**Windows.py**

import tkinter as tk

from PIL import Image, ImageTk

root = tk.Tk()

root.title("Suryanamskar Pose Detection Using Machine Learning") w,h = root.winfo\_screenwidth(),root.winfo\_screenheight() root.geometry("%dx%d+0+0"%(w,h))

img = Image.open("D:/23 Protech/100% code/Suryanamskar Pose/yoga\_pose\_detection/y1.ico")

**# Replace with your image path if** img.size != (32, 32):

img = img.resize((32, 32), Image.LANCZOS) photo = ImageTk.PhotoImage(img) root.iconphoto(False, photo)

frame\_alpr = tk.LabelFrame(root,width=1350, height=700, bd=5, font=('times', 14,

'bold'),bg="#152238") frame\_alpr.grid(row=0, column=0) frame\_alpr.place(x=0, y=0)

img = Image.open('y2.png')

img = img.resize((150,150), Image.LANCZOS) logo\_image = ImageTk.PhotoImage(img)

logo\_label = tk.Label(root, image=logo\_image) logo\_label.image = logo\_image logo\_label.place(x=600, y=100)

lbl = tk.Label(root, text="Welcome to Suryanamskar Pose Detection \nUsing Machine

Learning", font=('times', 25,' bold underline '), height=2, width=45,bg="#152238",fg="white") lbl.place(x=250, y=300)

def start():

from subprocess import call call(['python','gui main.py'])

btn = tk.Button(root, text="Let's Start", bg="Yellow",fg="Black",font=("",20), width=9, height=1, command=start) btn.place(x=600, y=450)

root.mainloop()

**Gui.main.py**

import tkinter as tk **#from tkinter import \***

from tkinter import messagebox as ms import sqlite3

from PIL import Image, ImageTk import re import random import os

window = tk.Tk() window.geometry("700x700") window.title("REGISTRATION FORM") window.configure(background="#CCCCFF") img = Image.open("D:/23 Protech/100% code/Suryanamskar Pose/yoga\_pose\_detection/y1.ico")

**# Replace with your image path if** img.size != (32, 32):

img = img.resize((32, 32), Image.LANCZOS) photo = ImageTk.PhotoImage(img) window.iconphoto(False, photo)

Fullname = tk.StringVar() address = tk.StringVar() username = tk.StringVar()

Email = tk.StringVar() Phoneno = tk.IntVar() var = tk.IntVar() age = tk.IntVar() password = tk.StringVar() password1 = tk.StringVar() policeno = tk.IntVar() value = random.randint(1, 1000) print(value)

**# database code**

db = sqlite3.connect('evaluation.db') cursor = db.cursor()

cursor.execute("CREATE TABLE IF NOT EXISTS registration"

"(Fullname TEXT, address TEXT, username TEXT, Email TEXT, Phoneno

TEXT,Gender TEXT,age TEXT , password TEXT)") db.commit()

def password\_check(passwd):

SpecialSym =['$', '@', '#', '%']

val = True

if len(passwd) < 6: print('length should be at least 6') val = False

if len(passwd) > 20:

print('length should be not be greater than 8')

val = False

if not any(char.isdigit() for char in passwd):

print('Password should have at least one numeral')

val = False

if not any(char.isupper() for char in passwd):

print('Password should have at least one uppercase letter') val = False

if not any(char.islower() for char in passwd):

print('Password should have at least one lowercase letter') val = False

if not any(char in SpecialSym for char in passwd):

print('Password should have at least one of the symbols $@#') val = False if val:

return val

def insert():

fname = Fullname.get() addr = address.get() un = username.get() email = Email.get() mobile = Phoneno.get() gender = var.get() time = age.get() pwd = password.get() cnpwd = password1.get() with sqlite3.connect('evaluation.db') as db:

c = db.cursor()

**# Find Existing username if any take proper action** find\_user = ('SELECT \* FROM registration WHERE username = ?')

c.execute(find\_user, [(username.get())])

**# else:**

**# ms.showinfo('Success!', 'Account Created Successfully !')**

**# to check mail**

**#regex = '^\w+([\.-]?\w+)\*@\w+([\.-]?\w+)\*(\.\w{2,3})+$' regex='^[a-z0-9]+[\.\_]?[a-z0-9]+[@]\w+[.]\w{2,3}$' if (re.search(regex, email)):**

a = True else:

a = False # validation if (fname.isdigit() or (fname == "")): ms.showerror("showerror", "Error")

ms.showinfo("Message", "please enter valid name") elif (addr == ""):

ms.showerror("showerror", "Error")

ms.showinfo("Message", "Please Enter Address") elif (email == "") or (a == False): ms.showerror("showerror", "Error")

ms.showinfo("Message", "Please Enter valid email") elif((len(str(mobile)))<10 or len(str((mobile)))>10):

ms.showerror("showerror", "Error")

ms.showinfo("Message", "Please Enter 10 digit mobile number") elif ((time > 100) or (time == 0)): ms.showerror("showerror", "Error")

ms.showinfo("Message", "Please Enter valid age") elif (c.fetchall()):

ms.showerror('Error!', 'Username Taken Try a Diffrent One.') elif (pwd == ""):

ms.showerror("showerror", "Error")

ms.showinfo("Message", "Please Enter valid password") elif (var == False):

ms.showinfo("Message", "Please Enter gender") elif(pwd=="")or(password\_check(pwd))!=True:

ms.showerror("showerror", "Error")

ms.showinfo("Message", "password must contain atleast 1 Uppercase letter,1 symbol,1 number") elif (pwd != cnpwd):

ms.showerror("showerror", "Error")

ms.showinfo("Message", "Password Confirm password must be same") else:

conn = sqlite3.connect('evaluation.db') with conn:

cursor = conn.cursor() cursor.execute(

'INSERT INTO registration(Fullname, address, username, Email, Phoneno,

Gender, age , password) VALUES(?,?,?,?,?,?,?,?)',

(fname, addr, un, email, mobile, gender, time, pwd))

conn.commit() db.close()

ms.askquestion("askquestion", "Are you sure?") ms.askokcancel("askokcancel", "Want to continue?") ms.showinfo('Success!', 'Account Created Successfully !') **# window.destroy()**

from subprocess import call

call(["python", "login.py"])

window.destroy()

#########################################################################################

############################################################

def login():

from subprocess import call call(["python", "login.py"])

**#def register():**

**# from subprocess import call**

**# call(["python", "lecture\_login.py"])**

**# assign and define variable # def login():**

**#####For background Image**

**# image2 = Image.open('re1.jpg')**

**# image2 = image2.resize((700, 700), Image.LANCZOS)**

**# background\_image = ImageTk.PhotoImage(image2)**

**# background\_label = tk.Label(window, image=background\_image)**

**# background\_label.image = background\_image**

**# background\_label.place(x=0, y=0) # , relwidth=1, relheight=1)**

l1 = tk.Label(window, text="Registration Form", font=("Times new roman", 30, "bold italic"), bg="#192841", fg="white") l1.place(x=190, y=50)

**# that is for label1 registration**

l2 = tk.Label(window, text="FULL NAME :", width=12, font=("Times new roman", 15, "bold"), bg="#CCCCFF") l2.place(x=130, y=150)

t1 = tk.Entry(window, textvar=Fullname, width=20, font=('', 15)) t1.place(x=330, y=150)

**# that is for label 2 (full name)**

l3 = tk.Label(window, text="ADDRESS :", width=12, font=("Times new roman", 15, "bold"), bg="#CCCCFF") l3.place(x=130, y=200)

t2 = tk.Entry(window, textvar=address, width=20, font=('', 15)) t2.place(x=330, y=200)

**# that is for label 3(address)**

**# that is for label 4(blood group)**

l5 = tk.Label(window, text="MAIL-ID :", width=12, font=("Times new roman", 15, "bold"), bg="#CCCCFF") l5.place(x=130, y=250)

t4 = tk.Entry(window, textvar=Email, width=20, font=('', 15)) t4.place(x=330, y=250) **# that is for email address**

l6 = tk.Label(window, text="CONTACT NO :", width=12, font=("Times new roman", 15,

"bold"), bg="#CCCCFF") l6.place(x=130, y=300)

t5 = tk.Entry(window, textvar=Phoneno, width=20, font=('', 15))

t5.place(x=330, y=300)

**# phone number**

l7 = tk.Label(window, text="GENDER :", width=12, font=("Times new roman", 15, "bold"), bg="#CCCCFF")

l7.place(x=130, y=350)

**# gender**

tk.Radiobutton(window, text="MALE", padx=5, width=5, bg="#CCCCFF", font=("bold", 15), variable=var, value=1).place(x=330,

y=350)

tk.Radiobutton(window, text="FEMALE", padx=20, width=4, bg="#CCCCFF", font=("bold", 15), variable=var, value=2).place( x=440, y=350)

l8 = tk.Label(window, text="AGE:", width=12, font=("Times new roman", 15, "bold"), bg="#CCCCFF") l8.place(x=130, y=400)

t6 = tk.Entry(window, textvar=age, width=20, font=('', 15)) t6.place(x=330, y=400)

l4 = tk.Label(window, text="USERNAME", width=12, font=("Times new roman", 15, "bold"), bg="#CCCCFF") l4.place(x=130, y=450)

t3 = tk.Entry(window, textvar=username, width=20, font=('', 15)) t3.place(x=330, y=450)

l9 = tk.Label(window, text="PASSWORD", width=12, font=("Times new roman", 15, "bold"), bg="#CCCCFF") l9.place(x=130, y=500)

t9 = tk.Entry(window, textvar=password, width=20, font=('', 15), show="\*") t9.place(x=330, y=500)

l10 = tk.Label(window, text="C-PASSWORD", width=13, font=("Times new roman", 15, "bold"), bg="#CCCCFF")

l10.place(x=130, y=550)

t10 = tk.Entry(window, textvar=password1, width=20, font=('', 15), show="\*") t10.place(x=330, y=550)

btn = tk.Button(window, text="SUBMIT", bg="green",font=("",20),fg="white", width=9, height=0, command = insert) btn.place(x=250, y=600)

#btn = tk.Button(window, text="login", bg="#192841",font=("",20),fg="white", width=9, height=0, command=login) #btn.place(x=350, y=600)

# tologin=tk.Button(window , text="Go To Login", bg ="dark green", fg = "white", width=15, height=2, command=login) # tologin.place(x=330, y=600) window.mainloop()

**registration.py**

import tkinter as tk **# from tkinter import \***

from tkinter import messagebox as ms import sqlite3

from PIL import Image, ImageTk import re import random import os

window = tk.Tk()

window.geometry("700x700") window.title("REGISTRATION FORM") window.configure(background="#CCCCFF")

img = Image.open("D:/23 Protech/100% code/Suryanamskar Pose/yoga\_pose\_detection/y1.ico")

**# Replace with your image path if** img.size != (32, 32):

img = img.resize((32, 32), Image.LANCZOS) photo = ImageTk.PhotoImage(img) window.iconphoto(False, photo)

Fullname = tk.StringVar() address = tk.StringVar() username = tk.StringVar()

Email = tk.StringVar() Phoneno = tk.IntVar() var = tk.IntVar() age = tk.IntVar() password = tk.StringVar() password1 = tk.StringVar() policeno = tk.IntVar() value = random.randint(1, 1000) print(value)

**# database code**

db = sqlite3.connect('evaluation.db') cursor = db.cursor()

cursor.execute("CREATE TABLE IF NOT EXISTS registration"

"(Fullname TEXT, address TEXT, username TEXT, Email TEXT, Phoneno

TEXT,Gender TEXT,age TEXT , password TEXT)") db.commit()

def password\_check(passwd):

SpecialSym =['$', '@', '#', '%']

val = True

if len(passwd) < 6: print('length should be at least 6') val = False

if len(passwd) > 20:

print('length should be not be greater than 8') val = False

if not any(char.isdigit() for char in passwd):

print('Password should have at least one numeral') val = False

if not any(char.isupper() for char in passwd):

print('Password should have at least one uppercase letter') val = False

if not any(char.islower() for char in passwd):

print('Password should have at least one lowercase letter') val = False

if not any(char in SpecialSym for char in passwd):

print('Password should have at least one of the symbols $@#') val = False if val:

return val

def insert():

fname = Fullname.get()

addr = address.get() un = username.get() email = Email.get() mobile = Phoneno.get() gender = var.get() time = age.get() pwd = password.get() cnpwd = password1.get() with sqlite3.connect('evaluation.db') as db:

c = db.cursor()

**# Find Existing username if any take proper action** find\_user = ('SELECT \* FROM registration WHERE username = ?')

c.execute(find\_user, [(username.get())])

**# else:**

**# ms.showinfo('Success!', 'Account Created Successfully !')**

**# to check mail**

**#regex = '^\w+([\.-]?\w+)\*@\w+([\.-]?\w+)\*(\.\w{2,3})+$'** regex='^[a-z0-9]+[\.\_]?[a-z0-9]+[@]\w+[.]\w{2,3}$' if (re.search(regex, email)):

a = True else:

a = False # validation if (fname.isdigit() or (fname == "")): ms.showerror("showerror", "Error")

ms.showinfo("Message", "please enter valid name") elif (addr == ""):

ms.showerror("showerror", "Error")

ms.showinfo("Message", "Please Enter Address") elif (email == "") or (a == False): ms.showerror("showerror", "Error")

ms.showinfo("Message", "Please Enter valid email") elif((len(str(mobile)))<10 or len(str((mobile)))>10):

ms.showerror("showerror", "Error")

ms.showinfo("Message", "Please Enter 10 digit mobile number") elif ((time > 100) or (time == 0)): ms.showerror("showerror", "Error")

ms.showinfo("Message", "Please Enter valid age") elif (c.fetchall()):

ms.showerror('Error!', 'Username Taken Try a Diffrent One.') elif (pwd == ""):

ms.showerror("showerror", "Error")

ms.showinfo("Message", "Please Enter valid password") elif (var == False):

ms.showinfo("Message", "Please Enter gender") elif(pwd=="")or(password\_check(pwd))!=True:

ms.showerror("showerror", "Error")

ms.showinfo("Message", "password must contain atleast 1 Uppercase letter,1 symbol,1 number") elif (pwd != cnpwd):

ms.showerror("showerror", "Error")

ms.showinfo("Message", "Password Confirm password must be same") else:

conn = sqlite3.connect('evaluation.db') with conn:

cursor = conn.cursor() cursor.execute(

'INSERT INTO registration(Fullname, address, username, Email, Phoneno,

Gender, age , password) VALUES(?,?,?,?,?,?,?,?)',

(fname, addr, un, email, mobile, gender, time, pwd))

conn.commit() db.close() ms.askquestion("askquestion", "Are you sure?") ms.askokcancel("askokcancel", "Want to continue?") ms.showinfo('Success!', 'Account Created Successfully !')

**# window.destroy()** from subprocess import call call(["python", "login.py"]) window.destroy()

#########################################################################################

############################################################ def login():

from subprocess import call call(["python", "login.py"])

**#def register():**

**# from subprocess import call**

**# call(["python", "lecture\_login.py"])**

**# assign and define variable # def login():**

**#####For background Image**

**# image2 = Image.open('re1.jpg')**

**# image2 = image2.resize((700, 700), Image.LANCZOS)**

**# background\_image = ImageTk.PhotoImage(image2)**

**# background\_label = tk.Label(window, image=background\_image)**

**# background\_label.image = background\_image**

**# background\_label.place(x=0, y=0) # , relwidth=1, relheight=1)**

l1 = tk.Label(window, text="Registration Form", font=("Times new roman", 30, "bold italic"), bg="#192841", fg="white") l1.place(x=190, y=50)

**# that is for label1 registration**

l2 = tk.Label(window, text="FULL NAME :", width=12, font=("Times new roman", 15, "bold"), bg="#CCCCFF") l2.place(x=130, y=150)

t1 = tk.Entry(window, textvar=Fullname, width=20, font=('', 15)) t1.place(x=330, y=150)

**# that is for label 2 (full name)**

l3 = tk.Label(window, text="ADDRESS :", width=12, font=("Times new roman", 15, "bold"), bg="#CCCCFF") l3.place(x=130, y=200)

t2 = tk.Entry(window, textvar=address, width=20, font=('', 15)) t2.place(x=330, y=200)

**# that is for label 3(address)**

**# that is for label 4(blood group)**

l5 = tk.Label(window, text="MAIL-ID :", width=12, font=("Times new roman", 15, "bold"), bg="#CCCCFF") l5.place(x=130, y=250)

t4 = tk.Entry(window, textvar=Email, width=20, font=('', 15)) t4.place(x=330, y=250) # that is for email address

l6 = tk.Label(window, text="CONTACT NO :", width=12, font=("Times new roman", 15,

"bold"), bg="#CCCCFF") l6.place(x=130, y=300)

t5 = tk.Entry(window, textvar=Phoneno, width=20, font=('', 15)) t5.place(x=330, y=300)

# phone number

l7 = tk.Label(window, text="GENDER :", width=12, font=("Times new roman", 15, "bold"), bg="#CCCCFF")

l7.place(x=130, y=350)

# gender

tk.Radiobutton(window, text="MALE", padx=5, width=5, bg="#CCCCFF", font=("bold", 15), variable=var, value=1).place(x=330,

y=350)

tk.Radiobutton(window, text="FEMALE", padx=20, width=4, bg="#CCCCFF", font=("bold", 15), variable=var, value=2).place( x=440, y=350)

l8 = tk.Label(window, text="AGE:", width=12, font=("Times new roman", 15, "bold"), bg="#CCCCFF") l8.place(x=130, y=400)

t6 = tk.Entry(window, textvar=age, width=20, font=('', 15)) t6.place(x=330, y=400)

l4 = tk.Label(window, text="USERNAME", width=12, font=("Times new roman", 15, "bold"), bg="#CCCCFF") l4.place(x=130, y=450)

t3 = tk.Entry(window, textvar=username, width=20, font=('', 15)) t3.place(x=330, y=450)

l9 = tk.Label(window, text="PASSWORD", width=12, font=("Times new roman", 15, "bold"), bg="#CCCCFF") l9.place(x=130, y=500)

t9 = tk.Entry(window, textvar=password, width=20, font=('', 15), show="\*") t9.place(x=330, y=500)

l10 = tk.Label(window, text="C-PASSWORD", width=13, font=("Times new roman", 15, "bold"), bg="#CCCCFF")

l10.place(x=130, y=550)

t10 = tk.Entry(window, textvar=password1, width=20, font=('', 15), show="\*") t10.place(x=330, y=550)

btn = tk.Button(window, text="SUBMIT", bg="green",font=("",20),fg="white", width=9, height=0, command = insert) btn.place(x=250, y=600)

**#btn = tk.Button(window, text="login", bg="#192841",font=("",20),fg="white", width=9, height=0, command=login) #btn.place(x=350, y=600)**

**# tologin=tk.Button(window , text="Go To Login", bg ="dark green", fg = "white", width=15, height=2, command=login) # tologin.place(x=330, y=600)** window.mainloop()

**Login.py**

import tkinter as tk

from tkinter import ttk, LEFT, END from tkinter import messagebox as ms import sqlite3

from PIL import Image, ImageTk import re ##############################################+==========================================

=================== root = tk.Tk()

root.configure(background="white") # root.geometry("1300x700")

img = Image.open("D:/23 Protech/100% code/Suryanamskar Pose/yoga\_pose\_detection/y1.ico")

**# Replace with your image path if** img.size != (32, 32):

img = img.resize((32, 32), Image.LANCZOS) photo = ImageTk.PhotoImage(img) root.iconphoto(False, photo)

w, h = root.winfo\_screenwidth(), root.winfo\_screenheight() root.geometry("550x400") root.title("Login Form")

username = tk.StringVar() password = tk.StringVar()

def registration():

from subprocess import call call(["python","registration.py"]) root.destroy()

def login():

**# Establish Connection**

with sqlite3.connect('evaluation.db') as db:

c = db.cursor()

**# Find user If there is any take proper action** db = sqlite3.connect('evaluation.db') cursor = db.cursor()

cursor.execute("CREATE TABLE IF NOT EXISTS registration"

"(Fullname TEXT, address TEXT, username TEXT, Email TEXT,

Phoneno TEXT,Gender TEXT,age TEXT , password TEXT)") db.commit()

find\_entry = ('SELECT \* FROM registration WHERE username = ? and password = ?')

c.execute(find\_entry, [(username.get()), (password.get())]) result = c.fetchall()

if result: msg = ""

**# self.logf.pack\_forget()**

**# self.head['text'] = self.username.get() + '\n Loged In'**

**# msg = self.head['text']**

**# self.head['pady'] = 150** print(msg)

ms.showinfo("messege", "you have successfully log-in") from subprocess import call call(['python','Home.py'])

**# ===========================================** root.destroy()

**# ================================================**  else:

ms.showerror('Oops!', 'Username Or Password Did Not Found/Match.') title=tk.Label(root, text="Login Here", font=("Algerian", 30,

"bold"),bd=5,bg="white",fg="blue") title.place(x=150,y=50,width=250)

Login\_frame=tk.Frame(root,bg="white")

Login\_frame.place(x=50,y=130)

lbluser=tk.Label(Login\_frame,text="Username",compound=LEFT,font=("Times new roman", 20,

"bold"),bg="white").grid(row=1,column=0,padx=20,pady=10)

txtuser=tk.Entry(Login\_frame,bd=5,textvariable=username,font=("",15)) txtuser.grid(row=1,column=1,padx=20)

lblpass=tk.Label(Login\_frame,text="Password",compound=LEFT,font=("Times new roman", 20,

"bold"),bg="white").grid(row=2,column=0,padx=50,pady=10)

txtpass=tk.Entry(Login\_frame,bd=5,textvariable=password,show="\*",font=("",15)) txtpass.grid(row=2,column=1,padx=20)

btn\_log=tk.Button(Login\_frame,text="SUBMIT",command=login,width=15,font=("Times new roman", 14, "bold"),bg="Green",fg="white") btn\_log.grid(row=3,column=1,pady=10)

btn\_reg=tk.Button(Login\_frame,text="SIGN-HERE",command=registration,width=15,font=("Times new roman", 14, "bold"),bg="red",fg="white") btn\_reg.grid(row=3,column=0,pady=10)

root.mainloop()

**home.py**

import tkinter as tk

from tkinter import ttk, LEFT, END from PIL import Image, ImageTk

from tkinter.filedialog import askopenfilename from tkinter import messagebox as ms import cv2 import sqlite3 import os import numpy as np import time import tkinter as tk

from moviepy.editor import VideoFileClip

root = tk.Tk() global fn fn = ""

**##############################################+========================================== ===================**

root.configure(background="brown")

**# root.geometry("1300x700")**

w, h = root.winfo\_screenwidth(), root.winfo\_screenheight() root.geometry("%dx%d+0+0" % (w, h))

root.title("Suryanamskar Pose Detection Using Machine Learning")

img = Image.open("D:/23 Protech/100% code/Suryanamskar Pose/yoga\_pose\_detection/y1.ico")

**# Replace with your image path if** img.size != (32, 32):

img = img.resize((32, 32), Image.LANCZOS) photo = ImageTk.PhotoImage(img) root.iconphoto(False, photo)

image2 = Image.open('y3.jpg')

image2 = image2.resize((1400,700))

background\_image = ImageTk.PhotoImage(image2)

background\_label = tk.Label(root, image=background\_image)

background\_label.image = background\_image

background\_label.place(x=0, y=0)

label\_l1 = tk.Label(root, text="Yoga is not a work-out, it is a work-in\nAnd this is the point of spiritual practice to open up your hearts",font=("Times New Roman", 25, 'bold italic'),

background="white", fg="black", width=75, height=2) label\_l1.place(x=0, y=0)

def play\_video():

cap = cv2.VideoCapture('D:/23 Protech/100% code/Suryanamskar

Pose/yoga\_pose\_detection/yoga1.mp4')

**# Check if camera opened successfully** if (cap.isOpened()== False):

print("Error opening video stream or file")

**# Read until video is completed** while(cap.isOpened()):

**# Capture frame-by-frame** ret, frame = cap.read() if ret == True:

**# Display the resulting frame** cv2.imshow('Frame',frame)

**# Press Q on keyboard to exit** if cv2.waitKey(40) & 0xFF == ord('q'):

break

**# Break the loop** else: break

**# When everything done, release the video capture object**  cap.release()

**# Closes all the frames** cv2.destroyAllWindows()

**################################$%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %%%%%%%%%%%%%%%%%%%%%%**

def action():

from subprocess import call call(["python","GUI\_Masters.py"])

**#########################################################################################**

**########################**

def window():

root.destroy()

frame = tk.LabelFrame(root, text=" --Lets Begin-- ", width=250, height=350, bd=5, font=('times', 14, ' bold '),bg="black",fg="white")

frame.grid(row=0, column=0, sticky='nw') frame.place(x=10, y=100)

button3 = tk.Button(frame, text="Detection",command=action, width=10, height=1, bg="white", fg="black",font=('times', 20, ' bold ')) button3.place(x=30, y=50)

button3 = tk.Button(frame, text="S-Video",command=play\_video, width=10, height=1, bg="white", fg="black",font=('times', 20, ' bold ')) button3.place(x=30, y=150)

exit = tk.Button(frame, text="Exit", command=window, width=10, height=1, font=('times',

20, ' bold '), bg="red",fg="white") exit.place(x=30, y=250)

root.mainloop()

**GUI.master.py**

import mediapipe as mp **# Import mediapipe** import cv2 **# Import opencv** import pandas as pd import numpy as np

from sklearn.model\_selection import train\_test\_split from sklearn.pipeline import make\_pipeline from sklearn.preprocessing import StandardScaler

from sklearn.linear\_model import LogisticRegression, RidgeClassifier from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier from sklearn.metrics import accuracy\_score **# Accuracy metrics** import pickle import cv2 import pyttsx3

**# Initialize the pyttsx3 engine**

engine = pyttsx3.init()

**# Initialize pyttsx3**

mp\_drawing = mp.solutions.drawing\_utils **# Drawing helpers**

mp\_holistic = mp.solutions.holistic **# Mediapipe Solutions**

with open('D:/23 Protech/100% code/Suryanamskar Pose/yoga\_pose\_detection/Training

Files/body\_language.pkl1', 'rb') as f: model = pickle.load(f) print(model)

cap = cv2.VideoCapture(0) height=900 **#for some reason, ANYTHING else works for my HD camera for example 1079..**

width=2000

cap.set(cv2.CAP\_PROP\_FRAME\_HEIGHT, height) cap.set(cv2.CAP\_PROP\_FRAME\_WIDTH, width)

**# Initiate holistic model**

with mp\_holistic.Holistic(min\_detection\_confidence=0.5, min\_tracking\_confidence=0.5) as holistic: while cap.isOpened():

ret, frame = cap.read()

**# Recolor Feed**

image = cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB) image.flags.writeable = False

**# Make Detections**

results = holistic.process(image)

**# print(results.face\_landmarks)**

**# face\_landmarks, pose\_landmarks, left\_hand\_landmarks, right\_hand\_landmarks**

**# Recolor image back to BGR for rendering** image.flags.writeable = True

image = cv2.cvtColor(image, cv2.COLOR\_RGB2BGR)

**# 1. Draw face landmarks**

mp\_drawing.draw\_landmarks(image, results.face\_landmarks, mp\_holistic.FACEMESH\_CONTOURS,

mp\_drawing.DrawingSpec(color=(80,110,10), thickness=1, circle\_radius=1),

mp\_drawing.DrawingSpec(color=(80,256,121), thickness=1, circle\_radius=1)

)

**# 2. Right hand**

mp\_drawing.draw\_landmarks(image, results.right\_hand\_landmarks, mp\_holistic.HAND\_CONNECTIONS,

mp\_drawing.DrawingSpec(color=(80,22,10), thickness=2, circle\_radius=4),

mp\_drawing.DrawingSpec(color=(80,44,121), thickness=2, circle\_radius=2)

)

**# 3. Left Hand**

mp\_drawing.draw\_landmarks(image, results.left\_hand\_landmarks, mp\_holistic.HAND\_CONNECTIONS,

mp\_drawing.DrawingSpec(color=(121,22,76), thickness=2, circle\_radius=4),

mp\_drawing.DrawingSpec(color=(121,44,250), thickness=2, circle\_radius=2)

)

**# 4. Pose Detections**

mp\_drawing.draw\_landmarks(image, results.pose\_landmarks, mp\_holistic.POSE\_CONNECTIONS,

mp\_drawing.DrawingSpec(color=(245,117,66), thickness=2, circle\_radius=4),

mp\_drawing.DrawingSpec(color=(245,66,230), thickness=2, circle\_radius=2)

) **# Export coordinates**  try:

**# Extract Pose landmarks**

pose = results.pose\_landmarks.landmark

pose\_row = list(np.array([[landmark.x, landmark.y, landmark.z, landmark.visibility] for landmark in pose]).flatten())

**# Extract Face landmarks**

face = results.face\_landmarks.landmark

face\_row = list(np.array([[landmark.x, landmark.y, landmark.z, landmark.visibility] for landmark in face]).flatten())

**# Concate rows**

row = pose\_row+face\_row

**# # Append class name**

**# row.insert(0, class\_name)**

**# # Export to CSV**

**# with open('coords.csv', mode='a', newline='') as f:**

**# csv\_writer = csv.writer(f, delimiter=',', quotechar='"',** quoting=csv.QUOTE\_MINIMAL)

**#csv\_writer.writerow(row)**

**# Make Detections** X = pd.DataFrame([row])

body\_language\_class = model.predict(X)[0] body\_language\_prob = model.predict\_proba(X)[0] print(body\_language\_class, body\_language\_prob)

**# Grab ear coords** coords = tuple(np.multiply( np.array(

(results.pose\_landmarks.landmark[mp\_holistic.PoseLandmark.LEFT\_EAR].x,

results.pose\_landmarks.landmark[mp\_holistic.PoseLandmark.LEFT\_EAR].y))

, [640,480]).astype(int))

cv2.rectangle(image,

(coords[0], coords[1]+5),

(coords[0]+len(body\_language\_class)\*20, coords[1]-30),

(245, 117, 16), -1)

cv2.putText(image, body\_language\_class, coords,

cv2.FONT\_HERSHEY\_SIMPLEX, 1, (255, 255, 255), 2, cv2.LINE\_AA)

**# Get status box**

cv2.rectangle(image, (0,0), (250, 60), (245, 117, 16), -1)

**# Display Class**

cv2.putText(image, 'CLASS'

, (95,12), cv2.FONT\_HERSHEY\_SIMPLEX, 0.5, (0, 0, 0), 1, cv2.LINE\_AA)

cv2.putText(image, body\_language\_class.split(' ')[0]

, (90,40), cv2.FONT\_HERSHEY\_SIMPLEX, 1, (255, 255, 255), 2, cv2.LINE\_AA)

**# Display Probability** cv2.putText(image, 'PROB'

, (15,12), cv2.FONT\_HERSHEY\_SIMPLEX, 0.5, (0, 0, 0), 1, cv2.LINE\_AA)

cv2.putText(image,

str(round(body\_language\_prob[np.argmax(body\_language\_prob)],2))

, (10,40), cv2.FONT\_HERSHEY\_SIMPLEX, 1, (255, 255, 255), 2, cv2.LINE\_AA)

cv2.putText(image, 'CLASS', (95, 12), cv2.FONT\_HERSHEY\_SIMPLEX, 0.5, (0, 0,

0), 1, cv2.LINE\_AA)

cv2.putText(image, body\_language\_class.split(' ')[0], (90, 40), cv2.FONT\_HERSHEY\_SIMPLEX, 1, (255, 255, 255), 2, cv2.LINE\_AA)

**# Convert the text to speech**

text = ' '.join([s for s in body\_language\_class.split(' ') if s not in

['Left', 'Right']]) engine.say(text)

cv2.imwrite('image.jpg', image)

except: pass

cv2.imshow('Raw Webcam Feed', image)

if cv2.waitKey(10) & 0xFF == ord('q'):

break

**# Play the audio**

**# engine.runAndWait()**

**# # Close the window if 'q' is pressed**

**# if engine == ord('q'): # break**

**# except:**

**# pass**

**# cv2.imshow('Raw Webcam Feed', image)**

**# if cv2.waitKey(10) & 0xFF == ord('q'):**

**# break cap.release()** cv2.destroyAllWindows()