## ****Enchanted Wings: Marvels of Butterfly Species****

### ****Phase 1: Brainstorming & Ideation****

#### 1. Problem Statement

Identifying butterfly species manually is a complex, time-consuming task that requires specialized expertise. This presents challenges for researchers in biodiversity and ecology and limits the participation of non-experts in citizen science and education. An automated, scalable, and accurate classification system is needed to bridge this gap and enhance public engagement in conservation.

#### 2. Proposed Solution

This project introduces an automated butterfly classification system using transfer learning with pre-trained Convolutional Neural Networks (CNNs) such as MobileNetV2 or EfficientNetB0. Trained on a labeled dataset of 6,499 butterfly images across 75 species, the model aims to deliver efficient and high-accuracy species identification from input images.

#### 3. Target Users

Field researchers and biodiversity scientists

Ecologists studying butterfly behavior and distribution

Educators and students in entomology or environmental studies

Citizen scientists and nature enthusiasts

#### 4. Expected Outcome

A lightweight, accurate, and user-friendly butterfly classification tool that can be integrated into educational platforms, field research tools, and citizen science applications—supporting real-time species identification and promoting ecological awareness.

### ****Phase 2: Requirement Analysis****

The classification system is developed using Python, leveraging libraries such as:

**TensorFlow & Keras** for model development

**Pandas & NumPy** for data handling

**Scikit-learn** for preprocessing and label encoding

**Joblib** and .save() for model persistence

**Jupyter Notebooks** or modular Python scripts for development

**TensorBoard** for training visualization

**Deployment Considerations:**

Lightweight frameworks: Flask, Streamlit, FastAPI

For mobile: TensorFlow Lite ensures offline accessibility

**Functional Requirements:**

Image loading with metadata from structured directories

Preprocessing: resizing, normalization, batching

CNN-based model with custom dense layers

Training logic: 80/20 split, early stopping, checkpoints

Inference capability: accept new images, return predictions

**Challenges:**

**Class imbalance**: addressed with augmentation and class weighting

**Small dataset size**: mitigated via regularization and dropout

**Hardware limitations**: resolved using platforms like Google Colab

### ****Phase 3: Project Design****

The system architecture is modular and includes:

Image input (upload or capture)

Preprocessing module

Trained CNN model (MobileNetV2 or EfficientNetB0)

Label encoder mapping predictions to species names

Result display via an intuitive UI

**UI/UX Focus:**

Minimalist, mobile-responsive interface

Sections: image upload, preview, prediction results

Educational add-ons (species info, past results)

Accessibility and user-friendliness prioritized

### ****Phase 4: Project Planning (Agile)****

The development followed Agile methodology with multiple sprints:

**Sprint Planning:** Tasks divided into dataset prep, model training, and deployment

**Team Roles:** Data preprocessing, model tuning, UI integration

**Deliverables:** Reviewed after each sprint for continuous improvement

This approach ensured flexibility, accountability, and timely progress.

### ****Phase 5: Project Development****

**Tech Stack:**

Python 3.x, TensorFlow (Keras), Pandas, NumPy, Scikit-learn

Joblib for encoder saving, TensorBoard for training tracking

**Workflow:**

1.Load and clean dataset using metadata

2.Encode labels, split data for balance

3.Build model with MobileNetV2 + custom layers

4.Train with early stopping, save best model

5.Modular design enables easy UI/API integration

**Challenges & Solutions:**

**Imbalanced dataset:** Stratified sampling and augmentation planned

**Overfitting:** Dropout, early stopping, regularization

**Slow training:** Used Google Colab for faster compute

### ****Phase 6: Functional & Performance Testing****

The model was tested across multiple parameters:

Preprocessing validation

Label encoding consistency

Prediction accuracy across all classes

Stability and reproducibility of results

Robust error handling (e.g., invalid images)

**Validation Results:**

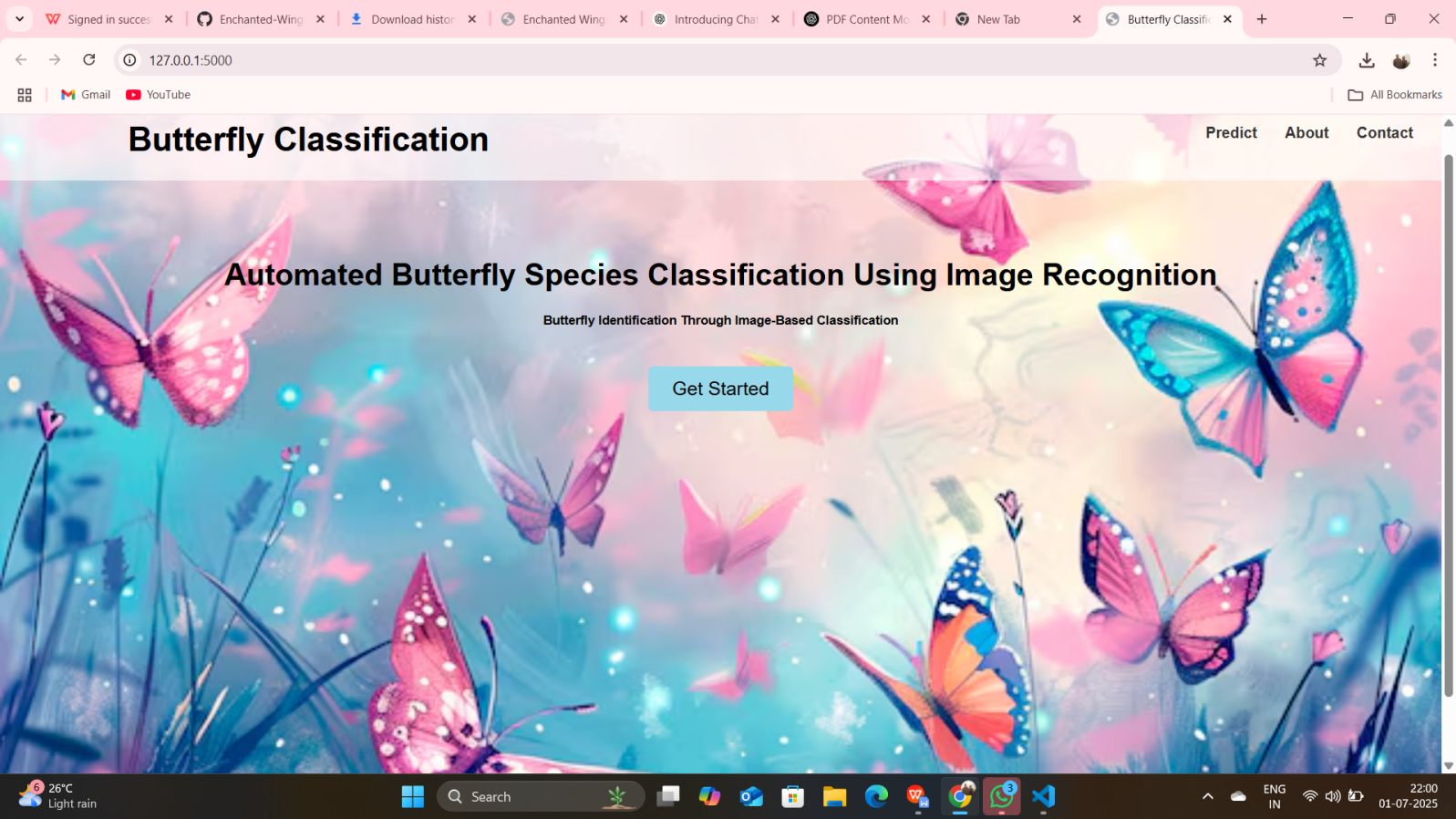
Model achieved ~80–90% accuracy

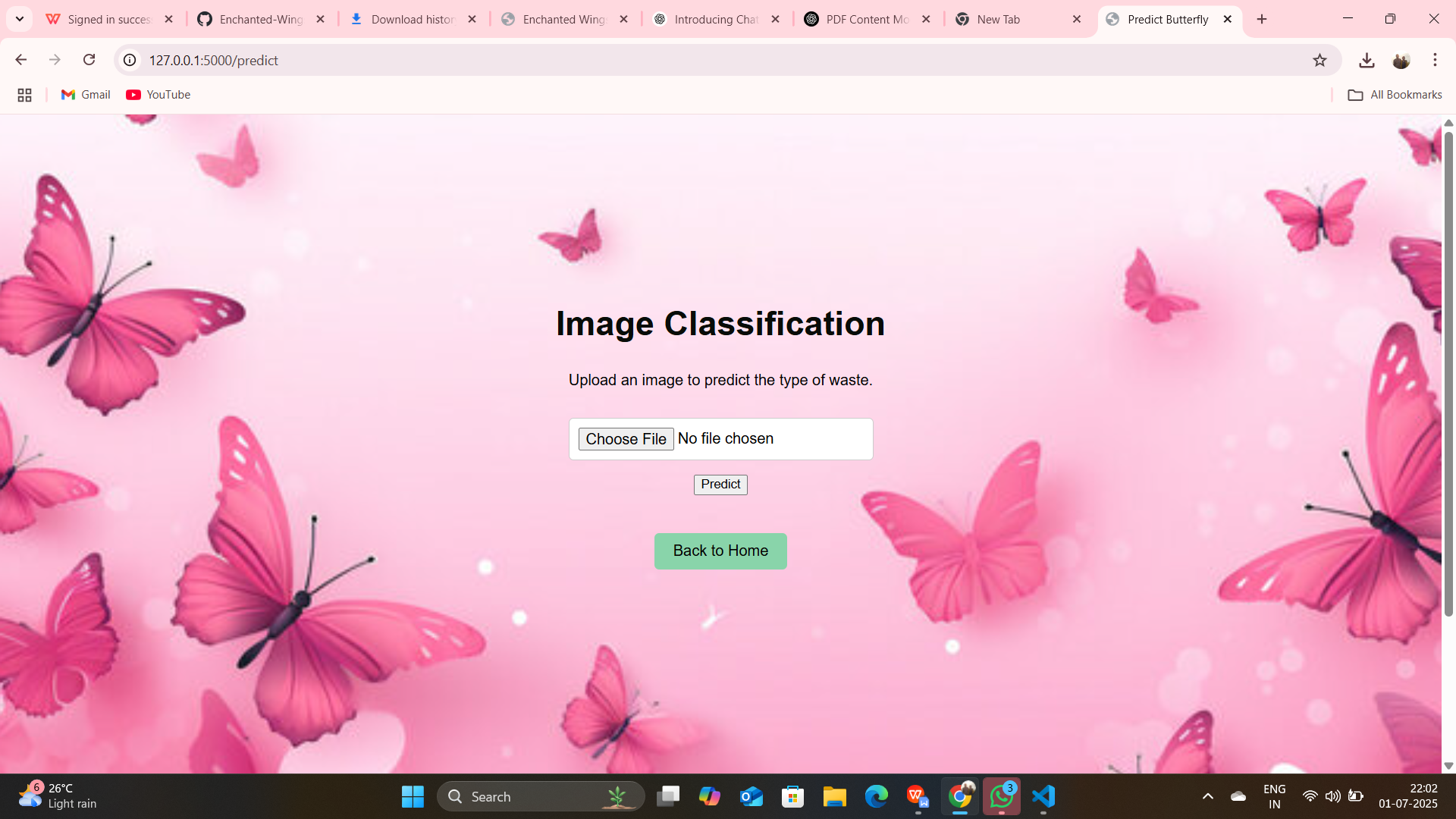
Ready for deployment using Streamlit/Flask

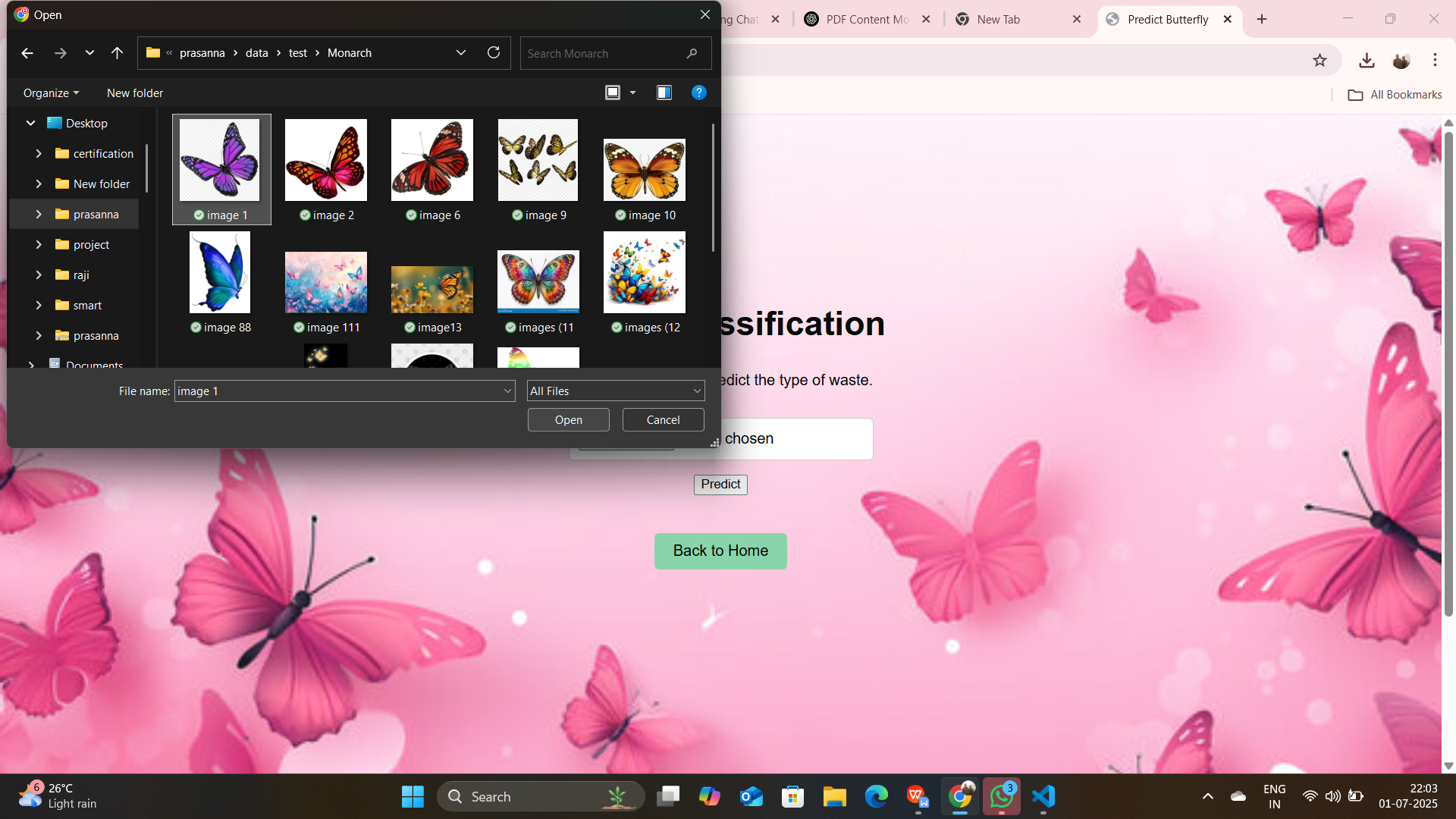
Supports image input → prediction → label output pipeline

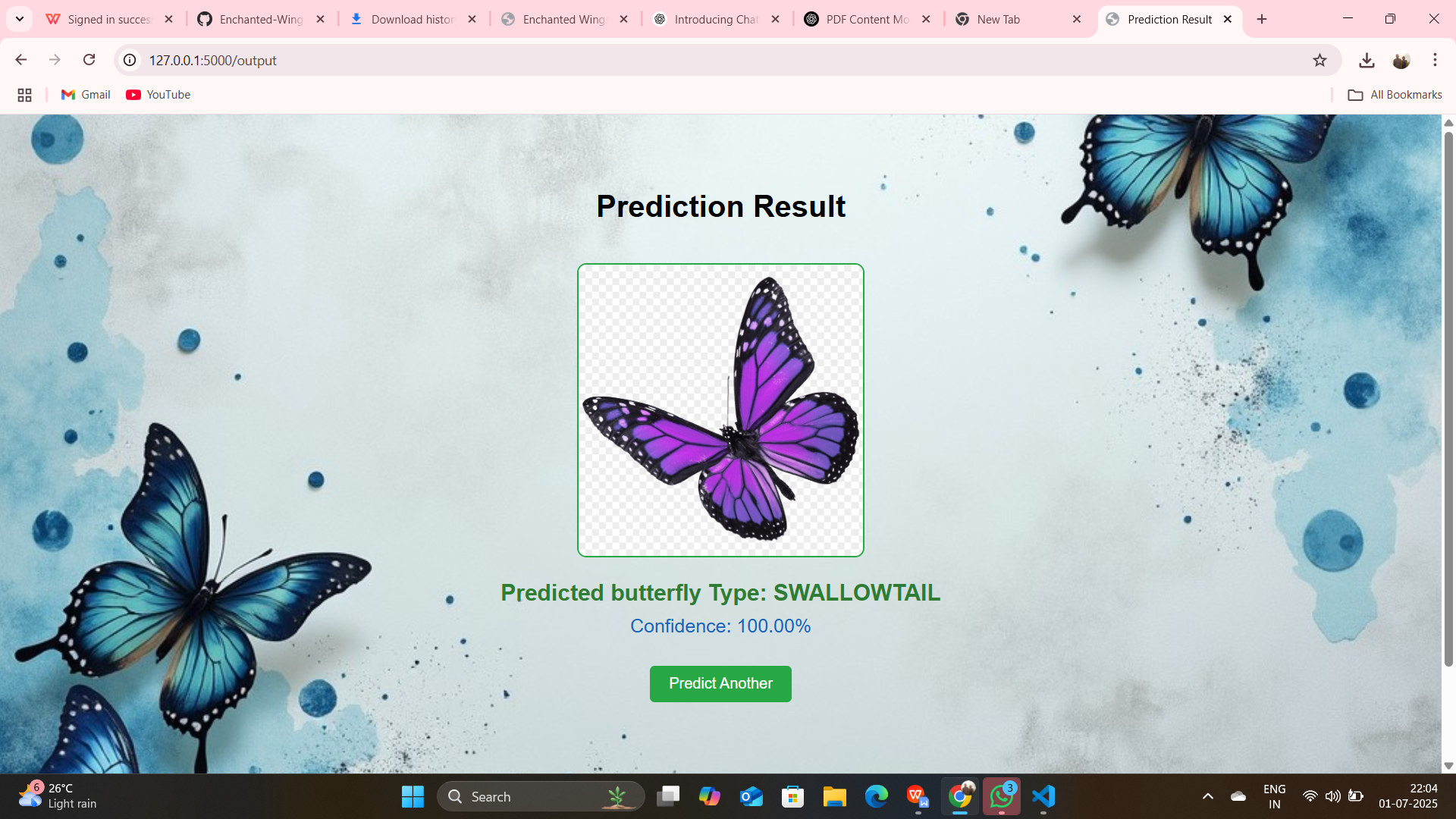
### ****Phase 7: Results & Conclusion****

The Enchanted Wings system successfully automates butterfly species classification using deep learning. With robust accuracy and modular architecture, it’s ready for research, education, and citizen science applications.









### ****Advantages :****

1.Fast and lightweight

2.Easy-to-use interface

3.Encourages ecological learning and public engagement

### ****Disadvantages :****

Accuracy depends on dataset diversity

Limited to species present in the training dataset

### ****Future Scope :****

Integrate **Grad-CAM** for visual explanations

Expand dataset to include moths and rare species

Deploy as a **Progressive Web App (PWA)** for offline use

Enable **user-contributed labeling** for dataset enrichment

### ****Appendix :****

📂 **GitHub Repository:** [Enchanted Wings – GitHub](https://github.com/akhil868825/Enchanted-Wings-Marvels-Of-Butterfly-Species)

📸 **Dataset Link:** Image Classification Dataset

🎥 **Demo Video:** [View on GitHub](https://github.com/akhil868825/Enchanted-Wings-Marvels-Of-Butterfly-Species/upload/main/Video Demo)