Project Design Phase-II
Technology Stack (Architecture & Stack)

Date	27th june 2025
Team ID	LTVIP2025TMID42332
Project Name	*Enchanted Wings: Marvels of Butterfly Species*
Maximum Marks	4 Marks

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

The system architecture is designed to deliver a lightweight, offline-capable, AI-powered butterfly species classifier with a clean UI and high-performance backend logic using CNN models.

High-Level Architecture Description

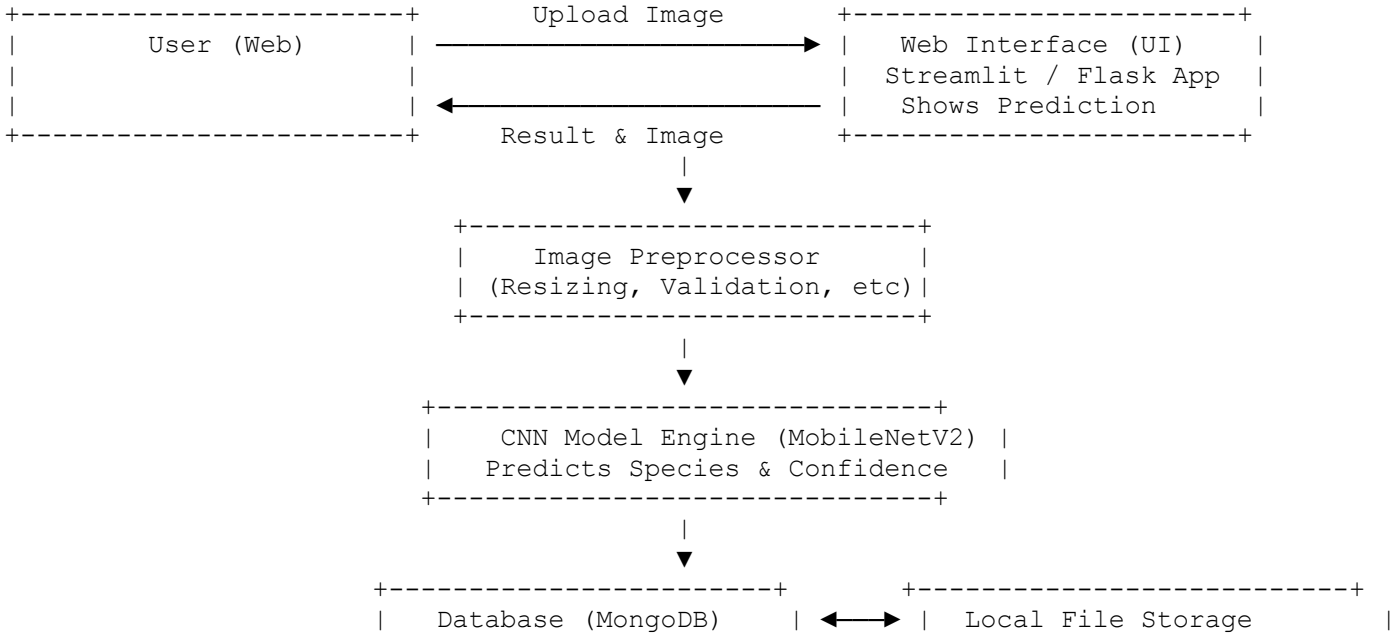

Table 1: Components & Technologies

S.No	Component	Description	Technology
1	**User Interface**	Interface for uploading images, receiving predictions	Streamlit / HTML / Flask
2	**Application Logic-1**	Handles image input, preprocessing, and user flow	Python
3	**Application Logic-2**	Prediction logic – classifying butterfly species using deep learning	TensorFlow / Keras
4	**Application Logic-3**	Visual output rendering and basic animations	Streamlit / Flask HTML Templates
5	**Database**	Stores logs of predictions, user input, and species metadata	MongoDB (or SQLite for local use)
6	**Cloud Database**	(Optional for future) Stores logs in cloud for scalability	MongoDB Atlas / Firebase Firestore
7	**File Storage**	Stores user-uploaded butterfly images and model files	Local Filesystem
8	**External API-1**	(Planned) Integrate map/geolocation for species distribution	Google Maps API / OpenStreetMap API (future use)
9	**External API-2**	(Planned) Fetch butterfly taxonomy or descriptions from open datasets	GBIF API / iNaturalist API (future enhancement)
10	**Machine Learning Model**	Classifies butterfly species from image input	CNN (MobileNetV2 / ResNet50 / VGG16)
11	**Infrastructure**	Hosts the app on local system or optionally deploys to cloud	Localhost / Flask Runtime / Docker (optional)

Table 2: Application Characteristics

S.No	Characteristic	Description	Technology
1	**Open-Source**	Frameworks used are freely available and modifiable	Streamlit, Flask, TensorFlow, Keras

S.No	Characteristic	Description	Technology
	Frameworks		
2	**Security Implementations**	Image input validation, path sanitization, basic user input checks	SHA256 (future), Input Sanitization
3	**Scalable Architecture**	Modular structure supports new model integration and cloud deployment	MVC Pattern, Modular Flask App
4	**Availability**	Fully functional offline; can optionally deploy to cloud for availability	Local System + Docker (future scope)
5	**Performance**	Loads pre-trained model in memory; responses within 2–3 seconds for typical image inputs	TensorFlow In-Memory Model Loading, FastAPI (optional)

Summary:

This architecture allows for:

- **Offline usability**
- **Fast AI-powered classification**
- **Smooth UI/UX for non-technical users**
- **Optional future integration with cloud, APIs, and real-time field deployment**