A Mini project Report On

MP3 PLAYER

A project report submitted in partial fulfilment of the academic requirements for the award of degree of

BACHELOR OF TECHNOLOGY

In

CIVIL ENGINEERING

By

T. LAKSHMI ALEKHYA **17H51A0157**

Under the guidance of

Mr. V. SRINIVAS (ASSISTANT PROFESSOR)

CMR COLLEGE OF ENGINEERING & TECHNOLOGY

(UGC Autonomous)

Approved by AICTE, permanently Affiliated to JNTUH, Hyderabad.

An NBA Accredited Institution

Kandlakoya (V), Medchal-501401.

DEPARTMENT OF CIVIL ENGINEERING

CMR COLLEGE OF ENGINEERING&TECHNOLOGY (Autonomous)

Approved by AICTE, Permanently Affiliated to JNTUH, Hyderabad.

An NBA Accredited Institution

Kandlakoya(V), Medchal District-501401

BONAFIDE CERTIFICATE

This is to certify that the dissertation entitled "**MP3 PLAYER"** is the bonafide done by **T.LAKSHMI ALEKHYA (17H51A0157)** in the partial fulfilment of the academic requirements for the award of the degree of Bachelor of Technology, in Civil Engineering of CMR College of Engineering & Technology, Hyderabad during the period 2020-2021.

Signature of the Guide	Signature of the HOD
Mr. V. SRINIVAS	**Dr. K. SURESH**
Asst. professor	Professor & Head
Department of Civil Engineering	Department of Civil Engineering
CMRCET	CMRCET

CMR COLLEGE OF ENGINEERING & TECHNOLOGY

DEPARTMENT OF CIVIL ENGINEERING

DECLARATION

We hereby declare that the work described in the project report **"MP3 PLAYER "**has been carried under the guidance and supervision of **Mr. V. SRINIVAS**, Assistant Professor, CMRCET towards the partial fulfilment of the award of the degree of bachelor of technology in civil engineering from CMR college of engineering and technology, Hyderabad. This work is the original and has not been submitted either in part or in full for the award of any other degree or diploma of any other university

M.SRIVIDYA (17H51A0137)

MANIKANTA RAO SURIGINENI (17H51A0156)

T. LAKSHMI ALEKHYA (17H51A0157)

PODILA NAGARATHAMMA (18H55A0118)

ACKNOWLEDGEMENT

I am deeply indebted to our guide **Mr. V. SRINIVAS, Assistant Professor**, whose help, stimulating suggestions and encouragement helped me in all the time for writing the report.

I would like to express my sincere thanks and gratitude to **Dr. K. SURESH**, **Professor and Head**, Department of Civil Engineering, for his valuable suggestions and advice for carrying out this thesis work.

I would like to thank the Faculty of Department of Civil Engineering for their positive and helpful approaches for my engineering education and also the non-teaching staff for their assistance.

I am grateful and thankful to **Dr. V. A. NARAYANA**, **Principal & Secretary** & Correspondent **CH. GOPAL REDDY**, **CMR COLLEGE OF ENGINEERING & TECHNOLOGY** and management for providing all the necessary facilities for carrying out this work.

Finally, I would like to express my deepest gratitude to my parents whose patient love enabled me to complete this work. And at last, but not the least I would like to thank God for the successful completion of the project.

ABSTRACT

Audio is an important source of communication and is as important as text in today's time. We know that the audio files are digital files. Therefore, there is a need of a tool to run the digital files or in other words, play the files. Without this tool or player, we'll never be able to listen to music, movies or the contents of any audio file. Thus, we need MP3 players. It is a device using to play MP3s and other digital audio files. We can build this by ourselves without have to download and install premium music players. The Mp3 player GUI project idea attempts to emulate the physical MP3 Player. This program will allow you to play songs, music, and all MP3 files on your desktop or laptops. MP3 player using Python is a basic programming application built using the programming language Python. It is a GUI program built by the means of Python libraries Tkinter, Pygame and Mutagen. The MP3 player application should have the capabilities of playing a song, create and display a playlist, pause and resume a long and change the song, that is, play the previous or next song.

TABLES OF CONTENTS

CHAPTER 1

INTRODUCTION

INTRODUCTION:

1.1 PYTHON:

Python is a widely used general-purpose, high level programming language. It was created by Guido van Rossum in 1991 and further developed by the python software foundation. It was designed with an emphasis on code readability, and its syntax allows programmers to express their concepts in fewer lines of code.

Python is a programming language that lets you work quickly and integrate systems more efficiently.

1.1.1 IT IS USED FOR:

1. Web development (server-side),
2. Software development,
3. Mathematics,
4. System scripting.

1.1.2 WHAT CAN PYTHON DO?

1. Python can be used on a server to create web applications.

Python is a programming language that lets you work quickly and integrate systems more efficiently.

1.1.3 IT IS USED FOR:

5. Web development (server-side),
6. Software development,
7. Mathematics,
8. System scripting.

1.1.4 WHAT CAN PYTHON DO?

2. Python can be used on a server to create web applications.
3. Python has a simple syntax similar to English language.
4. Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
5. Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.

6. Python can be treated in a procedural way, an object – oriented way or a fractional way.

We need an application that will allow us to play or listen to digital audio files. MP3 player is the device to play MP3s and other digital audio files. The MP3 GUI program application attempts to emulate the physical MP3 Player.

This program will allow you to play songs, music, and all MP3 files on your desktop or laptops. The main objective of this project is to allow users to play MP3 and digital audio files. To be engaging for users, the application has to have a simple but beautiful user interface.

This GUI project is developed using Python programming language. The GUI aspect of the application is built using the Tkinter library of Python. The interactive part of the application that handles the MP3 files uses the Pygame and Mutagen libraries.

You can have an interface for listing the available MP3 files. You can also give users the option to list other digital audio files that are not MP3. The users will also expect the MP3 Player to have an interface that shows information on the file that is playing. Some of the information you can include are the name of the file, its length, the amount played, and the amount not played, in minutes and seconds.

Python has libraries that can play audio files, such as Pygame, which allows you to work with multimedia files in few lines of code. Similar libraries are Pymedia and Simple audio. These libraries can handle a lot of digital audio files.

They can handle other file types, not just the MP3 files. You can also implement a feature that allows users to create a playlist. To do this, you'll need a database to store information on the created playlists.

Python's sqlite3 module allows you to use the SQLite database. The SQLite database is a better option in this case, because it is file based and easier to set up than other SQL databases. While SQLite is file based, it is better for saving data than a regular file.

USER REQUIREMENTS:

1. The first step is to create a directory called environments where to store all your virtual Environments and install the python modules. In this way the program is available only within the set environment and not throughout my computer.

2. The second step is to use a module **Tkinter** to create isolated virtual environments.

3.Now, we can create a virtual environment named **myenv.** If you have installed python 3 you can easily do it.

4. Inside the directory **myenv** there is a copy of Python interpreter, the standard library and the various supporting files.

5. Then, we need to install **Pygame.** Pygame is a module that is used for creating videogames. For creating our **MP3 Music Player** we will use the sound component.

CHAPTER 2

PROBLEM DEFINITION

Sony entered the digital audio **player** market in 1999 **with** the Vaio Music Clip and Memory Stick **Walkman**, however they were technically not **MP3 players** as it did not support the **MP3** format but instead Sony's own ATRAC format and WMA. The company's first **MP3**-supporting **Walkman player** did not come until 2004.

1. To build an MP3 player using Python programming language to be able to play and listen to songs, MP3 files and other digital audio files.

2. Determine the functionalities of the MP3 player.

3. The player should be have a simple and easy to use GUI with options for various functions, display screen to display the entire playlist and buttons to shut down the player.

4. The player should be able to play any song. It should be capable of playing MP3 files or any other digital audio files.

5. The player should allow the user to browse through the contents of the computer drive to choose song/s to be played or queued.

6. It should provide the user with option to pause or resume the song.

7. The user should be able to play the previous or the next song in the playlist.

8. Lastly, the user should get basic details about the current playing song. The details can include the song name, singer’s name, the duration of the song, size of the file, etc.

CHAPTER 3

ALGORITHM

1.Import the libraries.

2. Create an object of the tkinter and Pygame libraries.

3. Create a window using Tkinter object.

4. Add buttons that provide different functionalities.

- Add one song or multiple songs
- Play and stop the song
- Pause the song
- Play previous song
- Play next song

5. Add a song button when pressed should open a dialog box to browse and choose the file.

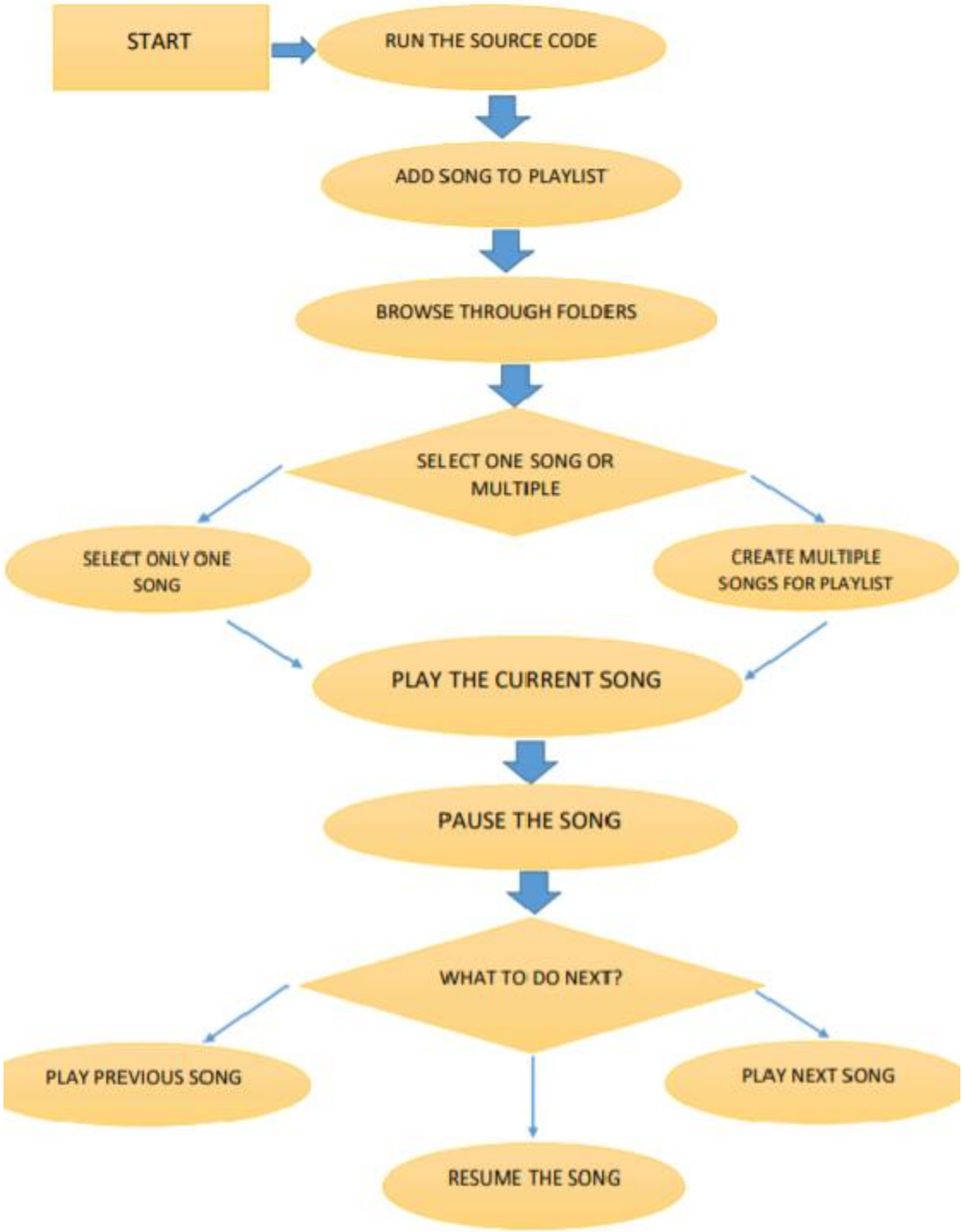
6. Add label to display the song's information.

- Name
- Singer
- Duration
- Size of the file, etc.

7. Display screen will display the details of the entire playlist.

8. Close button will automatically clear the song list and will stop playing the song

FLOW CHART


START
RUN THE SOURCE CODE
ADD SONG TO PLAYLIST
BROWSE THROUGH FOLDERS
SELECT ONE SONG OR MULTIPLE
SELECT ONLY ONE SONG
CREATE MULTIPLE SONGS FOR PLAYLIST
PLAY THE CURRENT SONG
PAUSE THE SONG
WHAT TO DO NEXT?
PLAY PREVIOUS SONG
PLAY NEXT SONG
RESUME THE SONG

CHAPTER 4
IMPLIMENTATION

```
from tkinter import *
import pygame
from tkinter import filedialog

root= Tk()
root.title('Mp3 Player')
root.iconbitmap('C:/Users/SAI TEJA')
root.geometry("500x400")

#Iitialize pygame Mixer
pygame.mixer.init()

#Add Song function
def add_song():

    song= filedialog.askopenfilename(initialdir='audio/',title="choose a song", filetype=(("mp3 Files","*.mp3"),))

    # removing the info and mp3 at the end
    song= song.replace("C:/Users/SAI TEJA/audio", "")
    song=song.replace(".mp3", "")

    #Add song to listbox
    song_box.insert(END, song)
#Add many songs
def add_many_songs():
    songs= filedialog.askopenfilenames(initialdir='audio/',title="choose a song", filetype=(("mp3 Files","*.mp3"),))
```

```
    #loop through song list and replace directory info and mp3
    for song in songs:
        song= song.replace("C:/Users/SAI TEJA/audio", "")
        song=song.replace(".mp3", "")

        #Insert into playlist
        song_box.insert(END, song)

#For playing selected song
def play():
    song=song_box.get(ACTIVE)
    song=f'C:/Users/SAI TEJA/audio{song}.mp3'

    #To play songs
    pygame.mixer.music.load(song)
    pygame.mixer.music.play(loops=0)



#for stop playing
def stop():
    pygame.mixer.music.stop()
    song_box.selection_clear(ACTIVE)

def next_song():
    #get the current song tuple number
    next_one = song_box.curselection()
    #add one to the current song number
    next_one = next_one[0]+1
    #grab song title from playlist
    song=song_box.get(next_one)
    #add the directory structure and mp3 as we did in play function
    song=f'C:/Users/SAI TEJA/audio{song}.mp3'
    print(song)
```

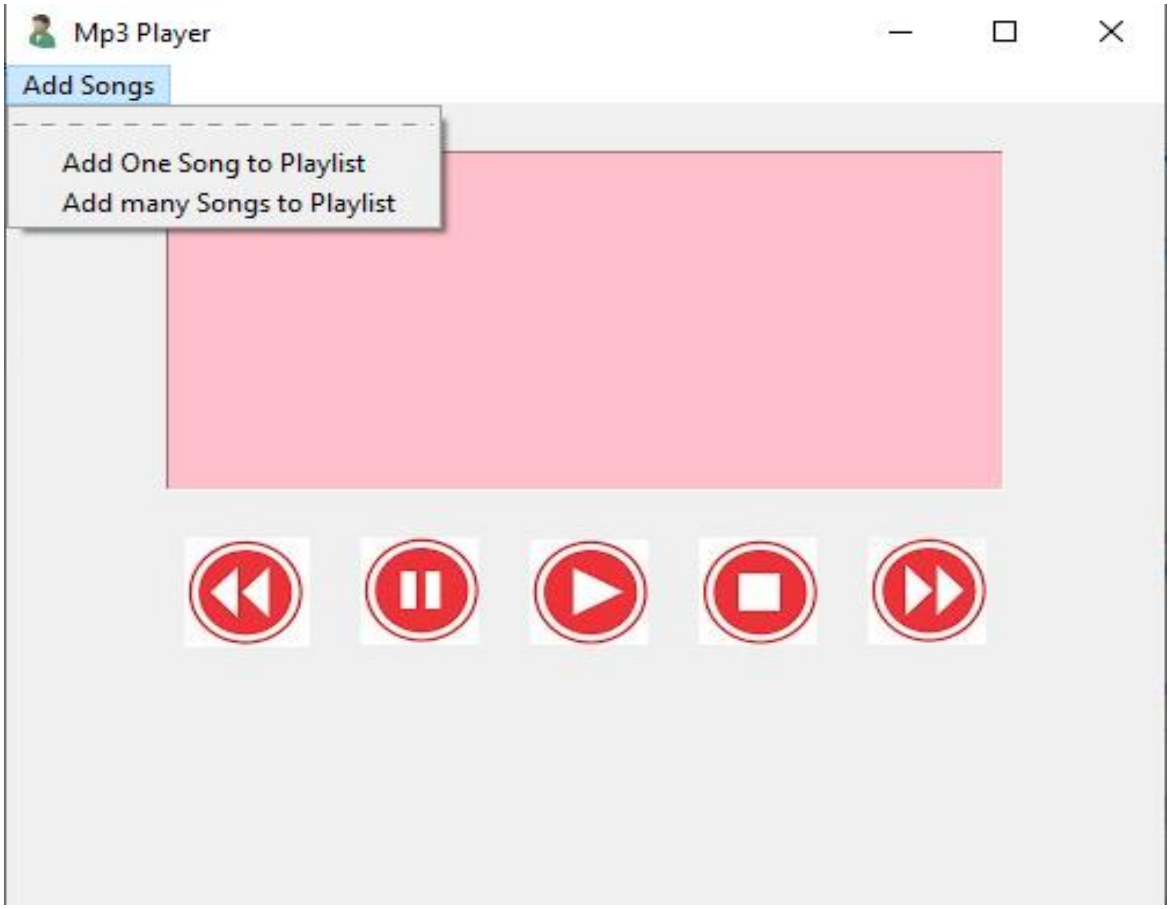
```
    pygame.mixer.music.load(song)
    pygame.mixer.music.play(loops=0)

    #clear move active bar in playlist
    song_box.selection_clear(0, END)
    #Activate new song bar
    song_box.activate(next_one)
    #Set active bar to next song
    song_box.selection_set(next_one, last=None)

#play previous song in playlist
def previous_song():
    #get the current song tuple number
    next_one = song_box.curselection()
    #add one to the current song number
    next_one = next_one[0]-1
    #grab song title from playlist
    song=song_box.get(next_one)
    #add the directory structure and mp3 as we did in play function
    song=f'C:/Users/SAI TEJA/audio{song}.mp3'
    print(song)
    pygame.mixer.music.load(song)
    pygame.mixer.music.play(loops=0)

    #clear move active bar in playlist
    song_box.selection_clear(0, END)
    #Activate new song bar
    song_box.activate(next_one)
    #Set active bar to next song
    song_box.selection_set(next_one, last=None)


#create global pause variable for using inside and outside of diff functions
```

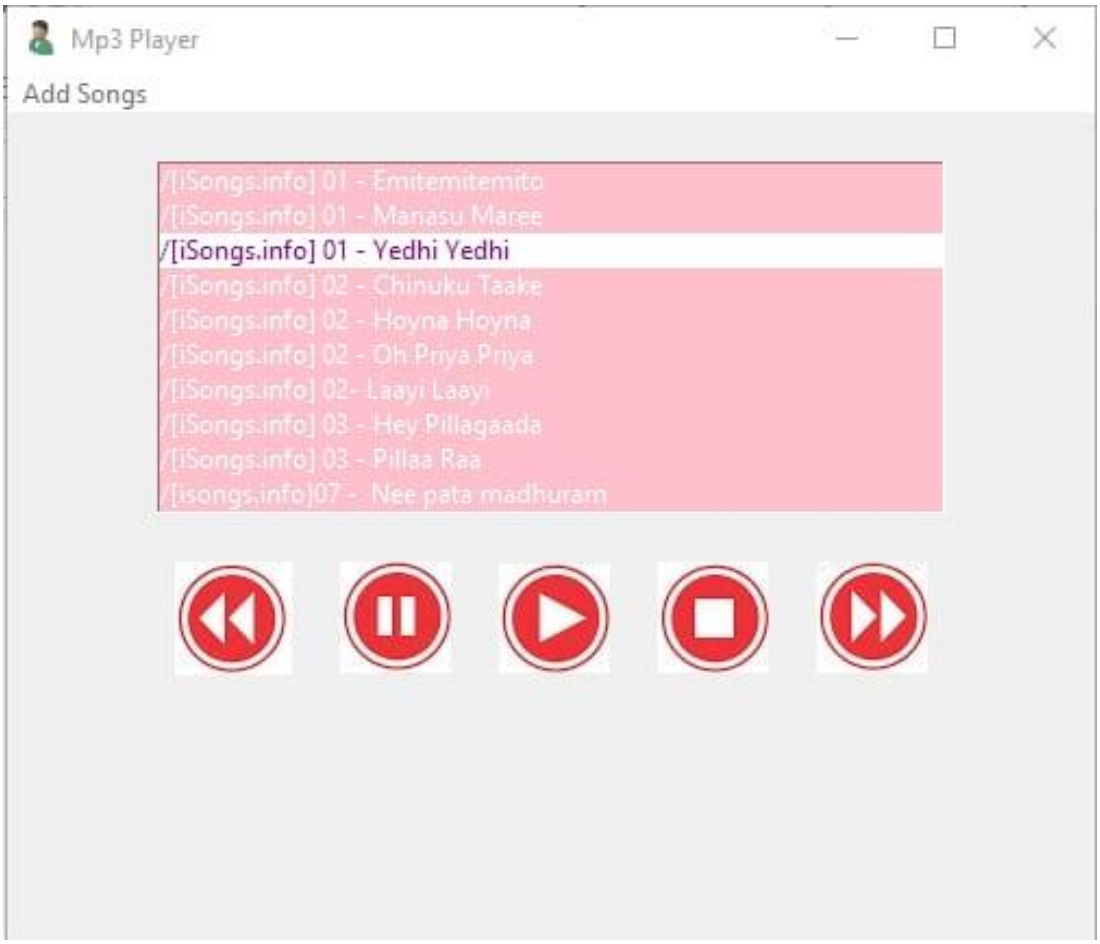
```
global paused
paused = False

#for pausing we use pygame
def pause(is_paused):
    global paused
    paused= is_paused
    if paused:
        # Unpause
        pygame.mixer.music.unpause()
        paused= False
    else:
        #Pause
        pygame.mixer.music.pause()
        paused=True

   #create a playlist Box by using tkinter widgets
#bg is background colour and fg is foreground for letters, selectbackground is selectingcolour and selectforeground is for text while selection
song_box=Listbox(root, bg="pink", fg="white", width=60, selectbackground="white", selectforeground="purple")
song_box.pack(pady=20)


# define player control button images
back_btn_img= PhotoImage(file='images/back2.png')
pause_btn_img= PhotoImage(file='images/pause3.png')
play_btn_img= PhotoImage(file='images/play4.png')
stop_btn_img= PhotoImage(file='images/stop5.png')
forward_btn_img= PhotoImage(file='images/forward2.png')

# create player control Frames to easily get centered(pack)
controls_frame = Frame(root)
```

```
controls_frame.pack()

# create player control Buttons
back_button=Button(controls_frame, image=back_btn_img, borderwidth=0, command=previous_song)
forward_button=Button(controls_frame, image=forward_btn_img, borderwidth=0, command=next_song)
pause_button=Button(controls_frame, image=pause_btn_img, borderwidth=0, command=lambda: pause(paused))
play_button=Button(controls_frame, image=play_btn_img, borderwidth=0, command=play)
stop_button=Button(controls_frame, image=stop_btn_img, borderwidth=0, command=stop)

back_button.grid(row=0, column=0, padx=10)
forward_button.grid(row=0, column=4, padx=10)
play_button.grid(row=0, column=2, padx=10)
pause_button.grid(row=0, column=1, padx=10)
stop_button.grid(row=0, column=3, padx=10)

#create Menu
my_menu = Menu(root)
root.config(menu=my_menu)

#Add song Menu
add_song_menu=Menu(my_menu)
my_menu.add_cascade(label="Add Songs", menu=add_song_menu)
add_song_menu.add_command(label="Add One Song to Playlist", command=add_song)
#Above we have add_song function we did not define so define at the top

#Add many songs to playlist
add_song_menu.add_command(label="Add many Songs to Playlist", command=add_many_songs)
#now again create add_many_songs function at top
root.mainloop()
```

CHAPTER 5

RESULT




Mp3 Player
Add Songs
/[iSongs.info] 01 - Emitemitemito
/[iSongs.info] 01 - Manasu Maree
/[iSongs.info] 01 - Yedhi Yedhi
/[iSongs.info] 02 - Chinuku Taake
/[iSongs.info] 02 - Hoyna Hoyna
/[iSongs.info] 02 - Oh Priya Priya
/[iSongs.info] 02- Laayi Laayi
/[iSongs.info] 03 - Hey Pillagaada
/[iSongs.info] 03 - Pillaa Raa
/[isongs.info]07 - Nee pata madhuram

CHAPTER 6

CONCLUSION

- MP3 player is a device built to play and listen to digital audio files. These can be either MP3 files or some other audio files. The player was built using Python language. A GUI implementation of the application was developed that is simple and easy to use.

- The application provides the user with five options — to add song to a playlist, to play the song, to pause or resume the song, to play the previous song and to play the next song.

- The player also has the capability to add multiple songs to the playlist at the same time. It has a large display area where the playlist is visible.

- Once a song is selected and played, we can hear it and can also see details about the song on top of the display. This information includes details about the song such as song name, singer's name, duration of the song, size of the file, etc.

- The Tkinter library of Python was used to create the GUI of the project. It was used to create the option buttons, the label and the display area.

- The Pygame and Mutagen library was used to add songs, play the songs, provide pause and resume options.

- In conclusion, a successful project was built in which songs will play one after the other automatically and the entire playlist will play all over again once concluded.