Full Stack Development with Flask

Project Documentation - PoultryDetect

1. Introduction

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

Team ID: LTVIP2025TMID42969
Project Duration: June 24-26, 2025
Location: Ongole, Andhra Pradesh
Institution: Rise Krishna Sai Prakasam Group of Institutions

Team Members:

- M. Karthik Reddy: Lead Developer & AI Integration
- P. Srinivasa Kalyan: Full Stack Development & UI/UX Design

2. Project Overview

Purpose: The PoultryDetect application aims to develop a Transfer learning-based system for classifying poultry diseases into four categories: Salmonella, Newcastle Disease, Coccidiosis, and Healthy. The solution provides farmers with an AI-powered tool to diagnose poultry diseases through image upload, enabling swift action to reduce disease spread and improve productivity.

Features:

- AI-powered image classification for poultry disease detection
- Real-time disease prediction with 4 disease categories
- Educational resources and research links
- Responsive web interface with glassmorphism design
- Interactive user journey for farmers and veterinary students
- Animated UI elements for enhanced user engagement

3. Architecture

Frontend:

- **Framework:** HTML5, CSS3, JavaScript with Tailwind CSS
- **Styling:** Glass morphism effects with backdrop blur
- **Components:** Responsive navigation, image upload interface, prediction display
- **Animation:** CSS animations and Lottie animations for enhanced UX

- **Design Pattern:** Mobile-first responsive design

Backend:

- **Framework:** Flask (Python web framework)
- **Model Integration:** Keras/TensorFlow for deep learning model
- **File Handling:** Werkzeug for secure file uploads
- **Image Processing:** PIL/Keras preprocessing for image manipulation
- **Architecture Pattern:** MVC (Model-View-Controller)

Database:

- **Type:** File-based storage for uploaded images
- **Storage:** Static file serving for images and assets
- **Model Storage:** H5 format for trained neural network model
- **File Management:** Secure filename handling and upload directory management

4. Setup Instructions

Prerequisites:

- Python 3.7 or higher
- pip (Python package installer)
- Virtual environment (recommended)
- Modern web browser

Installation:

bash

```bash
# Clone the repository
git clone <repository-url>
cd poultry-detect

# Create virtual environment
python -m venv venv

# Activate virtual environment
# Windows:
venv\Scripts\activate
# macOS/Linux:
source venv/bin/activate

# Install dependencies
pip install flask keras tensorflow numpy pillow werkzeug

# Create required directories
mkdir -p static/uploads

# Ensure model file is present
# Place healthy_vs_rotten.h5 in the root directory

# Set environment variables (optional)
export FLASK_ENV=development
export FLASK_DEBUG=1
```

5. Folder Structure

```
poultry-detect/
├── app.py                  # Main Flask application
├── healthy_vs_rotten.h5    # Trained ML model
├── requirements.txt        # Python dependencies
├── static/                 # Static assets
│   ├── uploads/            # Uploaded images storage
│   ├── farm.jpeg           # Background image
│   └── hen.jpeg            # Animation asset
└── templates/              # HTML templates
    ├── index.html          # Home page with upload
    ├── about.html          # About page
    ├── contact.html        # Contact form
    └── training.html       # Research and training page
```

6. Running the Application

Development Server:

```bash
# Navigate to project directory
cd poultry-detect

# Activate virtual environment
source venv/bin/activate  # or venv\Scripts\activate on Windows

# Run Flask application
python app.py

# Alternative method
flask run
```

Access Points:

- Local Development: `http://localhost:5000`
- Home Page: `http://localhost:5000/`
- About: `http://localhost:5000/about`
- Contact: `http://localhost:5000/contact`
- Training: `http://localhost:5000/training`

7. API Documentation

Endpoints

`GET /`

- **Description:** Serves the main application page with image upload functionality
- **Response:** HTML template with upload form
- **Template:** `index.html`

`GET /about`

- **Description:** Information about the PoultryDetect application
- **Response:** HTML template with project details
- **Template:** `about.html`

`GET /contact`

- **Description:** Contact form for user inquiries
- **Response:** HTML template with contact form

- **Template:** `contact.html`

`GET /training`

- **Description:** Educational resources and research links
- **Response:** HTML template with disease information
- **Template:** `training.html`

`POST /predict`

- **Description:** Handles image upload and disease prediction
- **Parameters:**
  - `file` (multipart/form-data): Image file for classification
- **Response:** HTML template with prediction results
- **Example Response:**

```
json

{
  "prediction": "Healthy",
  "img_path": "/static/uploads/chicken_image.jpg"
}
```

- **Error Handling:**
  - No file uploaded: Returns "No file uploaded" message
  - Invalid file: Returns "Invalid file" message
  - Processing error: Returns "Invalid image" message

8. Authentication

Current Implementation:

- No authentication system implemented
- Open access to all application features
- File upload security through `secure_filename()` function

Security Measures:

- Secure filename sanitization using Werkzeug
- File type validation through model processing
- Upload directory isolation
- Error handling for malicious file attempts

9. User Interface

Design Philosophy:

- Glassmorphism design with backdrop blur effects
- Nature-themed color scheme (green tones)
- Responsive design for mobile and desktop
- Interactive animations and micro-interactions

Key UI Features:

- Animated hen character walking across the screen
- Glass-effect navigation bar with blur backdrop
- Centered upload form with visual feedback
- Disease classification cards with research links
- Farmer journey timeline visualization

Screenshots Description:

- Home page with upload interface and glassmorphism effects
- Prediction results display with uploaded image
- About page with project information and features
- Training page with disease research cards
- Contact form with nature background

10. Testing

Testing Strategy:

- Manual testing of all routes and endpoints
- Image upload functionality testing with various file types
- UI responsiveness testing across different devices
- Model prediction accuracy validation
- Error handling verification

Test Cases:

- Valid image upload and prediction
- Invalid file type handling
- Large file size handling
- Network connectivity issues

- Model loading and prediction errors

11. Screenshots or Demo

Application Showcase:

- Landing page with clean, modern interface
- Real-time disease prediction results
- Educational content for veterinary training
- Responsive design across mobile and desktop
- Interactive elements and smooth animations

Demo Features:

- Live image upload and classification
- Instant AI-powered disease detection
- Educational resources integration
- User-friendly interface for farmers

12. Known Issues

Current Limitations:

- Model file (healthy_vs_rotten.h5) not included in repository
- Requirements.txt file is empty - needs population
- No database integration for storing user data or history
- Contact form is non-functional (frontend only)
- Limited error handling for edge cases
- No file size validation for uploads

Technical Debt:

- Model path hardcoded in application
- No configuration management system
- Limited logging and monitoring
- No unit tests implemented

13. Future Enhancements

Planned Features:

- User authentication and profile management

- Prediction history and analytics dashboard
- Mobile application development (React Native/Flutter)
- Real-time notifications for disease outbreaks
- Integration with veterinary consultation services
- Batch image processing capabilities
- API development for third-party integrations

Technical Improvements:

- Database integration (MongoDB/PostgreSQL)
- Containerization with Docker
- CI/CD pipeline implementation
- Comprehensive testing suite
- Performance optimization and caching
- Model versioning and A/B testing
- Multi-language support

Business Enhancements:

- Farmer community features
- Expert consultation booking
- Treatment tracking and management
- Farm management integration
- Marketplace for veterinary supplies
- Educational certification programs