A PRELIMINARY REPORT ON

"KAANSEN: THE RAGA RECOGNIZER"

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE
IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE

OF

BACHELOR OF ENGINEERING (COMPUTER ENGINEERING)

SUBMITTED BY

Students' Names	Roll No.:
Harsh Apte	17U248
Savani Gokhale	17U218
Sanket Arekar	17U238
Sarvajit Sankar	17U239

Sinhgad Institutes

DEPARTMENT OF COMPUTER ENGINEERING

STES'S SMT. KASHIBAINAVALECOLLEGE OF ENGINEERING

VADGAON BK, OFF SINHGAD ROAD, PUNE 411041
SAVITRIBAI PHULE PUNE UNIVERSITY
2019 -2020

Sinhgad Institutes

CERTIFICATE

This is to certify that the project report entitles

"KAANSEN: THE RAGA RECOGNIZER"

Submitted by

Students' Names	Roll No.:
Harsh Apte	17U248
Savani Gokhale	17U218
Sanket Arekar	17U238
Sarvajit Sankar	17U239

are bonafide students of this institute and the work has been carried out by them under the supervision of **Prof. K. N. Honwadkar** and it is approved for the partial fulfillment of the requirement of Savitribai Phule Pune University, for the award of the degree of **Bachelor of Engineering** (Computer Engineering).

(Dr. K. N Honwadkar)
Guide
Department of Computer Engineering

(Dr. P. N. Mahalle)
Head,
Department of Computer Engineering

(Dr. A. V. Deshpande)
Principal,
Smt.Kashibai Navale College of Engineering Pune – 41

Place: Pune

Date:

ACKNOWLEDGEMENT

The success of this project is the outcome of the enormous contribution of various people, sources involved directly or indirectly with this project work. It is immense pleasure to express our sincere gratitude to all of them. Our guide was the one who gave us the idea to work on this project. So, we do thank our guide Dr. K N Honwadkar for suggesting this project and giving as some ideas as to how to go about the project.

We thank the support staff and all our colleagues who were fruitful in helping us with the project phases by giving their valuable time even though they had their own work. We are thankful to and fortunate enough to get constant encouragement, support and guidance from all Teaching staffs of computer department who (if any) helped us in successfully completing our project work. Also, we would like to extend our sincere esteems to all staff in laboratory for their timely support.

We would also like to thank our Head of Department to Prof P N Mahalle sir, who guided us all along, till the completion of our project work by providing all the necessary information for developing a good system. We are grateful to SMT KASHIBAI NAVALE COLLEGE OF ENGINEERING PUNE for giving us an opportunity to develop this project.

NAME OF THE STUDENTS

Harsh Apte
Savani Gokhale
Sanket Arekar
Sarvajit Sankar

ABSTRACT

Indian classical music ragas are difficult to recognize even for the professionals involved in the industry. Every person who has interest in listening to classical music would want to know what raga was used in making of the song.

The only way to recognize the raga was asking some professionals in their vicinity, or checking YouTube videos on that if at all that exists. Thus, there was need for a way for everyone to be able to recognize ragas without need of lot of manual work.

Thus, the project aims to solve problem at a basic level. The software we are building involves accepting the input audio file. Processing the audio file for extracting the necessary information from it. Then, using the information gained and passing it to machine learning datasets and getting output as the identification of the raga recognized to the user.

Thus, we solve this problem of recognition for the benefit of people at large without the need for lot of manual intervention.

The software involves a dataset with information that it learned with the supply of information provided to it during training of the model phase. With this phase, it learns the different possibilities of that raga to occur, the forms, the various frequency ranges possible for it. Thus, during testing phase, it can successfully recognize the raga with good efficiency. The basic idea is to recognize the raga using “Bandish” and “Pakad

TABLE OF CONTENTS

LIST OF ABBREVATIONS

ABBREVIATION	ILLUSTRATION
SVM	Support Vector Machine
HMM	Hidden Markov Model
KNN	K-Nearest Neighbors
PCD	Pitch-class Distributions
PCDD	Pitch-class dyad Distributions
CFS	Correlation based Feature Selection
FCBF	Fast Correlation based Filter
RAM	Random Access Memory
CPU	Central Processing Unit
GB	Gigabyte
SRS	System Requirements Specifications
UML	Unified Modelling Language
NP-Hard	Non-Polynomial Hard

LIST OF FIGURES

LIST OF TABLES

01. INTRODUCTION

1.1 OVERVIEW

The melodic formalization of melodies found in Indian Classical music (Carnatic and Hindustani) is known as RAGA. It comprises of a sequence of swaras depicting the mood and sentiments. Indian Classical music has seven basic swaras (notes) namely (Sa, Re, Ga, Ma, Pa, Dha, Ni). There are several hundreds of ragas in Indian Classical music derived from 72 parent or Janaka ragas, formed by the combination of 12 swarasthanas. The raga system uses a method of organization of tunes or notes based on certain natural principles.

Notes in the same raga use the same swaras in various combinations and with practice, the listener can pick up the resemblance. Indian classical music is without a doubt defined by two basic elements - it must follow a Raga which is classical mode, and a specific rhythm, the Taal. In any Indian classical composition, the music is based on “drone” which is a continual pitch that sounds throughout the concert called a tonic. Each raga has a swaroopam (a musical form or image) that is defined by the swaras used, the gamakas given to these swaras, the progression in which the swaras occur etc. This classification is termed as the raga lakshanam. For example, Raga lakshanam has the “arohana” and “avarohana”, details of raga “chaya swaras” (the swaras which are chiefly responsible for the characteristic melody of the raga), gamakas, characteristic swara phrases and general usage notes. It is intended and inclined more towards the performer than for the listener.

Arohanam is the sequence of swaras used in a raga in the ascending passages which means increasing the pitch frequencies as the pitch goes up. Avarohana is the sequence of swaras to be used when we need to go down in pitch that is in descent. The arohana and avarohana (or the scale) of a raga provide only the outline or structure upon which the rest of the raga is formed.

1.2 MOTIVATION

Identifying ragas is a crucial and difficult problem for appreciating, comparing and learning Indian music. Due to the overwhelming number of ragas in classical music, complicated structure, sharp resemblance and minor variations of ragas, even humans find it difficult to identify them, without years of practice, that is unless the person is an expert.

Along with this, most of the Indian People are not well connected to the classical music. Encouraging the people to appreciate Indian Classical Music and its creative use in the songs is the motive behind this. To help people to recognize the ragas associated with musical tune/song. So that, they can understand the roots and enjoy the music. Also, particularly the users having no background in classical music will be able to enjoy and understand the classical aspects of songs. This will increase enthusiasm related to the field of Indian Classical Music. Our shared interest in the Classical music triggered us for creating this RAGA identifier.

As a result, raga identification can provide a basis for finding for similar songs suited for a certain beautiful theme. It can also be used by new musicians who find it difficult to differentiate ragas which are very similar to each other. This can also be used to check how accurately a person is performing.

1.2.1 PREVIOUS WORK

The sequence of the particular RAGA is same that's why SVM[1] and HMM[4] models can be used to distinguish between the inputs. The database consisted of 20 hours of RAGAs along with some commercial recordings which were split into 30 sec and 60 sec segments. Total 60% of accuracy was achieved.

Spot on accurate results were obtained using a K-NN classifier with 60/40% train/test split. This was further developed where PCDs[6] and PCDDs were used as features with more sophisticated learning algorithms. In a 17-target experiment with 142 segments, classification accuracy of a whopping 94% was attained using 10-fold cross validation. However, the significance of the results in both cases was limited by the size of thedatabase.

1.3 PROBLEM DEFINITION AND OBJECTIVES

Developing a music recognition software which will receive the input in the form of audio format (especially traditional songs in Hindi and Marathi language). By extracting the signature tune from the given input and processing it, product will give an output in form of RAGA name. Being concentrated on Indian Classical music, software will match extracted tune with the Raga Tunes stored in database and will suggest the user about belongingness of it to particular RAGA.

Raga Identification is a process of listening to a piece of music, synthesizing it into sequence of notes and analyzing the sequence of notes for identifying the raga it follows. The identification of ragas is very difficult, and comes only after ample amount of experience

To identify RAGAs using software, the characteristics of ragas have to be converted into suitable features. This becomes difficult for Indian music due to the following reasons:

(i) A musical input may be composed from multiple instruments during a performance.

(ii) Unlike Western music, the notes in Indian music are not on an absolute scale that is fixed but on a relative scale that is, it can largely vary with various parameters.

(iii) There is no fixed starting swara in a raga.

(iv) Notes in Indian music do not have a fixed frequency but rather band of frequencies (oscillations) around a note, that's why the raga can be constructed in any of the band.

(v)The sequence of swaras in the ragas are not fixed and various improvisations are allowed while citing a raga as long as the characteristics of raga are intact.

These factors make RAGA detection more difficult as these are not discrete quantities, which is otherwise easy to differentiate. The RAGAs can be mixed with each other and also more than two RAGAs can be used in a song, each RAGA at particular instant. This case make identification more difficult.

1.4 PROJECT SCOPE AND LIMITATIONS

1) Project Scope Objectives:

This product will increase the appreciation towards Indian Classical Music. This can work as an introduction for classical music. After using this product, the overall awareness related RAGAs will increase and people can understand the roots and enjoy the music. We want our product to bridge the gap between Classical music and contemporary music.

2) Project Scope Milestones:

a. Finalizing the Requirements:

The requirements should be based on amount of data we will be processing. The storage capacity depends on the size of dataset. The RAM and processing power of CPU depends on the Models we will be using. The Higher the processing power, the less time it will take to training and testing on models.

b. Use of efficient Model:

There are various models previously used for RAGA Identification like SVM, KNN, HMM etc. The type of model to be chosen highly depends on two factors, first is the type of features to be extracted from the input and second, the size of the dataset.

c. Training of the Model

The Dataset should be divided into training set and testing set. The proportion should be (60-40) % or (70-30) %. The training of the dataset should be done with proper and accurate samples so that the outcomes during the testing phase will be correct. The most decisive features should be used to feed the model, this will help the model to differentiate the samples easily. This helps in identifying the learning capacity of the model.

d. Testing on the Model

This includes feeding the model with unknown input samples. This helps in testing the learned knowledge of the model. The result of testing phase should be analyzed thoroughly to find out the advantages and limitations of the model. If the result is not favorable, the dataset or the model is changed, and training and testing phase is performed again till we get the highest accuracy.

e. Keeping Check on the Accuracy

The accuracy of the model should be checked after every change made in the model or dataset. The accuracy can be increased by selecting the most efficient and suitable model to the system. Sometimes models having more accuracy fail in generalization. This happens when there are too many features. They work extremely well on obvious inputs but fail badly on generalized inputs. This is known as Overfitting. Thus, moderate accuracy is accepted to prevent overfitting.

f. Embedding the system in the Software.

The Model used cannot be used by the user as it is not user friendly. We need to wrap the model so that an ordinary man can use our model with ease. Thus, we need to embed our model into a software. This will have a front-end and back-end. The interface should be informative and unambiguous.

LIMITATIONS:

This software isn't going to teach raaga recognition. This software will not help in learning raagas, it will merely help in identifying the ragas associated with a song.

At a time only 1 file can be processed, since backend includes audio rendering code which takes lots of CPU resources to render and give output. Hence, user cannot input multiple files at a time to recognize the raga of the concerned songs.

This software currently only accepts ".wav" file as input. So, if any other audio file is there with the user, we currently don't provide with audio conversion facilities. The user needs to convert his/her audio to wav format on his/her own. The reason is because the libraries used in backend works only for wav format of file.

The accuracy of the software depends on the size of the dataset we input for training and testing, hence the error rate is high as we have implemented our own dataset and it is very small in size.

The software will not be able to recognize songs which don't have Ragas in them. Since, this software uses a range of frequency to recognize the ragas, the software isn't trained to recognize any kind of music which isn't the set of ragas for which the software is trained for.

The software will fail to recognize Ragas for which it hasn't trained its modelyet. At present we aren't implementing for more Ragas.

The software doesn't have microphone listening capability to recognize raga on the go. User will have to input audio file manually.

The software can recognize a song with a Raga, but a case might be that there won't be raga in it. So, there is a possibility of false positive. The software efficiency depends on the dataset size, so it may not be able to detect all ragas.

1.5 METHODOLOGIES AND PROBLEM SOLVING

The research work of all these papers start from 2009, wherein the paper by Surendra Shetty, K.K. Acharya[6] discusses how they are solving the problem of recognition using aaroha and avaroha patterns in the music. The drawback of this method was that it used to though successfully recognize monotonic music, it did not always recognize in the same way polytonic music.

Another paper that was published during 2009 was by Shreyas Belle, Rushikesh Joshi, Preeti Rao[2] which uses swara intonation to recognize the ragas. But they failed to consider the possibilities that intonations don't need to be in just one frequency variations, but there can be various bands of intonations possible for a same raga. This is what we call a performers variability.

Paper by author K. Priya[3] and others in 2012 came up with better solution which gets better results than the previous research by using different Data Mining Techniques. Though this was done for Carnatic music which is different genre of music, the result and solutions can be applied to this too. The disadvantage was that even with using such advanced Data Mining techniques the resulting efficiency wasn't quite high. Hence, the recommendation to improve that was given as implementing genetic algorithm over their proposed solution.

A new way of approaching the Raga recognition was proposed in paper in 2012 by Joe Cheri Ross, Vinutha T. P., Preeti Rao[4], wherein they used method based on motifs of ragas for recognition. This is by far the most advanced in recognition of the ragas only by the basis of the constraints they have considered for the possibility of raga occurrence in a audio file. This proposed system has good results in polytonic detection and also considers the performance variability factors. The variability factors are the variations in voice because of which a raga is never in the same frequency spectrum for 2 singers or performers. This method is quite complex.

The discrepancies of 2009 paper by Shreyas Belle, Rushikesh Joshi, Preeti Rao[2] which uses swara intonation to recognize the ragas which had faults like they failed to consider the possibility that intonations doesn't need to be in just one frequency variations, but there can be various bands of intonations possible for a same raga. This is what we call a performers variability. Such discrepancies were solved in 2014 when a new proposed system by Vijay Kumar et al.[1] used nonlinear SVM framework using two kernels to propose that they have also considered the temporal information of the notes which was ignored till now, because of which they can now have information of performance variability. This method was implemented for Carnatic music, but of course the proposed system can be tweaked to satisfy for Hindustani music too.

A new approach to solve the raga recognition problem was suggested in 2019 paper by Gaurav Pandey et al.[5] did have a high efficiency rate. They used Hidden Markov Model enhanced with a string-matching algorithm. This did help them in retrieving the results faster. However, this failed to consider the fact that ragas can be in different frequency pitches which therefore causes problems in string matching. It also won't be able to recognize a raga if a performer changes a small swara within the raga to suit the narrative or the context of the surrounding and to have a poetic and emotional effect on the people immersed in to hearing the performer. The performer may tweak a swara to a sharp or flat frequency band for emphasis purposes, then the string-matching algorithm will not take into account this variable change.

02. LITERATURE SURVEY

Paper Name	Abstract	Author
1. Identifying Ragas in Indian Music	The paper provides raga classification problem in a non-linear SVM framework using a combination of two kernels that represent the similarities of a music signal using two different features pitch-class profile and n-gram distribution of notes.	Vijay Kumar, Harit Pandya, C.V. Jawahar
2. Raga Identification by using Swara Intonation	The paper describes an experiment that compares the intonation characteristics for distinct ragas with the same set of swaras. Features derived from swara intonation are used in a statistical classification framework to classify audio segments corresponding to different ragas with the same swaras.	Shreyas Belle, Rushikesh Joshi, Preeti Rao
3. Data Mining Techniques for Automatic recognition of Carnatic Raga Swaram notes	This research work deals with automatic identification of Carnatic raga Swaram notes through Data Mining algorithms. the swaram notes were also investigated through the use of appropriate feature relevance algorithms namely Feature ranking, Correlation based Feature Selection (CFS) filtering, and Fast Correlation based Filter (FCBF) filtering.	K. Priya, R. Geetha Ramani, Shomona Gracia Jacob

4. TANSEN : a system for automatic raga identification	This paper presents an approach to solve the problem of automatic identification of Ragas from audio samples. The system, named Tansen, is based on a Hidden Markov Model enhanced with a string matching algorithm.	Gaurav Pandey, Chaitanya Mishra, Paul Ipe
5. Detecting melodic motifs from audio for Hindustani Classical Music.	The motifs, or key phrases, provide strong cues to the identity of the underlying *raga* in both Hindustani and Carnatic styles of Indian music. Thus the automatic detection of such recurring basic melodic shapes from audio is of relevance in music information retrieval.	Joe Cheri Ross, Vinutha T. P. , Preeti Rao
6. Raga Mining of Indian Music by Extracting Arohana-Avarohana	Raga is an attractive combination of notes, engaging tonal context. Here the paper depicts a system which takes an audio file as an input and converts it into sequence of notes, identifies the raga by extracting its Arohana-avarohana pattern.	Surendra Shetty, K.K. Acharya

03. SOFTWARE REQUIREMENT SPECIFICATION

3.1 USER CLASSES & CHARACTERISTICS:

Classical Music Enthusiasts: Majority of the people can use this app, especially enthusiasts will love it. People will able to polish their musical skills and knowledge about Classical Music with the help of this app.

Ordinary people: People will gain lot of knowledge about the ancient music culture of Indian Subcontinent and thus will help in generating interest in Classical Music. This will also help in spreading awareness about Classical music.

Classical Music Experts: These people can test our software on different musical compositions thus helping them to understand the formation of the composition. They can use this Software to compose newer audio tracks.

The technical experience of these users will be good as the system will be straightforward and easy to use.

3.2 ASSUMPTIONS & DEPENDENCIES:

The project will be focusing only on the Hindustani Classical Music for RAGA Detection and Excluding other forms of music.

Therefore, the input is assumed to be a Classical music and any input other than Classical music will lead to erroneous deductions.

The Model may face problems when there are many ragas in a particular input sample and may detect the RAGA in the particular instant of the sample, which may not be prominent. Thus, more complex combinations and exceptions in the samples are excluded.

For inputs having no Classical composition the model will identify the RAGA whose frequency is more identical to the input. This may lead to erroneous deductions.

3.3 FUNCTIONAL REQUIREMENTS:

3.3.1 INPUT:

The user can give input in the form of mp3 file.
Input will consist of songs or a bandish (songs used while rendering the raga) which will be in a standard format. This input should have the phrases similar to a raga in order to detect the raga. A song which is based on some raga has characteristic motifs of the raga which consequently highlights the mood.

3.3.2 FREQUENCY EXTRACTION:

Input is a mp3 file.
It will compress the background frequencies of the mp3 file.
Output is extracted vocal frequencies. Such vocal frequencies will ease the operations for raga detection as they will provide selective frequencies of local maxima of the frequency graph afterwards. This step is necessary in order to extract patterns from the input song.

3.3.3 SHIFT OF SCALE:

Extracted vocal frequencies is the input for this stage.
Vocal frequencies extracted are shifted with respect to the base scale. This enables an efficient matching of input with the raga samples. This step is a good start as every song is of a different scale.
Output is a file which is used for further comparison.

3.3.4 KEY PHRASES EXTRACTION:

Input at this stage will be Hindi songs and bandish (songs used in classical music).
The key phrases are learned at this stage. These phrases are a combination of notes which are prominently seen in a particular raga.
To give an example, Raga Yaman:
This raga is one of the most common ragas used in Hindi songs and also it is popular in classical shows. For better understanding of how this function works, following information is necessary:
Aroha: Ni Re Ga Ma Dha Ni Sa … (Ascending order) Avaroh: Sa Ni Dha Pa Ma Ga Re Sa… (Descending order)
Both of these help for rendering this raga, as these are the basic orders of the swaras for raga yaman.
Songs consisting raga yaman might have the following phrases:
Ni Re Ga Ma
Pa Ma Ga Re
Ni Re Sa
Ga Ma Dha Ni Sa
Ni Dha Pa Ma Ga Re
Phrases like these should be extracted from the input given. Output of this stage is further used for clustering of similar phrases.

3.3.5 CLUSTERING:

Key phrases extracted from the previous stage is the input at this stage. Labelling according to the similarity between phases will be done here. The similar phrases will result to give similar labels for input files.

3.3.6 KEY PHRASES MATCHING:

Phrases from song and key phrases will be matched at this stage. Label of the key phrases will be given as output.
Section 4 gives how phrases are important to identify the raga. The phrases detected in the current input will be matched with the earlier inputs.
If the similarity found with respect to the earlier files is greater than that raga name will be displayed on the next stage for the user.

3.3.7 OUTPUT:

This stage simply shows the user the name of a raga. Raga which influences the input song will be displayed at this stage.

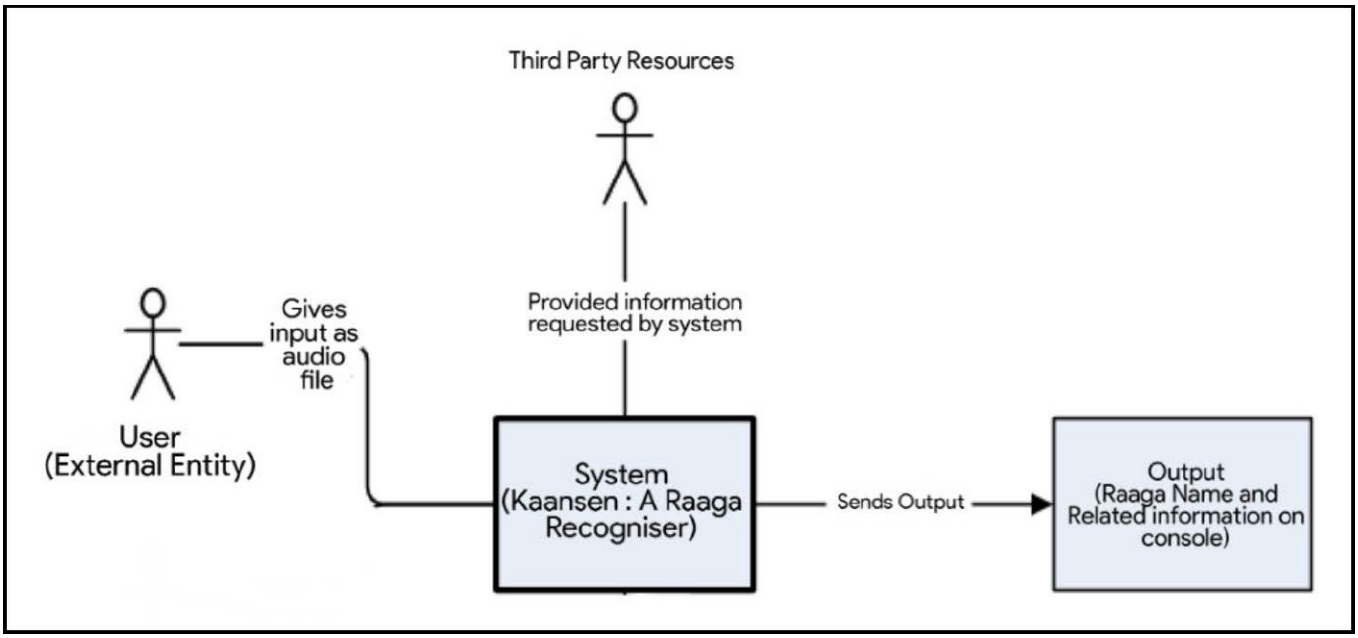


Figure 3.1 Context Diagram

As evident from the Context diagram i.e. **Figure 3.1** as shown above, user needs to input a song as audio file and the system will do the necessary processing with the help of 3rd party resources and will generate output as Raga name for the Raga associated with the song.

3.4 NON-FUNCTIONAL REQUIREMENTS:

3.4.1 PERFORMANCE REQUIREMENTS

1) Processing Speed-
   At a time, only 1 raga can be given as input, and its result can be evaluated. The processing power is totally catered towards finding that one answer only.

2) Reliability-
Software will be able to recognize the ragas effectively with minimum error. Because some things aren't taken into account while making this software such as performers style to tweak a raga as they please to satisfy the people with entertaining and mesmerizing effects.

3) Scalability-
The software can be scaled to recognize more ragas in the future easily by implementing necessary changes in the backend. Once such a scaling is done, it will help everyone to recognize any type of raga with some efficiency.

3.4.2 SAFETY REQUIREMENTS

The software shall generate output in textual format thereby avoiding potential high sound hazards, other than that, there are no safety requirements to be fulfilled by the software. The software makes sure that there are no environmental damages done by the actions performed by the same.

3.4.3 SECURITY REQUIREMENTS

The software doesn't take inputs from microphone, has no user login, so security requirements aren't present in this implementation. It is quite safe for user's data as it doesn't store any information of the user. This makes it unnecessary to implement different techniques for cyber security.

3.4.4 SOFTWARE QUALITY ATTRIBUTES

1) Verifiability-
The intended functioning of software can be easily checked by a classical music practitioner in the development team by checking the actual output with expected output.

2) Accessibility-
Anyone with any language background can input the music to recognize the ragas.

3) Usability-
Each and every person with little background in English can understand the functioning of the software and get well acquainted with the functioning of the software.

3.5 SYSTEM REQUIREMENTS:

These are the requirements which will be needed by both developer and user. The developer will need tools to develop the Models. We will use Anaconda Navigator for this purpose. **Anaconda** is a free and open-source distribution of the Python and R programming languages for scientific computing (machine learning and data science applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. Anaconda Navigator is the best as it contains all other tools which are necessary. It also has the Spyder IDE, which we will use for writing code for logic. Scikit-learn is a software machine learning library for the Python programming language. This library also contains codes for various Machine Learning mode.

Users will only need the Google Chrome for Running the Software as Software Requirements and need all the Hardware requirements as mentioned below.

3.5.1 SYSTEM DEVELOPER REQUIREMENTS:

Software Requirements:

Table No.2 System Developer Software Requirements

Name	**Version**	**Source**
Anaconda Navigator	1.9.7	https://www.anaconda.com/distribution/#download-section
Spyder	3.3.6	Inbuilt in Anaconda Navigator
Scikit-Learn package	0.21.2	https://scikit-learn.org/stable/install.html
Google Chrome Browser Mozilla Firefox	Version 76.0.3809.132 (Official Build) (64-bit)	https://www.google.com/chrome/

Hardware Requirements:

Table No.3 System Developer Hardware Requirements

Name	Specification
CPU and memory	Minimum (2 cores and 4 GB RAM) Recommended(2 cores and 8 GB RAM) **Note:** In a production environment, the minimum requirement would be 8 cores and 16 GB RAM for handling a 50-GB to 100-GB volume of data per day and up to four users.
Disk space	Approx. 10 GB
Operating System	Linux Red Hat Enterprise Linux versions 6.4, 6.5, 6.6, 6.7, 7.0, 7.1, and 7.2 (64-bit)

	Ubuntu 15.10 (64-bit) Amazon Linux AMI 2015.09 (64-bit) SUSE Linux Enterprise Server 11 (64-bit) CentOS 7.1-1503 (64-bit) Cloud Linux 7.2 (64-bit)
Processor	i5 / i7 /i9 or any latest version

3.5.2 SYSTEM USER REQUIREMENTS

Software Requirements:

Google Chrome Browser / Mozilla Firefox	Version 76.0.3809.132 (Official Build) (64-bit)	https://www.google.com/chrome/

Hardware requirements:

Table No. 4 System User Hardware Requirements

Name	Specification
CPU and memory	Minimum (2 cores and 4 GB RAM) Recommended (2 cores and 8 GB RAM)
Disk space	Approx. 10 GB
Operating System	Linux Red Hat Enterprise Linux versions 6.4, 6.5, 6.6, 6.7, 7.0, 7.1, and 7.2 (64-bit) Ubuntu 15.10 (64-bit) Amazon Linux AMI 2015.09 (64-bit) SUSE Linux Enterprise Server 11 (64-bit) CentOS 7.1-1503 (64-bit) Cloud Linux 7.2 (64-bit)
Processor	i5 / i7 /i9 or any latest version

3.6 SDLC MODEL

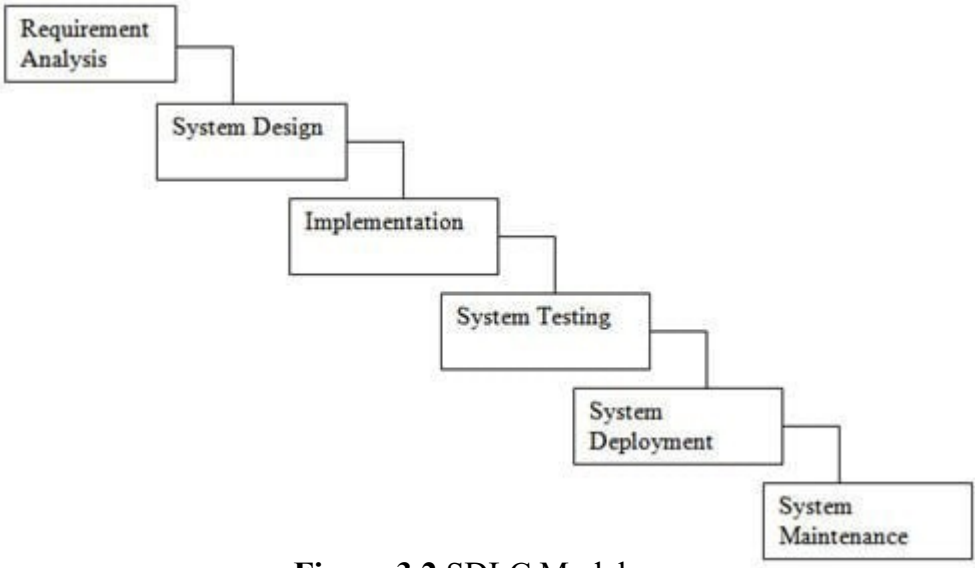


Figure 3.2 SDLC Model

Software Development Life Cycle Diagram

As evident from the **Figure 3.2** above, the Waterfall Model is sequential design process, often used in Software development processes, where progress is seen as the phase of conception, Requirement Analysis, System Design, Implementation, System Testing, System Deployment, System Maintenance. This Model is also called as the classic Life cycle model as it suggests a systematic sequential approach to software developments. This is one of the oldest models followed in software engineering.

There are 6 Phase of waterfall model:

1) Requirement Analysis

During this Phase all the relevant information is gathered related to the project, like what is required to make this project. There are various types of requirements.

a) System requirements

These are the requirements needed by Developers and Users to run the software on their respective machines. The Users machine should be compatible with the software specifications for smooth working and they should satisfy both software and hardware requirements. System requirements are further classified in two types:

i) Hardware Requirements

These are the hardware required to run the software. These are mostly same for both Users and Developers.

ii) Software Requirements

These are the software tools required to develop the software and run it on the machine.

Developers need more software specifications than the users.

b)User requirements

These are the expectations of the user from the product. This include the expected output from the software. This output should be clear and very accurate.

2) System Design:

This includes the architecture of the system. This gives an overall idea about the working of the software. There are various diagrams used to describe the system Design. These include UML diagrams like Data Flow diagrams, sequence diagrams, Use case diagrams etc.

The System Design for the Software includes following phases: -

i) extracting frequencies from the input so that the RAGA can be identified easily. This is basically cleaning of the data by removing unwanted frequencies.

ii) Shifting of the scale of frequencies to the base scale so that displacement from base scale can be easily calculated.

iii) Extracting of the Key Phase is done by extracting the prominent part of the input.

iv)Now all the similar phases are clustered into a group and then these groups are labelled.

v)These phrases are then matched with the dataset and system identifies the RAGA Composition and gives proper output.

3) Implementation

The phases of the System Design are implemented by writing the logic of the goal. This is done by writing the instructions in a suitable programming language. Python will be used for this project. All the Components of the design phase are implemented by translating the software Design into Source Code. For Each Phase, a separate module will be created and business logic will be written in it.

4) System Testing

System Testing is done when your project is completely in working state. The unknown inputs are fed to the system and outcomes are checked. The developers give their best to make software robust in nature. If these outputs are wrong then we need to redesign the system and start the development from the start.

System testing ensures that the entire integrated software system met requirements. It tests a configuration to ensure known and predictable results an example of system testing is the configuration-oriented system Integration test System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

i) White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at lost its purpose It is purpose. It is used to test areas that cannot be reached from a black box level

ii) *Black Box Testing*

Black Box Testing is testing the software without any knowledge of the inner working structure or language of the module being tested. Black box tests, most other kinds of tests, must be written from a definitive source document such specification or requirements document.

It is a testing in which the software under test is treated, as a black box you cannot see into it. The test provides inputs and responds to outputs without considering how the software works.

iii) *Unit Testing:*

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

iv) *Integration Testing*

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g. components in a software system or one step up software applications at the company level interact without error.

5) System Deployment

Here, now the system is bug free and is ready to be brought in the market to compete with other competitors. This is where marketing of the product takes place. The end product is deployed in the market environment and User Acceptance Testing is done. Acceptance testing is done by customers along with the developers. It also ensures that the system meets the functional requirements.

6) System Maintenance

The system is now in the market and if any of the customer faces any problem or any bug is encountered, it needs to be rectified. This helps in keeping the software bug-free for its whole lifetime and enhances the user experience.

04. PROPOSED SYSTEM

The proposed system takes in input as audio file, preferably in mp3 extension, and then processes its raw data, and gives a informed result.

For someone to be able to use the software and recognize a song, they must have a song present with them. So, first step is, accepting a file of which you want to recognize the raga.

Now, the basic element of a song is frequency variations. So, the way we can recognize the raga is through a band of frequencies which reflects a particular ragas pakad or bandish.

Hence, the next step would be to extract the frequencies that are in the audio file. Now, when that's done, next problem we face is that there are various pitches of various singers who sing at various frequency bands, so there is a need to check every specific frequency spectrum range or to scale down or scale up the song to a particular frequency range, and thus we get a baseband where we need to look.

Now, the next step is to compare frequencies such that it matches a set of frequencies which belong to one of the ragas for which we have stored the definitions of Bandish frequencies used in that song.

For this processing phase, we need to make use of frequency band in bandish, which would be included in the training phase of the model. The training phase will play an important role in clustering the different sets of frequency spectrums that can be possible for a raga.

With as much dataset that can be possibly provided to the training model, we train it to recognize a raga at all frequency fronts possible for it.

The bandish here referred to is the key phrases in a raga which is with maximum surety will occur in maximum of the songs.

Thus, we need to cluster all these frequency ranges of a ragas into one place. Hence, we make use of clustering to club all those possible outcomes together which will help in training the model with good efficiency.

Next step would now be to make use of the input file, and then, match its key phrases with the clustered keys available and then deriving the best conclusion with the best matched results, thus, getting the required result.

Now, the final step would be to generate output to the user as Raga name. All this is done with a single click by user.

For our proposed system, we expect that frequency extraction software would be needed, there will be need of using machine learning to train the software with dataset for it to be able to generate the result.

For this, the greater the dataset, greater is the efficiency in recognizing the raga successfully.

We also need to implement some clustering algorithm on the dataset, also, we need shift of scale of frequency software so successfully scale up or scale down frequencies successfully.

We need someone in the team with good classical music knowledge to set up different attributes to be needed in the dataset.

05. SYSTEM DESIGN

5.1 SYSTEM ARCHITECTURE:

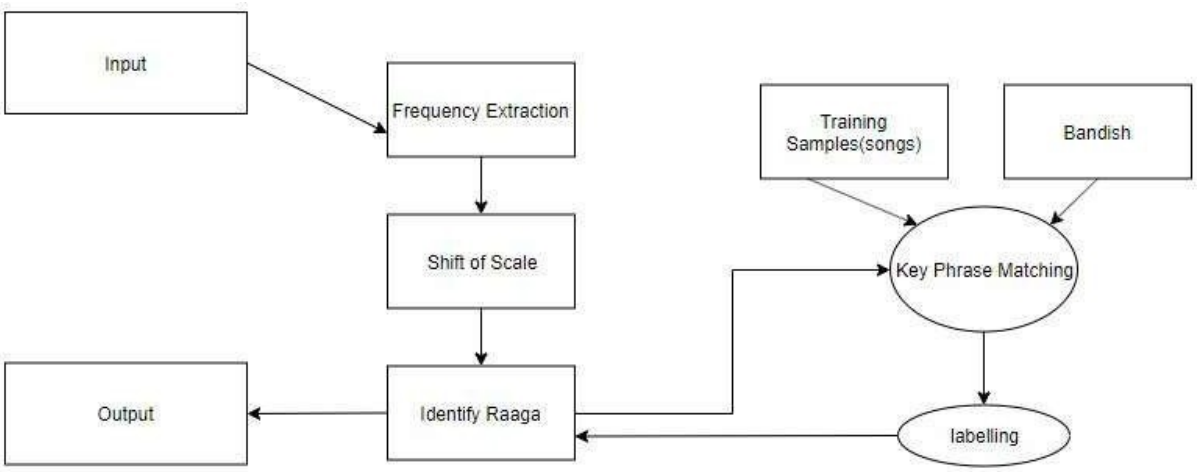


Figure 5.1 System Architecture

System Architecture Diagram

The **Figure 5.1** above is subdivided into following parts:

Input:

The user can give input in the form of mp3 file. Input will consist of songs or a bandish (songs used while rendering the raga) which will be in a standard format. This input should have the phrases similar to a raga in order to detect the raga. A song which is based on some raga has characteristic motifs of the raga which consequently highlights the mood.

Frequency Extraction:

Input is a mp3 file. It will compress the background frequencies of the mp3 file. Output is extracted vocal frequencies. Such vocal frequencies will ease the operations for raga detection as they will provide selective frequencies of local maxima of the frequency graph afterwards. This step is necessary in order to extract patterns from the input song.

Shift of Scale:

Extracted vocal frequencies is the input for this stage. Vocal frequencies extracted are shifted with respect to the base scale. This enables an efficient matching of input with the raga samples. This step is a good start as every song is of a different scale. Output is a file which is used for further comparison.

Key Phrases Extraction:

Input at this stage will be Hindi songs and bandish (songs used in classical music). The key phrases are learned at this stage. These phrases are a combination of notes which are prominently seen in a particular raga.

Clustering:

This is important phase of **Figure 6.1.** Key phrases extracted from the previous stage is the input at this stage. Labelling according to the similarity between phases will be done here.

Key phrases matching:

Phrases from song and key phrases will be matched at this stage. Label of the key phrases will be given as output. Section 4 gives how phrases are important to identify the raga. The phrases detected in the current input will be matched with the earlier inputs. If the similarity found with respect to the earlier files is greater than that raga name will be displayed on the next stage for the user.

Output:

This stage simply shows the user the name of a raga. Raga which influences the input song will be displayed at this stage.

5.2 DATA FLOW DIAGRAM:

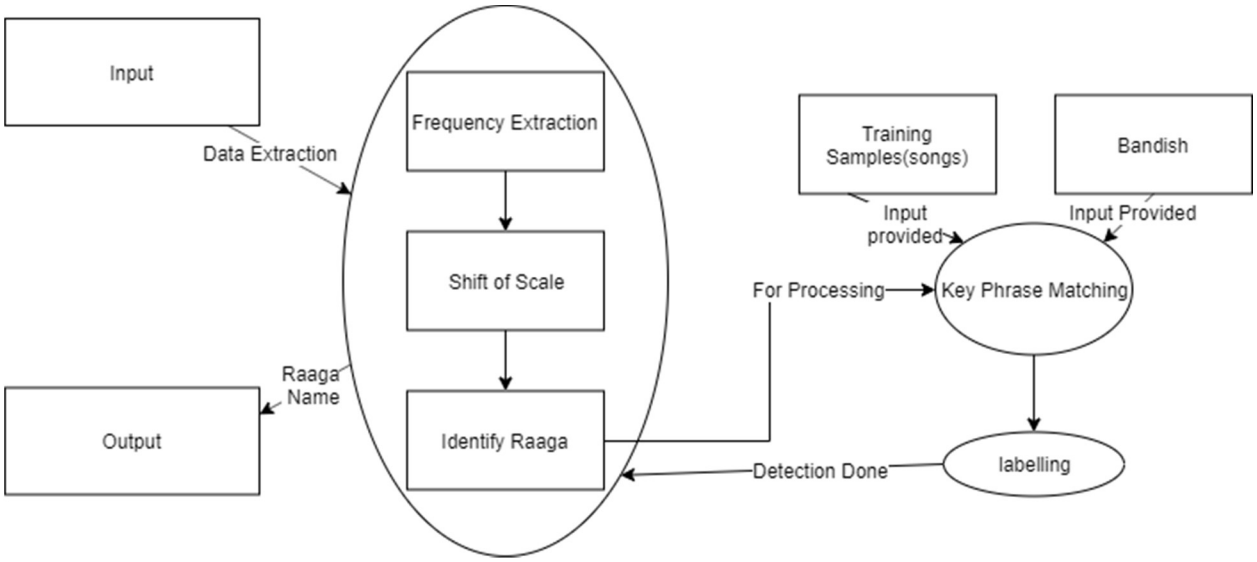


Figure 5.2 Data flow diagram

The user can give input in the form of mp3 file. Input will consist of songs or a bandish (songs used while rendering the raga) which will be in a standard format. This input should have the phrases similar to a raga in order to detect the raga. A song which is based on some raga has characteristic motifs of the raga which consequently highlights the mood. Input is a mp3 file. As seen from the diagram above, Output is data extracted in the form of vocal frequencies. This step is necessary in order to extract patterns from the input song. Extracted vocal frequencies is the input to Shift of Scale stage. Vocal frequencies extracted are shifted with respect to the base scale. This enables an efficient matching of input with the raga samples. This step is a good start as every song is of a different scale. Output is a file which is used for further comparison. Input at this stage will be Hindi songs and bandish which will be provided to this module during training phase to get it trained to identify raga successfully. **In Figure 5.2 ,** the key phrases are learned at this stage. These phrases are a combination of notes which are prominently seen in a particular raga. Key phrases of the input song is matched with the module trained and it identifies the raga, then the name is associated to the raga at the labelling phase and that output is sent to user.

The similar phrases will result to give similar labels for input files. Phrases from song and key phrases will be matched at this stage. Label of the key phrases will be given as output. Section 4 gives how phrases are important to identify the raga. The phrases detected in the current input will be matched with the earlier inputs. If the similarity found with respect to the earlier files is greater than that raga name will be displayed on the next stage for the user. This stage simply shows the user the name of a raga. Raga which influences the input song will be displayed at thisstage.

5.3 USE CASE DIAGRAM:

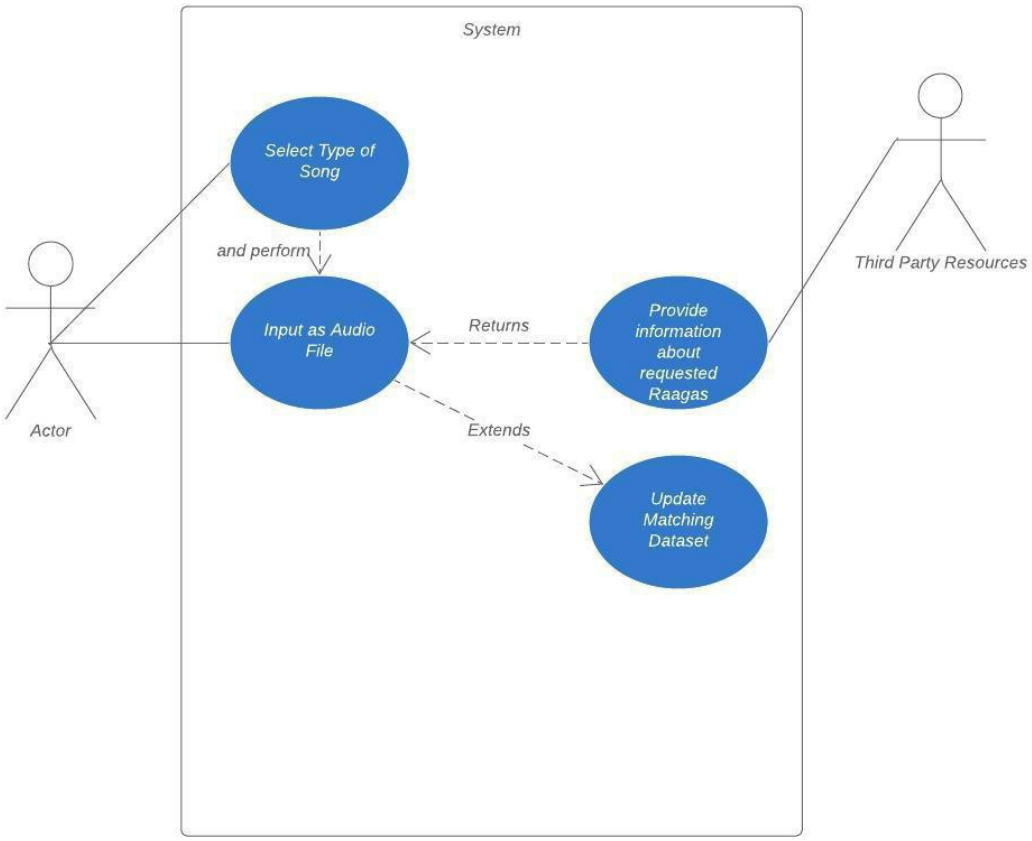


Figure 5.3 Use case diagram

As evident from the **Figure 5.3**, actor i.e. User has two actions to perform namely to select type of a song and accordingly provide an input to the system in audio file format. The input given by the user will be tested against the trained model and will be saved in the dataset for the future reference. When system processes the input and figures out which type of raga is this, the further information about it is extracted from third party resources. The extracted information is then returned to user on the console itself.

5.4 SEQUENCE DIAGRAM:

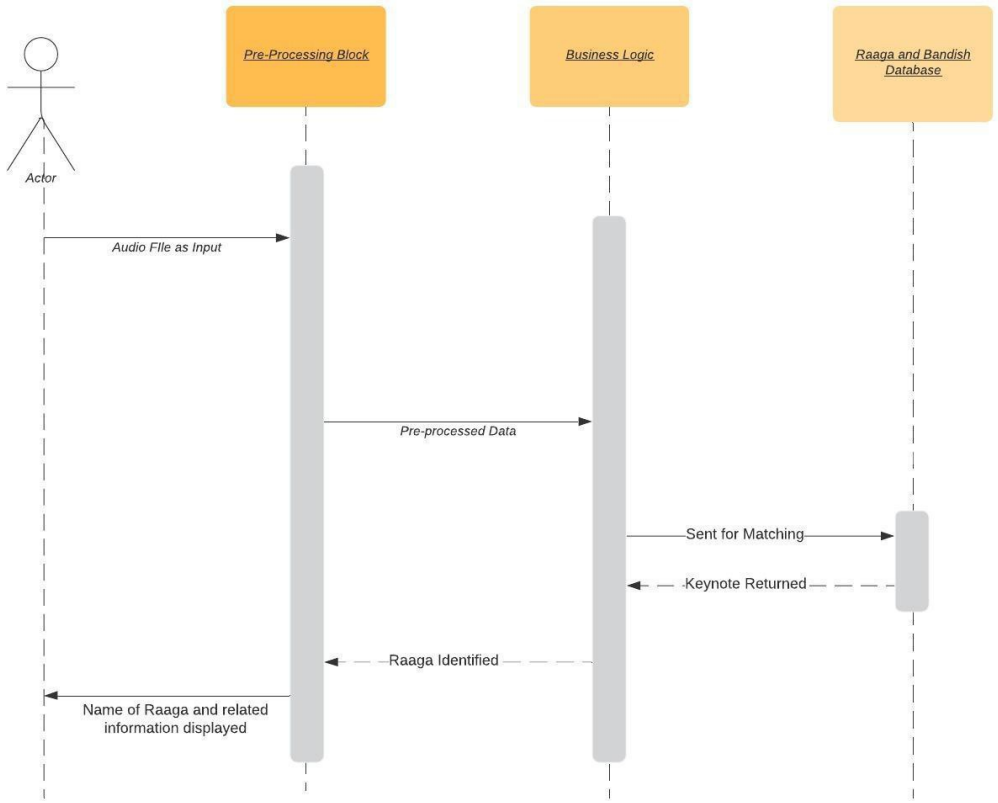


Figure 5.4 Sequence diagram

The **Figure 5.4** shows that actor i.e. user gives his audio file input to system. It is then treated in pre-processing block. Pre-processing block takes care of converting audio input and forwards it to main business logic. Business logic
i.e. main function block does the work of frequency extraction which will be further used on trained model to get desired output. Finally, model works on input given by previous phase and identifies the correct "Ragas". This is done by keynote matching. The generated output is returned to actor/user. As already mentioned, the related information of that specific Ragas is extracted from third party resources like dedicated music related websites, and collectively they are displayed on the user end/console.

06. PROJECT PLAN

6.1 PROJECT ESTIMATE

6.1.1 Reconciled Estimates

6.1.1.1 Cost Estimate

• Lines of code (LOC) is 700 (Approx.)
• LOC in KLOC (Kilos LOC) is 0.7 KLOC

Effort:

The Effort is calculated by formula

E = 2.4*(KLOC)^1.05
E = 2.4*(0.7) ^1.05
E = 1.65 Approx.

5.1.1.2 Time Estimates:

The Development Time is Calculated by formula

D=2.5*(E)^0.38
D= 2.5*(1.65) ^0.38
D=3.02 Approx.

5.1.2 Project Resources:

Hardware Resources Required:

• Processor	-	Pentium IV/Intel 13 core
• Speed	-	1.1 GHz
• RAM	-	512 MB (min)
• Hard Disk	-	20GB
• Keyboard	-	Standard keyboard
• Mouse	-	Two or Three Button Mouse
• Monitor	-	LED Monitor

Software Resources Required:

- Operating System - Windows
- Front End - HTML , CSS , FLASK
- Back End - PYTHON
- IDE - PYCHARM , JUPYTER NOTEBOOK

6.2 RISK MANAGEMENT W.R.T NP HARD ANALYSIS:

- The identification of Risk is central to the success and failure of the project, hence I have made a concentrated effort to minimize and even eliminate certain risk related Software risk could be classified into categories. Internal and External risk, those risk which arise from the risk factor within the organization can be defined internal risk and the risk coming from outside is called external risk Internal risk avoidance can be done by clear picturing the process, product risk.

6.2.1 Risk Identification:

For risks identification, review of scope document, requirements specifications and schedule is done. Answers to questionnaire revealed some risks. Each risk is cate- gorized as per the categories mentioned in [**?**]. Please refer table 5.1 for all the risks. You can refereed following risk identification questionnaire.

1. Have top software and customer managers formally committed to support the project?
2. Are end-users enthusiastically committed to the project and the system/product to be built?
3. Are requirements fully understood by the software engineering team and its customers?
4. Have customers been involved fully in the definition of requirements?
5. Do end-users have realistic expectations?
6. Does the software engineering team have the right mix of skills?
7. Are project requirements stable?

8. Is the number of people on the project team adequate to do the job?

9. Do all customer/user constituencies agree on the importance of the project and on the requirements for the system/product to be built?

6.2.2 Risk Analysis:

The risks for the Project can be analyzed within the constraints of time and quality

ID	Risk Description	Probability	Impact		
			Schedule	Quality	Overall
1	System Failure	Low	Low	High	High
2	Connection Failure	Low	Low	Low	Low

Table 5.1: Risk Table

Probability	Value	Description
High	Probability of occurrence is	> 75%
Medium	Probability of occurrence is	26 – 75%
Low	Probability of occurrence is	< 25%

Table 5.2: Risk Probability definitions

Impact	Value	Description
Very high	> 10%	Schedule impact or Unacceptable quality
High	5 – 10%	Schedule impact or Some parts of the project have low quality
Medium	< 5%	Schedule impact or Barely noticeable degradation in quality Low Impact on schedule or Quality can be incorporated

Table 5.3: Risk Impact definitions

6.2.3 Overview of Risk Mitigation, Monitoring, Management:

The identification of Risk is central to the success and failure of the project, bence1 have made a concentrated effort to minimize and even eliminate, Certain risk related Software nak could be classified into categories Internal and External risk, those risk which arise from the risk factor within the organization can be defined internal risk and the risk coming from outside is called external risk. Internal risk avoidance can be done by clear picturing the process, product risk.

Following are the details for each risk:

Risk ID	1
Risk Description	System Failure
Category	Requirements
Source	Identified during early development and testing
Probability	Low
Impact	High
Response	Accept
Strategy	internet connectivity must be better
Risk Status	Occurred

Risk ID	2
Risk Description	File Upload Failure
Category	Testing
Source	Early testing
Probability	High
Impact	High
Response	Mitigate
Strategy	Better testing will resolve this issue.
Risk Status	Identified

6.3 PROJECT SCHEDULE

6.3.1 PROJECT TASK SET:

Major Tasks in this section are:

Task 1: Requirement Analysis (Base Paper Explanation).

Base paper is thoroughly analyzed to find out the details of each and every method and process used in the paper. This helps in understanding the domain and we gain knowledge about the latest advancement in the domain of our project.

Task 2: Project Specification (Paper Work).

The domain of the project should be specified. The complete information of the project along with its working should be explained here.

Task 3: Technology Study and Design.

This includes the study of the technology that we will be using for the project and total system design of the project.

Task 4: Coding and Implementation (Module Development).

The codes are written in the files in separate modules. The code is tested to verify the working of the project.

6.3.2 TASK NETWORK:

Individual tasks and subtasks have interdependencies based on their sequence A task network is a graphic representation of the task flow for a project. Project tasks and their dependencies are noted.

6.3.2 TIMELINE CHART:

Activity	I week	II **week**	III **week**	IV **week**	V **week**	VI **week**	VII **week**	VIII **week**	IX **week**	X **week**	XI **week**
	AUG 4	AUG 11	AUG 18	AUG 25	SEPT 1	SEPT 8	SEPT 15	SEPT 22	OCT 1	OCT 8	OCT 11
Initiate the project											
Communication											
Literature survey											
Define scope											
Develop SRS											
Plan the project											
Design mathematical model											
Feasibility Analysis											
Planning schedule											
Design UML and other diagrams											

Table No. 6 Time-line Chart

Activity	XI week	XII week	XIII week	XIV week	XV week	XVI week	XVII week	XVIII week	XIX week	XX week	XXI week	XXII week
	Jan 5	Jan 15	Jan 19	Jan 26	Feb 2	Feb 9	Feb 16	Feb 23	Mar 2	Mar 9	Mar 16	April 25
Documentation												
Execute the Project												
Preprocessing and Creating the knowledge base.												
Training on different models to check the accuracy.												
Build and test basic functional unit												
Web Designing and UI integration												

Table No. 7 Project Implemntation Timeline

6.4 TEAM ORGANIZATION:

6.4.1 Team structure:

Harsh Apte:

Studied the previous work done in similar domains as a part of literature survey to derive the substantive findings which has been used for the progress of this project. Worked on the business logic of the project which includes pattern matching and recognition part for identification of the raaga. Also, developed a classification and clustering control flow for the final raaga recognition while displaying it to the user of the system. Assisted for the documentation part of the same.

Savani Gokhale:

Worked as a Domain Expert for the Indian Classical Music for developing the functional logic required for recognition of audio pieces. Created a unique dataset required for this particular project. Carried out the Data Pre-Processing and frequency extraction phases using python audio processing techniques. Contributed in documentation by describing the details of the algorithms used with the help of diagrams.

Sanket Arekar:

Tested various models suitable for detecting raaga and classification of various samples. The Accuracy of various models was observed and most suitable model was chosen. Extracted the significant features of the data available. Used various models to classify the similar pattern in the samples. Studied the Requirement Analysis and Development Model for the Project and prepared the documentation for the same. Prepared the timeline for the project. Beautified the Front-End UI using CSS and Bootstrap.

Sarvajit Sankar:

Developed the Front-End UI and the connectivity feature between front end and Back End. Developed the file Upload feature which restricts files of only type .wav audio format. Prepared the gap analysis for the research papers analyzing the advantages and efficiency of each paper in consideration. Did some parts of documentation referring to previous year project report documentation methods and wrote the mathematical model of the software in consideration.

6.4.2 Management reporting and communication:

Mechanism for progress reporting and inter/intra team communication are identified as per assessment sheet and lab time table.

Sr No.	Month	Description
1	June	• Description Discussion with guide regarding domain • Searching for IEEE paper for domain
2	July	• Short listing of IEEE papers within domain • Selection of IEEE paper
3	August	• Deciding Project name • Submission of Synopsis.
4	September	• Requirement analysis. • Designing of models
5	October	• Report preparation • Stage-l report submission

Table No.8 Management plan

07. PROJECT IMPLEMENTATION

7.1 OVERVIEW OF PROJECT MODULES:

This is one the important phases as the architecture of the system is designed in this phase. Analysis is carried out and depending on the analysis a software model is designed Different models for developing software are created depending on the requirements gathered in the first phase and the planning done in the second phase. In this phase we discuss about admin and user modules as well as functionality of that modules and implement this module.

7.2 TOOLS AND TECHNOLOGIES USED:

TOOLS USED ARE:

IDE – PyCharm , Jupyter Notebook

TECHNOLOGIES USED ARE:

FRONTEND – HTML , CSS , FLASK

BACKEND – PYTHON 3

7.3 ALGORITHM DETAILS:

7.3.1 ARTIFICIAL NEURAL NETWORK: ANN

1. Artificial neural networks (**ANN**) are computing systems are influenced by the neural networks which form the animal brain. These systems learn without pre specified or pre-programmed rules. For example, in case of image processing, ANN learn through analysing the input images which are usually labelled as, let's say "cats" and "not cats". Through this knowledge, these systems can identify a image having cat. An ANN is based on a cumulative connected units or nodes called artificial neurons, which are similar to the neurons in a biological brain. An artificial neuron receives a signal, processes it and can signal neurons connected to it.

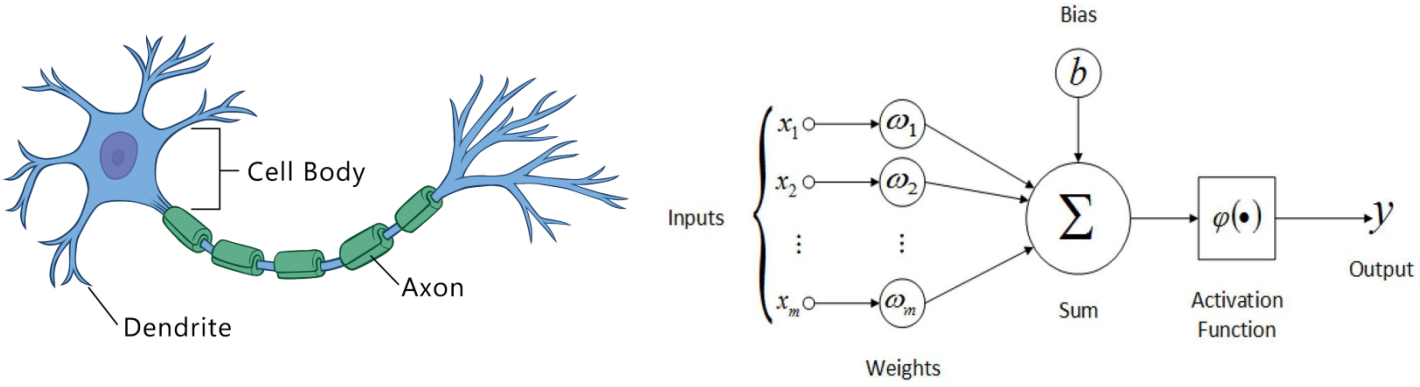
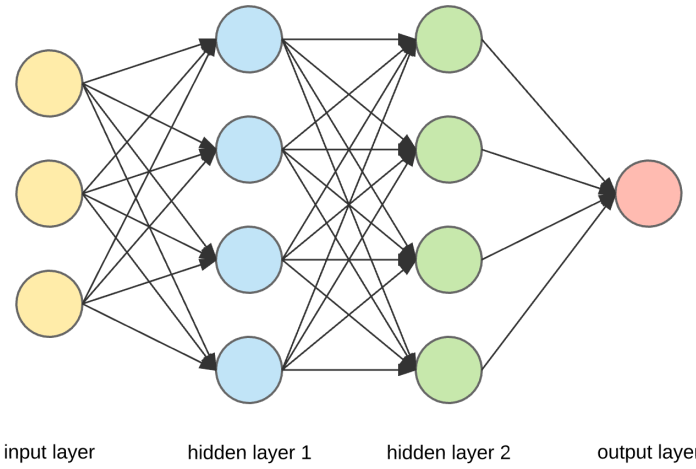


Figure 7.1 ANN



Artificial Neural Network consists multiple layers of neurons, which are also called as simple processors or nodes. These neurons are connected by links which pass signals to one neuron to another. These links are usually weighted. The outgoing connection can result in one output node or into a number of nodes. Perceptron is a primitive form of neural network.

Learning in ANN :

Three types of learning :

1. Supervised Learning
2. Un-Supervised Learning
3. Reinforcement Learning

Types of ANN :

1. Feed Forward Neural Network
2. Recurrent Neural Network

Advantages of ANN :

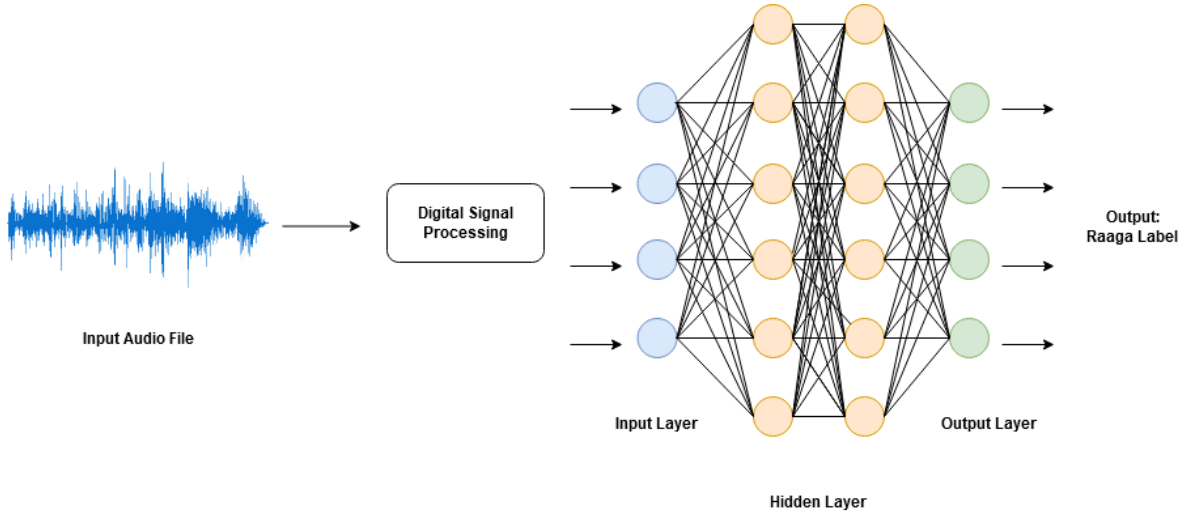
- It is highly non-linear modelling.
- It is easy to understand when compared to statistical methods.
- ANN does not need higher background of statistics.
- Back propagation can be used to solve various classification as well as forecasting problems.

Applications of ANN :

- Image Recognition
- Character Recognition
- Stock Market Prediction
- Travelling Salesman Problem
- Speech/Audio Recognition

Among the numerous conventional approaches of pattern recognition, the statistical approach is well established and is used for the real life problems. Recently, artificial neural network techniques are favoured because of their non-linear nature. Recognition is highly dependent on factors such as, definition of pattern classes, sensing environment, feature extraction and selection, pattern representation, cluster analysis, classifier design and learning and performance evaluation.

In Raaga Recognizer, ANN is used as follows :



ANN for Raaga Recognition

Fig 7.2 Use of ANN in Kaansen

The model requires an audio file as input. The audio file is pre-processed before feeding it to the ANN. The model gives output in the form of Raaga Label.

7.3.2 CONVOLUTIONAL NEURAL NETWORK: CNN

Convolutional Neural Networks (CNN) taking input as a spectrogram in the form of image and different structures are searched in these images. A spectrogram is a record generated by a sound spectrograph with time shown along the X-axis, frequency shown along the Y-axis, and intensity shown by different shades of darkness of the pattern. These structures are used for finding similar patterns in various musical spectrogram samples.
The Spectrogram contains the vital information about the features of the musical sample. These features get reflected in the spectrogram image and thus the pattern in the image has collinearity with the features. Thus extracting features from image helps in audio processing.

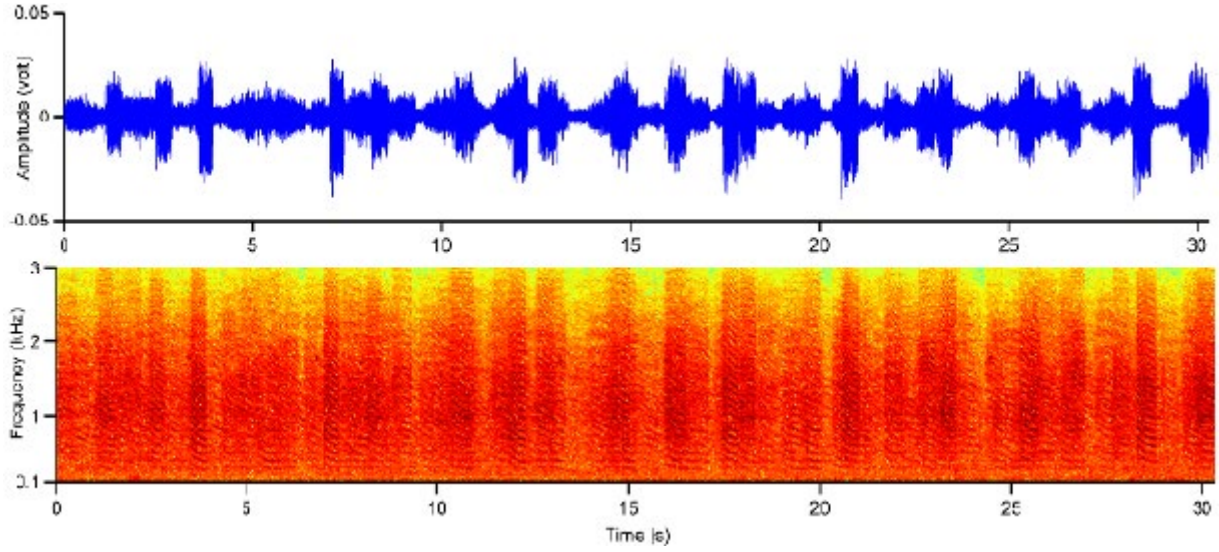


Figure 7.3 Amplitude to Frequency Conversion

Convolutional Neural Networks (CNN) are very similar to ordinary Neural Nets consisting of neurons that have weights and biases. Each neuron gets some input, performs a dot product(scalar multiplication) and follows it with a non-linearity, if necessary. The whole network still results in a single distinguishable score function: from the image pixels on one end to class scores on the other. There is a Loss function in the Final Layer.
CNN architectures make the definite assumption that the inputs are images, which explore and discover certain properties into the design. These then make the function more effective to implement and largely reduce the parameters in the network.
The CNN is able to identify the primary features in the image, then these attributes are passed on to the further layers and these layers find more complex features based on the information provided by the previous layers. As we proceed layer after layer more complex and novel features are discovered and these features play an important role in differentiating the samples.
Lot of features are discovered in the process, but only a few important or significant ones are selected for classification. Such features have a very large impact on the decision making. Our dataset consists of audio tracks each upto 40 seconds long. The tracks have the same Raagas but in the form of different frequencies. These audio files are in .wav format.

A filter is denoted by an array of weights with which we convolve the input. The filter, similar to a filter in signal processing, gives a measure for how close a part of input resembles a feature. This is then fed to the Maps, which is the immediate next layer. Maps are the representations of the input. The number of Think of filter like a membrane that allows only the desired qualities of the input to pass through it.
concatenate maps cnn.
A pooling operation called Max pooling that selects the maximum element from the region of the feature map covered by the filter. Thus, the output of max-pooling layer would be a feature map containing the most prominent features of the previous feature map.

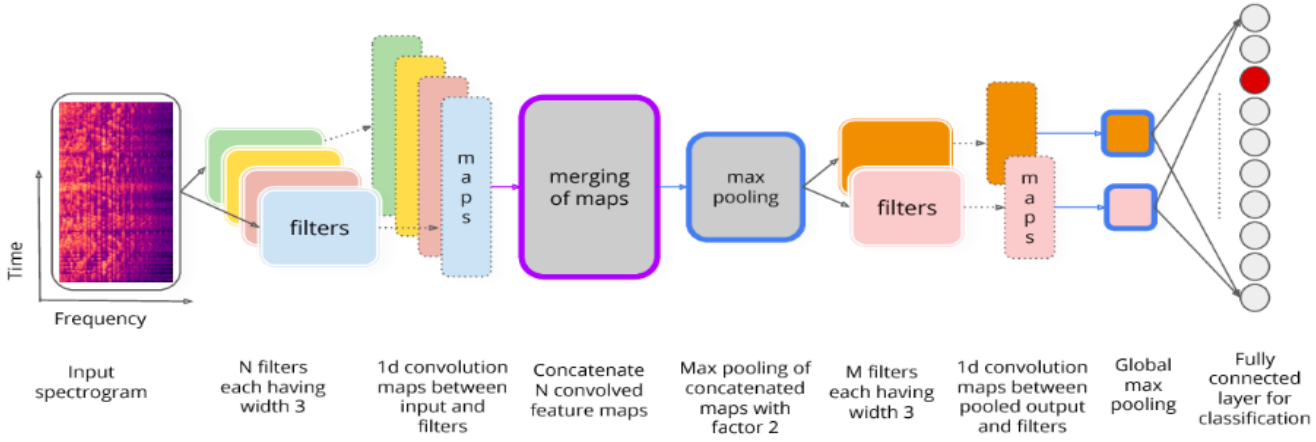


Figure 7.4 CNN

This CNN model can be combined with other models to give the desired results. This combination further increases the accuracy of the system.

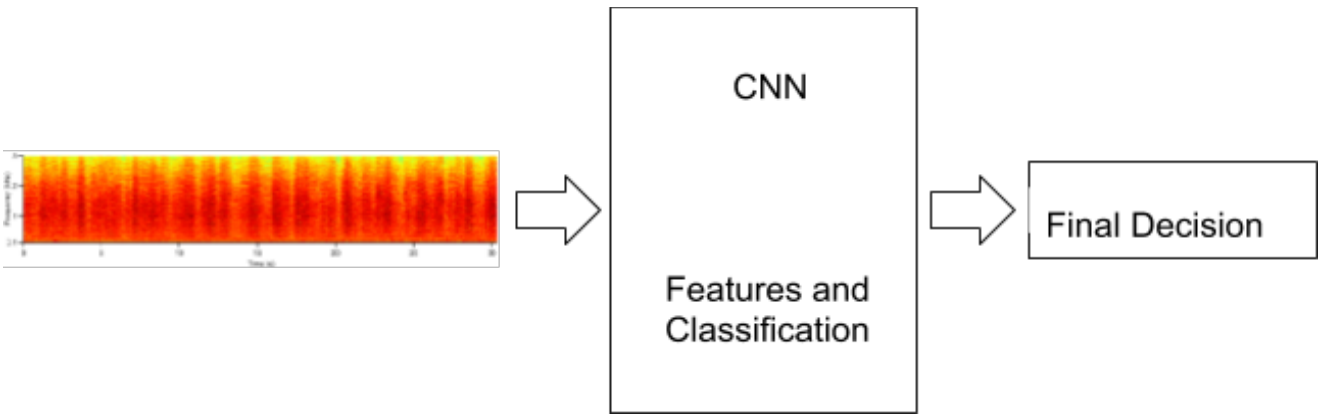


Figure 7.5 Use of CNN in Kaansen

This CNN model can be combined with other models to give the desired results. This combination further increases the accuracy of the system.

7.3.3 HIDDEN MARKOV MODEL(HMM):

Hidden Markov Model (HMM) is based on Markov Chains. Markov chain is a type of random process having states where probability of the current state depends on the previous state only and not on other prior states. The Markov chains remain hidden and cannot be observed. Each state generates a random one out of k observations which is visible.

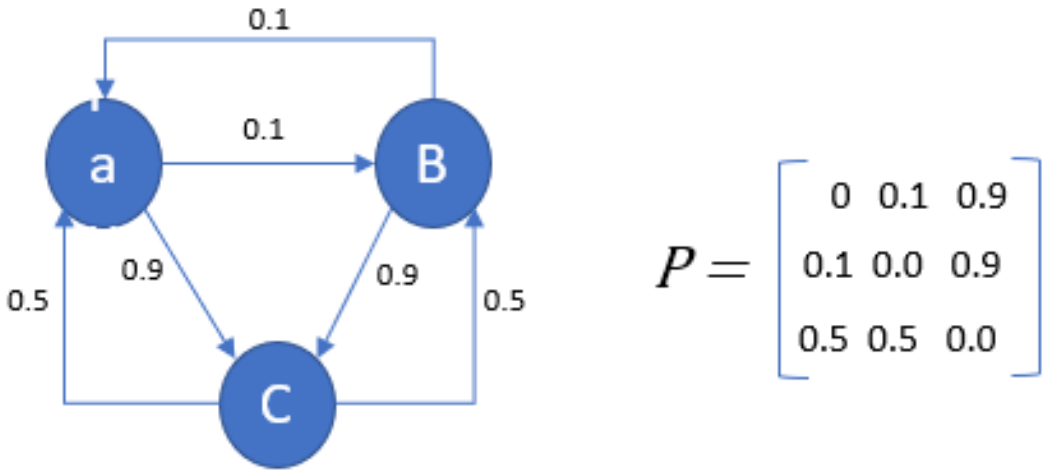


Figure 7.6 Probability Array in HMM

HMM is the type of probabilistic graphical model which help us to predict the sequence of unknown variables from observed variables. The reason of calling it hidden as it is an inference model based on Markov process.

HMMs are statistical models. We can compute the joint probability of a set of hidden states given a set of observed states. The hidden states are also called as latent states. The best possible sequence can be predicted by joint probability of the hidden states.

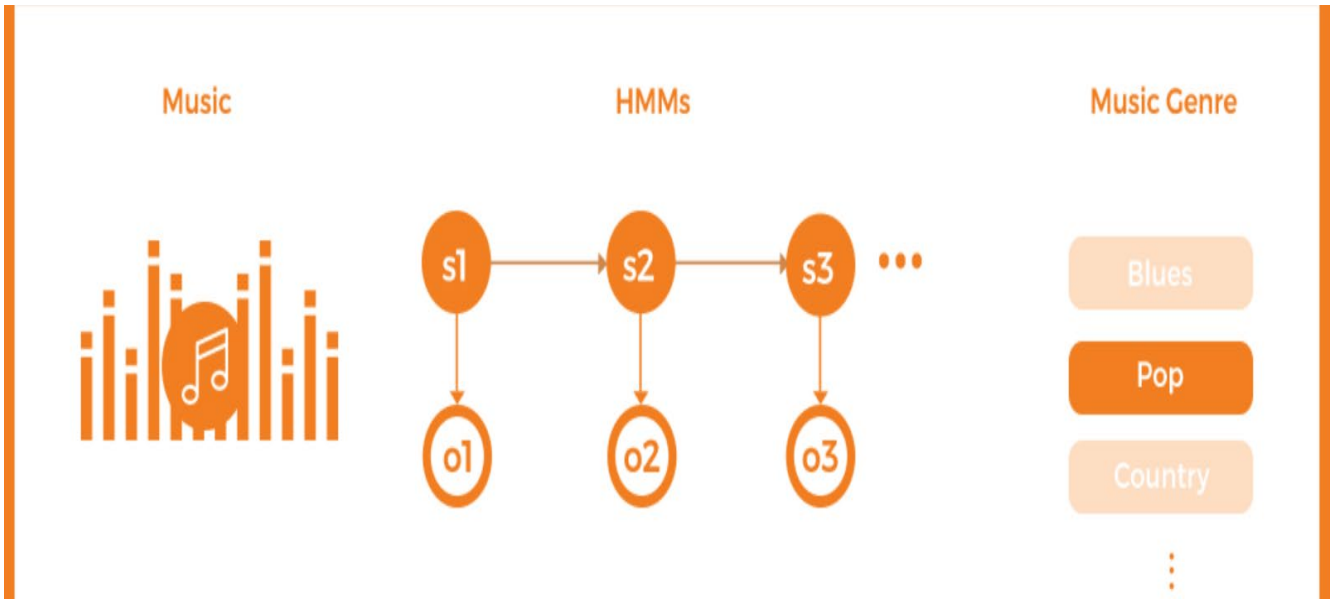


Figure 7.7 Hidden Markov Model Working

The following three aspects are need to be considered before finding the joint probability of the hidden states.

1. **Transition data**:
   Probability of transitioning into a different state based on present state.

2. **Emission data**:
   Probability of transitioning to an observed state based on hidden state.

3. **Initial state information:**

   Initial probability of transitioning into a hidden state.

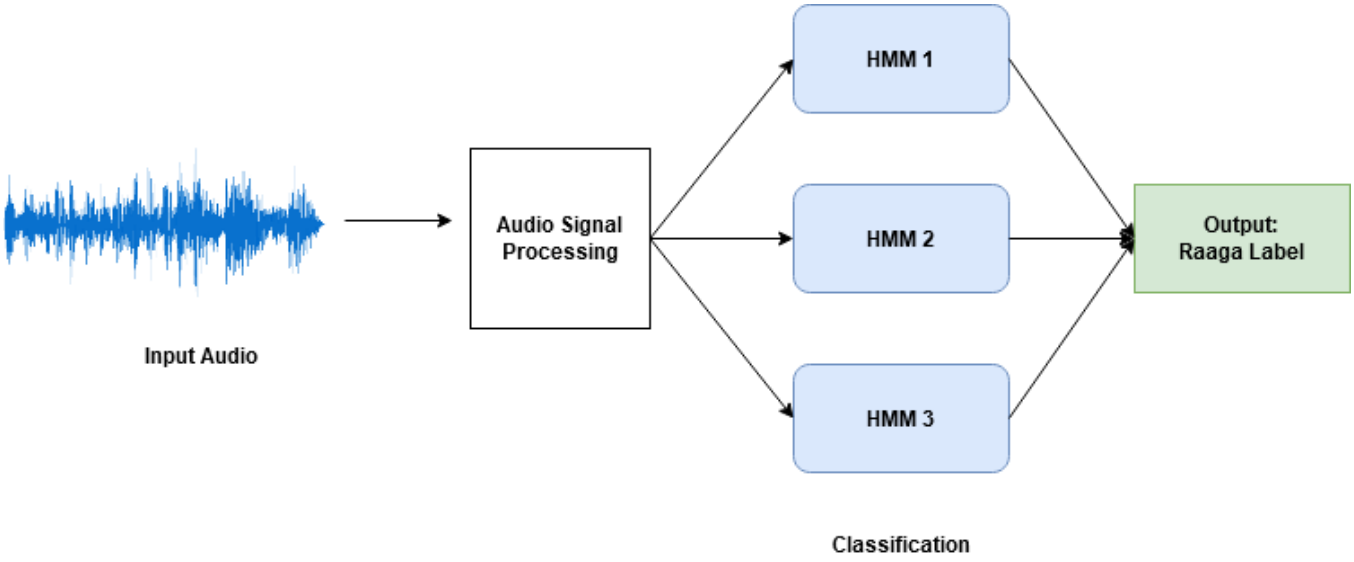


Figure 7.8 Use of HMM in Kaansen

7.3.4 K MEANS CLUSTERING:

Clustering is one of the most common exploratory data analysis technique used to get an intuition about the structure of data. It can be defined as the task of an identifying subgroups in the data such that data points in the same subgroup (cluster) are very similar while data points in different clusters are very different from each other. In other words, we try to find homogeneous subgroups within the data such that data points in each cluster are as similar and as possible according to a similarity measure such as euclidean-based distance or correlation-based distance. The decision of which similarity measure to use is application-specific.

Clustering analysis can be done on the basis of features where we try to find subgroups of samples based on features or it can also be done on the basis of samples where we try to find subgroups of features based on samples. We'll cover here clustering based on features. Clustering is used in market segmentation; where we try to find customers that are similar to each other whether in terms of behaviors or attributes, image segmentation/compression; where we try to group similar regions together, document clustering based on topics, etc.

Unlike supervised learning, clustering is considered an unsupervised learning method since it doesn't have the ground truth to compare the output of the clustering algorithm to the true labels to evaluate its performance. We only want to try to investigate the structure of the data by grouping it into distinct subgroups.

Kmeans algorithm is iterative algorithm which tries to partition the dataset into *K*pre-defined distinct non-overlapping subgroups (clusters) where each data point belongs to **only one group**. It tries to make the inter-cluster data points as similar as possible while also keeping the clusters as far from each other. It assigns data points to a cluster such that the sum/addition of the squared distance between those data points and the cluster's centroid (arithmetic mean of all the data points that belong to that cluster) is at found to be the minimum. The less variation we have within clusters, the more homogeneous (similar) the data points are within that cluster.

The way kmeans algorithm works is as follows:

1. Specify number of clusters *K*.
2. Initialize centroids by first shuffling the dataset and then randomly selecting *K* data points for the centroids without replacement.
3. Keep iterating until there is no change to the centroids. i.e assignment of data points to clusters isn't changing.

- Compute the sum of the squared distance between data points and all centroids.
- Assign each data point to the closest cluster (centroid).
- Compute the centroids for the clusters by taking the average of the all data points that belong to each cluster.

The approach kmeans follows to solve the problem is called **Expectation-Maximization**. The E-step is assigning the data points to the closest cluster. The M-step is computing the centroid of each cluster. Here in the raaga recognizer we'll be using the clustering technique for combining the outputs given by ANN, HMM and pattern matching & recognition models into specific clusters assigned for each particular raaga. This can be best described by the following figure.

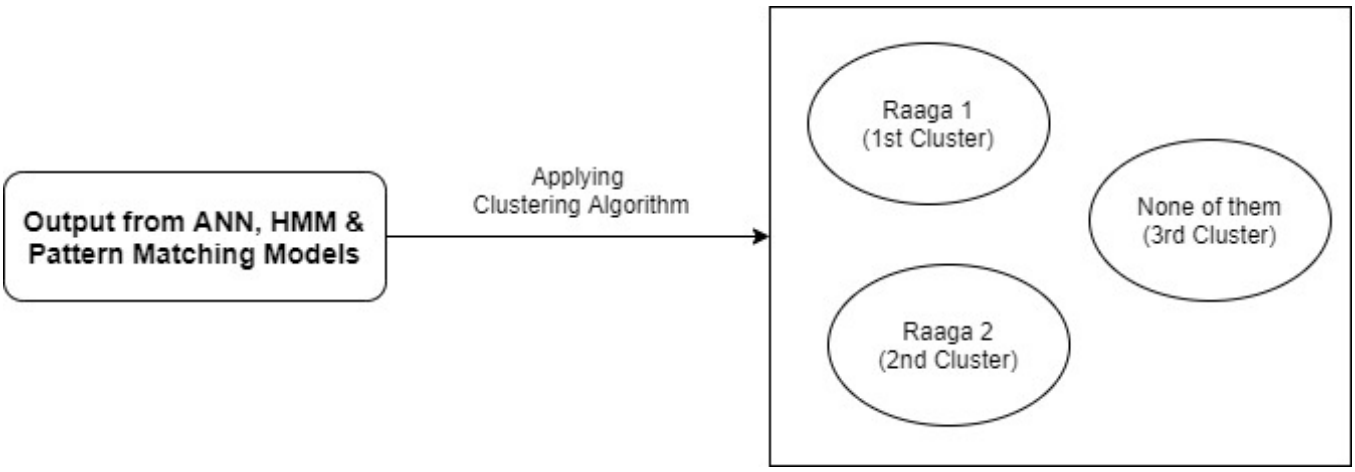


Figure 7.9 Use of K-means Clustering Kaansen

08.SOFTWARE TESTING

8.1 TYPES OF TESTING:

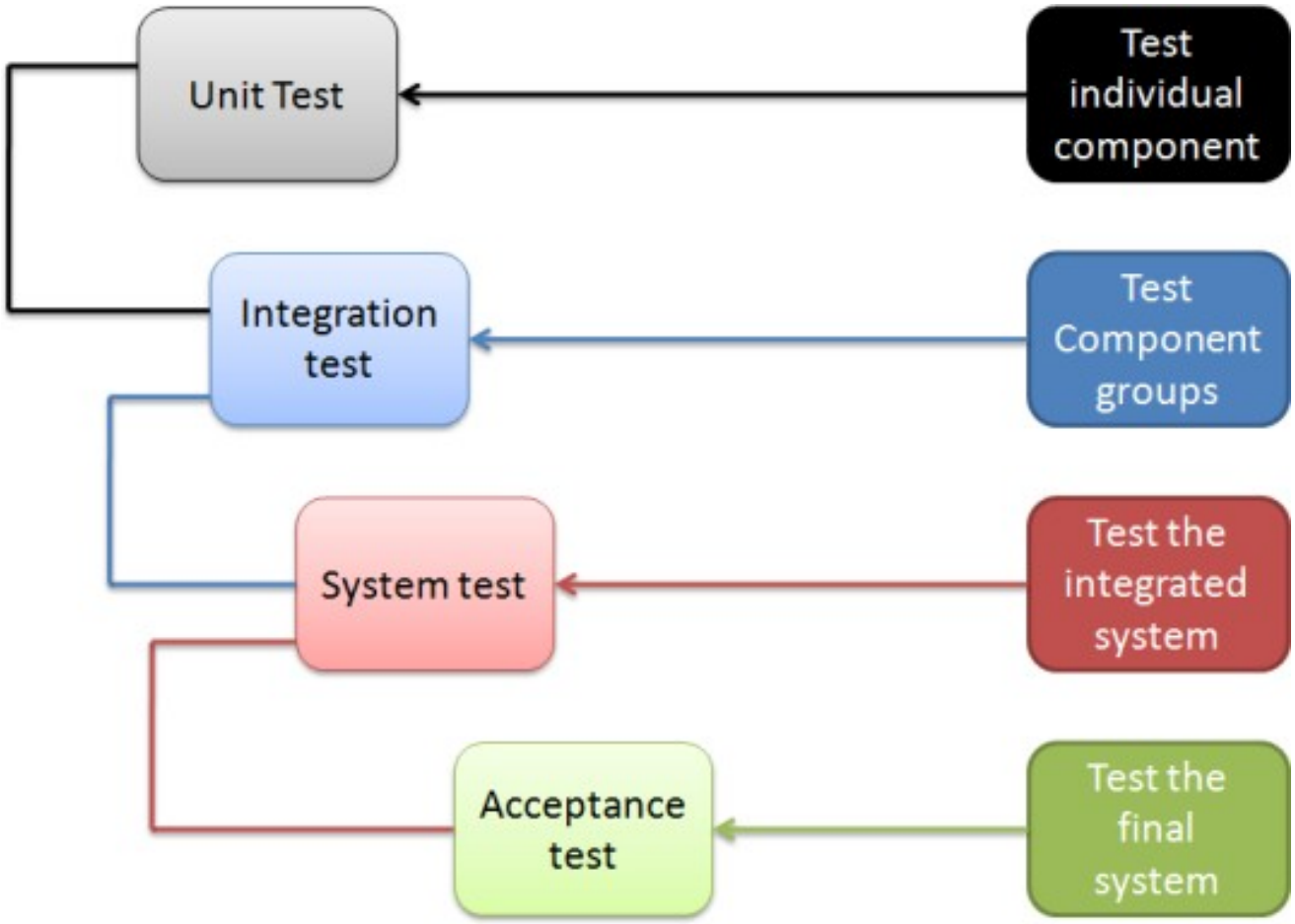


Figure 8.1 System Testing

A. Unit Testing –

- Unit testing is a type of software testing where each and every component of model is going to be tested. Mainly unit testing is performed at a development that is at the time of coding of the application. In software development life cycle, unit testing is always performed at the initial level. The main objective of this testing is to isolate each and every component of an application.

- Sometimes developer or tester skips unit testing to save time but it leads to higher defect fixing. Because unit testing improves the software by removing the defects at initial level. Hence unit testing is always performed at initial stage that is coding of an application or development of an application module. The main advantage of a unit testing is that we can test the modules of an application without waiting for other modules. Unit testing tools are available in market like Junit, Jtest, NUnit, EMMA, PHPUnit,etc.

B. Integration Testing-

- Integration is a second level of software testing where all individuals' units are integrated or combined and testing is performed on that combined units. Integration testing uses test drivers and test stubs for assisting. Integration testing plan is of 4 steps which includes prepare, review, rework and baseline Integration testing is performed by developers or individual tester. There are 4 approaches of integration testing which are Big Bang approach, Top Down approach, Bottom Up approach and Hybrid or Sandwich approach. For integration testing interaction between all units should be clearly defined. If information about this interaction is clearly defined the performance of integration testing will be degraded.

- Hybrid approach is combination of top down and bottom up approach. In bottom up approach, bottom level units are tested first and then testing is performed at upper level unit step by step after that. In top down approach, top level units are tested first and the lower level units are tested step by step after that. Big bang approach is performed when software is available as a bundle so that testing is performed in one go.

C. System Testing-

- System testing is performed at third level of software testing in which the complete application of software is tested. The main objective of system testing is to test that the integrated system is meeting the requirements of user or not. On the basis of system testing the deployment process software is decided System testing is performed by individual testers. System testing is a series of software testing whose main point is to test the complete software again and again

- There are 50 types of system testing which are Usability testing, Load testing. Regression testing, Recovery testing. Migration testing, Hardware testing Software testing. Functional testing. The type of system testing is selected based on the factors like who the tester works for time available for testing, software tester's knowledge, resources available to the user and the testing budget

- Usability testing tests the ability of system that it will meet the criteria or not. Load testing checks that software will be able to work in real life examples or not. Regression testing is actually the selection of already executed test cases which are re-executed again and again until all the defects are removed from system Recovery testing is used to check that system is really reliable, accurate, trustworthy and really able to satisfy the need or requirements of user in the day to day life.

- Functional testing is basically the addition of functionalities that should be added by developer or tester so that it will help to improve the performance of the system, Migration testing is used to check that software will able to perform on more than one machines that is it will have the ability of portability Hardware and software testing is mainly focuses on how the software and hardware will interact between each other and helps to check the coupling between hardware and software System testing is performed by professional testing agents.

- System testing is performed as a initial state in software development life cycle during which system is tested as a whole it enables to test, validate and verify both the application architecture as well as business requirements System testing is usually carried out by a team that is independent of the development environment.

D. Acceptance Testing –

- Acceptance testing is type of beta testing performed on product by the actual end users. Acceptance testing is testing is performed at the end of product deployment which is tested for acceptability of the software product.

- There are two types of acceptance testing such as internal acceptance testing and external acceptance testing. Internal acceptance testing is also called as alpha testing which is performed by the members of an organization who are not completely involved in building of software. External software testing is performed by the people who are not the employees of an organization they are actual end users which are going to use the product in their day to day life.

8.2 TEST CASES AND TEST RESULTS:

Test Case ID	1
Test Case Description	Checking Functionality of Upload option
Steps	1. Click upload button
Test Case Result	OS File Manager must open to choose a file
Action Result	File Manager Displayed
Status	Pass

Test Case ID	4
Test Case Description	Checking Functionality of File Manager to Only upload .wav files
Steps	1. Click upload button 2. Check to see if you can upload any other file
Test Case Result	No other type of file should be able to be uploaded
Action Result	Only wav files are displayed to be uploaded
Status	Pass

Test Case ID	3
Test Case Description	Checking Functionality of Submit button
Steps	1. Click upload button 2. Upload a File 3. Click on submit
Test Case Result	New page must be displayed stating that file uploaded successfully
Action Result	Message: File Uploaded Successfully
Status	Pass

Test Case ID	2
Test Case Description	Checking Functionality of Submit button
Steps	1. Click upload button 2. Upload a File 3. Click on submit
Test Case Result	New page must be displayed stating that file uploaded successfully
Action Result	Unable to Connect
Status	Fail
Problem	POST method not declared in APP ROUTE in FLASK
Solution	Mention methods as POST

Test Case ID	5
Test Case Description	Checking Functionality of Submit button
Steps	1. Click upload button 2. Upload a File 3. Click on submit 4. File must be stored in your local code repository under audio folder
Test Case Result	Audio files must stored under the audio folder
Action Result	Audio files stored under the audio folder
Status	Pass

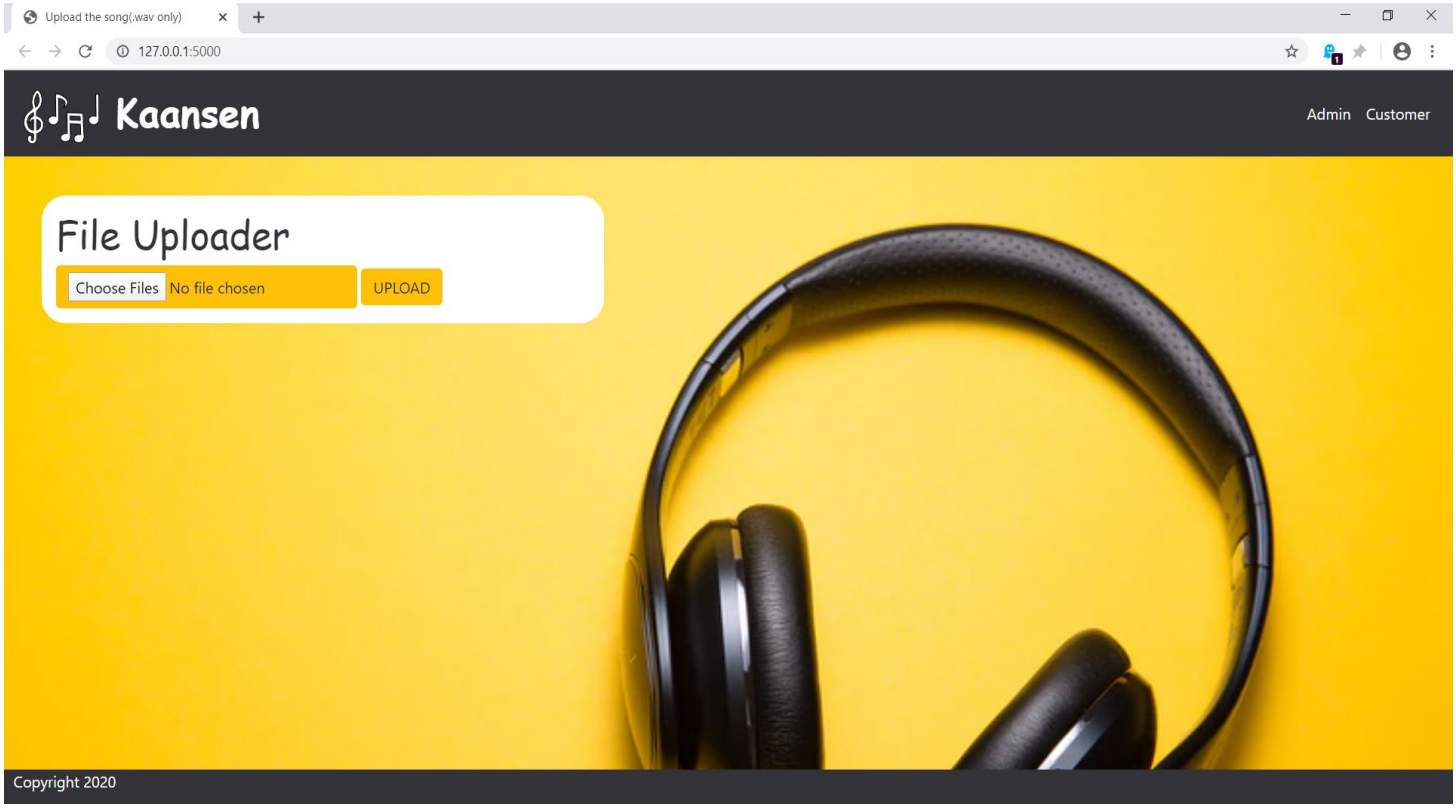
09. RESULTS

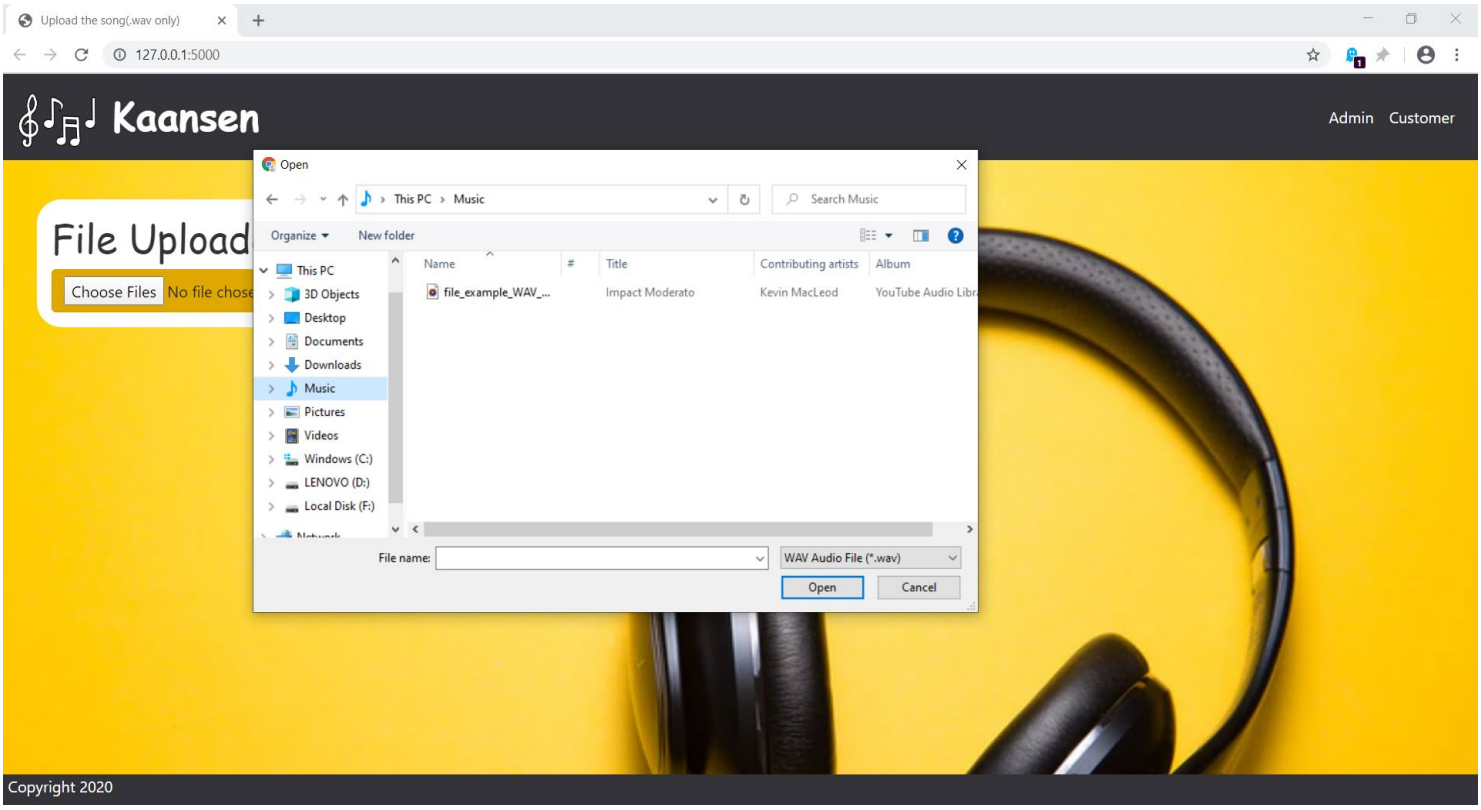
9.1 OUTCOMES

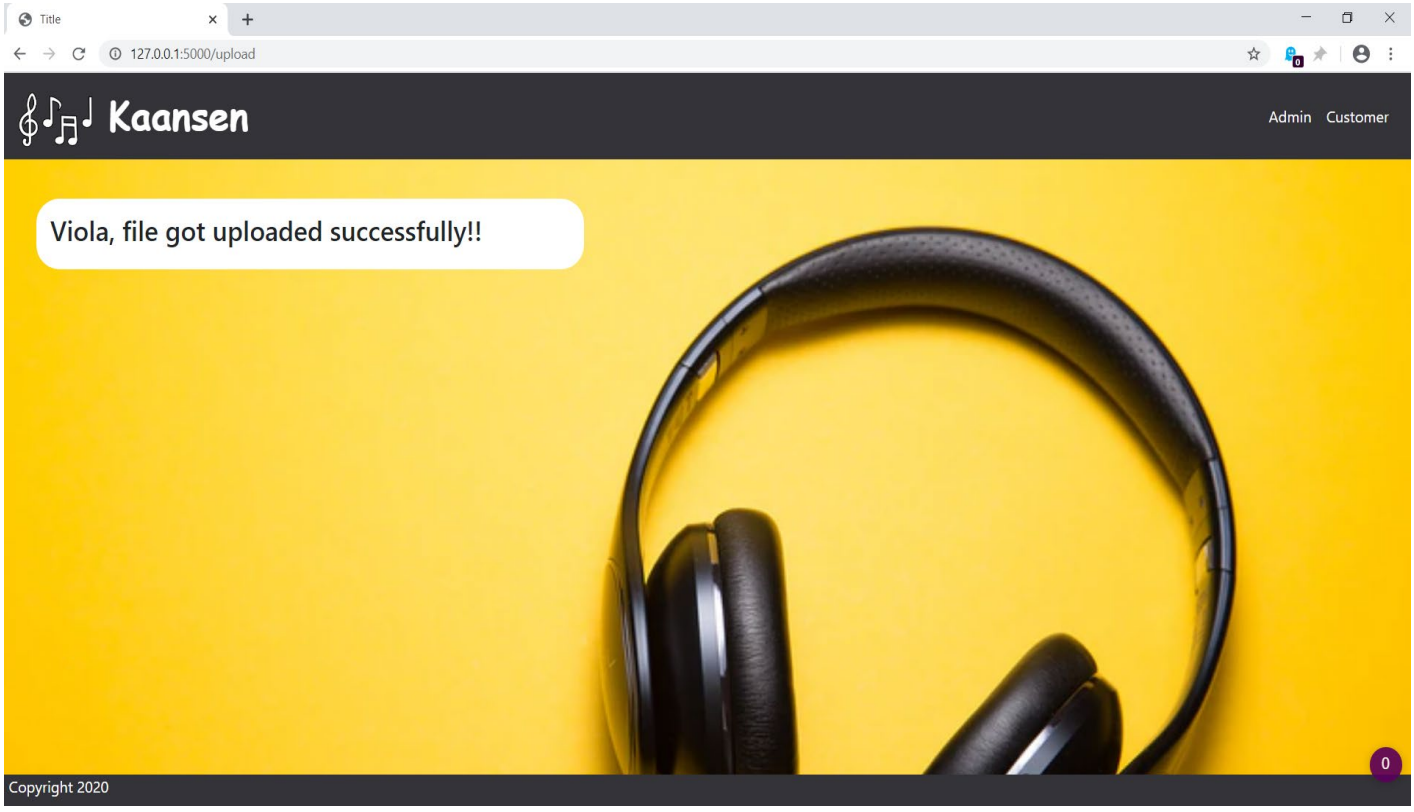
Since ancient times, recognition of Ragas in a song was only restricted to the people who are expert in that domain of music. Even if a common man wanted to know what kind of raag are used in a song, help isn't always available, the recognition of Ragas is tough, it takes years of knowledge to perfectly recognize a raga from a song. Not anymore, Raga recognizer, our software aims to solve or bridge that gap between an expert and a common man and provide help needed by common man to find the raagas contained in that song.

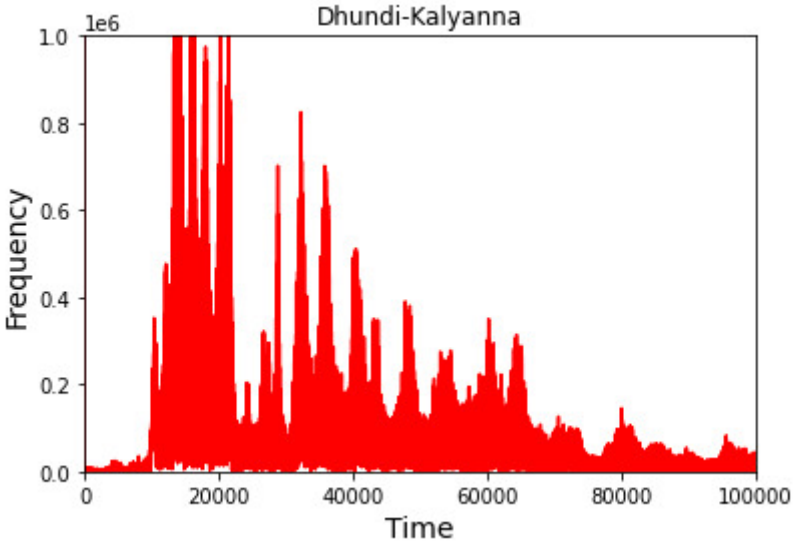
So, basically, we aim to recognize a raga from a given song and output the result to user. Depending on the input, our software will analyse the audio file for frequency pitches and match a range of frequencies to check if it belongs to a raga. If it does then we output the result to the user.

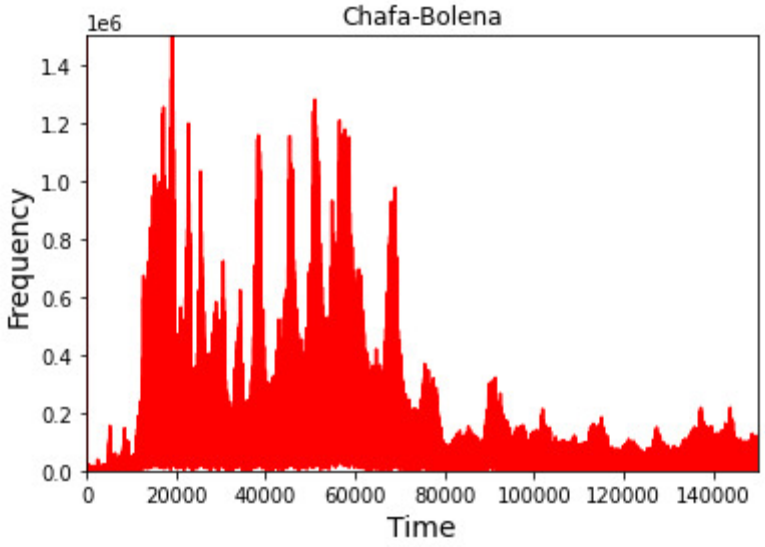
9.2 SCREEN SHOTS




Upload the song(.wav only)
127.0.0.1:5000
Kaansen
Admin
Customer
File Upload
Choose Files
No file chose
Open
This PC
Music
Search Music
Organize
New folder
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Windows (C:)
LENOVO (D:)
Local Disk (F:)
Name
#
Title
Contributing artists
Album
file_example_WAV_...
Impact Moderato
Kevin MacLeod
File name:
WAV Audio File (*.wav)
Open
Cancel
Copyright 2020


Title
127.0.0.1:5000/upload
Kaansen
Admin
Customer
Viola, file got uploaded successfully!!
0
Copyright 2020


Dhundi-Kalyanna
1e6
1.0
0.8
0.6
0.4
0.2
0.0
Frequency
0
20000
40000
60000
80000
100000
Time


Chafa-Bolena
1e6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
Frequency
0
20000
40000
60000
80000
100000
120000
140000
Time

10. CONCLUSIONS

10.1 CONCLUSION:

This software will help in recognizing few ragas effectively and thereby eliminating the manual work which would've been needed to be done that is, referring different sources in internet for information, asking an expert for an answer, etc. Thus, the software comes as a great help for people who are associated with the classical music industry as a professional or as a consumer of the services provided by this field at large.

10.2 ADVANTAGES

1. Recognition of Raga by any common man. Anyone who wants to recognize the songs they are listening to, can make use of this software to their advantage.

2. No need to have preexisting knowledge of Classical Music to use this software. Any layman in the field of music can use it, and also, they don't need to have experience in classical. Thus, this helps people to reduce their work for their end result which is to recognize the raga of a song.

3. Expense incurred by user for end goal of recognizing a Raga will be less. They won't have to spend money over learning classical music and spend years practicing it.

4. Users may be able to have new insights into the song they are listening.

10.3 FUTURE WORK:

- This software only recognizes few ragas. The future work in this would be-
    - To make the user interface much more user friendly. That is, new features can be implemented which include a better and responsive User Interface with more options to customize the section of the song to be recognized.
    - The output can be more informative than just giving out the raga name. It can include a number of more descriptive kind of information like what are the pakad and bandish involved in the raga, some video links which can shed more light on the raga, which all swaras are involved in it and many more information.

- To make more audio formats acceptable than the current accepting formats. In this way it will help the user to not convert his or her audio file to a specific format. Thus, we reduce the difficulties faced by the user in using this software, thereby improving on its usability parameter.

- To implement code to recognize a greater number of ragas. With the current context, its quite less ragas that are being recognized, and it won't help large section of people always. Hence, in future scope, the number of ragas it can recognize can be increased substantially thereby allowing users to enjoy full features of the software.

- To take input of audio from microphone of the user device and store the audio in the database and process and give output on the go. Sometimes user hears a song somewhere and wants to recognize the raga present in it then and there. So, if the user records the song, then converts to a supported audio format, it will not be fruitful.

- Hence, if this software has feature of listening to the audio from microphone access, it can process that information and give the result instantly. With this however we need to provide security such that we don't misuse the microphone access to develop a profile of the user.

- To make a full-fledged software which recognizes music other than classical ones. This will be final future scope, with the implementation of this concept, the software will be multipurpose and would open up to large section of the society. With such implementations, anyone can use the software.

To give out more information, external links of the ragas as output to the user to satisfy the research curious mind of the users.

10.3 APPLICATIONS

- Beginners in music curriculum learners can use this software to recognize the raga and update their music octave accordingly.

- Classical music enthusiasts can use this to recognize the raga they are listening to.

- Any person who is studying about ancient Classical culture and about music can use this software to get more information on type of music he or she is listening to.

APPENDIX A: PROBLEM STATEMENT FEASIBILITY ASSESSMENT USING, SATISFIABILITY ANALYSIS AND NP HARD, NP-COMPLETE OR P TYPE USING MODERN ALGEBRA AND RELEVANT MATHEMATICAL MODELS

What is P?

- P is set of all decision problems which can be solved in polynomial time by a deterministic
- Since it can be solved in polynomial time, it can be verified in polynomial time
- Therefore, P is a subset of NP

What is N?

- "N" in "NP" refers to the fact that you are not bound by the normal way a computer works which is step-by-step The "N" actually stands for "Non-deterministic" This means that you are dealing with an amazing kind of computer that can run things simultaneously or could somehow guess the right way to do things, or something like that
- So, this "N" computer can solve lots more problems in "P" time for example it can just clone copies of itself when needed
- So, programs that takes dramatically longer as the problem gets harder (i.e not in "P) could be solved quickly on this amazing "N" computer and so are in "NP"
- Thus "NP" means "we can solve it in polynomial time if we can break the normal rules of step-by-step computing"

What is NP?

- "NP" means "we can solve it in polynomial time if we can break the normal rules of step-by-step computing"

Project status:

- Reliable identification of raaga.

Solution:

- In the Feasibility Study stage, the assigned project is analyzed, the information about the project participants is collected, and the requirements for the system are gathered and analyzed.

- During the Feasibility Study stage, the project's goals, parameters and restraints are agreed and a conceptual problem solution is prepared. The proposed techniques provide a good quality tool to recognition of raagas.

- The proposed system consists of many steps such as frequency extraction, shift of scale, key phrase extraction, clustering, key phrase matching. The problem isn't NP-Hard. The determining of a Raga doesn't take infinite time or the system doesn't go into halting state. So, this project is NP Complete.

- It is shown in the following mathematical model:

Let S be the set of interfaces.
S={A}
Where
SD
A= System Module user side software
A= {I , O , F , Er}
I = Input
O = Output
F = Functions
Er = Error States
I = {Au}
Au = Audio file as input is supplied to software

Output: {D, N}
D= Raga was determined
N=Raga could not be determined

F={F1, F2, F3,F4,F5}
F1=frequency extraction
F2=shift of scale
F3=key phrases extraction
F4=clustering
F5=key phrase matching

Er= {interrupt on the device where the software is installed}

The heart of functioning is the F5 phase of the software. If the key phrases aren't matching with the input provided for any of the ragas in our dataset, then the result will be raga isn't recognized at that instant. If the key phrases do match, it will at max take exponential time. If it takes too much time, a specific timer will be set beyond which it should generate that the raga was not recognized. Thus, we eliminate the halting problem if at all present at F5 phase. Also, rest all functional phases do not halt or take much time as they are training phases, and F1 and F2 are done instantly. So, the project is feasible as the max time taken is exponential, and even then a result is generated without the system resorting to halting state.
In this way feasibility study is completed using NP Hard analysis with mathematical model to prove the problem statement is not NP-Hard.

APPENDIX B: DETAILS OF PAPER PUBLICATION: NAME OF THE CONFERENCE/JOURNAL, COMMENTS OF REVIEWERS, CERTIFICATE, PAPER

10.3.1 Paper title:

Computer Algorithm For Recognizing Raagas In Hindustani Classical Music Using Frequency Matching Techniques

10.3.2 Name of Conference/Journal where paper submitted:

International Conference on Internet of Things, Next Generation Networks and Cloud Computing (ICINC)

10.3.3 Paper accepted/rejected:

Accepted

APPENDIX C: PLAGIARISM REPORT

PLAGIARISM SCAN REPORT

Words	971	Date	October 11,2019
Characters	5857	Exclude Url	

2%	98%	1	46
Plagiarism	Unique	Plagiarized Sentences	Unique Sentences

PLAGIARISM SCAN REPORT

Words	999	Date	October 11,2019
Characters	6007	Exclude Url	

0%	100%	0	47
Plagiarism	Unique	Plagiarized Sentences	Unique Sentences

PLAGIARISM SCAN REPORT

Words	996	Date	October 11,2019
Characters	5695	Exclude Url	

0%	100%	0	50
Plagiarism	Unique	Plagiarized Sentences	Unique Sentences

PLAGIARISM SCAN REPORT

Words	943	Date	April 16,2020
Characters	5691	Exclude Url	

0%	100%	0	44
Plagiarism	Unique	Plagiarized Sentences	Unique Sentences

PLAGIARISM SCAN REPORT

Words	491	Date	April 16,2020
Characters	2997	Exclude Url	

0%	100%	0	26
Plagiarism	Unique	Plagiarized Sentences	Unique Sentences

PLAGIARISM SCAN REPORT

Words	202	Date	April 16,2020
Characters	1238	Exclude Url	

0%	100%	0	12
Plagiarism	Unique	Plagiarized Sentences	Unique Sentences

REFERENCES

1] V. Kumar, H. Pandya, C.V. Jawahar, Identifying Ragas in Indian Music, International Conference on Pattern Recognition ICPR 2014, August 2014, https://www.researchgate.net/publication/277258868_Identifying_Ragas_in_Indian_Music

2] S. Belle, R. Joshi, P. Rao, Raga Identification by using Swara Intonation, Journal of ITC Sangeet Research Academy, vol. 23, December, 2009, https://www.ee.iitb.ac.in/course/~daplab/publications/sb-pr-rkj-ninaad2009-v3.pdf

3] K. Priya, R. Geetha Ramani, S.G. Jacob, Data Mining Techniques for Automatic recognition of Carnatic Raga Swaram notes, International Journal of Computer Applications (0975 – 8887) Volume 52– No.10, August 2012, https://www.ijcaonline.org/archives/volume52/number10/8236-1444

4] G. Pandey, C. Mishra, P. Ipe, Tansen : A System For Automatic Raga Identification, 1st Indian International Conference on Artificial Intelligence, 2003, https://pdfs.semanticscholar.org/7045/3cb6e08661e98b867bdb6126853874d79a98.pdf

5] J.C. Ross, V.T.P , P. Rao, Detecting Melodic Motifs From Audio For Hindustani Classical Music, 13th International Society for Music Information Retrievel Conference (ISMIR 2012), August 2012, http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.294.3133&rep=rep1&type=pdf

6] S. Shetty, K.K. Acharya, Raga Mining of Indian Music by Extracting Arohana-Avarohana Pattern , International Journal of Recent Trends in Engineering Vol. 1, No. 1, May 2009, https://www.researchgate.net/publication/228885219_Raga_Mining_of_Indian_Music_by_Extracting_Arohana-Avarohana_Pattern.

7] Pranay Dighe, Parul Agrawal, Harish Karnick, Siddartha Thota and Bhiksha Raj, Scale independent raga identification using chromagram patterns and swara based features, IEEE International Conference on Multimedia and Expo Workshops, 2013, https://ieeexplore.ieee.org/abstract/document/6618238

8] Koduri Gopala-Krishna, Sankalp Gulati and Preeti Rao, A Survey of RAGA Recognition Techniques and Improvements to the State-of-the-Art, Sound and Music Computing, 2011, https://www.ee.iitb.ac.in/web/files/publications/KoduriP2011.pdf

9] Honglak Lee, Peter Pham, Yan Largman and Andrew Y Ng, Unsupervised feature learning for audio classification using convolutional deep belief networks, Neural Information Processing Systems, 2009, https://ai.stanford.edu/~ang/papers/nips09-AudioConvolutionalDBN.pdf

10] Gurav Pandey, Gaurav P, Chaitanya Mishra and Paul Ipe, Tansen : A System For Automatic Raga Identification, Indian International Conference on Artificial Intelligence, 2003, https://pdfs.semanticscholar.org/7045/3cb6e08661e98b867bdb6126853874d79a98.pdf

11] L. R. Rabiner: "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition": Proc. IEEE, Vol. 77, No. 2, pp. 257-286: February 1989, https://ieeexplore.ieee.org/document/18626

12] C. S. Iliopoulos and M. Kurokawa: "String Matching with Gaps for Musical Melodic Recognition": Proc. Prague Stringology Conference, pp. 55-64: 2002.

13] H. V. Sahasrabuddhe: "Searching for a Common Language of Ragas": Proc. Indian Music and Computers: Can 'Mindware' and Software Meet?: August 1994.

14] P. Chordia and A. Rae., "Raag recognition using pitch-class and pitch-class dyad distributions," in ISMIR 2007 8th Intl. Conf. on Music Information Retrieval, 2007.

15] A. K. Datta, R. Sengupta, N. Dey, and D. Nag. Experimental Analysis of Shrutis from Performances in Hindustani Music. Scientific Research Department, ITC Sangeet Research Academy, 1, N. S. C. Bose Road, Tollygunge, Kolkata 700040, India, 2006.

16] J. Chakravorty, B. Mukherjee and A. K. Datta: "Some Studies in Machine Recognition of Ragas in Indian Classical Music," Journal of the Acoust. Soc. India, Vol. 17, No.3&4, 1989.

17] Z. Juhasz: "Analysis Of Melody Roots In Hungarian Folk Music Using Self-Organizing Maps With Adaptively Weighted Dynamic Time Warping," Journal Applied Artificial Intelligence, Vol.21, No.1, 2007.

18] R. B. Dannenberg and N. Hu: "Pattern Discovery Techniques for Music Audio," Journal of New Music Research, Vol. 32, No.2, 2002.