```
from google.colab import drive
drive.mount('/content/drive',force_remount=True)  # Optional: If using Google Drive

!unzip "/content/drive/MyDrive/TOMATO LEAF DISEASE DETECTION.zip" -d "/content/dataset"
```

```
Mounted at /content/drive
Archive:  /content/drive/MyDrive/TOMATO LEAF DISEASE DETECTION.zip
   creating: /content/dataset/TOMATO LEAF DISEASE DETECTION/
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/cam_integration.py
   creating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/
   creating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/
   creating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/
   creating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Bacterial_spot/
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Bacterial_spot/Tomato___Bacterial_spot (1).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Bacterial_spot/Tomato___Bacterial_spot (10).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Bacterial_spot/Tomato___Bacterial_spot (11).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Bacterial_spot/Tomato___Bacterial_spot (12).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Bacterial_spot/Tomato___Bacterial_spot (13).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Bacterial_spot/Tomato___Bacterial_spot (14).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Bacterial_spot/Tomato___Bacterial_spot (15).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Bacterial_spot/Tomato___Bacterial_spot (2).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Bacterial_spot/Tomato___Bacterial_spot (3).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Bacterial_spot/Tomato___Bacterial_spot (4).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Bacterial_spot/Tomato___Bacterial_spot (5).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Bacterial_spot/Tomato___Bacterial_spot (6).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Bacterial_spot/Tomato___Bacterial_spot (7).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Bacterial_spot/Tomato___Bacterial_spot (8).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Bacterial_spot/Tomato___Bacterial_spot (9).JPG
   creating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Early_blight/
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Early_blight/Tomato___Early_blight (1).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Early_blight/Tomato___Early_blight (10).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Early_blight/Tomato___Early_blight (11).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Early_blight/Tomato___Early_blight (12).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Early_blight/Tomato___Early_blight (13).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Early_blight/Tomato___Early_blight (14).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Early_blight/Tomato___Early_blight (15).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Early_blight/Tomato___Early_blight (2).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Early_blight/Tomato___Early_blight (3).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Early_blight/Tomato___Early_blight (4).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Early_blight/Tomato___Early_blight (5).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Early_blight/Tomato___Early_blight (6).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Early_blight/Tomato___Early_blight (7).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Early_blight/Tomato___Early_blight (8).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Early_blight/Tomato___Early_blight (9).JPG
   creating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Healthy/
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Healthy/Tomato___healthy (1).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Healthy/Tomato___healthy (10).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Healthy/Tomato___healthy (11).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Healthy/Tomato___healthy (12).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Healthy/Tomato___healthy (13).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Healthy/Tomato___healthy (14).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Healthy/Tomato___healthy (15).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Healthy/Tomato___healthy (2).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Healthy/Tomato___healthy (3).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Healthy/Tomato___healthy (4).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Healthy/Tomato___healthy (5).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Healthy/Tomato___healthy (6).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Healthy/Tomato___healthy (7).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Healthy/Tomato___healthy (8).JPG
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Healthy/Tomato___healthy (9).JPG
   creating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Late_blight/
  inflating: /content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train/Tomato___Late_blight/Tomato___Late_blight (1).JPG
```

```
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator

# Define paths
train_path = "/content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train"
val_path = "/content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/val"

# Data Augmentation
datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True
)

train_generator = datagen.flow_from_directory(
    train_path,
    target_size=(128, 128),
    batch_size=32,
```

```
    class_mode="categorical"
)

val_datagen = ImageDataGenerator(rescale=1./255)
val_generator = val_datagen.flow_from_directory(
    val_path,
    target_size=(128, 128),
    batch_size=32,
    class_mode="categorical"
)

class_names = list(train_generator.class_indices.keys())  # Get class names
```

```
Found 150 images belonging to 10 classes.
Found 52 images belonging to 10 classes.
```

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout, BatchNormalization, Input

model = Sequential([
    Input(shape=(128, 128, 3)),  # Define input shape here
    Conv2D(32, (3,3), activation='relu'),
    BatchNormalization(),
    MaxPooling2D(2,2),

    Conv2D(64, (3,3), activation='relu'),
    BatchNormalization(),
    MaxPooling2D(2,2),

    Conv2D(128, (3,3), activation='relu'),
    BatchNormalization(),
    MaxPooling2D(2,2),

    Flatten(),
    Dense(128, activation='relu'),
    Dropout(0.5),
    Dense(len(class_names), activation='softmax')  # Output layer
])

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

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 126, 126, 32)	896
batch_normalization (BatchNormalization)	(None, 126, 126, 32)	128
max_pooling2d (MaxPooling2D)	(None, 63, 63, 32)	0
conv2d_1 (Conv2D)	(None, 61, 61, 64)	18,496
batch_normalization_1 (BatchNormalization)	(None, 61, 61, 64)	256
max_pooling2d_1 (MaxPooling2D)	(None, 30, 30, 64)	0
conv2d_2 (Conv2D)	(None, 28, 28, 128)	73,856
batch_normalization_2 (BatchNormalization)	(None, 28, 28, 128)	512
max_pooling2d_2 (MaxPooling2D)	(None, 14, 14, 128)	0
flatten (Flatten)	(None, 25088)	0
dense (Dense)	(None, 128)	3,211,392
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 10)	1,290

Total params: 3,306,826 (12.61 MB)

```
history = model.fit(
    train_generator,
    validation_data=val_generator,
    epochs=150
)

# Save the trained model
model.save("/content/tomato_leaf_disease_cnn.h5")
```

```
5/5 ━━━━━━━━━━━━━━━━━━━━ 8s 1s/step - accuracy: 0.6320 - loss: 1.1585 - val_accuracy: 0.3077 - val_loss: 10.0144
Epoch 99/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 10s 2s/step - accuracy: 0.6252 - loss: 1.1154 - val_accuracy: 0.3269 - val_loss: 13.9599
Epoch 100/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 10s 2s/step - accuracy: 0.6723 - loss: 1.5056 - val_accuracy: 0.3269 - val_loss: 17.4536
Epoch 101/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 16s 4s/step - accuracy: 0.5985 - loss: 1.2403 - val_accuracy: 0.2885 - val_loss: 12.2031
Epoch 102/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 15s 2s/step - accuracy: 0.6398 - loss: 1.3195 - val_accuracy: 0.2692 - val_loss: 5.8991
Epoch 103/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 8s 2s/step - accuracy: 0.6034 - loss: 1.1688 - val_accuracy: 0.2308 - val_loss: 3.7709
Epoch 104/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 9s 2s/step - accuracy: 0.6498 - loss: 0.9242 - val_accuracy: 0.4231 - val_loss: 2.0712
Epoch 105/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 8s 2s/step - accuracy: 0.6628 - loss: 1.0279 - val_accuracy: 0.4038 - val_loss: 2.6958
Epoch 106/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 9s 1s/step - accuracy: 0.6895 - loss: 1.0752 - val_accuracy: 0.4231 - val_loss: 2.3137
Epoch 107/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 10s 2s/step - accuracy: 0.6709 - loss: 1.0042 - val_accuracy: 0.4615 - val_loss: 2.7895
Epoch 108/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 8s 2s/step - accuracy: 0.7090 - loss: 0.9360 - val_accuracy: 0.3654 - val_loss: 2.3475
Epoch 109/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 10s 2s/step - accuracy: 0.7355 - loss: 1.1325 - val_accuracy: 0.4231 - val_loss: 2.9839
Epoch 110/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 10s 2s/step - accuracy: 0.7319 - loss: 0.9727 - val_accuracy: 0.3462 - val_loss: 4.7057
Epoch 111/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 8s 2s/step - accuracy: 0.6454 - loss: 0.9582 - val_accuracy: 0.5000 - val_loss: 3.1601
Epoch 112/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 10s 2s/step - accuracy: 0.7327 - loss: 0.8489 - val_accuracy: 0.4615 - val_loss: 2.9919
Epoch 113/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 9s 2s/step - accuracy: 0.6523 - loss: 1.0537 - val_accuracy: 0.4038 - val_loss: 3.0943
Epoch 114/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 9s 1s/step - accuracy: 0.6647 - loss: 1.0078 - val_accuracy: 0.4231 - val_loss: 3.4016
Epoch 115/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 10s 2s/step - accuracy: 0.7121 - loss: 0.8414 - val_accuracy: 0.3077 - val_loss: 6.6408
Epoch 116/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 9s 2s/step - accuracy: 0.6875 - loss: 0.8882 - val_accuracy: 0.3269 - val_loss: 8.9623
Epoch 117/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 9s 1s/step - accuracy: 0.6400 - loss: 0.9894 - val_accuracy: 0.3462 - val_loss: 7.5568
Epoch 118/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 9s 2s/step - accuracy: 0.6412 - loss: 1.0102 - val_accuracy: 0.3462 - val_loss: 5.0186
Epoch 119/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 8s 2s/step - accuracy: 0.7265 - loss: 0.7774 - val_accuracy: 0.3462 - val_loss: 4.2880
Epoch 120/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 9s 1s/step - accuracy: 0.6738 - loss: 0.9919 - val_accuracy: 0.3077 - val_loss: 4.3184
Epoch 121/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 9s 2s/step - accuracy: 0.6572 - loss: 0.9162 - val_accuracy: 0.3269 - val_loss: 4.7128
Epoch 122/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 8s 2s/step - accuracy: 0.7061 - loss: 0.9878 - val_accuracy: 0.3462 - val_loss: 5.0318
Epoch 123/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 9s 2s/step - accuracy: 0.6764 - loss: 1.2228 - val_accuracy: 0.3269 - val_loss: 3.0861
Epoch 124/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 11s 2s/step - accuracy: 0.7205 - loss: 0.8134 - val_accuracy: 0.3462 - val_loss: 2.8717
Epoch 125/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 8s 1s/step - accuracy: 0.6972 - loss: 0.9484 - val_accuracy: 0.3462 - val_loss: 3.1085
Epoch 126/150
5/5 ━━━━━━━━━━━━━━━━━━━━ 9s 2s/step - accuracy: 0.6896 - loss: 0.8977 - val_accuracy: 0.3654 - val_loss: 2.7755
Epoch 127/150
```

```
import os
train_dir = "/content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/train"
class_names = sorted(os.listdir(train_dir))
print(class_names)
```

```
['Tomato___Bacterial_spot', 'Tomato___Early_blight', 'Tomato___Healthy', 'Tomato___Late_blight', 'Tomato___Leaf_Mold', 'Tomato___Septoria_leaf_spot',
```

```
from tensorflow.keras.preprocessing import image
import numpy as np

def predict_image(image_path, model):
    img = image.load_img(image_path, target_size=(128, 128))  # Load image correctly
    img = image.img_to_array(img)
    img = np.expand_dims(img, axis=0) / 255.0  # Normalize

    prediction = model.predict(img)
    class_idx = np.argmax(prediction)
```

```
    return class_names[class_idx]
```

```
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras.preprocessing import image

# Load the trained model
cnn_model = tf.keras.models.load_model("/content/tomato_leaf_disease_cnn.h5", compile=False)

# Ensure class names match training labels
class_names = [
    'Tomato___Bacterial_spot', 'Tomato___Early_blight', 'Tomato___Healthy',
    'Tomato___Late_blight', 'Tomato___Leaf_Mold', 'Tomato___Septoria_leaf_spot',
    'Tomato___Spider_mites Two-spotted_spider_mite', 'Tomato___Target_Spot',
    'Tomato___Tomato_Yellow_Leaf_Curl_Virus', 'Tomato___Tomato_mosaic_virus'
]

def preprocess_image(image_path):
    img = image.load_img(image_path, target_size=(128, 128))  # Resize to match training size
    img_array = image.img_to_array(img) / 255.0  # Normalize (same as training)
    img_array = np.expand_dims(img_array, axis=0)  # Expand dimensions for batch
    return img_array, img

def predict_image(image_path, model):
    img_array, img = preprocess_image(image_path)

    # Debug: Print shape and values
    print("Image Shape:", img_array.shape)
    print("Image Min-Max Values:", img_array.min(), img_array.max())

    # Predict
    prediction = model.predict(img_array)
    class_idx = np.argmax(prediction)  # Get highest probability index
    confidence = prediction[0][class_idx]  # Get confidence score

    return class_names[class_idx], confidence, img, prediction[0]

# Test an image
test_image_path = "/content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/val/Tomato___Bacterial_spot/Tomato___Bacterial_spot (1).JPG"

predicted_class, confidence, img, all_predictions = predict_image(test_image_path, cnn_model)

# Display Image & Classification
plt.imshow(img)
plt.axis('off')  # Hide axis
plt.title(f"Predicted: {predicted_class}\nConfidence: {confidence:.2f}")
plt.show()

# Print all class probabilities
for class_name, prob in zip(class_names, all_predictions):
    print(f"{class_name}: {prob:.4f}")
```

```
Image Shape: (1, 128, 128, 3)
Image Min-Max Values: 0.015686275 0.7764706
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 209ms/step
```


```
Tomato___Bacterial_spot: 0.8429
Tomato___Early_blight: 0.0000
Tomato___Healthy: 0.0002
Tomato___Late_blight: 0.0000
Tomato___Leaf_Mold: 0.0000
Tomato___Septoria_leaf_spot: 0.0000
Tomato___Spider_mites Two-spotted_spider_mite: 0.0000
Tomato___Target_Spot: 0.0000
Tomato___Tomato_Yellow_Leaf_Curl_Virus: 0.1568
Tomato___Tomato_mosaic_virus: 0.0000
```

```
"""
    "/content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/val/Tomato___Bacterial_spot/Tomato___Bacterial_spot (1).JPG",
    "/content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/val/Tomato___Early_blight/Tomato___Early_blight (1).JPG",
    "/content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/val/Tomato___Healthy/Tomato___healthy (1).JPG",
    "/content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/val/Tomato___Late_blight/Tomato___Late_blight (1).JPG",
    "/content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/val/Tomato___Leaf_Mold/Tomato___Leaf_Mold (1).JPG",
    "/content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/val/Tomato___Septoria_leaf_spot/Tomato___Septoria_leaf_spot (1).JPG",
    "/content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/val/Tomato___Spider_mites Two-spotted_spider_mite/Tomato___Spider_mites Two-spotted_spi
    "/content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/val/Tomato___Target_Spot/Tomato___Target_Spot (1).JPG",
    "/content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/val/Tomato___Tomato_Yellow_Leaf_Curl_Virus/Tomato___Tomato_Yellow_Leaf_Curl_Virus (1).J
    "/content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/val/Tomato___Tomato_mosaic_virus/Tomato___Tomato_mosaic_virus (1).JPG",
"""
```

```
'\n    "/content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/val/Tomato___Bacterial_spot/Tomato___Bacterial_spot (1).JPG",\n    "/content/da
taset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/val/Tomato___Early_blight/Tomato___Early_blight (1).JPG",\n    "/content/dataset/TOMATO LEAF DISEA
SE DETECTION/Dataset/Dataset/val/Tomato___Healthy/Tomato___healthy (1).JPG",\n    "/content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/val/
Tomato___Late_blight/Tomato___Late_blight (1).JPG",\n    "/content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/val/Tomato___Leaf_Mold/Tomato
   Leaf Mold (1).JPG",\n    "/content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/val/Tomato   Septoria leaf spot/Tomato   Septoria leaf spo
```

```
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras.preprocessing import image

# Load the trained model
cnn_model = tf.keras.models.load_model("/content/tomato_leaf_disease_cnn.h5", compile=False)

# Ensure class names match training labels
class_names = [
    'Tomato___Bacterial_spot', 'Tomato___Early_blight', 'Tomato___Healthy',
    'Tomato___Late_blight', 'Tomato___Leaf_Mold', 'Tomato___Septoria_leaf_spot',
    'Tomato___Spider_mites Two-spotted_spider_mite', 'Tomato___Target_Spot',
    'Tomato___Tomato_Yellow_Leaf_Curl_Virus', 'Tomato___Tomato_mosaic_virus'
]

def preprocess_image(image_path):
    img = image.load_img(image_path, target_size=(128, 128))  # Resize to match training size
    img_array = image.img_to_array(img) / 255.0  # Normalize (same as training)
    img_array = np.expand_dims(img_array, axis=0)  # Expand dimensions for batch
    return img_array, img

def predict_image(image_path, model):
    img_array, img = preprocess_image(image_path)

    # Debug: Print shape and values
    print("Image Shape:", img_array.shape)
```

```
    print("Image Min-Max Values:", img_array.min(), img_array.max())

    # Predict
    prediction = model.predict(img_array)
    class_idx = np.argmax(prediction)  # Get highest probability index
    confidence = prediction[0][class_idx]  # Get confidence score

    return class_names[class_idx], confidence, img, prediction[0]

# Test an image
test_image_path = "/content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/val/Tomato___Early_blight/Tomato___Early_blight (1).JPG"

predicted_class, confidence, img, all_predictions = predict_image(test_image_path, cnn_model)

# Display Image & Classification
plt.imshow(img)
plt.axis('off')  # Hide axis
plt.title(f"Predicted: {predicted_class}\nConfidence: {confidence:.2f}")
plt.show()

# Print all class probabilities
for class_name, prob in zip(class_names, all_predictions):
    print(f"{class_name}: {prob:.4f}")
```

```
Image Shape: (1, 128, 128, 3)
Image Min-Max Values: 0.0 0.76862746
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 175ms/step
```


```
Tomato___Bacterial_spot: 0.0116
Tomato___Early_blight: 0.6695
Tomato___Healthy: 0.0002
Tomato___Late_blight: 0.1181
Tomato___Leaf_Mold: 0.0003
Tomato___Septoria_leaf_spot: 0.0005
Tomato___Spider_mites Two-spotted_spider_mite: 0.0002
Tomato___Target_Spot: 0.0015
Tomato___Tomato_Yellow_Leaf_Curl_Virus: 0.1979
Tomato___Tomato_mosaic_virus: 0.0002
```

```
import pickle
import tensorflow as tf

# Load the trained model
cnn_model = tf.keras.models.load_model("/content/tomato_leaf_disease_cnn.h5", compile=False)

# Save the model as a pickle file
pickle_filename = "/content/Tomato_Leaf_Pickle.pkl"
with open(pickle_filename, "wb") as file:
    pickle.dump(cnn_model, file)

print(f"✅ Model saved as pickle file: {pickle_filename}")
```

```
import pickle
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras.preprocessing import image

# Load the pickle model
pickle_filename = "/content/Tomato_Leaf_Pickle.pkl"  # Update this path if needed
with open(pickle_filename, "rb") as file:
    loaded_model = pickle.load(file)
```

```
print("✅ Pickle model loaded successfully!")

# Ensure class names match training labels
class_names = [
    'Tomato___Bacterial_spot', 'Tomato___Early_blight', 'Tomato___Healthy',
    'Tomato___Late_blight', 'Tomato___Leaf_Mold', 'Tomato___Septoria_leaf_spot',
    'Tomato___Spider_mites Two-spotted_spider_mite', 'Tomato___Target_Spot',
    'Tomato___Tomato_Yellow_Leaf_Curl_Virus', 'Tomato___Tomato_mosaic_virus'
]

def preprocess_image(image_path):
    img = image.load_img(image_path, target_size=(128, 128))  # Resize to match training size
    img_array = image.img_to_array(img) / 255.0  # Normalize (same as training)
    img_array = np.expand_dims(img_array, axis=0)  # Expand dimensions for batch
    return img_array, img

def predict_image(image_path, model):
    img_array, img = preprocess_image(image_path)

    # Predict
    prediction = model.predict(img_array)
    class_idx = np.argmax(prediction)  # Get highest probability index
    confidence = prediction[0][class_idx]  # Get confidence score

    return class_names[class_idx], confidence, img, prediction[0]

# Example: Test on a user-provided image URL
image_url = "/content/dataset/TOMATO LEAF DISEASE DETECTION/Dataset/Dataset/val/Tomato___Leaf_Mold/Tomato___Leaf_Mold (1).JPG"

predicted_class, confidence, img, all_predictions = predict_image(image_url, loaded_model)

# Display result
plt.imshow(img)
plt.axis('off')
plt.title(f"Predicted: {predicted_class}\nConfidence: {confidence:.2f}")
plt.show()
```