Crop Pest Classification Model Using Deep Learning Techniques

A MAIN PROJECT REPORT
submitted in partial fulfillment of the requirements for the award of the Degree of

Master Of Computer Applications
UNDER
APJ Abdul Kalam Technological University

BY

Yusra Yasar M
Register Number: MES21MCA-2060

DEPARTMENT OF COMPUTER APPLICATIONS
MES COLLEGE OF ENGINEERING
KUTTIPPURAM, MALAPPURAM - 679 582
May 2023

Declaration

I undersigned hereby declare that the main project report **Crop Pest Classification Model Using Deep Learning Techniques** submitted in partial fulfillment of the requirements for the award of *Degree of Master of Computer Applications* under *APJ Abdul Kalam Technological University* is a bona fide record done by me under the supervision of **Nowshad C V**. This submission represents my ideas in my own words and where ideas or words of others have been included, I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to the ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in my submission.I understand that any violation of the above will be a cause for disciplinary action by the institute and / or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other University.

Sincerely,
Yusra Yasar M
MES21MCA-2060

Place: Kuttippuram

Date:

MES COLLEGE OF ENGINEERING

KUTTIPPURAM, KERALA - 679 582

(A NAAC ACCREDITED INSTITUTION WITH NBA ACCREDITED DEPARTMENTS,
APPROVED BY AICTE AND AFFILIATED TO APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY)

DEPARTMENT OF COMPUTER APPLICATIONS

CERTIFICATE

This is to certify that the main project report entitled **Crop Pest Classification Model Using Deep Learning Techniques** is a bona fide record of the main project work carried out by **Yusra Yasar M (Register No: MES21MCA-2060)**, fourth semester student of the department, during the academic year 2022-23, in partial fulfillment of the requirements for the award of *Degree of Master of Computer Applications* under *APJ Abdul Kalam Technological University*. This report in any form has not been submitted to any other University or Institution for any purpose.

Internal Supervisor(s) External Supervisor(s)

External Examiner Head of the department

Acknowledgement

My endeavor stands incomplete without dedicating my gratitude to a few people who have contributed towards the successful completion of my main project. I pay my gratitude to the Almighty for His invisible help and blessing for the fulfillment of this work . At the outset I express my heart full thanks to our **Head of the Department**, **Prof.Hyderali K** for permitting me to do this main project. I would like to express my sincere gratitude to our project coordinator, **Mr.Vasudevan T V**,Assistant Professor for his exceptional support and encouragement throughout this project. I take this opportunity to express my profound gratitude to **Mr.Nowshad CV**, my **guide**, for his valuable support and help in my main project. I am also grateful to all our teaching and non teaching staff for their encouragement, guidance and whole-hearted support. Last but not least, I am gratefully indebted to my family and friends, who gave me their precious help in my main project.

Sincerely,
Yusra Yasar M
MES21MCA-20260

Synopsis

Crop pest classification is a crucial aspect of modern agriculture and plays a significant role in ensuring sustainable food production. The objective of this field is to identify different pests that threaten crops and to categorize them based on their characteristics, behavior, and impact. This information can then be used to develop effective pest management strategies and to minimize the use of harmful chemicals in agriculture.

The classification of crop pests typically involves the use of morphological, behavioral, and molecular methods to accurately identify and categorize different species. With advances in technology, machine learning and computer vision algorithms are increasingly being used for crop pest classification, providing a more efficient and accurate means of identifying and managing pests. Ultimately, the goal of crop pest classification is to help ensure the long-term sustainability of food production and to protect the environment from the harmful effects of pesticide use.

Table Of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Crop pest classification is a critical component of modern agriculture, as it helps in identifying and managing the various pests that threaten crop production. Traditional methods of pest classification, such as morphological and behavioral observations, have limitations in terms of accuracy and efficiency. With the advancement of deep learning algorithms, there is now an opportunity to provide a more accurate and efficient approach to crop pest classification.Deep learning is a subset of machine learning that uses artificial neural networks to perform tasks that were once thought to require human intelligence, such as image recognition. In the context of crop pest classification, deep learning algorithms are trained using large datasets of pest images and features, allowing them to accurately recognize and classify different pests. This technology has the potential to overcome the limitations of traditional methods and provide more robust and reliable pest classification results.

The introduction of deep learning for crop pest classification is an exciting development in the field of agriculture and has the potential to revolutionize the way we approach pest management. In this project, the potential of deep learning for crop pest classification and its importance in ensuring the long-term sustainability of food production and the protection of the environment from harmful pesticide use

1.1 Background

Crop pest classification based on deep learning is highly relevant in modern agriculture, as it provides a powerful tool for farmers and agricultural experts. Crop pest classification is relevant for a number of reasons, and they are: Minimization of pesticide use: Pesticides are commonly used to control pests, but their use can also have harmful effects on the environment and human health. Accurate pest classification can help to minimize the use of pesticides by providing more targeted and effective pest management strategies Economic benefits: Pests can cause significant financial losses to farmers and the agricultural industry as a whole. Accurate pest classification and effective pest management strategies can help to reduce these losses and contribute to the economic stability of the agricultural industry.

1.2 Objective

The objective of pest classification based on deep learning is to develop an accurate and efficient system that can quickly classify various pests . The primary objective of crop pest classification is to accurately identify and classify different pests that affect crops. This allows for effective pest management strategies to be developed and implemented, which can minimize the impact of pests on crop production. Pesticides are commonly used to control pests, but their use can also have harmful effects on the environment and human health. Accurate pest classification can help to minimize the use of pesticides by providing more targeted and effective pest management strategies The ultimate goal is to improve crop yields and reduce economic losses by providing an effective tool for precision agriculture management, and ensuring sustainable and environmentally friendly agriculture.
.

1.3 Contribution

In crop pest classification is to accurately and efficiently identify different pests that affect crops and classify them into different categories. Traditional methods of pest classification, such as morphological and behavioral observations, are time-consuming and have limitations in terms of accuracy. There is a need for a more robust and reliable method of crop pest classification that can

quickly and accurately identify different pests, allowing for effective pest management strategies to be developed and implemented. This will help to ensure the long-term sustainability of food production and minimize the impact of pests on crops

1.4 Report Organisation

The report on Crop Pest Classification Model Using Deep Learning Techniques is organized into several sections. The report starts with an introduction that outlines the background and motivation for the project, as well as the objectives and contributions of the research. This section also includes a brief overview of the report organization, highlighting the various sections and their contents.

The report then delves into the literature survey, which provides an in-depth analysis of the existing research and technology in the field of voice-based email for visually challenged individuals. The methodology section follows, detailing the various steps taken to implement the project, including the workflow, and implementation process. The report also includes a section on Agile methodology, which outlines the user stories, product backlog, project plan, sprint backlog, and product backlog review. Finally, the report concludes with a discussion of the testing and validation process, as well as an overview of the Git version control system used throughout the project. Overall, the report is well-structured and comprehensive, providing a detailed account of the project from start to finish.

Chapter 2

Literature Survey

The production rate of agricultural products is decreasing day by day due to many natural as well as human-generated causes. One of the major causes behind the decreased production rate is the pests' nuisance. So, specialists emphasize automatic pesticide classification to detect pests easily before crops are affected. Sometimes when farmers see a plague, they fertilize it without understanding whether it is helpful or detrimental and destroy it all. So, scientists are trying to establish models to classify the beneficial and harmful pests.

In 2015, Vijai et al. [1] developed a genetic algorithm for image segmentation techniques for automatic detection and classification of plant leaf disease. Ding et al. [2] presented a pipeline based on deep learning to automatically identify pests by the taken images from the field trip in 2016. They did not use any pest specific-engineering, so they can adopt any species with less human effort, which showed promising results. In 2017, Monzurul et al. [3] presented an integrated approach to image processing and machine learning to diagnose leaf disease from images. The segmentation approach and support vector machine was applied, showed disease classification over 300 images with an accuracy of 95

Edna et al in 2018, using a data set of 38 different classes with a healthy and diseased picture of 14 leaves established a disease detection model. DenseNet showed excellent performance where it took fewer parameters and time to achieve state-of-the-art performance, which beat the rest of the architectures by obtaining 99.75 accuracy.

Divyansh et al. used VGG-19 for fine-tuning to extract features from the potato leaf and applied a multi-classifier where logistic regression outperformed others by obtaining 97.8 accuracy. However, in the field of pest and disease detection, Convolutional Neural Network has made substantial progress

Chapter 3

Methodology

3.1 Introduction

The methodology for developing a crop pest classification model using deep learning techniques involves several key steps. Firstly, data collection is necessary to obtain a dataset of images of both healthy crops and those affected by pests. These images must then be preprocessed to resize, normalize, and augment the dataset, increasing its diversity and size. Next, an appropriate deep learning architecture is selected, and the model is trained on the preprocessed dataset. The model is then evaluated on a validation set, with performance metrics such as accuracy, precision, and recall used to optimize the model. Once the model has been evaluated and refined, it can be deployed to a web server or mobile device and used in the field. Maintenance of the model is also critical, as regular monitoring and retraining with new data is required to ensure continued optimal performance. Overall, the methodology for developing a crop pest classification model using deep learning techniques requires expertise in both agriculture and deep learning, careful attention to detail, and a willingness to experiment with different approaches and techniques to achieve the best possible performance.

In addition to the core steps outlined above, there are several other considerations that should be taken into account when developing a crop pest classification model using deep learning techniques.

Firstly, the selection of an appropriate dataset is crucial for the success of the model. The dataset should be large enough to ensure that the model can learn the

necessary features to accurately classify crops affected by pests. It should also be diverse enough to cover a wide range of pest types, crop species, and environmental conditions.

Secondly, careful attention must be paid to the training and optimization of the model. Techniques such as early stopping, regularization, and hyperparameter tuning should be used to prevent overfitting and improve generalization performance. Additionally, the choice of evaluation metrics should reflect the specific goals of the model, such as minimizing false positives or maximizing overall accuracy.

Thirdly, the deployment of the model should be user-friendly and accessible to farmers, who may not have a technical background. This may involve developing a simple and intuitive user interface or providing clear instructions on how to use the model.

Finally, ethical considerations must also be taken into account when developing a crop pest classification model. For example, privacy concerns must be addressed when collecting and processing images of crops, and the impact of the model on small-scale farmers must be carefully considered.

In summary, developing a crop pest classification model using deep learning techniques requires a rigorous methodology that takes into account data collection, preprocessing, model selection, training, evaluation, deployment, maintenance, and ethical considerations. By following best practices and leveraging the latest advances in deep learning, researchers can develop effective and practical solutions to help farmers protect their crops and increase their yields.

3.2 Workflow

Here is a typical workflow for building a crop pest classification model using deep learning techniques:
Data Collection: Identify the crop types and pest species that are important for your model. Collect a large dataset of crop images with pests from different sources such as online repositories, field surveys, or crowdsourcing platforms. Organize the dataset into train, validation, and test subsets.
Data Preprocessing: Resize the images to a fixed size to ensure consistency in

the input dimensions of the model. Normalize the pixel values to have zero mean and unit variance, which helps the model to converge faster during training.

Data Augmentation: Use techniques such as rotation, flipping, and cropping to create additional training samples. Apply random transformations such as brightness and contrast adjustments to increase the dataset diversity.

Model Selection: Choose an appropriate deep learning model such as a CNN for image classification. Consider using pre-trained models such as VGG, ResNet, or Inception, which have been trained on large-scale image datasets like ImageNet.

Model Training: Initialize the model parameters randomly or with pre-trained weights. Feed the model with input images and their corresponding labels. Use an optimization algorithm such as Stochastic Gradient Descent (SGD) to update the model parameters to minimize the loss. Monitor the model performance on a validation dataset and adjust the hyperparameters accordingly.

Model Evaluation: Evaluate the trained model on a separate test dataset to assess its performance on unseen data. Compute metrics such as accuracy, precision, recall, and F1-score to measure the model performance.

Model Deployment: Deploy the trained model in a production environment such as a web application, mobile app, or embedded system. Integrate the model with the data pipeline and user interface for crop pest classification. Monitor the model performance and retrain the model periodically with new data to maintain its accuracy and reliability.

3.3 Data Flow Diagram

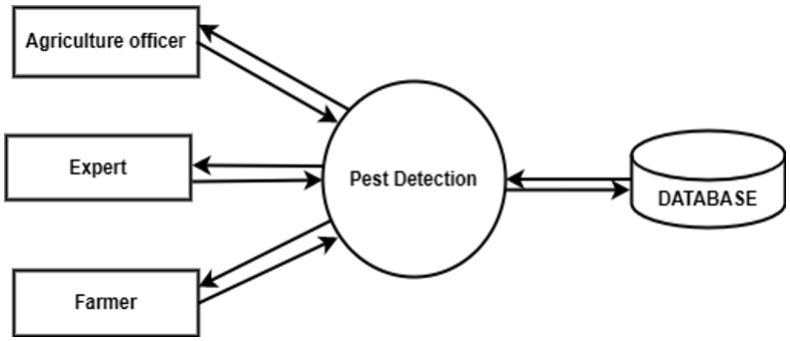

Figure 3.1: Level 0

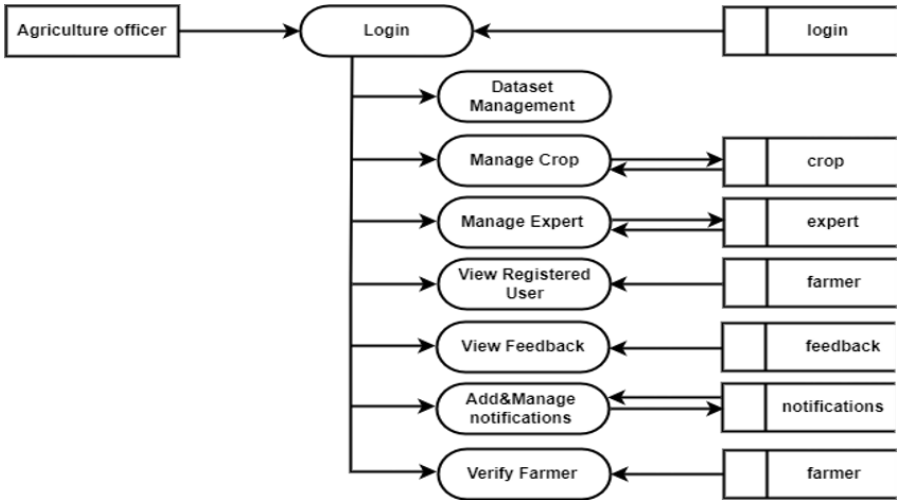

Figure 3.2: Level 1

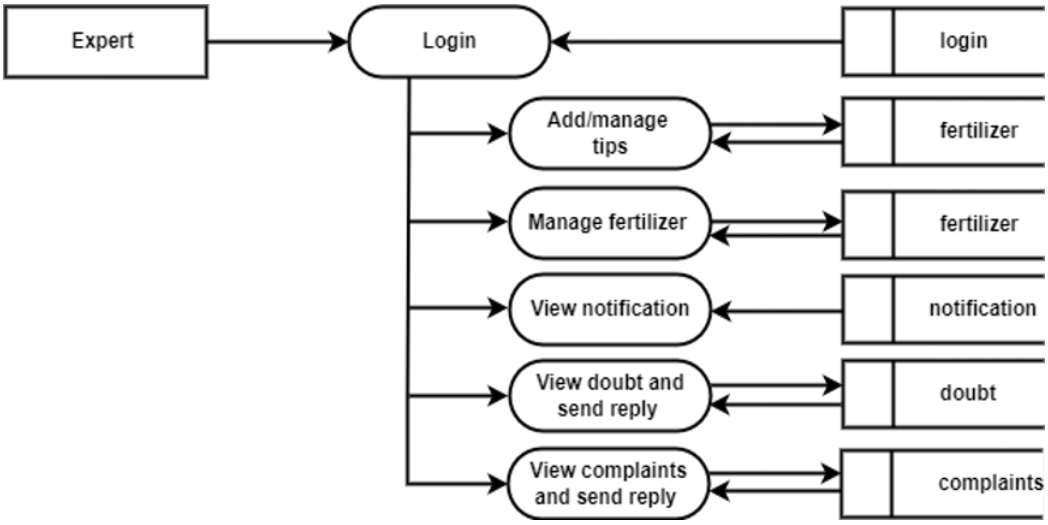

Figure 3.3: Level 1

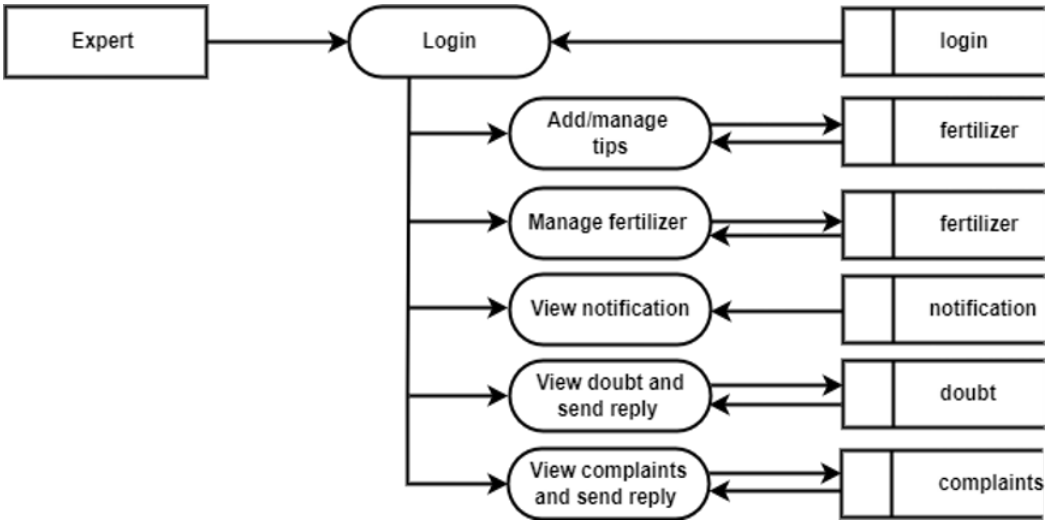

Figure 3.4: Level 1

3.4 Implementation

The implementation of the crop pest classification model using deep learning techniques involves several steps. Firstly, a large dataset of pest images and features must be collected and labeled. This dataset will be used to train the deep learning model. Preprocessing steps such as image resizing, normalization, and augmentation may also be applied to the dataset to improve the performance of the model.

Next, a deep learning architecture such as a convolutional neural network (CNN) can be used to build the model. The model is trained using the dataset, with the objective of minimizing the loss function and maximizing the accuracy of the model's predictions. The model is then evaluated using a separate testing dataset to determine its performance.

Once the model has been trained and evaluated, it can be deployed as a web application or a mobile app for use by farmers and agricultural experts. The application will allow users to upload images of pests and receive predictions on the type of pest present. The application may also provide recommendations for pest management strategies based on the type of pest identified.

To ensure the accuracy and effectiveness of the model, regular updates and improvements should be made based on feedback from users and new data collected in the field. This will help to ensure that the model remains up-to-date and relevant to the changing conditions of agriculture. Overall, the implementation of a crop pest classification model using deep learning techniques has the potential to revolutionize the way we approach pest management in agriculture, leading to more sustainable and environmentally friendly practices.

Chapter 4

Agile Methodology

4.1 Introduction

Agile methodology is a popular approach to software development that is well-suited for the development of a voice-based email system for visually impaired people. Agile methodology is an iterative and incremental approach to software development that focuses on delivering working software quickly and continuously improving it over time. It is based on the principles of collaboration, flexibility, and rapid feedback. Agile methodology is particularly useful for software development projects that involve a high level of complexity and uncertainty, such as the development of the voice-based email system.

The agile methodology consists of several key practices that are designed to promote collaboration, rapid feedback, and continuous improvement. The first key practice is iterative development, which involves breaking the development process into small, manageable pieces or iterations. Each iteration involves a short period of time, typically one to four weeks, during which the development team works to complete a specific set of tasks or deliverables. At the end of each iteration, the team reviews the work completed and adjusts the development plan as necessary.
The second key practice is continuous integration, which involves integrating the work of different developers into a single, unified code base on a regular basis. This helps to ensure that the code is always up-to-date and that any issues or conflicts are identified and resolved quickly. The third key practice is test-driven development, which involves writing automated tests for each new piece of code

before it is integrated into the code base. This helps to ensure that the code is of high quality and that any defects are identified and corrected quickly.
In the case of the voice-based email system, the agile methodology can be particularly useful in ensuring that the system is developed quickly, efficiently, and in a manner that is responsive to the needs of visually impaired users. The iterative development approach can be used to break the development process into smaller, more manageable pieces, such as developing individual features or functions of the system. Continuous integration can be used to ensure that the code is always up-to-date and that any issues or conflicts are identified and resolved quickly. Test-driven development can be used to ensure that the system is of high quality and that any defects are identified and corrected quickly.

4.2 User Story

A user story is a concise, high-level description of a feature or functionality of a product that is written from the perspective of the user or customer. It is used to capture the user's needs, goals, and desired outcomes, rather than specifying the technical details of how the feature will be implemented. User stories are a key component of agile software development methodologies, which emphasize collaboration and responding to change over following a rigid plan.
User stories typically follow a simple format, such as "As a [user], I want [goal], so that [benefit]." The user story describes a specific feature or functionality that the user needs in order to achieve a particular goal or outcome. By focusing on the user's needs and goals, user stories help ensure that development efforts are aligned with the user's needs and that the resulting product is more likely to be useful and valuable.
User stories are often written on index cards or sticky notes and can be easily prioritized and organized by the development team. They are used to drive the development process, helping to ensure that the team is building features and functionality that are aligned with the user's needs. As the product evolves and new requirements emerge, user stories can be added, updated, or removed, allowing the team to respond quickly and effectively to changing circumstances. Overall, user stories are an important tool for building better products that meet the needs and goals of the end users.

User Story ID	As a (type of user)	I want to	So that I can
1	Agriculture officer	Login	Access the System
2	Agriculture officer	Verify farmer	Verify farmer
3	Farmer	Registering	registering
4	Agriculture officer	Manage Dataset	Manage Dataset For pest prediction
5	Farmer	Login	Access the System
6	Farmer	Pest prediction	Pest prediction by dataset
7	Agriculture officer	Manage Crop	Add/ Delete Crop
8	Farmer	View Crop	View Crop
9	Agriculture officer	Manage Expert	Manage Expert
10	Expert	Login	Access the System
11	Agriculture officer	View farmer	View Registered farmer
12	Agculture oficer	View Feedback	View Feedback
	Farmer	Send feedback	Post feedback
14	Agriculture officer	Manage Notification	Add/delete Notification
15	Expert	View Notification	View Notification
16	Farmer	View Notification	View Notification
17	Expert	Manage Tips	Add or delete Tips
18	Farmer	View Tips	View Tips
19	Expert	Manage fertilizer	Add or delete fertilizer
20	Farmer	View fertilizer	View fertilizer
21	Expert	View doubts and reply	View doubts and reply
22	Farmer	Send doubts and reply	Send doubts and reply
23	Expert	View Complaints and reply	View Complaints and reply
24	Farmer	Send Complaints and reply	Send Complaints and reply

Table 4.1: User Story

4.3 Product Backlog

The product backlog is a prioritized list of requirements or features that need to be developed for a product. It is a key artifact in agile software development methodologies, such as Scrum, and serves as the single source of truth for all the work that needs to be done on the product.
The product backlog is typically owned by the product owner, who is responsible for prioritizing the items on the backlog based on the needs and goals of the stakeholders. The items on the backlog are typically user stories, which describe the features or functionality of the product from the perspective of the user.
The product backlog is dynamic and evolves over time as new information becomes available or as the product evolves. Items can be added, removed, or reprioritized based on feedback from stakeholders, changes in the market or technology, or other factors. The development team uses the product backlog to plan and execute their work, pulling items from the top of the backlog into each sprint or development cycle.

ID	**Name**	**Priority (High/Medium /Low)**	**Estimate (Hr)**	**Status (Planned/ Progressing/ Completed)**
1	Pest prediction	High	75	Completed
2	Manage Crops	High	30	Completed
3	Feedback	High	40	Completed
4	Manage Tips	High	40	Completed
5	Complaints and Doubt's	High	65	Completed
6	Notification	High	24	Completed
7	Manage Fertilizer	High	50	Completed

Table 4.2: Product Backlog

4.4 Project Plan

A project plan is a detailed document that outlines the objectives, tasks, timelines, resources, and potential risks associated with a specific project. It serves as a roadmap that guides the project team in executing and completing the project successfully within the defined constraints, such as budget, timeline, and scope.

A project plan is critical for ensuring that a project is completed efficiently, effectively, and on time. It helps project managers and team members stay focused on the project goals and objectives and provides a framework for managing tasks, timelines, and resources. A well-defined project plan also helps identify potential risks and challenges, allowing project teams to prepare and implement appropriate strategies to mitigate them. Ultimately, a project plan is essential for managing a project and ensuring that it meets or exceeds stakeholders' expectations

User Story ID	**Task Name**	**Start Date**	**End Date**	**Days**	**Status**
1,2,3	Sprint 1	01/02/2023	10/02/2023	10	Completed
4,5,6		11/02/2023	21/02/2023	11	Completed
7,8,9	Sprint 2	27/07/2023	08/03/2023	10	Completed
10,11,12		09/03/2023	20/03/2023	11	Completed
13,14,15	Sprint 3	22/03/2023	31/03/2023	10	Completed
16,17,18		01/04/2023	11/04/2023	11	Completed
19,20,21	Sprint 4	15/04/2023	24/04/2023	10	Completed
22,23,24		25/04/2023	05/05/2023	11	Completed

Table 4.3: Project Plan

4.5 Sprint Backlog (Plan)

The Sprint Backlog is a plan created by the Development Team at the beginning of each Sprint. It details the work that needs to be completed during the Sprint to meet the Sprint Goal. The Sprint Backlog is created during the Sprint Planning event, where the Development Team decides on the work that will be completed during the Sprint. The Product Owner presents the Product Backlog items that they want to be completed during the Sprint, and the Development Team determines how much of the work they can realistically complete during the Sprint.

Backlog item	Status and completion date	original Estimate (hours)	Day 1 07/02	Day 2 08/02	Day 3 09/02	Day 4 10/02	Day 5 13/02	Day 6 14/02	Day 7 15/02	Day 8 16/02	Day 9 17/02	Day 10 20/02	Day 11 21/02	Day 12 22/02	Day 13 23/02	Day 14 24/02	Day 15 27/02	Day 16 28/02	Day 17 01/03	Day 18 02/03	Day 19 03/03
UI designing	10/02/23	4	0	1	1	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Coding	21/02/23	8	0	0	0	0	1	2	0	2	0	2	1	0	0	0	0	0	0	0	0
Testing	03/02/23	6	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	2	2	0	0
Total		18	0	1	1	2	1	2	0	2	0	2	1	1	0	1	0	2	2	0	0

Table 4.4: Sprint Backlog(plan 1)

Backlog Item	Status and completion date	original Estimate (hours)	Day 1 06/03	Day 2 07/03	Day 3 08/02	Day 4 9/03	Day 5 10/03	Day 6 13/03	Day 7 14/03	Day 8 15/03	Day 9 16/03	Day 10 17/03	Day 11 20/03	Day 12 21/03	Day 13 22/03	Day 14 23/03
UI designing	11/03/23	4	0	0	0	0	0	0	0	0	0	2	1	1	0	0
Coding	17/03/23	12	0	3	0	0	1	2	3	0	1	2	0	0	0	0
Testing	23/03/23	4	0	0	0	0	0	0	0	0	0	0	1	1	0	2
Total		20	0	3	0	0	1	2	3	0	1	4	2	2	0	2

Table 4.5: Sprint Backlog(plan 2)

Backlog Item	Status and completion date	original Estimate (hours)	Day 1 24/03	Day 2 26/03	Day 3 28/03	Day 4 30/03	Day 5 03/04	Day 6 05/04	Day 7 06/04	Day 8 07/04	Day 9 08/04	Day 10 10/04	Day 11 11/04	Day 12 13/04	Day 13 14/04	Day 14 16/04
UI designing	26/03/23	4	0	0	2	2	0	0	0	0	0	0	0	0	0	0
Coding	17/04/23	16	2	2	2	2	2	2	2	0	2	0	0	0	0	0
Testing	16/04/23	8	0	0	0	0	1	1	0	0	1	0	0	2	1	2
Total		28	2	2	4	4	3	3	2	0	3	0	0	2	1	2

Table 4.6: Sprint Backlog(plan 3)

Backlog Item	Status and completion date	original Estimate (hours)	Day 1 18/04	Day 2 19/04	Day 3 22/04	Day 4 24/04	Day 5 26/04	Day 6 27/04	Day 7 28/04	Day 8 02/05	Day 9 04/04	Day 10 06/05	Day 11 09/05	Day 12 11/05	Day 13 13/05	Day 14 15/05
UI designing	24/04/23	8	0	2	0	2	0	2	0	2	0	0	0	0	0	0
Coding	10/05/23	18	2	2	2	2	2	2	2	2	0	2	0	0	0	0
Testing	15/05/23	8	0	0	1	0	0	1	0	1	0	0	2	0	1	2
Total		34	2	4	3	4	2	5	2	5	0	2	2	0	1	2

Table 4.7: Sprint Backlog(plan 4)

4.6 Sprint Backlog (Actual)

The Sprint Backlog is a subset of the Product Backlog, consisting of items that the Development Team has committed to completing during a Sprint. It is created during the Sprint Planning meeting, where the Development Team selects the highest priority Product Backlog items that they can complete within the upcoming Sprint. The Sprint Backlog is a plan for how the Development Team will accomplish the selected Product Backlog items and achieve the Sprint Goal..

Backlog item	Status and completion date	original Estimate (hours)	Day 1 07/02	Day 2 08/02	Day 3 09/02	Day 4 10/02	Day 5 13/02	Day 6 14/02	Day 7 15/02	Day 8 16/02	Day 9 17/02	Day 10 20/02	Day 11 21/02	Day 12 22/02	Day 13 23/02	Day 14 24/02	Day 15 27/02	Day 16 28/02	Day 17 01/03	Day 18 02/03	Day 19 03/03
UI designing	10/02/23	4	0	1	1	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Coding	21/02/23	8	0	0	0	0	1	2	0	2	0	2	1	0	0	0	0	0	0	0	0
Testing	03/02/23	6	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	2	2	0	0
Total		18	0	1	1	2	1	2	0	2	0	2	1	1	0	1	0	2	2	0	0

Table 4.8: Sprint Backlog(actual 1)

Backlog Item	Status and completion date	original Estimate (hours)	Day 1 06/03	Day 2 07/03	Day 3 08/02	Day 4 9/03	Day 5 10/03	Day 6 13/03	Day 7 14/03	Day 8 15/03	Day 9 16/03	Day 10 17/03	Day 11 20/03	Day 12 21/03	Day 13 22/03	Day 14 23/03
UI designing	11/03/23	4	0	0	0	0	0	0	0	0	0	2	1	1	0	0
Coding	17/03/23	12	0	3	0	0	1	2	3	0	1	2	0	0	0	0
Testing	23/03/23	4	0	0	0	0	0	0	0	0	0	0	1	1	0	2
Total		20	0	3	0	0	1	2	3	0	1	4	2	2	0	2

Table 4.9: Sprint Backlog(actual 2)

Backlog Item	Status and completion date	original Estimate (hours)	Day 1 24/03	Day 2 26/03	Day 3 28/03	Day 4 30/03	Day 5 03/04	Day 6 05/04	Day 7 06/04	Day 8 07/04	Day 9 08/04	Day 10 10/04	Day 11 11/04	Day 12 13/04	Day 13 14/04	Day 14 16/04
UI designing	26/03/23	4	0	0	2	2	0	0	0	0	0	0	0	0	0	0
Coding	17/04/23	16	2	2	2	2	2	2	2	0	2	0	0	0	0	0
Testing	16/04/23	8	0	0	0	0	1	1	0	0	1	0	0	2	1	2
Total		28	2	2	4	4	3	3	2	0	3	0	0	2	1	2

Table 4.10: Sprint Backlog(actual 3)

Backlog Item	Status and completion date	original Estimate (hours)	Day 1 18/04	Day 2 19/04	Day 3 22/04	Day 4 24/04	Day 5 26/04	Day 6 27/04	Day 7 28/04	Day 8 02/05	Day 9 04/04	Day 10 06/05	Day 11 09/05	Day 12 11/05	Day 13 13/05	Day 14 15/05
UI designing	24/04/23	8	0	2	0	2	0	2	0	2	0	0	0	0	0	0
Coding	10/05/23	18	2	2	2	2	2	2	2	2	0	2	0	0	0	0
Testing	15/05/23	8	0	0	1	0	0	1	0	1	0	0	2	0	1	2
Total		34	2	4	3	4	2	5	2	5	0	2	2	0	1	2

Table 4.11: Sprint Backlog(actual 4)

4.7 Product Backlog Review

Product Backlog Review is an essential ceremony in Agile software development, where the Product Owner and the Development Team collaborate to review the product backlog items. The primary goal of the Product Backlog Review is to ensure that the product backlog is up-to-date, prioritize the items, and provide a shared understanding of the product backlog among the team. The Product backlog review is given below.

REVIEW FORM

SPRINT 1

User Story ID	**Function which done**	**Comments from Scrum master and Product Owner**
1	Login	Satisfying result
2	Verify farmer	Satisfying result
3	Registering	Satisfying result
4	Manage Dataset	Satisfying result
5	Login	Satisfying result
6	Pest prediction	Satisfying result

Table 4.12: Product Backlog Review - Sprint 1

REVIEW FORM

SPRINT 2

User Story ID	Function which done	Comments from Scrum master and Product Owner
7	Manage Crop	Satisfying result
8	View Crop	Satisfying result
9	Manage Expert	Satisfying result
10	Login	Satisfying result
11	View farmer	Satisfying result
12	View feedback	Satisfying result

Table 4.13: Product Backlog Review - Sprint 2

REVIEW FORM

SPRINT 3

User Story ID	Function which done	Comments from Scrum master and Product Owner
13	Send feedback	Satisfying result
14	Manage Notification	Satisfying result
15	View Notification	Satisfying result
16	View Notification	Satisfying result
17	Manage Tips	Satisfying result
18	View Tips	Satisfying result

Table 4.14: Product Backlog Review - Sprint 3

REVIEW FORM

SPRINT 4

User Story ID	**Function which done**	**Comments from Scrum master and Product Owner**
, 19	Manage fertilizer	Satisfying result
20	View fertilizer	Satisfying result
21	View doubts and reply	Satisfying result
22	Send doubts and reply	Satisfying result
23	View Complaints and reply	Satisfying result
24	Send Complaints and reply	Satisfying result

Table 4.15: Product Backlog Review - Sprint 4

4.8 Sprint Review

Sprint Review is a meeting that takes place at the end of each sprint in an Agile Scrum project. The purpose of the Sprint Review is to inspect the Increment produced during the sprint, gather feedback, and adapt the Product Backlog if needed. The Sprint Review is attended by the Scrum Team, stakeholders, and customers, who review the Increment and provide feedback. Detailed sprint review is given below.

REVIEW FORM

SPRINT 1

User Story ID	Function which done	Comments from Scrum master and Product Owner
1	Login	Satisfying result
2	Verify farmer	Satisfying result
3	Registering	Satisfying result
4	Manage Dataset	Satisfying result
5	Login	Satisfying result
6	Pest prediction	Satisfying result

Table 4.16: Sprint Review - Sprint 1

REVIEW FORM

SPRINT 2

User Story ID	Function which done	Comments from Scrum master and Product Owner
7	Manage Crop	Satisfying result
8	View Crop	Satisfying result
9	Manage Expert	Satisfying result
10	Login	Satisfying result
11	View farmer	Satisfying result
12	View feedback	Satisfying result

Table 4.17: Sprint Review - Sprint 2

REVIEW FORM

SPRINT 3

User Story ID	Function which done	Comments from Scrum master and Product Owner
13	Send feedback	Satisfying result
14	Manage Notification	Satisfying result
15	View Notification	Satisfying result
16	View Notification	Satisfying result
17	Manage Tips	Satisfying result
18	View Tips	Satisfying result

Table 4.18: Sprint Review - Sprint 3

REVIEW FORM

SPRINT 4

User Story ID	Function which done	Comments from Scrum master and Product Owner
19	Manage fertilizer	Satisfying result
20	View fertilizer	Satisfying result
21	View doubts and reply	Satisfying result
22	Send doubts and reply	Satisfying result
23	View Complaints and reply	Satisfying result
24	Send Complaints and reply	Satisfying result

Table 4.19: Sprint Review - Sprint 4

4.9 Testing and Validation

Testing and validation are essential parts of any software development process. In the case of the voice-based email system for visually challenged people, testing and validation are crucial to ensure that the system is working as intended and meeting the requirements of the end-users.The testing process should begin with unit testing, where each module of the system is tested independently to ensure that it performs its functions correctly. The next step is integration testing, where the different modules are combined and tested as a single system. This testing process helps to identify any issues that may arise when the modules interact with each other.

Once the integration testing is complete, the system should undergo system testing to ensure that it meets all the functional and non-functional requirements. This testing process should also include testing the system's accessibility features to ensure that visually challenged people can use the system without difficulty.

Test#	**Date**	**Action**	**Result**	**Pass ? (Yes/No)**
1	21/02/23	Login	Access the System	Yes
2	24/02/23	Verify farmer	Verify farmer by officer	Yes
3	27/0223	Registering	Registering off all users	Yes
4	29/02/23	Manage Dataset	Manage Dataset For pest prediction	Yes
5	01/03/23	Login	Access the System	Yes
6	03/03/23	Pest prediction	Pest prediction by dataset	Yes

Table 4.20: Test Sprint 1

Test#	Date	Action	Result	Pass ? (Yes/No)
7	17/03/23	Manage Crop	Add/ Delete Crop	Yes
8	18/03/23	View Crop	View Crop by user	Yes
9	20/03/23	Manage Expert	Manage Expert by Officer	Yes
10	21/03/23	Login	Access the System to User	Yes
11	22/03/23	View farmer	View Registered farmer officer	Yes
12	23/03/23	View Feedback	View Feedback for officer	Yes

Table 4.21: Test Sprint 2

Test#	Date	Action	Result	Pass ? (Yes/No)
13	07/04/23	Send feedback	Post feedback by user	Yes
14	10/04/23	Manage Notification	Add/delete Notification officer	Yes
15	11/04/23	View Notification	View Notification to expert	Yes
16	13/04/23	View Notification	View Notification to farmer	Yes
17	14/04/23	Manage Tips	Add or delete Tips by expert	Yes
18	16/04/23	View Tips	View Tips user	Yes

Table 4.22: Test Sprint 3

Test#	Date	Action	Result	Pass ? (Yes/No)
19	10/05/23	Manage fertilizer	Add or delete fertilizer expert	Yes
20	11/05/23	View fertilizer	View fertilizer to user	Yes
21	12/05/23	View doubts and reply	View doubts and reply expert	Yes
22	13/05/23	Send doubts and reply	Send doubts and reply user	Yes
23	14/05/23	View Complaints and reply	View Complaints and reply expert	Yes
24	15/05/23	Send Complaints and reply	Send Complaints and reply user	Yes

Table 4.23: Test Sprint 4

4.10 Git

Git is a version control system used to track and manage changes in software code. In the context of the voice-based email system for visually impaired individuals, Git can be used to manage the project's source code and track changes made during the development process.

When working on the project, developers can use Git to create a local repository on their machine to store the project's code. They can make changes to the code, and then use Git to stage and commit those changes. Each commit represents a snapshot of the code at a specific point in time.

Once the developers have made several changes and commits, they can push their changes to a remote repository hosted on a platform like GitHub. This allows other team members to access the latest code changes and collaborate on the project. With Git, it's easy to track the history of the project, identify changes made by different team members, and revert to previous versions if necessary.

Chapter 5

Conclusion

Crop pest classification is a critical aspect of modern agriculture, and the use of deep learning techniques offers a more efficient and accurate means of identifying and managing pests. By accurately identifying and categorizing pests, effective pest management strategies can be developed and implemented, which can minimize the impact of pests on crop production and reduce the use of harmful pesticides. The implementation of a crop pest classification system based on deep learning requires the collection of data, the development of a deep learning model architecture, preprocessing of the data, training, and evaluation of the model, and finally, deployment of the system. The proposed software utility offers a user-friendly interface for agriculture officers, experts, and farmers to manage the classification of crop pests, providing a valuable tool for precision agriculture management and sustainable food production. Overall, the development and implementation of crop pest classification based on deep learning techniques have the potential to revolutionize the way we approach pest management and ensure the long-term sustainability of food production.

References

[1] V. Singh, V. and P. A. K. Misra *"Detection of unhealthy region of plant leaves using Image Processing and Genetic Algorithm," in 2015 International Conference on Advances in Computer Engineering and Applications (ICACEA), Ghaziabad, India, 2015*

[2] W. Ding and G. Taylor *"Automatic moth detection from trap images for pest management", Computers and Electronics in Agriculture, vol. 123, pp. 17-28, 2016. Available: 10.1016/j.compag.2016.02.003*.

[3] M. Islam, A. Dinh, K. Wahid and P. Bhowmik *"Detection of Potato Diseases Using Image Segmentation and Multiclass Support,Vector Machine," in 2017 IEEE 30th Canadian Conference of Electrical and Computer Engineering (CCECE), Saskatoon,Canada,2017*.

Appendix A

Source Code

```
# coding: utf-8

# In[ ]:
import os

import tensorflow as tf

import keras
from keras.engine.saving import load_model
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, AveragePooling2D
from keras.layers import Dense, Activation, Dropout, Flatten

from keras.preprocessing import image
from keras.preprocessing.image import ImageDataGenerator

import numpy as np

#------------------------------
# sess = tf.Session()
# keras.backend.set_session(sess)
#------------------------------
#variables
num_classes =9
batch_size = 40
```

```
epochs = 30
#-----------------------------

import os, cv2, keras
import numpy as np
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten
from keras.layers import Conv2D, MaxPooling2D
from keras.engine.saving import load_model
# manipulate with numpy, load with panda
import numpy as np
# import pandas as pd

# data visualization
import cv2

# import seaborn as sns

# get_ipython().run_line_magic('matplotlib', 'inline')


# Data Import
def read_dataset(path):
    data_list = []
    label_list = []
    i=-1
    my_list = os.listdir(r'C:\Users\hp\Downloads
\dataset\archive\pest\train')
    for pa in my_list:
        i=i+1
        print(pa,"==================")
        for root, dirs, files in os.walk(r'C:\Users
\hp\Downloads\dataset\archive\pest\train\\' + pa):

            for f in files:
                file_path = os.path.join(r'C:\Users\hp
\Downloads\dataset\archive\pest\train\\'+pa, f)
                img = cv2.imread(file_path, cv2.IMREAD_GRAYSCALE)
```

```
            res = cv2.resize(img, (48, 48),
 interpolation=cv2.INTER_CUBIC)
            data_list.append(res)
            # label = dirPath.split('/')[-1]
            label = i
            label_list.append(label)
            # label_list.remove("./training")
    return (np.asarray(data_list, dtype=np.float32),
 np.asarray(label_list))

def read_dataset1(path):
    data_list = []
    label_list = []

    file_path = os.path.join(path)
    img = cv2.imread
        (file_path, cv2.IMREAD_GRAYSCALE)
    res = cv2.resize(img, (48, 48),
        interpolation=cv2.INTER_CUBIC)
    data_list.append(res)
    # label = dirPath.split('/')[-1]

            # label_list.remove("./training")
    return (np.asarray(data_list, dtype=np.float32))

from sklearn.model_selection import train_test_split
# load dataset
x_dataset, y_dataset = read_dataset(r"C:\Users\hp
\Downloads\dataset\archive\pest\train")
X_train, X_test, y_train, y_test = train_test_split
(x_dataset, y_dataset, test_size=0.2, random_state=0)

y_train1=[]
for i in y_train:
    emotion = keras.utils.to_categorical(i, num_classes)
    print(i,emotion)
    y_train1.append(emotion)
```

```
y_train=y_train1
x_train = np.array(X_train, 'float32')
y_train = np.array(y_train, 'float32')
x_test = np.array(X_test, 'float32')
y_test = np.array(y_test, 'float32')

x_train /= 255  # normalize inputs between [0, 1]
x_test /= 255
print("x_train.shape",x_train.shape)
x_train = x_train.reshape(x_train.shape[0], 48, 48, 1)
x_train = x_train.astype('float32')
x_test = x_test.reshape(x_test.shape[0], 48, 48, 1)
x_test = x_test.astype('float32')

print(x_train.shape[0], 'train samples')
print(x_test.shape[0], 'test samples')
# ----------------------------
# construct CNN structure

model = Sequential()

# 1st convolution layer
model.add(Conv2D(64, (5, 5),
activation='relu', input_shape=
(48, 48, 1)))
model.add(MaxPooling2D(pool_size=
(5, 5), strides=(2, 2)))

# 2nd convolution layer
model.add(Conv2D(64, (3, 3),
activation='relu'))
model.add(Conv2D(64, (3, 3),
activation='relu'))
model.add(AveragePooling2D(pool_size=
(3, 3), strides=(2, 2)))

# 3rd convolution layer
model.add(Conv2D(128, (3, 3),
```

```
activation='relu'))
model.add(Conv2D(128, (3, 3),
activation='relu'))
model.add(AveragePooling2D(pool_size=
(3, 3), strides=(2, 2)))

model.add(Flatten())

# fully connected neural networks
model.add(Dense(1024, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(1024, activation='relu'))
model.add(Dropout(0.2))

model.add(Dense(num_classes, activation='softmax'))
# ------------------------------
# batch process

print(x_train.shape)

gen = ImageDataGenerator()
train_generator = gen.flow
(x_train, y_train, batch_size=batch_size)

# ------------------------------

model.compile(loss='categorical_crossentropy'
              , optimizer=keras.optimizers.Adam()
              , metrics=['accuracy']
              )

# ------------------------------

if not os.path.exists("model1.h5"):

    model.fit_generator(train_generator,
steps_per_epoch=batch_size, epochs=epochs)
    model.save("model1.h5")
```

```
# train for randomly selected one
else:
    model = load_model("model1.h5")  # load weights
from sklearn.metrics import confusion_matrix
yp=model.predict_classes(x_test,verbose=0)
cf=confusion_matrix(y_test,yp)
print(cf)
def predict(fn):
    dataset=read_dataset1(fn)
    (mnist_row, mnist_col, mnist_color) = 48, 48, 1

    dataset = dataset.reshape(dataset.shape[0],
 mnist_row, mnist_col, mnist_color)
    mo = load_model("model.h5")

    # predict probabilities for test set

    yhat_classes = mo.predict_classes(dataset, verbose=0)
    return yhat_classes
#
#     print(yhat_classes)

#predict(r"D:\vehicle classification with deep
learning\src\semi-semitrailer-truck-tractor-highway.jpg")
```

Appendix B

Output

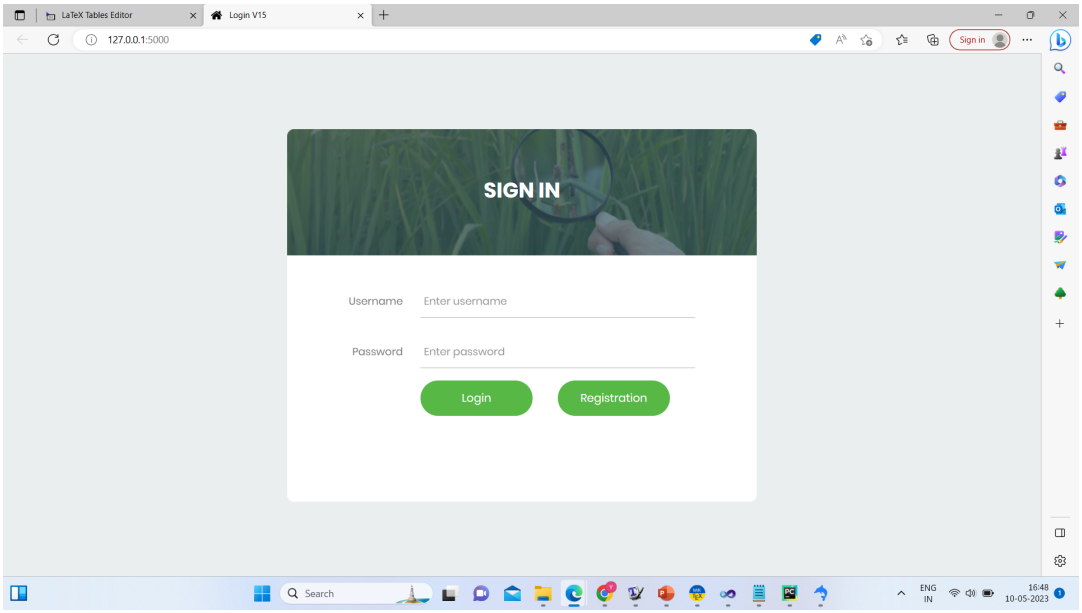

Figure B.1: Login

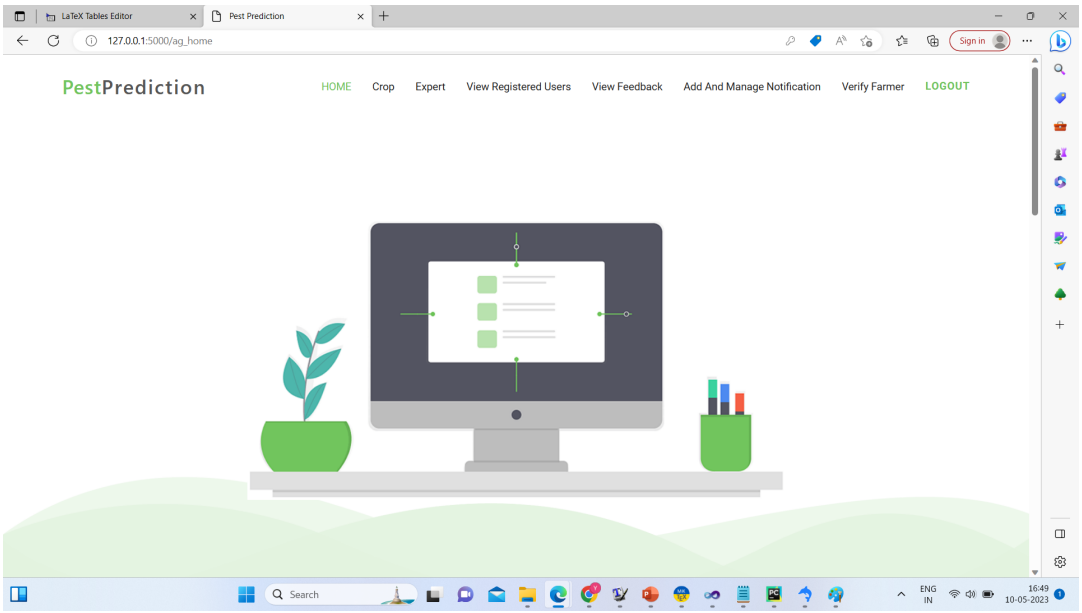

Figure B.2: Admin page

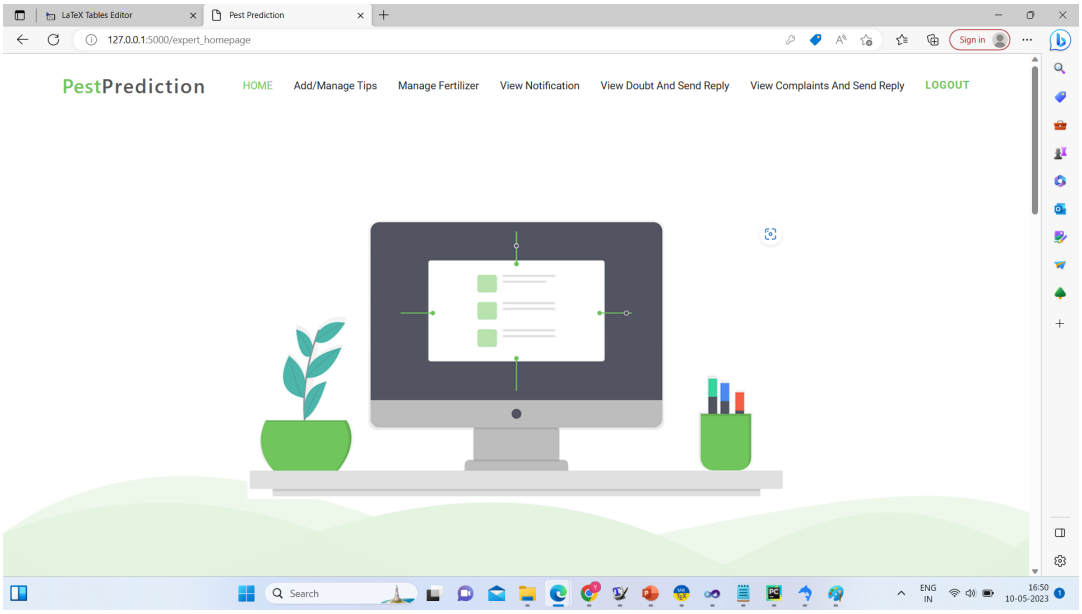

Figure B.3: Expert page

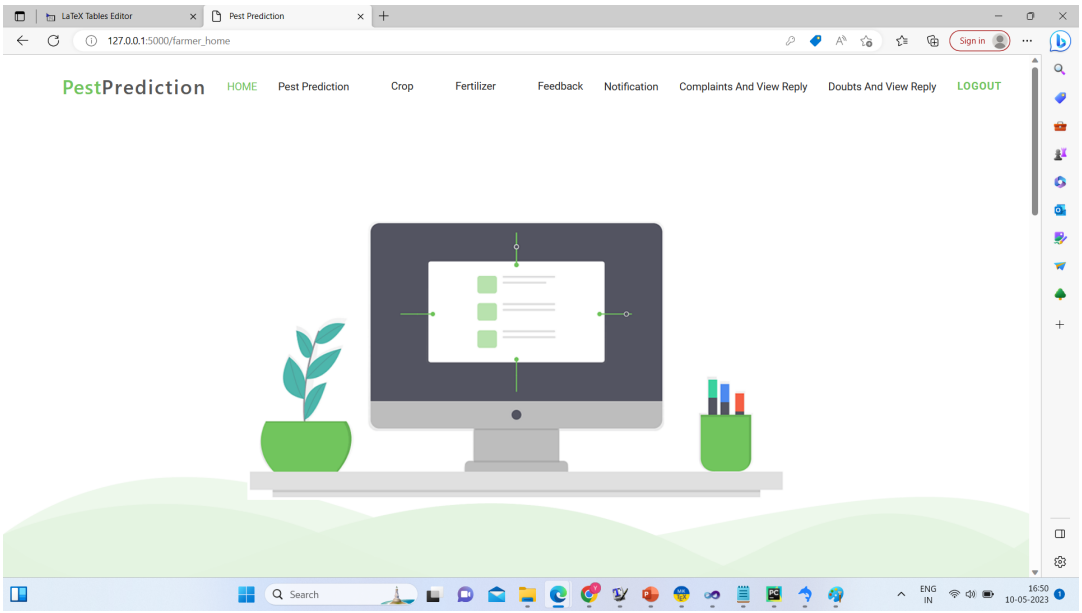

Figure B.4: User page

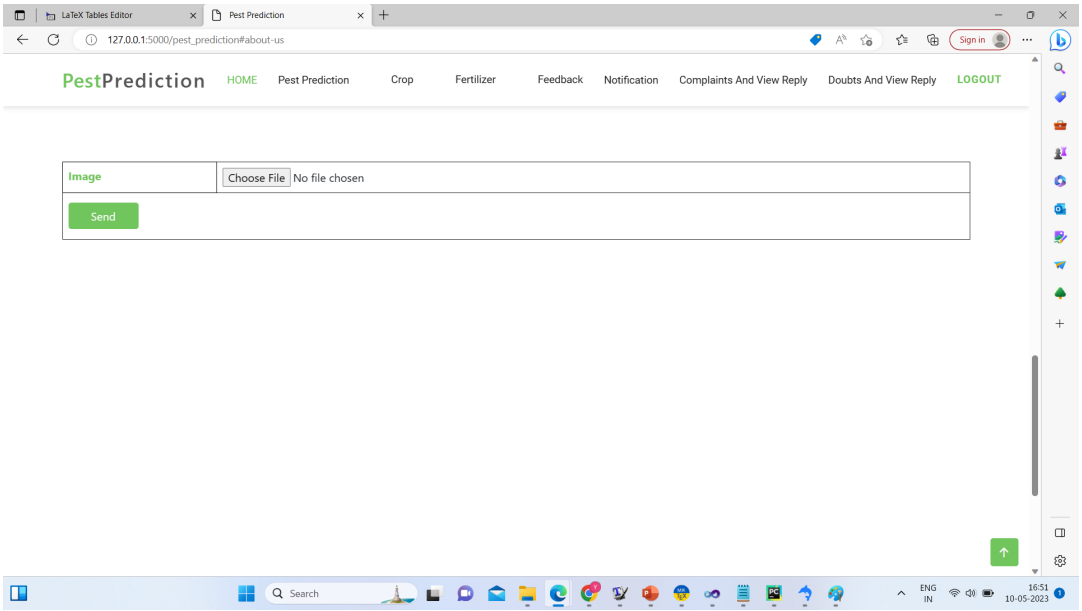

Figure B.5: Prediction page

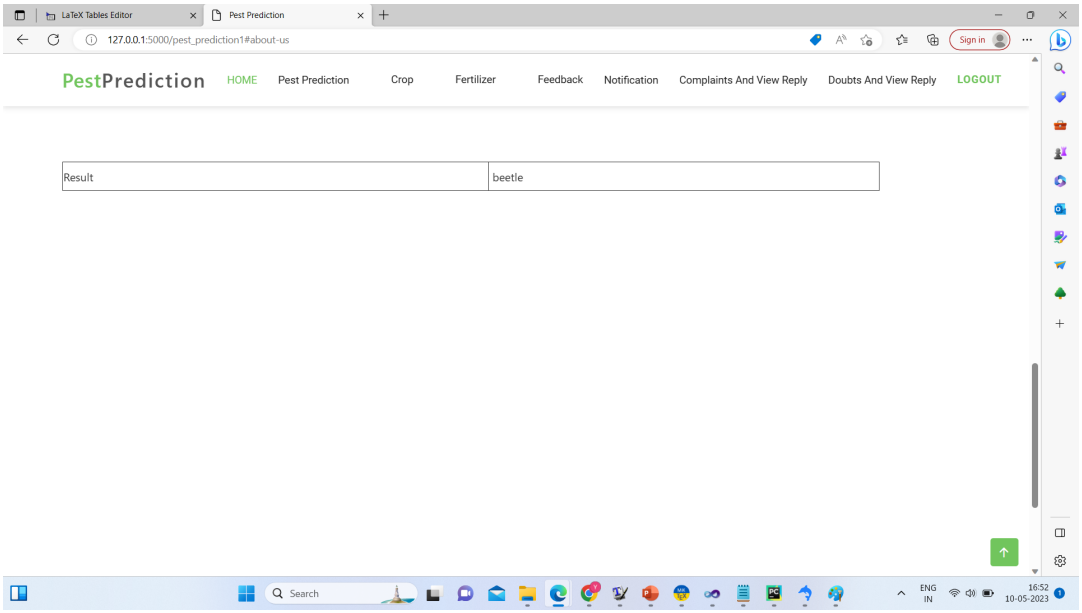

Figure B.6: Prediction result page

Appendix C

Git History

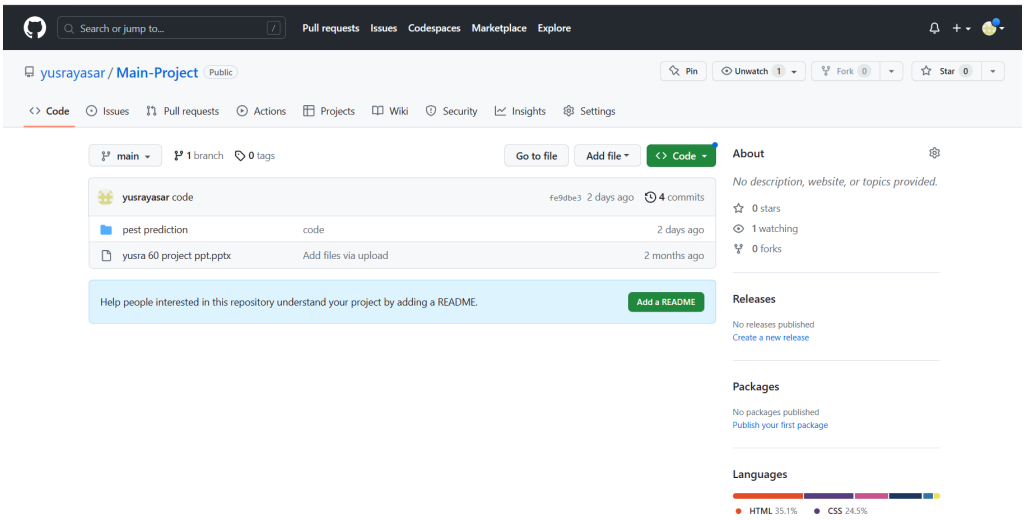

Figure C.1: Git Commits