**Sample Code**

### from tkinter import messagebox

### from tkinter import \*

### from tkinter import simpledialog

### import tkinter

### from tkinter import filedialog

### from imutils import paths

### import numpy as np

### from collections import defaultdict

### from tkinter.filedialog importaskopenfilename

### from tkinter import simpledialog

### from keras.preprocessing.image import img\_to\_array

### from keras.models import load\_model

### import imutils

### import cv2

### import numpy as np

### import sys

### from tkinter import ttk

### import os

### from playsound import playsound

### main = tkinter.Tk()

### main.title("EMOTIONBASED MUSICRECOMMENDATION SYSTEM")

### main.geometry("1200x1200")

### global value

### global filename

### global faces

### global frame

### detection\_model\_path = 'haarcascade\_frontalface\_default.xml'

### emotion\_model\_path = '\_mini\_XCEPTION.106-0.65.hdf5'

### face\_detection = cv2.CascadeClassifier(detection\_model\_path)

### emotion\_classifier = load\_model(emotion\_model\_path, compile=False)

### EMOTIONS = ["angry","disgust","scared","happy", "sad", "surprised","neutral"]

### def upload():

### global filename

### global value

### filename = askopenfilename(initialdir = "images"

### pathlabel.config(text=filename)

### def preprocess():

### global filename

### global frame

### global faces

### text.delete('1.0', END)

### orig\_frame = cv2.imread(filename)

### orig\_frame = cv2.resize(orig\_frame, (48, 48))

### frame = cv2.imread(filename,0)

### faces = face\_detection.detectMultiScale(frame,scaleFactor=1.1,minNeighbors=5,minSize=(30,30),flags=cv2.CASCADE\_SCALE\_IMAGE)

### text.insert(END,"Total number of faces detected : "+str(len(faces)))

### def detectEmotion():

### global faces

### if len(faces) > 0:

### faces = sorted(faces, reverse=True,key=lambda x: (x[2] - x[0]) \* (x[3] - x[1]))[0]

### (fX, fY, fW, fH) = face

### roi = frame[fY:fY + fH, fX:fX + fW]

### roi = cv2.resize(roi, (48, 48))

### roi = roi.astype("float") / 255.0

### roi = img\_to\_array(roi)

### roi = np.expand\_dims(roi, axis=0)

### preds = emotion\_classifier.predict(roi)[0]

### emotion\_probability = np.max(preds)

### label = EMOTIONS[preds.argmax()]

### messagebox.showinfo("Emotion Prediction Screen","Emotion Detected As : "+label)

### value.clear()

### path = 'songs'

### for r, d, f in os.walk(path):

### for file in f:

### if file.find(label) != -1:

### value.append(file)

### else:

### messagebox.showinfo("Emotion Prediction Screen","No face detected in uploaded image")

### def playSong():

### name = songslist.get()

### playsound('songs/'+name)

### font = ('times', 20, 'bold')

### title = Label(main, text=’EMOTION RECOGNITION FROM IMAGES')

### title.config(bg='brown', fg='white')

### title.config(font=font)

### title.config(height=3, width=80)

### title.place(x=5,y=5)

### font1 = ('times', 14, 'bold')

### upload = Button(main, text="Upload Image With Face", command=upload)

### upload.place(x=50,y=100)

### upload.config(font=font1)

### pathlabel = Label(main)

### pathlabel.config(bg='brown', fg='white')

### pathlabel.config(font=font1)

### pathlabel.place(x=300,y=100)

### preprocessbutton = Button(main, text="Preprocess & Detect Face in Image", command=preprocess)

### preprocessbutton.place(x=50,y=150)

### preprocessbutton.config(font=font1)

### emotion = Button(main, text="Detect Emotion", command=detectEmotion)

### emotion.place(x=50,y=200)

### emotion.config(font=font1)

### emotionlabel = Label(main)

### emotionlabel.config(bg='brown', fg='white')

### emotionlabel.config(font=font1)

### emotionlabel.place(x=610,y=200)

### emotionlabel.config(text="Predicted Song")

### value = ["Song List"]

### songslist =ttk.Combobox(main,values=value,postcommand=lambda: songslist.configure(values=value))

### songslist.place(x=760,y=210)

### songslist.current(0)

### songslist.config(font=font1)

### playsong = Button(main, text="Play Song", command=playSong)

### playsong.place(x=50,y=250)

### playsong.config(font=font1)

### font1 = ('times', 12, 'bold')

### text=Text(main,height=10,width=150)

### scroll= gure(yscrollcommand=scroll.set)

### text.place(x=10,y=300)

### text.config(font=font1)

### main.config(bg='brown')

### main.mainloop()

### Scrollbar(text)

### text.config(font=font1)

### main.config(bg='brown')

### main.mainloop()