In [1]:
```
pip install tensorflow numpy matplotlib
```

```
Requirement already satisfied: tensorflow in c:\users\shita\anaconda3\lib\site-packages (2.14.0)
Requirement already satisfied: numpy in c:\users\shita\anaconda3\lib\site-packages (1.24.3)
Requirement already satisfied: matplotlib in c:\users\shita\anaconda3\lib\site-packages (3.7.1)
Requirement already satisfied: tensorflow-intel==2.14.0 in c:\users\shita\anaconda3\lib\site-packages (from tensorflow) (2.14.0)
Requirement already satisfied: absl-py>=1.0.0 in c:\users\shita\anaconda3\lib\site-packages (from tensorflow-intel==2.14.0->tensorflow) (2.0.0)
Requirement already satisfied: astunparse>=1.6.0 in c:\users\shita\anaconda3\lib\site-packages (from tensorflow-intel==2.14.0->tensorflow) (1.6.3)
Requirement already satisfied: flatbuffers>=23.5.26 in c:\users\shita\anaconda3\lib\site-packages (from tensorflow-intel==2.14.0->tensorflow) (23.5.26)
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in c:\users\shita\anaconda3\lib\site-packages (from tensorflow-intel==2.14.0->tensorflow) (0.5.4)
Requirement already satisfied: google-pasta>=0.1.1 in c:\users\shita\anaconda3\lib\site-packages (from tensorflow-intel==2.14.0->tensorflow) (0.2.0)
Requirement already satisfied: h5py>=2.9.0 in c:\users\shita\anaconda3\lib\site-packages (from tensorflow-intel==2.14.0->tensorflow) (3.7.0)
Requirement already satisfied: libclang>=13.0.0 in c:\users\shita\anaconda3\lib\site-packages (from tensorflow-intel==2.14.0->tensorflow) (16.0.6)
Requirement already satisfied: ml-dtypes==0.2.0 in c:\users\shita\anaconda3\lib\site-packages (from tensorflow-intel==2.14.0->tensorflow) (0.2.0)
Requirement already satisfied: opt-einsum>=2.3.2 in c:\users\shita\anaconda3\lib\site-packages (from tensorflow-intel==2.14.0->tensorflow) (3.3.0)
Requirement already satisfied: packaging in c:\users\shita\appdata\roaming\python\python311\site-packages (from tensorflow-intel==2.14.0->tensorflow) (23.1)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<5.0.0dev,>=3.20.3 in c:\users\shita\anaconda3\lib\site-packages (from tensorflow-intel==2.14.0->tensorflow)
(4.24.4)
Requirement already satisfied: setuptools in c:\users\shita\anaconda3\lib\site-packages (from tensorflow-intel==2.14.0->tensorflow) (68.0.0)
Requirement already satisfied: six>=1.12.0 in c:\users\shita\appdata\roaming\python\python311\site-packages (from tensorflow-intel==2.14.0->tensorflow) (1.16.0)
Requirement already satisfied: termcolor>=1.1.0 in c:\users\shita\anaconda3\lib\site-packages (from tensorflow-intel==2.14.0->tensorflow) (2.3.0)
Requirement already satisfied: typing-extensions>=3.6.6 in c:\users\shita\anaconda3\lib\site-packages (from tensorflow-intel==2.14.0->tensorflow) (4.7.1)
Requirement already satisfied: wrapt<1.15,>=1.11.0 in c:\users\shita\anaconda3\lib\site-packages (from tensorflow-intel==2.14.0->tensorflow) (1.14.1)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in c:\users\shita\anaconda3\lib\site-packages (from tensorflow-intel==2.14.0->tensorflow) (0.31.0)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in c:\users\shita\anaconda3\lib\site-packages (from tensorflow-intel==2.14.0->tensorflow) (1.59.0)
Requirement already satisfied: tensorboard<2.15,>=2.14 in c:\users\shita\anaconda3\lib\site-packages (from tensorflow-intel==2.14.0->tensorflow) (2.14.1)
Requirement already satisfied: tensorflow-estimator<2.15,>=2.14.0 in c:\users\shita\anaconda3\lib\site-packages (from tensorflow-intel==2.14.0->tensorflow) (2.14.0)
Requirement already satisfied: keras<2.15,>=2.14.0 in c:\users\shita\anaconda3\lib\site-packages (from tensorflow-intel==2.14.0->tensorflow) (2.14.0)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\shita\anaconda3\lib\site-packages (from matplotlib) (1.0.5)
Requirement already satisfied: cycler>=0.10 in c:\users\shita\anaconda3\lib\site-packages (from matplotlib) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\shita\anaconda3\lib\site-packages (from matplotlib) (4.25.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\shita\anaconda3\lib\site-packages (from matplotlib) (1.4.4)
Requirement already satisfied: pillow>=6.2.0 in c:\users\shita\anaconda3\lib\site-packages (from matplotlib) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\shita\anaconda3\lib\site-packages (from matplotlib) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\shita\appdata\roaming\python\python311\site-packages (from matplotlib) (2.8.2)
Requirement already satisfied: wheel<1.0,>=0.23.0 in c:\users\shita\anaconda3\lib\site-packages (from astunparse>=1.6.0->tensorflow-intel==2.14.0->tensorflow) (0.38.4)
Requirement already satisfied: google-auth<3,>=1.6.3 in c:\users\shita\anaconda3\lib\site-packages (from tensorboard<2.15,>=2.14->tensorflow-intel==2.14.0->tensorflow) (2.23.3)
Requirement already satisfied: google-auth-oauthlib<1.1,>=0.5 in c:\users\shita\anaconda3\lib\site-packages (from tensorboard<2.15,>=2.14->tensorflow-intel==2.14.0->tensorflow) (1.0.0)
Requirement already satisfied: markdown>=2.6.8 in c:\users\shita\anaconda3\lib\site-packages (from tensorboard<2.15,>=2.14->tensorflow-intel==2.14.0->tensorflow) (3.4.1)
Requirement already satisfied: requests<3,>=2.21.0 in c:\users\shita\anaconda3\lib\site-packages (from tensorboard<2.15,>=2.14->tensorflow-intel==2.14.0->tensorflow) (2.31.0)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in c:\users\shita\anaconda3\lib\site-packages (from tensorboard<2.15,>=2.14->tensorflow-intel==2.14.0->tensorflow) (0.7.1)
Requirement already satisfied: werkzeug>=1.0.1 in c:\users\shita\anaconda3\lib\site-packages (from tensorboard<2.15,>=2.14->tensorflow-intel==2.14.0->tensorflow) (2.2.3)
Requirement already satisfied: cachetools<6.0,>=2.0.0 in c:\users\shita\anaconda3\lib\site-packages (from google-auth<3,>=1.6.3->tensorboard<2.15,>=2.14->tensorflow-intel==2.14.0->tensorflow) (5.3.1)
Requirement already satisfied: pyasn1-modules>=0.2.1 in c:\users\shita\anaconda3\lib\site-packages (from google-auth<3,>=1.6.3->tensorboard<2.15,>=2.14->tensorflow-intel==2.14.0->tensorflow) (0.2.8)
Requirement already satisfied: rsa<5,>=3.1.4 in c:\users\shita\anaconda3\lib\site-packages (from google-auth<3,>=1.6.3->tensorboard<2.15,>=2.14->tensorflow-intel==2.14.0->tensorflow) (4.9)
Requirement already satisfied: requests-oauthlib>=0.7.0 in c:\users\shita\anaconda3\lib\site-packages (from google-auth-oauthlib<1.1,>=0.5->tensorboard<2.15,>=2.14->tensorflow-intel==2.14.0->tensorflow)
(1.3.1)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\shita\anaconda3\lib\site-packages (from requests<3,>=2.21.0->tensorboard<2.15,>=2.14->tensorflow-intel==2.14.0->tensorflow) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\shita\anaconda3\lib\site-packages (from requests<3,>=2.21.0->tensorboard<2.15,>=2.14->tensorflow-intel==2.14.0->tensorflow) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\shita\anaconda3\lib\site-packages (from requests<3,>=2.21.0->tensorboard<2.15,>=2.14->tensorflow-intel==2.14.0->tensorflow) (1.26.16)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\shita\anaconda3\lib\site-packages (from requests<3,>=2.21.0->tensorboard<2.15,>=2.14->tensorflow-intel==2.14.0->tensorflow) (2023.7.22)
Requirement already satisfied: MarkupSafe>=2.1.1 in c:\users\shita\anaconda3\lib\site-packages (from werkzeug>=1.0.1->tensorboard<2.15,>=2.14->tensorflow-intel==2.14.0->tensorflow) (2.1.1)
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in c:\users\shita\anaconda3\lib\site-packages (from pyasn1-modules>=0.2.1->google-auth<3,>=1.6.3->tensorboard<2.15,>=2.14->tensorflow-intel==2.14.0->ten
sorflow) (0.4.8)
Requirement already satisfied: oauthlib>=3.0.0 in c:\users\shita\anaconda3\lib\site-packages (from requests-oauthlib>=0.7.0->google-auth-oauthlib<1.1,>=0.5->tensorboard<2.15,>=2.14->tensorflow-intel==2.1
4.0->tensorflow) (3.2.2)
Note: you may need to restart the kernel to use updated packages.
```

In [2]:
```
# Import necessary libraries
import tensorflow as tf
from tensorflow.keras import layers, models
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

In [3]:
```
# Define paths to your training and validation datasets
train_data_dir = "C:/Deep Learning/cotton/TRAIN"
validation_data_dir = "C:/Deep Learning/cotton/TEST"
```

In [4]:
```
# Set parameters
img_width, img_height = 224, 224
batch_size = 30
epochs = 10
```

In [5]:
```
# Data augmentation for training
train_datagen = ImageDataGenerator(
    rescale=1./255,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True
)
```

In [6]:
```
# Rescale validation data
validation_datagen = ImageDataGenerator(rescale=1./255)
```

In [7]:
```
# Load training data
train_generator = train_datagen.flow_from_directory(
    train_data_dir,
    target_size=(img_width, img_height),
    batch_size=batch_size,
    class_mode='categorical'
)
```

```
Found 1708 images belonging to 2 classes.
```

In [8]:
```
# Load validation data
validation_generator = validation_datagen.flow_from_directory(
    validation_data_dir,
    target_size=(img_width, img_height),
    batch_size=batch_size,
    class_mode='categorical'
)
```

```
Found 1708 images belonging to 2 classes.
```

In [9]:
```
# Define GoogLeNet (Inception) model
base_model = tf.keras.applications.InceptionV3(
    include_top=False,
    weights='imagenet',
    input_tensor=None,
    input_shape=(img_width, img_height, 3),
    pooling=None
)
```

In [10]:
```
# Freeze the layers of the base model
for layer in base_model.layers:
    layer.trainable = False
```

In [11]:
```
# Define the number of classes in your classification problem
num_classes = len(train_generator.class_indices)
```

In [12]:
```
# Create a custom model on top of the base model
model = models.Sequential([
    base_model,
    layers.GlobalAveragePooling2D(),
    layers.Dense(256, activation='relu'),
    layers.Dropout(0.5),
    layers.Dense(num_classes, activation='softmax')
])
```

In [13]:
```
# Compile the model
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

In [14]:
```
# Train the model
history = model.fit(
    train_generator,
    steps_per_epoch=train_generator.samples // batch_size,
    epochs=epochs,
    validation_data=validation_generator,
    validation_steps=validation_generator.samples // batch_size
)
```

```
Epoch 1/10
56/56 [==============================] - 410s 7s/step - loss: 0.4849 - accuracy: 0.8218 - val_loss: 0.1570 - val_accuracy: 0.9458
Epoch 2/10
56/56 [==============================] - 387s 7s/step - loss: 0.1806 - accuracy: 0.9327 - val_loss: 0.0981 - val_accuracy: 0.9649
Epoch 3/10
56/56 [==============================] - 383s 7s/step - loss: 0.1838 - accuracy: 0.9279 - val_loss: 0.0940 - val_accuracy: 0.9762
Epoch 4/10
56/56 [==============================] - 400s 7s/step - loss: 0.1168 - accuracy: 0.9529 - val_loss: 0.0565 - val_accuracy: 0.9804
Epoch 5/10
56/56 [==============================] - 419s 7s/step - loss: 0.0901 - accuracy: 0.9714 - val_loss: 0.0507 - val_accuracy: 0.9827
Epoch 6/10
56/56 [==============================] - 397s 7s/step - loss: 0.1014 - accuracy: 0.9613 - val_loss: 0.0464 - val_accuracy: 0.9863
Epoch 7/10
56/56 [==============================] - 431s 8s/step - loss: 0.0764 - accuracy: 0.9720 - val_loss: 0.0502 - val_accuracy: 0.9810
Epoch 8/10
56/56 [==============================] - 444s 8s/step - loss: 0.0854 - accuracy: 0.9690 - val_loss: 0.0363 - val_accuracy: 0.9857
Epoch 9/10
56/56 [==============================] - 331s 6s/step - loss: 0.0758 - accuracy: 0.9702 - val_loss: 0.0312 - val_accuracy: 0.9869
Epoch 10/10
56/56 [==============================] - 324s 6s/step - loss: 0.0691 - accuracy: 0.9785 - val_loss: 0.0493 - val_accuracy: 0.9815
```

In [15]:
```
# Evaluate the model
evaluation = model.evaluate(validation_generator)
print(f"Validation Accuracy: {evaluation[1] * 100:.2f}%")
```

```
57/57 [==============================] - 170s 3s/step - loss: 0.0493 - accuracy: 0.9819
Validation Accuracy: 98.19%
```

In [16]:
```
# Save the model
model.save('cotton_disease_model.h5')
```

```
C:\Users\shita\anaconda3\Lib\site-packages\keras\src\engine\training.py:3079: UserWarning: You are saving your model as an HDF5 file via `model.save()`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')`.
  saving_api.save_model(
```

In [60]:
```
from tensorflow.keras.preprocessing import image
from tensorflow.keras.applications.inception_v3 import InceptionV3, preprocess_input, decode_predictions
import numpy as np
```

In [61]:
```
# Load the pre-trained GoogLeNet (Inception) model
model = InceptionV3(weights='imagenet')
```

In [84]:
```
# Load and preprocess an image for prediction
img_path = "C:/Deep Learning/cotton/TEST/test/5586201.jpg"
img = image.load_img(img_path, target_size=(299, 299))  # GoogLeNet input size
img_array = image.img_to_array(img)
img_array = np.expand_dims(img_array, axis=0)
img_array = preprocess_input(img_array)
```

In [85]:
```
# Make predictions
predictions = model.predict(img_array)
```

```
1/1 [==============================] - 4s 4s/step
```

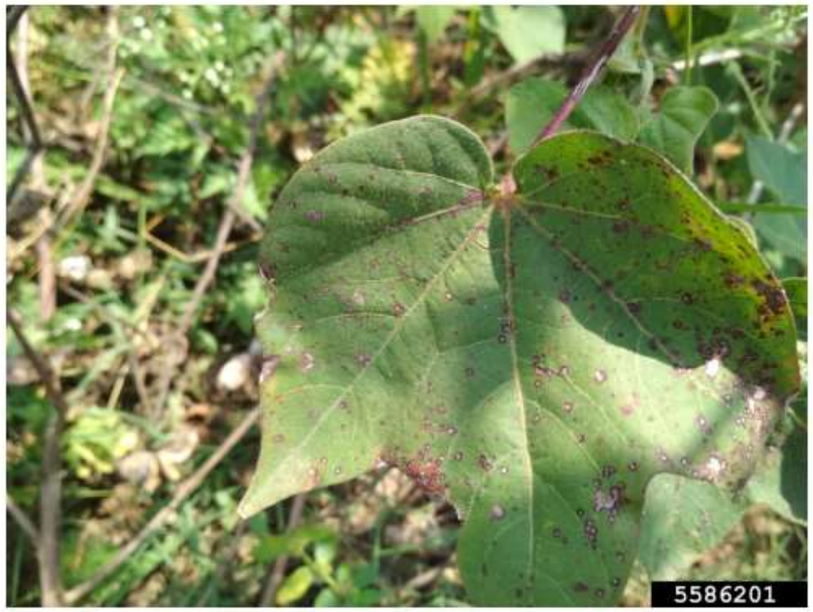
In [86]:
```python
model = tf.keras.models.load_model('cotton_disease_model.h5')
```

In [87]:
```python
# Function to preprocess an image for prediction
def preprocess_image(img_path):
    img = image.load_img(img_path, target_size=(224, 224))
    img_array = image.img_to_array(img)
    img_array = np.expand_dims(img_array, axis=0)
    img_array /= 255.0  # Rescale to [0, 1]
    return img_array
```

In [88]:
```python
import matplotlib.pyplot as plt
```

In [89]:
```python
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras.preprocessing import image

# Load the trained model
model = tf.keras.models.load_model('cotton_disease_model.h5')

# Function to preprocess an image for prediction
def preprocess_image(img_path):
    img = image.load_img(img_path, target_size=(224, 224))
    img_array = image.img_to_array(img)
    img_array = np.expand_dims(img_array, axis=0)
    img_array /= 255.0  # Rescale to [0, 1]
    return img_array

# Function to make predictions and display results
def predict_and_display(image_path):
    # Preprocess the image
    img = preprocess_image(image_path)

    # Make predictions
    predictions = model.predict(img)
    class_labels = ['Diseased', 'Healthy']

    # Display the original image
    img = image.load_img(image_path)
    plt.imshow(img)
    plt.axis('off')
    plt.show()

    # Display prediction results
    print("Predictions:")
    for i in range(len(class_labels)):
        print(f"{class_labels[i]}: {predictions[0][i]:.4f}")

    predicted_label = class_labels[np.argmax(predictions)]
    print(f"Predicted Label: {predicted_label}")

# Example usage
image_path ="C:/Deep Learning/cotton/TEST/test/5586201.jpg"
predict_and_display(image_path)
```

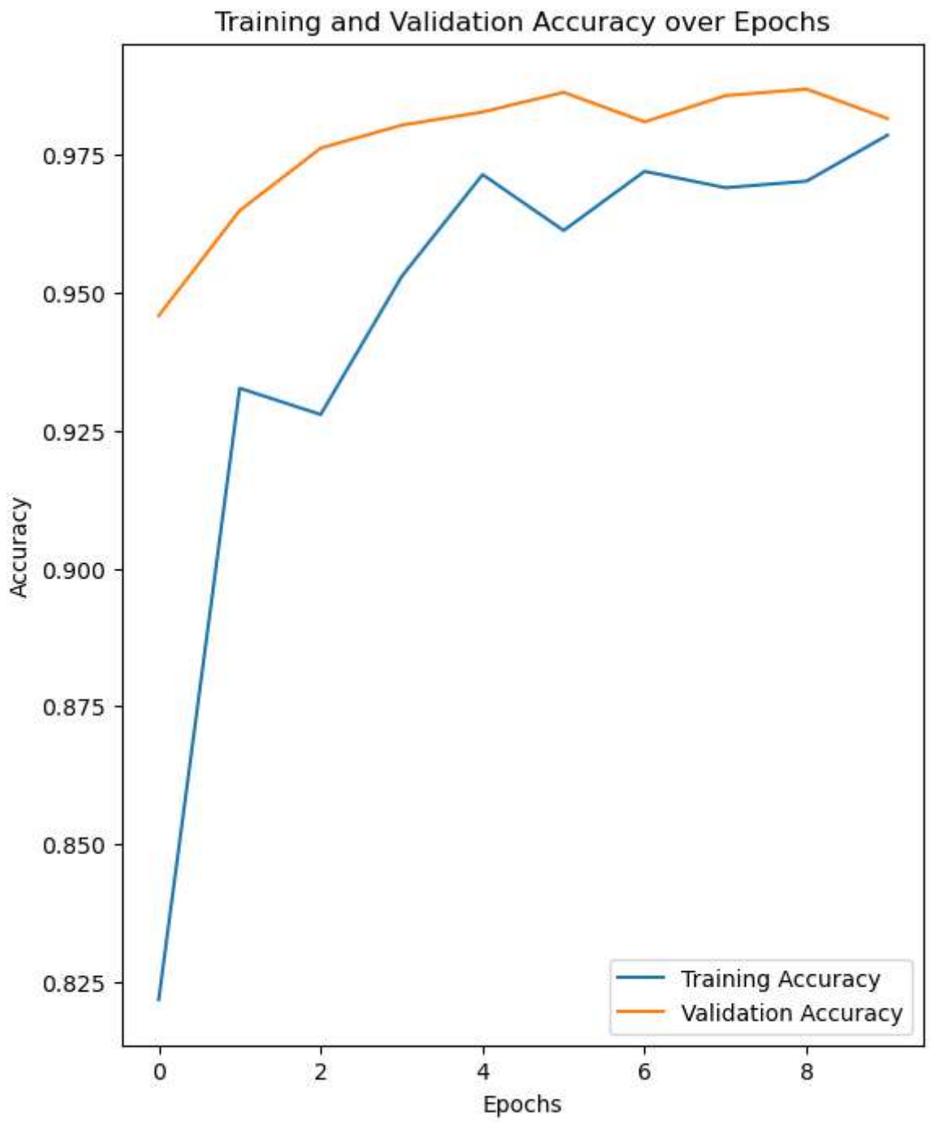
```
1/1 [==============================] - 3s 3s/step
```




```
Predictions:
Diseased: 0.9986
Healthy: 0.0014
Predicted Label: Diseased
```

In [106…

```python
history = history

# Accessing the training and validation accuracy
train_acc = history.history['accuracy']
val_acc = history.history['val_accuracy']

# Number of epochs
EPOCHS = len(train_acc)

# Plotting the accuracy
plt.figure(figsize=(14, 8))
plt.subplot(1, 2, 1)
plt.plot(range(EPOCHS), train_acc, label='Training Accuracy')
plt.plot(range(EPOCHS), val_acc, label='Validation Accuracy')
plt.legend(loc='lower right')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.title('Training and Validation Accuracy over Epochs')
plt.show()
```



In [90]:
```
from sklearn.metrics import confusion_matrix, classification_report
```

In [91]:
```
# Define paths to your test dataset
test_data_dir = "C:/Deep Learning/cotton/TEST"
```

In [93]:
```
# Set parameters
img_width, img_height = 299, 299  # GoogLeNet input size
batch_size = 30
```

In [94]:
```
# Data preprocessing for testing
test_datagen = image.ImageDataGenerator(preprocessing_function=preprocess_input)
```

In [95]:
```
# Load testing data
test_generator = test_datagen.flow_from_directory(
    test_data_dir,
    target_size=(img_width, img_height),
    batch_size=batch_size,
    shuffle=False,  # Important: Keep the order of predictions consistent with ground truth
    class_mode='categorical'
)
```

```
Found 1708 images belonging to 2 classes.
```

In [96]:
```
# Make predictions on the test set
predictions = model.predict(test_generator)
```

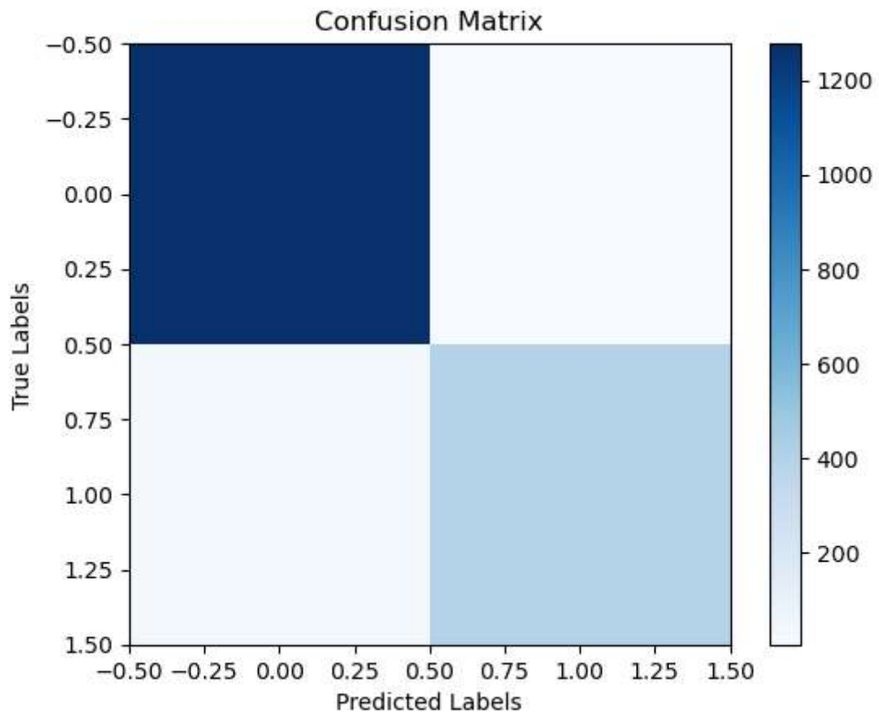
```
57/57 [==============================] - 350s 6s/step
```

In [97]:
```
# Decode predictions and ground truth labels
predicted_labels = np.argmax(predictions, axis=1)
true_labels = test_generator.classes
```

In [98]:
```
# Compute confusion matrix
cm = confusion_matrix(true_labels, predicted_labels)
```

In [99]:
```
# Plot confusion matrix
plt.imshow(cm, interpolation='nearest', cmap=plt.cm.Blues)
plt.title('Confusion Matrix')
plt.colorbar()
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
```

Out[99]:
```
Text(0, 0.5, 'True Labels')
```



In [105…
```
# Print classification report
print("Classification Report:")
print(classification_report(true_labels, predicted_labels))
```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.98      1.00      0.99      1283
           1       0.99      0.94      0.96       425

    accuracy                           0.98      1708
   macro avg       0.98      0.97      0.97      1708
weighted avg       0.98      0.98      0.98      1708
```