In [ ]:

Prepare libraries, import and check dataset

In [6]:
```
import tensorflow as tf
import tensorflow_datasets as tfds
import matplotlib.pyplot as plt
import numpy as np
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.layers import Dense, Flatten, Conv2D, MaxPooling2D, Dr
from tensorflow.keras.models import Sequential
import os
from tensorflow.keras.callbacks import ReduceLROnPlateau
from tensorflow.keras.callbacks import EarlyStopping
```

In [7]:
```
print(tf.__version__)
print(tfds.__version__)
```

```
2.9.2
4.6.0
```

In [8]:
```
# Specify the directory to download the dataset
data_dir = './malaria_datasets'
```

In [9]:
```
# Download the dataset to the specified directory
builder = tfds.builder('malaria', data_dir=data_dir)
builder.download_and_prepare()
```

In [10]:
```
# Load the dataset
dataset, info = tfds.load('malaria', data_dir=data_dir, with_info=True, as_s
full_dataset = dataset['train']
```

In [11]:
```
# Function to inspect a few samples from the dataset
def inspect_dataset(dataset, num_samples=5):
    for image, label in dataset.take(num_samples):
        print("Label: ", label.numpy())
        print("Image shape: ", image.numpy().shape)
        print()
```

In [12]:
```
# Function to inspect and display a few samples from the dataset
def display_dataset(dataset, num_samples=9):
    plt.figure(figsize=(10, 10))
    for i, (image, label) in enumerate(dataset.take(num_samples)):
        ax = plt.subplot(1, num_samples, i + 1)
        plt.imshow(image.numpy().astype("uint8"))
        plt.title(f"Label: {label.numpy()}")
        plt.axis("off")
    plt.show()
```

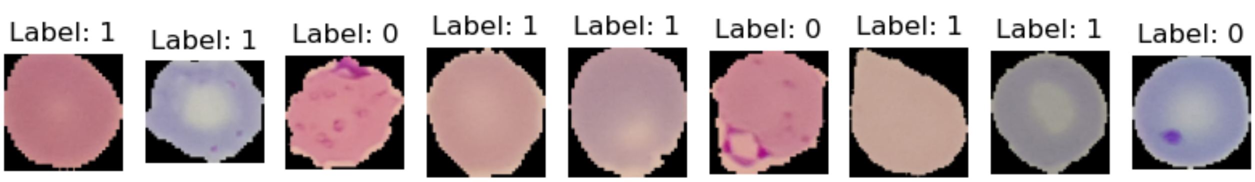
In [13]:
```
# Print dataset information
print(info)
```

```
tfds.core.DatasetInfo(
    name='malaria',
    full_name='malaria/1.0.0',
    description="""
    The Malaria dataset contains a total of 27,558 cell images
    with equal instances of parasitized and uninfected cells from the thin b
lood
    smear slide images of segmented cells.
    """,
    homepage='https://lhncbc.nlm.nih.gov/publication/pub9932',
    data_path='./malaria_datasets/malaria/1.0.0',
    file_format=tfrecord,
    download_size=337.08 MiB,
    dataset_size=317.62 MiB,
    features=FeaturesDict({
        'image': Image(shape=(None, None, 3), dtype=tf.uint8),
        'label': ClassLabel(shape=(), dtype=tf.int64, num_classes=2),
    }),
    supervised_keys=('image', 'label'),
    disable_shuffling=False,
    splits={
        'train': <SplitInfo num_examples=27558, num_shards=4>,
    },
    citation="""@article{rajaraman2018pre,
      title={Pre-trained convolutional neural networks as feature extractors
toward
      improved malaria parasite detection in thin blood smear images},
      author={Rajaraman, Sivaramakrishnan and Antani, Sameer K and Poostchi,
Mahdieh
      and Silamut, Kamolrat and Hossain, Md A and Maude, Richard J and Jaege
r,
      Stefan and Thoma, George R},
      journal={PeerJ},
      volume={6},
      pages={e4568},
      year={2018},
      publisher={PeerJ Inc.}
    }""",
)
```

In [14]:
```
# Inspect a few samples from the dataset
inspect_dataset(full_dataset)
```

```
Label:  1
Image shape:  (103, 103, 3)

Label:  1
Image shape:  (106, 121, 3)

Label:  0
Image shape:  (139, 142, 3)

Label:  1
Image shape:  (130, 118, 3)

Label:  1
Image shape:  (121, 109, 3)
```

In [15]:
```
# Display a few samples from the dataset
display_dataset(full_dataset)
```



Preprocessing dataset

In [16]:
```
# Calculate the number of examples
num_examples = info.splits['train'].num_examples
```

In [17]:
```
# Define the split sizes
train_size = int(0.7 * num_examples)
val_size = int(0.2 * num_examples)
test_size = num_examples - train_size - val_size
```

In [18]:
```
# Shuffle and split the dataset
full_dataset = full_dataset.shuffle(num_examples)
train_dataset = full_dataset.take(train_size)
remaining_dataset = full_dataset.skip(train_size)
val_dataset = remaining_dataset.take(val_size)
test_dataset = remaining_dataset.skip(val_size)
```

In [19]:
```
# Define the image size to uniform size of images
image_size = (130, 130)
```

In [20]:
```
# Function to preprocess the images, resize and uniform
def preprocess(image, label):
    image = tf.image.resize(image, image_size)
    image = image / 255.0  # Normalize the image
    return image, label
```

In [21]:
```
# Apply the preprocessing function to the datasets and batch them
train_dataset = train_dataset.map(preprocess).batch(16).prefetch(tf.data.exp
val_dataset = val_dataset.map(preprocess).batch(16).prefetch(tf.data.experim
test_dataset = test_dataset.map(preprocess).batch(16).prefetch(tf.data.exper
```

Model and training

In [22]:
```
# Function to create and compile the model
def create_model():
    model = Sequential()
    model.add(Conv2D(32, (3, 3), activation='relu', input_shape=image_size +
    model.add(MaxPooling2D((2, 2)))
    model.add(Conv2D(64, (3, 3), activation='relu'))
    model.add(MaxPooling2D((2, 2)))
    model.add(Conv2D(64, (3, 3), activation='relu'))
    model.add(MaxPooling2D((2, 2)))
    model.add(Flatten())
    model.add(Dense(128, activation='relu'))
    model.add(Dropout(0.5))
    model.add(Dense(1, activation='sigmoid'))
    model.compile(optimizer=Adam(learning_rate=0.0001), loss='binary_crossen
    return model
```

In [23]:
```
# Function to plot training progress and save the figure
def plot_training_history(history, model_name):
    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.figure(figsize=(12, 6))
    plt.subplot(1, 2, 1)
    plt.plot(epochs, acc, 'r', label='Training accuracy')
    plt.plot(epochs, val_acc, 'b', label='Validation accuracy')
    plt.title('Training and validation accuracy')
    plt.ylim(0.7, 1.005)
    plt.legend()

    plt.subplot(1, 2, 2)
    plt.plot(epochs, loss, 'r', label='Training loss')
    plt.plot(epochs, val_loss, 'b', label='Validation loss')
    plt.title('Training and validation loss')
    plt.ylim(-0.01, 0.6)
    plt.legend()

    # Save the figure
    if not os.path.exists('figures'):
        os.makedirs('figures')
    plt.savefig(f'figures/{model_name}_training_history.png')
    plt.show()
```

In [24]:
```
# Create the model
model = create_model()
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 conv2d (Conv2D)             (None, 128, 128, 32)      896       
                                                                 
 max_pooling2d (MaxPooling2D  (None, 64, 64, 32)       0         
 )                                                               
                                                                 
 conv2d_1 (Conv2D)           (None, 62, 62, 64)        18496     
                                                                 
 max_pooling2d_1 (MaxPooling  (None, 31, 31, 64)       0         
 2D)                                                             
                                                                 
 conv2d_2 (Conv2D)           (None, 29, 29, 64)        36928     
                                                                 
 max_pooling2d_2 (MaxPooling  (None, 14, 14, 64)       0         
 2D)                                                             
                                                                 
 flatten (Flatten)           (None, 12544)             0         
                                                                 
 dense (Dense)               (None, 128)               1605760   
                                                                 
 dropout (Dropout)           (None, 128)               0         
                                                                 
 dense_1 (Dense)             (None, 1)                 129       
                                                                 
=================================================================
Total params: 1,662,209
Trainable params: 1,662,209
Non-trainable params: 0
_________________________________________________________________
```

In [25]:
```python
# Train the model
history = model.fit(
    train_dataset,
    epochs=80,
    validation_data=val_dataset
)
```

```
Epoch 1/80
1206/1206 [==============================] - 29s 22ms/step - loss: 0.5763 - accuracy: 0.7039 - val_loss: 0.3725 - val_accuracy: 0.8684
Epoch 2/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.2611 - accuracy: 0.9016 - val_loss: 0.2001 - val_accuracy: 0.9227
Epoch 3/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.1829 - accuracy: 0.9346 - val_loss: 0.1632 - val_accuracy: 0.9445
Epoch 4/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.1717 - accuracy: 0.9405 - val_loss: 0.1557 - val_accuracy: 0.9467
Epoch 5/80
```

```
1206/1206 [==============================] - 27s 22ms/step - loss: 0.1632 -
accuracy: 0.9449 - val_loss: 0.1476 - val_accuracy: 0.9523
Epoch 6/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.1515 -
accuracy: 0.9498 - val_loss: 0.1251 - val_accuracy: 0.9583
Epoch 7/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.1442 -
accuracy: 0.9527 - val_loss: 0.1384 - val_accuracy: 0.9477
Epoch 8/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.1340 -
accuracy: 0.9559 - val_loss: 0.1232 - val_accuracy: 0.9586
Epoch 9/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.1362 -
accuracy: 0.9537 - val_loss: 0.1239 - val_accuracy: 0.9557
Epoch 10/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.1240 -
accuracy: 0.9578 - val_loss: 0.1148 - val_accuracy: 0.9608
Epoch 11/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.1160 -
accuracy: 0.9600 - val_loss: 0.1041 - val_accuracy: 0.9648
Epoch 12/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.1171 -
accuracy: 0.9593 - val_loss: 0.1033 - val_accuracy: 0.9637
Epoch 13/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.1086 -
accuracy: 0.9639 - val_loss: 0.0915 - val_accuracy: 0.9693
Epoch 14/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.1063 -
accuracy: 0.9636 - val_loss: 0.0914 - val_accuracy: 0.9661
Epoch 15/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0997 -
accuracy: 0.9653 - val_loss: 0.0908 - val_accuracy: 0.9697
Epoch 16/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.0970 -
accuracy: 0.9666 - val_loss: 0.0891 - val_accuracy: 0.9672
Epoch 17/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.0920 -
accuracy: 0.9688 - val_loss: 0.0761 - val_accuracy: 0.9717
Epoch 18/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.0883 -
accuracy: 0.9685 - val_loss: 0.0705 - val_accuracy: 0.9768
Epoch 19/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.0849 -
accuracy: 0.9703 - val_loss: 0.0728 - val_accuracy: 0.9746
Epoch 20/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.0784 -
accuracy: 0.9725 - val_loss: 0.0597 - val_accuracy: 0.9797
Epoch 21/80
1206/1206 [==============================] - 26s 21ms/step - loss: 0.0753 -
accuracy: 0.9738 - val_loss: 0.0593 - val_accuracy: 0.9770
Epoch 22/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0699 -
accuracy: 0.9754 - val_loss: 0.0518 - val_accuracy: 0.9817
```

```
Epoch 23/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0668 -
accuracy: 0.9769 - val_loss: 0.0558 - val_accuracy: 0.9815
Epoch 24/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0647 -
accuracy: 0.9783 - val_loss: 0.0472 - val_accuracy: 0.9840
Epoch 25/80
1206/1206 [==============================] - 26s 21ms/step - loss: 0.0602 -
accuracy: 0.9784 - val_loss: 0.0414 - val_accuracy: 0.9864
Epoch 26/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0541 -
accuracy: 0.9809 - val_loss: 0.0418 - val_accuracy: 0.9871
Epoch 27/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.0502 -
accuracy: 0.9819 - val_loss: 0.0461 - val_accuracy: 0.9849
Epoch 28/80
1206/1206 [==============================] - 26s 21ms/step - loss: 0.0467 -
accuracy: 0.9849 - val_loss: 0.0321 - val_accuracy: 0.9911
Epoch 29/80
1206/1206 [==============================] - 26s 21ms/step - loss: 0.0434 -
accuracy: 0.9851 - val_loss: 0.0338 - val_accuracy: 0.9895
Epoch 30/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.0420 -
accuracy: 0.9855 - val_loss: 0.0279 - val_accuracy: 0.9913
Epoch 31/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.0427 -
accuracy: 0.9862 - val_loss: 0.0248 - val_accuracy: 0.9926
Epoch 32/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.0391 -
accuracy: 0.9863 - val_loss: 0.0249 - val_accuracy: 0.9918
Epoch 33/80
1206/1206 [==============================] - 26s 21ms/step - loss: 0.0361 -
accuracy: 0.9866 - val_loss: 0.0218 - val_accuracy: 0.9942
Epoch 34/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0329 -
accuracy: 0.9893 - val_loss: 0.0146 - val_accuracy: 0.9953
Epoch 35/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.0302 -
accuracy: 0.9902 - val_loss: 0.0154 - val_accuracy: 0.9953
Epoch 36/80
1206/1206 [==============================] - 26s 21ms/step - loss: 0.0286 -
accuracy: 0.9900 - val_loss: 0.0170 - val_accuracy: 0.9955
Epoch 37/80
1206/1206 [==============================] - 26s 21ms/step - loss: 0.0269 -
accuracy: 0.9903 - val_loss: 0.0162 - val_accuracy: 0.9958
Epoch 38/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0256 -
accuracy: 0.9909 - val_loss: 0.0132 - val_accuracy: 0.9964
Epoch 39/80
1206/1206 [==============================] - 26s 21ms/step - loss: 0.0227 -
accuracy: 0.9923 - val_loss: 0.0140 - val_accuracy: 0.9951
Epoch 40/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.0239 -
```

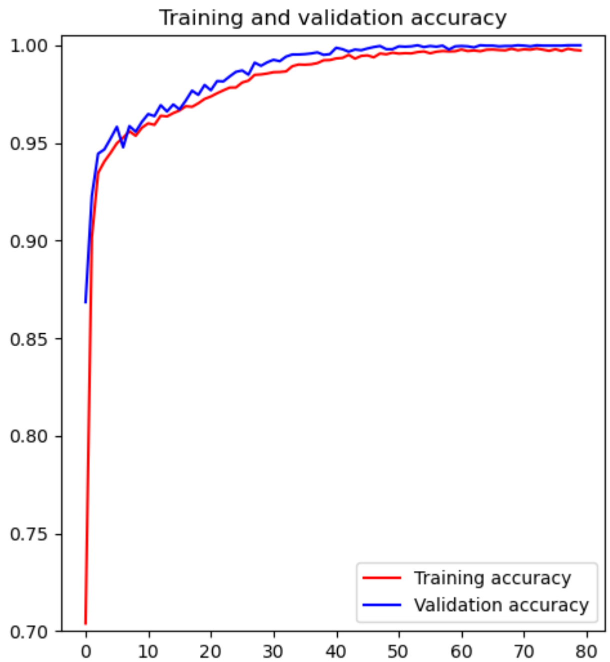
```
accuracy: 0.9924 - val_loss: 0.0135 - val_accuracy: 0.9955
Epoch 41/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0209 -
accuracy: 0.9933 - val_loss: 0.0078 - val_accuracy: 0.9987
Epoch 42/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0220 -
accuracy: 0.9935 - val_loss: 0.0074 - val_accuracy: 0.9980
Epoch 43/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.0153 -
accuracy: 0.9951 - val_loss: 0.0143 - val_accuracy: 0.9967
Epoch 44/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.0184 -
accuracy: 0.9932 - val_loss: 0.0130 - val_accuracy: 0.9978
Epoch 45/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0170 -
accuracy: 0.9946 - val_loss: 0.0075 - val_accuracy: 0.9975
Epoch 46/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0154 -
accuracy: 0.9948 - val_loss: 0.0061 - val_accuracy: 0.9984
Epoch 47/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0166 -
accuracy: 0.9938 - val_loss: 0.0048 - val_accuracy: 0.9991
Epoch 48/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.0138 -
accuracy: 0.9958 - val_loss: 0.0034 - val_accuracy: 0.9996
Epoch 49/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.0144 -
accuracy: 0.9954 - val_loss: 0.0095 - val_accuracy: 0.9980
Epoch 50/80
1206/1206 [==============================] - 26s 21ms/step - loss: 0.0113 -
accuracy: 0.9962 - val_loss: 0.0052 - val_accuracy: 0.9980
Epoch 51/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0131 -
accuracy: 0.9957 - val_loss: 0.0032 - val_accuracy: 0.9995
Epoch 52/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0127 -
accuracy: 0.9960 - val_loss: 0.0040 - val_accuracy: 0.9993
Epoch 53/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.0127 -
accuracy: 0.9959 - val_loss: 0.0026 - val_accuracy: 0.9995
Epoch 54/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0107 -
accuracy: 0.9965 - val_loss: 0.0022 - val_accuracy: 1.0000
Epoch 55/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0097 -
accuracy: 0.9968 - val_loss: 0.0039 - val_accuracy: 0.9991
Epoch 56/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0121 -
accuracy: 0.9959 - val_loss: 0.0021 - val_accuracy: 0.9996
Epoch 57/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0103 -
accuracy: 0.9967 - val_loss: 0.0040 - val_accuracy: 0.9993
Epoch 58/80
```

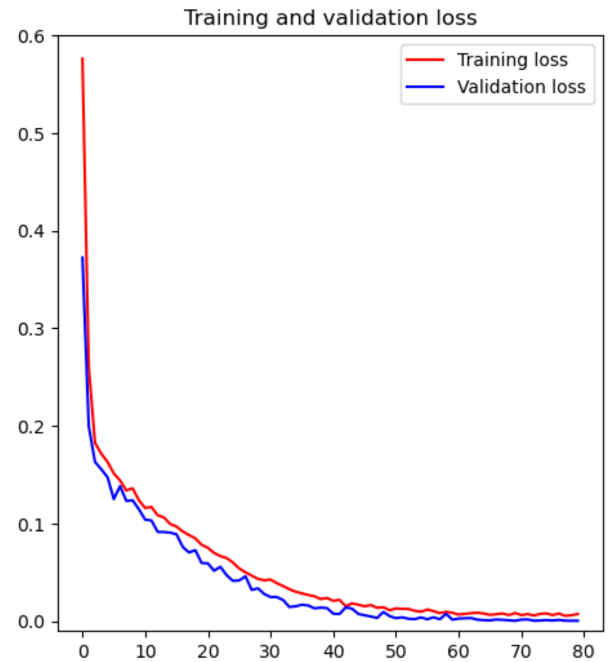
```
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0084 -
accuracy: 0.9970 - val_loss: 0.0021 - val_accuracy: 0.9998
Epoch 59/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0097 -
accuracy: 0.9968 - val_loss: 0.0076 - val_accuracy: 0.9978
Epoch 60/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0087 -
accuracy: 0.9969 - val_loss: 0.0016 - val_accuracy: 0.9995
Epoch 61/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0068 -
accuracy: 0.9978 - val_loss: 0.0028 - val_accuracy: 0.9996
Epoch 62/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.0076 -
accuracy: 0.9970 - val_loss: 0.0031 - val_accuracy: 0.9995
Epoch 63/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0083 -
accuracy: 0.9974 - val_loss: 0.0033 - val_accuracy: 0.9989
Epoch 64/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0088 -
accuracy: 0.9969 - val_loss: 0.0016 - val_accuracy: 1.0000
Epoch 65/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0080 -
accuracy: 0.9977 - val_loss: 0.0011 - val_accuracy: 0.9998
Epoch 66/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0065 -
accuracy: 0.9978 - val_loss: 9.2736e-04 - val_accuracy: 0.9998
Epoch 67/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0072 -
accuracy: 0.9976 - val_loss: 0.0018 - val_accuracy: 0.9995
Epoch 68/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0079 -
accuracy: 0.9974 - val_loss: 0.0015 - val_accuracy: 0.9996
Epoch 69/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0063 -
accuracy: 0.9982 - val_loss: 9.8334e-04 - val_accuracy: 0.9996
Epoch 70/80
1206/1206 [==============================] - 26s 21ms/step - loss: 0.0084 -
accuracy: 0.9975 - val_loss: 4.8609e-04 - val_accuracy: 1.0000
Epoch 71/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0063 -
accuracy: 0.9979 - val_loss: 0.0017 - val_accuracy: 0.9998
Epoch 72/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0074 -
accuracy: 0.9977 - val_loss: 0.0018 - val_accuracy: 0.9995
Epoch 73/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.0059 -
accuracy: 0.9983 - val_loss: 5.9385e-04 - val_accuracy: 1.0000
Epoch 74/80
1206/1206 [==============================] - 26s 21ms/step - loss: 0.0076 -
accuracy: 0.9978 - val_loss: 7.2975e-04 - val_accuracy: 0.9998
Epoch 75/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.0080 -
accuracy: 0.9971 - val_loss: 0.0012 - val_accuracy: 0.9998
```

```
Epoch 76/80
1206/1206 [==============================] - 26s 21ms/step - loss: 0.0064 - accuracy: 0.9980 - val_loss: 7.9143e-04 - val_accuracy: 0.9998
Epoch 77/80
1206/1206 [==============================] - 26s 21ms/step - loss: 0.0079 - accuracy: 0.9971 - val_loss: 0.0014 - val_accuracy: 0.9998
Epoch 78/80
1206/1206 [==============================] - 26s 21ms/step - loss: 0.0056 - accuracy: 0.9982 - val_loss: 5.8515e-04 - val_accuracy: 1.0000
Epoch 79/80
1206/1206 [==============================] - 26s 21ms/step - loss: 0.0062 - accuracy: 0.9976 - val_loss: 5.1499e-04 - val_accuracy: 1.0000
Epoch 80/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.0074 - accuracy: 0.9974 - val_loss: 5.2042e-04 - val_accuracy: 1.0000
```

In [26]:

```
# Plot training history and save the figure
plot_training_history(history, 'basic_model')
```





In [27]:

```
# Evaluate the model using the test data
loss, accuracy = model.evaluate(test_dataset)
print(f"Test Loss: {loss}")
print(f"Test Accuracy: {accuracy}")
```

```
173/173 [==============================] - 2s 7ms/step - loss: 5.7404e-04 - accuracy: 1.0000
Test Loss: 0.000574044301174581
Test Accuracy: 1.0
```

Introduce Learning Rate Scheduling and Early Stopping

In [28]:
```
reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.2, patience=3, mi
early_stopping = EarlyStopping(monitor='val_loss', patience=5, restore_best_
```

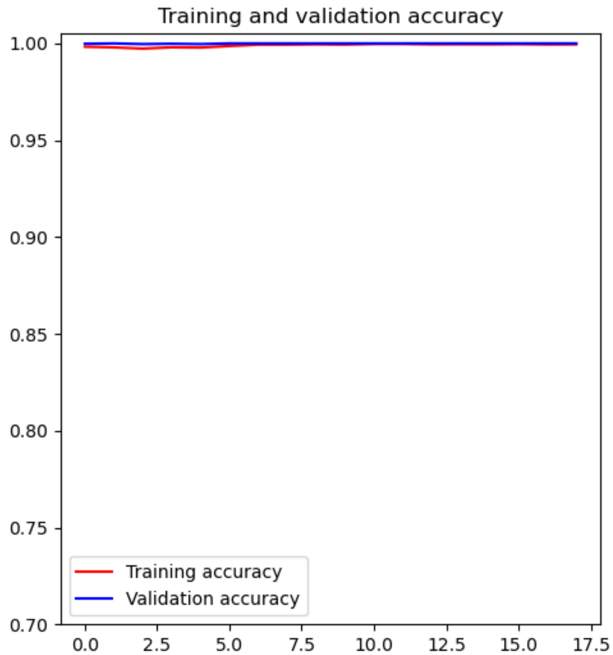
In [29]:
```
history = model.fit(
    train_dataset,
    epochs=80,
    validation_data=val_dataset,
    callbacks=[reduce_lr, early_stopping]
)
```

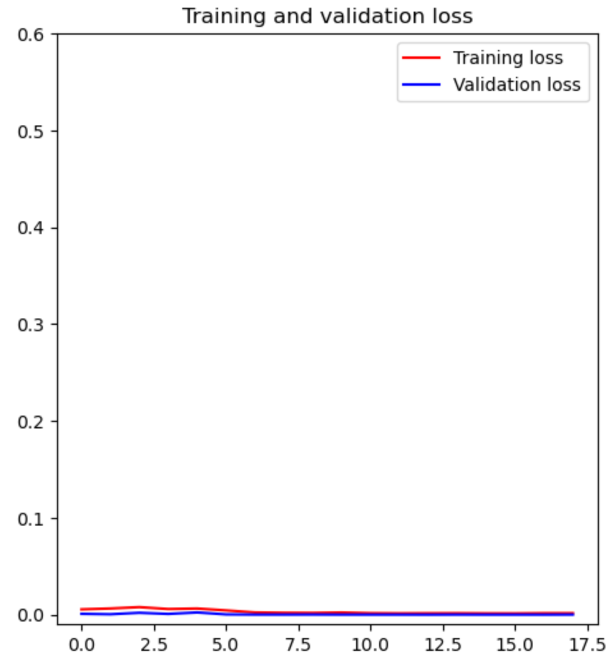
```
Epoch 1/80
1206/1206 [==============================] - 26s 21ms/step - loss: 0.0055 -
accuracy: 0.9983 - val_loss: 8.7567e-04 - val_accuracy: 0.9998 - lr: 1.0000e
-04
Epoch 2/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.0064 -
accuracy: 0.9979 - val_loss: 4.5340e-04 - val_accuracy: 1.0000 - lr: 1.0000e
-04
Epoch 3/80
1206/1206 [==============================] - 26s 21ms/step - loss: 0.0078 -
accuracy: 0.9973 - val_loss: 0.0020 - val_accuracy: 0.9996 - lr: 1.0000e-04
Epoch 4/80
1206/1206 [==============================] - 26s 21ms/step - loss: 0.0059 -
accuracy: 0.9980 - val_loss: 7.8061e-04 - val_accuracy: 0.9998 - lr: 1.0000e
-04
Epoch 5/80
1206/1206 [==============================] - 26s 21ms/step - loss: 0.0064 -
accuracy: 0.9978 - val_loss: 0.0023 - val_accuracy: 0.9996 - lr: 1.0000e-04
Epoch 6/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0044 -
accuracy: 0.9987 - val_loss: 2.9758e-04 - val_accuracy: 1.0000 - lr: 2.0000e
-05
Epoch 7/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.0023 -
accuracy: 0.9993 - val_loss: 1.2412e-04 - val_accuracy: 1.0000 - lr: 2.0000e
-05
Epoch 8/80
1206/1206 [==============================] - 26s 21ms/step - loss: 0.0020 -
accuracy: 0.9993 - val_loss: 1.0557e-04 - val_accuracy: 1.0000 - lr: 2.0000e
-05
Epoch 9/80
1206/1206 [==============================] - 26s 21ms/step - loss: 0.0019 -
accuracy: 0.9995 - val_loss: 1.6534e-04 - val_accuracy: 1.0000 - lr: 2.0000e
-05
Epoch 10/80
1206/1206 [==============================] - 26s 21ms/step - loss: 0.0022 -
```

```
accuracy: 0.9994 - val_loss: 9.3571e-05 - val_accuracy: 1.0000 - lr: 2.0000e
-05
Epoch 11/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0016 -
accuracy: 0.9997 - val_loss: 8.0285e-05 - val_accuracy: 1.0000 - lr: 1.0000e
-05
Epoch 12/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0014 -
accuracy: 0.9997 - val_loss: 7.3930e-05 - val_accuracy: 1.0000 - lr: 1.0000e
-05
Epoch 13/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0015 -
accuracy: 0.9995 - val_loss: 4.2267e-05 - val_accuracy: 1.0000 - lr: 1.0000e
-05
Epoch 14/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0016 -
accuracy: 0.9995 - val_loss: 1.0627e-04 - val_accuracy: 1.0000 - lr: 1.0000e
-05
Epoch 15/80
1206/1206 [==============================] - 26s 21ms/step - loss: 0.0014 -
accuracy: 0.9995 - val_loss: 5.8789e-05 - val_accuracy: 1.0000 - lr: 1.0000e
-05
Epoch 16/80
1206/1206 [==============================] - 26s 21ms/step - loss: 0.0014 -
accuracy: 0.9996 - val_loss: 6.3728e-05 - val_accuracy: 1.0000 - lr: 1.0000e
-05
Epoch 17/80
1206/1206 [==============================] - 27s 22ms/step - loss: 0.0016 -
accuracy: 0.9994 - val_loss: 4.9783e-05 - val_accuracy: 1.0000 - lr: 1.0000e
-05
Epoch 18/80
1206/1206 [==============================] - 26s 22ms/step - loss: 0.0016 -
accuracy: 0.9995 - val_loss: 1.1664e-04 - val_accuracy: 1.0000 - lr: 1.0000e
-05
```

In [30]:

```python
# Plot training history and save the figure
plot_training_history(history, 'basic_model_with_LRS_ES')
```





In [31]:
```
# Evaluate the model using the test data
loss, accuracy = model.evaluate(test_dataset)
print(f"Test Loss: {loss}")
print(f"Test Accuracy: {accuracy}")
```

```
173/173 [==============================] - 2s 6ms/step - loss: 5.2569e-05 -
accuracy: 1.0000
Test Loss: 5.256887379800901e-05
Test Accuracy: 1.0
```

Deeper model

In [32]:
```
# Function to create and compile the deeper model
def create_deeper_model():
    model = Sequential()
    model.add(Conv2D(32, (3, 3), activation='relu', input_shape=image_size +
    model.add(MaxPooling2D((2, 2)))
    model.add(Conv2D(64, (3, 3), activation='relu'))
    model.add(MaxPooling2D((2, 2)))
    model.add(Conv2D(128, (3, 3), activation='relu'))
    model.add(MaxPooling2D((2, 2)))
    model.add(Conv2D(128, (3, 3), activation='relu'))
    model.add(MaxPooling2D((2, 2)))
    model.add(Flatten())
    model.add(Dense(256, activation='relu'))
    model.add(Dropout(0.5))
    model.add(Dense(128, activation='relu'))
    model.add(Dropout(0.5))
    model.add(Dense(1, activation='sigmoid'))
    model.compile(optimizer=Adam(learning_rate=0.0001), loss='binary_crossen
    return model
```

In [33]:
```
# Create the deeper model
model = create_deeper_model()
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 conv2d_3 (Conv2D)           (None, 128, 128, 32)      896       
                                                                 
 max_pooling2d_3 (MaxPooling  (None, 64, 64, 32)       0         
 2D)                                                             
                                                                 
 conv2d_4 (Conv2D)           (None, 62, 62, 64)        18496     
                                                                 
 max_pooling2d_4 (MaxPooling  (None, 31, 31, 64)       0         
 2D)                                                             
                                                                 
 conv2d_5 (Conv2D)           (None, 29, 29, 128)       73856     
                                                                 
 max_pooling2d_5 (MaxPooling  (None, 14, 14, 128)      0         
 2D)                                                             
                                                                 
 conv2d_6 (Conv2D)           (None, 12, 12, 128)       147584    
                                                                 
 max_pooling2d_6 (MaxPooling  (None, 6, 6, 128)        0         
 2D)                                                             
                                                                 
 flatten_1 (Flatten)         (None, 4608)              0         
                                                                 
 dense_2 (Dense)             (None, 256)               1179904   
                                                                 
 dropout_1 (Dropout)         (None, 256)               0         
                                                                 
 dense_3 (Dense)             (None, 128)               32896     
                                                                 
 dropout_2 (Dropout)         (None, 128)               0         
                                                                 
 dense_4 (Dense)             (None, 1)                 129       
                                                                 
=================================================================
Total params: 1,453,761
Trainable params: 1,453,761
Non-trainable params: 0
_________________________________________________________________
```

In [34]:
```python
# Train the deeper model
history = model.fit(
    train_dataset,
    epochs=80,
    validation_data=val_dataset
)
```

```
Epoch 1/80
1206/1206 [==============================] - 33s 26ms/step - loss: 0.4940 - accuracy: 0.7405 - val_loss: 0.1872 - val_accuracy: 0.9216
Epoch 2/80
```

```
1206/1206 [==============================] - 32s 26ms/step - loss: 0.1793 -
accuracy: 0.9445 - val_loss: 0.1316 - val_accuracy: 0.9597
Epoch 3/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.1511 -
accuracy: 0.9536 - val_loss: 0.1314 - val_accuracy: 0.9555
Epoch 4/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.1358 -
accuracy: 0.9573 - val_loss: 0.1303 - val_accuracy: 0.9595
Epoch 5/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.1294 -
accuracy: 0.9579 - val_loss: 0.1247 - val_accuracy: 0.9581
Epoch 6/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.1256 -
accuracy: 0.9584 - val_loss: 0.1121 - val_accuracy: 0.9608
Epoch 7/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.1135 -
accuracy: 0.9626 - val_loss: 0.0990 - val_accuracy: 0.9650
Epoch 8/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.1152 -
accuracy: 0.9606 - val_loss: 0.1077 - val_accuracy: 0.9621
Epoch 9/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.1122 -
accuracy: 0.9631 - val_loss: 0.1034 - val_accuracy: 0.9628
Epoch 10/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.1032 -
accuracy: 0.9646 - val_loss: 0.0879 - val_accuracy: 0.9675
Epoch 11/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0996 -
accuracy: 0.9660 - val_loss: 0.0878 - val_accuracy: 0.9692
Epoch 12/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.1017 -
accuracy: 0.9653 - val_loss: 0.0857 - val_accuracy: 0.9717
Epoch 13/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0940 -
accuracy: 0.9668 - val_loss: 0.0801 - val_accuracy: 0.9737
Epoch 14/80
1206/1206 [==============================] - 33s 27ms/step - loss: 0.0874 -
accuracy: 0.9692 - val_loss: 0.0675 - val_accuracy: 0.9764
Epoch 15/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0865 -
accuracy: 0.9700 - val_loss: 0.0788 - val_accuracy: 0.9770
Epoch 16/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0829 -
accuracy: 0.9715 - val_loss: 0.0637 - val_accuracy: 0.9771
Epoch 17/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0750 -
accuracy: 0.9747 - val_loss: 0.0553 - val_accuracy: 0.9793
Epoch 18/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0702 -
accuracy: 0.9765 - val_loss: 0.0652 - val_accuracy: 0.9773
Epoch 19/80
1206/1206 [==============================] - 33s 27ms/step - loss: 0.0672 -
accuracy: 0.9766 - val_loss: 0.0486 - val_accuracy: 0.9822
```

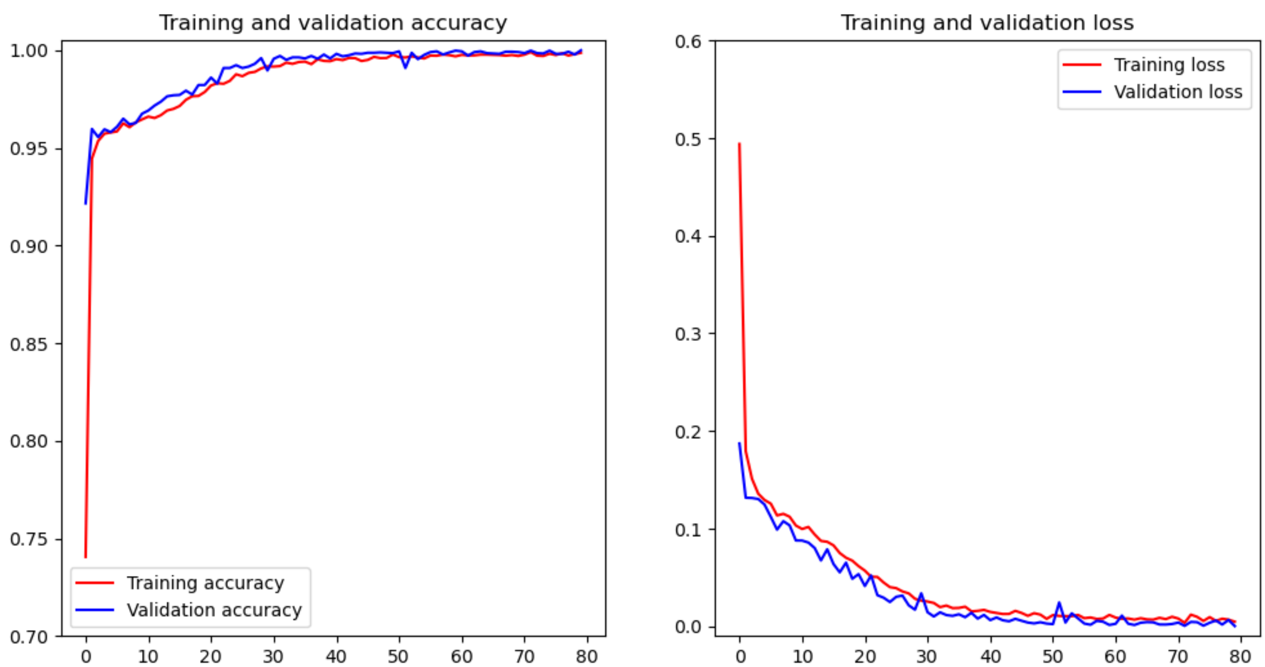
```
Epoch 20/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0616 -
accuracy: 0.9786 - val_loss: 0.0534 - val_accuracy: 0.9822
Epoch 21/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0570 -
accuracy: 0.9820 - val_loss: 0.0413 - val_accuracy: 0.9860
Epoch 22/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0507 -
accuracy: 0.9832 - val_loss: 0.0520 - val_accuracy: 0.9829
Epoch 23/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0505 -
accuracy: 0.9828 - val_loss: 0.0319 - val_accuracy: 0.9909
Epoch 24/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0448 -
accuracy: 0.9843 - val_loss: 0.0292 - val_accuracy: 0.9909
Epoch 25/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0401 -
accuracy: 0.9877 - val_loss: 0.0248 - val_accuracy: 0.9924
Epoch 26/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0391 -
accuracy: 0.9867 - val_loss: 0.0302 - val_accuracy: 0.9909
Epoch 27/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0357 -
accuracy: 0.9884 - val_loss: 0.0314 - val_accuracy: 0.9917
Epoch 28/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0337 -
accuracy: 0.9890 - val_loss: 0.0215 - val_accuracy: 0.9931
Epoch 29/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0281 -
accuracy: 0.9907 - val_loss: 0.0170 - val_accuracy: 0.9960
Epoch 30/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0263 -
accuracy: 0.9918 - val_loss: 0.0338 - val_accuracy: 0.9897
Epoch 31/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0253 -
accuracy: 0.9917 - val_loss: 0.0144 - val_accuracy: 0.9956
Epoch 32/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0239 -
accuracy: 0.9919 - val_loss: 0.0100 - val_accuracy: 0.9971
Epoch 33/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0196 -
accuracy: 0.9937 - val_loss: 0.0143 - val_accuracy: 0.9951
Epoch 34/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0212 -
accuracy: 0.9931 - val_loss: 0.0115 - val_accuracy: 0.9964
Epoch 35/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0186 -
accuracy: 0.9940 - val_loss: 0.0107 - val_accuracy: 0.9964
Epoch 36/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0188 -
accuracy: 0.9942 - val_loss: 0.0123 - val_accuracy: 0.9958
Epoch 37/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0201 -
```

```
accuracy: 0.9929 - val_loss: 0.0093 - val_accuracy: 0.9971
Epoch 38/80
1206/1206 [==============================] - 32s 27ms/step - loss: 0.0153 -
accuracy: 0.9954 - val_loss: 0.0138 - val_accuracy: 0.9956
Epoch 39/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0158 -
accuracy: 0.9946 - val_loss: 0.0078 - val_accuracy: 0.9978
Epoch 40/80
1206/1206 [==============================] - 35s 28ms/step - loss: 0.0168 -
accuracy: 0.9944 - val_loss: 0.0116 - val_accuracy: 0.9958
Epoch 41/80
1206/1206 [==============================] - 32s 27ms/step - loss: 0.0147 -
accuracy: 0.9955 - val_loss: 0.0063 - val_accuracy: 0.9982
Epoch 42/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0136 -
accuracy: 0.9950 - val_loss: 0.0090 - val_accuracy: 0.9969
Epoch 43/80
1206/1206 [==============================] - 33s 27ms/step - loss: 0.0126 -
accuracy: 0.9961 - val_loss: 0.0062 - val_accuracy: 0.9975
Epoch 44/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0126 -
accuracy: 0.9960 - val_loss: 0.0051 - val_accuracy: 0.9984
Epoch 45/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0158 -
accuracy: 0.9945 - val_loss: 0.0076 - val_accuracy: 0.9982
Epoch 46/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0136 -
accuracy: 0.9951 - val_loss: 0.0056 - val_accuracy: 0.9987
Epoch 47/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0107 -
accuracy: 0.9966 - val_loss: 0.0038 - val_accuracy: 0.9987
Epoch 48/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0134 -
accuracy: 0.9960 - val_loss: 0.0029 - val_accuracy: 0.9989
Epoch 49/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0119 -
accuracy: 0.9961 - val_loss: 0.0041 - val_accuracy: 0.9987
Epoch 50/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0076 -
accuracy: 0.9978 - val_loss: 0.0028 - val_accuracy: 0.9985
Epoch 51/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0115 -
accuracy: 0.9965 - val_loss: 0.0022 - val_accuracy: 0.9995
Epoch 52/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0106 -
accuracy: 0.9963 - val_loss: 0.0244 - val_accuracy: 0.9909
Epoch 53/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0104 -
accuracy: 0.9969 - val_loss: 0.0038 - val_accuracy: 0.9987
Epoch 54/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0104 -
accuracy: 0.9963 - val_loss: 0.0133 - val_accuracy: 0.9955
Epoch 55/80
```

```
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0116 -
accuracy: 0.9958 - val_loss: 0.0081 - val_accuracy: 0.9976
Epoch 56/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0083 -
accuracy: 0.9973 - val_loss: 0.0028 - val_accuracy: 0.9991
Epoch 57/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0090 -
accuracy: 0.9972 - val_loss: 0.0016 - val_accuracy: 0.9995
Epoch 58/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0075 -
accuracy: 0.9978 - val_loss: 0.0056 - val_accuracy: 0.9978
Epoch 59/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0079 -
accuracy: 0.9975 - val_loss: 0.0046 - val_accuracy: 0.9989
Epoch 60/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0116 -
accuracy: 0.9969 - val_loss: 0.0014 - val_accuracy: 0.9998
Epoch 61/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0089 -
accuracy: 0.9977 - val_loss: 0.0026 - val_accuracy: 0.9995
Epoch 62/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0077 -
accuracy: 0.9974 - val_loss: 0.0110 - val_accuracy: 0.9973
Epoch 63/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0080 -
accuracy: 0.9974 - val_loss: 0.0026 - val_accuracy: 0.9991
Epoch 64/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0068 -
accuracy: 0.9978 - val_loss: 0.0013 - val_accuracy: 0.9995
Epoch 65/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0082 -
accuracy: 0.9978 - val_loss: 0.0036 - val_accuracy: 0.9985
Epoch 66/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0071 -
accuracy: 0.9977 - val_loss: 0.0043 - val_accuracy: 0.9984
Epoch 67/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0068 -
accuracy: 0.9976 - val_loss: 0.0040 - val_accuracy: 0.9982
Epoch 68/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0087 -
accuracy: 0.9973 - val_loss: 0.0018 - val_accuracy: 0.9993
Epoch 69/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0073 -
accuracy: 0.9976 - val_loss: 0.0018 - val_accuracy: 0.9993
Epoch 70/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0097 -
accuracy: 0.9971 - val_loss: 0.0023 - val_accuracy: 0.9991
Epoch 71/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0079 -
accuracy: 0.9978 - val_loss: 0.0038 - val_accuracy: 0.9985
Epoch 72/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0035 -
accuracy: 0.9991 - val_loss: 4.3958e-04 - val_accuracy: 0.9998
```

```
Epoch 73/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0119 -
accuracy: 0.9973 - val_loss: 0.0045 - val_accuracy: 0.9985
Epoch 74/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0096 -
accuracy: 0.9971 - val_loss: 0.0042 - val_accuracy: 0.9984
Epoch 75/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0055 -
accuracy: 0.9982 - val_loss: 6.0558e-04 - val_accuracy: 0.9998
Epoch 76/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0094 -
accuracy: 0.9976 - val_loss: 0.0036 - val_accuracy: 0.9982
Epoch 77/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0055 -
accuracy: 0.9985 - val_loss: 0.0059 - val_accuracy: 0.9982
Epoch 78/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0077 -
accuracy: 0.9973 - val_loss: 0.0018 - val_accuracy: 0.9993
Epoch 79/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0068 -
accuracy: 0.9980 - val_loss: 0.0065 - val_accuracy: 0.9978
Epoch 80/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0047 -
accuracy: 0.9987 - val_loss: 1.9709e-04 - val_accuracy: 1.0000
```

In [35]:
```
# Plot training history and save the figure
plot_training_history(history, 'deeper_model')
```



In [36]:
```
# Evaluate the deeper model using the test data
loss, accuracy = model.evaluate(test_dataset)
print(f"Test Loss: {loss}")
print(f"Test Accuracy: {accuracy}")
```

```
173/173 [==============================] - 2s 8ms/step - loss: 0.0018 - accu
racy: 0.9996
Test Loss: 0.0018338192021474242
Test Accuracy: 0.9996373057365417
```

Introduce Learning Rate Scheduling and Early Stopping for deeper model

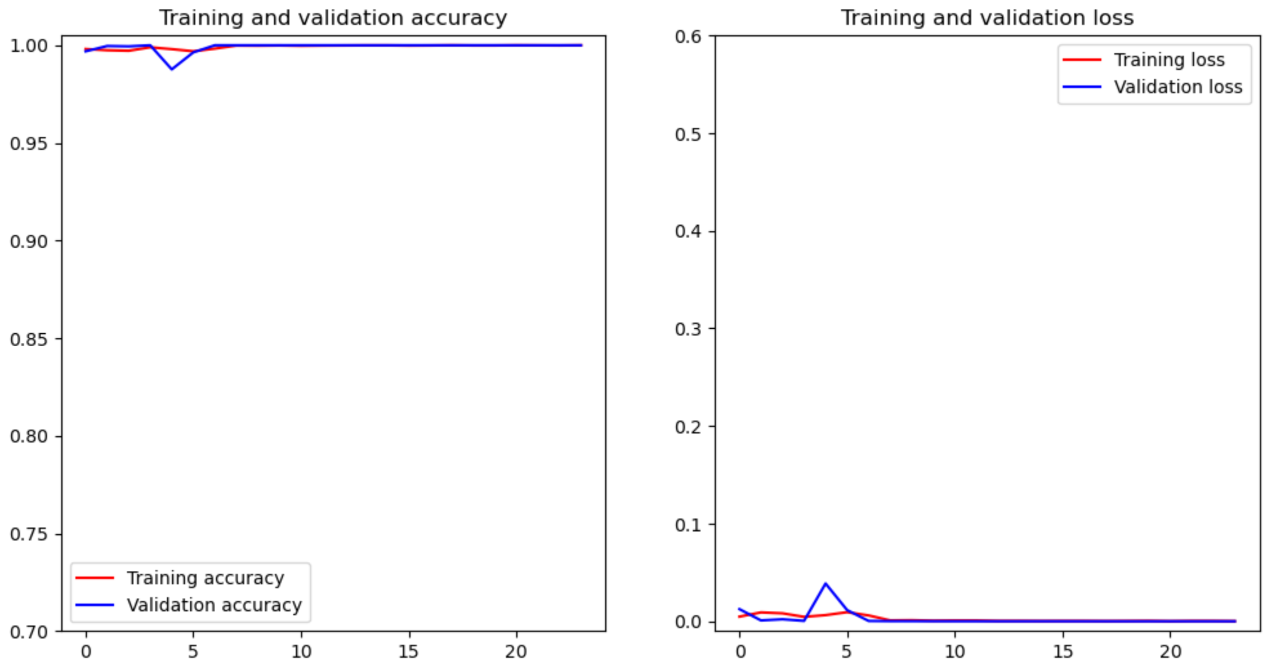
In [37]:
```
history = model.fit(
    train_dataset,
    epochs=80,
    validation_data=val_dataset,
    callbacks=[reduce_lr, early_stopping]
)
```

```
Epoch 1/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0047 -
accuracy: 0.9981 - val_loss: 0.0124 - val_accuracy: 0.9969 - lr: 1.0000e-04
Epoch 2/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0091 -
accuracy: 0.9975 - val_loss: 8.7114e-04 - val_accuracy: 0.9996 - lr: 1.0000e
-04
Epoch 3/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0081 -
accuracy: 0.9972 - val_loss: 0.0021 - val_accuracy: 0.9995 - lr: 1.0000e-04
Epoch 4/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0045 -
accuracy: 0.9990 - val_loss: 4.1087e-04 - val_accuracy: 1.0000 - lr: 1.0000e
-04
Epoch 5/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0063 -
accuracy: 0.9980 - val_loss: 0.0387 - val_accuracy: 0.9877 - lr: 1.0000e-04
Epoch 6/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0093 -
accuracy: 0.9969 - val_loss: 0.0114 - val_accuracy: 0.9964 - lr: 1.0000e-04
Epoch 7/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0060 -
accuracy: 0.9983 - val_loss: 3.3159e-04 - val_accuracy: 1.0000 - lr: 1.0000e
-04
Epoch 8/80
1206/1206 [==============================] - 32s 26ms/step - loss: 9.1099e-0
4 - accuracy: 0.9998 - val_loss: 1.4918e-04 - val_accuracy: 1.0000 - lr: 2.0
000e-05
Epoch 9/80
1206/1206 [==============================] - 32s 26ms/step - loss: 0.0011 -
accuracy: 0.9998 - val_loss: 1.1557e-04 - val_accuracy: 1.0000 - lr: 2.0000e
-05
Epoch 10/80
1206/1206 [==============================] - 32s 26ms/step - loss: 6.1037e-0
4 - accuracy: 0.9999 - val_loss: 8.9402e-05 - val_accuracy: 1.0000 - lr: 2.0
```

```
000e-05
Epoch 11/80
1206/1206 [==============================] - 32s 26ms/step - loss: 7.9199e-0
4 - accuracy: 0.9997 - val_loss: 5.3075e-05 - val_accuracy: 1.0000 - lr: 2.0
000e-05
Epoch 12/80
1206/1206 [==============================] - 32s 26ms/step - loss: 7.7600e-0
4 - accuracy: 0.9998 - val_loss: 3.8047e-05 - val_accuracy: 1.0000 - lr: 1.0
000e-05
Epoch 13/80
1206/1206 [==============================] - 32s 26ms/step - loss: 3.6677e-0
4 - accuracy: 0.9999 - val_loss: 2.6674e-05 - val_accuracy: 1.0000 - lr: 1.0
000e-05
Epoch 14/80
1206/1206 [==============================] - 32s 26ms/step - loss: 3.6784e-0
4 - accuracy: 0.9999 - val_loss: 3.0078e-05 - val_accuracy: 1.0000 - lr: 1.0
000e-05
Epoch 15/80
1206/1206 [==============================] - 32s 26ms/step - loss: 2.5355e-0
4 - accuracy: 0.9999 - val_loss: 4.9924e-05 - val_accuracy: 1.0000 - lr: 1.0
000e-05
Epoch 16/80
1206/1206 [==============================] - 32s 26ms/step - loss: 2.9930e-0
4 - accuracy: 0.9998 - val_loss: 4.0461e-05 - val_accuracy: 1.0000 - lr: 1.0
000e-05
Epoch 17/80
1206/1206 [==============================] - 32s 26ms/step - loss: 3.0150e-0
4 - accuracy: 0.9999 - val_loss: 2.8247e-05 - val_accuracy: 1.0000 - lr: 1.0
000e-05
Epoch 18/80
1206/1206 [==============================] - 32s 26ms/step - loss: 1.7783e-0
4 - accuracy: 1.0000 - val_loss: 2.0468e-05 - val_accuracy: 1.0000 - lr: 1.0
000e-05
Epoch 19/80
1206/1206 [==============================] - 32s 26ms/step - loss: 2.7635e-0
4 - accuracy: 0.9999 - val_loss: 9.2260e-06 - val_accuracy: 1.0000 - lr: 1.0
000e-05
Epoch 20/80
1206/1206 [==============================] - 32s 26ms/step - loss: 3.9879e-0
4 - accuracy: 0.9999 - val_loss: 1.4197e-05 - val_accuracy: 1.0000 - lr: 1.0
000e-05
Epoch 21/80
1206/1206 [==============================] - 32s 26ms/step - loss: 8.8338e-0
5 - accuracy: 1.0000 - val_loss: 1.1735e-05 - val_accuracy: 1.0000 - lr: 1.0
000e-05
Epoch 22/80
1206/1206 [==============================] - 32s 26ms/step - loss: 3.1754e-0
4 - accuracy: 0.9999 - val_loss: 1.7731e-05 - val_accuracy: 1.0000 - lr: 1.0
000e-05
Epoch 23/80
1206/1206 [==============================] - 32s 26ms/step - loss: 3.0307e-0
4 - accuracy: 0.9999 - val_loss: 1.1800e-05 - val_accuracy: 1.0000 - lr: 1.0
000e-05
```

```
Epoch 24/80
1206/1206 [==============================] - 32s 26ms/step - loss: 9.1791e-0
5 - accuracy: 1.0000 - val_loss: 9.9595e-06 - val_accuracy: 1.0000 - lr: 1.0
000e-05
```

In [38]:
```
# Plot training history and save the figure
plot_training_history(history, 'deeper_model_with_LRS_ES')
```



In [39]:
```
# Evaluate the model using the test data
loss, accuracy = model.evaluate(test_dataset)
print(f"Test Loss: {loss}")
print(f"Test Accuracy: {accuracy}")
```

```
173/173 [==============================] - 2s 8ms/step - loss: 1.0464e-05 -
accuracy: 1.0000
Test Loss: 1.0464496881468222e-05
Test Accuracy: 1.0
```

In [ ]:
```

```