**import numpy as np**

**import pandas as pd**

**import tensorflow as tf**

**from tensorflow.keras.preprocessing.image import ImageDataGenerator**

**from tensorflow.keras.applications import ResNet50**

**from tensorflow.keras.layers import Dense, GlobalAveragePooling2D**

**from tensorflow.keras.models import Model**

**# Load the dataset**

**# Replace this with your data loading code**

**data = pd.read\_csv("personality\_dataset.csv") # Assuming you have a CSV file with image paths and labels**

**# Data preprocessing**

**# Assuming you have a directory structure where images are stored in folders corresponding to their labels**

**train\_datagen = ImageDataGenerator(rescale=1./255, validation\_split=0.2)**

**train\_generator = train\_datagen.flow\_from\_dataframe(**

**dataframe=data,**

**directory="images/",**

**x\_col="image\_path",**

**y\_col="personality\_traits",**

**target\_size=(224, 224),**

**batch\_size=32,**

**class\_mode="categorical",**

**subset="training"**

**)**

**validation\_generator = train\_datagen.flow\_from\_dataframe(**

**dataframe=data,**

**directory="images/",**

**x\_col="image\_path",**

**y\_col="personality\_traits",**

**target\_size=(224, 224),**

**batch\_size=32,**

**class\_mode="categorical",**

**subset="validation"**

**)**

**# Load the pre-trained ResNet50 model without the top layer**

**base\_model = ResNet50(weights='imagenet', include\_top=False)**

**# Add a global average pooling layer and a dense layer for classification**

**x = base\_model.output**

**x = GlobalAveragePooling2D()(x)**

**predictions = Dense(num\_classes, activation='softmax')(x)**

**# Create the full model**

**model = Model(inputs=base\_model.input, outputs=predictions)**

**# Freeze the pre-trained layers**

**for layer in base\_model.layers:**

**layer.trainable = False**

**# Compile the model**

**model.compile(optimizer='adam',**

**loss='categorical\_crossentropy',**

**metrics=['accuracy'])**

**# Train the model**

**history = model.fit(**

**train\_generator,**

**steps\_per\_epoch=train\_generator.samples // train\_generator.batch\_size,**

**validation\_data=validation\_generator,**

**validation\_steps=validation\_generator.samples // validation\_generator.batch\_size,**

**epochs=10**

**)**

**# Save the trained model**

**model.save("personality\_prediction\_model.h5")**

**# Perform inference on new images**

**# Load the model**

**# model = tf.keras.models.load\_model("personality\_prediction\_model.h5")**

**# predicted\_traits = model.predict(new\_images)**