# Transfer Learning-Based Classification of Poultry Diseases for Enhanced Health Management

**Team ID**

**LTVIP2025TMID42076**

**Team Members**

T Baba (Lead)

Muthulurupadu Subbarayudu

K Rama Chaitanya

Bandi Viswanadha Reddy

Smart Internz

# 1. INTRODUCTION

## 1.1 Project Overview

This project focuses on developing a transfer learning-based machine learning system for classifying poultry diseases into four categories: Salmonella, New Castle Disease, Coccidiosis, and Healthy. The system will be integrated into a mobile application that allows farmers to input relevant data such as symptoms, environmental conditions, and biological indicators. Using this data, the model will provide an instant and accurate diagnosis along with recommended treatment measures. The project aims to bring cutting-edge technology into the hands of poultry farmers and veterinary professionals to improve animal health and reduce the economic impact of diseases.

In addition to providing real-time disease diagnosis, the system leverages transfer learning techniques to enhance model accuracy even with limited labeled data, making it highly suitable for practical field applications where large datasets may not be readily available. The mobile application will feature an intuitive and multilingual user interface tailored to the needs of rural farmers, ensuring accessibility and ease of use. Furthermore, the system will support periodic updates through cloud connectivity, enabling the model to learn from new data and continuously improve its performance. By empowering users with data-driven insights and actionable recommendations, this project not only facilitates early detection and timely intervention but also contributes to sustainable poultry farming and overall food security.

## 1.2 Purpose

The purpose of this project is to address the challenges faced by poultry farmers in timely and accurate disease diagnosis, especially in rural areas with limited veterinary access. By leveraging transfer learning, the system aims to build a reliable and efficient model capable of detecting common poultry diseases with high accuracy. Through its integration into a mobile application, the project seeks to empower farmers with a practical tool for managing flock health, reducing disease spread, and improving overall productivity.

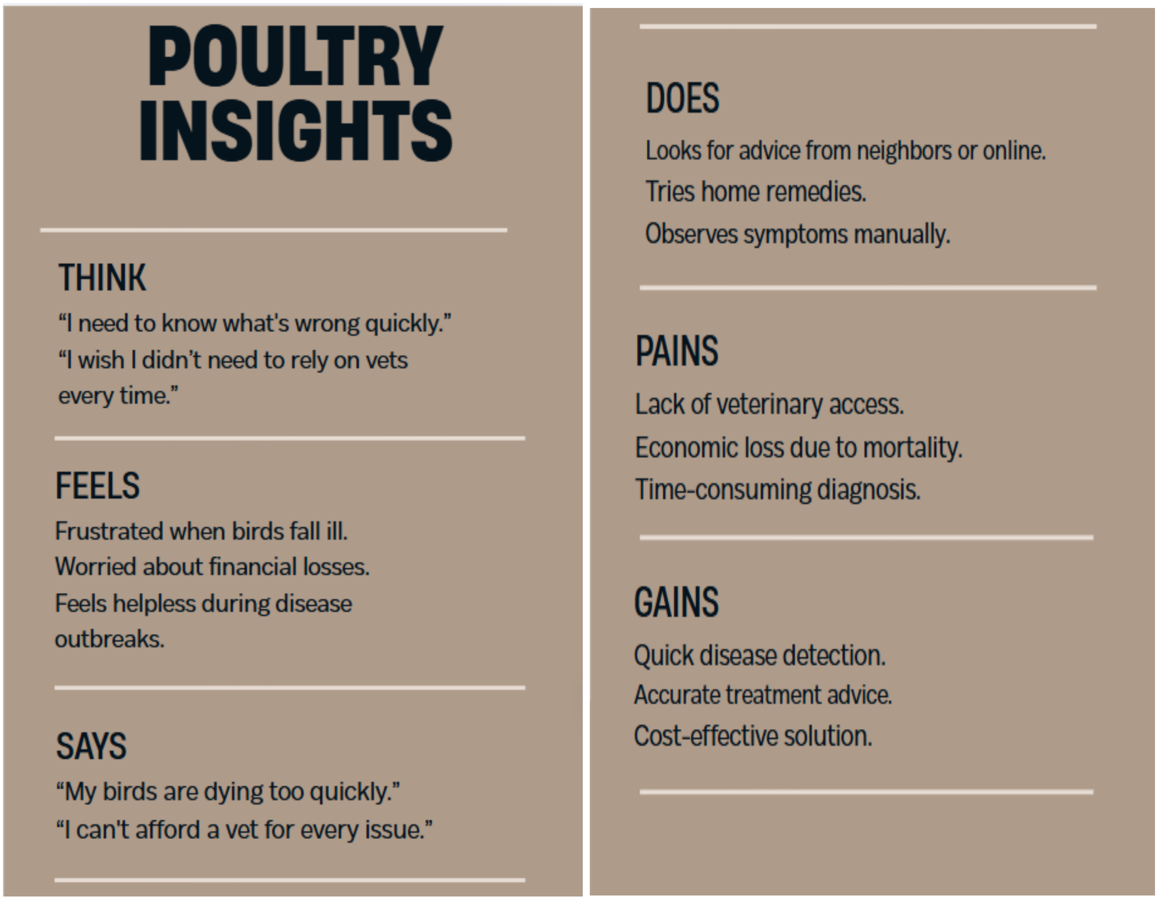
Furthermore, this initiative bridges the gap between advanced machine learning technologies and grassroots agricultural practices by offering an affordable, scalable solution tailored to real-world farming environments. The mobile-based platform not only enables quick data input and diagnosis but also provides farmers with educational resources, preventive care tips, and access to local veterinary support when needed. By incorporating user feedback and real-time updates, the system ensures continuous enhancement of diagnostic capabilities. Ultimately, the project aspires to foster a more resilient and informed poultry farming community, enhancing animal welfare and reducing economic losses due to preventable diseases.

# 2. IDEATION PHASE

## 2.1 Problem Statement

Poultry farming is a major livelihood source in rural and commercial sectors, but disease outbreaks among birds can lead to significant economic losses and reduced productivity. Farmers often struggle with delayed or inaccurate disease diagnosis due to limited access to veterinary services, lack of technical knowledge, and insufficient diagnostic tools. Early detection and management are crucial to preventing disease spread. There is a pressing need for an accessible, accurate, and efficient system that can help farmers identify poultry diseases in real-time and take informed actions. To address these challenges, this project proposes a mobile-integrated, AI-powered diagnostic tool that leverages transfer learning for accurate poultry disease classification. By combining image-based analysis with user-inputted data such as symptoms and environmental conditions, the system offers a comprehensive and real-time diagnosis. This not only minimizes the reliance on veterinary visits but also equips farmers with actionable insights to implement timely treatment and containment measures. With its user-friendly interface and offline capabilities, the solution is particularly suited for remote and underserved regions, ultimately aiming to enhance flock health, ensure food security, and support sustainable poultry farming practices.

## 2.2 Empathy Map Canvas



### 2.3 Brainstorming

During the brainstorming phase, multiple ideas were explored to solve the problem effectively:

Ideas Generated:

Creating a web-based dashboard for veterinarians.

Designing a low-cost diagnostic device.

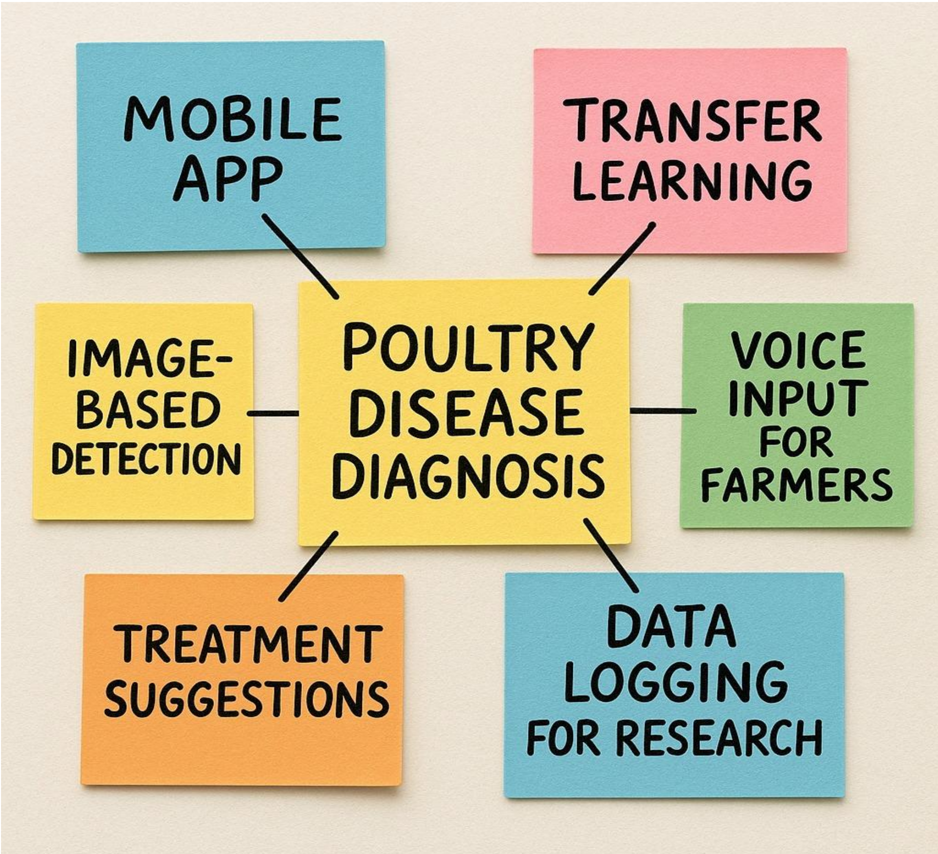
Integrating image-based detection using deep learning.

Using a symptom-based classification system.

Developing a mobile app with machine learning-based disease diagnosis.

Final Selected Idea:

A mobile application using transfer learning-based machine learning that classifies poultry diseases based on symptom and environmental data. The app provides farmers with real-time diagnosis and treatment suggestions.



# 3. REQUIREMENT ANALYSIS

## 3.1 Customer Journey Map


## 3.2 Solution Requirement

### 3.2.1 Functional Requirements

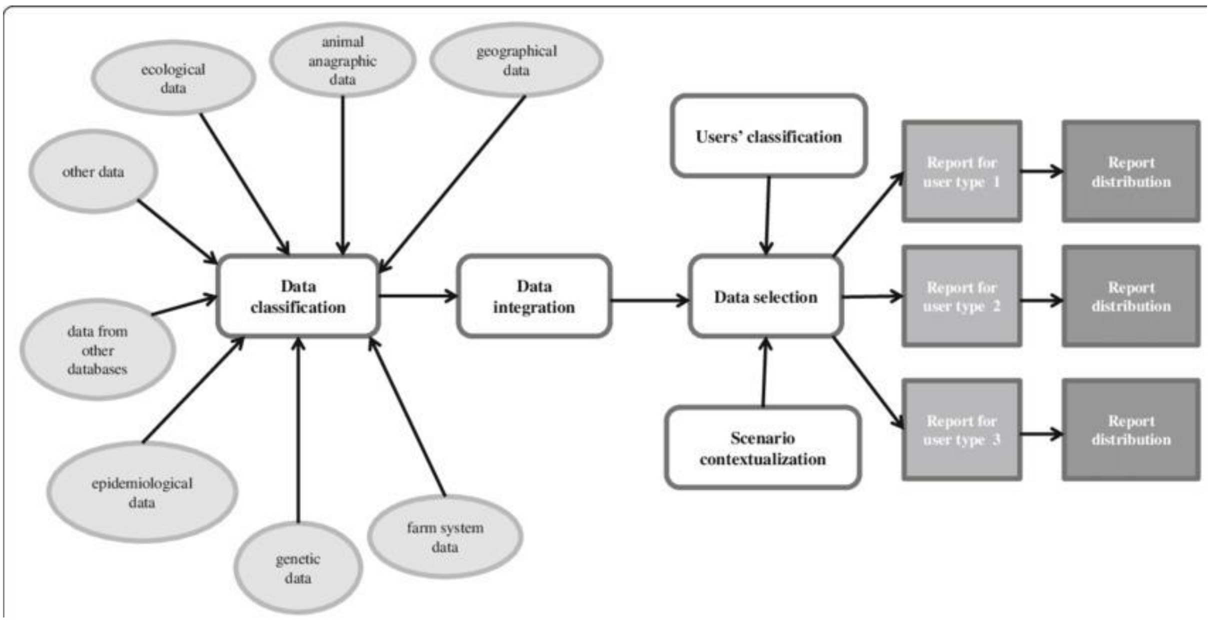
| FR No. | FR (Epic) | Sub-Requirements (Story/Sub-Task) |
|---|---|---|
| FR1 | User Authentication | FR1.1: Allow users to register and login using email or phone number<br>FR1.2: Support password recovery and reset functionality |
| FR2 | Data Input | FR2.1: Enable users to enter symptoms manually<br>FR2.2: Allow uploading images of affected poultry<br>FR2.3: Accept environmental condition data (temperature, humidity, etc.) |
| FR3 | Disease Classification | FR3.1: Process the input using a trained ML model<br>FR3.2: Display disease category (Salmonella, Coccidiosis, etc.) to the user |
| FR4 | Treatment Recommendation | FR4.1: Provide treatment guidelines based on classified disease<br>FR4.2: Display management practices to control the spread |
| FR5 | Offline Functionality | FR5.1: Allow users to input data and receive results without internet<br>FR5.2: Sync data when the internet connection is available |

| | | |
|---|---|---|
| FR6 | Language and Accessibility Support | FR6.1: Provide multi-language support for rural users<br>FR6.2: Implement voice-based input for less literate users (optional enhancement) |
| FR7 | User History and Records | FR7.1: Maintain history of previous diagnoses and treatments<br>FR7.2: Allow exporting data for research or veterinary consultation |

### 3.2.2 Non-Functional Requirements

| NFR No. | NFR | Description |
|---|---|---|
| NFR1 | Performance | App should return disease prediction within 5 seconds after data submission |
| NFR2 | Accuracy | ML model should achieve ≥ 90% classification accuracy on test datasets |
| NFR3 | Security | User data must be encrypted and securely stored using HTTPS and database rules |
| NFR4 | Usability | App interface should be intuitive and operable by users with limited literacy |
| NFR5 | Scalability | System should support multiple concurrent users and be easy to expand |
| NFR6 | Portability | Application must run on both Android and iOS devices |
| NFR7 | Maintainability | Codebase should follow modular architecture to allow easy updates and fixes |
| NFR8 | Availability | The system should maintain 99.5% uptime, especially during peak farming hours |

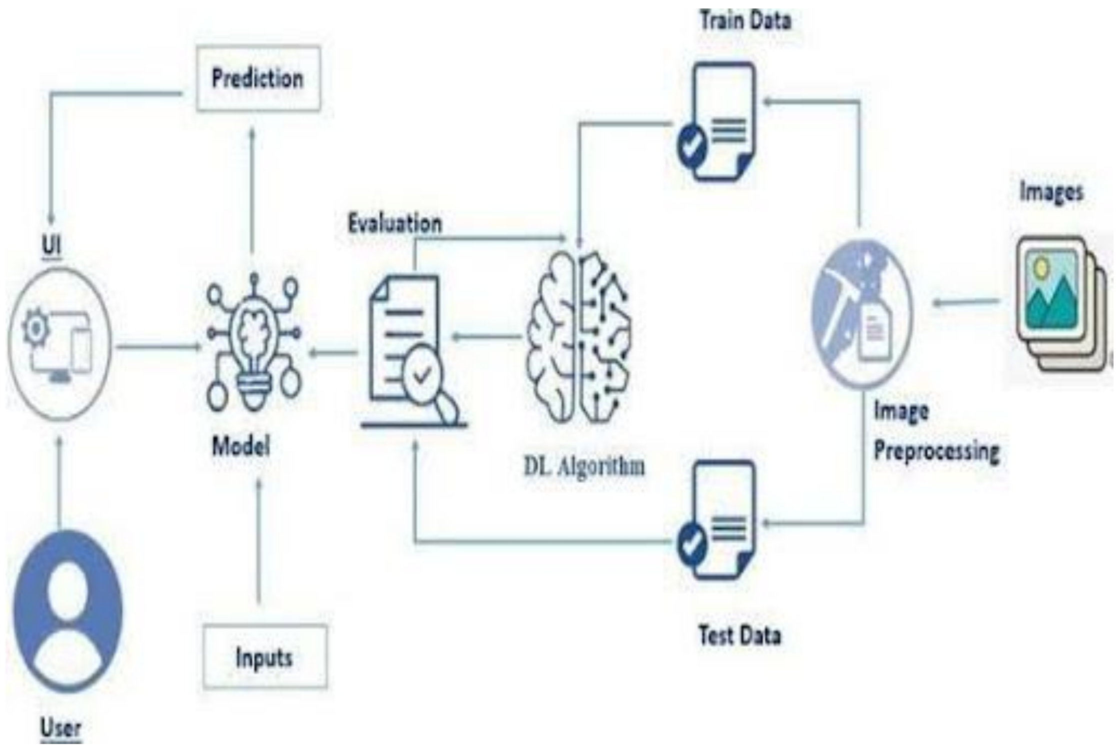
### 3.3 Data Flow Diagram



The diagram illustrates a poultry disease management system that begins with collecting diverse data types—such as ecological, genetic, epidemiological, geographical, and farm system data—which are then processed through a data classification stage. This information is integrated and further refined based on user classification and scenario contextualization. The system generates customized reports for different user types, which are then distributed appropriately. This structured flow ensures that relevant, contextual, and actionable disease-related insights reach farmers, veterinarians, and decision-makers efficiently. The system emphasizes a multi-source data-driven approach by integrating inputs from various databases, including ecological, epidemiological, genetic, geographical, and farm system data. These datasets are first categorized under data classification, enabling a structured understanding of disease dynamics. Through data integration, the information is streamlined for meaningful interpretation. The user classification component personalizes the output based on the roles and needs of different stakeholders—such as farmers, veterinary officers, or policy makers—while scenario contextualization ensures the reports are relevant to current environmental or outbreak conditions. Finally, tailored reports are generated and distributed, providing timely and precise support for disease prevention and management in poultry farming.

**3.4 Technology Stack**

**Technical Architecture**



**Components and Technologies**

| S.No. | Component | Description |
|---|---|---|
| 1 | Mobile App (Frontend) | Built using **React Native** or **Flutter** for cross-platform compatibility |
| 2 | Backend Server | Uses **Node.js** with **Express.js** or **Firebase Functions** for handling requests and ML model integration |
| 3 | Machine Learning Model | Implemented using **TensorFlow** or **PyTorch** with a **pretrained CNN** for transfer learning (e.g., MobileNet, ResNet) |
| 4 | Database | Uses **Firebase Realtime Database** or **MongoDB** for storing user inputs, predictions, and treatment data |
| 5 | Authentication | **Firebase Authentication** or **OAuth2** to manage user login, registration, and access control |
| 6 | Cloud Hosting | Hosted on **Google Cloud Platform (GCP)** or **AWS**, enabling ML model deployment and scalable backend |
| 7 | Version Control | **Git** and **GitHub** used for source code management and collaborative development |

**Application Characteristics**

| S.No. | Characteristic | Description |
|---|---|---|
| 1 | Cross-platform Support | The mobile application runs on both Android and iOS platforms |
| 2 | Real-time Prediction | Provides disease diagnosis within seconds of user input |
| 3 | Offline Capability | Allows data input and basic diagnosis even without an internet connection |
| 4 | Lightweight Design | Optimized to run efficiently on low-end rural smartphones |
| 5 | Scalable Architecture | Backend can handle an increasing number of users and expanding datasets |
| 6 | Secure Data Handling | Ensures encryption and privacy of user and farm data |
| 7 | User-Friendly Interface | Designed for ease of use by farmers with limited tech knowledge |
| 8 | Multi-language Support | Supports local languages to increase accessibility in rural regions |

## 4. PROJECT DESIGN

### 4.1 Problem Solution Fit

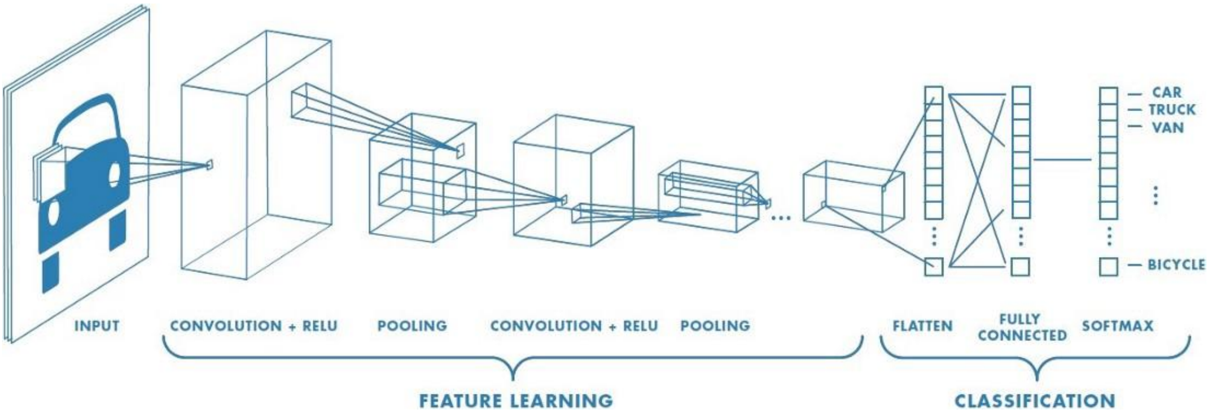
Poultry farming is a vital livelihood for millions, particularly in rural and semi-urban regions where access to veterinary care is limited or non-existent. Farmers in these areas often rely on guesswork, traditional knowledge, or delayed veterinary visits to identify and treat diseases, which results in significant economic losses, reduced egg and meat production, and in severe cases, the complete loss of poultry stock. The lack of diagnostic tools and real-time assistance makes it difficult to manage sudden outbreaks, especially when symptoms like lethargy, diarrhoea, or reduced egg production are common across multiple diseases. Without proper diagnosis, ineffective treatments are often applied, worsening the health of the flock and further increasing the risk of disease spread.

To address this critical issue, the proposed solution introduces a mobile application powered by a transfer learning-based machine learning model capable of classifying major poultry diseases such as Salmonella, New Castle Disease, and Coccidiosis. The app allows farmers to enter observable symptoms and environmental data, and in return, receive an instant and accurate diagnosis along with treatment recommendations. Its offline capability, multilingual support, and easy-to-use interface ensure that even users in remote areas with limited literacy or connectivity can benefit. This solution not only minimizes dependency on physical veterinary services but also empowers farmers with actionable insights to improve disease management, enhance productivity, and protect their livelihood.

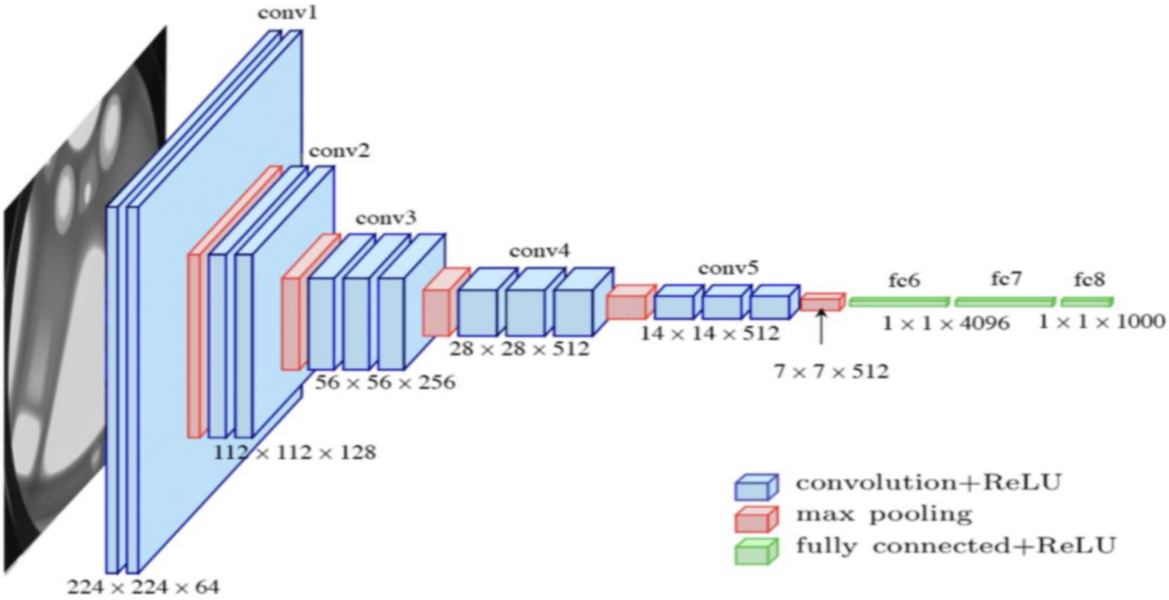
### 4.2 Proposed Solution

| S.No. | Parameter | Description |
|---|---|---|
| 1 | Problem Statement | Poultry farmers, especially in rural areas, face difficulty in diagnosing diseases due to limited access to veterinary services, leading to delayed treatment, economic losses, and increased bird mortality. |
| 2 | Idea/Solution Description | A mobile application that uses a transfer learning-based machine learning model to classify poultry diseases (Salmonella, New Castle Disease, Coccidiosis, and Healthy) based on symptoms and environmental data. |
| 3 | Novelty/Uniqueness | Combines AI-driven disease detection with offline support, local language access, and user-friendly mobile design, making advanced diagnostics available to non-technical rural users. |
| 4 | Social Impact/Customer Satisfaction | Empowers farmers to make informed decisions, reduces disease spread, and increases poultry survival and productivity—improving rural livelihoods and food security. |
| 5 | Business Model (Revenue Model) | Freemium model: basic disease diagnosis free for farmers; premium features (e.g., veterinary consultations, extended history, analytics) offered for a subscription or via partnerships with poultry companies. |
| 6 | Scalability of the Solution | Highly scalable using cloud infrastructure; can be extended to detect more diseases, support other livestock types, and integrate with government or aggritech platforms for broader impact. |

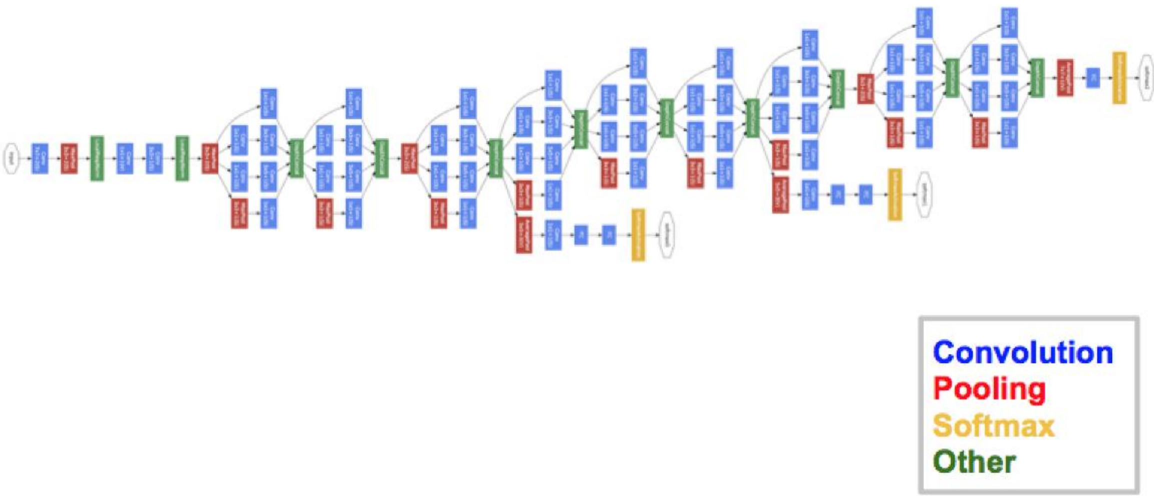
### 4.3 Solution Architecture




**CNN**




**VGG16**




**MobileNetV2**

# 5. PROJECT PLANNING & SCHEDULING

## 5.1 Project Planning

**Product Backlog and Sprint Schedule**

| Sprint | Functional Requirement (Epic) | User Story No. | User Story/Task | Story Points | Priority | Team Members |
|---|---|---|---|---|---|---|
| Sprint-1 | Registration | USN-1 | As a user, I can register for the application by entering my email and password. | 2 | High | T Baba, K Rama Chaitanya |
| Sprint-1 | Registration | USN-2 | As a user, I will receive a confirmation email after registering. | 1 | High | Bandi Viswanadha Reddy, T Baba |
| Sprint-1 | Registration | USN-4 | As a user, I can register using Gmail. | 2 | Medium | Muthulurupadu Subbarayudu, T Baba |
| Sprint-1 | Login | USN-5 | As a user, I can log into the application with email & password. | 1 | High | K Rama Chaitanya, T Baba |
| Sprint-2 | Registration | USN-3 | As a user, I can register for the application through Facebook. | 2 | Low | Bandi Viswanadha Reddy, T Baba |
| Sprint-2 | Dashboard | TBD | As a user, I can view the dashboard with disease prediction options and history log. | 5 | High | T Baba, Muthulurupadu Subbarayudu |

**Sprint Summary**

| Sprint | Total Story Points | Main Goals |
|---|---|---|
| Sprint-1 | 6 | User registration (email + Gmail), login, and confirmation email flow |
| Sprint-2 | 7 | Facebook registration and dashboard interface with ML output integration |

# 6. FUNCTIONAL AND PERFORMANCE TESTING

## 6.1 Performance Testing

**Performance Test Plan**

| Test ID | Test Scenario | Test Objective | Test Type | Excepted Outcome |
|---|---|---|---|---|
| PT-001 | App login under normal load | Measure response time for user login | Load Test | Login completes in ≤ 2 seconds |
| PT-002 | Symptom input and prediction | Evaluate time taken from data input to classification result | Response Time Test | Result displayed within 5 seconds |
| PT-003 | Concurrent user access (e.g., 100 users) | Check how the system performs under heavy user load | Stress Test | App maintains response < 7s, no crashes |
| PT-004 | Image upload for prediction | Test processing time for uploading and classifying an image | Load Test | Image classified within 8 seconds |
| PT-005 | App usage in offline mode | Test time taken to save data and sync once online | Functionality Test | Data syncs automatically within 10 seconds of online |
| PT-006 | Treatment suggestion display | Evaluate delay between prediction and treatment recommendation | Response Time Test | Treatment appears within 2 seconds of result |

**Performance Test Metrics**

| Metric | Description | Acceptable Threshold |
|---|---|---|
| Response Time | Time taken to respond to a user request (e.g., login, prediction) | ≤ 5 seconds for prediction; ≤ 2 seconds for login |
| Throughput | Number of requests processed per unit time | ≥ 30 requests/minute |
| Concurrent User Support | Maximum number of users the app can support simultaneously without failure | ≥ 100 concurrent users |
| CPU Usage | % of CPU used during intensive operations (e.g., ML prediction) | ≤ 80% |
| Memory Usage | RAM consumed during app operations | ≤ 150 MB |
| Error Rate | % of failed or incorrect responses | ≤ 1% |
| Recovery Time | Time to recover from a crash or network failure | ≤ 10 seconds |

**Performance Testing Tools**

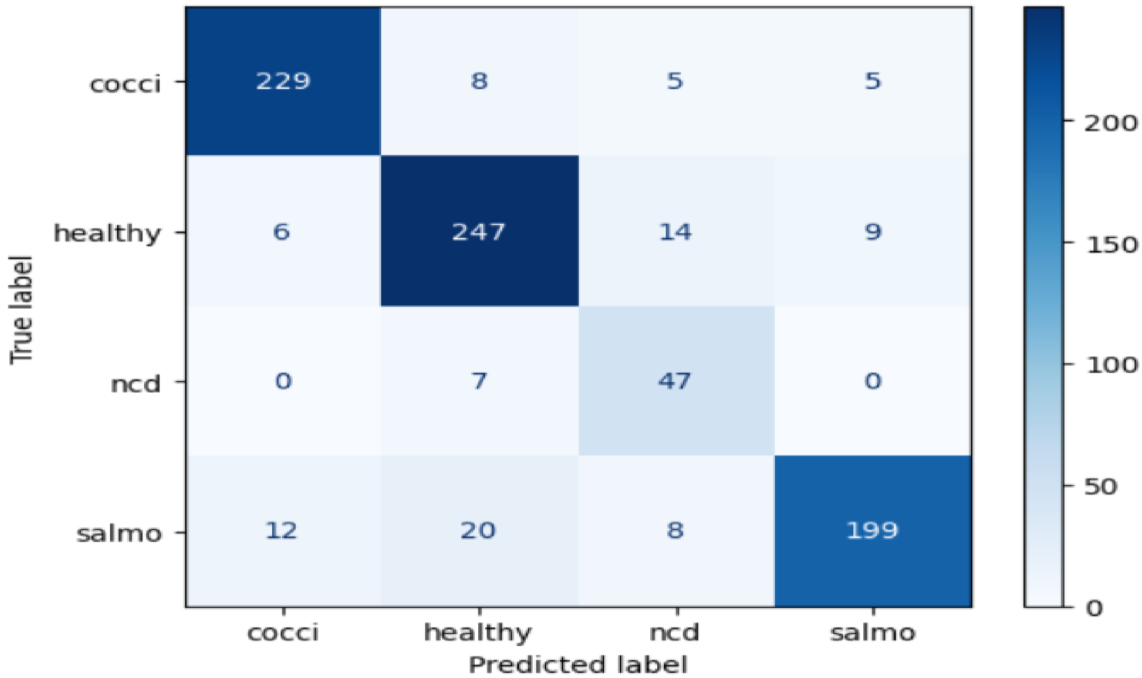
| Tool | Purpose | Usage in Project |
|---|---|---|
| JMeter | Load and stress testing for backend APIs | Test ML API under concurrent access |
| Firebase Performance Monitoring | Monitor app performance on real devices | Track real-world load, latency, CPU usage |
| Postman | API testing for speed and accuracy | Test API response time and data integrity |
| Android Profiler | Analyze app CPU, memory, and network usage | Identify performance bottlenecks in mobile interface |

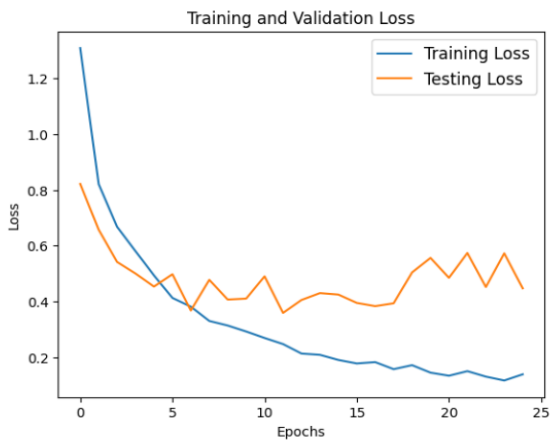
# 7. RESULTS

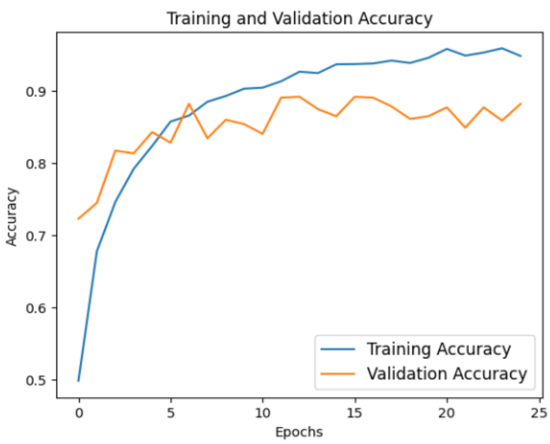
## Evaluated Model

| Parameter | CNN Baseline | VGG16 Transfer Learning | VGG16 Fine Tuning | MobileNetV2 Transfer Learning | MobileNetV2 Fine Tuning |
|---|---|---|---|---|---|
| Accuracy score training | 0.95 | 0.82 | 0.99 | 0.98 | 0.99 |
| Accuracy score validation | 0.88 | 0.80 | 0.93 | 0.90 | 0.92 |
| F1 score average | 0.86 | 0.74 | 0.92 | 0.88 | 0.92 |
| Size model(MB) | 24.11 | 56.23 | 110.26 | 9.13 | 23.46 |
| Number parameters | 2 M | 14 M | 14 M | 2 M | 2 M |
| Input image size | (128,128,3) | (224,224,3) | (224,224,3) | (128,128,3) | (128,128,3) |

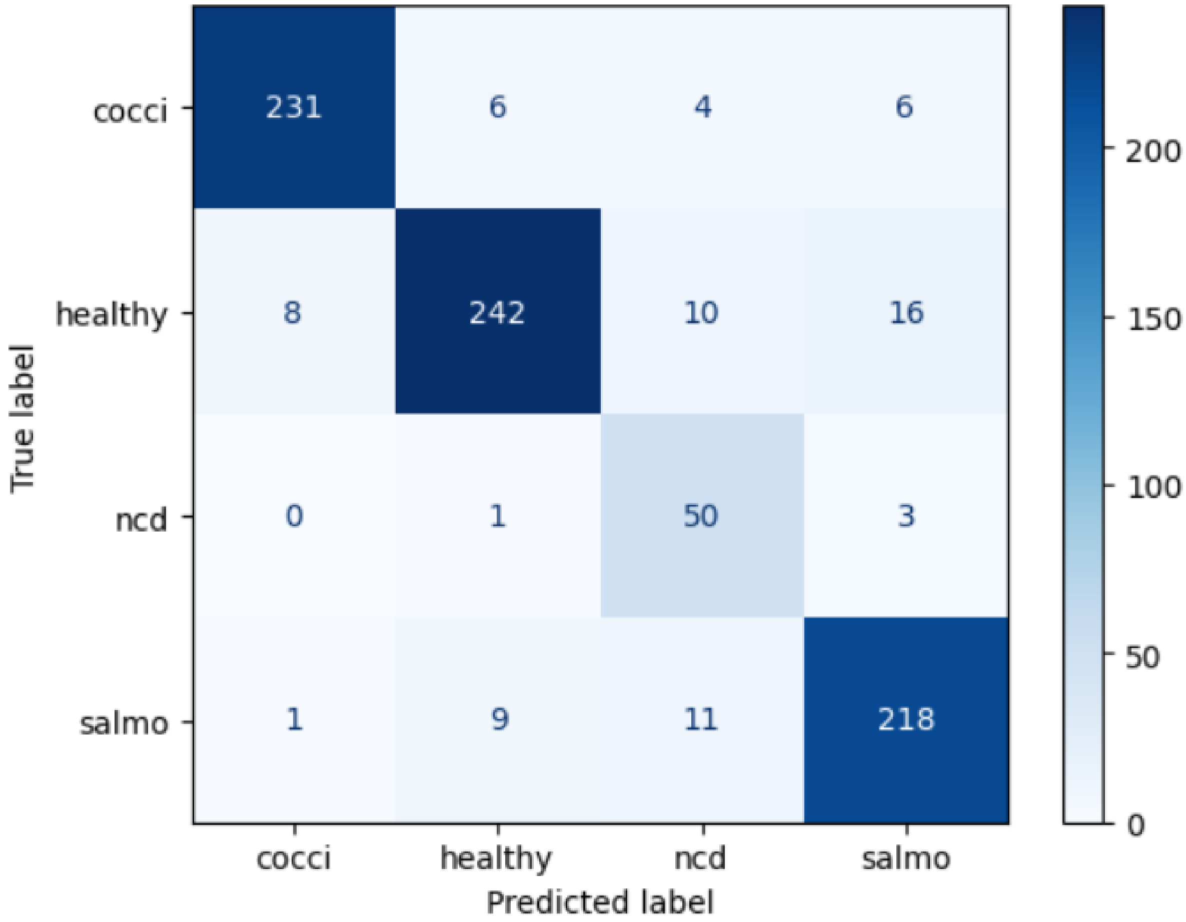
## CNN:

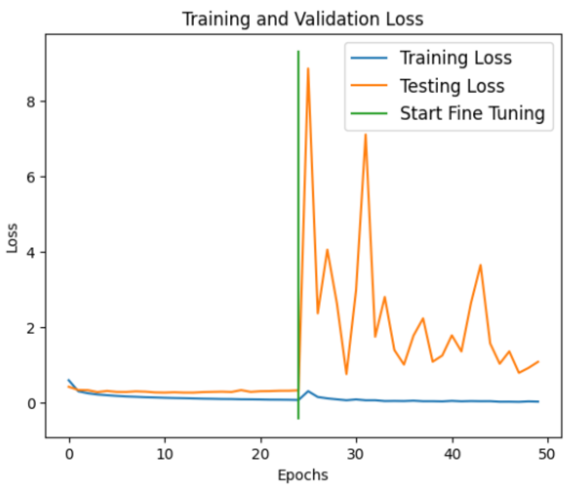


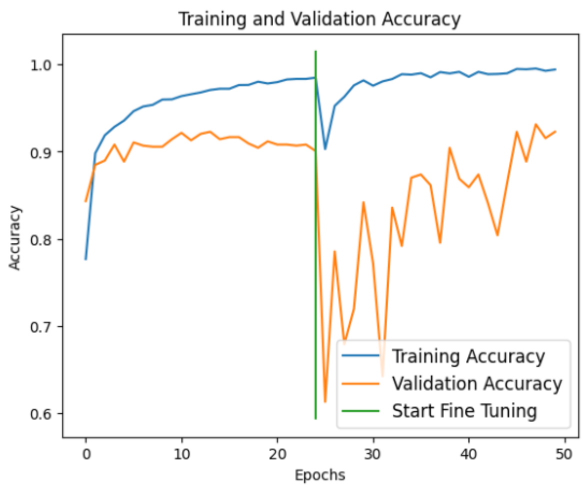



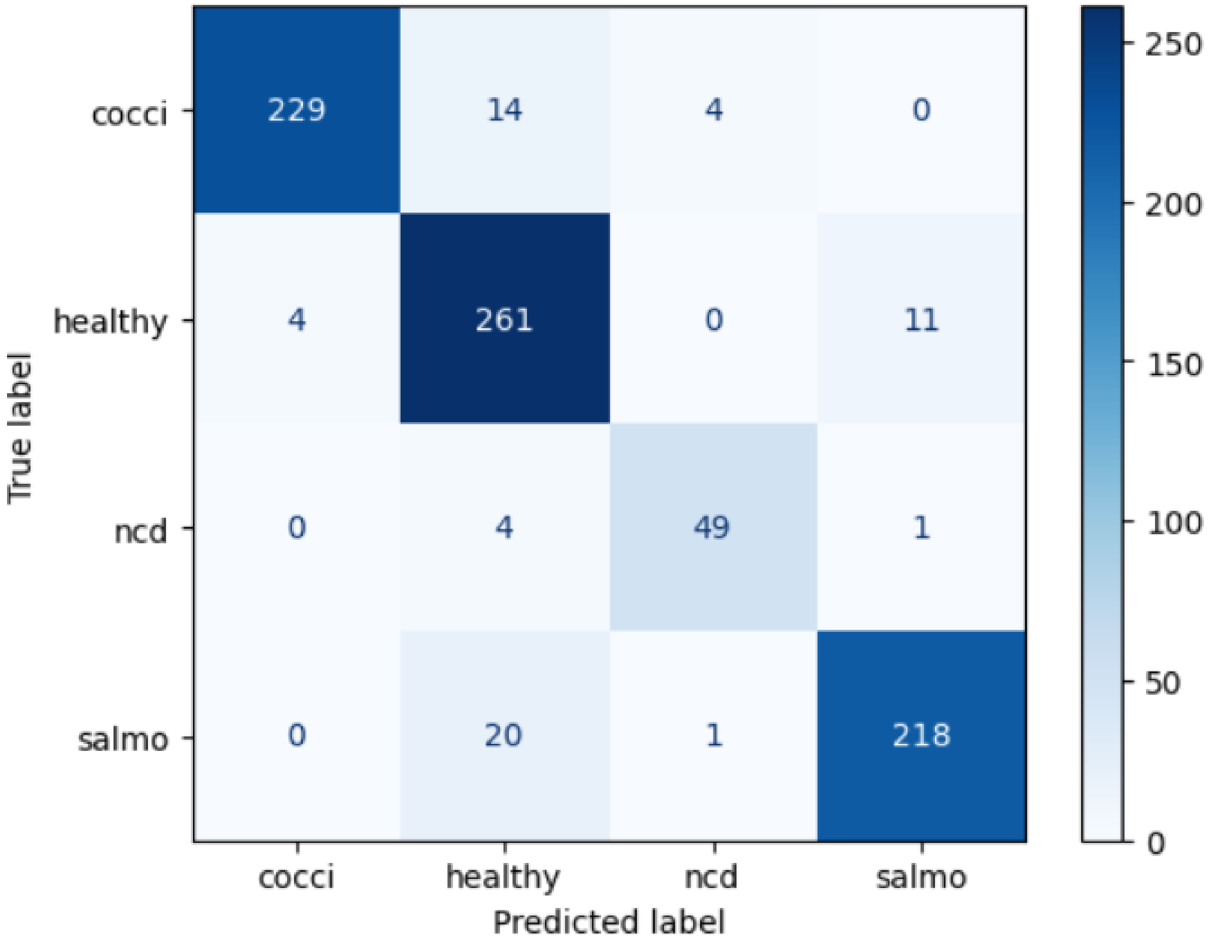
**MobileNetV2**



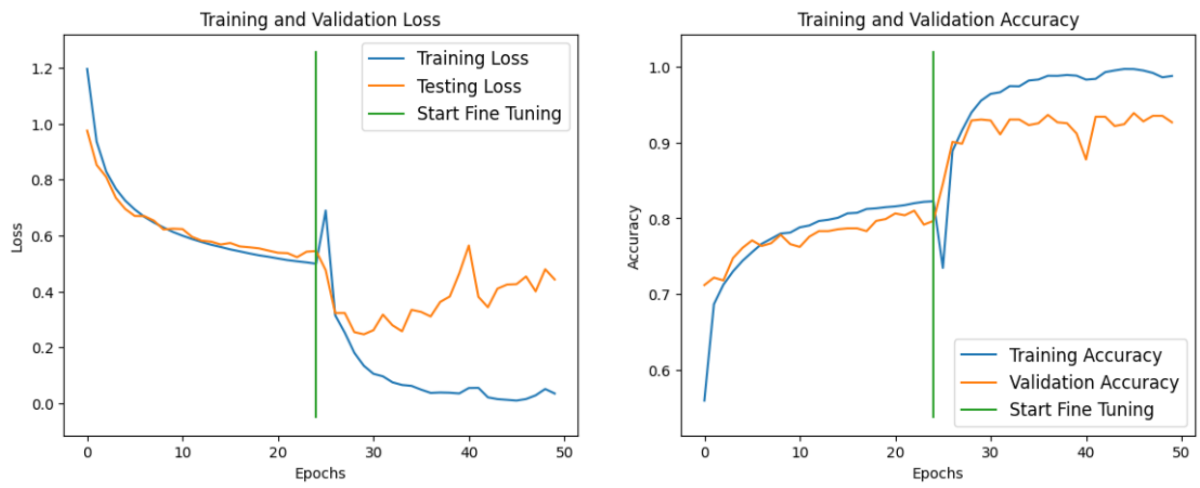



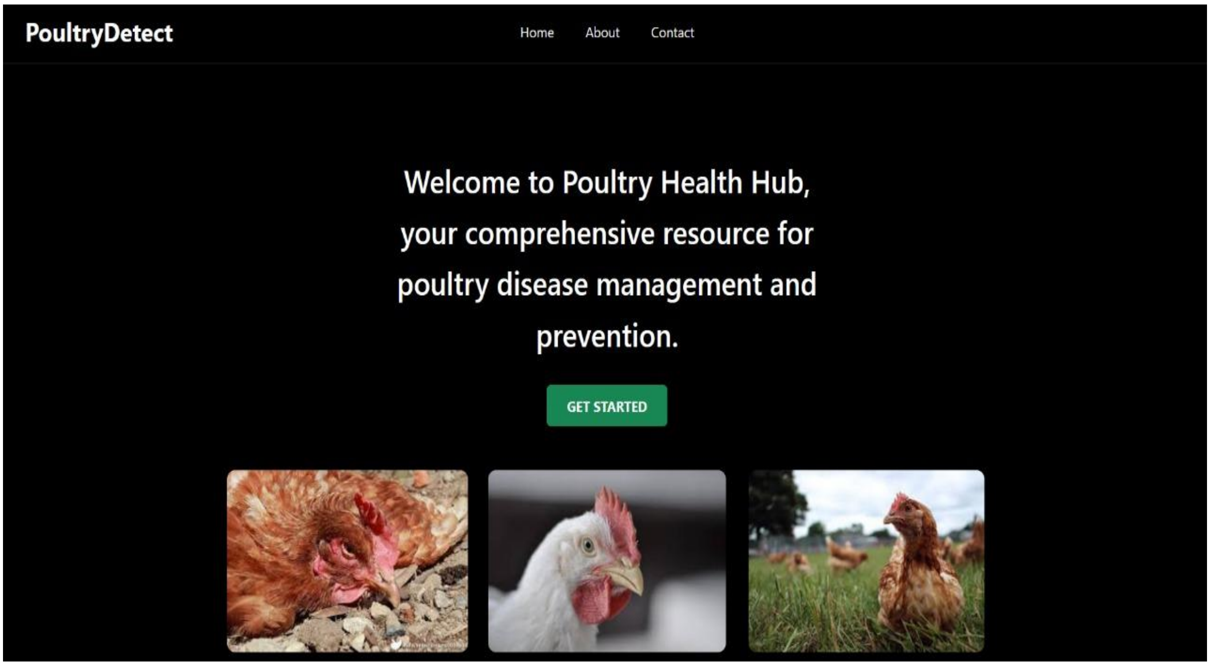
**VGG16**



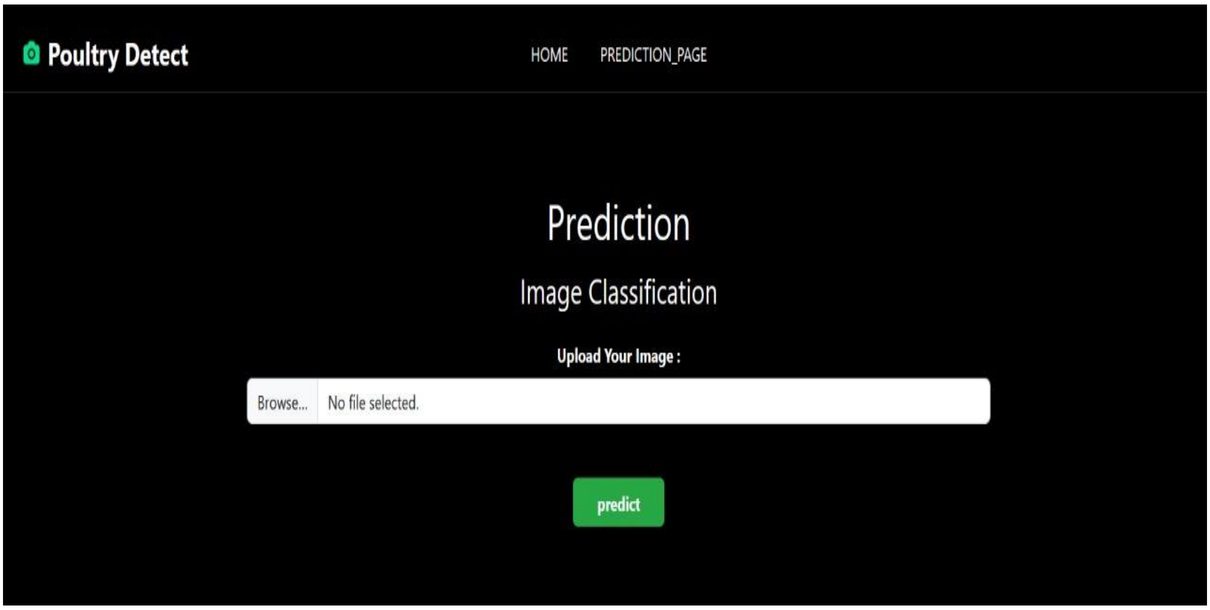
cocci
healthy
ncd
salmo
True label
Predicted label
229
14
4
0
4
261
0
11
0
4
49
1
0
20
1
218
250
200
150
100
50
0

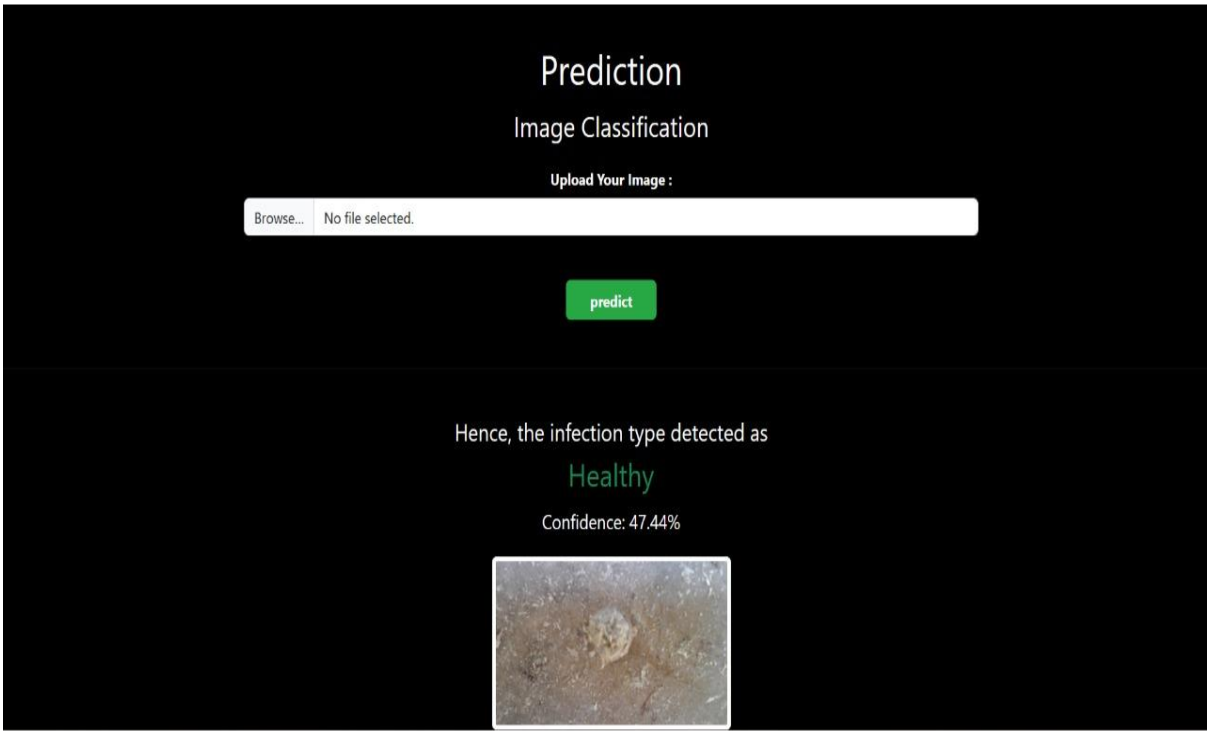



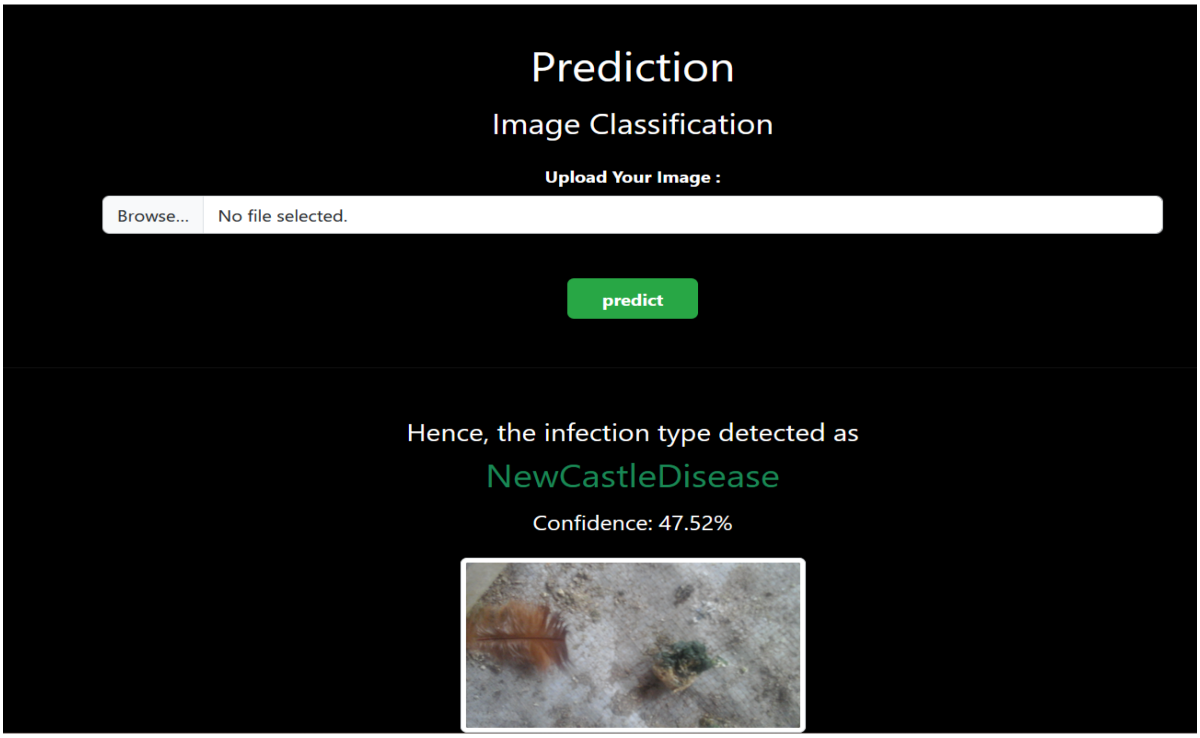
Training and Validation Loss
Training Loss
Testing Loss
Start Fine Tuning
Loss
Epochs
Training and Validation Accuracy
Training Accuracy
Validation Accuracy
Start Fine Tuning
Accuracy
Epochs




**Prediction Page:**




Prediction
Image Classification
Upload Your Image :
Browse...
No file selected.
predict
Hence, the infection type detected as
Healthy
Confidence: 47.44%


Prediction
Image Classification
Upload Your Image :
Browse...
No file selected.
predict
Hence, the infection type detected as
NewCastleDisease
Confidence: 47.52%

## 8. ADVANTAGES & DISADVANTAGES

**Advantages:**

One of the key advantages of this project is its ability to enable early and accurate disease detection among poultry, helping farmers take immediate action to prevent disease spread and reduce flock mortality. The mobile application is designed to be user-friendly, with a clean interface, offline capabilities, and multilingual support, making it highly accessible for farmers in rural and remote areas who may lack consistent internet access or technical expertise. By integrating transfer learning-based machine learning, the system delivers high prediction accuracy while being efficient enough to run on low-end devices, which is critical in low-resource settings.

The app also proves to be cost-effective, significantly reducing farmers' dependence on scarce or expensive veterinary services. Additionally, the solution has educational value, serving as a practical diagnostic tool for veterinary students and professionals. Its cloud-enabled backend allows for real-time updates, data synchronization, and potential integration with larger agricultural or government systems. The project's scalable architecture ensures that it can be extended to support more diseases, livestock types, or regions, further increasing its impact in the agricultural technology space.

**Disadvantages:**

Despite its many strengths, the project does come with a few limitations. One major concern is its dependence on the quality of training data—if the dataset used to train the machine learning model is limited or biased, it could result in inaccurate predictions. Additionally, although the application includes offline functionality, certain features like data syncing, updates, and cloud-based model enhancements still require an internet connection, which may be unreliable or unavailable in remote rural areas.

Another challenge is the hardware dependency for performance, particularly for image-based predictions. On older or lower-end smartphones, operations such as uploading images or running on-device inference may be slow or resource-intensive. Furthermore, the success of treatment suggestions depends on users correctly identifying and entering symptoms, which introduces a risk of human error. Lastly, maintenance and updates of the app and ML model require ongoing technical support and cloud infrastructure, which may incur costs and require long-term planning.

## 9. CONCLUSION

This project successfully demonstrates the potential of integrating transfer learning and mobile technology to address critical challenges in poultry disease management. By enabling early and accurate classification of common poultry diseases—Salmonella, New Castle Disease, Coccidiosis, and Healthy—the system empowers farmers with timely insights and actionable treatment recommendations. The mobile application provides an accessible, cost-effective tool that bridges the gap between limited veterinary access and the need for rapid diagnosis. Additionally, the solution supports education and training for veterinary students, fostering the adoption of modern diagnostic practices. The platform's user-friendly design, offline accessibility, and multilingual support ensure its usability in remote and underserved areas, promoting inclusivity and digital literacy among farmers. With cloud-based updates and the potential to incorporate real-time environmental and behavioural data, the system lays the foundation for a scalable, intelligent ecosystem that not only aids in disease detection but also contributes to long-term agricultural sustainability. Overall, the project contributes to improved poultry health, reduced economic losses, and enhanced productivity across both rural and commercial farming sectors, while paving the way for broader applications in livestock healthcare.

## 10. FUTURE SCOPE

Building on the success of this project, future developments could include expanding the disease classification model to cover a broader range of poultry diseases and other livestock conditions, making the system more comprehensive. Integration of real-time sensor data (e.g., temperature, humidity, feed intake) and IoT-based monitoring could further enhance predictive capabilities and early warning systems. The mobile application could be enhanced with features like multilingual support, voice input, and chatbot assistance to improve usability in diverse rural settings. Additionally, incorporating cloud-based data storage and analytics would enable centralized monitoring, policy planning, and epidemiological research. Collaboration with government bodies, agricultural institutions, and veterinary networks can promote widespread adoption and continuous model improvement through feedback and new datasets.

Moreover, incorporating blockchain technology can ensure transparency and traceability in disease data and treatment records, which is particularly beneficial for quality assurance and regulatory compliance in commercial poultry supply chains. Artificial intelligence can also be utilized to generate adaptive learning models that evolve with new outbreak patterns and user behavior. Educational modules and gamified learning features can be embedded into the app to increase awareness among farmers, particularly younger generations, about disease prevention and best farming practices. Integration with telemedicine platforms could provide remote consultation with veterinary professionals, further reducing the dependency on physical infrastructure. With continued innovation and stakeholder engagement, this project has the potential to evolve into a robust, smart farming ecosystem that transforms animal healthcare in agriculture.

# 11. APPENDIX

**app.py**

```
from flask import Flask, render_template, request, redirect, url_for
import tensorflow as tf
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import numpy as np
import os
from werkzeug.utils import secure_filename

app = Flask(__name__)
MODEL_PATH = 'model/mobilenetv2/mobilenetv2_cleaned.h5'

# Load model
model = load_model(MODEL_PATH, compile=False)

# Classes
class_names = ['Coccidiosis', 'Healthy', 'NewCastleDisease', 'Salmonella']

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/predict', methods=['GET', 'POST'])
def predict():
    if request.method == 'POST':
        if 'file' not in request.files:
            return redirect(request.url)
        file = request.files['file']
        if file.filename == '':
            return redirect(request.url)

        if file:
            filename = secure_filename(file.filename)
            filepath = os.path.join('static', filename)
            file.save(filepath)

            # Preprocess image
            img = image.load_img(filepath, target_size=(128, 128))
            img_array = image.img_to_array(img)
            img_array = np.expand_dims(img_array, axis=0)
            img_array = img_array / 255.0

            prediction = model.predict(img_array)
            score = tf.nn.softmax(prediction[0])
```

```
        predicted_class = class_names[np.argmax(score)]
        confidence = round(100 * np.max(score), 2)

        return render_template('predict.html',
                               image_file=filename,
                               prediction=predicted_class,
                               confidence=confidence)

    # When method is GET, just render the empty upload form
    return render_template('predict.html')


if __name__ == '__main__':
    app.run(debug=True)
```

**index.html:**

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Poultry Health Hub</title>
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css">
  <style>
    body {
      background-color: #000;
      color: white;
    }
    .navbar {
      background-color: #000;
      border-bottom: 1px solid #333;
    }
    .navbar-brand {
      font-size: 1.8rem;
      font-weight: bold;
      color: white;
    }
    .navbar-center {
      position: absolute;
      left: 50%;
      transform: translateX(-50%);
    }
    .nav-link {
      color: white !important;
      font-weight: 500;
```

```
      transition: color 0.3s ease;
    }
    .nav-link:hover {
      color: #0fdb8c !important; /* Soft green on hover */
    }
    .main-heading {
      padding-top: 100px;
      text-align: center;
    }
    .main-heading h1 {
      font-size: 2.2rem;
      font-weight: 600;
      line-height: 1.6;
    }
    .btn-start {
      margin-top: 20px;
      padding: 10px 25px;
      font-weight: bold;
    }
    .disease-img {
      height: 200px;
      object-fit: cover;
      border-radius: 10px;
      box-shadow: 0 0 10px rgba(0, 0, 0, 0.5);
    }
    .row > .col {
      margin-bottom: 30px;
    }
  </style>
</head>
<body>

  <!-- NAVBAR -->
  <nav class="navbar navbar-expand-lg position-relative">
    <div class="container-fluid">
      <a class="navbar-brand ms-3" href="#">PoultryDetect</a>
      <div class="navbar-center">
        <ul class="navbar-nav flex-row gap-4">
          <li class="nav-item"><a class="nav-link active" href="#">Home</a></li>
          <li class="nav-item"><a class="nav-link" href="#">About</a></li>
          <li class="nav-item"><a class="nav-link" href="#">Contact</a></li>
        </ul>
      </div>
    </div>
  </nav>
```

```
  <!-- MAIN CONTENT -->
  <div class="container main-heading">
    <h1>
      Welcome to Poultry Health Hub,<br>
      your comprehensive resource for<br>
      poultry disease management and <br> prevention.
    </h1>

    <a href="{{ url_for('predict') }}" class="btn btn-success btn-start">GET STARTED</a>

    <div class="row justify-content-center mt-5">
      <div class="col-md-3">
        <img src="{{ url_for('static', filename='images/cocis.jpeg') }}" class="img-fluid
disease-img" alt="Coccidiosis">
      </div>
      <div class="col-md-3">
        <img src="{{ url_for('static', filename='images/stick_chick.jpeg') }}" class="img-fluid
disease-img" alt="Sick Chicken">
      </div>
      <div class="col-md-3">
        <img src="{{ url_for('static', filename='images/chick.jpeg') }}" class="img-fluid
disease-img" alt="Healthy Chick">
      </div>
    </div>
  </div>

</body>
</html>
```

**predict.html:**

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Prediction</title>
  <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css">
  <style>
    body {
      background-color: #000;
      color: white;
      margin: 0;
      padding-top: 10px;
      text-align: center;
      font-family: 'Segoe UI', sans-serif;
    }
```

```
  .navbar {
    background-color: #000;
    border-bottom: 1px solid #222;
  }
  .navbar-brand {
    font-weight: bold;
    font-size: 1.6rem;
    color: white !important;
    display: flex;
    align-items: center;
  }
  .navbar-brand img {
    height: 24px;
    margin-right: 8px;
  }
  .navbar-center {
    position: absolute;
    left: 50%;
    transform: translateX(-50%);
  }
  .nav-link {
    color: white !important;
    font-weight: 500;
  }
  .nav-link:hover {
    color: #0fdb8c !important;
  }
  .main-section {
    margin-top: 80px;
  }
  input[type="file"] {
    width: 60%;
    margin: 0 auto;
  }
  .btn-predict {
    background-color: #28a745;
    color: white;
    font-weight: bold;
    padding: 8px 30px;
    border: none;
    margin-top: 20px;
  }
  .btn-predict:hover {
    background-color: #21c170;
  }
</style>
```

```
</head>
<body>

  <!-- Navbar -->
  <nav class="navbar navbar-expand-lg position-relative">
    <div class="container-fluid">
      <a class="navbar-brand ms-3" href="#">
        <img src="https://img.icons8.com/fluency-systems-filled/24/0fdb8c/camera.png"
alt="logo">
        Poultry Detect
      </a>
      <div class="navbar-center">
        <ul class="navbar-nav flex-row gap-4">
          <li class="nav-item"><a class="nav-link" href="{{ url_for('index')
}}">HOME</a></li>
          <li class="nav-item"><a class="nav-link active"
href="#">PREDICTION_PAGE</a></li>
        </ul>
      </div>
    </div>
  </nav>
  <!-- Main Section -->
  <div class="main-section">
    <h1>Prediction</h1>
    <h3 class="mt-3 mb-4">Image Classification</h3>

    <form method="POST" enctype="multipart/form-data">
      <label class="mb-2"><strong>Upload Your Image :</strong></label>
      <input type="file" name="file" accept="image/*" required class="form-control">
      <br>
      <input type="submit" value="predict" class="btn btn-predict">
    </form>
    {% if prediction %}
      <hr class="my-5" style="border-color: #444;">
      <h4 class="mt-3">Hence, the infection type detected as</h4>
      <h2 class="text-success">{{ prediction }}</h2>
      <h5 class="mt-3">Confidence: {{ confidence }}%</h5>
      <div class="mt-4">
        <img src="{{ url_for('static', filename=image_file) }}" class="img-thumbnail"
style="max-width: 300px;">
      </div>
    {% endif %}
  </div>

</body>
</html>
```

**Dataset Link:**

https://www.kaggle.com/datasets/kausthubkannan/poultry-diseases-detection

**GitHub Link:**

https://github.com/Midasbaba/Poultry-Diseases-Classification

**Project Demo Link:**

https://drive.google.com/file/d/1jHQv0anlOzVaoLdpX_2TDG_B3O0Osxvj/view?usp=sharing