In [1]:
```
import numpy as np
import pandas as pd
import os
```

In [2]:
```
def load_images_train(path):
    classes = ['PNEUMONIA','NORMAL']
    imgs = []
    labels = []
    for i in classes:
        l2 = os.listdir(path+'/'+i)
        c = 0
        for j in l2:
            img = Image.open(path+i+'/'+j).convert('RGB')
            img = img.resize(size=(32,32))
            imgs.append(np.array(img))
            labels.append(i)
            del img
    return np.array(imgs),labels
```

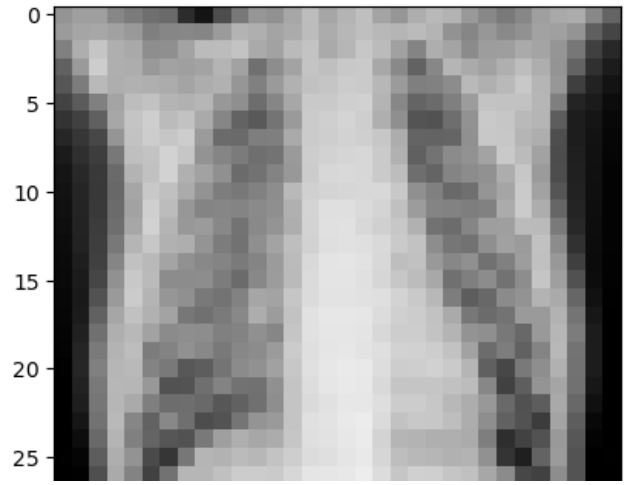
In [3]:
```
import matplotlib.pyplot as plt
from PIL import Image
from tensorflow.keras.layers import Conv2D,MaxPooling2D,Flatten,Dense
from keras.models import Sequential
from sklearn.model_selection import train_test_split
```

In [17]:
```
x_train,y_train = load_images_train("C:/Users/HP/Desktop/python projects/trair
```

In [18]:
```
print(x_train.shape)
```

```
(5216, 32, 32, 3)
```

In [19]:
```
import random
for i in range(1, 10):
  n = random.randint(0,1000)
  plt.imshow(x_train[n])
  plt.show()
```



In [21]:
```
t=target.cat.codes
```

In [20]:
```
target=pd.Series(y_train,dtype='category')
t
target
```

Out[21]:
```
0       1
1       1
2       1
3       1
4       1
       ..
5211    0
5212    0
5213    0
5214    0
5215    0
Length: 5216, dtype: int8
```

Out[20]:
```
0       PNEUMONIA
1       PNEUMONIA
2       PNEUMONIA
3       PNEUMONIA
4       PNEUMONIA
          ...
5211     NORMAL
5212     NORMAL
5213     NORMAL
5214     NORMAL
5215     NORMAL
Length: 5216, dtype: category
Categories (2, object): ['NORMAL', 'PNEUMONIA']
```

In [23]:
```
from PIL import Image
testim1 = Image.open("C:/Users/HP/Desktop/python projects/train/NORMAL/IM-0166
```

In [20]:
```
target=pd.Series(y_train,dtype='category')
t
target
```

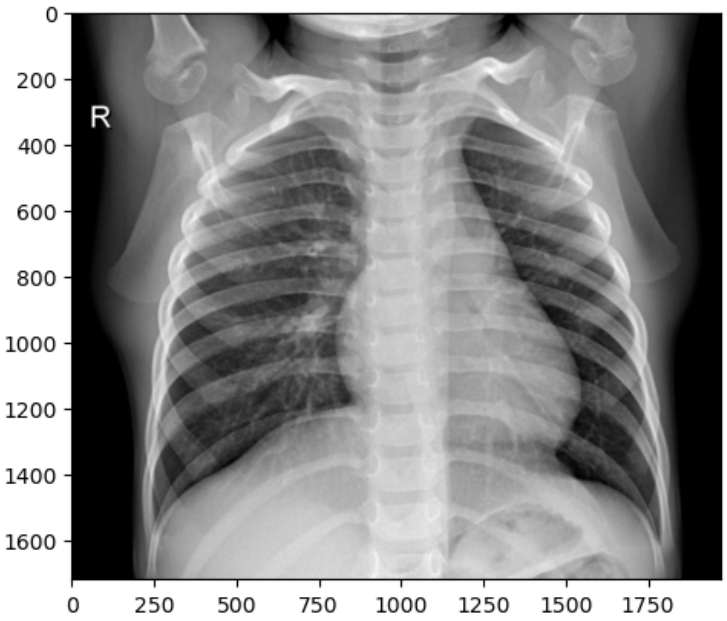
Out[20]:
```
0       PNEUMONIA
1       PNEUMONIA
2       PNEUMONIA
3       PNEUMONIA
4       PNEUMONIA
          ...
5211     NORMAL
5212     NORMAL
5213     NORMAL
5214     NORMAL
5215     NORMAL
Length: 5216, dtype: category
Categories (2, object): ['NORMAL', 'PNEUMONIA']
```

In [21]:
```
t=target.cat.codes
t
```

Out[21]:
```
0       1
1       1
2       1
3       1
4       1
       ..
5211    0
5212    0
5213    0
5214    0
5215    0
Length: 5216, dtype: int8
```

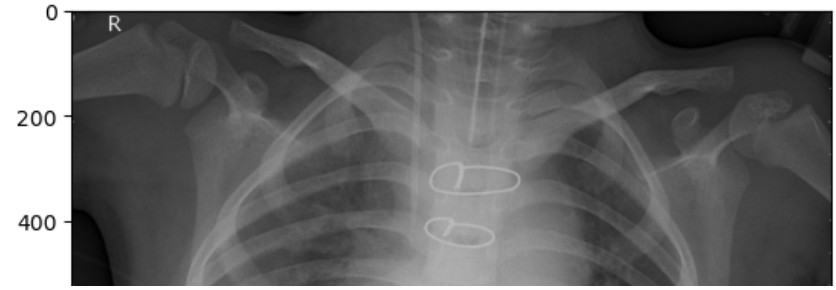
In [23]:
```
from PIL import Image
testim1 = Image.open("C:/Users/HP/Desktop/python projects/train/NORMAL/IM-0166
plt.imshow(testim1)
```

Out[23]:
```
<matplotlib.image.AxesImage at 0x14ce5cb9450>
```



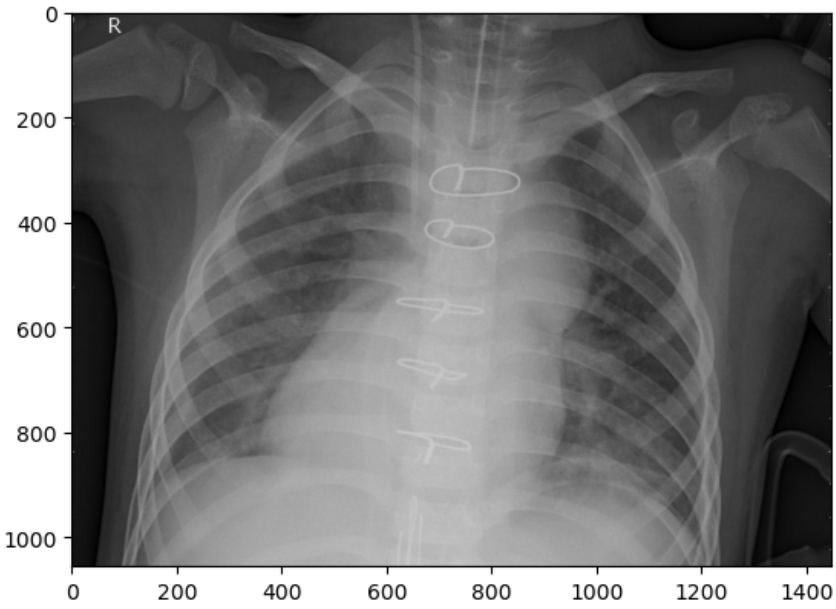
In [24]:
```
from PIL import Image
path = "C:/Users/HP/Desktop/python projects/train/PNEUMONIA/person2_bacteria_3
testim2 = Image.open(path).convert('RGB')
plt.imshow(testim2)
```

Out[24]:
```
<matplotlib.image.AxesImage at 0x14ce5c226d0>
```



In [24]:
```
from PIL import Image
path = "C:/Users/HP/Desktop/python projects/train/PNEUMONIA/person2_bacteria_3
testim2 = Image.open(path).convert('RGB')
plt.imshow(testim2)
```

Out[24]: <matplotlib.image.AxesImage at 0x14ce5c226d0>



In [25]:
```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense,Flatten,Conv2D,MaxPooling2D,Dropout,
```

In [27]:
```
model = Sequential()
model.add(Conv2D(16,activation="relu",kernel_size=3,input_shape=x_train.shape[
model.add(MaxPooling2D((2,2),padding='same'))
model.add(Activation("relu"))
model.add(Conv2D(32,activation="relu",kernel_size=3))
model.add(MaxPooling2D((2,2),padding='same'))
model.add(Conv2D(64,activation="relu",kernel_size=3))
model.add(MaxPooling2D((2,2),padding='same'))
model.add(Conv2D(128,activation="relu",kernel_size=3))
model.add(MaxPooling2D((2,2),padding='same'))
```

In [29]:
```
model.add(Flatten())
model.add(Dense(100,activation="relu"))
model.add(Dropout(0.15))
model.add(Dense(2,activation="softmax"))
model.compile(loss="sparse_categorical_crossentropy",optimizer="adam",metrics=
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_4 (Conv2D)           (None, 30, 30, 16)        448

 max_pooling2d_4 (MaxPoolin  (None, 15, 15, 16)        0
 g2D)

 activation_1 (Activation)   (None, 15, 15, 16)        0
```

In [29]:
```
model.add(Flatten())
model.add(Dense(100,activation="relu"))
model.add(Dropout(0.15))
model.add(Dense(2,activation="softmax"))
model.compile(loss="sparse_categorical_crossentropy",optimizer="adam",metrics=
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_4 (Conv2D)           (None, 30, 30, 16)        448

 max_pooling2d_4 (MaxPoolin  (None, 15, 15, 16)        0
 g2D)

 activation_1 (Activation)   (None, 15, 15, 16)        0

 conv2d_5 (Conv2D)           (None, 13, 13, 32)        4640

 max_pooling2d_5 (MaxPoolin  (None, 7, 7, 32)          0
 g2D)

 conv2d_6 (Conv2D)           (None, 5, 5, 64)          18496

 max_pooling2d_6 (MaxPoolin  (None, 3, 3, 64)          0
 g2D)

 conv2d_7 (Conv2D)           (None, 1, 1, 128)         73856

 max_pooling2d_7 (MaxPoolin  (None, 1, 1, 128)         0
 g2D)

 flatten (Flatten)           (None, 128)               0

 dense (Dense)               (None, 100)               12900

 dropout (Dropout)           (None, 100)               0

 dense_1 (Dense)             (None, 2)                 202

=================================================================
Total params: 110542 (431.80 KB)
Trainable params: 110542 (431.80 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

In [30]:
```
history = model.fit(x_train,t,epochs=20)
```

```
Epoch 1/20
163/163 [==============================] - 5s 20ms/step - loss: 0.9344 - acc
uracy: 0.8679
Epoch 2/20
163/163 [==============================] - 3s 20ms/step - loss: 0.1511 - acc
uracy: 0.9415
Epoch 3/20
163/163 [==============================] - 2s 12ms/step - loss: 0.1309 - acc
uracy: 0.9498
Epoch 4/20
163/163 [==============================] - 2s 12ms/step - loss: 0.1125 - acc
uracy: 0.9555
Epoch 5/20
163/163 [==============================] - 2s 13ms/step - loss: 0.0934 - acc
uracy: 0.9664
Epoch 6/20
```

In [30]:
```
history = model.fit(x_train,t,epochs=20)
```

```
Epoch 1/20
163/163 [==============================] - 5s 20ms/step - loss: 0.9344 - acc
uracy: 0.8679
Epoch 2/20
163/163 [==============================] - 3s 20ms/step - loss: 0.1511 - acc
uracy: 0.9415
Epoch 3/20
163/163 [==============================] - 2s 12ms/step - loss: 0.1309 - acc
uracy: 0.9498
Epoch 4/20
163/163 [==============================] - 2s 12ms/step - loss: 0.1125 - acc
uracy: 0.9555
Epoch 5/20
163/163 [==============================] - 2s 13ms/step - loss: 0.0934 - acc
uracy: 0.9664
Epoch 6/20
163/163 [==============================] - 3s 18ms/step - loss: 0.0876 - acc
uracy: 0.9666
Epoch 7/20
163/163 [==============================] - 3s 18ms/step - loss: 0.0844 - acc
uracy: 0.9693
Epoch 8/20
163/163 [==============================] - 3s 17ms/step - loss: 0.0753 - acc
uracy: 0.9695
Epoch 9/20
163/163 [==============================] - 3s 16ms/step - loss: 0.0698 - acc
uracy: 0.9735
Epoch 10/20
163/163 [==============================] - 2s 12ms/step - loss: 0.0719 - acc
uracy: 0.9720
Epoch 11/20
163/163 [==============================] - 2s 14ms/step - loss: 0.0542 - acc
uracy: 0.9781
Epoch 12/20
163/163 [==============================] - 3s 16ms/step - loss: 0.0508 - acc
uracy: 0.9818
Epoch 13/20
163/163 [==============================] - 3s 17ms/step - loss: 0.0551 - acc
uracy: 0.9808
Epoch 14/20
163/163 [==============================] - 3s 15ms/step - loss: 0.0453 - acc
uracy: 0.9829
Epoch 15/20
163/163 [==============================] - 3s 17ms/step - loss: 0.0388 - acc
uracy: 0.9856
Epoch 16/20
163/163 [==============================] - 2s 12ms/step - loss: 0.0328 - acc
uracy: 0.9889
Epoch 17/20
163/163 [==============================] - 2s 11ms/step - loss: 0.0450 - acc
uracy: 0.9820
Epoch 18/20
163/163 [==============================] - 2s 14ms/step - loss: 0.0289 - acc
uracy: 0.9904
Epoch 19/20
163/163 [==============================] - 2s 15ms/step - loss: 0.0242 - acc
uracy: 0.9908
Epoch 20/20
163/163 [==============================] - 2s 15ms/step - loss: 0.0459 - acc
uracy: 0.9827
```

In [34]:
```
def load_images_test(path):
    classes = ['PNEUMONIA','NORMAL']
    imgs = []
    labels = []
    for i in classes:
        l2 = os.listdir(path+'/'+i)
        c = 0
        for j in l2:
            img = Image.open(path+i+'/'+j).convert('RGB')
            img = img.resize(size=(32,32))
            imgs.append(np.array(img))
            labels.append(i)
            del img
    return np.array(imgs),labels
```

In [38]:
```
x_test,y_test = load_images_test("C:/Users/HP/Desktop/python projects/test/")
```

uracy: 0.9827

In [34]:
```
def load_images_test(path):
    classes = ['PNEUMONIA','NORMAL']
    imgs = []
    labels = []
    for i in classes:
        l2 = os.listdir(path+'/'+i)
        c = 0
        for j in l2:
            img = Image.open(path+i+'/'+j).convert('RGB')
            img = img.resize(size=(32,32))
            imgs.append(np.array(img))
            labels.append(i)
            del img
    return np.array(imgs),labels
```

In [38]:
```
x_test,y_test = load_images_test("C:/Users/HP/Desktop/python projects/test/")
```

In [39]:
```
target1=pd.Series(y_test,dtype='category')
target1
```

Out[39]:
```
0      PNEUMONIA
1      PNEUMONIA
2      PNEUMONIA
3      PNEUMONIA
4      PNEUMONIA
         ...
619       NORMAL
620       NORMAL
621       NORMAL
622       NORMAL
623       NORMAL
Length: 624, dtype: category
Categories (2, object): ['NORMAL', 'PNEUMONIA']
```

In [40]:
```
t1=target1.cat.codes
t1
```

Out[40]:
```
0      1
1      1
2      1
3      1
4      1
      ..
619    0
620    0
621    0
622    0
623    0
Length: 624, dtype: int8
```

In [41]:
```
y_pred = model.predict(x_test)
y_pred
```

```
20/20 [==============================] - 0s 5ms/step
```

Out[41]:
```
array([[1.6637307e-10, 1.0000000e+00],
       [4.4121360e-05, 9.9995589e-01],
       [2.7869487e-09, 1.0000000e+00],
       ...,
       [5.2364998e-02, 9.4763499e-01],
       [4.5901434e-03, 9.9540991e-01],
       [3.7854928e-09, 1.0000000e+00]], dtype=float32)
```

In [45]:
```
target2=pd.Series(y_val,dtype='category')
target2
```

Out[45]:
```
0     PNEUMONIA
1     PNEUMONIA
2     PNEUMONIA
3     PNEUMONIA
4     PNEUMONIA
5     PNEUMONIA
6     PNEUMONIA
7     PNEUMONIA
8        NORMAL
9        NORMAL
10       NORMAL
11       NORMAL
12       NORMAL
13       NORMAL
14       NORMAL
```

In [44]:
```
x_val,y_val = load_images_test("C:/Users/HP/Desktop/python projects/val/")
```

In [45]:
```
target2=pd.Series(y_val,dtype='category')
target2
```

```
       ...,
       [5.2364998e-02, 9.4763499e-01],
       [4.5901434e-03, 9.9540991e-01],
       [3.7854928e-09, 1.0000000e+00]], dtype=float32)
```

Out[45]:
```
0     PNEUMONIA
1     PNEUMONIA
2     PNEUMONIA
3     PNEUMONIA
4     PNEUMONIA
5     PNEUMONIA
6     PNEUMONIA
7     PNEUMONIA
8        NORMAL
9        NORMAL
10       NORMAL
11       NORMAL
12       NORMAL
13       NORMAL
14       NORMAL
15       NORMAL
dtype: category
Categories (2, object): ['NORMAL', 'PNEUMONIA']
```

In [44]:
```
x_val,y_val=load_images_test("C:/Users/HP/Desktop/python projects/val/")
```

In [46]:
```
t3 = target2.cat.codes
t3
```

Out[46]:
```
0     1
1     1
2     1
3     1
4     1
5     1
6     1
7     1
8     0
9     0
10    0
11    0
12    0
13    0
14    0
15    0
dtype: int8
```

In [47]:
```
val_pred = model.predict(x_val)
val_pred
```

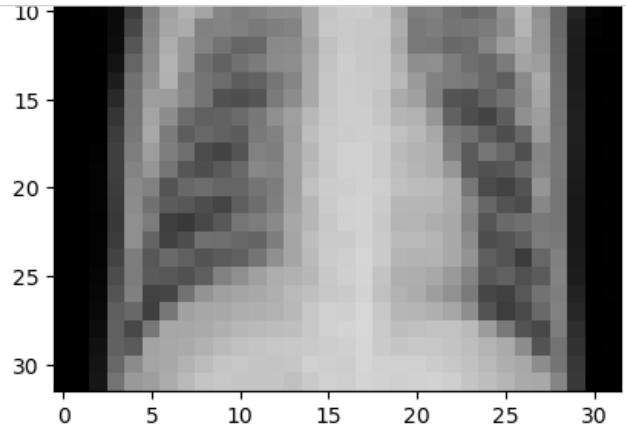
```
1/1 [==============================] - 0s 20ms/step
```

Out[47]:
```
array([[5.10323673e-07, 9.99999523e-01],
       [1.19066165e-07, 9.99999881e-01],
       [2.49461709e-06, 9.99997497e-01],
       [4.58868344e-05, 9.99954104e-01],
       [3.88961189e-06, 9.99996066e-01],
       [1.09179382e-07, 9.99999881e-01],
       [1.43378054e-06, 9.99998569e-01],
       [9.51579750e-06, 9.99990463e-01],
       [5.34985960e-03, 9.94650185e-01],
       [3.14582861e-03, 9.96854246e-01],
       [8.06390703e-01, 1.93609327e-01],
       [6.68178126e-02, 9.33182180e-01],
       [6.67043822e-03, 9.93329585e-01],
       [1.54765353e-01, 8.45234632e-01],
       [4.33896601e-01, 5.66103458e-01],
       [8.9990534e-01, 1.0009489e-01]], dtype=float32)
```

In [54]:
```
import random
for i in range(1,10):
    n = random.randint(0,634)
    img = plt.imshow(x_test[n])
    plt.imshow(x_test[n])
    plt.show()
#   Prediction(img)
    predicted_value = model.predict(x_test)
    if(predicted_value[n][1] == 1.0000000e+00):
        print("Pnuemonia detected")
    elif(predicted_value[n][0] > predicted_value[n][1]):
        print("No pnuemonia Detected")
    else:
        print("Pnuemonia detected")
```

In [54]:
```
import random
for i in range(1,10):
    n = [random.randint(0,620)]
    img = plt.imshow(x_test[n])
    plt.imshow(x_test[n])
    plt.show()
#   Prediction(img)
    predicted_value = model.predict(x_test)
    if(predicted_value[n][1] == 1.0000000e+00):
        print("Pnuemonia detected")
    elif(predicted_value[n][0] > predicted_value[n][1]):
        print("No pnuemonia Detected")
    else:
        print("Pnuemonia detected")
```

```
       [6.67043822e-03, 9.93329585e-01],
       [1.94765353e-01, 8.45234632e-01],
       [4.33896601e-01, 5.66103458e-01],
       [8.99050534e-01, 1.00949489e-01]], dtype=float32)
```

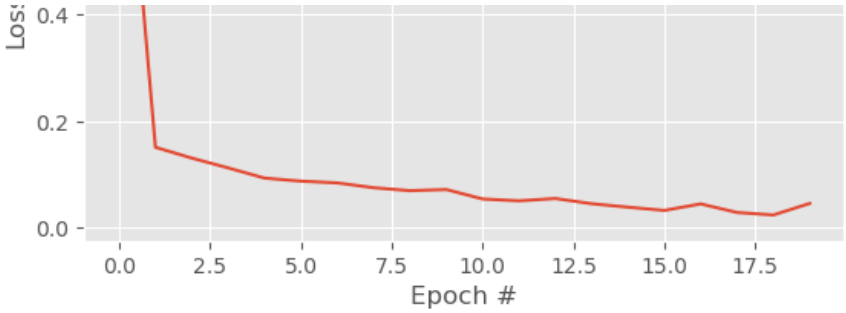



```
20/20 [==============================] - 0s 5ms/step
Pnuemonia detected
```

In [55]:
```
plt.style.use("ggplot")
plt.figure()
plt.plot(np.arange(0, 20), history.history["loss"], label="train_loss")
plt.plot(np.arange(0, 20), history.history["accuracy"], label="train_acc")
plt.title("Training Loss and Accuracy")
plt.xlabel("Epoch #")
plt.ylabel("Loss/Accuracy")
plt.legend()
```

Out[55]: <matplotlib.legend.Legend at 0x14ceef94ed0>

In [ ]:



In [ ]: