In [8]:
```
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.preprocessing import image
import numpy as np
import os
import random
from shutil import copyfile
import matplotlib.image  as mpimg
import matplotlib.pyplot as plt
```

In [13]:
```
to_create = [
    'leaf_disease',
    'leaf_disease/training',
    'leaf_disease/testing',
    'leaf_disease/training/Tomato_Bacterial_spot',
    'leaf_disease/training/Tomato_Late_blight',
    'leaf_disease/training/Tomato_Early_blight',
    'leaf_disease/training/Tomato_healthy',
    'leaf_disease/training/Tomato_Yellow_Leaf_Curl_Virus',
    'leaf_disease/training/Tomato_Leaf_Mold',
    'leaf_disease/testing/Tomato_Bacterial_spot',
    'leaf_disease/testing/Tomato_Late_blight',
    'leaf_disease/testing/Tomato_Early_blight',
    'leaf_disease/testing/Tomato_healthy',
    'leaf_disease/testing/Tomato_Yellow_Leaf_Curl_Virus',
    'leaf_disease/testing/Tomato_Leaf_Mold'

]
for directory in to_create:
    try:
        os.mkdir(directory)
        print(directory, 'created')
    except:
        print(directory, 'failed')
```

```
leaf_disease created
leaf_disease/training created
leaf_disease/testing created
leaf_disease/training/Tomato_Bacterial_spot created
leaf_disease/training/Tomato_Late_blight created
leaf_disease/training/Tomato_Early_blight created
leaf_disease/training/Tomato_healthy created
leaf_disease/training/Tomato_Yellow_Leaf_Curl_Virus created
leaf_disease/training/Tomato_Leaf_Mold created
leaf_disease/testing/Tomato_Bacterial_spot created
leaf_disease/testing/Tomato_Late_blight created
leaf_disease/testing/Tomato_Early_blight created
leaf_disease/testing/Tomato_healthy created
leaf_disease/testing/Tomato_Yellow_Leaf_Curl_Virus created
leaf_disease/testing/Tomato_Leaf_Mold created
```

In [14]:
```
def split_data(SOURCE, TRAINING, TESTING, SPLIT_SIZE):
    all_files = []

    for file_name in os.listdir(SOURCE):
        file_path = SOURCE + file_name

        if os.path.getsize(file_path):
            all_files.append(file_name)
        else:
            print('{} is zero length, so ignoring'.format(file_name))

    n_files = len(all_files)
    split_point = int(n_files * SPLIT_SIZE)

    shuffled = random.sample(all_files, n_files)

    train_set = shuffled[:split_point]
    test_set = shuffled[split_point:]

    for file_name in train_set:
        copyfile(SOURCE + file_name, TRAINING + file_name)

    for file_name in test_set:
        copyfile(SOURCE + file_name, TESTING + file_name)
```

In [15]:
```
SOURCE_DIR = "Tomato/Tomato_Late_blight/"
TRAINING_DIR = "leaf_disease/training/Tomato_Late_blight/"
TESTING_DIR ="leaf_disease/testing/Tomato_Late_blight/"

split_size = .8
split_data(SOURCE_DIR, TRAINING_DIR, TESTING_DIR, split_size)
```

In [16]:
```
SOURCE_DIR = "Tomato/Tomato_Bacterial_spot/"
TRAINING_DIR = "leaf_disease/training/Tomato_Bacterial_spot/"
TESTING_DIR ="leaf_disease/testing/Tomato_Bacterial_spot/"

split_size = .8
split_data(SOURCE_DIR, TRAINING_DIR, TESTING_DIR, split_size)
```

In [17]:
```
SOURCE_DIR = "Tomato/Tomato_Early_blight/"
TRAINING_DIR = "leaf_disease/training/Tomato_Early_blight/"
TESTING_DIR ="leaf_disease/testing/Tomato_Early_blight/"

split_size = .8
split_data(SOURCE_DIR, TRAINING_DIR, TESTING_DIR, split_size)
```

In [18]:
```
SOURCE_DIR = "Tomato/Tomato_healthy/"
TRAINING_DIR = "leaf_disease/training/Tomato_healthy/"
TESTING_DIR ="leaf_disease/testing/Tomato_healthy/"

split_size = .8
split_data(SOURCE_DIR, TRAINING_DIR, TESTING_DIR, split_size)
```

In [19]:
```
SOURCE_DIR = "Tomato/Tomato_Yellow_Leaf_Curl_Virus/"
TRAINING_DIR = "leaf_disease/training/Tomato_Yellow_Leaf_Curl_Virus/"
TESTING_DIR ="leaf_disease/testing/Tomato_Yellow_Leaf_Curl_Virus/"

split_size = .8
split_data(SOURCE_DIR, TRAINING_DIR, TESTING_DIR, split_size)
```

In [22]:
```
SOURCE_DIR = "Tomato/Tomato_Leaf_Mold/"
TRAINING_DIR = "leaf_disease/training/Tomato_Leaf_Mold/"
TESTING_DIR ="leaf_disease/testing/Tomato_Leaf_Mold/"

split_size = .8
split_data(SOURCE_DIR, TRAINING_DIR, TESTING_DIR, split_size)
```

In [23]:
```
training_dir = 'leaf_disease/training'
validation_dir = 'leaf_disease/testing'

train_datagen = ImageDataGenerator(rescale=1./255,
      rotation_range=40,
      width_shift_range=0.2,
      height_shift_range=0.2,
      zoom_range=0.2,
      horizontal_flip=True,
     )

train_generator = train_datagen.flow_from_directory(
        training_dir,
        target_size=(256, 256),
        batch_size=128,
        class_mode='categorical')

test_datagen = ImageDataGenerator(rescale=1./255)

validation_generator = test_datagen.flow_from_directory(
        validation_dir,
        target_size=(256, 256),
        batch_size=128,
        class_mode='categorical')
```

```
Found 10346 images belonging to 6 classes.
Found 2590 images belonging to 6 classes.
```

In [24]:
```
model = tf.keras.models.Sequential([
    tf.keras.layers.Conv2D(16, (3,3), activation='relu', input_shape=(256,
    tf.keras.layers.MaxPooling2D(2, 2),

    tf.keras.layers.Conv2D(32, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),

    tf.keras.layers.Conv2D(32, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),

    tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),

    tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),

    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(512, activation='relu'),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(6, activation='softmax')
])
```

In [25]:
```
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_5 (Conv2D)            (None, 254, 254, 16)      448
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 127, 127, 16)      0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 125, 125, 32)      4640
_________________________________________________________________
max_pooling2d_6 (MaxPooling2 (None, 62, 62, 32)        0
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 60, 60, 32)        9248
_________________________________________________________________
max_pooling2d_7 (MaxPooling2 (None, 30, 30, 32)        0
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 28, 28, 64)        18496
_________________________________________________________________
max_pooling2d_8 (MaxPooling2 (None, 14, 14, 64)        0
_________________________________________________________________
conv2d_9 (Conv2D)            (None, 12, 12, 64)        36928
_________________________________________________________________
max_pooling2d_9 (MaxPooling2 (None, 6, 6, 64)          0
_________________________________________________________________
flatten_1 (Flatten)          (None, 2304)              0
_________________________________________________________________
dense_2 (Dense)              (None, 512)               1180160
_________________________________________________________________
dropout_1 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_3 (Dense)              (None, 6)                 3078
=================================================================
Total params: 1,252,998
Trainable params: 1,252,998
Non-trainable params: 0
_________________________________________________________________
```

In [27]:

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

In [28]:

```
history = model.fit(train_generator,
                    epochs=50,
                    steps_per_epoch=68,
                    validation_data=validation_generator,
                    validation_steps=17,
                    verbose=1)
```

```
Epoch 1/50
68/68 [==============================] - 72s 1s/step - loss: 1.4170 - accur
acy: 0.4620 - val_loss: 0.8450 - val_accuracy: 0.6985
Epoch 2/50
68/68 [==============================] - 71s 1s/step - loss: 0.9010 - accur
acy: 0.6789 - val_loss: 0.6059 - val_accuracy: 0.7822
Epoch 3/50
68/68 [==============================] - 71s 1s/step - loss: 0.6514 - accur
acy: 0.7687 - val_loss: 0.5074 - val_accuracy: 0.8107
Epoch 4/50
```

```
68/68 [==============================] - 73s 1s/step - loss: 0.5223 - accur
acy: 0.8120 - val_loss: 0.7347 - val_accuracy: 0.7937
Epoch 5/50
68/68 [==============================] - 74s 1s/step - loss: 0.4554 - accur
acy: 0.8390 - val_loss: 0.3798 - val_accuracy: 0.8644
Epoch 6/50
68/68 [==============================] - 71s 1s/step - loss: 0.3666 - accur
acy: 0.8674 - val_loss: 0.4425 - val_accuracy: 0.8699
Epoch 7/50
68/68 [==============================] - 72s 1s/step - loss: 0.3147 - accur
acy: 0.8883 - val_loss: 0.4011 - val_accuracy: 0.8732
Epoch 8/50
68/68 [==============================] - 71s 1s/step - loss: 0.3384 - accur
acy: 0.8773 - val_loss: 0.6462 - val_accuracy: 0.8212
Epoch 9/50
68/68 [==============================] - 71s 1s/step - loss: 0.2714 - accur
acy: 0.9007 - val_loss: 0.2963 - val_accuracy: 0.8989
Epoch 10/50
68/68 [==============================] - 91s 1s/step - loss: 0.2486 - accur
acy: 0.9107 - val_loss: 0.4135 - val_accuracy: 0.8768
Epoch 11/50
68/68 [==============================] - 72s 1s/step - loss: 0.2690 - accur
acy: 0.9017 - val_loss: 0.3319 - val_accuracy: 0.9136
Epoch 12/50
68/68 [==============================] - 71s 1s/step - loss: 0.2132 - accur
acy: 0.9286 - val_loss: 0.1980 - val_accuracy: 0.9384
Epoch 13/50
68/68 [==============================] - 72s 1s/step - loss: 0.2169 - accur
acy: 0.9227 - val_loss: 0.2241 - val_accuracy: 0.9200
Epoch 14/50
68/68 [==============================] - 72s 1s/step - loss: 0.1886 - accur
acy: 0.9295 - val_loss: 0.5080 - val_accuracy: 0.8603
Epoch 15/50
68/68 [==============================] - 73s 1s/step - loss: 0.2167 - accur
acy: 0.9219 - val_loss: 0.2219 - val_accuracy: 0.9311
Epoch 16/50
68/68 [==============================] - 72s 1s/step - loss: 0.1823 - accur
acy: 0.9363 - val_loss: 0.2427 - val_accuracy: 0.9278
Epoch 17/50
68/68 [==============================] - 73s 1s/step - loss: 0.1572 - accur
acy: 0.9455 - val_loss: 0.1996 - val_accuracy: 0.9472
Epoch 18/50
68/68 [==============================] - 72s 1s/step - loss: 0.1945 - accur
acy: 0.9320 - val_loss: 0.2529 - val_accuracy: 0.9233
Epoch 19/50
68/68 [==============================] - 74s 1s/step - loss: 0.1380 - accur
acy: 0.9516 - val_loss: 0.4985 - val_accuracy: 0.8888
Epoch 20/50
68/68 [==============================] - 73s 1s/step - loss: 0.1447 - accur
acy: 0.9487 - val_loss: 0.6354 - val_accuracy: 0.8465
Epoch 21/50
68/68 [==============================] - 72s 1s/step - loss: 0.1645 - accur
acy: 0.9419 - val_loss: 0.3279 - val_accuracy: 0.9099
Epoch 22/50
68/68 [==============================] - 72s 1s/step - loss: 0.1393 - accur
acy: 0.9546 - val_loss: 0.1892 - val_accuracy: 0.9600
Epoch 23/50
68/68 [==============================] - 72s 1s/step - loss: 0.1204 - accur
acy: 0.9582 - val_loss: 0.2053 - val_accuracy: 0.9407
Epoch 24/50
68/68 [==============================] - 221s 3s/step - loss: 0.1360 - accu
```

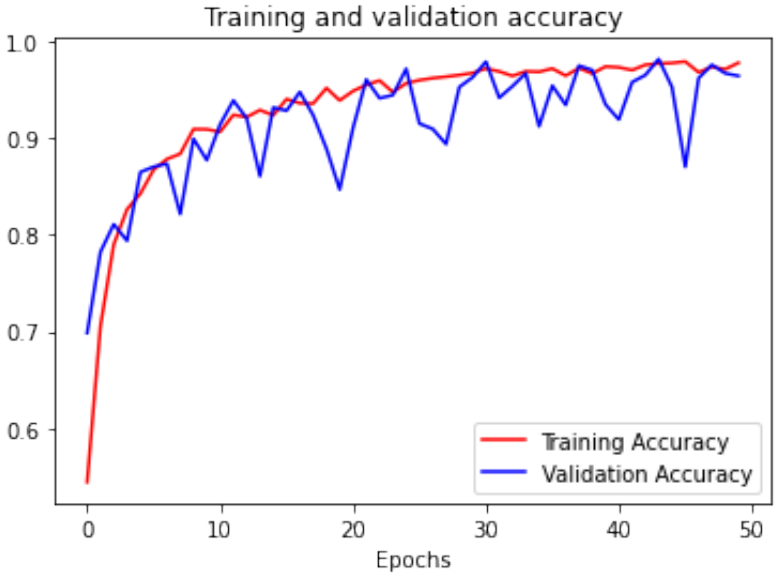
```
racy: 0.9497 - val_loss: 0.2187 - val_accuracy: 0.9439
Epoch 25/50
68/68 [==============================] - 72s 1s/step - loss: 0.1350 - accur
acy: 0.9520 - val_loss: 0.1327 - val_accuracy: 0.9710
Epoch 26/50
68/68 [==============================] - 72s 1s/step - loss: 0.0998 - accur
acy: 0.9621 - val_loss: 0.3747 - val_accuracy: 0.9145
Epoch 27/50
68/68 [==============================] - 304s 5s/step - loss: 0.1112 - accu
racy: 0.9625 - val_loss: 0.3705 - val_accuracy: 0.9090
Epoch 28/50
68/68 [==============================] - 72s 1s/step - loss: 0.1055 - accur
acy: 0.9623 - val_loss: 0.4909 - val_accuracy: 0.8934
Epoch 29/50
68/68 [==============================] - 73s 1s/step - loss: 0.1053 - accur
acy: 0.9636 - val_loss: 0.1503 - val_accuracy: 0.9522
Epoch 30/50
68/68 [==============================] - 71s 1s/step - loss: 0.0910 - accur
acy: 0.9681 - val_loss: 0.1508 - val_accuracy: 0.9623
Epoch 31/50
68/68 [==============================] - 71s 1s/step - loss: 0.0786 - accur
acy: 0.9717 - val_loss: 0.1258 - val_accuracy: 0.9784
Epoch 32/50
68/68 [==============================] - 71s 1s/step - loss: 0.0714 - accur
acy: 0.9745 - val_loss: 0.1743 - val_accuracy: 0.9412
Epoch 33/50
68/68 [==============================] - 71s 1s/step - loss: 0.1162 - accur
acy: 0.9581 - val_loss: 0.1847 - val_accuracy: 0.9531
Epoch 34/50
68/68 [==============================] - 72s 1s/step - loss: 0.0844 - accur
acy: 0.9707 - val_loss: 0.1778 - val_accuracy: 0.9665
Epoch 35/50
68/68 [==============================] - 72s 1s/step - loss: 0.0881 - accur
acy: 0.9697 - val_loss: 0.3513 - val_accuracy: 0.9118
Epoch 36/50
68/68 [==============================] - 71s 1s/step - loss: 0.0807 - accur
acy: 0.9704 - val_loss: 0.2342 - val_accuracy: 0.9536
Epoch 37/50
68/68 [==============================] - 71s 1s/step - loss: 0.1012 - accur
acy: 0.9661 - val_loss: 0.2784 - val_accuracy: 0.9338
Epoch 38/50
68/68 [==============================] - 72s 1s/step - loss: 0.0785 - accur
acy: 0.9715 - val_loss: 0.1351 - val_accuracy: 0.9743
Epoch 39/50
68/68 [==============================] - 71s 1s/step - loss: 0.0874 - accur
acy: 0.9693 - val_loss: 0.1393 - val_accuracy: 0.9697
Epoch 40/50
68/68 [==============================] - 72s 1s/step - loss: 0.0716 - accur
acy: 0.9739 - val_loss: 0.2578 - val_accuracy: 0.9343
Epoch 41/50
68/68 [==============================] - 72s 1s/step - loss: 0.0715 - accur
acy: 0.9746 - val_loss: 0.3685 - val_accuracy: 0.9187
Epoch 42/50
68/68 [==============================] - 71s 1s/step - loss: 0.0920 - accur
acy: 0.9692 - val_loss: 0.1782 - val_accuracy: 0.9573
Epoch 43/50
68/68 [==============================] - 72s 1s/step - loss: 0.0723 - accur
acy: 0.9758 - val_loss: 0.1831 - val_accuracy: 0.9646
Epoch 44/50
68/68 [==============================] - 72s 1s/step - loss: 0.0794 - accur
acy: 0.9752 - val_loss: 0.1342 - val_accuracy: 0.9812
```

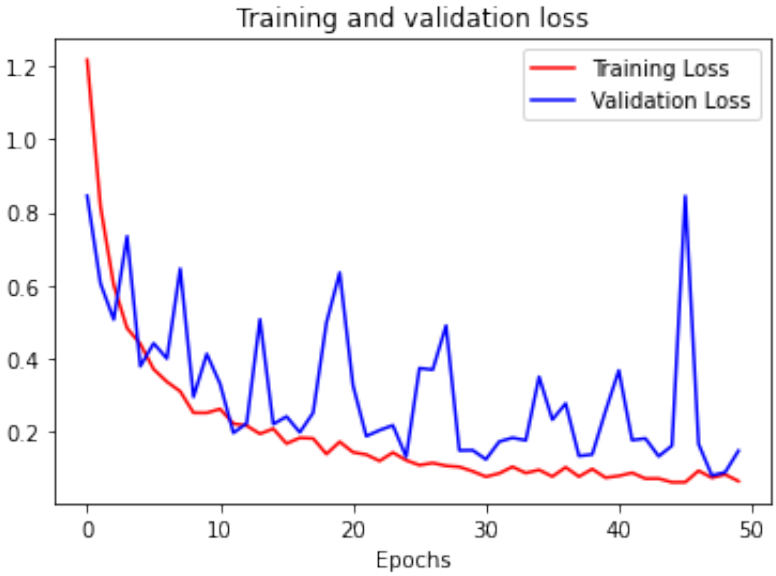
```
Epoch 45/50
68/68 [==============================] - 72s 1s/step - loss: 0.0594 - accur
acy: 0.9785 - val_loss: 0.1632 - val_accuracy: 0.9522
Epoch 46/50
68/68 [==============================] - 71s 1s/step - loss: 0.0529 - accur
acy: 0.9809 - val_loss: 0.8446 - val_accuracy: 0.8699
Epoch 47/50
68/68 [==============================] - 75s 1s/step - loss: 0.1059 - accur
acy: 0.9634 - val_loss: 0.1678 - val_accuracy: 0.9614
Epoch 48/50
68/68 [==============================] - 72s 1s/step - loss: 0.0732 - accur
acy: 0.9734 - val_loss: 0.0819 - val_accuracy: 0.9752
Epoch 49/50
68/68 [==============================] - 72s 1s/step - loss: 0.0988 - accur
acy: 0.9639 - val_loss: 0.0905 - val_accuracy: 0.9665
Epoch 50/50
68/68 [==============================] - 72s 1s/step - loss: 0.0735 - accur
acy: 0.9758 - val_loss: 0.1493 - val_accuracy: 0.9637
```

In [29]:

```
acc=history.history['accuracy']
val_acc=history.history['val_accuracy']
loss=history.history['loss']
val_loss=history.history['val_loss']

epochs=range(len(acc))

plt.plot(epochs, acc, 'r', label="Training Accuracy")
plt.plot(epochs, val_acc, 'b', label="Validation Accuracy")
plt.xlabel('Epochs')
plt.legend()
plt.title('Training and validation accuracy')
plt.figure()

plt.plot(epochs, loss, 'r', label="Training Loss")
plt.plot(epochs, val_loss, 'b', label="Validation Loss")
plt.xlabel('Epochs')
plt.legend()
plt.title('Training and validation loss')
```

Out[29]: Text(0.5, 1.0, 'Training and validation loss')





In [31]:
```
model.save('trained_model.h5')
```

In [ ]:
```
```