Artificial Intelligence and Machine Learning Documentation

Project Title: Transfer Learning-Based Classification of Poultry Diseases for Enhanced Health Management

1. Introduction

• Project Title: Transfer Learning-Based Classification of Poultry Diseases for Enhanced Health Management.

• Team Members:

- Team Leader: Mekala Naga Sai Gowtham Raj

- Team Member: Syed Karimulla

- Team Member: U Alekhya

2. Project Overview

• Purpose: Provide farmers with an on-site, AI-driven tool that classifies poultry diseases using images and symptom data.

• Features:

1. 1. Image upload & symptom entry
2. 2. Real-time disease prediction + confidence score
3. 3. Auto-generated treatment guide
4. 4. Disease-history log
5. 5. Offline mode (planned)

3. Architecture

• Frontend: HTML/JS + Tailwind-CSS

• Backend/API: Python (Flask)

• ML Model: Transfer Learning (EfficientNet/MobileNet)

• Data store: SQLite/JSON

• Deployment: Render.com (Docker)

4. Setup Instructions

• Prerequisites: Python 3.10+, pip, virtualenv, Git, TensorFlow

• Installation:

```
git clone https://github.com/GowthamRaj8886/PoultryDetect-
cd PoultryDetect-
python -m venv .venv && source .venv/bin/activate
pip install -r requirements.txt
python download_model.py
```

5. Folder Structure

```
PoultryDetect-/
├── app.py
├── /static/
│   └── uploads/
├── /templates/
├── /model/
│   ├── model.h5
│   └── labels.json
└── requirements.txt
```

6. Running the Application

• Development:

```
export FLASK_APP=app.py ; flask run
```

• Production:

```
# Build the Docker image

docker build -t poultrydetect .

# Run the Docker container

docker run -d -p 5000:5000 poultrydetect
```

7. API Documentation

• `/predict` (POST):

Accepts an image file (typically of a chicken) and optional symptom/environmental data in JSON format.

Returns a JSON response with the predicted disease and its confidence score.

Example response:

`{ "disease": "Coccidiosis", "confidence": 0.94 }`

• `/history` (GET):

Accepts a query parameter `farm_id`.

Returns a list of previous predictions made for that farm, including timestamps and diseases identified.

Useful for tracking recurring disease patterns.

• `/disease/<name>` (GET):

Returns detailed information in HTML/Markdown format about the specified disease name (e.g., `Coccidiosis`, `Salmonella`).

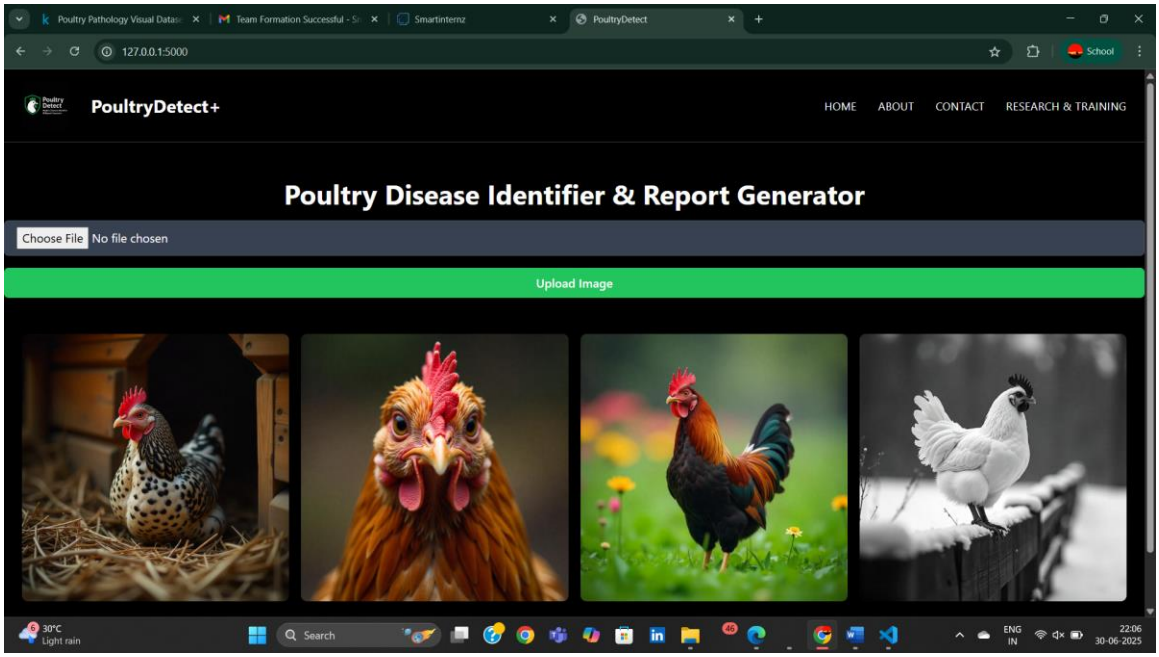
Includes symptoms, treatment steps, and preventive care guidelines.

8. Authentication

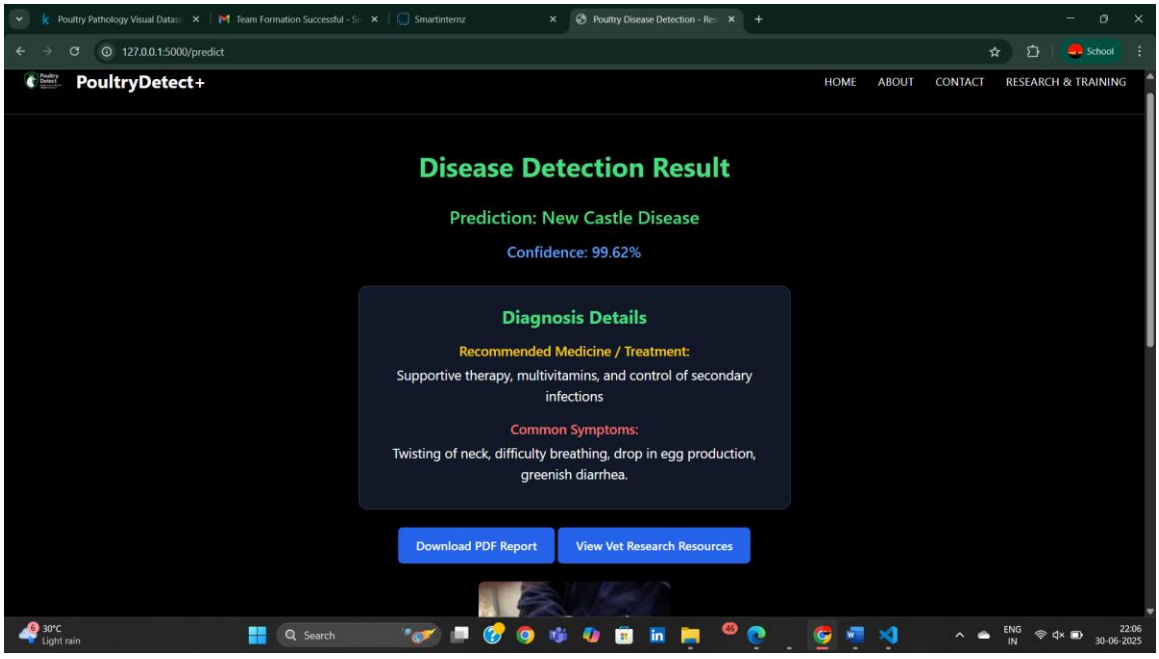
• No auth currently. Planned: JWT-based auth and HTTPS

9. User Interface

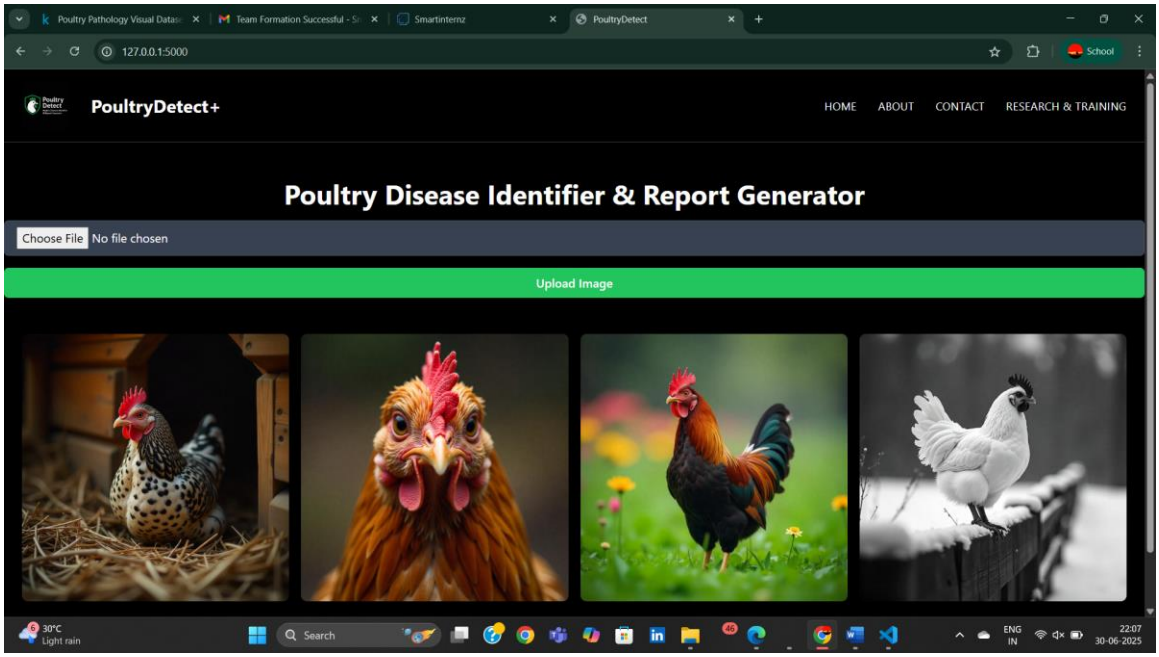
• Home / Upload



• Result – disease info, confidence, treatment



• Navigation – Home | Diseases | Contact | About



10. Testing

• Unit tests: pytest

• Performance: Flask locust

• Model metrics: 97.5% accuracy

• UI: Cypress E2E

11. Screenshots / Demo

• Live demo:
https://drive.google.com/drive/folders/1JQtPsVChqjSF5usGus3rTDMO8f0RkdBp

12. Known Issues

- Low-light images reduce accuracy
- Only 4 diseases supported
- English-only UI
- iOS offline mode not stable

13. Future Enhancements

- Add more diseases
- Publish Android app
- Drug-prescription module
- Push notifications
- Tele-consult & analytics dashboard