In [1]:

```
import tensorflow as tf
from tensorflow.keras import models, layers
import matplotlib.pyplot as plt
```

In [2]:

```
IMAGE_SIZE = 224
BATCH_SIZE = 32
CHANNELS = 3
```

In [3]:

```
dataset =  tf.keras.preprocessing.image_dataset_from_directory(
    "PlantVillage",
    shuffle=True,
    batch_size = BATCH_SIZE,
    image_size = (IMAGE_SIZE, IMAGE_SIZE)

)
```

```
Found 16011 files belonging to 10 classes.
```

In [4]:

```
classes = dataset.class_names
```

In [5]:

```
len(dataset)
```

Out[5]:

```
501
```

In [6]:

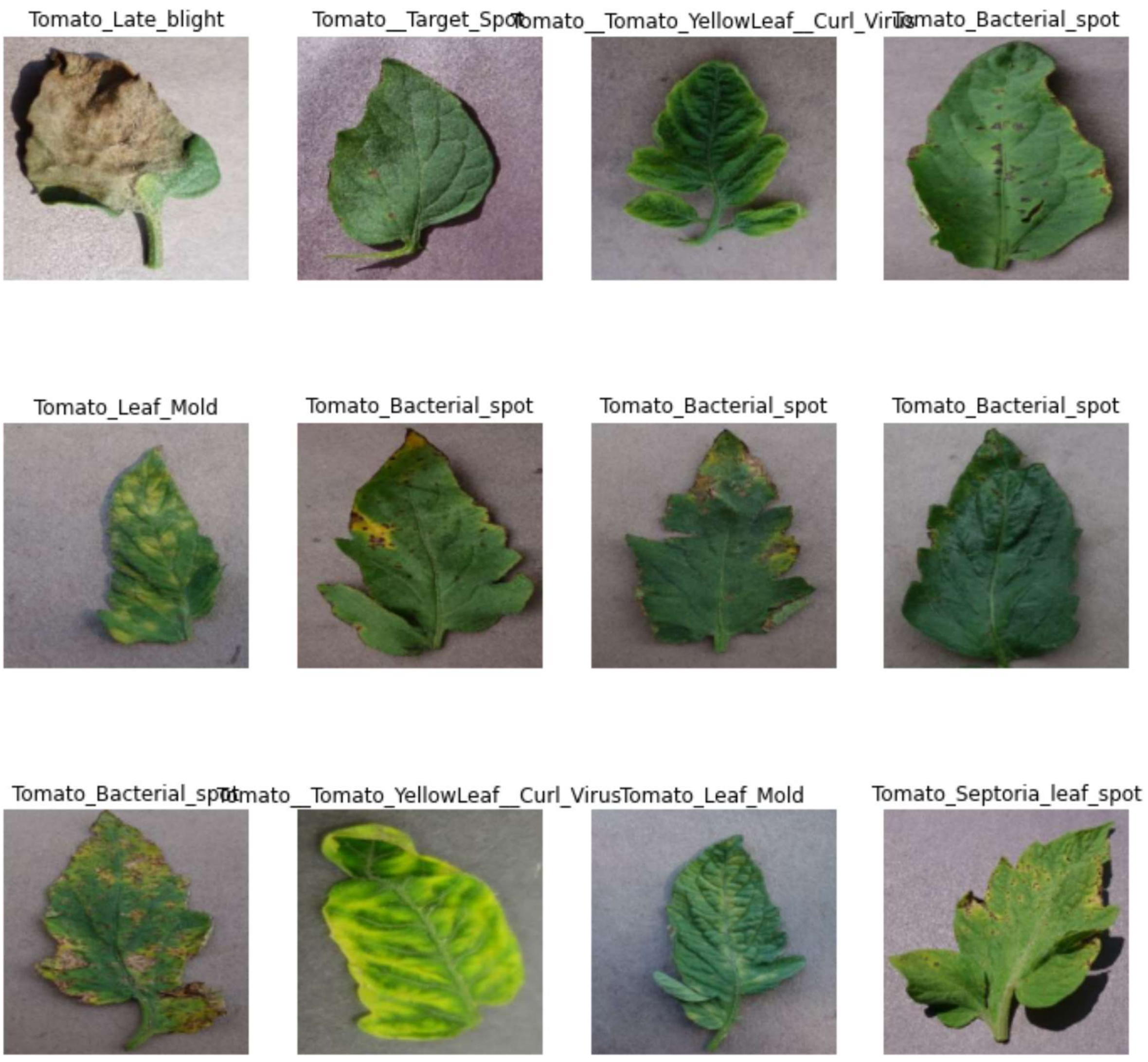
```
dataset
```

Out[6]:

```
<BatchDataset element_spec=(TensorSpec(shape=(None, 224, 224, 3), dtype=t
f.float32, name=None), TensorSpec(shape=(None,), dtype=tf.int32, name=Non
e))>
```

In [7]:

```
plt.figure(figsize=(12,12))
for image_batch, label_batch in dataset.take(1):
    for i in range(12) :
        plt.subplot(3,4,i+1)
        plt.imshow(image_batch[i].numpy().astype("uint8"))
        plt.title(classes[label_batch[i]])
        plt.axis("off")
```



In [8]:

```
def train_val_test(dataset, train_split = 0.8, val_split = 0.1, test_split = 0.1, shuffl

    if shuffle:
        dataset.shuffle(shuffle_size, seed = 12)

    dataset_len = len(dataset)
    train_size = int(dataset_len*train_split)
    val_size = int(dataset_len*val_split)
    test_size = int(dataset_len*test_split)

    train_ds =  dataset.take(train_size)
    val_ds = dataset.skip(train_size).take(val_size)
    test_ds = dataset.skip(train_size).skip(val_size)

    return train_ds, val_ds, test_ds
```

In [9]:

```
train_ds, val_ds, test_ds = train_val_test(dataset)
```

In [10]:

```
len(train_ds)
```

Out[10]:

```
400
```

In [11]:

```
train_ds = train_ds.cache().shuffle(1000).prefetch(tf.data.AUTOTUNE)
val_ds = val_ds.cache().shuffle(1000).prefetch(tf.data.AUTOTUNE)
test_ds = test_ds.cache().shuffle(1000).prefetch(tf.data.AUTOTUNE)
```

In [12]:

```
resize_rescale = tf.keras.Sequential([
    layers.experimental.preprocessing.Resizing(IMAGE_SIZE,IMAGE_SIZE),
    layers.experimental.preprocessing.Rescaling(1.0/255)
])
```

In [13]:

```
data_aug = tf.keras.Sequential(
    [
        tf.keras.layers.experimental.preprocessing.RandomFlip("horizontal_and_vertical")
        tf.keras.layers.experimental.preprocessing.RandomRotation(0.1),
        tf.keras.layers.experimental.preprocessing.RandomZoom(0.1),
        tf.keras.layers.experimental.preprocessing.RandomContrast(0.2),
        tf.keras.layers.experimental.preprocessing.RandomHeight(0.2),


    ]
)
```

In [24]:

```
from keras.applications.vgg19 import VGG19
from keras.models import Model
from keras.layers import Input, Lambda, Dense, Flatten
```

In [21]:

```
vgg = VGG19(input_shape=[IMAGE_SIZE,IMAGE_SIZE] + [3], weights='imagenet', include_top=

for layer in vgg.layers:
    layer.trainable = False
```

In [28]:

```
input_shape = (BATCH_SIZE, IMAGE_SIZE, IMAGE_SIZE, CHANNELS)
n_classes = 3

model = models.Sequential([
    resize_rescale,

    vgg,
    layers.Flatten(),
    layers.Dense(64, activation='relu'),
    layers.Dense(n_classes, activation='softmax'),
])

model.build(input_shape=input_shape)
```

our layers - you can add more if you want

```
#x = Flatten()(vgg.output)
dense_1 = Dense(1000, activation='relu')(x)
dense_2 = Dense(64, activation='relu')(dense_1)
prediction = Dense(n_classes, activation = 'softmax')
```

create a model object

```
model = Model(inputs=vgg.input, outputs=prediction)
```

In [17]:

```
model.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 sequential (Sequential)     (32, 224, 224, 3)         0         
                                                                 
 sequential_1 (Sequential)   (None, None, 224, 3)      0         
                                                                 
 vgg19 (Functional)          (None, 7, 7, 512)         20024384  
                                                                 
 flatten (Flatten)           (None, 25088)             0         
                                                                 
 dense (Dense)               (None, 64)                1605696   
                                                                 
 dense_1 (Dense)             (None, 3)                 195       
                                                                 
=================================================================
Total params: 21,630,275
Trainable params: 1,605,891
Non-trainable params: 20,024,384
_________________________________________________________________
```

In [29]:

```
model.compile(
    optimizer='adam',
    loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=False),
    metrics=['accuracy']
)
```

In [30]:

```
history = model.fit(
    train_ds,
    batch_size=BATCH_SIZE,
    validation_data=val_ds,
    verbose=1,
    epochs=50,
)
```

```
Epoch 1/50
```

In [ ]:

```

```