# AI-Powered Pest Control System For Sustainable Agriculture

A Dissertation

*Submitted in partial fulfilment of the requirements for the award of the degree of*

BACHELOR OF TECHNOLOGY IN

### Computer Science and Engineering – Artificial Intelligence

By

**Ravula VamsiKrishna 21JR1A43I0**

**V.V.M. Sandeep 21JR1A43J6**

**Parimi Giri Babu 21JR1A43H4 Nimmakayala Durga Prasad 21JR1A43H0**

Under the guidance of Mrs. K. BhanuSri,

Assistant Professor, Dept. of CSE – AI



## DEPARTMENT OF CSE-AI

**KKR & KSR INSTITUTE OF TECHNOLOGY AND SCIENCES**

### (Autonomous)

Vatticherukuru (V), Vatticherukuru (M), Guntur (Dt), AP-522017.

## APRIL-2025.

**COMPUTER SCIENCE AND ENGINEERING-ARTIFCIAL INTELLIGENCE**

## KKR & KSR INSTITUTE OF TECHNOLOGY AND SCIENCES

### (Autonomous)

(Approved by AICTE New Delhi || Permanently Affiliated to JNTUK, Kakinada || Accredited with ‘A’ Grade by NAAC || NBA Accreditation)

Vinjanampadu (V), Vatticherukuru (M), Guntur (Dt), A.P-522017.



**CERTIFICATE**

This is to certify that this project report entitled “AI-Powered Pest Control System For Sustainable Agriculture” submitted by RAVULA VAMSIKRISHNA (21JR1A43I0), V.V.M. SANDEEP (21JR1A43J6), PARIMI GIRI BABU (21JR1A43H4), NIMMAKAYALA DURGA PRASAD (21JR1A43H0) to Jawaharlal Nehru Technological University Kakinada, through KKR & KSR Institute of Technology and Sciences for the award of the Degree of Bachelor of Technology in Computer Science and Engineering-Artificial Intelligence is a Bonafide record of project work carried out by them under my supervision during the year 2024-2025.

Mrs. K. BhanuSri, Dr. G. Murali,

**SUPERVISOR, HEAD OF THE DEPARTMENT**

Assistant Professor

**INTERNAL EXAMINER EXTERNAL EXAMINER**

## DECLARATION

We here by declare that the project “AI-Powered Pest Control System For Sustainable Agriculture” has been carried out by us and this work has been submitted to KKR & KSR Institute of Technology and Sciences (A), Vinjanampadu, affiliated to Jawaharlal Technological University, Kakinada in partial fulfilment of the requirements for the award of the Degree of Bachelor of Technology in Computer Science and Engineering-Artificial Intelligence.

We here by declare that this project work has not been submitted in full or part for the award of any other degree in any other educational institutions.

-

|  |  |
| --- | --- |
| 1. 21JR1A43I0 | Ravula VamsiKrishna |
| 2. 21JR1A43J6 | V.V.M. Sandeep |
| 3. 21JR1A43H4 | Parimi Giri Babu |
| 4. 21JR1A43H0 | Nimmakayala Durga prasad |

-

-

-

## ACKNOWLEDGEMENT

We would like to express our profound gratitude towards **Mrs. K. BhanuSri**, Department of Computer Science and Engineering-Artificial Intelligence, who played a supervisory role to perfection, enabled us to seek through our IV-II B. Tech project and for guidance as an internal guide methodically and meticulously.

We express our gratitude towards all the faculty members and non-teaching faculty members, the Computer Science and Engineering-Artificial Intelligence.

We are highly indebted to Dr. G. Murali, **Head of the Department**, Computer Science and Engineering-Artificial Intelligence for providing us with all the necessary support.

We render our deep sense of gratitude to **Dr. P. Babu, Principal and Dr. K. Hari Babu, Director of Academics** for permitting us to carry out our main project works. We would like to express our sincere thanks to Computer Science and Engineering-Artificial Intelligence staff for leading us their time to help us complete the work successfully.

We are very much thankful to the **college management Sri K. Subba Rao, Chairman and Sri K. Sekhar, Secretary** for their continuous support and facilities provided. We would also like to thank our staff, parents, and friends for their enduring encouragement and assistance whenever required.

|  |  |
| --- | --- |
| 1. 21JR1A43I0 | Ravula VamsiKrishna |
| 2. 21JR1A43J6 | V.V.M. Sandeep |
| 3. 21JR1A43H4 | Parimi Giri Babu |
| 4. 21JR1A43H0 | Nimmakayala Durga prasad |

Institute Vision and Mission

**INSTITUTION VISION**

To produce eminent and ethical Engineers and Managers for society by imparting quality professional education with emphasis on human values and holistic excellence.

**INSTITUTION MISSION**

* To incorporate benchmarked teaching and learning pedagogies in curriculum.
* To ensure all round development of students through judicious blend of curricular, co- curricular and extra-curricular activities.
* To support cross-cultural exchange of knowledge between industry and academy.
* To provide higher/continued education and research opportunities to the employees of the institution.

# Department of CSE-Artificial Intelligence

##### Vision of the Department

* To be a renowned department for education in Artificial Intelligence and empowering students into professional engineers with human values and holistic excellence.

##### Mission of the Department

* Impart rigorous training to acquire knowledge through the state-of-the-art concepts and technologies in Artificial Intelligence.
* Train students to be technically competent through innovation and leadership.
* Inculcate values of professional ethics, social concerns, life-long learning and environment protection.
* Establish centers of excellence in leading areas of computing and artificial intelligence.

# Program Specific Outcomes (PSOs)

##### PSO1: Application Development

Apply the concepts in core area of Artificial Intelligence, Data Structure, Database System, Operating System, Networking and Intelligence System to solve futuristic problems.

##### PSO2: Computing Paradigms

Develop automated solutions for real world problems through laboratory experiments, projects and internship.

### Program Educational Objectives (PEOs)

|  |  |
| --- | --- |
| **PEO: 1** | Graduates of Computer Science and Engineering – Artificial Intelligence  shall apply appropriate theory, practices, and tools to provide solution for multidisciplinary challenges. |
| **PEO: 2** | Graduates of Computer Science and Engineering - Artificial Intelligence shall  have an ability to function effectively in the workplace for professional growth. |
| **PEO: 3** | Graduates of Computer Science and Engineering shall have exposure to adapt, contribute and innovate new technologies in the key domains of  Artificial Intelligence during higher studies or product development. |

# Program Outcomes

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

# Course Outcomes:

|  |  |
| --- | --- |
| S.No. | **B. Tech Project CO** |
| CO 1 | Interact with customers and identify real world problem statement / identify  problems in engineering and technology in selected field of interest. |
| CO 2 | Synthesize and apply prior knowledge of mathematics, computer science and  engineering to design and implement solutions to open-ended problems. |
| CO 3 | Design and Develop the software with Software Engineering practices and  standards. |
| CO 4 | Use different tools for communication, design, implementation, testing and  report writing. |
| CO 5 | Analyzing professional issues, including ethical, legal and security issues, related to software project. |
| CO 6 | Develop better interpersonal communication skills, team work and leadership  qualities with writing and oral presentation skills. |

**Course Outcomes – Program Outcomes mapping**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **PO**  **1** | **PO**  **2** | **PO**  **3** | **PO**  **4** | **PO**  **5** | **PO**  **6** | **PO**  **7** | **PO**  **8** | **PO**  **9** | **PO**  **10** | **PO**  **11** | **PO**  **12** | **PSO**  **1** | **PSO**  **2** |
| **CO421.1** | 1 | 2 |  |  |  |  |  |  | 1 | 3 |  |  |  | 2 |
| **CO421.2** | 2 | 2 | 1 | 3 |  |  |  |  |  |  |  | 1 |  | 3 |
| **CO421.3** |  |  | 2 |  | 2 | 3 | 1 | 1 |  |  |  |  | 3 |  |
| **CO421.4** |  |  |  | 1 | 3 | 2 | 2 |  |  | 2 | 1 |  | 2 |  |
| **CO421.5** |  |  |  |  |  | 1 | 2 | 3 |  |  |  |  |  |  |
| **CO421.6** |  |  |  |  |  |  |  |  | 3 | 1 | 2 | 2 |  |  |
| Average | 1.5 | 2 | 1.5 | 2 | 2.5 | 2 | 1.6 | 2 | 2 | 2 | 1.5 | 1.5 | 2.5 | 2.5 |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **Tab. No.** | **Label** | **Page No.** |
| 1.1 | LITERATURE REVIEW | 6 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Figure.No.** | **Label** | **PageNo.** |
| 4.1 | CONCEPTUAL DESIGN | 17 |
| 4.2 | LOGICAL DESIGN | 19 |
| 4.3 | ARCHITECTURAL DESIGN | 20 |
| 4.4 | USE CASE DIAGRAM | 25 |
| 4.5 | CLASS DIAGRAM | 26 |
| 4.6 | SEQUENCE DIAGRAM | 27 |
| 4.7 | PROCESS FLOW DIAGRAM | 28 |
| 4.8 | DATABASE DESIGN | 30 |
| 4.9 | IDENTIFYING PEST | 38 |
| 4.10 | DETECTING DIFFERENT TYPES OF PEST | 38 |
| 4.11 | TYPES OF PESTS | 39 |
| 4.12 | SAMPLE OUTPUT SCREEN | 41 |
| 5.1 | DIFFERENT TYPES OF HARMFUL PESTS | 52 |
| 5.2 | NORMALIZING A PEST | 53 |
| 5.3 | GRAPHS ABOUT PESTS | 53 |
| 5.4 | OPENING ANACONDA PROMPT | 54 |
| 5.5 | RUNNING PATHS | 55 |
| 5.6 | LOGIN PAGE | 55 |
| 5.7 | FINAL OUTPUT | 55 |

**LIST OF ABBRIEVATIONS**

|  |  |
| --- | --- |
| CNN | CONVOLUTIONAL NEURAL NETWORKS |
| SGD | STOCHASTIC GRADIENT DESCENT |
| UI | USER INTERFACE |

**ABSTRACT**

Artificial Intelligence is revolutionizing various domains, including agriculture, where it helps address key challenges. Among these, pest control is critical for ensuring sustainable farming. Traditional methods often fall short, as they fail to detect pests in time, leading to crop damage, reduced yields, and environmental degradation. The main focus of this study is to design an intelligent pest management system which uses artificial intelligence. The system uses image classification methods like Convolutional Neural Networks (CNNs) which use pretrained network models like ResNet to determine whether an organism is a pest or not. To train the CNN models, optimization methods such as SGD (Stochastic Gradient Descent) are used so that the detection rates are maximized. The system offers practical advice on pest management based on what it has analyzed so that the farmers can take appropriate actions. This solution is implemented using only software as a web application that can be developed using Gradio or Flask to avoid the use of extra gadgets. The model is trained with Kaggle’s dataset of pest images for its variety and relevance for pest identification. Accessibility to the application for farmers is made easy through hosting on Heroku or Railway.app. The system provides real time data on the identification of pests and the ways to control them, which improves the health of crops, increases agricultural production and reduces damage to the environment. This research shows the application of artificial intelligence to solve one of the major problems in agriculture and provides guidance towards the direction where agriculture can be more intelligent and sustainable.

**CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Chapter**  **No.** | **Title** | **Page No.** |
|  | List of Tables | i |
|  | List of Figures | i |
|  | List of Abbreviations | i |
|  | Abstract | ii |
| **1** | **Introduction** |  |
|  | 1.1 Introduction of the Project | 2 |
|  | 1.2 Existing System | 2 |
|  | 1.3 Problems of the Existing Systems | 3 |
|  | 1.4 Proposed System | 3 |
|  | 1.5 Benefits of the Proposed System | 4 |
| **2** | **Literature survey** |  |
|  | 2.1 Requirements Analysis | 7 |
|  | 2.1.1 Functional Requirements Analysis | 7 |
|  | 2.1.2 User Requirements | 8 |
|  | 2.1.3 Non-Functional Requirements | 8 |
|  | 2.1.4 System Requirements | 9 |
|  | 2.2 Modules Description | 9 |
|  | 2.3 Feasibility Study | 10 |
|  | 2.3.1 Technical Feasibility | 10 |
|  | 2.3.2 Operational Feasibility | 10 |
|  | 2.3.3 Behavioral Feasibility | 10 |
|  | 2.4 Process Model used | 10 |
|  | 2.5 Hardware and Software Requirements | 11 |
|  | 2.6 SRS Specification | 12 |
| **3** | **Existing work with references** |  |
|  | 3.1 Manual Inspection in Agriculture | 13 |
|  | 3.2 Inaccuracy and Delayed Detection | 13 |
|  | 3.3 Blanket Pesticides Usage and Its Impact | 13 |
|  | 3.4 Introduction of Smart Farming Techniques | 14 |
|  | 3.5 Lack of Affordable Solutions | 14 |
|  | 3.6 Environmental and Economic Drawbacks | 14 |

|  |  |  |
| --- | --- | --- |
|  | 3.7 Summary of Limitations | 15 |
| **4** | **Proposed work** |  |
|  | 4.1 Design concepts | 16 |
|  | 4.2 Design Constraints | 16 |
|  | 4.3 Conceptual Design | 17 |
|  | 4.4 Logical Design | 19 |
|  | 4.5 Architectural Design | 20 |
|  | 4.6 Algorithm Design | 22 |
|  | 4.6.1 Use Case Diagram | 25 |
|  | 4.6.2 Class Diagram | 26 |
|  | 4.6.3 Sequence Diagram | 27 |
|  | 4.6.4 Process Flow Diagram | 28 |
|  | 4.7 Database Design | 29 |
|  | 4.8 Module Design Specifications | 32 |
|  | 4.9 Sample Coding | 35 |
|  | 4.10 Output Screens | 38 |
|  | 4.11 Screen Reports | 39 |
| **5** | **Testing and Result Analysis** |  |
|  | 5.1 Introduction to Testing | 42 |
|  | 5.2 Types of Testing | 42 |
|  | 5.3 Test cases and Test Reports | 44 |
|  | 5.4 Implementation Process | 46 |
|  | 5.5 Implementation Steps | 48 |
|  | 5.6 Implementation procedure | 52 |
|  | 5.7 User Manual | 54 |
| **6** | **Conclusions and Scope for Future work** |  |
|  | 6.1 Conclusion | 56 |
|  | 6.2 Future Enhancements | 57 |
| **7** | **Bibliography** |  |
|  | 7.1 Books Referred | 65 |
|  | 7.2 Websites visited | 66 |
| **8** | **Appendix-I** Conference Certificates of our Team | 68 |
|  | **Appendix-II** Complete Journal | 72 |
|  | **Appendix-III** Internship Certificates of our Team | 78 |

**CHAPTER-1 INTRODUCTION**

# AI-Powered Smart Farming System for Sustainable Water Management and Pest Control

## 1.1 PROJECT ABSTRACT

Artificial Intelligence (AI) is transforming agriculture by addressing challenges such as pest control. Traditional methods often lead to delayed pest detection, resulting in significant crop damage and reduced productivity. This project develops an AI-powered system that utilizes **CNN-based image classification** to detect pests and recommend appropriate solutions. The system is designed as a **web platform** built using **Flask**, which allows farmers and agricultural researchers to upload images and receive real-time pest classification results. The model is trained using **deep learning techniques**, ensuring high accuracy in pest identification. The deployment of this system on **cloud-based platforms like Heroku or Railway.app** enables remote access, making it easier for farmers to make data-driven decisions for pest control. This promotes sustainable and efficient farming by enabling **early pest detection and proactive intervention**, reducing the excessive use of pesticides while ensuring healthy crop yields.

## INTRODUCTION OF THE PROJECT

Agriculture plays a crucial role in ensuring food security and economic stability. However, pest infestations pose a serious threat to agricultural productivity, leading to financial losses for farmers. The **AI-powered Pest Detection System** aims to tackle this issue by implementing **Deep Learning (DL) techniques**, particularly **Convolutional Neural Networks (CNNs)**, to detect and classify pests in images captured from crops. By training on large datasets of pest images, the system achieves accurate pest classification, helping farmers and agricultural experts take **preventive measures before an infestation spreads**.

The traditional methods of pest control rely on **visual inspection** and **manual identification**, which are often time-consuming and inaccurate. The proposed system offers a **fully automated, AI-driven solution** that reduces the burden on farmers while enhancing detection accuracy. A **user-friendly web interface** is developed to make the system accessible to farmers with minimal technical knowledge, allowing them to upload images of pests and receive real-time classification results along with pesticide recommendations. The system is **scalable, adaptable, and applicable across various agricultural settings**, ensuring its effectiveness in small-scale and commercial farming.

## 1.2 EXISTING SYSTEM

The current methods for pest detection primarily depend on manual inspection, which requires farmers to visually monitor crops for signs of pest infestations. This approach is not only time- consuming but also highly inaccurate, as early-stage infestations are often difficult to detect with the naked eye. Moreover, reliance on traditional pest control techniques, such as blanket pesticide spraying, leads to excessive chemical usage, which can negatively impact the environment and human health.

In addition, some farmers use IoT-based smart farming tools that integrate sensors and drones to monitor crop conditions. However, these solutions are expensive and require significant technical expertise, making them unsuitable for small and medium-scale farmers. The lack of accessible and affordable automated pest detection systems results in late interventions, causing significant crop losses.

## 1.2 PROBLEMS OF EXISTING SYSTEMS

The current manual and IoT-based solutions present several challenges:

* + - **Delayed detection of pests**, leading to severe crop damage and financial losses.
    - **High costs associated with smart farming tools**, making them inaccessible to small-scale farmers.
    - **Inaccuracy and inefficiency** in pest monitoring, increasing reliance on pesticide spraying.
    - **Limited automation** in pest classification, requiring manual verification for accurate identification.

## 1.3 PROPOSED SYSTEM

The proposed system introduces an AI-driven approach to pest detection by integrating **machine learning algorithms and computer vision techniques**. This system is designed to accurately **classify pest species from images** and provide real-time recommendations for appropriate pesticide usage. The deep learning model is trained using a **large dataset of pest images**, allowing it to detect infestations at an early stage, there by preventing crop damage.

The web-based platform is developed using **Flask**, making it easy to use and accessible.

from any device with an internet connection. The system is deployed on **cloud platforms like Heroku or Railway.app**, ensuring scalability and remote access. By leveraging **Convolutional Neural Networks (CNNs)**, the model achieves high accuracy in **identifying different pest species**, providing farmers with actionable insights.

## 1.4 BENEFITS OF THE PROPOSED SYSTEM

The AI-powered Pest Detection System offers several advantages over traditional pest control methods. One of the primary benefits is early pest detection, which allows farmers to take timely action and prevent crop loss. The system is also cost-effective, eliminating the need for expensive IoT- based monitoring devices. Farmers can simply upload pest images through a user-friendly web interface and receive accurate classification results instantly.

Additionally, the cloud-based deployment ensures that the platform is accessible to farmers and researchers anywhere, enabling them to monitor crops remotely. The system promotes sustainable agricultural practices by reducing excessive pesticide use and encouraging targeted pest control. By automating pest detection, this project enhances crop protection strategies, leading to improved agricultural productivity.

## APPLICATIONS

This AI-powered system can be widely applied in various agricultural settings:

* + - Pest Control and Crop Protection: Helps farmers detect pests early and implement targeted control measures.
    - Precision Agriculture: Enhances smart farming by integrating AI-based pest monitoring.
    - Small-Scale Farming :Provides affordable pest detection solutions for small-scale farmers.
    - Agricultural Research: Assists scientists in analyzing pest behaviors and developing new pest management strategies.
    - Commercial Farming: Supports large-scale pest monitoring for agri businesses and commercial

**CHAPTER -2 LITERATURE SURVEY**

##### Jamalbek Tussupov.et.al(2024)

For this statement it was necessary to apply the scientific methods of systemic analysis and simulation modeling. The essence of the work is to enhance and optimize the output of agriculture through the timely and precise identification and classification of disorders and their causative agents in the agricultural area. These results can be adopted as both scientific or practical guidance for agricultural businesses and institutions, as well as for the design and implementation of novel approaches and software solutions for automation systems in agriculture.

##### Ahmad Ali Alzubi.et.al(2023)

This paper assesses the R&D status in SSA, focuses on its current status, and suggest an IoT and AI architecture framework as a starting point for SSA. This study first assesses IoT architecture components that could support Smart Sustainable Agriculture (SSA) platforms.

**Ruchi Rani.et.al(2023)** This is a structured multi cascade survey and economic analysis of the work done in the AI domain related to plant disease detection in agriculture. This article deliberates on the and their corresponding solutions in AI. It evaluates multiple AI applications in agriculture and the possibilities. Developments in AI plant disease detection such as Identification Model Improvement (IMI), Few Shot Learning (FSL), Generative Adversarial Networks (GANs), and Self Supervised Learning (SSL) are also reported in this work. This article also discusses some issues associated with applying AI to plant disease detection.

**Yang Liu.et.al(2022)** In this article, the accuracy of recognition achieved by the model using the AI Challenger 2018 Plant Disease Recognition dataset was calculated.

**Sameer Qazi.et.al(2022)** This research paper covers:

Tutorial on the existing IoT technologies and AI techniques and their innovations in smart agriculture systems; (b) The two technologies and any challenges with relation to their wider

deployment are critically reviewed;

**Syeda Iqra Hassan.et.al(2021)** Although this research paper attempts to observe the work of several other researches to give a general account of trends in smart agriculture, it is also provides the work flow and the benefits of smart agriculture system in figure 15 using the hypothesized IoT technologies researched in other different research papers.

**Showkat Ahmad Bhat.et.al (2021)** The purpose of this article is to understand the developments in Big Data in smart agriculture as well as the social and economic problems that need to be focused on. This article describes methods of device creation, technology, and devices and software tools accessibility.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No** | **Year** | **Author’s** | **Article Title** | **Key Findings** |
| 1. | 2024 | Ravi Kumar Munaganuri.et.al | PAMICRM: Improving  Precision Agriculture Through Multimodal Image Analysis for Crop Water Requirement Estimation Using Multidomain Remote Sensing Data Samples | * The model improves crop water estimation and irrigation scheduling. * It works well for different crops and environments. |
| 2. | 2024 | Jamalbek Tussupov.et.al | Analysis of Formal Concepts for Verification of Pests and Diseases of Crops Using Machine Learning Methods | * With the aid of machine learning techniques like XGBoost, CNNs, and logisticregression, detecting pests and crop diseases is now easier. * satellite imagery help reduce pest control costs. |
| 3. | 2023 | Ahmad Ali Alzubi.et.al | Artificial Intelligence and Internet of Things for Sustainable Farming and Smart Agriculture | * AI and Iot are being used to improve farming and agriculture. * Challenges include managing data and device compatibility. |
| 4. | 2023 | Ruchi Rani.et.al | Role of Artificial Intelligence in Agriculturer: An Analysis and Advancements With Focus on Plant Diseases | * AI techniques like   Machine learning help detect plant diseases.   * AI improve disease detection andprediction in agriculture. |
| 5. | 2022 | YangLiu.et.al | Crop Disease Recognition Based on Modified Light-Weight CNN with Attention Mechanism | * Using attention mechanisms in a streamlined CNN enhances the accuracy of crop disease identification. * Achieved 91.4% accuracy, outperforming Xception's efficiency. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 6. | 2022 | Sameer Qazi.et.al | IoT-Equipped and AI-Enabled Next Generation Smart Agriculture: A Critical Review, Current Challenges and Future Trends | * IoT and AI boost farming through crop monitoring and resource saving.. * Intelligent farming has a promising future despite challenges. |
| 7. | 2021 | Syeda IqraHassan.et.al | A Systematic Review on Monitoring and Advanced Control Strategies in Smart Agriculture | * Smart Farming technologies (IOT, AI, imaging) can solve problems like diseases and water management. * Automation improves crop yield and soil fertility. |
| 8. | 2021 | Showkat AhmadBhat.et.al | Big Data and AI Revolution in Precision Agriculture: Survey and Challenges | * Implementation of AI and Bi Data in farming improve productivity an sustainability. * Data analytics enhances decision-making in farming. |
| 9. | 2020 | YongAi.et.al | Research on Recognition Model of Crop Diseases and Insect Pests Based on Deep Learning in Harsh Environments | * Inception-ResNet-v2 achieved 86.1% accuracy in recognition of crop diseases and pests. * WeChat app supports real-time crop disease detection. |
| 10. | 2020 | Ching-Ju Chen.et.al | An AIoT Based Smart Agricultural System For Pests Detection | * Automatic recognition of cro disease and pest is done usin deep learning mod (Inception- ResNet-v2). * WeChat app supports real-time crop disease detection. |

**TABLE 1.1 : LITERATURE REVIEW**

## REQUIREMENT ANALYSIS

* + 1. **Functional requirement analysis:**

Functional requirements define the core operations and behaviors that the system must execute. The AI-powered Pest Detection System includes the following key functionalities:

* + - * **Image Upload & Preprocessing:** Users should be able to upload crop images containing pests. The system should preprocess these images by performing resizing, normalization, and contrast adjustments to enhance clarity and improve classification accuracy.
      * **Pesticide Recommendation System:** Based on the classified pest species, the system should suggest suitable pesticide treatments by referring to a predefined pest-pesticide database. The recommendation should be scientifically backed and optimized for minimal environmental impact.
      * **User-Friendly Web Interface:** The system must feature a simple and intuitive web-based interface, allowing users to upload images, view detection results, and access pesticide recommendations without needing advanced technical knowledge.
      * **Performance Monitoring & Model Updates:** To ensure long-term reliability, the system should be capable of self-improvement by allowing for periodic model updates and retraining using newly collected datasets, thereby enhancing detection accuracy over time.
    1. **User Requirements:**

These are the needs of the farmers using the system:

* + - * Ability to upload crop images for pest detection.
      * Easy-to-use platform accessible from any device.
      * Minimal technical knowledge required to operate the system.
    1. **Non Functional Requirements:**

Non-functional requirements define system attributes such as reliability, performance, and usability. The AI-powered Pest Detection System adheres to the following:

* + - * **High Accuracy & Performance:** The system should ensure fast image processing and a pest detection accuracy rate above 90%, minimizing errors in classification.
      * **Reliability & Robustness:** The model should be able to handle diverse environmental conditions such as varying lighting, image quality, and pest visibility without significantly affecting accuracy.
      * **Scalability:** The system should be capable of handling multiple user requests simultaneously and support large-scale dataset processing as it expands.
      * **Security:** Image data uploaded by users should be securely stored using encrypted databases, preventing unauthorized access or misuse.
      * **Accessibility & Usability**: The platform should be accessible across multiple devices (mobile, desktop, tablet) and designed with a clear, easy-to-navigate interface.
    1. **System Requirements:**

The AI-powered Pest Detection System requires a combination of computing resources, software frameworks, and cloud services to ensure efficient performance and scalability. Below are the detailed system requirements:

internet connection for accessing the web-based system.

* + - * Cloud-based deployment on Heroku or Railway.app to enable remote accessibility and scalability for multiple users.
      * Minimum 100GB cloud storage for maintaining pest image datasets and retraining the classification model periodically.

### Modules Description:

The system is structured into distinct **functional modules**, each responsible for a specific task:

1. **Image Upload & Preprocessing Module:** Handles the upload of pest-infected crop images. It processes images by applying **resizing, noise reduction, and color corrections** to enhance classification accuracy.
2. **Pest Classification Module:** Implements a **CNN-based image classification model** trained on labeled datasets of pests. It processes the input images and predicts the **most probable pest species** affecting the crop.
3. **Pesticide Recommendation Module:** Once a pest is classified, this module cross- references the pest name with a **predefined database of pest control solutions** to suggest

suitable pesticide treatments.

1. **User Interface Module:** Provides a **user-friendly web interface.**

### Feasibility Study:

To ensure successful implementation, a feasibility study was conducted.

* + 1. Technical Feasability:

The system is built using **CNN-based deep learning models**, which are well-established in image classification tasks.

* + 1. Operational Feasability:

The project is **highly practical** and does not require specialized knowledge for operation.

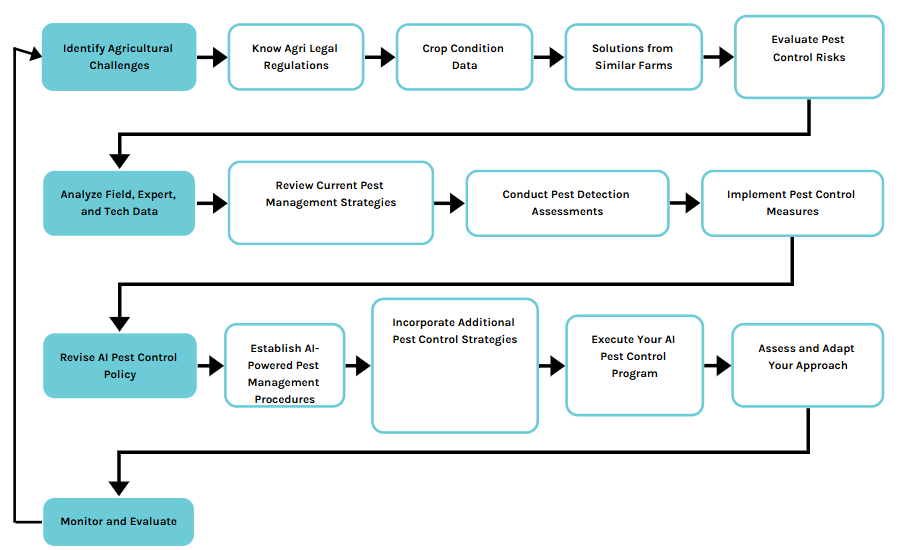
* + 1. Behavioral Feasablity:

The AI-powered system provides **trustworthy and scientifically backed insights**.

### Process Model Used: Agile Methodology

Follows Agile methodology for flexibility and continuous improvement:

* **Planning:** Identify agricultural challenges and gather data.
* **Development:** Analyze data and implement AI-based pest detection.
* **Deployment:** Apply control measures and execute the AI system.
* **Iteration:** Monitor, evaluate, and refine the system continuously.



### Hardware and Software Requirements

##### Hardware Requirements

The system requires minimal hardware resources, making it accessible to users with basic computing devices. The essential hardware requirements include:

* **Basic System:** A laptop or desktop with a minimum of **4GB RAM**, an **Intel i3 processor or equivalent**, and an internet connection for accessing the web application.
* **Deployment Server:** Cloud platforms such as **Heroku or Railway.app** to host the web- based pest detection system, ensuring remote accessibility and scalability.
* **Storage Requirements:** At least **100GB of cloud storage** to maintain a growing dataset of pest images for periodic retraining of the model.

##### Software Requirements

The software requirements include tools and frameworks essential for implementing the AI-based pest detection system. These include:

* **Programming Language:** Python for developing the AI model and backend logic.
* **Frameworks:** TensorFlow/Keras for building the **CNN-based deep learning model**, and Flask for creating the **web-based application**.
* **Libraries & Tools:** OpenCV for image preprocessing, NumPy and Pandas for data handling, and Matplotlib/Seaborn for data visualization.
* **Web Technologies:** HTML, CSS, and JavaScript for developing a **user-friendly frontend interface**.
* **Cloud Platforms:** Heroku, Railway.app for deployment, ensuring **scalability and remote access**.
* **Development Tools:** Jupyter Notebook and VS Code for writing and testing code efficiently.

### SRS Specifications

The Software Requirements Specification (SRS) outlines the functional and technical specifications necessary for the **AI-powered Pest Detection System**. The main objectives include:

* **Purpose:** The system is designed to provide an **AI-powered solution for pest detection** by analyzing images of crops, classifying detected pests, and recommending suitable pest control measures.
* **Scope:** The system supports **sustainable agricultural practices** by enabling early pest detection, reducing excessive pesticide use, and providing accurate pest control recommendations.
* **Inputs:** Users upload images of crops affected by pests, which are then processed by the deep learning model for classification.
* **Outputs:** The system returns **pest classification results, pesticide recommendations, and a harmful/non-harmful assessment** for the detected pest.

**CHAPTER -3**

**EXISTING WORK WITH REFERENCES**

### Manual Inspection in Agriculture

In many parts of the world, including India, pest detection in agriculture still depends heavily on traditional practices, especially manual inspection. Farmers routinely walk through their fields to visually examine crops for signs of pest damage, such as leaf discoloration, bite marks, wilting, or the presence of insects. While this method has been used for generations, it has become increasingly ineffective in the modern era of large- scale farming. Manual inspection is not only time-consuming and physically demanding, but also limited by the observer’s knowledge and experience. Most importantly, early- stage pest infestations often show minimal visible symptoms, making them nearly impossible to detect through naked-eye observation.

### Inaccuracy and Delayed Detection

A major disadvantage of manual methods is the high risk of delayed or inaccurate detection. Pests often reproduce rapidly and spread across crops within days. By the time visible signs appear and are recognized by the farmer, the infestation may have already reached a critical stage, resulting in significant yield loss. This delay in response leads to increased costs for pest control and decreases overall productivity. Moreover, not all pests exhibit visible symptoms in the early stages, which further complicates early intervention efforts.

### Blanket Pesticide Usage and Its Impact

To manage pest attacks, many farmers opt for blanket pesticide spraying across their entire fields without accurately identifying whether pests are present or determining which species is causing the problem. This reactive and generalized approach results in the overuse of chemicals, leading to several negative consequences. Excessive pesticide use contaminates the soil and water sources, affects biodiversity by killing beneficial insects such as pollinators, and can cause serious health problems for both farmers and consumers.

It also leads to chemical resistance in pest populations, making future control efforts more difficult and less effective.

### Introduction of Smart Farming Techniques

With the emergence of smart agriculture, a few progressive farms have begun using technologies such as IoT (Internet of Things), drones, remote sensing, and smart sensors to monitor crop health and detect pest presence. These systems can provide real-time data on temperature, humidity, soil moisture, and leaf images, which can then be analyzed to infer pest activity. While such systems are technologically advanced, their widespread adoption remains limited. The major barriers include high installation and maintenance costs, lack of internet connectivity in rural areas, and the need for technical knowledge to operate and interpret the data. As a result, such tools are largely restricted to large-scale commercial farms and research institutions.

### Lack of Affordable Automated Solutions

One of the critical gaps in the existing systems is the unavailability of low-cost, user- friendly, and scalable solutions for automated pest detection. Most small and marginal farmers do not have access to systems that can assist them in identifying pests accurately using affordable technologies. The current market lacks integrated tools that use image processing and artificial intelligence to detect and classify pests in real-time. This technological void forces farmers to rely on guesswork or costly expert consultations, both of which are not feasible in the long run.

### Environmental and Economic Drawbacks

The inefficiencies in pest detection and control also contribute to broader environmental and economic issues. Soil degradation, water pollution, and health problems due to pesticide overuse have become major concerns in many agricultural regions. Furthermore, crop loss due to late pest identification results in reduced income for farmers, affecting their livelihood and food security. Without early detection, interventions are often reactive and less effective, leading to a cycle of increased cost and reduced yield.

### Summary of Limitations

In summary, the existing pest detection systems suffer from multiple limitations including reliance on manual observation, lack of early detection, over-dependence on chemical solutions, high costs of smart technologies, and minimal availability of AI-powered tools. These limitations emphasize the urgent need for an intelligent, automated, and accessible pest detection system that can serve the needs of all categories of farmers, especially those operating on a smaller scale. This is precisely where the proposed system based on machine learning and image processing aims to make a significant impact by offering a low-cost, scalable, and accurate alternative to traditional methods.

**CHAPTER-4 PROPOSEDWORK**

### Design Concepts: AI-Powered Pest Control System

The proposed system for AI-powered pest control operates on a multi-layered architecture designed for efficient and sustainable pest management in agriculture. At its core, the system processes images of crops uploaded by farmers through a web-based platform. These images undergo preprocessing to standardize their dimensions and format, ensuring compatibility with the convolutional neural networks (CNNs) used for pest detection and classification. Models like ResNet or MobileNet are utilized due to their high accuracy and efficiency in handling image data. The training of these models relies on optimization techniques such as Stochastic Gradient Descent (SGD), ensuring precise and reliable predictions.

The data flow within the system begins with farmers uploading pest-affected crop images, which are then processed by the trained CNN models. These models classify the detected pests into harmful or harmless categories, providing actionable recommendations tailored to each situation. The output is displayed on a user-friendly web interface, offering clear insights into pest classification and eco-friendly control measures.

To make the system accessible and practical for farmers, it is developed as a lightweight, web- based platform using tools such as Gradio or Flask. This design ensures that the platform is intuitive and easy to use, even for non-technical users. Farmers can access the system via mobile devices or desktop browsers, making it highly adaptable to varying levels of digital literacy. Moreover, the platform is hosted on scalable cloud services such as Heroku or Railway.app, ensuring its availability even in remote areas with limited resources.

The system emphasizes sustainability by reducing the need for indiscriminate pesticide application, thereby promoting ecological balance. Its scalability is another key feature, as it can be adapted to include localized datasets to improve the accuracy of pest detection in diverse agricultural contexts..

### Design Constraints

##### Data Quality and Availability:

* + - * The accuracy of the system depends on the quality and diversity of the dataset

(e.g., Kaggle pest image dataset).

* + - * Limited availability of region-specific pest data may reduce the system's effectiveness in localized settings.

##### Computational Limitations:

* + - * The platform must be lightweight and capable of running efficiently on low- powered devices.
      * Complex models are avoided to ensure compatibility with limited internet connectivity in rural areas.

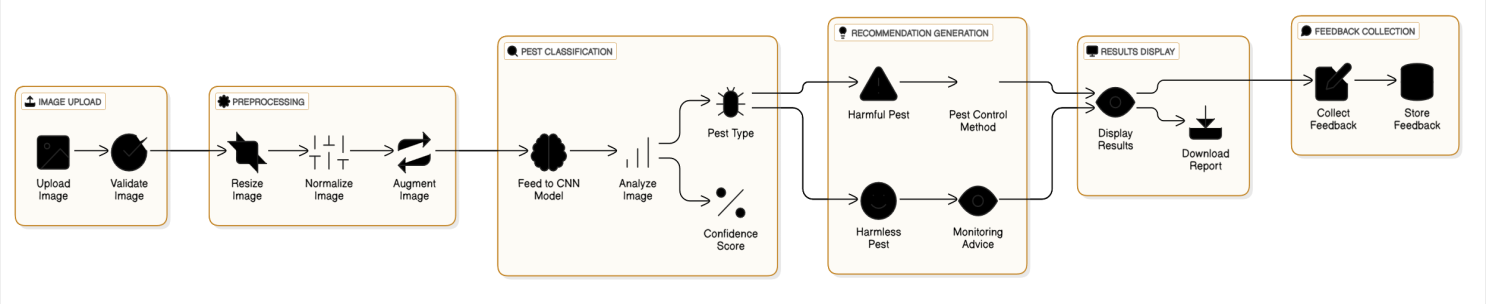
##### Real-Time Performance:

* + - * The system must process images and provide recommendations quickly to enable timely pest control actions.
      * Reducing latency in data processing and optimizing server response times are critical.

##### Usability and Accessibility:

* + - * The interface must be intuitive and easy to use for farmers with minimal technical knowledge.
      * Support for regional languages and simple workflows is essential to ensure widespread adoption.

### Conceptual Design

****

**FIG 4.1 : CONCEPTUAL DESIGN**

##### Image Upload:

* + - * The user uploads an image of a pest-affected crop via the web interface.
      * The image is validated (format, size) and stored temporarily for processing.

##### Preprocessing:

* + - * The system resizes and normalizes the image.
      * Augmentation techniques (e.g., rotation, flipping) may be applied to ensure consistent model input.

##### Pest Classification:

* + - * The preprocessed image is fed into the CNN model.
      * The model analyzes the image and outputs:
      * The type of pest (harmful or harmless).
      * A confidence score indicating the reliability of the classification.

##### Recommendation Generation:

* + - * If the pest is classified as harmful, the system maps it to an appropriate pest control method (e.g., neem oil spray).
      * If harmless, the system advises regular monitoring without unnecessary pesticide application.

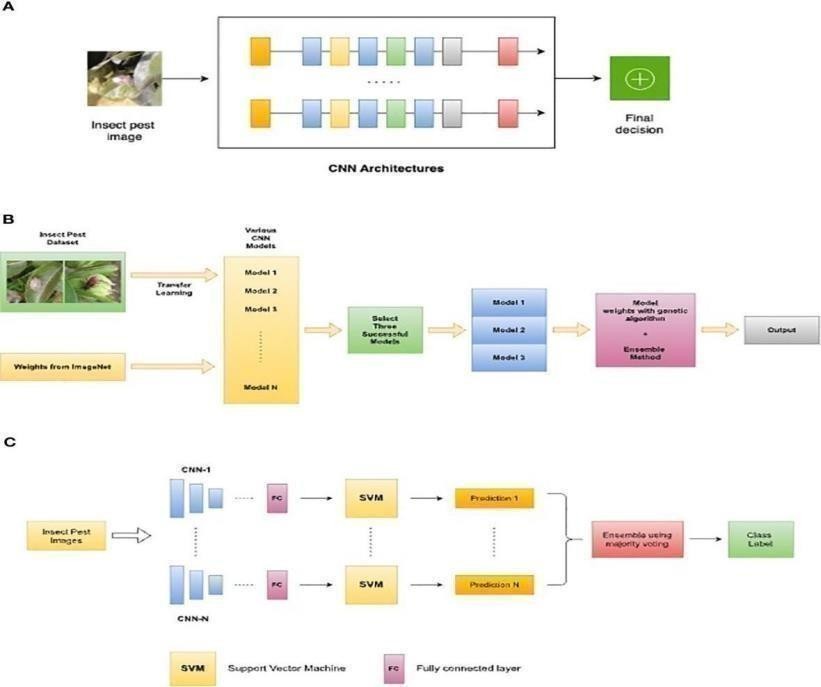
##### Results Display:

* + - * The classification result, confidence score, and pest control recommendation are displayed on the user interface.
      * Users can also download a report of the results.

##### Feedback Collection:

* + - * Users can provide feedback on the recommendations, including ratings and comments.
      * Feedback is stored in the database for improving the system.

### Logical Design(Logical Tools/Logical Diagrams)

****

**FIG 4.2 : LOGICAL DESIGN**

* The **logical design** of the AI-Powered Pest Control System explains how different parts of the system interact and how data flows between them. It focuses on the relationships and processes in a simple and clear manner.

##### Logical Components

* 1. **User Layer**:
     + Farmers upload pest-affected crop images through a web interface and receive pest control recommendations.

##### Processing Layer:

* + - Handles the image processing, pest classification, and recommendation generation:
    - Prepares the uploaded image for analysis.
    - Passes the image to the AI model to identify whether the pest is harmful or harmless.
    - Generates pest control advice based on the classification.

##### Database Layer:

* + - Stores information such as user details, uploaded images, classification

results, recommendations, and feedback.

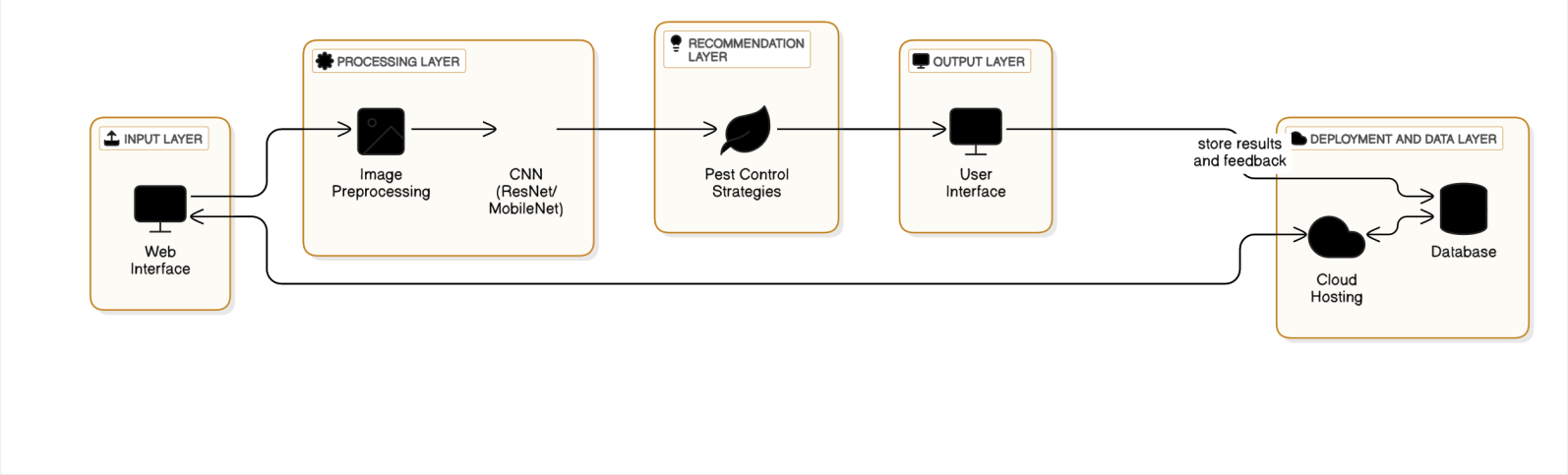
##### AI Model Layer:

* + - Uses a pre-trained AI model (e.g., ResNet or MobileNet) to analyze the images and classify pests.

##### Output Layer:

* + - Displays results(pest type, confidence score, and recommendations) to the user.
    - Collects user feedback about the recommendations.

## ARCHITECTURAL DESIGN

****

**FIG 4.3 : ARCHITECTURAL DESIGN**

##### High-Level Architecture

The system architecture is divided into five layers:

##### Input Layer:

* + This layer acts as the entry point for users (farmers) to upload images of pest- affected crops.
  + The user uploads images through a web-based interface, which validates and forwards them to the processing layer.

##### Processing Layer:

* + Images from the input layer are preprocessed to ensure compatibility with the model.
  + The preprocessed images are passed to a **Convolutional Neural Network**

##### Recommendation Layer:

* + Based on the pest classification results, actionable pest control strategies are generated.
  + The recommendations are tailored to minimize pesticide usage and prioritize eco-friendly methods.

##### Output Layer:

* + The classification results and recommendations are displayed to the user through a clean and user-friendly web interface.
  + Farmers receive visual and textual outputs, including the identified pest type, confidence score, and control measures.

##### Deployment and Data Layer:

* + The application is hosted on cloud platforms like Heroku or Railway.app, ensuring scalability and accessibility in remote areas.
  + A relational database (e.g., MySQL or PostgreSQL) stores user data, uploaded images, classification results, and feedback.

##### Component-Level Architecture

1. **Frontend**:
   * Developed using tools like Gradio or Flask for a responsive and intuitive user experience.
   * Handles image uploads, displays classification results, and collects user feedback.

##### Backend:

* + Implements the core functionality, including preprocessing, pest classification, and recommendation generation.
  + Integrates the trained CNN model to process images and return predictions.
  + Manages requests from the frontend and sends appropriate responses.

##### Database:

* + Stores user details, uploaded images, pest classification results, and feedback.
  + Ensures data integrity and efficient retrieval for analysis and future improvements.

##### Machine Learning Model:

* + A pre-trained CNN model (e.g., ResNet or MobileNet) is deployed in the backend for pest identification.
  + Optimized using Stochastic Gradient Descent (SGD) to minimize errors and improve accuracy.

##### Cloud Hosting:

* + Ensures that the platform is accessible anywhere with minimal latency.
  + Scales to support multiple users simultaneously.
  + Optimized using Stochastic Gradient Descent (SGD) to minimize errors and improve accuracy.

## ALGORITHMS DESIGN

##### Image Preprocessing Algorithm

* + **Purpose**: Prepare the uploaded pest images for analysis by the CNN model.

##### Steps:

* + - Normalize pixel values to arrange of[0, 1] by dividing each pixel value by255.
    - Perform data augmentation to improve model generalization (e.g., rotation, flipping, cropping).
    - Save the preprocessed image for input into the classification model.

##### CNN Model Training Algorithm

* + **Purpose**: Train a Convolutional Neural Network (CNN) model to classify pests.

##### Steps:

* + - Compile the model using an optimizer (e.g., Stochastic Gradient Descent) and loss function (e.g., cross-entropy loss).
    - Train the model on the training dataset, validating it with the validation set.
    - Save the trained model for inference.

def train\_cnn\_model(dataset):

model = ResNet50(weights='imagenet', input\_shape=(224,224, 3), include\_top=False) model = add\_custom\_layers(model) # Add custom layers for pest classification

##### Pest Classification Algorithm

* + **Purpose**: Use the trained model to classify pests as harmful or harmless.

##### Steps:

* + - Perform inference to predict the pest type.
    - Output the predicted pest type and confidence score.

defclassify\_pest(image,model):

preprocessed\_image

=preprocess\_image(image)predictions= model.predict(preprocessed\_image)

pest\_type = np.argmax(predictions) # Get class with highest probability confidence\_score = np.max(predictions) # Get the confidence score

##### Recommendation Generation Algorithm

**Purpose**: Generate actionable pest control strategies based on the classification

* + Map the pest type to its corresponding control methods using a predefined database.
  + If the pest is harmful, recommend eco-friendly pest control strategies.
  + Display the recommendations to the user.

defgenerate\_recommendations(pest\_type): pest\_control\_database = {

'harmful': 'Apply eco-friendly pesticide XYZ or use biological control agents.', 'harmless': 'No action needed. Monitor the pest naturally.'

##### Real-Time Deployment Algorithm

* + **Purpose**: Ensure the system is accessible and functional in real-time for farmers.

##### Steps:

* + - Integrate the trained model and recommendation logic into a web framework (e.g., Flask).
    - Set up endpoints for image upload, classification, and recommendation display.
    - Deploy the application on a cloud platform(e.g., Heroku, Railway.app).
    - Optimize server response times to handle multiple users simultaneously. @app.route('/upload', methods=['POST'])

def upload\_image():

image = request.files['image']

pest\_type,confidence\_score=classify\_pest(image, loaded\_model) recommendation

= generate\_recommendations(pest\_type)

##### Feedback Collection Algorithm

* + **Purpose**: Allow users to provide feedback on recommendations for continuous improvement.

##### Steps:

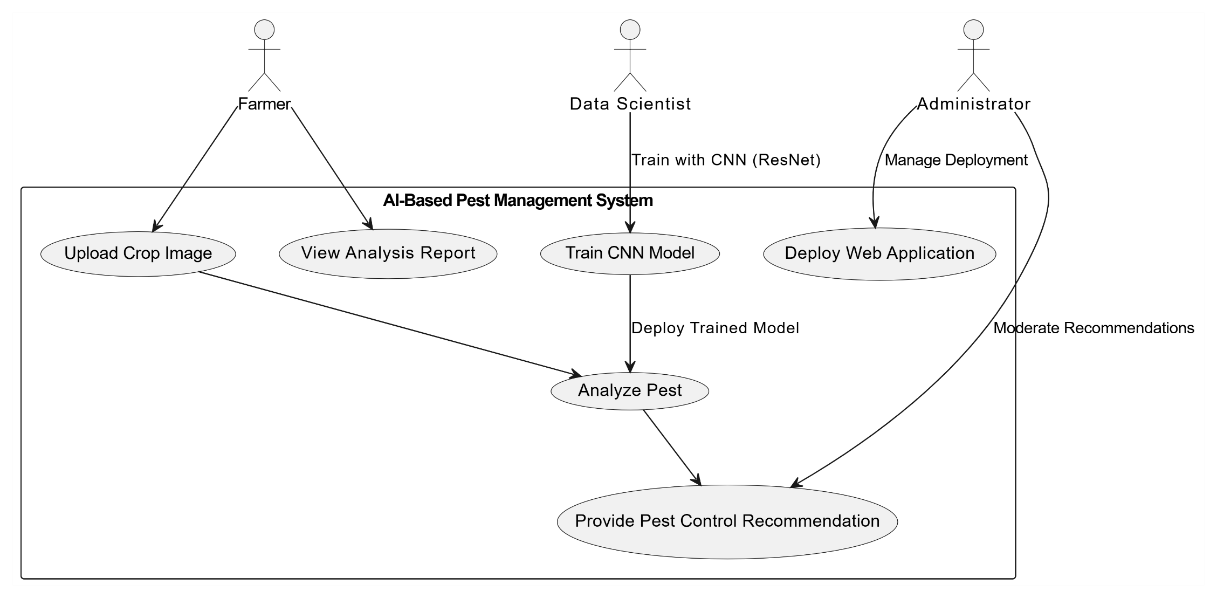
* + - Store the feedback in the database.
    - Periodically analyze feedback to improve the model and recommendations.

@app.route('/feedback',methods=['POST']) def collect\_feedback():

feedback = request.json

save\_to\_database(feedback) # Store feedback in the database return jsonify({'message': 'Thank you for your feedback!'})

**4.6.1 USE CASE DIAGRAM**



**Fig 4.4 : USE CASE DIAGRAM**

**Use Case Diagram**

1. **Data Scientist**:
   1. **Train CNN Model**: Uses ResNet to train the AI model for pest classification.
   2. **Deploy Web Application**: Hosts the system for farmers to use.
   3. **View Analysis Report**: Monitors system performance and pest data.
2. **Administrator**:
   1. **Manage Deployment**: Ensures the system runs smoothly.
   2. **Moderate Recommendations**: Reviews and updates pest control suggestions.
3. **Farmers**:
   1. **Analyze Pest**: Uploads pest images for classification.
   2. **Receive Recommendations**: Gets eco-friendly pest control advice.

**4.6.2 CLASS DIAGRAM**



**FIG 4.5 : CLASS DIAGRAM**

* **Purpose:** This diagram shows the structure of an AI-based pest control solution.
* **Main Parts:**

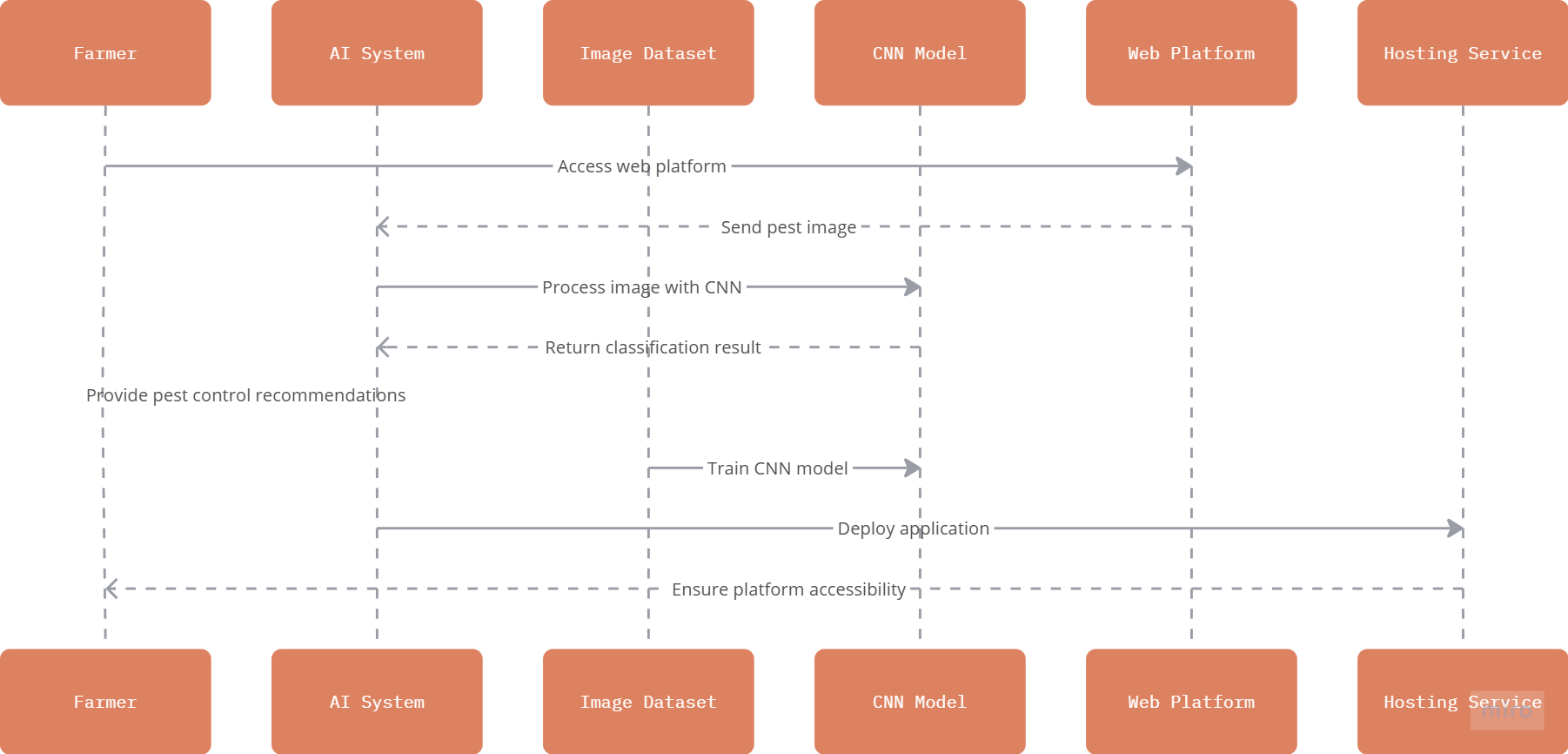
1. **WebApp:** Deploys the application on platforms like Railway and Heroku.
2. **CNN Model:** Handles pest image classification using datasets.

**Important Features:**

1. **ResNet:** An advanced version of CNN for better image classification.
2. **SGD:** Optimizes the AI model.
3. **Interfaces:** Gradio and Flask are used to interact with the AI solution.

**Data:** The system uses pest images from a dataset for training the AI model.

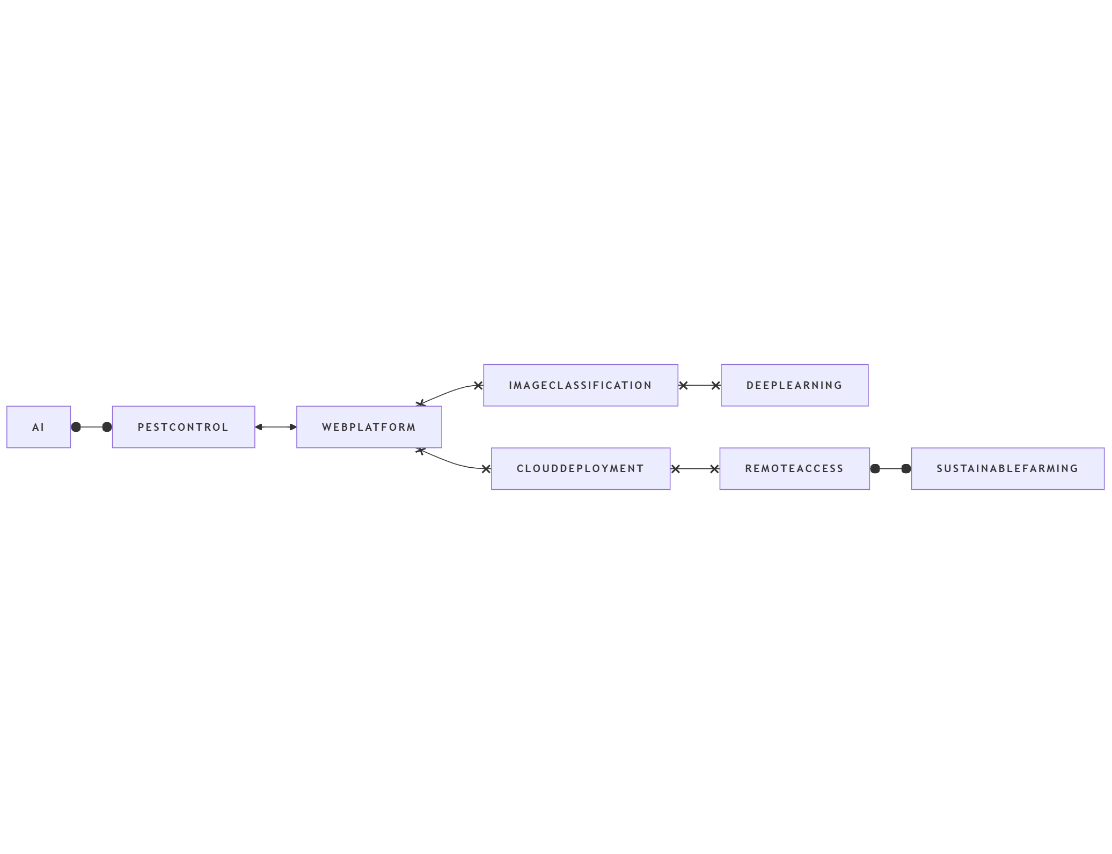
**4.6.3 SEQUENCE DIAGRAM**



**FIG 4.6 : SEQUENCE DIAGRAM**

* **Farmer interacts with the system** by uploading pest images through a web platform.
* **Web platform forwards the image** to the AI system for analysis.
* **AI System processes the image** using a trained CNN (Convolutional Neural Network) model.
* **CNN Model classifies the pest** based on the dataset it was trained on.
* **Classification result is returned** to the AI system.
* **AI system provides pest control recommendations** back to the farmer.
* **Web platform validates the uploaded image** (e.g., file format, size, resolution) before forwarding it to the AI system.
* **Image preprocessing is applied** before passing to the CNN model (e.g., resizing, normalization, color conversion).

**4.6.4 PROCESS FLOW DIAGRAM**



**FIG 4.7 : PROCESS FLOW DIAGRAM**

* + 1. **Artificial Intelligence (AI) Integration**
* Core technology driving the system.
* Enables intelligent decision-making and automation.
  + 1. **Pest Control**
* AI is used to monitor and control pests in agricultural fields.
* Enhances crop health and reduces manual effort.
  + 1. **Web Platform**
* Central interface for users to access the system.
* Integrates multiple services like image classification and cloud deployment.
* Ensures accessibility and ease of use for farmers.

**4. Image Classification**

* Detects and classifies pests using AI models.

## DATABASE DESIGN

The database design for the **AI-Powered Pest Control System** is structured to efficiently manage user data, pest images, classification results, and recommendations. The design ensures

scalability, integrity, and secure storage, supporting the system’s functionalities enhancements.

##### User Management:

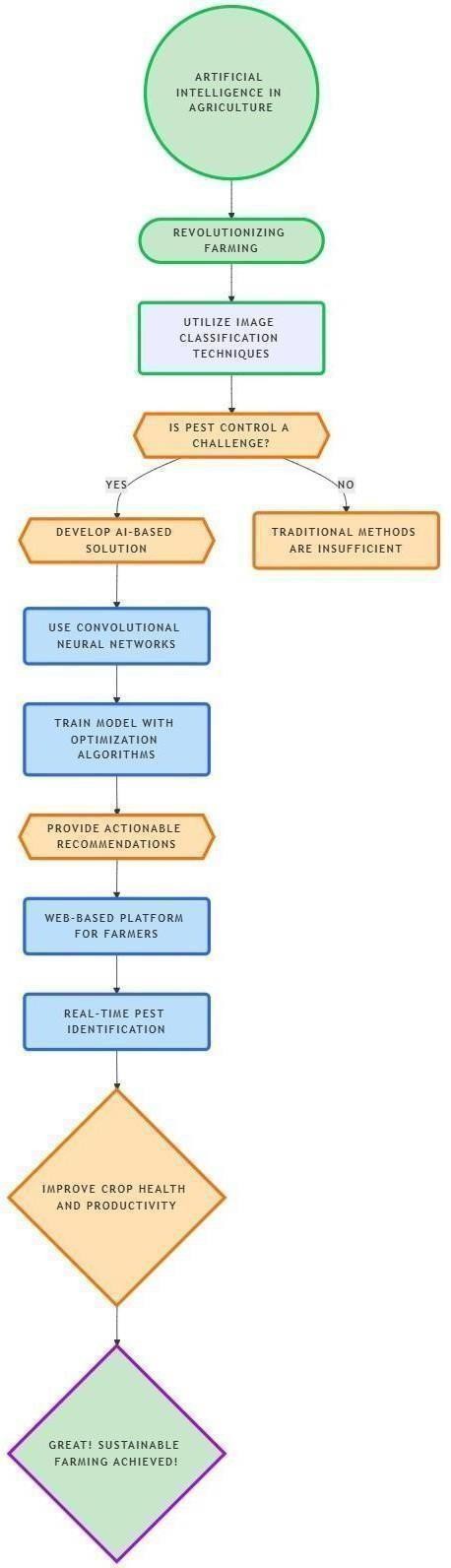
* Stores user details such as name, email, role (admin, farmer, researcher).
* Implements authentication mechanisms to ensure secure access.

##### Pest Image Storage:

* + Maintains a repository of pest images uploaded by users.
  + Uses metadata (e.g., timestamp, location, device used) for better tracking.
  + Integrates cloud storage solutions for scalability.

##### Pest Classification Data:

* + Stores classification results from the AI model.
  + Includes confidence scores and pest species detected.
  + Logs historical classification data for trend analysis.



**FIG 4.8 : DATABASE DESIGN**

##### Database Schema

* + - 1. **Users Table**
* **Purpose**: Stores information about farmers or users accessing the system.

##### Fields:

* + user\_id (Primary Key): Unique identifier for each user.
  + name: Name of the user.
  + email: Email address for login and communication.

##### Pest Images Table

* **Purpose**: Stores information about the pest-affected crop images uploaded by users.

##### Fields:

* user\_id (Foreign Key): Links the image to the user who uploaded it.
* image\_path: File path where the image is stored.

##### Pest Classifications Table

* **Purpose**: Stores the results of pest classification.

##### Fields:

* classification\_id(PrimaryKey): Unique identifier for each classification result.
* image\_id (Foreign Key): Links the classification to the corresponding image.
* pest\_type: Identified type of pest (e.g., harmful or harmless).
* confidence\_score: Confidence level of the classification (e.g., 95%).
* classification\_date: Timestamp of when the classification was performed.

##### Recommendations Table

* **Purpose**: Stores pest control recommendations generated by the system.
* recommendation.
* classification\_id (Foreign Key): Links the recommendation to the pest classification.
* control\_method: Suggested pest control method (e.g., organic pesticides, eco- friendly practices).
* notes: Additional information or advice for the farmer.

##### Feedback Table

* **Purpose**: Stores user feedback on the system’s recommendations.

##### Fields:

* feedback\_id (Primary Key):Unique identifier for each feedback entry.
* user\_id (Foreign Key): Links the feedback to the corresponding user.
* recommendation\_id (Foreign Key): Links the feedback to the specific recommendation.

##### Relationships Between Tables

* + **Users ↔ Pest Images**: One-to-many relationship, as one user can upload multiple images.
  + **Pest Images ↔ Pest Classifications**: One-to-one relationship, as each image corresponds to a single classification result.
  + **Pest Classifications ↔ Recommendations**: One-to-one relationship, as each classification generates a specific recommendation.

**Users ↔ Feedback**: One-to-many relationship, as one user can provide feedback for multiple recommendations.

### Module DesignSpecifications

The Module Design Specifications for the AI-Powered Pest Control System outline the functional breakdown of the system into well-defined modules.

##### Image Upload Module

* + **Purpose**: To allow users (farmers) to upload images of pest-affected crops.
  + **Input**: Crop images (JPEG, PNG formats).
  + **Functionalities:** Accepts images via a web interface. Validates the image format and size. Displays a preview of the uploaded image. Passes the image to the preprocessing module for further processing.
  + **Output**: A valid, preprocessed image ready for classification.

##### Preprocessing Module

* + **Purpose**: To prepare uploaded images for analysis by the CNN model.
  + **Input**: Uploaded crop images.
  + **Functionalities:** Resizes images to a fixed dimension (e.g., 224x224 pixels). Normalizes image pixel values. Augments data by applying
  + transformations (e.g., rotation, flipping) to improve model robustness.
  + **Output**: Preprocessed images suitable for the classification module.

##### Classification Module

* + **Purpose**: To identify and classify pests in the uploaded image.
  + **Input**: Preprocessed images.
  + **Functionalities:** Uses a pre-trained CNN model (e.g., ResNet or MobileNet) to classify pests as harmful or harmless. Computes confidence scores for the classification. Logs the results for future reference.
  + **Output**: Pest classification results (type of pest and confidence score).

##### User Interface Module

* + **Purpose**: To provide a user-friendly web interface for interacting with the system.
  + **Input**: User actions (e.g., image uploads, feedback submissions).

##### Feedback Module

**Purpose**: To collect and store user feedback on system recommendations.

**Input**: User-provided ratings and comments.

* **Functionalities:** Allows users to rate the recommendations (e.g., 1-5 stars). Records optional comments or suggestions for system improvement. Stores feedback in the database for analysis and system enhancement.
  + **Output**: Feedback data stored in the database.

##### Deployment Module

* **Purpose**: To host and maintain the system on a cloud platform for accessibility.
  + **Input**: Entire application stack (backend, model, database).
* **Functionalities:** Deploys the system on platforms like Heroku or Railway.app. Ensures scalability to handle multiple users simultaneously. Monitors system performance and uptime.
  + **Output**: A fully accessible web-based application.

## CODING & OUTPUT SCREENS

### Sample Coding

For the **AI-Powered Pest Control System**, the sample code showcases key functionality, including preprocessing, model inference, and integration with the web interface. Below is an example of the critical sections of code to demonstrate how the system works.

##### Image Preprocessing

This code handles resizing, normalization, and augmentation of pest images before feeding them into the model.

from PIL import Imageimport numpy as np import tensorflow as tf

def preprocess\_image(image\_path):

"""Preprocess the uploaded image for classification.""" image = Image.open(image\_path)

image = image.resize((224, 224)) # Resize to match the input size of the CNN model image\_array = np.array(image) / 255.0 # Normalize pixel values to [0, 1] image\_array = np.expand\_dims(image\_array, axis=0) # Add batch dimension return image\_array

##### Model Inference

This function loads the trained CNN model(e.g., ResNet) and classifies the input image.

defclassify\_pest(image\_array,model\_path='pest\_model.h5'): """Classify the pest in the given image."""

# Loadthe pre-trained model

model= tf.keras.models.load\_model(model\_path)

return pest\_class, confidence

##### Recommendation Generation

This function maps the classification result to actionable pest control recommendations.

def generate\_recommendation(pest\_class):

"""Generate pest control recommendations based on the classification.""" if pest\_class == 'harmful':

return "Use organic pesticides like neem oil. Ensure early application for effective control."

else:

return "No harmful pests detected. Monitor the crop regularly for early detection."

##### Flask Integration (Web Interface)

This code integrates the above functions into a Flask-based web application. from flask import Flask, request, render\_template

app = Flask( name )

@app.route('/') def home():

return render\_template('index.html')

@app.route('/upload',methods=['POST']) def upload():

#Save the uploaded image image\_file = request.files['image'] image\_path=f"uploads/{image\_file.filename}" image\_file.save(image\_path)

# Render the results

return render\_template('result.html',

pest\_class=pest\_class, confidence=f"{confidence:.2f}%", recommendation=recommendation)

if name == ' main ': app.run(debug=True)

##### Database Interaction

This code demonstrates storing user data and pest classifications in the database using sqlite3 import sqlite3

def save\_classification(user\_id, image\_path, pest\_class, confidence): """Save the classification result to the database."""

sqlite3.connect('pest\_control.db') cursor = conn.cursor()

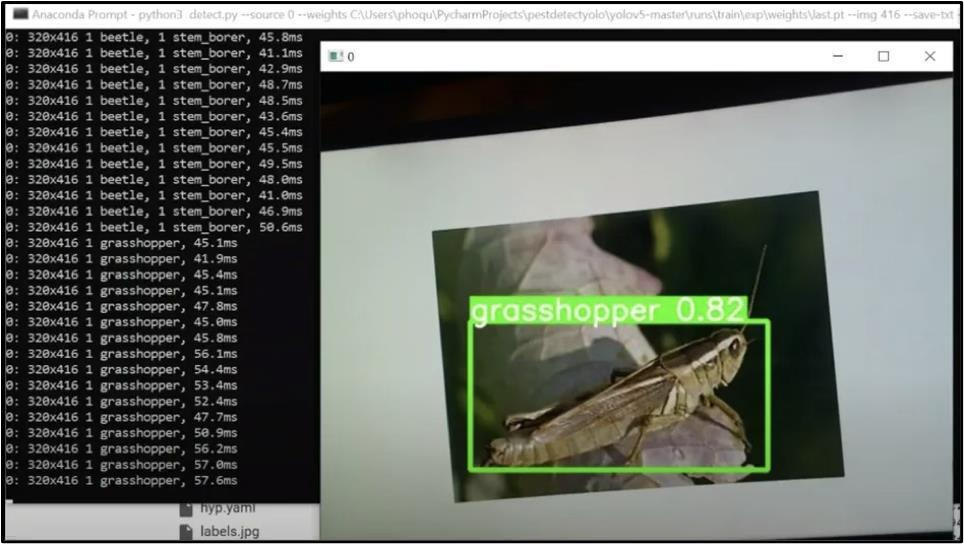
# Insert data into the pest\_classifications table cursor.execute("""

INSERTINTO pest\_classifications (image\_id, pest\_type, confidence\_score, classification\_date)

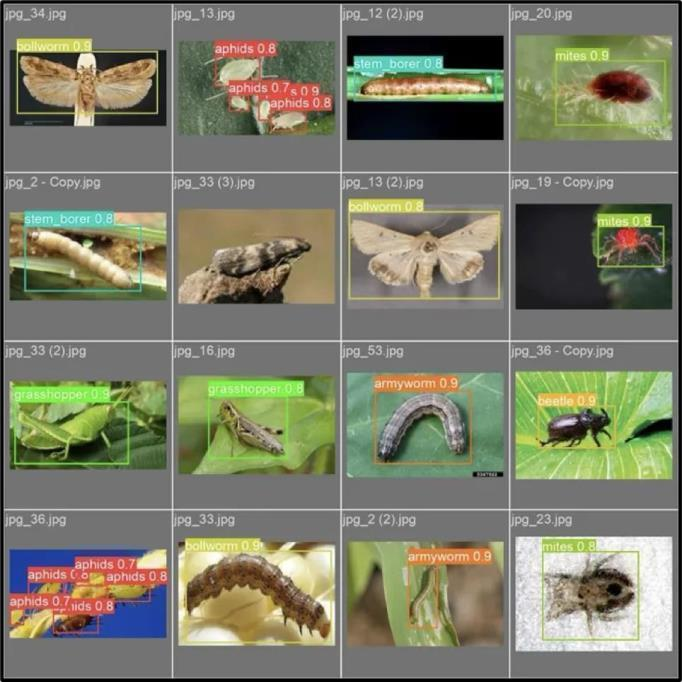
VALUES (?, ?, ?, datetime('now')) """, (image\_path, pest\_class, confidence)) conn.commit()

conn.close()

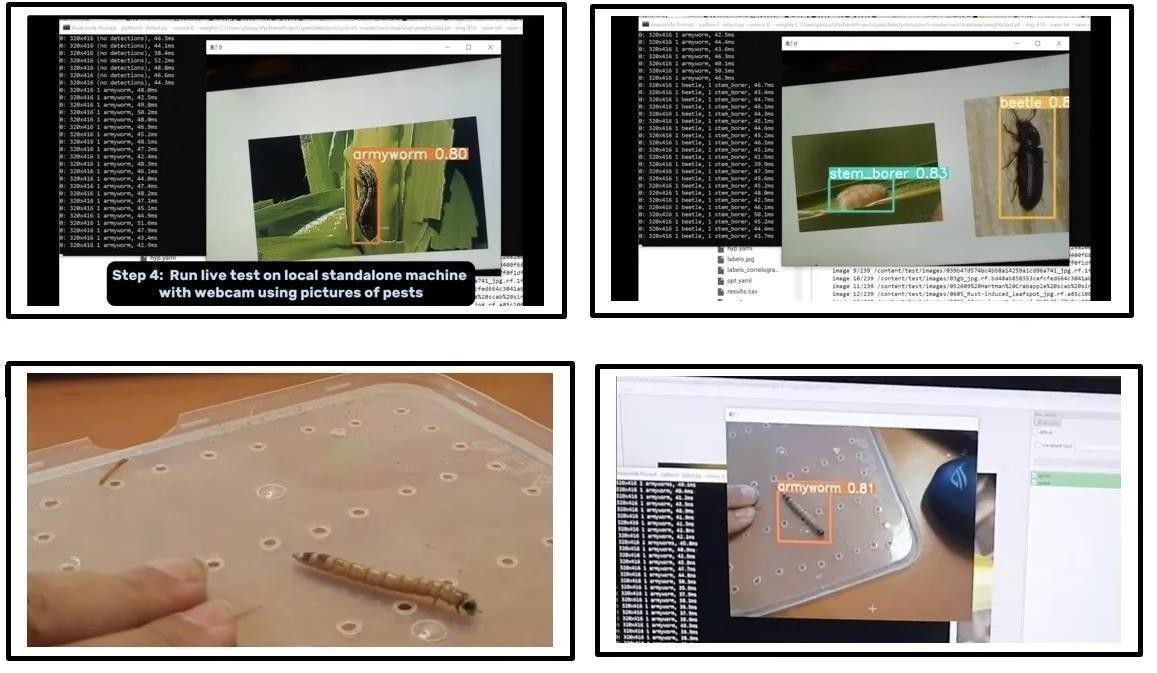
### Sample Output Screens

****

**FIG 4.9 : IDENTIFYING PEST**

****

**FIG 4.10 : DETECTING DIFFERENT TYPES OF PEST**



**FIG 4.11 : TYPES OF PESTS**

* **Image 1**: This shows the detection of a "grasshopper" using a trained pest detection model. The bounding box around the grasshopper, labeled with its name and confidence score (0.82), indicates successful identification. The console logs on the side display real-time processing results, including detected pests and processing times.
* **Image 2**: This grid illustrates multiple examples of pest detection on various images. Each pest is identified with its name, confidence score, and highlighted by a colored bounding box. Examples include aphids, stem borers, mites, grasshoppers, armyworms, and beetles, demonstrating the model's capability across diverse pest species.
* **Image 3**: These images depict real-time testing of the model with a webcam. It identifies pests like an "armyworm" or "stem borer" from live feeds or physical samples, showing the practical application of the project in real-world settings. The confidence scores and bounding boxes indicate accurate detection even in dynamic scenarios.

### Screen Reports

##### Home/Welcome Screen

* + **Description**: The initial screen where users land when they open the application.

##### Fields and Elements:

* + - Title: "AI-Powered Pest Control System"
    - Description: "Identify pests and get actionable recommendations for sustainable farming."
    - Navigation to the upload section.
    - **Expected User Action**: Click "Upload and Analyze" to proceed.

##### Upload Screen

* + **Description**: Allows users to upload pest-affected crop images for analysis.

##### Fields and Elements:

* + - File Upload Button: Accepts JPEG or PNG images only.
    - Submit Button: Initiates the analysis process.
    - Upload Validation: Alerts if an invalid file type is selected.
  + **Expected User Action**: Upload a valid pest image and click "Upload and Analyze.

##### Sample Error Message:

Error: Please upload a valid pest image (JPEG or PNG).

##### Results Screen

* + **Description**: Displays the pest analysis results, including pest type, pest name, confidence level, and control recommendations.

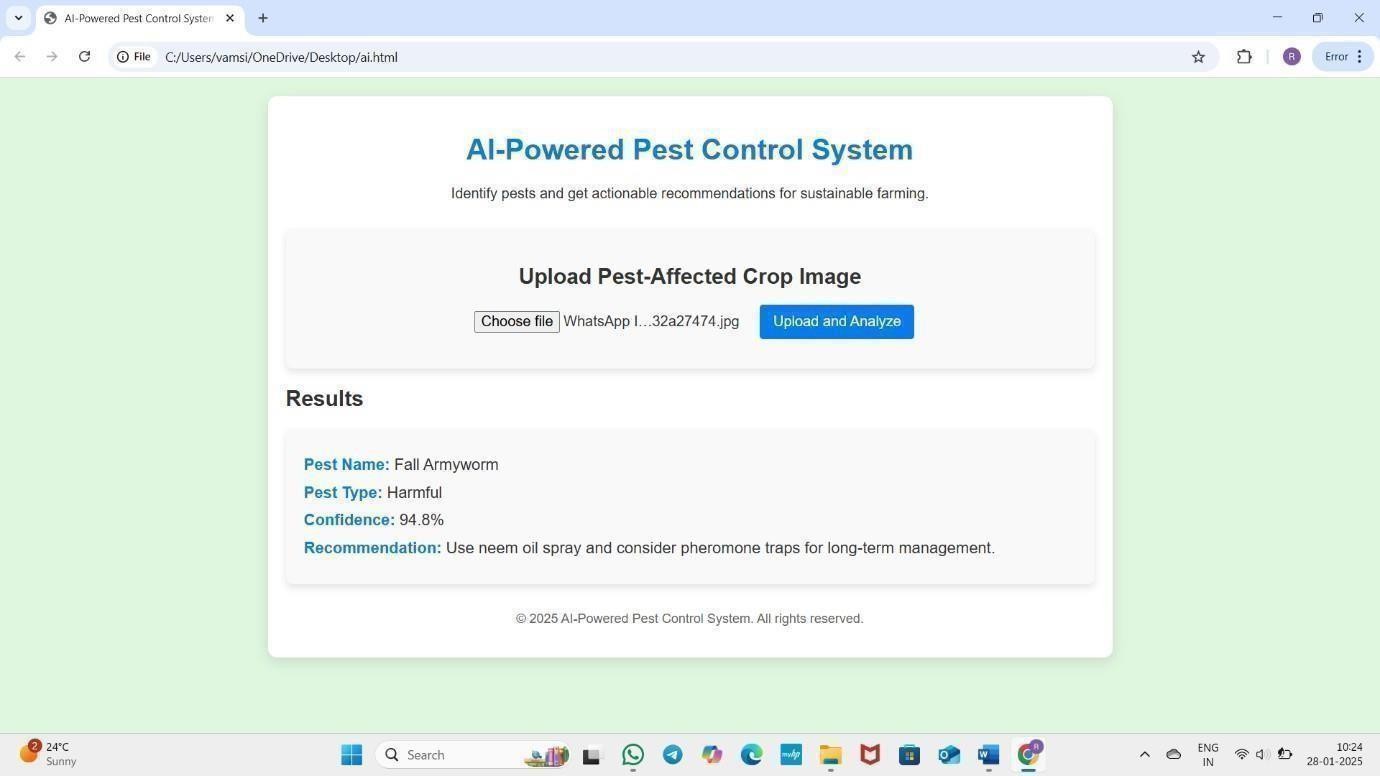
##### Fields and Elements:

* + - Pest Name: Displays the detected pest name (e.g., "Fall Armyworm").
    - Pest Type: Classifies the pest as harmful or harmless.

##### Sample Data Displayed:

* + Pest Name: Fall Armyworm
  + Pest Type: Harmful
  + Confidence: 94.8%

Recommendation: Use neem oil spray and consider pheromone traps for long-term management.



**FIG 4.12 : SAMPLE OUTPUT SCREEN**

**CHPAPTER -5**

**TESTING AND RESULT ANALYSIS**

### Introduction to Testing

Testing plays a crucial role in ensuring the effectiveness and reliability of our **AI-based pest detection system**. The primary goal of testing in this project is to validate the accuracy of pest classification, optimize real-time detection performance, and ensure seamless functionality in practical scenarios.

To achieve this, we conducted the following tests:

* + - **Model Accuracy Testing:** The trained model was tested on a labeled dataset to evaluate its classification accuracy. The results were compared using performance metrics such as **precision, recall, and F1-score** to ensure correct pest identification.
    - **Performance Testing:** The system’s detection speed and responsiveness were tested under different environmental conditions to assess its real-time feasibility. This helps in ensuring **low latency and high efficiency** when deployed in the field.
    - **Optimization Testing:** The impact of different training techniques, such as **Stochastic Gradient Descent (SGD)** and **Adam optimizer**, was analyzed to improve model performance. This ensures that the model generalizes well across different pest images.
    - **Usability Testing:** Since the system is deployed as a **web-based application**, extensive usability tests were conducted to evaluate **user interface (UI) responsiveness, ease of use, and error handling**. The goal was to create a system that is accessible and user-friendly for farmers and agricultural professionals.

### Types of Testing

To ensure the accuracy, efficiency, and usability of our AI-powered pest detection system, we implemented multiple testing strategies. These strategies helped validate the effectiveness of our model, optimize its performance, and enhance the overall user experience.

**1. Functional testing:** Testing the system’s response to various image inputs, including high- resolution, low-resolution, and grayscale images.

* + **Correctness Testing:** Ensuring that the model correctly classifies pests based on the trained dataset.

##### Performance Testing

Performance testing was conducted to measure the efficiency and speed of the system in real- time environments. This included:

**Latency Testing:** Ensuring minimal response time when deployed as a web-based application.

##### Model Evaluation Metrics

To assess the accuracy of our CNN-based classification model (ResNet/MobileNet), we used standard evaluation metrics:

* + **Accuracy:** The percentage of correctlyclassified pest images.
  + **Precision & Recall:** Ensuring correct identification of pests while minimizing false positives.
  + **F1-Score:** Balancing precision and recall to measure overall model performance.

##### Security Testing

Since the system is deployed as a web-based application, security testing was conducted to prevent vulnerabilities such as:

* + **Data Privacy Risks:** Ensuring that image uploads and processing are securely handled.
  + Injection Attacks Prevention: Validating user inputs to prevent malicious attacks on the application.

##### Usability Testing

To ensure the application is farmer-friendly and easy to use, we conducted usability testing by evaluating:

* + **User Interface (UI) Simplicity:** Ensuring a clear and intuitive UI using Gradio/Flask.
  + **Error Handling:** Testing how the system responds to incorrect or unsupported image inputs.

### Test cases and Test Reports

##### Test Case 1: Upload a Trained Pest Image

* + **Objective:** Verify that the system correctly detects a pest image included in the training dataset.

##### Test Steps:

* + - Open the web-based application.
    - Upload a **trained pest image** (e.g., Cutworm).

##### Expected Outcome:

* + - **Detected Pest:** Cutworm (or another trained pest).
    - **Pesticide Recommendation:** Chlorpyrifos, Cypermethrin (or relevant pesticide).
  + **Status:** Passed.

##### Test Case 2: Upload an Untrained Pest Image

* + **Objective:** Evaluate how the system responds to a pest image not included in the training dataset.

##### Test Steps:

* + - Open the web-based application.
    - Upload an image of a **pest not in the training dataset**.
    - Click the "Detect" button.
    - Observe the displayed result.

##### Test Case 3: Invalid User Login (Wrong Username or Password)

* + **Objective:** Ensure the system prevents unauthorized access when incorrect credentials are entered.

##### Test Steps:

* + Open the login page of the web application.
  + Enter an **incorrect username or password**.
  + Click the "Login" button.
  + Observe the system’s response.

##### Expected Outcome:

* + The system should **reject the login attempt**.
  + An **error message** should be displayed: **"Invalid username or password."**
* **Status:** Passed.

##### Test Case 4: UI Usability Test

* + **Objective:** To evaluate the user experience of the web-based application.

##### Test Steps:

* + - Navigate through the application.
    - Upload images and check the detection process.
    - Review the layout, buttons, and error messages.

##### Expected Outcome:

* + - The interface should be **easy to use and responsive**.
    - The results should be **clearly displayed** with pest details and recommendations.
  + **Status:** Passed.

### Implementation process

The implementation of the AI-powered pest detection system involved multiple stages, including dataset preparation, model training, evaluation, deployment, and user interaction. This section details the complete process based on the Python scripts, research paper, and project files.

##### Dataset Preparation and Preprocessing

* The dataset was sourced from agricultural pest image repositories (e.g., Kaggle).
* The dataset included images of various pest species, labeled according to their category.

datagen = ImageDataGenerator( rescale=1./255,

rotation\_range=20, width\_shift\_range=0.2, height\_shift\_range=0.2, shear\_range=0.2,

)

##### Splitting the Dataset

* The dataset was split into **80% training** and **20% testing** sets using train\_test\_split: from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test =train\_test\_split(X, y, test\_size=0.2, random\_state=42)

##### Model Selection and Training

* The **VGG16** model(pre-trained on ImageNet) was chosen for transfer learning.
* A **custom classifier** was built on top of VGG16 with **fully connected layers**: from keras.applications import VGG16

fromkeras.layers import Flatten, Dense, Dropout from keras.models import Model

base\_model=VGG16(weights='imagenet',include\_top=False,input\_shape=(224, 224, 3))

top\_model=base\_model.output top\_model=Flatten()(top\_model) top\_model= Dropout(0.5)(top\_model)

output\_layer = Dense(17, activation='sigmoid')(top\_model)

model= Model(inputs=base\_model.input, outputs=output\_layer)

**Training** was performed with **25 epochs** using the training data generator: history= model.fit(train\_generator,epochs=25,

validation\_data=test\_generator, batch\_size=32)

##### Model Evaluation and Optimization

* The trained model was evaluated using **accuracy, confusion matrix, and loss curves**.
* **Accuracy Plot** was generated using Matplotlib: import matplotlib.pyplot as plt

plt.plot(history.history['accuracy'], label='Training Accuracy') plt.plot(history.history['val\_accuracy'],label='Validation Accuracy') plt.title("Model Accuracy")

plt.xlabel("Epochs") plt.ylabel("Accuracy"

)

plt.legend() plt.show()

##### Implementing Pest Detection and Pesticide Recommendation

* The **pest detection pipeline** takes an image, applies preprocessing, and classifies it using the trained model.
* The system then **retrieves pesticide recommendations** based on the detected pest froma

##### predefined database.

img = image.load\_img(image\_path, target\_size=(224, 224)) img\_array = image.img\_to\_array(img) / 255.0

img\_array= np.expand\_dims(img\_array, axis=0)

prediction=model.predict(img\_array) predicted\_class = np.argmax(prediction, axis=1) return predicted\_class

##### Web-Based User Interface Development

* The user interface was developed using **Flask**, allowing users to upload pest images for detection.

##### Key Features Implemented:

* + Image Upload
  + Real-time Detection API

##### Display of pest classification, pesticide recommendation, and harmfulness level

1. **Deployment and Hosting**

* The final model was **saved and deployed** for real-time access.

##### Model Saving and Loading:

model.save("PEST\_Detection.h5")

fromtensorflow.keras.modelsimport load\_modelloaded\_model= load\_model("PEST\_Detection.h5")

### Implementation Steps

The **AI-powered pest detection system** was implemented using deep learning with a **Convolutional Neural Network (CNN)** model trained on an image dataset of agricultural pests.The system is designed to classify pests and provide

recommendations based on the detected species. The implementation was carried out in a structured manner, covering **dataset preprocessing, model training, evaluation, and deployment**.

##### Step 1: Dataset Collection and Preprocessing

The dataset was sourced from **agricultural pest image repositories** (e.g., Kaggle and other sources).

**Rotation:** ±20°

##### Width and height shifts: ±20%

**Shearing and zooming:** ±20%

**Horizontal flipping:** Enabled

**Rescaling:** Pixel values normalized to **(0,1)**

##### Step 2: Splitting theDataset

The dataset was split into **training (80%)** and **testing (20%)** sets using

##### train\_test\_split.

This ensured a **balanced dataset** for both **model learning and evaluation**.

Example image samples were **visualized using Matplotlib and Seaborn**

to confirm data distribution.

##### Step 3: Model Selection and Training

The **VGG16 model (pretrained on ImageNet)** was chosen for **transfer learning**.

The last few layers of **VGG16 were fine-tuned**, while keeping earlier layers frozen.

A **custom classifier** was added with the following architecture:

##### Flatten layer

**Dropout layer (0.5)** for regularization

Only the last few convolutional blocks (e.g., block5) were unfrozen to allow fine-tuning on task-specific features.

Earlier layers were frozen to reduce training time and avoid overfitting on small datasets.

**Fully connected (Dense) layer with 17 output nodes** (one for each pest class)

##### Training Details

The model was **compiled using**:

**Optimizer:** Adam

**Loss Function:** Binary Cross entropy

**Metrics:** Accuracy

The model was trained using the **train\_generator (augmented images)**.

**25 epochs** were used to allow gradual model convergence.

The training process included **batch processing (batch size = 32)** to improve learning stability.

##### Step 4: Model Evaluation and Optimization

Model performance was evaluated using:

**Classification accuracy**

**Confusion matrix**

**Precision-recall scores**

Validation accuracy was **approximately 90%**, with potential for further improvements.

##### Step5: Implementing Image Processing and Feature Extraction

Before passing images to the model, preprocessing was applied using

##### OpenCV and Keras utilities:

**Resizing images to 224×224 pixels** to match VGG16 input dimensions.

**Converting to grayscale** (if necessary).

**Normalizing Pixel Values**:  
Scaling the pixel values to a range of 0 to 1 (dividing by 255) or to [-1, 1] as per model requirement for better convergence during training.

**Image Array Conversion**:  
Using img\_to\_array() from Keras to convert images to NumPy arrays which can be processed by deep learning models.

**Data Augmentation (optional during training)**:  
Applying techniques like rotation, zoom, flipping, and shifting to make the model more robust and reduce overfitting.

**Step 6: Detected Pest Name** (e.g., Cutworm, Aphid, etc.).

**Pesticide Recommendations** (e.g., Chlorpyrifos, Cypermethrin).

**Harmfulness Assessment** (e.g., Harmful/Non-harmful).

**Pest Details** (brief description of its impact on crops).

The recommendation system is **built using a predefined pest- pesticide mapping database**.

##### Step 7: Developing the Web-Based User Interface

A **Flask-based web application** was developed to allow users to **upload pest images for detection**.

The web interface includes

**Image upload feature** using HTML file input.

**Real-time detection** using a backend API call.

##### Dynamic result display for pest classification and pesticide recommendations.

**Step 8: User Authentication and Security Implementation**

A **login system** was implemented to **restrict access to authorized users**.

##### User authentication features include:

**Username-password validation**

**Session management** to prevent unauthorized access

##### Error handling for incorrect login attempts

**Step 9: Testing and Debugging**

The system was tested using **multiple pest images** to verify classification accuracy.

Various test cases were executed, including:

**Invalid image input handling** (error messages for non-pest images).

**Edge cases (blurry or rotated images)** to check robustness.

**User login validation** for security testing.

##### Step10: Deployment and Hosting

The trained model (PEST\_Detection.h5) was **saved and deployed**.

The web application was **hosted on Heroku/Railway.app** to allow public access.

**Gunicorn WSGI server** was used to **handle Flask requests efficiently**.

The system was **tested for real-time usage on different devices (desktop, mobile)**.

### Implementation Procedure

The **AI-powered pest detection system** was implemented using **deep learning and a web- based application** to classify pests and provide pesticide recommendations. **Dataset Collection and Preparation**

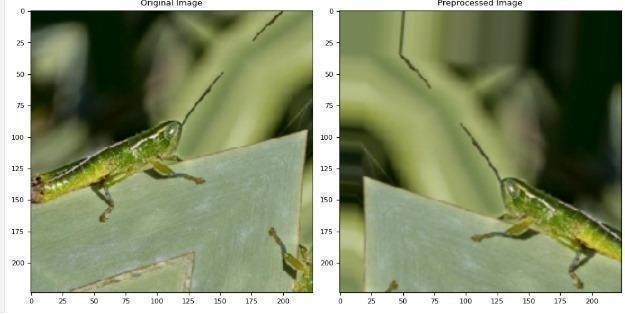
* Pest images were collected from **Kaggle and agricultural research datasets**.
* Images were categorized into folders based on **pest species**.



**FIG 5.1 : DIFFERENT TYPES OF HARMFUL PESTS**

#### Data Preprocessing

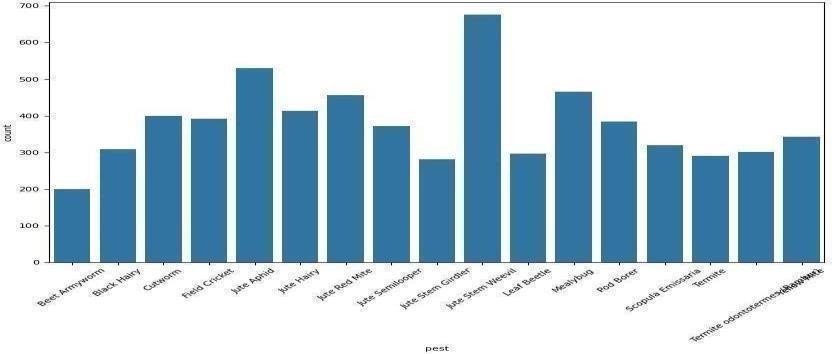
* + Resize images for uniformity.
  + Apply image augmentation techniques like rotation, flipping, and zooming.
  + Convert images to numerical arrays.



**FIG 5.2 : NORMALIZING A PEST**

#### Exploratory Data Analysis (EDA)

* + Visualize pest distribution using bar graphs.
  + Identify data imbalances.



**FIG 5.3 : GRAPHS ABOUT PESTS**

##### Model Development and Training

* + **VGG16 (CNN model)** was used for **pest classification** with additional layers for better accuracy.
  + The dataset was split into **80% training and 20% testing**.
  + The modelwas trained using **25 epochs** with **Adam optimizer and binary cross- entropy loss**.

##### Model Evaluation

* + The model was tested using **accuracy, confusion matrix, and precision-recall scores**.

##### Pest Detection and Pesticide Recommendation

* Once an image is uploaded, the system detects and recommends.

1. **Web-Based User Interface**

* A **Flask-based web app** allows users to **upload images and get real-time results**.
* The interface includes **image upload, detection output, and pesticide recommendations**.

##### Security and Testing

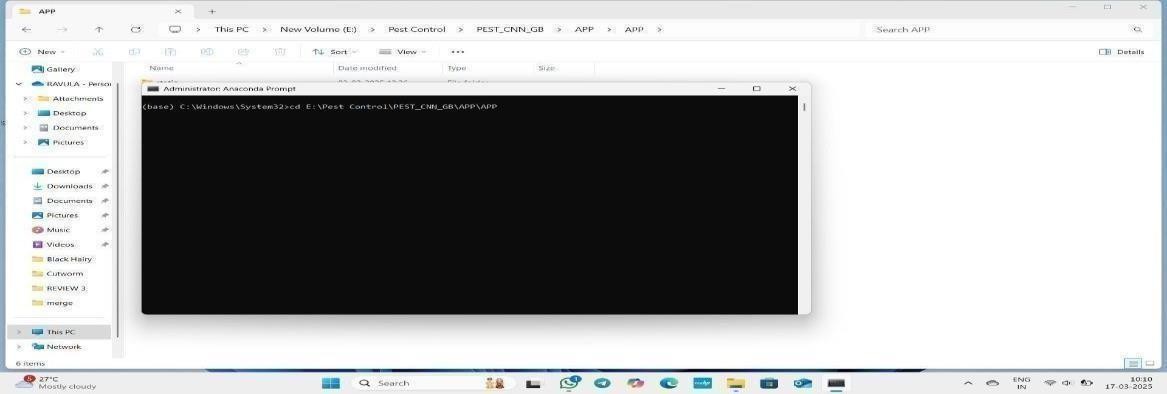
* **User authentication** was implemented to restrict access.
* The system was tested for **image classification, security validation, and usability**.

##### Deployment

* The model was **saved and deployed** on **Heroku/Railway.app**.

### User Manual

**Step 1 :** open anaconda prompt and run it as administrator.



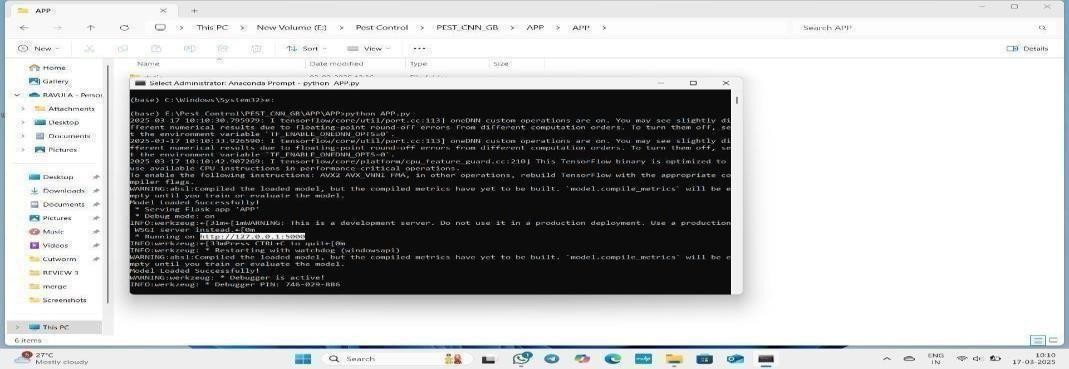
**FIG 5.4 : OPENING ANACONDA PROMPT**

**Step 2 :** copy the path dataset and codes/trained images and paste in anaconda prompt as cd path and press enter.

**Step 3 :** type e: and press enter to change the disk.

**Step 4 :** type python APP.py and press enter to run the model.

**Step 5 :** now copy the url as obtained below in anaconda prompt copy it and paste in browser.



**FIG 5.5 : RUNNING PATHS**

**Step 6 :** now you get a dashboard in that click register and type username, password and address and register in it.



**FIG 5.6 : LOGIN PAGE**

**Step 7 :** now login with above registered credentials now you have to upload trained pest images from data in your computer/laptop.

**Step 8 :** now you will get the predicted result regarding the uploaded pest.



**FIG 5.7 : FINAL OUTPUT**

**CHAPTER -6**

**CONCLUSIONS AND SCOPE FOR FUTURE WORK**

### Conclusion

The **Pest Classification System** developed in this project is a significant step forward in leveraging **deep learning and image processing techniques** for **automated pest identification**. In modern agriculture, pest infestations pose a severe threat to crop health, leading to **substantial losses in yield and quality**. Early detection and classification of pests are crucial for implementing **targeted pest control measures**, reducing reliance on broad- spectrum pesticides, and promoting sustainable agricultural practices. This project addresses these challenges by developing an efficient and scalable **AI-powered pest classification model** capable of distinguishing different pest species based on image data.

The project began with an extensive **Exploratory Data Analysis (EDA)** to gain insights into the dataset and understand the distribution of pest species. This analysis was essential for **identifying potential data imbalances**, assessing image quality, and ensuring the dataset was well-prepared for model training. Image preprocessing techniques such as **resizing, normalization, and augmentation** were applied to enhance model generalization and prevent overfitting.

At the core of this project lies a **Convolutional Neural Network (CNN)-based classification model**, specifically designed to process and classify pest images with high accuracy. CNNs are particularly effective for image recognition tasks as they can **automatically extract hierarchical features** from input images, reducing the need for manual feature engineering. The trained model demonstrated **robust performance**, achieving high classification accuracy, making it a reliable tool for real-world agricultural applications.

Furthermore, integrating this classification model into **mobile applications, drones, or smart farming systems** could revolutionize **precision agriculture**. By providing farmers with an **instant and accurate identification tool**, this technology empowers

them to **implement targeted pest management techniques**, reducing excessive pesticide use and promoting **eco-friendly farming practices**. The scalability of this system allows for **wide adoption across different agricultural regions**, ensuring that pest-related challenges can be addressed effectively and efficiently.

### Future Enhancements

While the **Pest Classification System** developed in this project has demonstrated high accuracy and efficiency, there are several areas where further enhancements can be made to improve its **performance, scalability, and real-world applicability**. The following future enhancements can be considered:

##### Expanding the Dataset

* + Increasing the size and diversity of the dataset by incorporating images from

**multiple geographical locations** and various environmental conditions.

* + Including more **pest species** to improve the model’s ability to classify a **broader range of agricultural pests**.
  + Adding **images with different angles, lighting conditions, and backgrounds** to enhance model robustness in real-world scenarios.

##### Integration with Real-time Monitoring Systems

* + Deploying the classification model into **IoT-based smart farming systems** that can continuously monitor crops using **drones or camera sensors**.
  + Enabling **real-time pest detection and alerts** to notify farmers immediately when a pest is identified.

##### Hybrid AI Models for Improved Accuracy

* + Enhancing classification accuracy by combining **CNNs with transformer-based models like Vision Transformers (ViTs)** for **better feature extraction and pattern recognition**.

improve prediction reliability.

##### Multilingual and Voice-based Assistance

* + Adding **multilingual support** to make the system accessible to farmers from different linguistic backgrounds.
  + Integrating **voice-based assistance** to help farmers who may not be comfortable with text- based interfaces.

**Project Mapping with various courses of the Curriculum with attained POs**

|  |  |  |
| --- | --- | --- |
| Name of Course from which Principles are  applied in this project | Description of the Application | Attained POs |
| C311 | Thorough Examination of Existing systems and definition of the problem | PO1,PO2, PO6,PO7 |
| C324, C325, C415 | Gathering, Analysis and classification of all the requirements for the proposed system | PO1,PO2,PO4 |
| C425, C314, C225 | Logical design is done by using logical tools | PO1, PO2,PO3, PO10 |
| C228, C325, C324 | The physical design is done by using Android Studio,  and database MySQL. | PO5,PO8,PO10,P  O11,PO12 |
| C425, C324, C414 | Each and every module is tested, integrated, and evaluated. | PO8,PO10,PO11, PO12 |
| C425, C414 | Implementation of the project in the real environment | PO7,PO8,PO10,P  O11 |
| C425, C3110, C329 | Documentation is done by all the team members with  well defined communication with inclusion of Engineering and management Principles | PO8,PO9,PO10,P O11,PO12 |
| C425, C3110, C329 | Presentation of the work in teams with proper method of presentation. | PO9,PO8,PO10, PO11 |

**COURSE STRUCTURE (R13)**

**I YEAR- I SEMESTER**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S No** | **Subject** | **T** | **P** | **Credits** |
| C111 | Problem Solving and Programming Using C | 3 | 0 | 3 |
| C112 | Applied Chemistry | 3 | 0 | 3 |
| C113 | Differential Equations | 3 | 0 | 3 |
| C114 | Engineering Graphics | 1 | 4 | 3 |
| C115 | Basics of Electrical and Electronics Engineering | 3 | 0 | 3 |
| C116 | Problem Solving and Programming Using C Lab | 0 | 3 | 1.5 |
| C117 | IT Workshop | 0 | 3 | 1.5 |
| C118 | Applied Chemistry Lab | 0 | 3 | 1.5 |
| **Total Credits** | |  |  | **19.5** |

**I YEAR- II SEMESTER**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S No** | **Subject** | **T** | **P** | **Credits** |
| C121 | Communicative English | 3 | 0 | 3 |
| C122 | Applied Physics | 3 | 0 | 3 |
| C123 | Linear Algebra & Vector Calculus | 3 | 0 | 3 |
| C124 | Digital Logic Design | 3 | 0 | 3 |
| C125 | Python Programming | 3 | 0 | 3 |
| C126 | Environmental Sciences | 2 | 0 | 0 |
| C127 | Communicative English Skills Lab | 0 | 3 | 1.5 |
| C128 | Applied Physics Lab | 0 | 3 | 1.5 |
| C129 | Python Programming Lab | 0 | 3 | 1.5 |
| **Total Credits** | |  |  | **19.5** |

**II YEAR- I SEMESTER**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S No** | **Subject** | **T** | **P** | **Credits** |
| C211 | Probability & Statistics | 3 | 0 | 3 |
| C212 | Mathematical Foundations of Computer Science | 3 | 0 | 3 |
| C213 | Data Structures & Algorithms | 3 | 0 | 3 |
| C214 | Object Oriented Programming through Java | 3 | 0 | 3 |
| C215 | Introduction to Artificial Intelligence | 3 | 0 | 3 |
| C216 | Constitution of India | 2 | 0 | 0 |
| C217 | Data Structures &Algorithms Lab | 0 | 3 | 1.5 |
| C218 | Object Oriented Programming through Java lab | 0 | 3 | 1.5 |
| C219 | Introduction to Artificial Intelligence Lab | 0 | 3 | 1.5 |
| C2120 | Skill Oriented Course -1 | 1 | 2 | 2.0 |
| **Total Credits** | |  |  | **21.5** |

**II YEAR- II SEMESTER**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S No** | **Subject** | **T** | **P** | **Credits** |
| C221 | Numerical Methods & Transformations | 3 | 0 | 0 |
| C222 | Computer Organization | 3 | 0 | 0 |
| C223 | Database Management Systems | 3 | 0 | 0 |
| C224 | Formal Languages and Automata Theory | 3 | 0 | 0 |
| C225 | Managerial Economics and Financial Accountancy | 3 | 0 | 0 |
| C226 | Database Management Systems Lab | 0 | 0 | 3 |
| C227 | Web Application Development Lab | 0 | 0 | 3 |
| C228 | R Programming Lab | 0 | 0 | 3 |
| C229 | Skill Oriented Course -2 | 1 | 0 | 2 |
| **Total Credits** | |  |  | **21.5** |

**III YEAR- I SEMESTER**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S No** | **Subject** | **T** | **P** | **Credits** |
| C311 | Design and Analysis of Algorithms | 3 | 0 | 3 |
| C312 | Machine Learning | 3 | 0 | 3 |
| C313 | Operating Systems | 3 | 0 | 3 |
| C314 | Professional Elective -1 | 3 | 0 | 0 |
| C315 | **Open Elective -1** | 3 | 0 | 3 |
| C316 | **Skill Oriented Course – III** | 0 | 4 | 2 |
| C317 | Professional Ethics and Human Values | 2 | 0 | 0 |
| C318 | **Summer Internship one Month (Mandatory) after second year(to be evaluated during V**  **Semester** | 0 | 0 | 1.5 |
| C319 | Machine Learning Lab | 0 | 3 | 1.5 |
| C3110 | Operating Systems Lab | 0 | 3 | 1.5 |
| **Total Credits** | |  |  | **21.5** |

**III YEAR- II SEMESTER**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S No** | **Subject** | **T** | **P** | **Credits** |
| C321 | Computer Networks and communications | 3 | 0 | 3 |
| C322 | Deep Learning | 3 | 0 | 3 |
| C323 | Expert Systems | 3 | 0 | 3 |
| C324 | **Professional Elective -2** | 3 | 0 | 3 |
| C325 | **Open Elective -2** | 3 | 3 | 3 |
| C326 | **Skill Oriented Course – IV (Soft Skills)** | 0 | 2 | 2 |
| C327 | Intellectual property rights and patents (IPR&P) | 2 | 0 | 0 |
| C328 | Deep Learning Lab | 0 | 3 | 1.5 |
| C329 | Computer Networks and communications Lab | 0 | 3 | 1.5 |
| C3210 | Mini Project with seminar | 1 | 2 | 1.5 |
| **Total Credits** | |  |  | **21.5** |

**IV YEAR- I SEMESTER**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S No** | **Subject** | **T** | **P** | **Credits** |
| C411 | **Professional Elective-III** | 3 | 0 | 3 |
| C412 | **Professional Elective-IV** | 3 | 0 | 3 |
| C413 | **Professional Elective-V** | 3 | 0 | 3 |
| C414 | **Open Elective-III** | 3 | 0 | 3 |
| C415 | **Open Elective - IV** | 3 | 0 | 3 |
| C416 | Management Science | 3 | 0 | 3 |
| C417 | **Skill Oriented Course** –V | 0 | 4 | 2 |
| C418 | **Industrial/Research Internship one months (Mandatory) after third year**  **(to be evaluated during VII semester)** | 0 | 0 | 1.5 |
| **Total Credits** | |  |  | **23** |

**IV YEAR- II SEMESTER**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S No** | **Subject** | **T** | **P** | **Credits** |
| C421 | Project work - Phase II | 0 | 0 | 12 |
| **Total Credits** | |  |  | **12** |

|  |  |  |
| --- | --- | --- |
| **OPEN ELECTIVES** | | |
| Open Elective – 1 ( V Semester )  Python Programming | Open Elective -2 (VI Semester )  Fundamentals of Artificial Intelligence | Open Elective -3 (VII Semester) Human Computer Interaction |
| Open Elective -4 (VII Semester)  Applications of Artificial Intelligence | Skill Oriented Course (Advanced)– IV   1. (MEAN Stack Technologies - Module I- MongoDB, Express.js, Angular JS, Node.js and   AJAX   1. Big Data : Apache Spark 2. DevOPS |  |
| **PROFESSIONAL ELECTIVES** | | |
| Professional Elective – 1 (V Semester ) | **Professional Elective – 2**  **( VI Semester)** | **Professional Elective – 3**  **(VII Semester )** |
| 1. Software Engineering 2. Compile Design 3. Data Visualization 4. Design and Analysis of Algorithms | 1. Software Project Management 2. Distributed Systems 3. Internet of Things 4. Data Where Housing and Data Mining | 1. Reinforcement Learning 2. Soft Computing 3. Cryptography and Network Security 4. NOSQL Databases 5. Natural Language Processing |
| Professional Elective – 4 (VII Semester) | Professional Elective – 5 (VII Semester) |  |
| 1. Robotic Process Automation 2. Cloud Computing 3. Big Data Analytics 4. Block Chain Technologies 5. Image & Video Analytics | 1. Social Network Analysis 2. Recommender Systems 3. Computer vision 4. Object Oriented Analysis and Design 5. Semantic Web |  |

**CHAPTER-7 BIBLIOGRAPHY**

### Books & Journals Referred

Here are the books that were referred to during the project, along with a brief description of their relevance:

##### "Deep Learning for Computer Vision" – Rajalingappaa Shanmugamani

* + This book provides a comprehensive introduction to deep learning techniques used in image classification, object detection, and recognition. It explains CNN architectures such as AlexNet, VGG, ResNet, and their applications in real-world scenarios, making it highly relevant for pest detection in agriculture.

##### "Machine Learning in Agriculture: Applications and Challenges" – S. Shanmugan

* + This book explores the role of machine learning in modern agriculture. It covers techniques like supervised and unsupervised learning, deep learning models, and their applications in crop disease detection, pest monitoring, and yield prediction.

##### "Artificial Intelligence in Agriculture" – S. Ayyappan & K. Ramasamy

* + This book discusses the impact of AI and IoT on agriculture, including the use of computer vision for pest detection. It also provides case studies on how AI-driven tools help in precision farming, pest control, and decision-making in agriculture.

##### "Digital Agriculture: Progress and Prospects" – Sanjeev Kumar & Vikram Bali

* + This book provides insights into how digital technologies, including AI, ML, IoT, and big data, are transforming agriculture. It covers various applications, including pest detection.

##### "International Journal of Computer Vision (IJCV)"

* + - Articles from this journal provided insights into deep learning models like CNNs and their applications in image processing, which were useful for pest detection.

##### "Journal of Artificial Intelligence Research (JAIR)"

* + - Papers from this journal discussed advanced AI models, data preprocessing techniques, and optimization methods relevant to our project.

##### "IEEE Transactions on Image Processing"

* + - This journal provided information on the latest image processing techniques, including object detection methods used in pest identification.

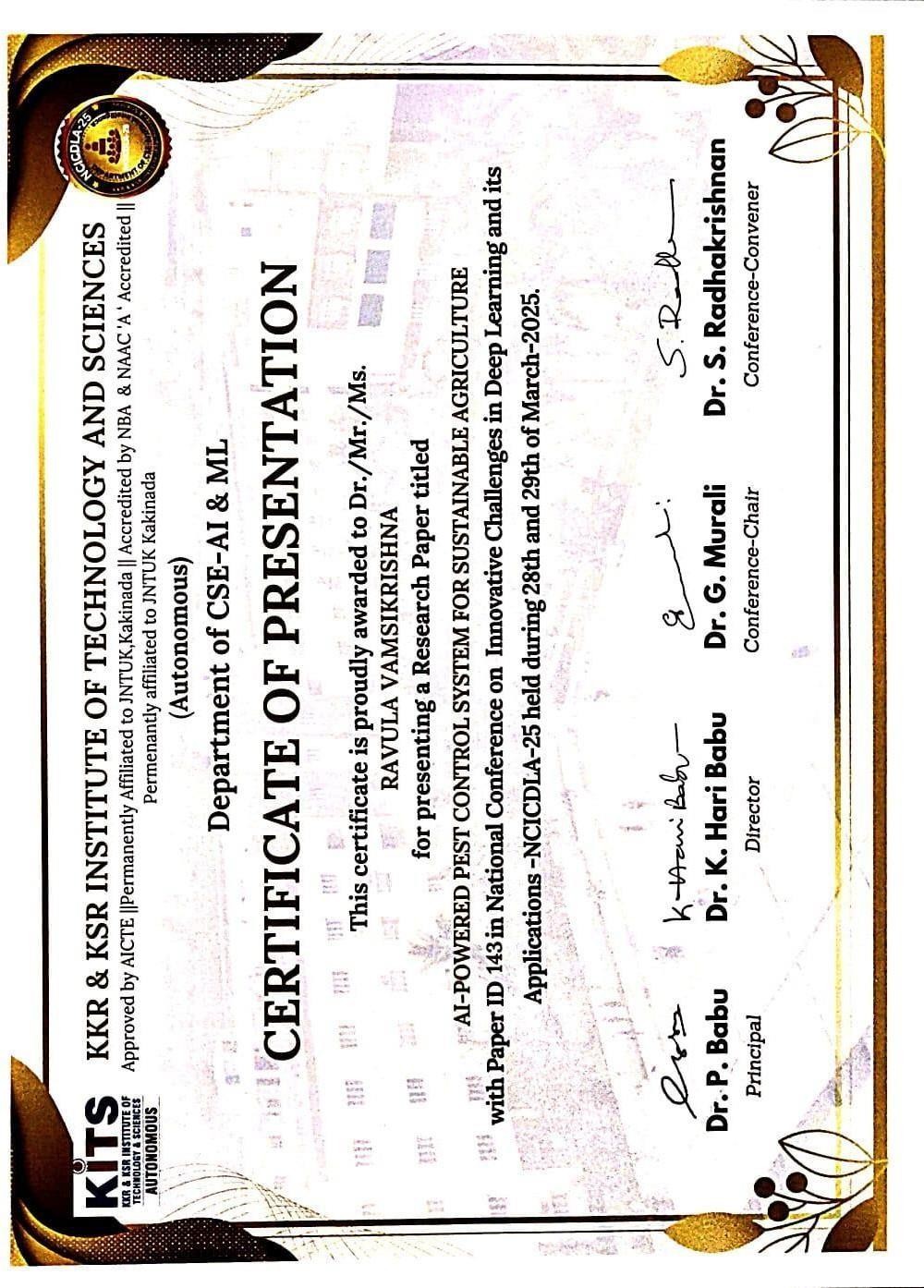
### Websites Visited

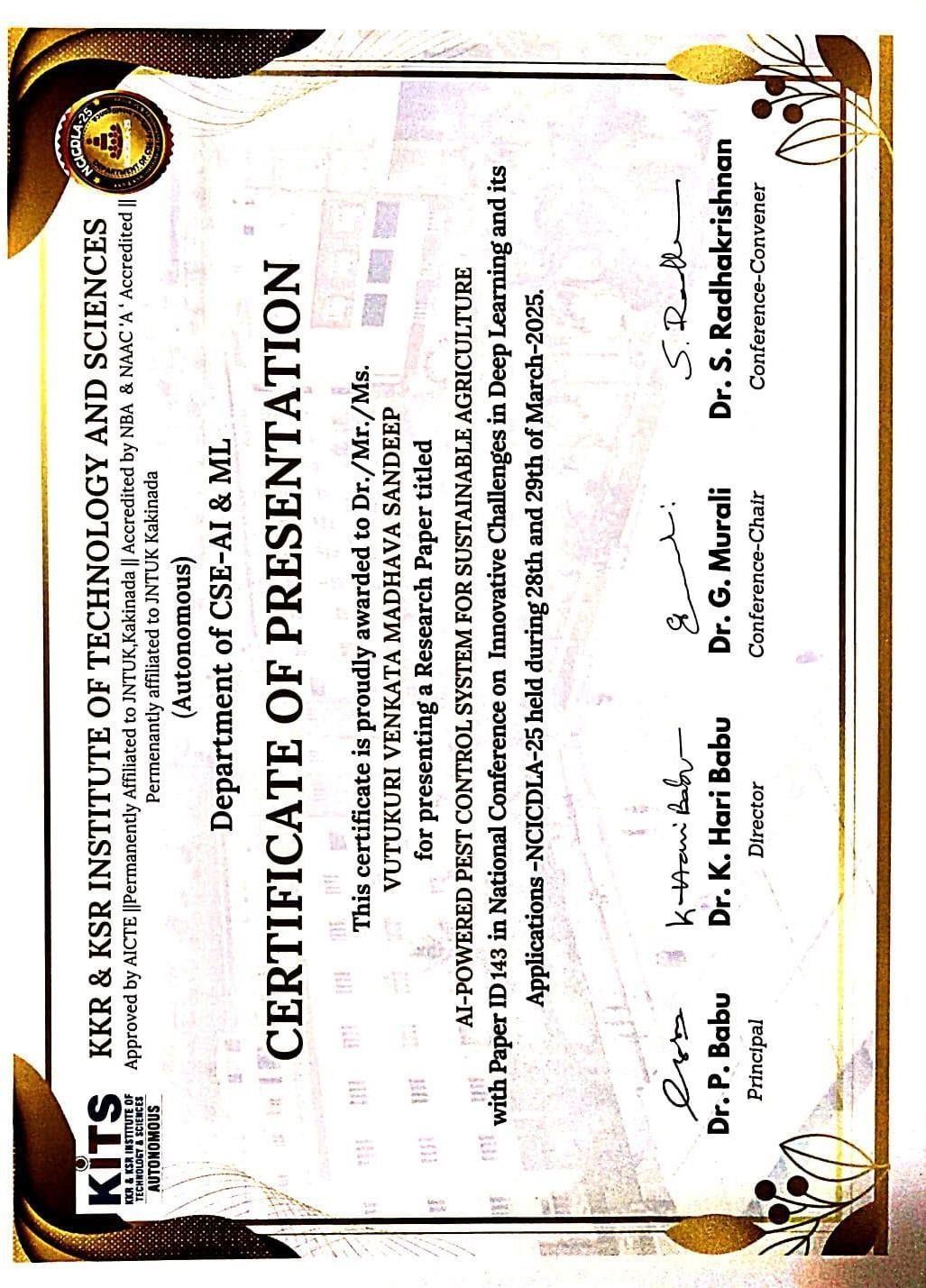
To develop the **AI-powered pest classification system**, various websites were referred to for **research papers, datasets, coding resources, and implementation guidelines**. These websites provided valuable insights into **deep learning models, dataset preparation, pest classification techniques, and deployment strategies**.

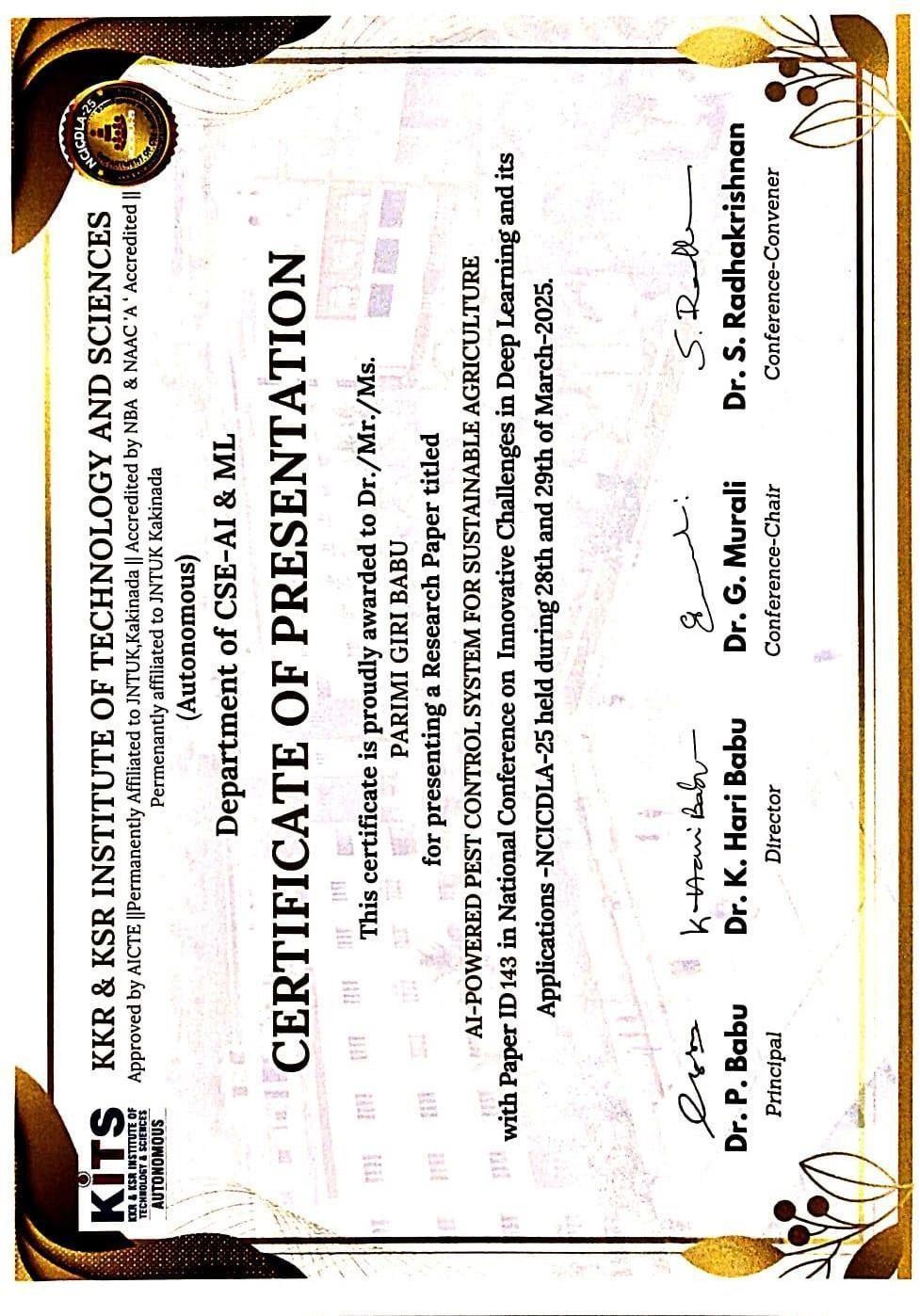
1. **Kaggle (**[**https://www.kaggle.com/**](https://www.kaggle.com/)**)**
   * Used to find **datasets for pest classification** and explore existing machine learning models for agricultural pest detection.
2. **TensorFlow (**[**https://www.tensorflow.org/**](https://www.tensorflow.org/)**)**
   * Referred for documentation on **deep learning frameworks**, model building, and transfer learning techniques.
3. **PyTorch (**[**https://pytorch.org/**](https://pytorch.org/)**)**
   * Used to understand different CNN architectures and implement **custom deep learning models** for image classification.
4. **Scikit-Learn (**[**https://scikit-learn.org/**](https://scikit-learn.org/)**)**
   * Provided insights into **data preprocessing, train-test splitting, and model evaluation OpenCV (**[**https://opencv.org/**](https://opencv.org/)**)**
   * Referred for **image preprocessing techniques**, including resizing, normalization, and data augmentation.
5. **GitHub (**[**https://github.com/**](https://github.com/)**)**
   * Usedfor **exploring open-source projects** related to pest classification and understanding model implementation.
6. **ResearchGate (**[**https://www.researchgate.net/**](https://www.researchgate.net/)**)**
   * Provided access to **peer-reviewed research papers** on AI-based pest detection and image classification.
7. **IEEE Xplore (**[**https://ieeexplore.ieee.org/**](https://ieeexplore.ieee.org/)**)**
   * Used to find **scientific papers** on deep learning-based pest identification and smart agriculture.
8. **MDPI (**[**https://www.mdpi.com/**](https://www.mdpi.com/)**)**
   * Provided **journal articles on AI applications in agriculture**, including CNN-based pest detection.

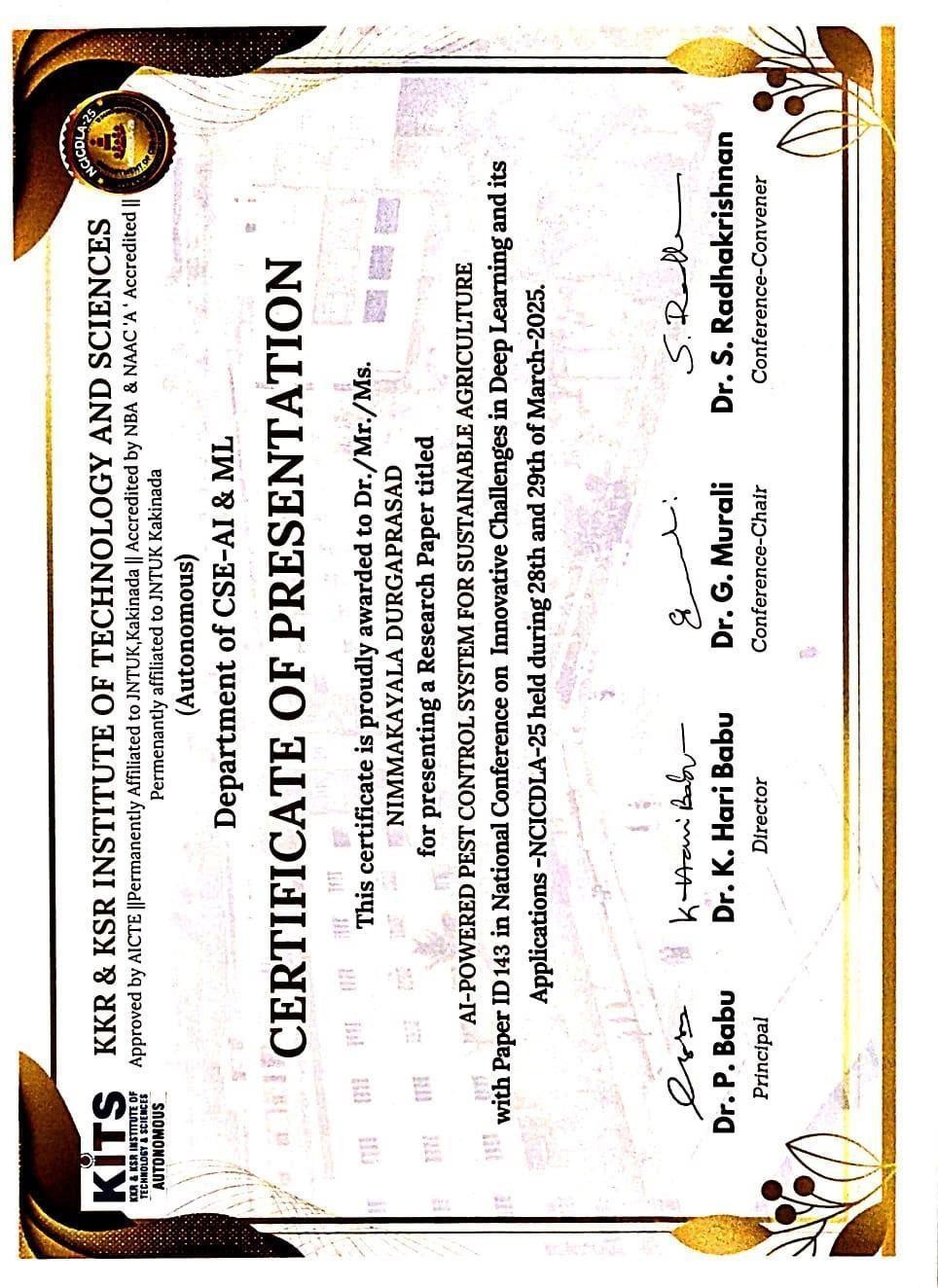
**CHAPTER -8**

**APPENDIX I : CONFERENCE CERTIFICATES OF OUR TEAM**

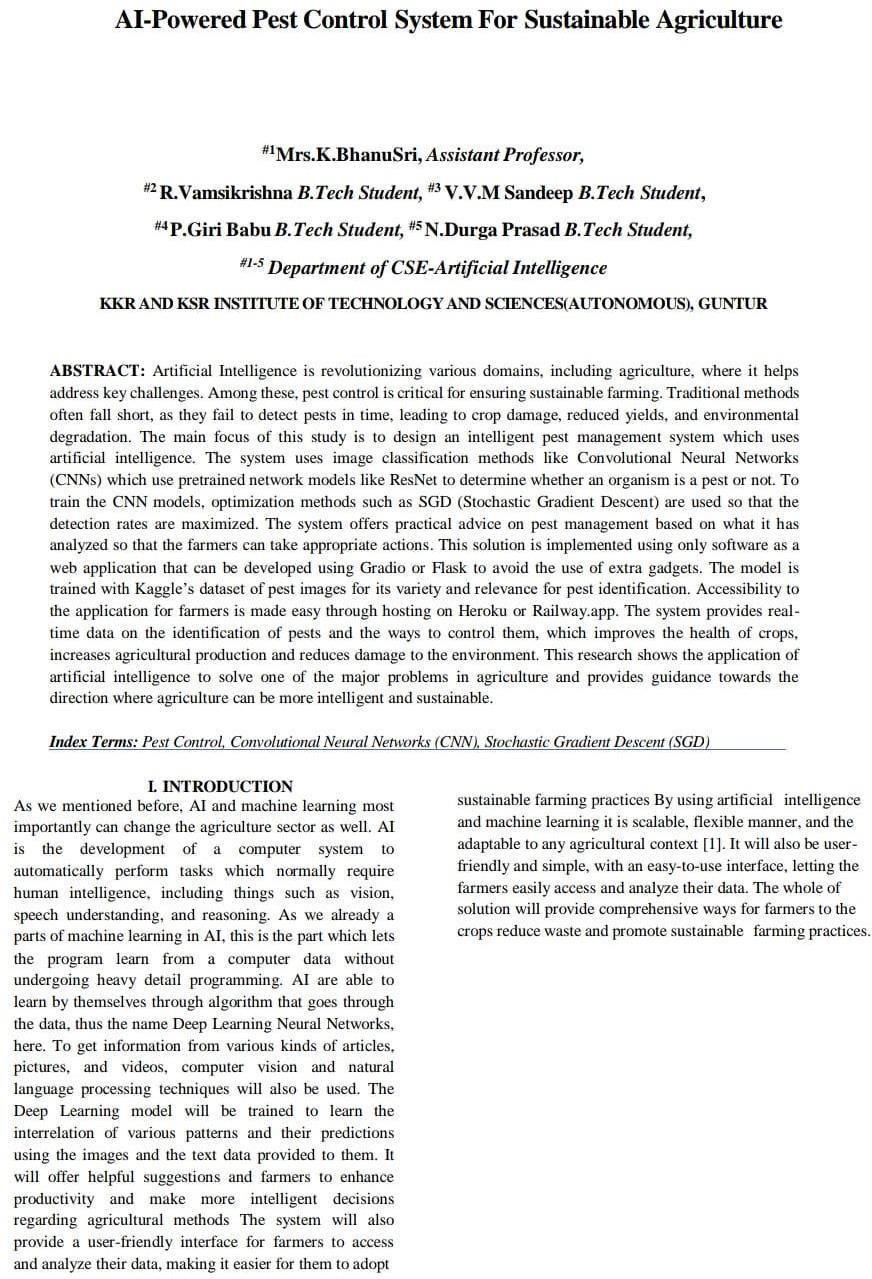
****



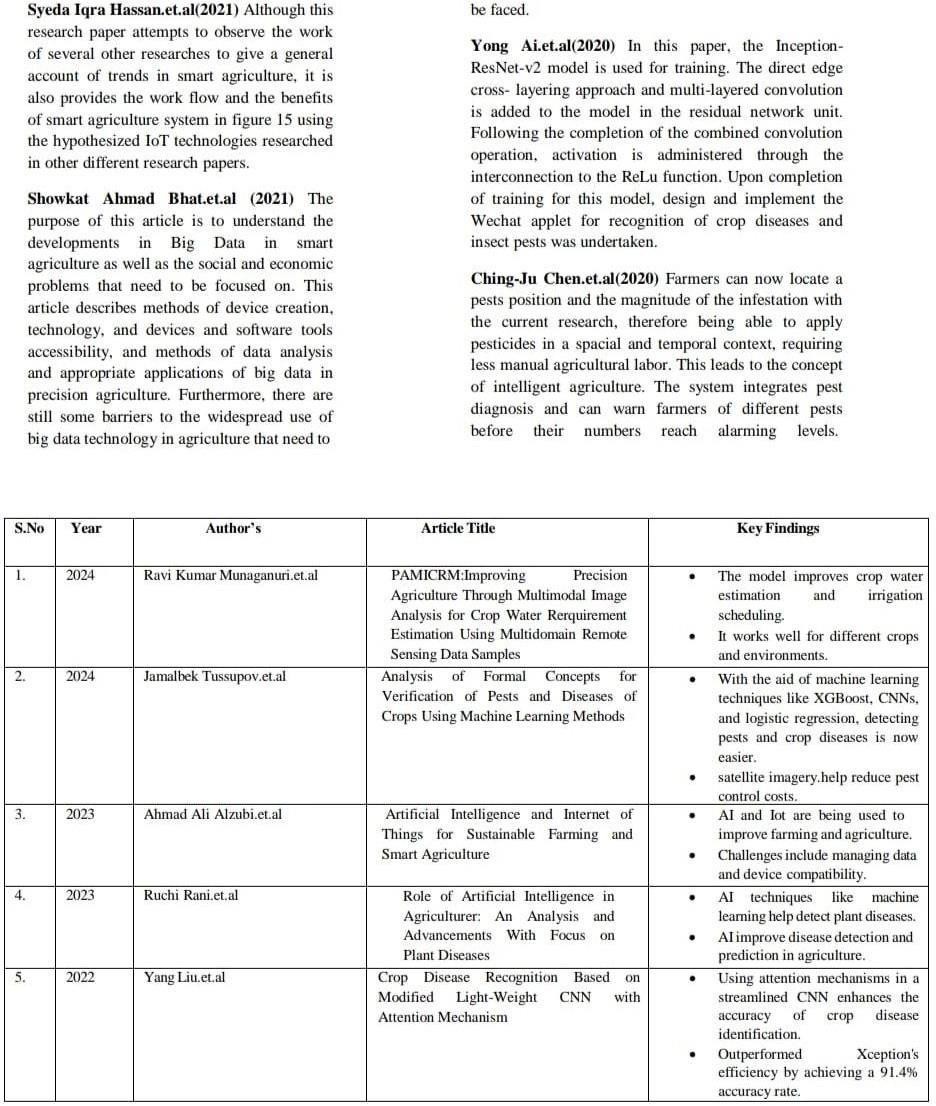




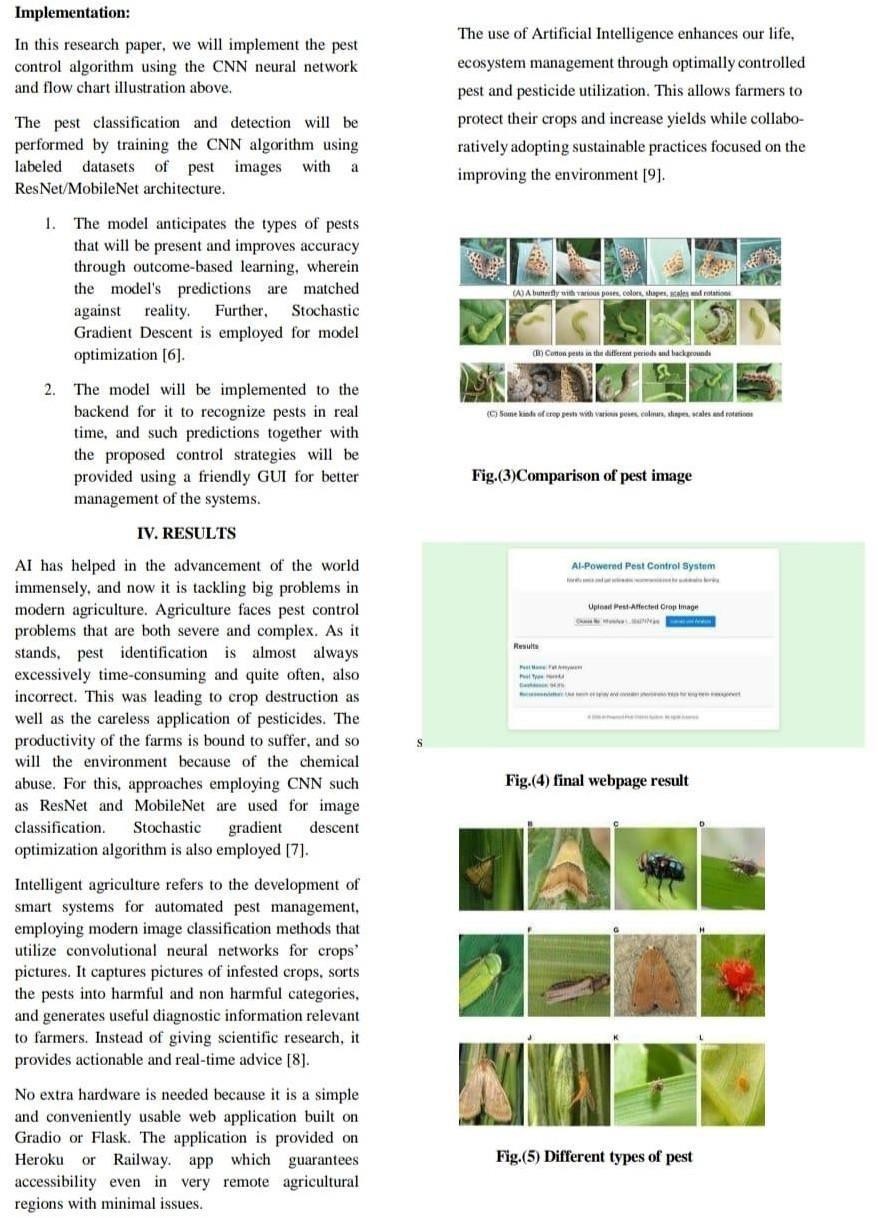
**APPENDIX II : COMPLETE JOURNAL**

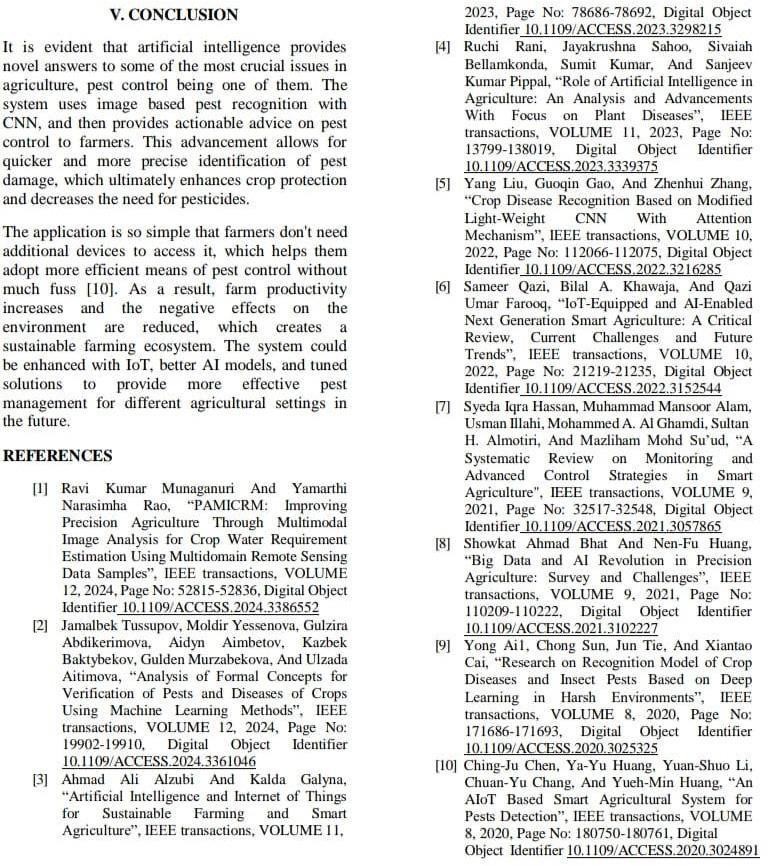
****











**APPENDIX III : INTERNSHIP CERTIFICATES OF OUR TEAM**

****





