**A Project report on**

# Song Recommender Chat Bot

### A Dissertation submitted to JNTU Hyderabad in partial fulfillment of the academic requirements for the award of the degree.

**Bachelor of Technology**

**IN**

**Computer Science and Engineering**

**Submitted by**

(Student Name) (Roll No)

#### P.ANUSHA 20H51A05J3 N.SAI ADITYA VARDHAN 20H51A05J5 S.BHAGYASHREE 20H51A05J7

Under the esteemed guidance of

### Dr. S. Siva Skandha

**HOD, DEPT. OF CSE**



**Department of Computer Science and Engineering**

# CMR COLLEGE OF ENGINEERING AND TECHNOLOGY

(An Autonomous Institution, Approved by AICTE, Affiliated to JNTUH, NAAC ‘A+’)

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD - 501401.

2020- 2024



**CMR COLLEGE OF ENGINEERING & TECHNOLOGY**

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD – 501401

##### DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



CERTIFICATE

This is to certify that the Mini Project-1 report entitled **"SONG RECOMMENDER CHAT BOT"** being submitted by **P.ANUSHA (20H51A05J3), N.SAI ADITYA VARDHAN (20H51A05J5), S.BHAGYASHREE (20H51A05J7)**

in partial fulfillment for the award of **Bachelor of Technology in Computer Science and Engineering** is a record of bonafide work carried out his/her under my guidance and supervision.

The results embodies in this project report have not been submitted to any other University or Institute for the award of any Degree.

**Dr. S. Siva Skandha Associate Professor and HOD Dept. of CSE**

## ACKNOWLEDGEMENT

With great pleasure We want to take this opportunity to express my heartfelt gratitude to all the people who helped in making this project work a grand success.

We are grateful to **Dr. S. Siva Skandha**, Associate Professor and Head of the Department of Computer Science and Engineering for his valuable technical suggestions and guidance during the execution of this project work.

We are very grateful to **Dr. Vijaya Kumar Koppula**, Dean-Academic, CMR College of Engineering and Technology, for his constant support and motivation in carrying out the project work successfully.

We are highly indebted to **Dr. V A Narayana,** Principal, CMR College of Engineering and Technology, for giving permission to carry out this project in a successful and fruitful way.

We would like to thank the Teaching & Non- teaching staff of Department of Computer Science and Engineering for their co-operation

Finally, I express my sincere thanks to **Mr. Ch. Gopal Reddy**, Secretary, CMR Group of Institutions, for his continuous care. I sincerely acknowledge and thank all those who gave support directly and indirectly in completion of this project work.

**P.ANUSHA (20H51A05J3) N.SAI ADITYA VARDHAN (20H51A05J5) S.BHAGYASHREE (20H51A05J7)**



## DECLARATION

We hereby declare that results embodied in this Report of Project on **“SONG RECOMMENDER CHAT BOT”** are from work carried out by using partial fulfillment of the requirements for the award of B. Tech degree. We have not submitted this report to any other university/institute for the award of any other degree.

## NAME SIGNATURE

P.ANUSHA (20H51A05J3)

N.SAI ADITYA VARDHAN (20H51A05J5)

S.BHAGYASHREE (20H51A05J7)

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER**  **NO.** | **TITILES** | **PAGE NO.** |
|  | **Abstract** | 7 |
| **1** | **Introduction** | 8 |
| **2** | **Background Work** | 10 |
|  | 2.1 Existing system | 15 |
| **3** | **Proposed System** | 16 |
|  | 3.1 Proposed solution | 16 |
|  | 3.2 System Design | 16 |
|  | 3.3 Modules in system | 18 |
| **4** | **Designing** | 22 |
|  | 4.1 Construction of Use Case Diagram | 22 |
|  | 4.2 Sequence Diagram | 23 |
|  | 4.3 Class Diagram | 24 |
|  | 4.4 Activity Diagram | 25 |
| **5** | **Result and Discussion** | 28 |
| **6** | **Conclusion & Future Scope** | 31 |
| **7** | **References** | 32 |

**FIGURES**

|  |  |  |
| --- | --- | --- |
| **CHAPTERS NO.** | **TITLES** | **PAGE NO** |
| 2 | 2.1 DIAGRAM | 14 |
| 3 | 3.2.1 DIAGRAM | 17 |
|  | 3.4.1 DIAGRAM | 19 |
|  | 3.4.2 DIAGRAM | 21 |
| 4 | 4.1 DIAGRAM | 22 |
|  | 4.2 DIAGRAM | 23 |
|  | 4.3 DIAGRAM | 24 |
|  | 4.4 DIAGRAM | 25 |
|  | 4.3 DIAGRAM | 26 |
| 5 | 5.1 SCREENSHOT OF EXECUTION | 28 |
|  | 5.2 SCREENSHOT OF EXECUTION | 28 |
|  | 5.3 SCREENSHOT OF EXECUTION | 29 |
|  | 5.4 SCREENSHOT OF DATABASE | 29 |

**ABSTRACT**

Technology has a great impact on every part of lives, which also includes our day-to-day habits. In present scenario, fields like artificial intelligence and machine learning are on great boom. With the help of advancement in these fields, large number of people are interacting through the system via chat bots and voice assistants. In this era of technological advancements, music recommendation based on mood is much needed at it will help humans relieve stress and listen to soothing music according to their mood. Considering above factors, this project is aimed to implement the Chatbot Song Recommender System that includes chatbot to assist user and recommend songs. The chatbot identifies the user's sentiment by asking some general questions. Based on the input provided by the user, the chatbot will generate some list of songs. The proposed recommendation system utilizes the APIs for the playlist generation and recommendation.

## CHAPTER 1 INTRODUCTION

Now-a-days, we all are living in the time where we know that nothing is certain. Same goes with our mind, at regular instances of time our mood, our choices and our priorities changes. Considering the constant changing behavior of human being we have developed our system. We have made our system considering that the humans experience frequently changes in their mood and somehow, at particular moment of time, frequently changing of mood would also result in change in mood of music of their choice. Hence, with the help of our system you can listen music according to your mood. In addition, we have also provided the facility to user to chat with the chat bot after all texting makes conversation between chat bot and user more interactive and it will efficiently help in analyzing the current mood of the user and based on that chat bot will recommend songs. Besides these, our system comprises of four modules. Initially for a new user we have provided a sign-up page. After completing sign-up , user can login to our system. In addition to our system, we’ve also provided the password recovery facility to user, in case user forgot the password. After chatting with the chat bot, current mood of the user is analyzed and list of songs is suggested to user, using the concept of NLP (Natural Language Processing). Based on the list of songs user can choose the song to be played based on his or her choice. We have used Python as our prime language because it supports an extensive set of open-source libraries which can be used by our system.

Every day, each and every person undergoes lot of troubles and the reliever of all the stress that are encountered is Music. If it is so, the vital part of hearing the song has to be in a facilitated way, that is player able to play the song in accordance to the person’s mood. The paper proposes such a player and hence named Emotion based music player. Recognizing the human emotions is considered to be a global consistency but depict variability among the humans on the basis of their abilities.

The different approaches being the classification of emotions that are existing are depicted as knowledge-based, statistical and hybrid techniques. However there relies several difficulties while retrieving the music information such as querying by singing, genre classification, etc. The most possible implementation is by producing music suggestion that is based on the content. The multiple efforts to detect and describe the feeling, to classify based on features are all wanted to obtain outstanding music recommendation system. For describing a music taxonomy, a feeling descriptor is found to be useful.

Problem, Identification & Objective:

Emotion detection is an important process in our project which requires accuracy and this can be done eﬀectively with the help of facial expressions which is how humans understand and interpret an emotion. Humans are capable of perceiving emotions which is exceedingly important for a communication to be a success and hence in a typical conversation almost 93% of communication depends on the emotion being expressed.

Chat-bots have 24/7 Availability: Chat-bots are available to solve customer problems 24/7 whether it is day or night! They don’t need to sleep after all! This is much more difficult to achieve using human customer service as it would require rotating teams that would be more complicated to manage as well. This means that chat-bots can answer customer queries whenever customers have queries which help in increasing customer loyalty. If companies don’t address customer problems as soon as possible, then the customers may switch to another company that provides better customer service.

In this project, we would be combining multiple services and open-source tools to make a Chatbot that recommends songs based on the tone of the conversation which the user is having with the chatbot.

# CHAPTER 2 BACKGROUND WORK

###### Python Technology

Python technology is both a programming language and a platform.

###### The Python Programming Language

THE PYTHON PROGRAMMING LANGUAGE IS A HIGH-LEVEL LANGUAGE THAT CAN BE CHARACTERISED BY ALL OF THE FOLLOWING BUZZWORDS:

* Simple
* Architecture neutral
* Object oriented
* Portable
* Distributed
* High performance
* Interpreted
* Multithreaded
* Robust
* Dynamic
* Secure

###### Python

Python is high level language and it is also integrated version of the program. Python is an object-oriented approach and its main aim to help programmers to write the code clearly, logical code for small and large scale of project.

* Python is simple, object-oriented programming language.
* The language and implementation should provide support for software engineering principles such as strong type library preset for different machine learning algorithm, and all other algorithm in simple manner.
* Coding will be smooth in python and the data analysis can be easily done in python.

###### Anaconda:

Anaconda is free and open-source distribution of the Python and R programming languages for scientific computing (data science, machine Learning applications, Large- scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. It is developed and maintained by Anaconda, Inc. The distribution incudes data-science packages suitable for Windows, Linux, and macOS. Packaged versions are required and are managed by the package management system anaconda. This package manager was spun out as a separate open-source package as it ended up being useful on its own and for other things than Python. There is also a small, bootstrap version of Anaconda called Miniconda, which includes only conda, Python, the packages they depends on, and a small number of other packages.

###### import numpy as np:

* + NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.
  + At the core of the NumPy package, is the nd array object. This encapsulates n-dimensional arrays of homogeneous data types, with many operations being performed in compiled code for performance. There are several important differences between NumPy arrays and the standard Python sequences:
    - NumPy arrays have a fixed size at creation, unlike Python lists (which can grow dynamically). Changing the size of a ndarray will create a new array and delete the original.
    - The elements in a NumPy array are all required to be of the same data type, and thus will be the same size in memory. The exception: one can have arrays of (Python, including NumPy) objects, thereby allowing for arrays of different sized elements.
    - NumPy arrays facilitate advanced mathematical and other types of operations on large numbers of data. Typically, such operations are executed more efficiently and with less code than is possible using Python’s built-in sequences.
    - A growing plethora of scientific and mathematical Python-based packages are using NumPy arrays; though these typically support Python-sequence input, they convert such input to NumPy arrays prior to processing, and they often output NumPy arrays. In other words, in order to efficiently use much (perhaps even most) of today’s scientific/mathematical Python-based software, just knowing how to use Python’s built-in sequence types is insufficient - one also needs to know how to use NumPy arrays.

###### import time:

This module provides various time-related functions. For related functionality, see also the date , time and calendar modules.

Although this module is always available, not all functions are available on all platforms. Most of the functions defined in this module call platform C library functions with the same name. It may sometimes be helpful to consult the platform documentation, because the semantics of these functions varies among platforms.

An explanation of some terminology and conventions is in order.

###### import os:

This module provides a portable way of using operating system dependent functionality. If you just want to read or write a file see open(), if you want to manipulate paths, see the os.path module, and if you want to read all the lines in all the files on the command line see the fileinput module. For creating temporary files and directories see the tempfile module, and for high-level file and directory handling see the shutil module.

Notes on the availability of these functions:

* + The design of all built-in operating system dependent modules of Python is such that as long as the same functionality is available, it uses the same interface; for example, the function os.stat(path) returns stat information about path in the same format (which happens to have originated with the POSIX interface).
  + Extensions peculiar to a particular operating system are also available through the os module, but using them is of course a threat to portability.
  + All functions accepting path or file names accept both bytes and string objects, and result in an

object of the same type, if a path or file name is returned.

* + On VxWorks, os.popen, os.fork, os.execv and os.spawn\*p\* are not supported.

Import chatbot:

ChatterBot is a Python library built based on machine learning with an inbuilt conversational dialog flow and training engine. The bot created using this library will get trained automatically with the response it gets from the user.

###### Why Chatbots are important for a Business or a Website

* Quick resolution for a complaint or a problem.
* Improve business branding thereby achieving great customer satisfaction.
* Answering questions and answers for customers.
* Making a reservation at hotel or at restaurant.
* Save human effort 24×7.
* Enhance business revenue by providing ideas and inspirations.
* Finding details about business such as hours of operation, phone number and address.
* Automate sales and lead generation process.
* Reduce customer agents waiting time answering phone calls.

###### Import ListTrainer:

ChatterBot is a Python library that makes it easy to generate automated responses to a user’s input. ChatterBot uses a selection of machine learning algorithms to produce different types of responses. This makes it easy for developers to create chat bots and automate conversations with users.

ChatterBot is a Python library designed to make it easy to create software that can engage in conversation. An untrained instance of ChatterBot starts off with no knowledge of how to communicate. Each time a user enters a statement, the library saves the text that they entered and the text that the statement was in response to. As ChatterBot receives more input the number of responses that it can reply and the accuracy of each response in relation to the input statement increase. The program selects

the closest matching response by searching for the closest matching known statement that matches the input, it then chooses a response from the selection of known responses to that statement.

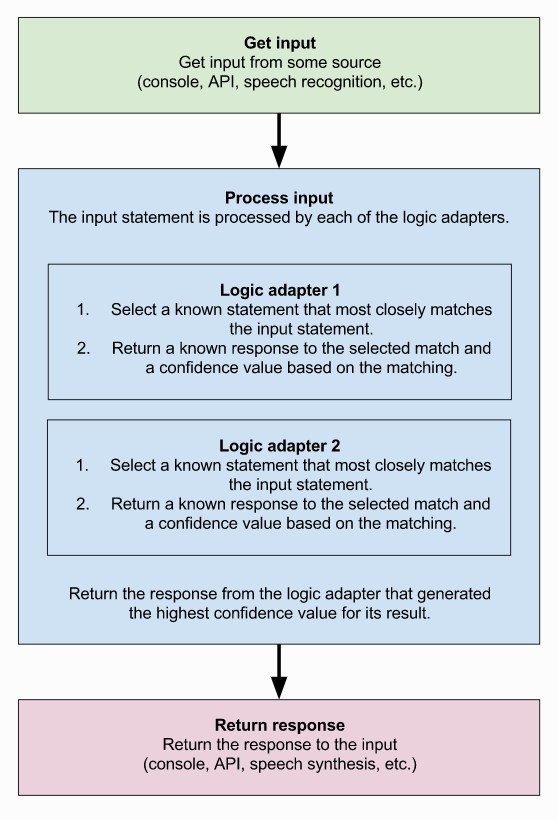


Fig: 2.1 Importing List Trainer

###### Import NLTK:

The Natural Language Toolkit (NLTK) is a platform used for building Python programs that work with human language data for applying in statistical natural language processing (NLP). It contains text processing libraries for tokenization, parsing, classification, stemming, tagging and semantic reasoning. It also includes graphical demonstrations and sample data sets as well as accompanied by a cook book and a book which explains the principles behind the underlying language processing tasks that NLTK supports.

The Natural Language Toolkit is an open-source library for the Python programming language originally written by Steven Bird, Edward Loper and Ewan Klein for use in development and education. It comes with a hands-on guide that introduces topics in computational linguistics as well as

programming fundamentals for Python which makes it suitable for linguists who have no deep

knowledge in programming, engineers and researchers that need to delve into computational linguistics, students and educators.

NLTK includes more than 50 corpora and lexical sources such as the Penn Treebank Corpus, Open Multilingual Wordnet, Problem Report Corpus, and Lin’s Dependency Thesaurus.

## EXISTING SYSTEM:

* Collaborative filtering involves collecting information from many users and then making predictions based on some similarity measures between users and between items. This can be classified into user-based and item-based models.
* In item-based model it is assumed that songs that are often listened together by some users tend

to be similar and are more likely to be listened together in future also by some other user. According to user-based similarity model [Figure2][8], users who have similar listening histories, i.e., have listened to the same songs in the past tend to have similar interests and will probably listen to the same songs in future too.

## DISADVANTAGES OF EXISTING SYSTEM:

* + - * We need some similarity measure to compare between two songs or between two users. Cosine similarity weighs each of the users equally which is usually not the case. User should be weighted less if he has shown interests to many varieties of items (it shows that either she does not discern between songs based on their quality, or just likes to explore). Likewise, user is weighted more if listens to very limited set of songs. T
      * Similarity calculation is time taking process.
      * It takes more time to recommend music to users.

## CHAPTER 3 PROPOSED SYSTEM

##### PROPOSED SOLUTION:

The natural language processing (NLP) technology can serve as an interaction between computers and humans using linguistic analysis and deep learning methods to obtain knowledge from an unstructured free text. The NLP systems have shown their uniqueness and importance in the areas of information retrieval mostly in the retrieval and processing of large amount of unstructured music records and return structured information by user-defined queries. In general, the NLP system is aimed at representing explicitly the knowledge that is expressed by the text written in a natural language. There are few applications of the NLP techniques in recommending music despite the enormous amount of text-based information, which can be retrieved from user given dataset. The main challenges addressed by the application of NLP for music recommendation are flexible formatting, structure without sentences, missing expected words and punctuation, unusual parts of speech (POS), medical jargon, and misspellings.

###### Advantages Of Proposed System:

* + - The system is able to recommend music using a direct approach of the question and answering technique to suggest a mood-based music.
    - Moreover, the recommendation systems experts need a platform to keep track of large text-based

chunk of knowledge narrated by patients in a natural language, hence improving healthcare delivery for music.

##### SYSTEM DESIGN INTRODUCTION:

The System Design Document describes the system requirements, operating environment, system and subsystem architecture, files and database design, input formats, output layouts, human- machine interfaces, detailed design, processing logic, and external interfaces.

##### : ALGORITHM:

###### Methodology:

The proposed system work develops a personalized system, where the user's current emotion is analyzed with the help of the chat bot. The chat bot identiﬁes the user's sentiment by chat conversation with the user. Based on the input provided by the user, current emotion or mood is analyzed by the chat bot and it will suggest song to the user. The objective of our application is to identify the mood expressed by the user and once the mood is identiﬁed, songs are played by the application. The application solves the basic needs of music listeners without troubling them as existing applications do.

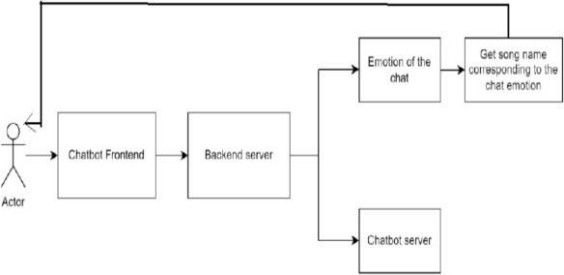


Fig: 3.2.1 Methodology

Natural Language Processing: Natural language processing (NLP) Sentiment analysis (or opinion mining) is a natural language processing (NLP) technique used to determine whether data is positive, negative or neutral. Sentiment analysis is often performed on textual data to help businesses monitor brand and product sentiment in customer feedback, and understand customer needs. Sentiment Analysis is a procedure used to determine if a chunk of text is positive, negative or neutral. In text analytics, natural language processing (NLP) and machine learning (ML) techniques are combined to assign sentiment scores to the topics, categories or entities within a phrase.

* 1. **SYSTEM REQUIREMENTS:**

##### HARDWARE REQUIREMENTS:

* System : Intel(R) Core (TM) i3-7020U CPU @ 2.30GHz
* Hard Disk : 1 TB.
* Input Devices : Keyboard, Mouse
* Ram : 4 GB.

##### SOFTWARE REQUIREMENTS:

* Operating system : Windows XP/7/10/11.
* Coding Language : Python
* Tool : Anaconda
* Interface : Flask webapp
* Database : SQL lite

##### MODULES IN SYSTEM:

* + 1. **: DATASET:**

Text file with questions and answers are used as dataset in this project. Song

recommendation related queries with links and answers are given in text file which is prepared by taking data from Spotify website. Data set can be increased by adding more records to the txt files.

###### Pre-processing:

In this step required chat bot libraries are initialized and text data is taken as input to NLP module and preprocessing is performed. Data set question and answers are stored in database. While application I executed preprocessing is performed.

###### Chat Bot Web APP:

Flask framework is used to develop chat bot application which has input option from user and question posted by user is processed and verified with the answer which is trained using NLP model and result is displayed to user based in input question.

### Development Process:

**Chatter Bot** is a library in python which generates a response to user input. It used a number of machine learning algorithms to generates a variety of responses. It makes it easier for the user to make a chatbot using the chatterbot library for more accurate responses. The design of the chatbot is such that it allows the bot to interact in many languages which include Spanish, German, English, and a lot of regional languages. The Machine Learning Algorithms also make it easier for the bot to improve on its own with the user input.

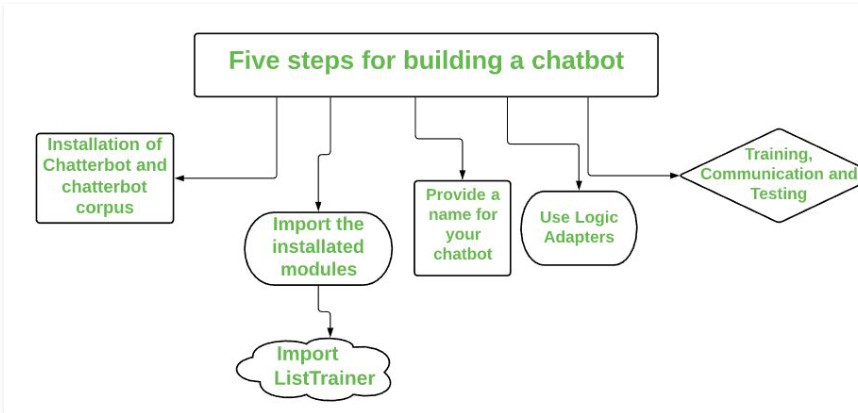


Fig: 3.4.1 Process of Developing Chat Bot

Let’s begin the journey of our own chatbot in the shortest way possible: -

###### Step 1. Install the Chatterbot and chatterbot corpus module:

The first and foremost step is to install the chatterbot library. You also need to install the chatterbot corpus library. Basically, Corpus means a bunch of words. The Chatterbot corpus contains a bunch of data that is included in the chatterbot module. The corpus is used by bots to train themselves.

Run the following pip commands on the terminal for installation:

* pip install chatterbot
* pip install chatterbot corpus

we have to import two classes: Chatbot from chatterbot and ListTrainer from chatterbot.trainers.

**List Trainer:** Allows a chatbot to be trained using a list of strings where the list represents a conversation.

###### Step 2. Name our Chatbot:

Now, we will give any name to the chatbot of our choice. Just create a Chatbot object. Here the chatbot is maned as “Bot” just to make it understandable.

###### Step 3. Use of Logic Adapter:

The Logical Adapter regulates the logic behind the chatterbot that is, it picks responses for any input provided to it. This parameter contains a list of all the logical operators. When more than one logical adapter is put to use, the chatbot will calculate the confidence level, and the response with the highest calculated confidence will be returned as output.

Here we have used two logical adapters:

* + - 1. **Best Match:** The BestMatchAdapter helps it to choose the best match from the list of responses already provided.

2. **Time Logic Adapter:** The TimeLogicAdapter identifies statements in which a question about the current time is asked. If a matching question is detected, then a response containing the current time is returned.

###### Step 4. Training, Communication, and Testing:

For the training process, you will need to pass in a list of statements where the order of each statement is based on its placement in a given conversation. We have to train the bot to improve its performance for this we need to call the **train()** method by passing a list of sentences. Training ensures that the bot has enough knowledge to get started with specific responses to specific inputs. After training, let’s check its communication skills. And the last step is to do testing.

### System architecture:

A system architecture or systems architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system. Organized in a way that supports reasoning about the structures and behaviors of the system.

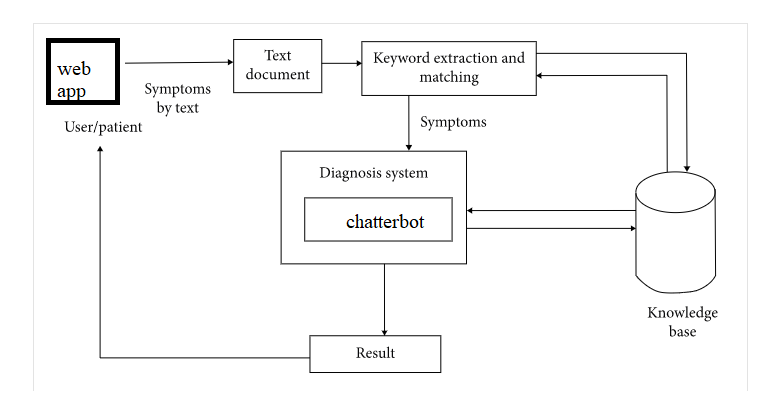
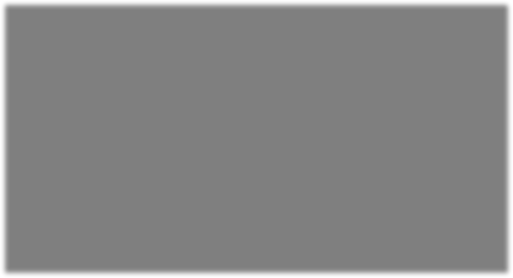
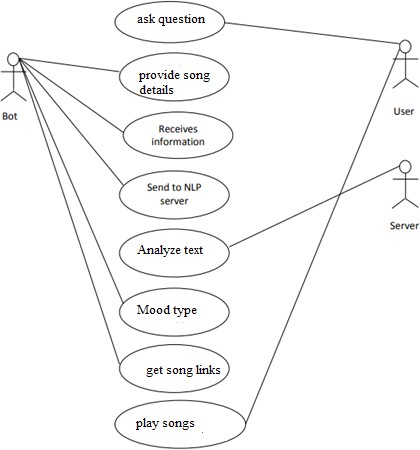


Fig 3.4.2: System Architecture

## CHAPTER 4 DESIGNING:

###### Construction of Use case diagrams:

A use case diagram in the Unified Modelling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



#### Fig 4.1: Use Case Diagram

##### SEQUENCE DIAGRAMS:

A sequence diagram in Unified Modelling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

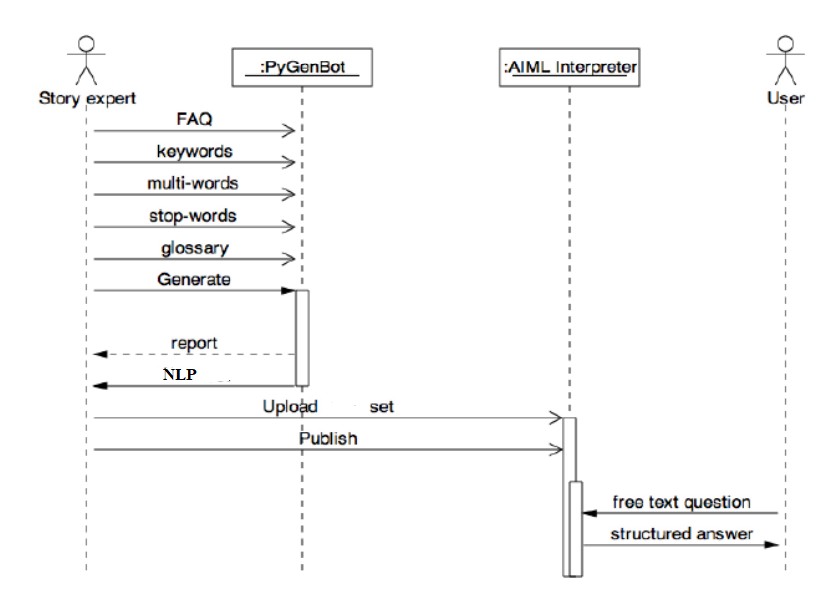
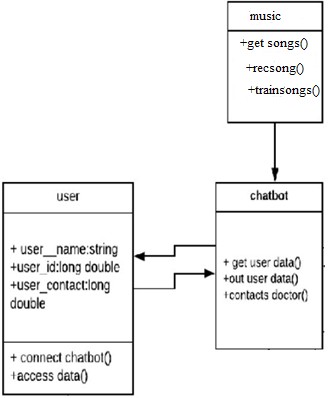


Fig 4. 2 Sequence diagram

## CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



#### Fig4.3 Class Diagram

##### ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

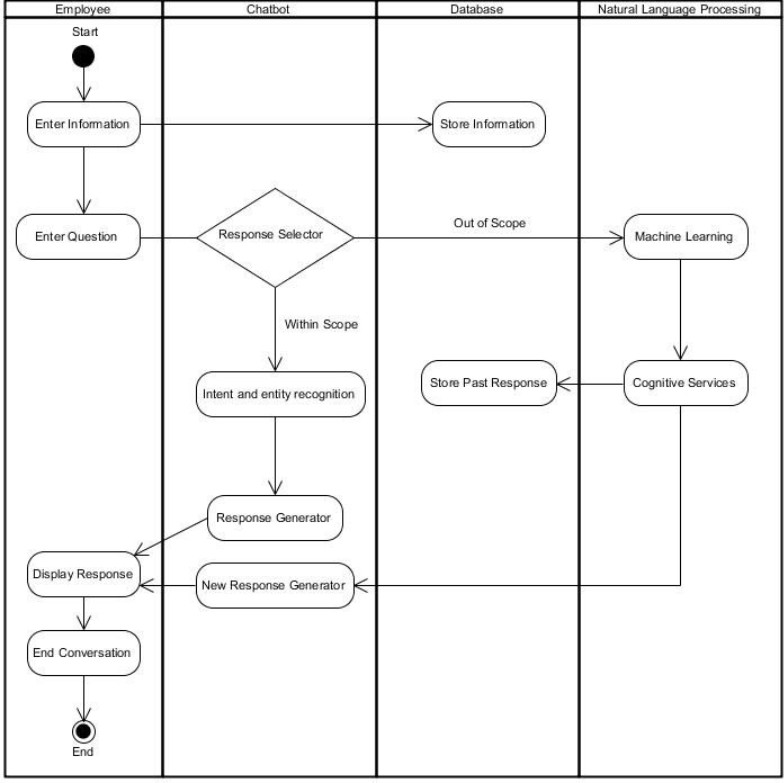
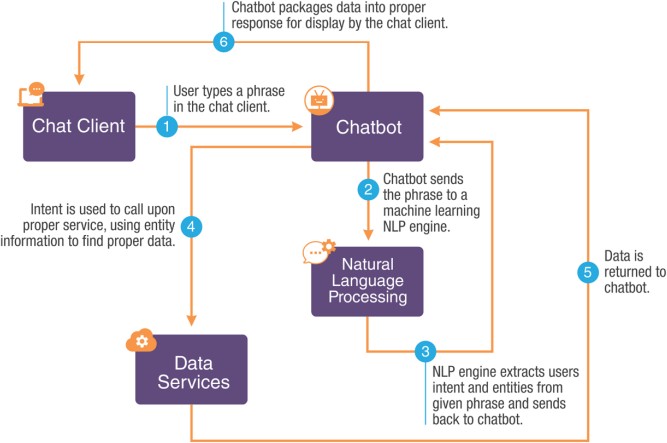


Fig4.4 Activity Diagram

### Data Flow Diagram:



#### Fig 4.5 Data Flow Diagram

**CODE:**

from flask import Flask, render\_template, request #from chatbot import chatbot

import os

from chatterbot import ChatBot

from chatterbot.trainers import ListTrainer

from chatterbot.trainers import ChatterBotCorpusTrainer import os

# Creating ChatBot Instancep

chatbot = ChatBot(

'SONG RECOMMENDATION CHATBOT',

storage\_adapter='chatterbot.storage.SQLStorageAdapter', logic\_adapters=[

'chatterbot.logic.BestMatch',

{

'import\_path': 'chatterbot.logic.BestMatch',

'default\_response': 'I am sorry, but I do not understand. I am still learning.', 'maximum\_similarity\_threshold': 0.90

}

],

database\_uri='sqlite:///database.sqlite3'

)

# Training with Personal Ques & Ans

training\_data\_quesans = open('training\_data/SONG.txt').read().splitlines() training\_data = training\_data\_quesans

print(training\_data)

trainer = ListTrainer(chatbot) trainer.train(training\_data)

# Training with English Corpus Data trainer\_corpus = ChatterBotCorpusTrainer(chatbot) app = Flask(\_name\_)

app.static\_folder = 'static'

@app.route("/") def home():

return render\_template("index.html")

@app.route("/get")

def get\_bot\_response():

userText = request.args.get('msg') #print(userText)

return str(chatbot.get\_response(userText))

## CHAPTER 5 RESULTS AND DISCUSSIONS

### Working Prototype Images:

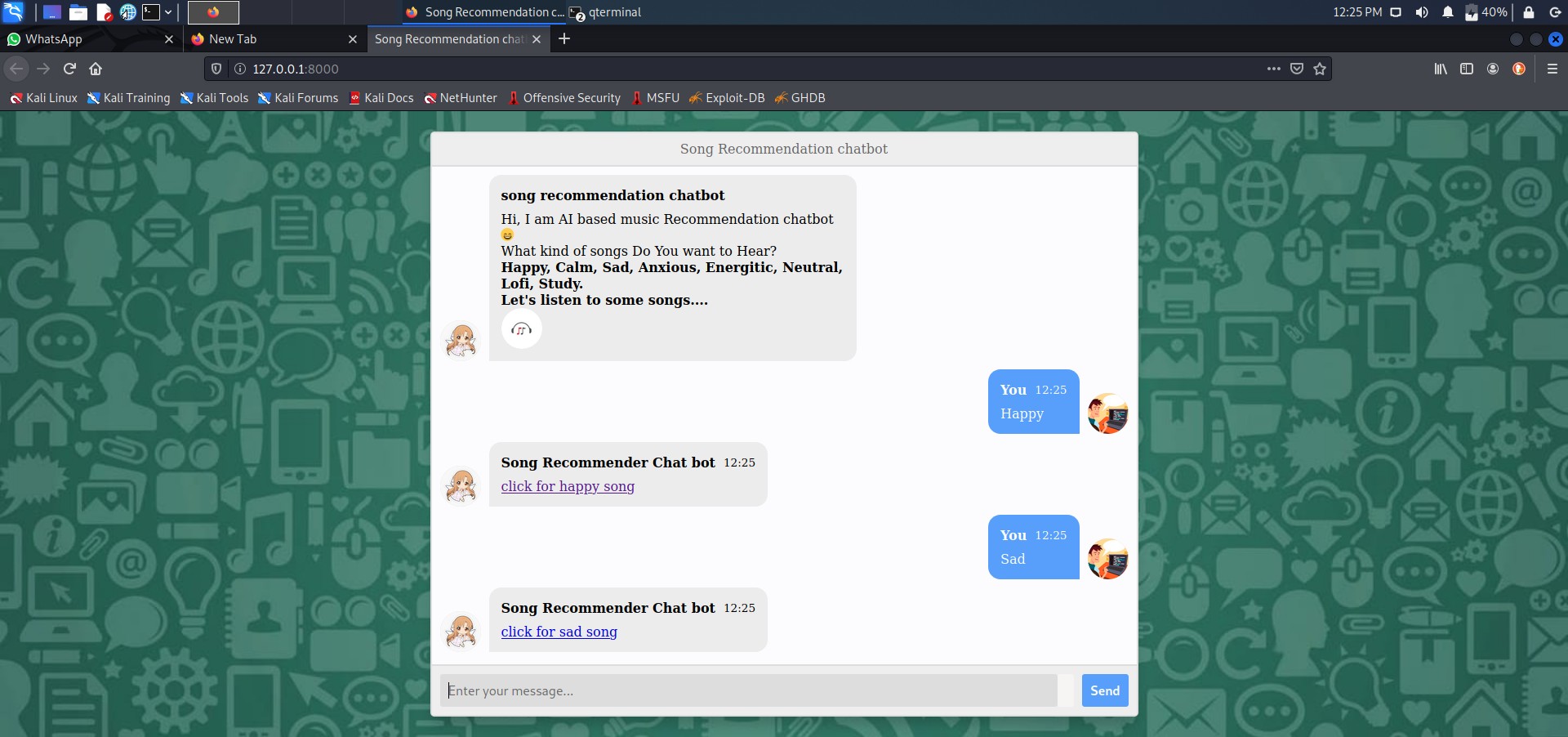


Fig 5.1 Screenshot of Execution

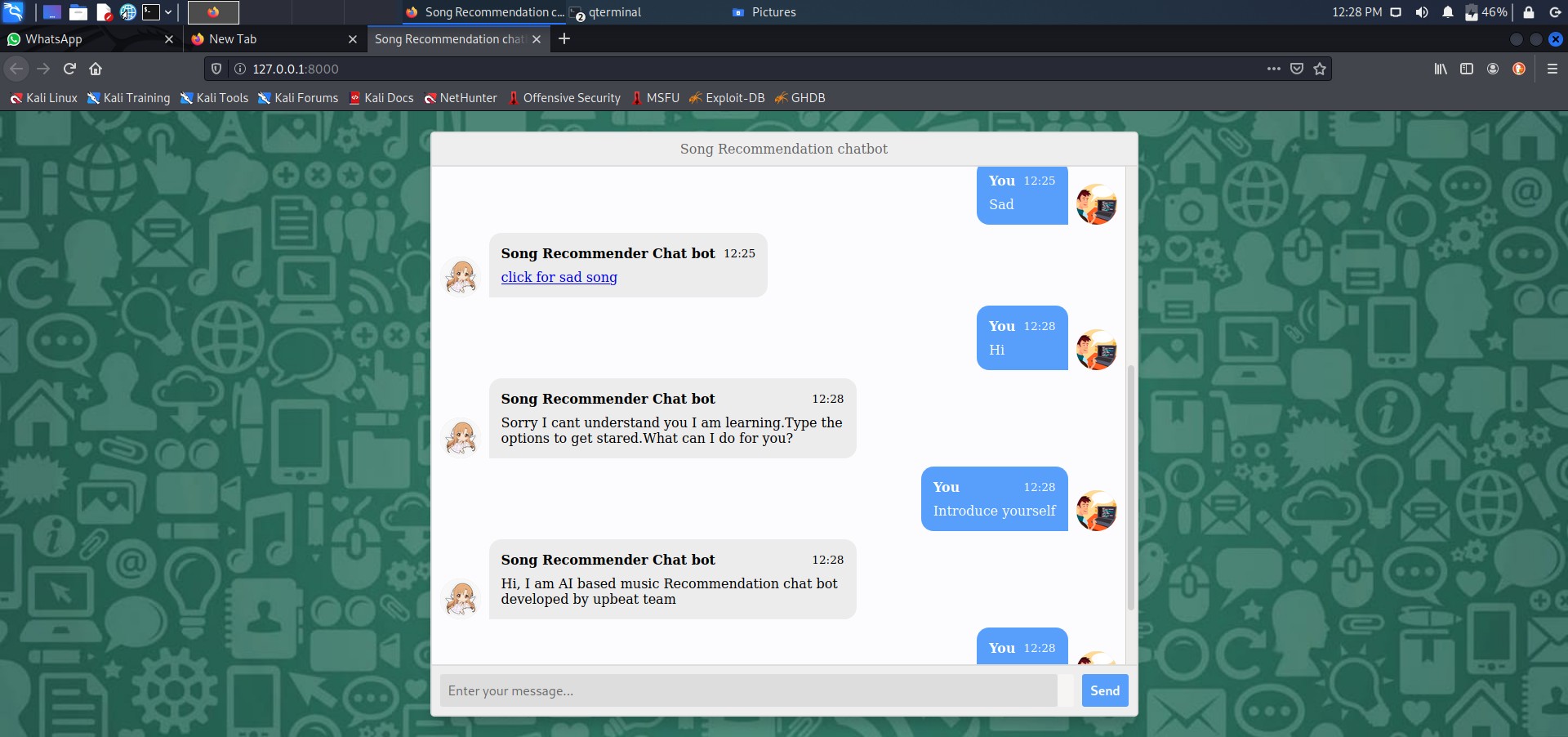


Fig 5.2 Screenshot of Execution

### Anaconda Prompt:

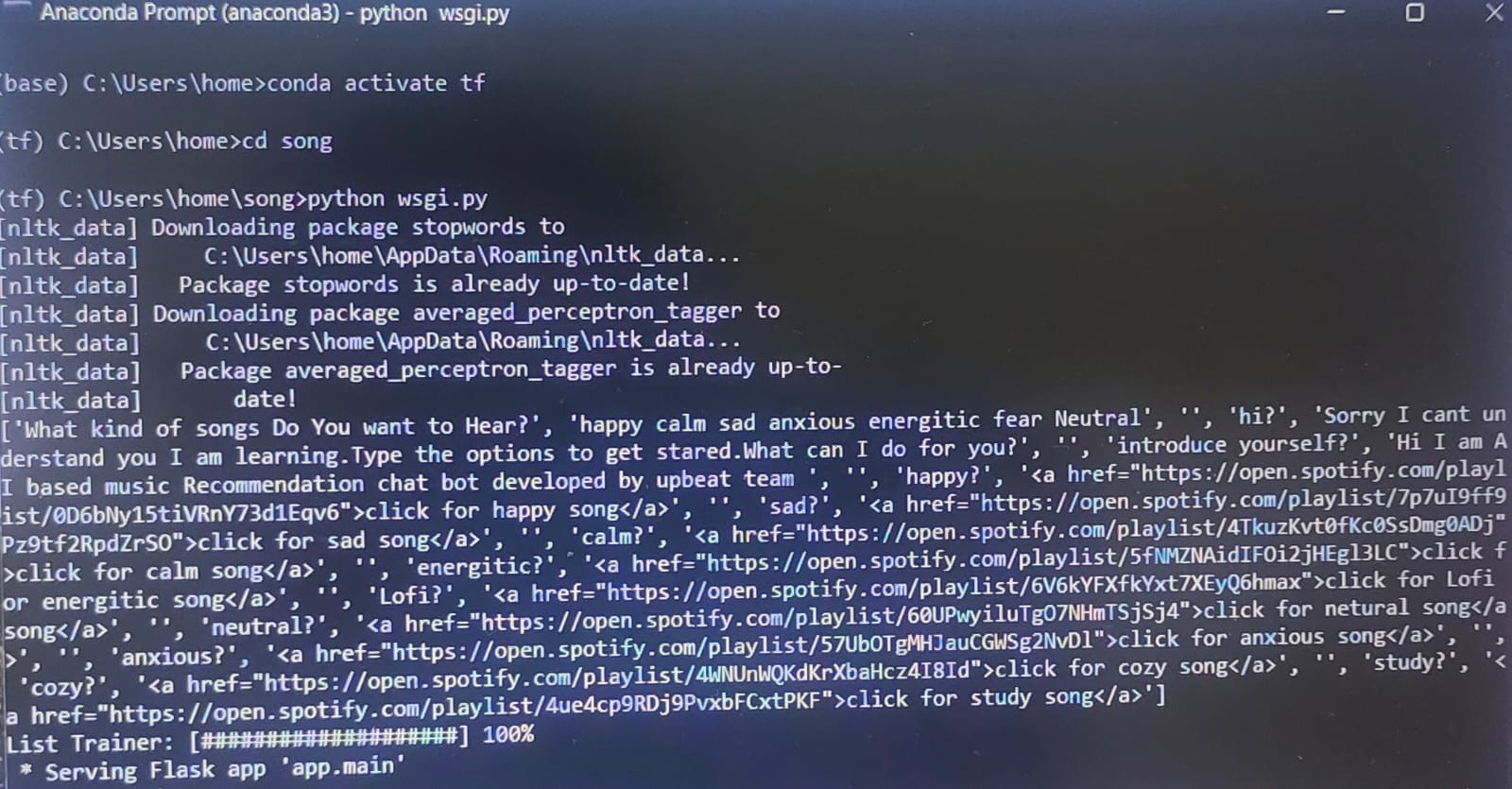


Fig 5.3 Screenshot of Execution

**SQL Lite Database:**

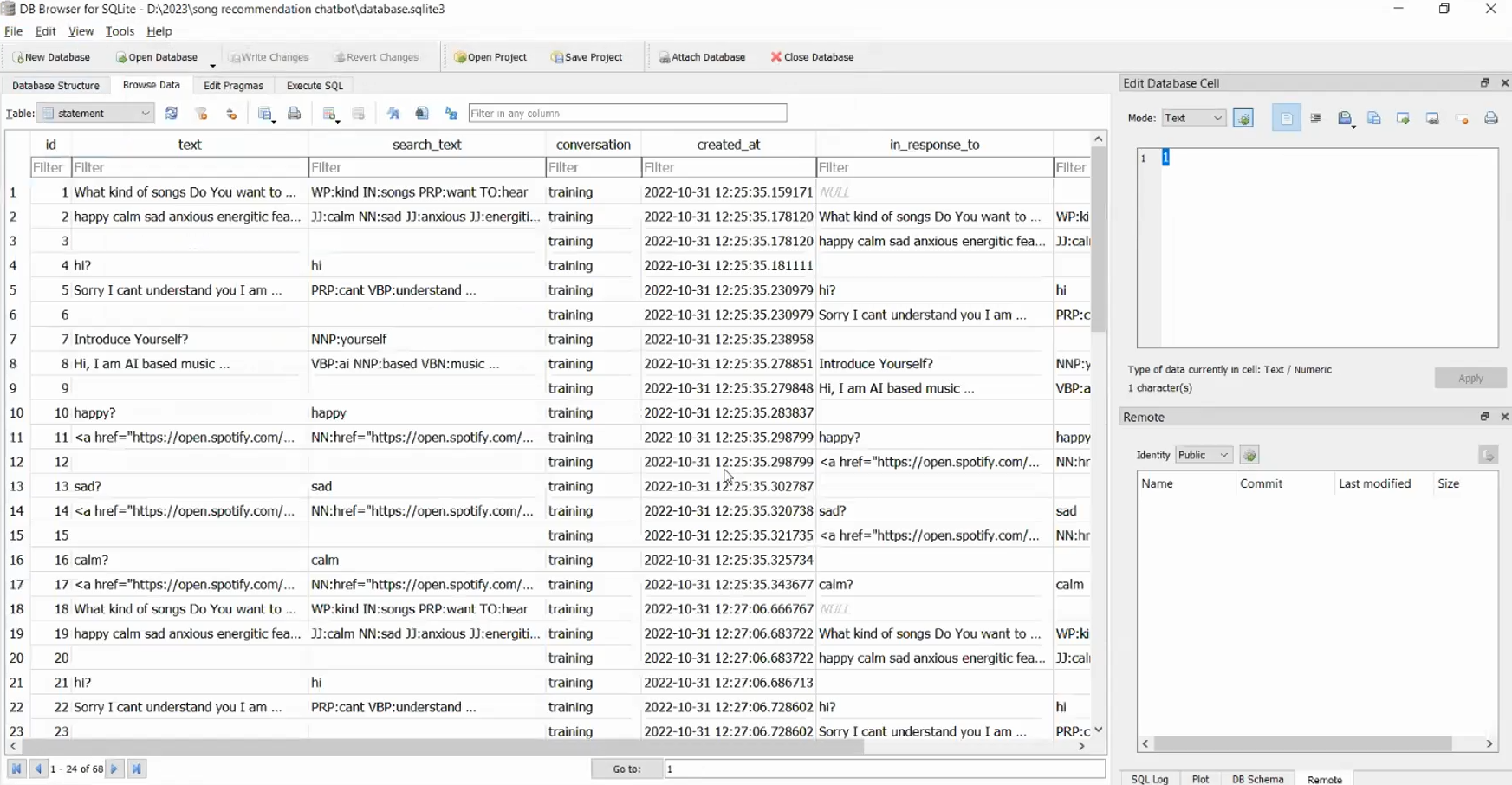


Fig 5.4 Screenshot of Database

##### DISCUSSIONS:

Technology has a great impact on every part of lives, which also includes our day-to-day habits. In present scenario, fields like artificial intelligence and machine learning are on great boom. With the help of advancement in these fields, large number of people are interacting through the system via chat bots and voice assistants. Considering above factors, this project is aimed to implement the Machine Learning based Song Recommender Chat Bot System that includes chat bot to assist user and recommend songs using the Natural Language Processing.

We facilitate the user to chat with the chat bot after all texting makes conversation between chat bot and user more interactive and it will efficiently help in analyzing the current mood of the user and based on that chat bot will recommend songs. After chatting with the chat bot, current mood of the user is analyzed and list of songs is suggested to user, using the concept of NLP (Natural Language Processing). Based on the list of songs user can choose the song to be played based on his or her choice. We have used Python as our prime language because it supports an extensive set of open-source libraries which can be used by our system.

## CHAPTER 6

## CONCLUSION & FUTURE SCOPE:

**CONCLUSION:**

We have presented a survey and methodology for building the chat bot song recommender system. To perform this, we first identified various approaches for building a chat bot known to date. We then evaluated the considered algorithms which are useful in building of our system in terms of their ability to work on the recommendation process of the system. We also gathered all the requirements needed for building our system and studied the overall process involved in chat bot's working. Lastly, we summarized the deployment requirements of our system. On the conclusion note our “Song Recommender Chat Bot System” is used to facilitate the use by people to automate and give them better music player experience. The application solves the basic needs of music listeners without troubling them as existing applications do.

##### FUTURE SCOPE:

In the future, we would like to try the following things:

* Training corpus with deep learning algorithms to improve accuracy
* Adding move recommendation system to application can be useful for users
* Android application can be developed for user friendly access.
* Trying clustering techniques to recommend music**.**

# REFERENCES:

* + 1. J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen, “Collaborative filtering recommender systems,” The Adaptive Web: Methods and Strategies of Web Personalization, pp. 291-324,2007.
    2. M. J. Pazzani and D. Billsus, “Contentbased recommendation systems,” TheAdaptive Web: Methods and Strategies ofWeb Personalization, pp. 325-341, 2007.
    3. E. J. Humphrey, J. P. Bello, and Y. LeCun, “Moving beyond feature design: deep architectures and automatic feature learning inmusic informatics,” in Proc. 13th Int‟l Conf. Music Info.Retrieval, pp. 403- 408, October2012.
    4. E. J. Humphrey, J. P. Bello, and Y. LeCun, “Moving beyond feature design: deep architectures and automatic feature learning inmusic informatics,” in Proc. 13th Int‟l Conf. Music Info.Retrieval, pp. 403- 408, October2012.
    5. W. Hsu, and C. J. Lin, “A comparison of methods for multiclass support vector machines,”IEEE Tran. Neural Networks, vol. 13, no. 2, pp. 415-425, 2002.
    6. Abbas SaliimiLokman, JasniMohamadZain,FakultiSistemKom puter, KejuruteraanPerisian,” Designing a Chatbot for Diabetic Patients”, ACM Transactions on Management Information Systems (TMIS), Volume 4, Issue 2, August 2015 .
    7. PavlidouMeropi,Antonis S. Billis,Nicolas D.Hasanagas, CharalambosBratsas,Ioanni sAntoniou,Panagiotis D.Bamidis,” Conditional Entropy Based Retrieval Model in Patient- CarerConversational Cases”, 2017 IEEE 30th International conference on Computer-Based Medical System