In [4]:
```python
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split

import os
directory = r"C:\Users\matim\Downloads\Agritech project"
print("Files in directory:")
for file in os.listdir(directory):
    print(file)


# Load the dataset
file_path = r"C:\Users\matim\Downloads\Agritech project\Complete_Dataset - Merged_Dataset.csv.csv"
df = pd.read_csv(file_path)

# Display basic information
print("Dataset shape:", df.shape)
print("\nFirst few rows:")
df.head()

# Display basic information about the dataset
print("Dataset shape:", df.shape)
print("\nFirst few rows:")
df.head()
```

```
Files in directory:
.git
.ipynb_checkpoints
Automated Data Collection and Organisation.ipynb
Complete_Dataset - Merged_Dataset.csv.csv
Grapevine_Annotated_Dataset
Grapevine_Annotated_Dataset.zip
image_metadata.csv
organized_dataset
raw_dataset
Untitled.ipynb
Dataset shape: (99, 25)

First few rows:
Dataset shape: (99, 25)

First few rows:
```

Out[4]:

	Leaf_ID	Image_Name	Leaf_Condition	Z_order	Width	Height	Source	Occluded	Spot Color	Spot Shape	...	Yellowing of Leaves	Leaf_Color	Leaf Texture	Leaf Shape	Vine vitality	Absence of Spots	Discoloration Pattern	Leaf Curling	Presence of Lesions
0	2	BlackRot_3.JPG	BR_Advanced	0	256	256	semi-auto	0	Back	Irregular	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	3	BlackRot_4.JPG	BR_Advanced	0	256	256	semi-auto	0	Black	Irregular	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	4	BlackRot_5.JPG	BR_Advanced	0	256	256	semi-auto	0	Black	Irregular	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	5	BlackRot_6.JPG	BR_Advanced	0	256	256	semi-auto	0	Black	Round	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	6	BlackRot_7.JPG	BR_Advanced	0	256	256	semi-auto	0	Black	Round	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

5 rows × 25 columns

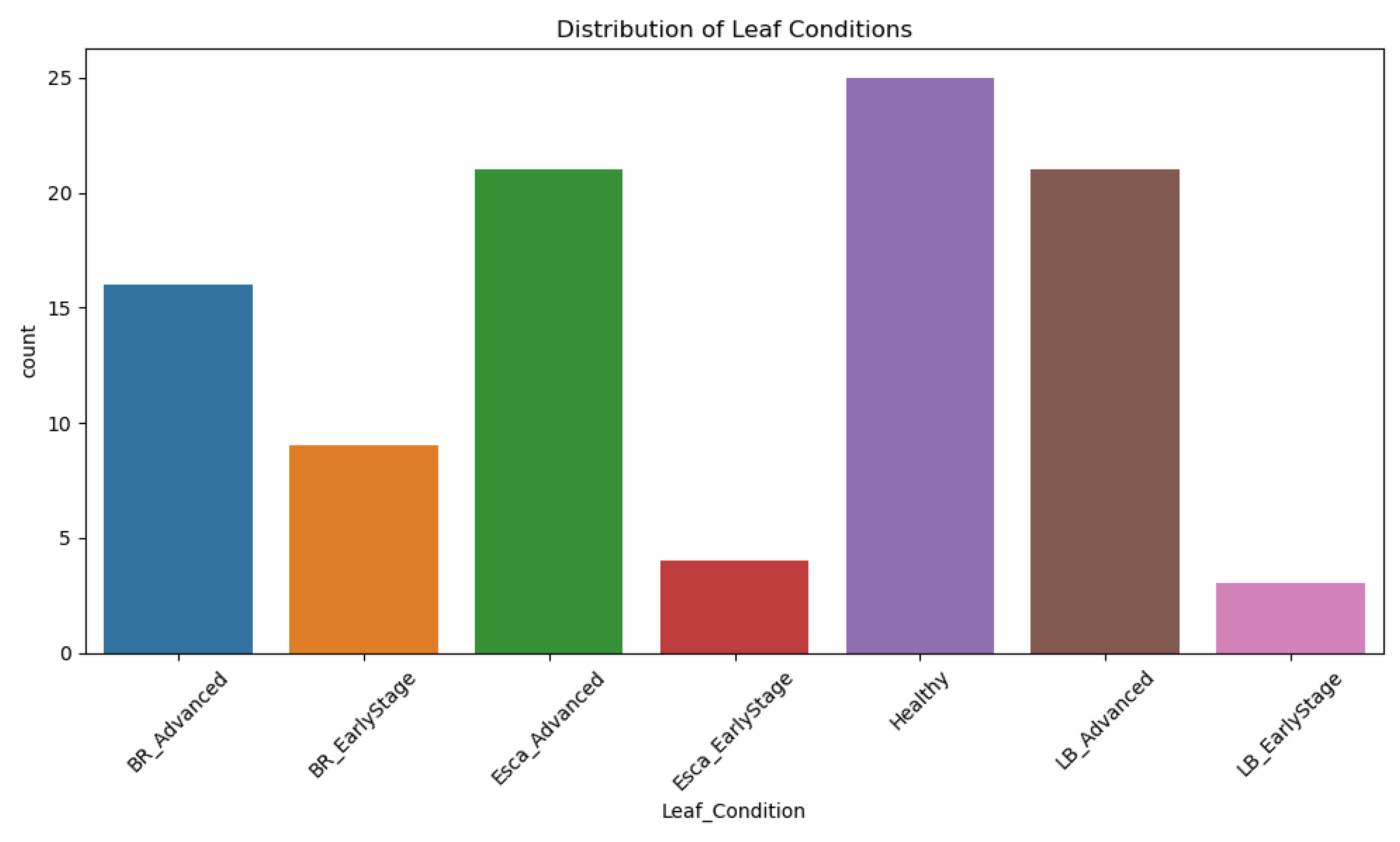
In [5]:
```python
#Exploratory Data Analysis

# Display information about the dataset
print("\nDataset Info:")
df.info()

# Display statistical summary
print("\nStatistical Summary:")
df.describe()

# Check for missing values
print("\nMissing values in each column:")
df.isnull().sum()
```

```
Dataset Info:
&lt;class 'pandas.core.frame.DataFrame'&gt;
RangeIndex: 99 entries, 0 to 98
Data columns (total 25 columns):
 #   Column                      Non-Null Count  Dtype  
---  ------                      --------------  -----  
 0   Leaf_ID                     99 non-null     int64  
 1   Image_Name                  99 non-null     object 
 2   Leaf_Condition              99 non-null     object 
 3   Z_order                     99 non-null     int64  
 4   Width                       99 non-null     int64  
 5   Height                      99 non-null     int64  
 6   Source                      99 non-null     object 
 7   Occluded                    99 non-null     int64  
 8   Spot Color                  16 non-null     object 
 9   Spot Shape                  25 non-null     object 
 10  Leaf Area Affected(%)       49 non-null     float64
 11  Presence of Fungal Growth   25 non-null     object 
 12  Spot_Color                  9 non-null      object 
 13  Leaf Vein Color             25 non-null     object 
 14  Visible White/Black Growth  25 non-null     object 
 15  Yellowing of Leaves         25 non-null     object 
 16  Leaf_Color                  25 non-null     object 
 17  Leaf Texture                25 non-null     object 
 18  Leaf Shape                  25 non-null     object 
 19  Vine vitality               25 non-null     object 
 20  Absence of Spots            25 non-null     object 
 21  Discoloration Pattern       24 non-null     object 
 22  Leaf Curling                24 non-null     object 
 23  Presence of Lesions         24 non-null     object 
 24  LeafAnnotated_points        99 non-null     object 
dtypes: float64(1), int64(5), object(19)
memory usage: 19.5+ KB

Statistical Summary:

Missing values in each column:
```

Out[5]:

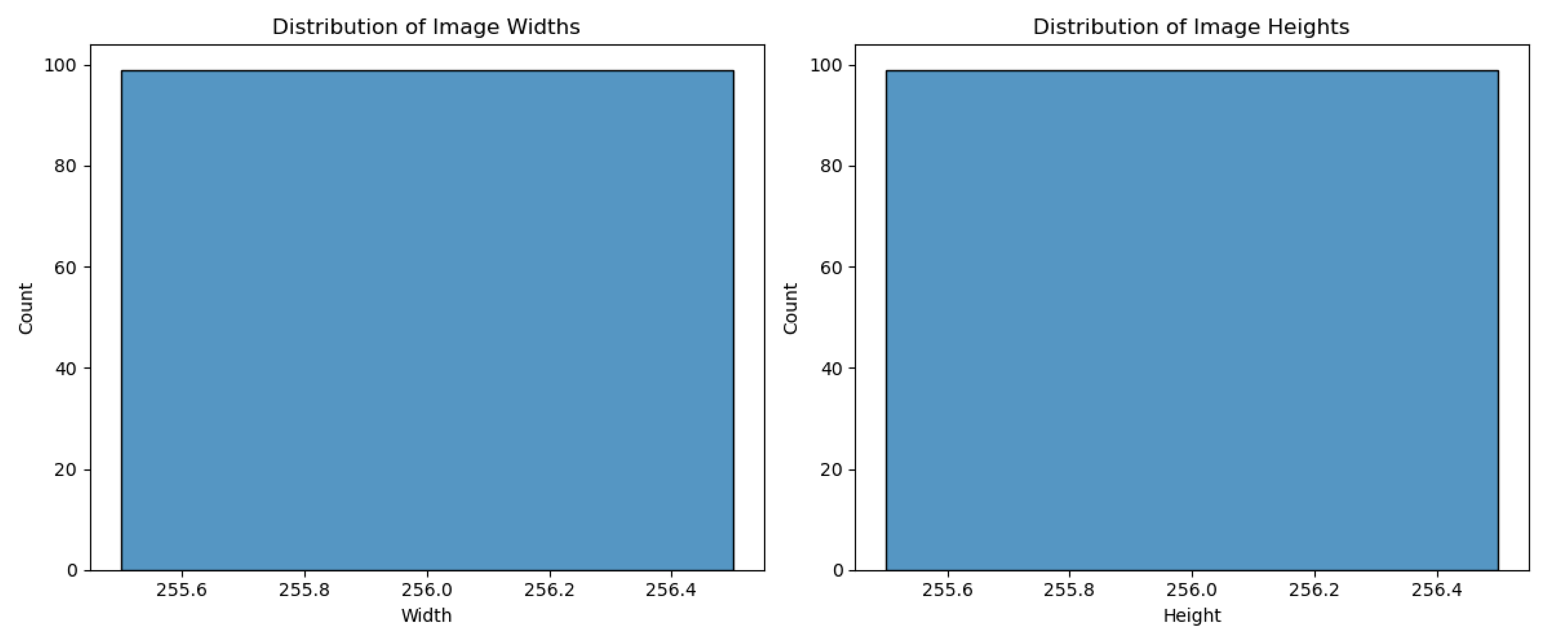
```
Leaf_ID                        0
Image_Name                     0
Leaf_Condition                 0
Z_order                        0
Width                          0
Height                         0
Source                         0
Occluded                       0
Spot Color                    83
Spot Shape                    74
Leaf Area Affected(%)         50
Presence of Fungal Growth     74
Spot_Color                    90
Leaf Vein Color               74
Visible White/Black Growth    74
Yellowing of Leaves           74
Leaf_Color                    74
Leaf Texture                  74
Leaf Shape                    74
Vine vitality                 74
Absence of Spots              74
Discoloration Pattern         75
Leaf Curling                  75
Presence of Lesions           75
LeafAnnotated_points           0
dtype: int64
```

In [6]:
```python
# Count the distribution of Leaf_Condition
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x='Leaf_Condition')
plt.xticks(rotation=45)
plt.title('Distribution of Leaf Conditions')
plt.tight_layout()
plt.show()

# Print the counts
print("\nLeaf Condition Distribution:")
print(df['Leaf_Condition'].value_counts())
```

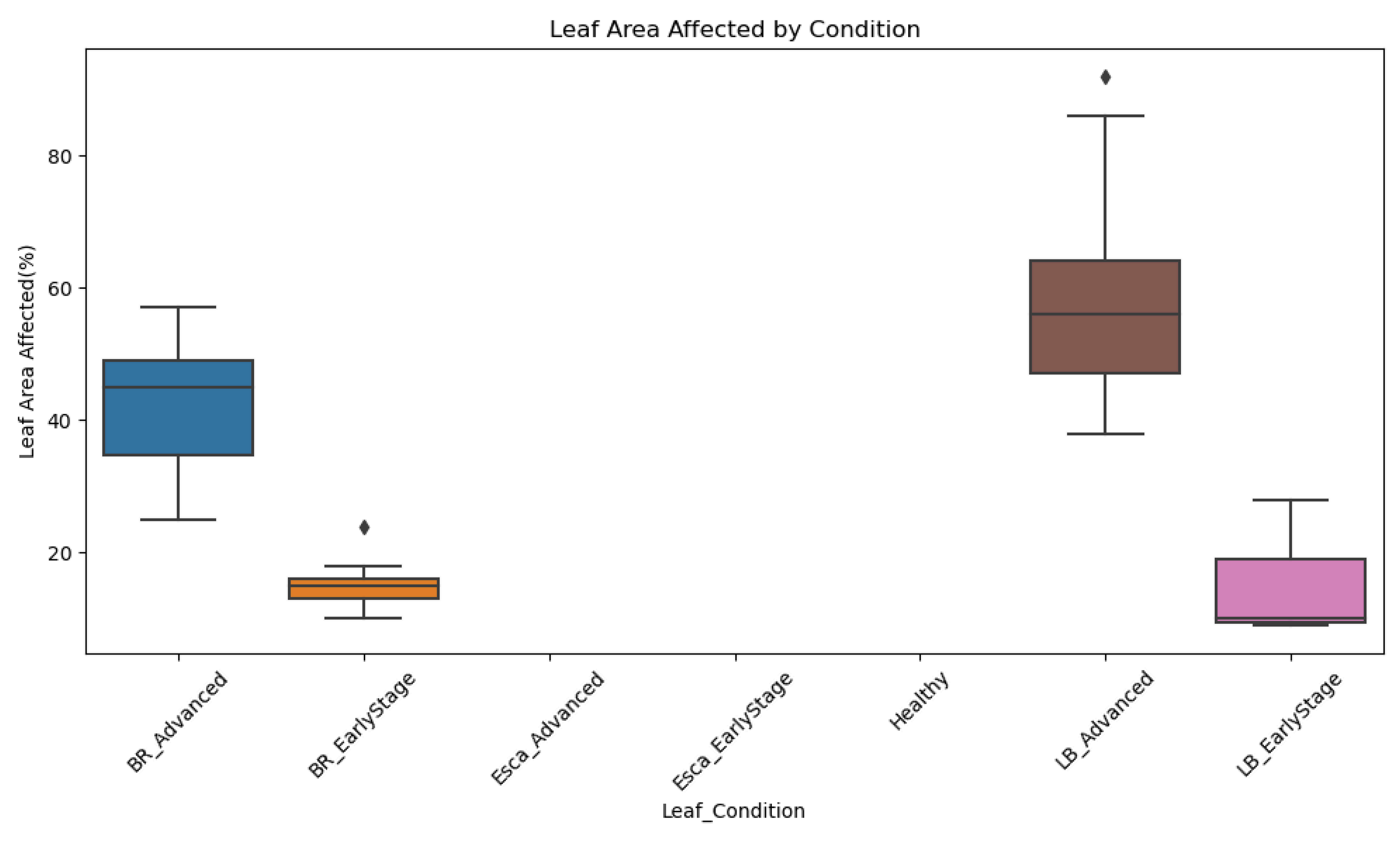



```
Leaf Condition Distribution:
Leaf_Condition
Healthy            25
Esca_Advanced      21
LB_Advanced        21
BR_Advanced        16
BR_EarlyStage       9
Esca_EarlyStage     4
LB_EarlyStage       3
Name: count, dtype: int64
```

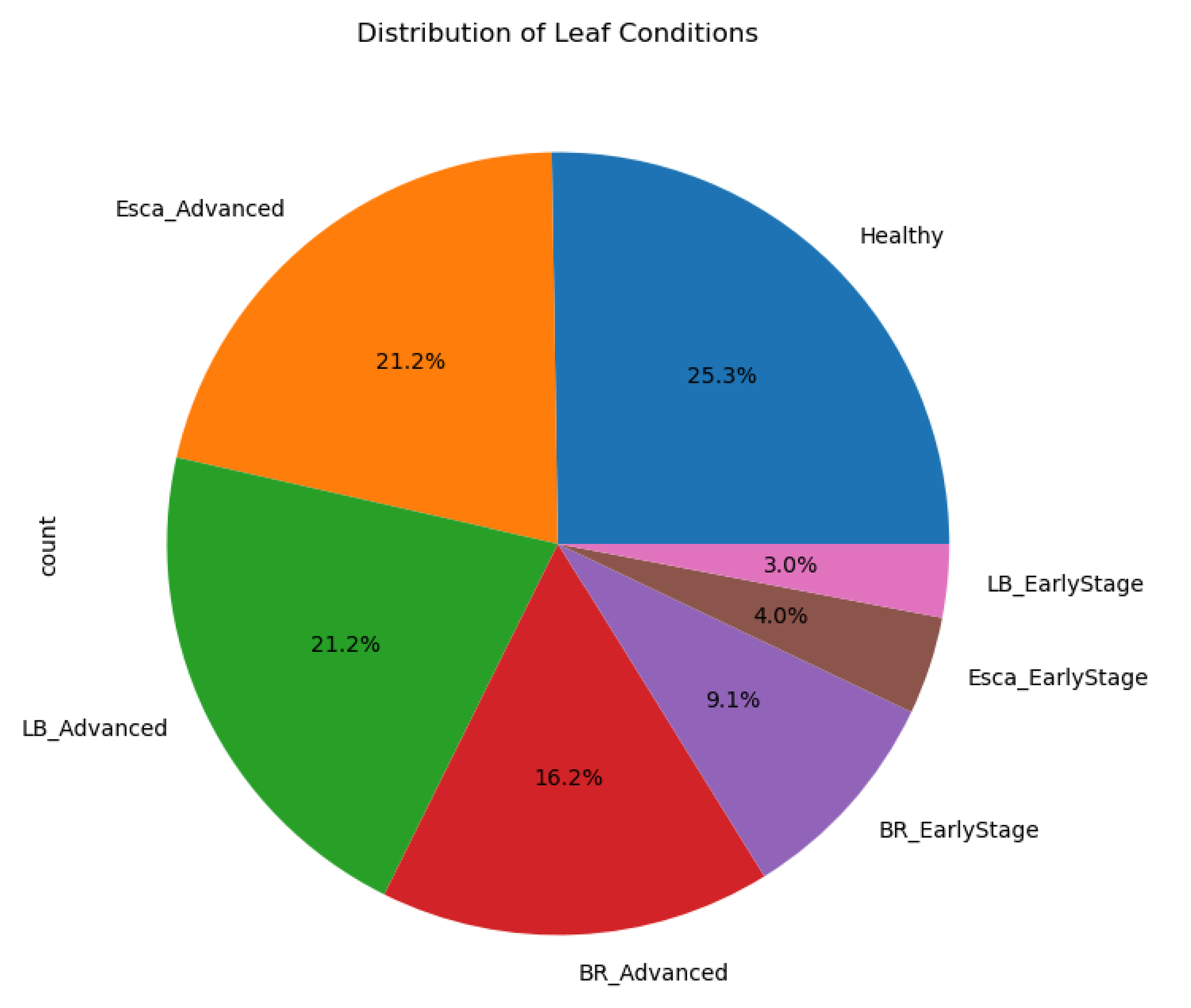
In [7]:
```python
# Analyze the image dimensions
plt.figure(figsize=(12, 5))

plt.subplot(1, 2, 1)
sns.histplot(data=df, x='Width')
plt.title('Distribution of Image Widths')

plt.subplot(1, 2, 2)
sns.histplot(data=df, x='Height')
plt.title('Distribution of Image Heights')

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



In [8]:
```python
# Distribution of Leaf Area Affected
plt.figure(figsize=(10, 6))
sns.boxplot(x='Leaf_Condition', y='Leaf Area Affected(%)', data=df)
plt.xticks(rotation=45)
plt.title('Leaf Area Affected by Condition')
plt.tight_layout()
plt.show()
```



In [11]:
```python
# Pie chart for leaf conditions
plt.figure(figsize=(10, 8))
df['Leaf_Condition'].value_counts().plot(kind='pie', autopct='%1.1f%%')
plt.title('Distribution of Leaf Conditions')
plt.show()
```



In [13]:
```python
# Pie chart code for missing vs. available data
def create_missing_data_pie(column_name, missing_count, total_count):
    labels = ['Available', 'Missing']
    sizes = [total_count - missing_count, missing_count]
    colors = ['#66b3ff', '#ff9999']

    plt.figure(figsize=(8, 6))
    plt.pie(sizes, labels=labels, colors=colors, autopct='%1.1f%%')
    plt.title(f'Data Availability: {column_name}')
    plt.axis('equal')
    plt.show()
```

In [9]:
```python
# Create metadata summary
metadata = {
    'total_samples': len(df),
    'conditions': df['Leaf_Condition'].unique().tolist(),
    'image_dimensions': {
        'width_range': f"{df['Width'].min()} to {df['Width'].max()}",
        'height_range': f"{df['Height'].min()} to {df['Height'].max()}"
    },
    'missing_value_summary': {
        column: {
            'missing_count': int(missing),
            'missing_percentage': f"{(missing/len(df))*100:.2f}%"
        }
        for column, missing in df.isnull().sum().items()
        if missing > 0
    }
}

print("\nDataset Metadata:")
for key, value in metadata.items():
    print(f"\n{key}:")
    print(value)
```

```
Dataset Metadata:

total_samples:
99

conditions:
['BR_Advanced', 'BR_EarlyStage', 'Esca_Advanced', 'Esca_EarlyStage', 'Healthy', 'LB_Advanced', 'LB_EarlyStage']

image_dimensions:
{'width_range': '256 to 256', 'height_range': '256 to 256'}

missing_value_summary:
{'Spot Color': {'missing_count': 83, 'missing_percentage': '83.84%'}, 'Spot Shape': {'missing_count': 74, 'missing_percentage': '74.75%'}, 'Leaf Area Affected
(%)': {'missing_count': 50, 'missing_percentage': '50.51%'}, 'Presence of Fungal Growth': {'missing_count': 74, 'missing_percentage': '74.75%'}, 'Spot_Color':
{'missing_count': 90, 'missing_percentage': '90.91%'}, 'Leaf Vein Color': {'missing_count': 74, 'missing_percentage': '74.75%'}, 'Visible White/Black Growth':
{'missing_count': 74, 'missing_percentage': '74.75%'}, 'Yellowing of Leaves': {'missing_count': 74, 'missing_percentage': '74.75%'}, 'Leaf_Color': {'missing_c
ount': 74, 'missing_percentage': '74.75%'}, 'Leaf Texture': {'missing_count': 74, 'missing_percentage': '74.75%'}, 'Leaf Shape': {'missing_count': 74, 'missin
g_percentage': '74.75%'}, 'Vine vitality': {'missing_count': 74, 'missing_percentage': '74.75%'}, 'Absence of Spots': {'missing_count': 74, 'missing_percentag
e': '74.75%'}, 'Discoloration Pattern': {'missing_count': 75, 'missing_percentage': '75.76%'}, 'Leaf Curling': {'missing_count': 75, 'missing_percentage': '7
5.76%'}, 'Presence of Lesions': {'missing_count': 75, 'missing_percentage': '75.76%'}}
```

In [10]:
```python
# First, let's identify which features we want to keep
# We'll focus on columns with complete or near-complete data
complete_columns = df.columns[df.isnull().sum() < len(df)/2].tolist()
print("\nColumns with less than 50% missing values:")
print(complete_columns)

# Create a clean dataset with selected features
df_clean = df[complete_columns]

# Split the data (60% train, 20% validation, 20% test)
X = df_clean.drop(['Leaf_Condition', 'Image_Name', 'LeafAnnotated_points'], axis=1)  # adjust features as needed
y = df_clean['Leaf_Condition']

# First split: separate test set
X_temp, X_test, y_temp, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Second split: create validation set
X_train, X_val, y_train, y_val = train_test_split(X_temp, y_temp, test_size=0.25, random_state=42)  # 0.25 of 0.8 = 0.2

print("\nData split sizes:")
print(f"Training set: {len(X_train)} samples")
print(f"Validation set: {len(X_val)} samples")
print(f"Test set: {len(X_test)} samples")
```

```
Columns with less than 50% missing values:
['Leaf_ID', 'Image_Name', 'Leaf_Condition', 'Z_order', 'Width', 'Height', 'Source', 'Occluded', 'LeafAnnotated_points']

Data split sizes:
Training set: 59 samples
Validation set: 20 samples
Test set: 20 samples
```

In [ ]: