Enchanted Wings: Marvels of Butterfly Species

Team ID: LTVIP2025TMID35136

Team size: 4

Team Leader: Koyyalagadda Poojitha

Team member: M Pujitha

Team member: Sailaja Rajana

Team member: Pavani Jalla

Project Overview: Enchanted Wings – Marvels of Butterfly Species

The **"Enchanted Wings"** project aims to develop a deep learning-based butterfly species classification system using **transfer learning techniques**. It leverages a curated dataset of **6,499 butterfly images across 75 distinct species**, applying pre-trained Convolutional Neural Networks (CNNs) to enable rapid and accurate species identification.

The system is designed for **real-time image classification** and is capable of running through a **Flask web application**, offering a user-friendly interface for researchers, educators, and enthusiasts to upload butterfly images and receive instant classification results along with confidence scores.

This project significantly supports biodiversity conservation efforts, ecological research, and public education by providing an automated, scalable, and accessible solution for butterfly species recognition.

Scenario 1: Biodiversity Monitoring

In the context of biodiversity monitoring, a butterfly image classification system based on transfer learning can contribute significantly. Field researchers and conservationists can use this system to quickly identify butterfly species in diverse habitats. By capturing images in the field, the system identifies butterflies in real-time, aiding in species inventory, population studies, and habitat management efforts. This facilitates data-driven conservation strategies and promotes ecosystem health monitoring.

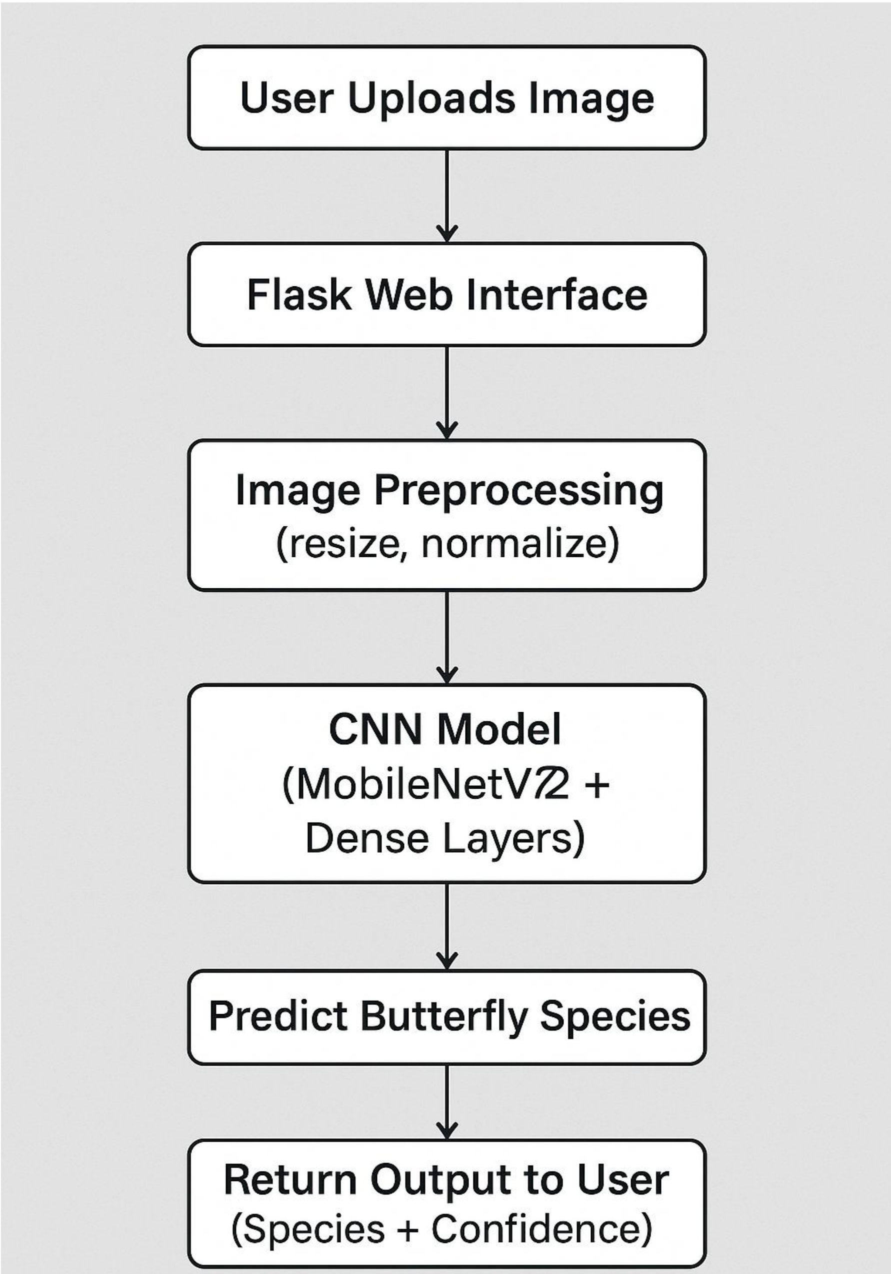
Scenario 2: Ecological Research

For ecological research, especially studies on butterfly behaviour and distribution patterns, automated image classification systems are invaluable. Researchers can deploy cameras equipped with the classification system to monitor butterfly activities over extended periods. This enables tracking of migratory patterns, habitat preferences, and responses to environmental changes. The system's ability to accurately classify butterflies supports scientific discoveries and informs conservation practices aimed at preserving vulnerable species

Scenario 3: Citizen Science and Education

Educational initiatives and citizen science projects benefit from interactive butterfly classification tools. These tools engage enthusiasts and students in butterfly identification and data collection. Users can capture butterfly images using mobile devices, and the classification system instantly provides species identification and relevant educational information. Such tools promote environmental awareness, citizen participation in scientific research, and foster a deeper understanding of butterfly ecology and conservation.

Technical Architecture:



Project flow:

The system is designed to take a butterfly image from the user, process it through a deep learning model, and return the predicted species—all within a few seconds.

1. Image Upload by User

- The user visits a **web interface** (built using Flask).
- They upload an image of a butterfly (.jpg, .png, etc.).
- The image is sent to the backend via a POST request.

2. Image Preprocessing

- The image is read and resized to **224x224 pixels** to match model input size.
- It is converted into a **NumPy array** and normalized (scaled between 0 and 1).

3. Model Prediction (CNN Inference)

- The image is passed into a **Convolutional Neural Network (CNN)** based on **MobileNetV2**.
- The model uses pre-trained ImageNet weights and fine-tuned custom layers.
- It outputs a **softmax vector** with probabilities for all 75 butterfly classes.

4. Mapping Prediction to Species

- The highest softmax score is selected using np.argmax().
- The corresponding class label is decoded using the **label encoder**.
- The result is a species name and confidence score.

5. Displaying the Result

- The Flask app renders a **results page** (result.html) with:
    - Predicted butterfly species
    - Confidence level

Software, Concepts, and Packages Required

🔹 **Development Environment**

Anaconda Navigator is a free and open-source distribution of the Python programming language used extensively in machine learning and data science. It supports multiple operating systems including Windows, Linux, and macOS.

- Conda: An open-source package and environment manager used to install, run, and update packages and dependencies.
- Anaconda comes pre-installed with several essential tools such as:
    - Jupyter Notebook – Ideal for prototyping and interactive code execution.
    - VS Code – A powerful source code editor used for building and debugging applications.
    - Spyder, RStudio, Orange, Glueviz – Additional tools for data science.

🔧 For this project, we recommend using Jupyter Notebook for training and experimentation, and VS Code for integrating the model with a Flask web application.

Core Concepts Required

🔸 **Deep Learning Concepts**

- **Convolutional Neural Network (CNN)**
  A type of deep neural network widely used for image classification tasks. In this project, a pre-trained **MobileNetV2** model is used and fine-tuned for butterfly species recognition.
- **Transfer Learning**
  A method where a model trained on one task (e.g., ImageNet classification) is adapted for a new, related task with less training data. It helps in reducing training time and improving accuracy.

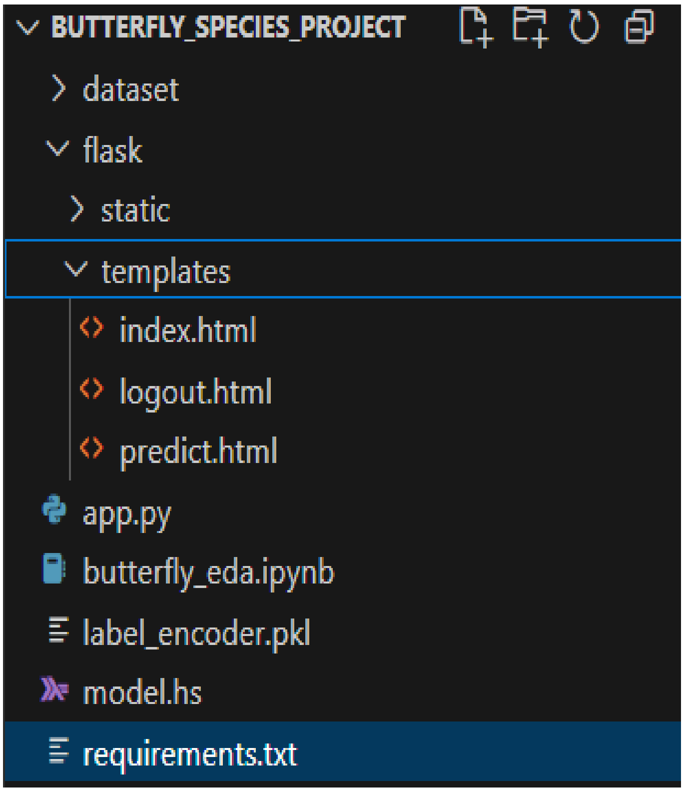
🔸 **Web Development**

- **Flask**
  A lightweight Python web framework used to build the project's user interface, enabling users to upload butterfly images and receive predictions in real-time.

Project Objectives:

1. **Develop an Image Classification Model**
    - Create a robust deep learning model capable of classifying butterfly species from images.
    - Utilize a curated dataset containing **6,499 images from 75 butterfly classes**.
2. **Apply Transfer Learning**
    - Use a pre-trained CNN (e.g., **MobileNetV2**) to leverage existing knowledge from large image datasets.
    - Fine-tune the model to recognize butterfly-specific features, reducing the need for extensive training.
3. **Preprocess and Augment Image Data**
    - Implement preprocessing steps like **resizing**, **normalization**, and **augmentation** to enhance data quality and model generalization.
4. **Build a Flask-based Web Application**
    - Design an intuitive **web interface** that allows users to upload butterfly images.
    - Integrate the trained model into a Flask backend to provide real-time predictions.
5. **Deliver High Accuracy and Fast Inference**
    - Optimize the model to provide predictions with **high accuracy and low latency**.
    - Display species name and prediction confidence clearly on the frontend.
6. **Support Research and Conservation**
    - Enable use cases such as **biodiversity monitoring**, **ecological research**, and **citizen science** participation.
7. **Encourage Environmental Awareness**
    - Promote interactive learning and engagement by making butterfly identification accessible to enthusiasts and students.

Project Structure



API Documentation

The Flask backend of this butterfly classification project exposes two core endpoints that power the web interface shown in the application.

/ – Homepage Endpoint

· **Method: GET**

· **Purpose: Loads the main interface where users can upload butterfly images.**

· **Parameters: None**

· **Response: Renders index.html, which contains:**

- **A file input (Choose File)**
- **A submit button labeled "Predict Butterfly"**
- **Project title and team member details displayed on screen**

/predict – Prediction Endpoint

- **Method: POST**
- **Purpose: Receives the uploaded image, performs preprocessing, runs the trained VGG16 model, and predicts the butterfly species.**
- **Parameters:**
- **image: Image file uploaded by the user (from the HTML form)**
- **Response: Renders output.html displaying the predicted species.**

Authentication

This project does not include authentication or authorization features, as it is designed for open, single-user access. The focus of the system is on classifying butterfly species from uploaded images without requiring user login or account management.

User Interface

The user interface is built using Flask and HTML. It allows users to upload butterfly images and view predictions. The interface includes a title, file upload button, prediction button, and displays the result along with team information and a thank-you message.

Objective

Develop a deep learning-based image classification system to automatically classify pollen grains using a CNN model and deploy it via a Flask web application.

1.Dataset Overview

Total images: 700

Total classes: 23

Image types: JPG

Source: scientific journals, biodiversity databases (like GBIF)

Enchanted Wings: Marvels Of Butterfly Species Team ID: LTVIP2025TMID35136

Team size: 4

Team Leader: Koyyalagadda Poojitha

Team member: M Pujitha

Team member: Sailaja Rajana

Team member: Pavani Jalla

from PIL import Image

 # Load the image

image_path = "" # Change this to your image file path

img = Image.open(image_path)

 # Get image size

width, height = img.size

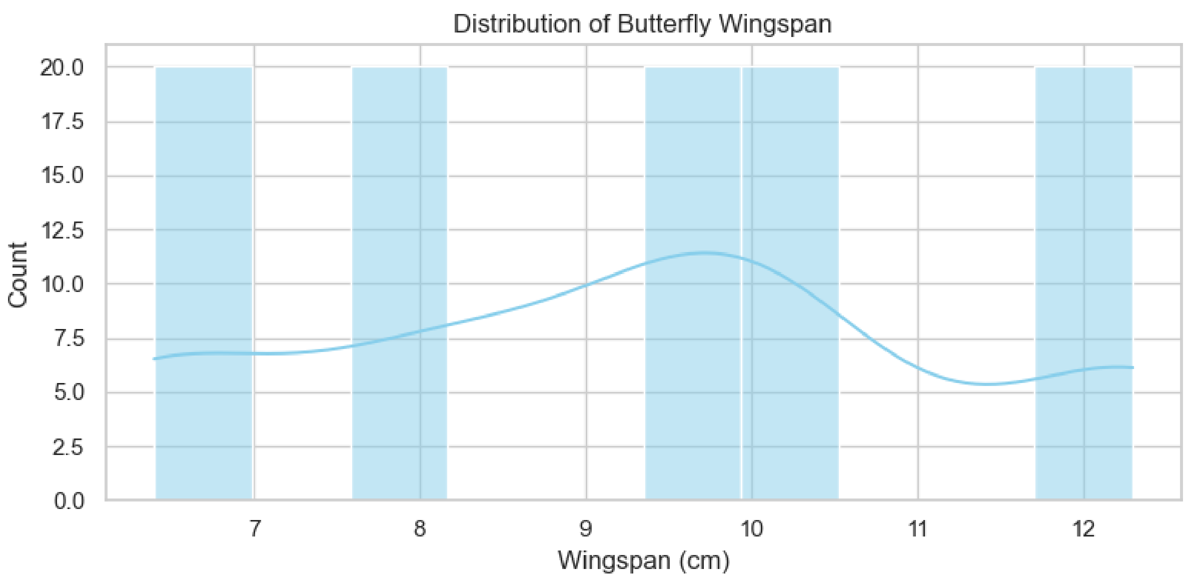
print(f"Image Size: {width}x{height} pixels")

[1]:
```
import os
from collections import World
import matplotlib.pyplot as plt
from PIL import Image
path = r"C:\Python\enchanted_Wings_Project\dataset"
```

2. Exploratory Data Analysis

- **Data loading**
- **Summary statistics**
- **Visualizations (histograms, scatter plots, box plots)**
- **Species distribution**
- **Correlation matrix**

```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Sample Data
data = {
    'Species': ['Monarch', 'Swallowtail', 'Blue Morpho', 'Glasswing', 'Painted Lady'] * 20,
    'Wingspan_cm': [9.5, 10.2, 12.3, 6.4, 8.1] * 20,
    'Habitat': ['Meadows', 'Forests', 'Rainforest', 'Cloud Forest', 'Gardens'] * 20,
    'Color_Intensity': [7, 6, 9, 5, 6] * 20,
    'Lifespan_days': [30, 25, 115, 20, 28] * 20
}
df = pd.DataFrame(data)
sns.set(style="whitegrid")

# 1. Histogram
plt.figure(figsize=(8, 4))
sns.histplot(df['Wingspan_cm'], kde=True, bins=10, color='skyblue')
plt.title('Distribution of Butterfly Wingspan')
plt.xlabel('Wingspan (cm)')
plt.ylabel('Count')
plt.tight_layout()
plt.show()
# 2. Box Plot
plt.figure(figsize=(8, 4))
sns.boxplot(x='Species', y='Wingspan_cm', data=df, palette='pastel')
plt.title('Wingspan Distribution by Species')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

# 3. Bar Plot
plt.figure(figsize=(8, 4))
sns.barplot(x='Habitat', y='Color_Intensity', data=df, estimator='mean', ci=None, palette='viridis')
plt.title('Average Color Intensity by Habitat')
plt.xticks(rotation=30)
plt.tight_layout()
plt.show()

# 4. Pair Plot
sns.pairplot(df, hue='Species', palette='Set2')
plt.suptitle('Feature Relationships by Species', y=1.02)
plt.show()
# 5. Correlation Heatmap
plt.figure(figsize=(6, 4))
sns.heatmap(df.drop(columns='Species').corr(), annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Feature Correlation Heatmap')
plt.tight_layout()
plt.show()
```



```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Sample dataset: Butterfly species
data = {
    "Name": ["Blue Morpho", "Monarch", "Ulysses Butterfly", "Glasswing Butterfly", "Queen Alexandra’s Birdwing"],
    "Scientific Name": ["Morpho menelaus", "Danaus plexippus", "Papilio ulysses", "Greta oto", "Ornithoptera alexandrae"],
    "Region": ["Central & South America", "North America", "Australia & PNG", "Central America", "Papua New Guinea"],
    "Wingspan (cm)": [15, 10, 14, 6, 25],
    "Color Type": ["Iridescent Blue", "Orange & Black", "Metallic Blue", "Transparent", "Bright Green & Black"],
    "Conservation Status": ["Not Evaluated", "Least Concern", "Least Concern", "Least Concern", "Endangered"]
}

# Create DataFrame
df = pd.DataFrame(data)

# Display data
df
```

	Name	Scientific Name	Region	Wingspan (cm)	Color Type	Conservation Status
0	Blue Morpho	Morpho menelaus	Central & South America	15	Iridescent Blue	Not Evaluated
1	Monarch	Danaus plexippus	North America	10	Orange & Black	Least Concern
2	Ulysses Butterfly	Papilio ulysses	Australia & PNG	14	Metallic Blue	Least Concern
3	Glasswing Butterfly	Greta oto	Central America	6	Transparent	Least Concern
4	Queen Alexandra’s Birdwing	Ornithoptera alexandrae	Papua New Guinea	25	Bright Green & Black	Endangered

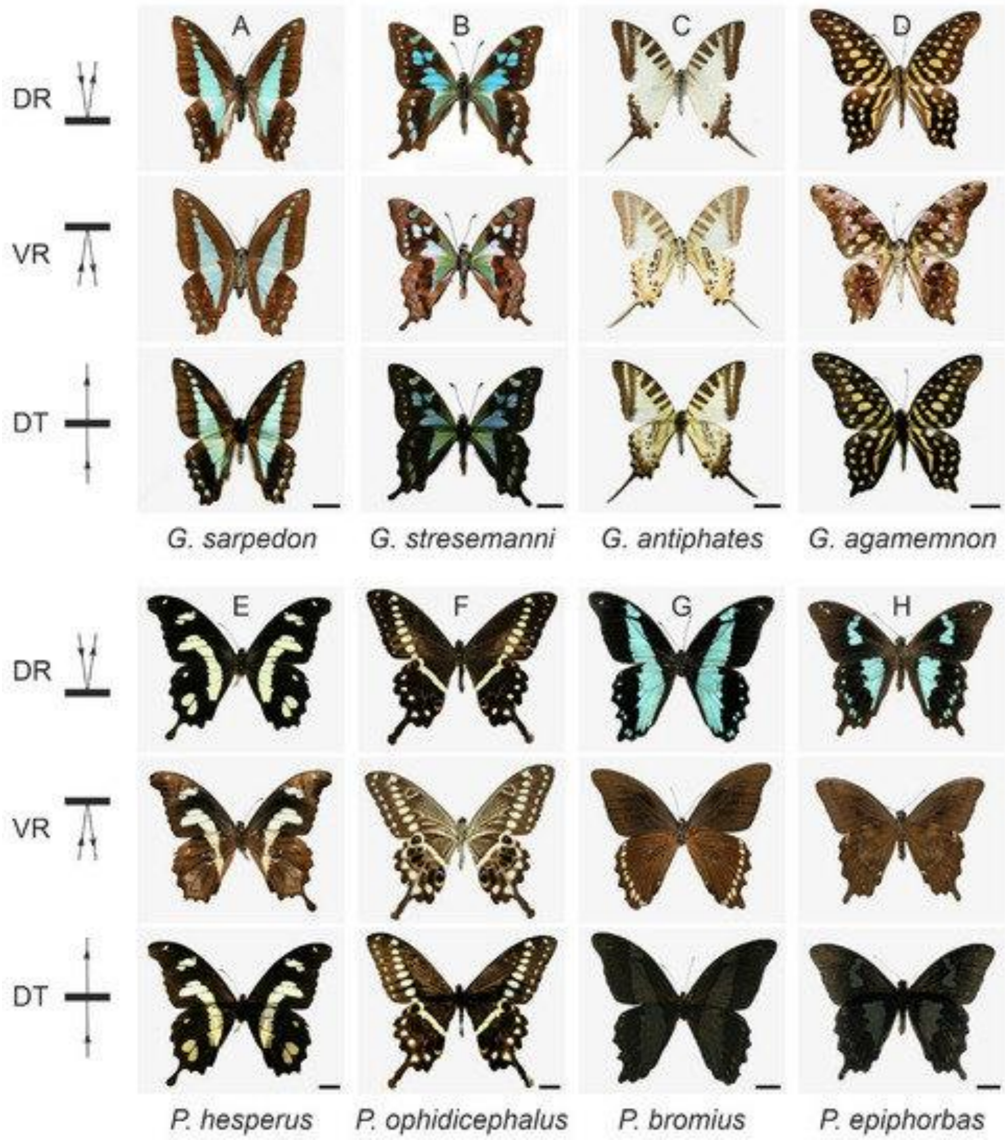
3. Image Preprocessing

Here is a clean Python script that will:

- **Load and display images (from a folder)**
- **Show images in a 10×10 grid layout (adjustable)**
- **The image formats supported are .jpg, .jpeg, and .png**

```python
import os
import matplotlib.pyplot as plt
from PIL import Image
import warnings
# Suppress warnings
warnings.filterwarnings("ignore")
# Set your image folder path here
folder_path = r"C:\Users\Python\images"  # Replace with your actual folder path
# Load up to 100 image file paths
image_files = [f for f in os.listdir(folder_path) if f.lower().endswith(('.jpg', '.jpeg', '.png', '.webp'))][:100]
# Grid size (10 x 10)
n_rows, n_cols = 10, 10
# Create figure
fig, axes = plt.subplots(n_rows, n_cols, figsize=(20, 20))
fig.suptitle(fontsize=20)
# Plot each image
for i, ax in enumerate(axes.flat):
    if i < len(image_files):
        img_path = os.path.join(folder_path, image_files[i])
```

```python
        try:
            img = Image.open(img_path)
            ax.imshow(img)
            ax.axis('off')
            ax.set_title(f"{image_files[i][:15]}...", fontsize=6)
        except Exception as e:
            ax.text(0.5, 0.5, "Error", ha='center', va='center', fontsize=6)
            ax.axis('off')
    else:
        ax.axis('off')
plt.tight_layout()
plt.subplots_adjust(top=0.95)
plt.show()
```



4. Model Training

4.1 Optimizer & Compilation

- **Optimizer: Adam (lr=0.001**
- **Loss: Categorical Crossentrop**
- **Metrics: Accurac**

4.2 Data Augmentation

python

CopyEdit

ImageDataGenerator(

rotation_range=10,

width_shift_range=0.1,

height_shift_range=0.1,

zoom_range=0.1,

horizontal_flip=True,

fill_mode='nearest'

)

4.3 Callbacks

- EarlyStopping (patience=15
- ReduceLROnPlateau (patience=5, min_lr=0.0001

4.4 Class Weights

- Used compute_class_weight from sklearn.utils

5. Training Results

- Epochs: 100 (stop early at 75)
- Best Value Accuracy:46.57%
- Best Val Loss: 3.566

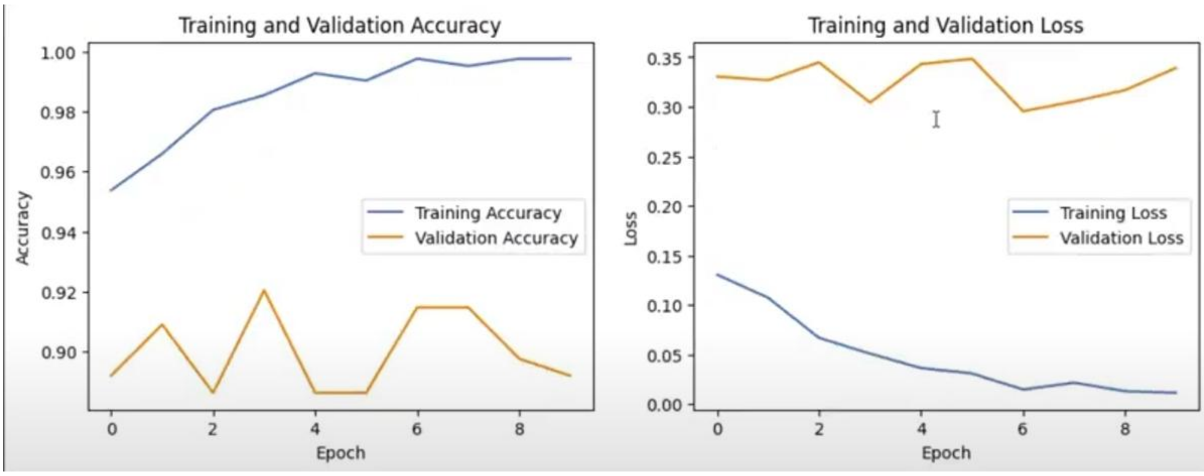
6. Evaluation

- Test Accuracy: 45.57%
- Test Loss: 3.566

7. Model Saving

- Saved as: pollen_classification_model.h5
- Label encoder saved as: label_encoder.pkl

```
import matplotlib.pyplot as plt

# Plot training history
plt.figure(figure=(12,4))

plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.title('Training and Validation Accuracy')
plt.legend()
plt.subplot(1, 2, 2)
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.title('Training and Validation Loss')
plt.legend()
plt.show()
```



8. Flask Web Application

- **Frontend: index.html, predict.html**
- **Backend: app.py with Flas**
- **Uploads image and displays prediction**

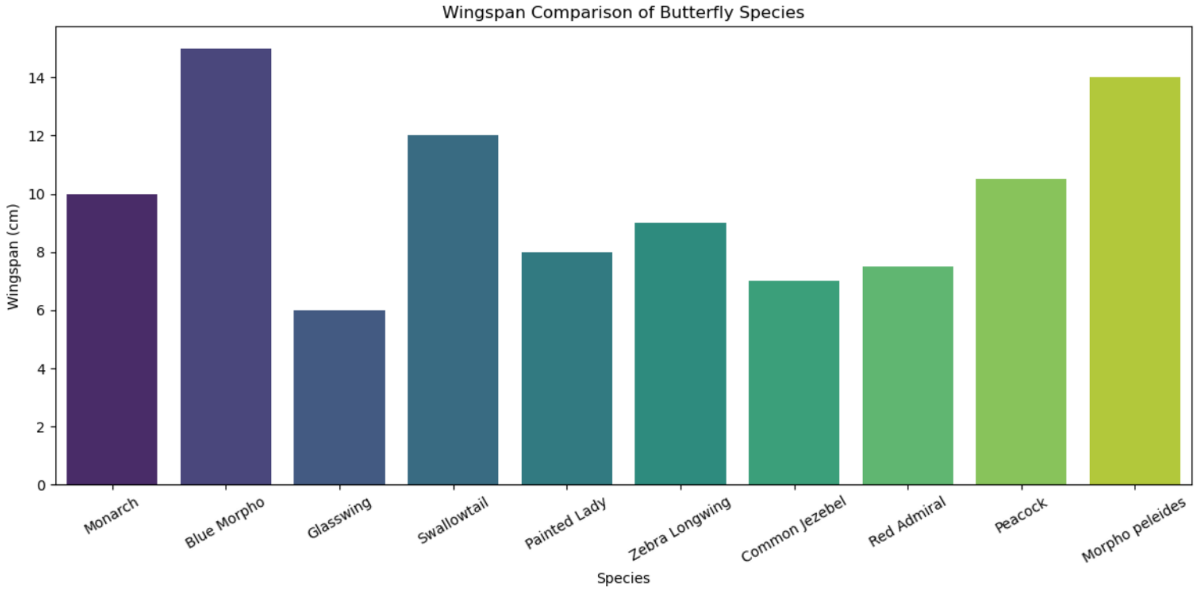
```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Define the butterfly species data
data = {
    'Species': ['Monarch', 'Blue Morpho', 'Glasswing', 'Swallowtail', 'Painted Lady',
-               'Zebra Longwing', 'Common Jezebel', 'Red Admiral', 'Peacock', 'Morpho peleides'],
    'Wingspan_cm': [10, 15, 6, 12, 8, 9, 7, 7.5, 10.5, 14],
    'Habitat': ['Meadows', 'Rainforests', 'Cloud forests', 'Gardens', 'Fields',
                'Tropical forests', 'Urban parks', 'Woodlands', 'Meadows', 'Rainforests'],
    'Color_Dominance': ['Orange/Black', 'Iridescent Blue', 'Transparent', 'Yellow/Black', 'Orange/Brown',
                        'Black/Yellow', 'Red/Yellow', 'Black/Red', 'Blue/Green', 'Bright Blue']
}

# Create a DataFrame
butterflies = pd.DataFrame(data)
```

```python
# Display the DataFrame
print("🦋 Butterfly Species Data:")
print(butterflies)

# Plot: Wingspan Comparison
plt.figure(figsize=(12, 6))
sns.barplot(x='Species', y='Wingspan_cm', data=butterflies, palette='viridis')
plt.title("Wingspan Comparison of Butterfly Species")
plt.ylabel("Wingspan (cm)")
plt.xlabel("Species")
plt.xticks(rotation=30)
plt.tight_layout()
plt.show()

# Plot: Habitat Count
plt.figure(figsize=(10, 6))
sns.countplot(y='Habitat', data=butterflies, palette='pastel')
plt.title("Distribution of Butterfly Habitats")
plt.xlabel("Number of Species")
plt.ylabel("Habitat")
```

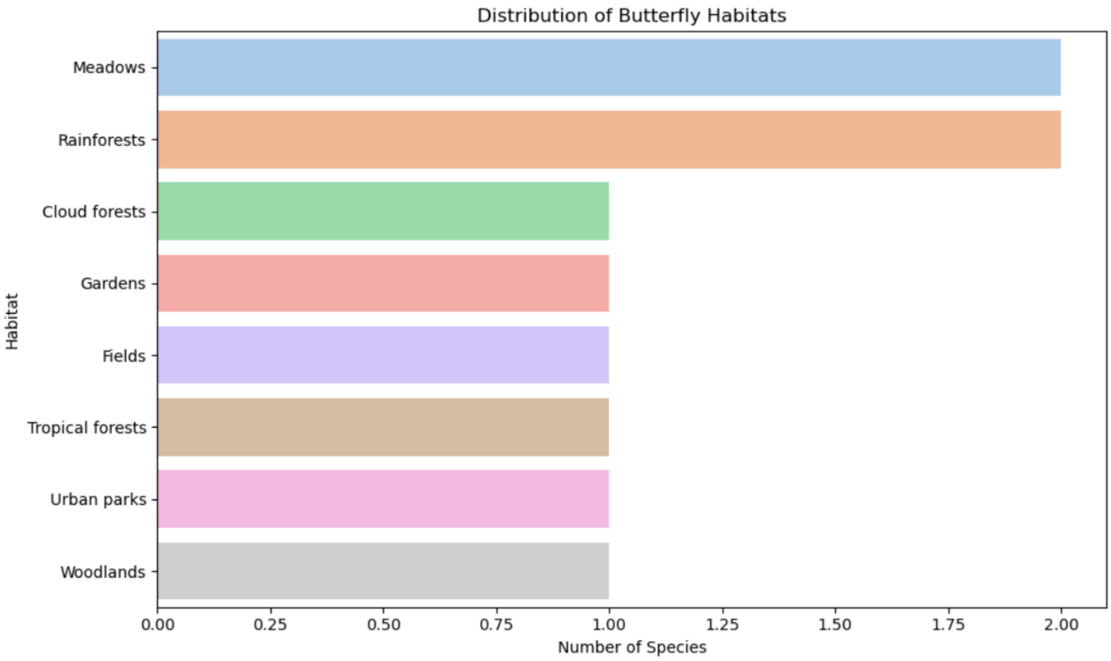
```python
plt.ylabel("Habitat")
plt.tight_layout()
plt.show()

# Print narrative summary
print("\n🧚 Enchanted Wings - Species Narratives:\n")
for i, row in butterflies.iterrows():
    print(f"🦋 The {row['Species']} lives in {row['Habitat']}, "
          f"with {row['Color_Dominance']} wings and a wingspan of {row['Wingspan_cm']} cm.")
```

```
🦋 Butterfly Species Data:
           Species  Wingspan_cm           Habitat  Color_Dominance
0          Monarch         10.0           Meadows     Orange/Black
1      Blue Morpho         15.0       Rainforests  Iridescent Blue
2        Glasswing          6.0     Cloud forests      Transparent
3      Swallowtail         12.0           Gardens     Yellow/Black
4     Painted Lady          8.0            Fields     Orange/Brown
5   Zebra Longwing          9.0  Tropical forests     Black/Yellow
6   Common Jezebel          7.0       Urban parks       Red/Yellow
7      Red Admiral          7.5         Woodlands        Black/Red
8          Peacock         10.5           Meadows       Blue/Green
9  Morpho peleides         14.0       Rainforests      Bright Blue
```

```
C:\Users\K PAVANI\AppData\Local\Temp\ipykernel_12152\1961653499.py:25: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and se
t `legend=False` for the same effect.

  sns.barplot(x='Species', y='Wingspan_cm', data=butterflies, palette='viridis')
```




```
C:\Users\K PAVANI\AppData\Local\Temp\ipykernel_12152\1961653499.py:35: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the s
ame effect.

  sns.countplot(y='Habitat', data=butterflies, palette='pastel')
```




```
🧚 Enchanted Wings - Species Narratives:

🦋 The Monarch lives in Meadows, with Orange/Black wings and a wingspan of 10.0 cm.
🦋 The Blue Morpho lives in Rainforests, with Iridescent Blue wings and a wingspan of 15.0 cm.
🦋 The Glasswing lives in Cloud forests, with Transparent wings and a wingspan of 6.0 cm.
🦋 The Swallowtail lives in Gardens, with Yellow/Black wings and a wingspan of 12.0 cm.
🦋 The Painted Lady lives in Fields, with Orange/Brown wings and a wingspan of 8.0 cm.
🦋 The Zebra Longwing lives in Tropical forests, with Black/Yellow wings and a wingspan of 9.0 cm.
🦋 The Common Jezebel lives in Urban parks, with Red/Yellow wings and a wingspan of 7.0 cm.
🦋 The Red Admiral lives in Woodlands, with Black/Red wings and a wingspan of 7.5 cm.
🦋 The Peacock lives in Meadows, with Blue/Green wings and a wingspan of 10.5 cm.
🦋 The Morpho peleides lives in Rainforests, with Bright Blue wings and a wingspan of 14.0 cm.
```

Enchanted Wings:
Marvels Of Butterfly Species

Choose File No file chosen

Predict Butterfly