

**RAPPORT PROJET ANNUEL**

**RECOMMANDATION MUSICALE**

**PERSONNALISÉE**

**GROUPE 6 :**

**AMORTILA Guillaume**

**IHOULINE Ines**

**LAMA Graih Jules**

**Année Scolaire : 2019 -2020**

<b>I- INTRODUCTION</b>	<b>3</b>
CONTEXTE	3
OBJECTIF	4
<b>II- DESCRIPTION FONCTIONNELLE</b>	<b>4</b>
Aperçu des contenus	4
La page d'accueil principale de la plateforme :	4
La page de connexion :	5
Espace de l'utilisateur une fois connecté :	6
Page "Mes Like" de l'espace utilisateur :	7
Page "mon compte" :	8
Page "Modification" :	8
Page "Mes propositions" :	9
Page "ajouter une musique" :	9
Modus operandi	10
<b>III- ARCHITECTURE TECHNIQUE</b>	<b>10</b>
Front-end	10
Back-end	11
<b>IV- DONNÉES MISES EN OEUVRE</b>	<b>13</b>
<b>V- ALGORITHMES MIS EN OEUVRE</b>	<b>15</b>
<b>VI- GESTION DE PROJETS</b>	<b>20</b>
Planning de réalisation	20
Répartition du travail	21
Problèmes rencontrés	23
<b>VII- RESULTATS OBTENUS</b>	<b>24</b>
Bilan fonctionnel	24
Améliorations possibles	25
Adresse des dépôts de code	26
<b>CONCLUSION :</b>	<b>26</b>

# I- INTRODUCTION

*Ludwig Van Beethoven* disait que "*La musique est une révélation plus haute que toute sagesse et que toute philosophie.*" L'encyclopédie du web *Wikipédia* mentionne que la musique est "*l'art consistant à arranger et à ordonner ou désordonner sons et silences au cours du temps : le rythme est le support de cette combinaison dans le temps, la hauteur, celle dans la combinaison des fréquences etc.* La musique faisant donc parti prenante de notre vie quotidienne, nous avons besoin d'elle pendant les fêtes, les cérémonies, les moments de relaxation et de détente, le jogging, la cuisine et bien d'autres.

**MELYDO** intervient en ce moment donc comme une Intelligence Artificielle consistant à proposer de nouvelles musiques pertinentes à écouter aux utilisateurs de notre plateforme.

Une **plateforme web** sera mise à disposition des utilisateurs. Elle permet de proposer de nouvelles musique en fonction d'une note qui sera calculée en fonction de nombreux paramètre.

## 1. CONTEXTE

De nombreux applications ou plateformes web existant sur le marché s'efforcent de proposer au utilisateurs des séries de playlist à travers une large diversité musicale. ces utilisateurs sont souvent contraint de se contenter à des contenus proposés limités aux simples fonctionnalités et options de sélection de genre musicaux ou encore de sélection d'artistes. De plus, de nombreux utilisateurs ont besoin d'avoir sous la main une plateforme de musique qui soit facile d'utilisation et qui puisse les fournir et les proposer instantanément les musiques de leur choix et de leur goût et ceci sans devoir fournir le moindre effort.

Dans la perspective d'évoluer dans la multiplicité d'options et de proposer un contenus plus adapté aux choix des utilisateurs il devient impératif de proposer une solution de type librairie qui pourrait apporter un plus dans les propositions musicales.

## 2. OBJECTIF

L'objectif principal que nous nous sommes fixés afin de mener à terme notre projet MELYDO est qu'à travers une architecture bien définie et une combinaison de technologies, qu'on puisse implémenter une plateforme web de recommandation spécialisée de musique de tout genre essentiellement basée sur un modèle bien spécifique d'intelligence artificielle.

## II- DESCRIPTION FONCTIONNELLE

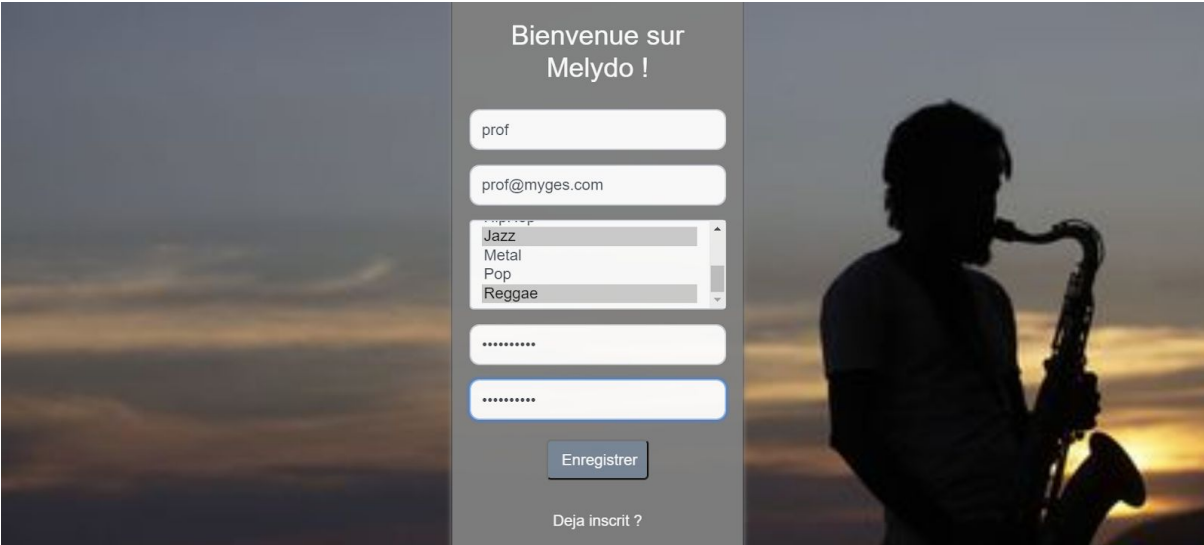
Notre plateforme web de recommandation musicale ainsi mise sur pied intègre les différentes fonctionnalités suivantes :

### 1. Aperçu des contenus

Plusieurs contenus de notre plateforme web de recommandation musicale personnalisé sont à présenter, en fonction de la fonctionnalité sollicitée par l'utilisateur, parmi lesquels :

- **La page d'accueil principale de la plateforme :**

Il s'agit de la principale page qui apparaît lorsque l'utilisateur va sur le site.



Bienvenue sur  
Melydo !

prof

prof@myges.com

Jazz  
Metal  
Pop  
Reggae

\*\*\*\*\*

\*\*\*\*\*

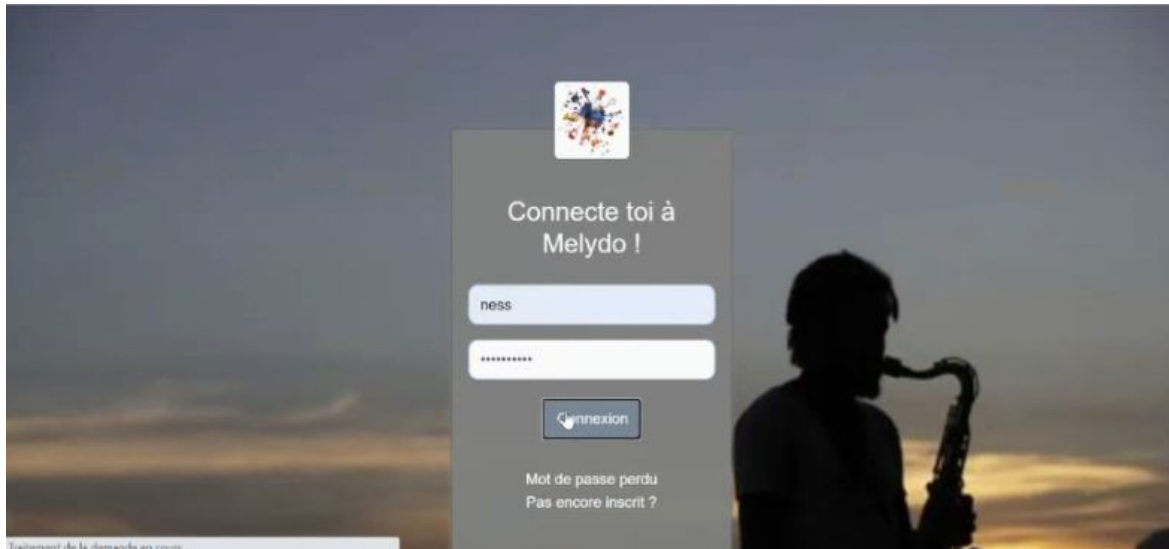
Enregistrer

Deja inscrit ?

- **La page de connexion :**

Page qui s'affiche après l'inscription et qui permet à l'utilisateur de se connecter sur son espace **MELYDO**

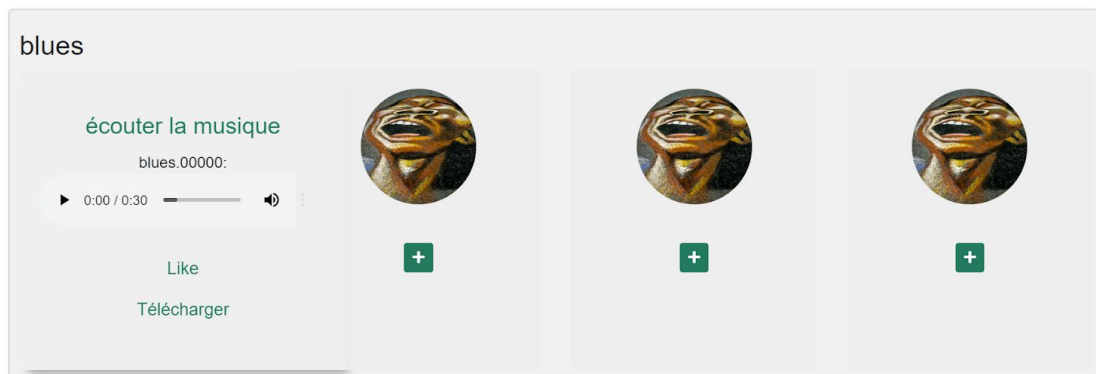
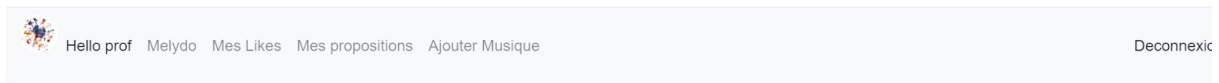
Lorsque nous venons de nous inscrire, cela nous permet de nous connecter à la plateforme.



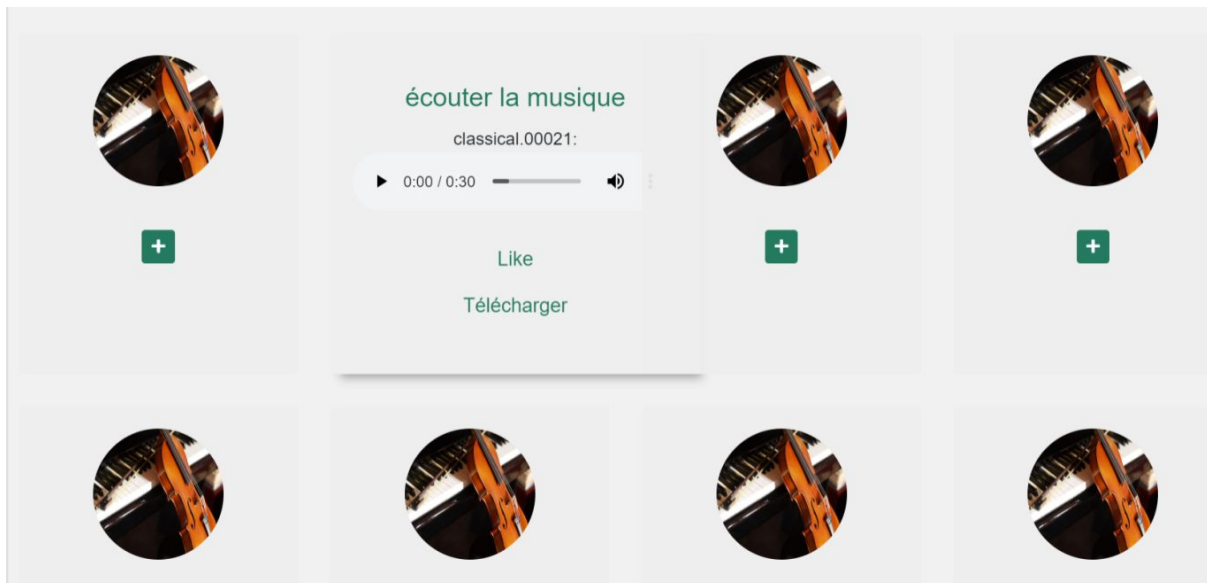
Si le mot de passe ou l'identifiant ***n'est pas*** dans la base de donnée.

- **Espace de l'utilisateur une fois connecté :**

Sur cet espace utilisateur se présente les premières musiques recommandées grâce au choix de l'utilisateur effectué pendant l'inscription sur la plateforme.



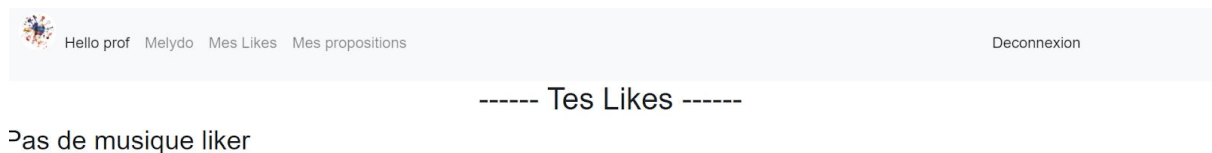
Et lorsque l'utilisateur clique sur play il peut suivre la musique :



- **Page “Mes Like” de l’espace utilisateur :**

Dans cette page, s’affiche la liste des musiques aimée ou likée par l’utilisateur. Notons qu’il est possible de revenir par exemple sur sa décision et de ne plus alors aimer la musique alors cette dernière est tout simplement retirée de la page par un simple clic sur “*je n’aime plus*”.

*Ici nous avons une liste des musiques aimé par l'utilisateur encore vide*

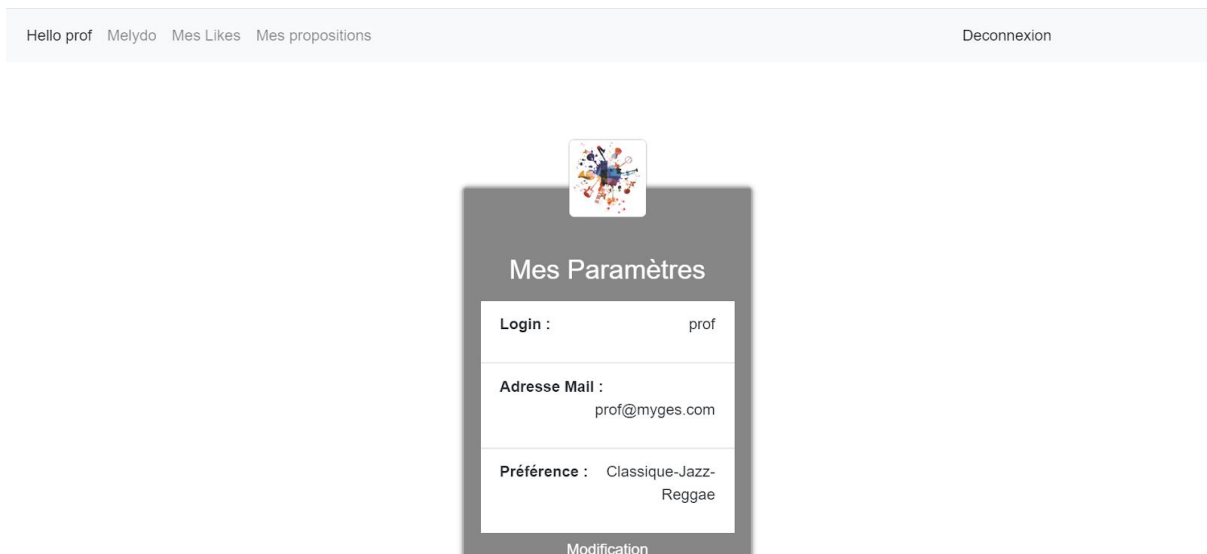


*Mais lorsque l'utilisateur aimé déjà une musique, cette dernière s'ajoute à sa liste de ses Likes :*



- **Page “mon compte” :**

Ici l'utilisateur a la possibilité de voir les paramètres de son compte. Les informations telles que le Login, l'adresse mail et les préférences musicales de l'utilisateur y sont présentes.



Hello prof Melydo Mes Likes Mes propositions Deconnexion

**Mes Paramètres**

Login : prof

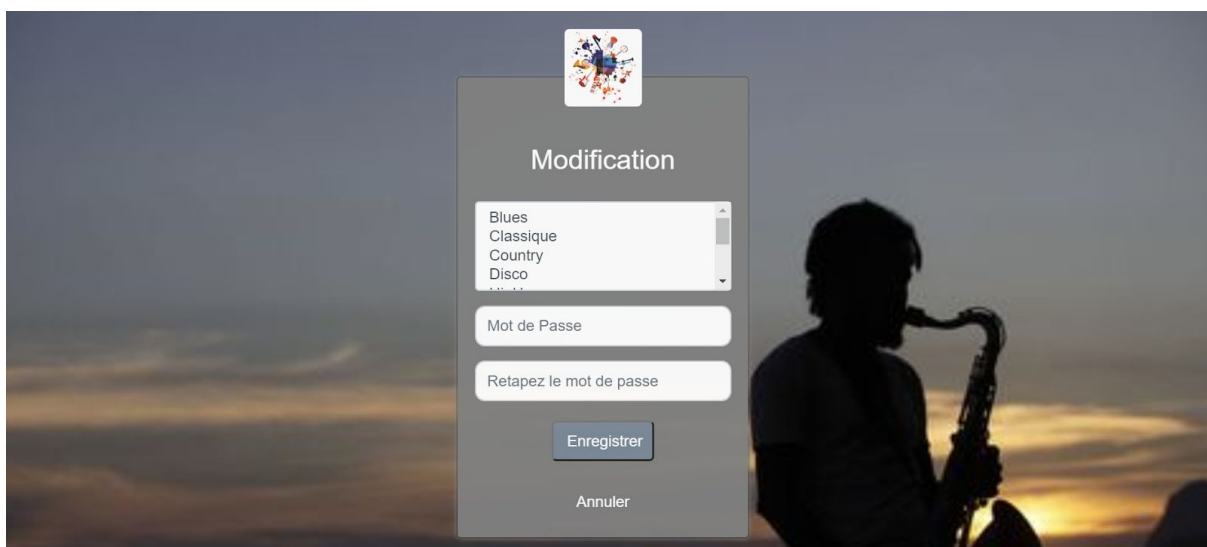
Adresse Mail : prof@myges.com

Préférence : Classique-Jazz-Reggae

Modification

- **Page “Modification” :**

Ici l'utilisateur a la possibilité de modifier son *mot de passe*, et ses *préférences* musicale pour une nouvelle recommandation.



**Modification**

Blues  
Classique  
Country  
Disco  
etc...

Mot de Passe

Retapez le mot de passe

Enregistrer

Annuler



- **Page “Mes propositions” :**

On retrouve dans cette page, les musiques proposées à l'utilisateur par le biais d'un modèle d'intelligence artificielle en fonction de certains critères.

Ici nous avons plusieurs solutions :

- Si l'utilisateur n'a pas écouté plusieurs musiques (plus de 10), nous nous basons sur ses préférences, et ainsi que les musiques 'liker' des autres utilisateurs ayant le même genre de préférences.
- Si l'utilisateur a écouté plus de 10 musiques, nous avons créé un moyen de prédire une note en fonction de plusieurs critères, notamment le nombre et le registre des écoutes de l'utilisateur, le nombre de likes ainsi que le nombre de téléchargements des autres utilisateurs.

Evidemment, dans les deux cas, nous faisons attention à ne pas proposer des musiques déjà écoutées.

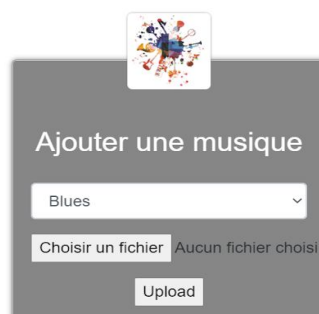
Nous avons également commencé à proposer une dernière proposition, avec de l'IA que nous développerons plus tard.

## 2. Page “ajouter une musique” :

Ici l'utilisateur a la possibilité d'ajouter une musique à la base de données en choisissant son genre. Puisque nos musiques sont triées par genre, cela permet d'ajouter la musique directement dans le bon dossier.

Hello prof Melydo Mes Likes Mes propositions

Deconnexion



Ajouter une musique

Blues

Choisir un fichier Aucun fichier choisi

Upload

### **3. Modus operandi**

Pour profiter à fond de l'expérience offerte par notre plateforme web de recommandation musicale un utilisateur doit pouvoir respecter un mode opératoire bien précise.

Lorsqu'un utilisateur arrive sur la page d'accueil de la plateforme, il doit premièrement s'inscrire en insérant les informations sollicitées. Ensuite, une fois inscrit, il peut se connecter et bénéficier des premières recommandations musicales qui lui sont proposées dans son espace personnel. Il a la possibilité d'écouter les musiques qui lui sont recommandés, les aimer ou mieux encore les télécharger.

## **III- ARCHITECTURE TECHNIQUE**

Pour maximiser l'implémentation de notre plateforme web, nous avons opté pour une architecture technique qui se repose sur deux partie à savoir la partie front-end et la partie back-end.

Ainsi donc plusieurs moyens techniques ont été utilisés pour le développement de de ces deux parties.

### **1. Front-end**

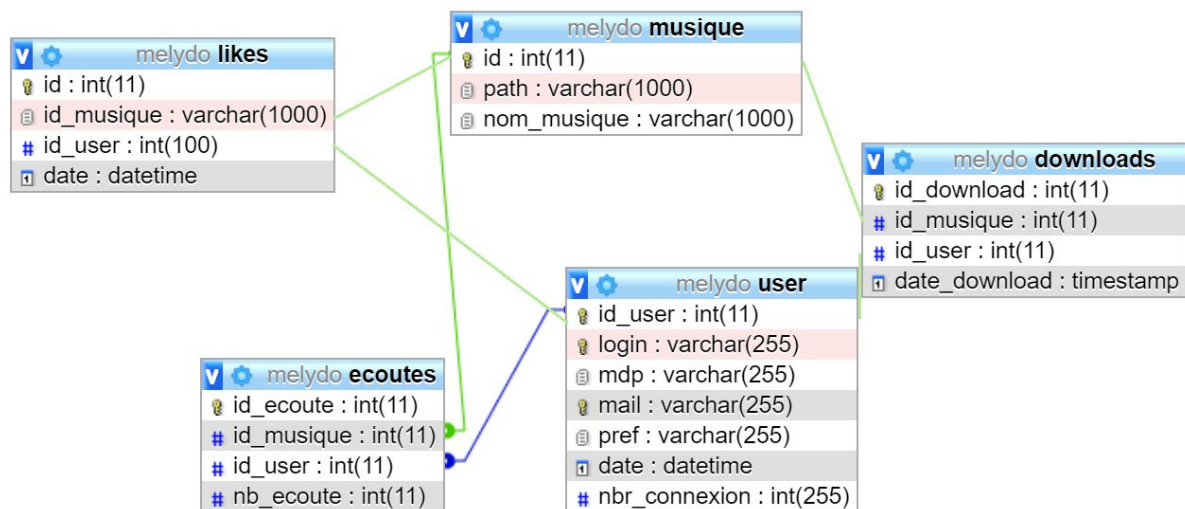
Nous avons commencé basiquement par faire un front en HTML et CSS, avec un lien entre chaque page, afin de pouvoir lister toute les fonctionnalités que nous allons avoir besoin pour le bon fonctionnement du site.

Nous avons besoin de:

- Les likes
- Inscription
- Connexion

- Déconnexion
- Télécharger musique

Grâce à cela nous avons pu créer notre base de donnée, nous avons choisi d'utiliser phpMyAdmin afin de créer une base mySQL. Voici nos différents tables.



Ensuite, pour la suite du développement du front-end, nous avons commencé par le faire en réact ainsi qu'avec du node.js, mais nous nous sommes ensuite redirigé sur le *framework Flask* de développement web en Python pour la construction dynamique des pages HTML

## 2. Back-end

Pour chaque fonctionnalité que nous avons établis préalablement, nous avons créé des servlets ainsi que des services afin de pouvoir les mettre en place.

Nous avons :

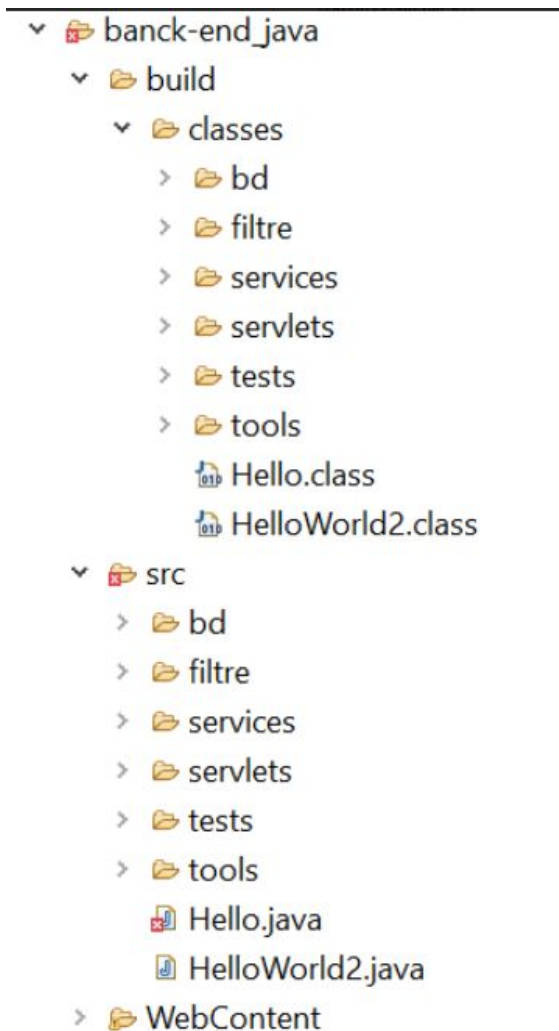
- CreateUser
- DeleteUser
- Login
- Logout
- ListMusique
- AjouterMusique
- SupprimerMusique

- ListerLike
- ajouterLike
- SupprimerLike

Chaque service appelle entre autre la connection avec la base de donnée.

Pour les servlets, ils récupèrent les données présent sur l'url et appelle le service correspondant avec comme paramètre, ce que nous venons de récupérer.

Nous avons commencé par faire des tests de tous les services en JAVA, avant de déployer sur TomCat l'application et ainsi tester les différents servlets grâce à leur url. Ses dernier test ont été réalisé grâce à Postman, afin de pouvoir tester les les différents requête : les "Post" ainsi que les "Get".



Lorsque nous nous avons changé de front-end, nous avons décidé de changer également de back-end afin de tout faire en réact, grâce au *framework Flask*.

## IV- DONNÉES MISES EN OEUVRE

Pour la mise en oeuvre de notre plateforme, nous avons utilisé plusieurs données.

Premièrement nous avons utilisé les données de type audio au format **.wav** stockées dans un dossier nommé **genres** téléchargées sur <http://opihi.cs.uvic.ca/sound/genres.tar.gz>

Nous avons également utilisé les données du dataset data.csv contenant les composants (*filename,chroma\_stft,rmse,spectral\_centroid,spectral\_bandwidth,rolloff,zero\_crossing\_rate,mfcc1,mfcc2,mfcc3,mfcc4,mfcc5,mfcc6,mfcc7,mfcc8,mfcc9,mfcc10,mfcc11,mfcc12,mfcc13,mfcc14,mfcc15,mfcc16,mfcc17,mfcc18,mfcc19,mfcc20,label*) des différents son audio.

Afin d'enrichir notre base de donnée MySQL, nous avons également généré des données utilisateurs comme :

- id\_download
- id\_musique
- id\_user
- nom\_musique
- date\_download
- id\_ecoute
- date\_ecoute

*Exemple des données dans la table écoute de notre base de données.*

Serveur: MariaDB:3306 » Base de données: melydo » Table: ecoutes					
Parcourir	Structure	SQL	Rechercher	Insérer	Exporter
Options					
id_ecoute id_musique id_user nb_ecoute					
<input type="checkbox"/>	Éditer	Copier	Supprimer	1	58 1 3
<input type="checkbox"/>	Éditer	Copier	Supprimer	2	80 1 2
<input type="checkbox"/>	Éditer	Copier	Supprimer	3	765 1 16
<input type="checkbox"/>	Éditer	Copier	Supprimer	4	389 1 14
<input type="checkbox"/>	Éditer	Copier	Supprimer	5	7 1 15
<input type="checkbox"/>	Éditer	Copier	Supprimer	6	702 1 5
<input type="checkbox"/>	Éditer	Copier	Supprimer	7	23 1 13
<input type="checkbox"/>	Éditer	Copier	Supprimer	8	88 1 7

**Données relatives aux informations personnelles de l'utilisateur (login, mdp etc...) :**

<< < 201 > | Nombre de lignes : 25 > Filtrer les lignes: Chercher dans cette table Trier sur l'index

Options

<

**Dans le web.xml on défini les servlets qui vont être appelés :**

```
web.xml
dem/banck-end java/WebContent/WEB-INF/web.xml
2= <web-app xmlns:web="http://xmlns.jcp.org/xml/ns/javaee">
3
4     <display-name>melydo</display-name>
5     <servlet>
6         <servlet-name>Test</servlet-name>
7         <servlet-class>Hello</servlet-class>
8     </servlet>
9     <servlet>
10        <servlet-name>CreateUser</servlet-name>
11        <servlet-class>servlets.CreateUser</servlet-class>
12    </servlet>
13    <servlet>
14        <servlet-name>Login</servlet-name>
15        <servlet-class>servlets.Login</servlet-class>
16    </servlet>
17    <servlet>
18        <servlet-name>LogOut</servlet-name>
19        <servlet-class>servlets.LogOut</servlet-class>
20    </servlet>
21    <servlet>
22        <servlet-name>DeleteUser</servlet-name>
23        <servlet-class>servlets.DeleteUser</servlet-class>
24    </servlet>
25
26    <servlet>
27        <servlet-name>AjouterLike</servlet-name>
28        <servlet-class>servlets.AjouterLike</servlet-class>
29    </servlet>
30    <servlet>
31        <servlet-name>AjouterMusique</servlet-name>
32        <servlet-class>servlets.AjouterMusique</servlet-class>
33    </servlet>
34    <servlet>
```

*Nous avons aussi dans un même fichier les servlets mapping, qui permettent de pouvoir l'appeler en tant qu'url :*

```
<servlet-mapping>
  <servlet-name>Test</servlet-name>
  <url-pattern>/Test</url-pattern>
</servlet-mapping>
<servlet-mapping>
  <servlet-name>CreateUser</servlet-name>
  <url-pattern>/CreateUser</url-pattern>
</servlet-mapping>
<servlet-mapping>
  <servlet-name>Login</servlet-name>
  <url-pattern>/Login</url-pattern>
</servlet-mapping>
<servlet-mapping>
  <servlet-name>LogOut</servlet-name>
  <url-pattern>/LogOut</url-pattern>
</servlet-mapping>
<servlet-mapping>
  <servlet-name>DeleteUser</servlet-name>
  <url-pattern>/DeleteUser</url-pattern>
</servlet-mapping>
<servlet-mapping>
  <servlet-name>AjouterLike</servlet-name>
  <url-pattern>/AjouterLike</url-pattern>
</servlet-mapping>
<servlet-mapping>
  <servlet-name>AjouterMusique</servlet-name>
  <url-pattern>/AjouterMusique</url-pattern>
</servlet-mapping>
```

## V- ALGORITHMES MIS EN OEUVRE

Pour l'implémentation de la partie intelligence artificielle dans notre back-end intégralement fait en Python, nous avons fait recours à plusieurs algorithmes et bibliothèques dont les plus sollicités étaient :

- **LIBROSA** : pour l'analyse musicale et audio. Il fournit les éléments de base nécessaires à la création de systèmes de recherche d'informations musicales (*spectral\_bandwidth, zero\_crossing\_rate, mfcc1, mfcc2 etc...*)
- **NUMPY** : destinée à manipuler des matrices ou tableaux multidimensionnels ainsi que des fonctions mathématiques opérant sur ces tableaux
- **PANDAS** : pour la manipulation et l'analyse des données. Elle propose en particulier des structures de données et des opérations de manipulation de tableaux numériques et de séries temporelles.
- **SCIPY** : Pour son module sur le traitement du signal

Parmi les algorithmes et scripts que nous avons mis en oeuvre pour ce projet nous avons :

- **Algorithme de transformation de pistes musicales en spectres interprétables**

Grâce à la librairie python Librosa, nous avons pu générer pour chaque musique de la bibliothèque un spectre.

- **Algorithme de recommandation basé sur la comparaison des spectres**

Dans un premier algorithme de recommandation, nous avons essayé de rassembler les spectres des utilisateurs pour ensuite comparer les spectres musicaux directement à ceux-ci. Nous avons utilisé la distance euclidienne pour obtenir des propositions de spectres similaires.

Cet algorithme fut un échec, les essais ont vite montré qu'il proposait souvent les mêmes musiques, et qu'il y avait peu de pertinence avec le résultat attendu.

- **Algorithme basé sur les spectres et les utilisateurs**

Après avoir constaté la faible efficacité de la comparaison des spectres d'utilisateurs (moyenne des spectres musicaux de ses musiques) avec les musiques, nous avons fait une tentative par similarités de spectres entre utilisateurs. Peu concluant également, nous avons vite abandonné ce test.

- **Algorithme de recommandation basé sur les préférences des utilisateurs**

Enfin, après de nombreuses tentatives vaines avec les spectres des musique pour avoir un algorithme fonctionnel, nous nous sommes basé sur les algorithmes ci dessous.



- **Algorithme de classification KNN - KNeighborsClassifier pour la prédiction des genres musicales des utilisateurs**

Cet algorithme permet à partir des données d'une musique (informations issues de l'analyse spectral avec la librairie **LIBROSA** : *chroma\_stft*, *rmse*, *spectral\_centroid*, *spectral\_bandwidth*, *rolloff*, *zero\_crossing\_rate*, *mfcc1*, *mfcc2*, *mfcc3*, *mfcc4*, *mfcc5*, *mfcc6*), de pouvoir prédire le genre musicale adapté à l'utilisateur.

```
import pandas as pd

from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier

if __name__ == "__main__":
    df = pd.read_csv('C:/Users/JULES LAMA/Documents/Dossier PA/data.csv', header = None, sep=',')
    df.drop(index=0,inplace=True)
    df = df.to_numpy()
    df_X, df_Y = df[:, 1:-1], df[:, -1]

    X_train, X_test, y_train, y_test = train_test_split(df_X, df_Y, test_size=0.1)

    knn = KNeighborsClassifier()
    knn.fit(X_train, y_train)

    y_predict = knn.predict(X_test)
    print('-----predict value is -----')
    print(y_predict)
    print('-----actual value is -----')
    print(y_test)
    count = 0

    for i in range(len(y_predict)):
        if(y_predict[i] == y_test[i]):
            count += 1

    print('accuracy is %0.2f%%'%(100*count/len(y_predict)))
```

- **Algorithme KNN - KNeighborsClassifier pour la prédiction des musiques que l'utilisateur pourra aimer :**

```

for line in ecouter.readlines():
    line = line.split(",")
    user_id = int(line[1])
    musique_id = int(line[0])
    ecouter_nb = int(line[2][:-1])

    if(user_id not in data_ecoutes.keys()):
        data_ecoutes[user_id] = []

    data_ecoutes[user_id].append(musique_id)
    data_ecoutes[user_id].append(ecouter_nb)

downloads.close()
ecouter.close()
likes.close()

data_dict = dict()

init_data_dict(data_dict, data_downloads, 0)
init_data_dict(data_dict, data_ecoutes, 73)

data_X = []
data_Y = []

max_len = get_max_len(data_likes)
for key in data_dict.keys():
    likes_key = data_likes.keys()
    if(key in likes_key):
        value_like = data_likes[key]
        value_len = len(value_like)
        for i in value_like:
            temp_value = copy.copy(value_like)
            temp_value.remove(i)
            data_sample = [key] + data_dict[key] + temp_value

            data_sample = data_sample + [0 for j in range(max_len - value_len - 1)]
            data_X.append(data_sample)
            data_Y.append(i)
            if(len(data_sample) != 533):
                data_sample.pop()

X_train, X_test, y_train, y_test = train_test_split(data_X, data_Y, test_size=0.2)

knn = KNeighborsClassifier()
knn.fit(X_train, y_train)

y_predict = knn.predict(X_test)

count = 0

for i in range(len(y_predict)):
    if(y_predict[i] == y_test[i]):
        count += 1

print('accuracy is %0.2f%%'%(100*count/len(y_predict)))

print("user id :", user_id)
print("value like :", value_like)
print("value like predict :", y_predict)

```

On a donc le résultat ci-dessous de notre modèle de prédiction avec une accuracy de **93%**. Ici l'utilisateur avec pour *id 4999* a aimé les musiques *163, 424 et 296* et le modèle lui

prédit à travers ses écoutes, ses téléchargements, ses mentions j'aime, les musiques 277, 238 569 etc..

```
accuracy is 0.93%  
user id : 4999  
value like : [163, 424, 296]  
value like predict : [277 238 569 ... 226 85 107]
```

In [29]:

- *Script de génération de données utilisateurs et de données du site :*

Ce code python nous permet de générer de manière aléatoire les données telles que l'identifiant de la musique, le titre de la musique, l'identifiant des utilisateurs et le nombre d'écoute d'une musique par utilisateur.

```
#### GENERATE USER DATA ####  
  
# users_songs : [[[id_song, note]]] => note 1..10  
  
users_genre = []  
users_genre_part = []  
users_nb_songs = []  
users_songs = []  
  
gen_files = 0  
  
if gen_files:  
    ltf = open("likes_db.csv", "w")  
    ltf.write("id_musique,title,_songs,user_id\n")  
  
    lf = open("likes.csv", "w")  
    lf.write("id_musique,id_user\n")  
  
    df = open("downloads.csv", "w")  
    df.write("id_musique,id_user\n")  
  
    ef = open("ecoutes.csv", "w")  
    ef.write("id_musique,id_user,nb_ecoutes\n")
```

```

for i in range (1,nb_users):
    user_nb_songs = 5 + int(expovariate(0.02))
    if user_nb_songs > max_listened_musics_per_users:
        user_nb_songs = max_listened_musics_per_users
    users_nb_songs.append(user_nb_songs)

    user_genre = randrange(10)
    users_genre.append(user_genre)

    user_genre_part = randrange(7)+2 # 2 .. 8
    users_genre_part.append(user_genre_part)

    user_songs = []

    for j in range(user_nb_songs):

        if (randrange(10) < user_genre_part): # take music from specific genre
            id_song = randrange(100)+(user_genre*100)
        else:
            id_song = randrange(1000)

        if (id_song == 0):
            continue

        note = int(expovariate(0.1)) + 1
        while note > 10:
            note = int(expovariate(0.1)) + 1

        if (randrange(10)+3 < note):
            like = 1
        else:
            like = 0

        if (randrange(10)+2 < note):
            download = 1
        else:
            download = 0

        #nb_ecoutes = randrange(int(note*1.3 * 1.6)) + note

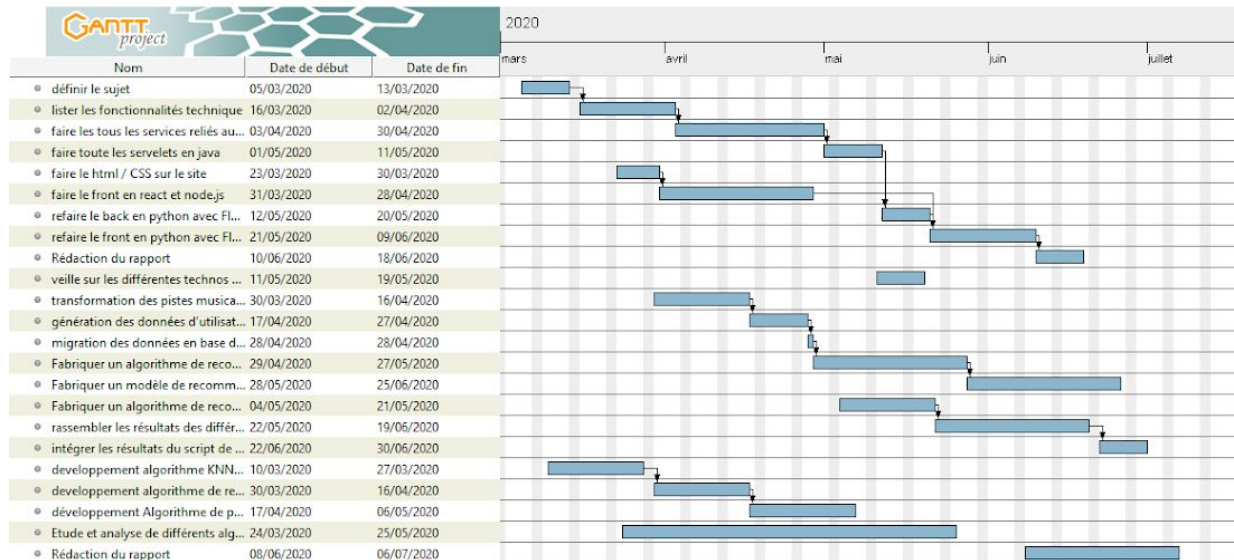
```

## VI- GESTION DE PROJETS

### 1. Planning de réalisation

Pour atteindre l'objectif fixé, nous avons organisé le déroulement du projet dans le temps. Nous nous sommes également servi de l'outil **Gantt Project** qui est un outil gratuit de gestion et de planification afin de mieux planifier la mise sur pied de notre projet dans le temps ainsi que la gestion des ressources.

En résumé, nous avons commencé par la phase de spécifications techniques et fonctionnelles que nous avons étalé sur 1 mois environ, ensuite l'implémentation des parties front-end et back-end sur 3 mois et semaine et enfin les tests sur 1 semaine



## 2. Répartition du travail

Bien qu'au départ les rôles n'étaient encore bien fixés, nous avons dû par la suite pour une bonne conduite de la mise en pratique de notre planning, nous répartir le travail en deux parties principalement.

D'une part, une personne **Inès**, en charge du développement de la partie Front-end de la plateforme web c'est à dire le développement de tout ce qui est design visuel et graphique, des pages web, de la mise en forme de l'interface etc..

*Parmi les tâches effectuées par Inès, nous avons :*

- Définir le sujet
- Lister les fonctionnalités technique
- Faire les tous les services reliés aux différentes fonctionnalités en java
- Faire toute les servlets en java
- Faire le html / CSS de lu site
- Faire le front en react et node.js
- Refaire le back en python avec Flask

- Refaire le front en python avec Flask, html, javascript
- Intégrés les modèles de données et les scripts au site web

D'autre part, deux personnes **Guillaume** et **Jules** en charge du développement de la partie back-end de la plateforme. Il s'agissait plus précisément dans cette partie de concevoir l'architecture du backend de la plateforme et l'implémentation de la partie intelligence artificielle de la plateforme à travers des algorithmes et des scripts.

***On peut citer les tâches réalisées par Guillaume :***

- Veille sur les différentes technos disponibles pour l'analyse de musique
- Transformation des pistes musicales en spectres interprétables
- Génération des données d'utilisateurs et données de site
- Migration des données en base de donnée
- Aider Ines sur le site
- Fabriquer un algorithme de recommandation basé sur la comparaison des spectres
- Fabriquer un modèle de recommandation de ML basé sur les spectres
- Fabriquer un algorithme de recommandation basé sur les préférences des utilisateurs
- Rassembler les résultats des différents algorithmes et modèles pour proposer un résultat de recommandation final
- Intégrer les résultats du script de recommandation au site

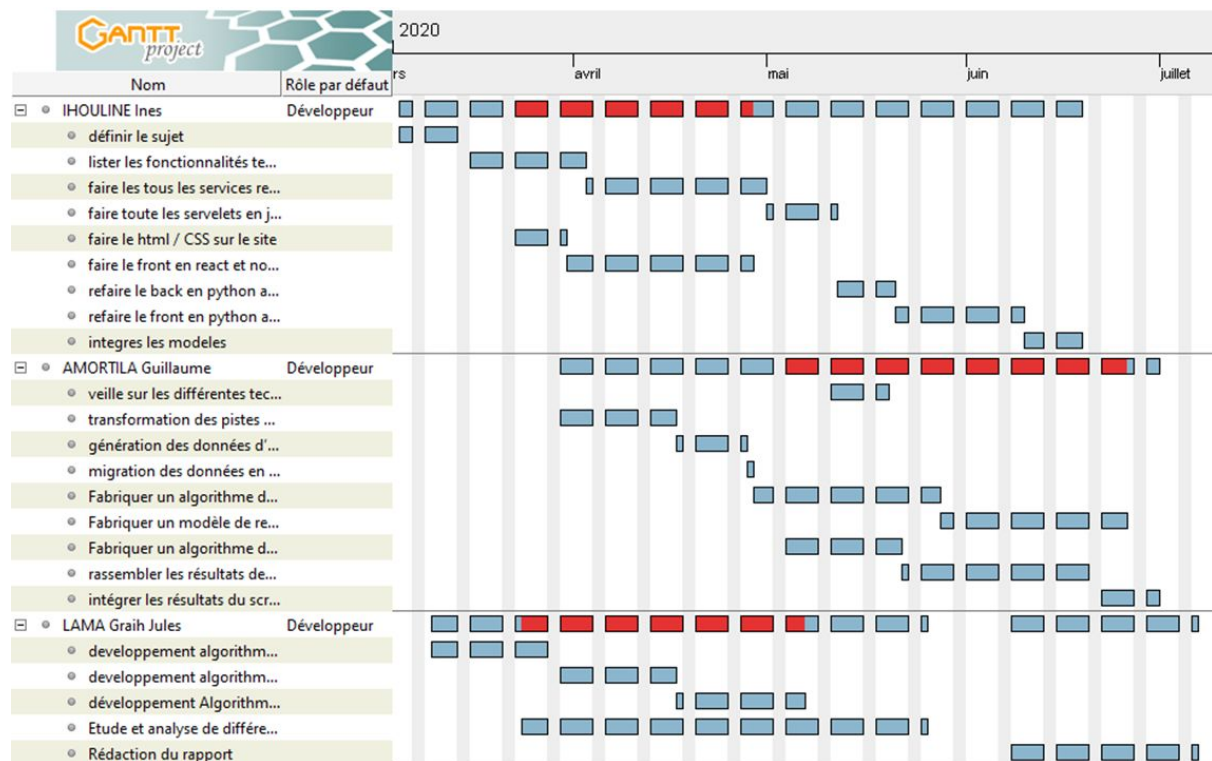
***Et les tâches réalisées par Jules :***

- Développement de l'algorithme KNN - KNeighborsClassifier
- Développement algorithme de recommandation basique
- Développement Algorithme de prédiction de genre musicale
- Développement algorithme de prédiction de like des utilisateurs
- Etude et analyse de différents algorithmes de prédiction à implémenter
- Rédaction du rapport

Ainsi pour la gestion des ressources ou plus précisément le suivi de l'évolution des différentes tâches que nous nous sommes données, nous avons élaboré un **diagramme de**



**Ressource** ci dessous, qui montre les personnes affectées à une ou plusieurs tâches en fonction du temps.



### 3. Problèmes rencontrés

Comme tous projets, nous nous sommes à plusieurs reprises heurté à de nombreuses difficultés tant sur le plan technique et fonctionnelle que sur le plan organisationnel.

On peut citer quelques unes :

- Intégration des scripts et algorithmes sur le site
- Difficultés techniques (code, nouvelles technos)
- Changement de technologies (Framework site web)
- IA : Sentiment d'impuissance face au but fixé
- Difficultés matérielles (bugs PC)
- Difficulté au niveau de la gestion de la bd
- Difficulté sur l'intégration des algorithmes de prédiction dans notre modèle

→ Etc...

## VII- RESULTATS OBTENUS

### 1. Bilan fonctionnel

Notre plateforme web de recommandation musicale ainsi implémentée, intègre plusieurs fonctionnalités côté utilisateur :

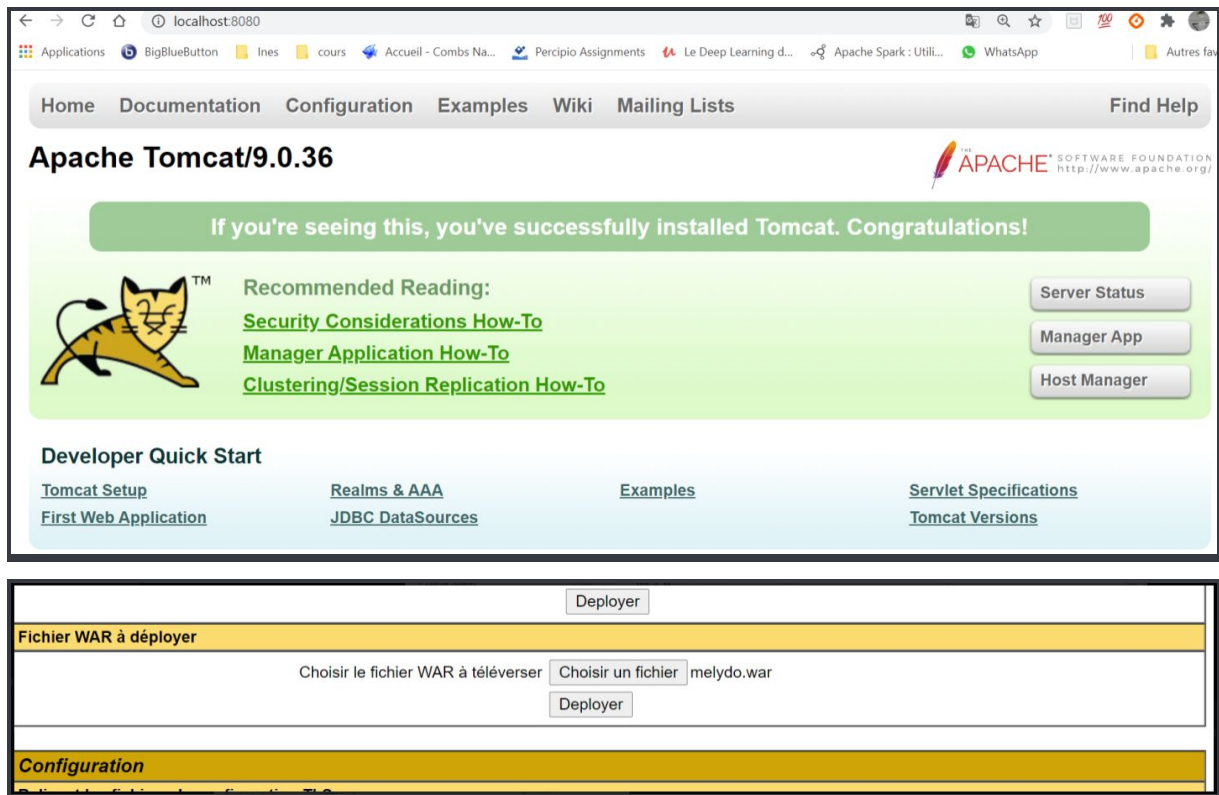
- Inscription sur la plateforme
- Connexion à la plateforme
- Modification des paramètres
- Téléchargement des musiques
- Ajout et suppression des musiques dans la liste des likes
- Déconnexion de la plateforme

Avec des tests validés, nous pouvons être sûr et certain que les fonctionnalités de notre application sont opérationnelles. On peut le voir sur les images ci-dessous :

```
1 package tests;
2
3 import java.sql.SQLException;
4
5
6
7 public class TestLogin {
8     public static void main(String in[]) throws SQLException {
9         try {
10             System.out.println(User.creerSession("anni","1234"));
11             //System.out.println(User.creerSession("Namide","Godofwar"));
12         } catch (Exception e) {
13             e.printStackTrace();
14         }
15     }
16 }
17 }
18
```

*Après, la phase des tests par la méthode post que toutes fonctionnalités répondent on procède au déploiement sur tomcat comme le montre les images ci-dessous :*





Dans l'ensemble des fonctionnalités attendus pour notre plateforme, on estime à 90% les fonctionnalités implémentées et opérationnelles.

## 2. Améliorations possibles

Aucun projet n'étant parfait, plusieurs points et aspects restent à améliorer au de notre plateforme afin de répondre à un maximum de besoin et de fonctionnalité.

- Amélioration du design visuel du front-end
- Amélioration de la pertinence des recommandation à travers la l'utilisation des algorithmes de prédiction plus efficaces
- Intégration de plus fonctionnalités :
  - Partager une musique
  - classement des musiques aimées par genre musicales dans l'espace utilisateur.
  - Sélection de plusieurs genres musicales
  - Présentation du résultat (pourcentage) de la recommandation issu de l'IA.

- Amélioration de la sécurité des données utilisateurs

### **3. Adresse des dépôts de code**

<https://github.com/IhoulineInes/MELYDO>

## **CONCLUSION :**

En conclusion, ce projet nous a permis de nous rendre compte de nous facilité mais aussi et surtout, à voir nos axes d'amélioration. Nous avons appris à appréhender des technologies dont nous n'avons pas l'habitude de travailler avec et de manipuler.

Nous avons aussi appris à nous connaitres autrement, et travailler en équipe, et ainsi essayer au mieux de mettre les qualités et les formation de chacun.

Nous ne pouvions terminer sans rappeler que malgré la pression avec les autres projets, nous avons su garder le morale haut et continuer le développement de notre projet annuel. Nous avons gagné en sens de l'organisation et un plus dans la gestion des projets en général