In [2]:
```
import pandas as pd
import numpy as np
import os
import seaborn as sns
import matplotlib.pyplot as plt
!pip install -U librosa
import librosa
import librosa.display
from IPython.display import Audio
import warnings
warnings.filterwarnings('ignore')
```

```
Requirement already satisfied: librosa in d:\anaconda3\lib\site-packages (0.10.1)
Requirement already satisfied: audioread>=2.1.9 in d:\anaconda3\lib\site-packages (fr
om librosa) (3.0.1)
Requirement already satisfied: numpy!=1.22.0,!=1.22.1,!=1.22.2,>=1.20.3 in d:\anacond
a3\lib\site-packages (from librosa) (1.24.3)
Requirement already satisfied: scipy>=1.2.0 in d:\anaconda3\lib\site-packages (from l
ibrosa) (1.11.1)
Requirement already satisfied: scikit-learn>=0.20.0 in d:\anaconda3\lib\site-packages
(from librosa) (1.3.0)
Requirement already satisfied: joblib>=0.14 in d:\anaconda3\lib\site-packages (from l
ibrosa) (1.2.0)
Requirement already satisfied: decorator>=4.3.0 in d:\anaconda3\lib\site-packages (fr
om librosa) (5.1.1)
Requirement already satisfied: numba>=0.51.0 in d:\anaconda3\lib\site-packages (from
librosa) (0.57.1)
Requirement already satisfied: soundfile>=0.12.1 in d:\anaconda3\lib\site-packages (f
rom librosa) (0.12.1)
Requirement already satisfied: pooch>=1.0 in d:\anaconda3\lib\site-packages (from lib
rosa) (1.8.0)
Requirement already satisfied: soxr>=0.3.2 in d:\anaconda3\lib\site-packages (from li
brosa) (0.3.7)
Requirement already satisfied: typing-extensions>=4.1.1 in d:\anaconda3\lib\site-pack
ages (from librosa) (4.7.1)
Requirement already satisfied: lazy-loader>=0.1 in d:\anaconda3\lib\site-packages (fr
om librosa) (0.2)
Requirement already satisfied: msgpack>=1.0 in d:\anaconda3\lib\site-packages (from l
ibrosa) (1.0.3)
Requirement already satisfied: llvmlite<0.41,>=0.40.0dev0 in d:\anaconda3\lib\site-pa
ckages (from numba>=0.51.0->librosa) (0.40.0)
Requirement already satisfied: platformdirs>=2.5.0 in d:\anaconda3\lib\site-packages
(from pooch>=1.0->librosa) (3.10.0)
Requirement already satisfied: packaging>=20.0 in d:\anaconda3\lib\site-packages (fro
m pooch>=1.0->librosa) (23.1)
Requirement already satisfied: requests>=2.19.0 in d:\anaconda3\lib\site-packages (fr
om pooch>=1.0->librosa) (2.31.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in d:\anaconda3\lib\site-packages
(from scikit-learn>=0.20.0->librosa) (2.2.0)
Requirement already satisfied: cffi>=1.0 in d:\anaconda3\lib\site-packages (from soun
dfile>=0.12.1->librosa) (1.15.1)
Requirement already satisfied: pycparser in d:\anaconda3\lib\site-packages (from cffi
>=1.0->soundfile>=0.12.1->librosa) (2.21)
Requirement already satisfied: charset-normalizer<4,>=2 in d:\anaconda3\lib\site-pack
ages (from requests>=2.19.0->pooch>=1.0->librosa) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in d:\anaconda3\lib\site-packages (from r
equests>=2.19.0->pooch>=1.0->librosa) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in d:\anaconda3\lib\site-packages
(from requests>=2.19.0->pooch>=1.0->librosa) (1.26.16)
Requirement already satisfied: certifi>=2017.4.17 in d:\anaconda3\lib\site-packages
(from requests>=2.19.0->pooch>=1.0->librosa) (2023.7.22)
```

In [3]:

```
paths = []
labels = []
for dirname, _, filenames in os.walk(r'C:\Users\ASUS\Desktop\ml project\TESS Toronto e
    for filename in filenames:
        paths.append(os.path.join(dirname, filename))
        label = filename.split('_')[-1]
        label = label.split('.')[0]
        labels.append(label.lower())
    if len(paths) == 2800:
```

```
        break
print('Dataset is Loaded')
```

```
Dataset is Loaded
```

In [4]:
```
len(paths)
```

Out[4]:
```
2800
```

In [5]:
```
paths[:5]
```

Out[5]:
```
['C:\\Users\\ASUS\\Desktop\\ml project\\TESS Toronto emotional speech set data\\OAF_angry\\OAF_back_angry.wav',
 'C:\\Users\\ASUS\\Desktop\\ml project\\TESS Toronto emotional speech set data\\OAF_angry\\OAF_bar_angry.wav',
 'C:\\Users\\ASUS\\Desktop\\ml project\\TESS Toronto emotional speech set data\\OAF_angry\\OAF_base_angry.wav',
 'C:\\Users\\ASUS\\Desktop\\ml project\\TESS Toronto emotional speech set data\\OAF_angry\\OAF_bath_angry.wav',
 'C:\\Users\\ASUS\\Desktop\\ml project\\TESS Toronto emotional speech set data\\OAF_angry\\OAF_bean_angry.wav']
```

In [6]:
```
labels[:5]
```

Out[6]:
```
['angry', 'angry', 'angry', 'angry', 'angry']
```

In [7]:
```
## Create a dataframe
df = pd.DataFrame()
df['speech'] = paths
df['label'] = labels
df.head()
```

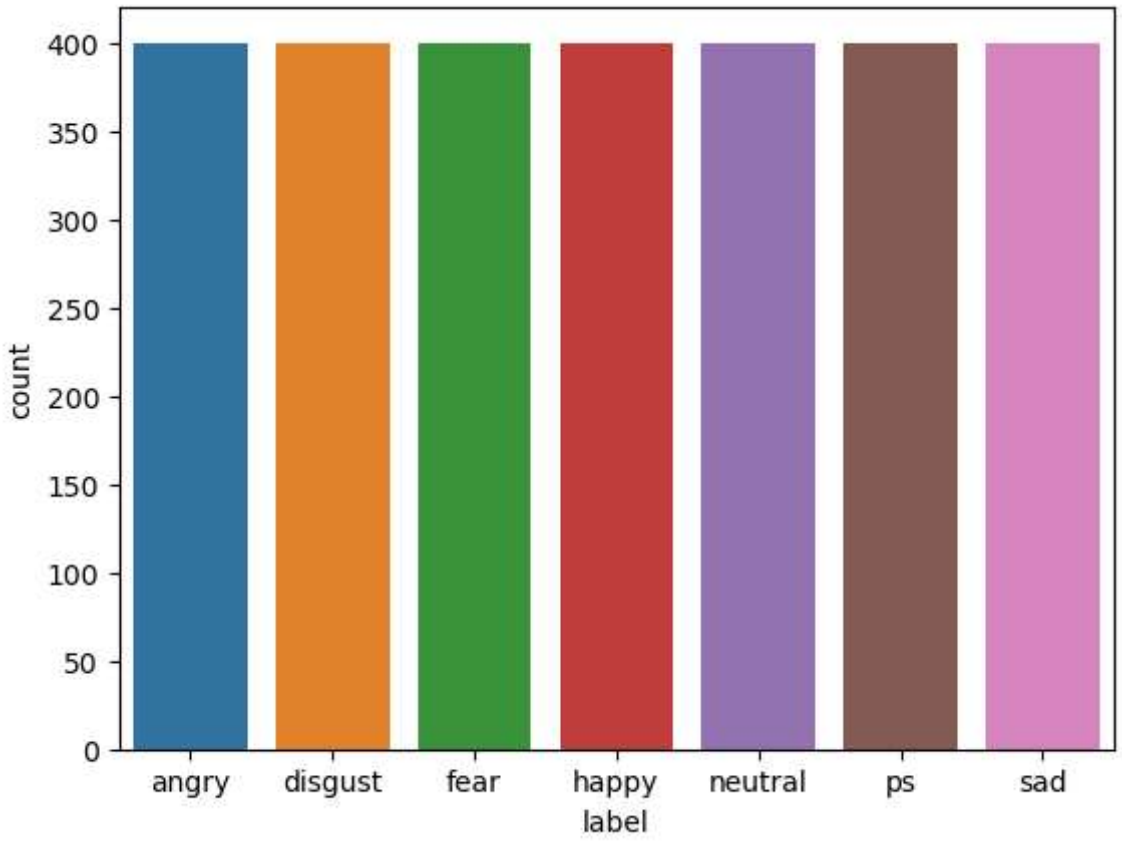
Out[7]:

	speech	label
0	C:\Users\ASUS\Desktop\ml project\TESS Toronto ...	angry
1	C:\Users\ASUS\Desktop\ml project\TESS Toronto ...	angry
2	C:\Users\ASUS\Desktop\ml project\TESS Toronto ...	angry
3	C:\Users\ASUS\Desktop\ml project\TESS Toronto ...	angry
4	C:\Users\ASUS\Desktop\ml project\TESS Toronto ...	angry

In [8]:
```
df['label'].value_counts()
```

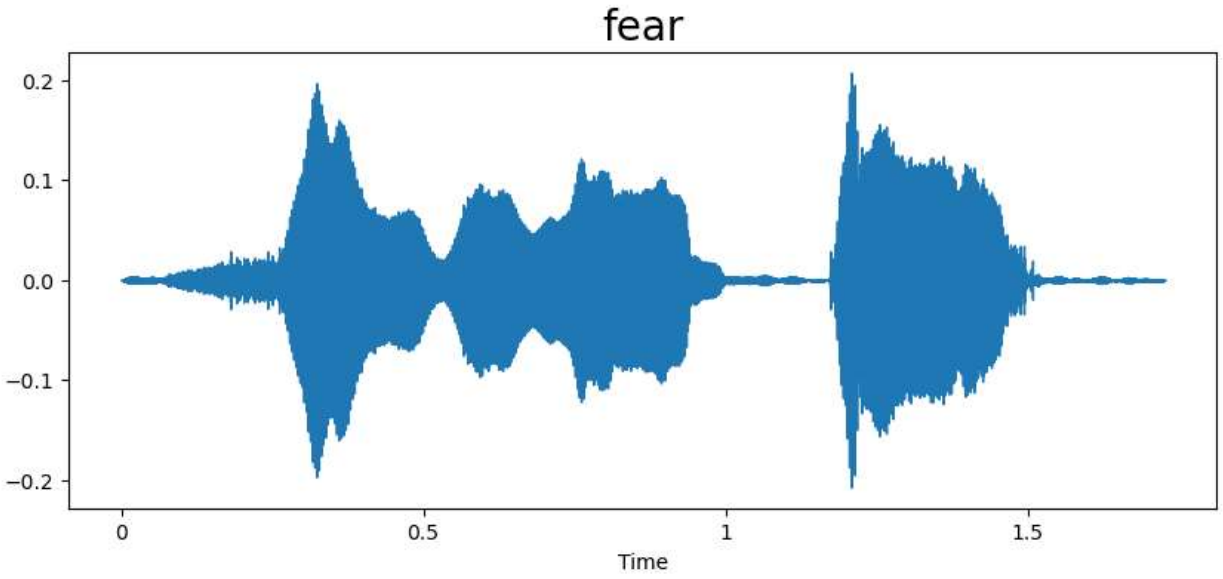
Out[8]:
```
label
angry      400
disgust    400
fear       400
happy      400
neutral    400
ps         400
sad        400
Name: count, dtype: int64
```

In [9]:
```
sns.countplot(x='label', data=df)
plt.show()
```



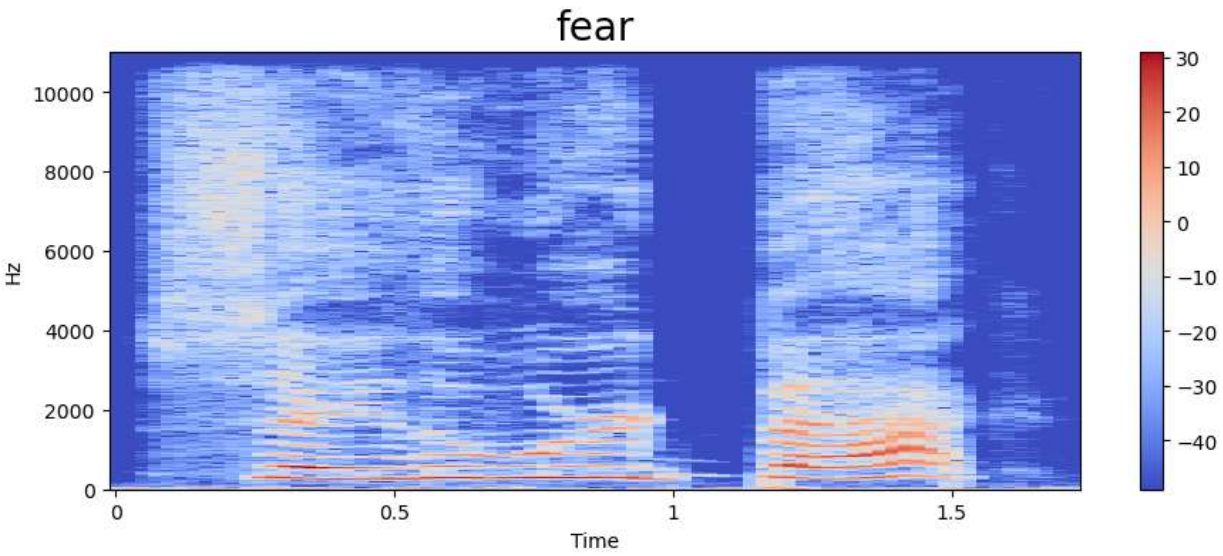
In [10]:
```
def waveplot(data, sr, emotion):
    plt.figure(figsize=(10,4))
    plt.title(emotion, size=20)
    librosa.display.waveshow(data, sr=sr)
    plt.show()

def spectogram(data, sr, emotion):
    x = librosa.stft(data)
    xdb = librosa.amplitude_to_db(abs(x))
    plt.figure(figsize=(11,4))
    plt.title(emotion, size=20)
    librosa.display.specshow(xdb, sr=sr, x_axis='time', y_axis='hz')
    plt.colorbar()
```

In [11]:
```
emotion = 'fear'
path = np.array(df['speech'][df['label']==emotion])[0]
data, sampling_rate = librosa.load(path)
waveplot(data, sampling_rate, emotion)
spectogram(data, sampling_rate, emotion)
Audio(path)
```



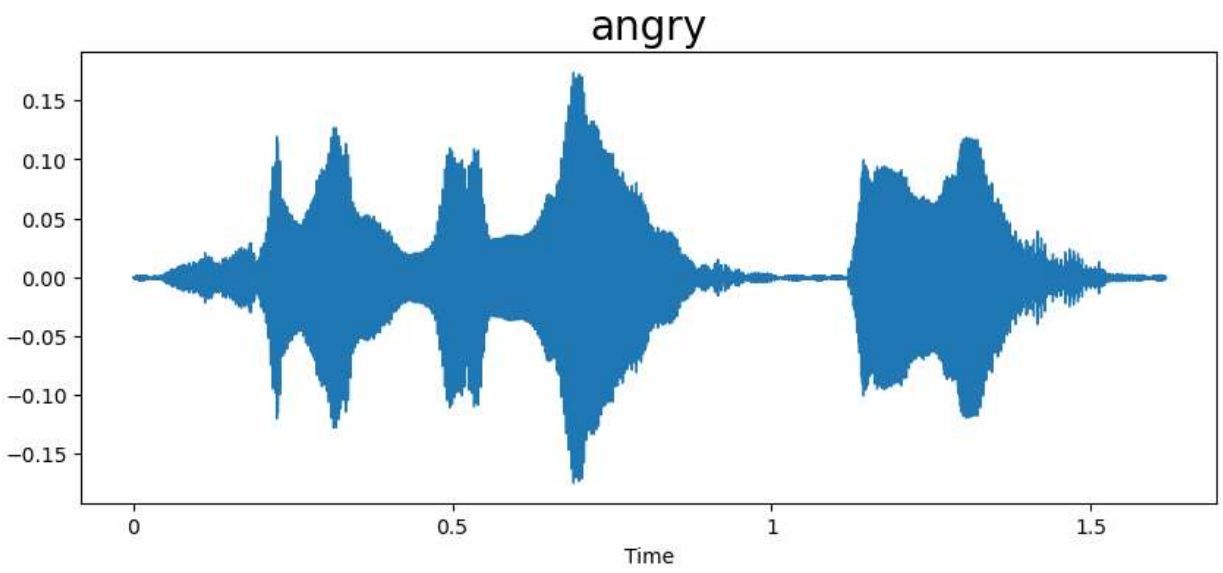
Out[11]:

0:00 / 0:01

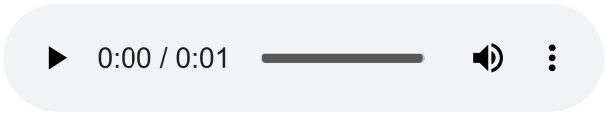


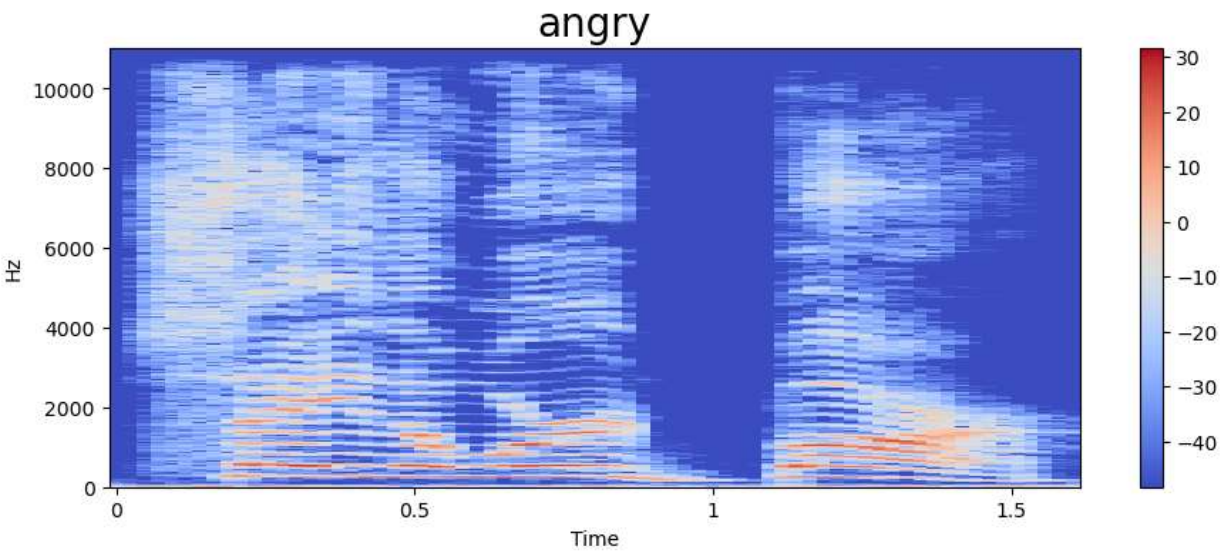
In [12]:

```
emotion = 'angry'
path = np.array(df['speech'][df['label']==emotion])[1]
data, sampling_rate = librosa.load(path)
waveplot(data, sampling_rate, emotion)
spectogram(data, sampling_rate, emotion)
Audio(path)
```



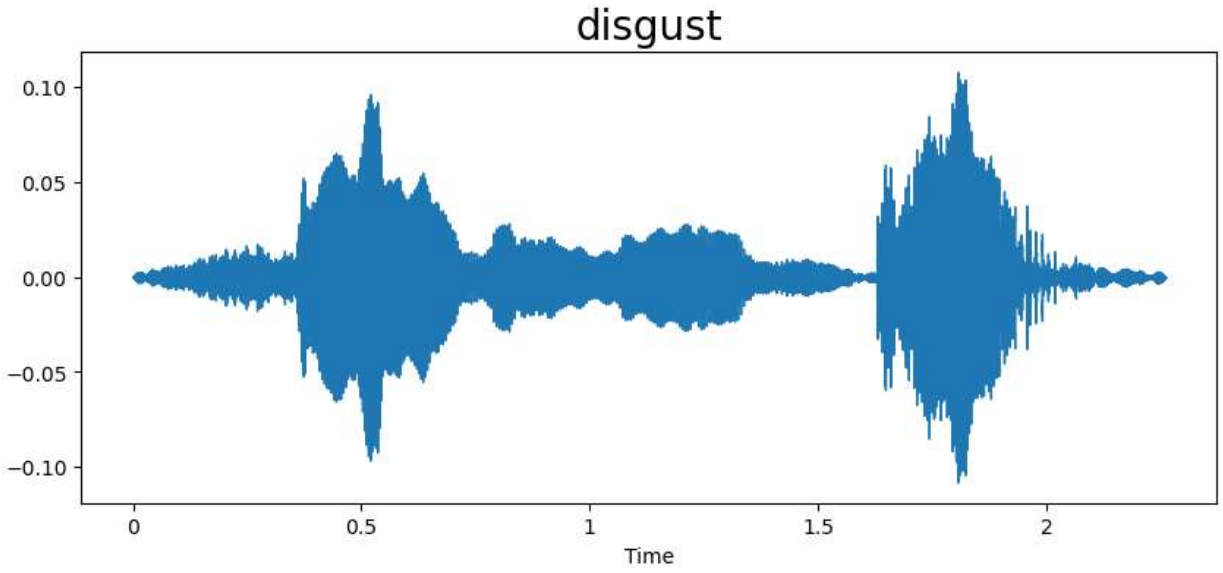
Out[12]:



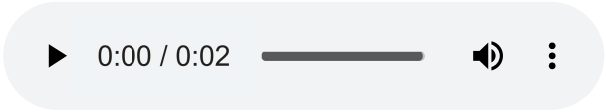


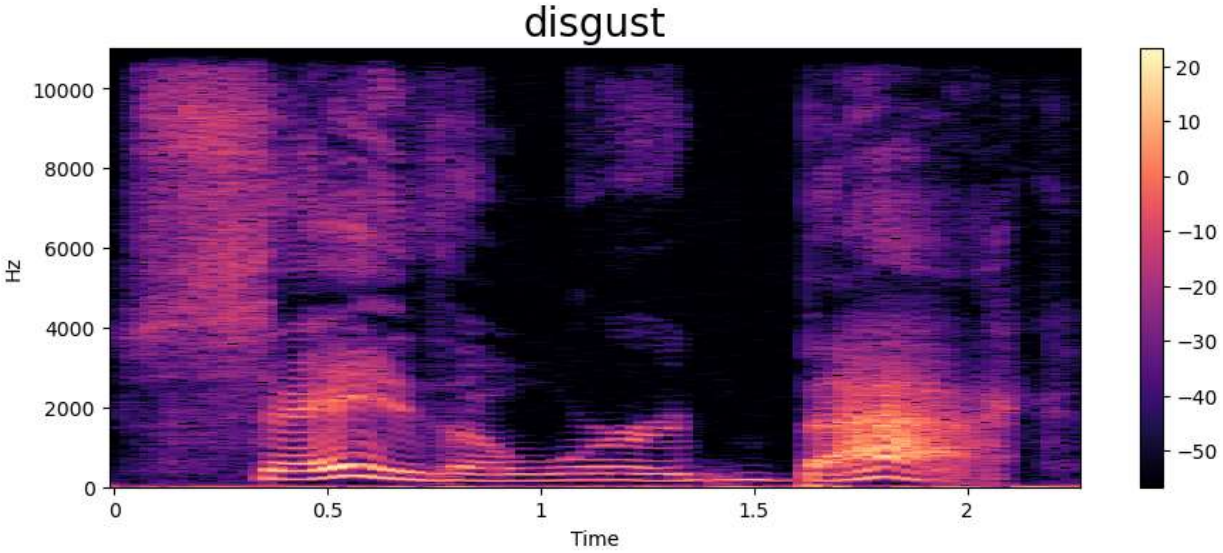
In [14]:

```
emotion = 'disgust'
path = np.array(df['speech'][df['label']==emotion])[0]
data, sampling_rate = librosa.load(path)
waveplot(data, sampling_rate, emotion)
spectogram(data, sampling_rate, emotion)
Audio(path)
```



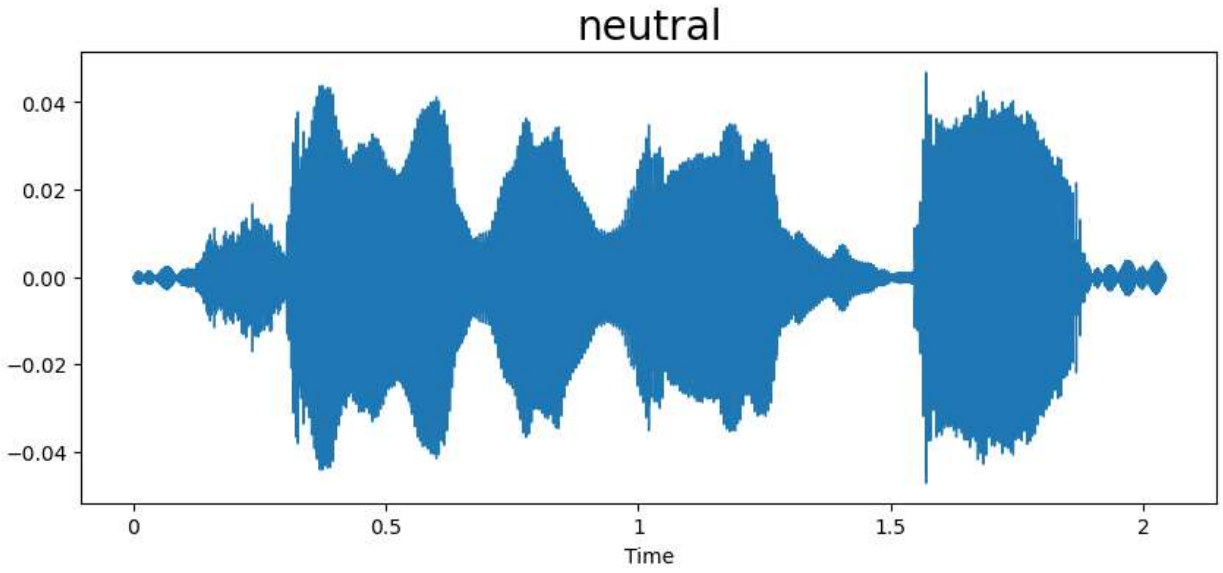
Out[14]:



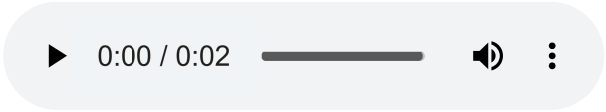


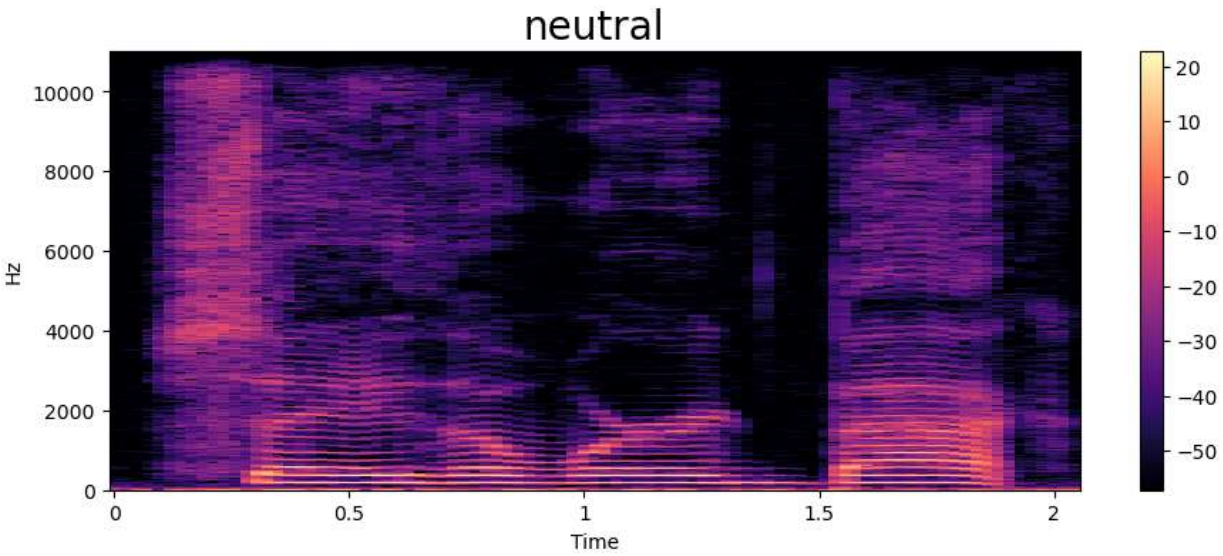
In [15]:

```
emotion = 'neutral'
path = np.array(df['speech'][df['label']==emotion])[0]
data, sampling_rate = librosa.load(path)
waveplot(data, sampling_rate, emotion)
spectogram(data, sampling_rate, emotion)
Audio(path)
```



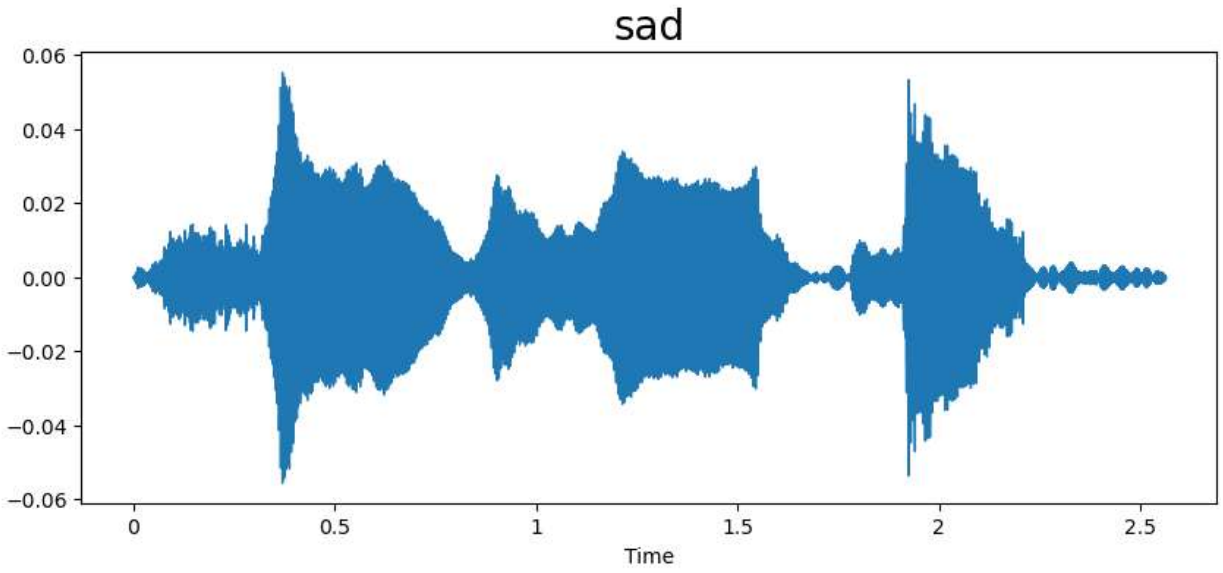
Out[15]:



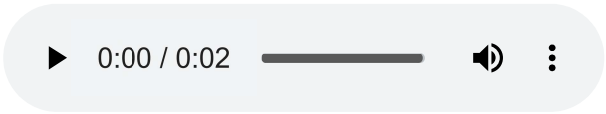


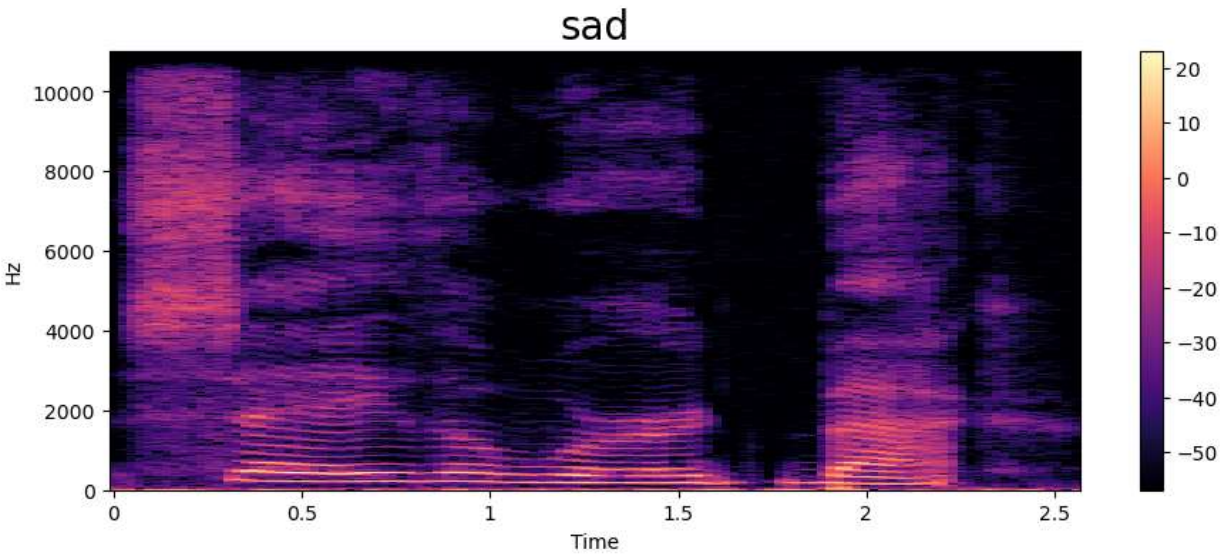
In [16]:

```
emotion = 'sad'
path = np.array(df['speech'][df['label']==emotion])[0]
data, sampling_rate = librosa.load(path)
waveplot(data, sampling_rate, emotion)
spectogram(data, sampling_rate, emotion)
Audio(path)
```



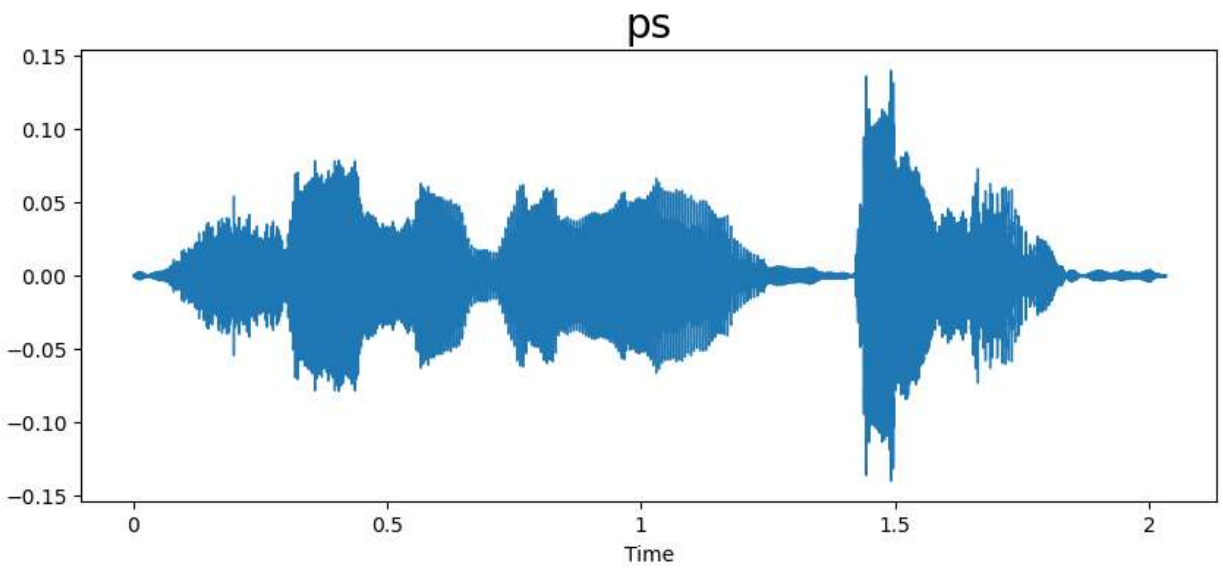
Out[16]:



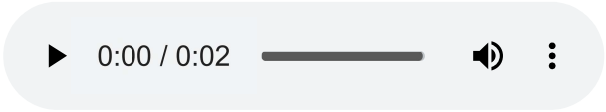


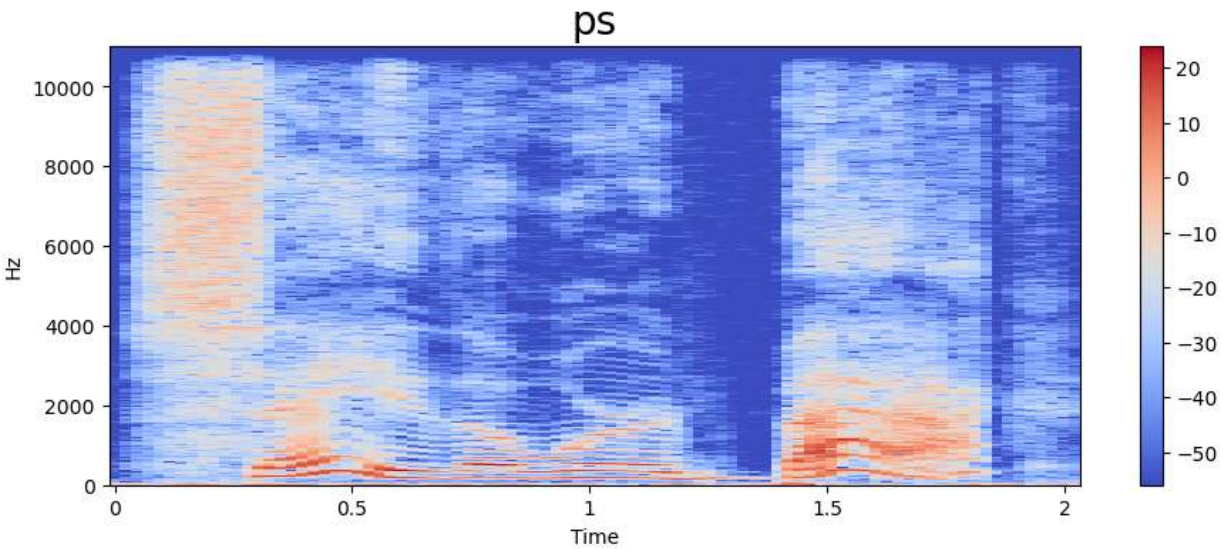
In [17]:

```
emotion = 'ps'
path = np.array(df['speech'][df['label']==emotion])[0]
data, sampling_rate = librosa.load(path)
waveplot(data, sampling_rate, emotion)
spectogram(data, sampling_rate, emotion)
Audio(path)
```



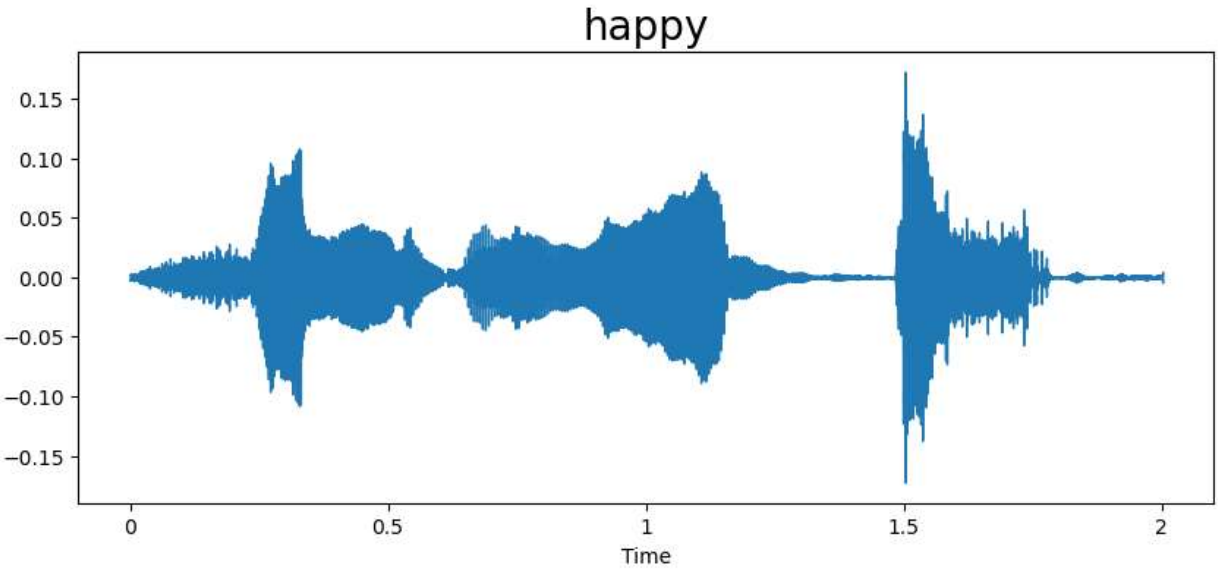
Out[17]:



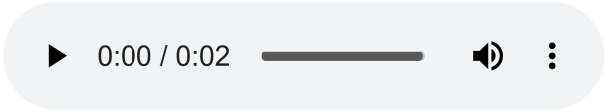


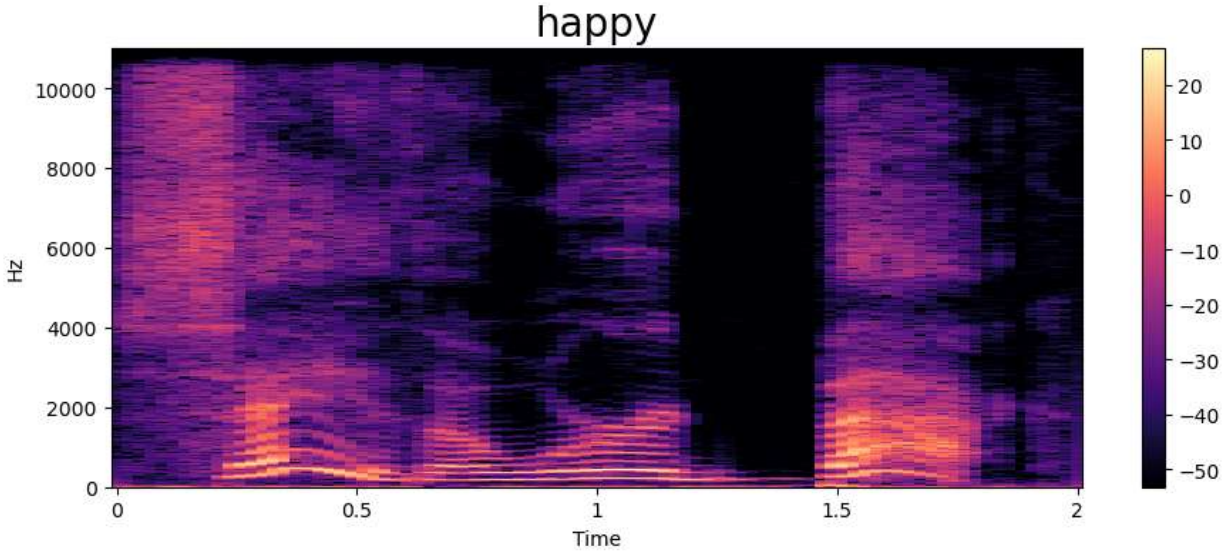
In [18]:

```
emotion = 'happy'
path = np.array(df['speech'][df['label']==emotion])[0]
data, sampling_rate = librosa.load(path)
waveplot(data, sampling_rate, emotion)
spectogram(data, sampling_rate, emotion)
Audio(path)
```



Out[18]:





In [19]:
```
def extract_mfcc(filename):
    y, sr = librosa.load(filename, duration=3, offset=0.5)
    mfcc = np.mean(librosa.feature.mfcc(y=y, sr=sr, n_mfcc=40).T, axis=0)
    return mfcc
```

In [20]:
```
extract_mfcc(df['speech'][0])
```

Out[20]:
```
array([-3.96986206e+02,  7.74405365e+01, -1.95927906e+01, -2.16666889e+01,
       -2.11275506e+00,  1.00753632e+01, -2.03667068e+01, -6.09244919e+00,
       -7.21228361e+00, -5.57365894e-01, -1.83255327e+00,  2.02101514e-01,
        7.27551281e-01,  1.31773770e+00,  2.88633752e+00,  2.85579133e+00,
       -4.71292162e+00, -4.43651056e+00, -1.62115920e+00, -1.02398405e+01,
       -7.55126143e+00, -1.79688025e+00, -7.03765202e+00,  9.43658447e+00,
        8.35585499e+00,  2.17123604e+01,  1.92169895e+01,  2.03489265e+01,
        1.34133663e+01,  8.33917141e+00,  3.94722402e-01,  5.11131430e+00,
        9.56873894e+00,  5.45486832e+00,  2.50996375e+00, -1.82390714e+00,
        4.86896276e+00,  9.31392384e+00,  2.08915019e+00, -1.90649128e+00],
      dtype=float32)
```

In [21]:
```
X_mfcc = df['speech'].apply(lambda x: extract_mfcc(x))
```

In [22]:
```
X_mfcc
```

Out[22]:
```
0       [-396.9862, 77.44054, -19.59279, -21.666689, -...
1       [-465.73267, 98.77373, 0.65600896, -32.74544, ...
2       [-429.79196, 46.124, 1.5550478, -0.21709539, 2...
3       [-403.46118, 76.32369, -12.531774, -22.288858,...
4       [-434.05756, 77.4455, 10.8655, 16.092943, 8.04...
                              ...
2795    [-553.2201, 89.83577, 27.215466, 16.407124, 19...
2796    [-589.23676, 96.20408, 36.96118, 15.014447, 28...
2797    [-533.41815, 85.43242, 27.791998, 19.307178, 2...
2798    [-548.6142, 110.16424, 31.910236, 12.572517, 2...
2799    [-549.2962, 102.374565, 32.268833, 26.261616, ...
Name: speech, Length: 2800, dtype: object
```

In [23]:
```
X = [x for x in X_mfcc]
X = np.array(X)
X.shape
```

Out[23]:
```
(2800, 40)
```

In [24]:
```
## input split
X = np.expand_dims(X, -1)
X.shape
```

Out[24]:
```
(2800, 40, 1)
```

In [25]:
```
from sklearn.preprocessing import OneHotEncoder
enc = OneHotEncoder()
y = enc.fit_transform(df[['label']])
```

In [26]:
```
y = y.toarray()
```

In [27]:
```
y.shape
```

Out[27]:
```
(2800, 7)
```

In [32]:
```
from keras.models import Sequential
from keras.layers import Dense, LSTM, Dropout

model = Sequential([
    LSTM(256, return_sequences=False, input_shape=(40,1)),
    Dropout(0.2),
    Dense(128, activation='relu'),
    Dropout(0.2),
    Dense(64, activation='relu'),
    Dropout(0.2),
    Dense(7, activation='softmax')
])

model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_1 (LSTM)               (None, 256)               264192

 dropout_3 (Dropout)         (None, 256)               0

 dense_3 (Dense)             (None, 128)               32896

 dropout_4 (Dropout)         (None, 128)               0

 dense_4 (Dense)             (None, 64)                8256

 dropout_5 (Dropout)         (None, 64)                0

 dense_5 (Dense)             (None, 7)                 455

=================================================================
Total params: 305799 (1.17 MB)
Trainable params: 305799 (1.17 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

In [29]:
```
# Train the model
history = model.fit(X, y, validation_split=0.2, epochs=50, batch_size=64)
```

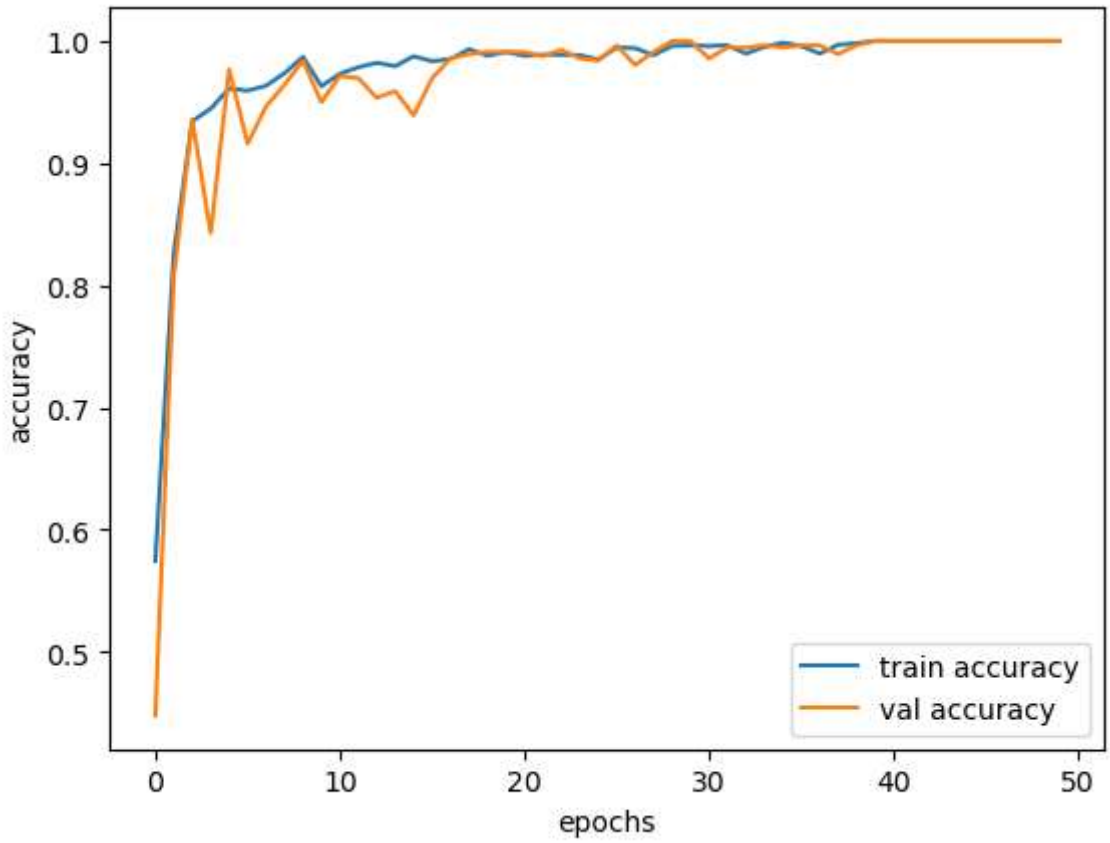
```
Epoch 1/50
35/35 [==============================] - 5s 96ms/step - loss: 1.1602 - accuracy: 0.57
46 - val_loss: 0.9828 - val_accuracy: 0.4482
Epoch 2/50
35/35 [==============================] - 3s 80ms/step - loss: 0.4982 - accuracy: 0.82
63 - val_loss: 0.5170 - val_accuracy: 0.8071
Epoch 3/50
35/35 [==============================] - 3s 84ms/step - loss: 0.2309 - accuracy: 0.93
44 - val_loss: 0.2209 - val_accuracy: 0.9357
Epoch 4/50
35/35 [==============================] - 3s 83ms/step - loss: 0.1736 - accuracy: 0.94
46 - val_loss: 0.5412 - val_accuracy: 0.8429
Epoch 5/50
35/35 [==============================] - 3s 79ms/step - loss: 0.1468 - accuracy: 0.96
12 - val_loss: 0.0871 - val_accuracy: 0.9768
Epoch 6/50
35/35 [==============================] - 3s 90ms/step - loss: 0.1255 - accuracy: 0.95
94 - val_loss: 0.2188 - val_accuracy: 0.9161
Epoch 7/50
35/35 [==============================] - 3s 88ms/step - loss: 0.1129 - accuracy: 0.96
34 - val_loss: 0.1673 - val_accuracy: 0.9464
Epoch 8/50
35/35 [==============================] - 3s 90ms/step - loss: 0.0949 - accuracy: 0.97
37 - val_loss: 0.0924 - val_accuracy: 0.9643
Epoch 9/50
35/35 [==============================] - 3s 87ms/step - loss: 0.0485 - accuracy: 0.98
71 - val_loss: 0.0557 - val_accuracy: 0.9839
Epoch 10/50
35/35 [==============================] - 3s 79ms/step - loss: 0.1220 - accuracy: 0.96
29 - val_loss: 0.1418 - val_accuracy: 0.9500
Epoch 11/50
35/35 [==============================] - 3s 86ms/step - loss: 0.0847 - accuracy: 0.97
32 - val_loss: 0.0613 - val_accuracy: 0.9714
Epoch 12/50
35/35 [==============================] - 3s 84ms/step - loss: 0.0695 - accuracy: 0.97
86 - val_loss: 0.0702 - val_accuracy: 0.9696
Epoch 13/50
35/35 [==============================] - 3s 83ms/step - loss: 0.0663 - accuracy: 0.98
21 - val_loss: 0.1301 - val_accuracy: 0.9536
Epoch 14/50
35/35 [==============================] - 3s 78ms/step - loss: 0.0649 - accuracy: 0.97
95 - val_loss: 0.1238 - val_accuracy: 0.9589
Epoch 15/50
35/35 [==============================] - 3s 86ms/step - loss: 0.0406 - accuracy: 0.98
75 - val_loss: 0.1753 - val_accuracy: 0.9393
Epoch 16/50
35/35 [==============================] - 3s 91ms/step - loss: 0.0527 - accuracy: 0.98
35 - val_loss: 0.0900 - val_accuracy: 0.9696
Epoch 17/50
35/35 [==============================] - 4s 104ms/step - loss: 0.0459 - accuracy: 0.9
853 - val_loss: 0.0301 - val_accuracy: 0.9857
Epoch 18/50
35/35 [==============================] - 3s 78ms/step - loss: 0.0237 - accuracy: 0.99
33 - val_loss: 0.0270 - val_accuracy: 0.9893
Epoch 19/50
35/35 [==============================] - 3s 77ms/step - loss: 0.0416 - accuracy: 0.98
79 - val_loss: 0.0207 - val_accuracy: 0.9911
Epoch 20/50
35/35 [==============================] - 3s 85ms/step - loss: 0.0290 - accuracy: 0.99
11 - val_loss: 0.0427 - val_accuracy: 0.9911
```

```
Epoch 21/50
35/35 [==============================] - 4s 101ms/step - loss: 0.0432 - accuracy: 0.9
879 - val_loss: 0.0177 - val_accuracy: 0.9911
Epoch 22/50
35/35 [==============================] - 3s 99ms/step - loss: 0.0298 - accuracy: 0.98
88 - val_loss: 0.0378 - val_accuracy: 0.9875
Epoch 23/50
35/35 [==============================] - 3s 76ms/step - loss: 0.0459 - accuracy: 0.98
88 - val_loss: 0.0227 - val_accuracy: 0.9929
Epoch 24/50
35/35 [==============================] - 3s 86ms/step - loss: 0.0361 - accuracy: 0.98
84 - val_loss: 0.0334 - val_accuracy: 0.9857
Epoch 25/50
35/35 [==============================] - 3s 87ms/step - loss: 0.0487 - accuracy: 0.98
48 - val_loss: 0.0562 - val_accuracy: 0.9839
Epoch 26/50
35/35 [==============================] - 3s 90ms/step - loss: 0.0189 - accuracy: 0.99
46 - val_loss: 0.0118 - val_accuracy: 0.9964
Epoch 27/50
35/35 [==============================] - 4s 108ms/step - loss: 0.0147 - accuracy: 0.9
942 - val_loss: 0.0372 - val_accuracy: 0.9804
Epoch 28/50
35/35 [==============================] - 3s 81ms/step - loss: 0.0274 - accuracy: 0.98
84 - val_loss: 0.0440 - val_accuracy: 0.9911
Epoch 29/50
35/35 [==============================] - 3s 91ms/step - loss: 0.0143 - accuracy: 0.99
60 - val_loss: 0.0023 - val_accuracy: 1.0000
Epoch 30/50
35/35 [==============================] - 3s 93ms/step - loss: 0.0191 - accuracy: 0.99
64 - val_loss: 0.0047 - val_accuracy: 1.0000
Epoch 31/50
35/35 [==============================] - 3s 88ms/step - loss: 0.0207 - accuracy: 0.99
60 - val_loss: 0.0406 - val_accuracy: 0.9857
Epoch 32/50
35/35 [==============================] - 3s 73ms/step - loss: 0.0146 - accuracy: 0.99
64 - val_loss: 0.0354 - val_accuracy: 0.9946
Epoch 33/50
35/35 [==============================] - 3s 81ms/step - loss: 0.0313 - accuracy: 0.98
97 - val_loss: 0.0166 - val_accuracy: 0.9946
Epoch 34/50
35/35 [==============================] - 3s 91ms/step - loss: 0.0128 - accuracy: 0.99
51 - val_loss: 0.0083 - val_accuracy: 0.9964
Epoch 35/50
35/35 [==============================] - 3s 80ms/step - loss: 0.0059 - accuracy: 0.99
87 - val_loss: 0.0102 - val_accuracy: 0.9946
Epoch 36/50
35/35 [==============================] - 3s 78ms/step - loss: 0.0092 - accuracy: 0.99
60 - val_loss: 0.0110 - val_accuracy: 0.9964
Epoch 37/50
35/35 [==============================] - 3s 80ms/step - loss: 0.0353 - accuracy: 0.98
97 - val_loss: 0.0153 - val_accuracy: 0.9964
Epoch 38/50
35/35 [==============================] - 3s 89ms/step - loss: 0.0113 - accuracy: 0.99
69 - val_loss: 0.0445 - val_accuracy: 0.9893
Epoch 39/50
35/35 [==============================] - 3s 81ms/step - loss: 0.0063 - accuracy: 0.99
82 - val_loss: 0.0064 - val_accuracy: 0.9964
Epoch 40/50
35/35 [==============================] - 3s 78ms/step - loss: 0.0015 - accuracy: 1.00
00 - val_loss: 2.3601e-04 - val_accuracy: 1.0000
```

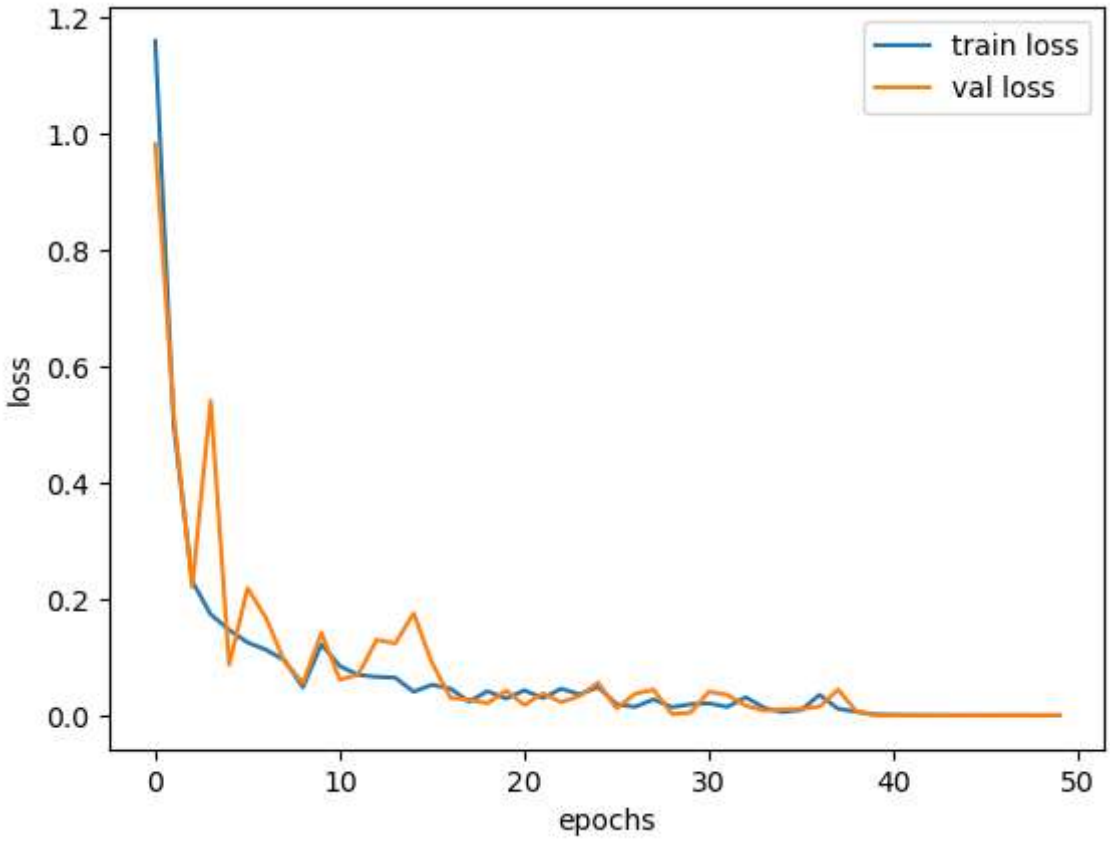
```
Epoch 41/50
35/35 [==============================] - 3s 80ms/step - loss: 0.0011 - accuracy: 1.00
00 - val_loss: 1.4100e-04 - val_accuracy: 1.0000
Epoch 42/50
35/35 [==============================] - 3s 83ms/step - loss: 5.4420e-04 - accuracy:
1.0000 - val_loss: 9.3034e-05 - val_accuracy: 1.0000
Epoch 43/50
35/35 [==============================] - 3s 80ms/step - loss: 4.4372e-04 - accuracy:
1.0000 - val_loss: 7.9626e-05 - val_accuracy: 1.0000
Epoch 44/50
35/35 [==============================] - 3s 78ms/step - loss: 3.2056e-04 - accuracy:
1.0000 - val_loss: 5.6659e-05 - val_accuracy: 1.0000
Epoch 45/50
35/35 [==============================] - 3s 82ms/step - loss: 3.2224e-04 - accuracy:
1.0000 - val_loss: 3.3183e-05 - val_accuracy: 1.0000
Epoch 46/50
35/35 [==============================] - 3s 78ms/step - loss: 2.1939e-04 - accuracy:
1.0000 - val_loss: 2.7399e-05 - val_accuracy: 1.0000
Epoch 47/50
35/35 [==============================] - 3s 79ms/step - loss: 2.7398e-04 - accuracy:
1.0000 - val_loss: 2.9215e-05 - val_accuracy: 1.0000
Epoch 48/50
35/35 [==============================] - 3s 81ms/step - loss: 2.9799e-04 - accuracy:
1.0000 - val_loss: 2.7641e-05 - val_accuracy: 1.0000
Epoch 49/50
35/35 [==============================] - 3s 82ms/step - loss: 1.6644e-04 - accuracy:
1.0000 - val_loss: 2.8072e-05 - val_accuracy: 1.0000
Epoch 50/50
35/35 [==============================] - 3s 79ms/step - loss: 1.9772e-04 - accuracy:
1.0000 - val_loss: 2.2136e-05 - val_accuracy: 1.0000
```

In [34]:

```
epochs = list(range(50))
acc = history.history['accuracy']
val_acc = history.history['val_accuracy']

plt.plot(epochs, acc, label='train accuracy')
plt.plot(epochs, val_acc, label='val accuracy')
plt.xlabel('epochs')
plt.ylabel('accuracy')
plt.legend()
plt.show()
```



In [35]:
```
loss = history.history['loss']
val_loss = history.history['val_loss']

plt.plot(epochs, loss, label='train loss')
plt.plot(epochs, val_loss, label='val loss')
plt.xlabel('epochs')
plt.ylabel('loss')
plt.legend()
plt.show()
```


train loss
val loss
loss
epochs
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0
10
20
30
40
50

In [ ]:

In [ ]: