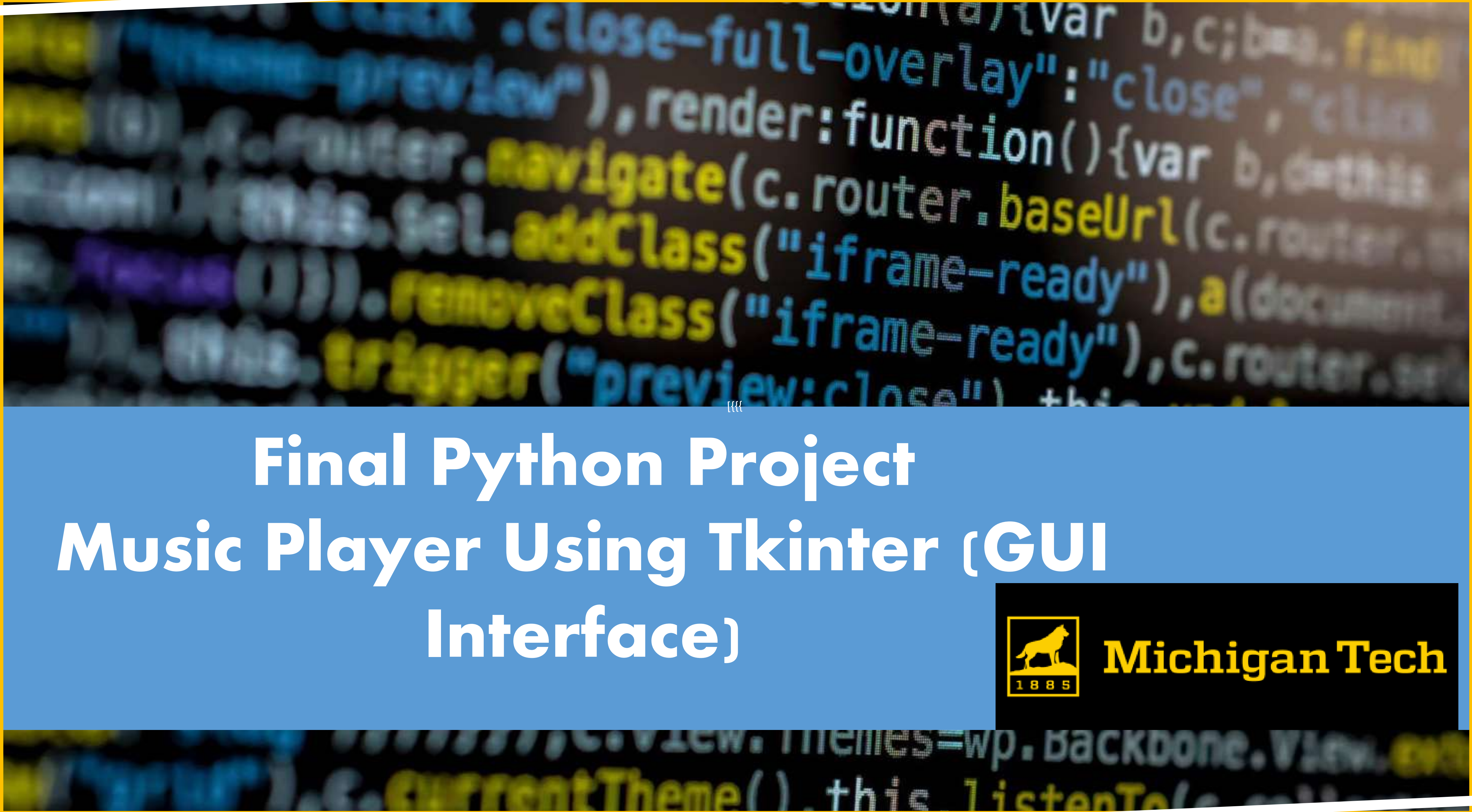


Final Python Project
Music Player Using Tkinter (GUI Interface)

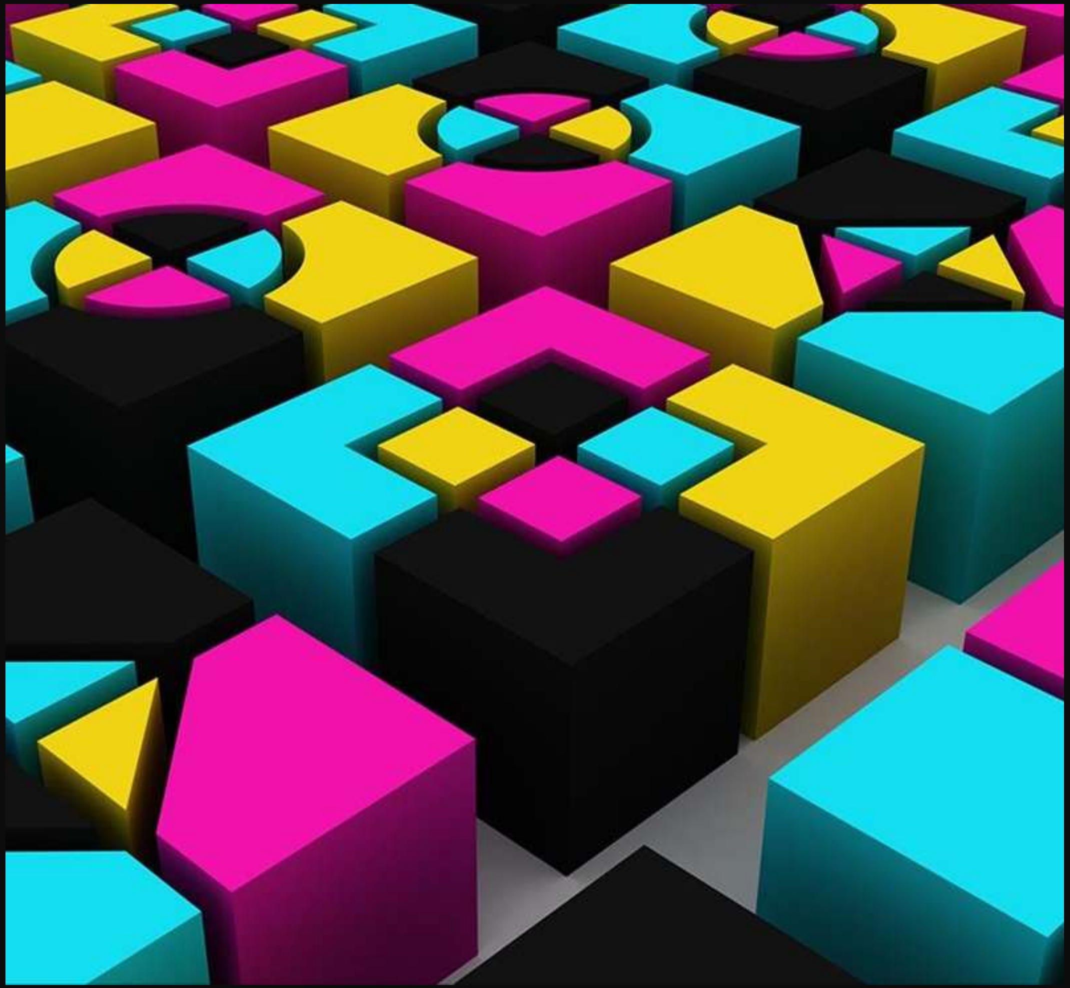
Long term goal & major application

- ✓ Music Playback
- ✓ Music Library management
- ✓ User Interface
- ✓ Audio Equalization
- ✓ Cross platform Compatibility
- ✓ Media Control

Michigan Tech

Libraries & Modules

- ✓Tkinter
- ✓os
- ✓Pygame
- ✓Pydub
- ✓PIL



Michigan Tech

Modules Description

- ✓Pygame is a game development toolkit for producing 2D games and multimedia applications.
- ✓Tkinter is a GUI library for creating desktop applications with graphical user interfaces this is inbuilt module in the python and no requirement is needed for installation. It’s a standard library.
- ✓Pydub is an audio processing library for reading, modifying, and exporting audio files in Python. Its a function that is used to manipulate, read & export audio files in various formats. This library function allows developers to slicing, concatenating and exporting.

- ✓In order to communicate with the operating system, the os module is imported. In particular, it is used to extract a list of files from a directory and add them to the music player GUI's song list box.
- ✓Image: This module offers classes and techniques for navigating, editing, and storing a variety of image file types. It is frequently employed for image processing operations like scaling, cropping, filtering, and more.
- ✓ImageTk: This module offers classes and procedures for incorporating PIL pictures into tkinter GUI programs. In tkinter windows and widgets like labels, buttons, and canvases, you may show images.

Michigan Tech

Understanding the code lines

✓self.disc_image = Image.open(disc_image_path).
✓self.disc_image = self.disc_image.resize(300, 300)
✓self.disc_photo = ImageTk.PhotoImage(self.disc_image)
✓self.disc_label = tk.Label(self.root, image=self.disc_photo, bg="black")
✓self.song_listbox = tk.Listbox(self.root, height=15, width=50, font=("Times New Roman", 12), bg="#2C2C2C", fg="white", selectbackground="#004BA8", selectforeground="white")
✓self.song_listbox.insert(tk.END, song)
✓self.play_button = tk.Button(self.root, text="PLAY", width=10, font=("Times New Roman", 14, "bold"), bg="#004BA8", fg="white", activebackground="#00336E", command=self.play_song)

Michigan Tech

- ✓ self.pause_button = tk.Button(self.root, text="PAUSE", width=10, font=("Times New Roman", 14, "bold"), bg="#004BA8", fg="white", activebackground="#00336E", command=self.pause_song)
- ✓ self.stop_button = tk.Button(self.root, text="STOP", width=10, font=("Times New Roman", 14, "bold"), bg="#004BA8", fg="white", activebackground="#00336E", command=self.stop_song)
- ✓ self.volume_slider = tk.Scale(self.root, from_=0, to=100, orient=tk.HORIZONTAL, length=500, sliderlength=20, width=15, font=("Times New Roman", 12), bg="blue", fg="white", highlightbackground="WHITE", troughcolor="black", command=self.set_volume)
- ✓ self.theme_label, self.theme_var, self.theme_var.set("Default"), self.theme_menu, self.change_theme(self, theme)

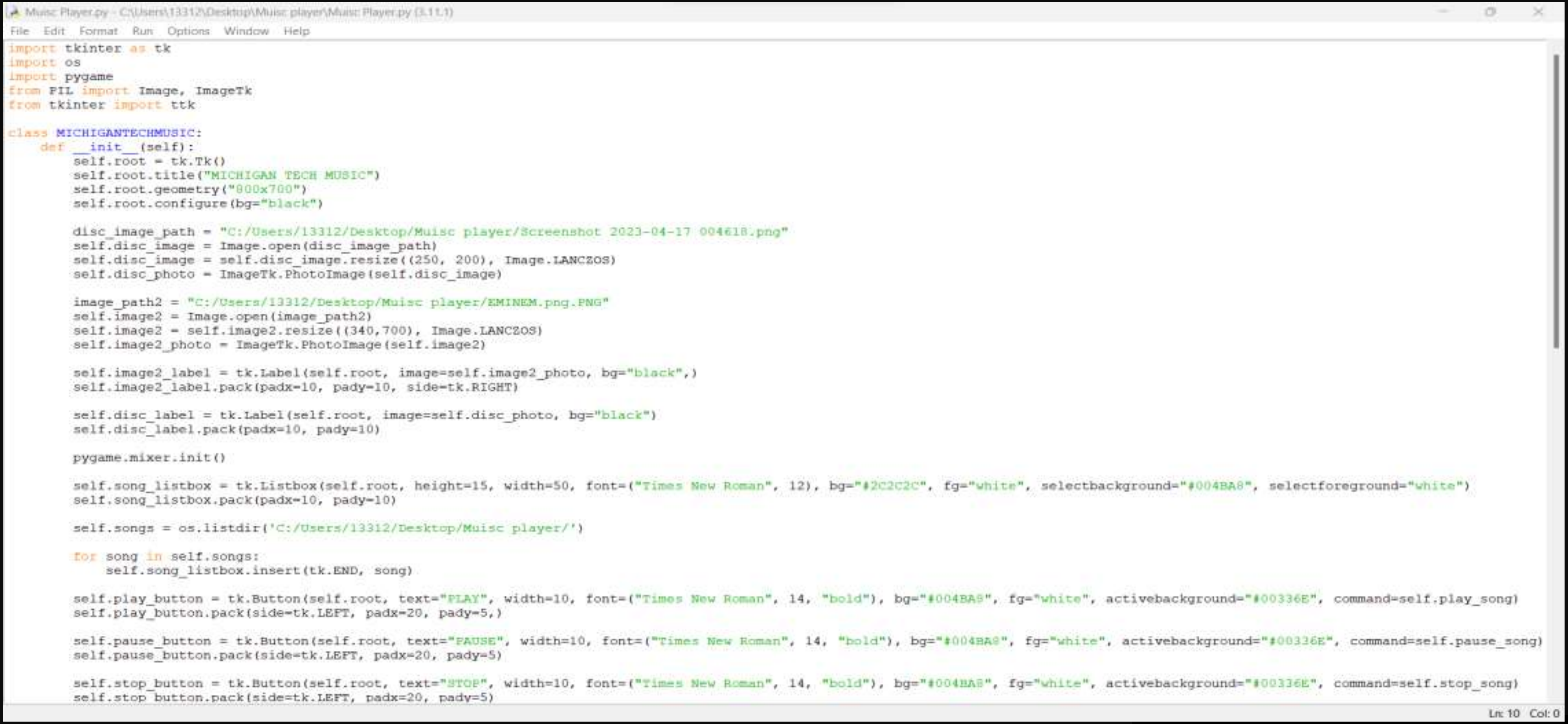
Michigan Tech

Challenges Encountered

✓File paths
✓Error handling
✓UI design
✓Music file format support
✓Organization and modularity of the code
✓Code efficiency
✓Platform compatibility

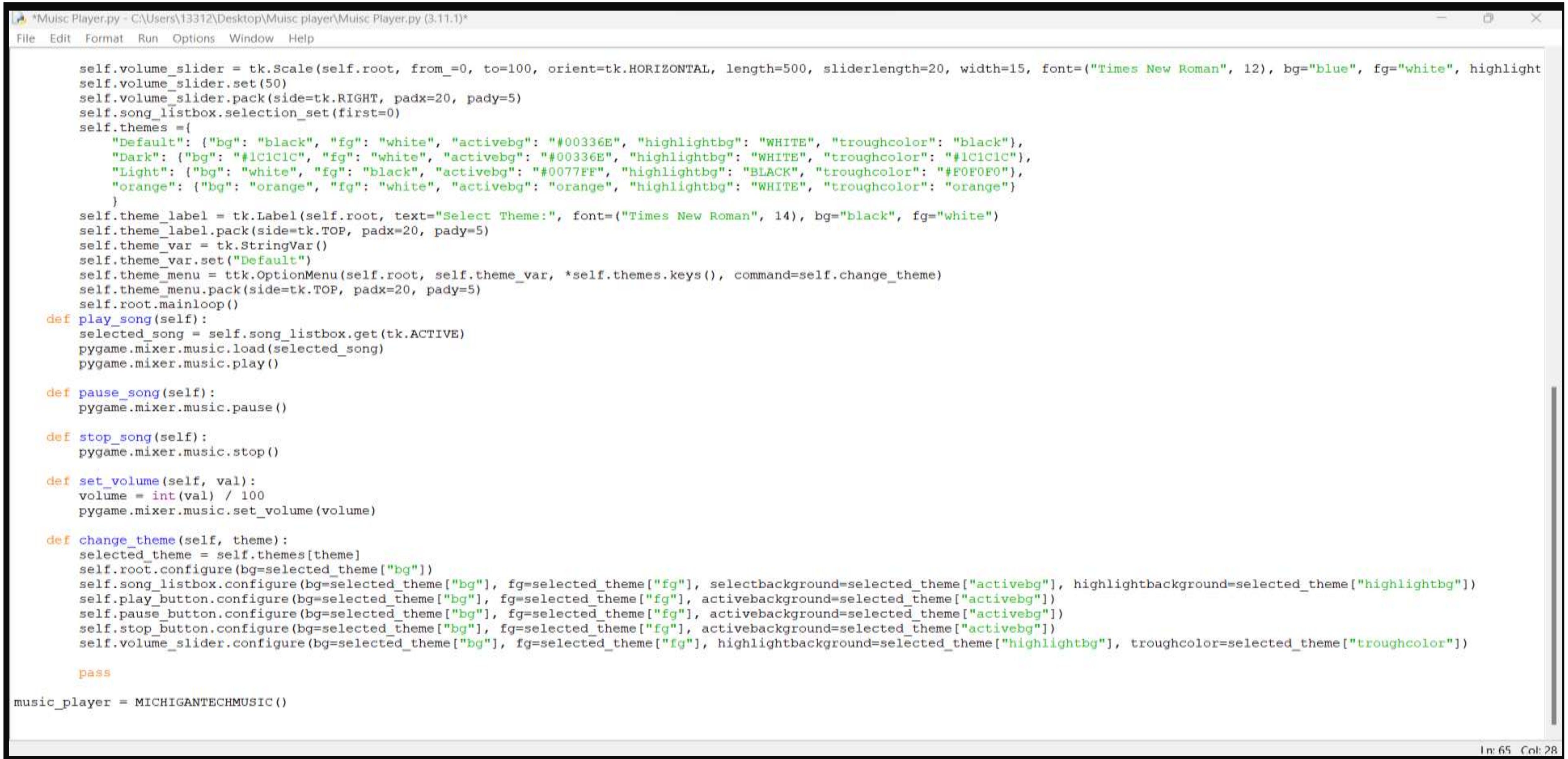
Michigan Tech

CODE


Muisc Player.py - C:\Users\13312\Desktop\Muisc player\Muisc Player.py (3.11.1)
File Edit Format Run Options Window Help
Ln: 10 Col: 0

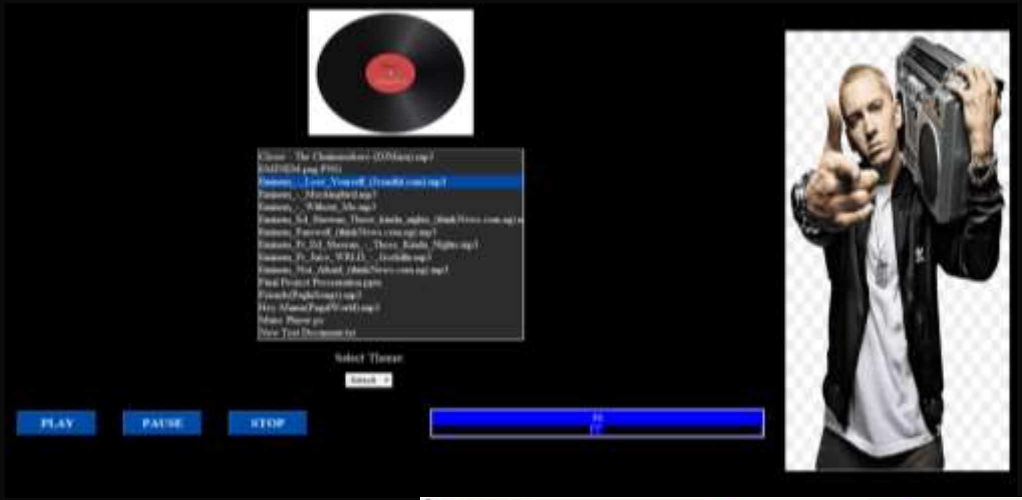
```
import tkinter as tk
import os
import pygame
from PIL import Image, ImageTk
from tkinter import ttk

class MICHIGANTECHMUSIC:
    def __init__(self):
        self.root = tk.Tk()
        self.root.title("MICHIGAN TECH MUSIC")
        self.root.geometry("800x700")
        self.root.configure(bg="black")

        disc_image_path = "C:/Users/13312/Desktop/Muisc player/Screenshot 2023-04-17 004618.png"
        self.disc_image = Image.open(disc_image_path)
        self.disc_image = self.disc_image.resize((250, 200), Image.LANCZOS)
        self.disc_photo = ImageTk.PhotoImage(self.disc_image)

        image_path2 = "C:/Users/13312/Desktop/Muisc player/EMINEM.png.PNG"
        self.image2 = Image.open(image_path2)
        self.image2 = self.image2.resize((340,700), Image.LANCZOS)
        self.image2_photo = ImageTk.PhotoImage(self.image2)

        self.image2_label = tk.Label(self.root, image=self.image2_photo, bg="black",)
        self.image2_label.pack(padx=10, pady=10, side=tk.RIGHT)

        self.disc_label = tk.Label(self.root, image=self.disc_photo, bg="black")
        self.disc_label.pack(padx=10, pady=10)

        pygame.mixer.init()

        self.song_listbox = tk.Listbox(self.root, height=15, width=50, font=("Times New Roman", 12), bg="#2C2C2C", fg="white", selectbackground="#004BA8", selectforeground="white")
        self.song_listbox.pack(padx=10, pady=10)

        self.songs = os.listdir('C:/Users/13312/Desktop/Muisc player/')

        for song in self.songs:
            self.song_listbox.insert(tk.END, song)

        self.play_button = tk.Button(self.root, text="PLAY", width=10, font=("Times New Roman", 14, "bold"), bg="#004BA8", fg="white", activebackground="#00336E", command=self.play_song)
        self.play_button.pack(side=tk.LEFT, padx=20, pady=5,)

        self.pause_button = tk.Button(self.root, text="PAUSE", width=10, font=("Times New Roman", 14, "bold"), bg="#004BA8", fg="white", activebackground="#00336E", command=self.pause_song)
        self.pause_button.pack(side=tk.LEFT, padx=20, pady=5)

        self.stop_button = tk.Button(self.root, text="STOP", width=10, font=("Times New Roman", 14, "bold"), bg="#004BA8", fg="white", activebackground="#00336E", command=self.stop_song)
        self.stop_button.pack(side=tk.LEFT, padx=20, pady=5)
```

1885 Michigan Tech

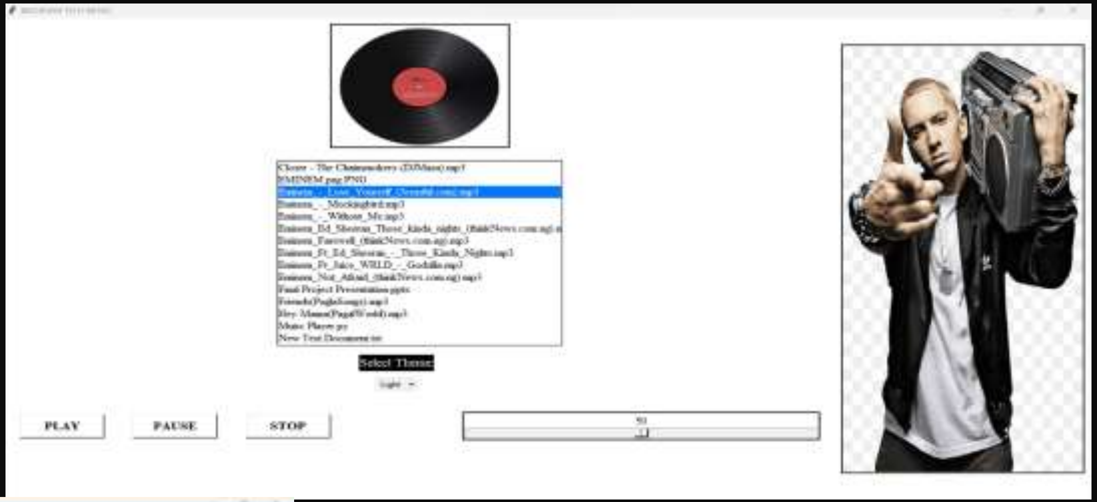

Muisc Player.py - C:\Users\13312\Desktop\Muisc player\Muisc Player.py (3.11.1)
File Edit Format Run Options Window Help
Ln: 65 Col: 28

```
        self.volume_slider = tk.Scale(self.root, from_=0, to=100, orient=tk.HORIZONTAL, length=500, sliderlength=20, width=15, font=("Times New Roman", 12), bg="blue", fg="white", highlight
        self.volume_slider.set(50)
        self.volume_slider.pack(side=tk.RIGHT, padx=20, pady=5)
        self.song_listbox.selection_set(first=0)
        self.themes ={
            "Default": {"bg": "black", "fg": "white", "activebg": "#00336E", "highlightbg": "WHITE", "troughcolor": "black"},
            "Dark": {"bg": "#1C1C1C", "fg": "white", "activebg": "#00336E", "highlightbg": "WHITE", "troughcolor": "#1C1C1C"},
            "Light": {"bg": "white", "fg": "black", "activebg": "#0077FF", "highlightbg": "BLACK", "troughcolor": "#F0F0F0"},
            "orange": {"bg": "orange", "fg": "white", "activebg": "orange", "highlightbg": "WHITE", "troughcolor": "orange"}
            }
        self.theme_label = tk.Label(self.root, text="Select Theme:", font=("Times New Roman", 14), bg="black", fg="white")
        self.theme_label.pack(side=tk.TOP, padx=20, pady=5)
        self.theme_var = tk.StringVar()
        self.theme_var.set("Default")
        self.theme_menu = ttk.OptionMenu(self.root, self.theme_var, *self.themes.keys(), command=self.change_theme)
        self.theme_menu.pack(side=tk.TOP, padx=20, pady=5)
        self.root.mainloop()
    def play_song(self):
        selected_song = self.song_listbox.get(tk.ACTIVE)
        pygame.mixer.music.load(selected_song)
        pygame.mixer.music.play()

    def pause_song(self):
        pygame.mixer.music.pause()

    def stop_song(self):
        pygame.mixer.music.stop()

    def set_volume(self, val):
        volume = int(val) / 100
        pygame.mixer.music.set_volume(volume)

    def change_theme(self, theme):
        selected_theme = self.themes[theme]
        self.root.configure(bg=selected_theme["bg"])
        self.song_listbox.configure(bg=selected_theme["bg"], fg=selected_theme["fg"], selectbackground=selected_theme["activebg"], highlightbackground=selected_theme["highlightbg"])
        self.play_button.configure(bg=selected_theme["bg"], fg=selected_theme["fg"], activebackground=selected_theme["activebg"])
        self.pause_button.configure(bg=selected_theme["bg"], fg=selected_theme["fg"], activebackground=selected_theme["activebg"])
        self.stop_button.configure(bg=selected_theme["bg"], fg=selected_theme["fg"], activebackground=selected_theme["activebg"])
        self.volume_slider.configure(bg=selected_theme["bg"], fg=selected_theme["fg"], highlightbackground=selected_theme["highlightbg"], troughcolor=selected_theme["troughcolor"])

        pass

music_player = MICHIGANTECHMUSIC()
```

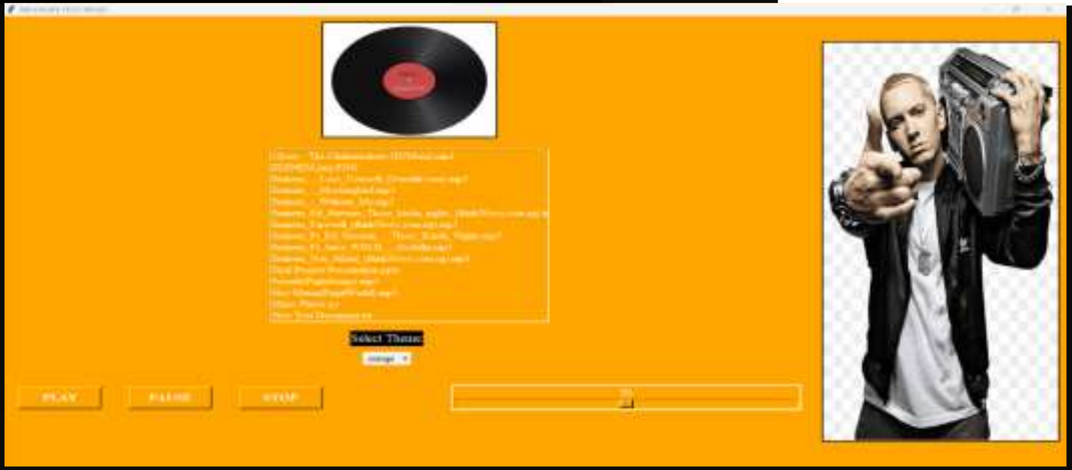
Output – Music interface


Closer - The Chainsmokers (DJMaza).mp3
EMINEM png PNG
Eminem_-_Lose_Yourself_(Svendfd.com).mp3
Eminem_-_Mockingbird.mp3
Eminem_-_Without_Me.mp3
Eminem_Ed_Sheeran_Those_kinda_nights_(thinkNews.com.ng).m
Eminem_Farewell_(thinkNews.com.ng).mp3
Eminem_Ft_Ed_Sheeran_-_Those_Kinda_Nights.mp3
Eminem_Ft_Juice_WRLD_-_Godzilla.mp3
Eminem_Not_Afraid_(thinkNews.com.ng).mp3
Final Project Presentation.pptx
Friends(PagalSongs).mp3
Hey Mama(PagalWorld).mp3
Music Player.py
New Text Document.txt

Select Theme
Default


Closer - The Chainsmokers (DJMaza).mp3
EMINEM png PNG
Eminem_-_Lose_Yourself_(Svendfd.com).mp3
Eminem_-_Mockingbird.mp3
Eminem_-_Without_Me.mp3
Eminem_Ed_Sheeran_Those_kinda_nights_(thinkNews.com.ng).m
Eminem_Farewell_(thinkNews.com.ng).mp3
Eminem_Ft_Ed_Sheeran_-_Those_Kinda_Nights.mp3
Eminem_Ft_Juice_WRLD_-_Godzilla.mp3
Eminem_Not_Afraid_(thinkNews.com.ng).mp3
Final Project Presentation.pptx
Friends(PagalSongs).mp3
Hey Mama(PagalWorld).mp3
Music Player.py
New Text Document.txt

Select Theme:
Light


Closer - The Chainsmokers (DJMaza).mp3
EMINEM png PNG
Eminem_-_Lose_Yourself_(Svendfd.com).mp3
Eminem_-_Mockingbird.mp3
Eminem_-_Without_Me.mp3
Eminem_Ed_Sheeran_Those_kinda_nights_(thinkNews.com.ng).m
Eminem_Farewell_(thinkNews.com.ng).mp3
Eminem_Ft_Ed_Sheeran_-_Those_Kinda_Nights.mp3
Eminem_Ft_Juice_WRLD_-_Godzilla.mp3
Eminem_Not_Afraid_(thinkNews.com.ng).mp3
Final Project Presentation.pptx
Friends(PagalSongs).mp3
Hey Mama(PagalWorld).mp3
Music Player.py
New Text Document.txt

Select Theme:
orange

Michigan Tech
1885

Current Music Applications & Trends

The backend programming for Spotify & Apple music was created using a microservices architecture. This indicates that the code has been divided into a number of separate, independent services, each of which is responsible for handling a particular task, such as playlist generation, user authentication, and music playback. Many computer languages, including Java, Scala, and Python, are used to create these services. They use a variety of data storage technologies, including Cassandra, MySQL, and Kafka, and they connect with one another through APIs (Application Programming Interfaces).A sizable, distributed database that is built to accommodate the enormous amounts of data that Spotify maintains houses the music data. To assess user listening patterns and produce tailored recommendations, Spotify combines its own algorithms with machine learning.

Michigan Tech

Thank You

Michigan Tech