Enchanted Wings

Marvels of Butterfly Species

An Internship Project by

K Akhilesh

June 2025

Table of Contents

1. Phase-1: Brainstorming & Ideation
2. Phase-2: Requirement Analysis
3. Phase-3: Project Design
4. Phase-4: Project Planning (Agile Methodologies)
5. Phase-5: Project Development
6. Phase-6: Functional & Performance Testing

Phase-1: Brainstorming & Ideation

Identifying butterfly species manually is a time-consuming and expertise-driven process.
This project aims to solve this by building an automated butterfly image classification system using pre-trained CNNs like MobileNetV2 or EfficientNetB0. The system will assist researchers, ecologists, educators, and citizen scientists in identifying butterfly species efficiently.
The goal is to deliver a lightweight, accurate tool usable in field research and education.

Phase-2: Requirement Analysis

This phase defines the technical and functional requirements using Python, TensorFlow (Keras), Pandas, NumPy, and Scikit-learn. The model will handle image preprocessing, label encoding, training with CNNs, and exporting models using .save() and joblib. Anticipated challenges include dataset imbalance and overfitting, mitigated by augmentation, dropout, and early stopping.

Phase-3: Project Design

The system architecture uses a modular pipeline from image input, preprocessing, to CNN model prediction and label decoding. A responsive UI (via Streamlit or Flask) allows users to upload images and receive real-time predictions. The design emphasizes usability, responsiveness, and accessibility.

Phase-4: Project Planning (Agile Methodologies)

Development is broken into 1–2 week sprints focusing on data preparation, modeling, evaluation, and deployment. Team roles are defined by component (data pipeline, model training, UI), and deliverables are reviewed after each sprint for continuous improvement and iteration.

Phase-5: Project Development

Core model built with MobileNetV2 and custom dense layers, using stratified sampling and TensorFlow pipelines. Overfitting was mitigated using dropout and early stopping. Model

training was performed on Google Colab. The model and label encoder were saved and prepared for deployment.

Phase-6: Functional & Performance Testing

The system was tested for prediction accuracy, input handling, and reproducibility. Label encoding mismatches and image path errors were fixed. The final model achieved 80–90% accuracy, confirming readiness for research and educational deployment, with Streamlit or Flask as deployment options.