In [1]:

```
## importing libraries
import os
import numpy as np
import pandas as pd
import random
import matplotlib.pyplot as plt
```

In [2]:

```
## importing tensorflow libraries
import tensorflow as tf
import tensorflow.keras as keras
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Sequential, Model, load_model
from tensorflow.keras.layers import Input, Dense, Dropout, Flatten, Conv2D, MaxPooli
```

In [3]:

```
pip install opencv-python
```

```
Requirement already satisfied: opencv-python in c:\users\kapoor\anaconda3\lib\site-p
ackages (4.5.4.58)
Requirement already satisfied: numpy>=1.17.3 in c:\users\kapoor\anaconda3\lib\site-p
ackages (from opencv-python) (1.20.1)
Note: you may need to restart the kernel to use updated packages.
```

In [4]:

```
## importing cv2 library
import cv2
```

In [5]:

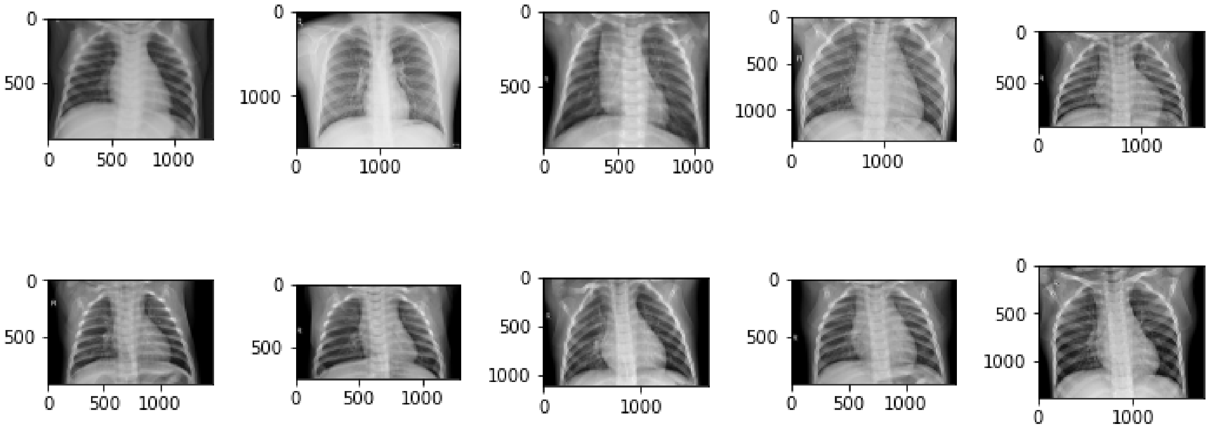
```
# Load Dataset
labels = 4 # HEALTHY, PNEUMONIA, COVID19 ##num_classes
image_size = (224,224) ##target_size
## path = "" ##chest_xray_dir
train_path = "Training_Data/" ##train_dir
test_path   = "Testing_Data/" ##val_dir
```

In [6]:

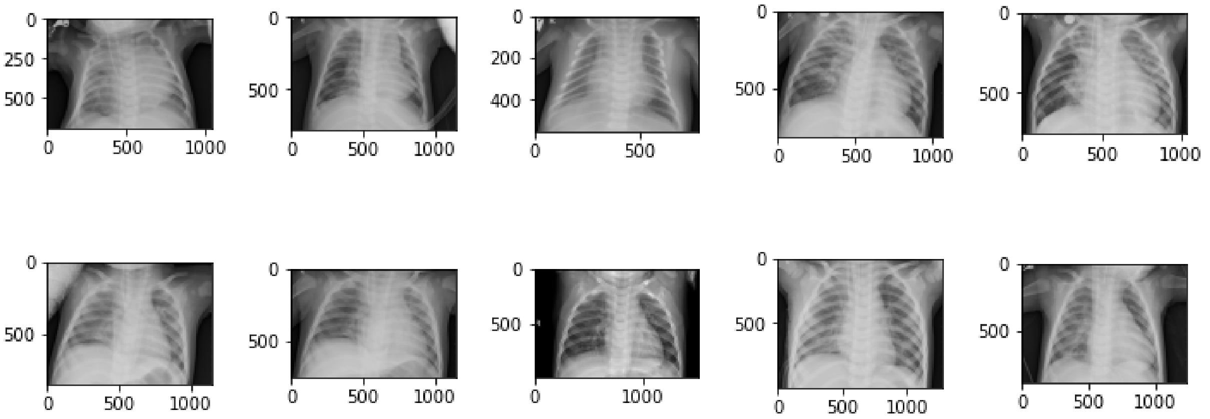
```
def plot_imgs(item_dir, top=10):
    all_item_dirs = os.listdir(item_dir)
    item_files = [os.path.join(item_dir, file) for file in all_item_dirs][:10]

    plt.figure(figsize=(10, 10))

    for idx, img_path in enumerate(item_files):
        plt.subplot(5, 5, idx+1)

        img = plt.imread(img_path)
        plt.tight_layout()
        plt.imshow(img, cmap='gray')
```

In [7]:

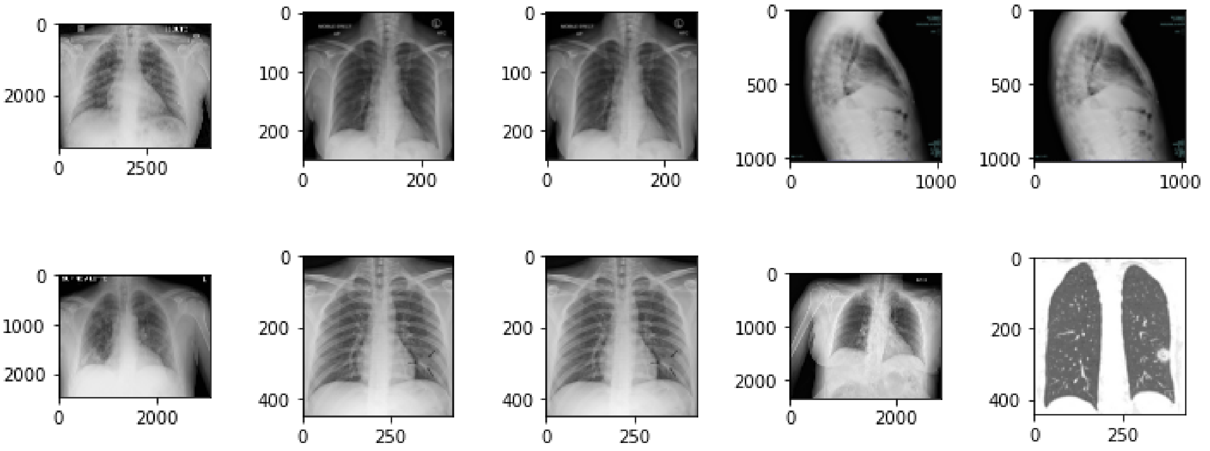
```
plot_imgs(train_path+'Healthy')
```



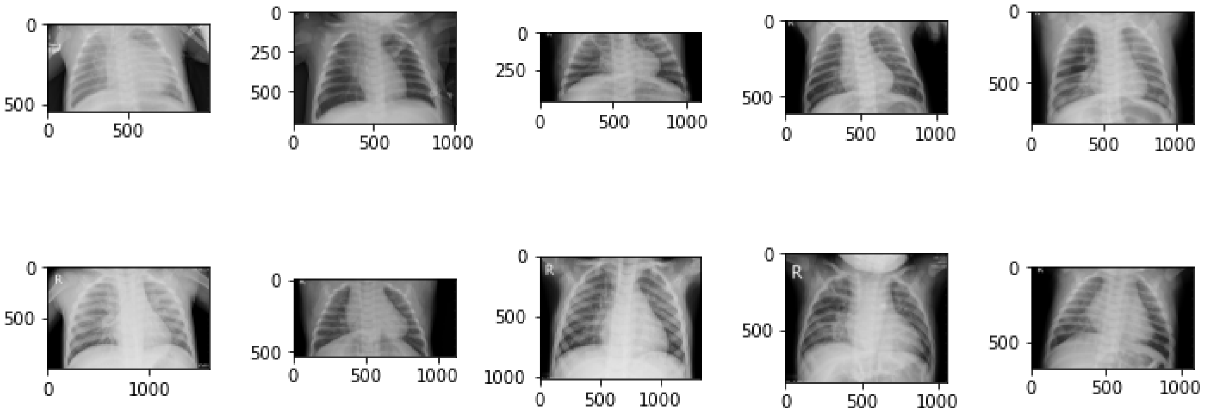
```
In [8]: plot_imgs(train_path+'Bacterial_Pneumonia')
```



```
In [9]: plot_imgs(train_path+'covid19')
```



```
In [10]: plot_imgs(train_path+'Viral_Pneumonia')
```



In [21]:

```
# Data Generator
rescale = 1./255
```

In [22]:

```
train_datagen = ImageDataGenerator(
    rescale=rescale,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True)
```

In [23]:

```
train_generator = train_datagen.flow_from_directory(
    train_path,
    target_size=image_size,
    class_mode='categorical',
    batch_size=32,
    color_mode="grayscale",
    shuffle=True)
```

```
Found 542 images belonging to 3 classes.
```

In [26]:

```
test_datagen = ImageDataGenerator(rescale=rescale)
```

In [27]:

```
test_generator = test_datagen.flow_from_directory(
    test_path,
    target_size=image_size,
    class_mode='categorical',
    batch_size=8,
    color_mode="grayscale",
    shuffle = False)
```

```
Found 40 images belonging to 3 classes.
```

In [28]:

```
# Build Model
model = Sequential()

# 1st Conv Layer
model.add(Conv2D(16, kernel_size=(3, 3), activation='relu', input_shape=(224, 224, 1
model.add(Conv2D(16, kernel_size=(3, 3), activation='relu', padding='same'))
model.add(MaxPooling2D(pool_size=(2, 2), padding='same'))
# 2nd Conv Layer
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', padding='same'))
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', padding='same'))
model.add(MaxPooling2D(pool_size=(2, 2), padding='same'))
# 3rd Conv Layer
model.add(Conv2D(64, kernel_size=(3, 3), activation='relu', padding='same'))
model.add(Conv2D(64, kernel_size=(3, 3), activation='relu', padding='same'))
model.add(MaxPooling2D(pool_size=(2, 2), padding='same'))
# 4th Conv layer
model.add(Conv2D(96, kernel_size=(3, 3), activation='relu', padding='same'))
model.add(Conv2D(96, kernel_size=(3, 3), activation='relu', padding='same'))
model.add(MaxPooling2D(pool_size=(2, 2), padding='same'))
# 5th Conv layer
model.add(Conv2D(128, kernel_size=(3, 3), activation='relu', padding='same'))
model.add(Conv2D(128, kernel_size=(3, 3), activation='relu', padding='same'))
model.add(MaxPooling2D(pool_size=(2, 2), padding='same'))
```

In [31]:

```
# Fully-Connected layer
model.add(Flatten())
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.4))
model.add(Dense(labels, activation='softmax'))
```

In [32]:

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 224, 224, 16)      160
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 224, 224, 16)      2320
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 112, 112, 16)      0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 112, 112, 32)      4640
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 112, 112, 32)      9248
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 56, 56, 32)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 56, 56, 64)        18496
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 56, 56, 64)        36928
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 28, 28, 64)        0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 28, 28, 96)        55392
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 28, 28, 96)        83040
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 14, 14, 96)        0
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 14, 14, 128)       110720
_________________________________________________________________
conv2d_9 (Conv2D)            (None, 14, 14, 128)       147584
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 7, 7, 128)         0
_________________________________________________________________
flatten (Flatten)            (None, 6272)              0
_________________________________________________________________
dense (Dense)                (None, 64)                401472
_________________________________________________________________
dropout (Dropout)            (None, 64)                0
_________________________________________________________________
flatten_1 (Flatten)          (None, 64)                0
_________________________________________________________________
dense_1 (Dense)              (None, 64)                4160
_________________________________________________________________
dropout_1 (Dropout)          (None, 64)                0
_________________________________________________________________
```

```
flatten_2 (Flatten)          (None, 64)                0
_________________________________________________________________
dense_2 (Dense)              (None, 64)                4160
_________________________________________________________________
dropout_2 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_3 (Dense)              (None, 3)                 195
=================================================================
Total params: 878,515
Trainable params: 878,515
Non-trainable params: 0
_________________________________________________________________
```

In [33]:

```
# Compile Model
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'
```

In [36]:

```
# Train Model
num_epochs = 11
history = model.fit_generator(generator=train_generator,
                    steps_per_epoch=train_generator.n // train_generator.batch_size,
                    epochs=num_epochs,
                    validation_data=test_generator,
                    validation_steps=test_generator.n // test_generator.batch_size)
```

```
WARNING:tensorflow:From <ipython-input-36-551dc68e5a73>:3: Model.fit_generator (from
tensorflow.python.keras.engine.training) is deprecated and will be removed in a futu
re version.
Instructions for updating:
Please use Model.fit, which supports generators.
Epoch 1/11
16/16 [==============================] - 40s 3s/step - loss: 1.0831 - accuracy: 0.42
75 - val_loss: 1.0701 - val_accuracy: 0.5000
Epoch 2/11
16/16 [==============================] - 35s 2s/step - loss: 1.0678 - accuracy: 0.49
61 - val_loss: 1.0513 - val_accuracy: 0.5000
Epoch 3/11
16/16 [==============================] - 33s 2s/step - loss: 1.0564 - accuracy: 0.49
22 - val_loss: 1.0431 - val_accuracy: 0.5000
Epoch 4/11
16/16 [==============================] - 31s 2s/step - loss: 1.0582 - accuracy: 0.48
82 - val_loss: 1.0405 - val_accuracy: 0.5000
Epoch 5/11
16/16 [==============================] - 30s 2s/step - loss: 1.0315 - accuracy: 0.49
22 - val_loss: 1.0484 - val_accuracy: 0.5000
Epoch 6/11
16/16 [==============================] - 31s 2s/step - loss: 0.9936 - accuracy: 0.49
02 - val_loss: 1.0601 - val_accuracy: 0.5000
Epoch 7/11
16/16 [==============================] - 30s 2s/step - loss: 1.0033 - accuracy: 0.49
41 - val_loss: 1.0498 - val_accuracy: 0.5000
Epoch 8/11
16/16 [==============================] - 30s 2s/step - loss: 0.9667 - accuracy: 0.48
82 - val_loss: 1.0047 - val_accuracy: 0.5000
Epoch 9/11
16/16 [==============================] - 30s 2s/step - loss: 0.9424 - accuracy: 0.49
80 - val_loss: 1.0361 - val_accuracy: 0.5000
Epoch 10/11
16/16 [==============================] - 31s 2s/step - loss: 0.9348 - accuracy: 0.48
24 - val_loss: 0.9239 - val_accuracy: 0.5000
Epoch 11/11
16/16 [==============================] - 29s 2s/step - loss: 0.9351 - accuracy: 0.47
45 - val_loss: 0.8838 - val_accuracy: 0.5000
```

In [37]:

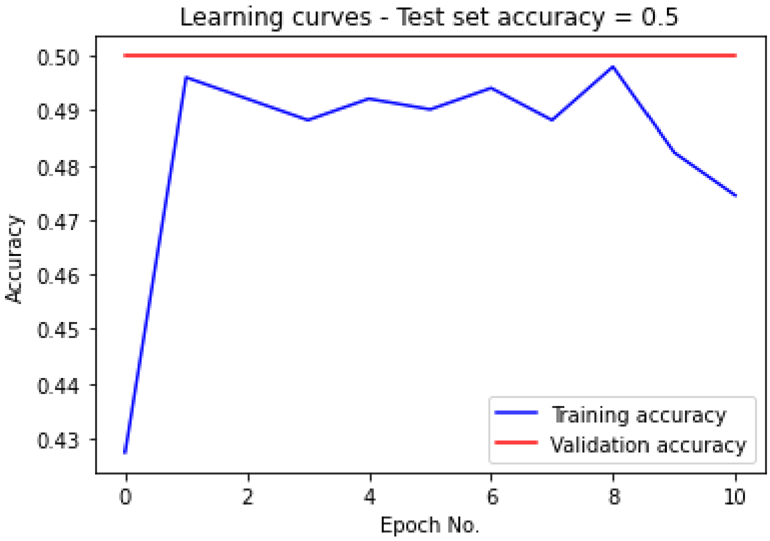
```
# Evaluate Model
result = model.evaluate_generator(test_generator, steps=len(test_generator))
```

```
print("%s%.2f  " % ("Loss     : ", result[0]))
print("%s%.2f%s" % ("Accuracy : ", result[1]*100, "%"))
```

```
WARNING:tensorflow:From <ipython-input-37-c553cdd6946a>:2: Model.evaluate_generator
(from tensorflow.python.keras.engine.training) is deprecated and will be removed in
a future version.
Instructions for updating:
Please use Model.evaluate, which supports generators.
Loss     : 0.88
Accuracy : 50.00%
```

In [38]:

```
# Plot learning curves and compute accuracy on the test set
accuracy = history.history['accuracy']
val_accuracy = history.history['val_accuracy']
test_accuracy = result[1]
plt.plot(range(len(accuracy)), accuracy, color='blue', label='Training accuracy')
plt.plot(range(len(accuracy)), val_accuracy, color='red', label='Validation accuracy
plt.title('Learning curves - Test set accuracy = ' + str(round(test_accuracy, 3)))
plt.xlabel('Epoch No.')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



In [39]:

```
# Predict
y_pred = model.predict_generator(test_generator, steps=len(test_generator), verbose=
y_pred = y_pred.argmax(axis=-1)
y_true=test_generator.classes

numofbatch = len(test_generator)
batch_no = random.randint(0, numofbatch-1)

y_img_batch, y_true_batch = test_generator[batch_no]
y_true_batch = y_true_batch.argmax(axis=-1)

y_pred_batch = model.predict(y_img_batch)
y_pred_batch = y_pred_batch.argmax(axis=-1)
sizeofbatch = len(y_true_batch)
```

```
WARNING:tensorflow:From <ipython-input-39-171b81805e4a>:2: Model.predict_generator
(from tensorflow.python.keras.engine.training) is deprecated and will be removed in
a future version.
Instructions for updating:
Please use Model.predict, which supports generators.
5/5 [==============================] - 1s 145ms/step
```

In [40]:

```
print("-"*35)
print("%s%d"%     ("Selected Batch No       : ", batch_no))
print("-"*35)
print("%s%d"%     ("Batch Size              : ", len(y_pred_batch)))
print("-"*35)
print("%s%.2f%s"% ("Accuracy                : ", np.mean(y_true==y_pred)*100, "%"))
print("-"*35)
```

```
-----------------------------------
Selected Batch No       : 2
-----------------------------------
Batch Size              : 8
-----------------------------------
Accuracy                : 50.00%
-----------------------------------
```

In [ ]: