**Machine Learning Project Documentation format**

# 1. Introduction

* **Project Title:** Pollen's Profiling: Automated Classification of Pollen Grains
* **Team Members:**

1. Eduri Maryjones (Data collection,Train the model,Application building)
2. Idimukkala Yasasswini (Train the model,save the model,Test the model,Application building)
3. Mamidela Venkata Naga Suseel Kumar (Read the data,Image pre-processing,Training the model,Save the model,Test the model,Application building)
4. Inturi Venkata Vikash (Data Collection,Exploratory Data Analysis,Image Pre-processing,Application Building)
5. J Pushpitha (Data Collection,Exploratory Data Analysis,Image Pre-processing,Application Building)

# 2. Project Overview

* **Purpose:** Automate pollen grain identification using a deep learning CNN model

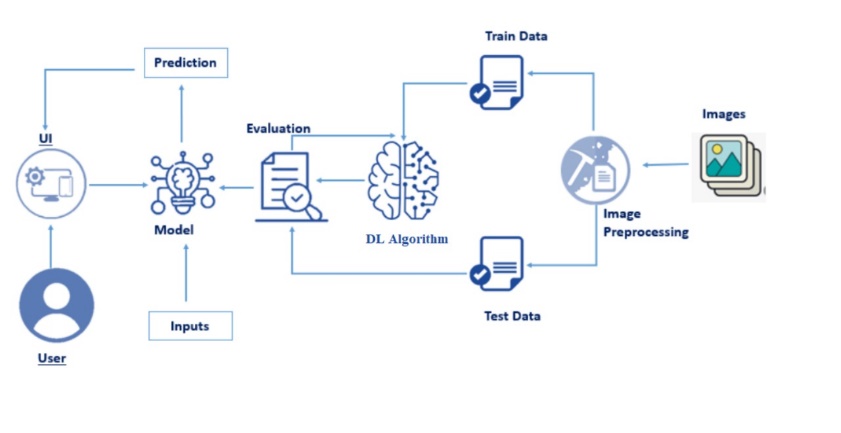
based on image classification.

* **Features:** Image upload for pollen classification.

Prediction of pollen type.

Web-based interface using Flask.

# 3. Architecture



# 4. Setup Instructions

* **Prerequisites:** Python, TensorFlow, Flask, OpenCV, Scikit-learn, Numpy, Pandas,

Matplotlib.

* **Installation:**

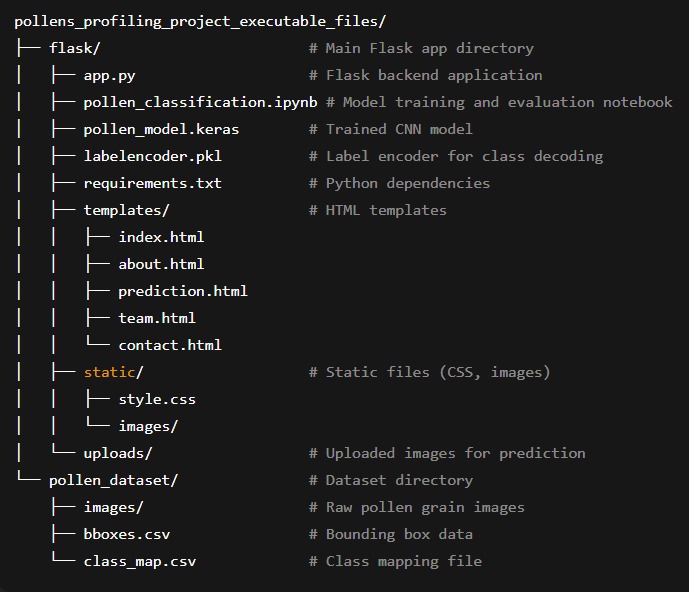
git clone https://github.com/EduriMaryJones/Pollen-s-Profiling-Automated-Classification-of-Pollen-Grains.git

cd Pollen-s-Profiling-Automated-Classification-of-Pollen-Grains

cd flask

pip install -r requirements.txt

# 5. Folder Structure



# 6. Running the Application

• > python app.py

Type this in the terminal and click the link

# 7. API Documentation

* Base URL: http://127.0.0.1:5000/
* 🚩 Endpoints:

/ → Home (GET)

/about → About page (GET)

/prediction→ Prediction page UI (GET)

/team → Team page (GET)

/contact → Contact page (GET)

/predict → Prediction API (POST)

* 🔥 /predict API:

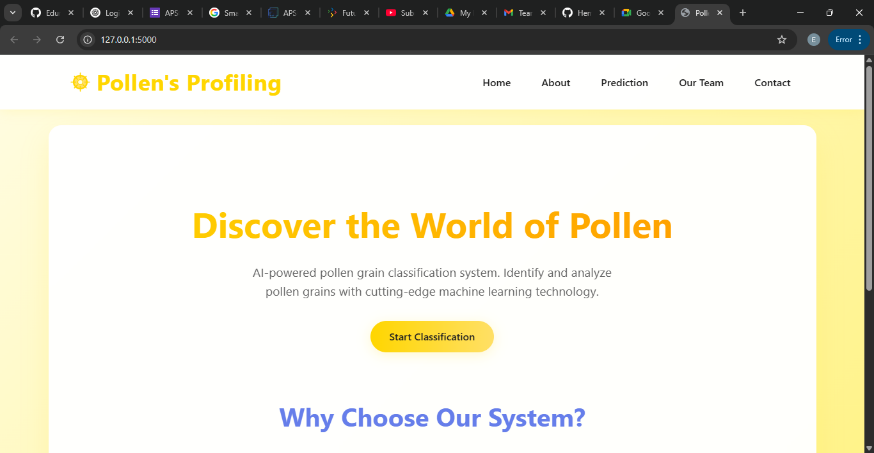
Method: POST

Input: Image file (file)

# 8. Authentication

* No Authentication required

1. **User Interface**



1. **Testing**

🧪 Testing Strategy

✅ Manual Testing

* + Each feature of the web application (Homepage, Prediction, Upload, Result display, Contact form) was manually tested across different browsers (Chrome, Edge, Firefox).
  + File upload validation was tested with:
  + Valid image formats (.jpg, .png)
  + Invalid inputs (non-image files, empty uploads)

✅ Model Evaluation

* + The CNN model was evaluated using

Train/Test Split (80/20) stratified by class

Metrics Used:

* + Accuracy
  + Confusion Matrix
  + Classification Report (Precision, Recall, F1-score)
  + Visual performance tracking was done using Matplotlib plots for training/validation loss and accuracy.

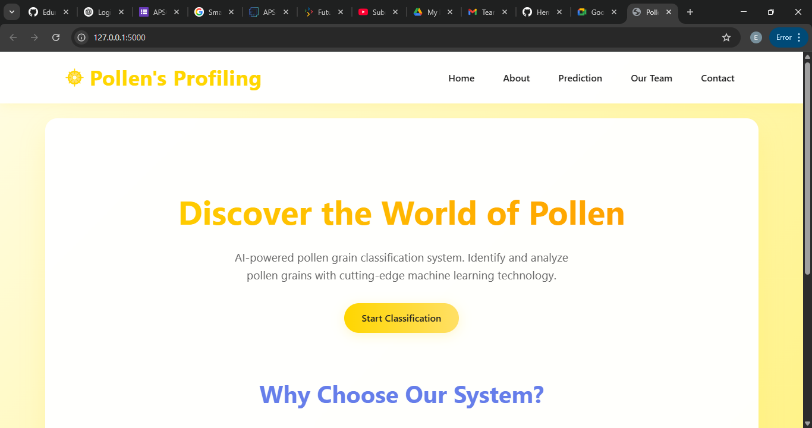
✅ Prediction Testing

* + Real-time predictions were tested by:
  + Uploading test images via the /predict endpoint
  + Verifying output class labels against known inputs
  + Cross-checking consistency with model evaluation results in the notebook

🧰 Tools Used for Testing

* + Tool / Library Purpose
  + Jupyter Notebook Model training, evaluation, and testing
  + Matplotlib Visualization of training metrics
  + Scikit-learn Confusion matrix, classification report
  + Browser Developer Tools UI inspection and validation
  + Flask Debug Mode Monitoring and catching runtime errors

1. **Screenshots or Demo**



1. **Known Issues** 
   * Accuracy depends on dataset quality.
   * Misclassification in very similar pollen types.

# 13. Future Enhancements

* Deploy on cloud (Render, Heroku)
* Add user login to save predictions.
* Improve model with a larger dataset.