Project Design Phase
Solution Architecture

Date	27th june 2025
Team ID	LTIVP2025TMID42332
Project Name	Enchanted Wings: Marvel species Butterfly project
Maximum Marks	4 Marks

Solution Architecture – *Enchanted Wings: Marvels of Butterfly Species:*

Overview:

The solution architecture is designed as a modular and scalable AI-based image classification system. It uses deep learning to identify butterfly species from user-uploaded images and delivers results through an intuitive web interface. The architecture ensures smooth interaction between the user, AI model, and data storage, supporting both offline and online functionality.

Architecture Components:

1.Frontend (User Interface Layer)

Technology Used: Streamlit / Flask with HTML and CSS
Role:

- Enables users to upload butterfly images.
- Displays predicted species and confidence scores.
- Presents prediction history, animated backgrounds, and visual results.
- Ensures minimal technical skill requirement via a clean, interactive UI.

2.Backend (Application Logic Layer)

Technology Used: Python with Flask
Role:

- Accepts and processes image requests from frontend.
- Loads and applies the trained deep learning model.
- Handles input validation, error management, and prediction logic.
- Logs each request with image metadata, timestamp, and prediction result.

3.Model Layer (AI/ML Engine)

Technology Used: TensorFlow / Keras
Model Type: Convolutional Neural Network (CNN) – MobileNetV2 / ResNet50 / VGG16
Role:

- Classifies butterfly species from input images.

- Trained on a curated dataset of butterfly images.
- Achieves high prediction accuracy through data augmentation, normalization, and fine-tuning.
- Loaded in-memory for fast runtime inference.

Deployment & Usage:

- **Offline Use:** Entire app including model, static assets, and templates stored locally. Requires no internet access once installed.
- **Online/Cloud Option (Future Scope):** Dockerized deployment for server-based or cloud-based scalability with API support.
- **System Requirements:** Python 3.x, Anaconda environment, GPU-enabled system or Google Colab (for training).

Data Flow Summary:

```
[User Uploads Image]
        ↓
[Frontend (Streamlit/Flask)]
        ↓
[Backend (Flask App Logic)]
        ↓
[Trained CNN Model (MobileNetV2/ResNet50)]
        ↓
[Prediction Output (Species + Confidence)]
        ↓
[Frontend Displays Results + Stores in DB]
```