langagière. Il s'agit de concevoir un dispositif informatique à moindre coût, proposant des applications au plus près de la recherche et des utilisateurs, ouvert, adaptable et évolutif qui vont permettre l'éveil, le jeu, l'éducation mais aussi le développement communicatif et langagier des enfants avec un polyhandicap.

Le projet se positionne au carrefour de la cognition, de l'interaction homme-machine et du traitement automatique de la langue et de la parole. Il fédère les forces de chercheurs issus de plusieurs laboratoires de l'Université Grenoble Alpes, informaticiens, traducteurs, linguistes et psycholinguistes, dont certains sont aussi parents d'enfants en situation de polyhandicap. Le projet est aussi développé en collaboration avec des professionnels du terrain en orthophonie, ergothérapie, orthoptie et neuropsychologie ainsi que des personnes proches du quotidien des personnes concernées (éducateurs et autres professionnels intervenant en institution et des aidants).

Cet appareil intègre un écran tactile et est compatible avec des appareils oculométriques, précis, robustes, et peu onéreux. Le but est de faire un prix avoisinant les 500 € mais pour le moment on est un peu plus haut.

Sur le plan logiciel, il s'agit de concevoir et d'implémenter une batterie de logiciels libres et ouverts. Ces logiciels sont actuellement développés au Laboratoire d'Informatique de Grenoble, en collaboration avec des membres du Gipsa-Lab en interaction avec des étudiants de différents cursus et avec des objectifs d'apprentissage, de divertissement mais aussi d'évaluation.

On y retrouve :

- GazePlay (développé en Java - JavaFx), une plateforme de jeux utilisables par oculométrie en développement depuis 2016 et en constante évolution proposant à ce jour plus de 60 jeux créatifs, ludiques ou sérieux permettant d'aider à l'acquisition de compétences (action-réaction, sélection, littératie, mémorisation, etc.). Sur le long terme, GazePlay comportera aussi une partie évaluation (GazePlay-Eval) qui fait l'objet de projets de mémoire en neuropsychologie ;
- VisualSceneDisPlay (développé en angular - un Framework JavaScript), un logiciel interactif et configurable de scènes visuelles pour apprendre le vocabulaire de base aux enfants, tout en faisant un premier pas vers la Communication Alternative et Augmentée ;
- Augcom (développé en angular - un Framework JavaScript), un outil de grille de Communication Alternative et Augmenté, basé sur des études comparatives des différents logiciels existants, sur les besoins exprimés par les familles et qui continuera à s'améliorer en proposant notamment un lexique et une organisation optimaux basés sur la recherche ;
- GazeMedia Player (mon projet de stage développé en angular), un site web qui assurera un accès aux sites de streaming musicaux préférés des utilisateurs, tels que Youtube, Spotify ou encore Deezer. Sur le long terme, ce projet sera utilisable par oculométrie.

Le stage consistera à renforcer l'équipe de développement actuellement constituée d'un ingénieur en informatique, de chercheurs et de bénévoles (par ailleurs informaticiens seniors dans leurs entreprises).

Il s'agira de développer de nouvelles fonctionnalités, de corriger les bugs découverts dans les outils.

L'étudiant(e) sera amené à utiliser les outils de développement mis à disposition de l'équipe (GitHub, groupes discord, ...).

L'ensemble de ces outils permet un passage facilité au télétravail total ou partiel le cas échéant.

2 - Contributions apportées durant le stage

2-1 GazePlay

2-1-1 Qu'est-ce que c'est ?

GazePlay est un logiciel libre et gratuit qui rassemble une soixantaine de mini-jeux jouables avec un oculomètre. Il est compatible avec tous les oculomètres pouvant contrôler le curseur de la souris et avec le Tobii EyeX et le Tobii 4C sur Windows et l'Eye Tribe Tracker sur Windows ou MacOs X.

Pour augmenter la motivation des joueurs, on peut modifier les images par défaut du jeu afin que les joueurs jouent avec des images de leur choix (image de leur famille ou de leurs thérapeutes par exemple). Une autre façon d'aider à motiver les joueurs est de leur donner une récompense (un grand emoji heureux et des applaudissements).

Les jeux et l'interface peuvent être affichés en français, anglais, allemand, et néerlandais et sont également partiellement traduits dans une vingtaine d'autres langues.

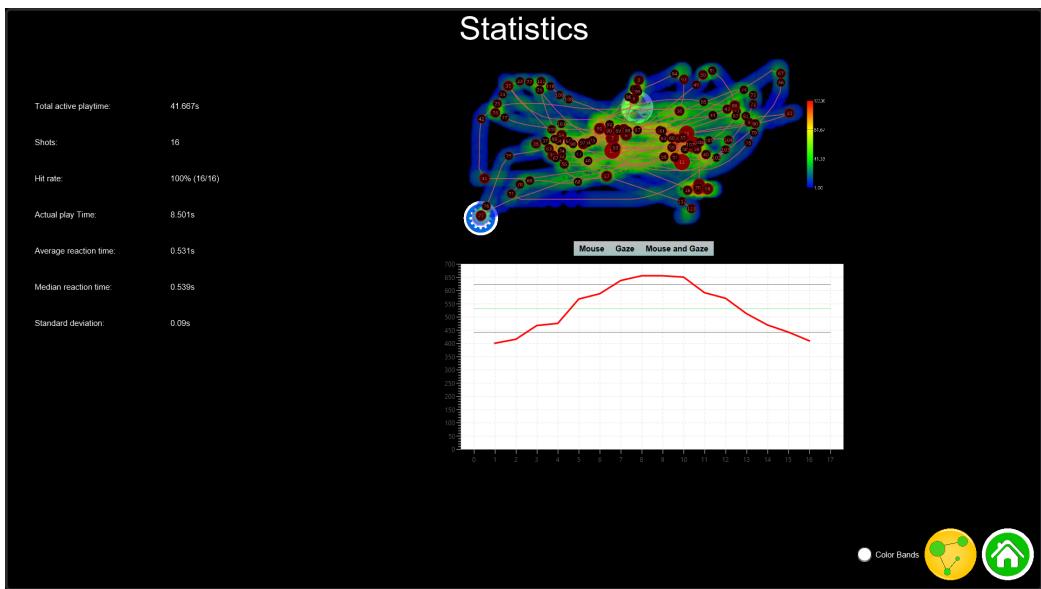
2-1-2 Existant

GazePlay rassemble plein de jeux , chacun proposant plusieurs variantes de difficultés, de tailles, etc. On compte ainsi 151 jeux et variantes. Pour chacun d'entre eux, nous visons à développer une ou plusieurs compétences chez les enfants.

On peut actuellement considérer cinq grands types de compétences pouvant être développés grâce à GazePlay :

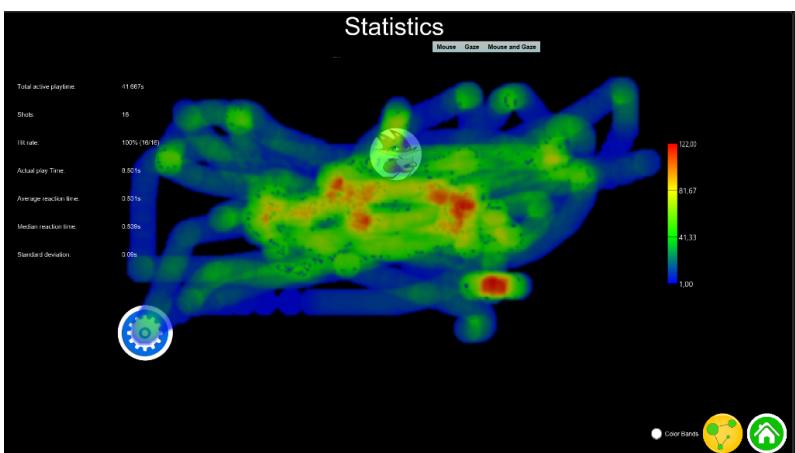
- les compétences d'action-réaction,
- les compétences de sélection,
- les compétences de mémorisation,
- les compétences de littératie,
- et les compétences de Logique & Mathématiques.

Lorsqu'un jeu est arrêté, plusieurs statistiques sont affichées et sauvegardées automatiquement.

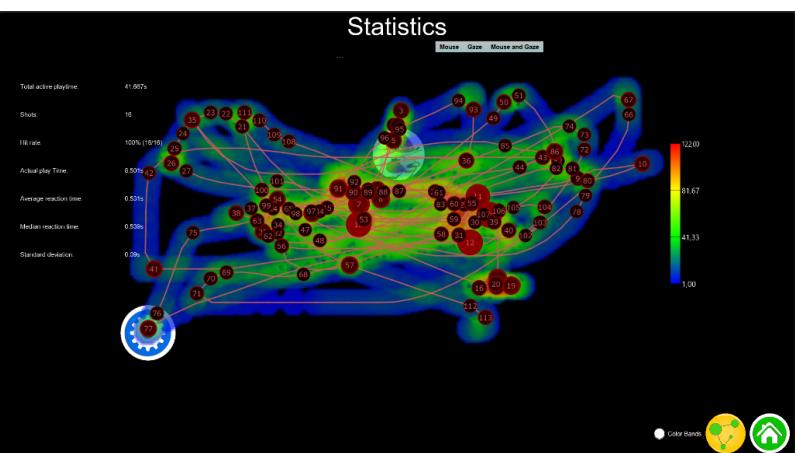


Capture des statistiques affichées à la suite du jeu Ninja

Elles présentent, suivant le jeu, la durée de temps de jeu actif, la durée de temps de jeu réel, la durée de réaction, l'écart type, le nombre de bonnes réponses... .



Capture de la carte de chaleur d'une partie de Ninja



Capture de la souris d'une partie de Ninja

Une carte de chaleur améliorée montre également les positions qui ont été regardées à l'écran et un tracé permet de connaître plus en détail le chemin qu'a emprunté le regard de l'utilisateur et les temps de fixation pour chacune des zones sur lesquelles son regard s'est arrêté. Ces zones sont appelées des zones d'intérêt et sont des données qui peuvent s'avérer très intéressantes pour le suivi de l'enfant, par exemple, pour observer le fait qu'il soit bien dans la capacité d'observer l'entièreté de l'écran ou si au contraire, certaines zones semblent plus difficiles d'accès.

L'enregistrement vidéo de l'écran et du regard est également disponible pour chaque partie. Grâce à ces statistiques, l'entourage parental et thérapeutique peut obtenir des informations objectives pour évaluer l'évolution et/ou la motivation des enfants.

2-1-3 Ma contribution

ProgressIndicator :

Lors de mon arrivée au LIG, j'ai commencé par l'implémentation de progress Indicator dans les jeux GazePlay qui n'en avait pas.

Pour rappel, le ProgressIndicator fait partie du package JavaFx. C'est un contrôle circulaire qui est utilisé pour indiquer les progrès, infinis ou finis (ce qui est le cas dans GazePlay). Il montre le degré d'achèvement d'une tâche.

Ils permettent à l'utilisateur, lorsqu'il utilise soit la souris soit un oculomètre, de pouvoir:

- comprendre qu'il va utiliser le bouton choisi lorsque celui-ci sera rempli;
- voir qu'il se trompe de cible;
- corriger son choix avant que celui-ci soit rempli;
- Et notamment comprendre que tels choix effectuent telles actions.



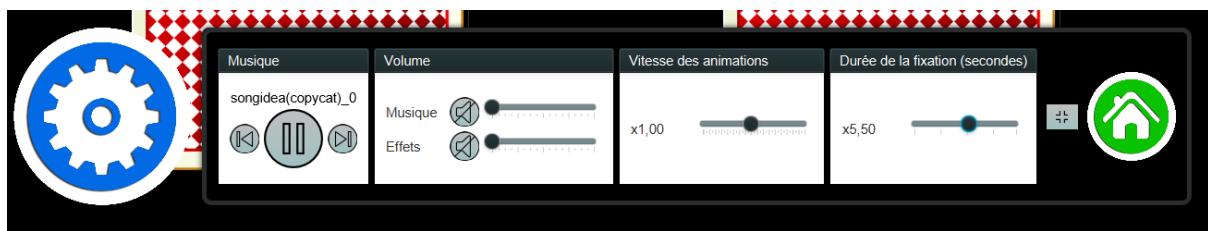
Exemple de l'utilisation du progress indicator dans le jeu Fabrique de gâteaux

Dans l'image, juste au-dessus, on remarque que l'utilisateur a choisi le 2ème bouton en partant de la gauche et que le ProgressIndicator de ce bouton a commencé à se remplir. Lorsque celui-ci sera rempli, alors l'action correspondante à ce bouton se lancera.

Durée de fixation :

Par la suite, j'ai créé et implémenté un système de slider bar (curseur de plage personnalisé), le dernier dans l'image ci-dessous, accessible depuis n'importe quel jeu via le système d'option (engrenage bleu).

Celui-ci permet à l'utilisateur de pouvoir régler la vitesse à laquelle les boutons se remplissent (via les ProgressIndicator).



Exemple d'une durée de fixation fixer à 5s 50

L'intérêt de ce système est de pouvoir adapter la vitesse de remplissage des boutons en fonction de la vitesse de réaction de l'utilisateur.

Si celui-ci a un temps de réaction élevé, il faut alors augmenter le temps de fixation.

A contrario, si il a un temps de réaction faible, il faut alors diminuer le temps de fixation.

2-2 GazeMedia Player

2-2-1 Qu'est-ce que c'est ?

GazeMedia Player est un site web libre et gratuit qui rassemble trois grosses plateformes de musique/vidéo (YouTube, Deezer et Spotify).

L'interface et les notifications peuvent être affichées en français ou en anglais.

Pour augmenter le confort des utilisateurs, ils peuvent aussi importer leurs propres musiques et/ou vidéos. De plus, un visualizer a été mis en place pour qu'ils ne s'ennuient pas pendant l'écoute de leur musique.

Plus tard, il sera compatible avec tous les oculomètres pouvant contrôler le curseur de la souris (le Tobii EyeX et le Tobii 4C sur Windows et l'Eye Tribe Tracker sur Windows ou MacOs X) et il y aura plus de langues traduites.

2-2-2 Existant

Au début, on envoyait directement l'utilisateur sur une playlist soit sur le site Spotify soit sur le site YouTube.

Le problème est que ces playlists peuvent à tout moment être supprimées, voir même n'être pas accessibles à tout public (limitation d'âge, devient une playlist privée, etc ...).

On a donc eu l'idée de créer notre propre application regroupant Spotify, YouTube et Deezer afin de permettre de créer nos propres playlists en allant chercher nos vidéos/musiques nous-mêmes.

De plus, cette application à part entière est pensée spécifiquement pour nos profils d'utilisateurs handicapés.

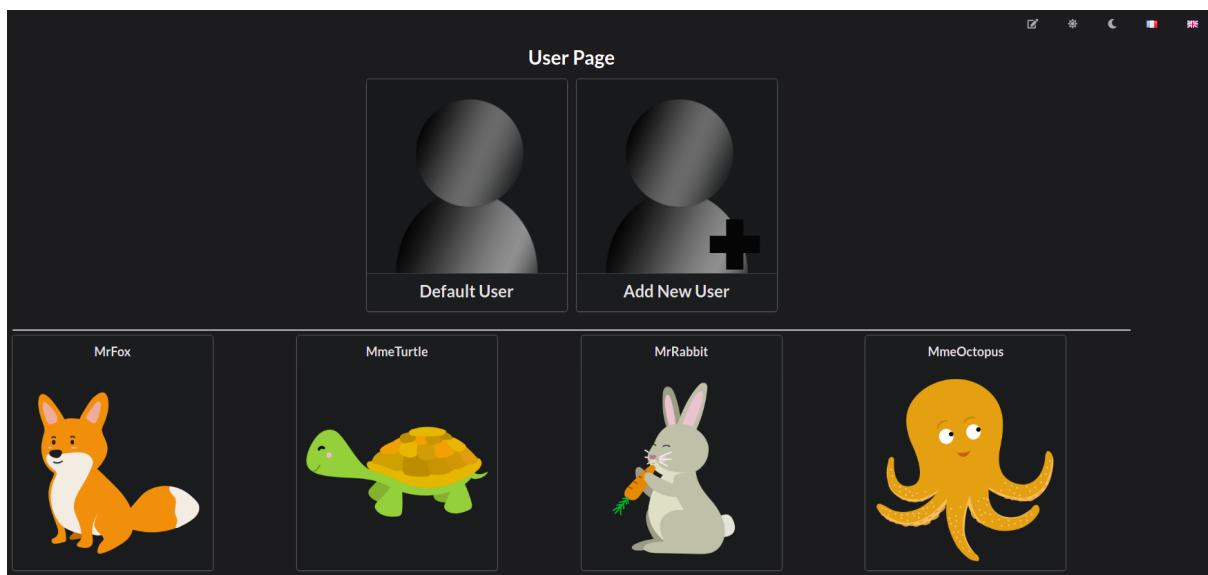
2-2-3 Ma contribution

Page Connexion :

Avant de pouvoir accéder à la playlist, chaque utilisateur commence par la page de connexion où il doit :

- Soit créer un compte utilisateur et se connecter avec;
- Soit se connecter avec son compte;
- Soit se connecter en tant qu'invité (via le compte invité).

Par défaut, la page de connexion est en mode thème sombre et la langue en Anglais. Mais à tout moment, via la barre d'options en haut à droite, on peut modifier ces éléments comme on veut.



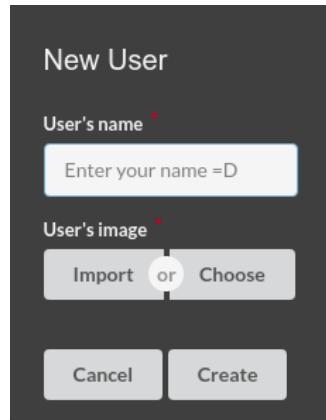
Exemple d'une page de connexion à GazeMedia Player avec 4 comptes

De plus, j'ai mis en place un système qui empêche l'utilisateur d'accéder aux autres pages de l'application s'il ne s'est pas connecté via son compte ou via le compte invité.

Parlons maintenant du système de connexion :

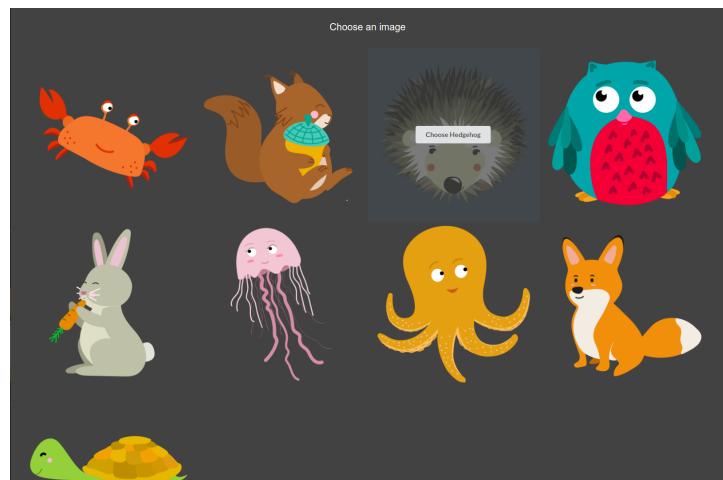
- Si l'utilisateur choisit de se connecter en tant qu'invité alors il accède à la playlist. Celle-ci est vierge, il peut utiliser les fonctionnalités liées à la playlist mais en revanche tout ce qu'il fait ne sera pas sauvegardé;
- Sinon si l'utilisateur choisit de se connecter en utilisant son compte, il accède lui aussi à la playlist. Mais il récupère sa playlist là où il l'avait laissée la dernière fois, de plus tout ce qu'il fait sera sauvegardé;

- Sinon si l'utilisateur choisit de se créer un compte, une fenêtre s'ouvre et lui demande son nom et une image. Ensuite, il peut se connecter en utilisant son compte fraîchement créé pour accéder à la playlist. Celle-ci sera vierge mais tout ce qu'il effectuera sur cette playlist sera sauvegardé.



Menu de création de compte

Lorsque que la fenêtre de création de compte s'ouvre, elle demande à l'utilisateur de rentrer son nom et soit d'importer une image soit de sélectionner une des images que l'on propose.



Liste des images que l'on propose à l'utilisateur

Une fois que l'utilisateur a défini son image de profil, on aura une prévisualisation de celle-ci dans le menu de création de compte.



Prévisualisation de l'image écureuil que l'utilisateur a choisi pour son compte

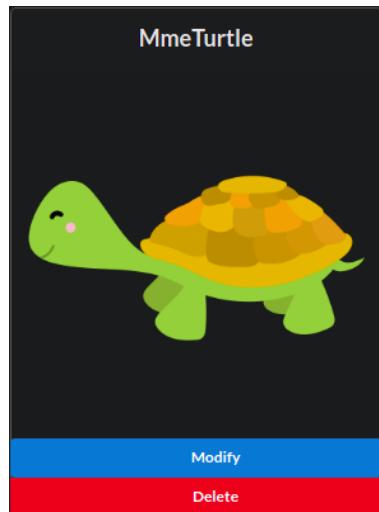
Si jamais, l'utilisateur essaye de créer un compte sans nom ou sans image, un message d'erreur apparaît dans la fenêtre de création lui expliquant ce qui ne va pas !

The image contains two side-by-side screenshots of a 'New User' creation dialog box. Both screenshots show the same fields: 'User's name' (with placeholder 'Enter your name =D') and 'User's image' (with 'Import' and 'Choose' buttons). In the left screenshot, both fields are empty, and red error messages 'Name is empty!' and 'Image is empty!' are displayed below their respective fields. In the right screenshot, the 'User's name' field contains the value 'Jordan', while the 'User's image' field remains empty, still displaying the 'Image is empty!' error message.

Erreurs lors de la création d'un compte

Il se peut, qu'après plusieurs jours, l'utilisateur remarque qu'il a mal écrit son nom ou que finalement l'image ne lui plaît pas.

Alors au lieu de supprimer son compte (perdre toutes ses données au passage) et d'en créer un nouveau, il peut utiliser le bouton bleu "modifier" afin de changer soit son nom soit son image soit les 2 à la fois.



Exemple d'un compte que l'on peut modifier ou supprimer

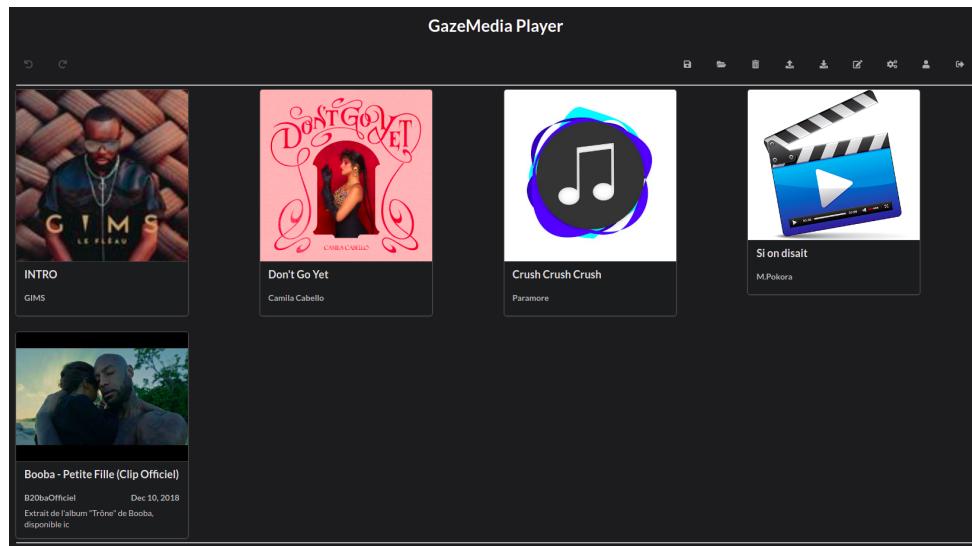
Etant donné que la base de données est en local (utilisation de IndexedDB), l'utilisateur peut supprimer n'importe quel compte. Évidemment un message d'alerte apparaît avant la suppression afin de lui demander s'il est sûr de cette action irréversible, de plus le bouton pour valider la suppression est cliquable seulement après 5s d'attente.

Page Playlist :

J'ai créé et mis en place la page web "Playlist" qui permet :

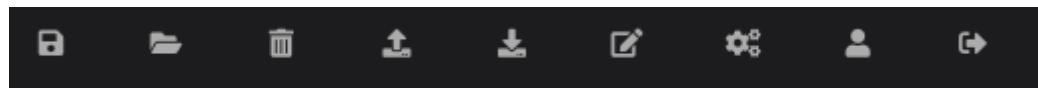
- de voir sa playlist;
- de sauvegarder sa playlist;
- de supprimer sa playlist;
- de charger une playlist sauvegardée;
- d'éditer sa playlist;
- d'ordonner sa playlist;
- d'importer des fichiers dans sa playlist;
- d'exporter sa playlist;
- de configurer sa playlist;
- de regarder ses vidéos;
- d'écouter ses musiques;
- de se déplacer facilement à l'aide d'une barre de navigation;
- de se connecter facilement à Deezer et/ou Spotify depuis la page web playlist;
- d'avoir des notifications;
- d'avoir des messages d'alertes (activable ou non) lors d'une suppression;
- de se déconnecter de son compte.

Après s'être connecté, l'utilisateur arrive sur la page de sa playlist qu'il retrouve dans l'état où il l'avait laissé lors de sa dernière session. Si le compte est tout nouveau alors la playlist est par défaut vide.



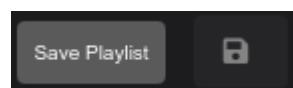
Exemple d'une page d'accueil de la page web "Playlist"

Grâce à une barre de menu horizontal, juste au-dessus de la playlist, l'utilisateur a accès à pleins de fonctionnalités.



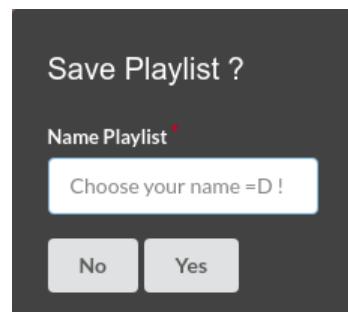
Barre de menu de la playlist

Chaque bouton du menu possède une info-bulle, qui donne le nom du bouton, lorsque l'on passe la souris dessus.



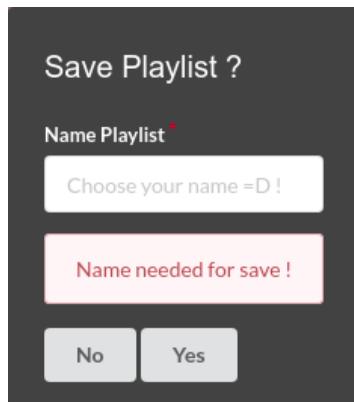
Commençons par le premier bouton, sauvegarder sa playlist :

Lorsque l'on clique sur ce bouton, une fenêtre apparaît demandant à l'utilisateur un nom pour la playlist actuelle que l'on souhaite sauvegarder.

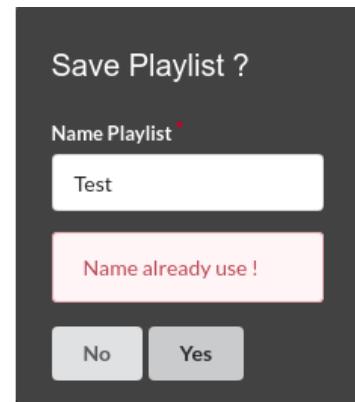


Fenêtre de sauvegarde

Si jamais l'utilisateur essaie de sauvegarder sa playlist sans mettre de nom ou en utilisant un nom déjà pris alors un message d'erreur apparaît lui expliquant son erreur;



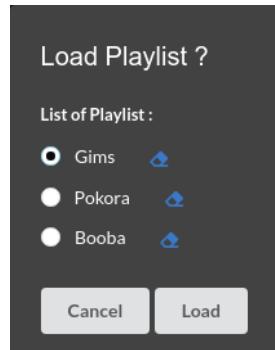
Exemple erreur : nom vide



Exemple erreur : nom déjà utilisé

Continuons avec le deuxième bouton, charger une playlist :

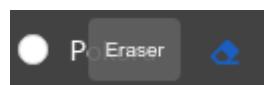
Lorsque l'on clique sur ce bouton, une fenêtre apparaît nous permettant de charger des playlists que l'on avait préalablement sauvegardées.



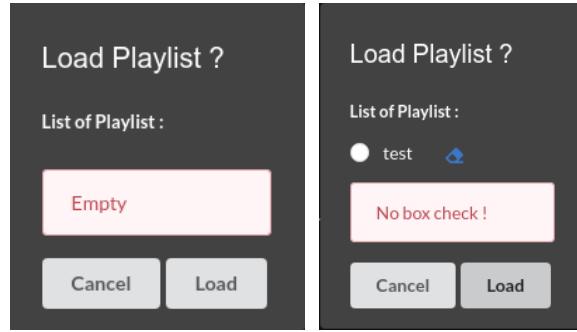
Fenêtre permettant de charger une playlist

Un fois la playlist choisie, au moment du chargement, si jamais on a déjà une playlist active alors un message d'alerte apparaît demandant à l'utilisateur si il est sûr de vouloir écraser la playlist actuelle par la playlist qu'il souhaite charger.

A tout moment l'utilisateur peut choisir de supprimer une playlist sauvegardée en cliquant sur le bouton en forme de gomme bleu :

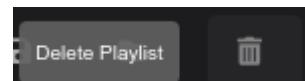


Si l'utilisateur n'avait pas de playlist sauvegardée ou qu'il essaie de charger une playlist sans en avoir sélectionnée une, alors un message apparaît expliquant l'erreur.



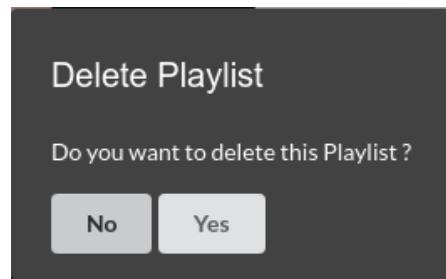
Erreur liste vide

Erreur pas de sélection

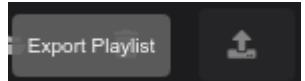


Continuons avec le troisième bouton, supprimer la playlist :

Lorsque l'utilisateur clique dessus, une fenêtre apparaît lui demandant si il veut effectivement supprimer sa playlist actuelle.

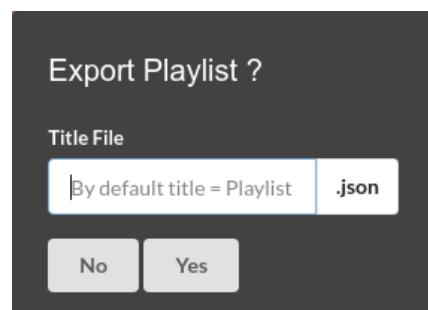


Fenêtre de suppression de playlist



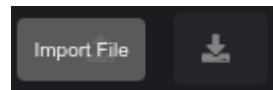
Continuons avec le quatrième bouton, exporter sa playlist :

Lorsque l'utilisateur clique dessus, une fenêtre apparaît lui demandant alors le nom qu'il veut donner à son fichier qu'il va recevoir (le fichier récupéré sera au format JSON et contiendra sa playlist actuelle).



Fenêtre d'export de la playlist actuelle

Si jamais l'utilisateur ne rentre pas de nom pour le fichier et tente de l'exporter quand même, alors par défaut le nom du fichier sera "Playlist.json".

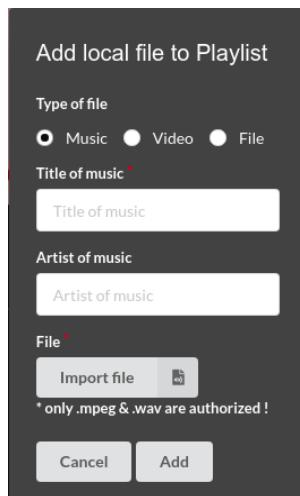


Continuons avec le cinquième bouton, importer un fichier :

Lorsque l'utilisateur clique dessus, une fenêtre apparaît lui demandant quel type de fichier il veut importer (musique, vidéo ou playlist).

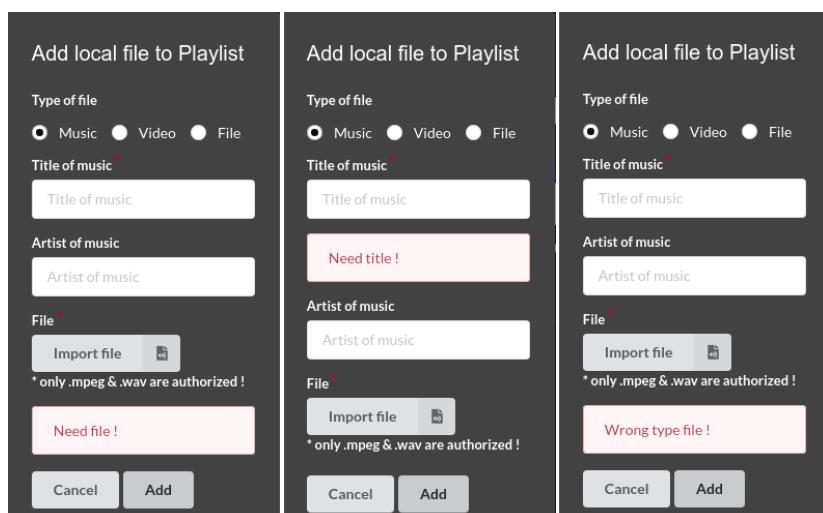
Si l'utilisateur veut importer une musique ou une vidéo, alors il aura une fenêtre lui demandant le nom de la musique/vidéo, le nom de l'artiste et le fichier à importer.

Pour les fichiers musicaux à importer, ils devront être soit de type .mpeg ou .wav sinon ils ne seront pas autorisés (pour les fichiers vidéos, c'est soit .mp4 soit .webm).



Exemple fenêtre import d'une musique

Si l'utilisateur oublie de rentrer le nom de la musique, de donner le fichier qu'il veut importer ou encore qu'il donne un mauvais fichier, alors des messages d'erreurs apparaîtront lui expliquant l'erreur qu'il a commise.



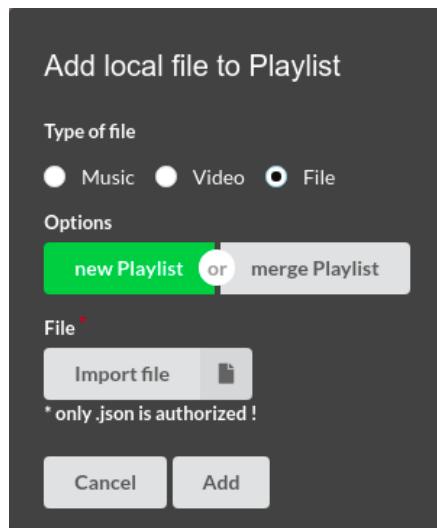
Erreur pas de fichier / Erreur pas de titre / Erreur mauvais fichier

Pour voir la fenêtre d'import d'une vidéo, qui est exactement la même que celle de l'import d'une musique (visuel comme erreurs), rendez-vous dans les annexes page 40.

Mais si l'utilisateur veut importer une playlist, alors il aura une autre fenêtre qui lui demandera la playlist au format JSON et l'option qu'il veut choisir.

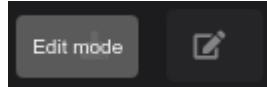
- Première option, "nouvelle playlist", celle-ci remplace sa playlist actuelle par la nouvelle playlist qu'il souhaite importer.
- Deuxième option, "fusionner playlist", celle-ci fusionne sa playlist actuelle avec la playlist qu'il veut importer en supprimant les doublons.

De plus, le fichier au format json qu'il importe doit correspondre à mon schéma de validation pour json (voir annexes page 40), sinon son fichier sera refusé et un message d'erreur apparaîtra comme pour l'importation d'une musique.



Fenêtre ajout d'une playlist avec l'option "nouvelle playlist"

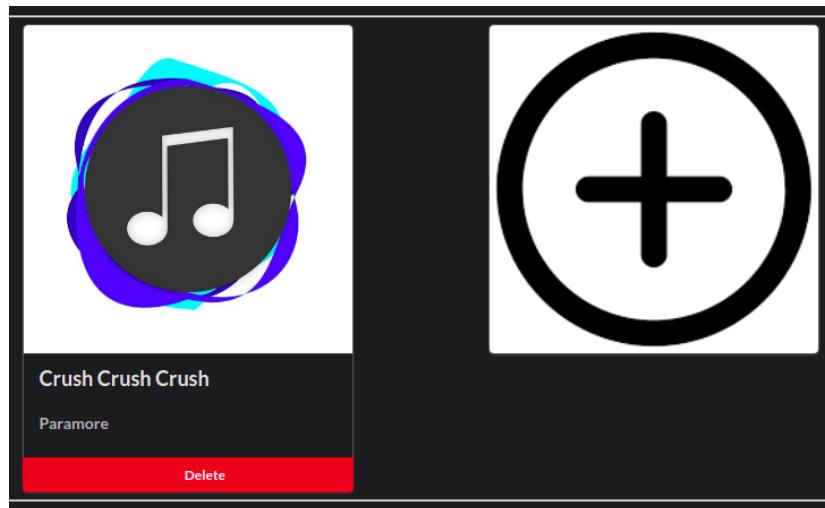
Une fois que l'utilisateur a tout renseigné et clique sur le bouton "ajouter" alors la/les musiques, vidéos et/ou playlists qu'il a choisi seront disponibles à l'écoute et au visionnage.



Continuons avec le sixième bouton, mode édition :

En cliquant sur ce bouton, on passe du mode normal au mode édition et vice-versa.

En mode édition, un bouton rouge "supprimer de la playlist" apparaît sous les musiques et vidéos. Ce bouton permet de supprimer de la playlist la musique/vidéo associée au bouton. De plus, pendant le mode édition, un bouton "+" est affiché et cliquable.

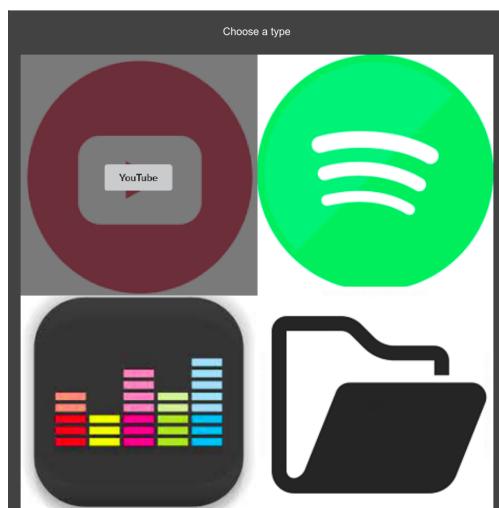


Exemple du bouton rouge "supprimer" sous une musique + affichage du bouton "+" en mode édition

Si l'on clique sur le bouton "+", une fenêtre apparaît permettant à l'utilisateur d'aller :

- soit sur ma page web YouTube et d'ajouter une vidéo youtube à sa playlist;
- soit sur ma page web Spotify et d'ajouter une musique spotify à sa playlist;
- soit sur ma page web Deezer et d'ajouter une musique deezer à sa playlist;

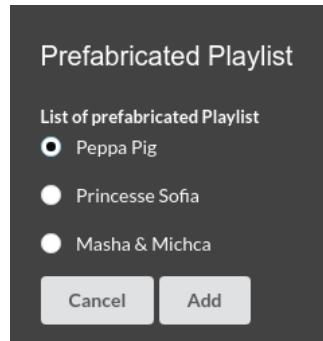
Si l'on clique sur le bouton "+" et que sa playlist est vide alors un quatrième choix lui est proposé, celui de charger une playlist préfabriquée.



Fenêtre permettant de choisir une plateforme

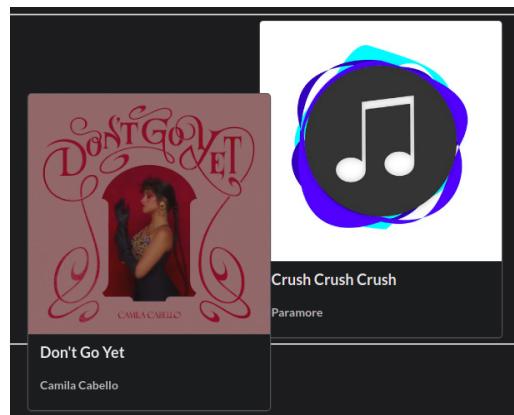
Lorsque l'utilisateur clique sur YouTube (en haut à gauche), Spotify (en haut à droite) ou Deezer (en bas à gauche), il est redirigé vers la page correspondante.

Mais si il clique sur le dossier des playlist préfabriquées (en bas à droite), alors une autre fenêtre s'ouvre lui demandant quelle playlist préfabriquée il veut ajouter à sa playlist.



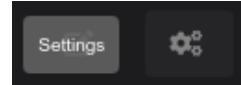
Fenêtre des playlist préfabriquées

De plus, en mode édition uniquement, l'utilisateur peut effectuer du Drag&Drop afin d'agencer sa playlist comme il le souhaite.

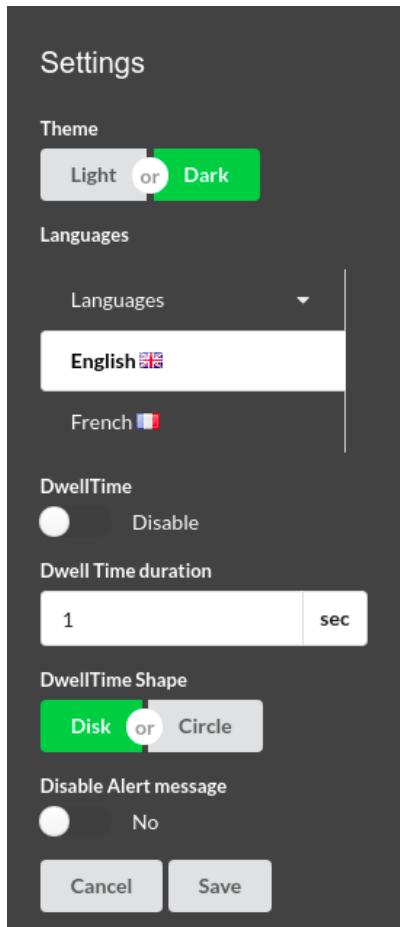


Drag&Drop de "Don't Go Yet" à la place de "Crush Crush Crush"

Sur l'image, pendant le Drag&Drop, on peut remarquer que l'élément que l'on déplace devient plus foncé afin d'indiquer à l'utilisateur que celui-ci est bien en mouvement.



Continuons avec le septième bouton, le panneau des paramètres :
En cliquant sur ce bouton, l'utilisateur ouvre une fenêtre permettant de modifier le site web.

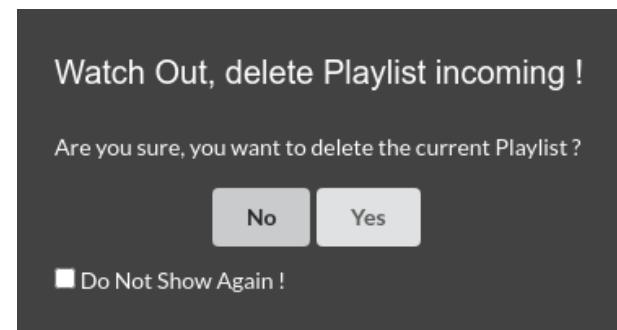


Fenêtre des paramètres

Dans cette fenêtre on peut :

- Modifier le thème du site, clair ou sombre;
- Choisir la langue à utiliser, Francais ou Anglais;
- Activer ou non le Dwell Time;
- Choisir la durée du Dwell Time;
- Choisir la forme du Dwell Time, un disque ou un cercle qui se remplit;
- Activer ou non les messages d'alerte.

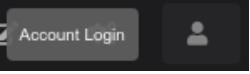
Chaque compte a ses propres paramètres et ceux-ci sont sauvegardés une fois que l'utilisateur clique sur le bouton "sauvegarder" en bas de la fenêtre.



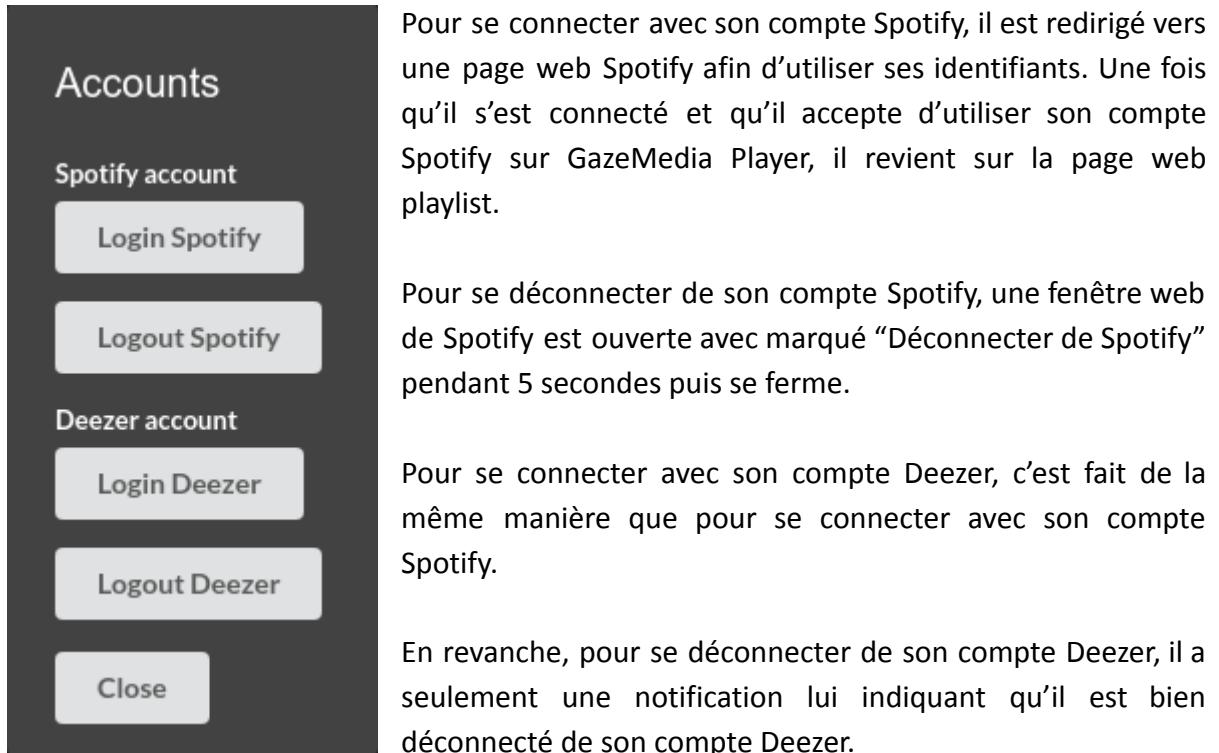
Exemple d'un message d'alerte -> la suppression de la playlist

On peut voir dans l'image ci-dessus que l'on peut désactiver les messages d'alerte en cochant la case en bas à gauche "Ne plus afficher de nouveau" avant de choisir une réponse (oui ou non dans l'exemple).

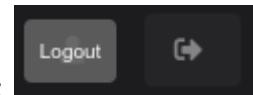
Le fait de désactiver les messages d'alerte rend toutes les actions instantanées et si jamais l'utilisateur veut les réactiver, il lui suffit juste de les remettre dans le panneau des paramètres.

Continuons avec le huitième bouton, état des comptes : 

Lorsque l'utilisateur clique sur ce bouton, une fenêtre apparaît lui permettant facilement de se connecter/déconnecter de ses comptes Spotify et/ou Deezer.



Fenêtre des comptes Spotify/Deezer



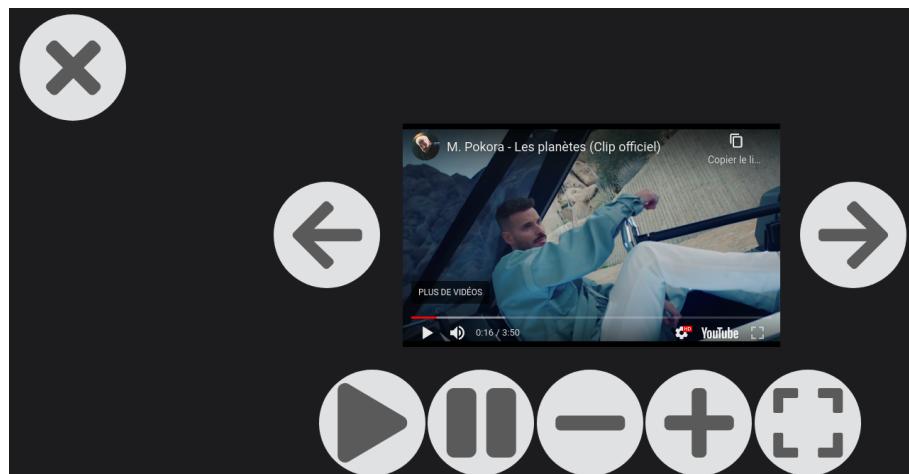
Finissons avec le neuvième et dernier bouton du menu, déconnexion :

Lorsque l'utilisateur clique sur ce bouton, il est ramené sur la page de connexion.

De plus, au moment de la déconnexion on vérifie que les comptes Spotify et Deezer de l'utilisateur soient eux aussi déconnectés, sinon on le fait.

C'est pour éviter que si un autre utilisateur se connecte avec un autre compte, il ne puisse pas utiliser les comptes Spotify et Deezer d'une autre personne.

Maintenant, regardons comment écouter/regarder une musique/vidéo depuis notre playlist. Lorsque l'on clique sur un élément de notre playlist, on affiche en dessous de celle-ci soit un lecteur audio soit un lecteur vidéo en fonction du type de l'élément sélectionné.



Exemple de l'affichage d'une vidéo YouTube

De plus, on affiche des boutons permettant à l'utilisateur de :

- lancer la musique ou la vidéo;
- mettre en pause la musique ou la vidéo
- augmenter ou diminuer le volume de la musique/vidéo;
- passer à la musique ou la vidéo suivante;
- retourner à la musique ou la vidéo précédente;
- fermer le lecteur audio ou vidéo;
- passer en mode plein écran (uniquement pour les vidéos);
- sortir du mode plein écran.

Pour voir à quoi ressemble l'affichage des autres lecteurs, rendez-vous dans les annexes :

- Lecteur Spotify -> page 41;
- Lecteur Deezer -> page 41;
- Lecteur musique importée -> page 42 ;
- Lecteur vidéo importée -> page 42.

La particularité du mode plein écran pour une vidéo est que l'on garde la possibilité d'utiliser les boutons (les mêmes que ceux montrés dans l'image ci-dessus).



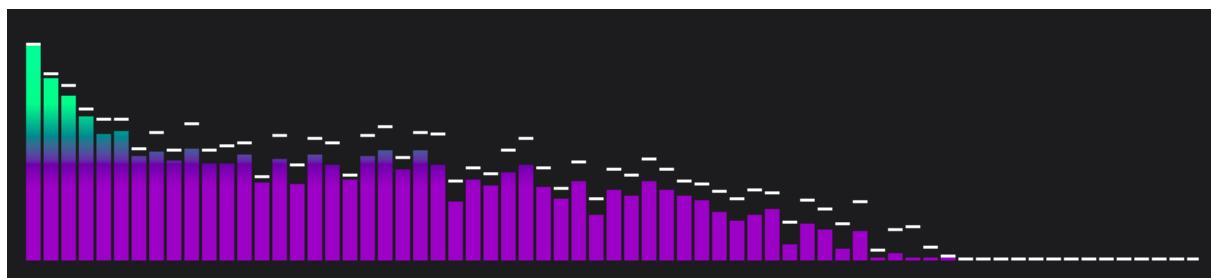
Mode plein écran d'une vidéo

En revanche, pendant le mode plein écran, les boutons s'effacent petit à petit jusqu'à devenir invisibles. Mais ils redeviennent visibles lorsque l'on passe la souris ou le regard dessus, permettant ainsi de pouvoir se servir des boutons tout en regardant la vidéo dans les meilleures conditions.

De plus, le lecteur audio a lui aussi une particularité : un visualizer.

Celui-ci permet de générer des images animées basées sur un morceau de musique, en l'occurrence la musique choisie par l'utilisateur.

Les images sont générées et rendues en temps réel et de manière synchronisée avec la musique quand elle est jouée.



Exemple du visualizer en cours de fonctionnement

Les couleurs choisies sont générées aléatoirement, du coup il est très rare de tomber 2 fois sur un visualizer ayant les mêmes couleurs.



Playlist



Up



Down



Music

Parlons maintenant de ma deuxième barre de menu qui est verticale cette fois-ci.

Dans un premier temps, cette barre n'est affichée que si la hauteur de la page web playlist est supérieure à la hauteur de l'écran de l'utilisateur.

Ensuite, cette barre comporte 4 boutons:

- un bouton "Playlist" avec l'icône d'une maison;
- un bouton "Monter" avec l'icône d'une flèche vers le haut;
- un bouton "Descendre" avec l'icône d'une flèche vers le bas;
- un bouton qui est soit "Musique" avec l'icône d'une note de musique soit un bouton "Vidéo" avec l'icône de film.

Le premier bouton permet, lorsque l'utilisateur clique dessus, de revenir tout en haut de la page web playlist à l'emplacement de la playlist.

Si il est déjà à l'emplacement de la playlist et qu'il clique dessus, il ne se passera rien.

Le deuxième bouton permet de remonter de 10 pixels par 10 pixels par rapport à là où l'on se trouve.

Si il est déjà tout en haut de la page web et qu'il clique sur ce bouton, alors il ne se passera rien.

Le troisième bouton est comme le deuxième mais cette fois-ci pour descendre.

Si il est déjà tout en bas de la page web et qu'il clique sur ce bouton, alors il ne se passera rien.

Et enfin le quatrième bouton, celui-ci change en fonction du lecteur actuellement en cours :

- Bouton musique si c'est une musique Spotify, Deezer ou importée.
- Bouton vidéo si c'est une vidéo Youtube ou importée.

De plus, ce bouton apparaît seulement quand une vidéo ou musique est en cours.

Lorsqu'il clique dessus, ce bouton l'emmène directement sur le lecteur audio ou vidéo en cours.

Pour finir, cette barre de menu le suit quand il scroll sur la page web afin d'y avoir toujours accès sans être à chaque fois obligé de remonter.

Page Youtube :

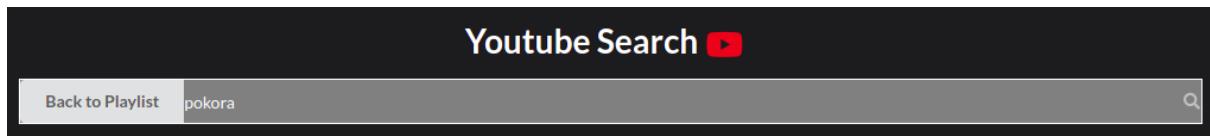
J'ai créé et mis en place la page web "YouTube" qui récupère une liste de vidéos de YouTube en transmettant une requête de recherche et d'autres paramètres à l'API de recherche YouTube.

Pour créer la page j'ai utilisé , en plus d'Angular CLI et Typescript, RxJS qui est une bibliothèque pour composer des programmes asynchrones et basés sur des événements en utilisant des séquences observables.

Pour faire tout ça, nous avons 3 composants principaux et un service :

- 1 composant pour l'élément d'entrée (l'input);
- 1 composant pour le rendu de la liste des vidéos;
- 1 composant parent qui rend les 2 à la fois;
- et enfin 1 service , c'est là que nous mettrons en œuvre toutes les fonctionnalités pertinentes qui nous permettront de communiquer avec l'API de recherche YouTube et de gérer les réponses de cette API.

Nous avons donc un élément d'entrée (un input) qui permet à l'utilisateur de saisir une requête de recherche. De plus, à gauche de cette input, nous avons un bouton permettant à l'utilisateur de retourner sur la page de la Playlist.



Recherche des vidéos de Pokora sur l'API YouTube

Cette valeur sera envoyée à notre service qui l'utilisera pour construire une URL et communiquer avec l'API de recherche de YouTube. Si l'appel à l'API réussit, il renverra une liste de vidéos liée à la valeur envoyée que nous pourrons ensuite afficher sur la page.

Mais avant d'obtenir ça, on doit créer un fichier Interface qui nous permet de définir la syntaxe à laquelle toute entité doit adhérer. Dans notre cas, j'ai défini certaines propriétés que chaque objet vidéo récupéré à partir de l'API Youtube doit contenir.

Les propriétés que je veux sont :

- L'id de la vidéo;
- L'url de la vidéo;
- L'id de la chaîne;
- L'url de la chaîne;
- Le titre de la chaîne;
- Le titre de la vidéo;
- La date de publication de la vidéo;
- La description de la vidéo;
- Et la vignette de la vidéo.

Avec toutes ces informations, cela me permet d'avoir des identifiants uniques afin de pouvoir être sûr qu'une vidéo est déjà dans ma playlist ou non.

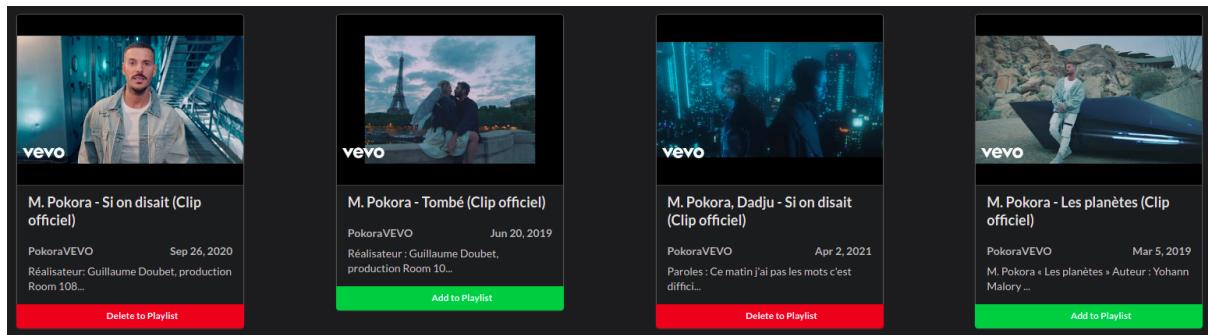
Ça me permet aussi de pouvoir créer mes cartes de vidéo en y ajoutant toutes les informations que je récupère.



Exemple d'une carte vidéo

Ensuite lorsque l'on obtient notre liste de vidéos, on parcourt notre liste et on regarde chaque vidéo pour voir si elle n'est pas déjà dans notre playlist.

Si elle n'y est pas alors le bouton "ajouter à la playlist" apparaît (bouton vert) sinon le bouton "supprimer de la playlist" apparaît (bouton rouge).



Liste des vidéos en lien avec Pokora

A chaque action avec les boutons “ajouter” et “supprimer”, une notification apparaît en bas à gauche de l’écran, expliquant à l’utilisateur ce qu’il vient de faire.



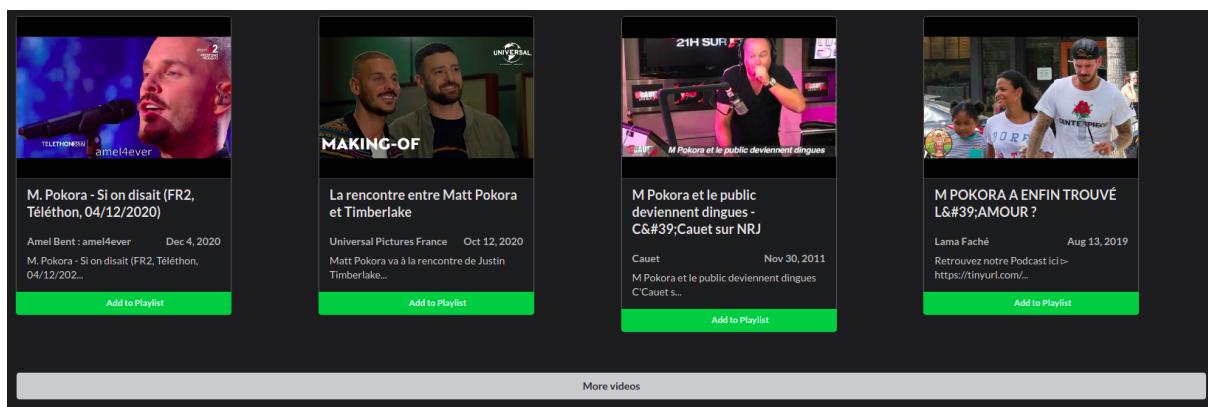
Notification ajout de la vidéo

Notification suppression de la vidéo

Pour éviter d’avoir trop de vidéos d’un coup, j’ai décidé de limiter l’affichage des vidéos à 12 (4 vidéos par ligne).

Puis j’ai mis en place un bouton “plus de vidéos” qui , lorsque l’on clique dessus, permet d’afficher 8 vidéos de plus et ainsi de suite jusqu’à afficher la totalité de la liste.

Au maximum une liste peut contenir 50 vidéos, cette limite est imposée par l’API YouTube elle-même.



Bouton “plus de vidéos” permettant d'afficher 8 vidéos en plus

Page Spotify :

J'ai créé et mis en place la page web "Spotify" qui utilise l'API Spotify pour :

- faire des recherches d'artistes, d'albums ou de morceaux de musiques;
- visualiser les dernières sorties trouvées sur la plateforme;
- et pouvoir ajouter une ou des musiques à la playlist.

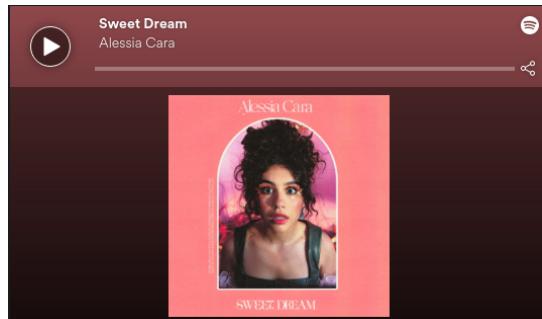
L'API Spotify fournit un ensemble de points de terminaison, chacun avec son propre chemin d'accès unique. Toutes les requêtes à l'API nécessitent une authentification. Ceci est réalisé en envoyant un jeton d'accès (token) valide dans l'en-tête de la demande.

Afin de centraliser les requêtes vers Spotify, j'ai créé un service global dans lequel nous avons défini l'url commune et l'en-tête pour spécifier le jeton d'accès.

Ce service sera appelé par les autres services des modules afin d'exécuter chaque requête spécifique.

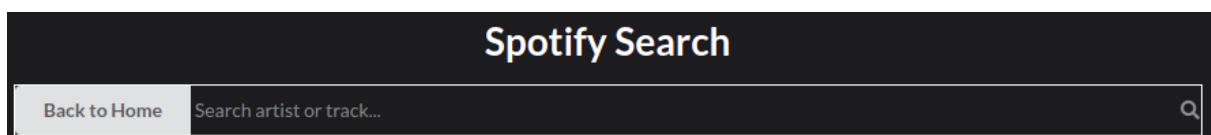
Grâce aux requêtes HTTP, j'ai été en mesure d'obtenir les informations spécifiques que je voulais et donc de travailler sur la gestion des données asynchrones.

Pour écouter les musiques de Spotify, j'ai utilisé leur widget qui fournit un lecteur audio intégrable facilement dans mon projet Web.



*Widget Spotify pour la musique
Sweet Dream de l'artiste Alessia Cara*

Pour rechercher un artiste, un album ou une musique chez Spotify, nous avons un élément d'entrée (un input) qui permet à l'utilisateur de saisir une requête de recherche.



Barre de recherche de mon application qui utilise l'API Spotify

Ensuite lorsque j'affiche un artiste, un album ou une musique que l'utilisateur recherche, il se peut que l'image associée soit endommagée ou n'a pas d'image du tout.

J'ai donc implémenter un Pipe, ci celui-ci ne reçoit pas d'image alors je met une image par défaut sinon je met l'image correspondante à l'artiste, album ou musique.

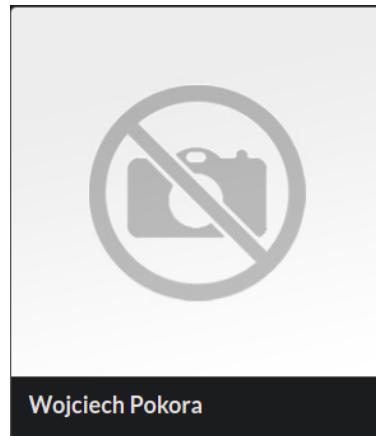
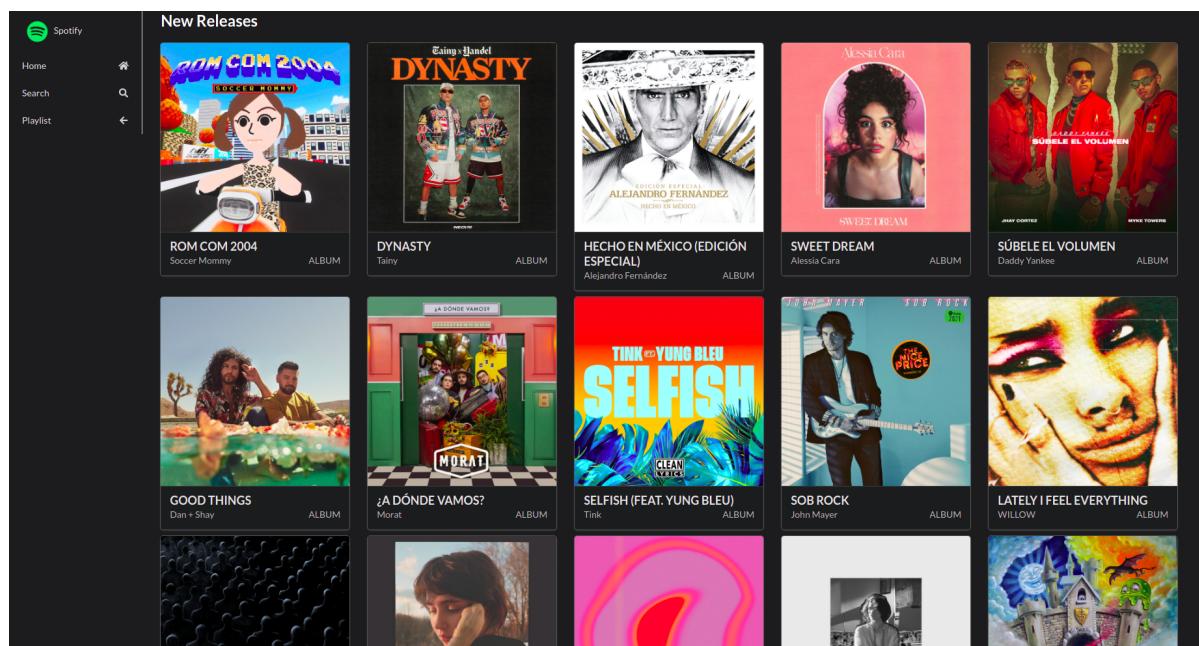


Image par défaut

Les audios seront affichés et lus en tant que widget Spotify. Pour les écouter, j'ai utilisé l'uri qui est une donnée de l'API Spotify qui peut être récupérée lors d'une demande de musiques.

La page d'accueil de cette application contient :

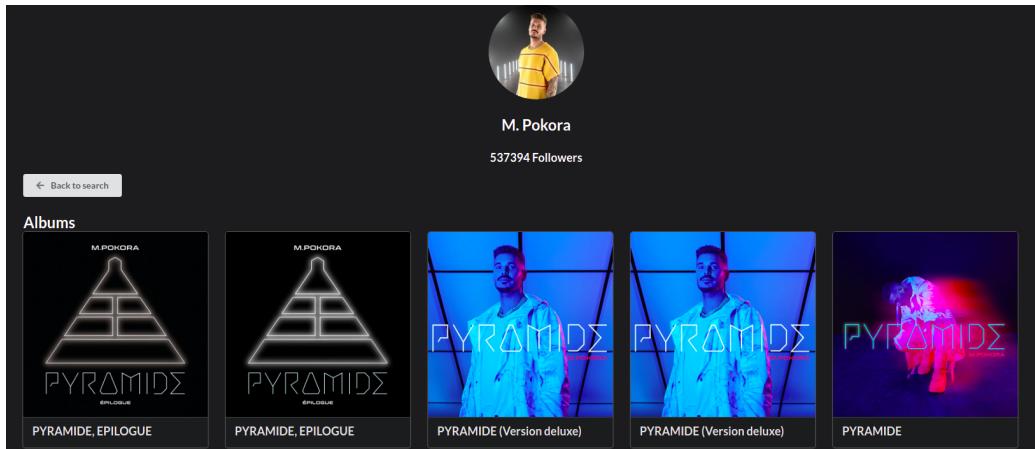
- un body avec une liste des tous les derniers albums récemment sortis;
- une sidebar permettant de soit retourner à l'accueil, soit aller sur la page de recherche de musique/artistes/albums ou soit de retourner sur la page de la playlist.



Page d'accueil de mon application Spotify

Lorsque que l'on clique sur un artiste, on arrive sur un page contenant :

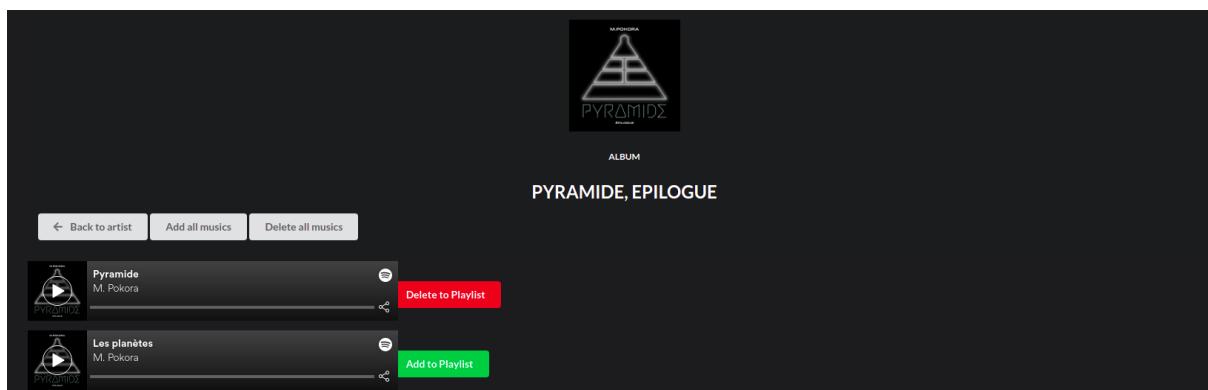
- Une image de l'artiste;
- Son Nom et Prénom;
- Son nombre de followers
- Une liste de tous ses albums;
- Un bouton permettant de retourner sur la barre de recherche.



Exemple de la page de l'artiste M.Pokora

Puis si on clique sur l'un de ses albums, on arrive sur une page contenant :

- L'image de l'album choisie;
- Le nom de l'album;
- Des boutons permettant de retourner sur la page de l'artiste, ajouter toutes les musiques dans la playlist ou de toutes les enlever;
- La liste de toutes les musiques de l'album choisi.



Exemple de 2 musiques de l'album Pyramide, Epilogue de Pokora

De plus, nous pouvons écouter 30 secondes de chaque musique grâce au widget Spotify. Si la musique nous plaît, on peut alors l'ajouter à notre playlist ou là elle sera écoutable en entier.

Page Deezer :

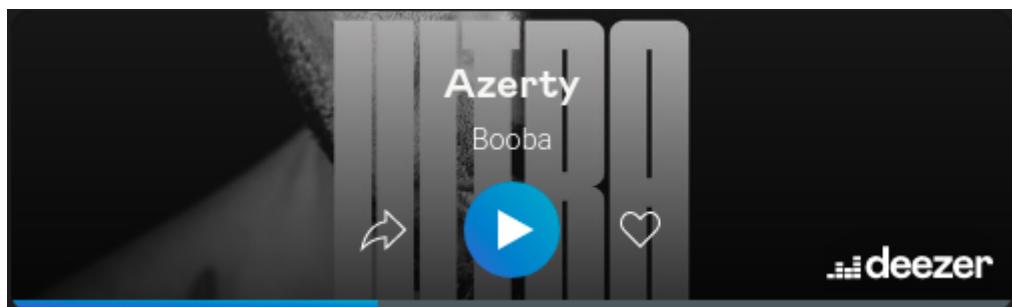
Comme pour Spotify, j'ai créé et mis en place la page Web Deezer qui utilise l'API Deezer pour :

- faire des recherches d'artistes, d'albums ou de morceaux de musiques;
- et pouvoir ajouter une ou des musiques à la playlist.

L'API Deezer fournit un accès illimité et sans authentification pour l'ensemble de ses services, nous permettant de créer une application web qui parcourt son catalogue musical.

En revanche, il faut être connecté pour pouvoir écouter les musiques en entier sinon nous n'avons que 30 secondes d'écoute.

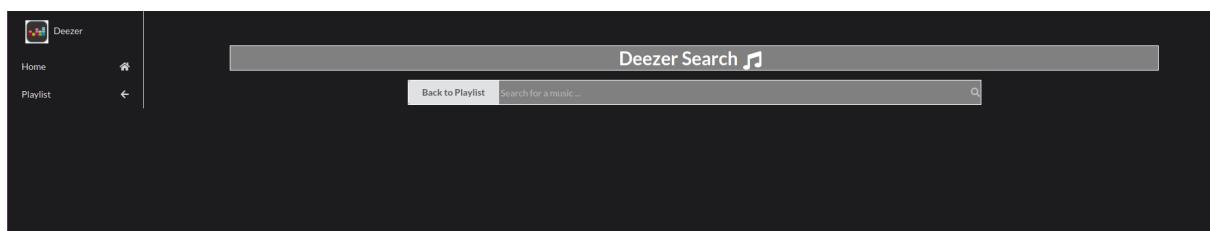
Pour écouter les musiques de Deezer, j'ai utilisé leur widget qui fournit un lecteur audio intégrable facilement dans mon projet Web.



Widget Deezer pour la musique Azerty de l'artiste Booba

La page d'accueil de cette application contient :

- un body avec une barre de recherche;
- une sidebar permettant de soit retourner à l'accueil soit de retourner sur la page de la playlist.



Page d'accueil de mon application Deezer

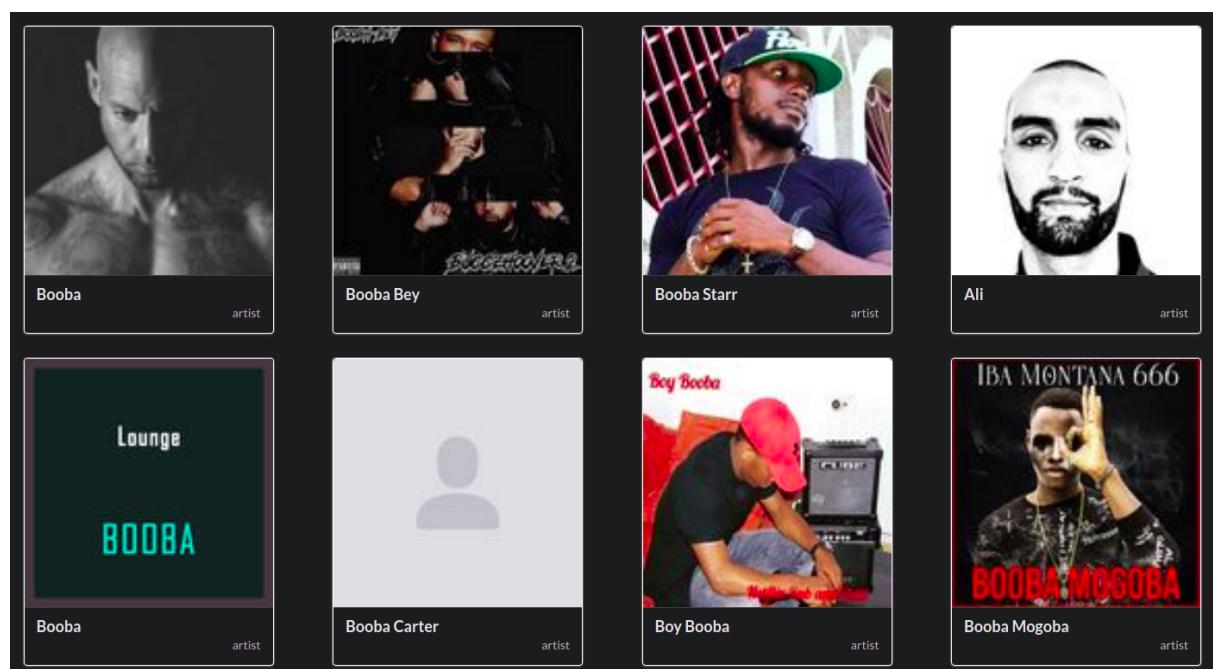
Nous avons donc un élément d'entrée (un input) qui permet à l'utilisateur de saisir une requête de recherche.

De plus, à gauche de cette input, nous avons un bouton permettant à l'utilisateur de retourner sur la page de la Playlist.



Barre de recherche de mon application qui utilise l'API Deezer

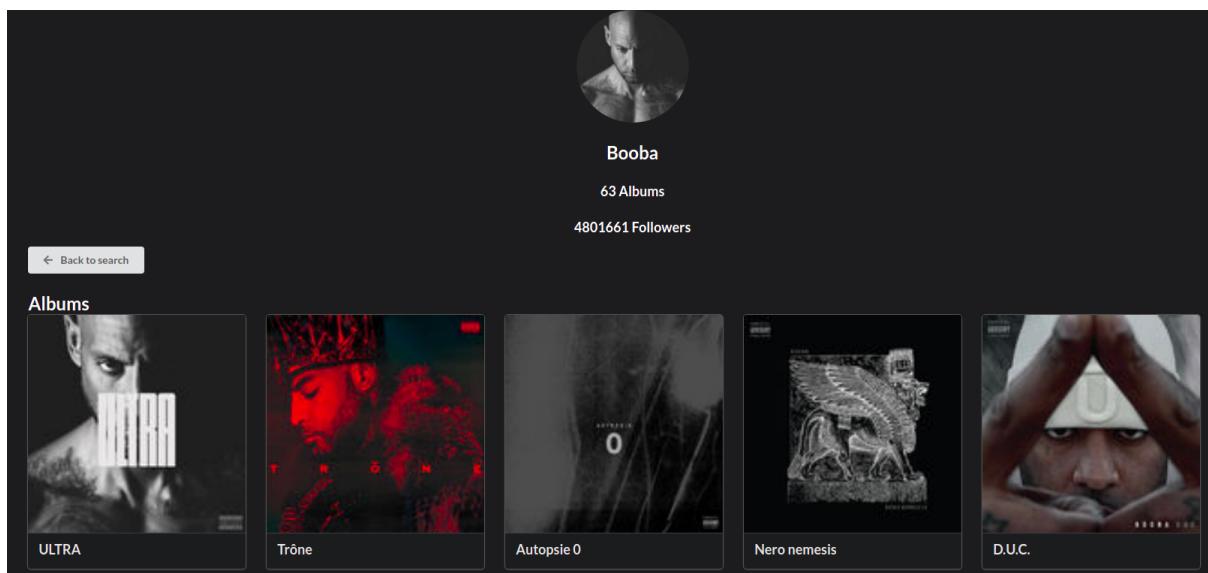
Une fois la recherche lancée, une liste d'artistes/d'albums est alors affichée en corrélation avec ce que l'on cherche.



Liste des artistes liée au mot de recherche "booba"

Lorsque que l'on clique sur un artiste, on arrive sur une page contenant :

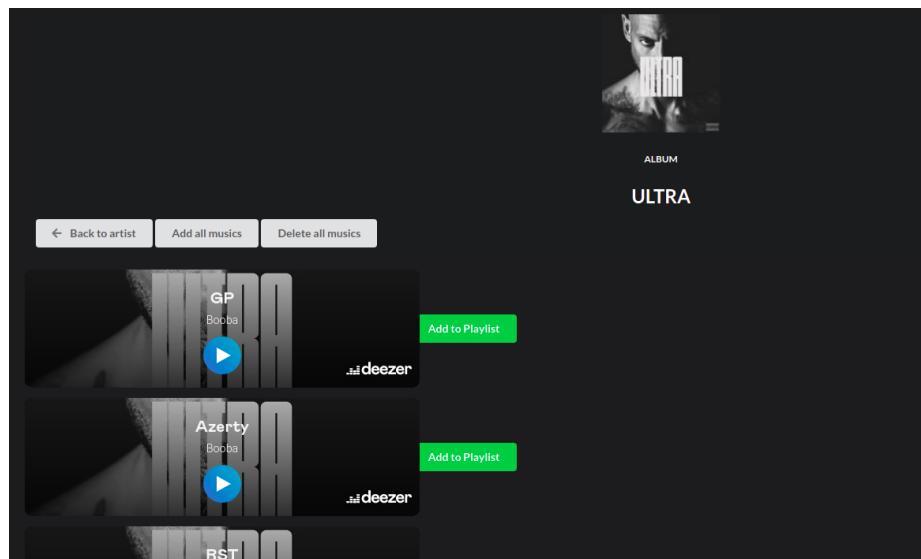
- Une image de l'artiste;
- Son Nom et Prénom;
- Son nombre de followers;
- Une liste de tous ses albums;
- Un bouton permettant de retourner sur la barre de recherche.



Exemple de la page de l'artiste M.Pokora

Puis si on clique sur l'un de ces albums, on arrive sur une page contenant :

- L'image de l'album choisi;
- Le nom de l'album;
- Des boutons permettant de retourner sur la page de l'artiste, ajouter toutes les musiques dans la playlist ou toutes les enlever de la playlist;
- La liste de toutes les musiques de l'album choisi.



Exemple de 2 musiques de l'album Ultra de Booba

De plus, nous pouvons écouter 30 secondes de chaque musique grâce au widget Deezer. Si la musique nous plaît, on peut alors l'ajouter à notre playlist ou là elle sera écoutable en entier.

3 - Technologies utilisées

3-1 Langages utilisés



GazePlay a été développé avec Java qui est un langage de programmation orienté objet. On utilise l'API JavaFx de Java car celui-ci contient des outils très divers, notamment pour les médias audio/vidéo et le graphisme 2D/3D.



Mon projet a été développé avec Angular car celui-ci permet :

- d'avoir des composants pour créer facilement des applications Web évolutives;
- d'avoir recours à une collection de bibliothèques bien intégrées qui couvrent une grande variété de fonctionnalités (routage, formulaire, communication client-serveur, etc ...);
- d'avoir une suite d'outils de développement qui aident à développer, construire, tester et mettre à jour facilement notre code.

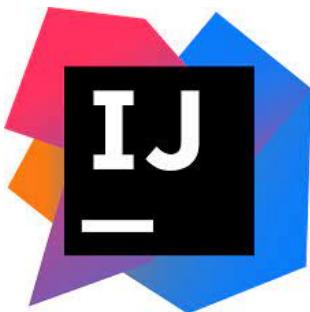
Plus précisément, c'est Angular CLI que j'ai utilisé car celui-ci facilite un certain nombre de tâches et celui-ci comprend le Html5, Css et TypeScript.



Dans mon projet, j'ai utilisé IndexedDB qui est une API permettant le stockage côté client de quantités importantes de données structurées.

De plus, IndexedDB est une base de données orientée objet basée sur JavaScript.

3-2 Logiciels utilisés



Pour développer, j'ai utilisé IntelliJ IDEA qui est un environnement de développement intégré de technologie Java destiné au développement de logiciels informatiques.

Mais je l'utilise surtout car celui-ci prend en charge Angular CLI et son utilisation est plutôt simple pour moi car c'est l'IDE que j'ai utilisé majoritairement durant toute ma scolarité.



En raison de la pandémie (COVID-19), nous avons utilisé Discord qui est un logiciel gratuit afin de pouvoir travailler en distanciel.

Discord est basé sur un principe de serveurs, ce qui signifie que l'entreprise a son propre serveur que j'ai dû rejoindre.

Nous faisons nos réunions hebdomadaires là-dessus en visioconférence en partageant nos avancements via un partage d'écran.



Pour notre versionning, nous avons utilisé GitHub, qui est un service web d'hébergement et de gestion de développement de logiciels.

4 - Gestion du travail

4-1 Organisation générale

Chaque semaine, nous avons une réunion afin de présenter à notre client ce que l'on a fait durant la semaine de travail.

Chaque fonctionnalité est regardée puis elle est soit valider soit à modifier.

Enfin nous organisons la prochaine semaine de travail, c'est-à -dire, quelles sont les fonctionnalités prioritaires à faire.

4-2 Organisation d'une fonctionnalité

création de la branche

- ↳ commits/pushes de notre code
- ↳ création d'une pull request
- ↳ révisions de notre pull request
- ↳ changements de notre code jusqu'à validation
- ↳ merge de notre code dans le projet

5 - Bilan et perspectives

5-1 Bilan

Durant ce stage de 6 mois, j'ai pu apprendre de nouveaux langages, de nouveaux logiciels et améliorer mes acquis obtenus durant ma scolarité.

J'ai commencé mon stage par GazePlay, celui-ci m'a permis d'apprendre comment gérer un projet quand plusieurs personnes travaillent dessus en même temps et de me familiariser avec l'environnement de l'entreprise.

Puis durant les 4 derniers mois de mon stage, je suis passé à plein temps sur le nouveau projet GazeMedia Player.

Dans l'informatique, je souhaite m'orienter vers la programmation web, donc lorsque mon tuteur m'a proposé un projet sous Angular j'ai toute suite voulu le faire afin d'apprendre ce langage .

Grâce à ce stage, j'ai énormément progressé sur l'utilisation de Git (Pull Request, Code review, Création d'issue, Forker un projet, Gérer des branches, etc ...) et surtout comment créer un projet de zéro.

5-2 Perspectives

Le projet GazeMedia player peut être amélioré sur plusieurs points :

- L'ajout de plus de langues : actuellement, seuls le Français et l'Anglais sont disponibles alors que sur GazePlay il y en a plus.
- Pouvoir tester mon site sur un serveur : durant toute la durée du stage (6 mois), j'ai uniquement testé mon site en localhost. Bien qu'une demande de serveur ait été faite, je n'ai pas eu la chance de l'avoir durant la durée de mon stage.
- Corriger les boutons avancer/reculer : pendant le stage, j'ai mis en place un système permettant de revenir en arrière ou en avant après avoir fait une modification dans la playlist (suppression, déplacement, ajout d'un élément dans la playlist, etc ...). En revanche, celui-ci possède encore quelques bugs, que je n'ai pas eu le temps de corriger.
- Continuer de pouvoir écouter la musique en cours lorsque que l'on se déplace sur les autres pages du site web : une fonctionnalité sur laquelle j'étais en train de travailler n'est pas totalement finie à cause de la fin de mon contrat de stage. Cette fonctionnalité avait aussi une suite, le fait de pouvoir continuer à regarder sa vidéo en cours lorsque l'on se déplace de page en page sur le site web.
- Rendre l'oculomètre utilisable : pour le moment, l'eye tracker n'est pas utilisable sur mon site web mais les bases sont présentes.

6 - Remerciements

Remerciements

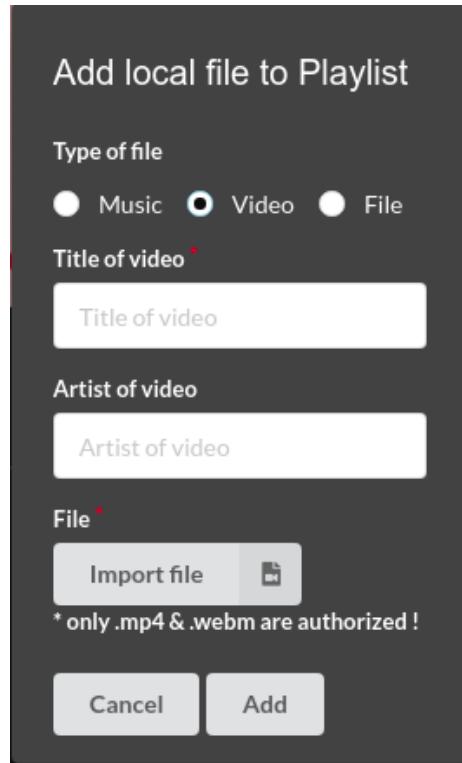
Je tiens à remercier le LIG de l'Université de Grenoble de m'avoir accueilli pour ce stage.

Je tiens aussi à remercier SCHWAB Didier pour avoir été mon tuteur industriel, ainsi que RIOU Sébastien qui a été mon autre tuteur industriel.

Enfin, je remercie DUMOND Yves pour avoir été mon tuteur académique pour ce stage.

7 - Annexes

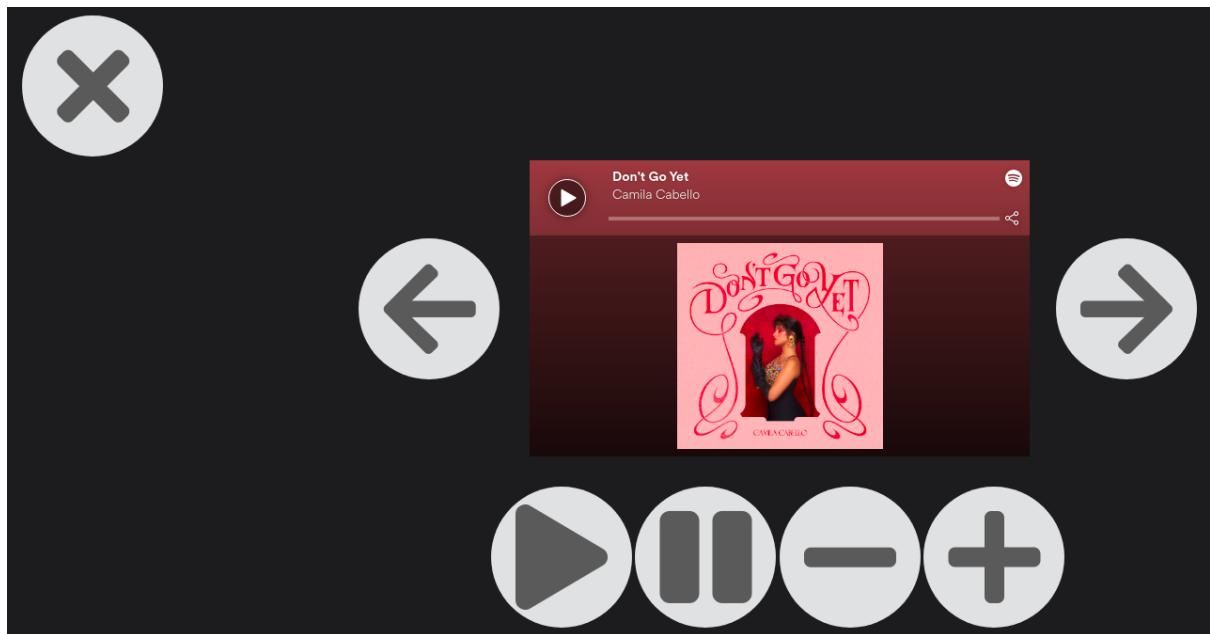
7-1 Fenêtre importation vidéo



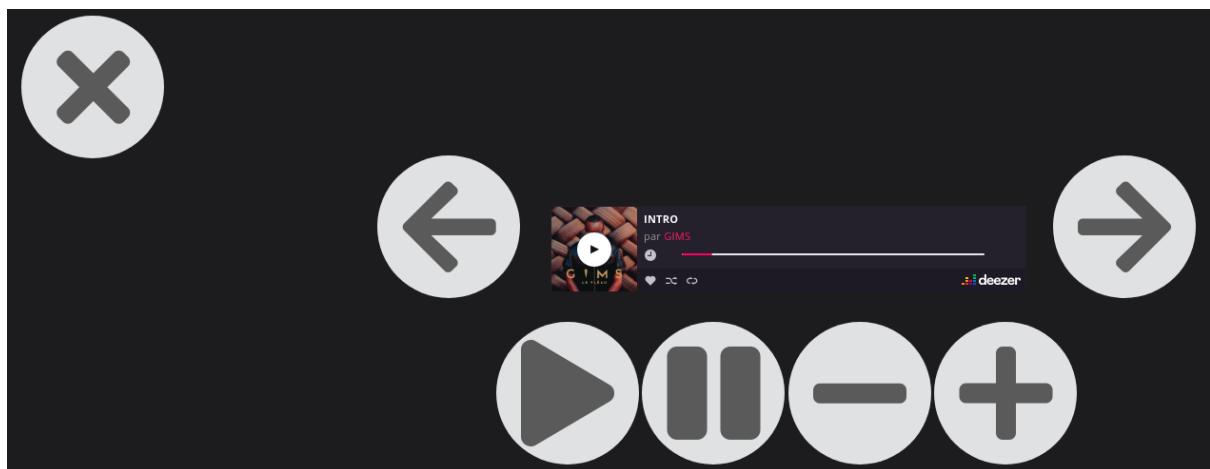
7-2 Json Validator

```
// Schema for the json validator
schemaPlaylist = {
  "properties": {
    types: { "type": "string" },
    id: { "type": "string" },
    artists: { "type": "string" },
    title: { "type": "string" },
    publishedAt: {
      type: "string",
      format: "date-time",
    },
    description: { "type": "string" },
    thumbnail: { "type": "string" }
  },
  "required": ["types", "id", "artists", "title", "publishedAt", "description", "thumbnail"],
  "additionalProperties": false
}
```

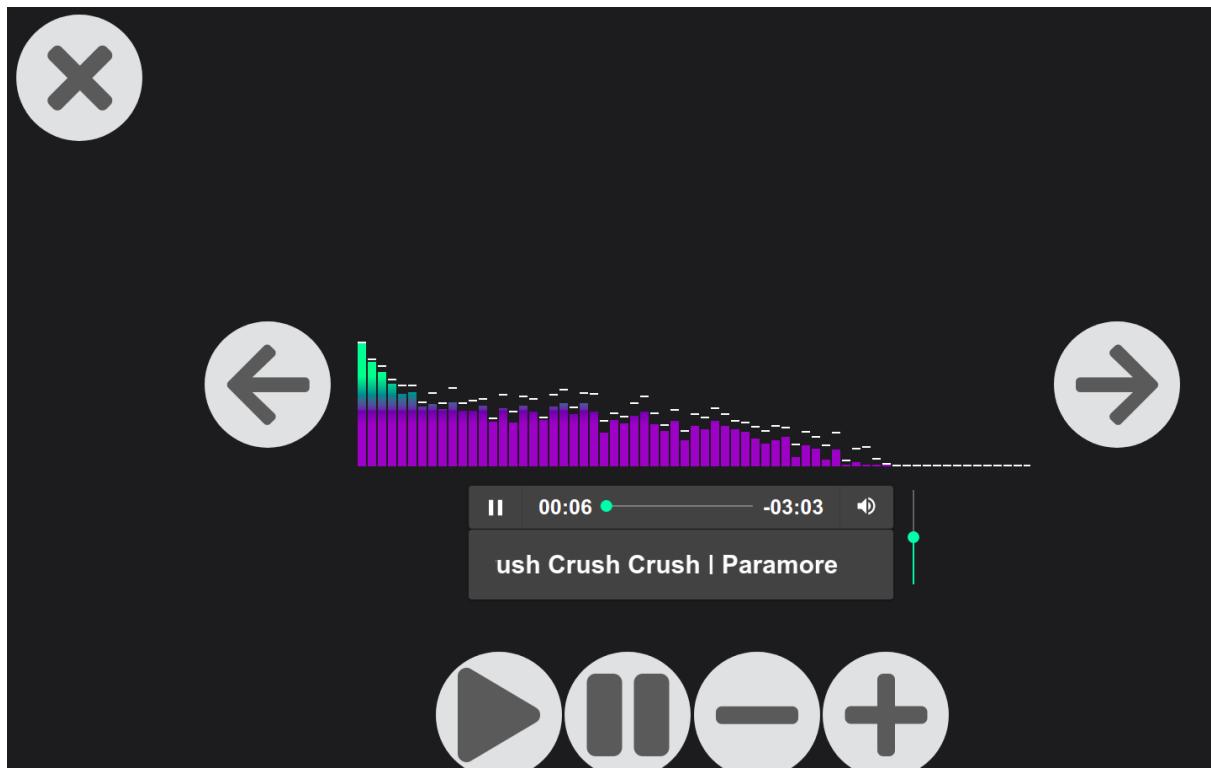
7-3 Lecteur Spotify



7-4 Lecteur Deezer



7-5 Lecteur musique importé



7-6 Lecteur vidéo importé

