**LYGEN – DOCUMENTATION**

**HASSANI Leila – M2 TAL**

**Interfaces web pour le TAL & Arbres Graphes et Réseaux**

Table des matières

[Objectif du projet 1](#_Toc127789074)

[Données utilisées 1](#_Toc127789075)

[Important : obtention d’une clé API Genius 1](#_Toc127789076)

[Méthodologie et implémentation 3](#_Toc127789077)

[Etape 1 – Récupération des paroles de chanson 3](#_Toc127789078)

[Etape 2 – Entraînement du modèle de base 3](#_Toc127789079)

[Etape 3 – Finetuner GPT2 3](#_Toc127789080)

[Etape 4 – Hidden Markov Model 3](#_Toc127789081)

[Etape 5 – Interface Web 3](#_Toc127789082)

[Résultats 4](#_Toc127789083)

[Logs 4](#_Toc127789084)

[Modèles 4](#_Toc127789085)

[Génération 4](#_Toc127789086)

[Améliorations 5](#_Toc127789087)

# Objectif du projet

Personnaliser la génération de paroles de chanson en fonction du style d’un artiste.

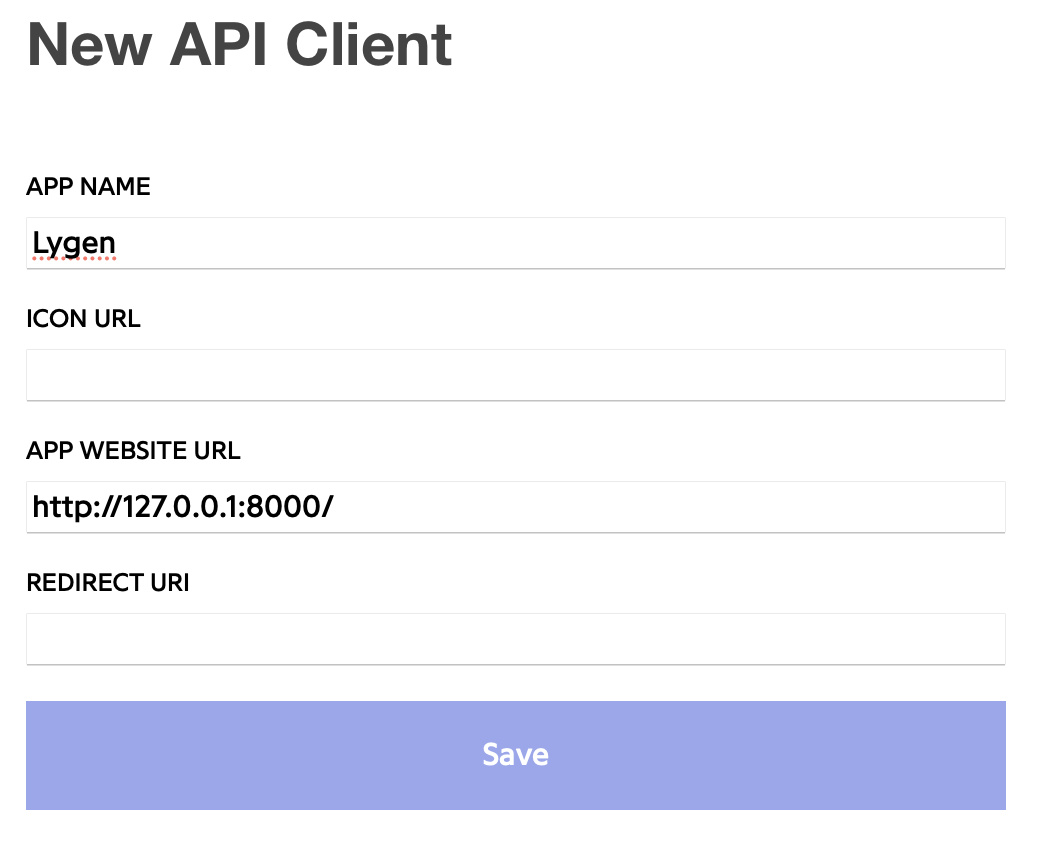
# Données utilisées

1. Un dataset [trouvé sur Kaggle](https://www.kaggle.com/code/karnikakapoor/lyrics-generator-rnn/data) afin d’entraîner le modèle Bi-LSTM (dans *base\_model/dataset/Songs.csv*)

2. Des paroles récupérées à travers l’API de Genius avec la librairie Python *lyricsgenius* afin de finetuner les modèles

# **Important : obtention d’une clé API Genius**

Pour obtenir une clé API Genius, aller sur <https://genius.com> et cliquer sur « Sign Up ». Après s’être crée un compte, aller sur <https://genius.com/developers> et cliquer sur « Create an API Client. » Taper ces informations et cliquer sur « Save » :



Il suffit ensuite de cliquer sur « Generate Access Token » :

Une image contenant texte

Description générée automatiquement

Dans le fichier lygen.py, copier l’Access Token à la place de YOUR\_API\_KEY (ligne 37) :

API\_KEY = "YOUR\_API\_KEY"

Une fois la clé API obtenue et ajoutée au fichier Python, créer un environnement virtuel et suivre ces deux étapes sur la ligne de commande :

* pip install -r requirements.txt
* uvicorn lygen:app

Se rendre sur **http://127.0.0.1:8000/.**

# Méthodologie et implémentation

## Etape 1 – Récupération des paroles de chanson

J’ai d’abord tenté d’utiliser les paroles de chansons à travers l’API de Genius directement, mais le site ne propose plus les paroles de chansons depuis plusieurs années. J’ai donc tenté de faire le scrapping moi-même à l’aide de *beautifulsoup*, mais le site n’autorise plus le scrapping de son site. Avant d’abandonner l’idée, j’ai découvert la librairie *lyricsgenius* qui – je ne sais comment – parviens à récupérer les paroles de chansons à travers l’API de Genius. Petit bémol : les paroles sont très mal nettoyées.

## Etape 2 – Entraînement du modèle de base

Le modèle a été entraîné sur Google Colab à l’aide du notebook *lyrics\_generator.ipynb*, puisque mon ordinateur personnel n’avait pas les capacités pour l’entraînement du modèle. Google Colab n’est parvenu qu’à 14 epochs sur les 30 programmées.

Une fois ce modèle de base entraîné, il a fallu régler les problèmes de compatibilité entre Mac OS et *tensorflow* afin de pouvoir lancer un *finetuning* en local après le choix d’un artiste. Une fois ces problèmes réglés, le finetuning du modèle est effectué en utilisant en entrée les données récupérées par l’API.

Pour avoir des résultats « convenables » avec ce modèle, il faut aller dans les 100 chansons et 50 epochs, mais le temps d’entraînement devient considérable et la qualité des résultats n’en vaut pas forcément le coût.

Les résultats étant très mauvais avec le Bi-LSTM, j’ai choisi d’ajouter la possibilité de finetuner GPT2.

## Etape 3 – Finetuner GPT2

L’idée m’est venue grâce au package [gpt-2-simple](https://github.com/minimaxir/gpt-2-simple) qui avait l’air d’être une manière assez simple de finetuner GPT2. Mais le package étant obsolète, je me suis finalement inspirée de [ce notebook](https://github.com/falloutdurham/beginners-pytorch-deep-learning/blob/master/chapter9/Chapter9.5.ipynb) pour le finetuning et que j’ai adapté pour la génération de paroles de chansons.

Les résultats sont meilleurs qu’avec le Bi-LSTM mais la qualité des résultats varie de génération en génération, d’où l’ajout de l’étape suivante.

## Etape 4 – Hidden Markov Model

J’ai tout simplement utilisé la librairie [*markovify*](https://github.com/jsvine/markovify). Elle offre (ironiquement) les meilleurs résultats des trois solutions, bien que ce soit la seule des trois méthodes qui n’utilise pas de Deep Learning.

Pour avoir des résultats avec ce modèle, il faut minimum 20 chansons (sinon il n’y a pas de génération), mais je recommande minimum 100 chansons pour avoir de bons résultats (ce qui peut prendre un peu de temps).

## Etape 5 – Interface Web

L’interface web a été faite avec FastAPI et Jinja. J’ai tenté de faire une barre de téléchargement (car le *finetuning* des modèles peut prendre du temps, temps pendant lequel la page apparait « figée ») mais la solution avait l’air trop complexe pour mes compétences en JavaScript et FastAPI. Le reste n’a pas posé trop de problème.

Les fichiers JavaScript et CSS se trouvent dans le dossier static. Il n’y a qu’un seul script JavaScript contenant plusieurs fonctions (*js/script.js*), et trois fichiers CSS, un fichier de style de base commun à toutes les pages (style.css, qui établit la couleur de fond d’écran, la police d’écriture, etc.) et les fichiers *css/main.css* (spécifique au fichier HTML *templates/index.html*) et *css/error.css* (spécifique au fichier HTML *templates/error.html*, qui permet d’afficher les messages d’erreur).

# Résultats

## Logs

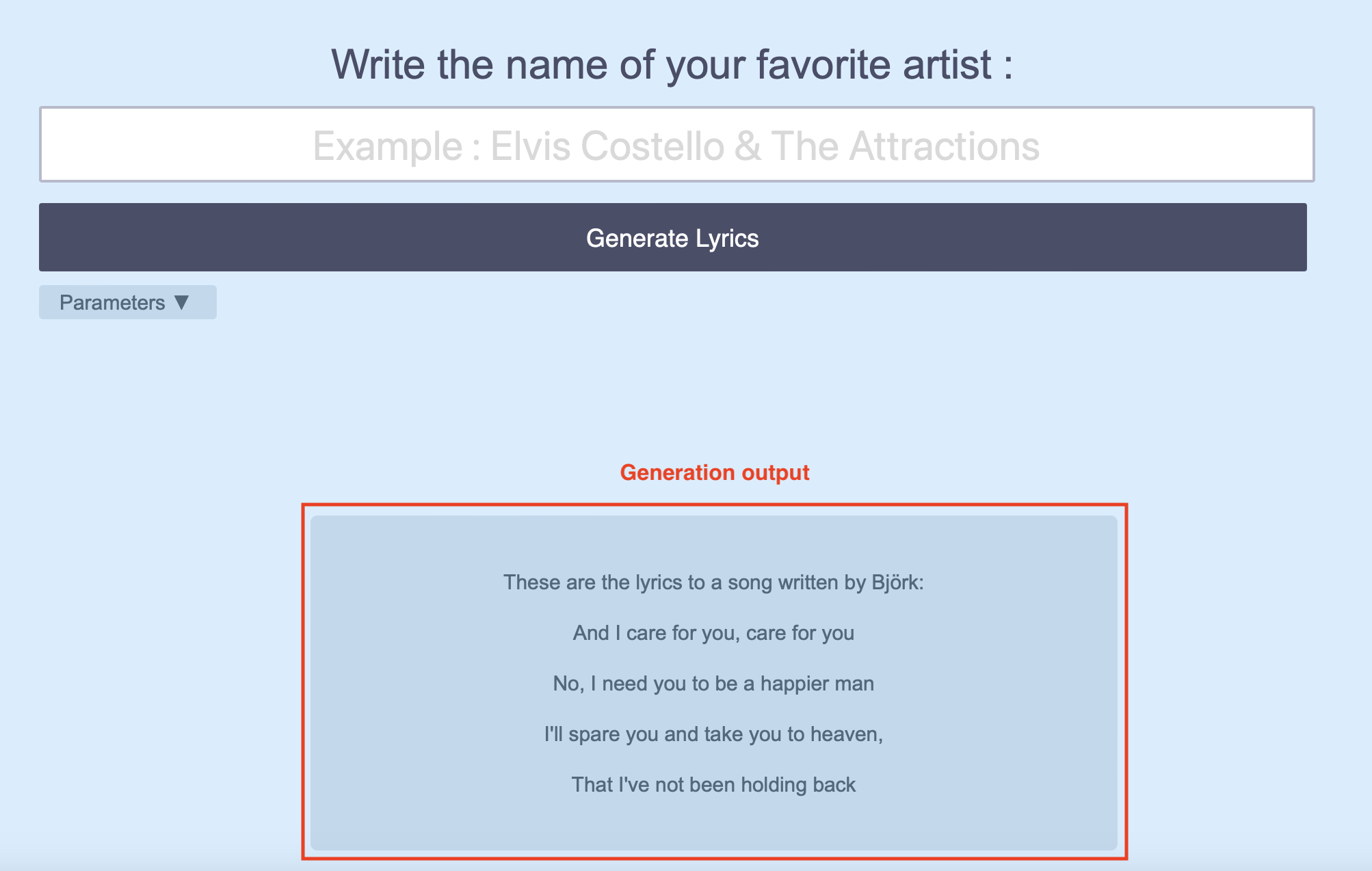
Les logs contiennent la dernière requête de l’utilisateur ainsi que les fichiers de chanson de l’artiste requêté au format CSV. Elles sont supprimées automatiquement à chaque fermeture de l’application.

## Modèles

Les modèles finetunés avec le Bi-LSTM sont dans *base\_model/finetuned\_model*s et les modèles finetunés avec GPT2 sont dans *gpt2\_finetuned\_model/finetuned\_models*.

## Génération

Les générations ne sont pas sauvegardées, elles apparaissent après le *finetuning* du modèle sur la page web.



## Améliorations

* Je ne suis pas sûre de l’utilité ou de la faisabilité, mais j’ai pensé à un pré-finetuning de GPT2
* Un meilleur modèle Bi-LSTM pré-entraîné (mais qui requiert des ressources plus importantes)
* Les modèles n’étant entraîné que pour de l’anglais, ajouter une étape de détection de la langue. Si la langue de l’artiste n’est pas l’anglais, envoyer un message d’erreur.