

# Software Project Report

Shirsendu Pal CH22BTECH11033

**AIM :-** To make an audioplayer in which songs can be played in random order.

**Scripting Language :-** Python (3.10.6)

**Libraries Used :-**

- 1) Numpy Library,
- 2) Playsound Library,
- 3) Pygame,
- 4) Tkinter.

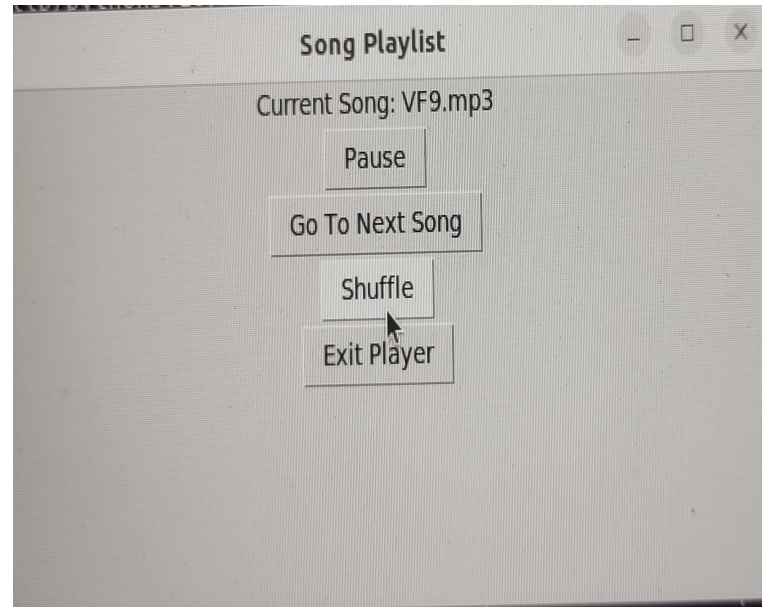


Image of the output of the code

**Mechanism :-**

- 1) Import numpy, pygame, random and tkinter library using 'pip install'.
- 2) Give the directory of the songs to the directory locator so that songs can be accessed.
- 3) Making a list of audio files.
- 4) Buttons for 'Play Next Song' and
- 5) Play the songs according to their position in the playlist.
- 6) If all the songs are played once in the loop, then a new random list of the songs is generated.
- 7) This process continues till the program is exited.

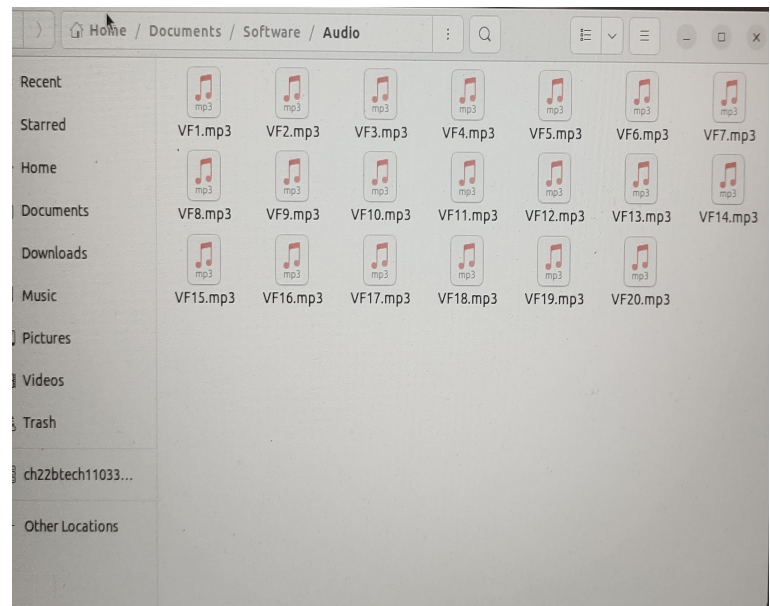


Image of the playlist folder

**Conclusion :-** Thus, we have created a program which plays songs randomly, selected from the created playlist in the Music Folder.

**Output Image :-**

```

import pygame
import random
import os
import numpy as np
import soundfile as sf
from tkinter import Tk, filedialog
import tkinter as tk

os.environ['SDL_VIDEODRIVER'] = 'x11'

def choose_music_folder():
    Tk().withdraw()
    folder_path = filedialog.askdirectory()
    return folder_path

def get_music_files(folder_path):
    music_files = []
    for file in os.listdir(folder_path):
        if file.endswith('.mp3'):
            music_files.append(os.path.join(folder_path, file))
    return music_files

def play_next():
    global current_song_index
    current_song_index = (current_song_index + 1) % len(playlist)
    pygame.mixer.music.load(playlist[current_song_index])
    pygame.mixer.music.play()

def shuffle_playlist():
    global playlist, shuffled_playlist, current_song_index
    shuffled_playlist = playlist.copy()
    np.random.shuffle(shuffled_playlist)
    current_song_index = 0
    pygame.mixer.music.load(shuffled_playlist[current_song_index])
    pygame.mixer.music.play()

def update_current_song_label():

```

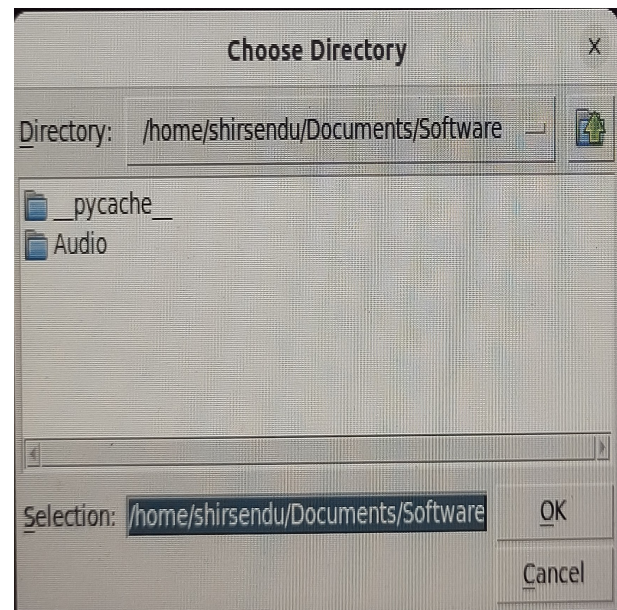


Image of the Playlist Directory Selector

```

    pygame.mixer.music.load(shuffled_playlist[current_song_index])
    pygame.mixer.music.play()

def update_current_song_label():
    current_song_label.config(text=f"Current Song: {os.path.basename(playlist[current_song_index])}")

def start_stop():
    global paused
    if paused:
        pygame.mixer.music.unpause()
        paused = False
        play_pause_button.config(text="Pause")
    else:
        pygame.mixer.music.pause()
        paused = True
        play_pause_button.config(text="Play")

def next_song():
    play_next()
    update_current_song_label()

def shuffle_songs():
    shuffle_playlist()
    update_current_song_label()

def exit_player():
    pygame.mixer.music.stop()
    window.quit()

pygame.init()
music_folder = choose_music_folder()

playlist = get_music_files(music_folder)
current_song_index = 0
shuffle_playlist()

# Create the GUI
window = tk.Tk()

```

```

    pygame.mixer.music.stop()
    window.quit()

pygame.init()
music_folder = choose_music_folder()

playlist = get_music_files(music_folder)
current_song_index = 0
shuffle_playlist()

# Create the GUI
window = tk.Tk()
window.title("Song Playlist")
window.geometry("500x250")
current_song_label = tk.Label(window, text="Playing: ")
current_song_label.pack()
paused = False
play_pause_button = tk.Button(window, text="Pause", command=start_stop)
play_pause_button.pack()

next_song_button = tk.Button(window, text="Go To Next Song", command=next_song)
next_song_button.pack()

shuffle_button = tk.Button(window, text="Shuffle", command=shuffle_songs)
shuffle_button.pack()

exit_button = tk.Button(window, text="Exit Player", command=exit_player)
exit_button.pack()

update_current_song_label()

pygame.mixer.init()
pygame.mixer.music.load(playlist[current_song_index])
pygame.mixer.music.play()

window.mainloop()
pygame.quit()

```

Images of the code