MUSIC EMOTION RECOGNITION ALGORITHM (MERA) USING DEEP LEARNING

BONFACE MARTIN OYWA N11/3/0486/017

Software project in partial fulfillment of the requirements for the award of Bachelor of Science in Computer Science degree of Laikipia University

DECEMBER 2021

# DECLARATIONS

I hereby declare that this project is my work and has not been submitted to any other university for purpose of examination. All the information given is my own and all the cited sources are quoted and acknowledged accordingly.

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# RECOMMENDATION

The project “Music Emotion Recognition Algorithm (MERA) using Deep Learning” has been presented to the Computing and Informatics Department of Laikipia University. We have received the thesis and recommend it to be accepted in partial fulfillment of the requirements for the award of Bachelor of Science in Computer Science.

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ms. Lorna Ogake

Department of Computing and Informatics, Laikipia University

# COPYRIGHT

All rights reserved no part of this thesis may be reproduced or stored in any retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, and recording without prior written permission of the author or Laikipia University on that behalf.

# ACKNOWLEDGMENT

This is to acknowledge the Almighty for His providence and blessings during the whole course of undertaking our project. I acknowledge all the lecturers in the Department of Computing and Informatics, Laikipia University who impacted me with the skills that were essential in executing the project. It would have been an uphill task to work on the project without the prerequisite programming, Software Engineering, and AI knowledge.

Specially, I acknowledge my supervisor Ms. Lorna Ogake for her guidance, all-time correction, and the timely recommendations that enabled the successful completion of the project. I gladly acknowledge my course mates who gave their insights and advice during the implementation of this project.

I also acknowledge the researchers at Deezer, and researchers whose research papers heavily drove the implementation of this project, for their works in the field of Neural Networks for Music Information Retrieval. They were the greatest source of inspiration.

# DEDICATION

I dedicate this project to the Department of Computing and Informatics, Laikipia University. To my supervisor Ms. Lorna Ogake who patiently listened to my presentations and gave me the guidance I needed to complete the project and make it a success.

To my fellow students who supported me through words of encouragement and technical assistance whenever I reached out to them. I cannot also forget my mother and grandparents who contributed immensely to my learning process. The financial and emotional aid ensured that I had peace of mind as I worked on my project and had access to all the necessary materials.

# ABSTRACT

Music Emotion Recognition is a concept in Music Information Retrieval which hasn’t been around for a very long time. It is involved in determining or classifying emotions or moods that humans perceive in music using computer programs.

In this project, I focused on the use of Deep Learning, a field in Artificial Intelligence that applies a concept called Deep Neural Networks (DNNs) for learning representations or patterns from data, which helps the networks perform their specified tasks such as classification of labels. Originally DNNs were only used in the classification of images, but can now be used even on the classification of text as part of Natural Language Processing (NLP), more specifically Natural Language Understanding. Both tasks will be used for emotion or mood recognition, later described in this project.

The project is further built on the concepts of valence and arousal in music, which represent negative to positive moods, and calm to energetic moods respectively. Both the audio and lyrics of music are used since arousal correlates well to the audio while valence correlates well to both audio and lyrics. The goal here is to use Deep Learning for the classification of mel-spectrograms defined from music audio in image classification, and classification of music lyrics in text classification then averaging out the results to come up with a prediction of emotion. Two models will therefore be used in this project.

TABLE OF CONTENTS

[DECLARATIONS I](#_Toc93838777)

[RECOMMENDATION II](#_Toc93838778)

[COPYRIGHT III](#_Toc93838779)

[ACKNOWLEDGMENT IV](#_Toc93838780)

[DEDICATION V](#_Toc93838781)

[ABSTRACT VI](#_Toc93838782)

[CHAPTER 1 – INTRODUCTION 1](#_Toc93838783)

[1.1 BACKGROUND OF STUDY 1](#_Toc93838784)

[1.2 PROBLEM DEFINITION 1](#_Toc93838785)

[1.3 SYSTEM OBJECTIVES 1](#_Toc93838786)

[1.3.1 GENERAL OBJECTIVE 1](#_Toc93838787)

[1.3.2 SPECIFIC OBJECTIVES 2](#_Toc93838788)

[1.4 JUSTIFICATION 2](#_Toc93838789)

[1.5 PROJECT SCOPE 2](#_Toc93838790)

[1.6 RESEARCH LIMITATIONS 2](#_Toc93838791)

[CHAPTER 2 – LITERATURE REVIEW 3](#_Toc93838792)

[2.1 INTRODUCTION 3](#_Toc93838793)

[2.2 MUSIC EMOTION RECOGNITION (MER) 3](#_Toc93838794)

[2.3 DEEP LEARNING 4](#_Toc93838795)

[2.4 FASTTEXT, LSTM + GLOVE AND BERT 4](#_Toc93838796)

[2.5 TRANSFER LEARNING 7](#_Toc93838797)

[CHAPTER 3 – METHODOLOGY 9](#_Toc93838798)

[3.1 DATA COLLECTION AND ANALYSIS 9](#_Toc93838799)

[3.1.1 COLLECTION OF MUSIC LYRICS 10](#_Toc93838800)

[3.1.2 COLLECTION OF SONG AUDIO 11](#_Toc93838801)

[3.1.3 QUADRANTS AS LABELS 12](#_Toc93838802)

[3.1.4 CLEANING DATA 13](#_Toc93838803)

[3.2 MODEL ARCHITECTURES, TRAINING AND EVALUATION 13](#_Toc93838804)

[3.2.1 AUDIO MODEL 13](#_Toc93838805)

[3.2.2 LYRICS MODELS 15](#_Toc93838806)

[3.3 SOFTWARE REQUIREMENTS 16](#_Toc93838807)

[3.4 HARDWARE REQUIREMENTS 17](#_Toc93838808)

[CHAPTER 4 – PROJECT PRESENTATION 18](#_Toc93838809)

[4.1 DATA FLOW DIAGRAMS (DFDs) 18](#_Toc93838810)

[4.2 PROJECT SCREENSHOTS 22](#_Toc93838811)

[CHAPTER 5 – RECOMMENDATIONS AND CONCLUSION 29](#_Toc93838812)

[REFERENCES 30](#_Toc93838813)

# CHAPTER 1 – INTRODUCTION

This chapter gives an introduction to music recommendations. The chapter further gives background information of the project and a problem statement, as well as the objectives and a justification of the project if implemented.

# 1.1 BACKGROUND OF STUDY

Music recommendation has come a long way such that a single search of a song almost guarantees you a continuous stream of other songs that you might like, and almost always do. These recommender systems usually use techniques such as collaborative filtering, and audio models to recommend music. This means that the music that gets recommended always has similarities with music you already listen to or what group of users (in collaborative filtering) you belong to.

Emotions in humans usually have a tendency to be erratic, and recommender systems sometimes won’t always be able to pick up on this since a user's current emotional state may force them to deviate from the usual user space or profile the system has built on them, meaning that a song being recommended may not be able to fit in well with a user’s current emotion, thus using music emotion recognition seems like something that will always be able to tailor music based on emotions, and be used by existing recommender systems to determine which songs based on the emotion they should recommend.

# 1.2 PROBLEM DEFINITION

For most music streaming companies, the above-mentioned methods of music recommendation are mainly for financial gain, so they tend to be very effective in terms of getting music in front of the users of their platform, thus there is some margin of error in terms of the experience they are willing to forgo regardless of how it affects a user's experience.

From my experience, their systems cannot usually create recommendations that necessarily have the emotional impact which as a listener, I may be in search of, especially when I need to slip into and out of certain moods.

# 1.3 SYSTEM OBJECTIVES

## 1.3.1 GENERAL OBJECTIVE

To study the techniques of Deep Learning applications in music emotion recognition and how they can be used by recommender systems.

## 1.3.2 SPECIFIC OBJECTIVES

1. To implement a music emotion recognition algorithm using audio mel-spectrograms and lyrics from songs.
2. To apply the algorithm in the creation of playlists or clusters based on emotion.
3. To create two models whose outputs can be used to classify new music’s cluster.
4. To explain how existing recommender systems can tap into these models for better music recommendations based on emotions or moods.

# 1.4 JUSTIFICATION

Most users have a tendency of searching for music that is in sync with their current mood or emotional state, and since music streaming platforms usually recommends music based on past listens and a listener’s similarity with others, they usually miss to meet these states such that a listener gets presented with a mix-match of what they might like which may deviate from their current interests emotionally.

This project’s justification is tied in well into using Deep Learning to aid in music classification or clustering and the creation of playlists or clusters that the streaming platforms can tap into to recommend music based on emotion.

# 1.5 PROJECT SCOPE

This project only focuses on the creation of the models, their deployment on a web application, and their use in clustering music based on their predicted value. No recommender system will be built but as per specific objective iv., an explanation on how they can be used would be discussed.

# 1.6 RESEARCH LIMITATIONS

The main challenges encountered during research or study included the following;

1. Getting clear access to how their systems currently handle recommendation of music based on emotions since this may be considered confidential as it’s may be a revenue-generating feature.
2. Accessing music audio and lyrics was limited due to copyright issues, so data collected may have not been sufficient enough to get very accurate models.
3. Most lyrics collected were only 30% of the actual song lyrics, thus mapping 30% of a song to actual audio was challenging thus entire audio was used.

# CHAPTER 2 – LITERATURE REVIEW

# 2.1 INTRODUCTION

This section presents a comprehensive overview of Music Information Retrieval (MIR) and its ties to Music Emotion Recognition (MER), Deep Learning (DL), and an in-depth look into Transfer Learning for multi-class image classification in Computer Vision, fastText, Long Short-Term Memory (LSTM) + Glove, and Bidirectional Encoder Representations Transformer (BERT) for multi-class text classification in Natural Language Processing, which are heavily applied in this project.

# 2.2 MUSIC EMOTION RECOGNITION (MER)

Music Information Retrieval (MIR) is an interdisciplinary science of retrieving information from music, whose aim is to develop computational tools for processing, searching, organizing, and accessing music-related data.

The field of Music Emotion Recognition (MER) has been evaluated consistently since 2007 in the Music Information Retrieval Evaluation Exchange (MIREX) Audio Mood Classification task. This task consists of the classification of audio into five mood categories or clusters, containing different emotional tags. Emotions in this context refer to what listeners perceive since; 1. It’s relatively less influenced by situational factors of listening e.g environment, and 2. Listeners are generally consistent in their ratings of the emotional expression of music [1].

In terms of the definition of emotions, the following attributes and characteristics of emotions were considered; 1. Emotions appear to vary their intensity e.g irritation – rage, 2. Emotions appear to involve distinct qualitative feelings e.g feeling of being afraid – nostalgic, 3. Some pairs of emotions appear more similar than others e.g joy and contentment versus joy and disgust, and 4. Some emotions appear to have opposites e.g happiness – sadness, love – hate, calm – worry. These attributes altogether led to the definition of the following two approaches for representing emotions;

1. Categorical approach where emotions are represented as categories distinct from each other, i.e happiness, sadness, anger, surprise, and fear.
2. Dimensional approach, on the other hand, is where we have emotions conceptualized based on their positions on a small number of dimensions i.e valence and arousal, which is what I’ll be using in this project as it helps reduce the dimension of training a MER model.

Valence refers to the space from negative to positive mood, and arousal is the space from calm to energetic. The paper [2] further explains that Arousal is highly correlated to the audio while valence is highly correlated to both audio and lyrics. The project decision, therefore, was to base the mood or emotion detection based on audio and lyrics.

For this project, I’ll be applying a concept in the paper [1] called 4Q, where emotional categories are further categorized into 4 Quadrant Dimensions of Arousal and Valence as shown in the table below;

|  |  |  |
| --- | --- | --- |
| **Quadrant** | **Emotions** | **Synonyms** |
| Q1 (A+V+) | Joyful activation | joy |
| Power | - |
| Surprise | - |
| Q2 (A+V-) | Anger | angry |
| Fear | anguished |
| Tension | tense |
| Q3 (A-V-) | Bitterness | bitter |
| Sadness | sad |
| Q4 (A-V+) | Tenderness | gentle |
| Peace | - |
| Transcendence | spiritual |

**Table 1.0 Quadrant Table**

# 2.3 DEEP LEARNING

Deep Learning is an extension of Machine Learning, that focuses on the use of multi-layer neural networks. These neural networks attempt to simulate the behavior of the human brain, allowing them to “learn” for large amounts of data.

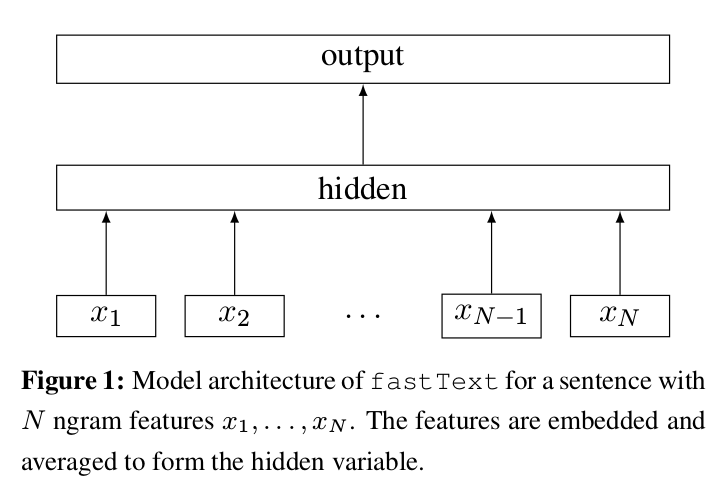
Some Deep Learning architectures include Deep Neural Networks, Recurrent Neural Networks, and Convolution Neural Networks, and they have been applied widely in fields such as Computer Vision, Speech Recognition, and Natural Language Processing, among others.

Music Information Retrieval (MIR), and more specifically Music Emotion Recognition, since it involves the automatic classification of music, using Audio (can be represented as an image) and Lyrics (text), can apply Deep Learning in Image and Text Classification, which is used in this project. Some concepts on how this would be done are described next.

# 2.4 FASTTEXT, LSTM + GLOVE AND BERT

The named models in the subtitle were used in this project to compare their performances in Music Emotion Recognition using lyrics. More on their comparison will be discussed later.

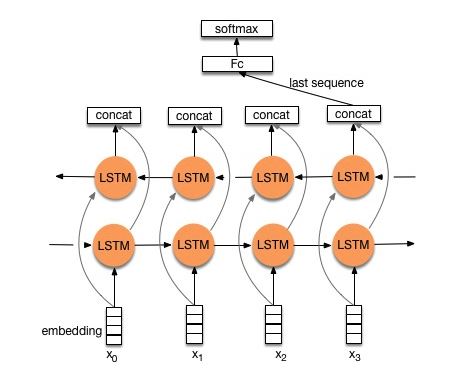
FastText is a library for efficient learning of word representations and sentence classification. It is a simple yet powerful model proposed by Facebook Research, and it is one of the fastest in sentence classification comparable in performance to much complex neural network-based ones. Its architecture is shown in the figure below;



**Figure 2.0 fastText model architecture**

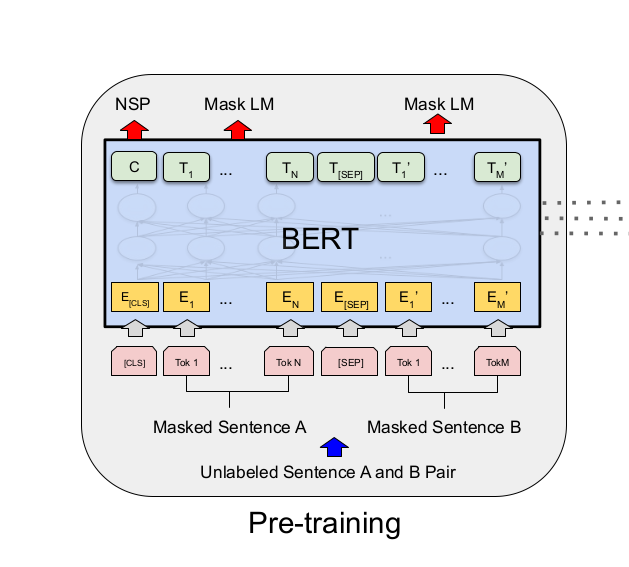
Since finding the best hyperparameter is crucial for building efficient models, this library has an autotune feature that allows the automatic finding of the best hyperparameters for a given dataset. This feature was applied in the creation of this model for the project.

LSTM or Long Short Term Memory is a type of Recurrent Neural Network that has feedback connections that helps it in processing not only a single data point but also entire sequences such as text. GloVe on the other hand is an unsupervised learning algorithm for obtaining vector representations for words (<https://nlp.stanford.edu/projects/glove/>). LSTM+GloVe refers to the technique of using pre-trained GloVe word vectors or embeddings. This helps in training models without necessarily having to create your own word embeddings which can be computing intensive and time-consuming. The architecture of this model is as shown in the figure below;



**Figure 2.1 LSTM model architecture**

The last model used in the project for lyrics classification is BERT or Bidirectional Encoder Representation Transformer. It is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering, text classification, and language inference, without substantial task specific architecture modifications [3]. The architecture of BERT is as shown in the figure below;

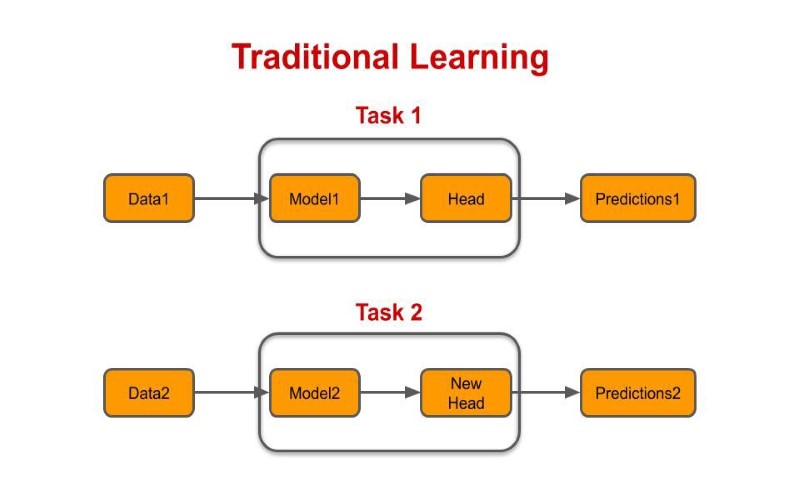


**Figure 2.2 BERT model architecture**

A variation of BERT called bert-base-uncased pre-trained model from the HuggingFace transformers library was used in the project, where I assumed that for example, the words BAD and bad, both shared the same sentiment and since the base model only had 110 million parameters, I could train locally on my computer.

# 2.5 TRANSFER LEARNING

Transfer learning is a research problem in Machine Learning (ML) that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem. For example, knowledge gained while learning to recognize cars could apply when trying to recognize trucks. It can be considered the reuse of a pre-trained Deep Learning (DL) or ML on a new problem. A simple illustration is as shown in the figure below;



**Figure 2.3 Transfer Learning illustration**

For this project, only the BERT model will be used for transfer learning.

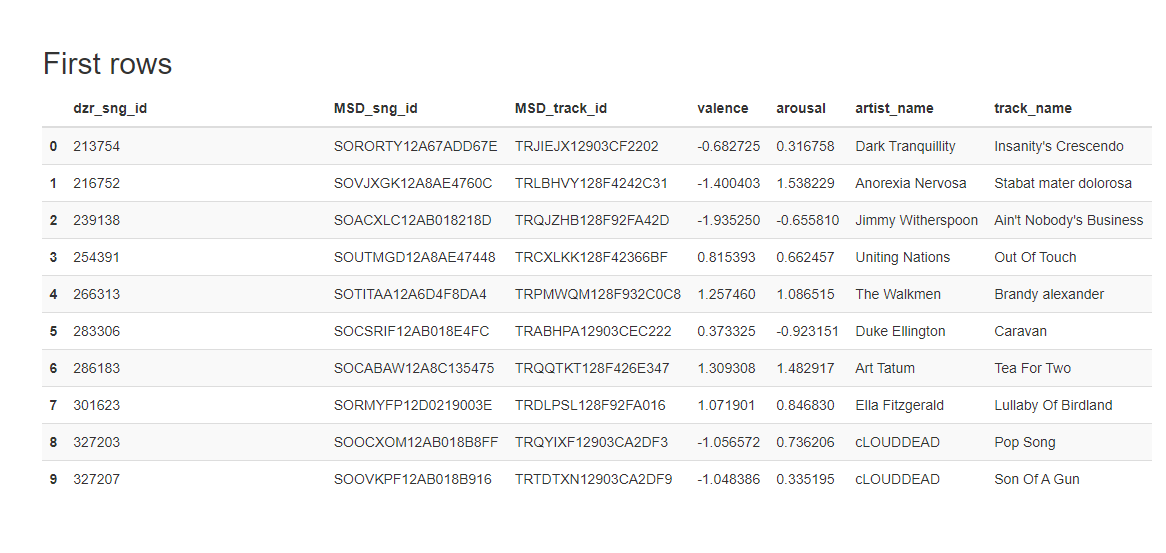
# CHAPTER 3 – METHODOLOGY

This section contains details on data collection and analysis, models architecture design decisions, training and evaluation, and model deployment.

# 3.1 DATA COLLECTION AND ANALYSIS

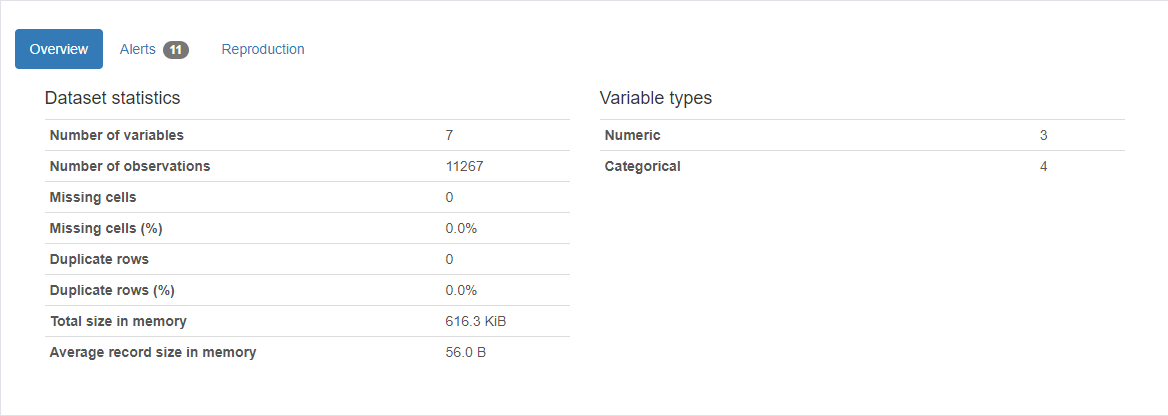
The objectives of this sub-section include; collection of lyrics, collection of song audio, setting the Valence and Arousal values into their respective Quadrants, described in the previous chapter, as the label column of our data, and finally cleaning the data.

The dataset selected for this project was the Deezer Music Emotion Recognition (MER) Dataset which was mentioned in [2] and was publicly available in the GitHub repository (<https://github.com/deezer/deezer_mood_detection_dataset.git>). The dataset came already split into training, validation, and testing sets, with data points 11000, 3000, and 3000 respectively. The dataset contained the following columns; dzr\_sng\_id, MSD\_sng\_id, MSD\_track\_id, valence, arousal, artist\_name, track\_name. A sample header of the dataset is as in the figure below;



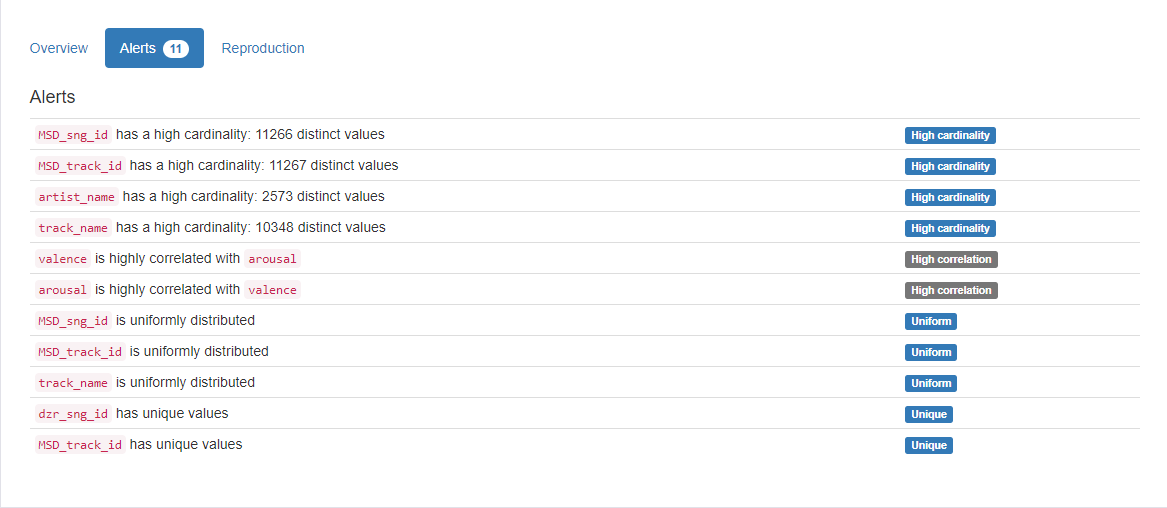
**Figure 3.0 Sample Dataset Header**

An overview of the data is shown in the image below. We get to see that there were no duplicates or missing data. The necessary numeric types included dzr\_sng\_id, valence, arousal, and categorical were artist\_name, and track\_name fields.



**Figure 3.1 Sample Dataset Statistics**

Some alerts raised by the profiler were as in the image below. Some fields such as artist\_name, track\_name, and lyrics had high cardinality, which was expected as having uncommon datapoint values for these was rather expected. It is also notable to mention the high correlation between arousal and valence. This shows that using the two to train the model would yield much as the quadrant which will be a direct derivation of the two, as shown in the quadrant table in chapter 2.



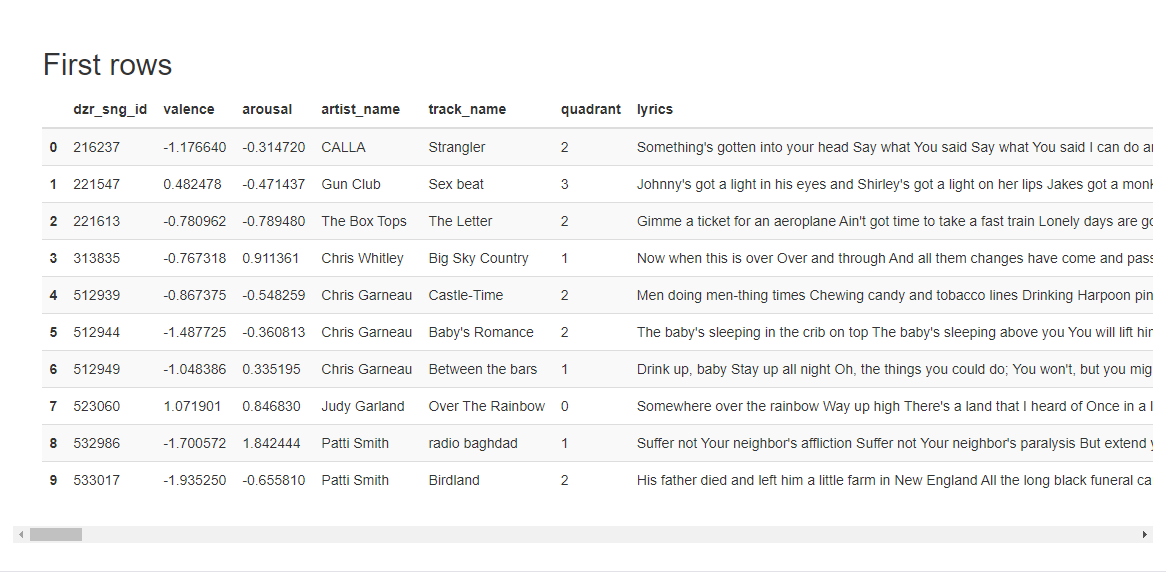
**Figure 3.2 Variable Related Alerts**

## 3.1.1 COLLECTION OF MUSIC LYRICS

In this particular task, I needed to add a lyrics column to the dataset. The API selected for this task was the MusixMatch API, more specifically the lyrics endpoint. More details about the API can be found by following the following link to the API’s documentation <https://developer.musixmatch.com/documentation/api-reference/matcher-lyrics-get>.

Since I could not hit the endpoint and directly add lyrics to our dataset, I build a lyrics\_service.py program that could use query the endpoint using the track\_name and return the collected lyrics. These lyrics were then added to a new data feature/column named lyrics using the Numpy library.

A Sample dataset containing the lyrics feature/column is as in the figure below;



**Figure 3.3 Sample Training Dataset with Lyrics Feature**

Since I was using the basic plan version of the MusixMatch API, I could only obtain 30% of a single song’s lyrics, which may be sufficient for the project.

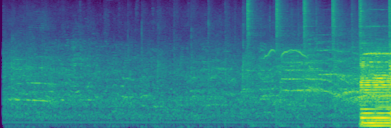
## 3.1.2 COLLECTION OF SONG AUDIO

Originally, the plan I had for this sub-section was to use the Million Song Dataset (MSD) (<https://millionsongdataset.com/>) to get the song’s audio due to the Deezer MER Dataset including MSD Track Ids for each datapoint. I although, ran into a problem of extracting the compressed audios from their HD5 format. Documentation from the MSD website included instructions in a wrapper Python file that I could have used, although it proved to be a challenge.

I settled for building my program or service for downloading song audios. The workflow of the service included; being able to search for a song using the song’s track\_name and artist\_name, downloading the music in .MP3 format, and finally converting the audio into a mel-spectrogram, a format used for audio classification in Deep Learning. The mel-spectrogram conversion was necessary since my audio model will be treating every song audio as an image. Libraries used included; YouTube-Search library responsible for finding a song’s YouTube link, the YouTube-DL library for downloading the songs using the located link, and torchaudio (PyTorch’s audio processing library) and librosa for creating and saving the mel-spectrograms.

***The number of samples to get from an audio signal helped with cutting down or right padding the audio samples to a 30% length. I can use the average sampling rate of the songs in the dataset as the base, then get the 30% from it.***

Mel-spectrograms were used instead of regular spectrogram since the former is used in applications where we need to model human hearing perception, and audio classification applications, which is necessary for this project. Sample mel-spectrogram is as shown in the figure below;



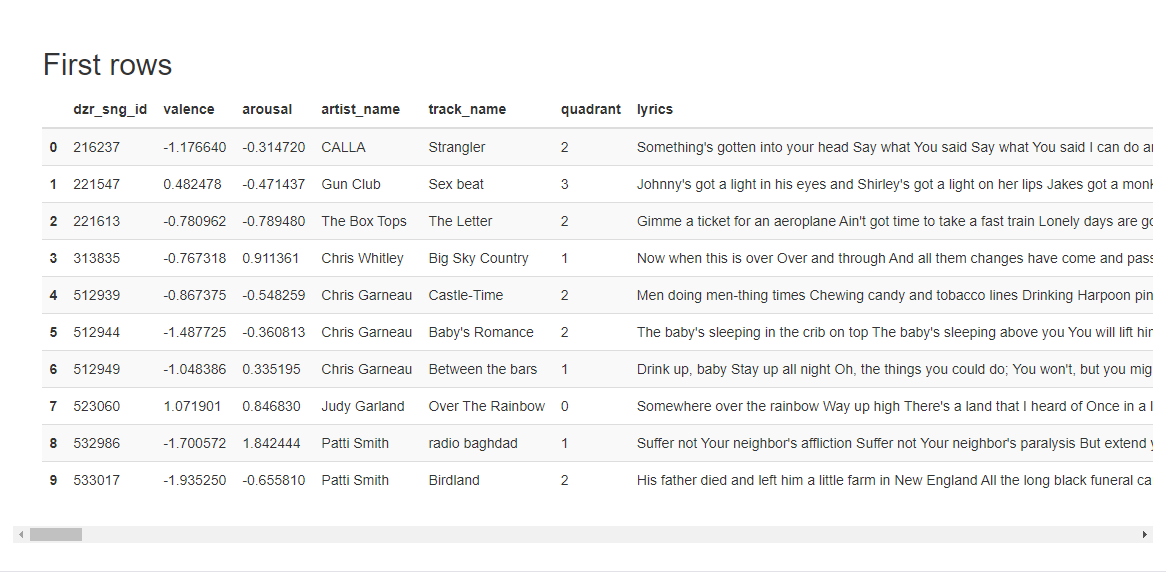
**Figure 3.4 Sample Mel-Spectrogram**

Due to copyright reasons, all song audio files are deleted after their spectrograms have been saved. The spectrograms were all saved using their dzr\_sng\_id value.

## 3.1.3 QUADRANTS AS LABELS

As shown in **Figure 3.0,** the dataset contains valence and arousal features/columns. These values can either be positive or negative values to match the Quadrant divisions when combined, as described in the previous chapter 2 under the **Quadrant Table**.

I created a new data label feature/column called quadrant, which contained integer-values ***i*** of range ***0* *to 3*** representing the ***Qi*** a data point belongs to, such that ***Qi = 0*** *==* ***Q1***… ***Qi=3*** *==* ***Q4.*** All this was done using a function utilizing the Pandas libraryand sample results is as in the figure below;



**Figure 3.5 Sample Dataset with Quadrant Feature**

## 3.1.4 CLEANING DATA

For this process, I deleted all columns that were no longer necessary to train the models, which included; dzr\_sng\_id, MSD\_sng\_id, MSD\_track\_id, valence, arousal, artist\_name, and track\_name. Other data points containing missing or Nan values were also dropped.

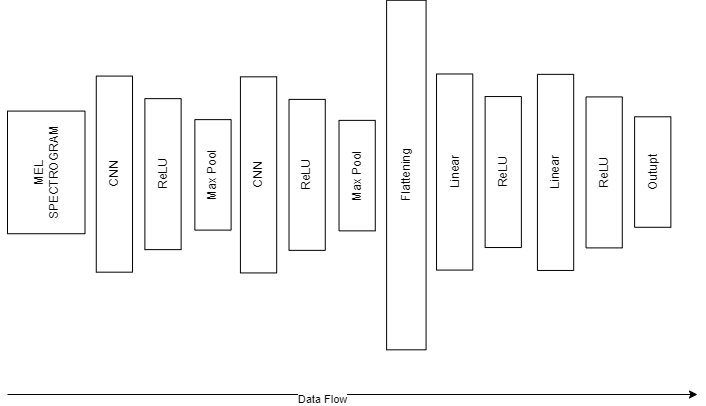
In the lyrics feature column, the lyrics datapoints contained the text **“*\*\*\*\*\*\*\* This Lyrics is NOT for Commercial use \*\*\*\*\*\*\* (1409621065547)*”,** at the end of every song lyrics, which I considered as unnecessary information that might lead to some skewness in the data, so I trimmed it.

# 3.2 MODEL ARCHITECTURES, TRAINING, AND EVALUATION

As described in the paper [2], there are two ways in which Music Emotion Recognition can be accomplished using Deep Learning, one is having two models, one for lyrics and the other for audio-only, and two having a combined version of the two. They described that having the combined version seemed to have a better performance than separate models. For my algorithm, I decided to use the two models' approach as it would help in faster debugging and development, and also help in improving the results of the models described in the paper by aggregating the outputs of the two models. This also acts as a form of regularization of the generated results.

## 3.2.1 AUDIO MODEL

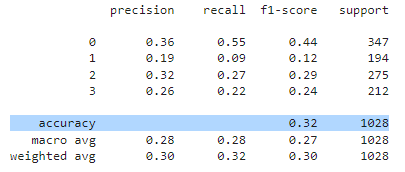
The audio model is Convolution Neural Network (CNN) based model. The architecture is as shown in the figure below. The reason behind this architecture was mostly resource-based, having a deeper model led to some crashes later in the training step.



**Figure 3.6 Audio Model Architecture**

The pipeline followed for this model is as below;

1. Preprocessing. This step involves converting the images to Tensor format. This is necessary as PyTorch can only process Tensor data.
2. Hyperparameter selection. The loss function used was the Cross-Entropy Loss, and the optimization function selected was the Stochastic Gradient Descent (SDG) function, as they perform really well for multi-class classification tasks.
3. Training. The model was trained over 25 epochs and only the loss metric was being monitored during this phase.
4. Evaluation. In this phase, the model’s performance on the test set was calculated, and the classification report was as shown in the figure below;



**Figure 3.7 Audio Model Classification Report**

## 3.2.2 LYRICS MODELS

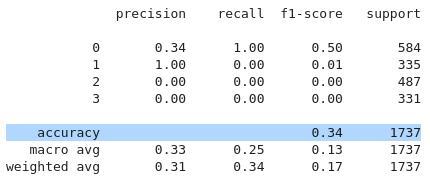
In this subsection, I’ll be comparing the performance of the fastText model, LSTM+GloVe model, and BERT model. The pipeline followed for all models is as below;

1. Preprocessing. This step involves preprocessing text to remove special characters, numbers, spaces, and stopwords such as “and”/ “or”. This step is similar to all models.
2. Tokenization. This step involves converting the text to a format that can be used by the models such as representing each word as a number. For fastText no tokenization was necessary. The LSTM+GloVe model used the Spacy library to tokenize the lyrics, and for the BERT model the entire corpus of words was tokenized or encoded using transformers’ BertTokenizer which is based on WordPiece, a subword segmentation algorithm used in natural language processing where the vocabulary is initialized with individual characters in the language, then the most frequent combinations of symbols in the vocabulary are iteratively added to the vocabulary.
3. Training. During training, the optimization function selected was Adam similar to the audio model, and the loss metric was used to measure the model’s performance. Weighted F1 score was also used for the BERT, as it tells you how precise your model is (how many instances it classifies correctly), as well as how robust it is (it does not miss a significant number of instances). Since automatic hyperparameter tuning was used for the fastText model, no optimization function or loss metric was specified.
4. Evaluation. All model evaluation was similar, where the precision and recall of the models were calculated. Precision refers to the number of correct labels among the labels predicted, while recall refers to the number of labels that were successfully were predicted, among all the real labels.

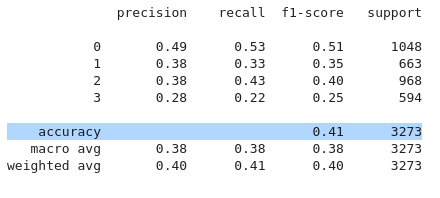
A comparison of the model classification reports are as in the figures that follow;



**Figure 3.8 Classification Report for fastText model. (Dataset Size, Precision, Recall)**



**Figure 3.9 Classification Report for LSTM+GloVe model.**



**Figure 3.10 Classification Report for BERT model.**

Accuracy was calculated although not used as the main measure of model performance since there is an existence of class imbalance in the dataset, thus doing having recall and precision will help in identifying the model’s performance on individual classes.

It is evident that the BERT model had the highest Precision and Recall averages, hence it was selected as the final model. No model was able to reach an average accuracy greater than 42%, although there were some instances where the individual accuracy for classes 0 and 2, were higher than 50% which was expected because;

1. Those two classes represent music which could be classified as Happy and Sad respectively, which were very high in the dataset. There was class imbalance.
2. The total dataset size was very small, hence could not obtain very high accuracies. More data should be able to help solve this.

The above conclusion also applies to the Audio Model.

# 3.3 SOFTWARE REQUIREMENTS

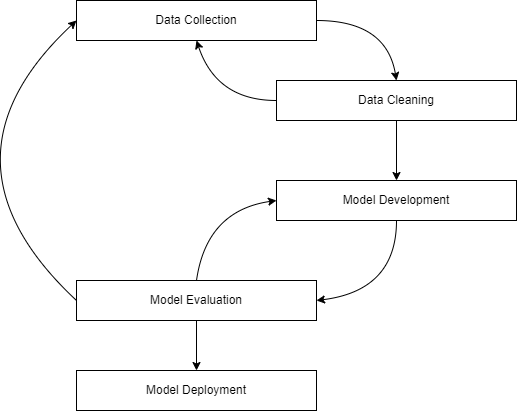
* Ubuntu 20.04 LTS
* Python 3.9
* PyTorch 1.9
* HuggingFace Transformers 4.9.1
* Scikit Learn 1.0.1
* Flask 2.0.1
* Numpy 1.20.3
* Pandas (any version)
* SQLite 3

# 3.4 HARDWARE REQUIREMENTS

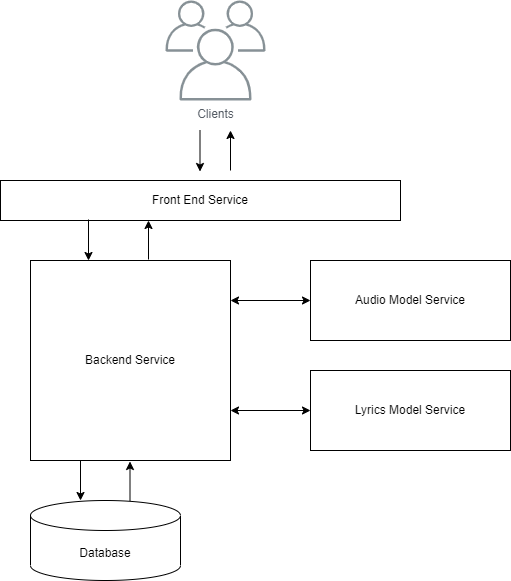
* NVIDIA 1660 GPU, version 470 CUDA 11.0, RAM 6 GB
* AMD Ryzen 3600 7 16 Cores
* 16 GB DDR3 RAM

# CHAPTER 4 – PROJECT PRESENTATION

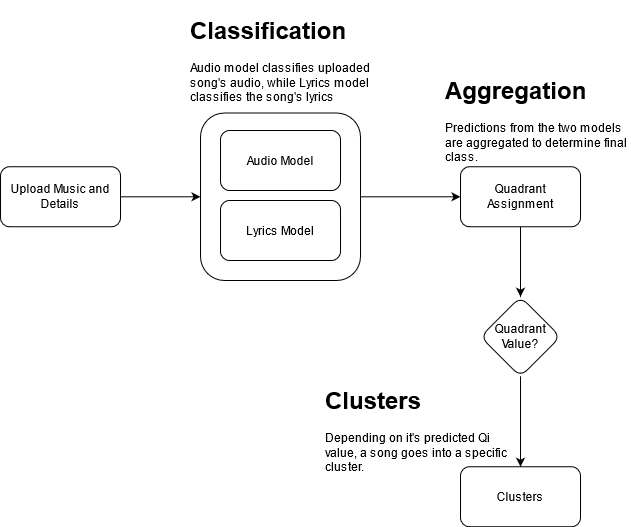
# 4.1 DATA FLOW DIAGRAMS (DFDs)



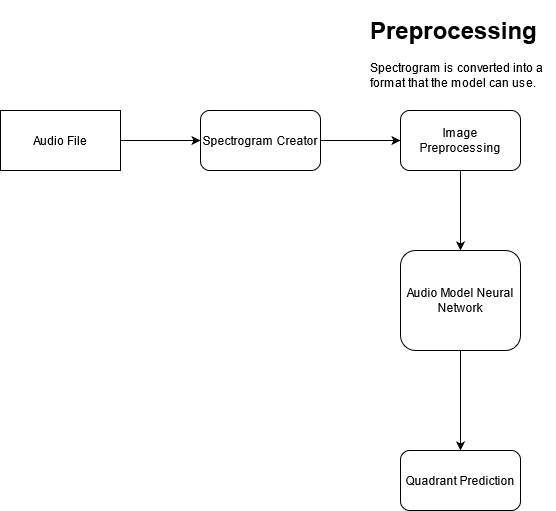
**Figure 4.0 Project Development Cycle AI Models Diagram**



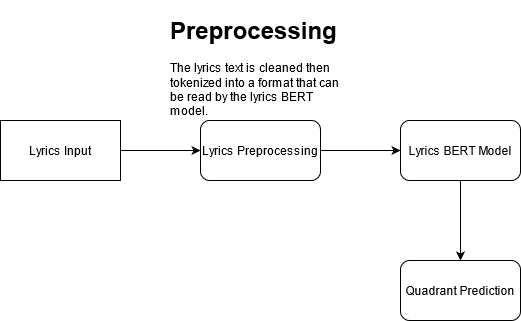
**Figure 4.1 Web Application Architecture**



**Figure 4.2 Clustering Pipeline**

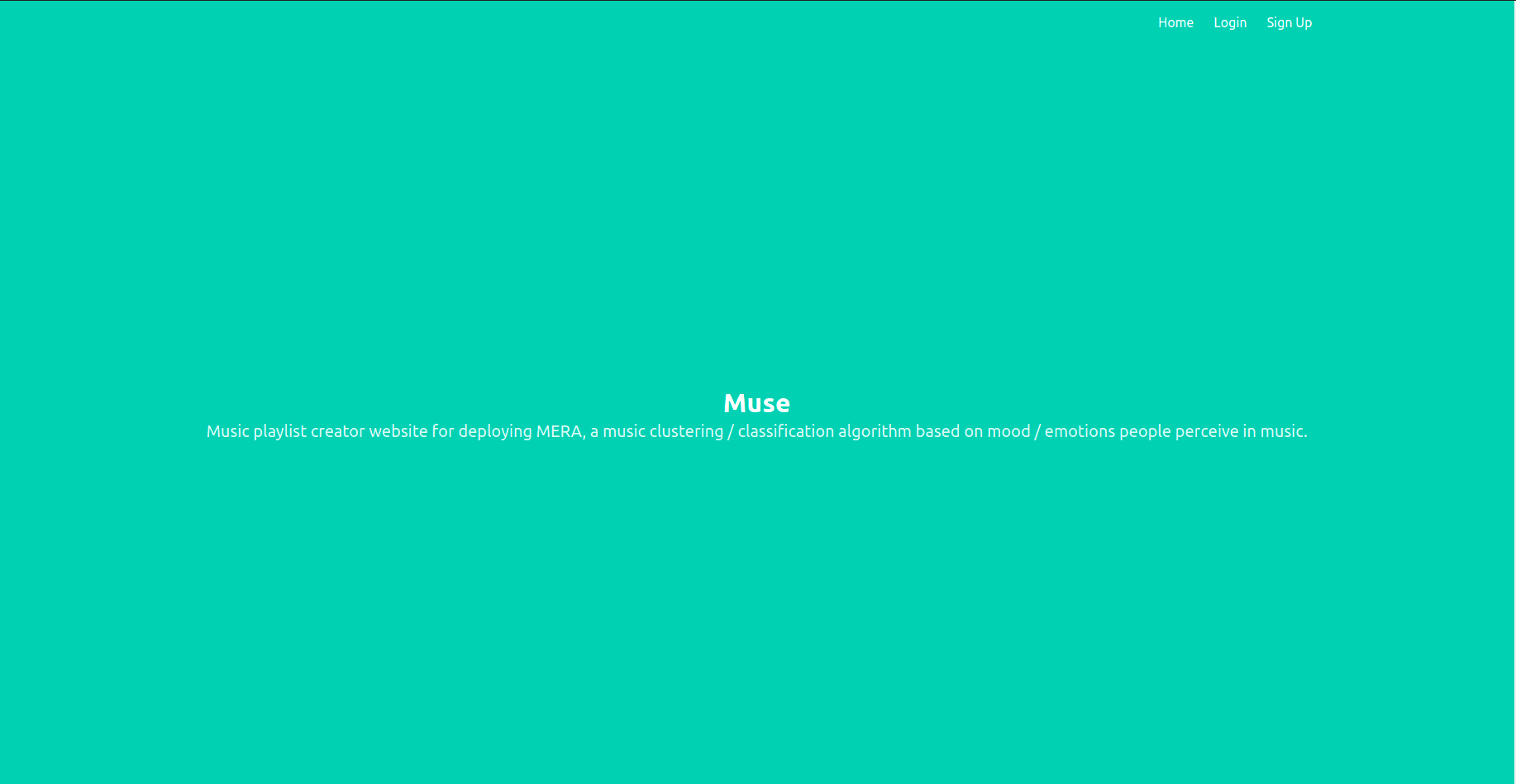


**Figure 4.3 Audio Model Pipeline**

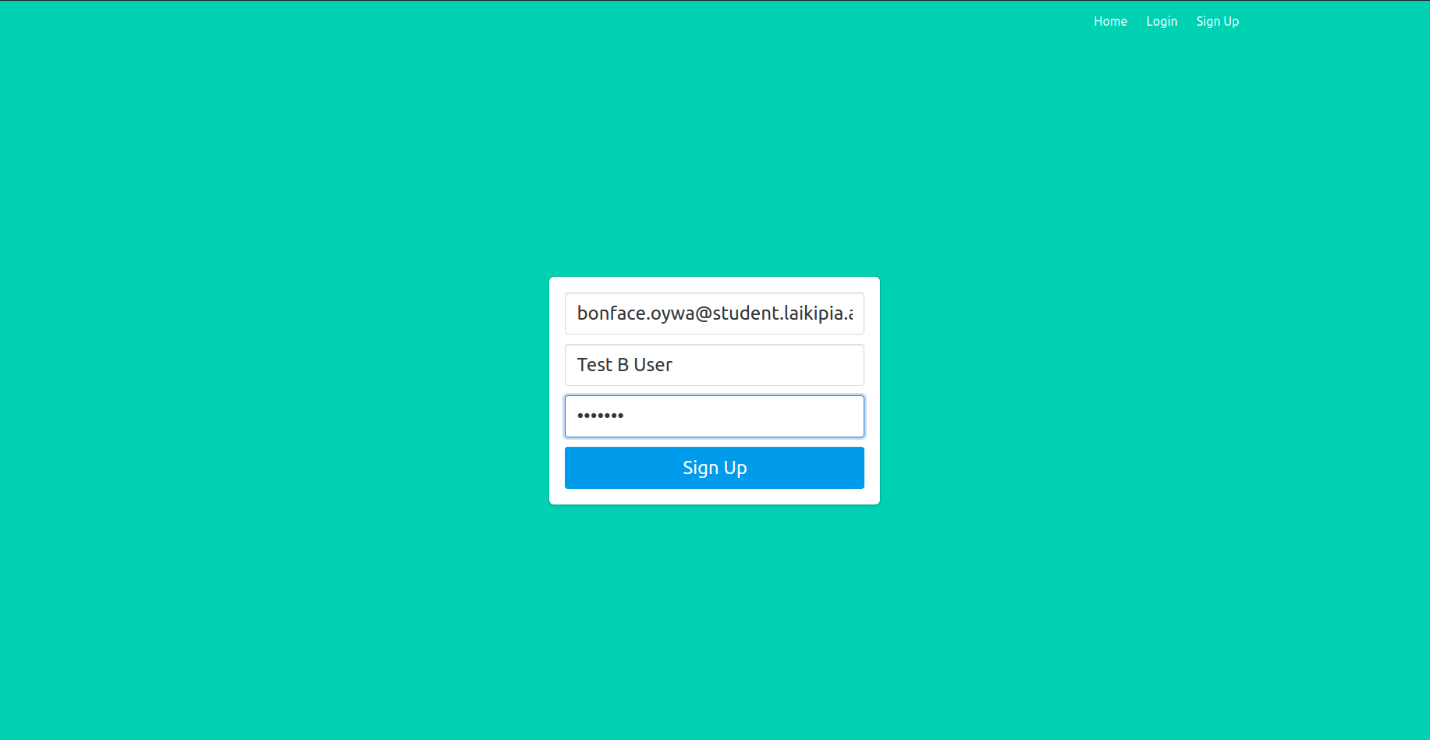


**Figure 4.4 Lyrics Model Pipeline**

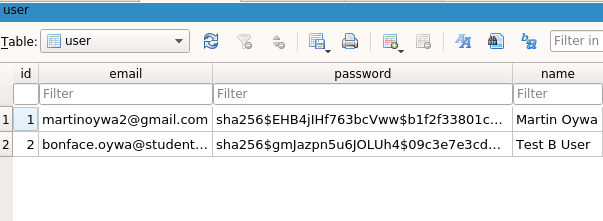
# 4.2 PROJECT SCREENSHOTS



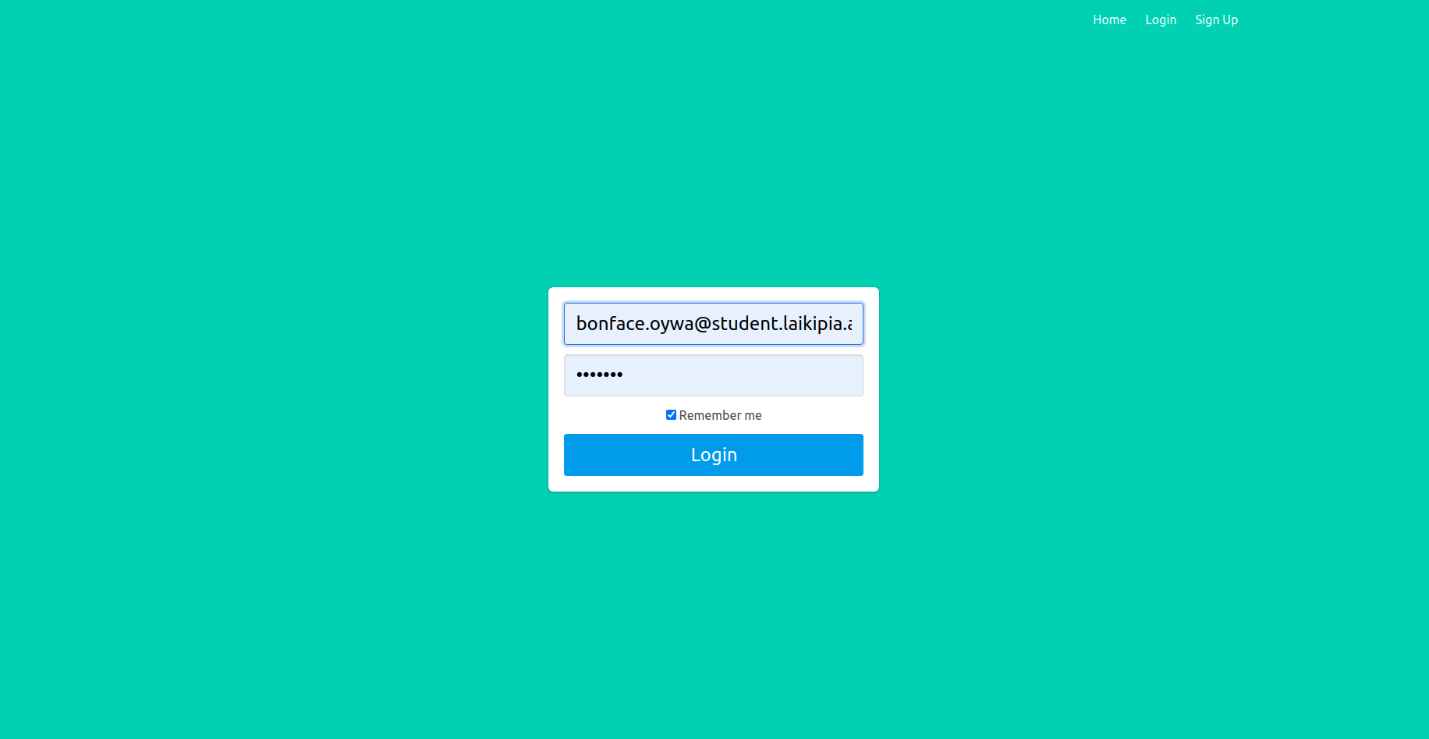
**Figure 4.5 Web Application Home Page**



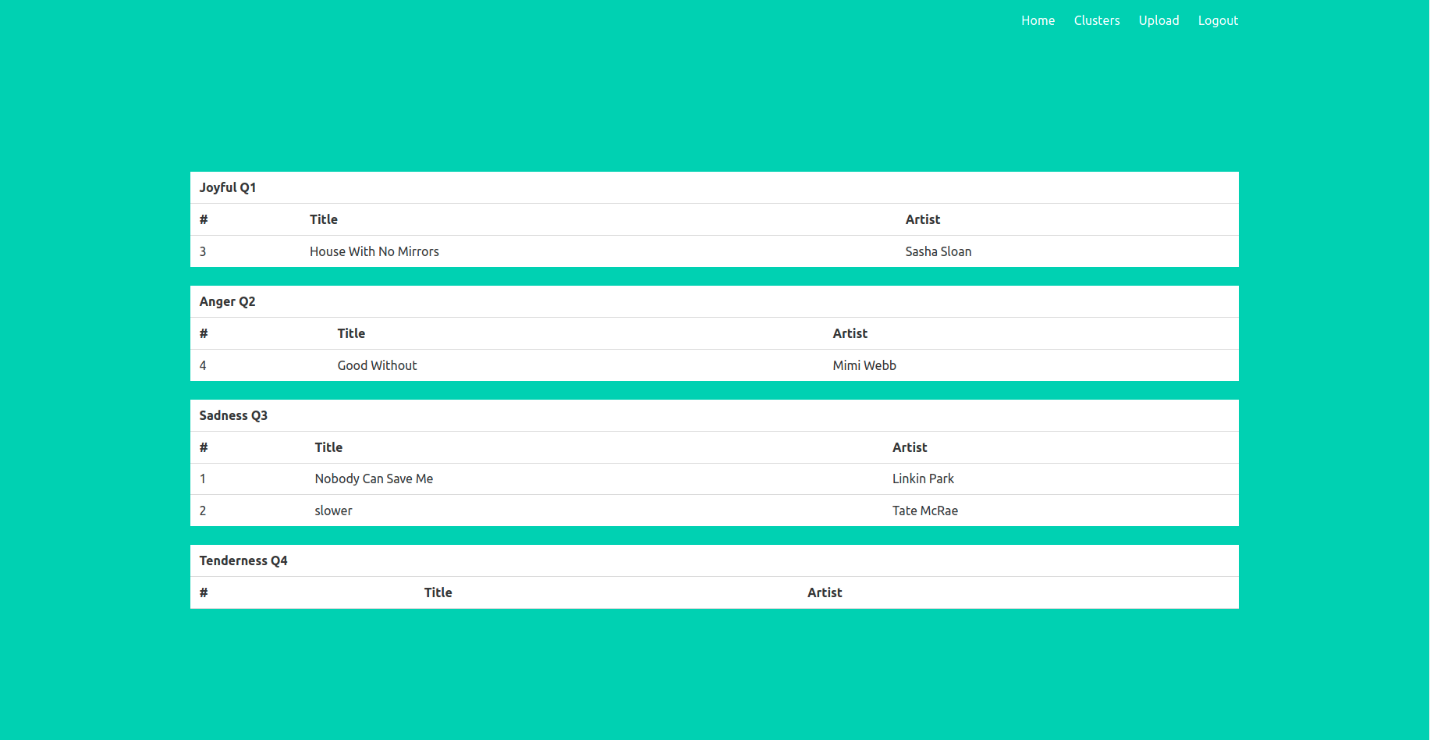
**Figure 4.6 Web Application Sign Up Page**



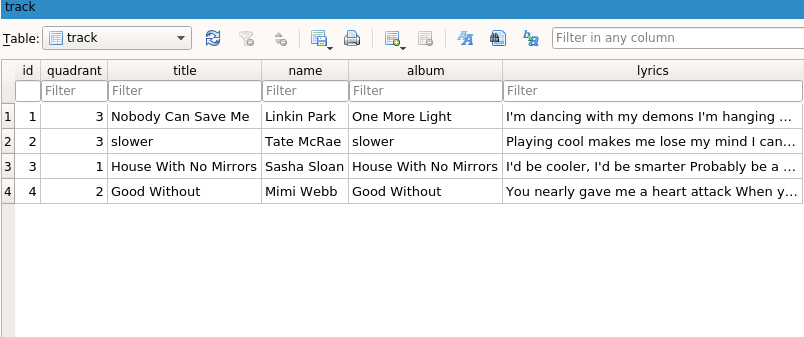
**Figure 4.7 User Database Table**



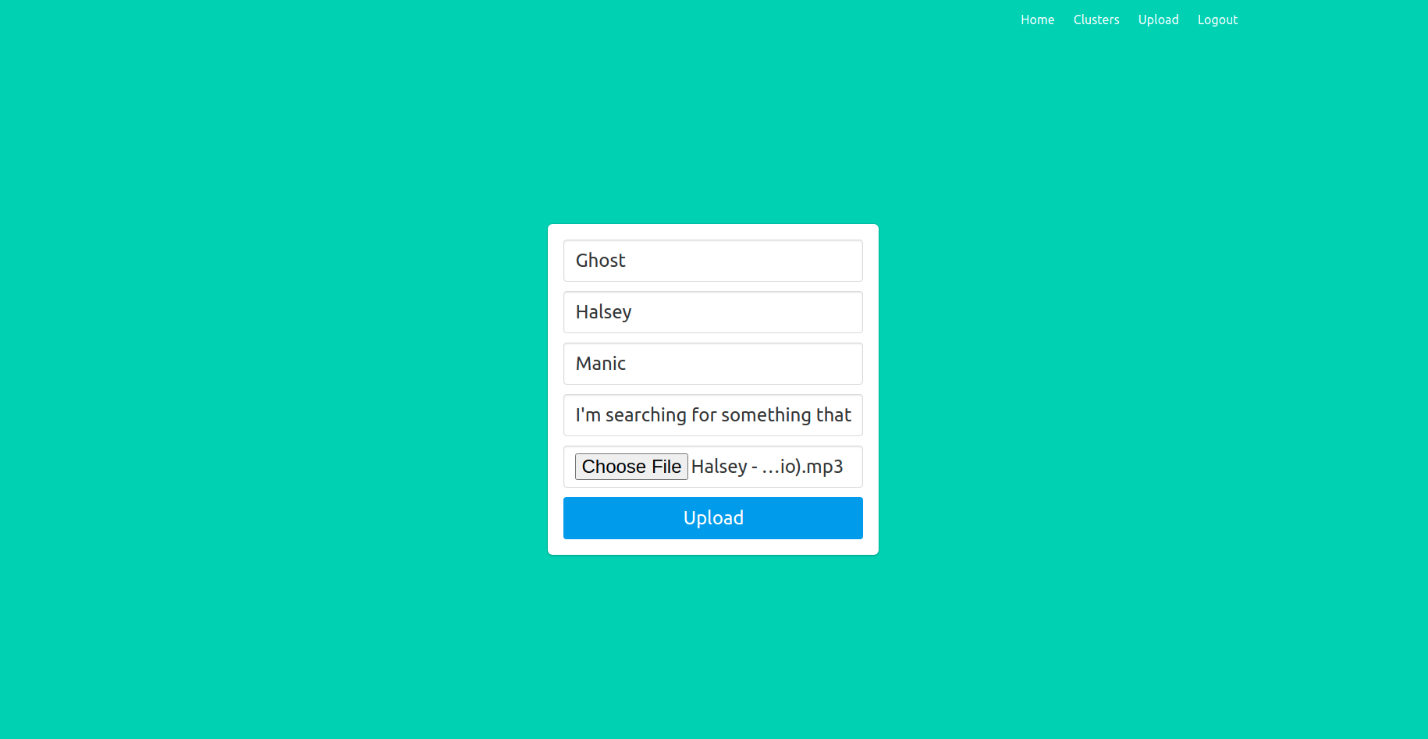
**Figure 4.8 Web Application Log In Page**



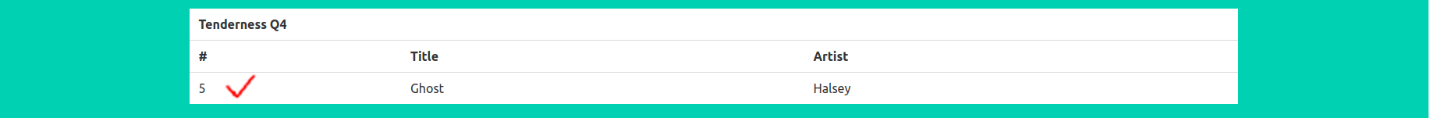
**Figure 4.9 Web Application Clusters Page**



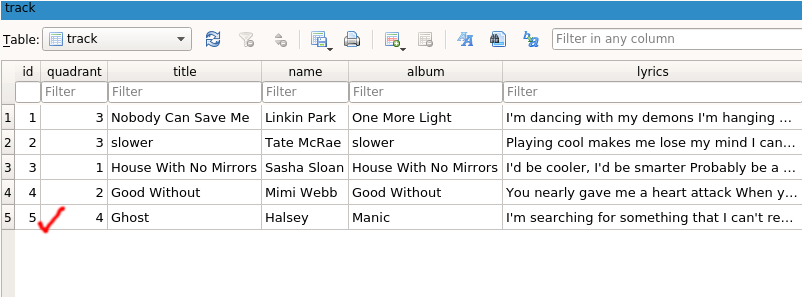
**Figure 4.10 Tracks Database Table**



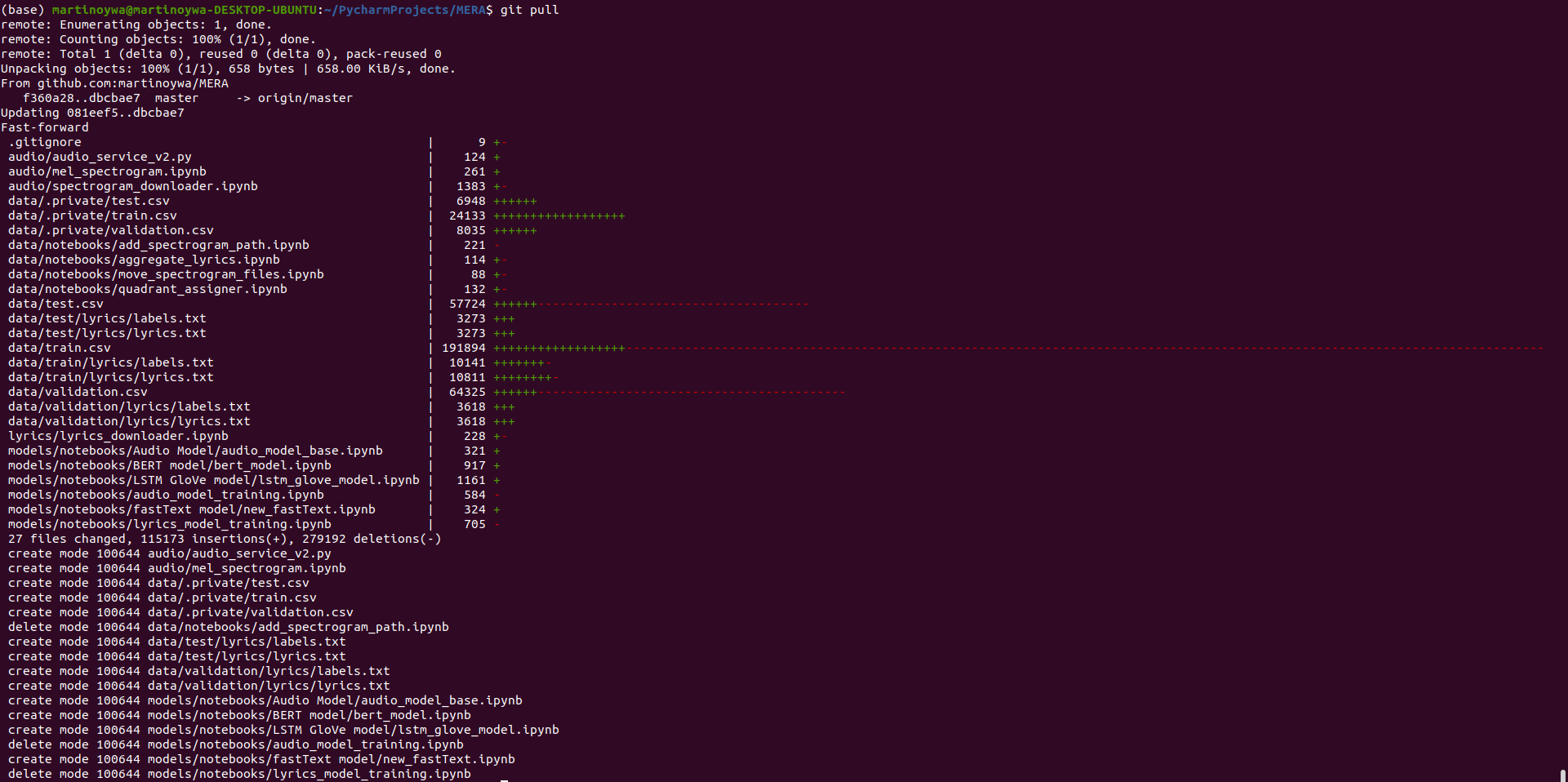
**Figure 4.11 Web Application Upload Page**



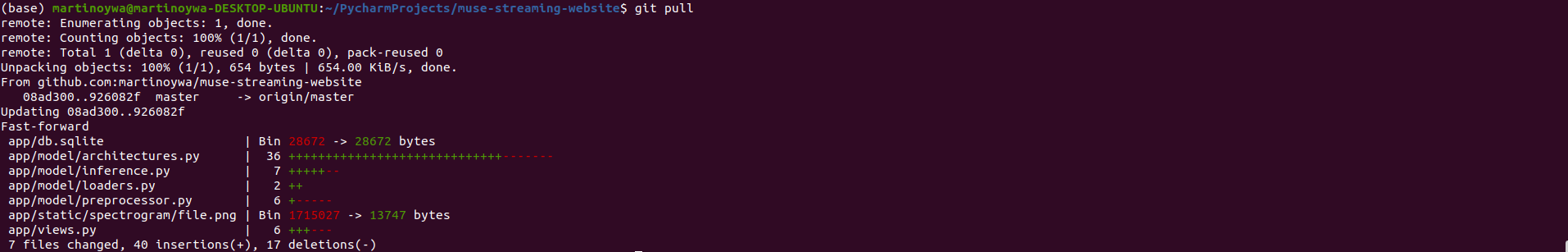
**Figure 4.12 Web Application New Track Details**



**Figure 4.13 Web Application New Track Details in Tracks Database Table**



**Figure 4.1 MERA Repository Code Structure**



**Figure 4.1 Muse Web Application Code Structure**

All project code can be found in the following GitHub repositories;

1. MERA: <https://github.com/martinoywa/MERA>
2. Muse Web Application: <https://github.com/martinoywa/muse-web-app>

# CHAPTER 5 – RECOMMENDATIONS AND CONCLUSION

The MERA algorithm presented in this project can be applied in existing recommender systems whereby the pool of music clustered into the respective quadrants can be used by the systems to decide on which music should they recommend. This means that the whole aspect of this algorithm is to add functionality on top of existing recommendation techniques. An example would be; In Spotify, all music and their details are considered documents, this is a pool. The MERA algorithm can then create clusters of these documents, where techniques such as collaborative filtering can then be used to recommend music to a user. This is necessary since not all happy songs, for instance, are pleasing to all users.

It is evident that the MERA algorithm still has a long way to go in terms of improving its precision and recall values for both mel-spectrogram and lyrics classification. More data may need to be collected to aid in reducing the margin of class imbalance in the datasets, but there may be more research needed especially on the research limitation in the first chapter, to achieve better results in the future.

# REFERENCES

|  |  |
| --- | --- |
| [1] | P. H. E. G. E. C. Juan Sebastián Gómez Cañón, The emotions that we perceive in music: the influence of language and lyrics comprehension on agreement, 2019. |
| [2] | R. H. F. P. J. R.-L. M. M. Rémi Delbouys, Music Mood Detection Based On Audio And Lyrics With Deep Neural Net, 2018. |
| [3] | M.-W. C. K. L. K. T. Jacob Devlin, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018. |