import tkinter as tk

from tkinter import filedialog

import librosa

import librosa.display

import matplotlib.pyplot as plt

import numpy as np

import pygame

import wave

class AudioApp:

def \_init(self, root):

self.root = root\_

self.root.title("Audio Processing App")

pygame.mixer.init()

buttons = [

("Load Audio", self.load\_audio),

("Transform Audio", self.transform\_audio),

("Play Original Audio", self.play\_original\_audio),

("Play Transformed Audio", self.play\_transformed\_audio),

("Plot Spectrogram", self.plot\_spectrogram)

]

for text, command in buttons:

button = tk.Button(root\_, text=text, command=command)

button.pack(pady=10)

self.audio\_data = None

self.sample\_rate = None

self.transformed\_data = None

self.original\_file\_path = None

def load\_audio(self):

file\_path = filedialog.askopenfilename(title="Select Audio File", filetypes=[("Audio Files", ".mp3;.wav")])

if file\_path:

self.original\_file\_path = file\_path

self.audio\_data, self.sample\_rate = librosa.load(file\_path, sr=None)

print(f"Loaded {file\_path}")

self.transformed\_data = self.audio\_data.copy()

def transform\_audio(self):

if self.transformed\_data is not None:

self.transformed\_data = librosa.effects.pitch\_shift(self.transformed\_data, sr=self.sample\_rate, n\_steps=1)

print("Audio transformed")

def plot\_spectrogram(self):

if self.transformed\_data is not None:

stft = librosa.stft(self.transformed\_data)

spectrogram = librosa.amplitude\_to\_db(np.abs(stft), ref=np.max)

plt.figure(figsize=(8, 6))

librosa.display.specshow(spectrogram, sr=self.sample\_rate, x\_axis='time', y\_axis='log')

plt.colorbar(label='dB')

plt.title("Spectrogram")

plt.show()

@staticmethod

def play\_audio(file\_path):

pygame.mixer.music.load(file\_path)

pygame.mixer.music.play()

def play\_original\_audio(self):

if self.original\_file\_path is not None:

self.play\_audio(self.original\_file\_path)

def play\_transformed\_audio(self):

if self.transformed\_data is not None:

transformed\_file\_path = "transformed\_audio.wav" # Изменен путь сохранения

with wave.open(transformed\_file\_path, 'w') as wf:

wf.setnchannels(1) # моно

wf.setsampwidth(2) # 16-битные аудиоданные

wf.setframerate(self.sample\_rate)

wf.writeframes(self.transformed\_data.tobytes())

self.play\_audio(transformed\_file\_path)

if \_name\_ == "\_main\_":

root = tk.Tk()

app = AudioApp(root)

root.mainloop()