Import necessary libraries

In [9]:

```
import warnings
warnings.filterwarnings("ignore", category=DeprecationWarning)

import tensorflow as tf
import numpy as np
from tensorflow import keras
import matplotlib.pyplot as plt
import random
import os
import itertools
import datetime
from tensorflow.keras.layers.experimental.preprocessing import Rescaling
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from sklearn.model_selection import train_test_split
from tensorflow.keras import layers
from sklearn.metrics import precision_score, accuracy_score, recall_score, confusion_mat
```

Step 1: Data Augmentation

Define directories for dataset

In [7]:

```
dataset_dir = 'Fruit And Vegetable Diseases Dataset'
```

Balance the dataset by oversampling

In [8]:

```
def oversample_dataset(directory, target_size):
    datagen = ImageDataGenerator(
        rotation_range=20,
        width_shift_range=0.2,
        height_shift_range=0.2,
        shear_range=0.2,
        zoom_range=0.2,
        horizontal_flip=True,
        fill_mode='nearest'
    )

    class_folders = os.listdir(directory)

    for folder in class_folders:
        path = os.path.join(directory, folder)
        images = [os.path.join(path, img) for img in os.listdir(path)]

        # Check if oversampling is needed for the current class
        if len(images) < target_size:
            samples_to_add = target_size - len(images)
            if samples_to_add > 0:
                # Apply data augmentation for oversampling
                augmentation_gen = datagen.flow_from_directory(
                    directory=directory,
                    classes=[folder],
                    target_size=(224, 224),
                    batch_size=samples_to_add,
                    class_mode='categorical'
                )

                num_generated_images = 0
                while num_generated_images < samples_to_add:
                    batch = augmentation_gen.next()
                    num_batch_images = batch[0].shape[0]
                    for i in range(num_batch_images):
                        if num_generated_images >= samples_to_add:
                            break

                        fruit_name = folder

                        image = batch[0][i].squeeze()
                        image_filename = f'{fruit_name}_augmented_{num_generated_images}
                        tf.keras.preprocessing.image.save_img(os.path.join(path, image_f
                        images.append(os.path.join(path, image_filename))
                        num_generated_images += 1
```

Balance and oversample the dataset

In [12]:

```
target_size = 2000
oversample_dataset(dataset_dir, target_size)
```

```
Found 1641 images belonging to 1 classes.
Found 611 images belonging to 1 classes.
Found 591 images belonging to 1 classes.
Found 579 images belonging to 1 classes.
Found 619 images belonging to 1 classes.
Found 608 images belonging to 1 classes.
Found 593 images belonging to 1 classes.
Found 200 images belonging to 1 classes.
Found 200 images belonging to 1 classes.
Found 200 images belonging to 1 classes.
Found 200 images belonging to 1 classes.
Found 200 images belonging to 1 classes.
Found 200 images belonging to 1 classes.
Found 1813 images belonging to 1 classes.
Found 200 images belonging to 1 classes.
Found 200 images belonging to 1 classes.
Found 614 images belonging to 1 classes.
C:\Users\PMLS\anaconda3\Lib\site-packages\PIL\Image.py:970: UserWarning: Palette images
with Transparency expressed in bytes should be converted to RGBA images
  warnings.warn(
Found 584 images belonging to 1 classes.
Found 1603 images belonging to 1 classes.
Found 1596 images belonging to 1 classes.
Found 604 images belonging to 1 classes.
Found 595 images belonging to 1 classes.
```

Seperate Train and Valid Dataset

In [13]:

```
import os
import shutil
from sklearn.model_selection import train_test_split

def split_data(input_folder, output_folder, split_ratio):
    # List all subdirectories in the input folder
    subdirectories = [f.path for f in os.scandir(input_folder) if f.is_dir()]

    # Iterate through each subdirectory
    for subdirectory in subdirectories:
        # Get the class/category name from the subdirectory path
        class_name = os.path.basename(subdirectory)

        # List all files in the current subdirectory
        files = [f.path for f in os.scandir(subdirectory) if f.is_file()]
        # Split files into training and testing sets
        train_files, test_files = train_test_split(files, test_size=split_ratio, random_

        # Create output folders for training and testing sets
        train_output_folder = os.path.join(output_folder, 'train', class_name)
        test_output_folder = os.path.join(output_folder, 'valid', class_name)
        os.makedirs(train_output_folder, exist_ok=True)
        os.makedirs(test_output_folder, exist_ok=True)

        # Copy training files to the training output folder
        for train_file in train_files:
            shutil.copy(train_file, train_output_folder)

        # Copy testing files to the testing output folder
        for test_file in test_files:
            shutil.copy(test_file, test_output_folder)

# Path to the output directory where the train and valid folders will be created
output_dir = 'Fruit And Veg Diseases Dataset'
split_ratio = 0.2  # Adjust the split ratio as needed

split_data(dataset_dir, output_dir, split_ratio)
```

Step 2: Data Preparation

Define directories for training and testing data

In [1]:

```
train_dir = 'Fruit And Veg Diseases Dataset/train'
test_dir = 'Fruit And Veg Diseases Dataset/valid'
```

In [4]:

```
from PIL import Image
import os

def filter_images_by_format(directory, allowed_formats):
    for root, dirs, files in os.walk(directory):
        for file in files:
            file_path = os.path.join(root, file)
            try:
                # Open the image to check its format
                with Image.open(file_path) as img:
                    img_format = img.format.upper()

                # Check if the format is not in the allowed_formats list
                if img_format not in allowed_formats:
                    print(f"Deleting {file_path} (unsupported format: {img_format})")
                    os.remove(file_path)

            except Exception as e:
                print(f"Error processing {file_path}: {e}")

# Specify the allowed image formats
allowed_formats = ['JPEG', 'PNG', 'JPG', 'GIF', 'BMP']

# Apply the filter to the training directory
filter_images_by_format(train_dir, allowed_formats)

# Apply the filter to the testing directory
filter_images_by_format(test_dir, allowed_formats)
```

```
Deleting Fruit And Veg Diseases Dataset/train\Potato__Rotten\rottenPotato (1).webp (unsu
pported format: WEBP)
Deleting Fruit And Veg Diseases Dataset/train\Tomato__Rotten\rottenTomato (1).webp (unsu
pported format: WEBP)
Deleting Fruit And Veg Diseases Dataset/valid\Banana__Healthy\freshBanana (1).webp (unsu
pported format: WEBP)
Deleting Fruit And Veg Diseases Dataset/valid\Banana__Rotten\rottenBanana (1).webp (unsu
pported format: WEBP)
Deleting Fruit And Veg Diseases Dataset/valid\Carrot__Healthy\freshCarrot (415).jpg (uns
upported format: WEBP)
```

In [10]:
```
train_dir = 'Fruit And Veg Diseases Dataset/train'
test_dir = 'Fruit And Veg Diseases Dataset/valid'
```

Create image datasets for training and testing

In [11]:
```
train_data = keras.utils.image_dataset_from_directory(train_dir,
                                                      image_size=(224, 224),
                                                      label_mode='categorical',
                                                      batch_size=32)
```

```
Found 46924 files belonging to 28 classes.
```

In [12]:
```
test_data = keras.utils.image_dataset_from_directory(test_dir,
                                                     image_size=(224, 224),
                                                     label_mode='categorical',
                                                     batch_size=32)
```

```
Found 11734 files belonging to 28 classes.
```

Define class names based on the directory structure

In [13]:
```
class_names = train_data.class_names
class_names
```

Out[13]:
```
['Apple__Healthy',
 'Apple__Rotten',
 'Banana__Healthy',
 'Banana__Rotten',
 'Bellpepper__Healthy',
 'Bellpepper__Rotten',
 'Carrot__Healthy',
 'Carrot__Rotten',
 'Cucumber__Healthy',
 'Cucumber__Rotten',
 'Grape__Healthy',
 'Grape__Rotten',
 'Guava__Healthy',
 'Guava__Rotten',
 'Jujube__Healthy',
 'Jujube__Rotten',
 'Mango__Healthy',
 'Mango__Rotten',
 'Orange__Healthy',
 'Orange__Rotten',
 'Pomegranate__Healthy',
 'Pomegranate__Rotten',
 'Potato__Healthy',
 'Potato__Rotten',
 'Strawberry__Healthy',
 'Strawberry__Rotten',
 'Tomato__Healthy',
 'Tomato__Rotten']
```

Step 2: Model Creation

Define the input image shape

```
In [14]: image_shape = (224, 224, 3)
```

Create a base model (EfficientNetB0) for feature extraction

```
In [15]: base_model = tf.keras.applications.EfficientNetB0(include_top=False, weights='imagenet')
         base_model.trainable = False
```

```
WARNING:tensorflow:From C:\Users\PMLS\anaconda3\Lib\site-packages\keras\src\backend.py:1
398: The name tf.executing_eagerly_outside_functions is deprecated. Please use tf.compa
t.v1.executing_eagerly_outside_functions instead.

WARNING:tensorflow:From C:\Users\PMLS\anaconda3\Lib\site-packages\keras\src\layers\norma
lization\batch_normalization.py:979: The name tf.nn.fused_batch_norm is deprecated. Plea
se use tf.compat.v1.nn.fused_batch_norm instead.

Downloading data from https://storage.googleapis.com/keras-applications/efficientnetb0_n
otop.h5
16705208/16705208 [==============================] - 19s 1us/step
```

Create the main model by adding layers on top of the base model

```
In [16]: inputs = layers.Input(shape=image_shape, name='input_layer')
         x = base_model(inputs, training=False)
         x = layers.GlobalAveragePooling2D(name='GlobalAveragePooling2D_layer')(x)
         outputs = layers.Dense(len(class_names), activation='softmax', name='output_layer')(x)
         feature_model = tf.keras.Model(inputs, outputs, name='Fruit_Vegetable_Diseases_Detection
```

Set some layers in the base model as trainable

```
In [17]: base_model.trainable = True
         for layer in base_model.layers[:-20]:
             layer.trainable = False
```

Compile the model

```
In [18]: feature_model.compile(
             loss='categorical_crossentropy',
             optimizer=tf.keras.optimizers.Adam(learning_rate=0.0001),
             metrics=['accuracy']
         )
```

```
In [19]: base_model.summary()
```

```
Model: "efficientnetb0"
________________________________________________________________________________________
__________
 Layer (type)                Output Shape                 Param #   Connected to

========================================================================================
==========
 input_1 (InputLayer)        [(None, None, None, 3)]      0         []


 rescaling (Rescaling)       (None, None, None, 3)        0         ['input_1[0][0]']


 normalization (Normalizati  (None, None, None, 3)        7         ['rescaling[0][0]']
 on)


 rescaling_1 (Rescaling)     (None, None, None, 3)        0         ['normalization[0]
[0]']


 stem_conv_pad (ZeroPadding  (None, None, None, 3)        0         ['rescaling_1[0]
[0]']
 2D)


 stem_conv (Conv2D)          (None, None, None, 32)       864       ['stem_conv_pad[0]
[0]']


 stem_bn (BatchNormalizatio  (None, None, None, 32)       128       ['stem_conv[0][0]']

 n)


 stem_activation (Activatio  (None, None, None, 32)       0         ['stem_bn[0][0]']

 n)


 block1a_dwconv (DepthwiseC  (None, None, None, 32)       288       ['stem_activation[0]
[0]']
 onv2D)


 block1a_bn (BatchNormaliza  (None, None, None, 32)       128       ['block1a_dwconv[0]
[0]']
 tion)


 block1a_activation (Activa  (None, None, None, 32)       0         ['block1a_bn[0][0]']

 tion)


 block1a_se_squeeze (Global  (None, 32)                   0         ['block1a_activation
[0][0]']
 AveragePooling2D)


 block1a_se_reshape (Reshap  (None, 1, 1, 32)             0         ['block1a_se_squeeze
[0][0]']
 e)


 block1a_se_reduce (Conv2D)  (None, 1, 1, 8)              264       ['block1a_se_reshape
[0][0]']


 block1a_se_expand (Conv2D)  (None, 1, 1, 32)             288       ['block1a_se_reduce
[0][0]']
```

```
 block1a_se_excite (Multipl  (None, None, None, 32)       0          ['block1a_activation
[0][0]',
 y)                                                                   'block1a_se_expand
[0][0]']

 block1a_project_conv (Conv  (None, None, None, 16)       512        ['block1a_se_excite
[0][0]']
 2D)


 block1a_project_bn (BatchN  (None, None, None, 16)       64         ['block1a_project_co
nv[0][0]']
 ormalization)


 block2a_expand_conv (Conv2  (None, None, None, 96)       1536       ['block1a_project_bn
[0][0]']
 D)


 block2a_expand_bn (BatchNo  (None, None, None, 96)       384        ['block2a_expand_con
v[0][0]']
 rmalization)


 block2a_expand_activation   (None, None, None, 96)       0          ['block2a_expand_bn
[0][0]']
 (Activation)


 block2a_dwconv_pad (ZeroPa  (None, None, None, 96)       0          ['block2a_expand_act
ivation[0]
 dding2D)                                                            [0]']


 block2a_dwconv (DepthwiseC  (None, None, None, 96)       864        ['block2a_dwconv_pad
[0][0]']
 onv2D)


 block2a_bn (BatchNormaliza  (None, None, None, 96)       384        ['block2a_dwconv[0]
[0]']
 tion)


 block2a_activation (Activa  (None, None, None, 96)       0          ['block2a_bn[0][0]']

 tion)


 block2a_se_squeeze (Global  (None, 96)                   0          ['block2a_activation
[0][0]']
 AveragePooling2D)
 block2a_se_reshape (Reshap  (None, 1, 1, 96)             0          ['block2a_se_squeeze
[0][0]']
 e)



 block2a_se_reduce (Conv2D)  (None, 1, 1, 4)              388        ['block2a_se_reshape
[0][0]']


 block2a_se_expand (Conv2D)  (None, 1, 1, 96)             480        ['block2a_se_reduce
[0][0]']


 block2a_se_excite (Multipl  (None, None, None, 96)       0          ['block2a_activation
[0][0]',
 y)                                                                   'block2a_se_expand
[0][0]']


 block2a_project_conv (Conv  (None, None, None, 24)       2304       ['block2a_se_excite
[0][0]']
 2D)



 block2a_project_bn (BatchN  (None, None, None, 24)       96         ['block2a_project_co
nv[0][0]']
 ormalization)



 block2b_expand_conv (Conv2  (None, None, None, 144)      3456       ['block2a_project_bn
[0][0]']
 D)



 block2b_expand_bn (BatchNo  (None, None, None, 144)      576        ['block2b_expand_con
v[0][0]']
 rmalization)



 block2b_expand_activation   (None, None, None, 144)      0          ['block2b_expand_bn
[0][0]']
 (Activation)



 block2b_dwconv (DepthwiseC  (None, None, None, 144)      1296       ['block2b_expand_act
ivation[0]
 onv2D)                                                              [0]']



 block2b_bn (BatchNormaliza  (None, None, None, 144)      576        ['block2b_dwconv[0]
[0]']
 tion)



 block2b_activation (Activa  (None, None, None, 144)      0          ['block2b_bn[0][0]']
 tion)


 block2b_se_squeeze (Global  (None, 144)                  0         ['block2b_activation
[0][0]']
 AveragePooling2D)


 block2b_se_reshape (Reshap  (None, 1, 1, 144)            0         ['block2b_se_squeeze
[0][0]']
 e)


 block2b_se_reduce (Conv2D)  (None, 1, 1, 6)              870       ['block2b_se_reshape
[0][0]']

 block2b_se_expand (Conv2D)  (None, 1, 1, 144)            1008      ['block2b_se_reduce
[0][0]']

 block2b_se_excite (Multipl  (None, None, None, 144)      0         ['block2b_activation
[0][0]',
 y)                                                                  'block2b_se_expand
[0][0]']

 block2b_project_conv (Conv  (None, None, None, 24)       3456      ['block2b_se_excite
[0][0]']
 2D)


 block2b_project_bn (BatchN  (None, None, None, 24)       96        ['block2b_project_co
nv[0][0]']
 ormalization)


 block2b_drop (Dropout)      (None, None, None, 24)       0         ['block2b_project_bn
[0][0]']

 block2b_add (Add)           (None, None, None, 24)       0         ['block2b_drop[0]
[0]',
                                                                     'block2a_project_bn

[0][0]']

 block3a_expand_conv (Conv2  (None, None, None, 144)      3456      ['block2b_add[0]
[0]']
 D)


 block3a_expand_bn (BatchNo  (None, None, None, 144)      576       ['block3a_expand_con
v[0][0]']
 rmalization)


 block3a_expand_activation   (None, None, None, 144)      0         ['block3a_expand_bn
[0][0]']
```

```
 (Activation)



 block3a_dwconv_pad (ZeroPa  (None, None, None, 144)      0          ['block3a_expand_act
ivation[0]
 dding2D)                                                             [0]']



 block3a_dwconv (DepthwiseC  (None, None, None, 144)      3600       ['block3a_dwconv_pad
[0][0]']
 onv2D)



 block3a_bn (BatchNormaliza  (None, None, None, 144)      576        ['block3a_dwconv[0]
[0]']
 tion)



 block3a_activation (Activa  (None, None, None, 144)      0          ['block3a_bn[0][0]']

 tion)



 block3a_se_squeeze (Global  (None, 144)                  0          ['block3a_activation
[0][0]']
 AveragePooling2D)



 block3a_se_reshape (Reshap  (None, 1, 1, 144)            0          ['block3a_se_squeeze
[0][0]']
 e)



 block3a_se_reduce (Conv2D)  (None, 1, 1, 6)              870        ['block3a_se_reshape
[0][0]']


 block3a_se_expand (Conv2D)  (None, 1, 1, 144)            1008       ['block3a_se_reduce
[0][0]']


 block3a_se_excite (Multipl  (None, None, None, 144)      0          ['block3a_activation
[0][0]',
 y)                                                                   'block3a_se_expand
[0][0]']


 block3a_project_conv (Conv  (None, None, None, 40)       5760       ['block3a_se_excite
[0][0]']
 2D)



 block3a_project_bn (BatchN  (None, None, None, 40)       160        ['block3a_project_co
nv[0][0]']
 ormalization)
 block3b_expand_conv (Conv2  (None, None, None, 240)      9600      ['block3a_project_bn
[0][0]']
 D)



 block3b_expand_bn (BatchNo  (None, None, None, 240)      960       ['block3b_expand_con
v[0][0]']
 rmalization)



 block3b_expand_activation   (None, None, None, 240)      0         ['block3b_expand_bn
[0][0]']
 (Activation)



 block3b_dwconv (DepthwiseC  (None, None, None, 240)      6000      ['block3b_expand_act
ivation[0
 onv2D)                                                              [0]']



 block3b_bn (BatchNormaliza  (None, None, None, 240)      960       ['block3b_dwconv[0]
[0]']
 tion)



 block3b_activation (Activa  (None, None, None, 240)      0         ['block3b_bn[0][0]']

 tion)



 block3b_se_squeeze (Global  (None, 240)                  0         ['block3b_activation
[0][0]']
 AveragePooling2D)



 block3b_se_reshape (Reshap  (None, 1, 1, 240)            0         ['block3b_se_squeeze
[0][0]']
 e)



 block3b_se_reduce (Conv2D)  (None, 1, 1, 10)             2410      ['block3b_se_reshape
[0][0]']


 block3b_se_expand (Conv2D)  (None, 1, 1, 240)            2640      ['block3b_se_reduce
[0][0]']


 block3b_se_excite (Multipl  (None, None, None, 240)      0         ['block3b_activation
[0][0]',
 y)                                                                  'block3b_se_expand
[0][0]']


 block3b_project_conv (Conv  (None, None, None, 40)       9600      ['block3b_se_excite
[0][0]']
 2D)
 block3b_project_bn (BatchN  (None, None, None, 40)       160       ['block3b_project_co
nv[0][0]']
 ormalization)



 block3b_drop (Dropout)      (None, None, None, 40)       0         ['block3b_project_bn
[0][0]']


 block3b_add (Add)           (None, None, None, 40)       0         ['block3b_drop[0]
[0]',
                                                                     'block3a_project_bn
[0][0]']


 block4a_expand_conv (Conv2  (None, None, None, 240)      9600      ['block3b_add[0]
[0]']
 D)



 block4a_expand_bn (BatchNo  (None, None, None, 240)      960       ['block4a_expand_con
v[0][0]']
 rmalization)



 block4a_expand_activation   (None, None, None, 240)      0         ['block4a_expand_bn
[0][0]']
 (Activation)



 block4a_dwconv_pad (ZeroPa  (None, None, None, 240)      0         ['block4a_expand_act
ivation[0]
 dding2D)                                                           [0]']



 block4a_dwconv (DepthwiseC  (None, None, None, 240)      2160      ['block4a_dwconv_pad
[0][0]']
 onv2D)



 block4a_bn (BatchNormaliza  (None, None, None, 240)      960       ['block4a_dwconv[0]
[0]']
 tion)



 block4a_activation (Activa  (None, None, None, 240)      0         ['block4a_bn[0][0]']

 tion)



 block4a_se_squeeze (Global  (None, 240)                  0         ['block4a_activation
[0][0]']
 AveragePooling2D)
 block4a_se_reshape (Reshap  (None, 1, 1, 240)            0          ['block4a_se_squeeze
[0][0]']
 e)



 block4a_se_reduce (Conv2D)  (None, 1, 1, 10)             2410       ['block4a_se_reshape
[0][0]']


 block4a_se_expand (Conv2D)  (None, 1, 1, 240)            2640       ['block4a_se_reduce
[0][0]']


 block4a_se_excite (Multipl  (None, None, None, 240)      0          ['block4a_activation
[0][0]',
 y)                                                                   'block4a_se_expand
[0][0]']


 block4a_project_conv (Conv  (None, None, None, 80)       19200      ['block4a_se_excite
[0][0]']
 2D)



 block4a_project_bn (BatchN  (None, None, None, 80)       320        ['block4a_project_co
nv[0][0]']
 ormalization)



 block4b_expand_conv (Conv2  (None, None, None, 480)      38400      ['block4a_project_bn
[0][0]']
 D)



 block4b_expand_bn (BatchNo  (None, None, None, 480)      1920       ['block4b_expand_con
v[0][0]']
 rmalization)



 block4b_expand_activation   (None, None, None, 480)      0          ['block4b_expand_bn
[0][0]']
 (Activation)



 block4b_dwconv (DepthwiseC  (None, None, None, 480)      4320       ['block4b_expand_act
ivation[0]
 onv2D)                                                              [0]']



 block4b_bn (BatchNormaliza  (None, None, None, 480)      1920       ['block4b_dwconv[0]
[0]']
 tion)



 block4b_activation (Activa  (None, None, None, 480)      0          ['block4b_bn[0][0]']

 tion)
 block4b_se_squeeze (Global  (None, 480)                 0          ['block4b_activation
[0][0]']
 AveragePooling2D)



 block4b_se_reshape (Reshap  (None, 1, 1, 480)           0          ['block4b_se_squeeze
[0][0]']
 e)



 block4b_se_reduce (Conv2D)  (None, 1, 1, 20)            9620       ['block4b_se_reshape
[0][0]']


 block4b_se_expand (Conv2D)  (None, 1, 1, 480)           10080      ['block4b_se_reduce
[0][0]']


 block4b_se_excite (Multipl  (None, None, None, 480)     0          ['block4b_activation
[0][0]',
 y)                                                                  'block4b_se_expand
[0][0]']


 block4b_project_conv (Conv  (None, None, None, 80)      38400      ['block4b_se_excite
[0][0]']
 2D)



 block4b_project_bn (BatchN  (None, None, None, 80)      320        ['block4b_project_co
nv[0][0]']
 ormalization)



 block4b_drop (Dropout)      (None, None, None, 80)      0          ['block4b_project_bn
[0][0]']


 block4b_add (Add)           (None, None, None, 80)      0          ['block4b_drop[0]
[0]',
                                                                     'block4a_project_bn

[0][0]']


 block4c_expand_conv (Conv2  (None, None, None, 480)     38400      ['block4b_add[0]
[0]']
 D)



 block4c_expand_bn (BatchNo  (None, None, None, 480)     1920       ['block4c_expand_con
v[0][0]']
 rmalization)



 block4c_expand_activation   (None, None, None, 480)     0          ['block4c_expand_bn
[0][0]']
 (Activation)
 block4c_dwconv (DepthwiseC  (None, None, None, 480)      4320      ['block4c_expand_act
ivation[0]
 onv2D)                                                             [0]']


 block4c_bn (BatchNormaliza  (None, None, None, 480)      1920      ['block4c_dwconv[0]
[0]']
 tion)


 block4c_activation (Activa  (None, None, None, 480)      0         ['block4c_bn[0][0]']

 tion)


 block4c_se_squeeze (Global  (None, 480)                  0         ['block4c_activation
[0][0]']
 AveragePooling2D)


 block4c_se_reshape (Reshap  (None, 1, 1, 480)            0         ['block4c_se_squeeze
[0][0]']
 e)


 block4c_se_reduce (Conv2D)  (None, 1, 1, 20)             9620      ['block4c_se_reshape
[0][0]']

 block4c_se_expand (Conv2D)  (None, 1, 1, 480)            10080     ['block4c_se_reduce
[0][0]']

 block4c_se_excite (Multipl  (None, None, None, 480)      0         ['block4c_activation
[0][0]',
 y)                                                                  'block4c_se_expand
[0][0]']

 block4c_project_conv (Conv  (None, None, None, 80)       38400     ['block4c_se_excite
[0][0]']
 2D)


 block4c_project_bn (BatchN  (None, None, None, 80)       320       ['block4c_project_co
nv[0][0]']
 ormalization)


 block4c_drop (Dropout)      (None, None, None, 80)       0         ['block4c_project_bn
[0][0]']

 block4c_add (Add)           (None, None, None, 80)       0         ['block4c_drop[0]
[0]',
                                                                     'block4b_add[0]

[0]']
```

```
 block5a_expand_conv (Conv2  (None, None, None, 480)      38400     ['block4c_add[0]
[0]']
 D)



 block5a_expand_bn (BatchNo  (None, None, None, 480)      1920      ['block5a_expand_con
v[0][0]']
 rmalization)



 block5a_expand_activation   (None, None, None, 480)      0         ['block5a_expand_bn
[0][0]']
 (Activation)



 block5a_dwconv (DepthwiseC  (None, None, None, 480)      12000     ['block5a_expand_act
ivation[0]
 onv2D)                                                             [0]']



 block5a_bn (BatchNormaliza  (None, None, None, 480)      1920      ['block5a_dwconv[0]
[0]']
 tion)



 block5a_activation (Activa  (None, None, None, 480)      0         ['block5a_bn[0][0]']

 tion)



 block5a_se_squeeze (Global  (None, 480)                  0         ['block5a_activation
[0][0]']
 AveragePooling2D)



 block5a_se_reshape (Reshap  (None, 1, 1, 480)            0         ['block5a_se_squeeze
[0][0]']
 e)



 block5a_se_reduce (Conv2D)  (None, 1, 1, 20)             9620      ['block5a_se_reshape
[0][0]']

 block5a_se_expand (Conv2D)  (None, 1, 1, 480)            10080     ['block5a_se_reduce
[0][0]']

 block5a_se_excite (Multipl  (None, None, None, 480)      0         ['block5a_activation
[0][0]',
 y)                                                                  'block5a_se_expand
[0][0]']

 block5a_project_conv (Conv  (None, None, None, 112)      53760     ['block5a_se_excite
[0][0]']
```

```
 2D)



 block5a_project_bn (BatchN  (None, None, None, 112)      448       ['block5a_project_co
nv[0][0]']
 ormalization)



 block5b_expand_conv (Conv2  (None, None, None, 672)      75264     ['block5a_project_bn
[0][0]']
 D)



 block5b_expand_bn (BatchNo  (None, None, None, 672)      2688      ['block5b_expand_con
v[0][0]']
 rmalization)



 block5b_expand_activation   (None, None, None, 672)      0         ['block5b_expand_bn
[0][0]']
 (Activation)



 block5b_dwconv (DepthwiseC  (None, None, None, 672)      16800     ['block5b_expand_act
ivation[0
 onv2D)                                                              [0]']



 block5b_bn (BatchNormaliza  (None, None, None, 672)      2688      ['block5b_dwconv[0]
[0]']
 tion)



 block5b_activation (Activa  (None, None, None, 672)      0         ['block5b_bn[0][0]']

 tion)



 block5b_se_squeeze (Global  (None, 672)                  0         ['block5b_activation
[0][0]']
 AveragePooling2D)



 block5b_se_reshape (Reshap  (None, 1, 1, 672)            0         ['block5b_se_squeeze
[0][0]']
 e)



 block5b_se_reduce (Conv2D)  (None, 1, 1, 28)             18844     ['block5b_se_reshape
[0][0]']

 block5b_se_expand (Conv2D)  (None, 1, 1, 672)            19488     ['block5b_se_reduce
[0][0]']
```

```
 block5b_se_excite (Multipl  (None, None, None, 672)      0         ['block5b_activation
[0][0]',
 y)                                                                  'block5b_se_expand
[0][0]']


 block5b_project_conv (Conv  (None, None, None, 112)      75264     ['block5b_se_excite
[0][0]']
 2D)



 block5b_project_bn (BatchN  (None, None, None, 112)      448       ['block5b_project_co
nv[0][0]']
 ormalization)



 block5b_drop (Dropout)      (None, None, None, 112)      0         ['block5b_project_bn
[0][0]']


 block5b_add (Add)           (None, None, None, 112)      0         ['block5b_drop[0]
[0]',
                                                                     'block5a_project_bn

[0][0]']


 block5c_expand_conv (Conv2  (None, None, None, 672)      75264     ['block5b_add[0]
[0]']
 D)



 block5c_expand_bn (BatchNo  (None, None, None, 672)      2688      ['block5c_expand_con
v[0][0]']
 rmalization)



 block5c_expand_activation   (None, None, None, 672)      0         ['block5c_expand_bn
[0][0]']
 (Activation)



 block5c_dwconv (DepthwiseC  (None, None, None, 672)      16800     ['block5c_expand_act
ivation[0]
 onv2D)                                                             [0]']



 block5c_bn (BatchNormaliza  (None, None, None, 672)      2688      ['block5c_dwconv[0]
[0]']
 tion)



 block5c_activation (Activa  (None, None, None, 672)      0         ['block5c_bn[0][0]']

 tion)



 block5c_se_squeeze (Global  (None, 672)                  0         ['block5c_activation
[0][0]']
```

```
 AveragePooling2D)



 block5c_se_reshape (Reshap  (None, 1, 1, 672)            0          ['block5c_se_squeeze
[0][0]']
 e)



 block5c_se_reduce (Conv2D)  (None, 1, 1, 28)             18844      ['block5c_se_reshape
[0][0]']


 block5c_se_expand (Conv2D)  (None, 1, 1, 672)            19488      ['block5c_se_reduce
[0][0]']


 block5c_se_excite (Multipl  (None, None, None, 672)      0          ['block5c_activation
[0][0]',
 y)                                                                   'block5c_se_expand
[0][0]']


 block5c_project_conv (Conv  (None, None, None, 112)      75264      ['block5c_se_excite
[0][0]']
 2D)



 block5c_project_bn (BatchN  (None, None, None, 112)      448        ['block5c_project_co
nv[0][0]']
 ormalization)



 block5c_drop (Dropout)      (None, None, None, 112)      0          ['block5c_project_bn
[0][0]']


 block5c_add (Add)           (None, None, None, 112)      0          ['block5c_drop[0]
[0]',
                                                                      'block5b_add[0]
[0]']


 block6a_expand_conv (Conv2  (None, None, None, 672)      75264      ['block5c_add[0]
[0]']
 D)



 block6a_expand_bn (BatchNo  (None, None, None, 672)      2688       ['block6a_expand_con
v[0][0]']
 rmalization)



 block6a_expand_activation   (None, None, None, 672)      0          ['block6a_expand_bn
[0][0]']
 (Activation)



 block6a_dwconv_pad (ZeroPa  (None, None, None, 672)      0          ['block6a_expand_act
ivation[0]
```

```
 dding2D)                                                           [0]']


 block6a_dwconv (DepthwiseC  (None, None, None, 672)      16800     ['block6a_dwconv_pad
[0][0]']
 onv2D)


 block6a_bn (BatchNormaliza  (None, None, None, 672)      2688      ['block6a_dwconv[0]
[0]']
 tion)


 block6a_activation (Activa  (None, None, None, 672)      0         ['block6a_bn[0][0]']

 tion)


 block6a_se_squeeze (Global  (None, 672)                  0         ['block6a_activation
[0][0]']
 AveragePooling2D)


 block6a_se_reshape (Reshap  (None, 1, 1, 672)            0         ['block6a_se_squeeze
[0][0]']
 e)


 block6a_se_reduce (Conv2D)  (None, 1, 1, 28)             18844     ['block6a_se_reshape
[0][0]']

 block6a_se_expand (Conv2D)  (None, 1, 1, 672)            19488     ['block6a_se_reduce
[0][0]']

 block6a_se_excite (Multipl  (None, None, None, 672)      0         ['block6a_activation
[0][0]',
 y)                                                                  'block6a_se_expand
[0][0]']

 block6a_project_conv (Conv  (None, None, None, 192)      129024    ['block6a_se_excite
[0][0]']
 2D)


 block6a_project_bn (BatchN  (None, None, None, 192)      768       ['block6a_project_co
nv[0][0]']
 ormalization)


 block6b_expand_conv (Conv2  (None, None, None, 1152)     221184    ['block6a_project_bn
[0][0]']
 D)
 block6b_expand_bn (BatchNo  (None, None, None, 1152)     4608      ['block6b_expand_con
v[0][0]']
 rmalization)



 block6b_expand_activation   (None, None, None, 1152)     0         ['block6b_expand_bn
[0][0]']
 (Activation)



 block6b_dwconv (DepthwiseC  (None, None, None, 1152)     28800     ['block6b_expand_act
ivation[0]
 onv2D)                                                             [0]']



 block6b_bn (BatchNormaliza  (None, None, None, 1152)     4608      ['block6b_dwconv[0]
[0]']
 tion)



 block6b_activation (Activa  (None, None, None, 1152)     0         ['block6b_bn[0][0]']

 tion)



 block6b_se_squeeze (Global  (None, 1152)                 0         ['block6b_activation
[0][0]']
 AveragePooling2D)



 block6b_se_reshape (Reshap  (None, 1, 1, 1152)           0         ['block6b_se_squeeze
[0][0]']
 e)



 block6b_se_reduce (Conv2D)  (None, 1, 1, 48)             55344     ['block6b_se_reshape
[0][0]']


 block6b_se_expand (Conv2D)  (None, 1, 1, 1152)           56448     ['block6b_se_reduce
[0][0]']


 block6b_se_excite (Multipl  (None, None, None, 1152)     0         ['block6b_activation
[0][0]',
 y)                                                                  'block6b_se_expand
[0][0]']


 block6b_project_conv (Conv  (None, None, None, 192)      221184    ['block6b_se_excite
[0][0]']
 2D)



 block6b_project_bn (BatchN  (None, None, None, 192)      768       ['block6b_project_co
nv[0][0]']
 ormalization)
 block6b_drop (Dropout)      (None, None, None, 192)      0         ['block6b_project_bn
[0][0]']


 block6b_add (Add)           (None, None, None, 192)      0         ['block6b_drop[0]
[0]',

                                                                     'block6a_project_bn
[0][0]']


 block6c_expand_conv (Conv2  (None, None, None, 1152)     221184    ['block6b_add[0]
[0]']
 D)



 block6c_expand_bn (BatchNo  (None, None, None, 1152)     4608      ['block6c_expand_con
v[0][0]']
 rmalization)



 block6c_expand_activation   (None, None, None, 1152)     0         ['block6c_expand_bn
[0][0]']
 (Activation)



 block6c_dwconv (DepthwiseC  (None, None, None, 1152)     28800     ['block6c_expand_act
ivation[0]
 onv2D)                                                             [0]']



 block6c_bn (BatchNormaliza  (None, None, None, 1152)     4608      ['block6c_dwconv[0]
[0]']
 tion)



 block6c_activation (Activa  (None, None, None, 1152)     0         ['block6c_bn[0][0]']

 tion)



 block6c_se_squeeze (Global  (None, 1152)                 0         ['block6c_activation
[0][0]']
 AveragePooling2D)



 block6c_se_reshape (Reshap  (None, 1, 1, 1152)           0         ['block6c_se_squeeze
[0][0]']
 e)



 block6c_se_reduce (Conv2D)  (None, 1, 1, 48)             55344     ['block6c_se_reshape
[0][0]']


 block6c_se_expand (Conv2D)  (None, 1, 1, 1152)           56448     ['block6c_se_reduce
[0][0]']
```

```
 block6c_se_excite (Multipl  (None, None, None, 1152)     0          ['block6c_activation
[0][0]',
 y)                                                                   'block6c_se_expand
[0][0]']


 block6c_project_conv (Conv  (None, None, None, 192)      221184     ['block6c_se_excite
[0][0]']
 2D)



 block6c_project_bn (BatchN  (None, None, None, 192)      768        ['block6c_project_co
nv[0][0]']
 ormalization)



 block6c_drop (Dropout)      (None, None, None, 192)      0          ['block6c_project_bn
[0][0]']


 block6c_add (Add)           (None, None, None, 192)      0          ['block6c_drop[0]
[0]',
                                                                      'block6b_add[0]
[0]']


 block6d_expand_conv (Conv2  (None, None, None, 1152)     221184     ['block6c_add[0]
[0]']
 D)



 block6d_expand_bn (BatchNo  (None, None, None, 1152)     4608       ['block6d_expand_con
v[0][0]']
 rmalization)



 block6d_expand_activation   (None, None, None, 1152)     0          ['block6d_expand_bn
[0][0]']
 (Activation)



 block6d_dwconv (DepthwiseC  (None, None, None, 1152)     28800      ['block6d_expand_act
ivation[0]
 onv2D)                                                              [0]']



 block6d_bn (BatchNormaliza  (None, None, None, 1152)     4608       ['block6d_dwconv[0]
[0]']
 tion)



 block6d_activation (Activa  (None, None, None, 1152)     0          ['block6d_bn[0][0]']

 tion)
 block6d_se_squeeze (Global  (None, 1152)                 0         ['block6d_activation
[0][0]']
 AveragePooling2D)



 block6d_se_reshape (Reshap  (None, 1, 1, 1152)           0         ['block6d_se_squeeze
[0][0]']
 e)



 block6d_se_reduce (Conv2D)  (None, 1, 1, 48)             55344     ['block6d_se_reshape
[0][0]']


 block6d_se_expand (Conv2D)  (None, 1, 1, 1152)           56448     ['block6d_se_reduce
[0][0]']


 block6d_se_excite (Multipl  (None, None, None, 1152)     0         ['block6d_activation
[0][0]',
 y)                                                                  'block6d_se_expand
[0][0]']


 block6d_project_conv (Conv  (None, None, None, 192)      221184    ['block6d_se_excite
[0][0]']
 2D)



 block6d_project_bn (BatchN  (None, None, None, 192)      768       ['block6d_project_co
nv[0][0]']
 ormalization)



 block6d_drop (Dropout)      (None, None, None, 192)      0         ['block6d_project_bn
[0][0]']


 block6d_add (Add)           (None, None, None, 192)      0         ['block6d_drop[0]
[0]',
                                                                     'block6c_add[0]
[0]']


 block7a_expand_conv (Conv2  (None, None, None, 1152)     221184    ['block6d_add[0]
[0]']
 D)



 block7a_expand_bn (BatchNo  (None, None, None, 1152)     4608      ['block7a_expand_con
v[0][0]']
 rmalization)



 block7a_expand_activation   (None, None, None, 1152)     0         ['block7a_expand_bn
[0][0]']
 (Activation)
 block7a_dwconv (DepthwiseC  (None, None, None, 1152)     10368      ['block7a_expand_act
ivation[0]
 onv2D)                                                              [0]']



 block7a_bn (BatchNormaliza  (None, None, None, 1152)     4608       ['block7a_dwconv[0]
[0]']
 tion)



 block7a_activation (Activa  (None, None, None, 1152)     0          ['block7a_bn[0][0]']

 tion)



 block7a_se_squeeze (Global  (None, 1152)                 0          ['block7a_activation
[0][0]']
 AveragePooling2D)



 block7a_se_reshape (Reshap  (None, 1, 1, 1152)           0          ['block7a_se_squeeze
[0][0]']
 e)



 block7a_se_reduce (Conv2D)  (None, 1, 1, 48)             55344      ['block7a_se_reshape
[0][0]']


 block7a_se_expand (Conv2D)  (None, 1, 1, 1152)           56448      ['block7a_se_reduce
[0][0]']


 block7a_se_excite (Multipl  (None, None, None, 1152)     0          ['block7a_activation
[0][0]',
 y)                                                                   'block7a_se_expand
[0][0]']


 block7a_project_conv (Conv  (None, None, None, 320)      368640     ['block7a_se_excite
[0][0]']
 2D)



 block7a_project_bn (BatchN  (None, None, None, 320)      1280       ['block7a_project_co
nv[0][0]']
 ormalization)



 top_conv (Conv2D)           (None, None, None, 1280)     409600     ['block7a_project_bn
[0][0]']


 top_bn (BatchNormalization  (None, None, None, 1280)     5120       ['top_conv[0][0]']

 )
 top_activation (Activation  (None, None, None, 1280)     0         ['top_bn[0][0]']

 )



========================================================================================
==========
Total params: 4049571 (15.45 MB)
Trainable params: 1350960 (5.15 MB)
Non-trainable params: 2698611 (10.29 MB)
________________________________________________________________________________________
__________
```

Step 3: Model Training

Create a function to set up TensorBoard logging

In [20]:

```
def create_tensorboard_callback(dir_name, experiment_name):
    log_dir = dir_name + "/" + experiment_name + "/" + datetime.datetime.now().strftime(
    tensorboard_callback = tf.keras.callbacks.TensorBoard(
        log_dir=log_dir
    )
    print(f"Saving TensorBoard log files to: {log_dir}")
    return tensorboard_callback
```

Set up callbacks for training

In [21]:

```
early_stopping = tf.keras.callbacks.EarlyStopping(monitor="val_loss", patience=3)
reduce_lr = tf.keras.callbacks.ReduceLROnPlateau(monitor="val_loss", factor=0.2, patienc
checkpoint_path = "fine_tune_checkpoints/"
model_checkpoint = tf.keras.callbacks.ModelCheckpoint(
    checkpoint_path,
    save_weights_only=True,
    save_best_only=True,
    monitor="val_loss"
)
```

Train the model with early stopping, learning rate reduction, and checkpointing

In [22]:

```
epochs = 10
history = feature_model.fit(train_data, epochs=epochs,
                            steps_per_epoch=len(train_data),
                            validation_data=test_data,
                            validation_steps=len(test_data),
                            callbacks=[early_stopping, model_checkpoint, reduce_lr,
                                       create_tensorboard_callback('Fruit_Vegetable_Dise
)
```

```
Saving TensorBoard log files to: Fruit_Vegetable_Diseases_Detection_Model/EfficientNetB0
10/20231115-180057
Epoch 1/10
WARNING:tensorflow:From C:\Users\PMLS\anaconda3\Lib\site-packages\keras\src\utils\tf_uti
ls.py:492: The name tf.ragged.RaggedTensorValue is deprecated. Please use tf.compat.v1.r
agged.RaggedTensorValue instead.

WARNING:tensorflow:From C:\Users\PMLS\anaconda3\Lib\site-packages\keras\src\engine\base_
layer_utils.py:384: The name tf.executing_eagerly_outside_functions is deprecated. Pleas
e use tf.compat.v1.executing_eagerly_outside_functions instead.

1467/1467 [==============================] - 1524s 1s/step - loss: 0.2617 - accuracy: 0.
```

```
9256 - val_loss: 0.0801 - val_accuracy: 0.9741 - lr: 1.0000e-04
Epoch 2/10
1467/1467 [==============================] - 1520s 1s/step - loss: 0.0423 - accuracy: 0.
9873 - val_loss: 0.0436 - val_accuracy: 0.9864 - lr: 1.0000e-04
Epoch 3/10
1467/1467 [==============================] - 1543s 1s/step - loss: 0.0186 - accuracy: 0.
9946 - val_loss: 0.0339 - val_accuracy: 0.9886 - lr: 1.0000e-04
Epoch 4/10
1467/1467 [==============================] - 1498s 1s/step - loss: 0.0099 - accuracy: 0.
9974 - val_loss: 0.0392 - val_accuracy: 0.9882 - lr: 1.0000e-04
Epoch 5/10
1467/1467 [==============================] - 1469s 1s/step - loss: 0.0065 - accuracy: 0.
9983 - val_loss: 0.0313 - val_accuracy: 0.9906 - lr: 1.0000e-04
Epoch 6/10
1467/1467 [==============================] - 1462s 996ms/step - loss: 0.0061 - accuracy:
0.9984 - val_loss: 0.0356 - val_accuracy: 0.9886 - lr: 1.0000e-04
Epoch 7/10
1467/1467 [==============================] - 1486s 1s/step - loss: 0.0051 - accuracy: 0.
9986 - val_loss: 0.0236 - val_accuracy: 0.9935 - lr: 1.0000e-04
Epoch 8/10
1467/1467 [==============================] - 1479s 1s/step - loss: 0.0047 - accuracy: 0.
9986 - val_loss: 0.0273 - val_accuracy: 0.9927 - lr: 1.0000e-04
Epoch 9/10
1467/1467 [==============================] - ETA: 0s - loss: 0.0020 - accuracy: 0.9995
Epoch 9: ReduceLROnPlateau reducing learning rate to 1.9999999494757503e-05.
1467/1467 [==============================] - 1467s 1s/step - loss: 0.0020 - accuracy: 0.
9995 - val_loss: 0.0282 - val_accuracy: 0.9911 - lr: 1.0000e-04
Epoch 10/10
1467/1467 [==============================] - 1470s 1s/step - loss: 6.9776e-04 - accurac
y: 0.9999 - val_loss: 0.0189 - val_accuracy: 0.9946 - lr: 2.0000e-05
```

Step 4: Model Evaluation

Load the best model checkpoint

In [23]:

```
feature_model.load_weights(checkpoint_path)
```

```
WARNING:tensorflow:From C:\Users\PMLS\anaconda3\Lib\site-packages\keras\src\saving\legac
y\save.py:538: The name tf.train.NewCheckpointReader is deprecated. Please use tf.compa
t.v1.train.NewCheckpointReader instead.
```

Out[23]:

```
<tensorflow.python.checkpoint.checkpoint.CheckpointLoadStatus at 0x26801769290>
```

Evaluate the model on the test data

In [24]:

```
test_loss, test_accuracy = feature_model.evaluate(test_data)
```

```
367/367 [==============================] - 270s 735ms/step - loss: 0.0189 - accuracy: 0.
9946
```

Print the evaluation results

In [25]:

```
print(f"Test Loss: {test_loss:.2f}")
print(f"Test Accuracy: {test_accuracy * 100:.2f}%")
```

```
Test Loss: 0.02
Test Accuracy: 99.46%
```

Step 5: Data Visualization and Model Metrics

Define a function to plot training history

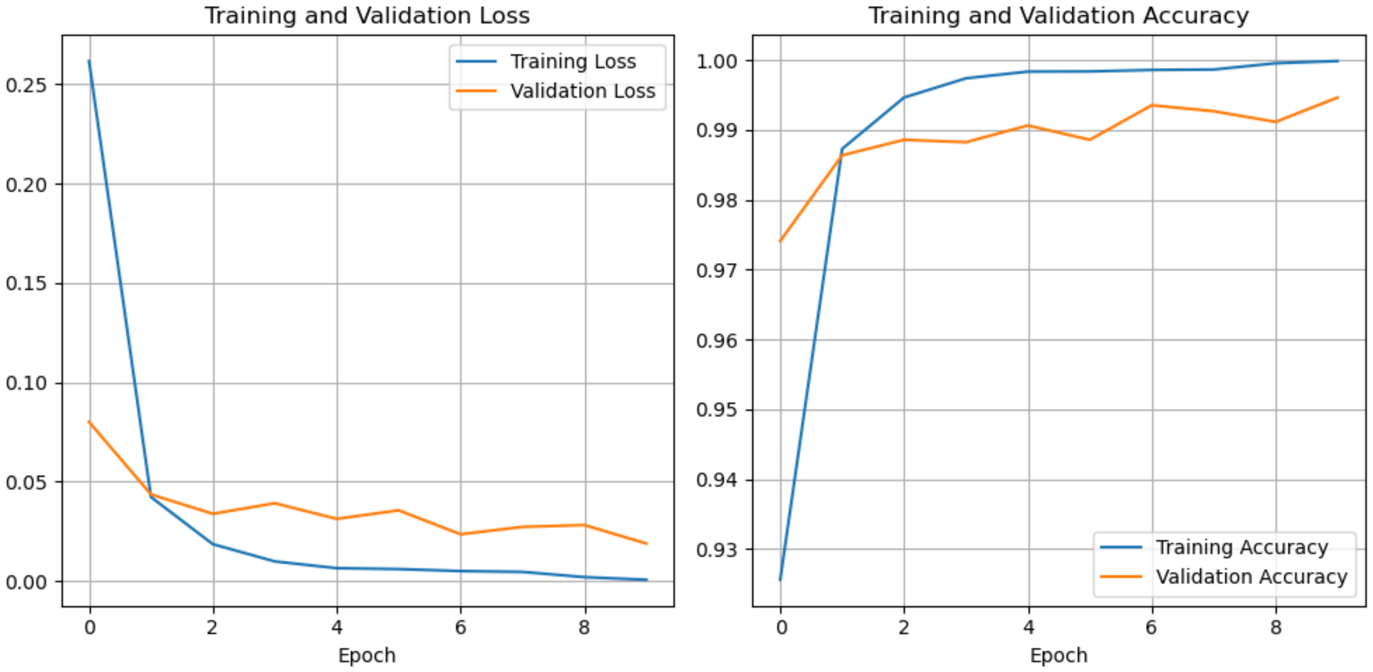
In [26]:

```
def plot_history(history):
    loss = history.history['loss']
    val_loss = history.history['val_loss']
    epochs = history.epoch
    acc = history.history['accuracy']
    val_acc = history.history['val_accuracy']

    plt.figure(figsize=(10, 5))
    plt.subplot(1, 2, 1)
    plt.plot(epochs, loss, label='Training Loss')
    plt.plot(epochs, val_loss, label='Validation Loss')
    plt.title('Training and Validation Loss')
    plt.xlabel('Epoch')
    plt.legend()
    plt.grid(True)

    plt.subplot(1, 2, 2)
    plt.plot(epochs, acc, label='Training Accuracy')
    plt.plot(epochs, val_acc, label='Validation Accuracy')
    plt.title('Training and Validation Accuracy')
    plt.xlabel('Epoch')
    plt.legend()
    plt.grid(True)

    plt.tight_layout()
    plt.show()
```

Plot the training history

In [27]:

```
plot_history(history)
```



Calculate additional metrics for model evaluation

In [28]:

```
from sklearn.metrics import classification_report

def calculate_metrics(model, test_data):
    y_true = []
    y_pred = []
    for images, labels in test_data:
        y_true.extend(np.argmax(labels, axis=1))
        y_pred.extend(np.argmax(model.predict(images), axis=1))

    return y_true, y_pred
```

In [29]:

```
y_true, y_pred = calculate_metrics(feature_model, test_data)
```

```
1/1 [==============================] - 3s 3s/step
1/1 [==============================] - 1s 781ms/step
1/1 [==============================] - 1s 860ms/step
1/1 [==============================] - 1s 742ms/step
1/1 [==============================] - 1s 760ms/step
1/1 [==============================] - 1s 704ms/step
1/1 [==============================] - 1s 711ms/step
1/1 [==============================] - 1s 783ms/step
1/1 [==============================] - 1s 726ms/step
1/1 [==============================] - 1s 824ms/step
1/1 [==============================] - 1s 831ms/step
1/1 [==============================] - 1s 825ms/step
1/1 [==============================] - 1s 832ms/step
1/1 [==============================] - 1s 871ms/step
1/1 [==============================] - 1s 784ms/step
1/1 [==============================] - 1s 776ms/step
1/1 [==============================] - 1s 740ms/step
1/1 [==============================] - 1s 731ms/step
1/1 [==============================] - 1s 730ms/step
1/1 [==============================] - 1s 719ms/step
1/1 [==============================] - 1s 733ms/step
1/1 [==============================] - 1s 799ms/step
1/1 [==============================] - 1s 796ms/step
1/1 [==============================] - 1s 733ms/step
1/1 [==============================] - 1s 723ms/step
1/1 [==============================] - 1s 788ms/step
1/1 [==============================] - 1s 764ms/step
1/1 [==============================] - 1s 793ms/step
1/1 [==============================] - 1s 817ms/step
1/1 [==============================] - 1s 786ms/step
1/1 [==============================] - 1s 901ms/step
1/1 [==============================] - 1s 838ms/step
1/1 [==============================] - 1s 892ms/step
1/1 [==============================] - 1s 814ms/step
1/1 [==============================] - 1s 770ms/step
1/1 [==============================] - 1s 821ms/step
1/1 [==============================] - 1s 800ms/step
1/1 [==============================] - 1s 754ms/step
1/1 [==============================] - 1s 731ms/step
1/1 [==============================] - 1s 803ms/step
1/1 [==============================] - 1s 774ms/step
1/1 [==============================] - 1s 728ms/step
1/1 [==============================] - 1s 732ms/step
1/1 [==============================] - 1s 807ms/step
1/1 [==============================] - 1s 767ms/step
1/1 [==============================] - 1s 783ms/step
1/1 [==============================] - 1s 837ms/step
1/1 [==============================] - 1s 834ms/step
1/1 [==============================] - 1s 828ms/step
1/1 [==============================] - 1s 904ms/step
1/1 [==============================] - 1s 854ms/step
1/1 [==============================] - 1s 830ms/step
1/1 [==============================] - 1s 744ms/step
1/1 [==============================] - 1s 813ms/step
1/1 [==============================] - 1s 783ms/step
1/1 [==============================] - 1s 870ms/step
1/1 [==============================] - 1s 740ms/step
```

```
1/1 [==============================] - 1s 750ms/step
1/1 [==============================] - 1s 784ms/step
1/1 [==============================] - 1s 707ms/step
1/1 [==============================] - 1s 710ms/step
1/1 [==============================] - 1s 697ms/step
1/1 [==============================] - 1s 702ms/step
1/1 [==============================] - 1s 703ms/step
1/1 [==============================] - 1s 735ms/step
1/1 [==============================] - 1s 766ms/step
1/1 [==============================] - 1s 761ms/step
1/1 [==============================] - 1s 892ms/step
1/1 [==============================] - 1s 770ms/step
1/1 [==============================] - 1s 753ms/step
1/1 [==============================] - 1s 702ms/step
1/1 [==============================] - 1s 702ms/step
1/1 [==============================] - 1s 700ms/step
1/1 [==============================] - 1s 766ms/step
1/1 [==============================] - 1s 691ms/step
1/1 [==============================] - 1s 734ms/step
1/1 [==============================] - 1s 716ms/step
1/1 [==============================] - 1s 696ms/step
1/1 [==============================] - 1s 725ms/step
1/1 [==============================] - 1s 694ms/step
1/1 [==============================] - 1s 701ms/step
1/1 [==============================] - 1s 775ms/step
1/1 [==============================] - 1s 691ms/step
1/1 [==============================] - 1s 703ms/step
1/1 [==============================] - 1s 772ms/step
1/1 [==============================] - 1s 791ms/step
1/1 [==============================] - 1s 794ms/step
1/1 [==============================] - 1s 791ms/step
1/1 [==============================] - 1s 790ms/step
1/1 [==============================] - 1s 834ms/step
1/1 [==============================] - 1s 722ms/step
1/1 [==============================] - 1s 703ms/step
1/1 [==============================] - 1s 772ms/step
1/1 [==============================] - 1s 700ms/step
1/1 [==============================] - 1s 702ms/step
1/1 [==============================] - 1s 702ms/step
1/1 [==============================] - 1s 707ms/step
1/1 [==============================] - 1s 712ms/step
1/1 [==============================] - 1s 700ms/step
1/1 [==============================] - 1s 705ms/step
1/1 [==============================] - 1s 711ms/step
1/1 [==============================] - 1s 697ms/step
1/1 [==============================] - 1s 702ms/step
1/1 [==============================] - 1s 707ms/step
1/1 [==============================] - 1s 702ms/step
1/1 [==============================] - 1s 749ms/step
1/1 [==============================] - 1s 797ms/step
1/1 [==============================] - 1s 803ms/step
1/1 [==============================] - 1s 780ms/step
1/1 [==============================] - 1s 770ms/step
1/1 [==============================] - 1s 775ms/step
1/1 [==============================] - 1s 717ms/step
1/1 [==============================] - 1s 782ms/step
1/1 [==============================] - 1s 758ms/step
1/1 [==============================] - 1s 849ms/step
1/1 [==============================] - 1s 848ms/step
1/1 [==============================] - 1s 822ms/step
1/1 [==============================] - 1s 727ms/step
1/1 [==============================] - 1s 807ms/step
1/1 [==============================] - 1s 757ms/step
1/1 [==============================] - 1s 800ms/step
1/1 [==============================] - 1s 746ms/step
1/1 [==============================] - 1s 763ms/step
```

```
1/1 [==============================] - 1s 825ms/step
1/1 [==============================] - 1s 813ms/step
1/1 [==============================] - 1s 785ms/step
1/1 [==============================] - 1s 802ms/step
1/1 [==============================] - 1s 796ms/step
1/1 [==============================] - 1s 810ms/step
1/1 [==============================] - 1s 696ms/step
1/1 [==============================] - 1s 708ms/step
1/1 [==============================] - 1s 718ms/step
1/1 [==============================] - 1s 712ms/step
1/1 [==============================] - 1s 723ms/step
1/1 [==============================] - 1s 804ms/step
1/1 [==============================] - 1s 753ms/step
1/1 [==============================] - 1s 714ms/step
1/1 [==============================] - 1s 710ms/step
1/1 [==============================] - 1s 707ms/step
1/1 [==============================] - 1s 697ms/step
1/1 [==============================] - 1s 702ms/step
1/1 [==============================] - 1s 705ms/step
1/1 [==============================] - 1s 709ms/step
1/1 [==============================] - 1s 706ms/step
1/1 [==============================] - 1s 763ms/step
1/1 [==============================] - 1s 843ms/step
1/1 [==============================] - 1s 814ms/step
1/1 [==============================] - 1s 784ms/step
1/1 [==============================] - 1s 753ms/step
1/1 [==============================] - 1s 717ms/step
1/1 [==============================] - 1s 696ms/step
1/1 [==============================] - 1s 695ms/step
1/1 [==============================] - 1s 716ms/step
1/1 [==============================] - 1s 701ms/step
1/1 [==============================] - 1s 703ms/step
1/1 [==============================] - 1s 706ms/step
1/1 [==============================] - 1s 703ms/step
1/1 [==============================] - 1s 692ms/step
1/1 [==============================] - 1s 701ms/step
1/1 [==============================] - 1s 696ms/step
1/1 [==============================] - 1s 702ms/step
1/1 [==============================] - 1s 703ms/step
1/1 [==============================] - 1s 712ms/step
1/1 [==============================] - 1s 701ms/step
1/1 [==============================] - 1s 733ms/step
1/1 [==============================] - 1s 788ms/step
1/1 [==============================] - 1s 772ms/step
1/1 [==============================] - 1s 786ms/step
1/1 [==============================] - 1s 796ms/step
1/1 [==============================] - 1s 723ms/step
1/1 [==============================] - 1s 704ms/step
1/1 [==============================] - 1s 709ms/step
1/1 [==============================] - 1s 715ms/step
1/1 [==============================] - 1s 721ms/step
1/1 [==============================] - 1s 710ms/step
1/1 [==============================] - 1s 719ms/step
1/1 [==============================] - 1s 770ms/step
1/1 [==============================] - 1s 765ms/step
1/1 [==============================] - 1s 723ms/step
1/1 [==============================] - 1s 703ms/step
1/1 [==============================] - 1s 686ms/step
1/1 [==============================] - 1s 702ms/step
1/1 [==============================] - 1s 707ms/step
1/1 [==============================] - 1s 757ms/step
1/1 [==============================] - 1s 711ms/step
1/1 [==============================] - 1s 807ms/step
1/1 [==============================] - 1s 782ms/step
1/1 [==============================] - 1s 796ms/step
1/1 [==============================] - 1s 792ms/step
```

```
1/1 [==============================] - 1s 743ms/step
1/1 [==============================] - 1s 711ms/step
1/1 [==============================] - 1s 712ms/step
1/1 [==============================] - 1s 697ms/step
1/1 [==============================] - 1s 701ms/step
1/1 [==============================] - 1s 710ms/step
1/1 [==============================] - 1s 691ms/step
1/1 [==============================] - 1s 713ms/step
1/1 [==============================] - 1s 706ms/step
1/1 [==============================] - 1s 704ms/step
1/1 [==============================] - 1s 695ms/step
1/1 [==============================] - 1s 703ms/step
1/1 [==============================] - 1s 708ms/step
1/1 [==============================] - 1s 702ms/step
1/1 [==============================] - 1s 696ms/step
1/1 [==============================] - 1s 721ms/step
1/1 [==============================] - 1s 767ms/step
1/1 [==============================] - 1s 782ms/step
1/1 [==============================] - 1s 788ms/step
1/1 [==============================] - 1s 779ms/step
1/1 [==============================] - 1s 849ms/step
1/1 [==============================] - 1s 783ms/step
1/1 [==============================] - 1s 697ms/step
1/1 [==============================] - 1s 701ms/step
1/1 [==============================] - 1s 705ms/step
1/1 [==============================] - 1s 733ms/step
1/1 [==============================] - 1s 715ms/step
1/1 [==============================] - 1s 698ms/step
1/1 [==============================] - 1s 696ms/step
1/1 [==============================] - 1s 702ms/step
1/1 [==============================] - 1s 715ms/step
1/1 [==============================] - 1s 719ms/step
1/1 [==============================] - 1s 717ms/step
1/1 [==============================] - 1s 698ms/step
1/1 [==============================] - 1s 709ms/step
1/1 [==============================] - 1s 694ms/step
1/1 [==============================] - 1s 749ms/step
1/1 [==============================] - 1s 777ms/step
1/1 [==============================] - 1s 792ms/step
1/1 [==============================] - 1s 789ms/step
1/1 [==============================] - 1s 770ms/step
1/1 [==============================] - 1s 701ms/step
1/1 [==============================] - 1s 706ms/step
1/1 [==============================] - 1s 705ms/step
1/1 [==============================] - 1s 697ms/step
1/1 [==============================] - 1s 732ms/step
1/1 [==============================] - 1s 688ms/step
1/1 [==============================] - 1s 719ms/step
1/1 [==============================] - 1s 694ms/step
1/1 [==============================] - 1s 694ms/step
1/1 [==============================] - 1s 720ms/step
1/1 [==============================] - 1s 694ms/step
1/1 [==============================] - 1s 694ms/step
1/1 [==============================] - 1s 697ms/step
1/1 [==============================] - 1s 693ms/step
1/1 [==============================] - 1s 699ms/step
1/1 [==============================] - 1s 718ms/step
1/1 [==============================] - 1s 784ms/step
1/1 [==============================] - 1s 796ms/step
1/1 [==============================] - 1s 780ms/step
1/1 [==============================] - 1s 785ms/step
1/1 [==============================] - 1s 759ms/step
1/1 [==============================] - 1s 706ms/step
1/1 [==============================] - 1s 727ms/step
1/1 [==============================] - 1s 720ms/step
1/1 [==============================] - 1s 719ms/step
```

```
1/1 [==============================] - 1s 694ms/step
1/1 [==============================] - 1s 701ms/step
1/1 [==============================] - 1s 717ms/step
1/1 [==============================] - 1s 707ms/step
1/1 [==============================] - 1s 695ms/step
1/1 [==============================] - 1s 710ms/step
1/1 [==============================] - 1s 707ms/step
1/1 [==============================] - 1s 719ms/step
1/1 [==============================] - 1s 715ms/step
1/1 [==============================] - 1s 709ms/step
1/1 [==============================] - 1s 701ms/step
1/1 [==============================] - 1s 783ms/step
1/1 [==============================] - 1s 765ms/step
1/1 [==============================] - 1s 778ms/step
1/1 [==============================] - 1s 775ms/step
1/1 [==============================] - 1s 768ms/step
1/1 [==============================] - 1s 787ms/step
1/1 [==============================] - 1s 772ms/step
1/1 [==============================] - 1s 702ms/step
1/1 [==============================] - 1s 718ms/step
1/1 [==============================] - 1s 726ms/step
1/1 [==============================] - 1s 698ms/step
1/1 [==============================] - 1s 703ms/step
1/1 [==============================] - 1s 701ms/step
1/1 [==============================] - 1s 713ms/step
1/1 [==============================] - 1s 710ms/step
1/1 [==============================] - 1s 702ms/step
1/1 [==============================] - 1s 699ms/step
1/1 [==============================] - 1s 703ms/step
1/1 [==============================] - 1s 701ms/step
1/1 [==============================] - 1s 718ms/step
1/1 [==============================] - 1s 727ms/step
1/1 [==============================] - 1s 787ms/step
1/1 [==============================] - 1s 775ms/step
1/1 [==============================] - 1s 805ms/step
1/1 [==============================] - 1s 775ms/step
1/1 [==============================] - 1s 707ms/step
1/1 [==============================] - 1s 704ms/step
1/1 [==============================] - 1s 707ms/step
1/1 [==============================] - 1s 705ms/step
1/1 [==============================] - 1s 708ms/step
1/1 [==============================] - 1s 711ms/step
1/1 [==============================] - 1s 701ms/step
1/1 [==============================] - 1s 705ms/step
1/1 [==============================] - 1s 703ms/step
1/1 [==============================] - 1s 705ms/step
1/1 [==============================] - 1s 692ms/step
1/1 [==============================] - 1s 707ms/step
1/1 [==============================] - 1s 712ms/step
1/1 [==============================] - 1s 699ms/step
1/1 [==============================] - 1s 692ms/step
1/1 [==============================] - 1s 700ms/step
1/1 [==============================] - 1s 764ms/step
1/1 [==============================] - 1s 787ms/step
1/1 [==============================] - 1s 780ms/step
1/1 [==============================] - 1s 787ms/step
1/1 [==============================] - 1s 772ms/step
1/1 [==============================] - 1s 737ms/step
1/1 [==============================] - 1s 701ms/step
1/1 [==============================] - 1s 702ms/step
1/1 [==============================] - 1s 720ms/step
1/1 [==============================] - 1s 716ms/step
1/1 [==============================] - 1s 705ms/step
1/1 [==============================] - 1s 708ms/step
1/1 [==============================] - 1s 691ms/step
1/1 [==============================] - 1s 703ms/step
```

```
1/1 [==============================] - 1s 706ms/step
1/1 [==============================] - 1s 720ms/step
1/1 [==============================] - 1s 700ms/step
1/1 [==============================] - 1s 688ms/step
1/1 [==============================] - 1s 702ms/step
1/1 [==============================] - 1s 708ms/step
1/1 [==============================] - 1s 751ms/step
1/1 [==============================] - 1s 790ms/step
1/1 [==============================] - 1s 796ms/step
1/1 [==============================] - 1s 781ms/step
1/1 [==============================] - 1s 785ms/step
1/1 [==============================] - 1s 715ms/step
1/1 [==============================] - 1s 701ms/step
1/1 [==============================] - 1s 696ms/step
1/1 [==============================] - 1s 708ms/step
1/1 [==============================] - 1s 720ms/step
1/1 [==============================] - 1s 781ms/step
1/1 [==============================] - 1s 688ms/step
1/1 [==============================] - 1s 723ms/step
1/1 [==============================] - 1s 778ms/step
1/1 [==============================] - 1s 712ms/step
1/1 [==============================] - 1s 717ms/step
1/1 [==============================] - 1s 721ms/step
1/1 [==============================] - 1s 708ms/step
1/1 [==============================] - 1s 703ms/step
1/1 [==============================] - 1s 709ms/step
1/1 [==============================] - 1s 788ms/step
1/1 [==============================] - 1s 779ms/step
1/1 [==============================] - 1s 776ms/step
1/1 [==============================] - 1s 773ms/step
1/1 [==============================] - 1s 771ms/step
1/1 [==============================] - 1s 718ms/step
1/1 [==============================] - 1s 704ms/step
1/1 [==============================] - 1s 722ms/step
1/1 [==============================] - 1s 717ms/step
1/1 [==============================] - 1s 721ms/step
1/1 [==============================] - 1s 694ms/step
1/1 [==============================] - 1s 704ms/step
1/1 [==============================] - 1s 692ms/step
1/1 [==============================] - 1s 711ms/step
1/1 [==============================] - 1s 711ms/step
1/1 [==============================] - 1s 710ms/step
1/1 [==============================] - 1s 706ms/step
1/1 [==============================] - 1s 711ms/step
1/1 [==============================] - 1s 695ms/step
1/1 [==============================] - 2s 2s/step
```

Print classification report

In [30]:
```
print(classification_report(y_true, y_pred, target_names=class_names))
```

```
                      precision    recall  f1-score   support

      Apple__Healthy       1.00      1.00      1.00       488
       Apple__Rotten       1.00      0.98      0.99       586
     Banana__Healthy       1.00      1.00      1.00       399
      Banana__Rotten       1.00      1.00      1.00       559
 Bellpepper__Healthy       1.00      1.00      1.00       400
  Bellpepper__Rotten       0.97      0.99      0.98       400
     Carrot__Healthy       0.97      0.99      0.98       399
      Carrot__Rotten       0.99      0.96      0.98       400
   Cucumber__Healthy       0.99      1.00      1.00       400
    Cucumber__Rotten       0.99      0.99      0.99       400
      Grape__Healthy       1.00      1.00      1.00       400
       Grape__Rotten       1.00      1.00      1.00       400
      Guava__Healthy       1.00      1.00      1.00       400
       Guava__Rotten       1.00      1.00      1.00       400
     Jujube__Healthy       1.00      1.00      1.00       400
      Jujube__Rotten       1.00      1.00      1.00       400
      Mango__Healthy       0.99      0.99      0.99       400
       Mango__Rotten       1.00      0.99      0.99       450
     Orange__Healthy       1.00      1.00      1.00       415
      Orange__Rotten       0.99      1.00      1.00       438
Pomegranate__Healthy       1.00      1.00      1.00       400
 Pomegranate__Rotten       1.00      1.00      1.00       400
     Potato__Healthy       0.99      0.99      0.99       400
      Potato__Rotten       0.99      0.99      0.99       400
 Strawberry__Healthy       1.00      1.00      1.00       400
  Strawberry__Rotten       1.00      1.00      1.00       400
     Tomato__Healthy       1.00      1.00      1.00       400
      Tomato__Rotten       1.00      1.00      1.00       400

            accuracy                           0.99     11734
           macro avg       0.99      0.99      0.99     11734
        weighted avg       0.99      0.99      0.99     11734
```

Compute the confusion matrix

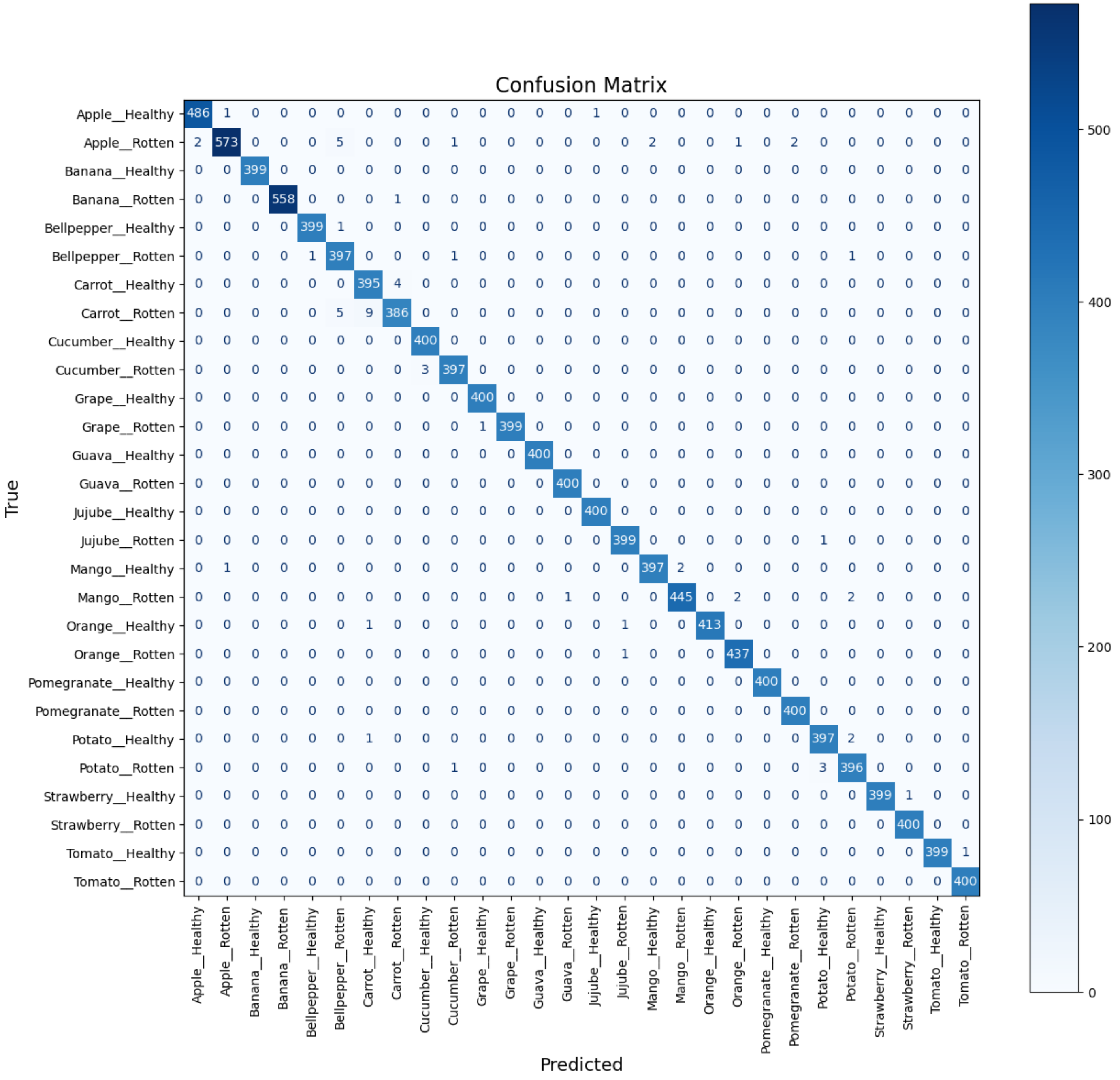
In [31]:

```python
confusion = confusion_matrix(y_true, y_pred)
```

Plot the confusion matrix

In [32]:

```python
from sklearn.metrics import ConfusionMatrixDisplay

def plot_confusion_matrix(confusion, class_names):
    num_classes = len(class_names)
    fig, ax = plt.subplots(figsize=(14, 14))

    # Convert the confusion matrix values to integers
    confusion = confusion.astype(int)

    disp = ConfusionMatrixDisplay(confusion, display_labels=class_names)
    disp = disp.plot(cmap=plt.get_cmap("Blues"), values_format="d", ax=ax)

    # Rotate y-axis class names to be straight at 90 degrees
    ax.set_yticklabels(class_names, rotation=0, fontsize=10)

    # Set the tick labels and fontsize for x-axis
    tick_marks = np.arange(num_classes)
    plt.xticks(tick_marks, class_names, rotation=90, fontsize=10)

    plt.title("Confusion Matrix", fontsize=16)
    plt.xlabel("Predicted", fontsize=14)
    plt.ylabel("True", fontsize=14)
    plt.show()

plot_confusion_matrix(confusion, class_names)
```



Step 6: Save Model

In [33]:
```
tf.saved_model.save(feature_model, 'fruit_vegetable_disease_detection_model')
```

```
INFO:tensorflow:Assets written to: fruit_vegetable_disease_detection_model\assets
INFO:tensorflow:Assets written to: fruit_vegetable_disease_detection_model\assets
```

In [34]:
```
h5_model_path = 'fruit_vegetable_disease_detection_model.h5'
feature_model.save(h5_model_path)
```

```
C:\Users\PMLS\anaconda3\Lib\site-packages\keras\src\engine\training.py:3103: UserWarnin
g: You are saving your model as an HDF5 file via `model.save()`. This file format is con
sidered legacy. We recommend using instead the native Keras format, e.g. `model.save('my
_model.keras')`.
  saving_api.save_model(
```

In [35]:
```
tflite_model_path = 'fruit_vegetable_disease_detection_model.tflite'
converter = tf.lite.TFLiteConverter.from_keras_model(feature_model)
tflite_model = converter.convert()
```

```
with open(tflite_model_path, 'wb') as f:
    f.write(tflite_model)
```

```
INFO:tensorflow:Assets written to: C:\Users\PMLS\AppData\Local\Temp\tmpqqb4bq3y\assets
```

```
INFO:tensorflow:Assets written to: C:\Users\PMLS\AppData\Local\Temp\tmpqqb4bq3y\assets
```

Step 7: Image Prediction

Define a function to load and preprocess an image

In [106…

```
def load_prep(img_path):
    img = tf.io.read_file(img_path)
    img = tf.image.decode_image(img)
    img = tf.image.resize(img, size=(224, 224))
    return img
```

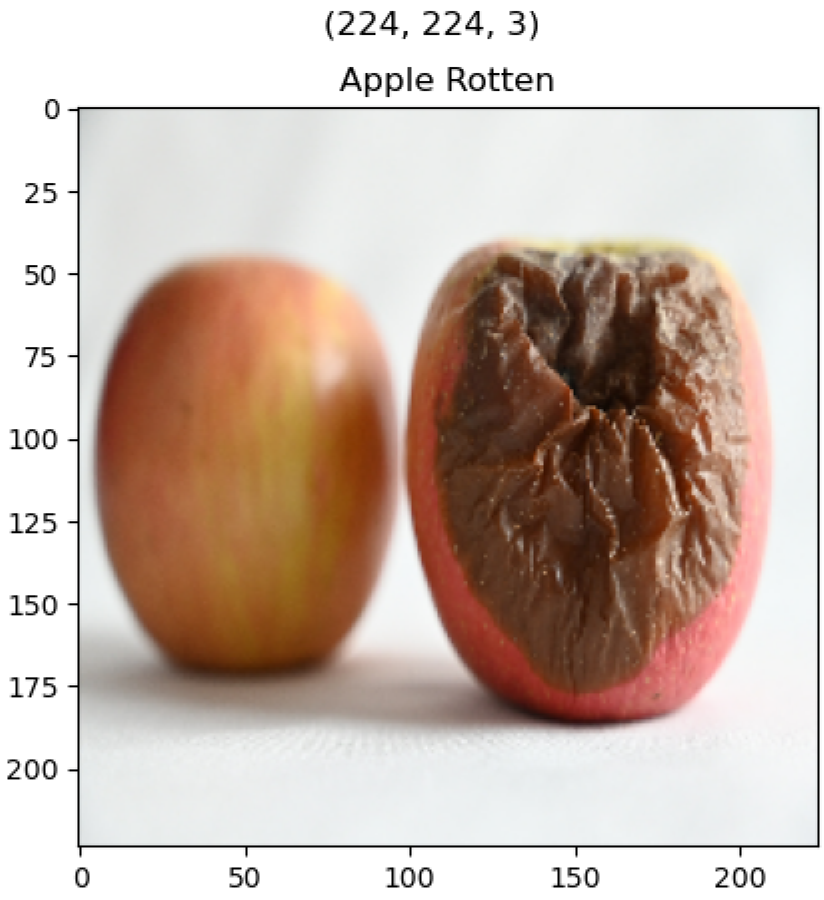
Load and preprocess an image, and make a prediction

In [107…

```
image = load_prep('test/Apple Rotten 1.jpg')
plt.imshow(image / 255.)
plt.title('Apple Rotten')
plt.suptitle(image.shape)
```

Out[107]:

```
Text(0.5, 0.98, '(224, 224, 3)')
```



In [108…

```
pred = feature_model.predict(tf.expand_dims(image, axis=0))
predicted_class = class_names[pred.argmax()]
predicted_prob = pred.max()
```

```
1/1 [==============================] - 1s 883ms/step
```

Print the predicted class and probability

In [109…

```
print(f'Predicted Class: {predicted_class}')
print(f'Predicted Probability: {predicted_prob * 100:.2f}%')
```

```
Predicted Class: Apple__Rotten
Predicted Probability: 100.00%
```

Define a function to randomly select an image from the test data and make a prediction

In [110…

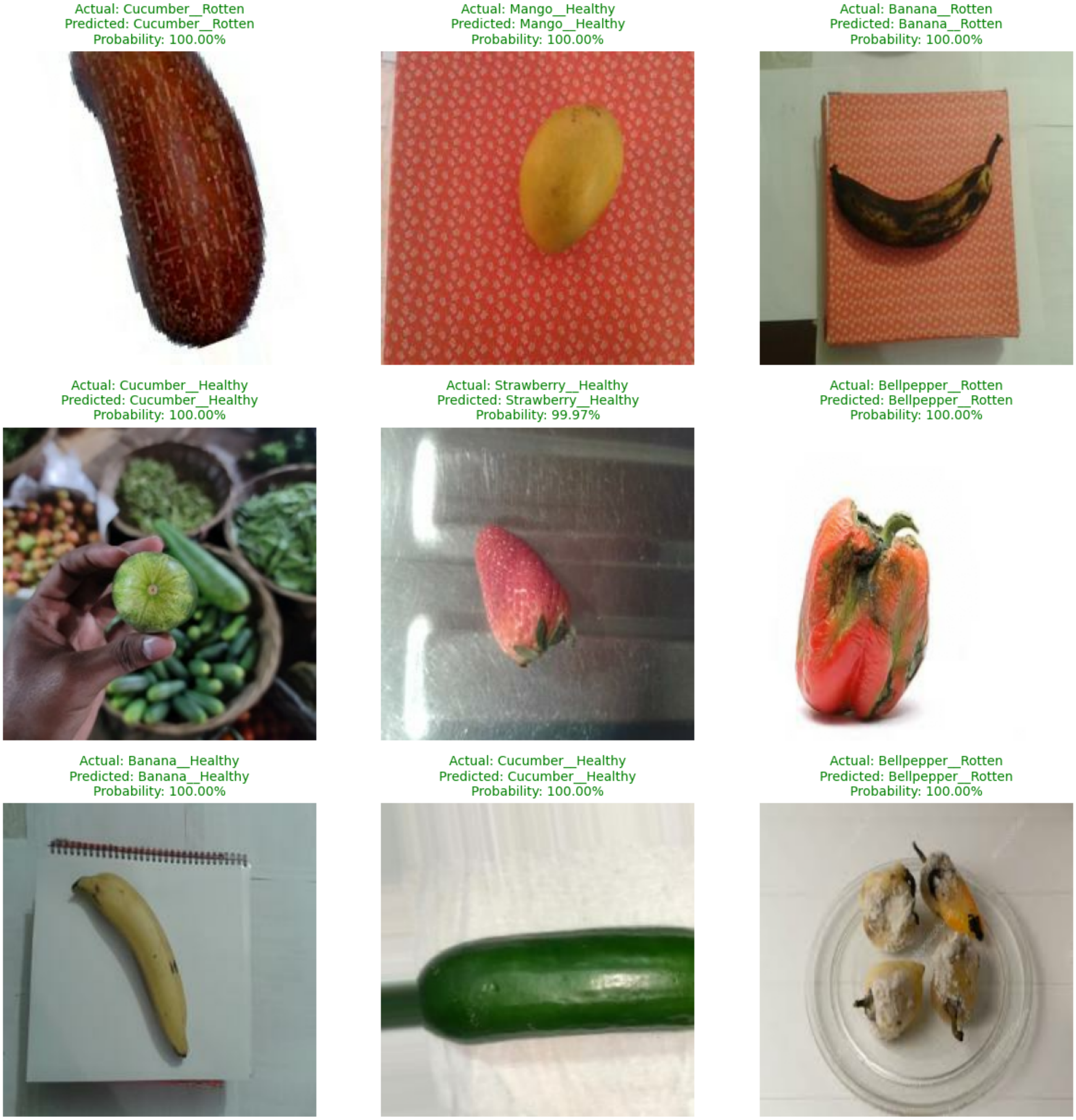
```
def random_image_predict(model, test_dir=test_dir, class_names=class_names, rand_class=T
    if rand_class:
        ran_cls = random.randint(0, len(class_names) - 1)
        cls = class_names[ran_cls]

        # Get a list of all files in the class directory
        class_dir = os.path.join(test_dir, cls)
        files = os.listdir(class_dir)

        # Choose a random file from the list
        random_file = random.choice(files)

        # Create the full path to the random file
        ran_path = os.path.join(class_dir, random_file)
    else:
        cls = class_names[cls_name]

        # Get a list of all files in the class directory
        class_dir = os.path.join(test_dir, cls)
        files = os.listdir(class_dir)

        # Choose a random file from the list
        random_file = random.choice(files)

        # Create the full path to the random file
        ran_path = os.path.join(class_dir, random_file)

    prep_img = load_prep(ran_path)

    pred = model.predict(tf.expand_dims(prep_img, axis=0))
    pred_cls = class_names[pred[0].argmax()]
    pred_percent = pred[0][pred[0].argmax()] * 100
    plt.imshow(prep_img / 255.)
    if pred_cls == cls:
        c = 'g'
    else:
        c = 'r'
    plt.title(f'Actual: {cls}\nPredicted: {pred_cls}\nProbability: {pred_percent:.2f}%',
    plt.axis(False)
```

Display 9 randomly predicted images from the test data

In [111…

```
plt.figure(figsize=(15, 15))
for i in range(9):
    plt.subplot(3, 3, i + 1)
    random_image_predict(feature_model, test_dir)
```

```
1/1 [==============================] - 0s 271ms/step
1/1 [==============================] - 0s 232ms/step
1/1 [==============================] - 0s 242ms/step
1/1 [==============================] - 0s 336ms/step
1/1 [==============================] - 0s 230ms/step
1/1 [==============================] - 0s 297ms/step
1/1 [==============================] - 0s 279ms/step
1/1 [==============================] - 0s 218ms/step
1/1 [==============================] - 0s 231ms/step
```


Actual: Cucumber__Rotten
Predicted: Cucumber__Rotten
Probability: 100.00%
Actual: Mango__Healthy
Predicted: Mango__Healthy
Probability: 100.00%
Actual: Banana__Rotten
Predicted: Banana__Rotten
Probability: 100.00%
Actual: Cucumber__Healthy
Predicted: Cucumber__Healthy
Probability: 100.00%
Actual: Strawberry__Healthy
Predicted: Strawberry__Healthy
Probability: 99.97%
Actual: Bellpepper__Rotten
Predicted: Bellpepper__Rotten
Probability: 100.00%
Actual: Banana__Healthy
Predicted: Banana__Healthy
Probability: 100.00%
Actual: Cucumber__Healthy
Predicted: Cucumber__Healthy
Probability: 100.00%
Actual: Bellpepper__Rotten
Predicted: Bellpepper__Rotten
Probability: 100.00%

Define a directory containing images for prediction

In [112…

```
data_dir = 'test'
plt.figure(figsize=(15, 10))
for i in range(9):
    plt.subplot(3, 3, i + 1)
    rn = random.choice(os.listdir(data_dir))
    image_path = os.path.join(data_dir, rn)
    img = load_prep(image_path)
    pred = feature_model.predict(tf.expand_dims(img, axis=0))
    pred_name = class_names[pred.argmax()]
    plt.imshow(img / 255.)
    plt.title(f'True: {rn}\nPredicted Class: {pred_name}')
    plt.axis(False)
```

```
1/1 [==============================] - 0s 237ms/step
1/1 [==============================] - 0s 249ms/step
1/1 [==============================] - 0s 260ms/step
1/1 [==============================] - 0s 298ms/step
```

```
1/1 [==============================] - 0s 231ms/step
1/1 [==============================] - 0s 216ms/step
1/1 [==============================] - 0s 232ms/step
1/1 [==============================] - 0s 272ms/step
1/1 [==============================] - 0s 234ms/step
```

True: Grape Healthy 4.jpg
Predicted Class: Jujube__Healthy

True: Potato Healthy 5.jpg
Predicted Class: Potato__Healthy

True: Potato Healthy 4.jpg
Predicted Class: Potato__Healthy

True: Mango Healthy 3.jpg
Predicted Class: Mango__Healthy

True: Tomato Healthy 6.jpg
Predicted Class: Tomato__Healthy

True: Tomato Healthy 6.jpg
Predicted Class: Tomato__Healthy

True: Apple Healthy 1.jpg
Predicted Class: Orange__Healthy

True: Potato Rotten 2.jpg
Predicted Class: Potato__Rotten

True: Cucumber Healthy 7.jpg
Predicted Class: Cucumber__Healthy

Define a function to predict an image from a given path

In [113…

```
def predict_img(img_path, model=feature_model):
    img = load_prep(img_path)
    pred = model.predict(tf.expand_dims(img, axis=0))
    pred_name = class_names[pred.argmax()]
    plt.imshow(img / 255.)
    plt.title(f'Predicted Class: {pred_name}')
    plt.axis(False)
```

Step 9: Image Prediction for load Crop Diseases Detection model

In [141…

```
loaded_model = tf.saved_model.load('fruit_vegetable_disease_detection_model')
```

Define a function to load and preprocess an image

In [147…

```
def load_prep(img_path):
    img = tf.io.read_file(img_path)
    img = tf.image.decode_image(img)
    img = tf.image.resize(img, size=(224, 224))
    return img
```

Define the directory containing the images for prediction

In [148…

```
test_directory = 'test'
```

Get a list of image file paths

In [149…

```
image_paths = [os.path.join(test_directory, img) for img in os.listdir(test_directory)]
```

Make predictions on each image

In [150…

```
predictions = []

for img_path in image_paths:
    img = load_prep(img_path)
    img = tf.expand_dims(img, axis=0)

    # Run inference using the loaded model
    prediction = loaded_model(img)
    predicted_class = class_names[np.argmax(prediction)]
    predictions.append((img_path, predicted_class))
```

Display the predictions

In [151…

```
for img_path, predicted_class in predictions:
    print(f'Image: {os.path.basename(img_path)} - Predicted Class: {predicted_class}')
```

```
Image: Apple Healthy 1.jpg - Predicted Class: Orange__Healthy
Image: Apple Healthy 2.jpg - Predicted Class: Potato__Healthy
Image: Apple Healthy 3.jpg - Predicted Class: Apple__Healthy
Image: Apple Healthy 4.jpg - Predicted Class: Apple__Healthy
Image: Apple Healthy 5.jpg - Predicted Class: Strawberry__Healthy
Image: Apple Healthy 6.jpg - Predicted Class: Apple__Healthy
Image: Apple Rotten 1.jpg - Predicted Class: Apple__Rotten
Image: Apple Rotten 2.jpg - Predicted Class: Apple__Rotten
Image: Apple Rotten 3.jpg - Predicted Class: Apple__Rotten
Image: Apple Rotten 4.jpg - Predicted Class: Apple__Rotten
Image: Apple Rotten 5.jpg - Predicted Class: Tomato__Rotten
Image: Apple Rotten 6.jpg - Predicted Class: Apple__Rotten
Image: Banana Healthy 1.jpg - Predicted Class: Banana__Healthy
Image: Banana Healthy 2.jpg - Predicted Class: Banana__Healthy
Image: Banana Healthy 3.jpg - Predicted Class: Banana__Healthy
Image: Banana Healthy 4.jpg - Predicted Class: Banana__Healthy
Image: Banana Healthy 5.jpg - Predicted Class: Banana__Healthy
Image: Banana Healthy 6.jpg - Predicted Class: Banana__Healthy
Image: Banana Rotten 1.jpg - Predicted Class: Banana__Rotten
Image: Banana Rotten 2.jpg - Predicted Class: Banana__Rotten
Image: Banana Rotten 3.jpg - Predicted Class: Banana__Rotten
Image: Banana Rotten 4.jpg - Predicted Class: Banana__Rotten
Image: Banana Rotten 5.jpg - Predicted Class: Banana__Rotten
Image: Banana Rotten 6.jpg - Predicted Class: Banana__Rotten
Image: Bellpepper Healthy 1.jpg - Predicted Class: Bellpepper__Healthy
Image: Bellpepper Healthy 2.jpg - Predicted Class: Bellpepper__Healthy
Image: Bellpepper Healthy 3.jpg - Predicted Class: Bellpepper__Healthy
Image: Bellpepper Healthy 4.jpg - Predicted Class: Bellpepper__Healthy
Image: Bellpepper Healthy 5.jpg - Predicted Class: Bellpepper__Healthy
Image: Bellpepper Healthy 6.jpg - Predicted Class: Bellpepper__Rotten
Image: Bellpepper Healthy 7.jpg - Predicted Class: Bellpepper__Healthy
Image: Bellpepper Rotten 1.jpg - Predicted Class: Bellpepper__Rotten
Image: Bellpepper Rotten 2.jpg - Predicted Class: Bellpepper__Rotten
Image: Bellpepper Rotten 3.jpg - Predicted Class: Bellpepper__Rotten
Image: Bellpepper Rotten 4.jpg - Predicted Class: Bellpepper__Rotten
Image: Bellpepper Rotten 5.jpg - Predicted Class: Bellpepper__Rotten
```

```
Image: Bellpepper Rotten 6.jpg - Predicted Class: Bellpepper__Rotten
Image: Bellpepper Rotten 7.jpg - Predicted Class: Bellpepper__Rotten
Image: Carrot Healthy 2.jpg - Predicted Class: Carrot__Healthy
Image: Carrot Healthy 3.jpg - Predicted Class: Carrot__Healthy
Image: Carrot Healthy 4.jpg - Predicted Class: Carrot__Healthy
Image: Carrot Healthy 5.jpg - Predicted Class: Carrot__Healthy
Image: Carrot Healthy 6.jpg - Predicted Class: Carrot__Healthy
Image: Carrot Rotten 1.jpg - Predicted Class: Carrot__Rotten
Image: Carrot Rotten 2.jpg - Predicted Class: Carrot__Rotten
Image: Carrot Rotten 3.jpg - Predicted Class: Carrot__Rotten
Image: Carrot Rotten 4.jpg - Predicted Class: Carrot__Rotten
Image: Cucumber  Healthy  4.jpg - Predicted Class: Cucumber__Healthy
Image: Cucumber  Healthy  5.jpg - Predicted Class: Cucumber__Healthy
Image: Cucumber  Healthy  6.jpg - Predicted Class: Cucumber__Healthy
Image: Cucumber  Healthy  7.jpg - Predicted Class: Cucumber__Healthy
Image: Cucumber  Healthy 1.jpg - Predicted Class: Cucumber__Healthy
Image: Cucumber  Healthy 2.jpg - Predicted Class: Cucumber__Healthy
Image: Cucumber  Healthy 3.jpg - Predicted Class: Cucumber__Healthy
Image: Cucumber Rotten 1.jpg - Predicted Class: Cucumber__Rotten
Image: Cucumber Rotten 2.jpg - Predicted Class: Cucumber__Rotten
Image: Cucumber Rotten 3.jpg - Predicted Class: Cucumber__Rotten
Image: Cucumber Rotten 4.jpg - Predicted Class: Cucumber__Rotten
Image: Cucumber Rotten 5.jpg - Predicted Class: Cucumber__Rotten
Image: Cucumber Rotten 6.jpg - Predicted Class: Cucumber__Rotten
Image: Grape Healthy 1.jpg - Predicted Class: Apple__Rotten
Image: Grape Healthy 2.jpg - Predicted Class: Grape__Healthy
Image: Grape Healthy 3.jpg - Predicted Class: Tomato__Healthy
Image: Grape Healthy 4.jpg - Predicted Class: Jujube__Healthy
Image: Grape Healthy 5.jpg - Predicted Class: Grape__Healthy
Image: Grape Healthy 6.jpg - Predicted Class: Jujube__Healthy
Image: Guava Healthy 1.jpg - Predicted Class: Guava__Healthy
Image: Guava Healthy 2.jpg - Predicted Class: Guava__Healthy
Image: Guava Healthy 3.jpg - Predicted Class: Cucumber__Rotten
Image: Guava Healthy 4.jpg - Predicted Class: Guava__Healthy
Image: Guava Rotten 1.jpg - Predicted Class: Apple__Rotten
Image: Guava Rotten 2.jpg - Predicted Class: Mango__Rotten
Image: Guava Rotten 3.jpg - Predicted Class: Guava__Rotten
Image: Guava Rotten 4.jpg - Predicted Class: Mango__Rotten
Image: Guava Rotten 5.jpg - Predicted Class: Apple__Rotten
Image: Jujube Healthy 1.jpg - Predicted Class: Tomato__Healthy
Image: Jujube Healthy 2.jpg - Predicted Class: Jujube__Healthy
Image: Jujube Healthy 3.jpg - Predicted Class: Jujube__Healthy
Image: Jujube Healthy 4.jpg - Predicted Class: Jujube__Healthy
Image: Jujube Healthy 5.jpg - Predicted Class: Jujube__Healthy
Image: Jujube Rotten 1.jpg - Predicted Class: Apple__Rotten
Image: Jujube Rotten 2.jpg - Predicted Class: Apple__Rotten
Image: Jujube Rotten 3.jpg - Predicted Class: Jujube__Rotten
Image: Jujube Rotten 4.jpg - Predicted Class: Jujube__Rotten
Image: Mango Healthy 1.jpg - Predicted Class: Mango__Healthy
Image: Mango Healthy 2.jpg - Predicted Class: Mango__Healthy
Image: Mango Healthy 3.jpg - Predicted Class: Mango__Healthy
Image: Mango Healthy 4.jpg - Predicted Class: Mango__Healthy
Image: Mango Rotten 1.jpg - Predicted Class: Mango__Rotten
Image: Mango Rotten 2.jpg - Predicted Class: Mango__Rotten
Image: Mango Rotten 3.jpg - Predicted Class: Mango__Rotten
Image: Mango Rotten 4.jpg - Predicted Class: Mango__Rotten
Image: Orange Healthy 1.png - Predicted Class: Orange__Healthy
Image: Orange Healthy 5.png - Predicted Class: Orange__Healthy
Image: Orange Rotten 1.jpg - Predicted Class: Orange__Rotten
Image: Orange Rotten 2.jpg - Predicted Class: Orange__Rotten
Image: Orange Rotten 3.jpg - Predicted Class: Orange__Rotten
Image: Orange Rotten 4.jpg - Predicted Class: Orange__Rotten
Image: Orange Rotten 5.jpg - Predicted Class: Orange__Rotten
Image: Orange Rotten 6.jpg - Predicted Class: Orange__Rotten
Image: Pomegranate Healthy 1.jpg - Predicted Class: Tomato__Rotten
Image: Pomegranate Healthy 2.jpg - Predicted Class: Pomegranate__Healthy
```

```
Image: Pomegranate Healthy 3.jpg - Predicted Class: Apple__Healthy
Image: Pomegranate Healthy 4.jpg - Predicted Class: Tomato__Rotten
Image: Pomegranate Rotten 1.jpg - Predicted Class: Pomegranate__Rotten
Image: Pomegranate Rotten 2.jpg - Predicted Class: Pomegranate__Rotten
Image: Pomegranate Rotten 3.jpg - Predicted Class: Pomegranate__Rotten
Image: Potato Healthy 1.jpg - Predicted Class: Potato__Healthy
Image: Potato Healthy 2.jpg - Predicted Class: Potato__Healthy
Image: Potato Healthy 3.jpg - Predicted Class: Potato__Healthy
Image: Potato Healthy 4.jpg - Predicted Class: Potato__Healthy
Image: Potato Healthy 5.jpg - Predicted Class: Potato__Healthy
Image: Potato Healthy 6.jpg - Predicted Class: Potato__Healthy
Image: Potato Rotten 1.jpg - Predicted Class: Potato__Rotten
Image: Potato Rotten 2.jpg - Predicted Class: Potato__Rotten
Image: Potato Rotten 3.jpg - Predicted Class: Potato__Rotten
Image: Strawberry Healthy  1.jpg - Predicted Class: Strawberry__Rotten
Image: Strawberry Healthy 2.jpg - Predicted Class: Strawberry__Rotten
Image: Strawberry Healthy 3.jpg - Predicted Class: Strawberry__Healthy
Image: Strawberry Healthy 4.jpg - Predicted Class: Strawberry__Rotten
Image: Strawberry Healthy 5.jpg - Predicted Class: Strawberry__Healthy
Image: Strawberry Rotten 1.jpg - Predicted Class: Strawberry__Rotten
Image: Strawberry Rotten 2.jpg - Predicted Class: Strawberry__Rotten
Image: Strawberry Rotten 3.jpg - Predicted Class: Strawberry__Rotten
Image: Strawberry Rotten 4.jpg - Predicted Class: Strawberry__Rotten
Image: Strawberry Rotten 5.jpg - Predicted Class: Potato__Rotten
Image: Strawberry Rotten 6.jpg - Predicted Class: Strawberry__Rotten
Image: Tomato Healthy 1.jpg - Predicted Class: Tomato__Healthy
Image: Tomato Healthy 2.jpg - Predicted Class: Tomato__Healthy
Image: Tomato Healthy 3.jpg - Predicted Class: Tomato__Healthy
Image: Tomato Healthy 4.jpg - Predicted Class: Tomato__Healthy
Image: Tomato Healthy 5.jpg - Predicted Class: Tomato__Healthy
Image: Tomato Healthy 6.jpg - Predicted Class: Tomato__Healthy
Image: Tomato Rotten 1.jpg - Predicted Class: Tomato__Rotten
Image: Tomato Rotten 2.jpg - Predicted Class: Tomato__Rotten
Image: Tomato Rotten 3.jpg - Predicted Class: Tomato__Rotten
Image: Tomato Rotten 4.jpg - Predicted Class: Tomato__Rotten
Image: Tomato Rotten 5.jpg - Predicted Class: Tomato__Rotten
```