* Data Cleaning:
  + It checks images in a specified directory ('data') for their format and removes those not in the list of allowed extensions ('jpeg', 'jpg', 'bmp', 'png').
  + It uses OpenCV (cv2) and imghdr for image processing.
* Data Loading:
  + It uses TensorFlow's image\_dataset\_from\_directory to load images from the 'data' directory as a dataset.
* Data Preprocessing:
  + It normalizes the image data by dividing it by 255.
  + It splits the dataset into training, validation, and test sets.
* Model Definition:
  + It defines a convolutional neural network (CNN) model using Keras Sequential API.
  + The model consists of convolutional layers with max-pooling, followed by flattening and dense layers.
  + It uses the Adam optimizer and binary cross-entropy loss for compilation.
* Training the Model:
  + The model is trained using the training set (train) for 20 epochs, with validation on the validation set (val).
* Visualization of Training History:
  + The code includes commented-out sections for plotting training and validation loss, as well as accuracy over epochs using Matplotlib.
* Evaluation Metrics:
  + It computes precision, recall, and accuracy on the test set after training.
* Testing on New Image:
  + It loads a sample image ('happytest.jpg'), performs color correction, and resizes it to the model's input shape.
  + It uses the trained model to predict whether the image represents a "happy" or "sad" class based on the threshold of 0.5.
* Model Saving and Loading:
  + It saves the trained model to a file ('happysadmodel.h5') in the 'models' directory.
  + It loads the saved model and makes predictions on the previously mentioned test image.
* Display Predictions:
  + The final output prints whether the predicted class is "sad" or "happy" based on the loaded model.