TABLE OF CONTENTS

Contents	Page.No.
Certificate	i
Declaration	ii
Acknowledgement	iii
Abstract	iv
List of Tables	v
List of Figures	vi
List of Abbreviations	vii
CHAPTER 1: INTRODUCTION	4
1.1 Introduction	4
1.2 Problem Statement	8
1.2.1 Problem Definition	9
1.3 Objective	10
1.4 Scope	12
1.5 Infrastructure	13
1.6 Literature Survey	14
CHAPTER 2: REVIEW OF LITERATURE	17
2.1 Research Methodologies	17
2.2 Feasibility Studies	18
CHAPTER 3: PROPOSED SYSTEM	20
3.1 SDLC	20
3.2 Existing System	24
3.2.1 Disadvantages of Existing System	25
3.3 Proposed Methods	27

3.3.1 Module Description 28
3.4 System Architecture 30
3.4.1 Proposed System Advantages 31
3.5 Project Description 31
3.5.1 General 32
3.5.2 Modules Description 32
3.5.3 Given Input Expected Output 33
3.6 System Requirement Specification 34
3.6.1 Hardware Requirements 34
3.6.2 Software Requirements 34
3.6.3 Deliverables 40
CHAPTER 4: TEST STRATEGIES AND METHODOLOGIES 41
4.1 Test Strategies And Methods 41
4.2 Testing Strategy 44
4.3 Test Cases 45
CHAPTER 5: DESIGN AND IMPLEMENTATION 48
5.1 Unified Modelling Language 48
5.1.1 Model 48
5.1.2 Principles of Modelling 48
5.1.3 Applications of UML 49
5.1.4 Use Case Diagram 50
5.1.5 Class Diagram 51
5.1.6 Sequence Diagram 52
5.1.7 Activity Diagram 53
5.2 Sample Code 54
5.3 Screenshots 61
5.3.1 Home page 61
5.3.2 About Us 62
5.3.3 Fertilizer Recommendation 62
5.3.4 Plant Disease Prediction 63
CHAPTER 6: JUSTIFICATION AND DISCUSSION OF THE RESULTS 66
6.1 Theoretical Justification 66

6.2 Benefit Discussion of The Scheme 68

CHAPTER 7: CONCLUSION AND FUTURE ENHANCEMENT 71

7.1 Conclusion 71

7.2 Future Enhancement 72

7.3 References 73

CHAPTER 1: INTRODUCTION

1.1 Introduction

The Agronomisage Platform leverages the power of Machine Learning (ML) and Deep Learning (DL) to provide data-driven recommendations for farmers. By integrating advanced technologies, this platform offers a range of features aimed at optimizing agricultural practices and enhancing crop yield. Here is a detailed explanation of the key functionalities of the Agronomisage Platform:

- **Smart Farming Assistant:** The platform serves as a smart farming assistant, utilizing artificial intelligence (AI) to assist farmers in making informed decisions regarding their agricultural activities. By analyzing data and providing insights, it helps farmers optimize their farming practices for better outcomes.
- **AI-powered Crop and Fertilizer Recommendation System:** One of the core features of the platform is its AI-powered crop and fertilizer recommendation system. By inputting information about the location and soil conditions, the ML model within the platform suggests the ideal crop for optimal yield. This feature enables farmers to make data-driven decisions when selecting crops to cultivate and associate fertilizers to be used to upgrade the quality of yield.
- **Data-driven Disease Detection for Farms:** The platform incorporates a sophisticated DL model trained on a vast image database to detect potential diseases in crops. Farmers can upload images of their crops, and the system uses deep learning algorithms to analyze the images and predict any diseases present. This proactive approach helps farmers identify and address crop diseases early, minimizing losses.
- **Precision Agriculture Platform:** Agronomisage Platform functions as a precision agriculture platform, emphasizing the use of data analysis to optimize farm management practices. By leveraging ML and DL capabilities, the platform enables farmers to make precise decisions based on data insights, leading to improved efficiency and productivity in farming operations.

- **Agricultural Decision Support System (ADSS):** The platform acts as an agricultural decision support system, providing farmers with valuable recommendations and insights to enhance their decision-making processes. By integrating ML and DL technologies, the platform offers a comprehensive solution for farmers to manage their farms effectively and sustainably.

In summary, the Agronomisage Platform is a comprehensive tool that harnesses the capabilities of ML and DL to empower farmers with data-driven recommendations, crop suggestions, disease detection capabilities, and precision agriculture practices. By utilizing advanced technologies, this platform serves as a valuable resource for farmers seeking to optimize their farming operations and achieve better outcomes in crop cultivation.

THIS PROJECT CAN BE CATEGORIZED INTO TWO MAIN TYPES:

- **Machine Learning (ML):** Crop Recommendation: This functionality utilizes supervised learning, likely with an algorithm like Random Forest. The model is trained on a large agricultural dataset that considers factors like location, soil type, historical weather, and optimal crop yields.
- **Deep Learning (DL):** Disease Prediction: This feature uses image recognition and requires a Convolutional Neural Network (CNN). The CNN is trained on a massive dataset of labeled images containing healthy and diseased crops. By analyzing user-uploaded crop photos, the model predicts potential diseases.

CAPABILITIES

The Agronomisage Platform is a comprehensive agricultural technology solution that leverages the power of Machine Learning (ML) and Deep Learning (DL) to empower farmers with data-driven recommendations and insights. Here is a detailed explanation of its key functionalities:

- **Data-Driven Crop Recommendation:**
    - The platform's ML model analyzes various factors, such as the farmer's location, soil type, and historical weather data (if provided), to suggest the optimal crop for the specific conditions.
    - By considering these variables, the platform can recommend crops that are likely to yield the best results, maximizing the farmer's productivity and profitability.
- **Intelligent Fertilizer Recommendation:**
    - Based on the chosen crop and the analysis of the farmer's soil, the platform provides recommendations for the most suitable fertilizer type and quantity.
    - This feature ensures that the plants receive the necessary nutrients in the right proportions, leading to improved crop growth and yield.
- **Advanced Disease Detection:**
    - The platform's DL model, trained on a vast database of crop images, can identify potential diseases in the farmer's crops by analyzing uploaded photographs.
    - This early detection capability allows farmers to take proactive measures to address the issues, minimizing crop losses and maximizing the overall health of their plants.

The Agronomisage Platform is a powerful agricultural decision support system that leverages the latest advancements in ML and DL to provide farmers with data-driven recommendations and insights. By integrating these intelligent functionalities, the platform aims to empower farmers to make more informed decisions, optimize their farming practices, and achieve better crop yields and overall farm productivity.

OVERALL BENEFITS:

The Agronomisage Platform offers several key benefits to farmers by leveraging Machine Learning (ML) and Deep Learning (DL) technologies:

1. **Increased Yields:**
    - The platform's ML model analyzes factors like location, soil type, and historical weather data to suggest the optimal crop for specific conditions, maximizing yield potential.
    - By growing the right crops under the right conditions, farmers can optimize their production and achieve higher yields.

2. **Reduced Costs:**
    - Based on the chosen crop and soil analysis, the platform recommends the most suitable fertilizer type and quantity, ensuring efficient nutrient delivery to plants.
    - By using the appropriate fertilizers, farmers can save money and minimize waste.
    - The advanced DL model for disease detection allows for early intervention, reducing crop losses and the associated costs.

3. **Improved Farm Management:**
    - The platform provides data-driven insights and recommendations, empowering farmers to make informed decisions about their agricultural practices.
    - By leveraging the power of AI, farmers can optimize their decision-making processes and improve overall farm management.

4. **Enhanced Farm Productivity:**
    - The intelligent functionalities of the Agronomisage Platform, such as crop recommendation and disease detection, streamline farming processes and help farmers allocate resources more efficiently.
    - By optimizing resource allocation and minimizing waste, farmers can enhance their overall farm productivity and profitability.

1.2 Problem Statement

To address the issue of farmers struggling to decide on the best-suited crop for their land due to traditional and non-scientific methods, a proposed system aims to provide predictive insights on crop sustainability and recommendations based on machine learning models trained with essential environmental and economic parameters. This system leverages predictive analytics in agriculture, utilizing statistical and machine learning techniques to analyze current and historical data for predicting future events and conditions in farming. By integrating different technologies and data sources like weather patterns, soil quality, and historical machine learning algorithms, farmers can make informed decisions to increase crop yields, lower costs, and improve crop management.

The system's predictive modeling techniques include time series models, regression models, random forest models, recurrent neural networks, and physics-based neural networks to capture seasonal patterns, soil properties, fertilizer inputs, irrigation, and complex temporal dependencies in crop yield data. These models help in predicting crop sustainability and productivity by processing historical and up-to-date information on weather, soil conditions, and crop health with high classification accuracy, enabling farmers to make early and informed decisions regarding crop selection and management.

Moreover, the system addresses challenges such as upfront costs, change management, data quality, and the need for agricultural expertise by emphasizing the importance of robust data sets, participatory development, and the integration of predictive tools into existing farming practices through collaborative frameworks. By democratizing access to advanced analytics and focusing on inclusive data practices, the future of predictive analytics in agriculture holds promise for increasing production efficiency, sustainability, and resilience in the face of evolving food demand and climate variability.

1.2.1 Problem Definition

To address the limitations of current data and enhance precision farming through robust ML/DL models, the focus is on developing models based on high-quality, verified data. These advanced models aim to empower farmers by optimizing resource use, improving crop yields, and enhancing overall farm productivity while increasing resilience to environmental challenges.

- **Optimizing Resource Use:** By leveraging real-time data and specific field conditions, ML/DL models can enable farmers to make data-driven decisions regarding resource allocation. These models analyze a variety of factors such as soil quality, weather patterns, crop health, and historical data to provide insights on the optimal use of resources like water, fertilizers, and pesticides. This precision in resource management helps in reducing waste, lowering costs, and maximizing efficiency in farming operations.
- **Improving Crop Yields:** The development of reliable ML/DL models for precision farming allows for the prediction of crop yields with greater accuracy. By integrating data on soil conditions, weather forecasts, and crop health, these models can forecast potential yields, identify areas for improvement, and recommend strategies to enhance productivity. Farmers can adjust their practices, planting schedules, and management techniques based on these predictions to achieve higher yields and better quality crops.
- **Increasing Resilience to Environmental Challenges:** ML/DL models play a crucial role in enhancing the resilience of farming operations to environmental challenges such as climate change, pests, and diseases. By analyzing historical data and real-time information, these models can predict and mitigate risks associated with environmental factors. Farmers can proactively respond to changing conditions, implement preventive measures against pests and diseases, and adapt their practices to ensure sustainable farming practices even in the face of uncertainties.

In conclusion, the development of robust ML/DL models for precision farming based on high-quality, verified data offers significant benefits to farmers by enabling them to optimize resource use, improve crop yields, and increase resilience to environmental challenges. By harnessing the power of advanced analytics and data-driven insights, farmers can enhance their decision-making processes, drive efficiency in farming practices, and ensure sustainable agricultural production in the long run. challenges.

1.3 Objective

To address the limitations of current data and enhance precision farming through robust ML/DL models, the focus is on developing models based on large-scale, high-quality, and verified data. These advanced models aim to empower farmers by optimizing resource use, improving crop yields, increasing farm profitability, and enhancing environmental sustainability.

- **Optimizing Resource Use:** By leveraging high-quality data from sensors, imagery, and other sources, ML/DL models can analyze field-level conditions in detail. This allows the models to identify specific areas within a field that require precise amounts of water, fertilizers, and pesticides. The models can then provide recommendations to farmers on the optimal application of these resources, leading to reduced waste and more efficient use. For example, the models can detect areas with poor soil moisture and recommend targeted irrigation, or identify nutrient-deficient zones and suggest localized fertilizer application. This precision in resource management helps in lowering costs, minimizing environmental impact, and maximizing efficiency in farming operations.

- **Improving Crop Yields:** With access to large-scale, verified data on soil conditions, weather patterns, and historical crop performance, ML/DL models can forecast crop yields with a high degree of accuracy. These models can integrate data on factors such as soil composition, nutrient levels, pest infestations, and weather forecasts to predict potential yields and identify areas for improvement. Farmers can then adjust their planting schedules, management techniques, and pest control strategies based on these predictions to achieve higher yields and better quality crops. The models can also help in early detection of crop diseases and pests, enabling timely intervention and mitigation of potential losses.
- **Increasing Farm Profitability:** By optimizing resource use and improving crop yields, the ML/DL models can directly contribute to increased farm profitability. The cost savings from efficient resource utilization, combined with the higher yields and quality of crops, can lead to improved financial outcomes for farmers. Additionally, the models can provide insights into market trends, consumer preferences, and pricing strategies, allowing farmers to make more informed decisions about crop selection, timing of harvests, and pricing of their produce.
- **Enhancing Environmental Sustainability:** Precision farming practices enabled by ML/DL models can have a positive impact on the environment. By optimizing the use of water, fertilizers, and pesticides, the models can help reduce the environmental footprint of agricultural activities. This includes minimizing water waste, preventing excessive fertilizer runoff, and reducing the reliance on harmful chemicals. Furthermore, the models can identify sustainable farming practices, such as crop rotation and cover cropping, that can improve soil health and biodiversity, contributing to the overall environmental responsibility of the farming operations.

The development of robust ML/DL models for precision farming based on large-scale, high-quality, and verified data offers significant benefits to farmers and the agricultural sector as a whole. By harnessing the power of advanced analytics and data-driven insights, farmers can enhance their decision-making processes, drive efficiency in farming practices, and ensure sustainable agricultural production in the long run, ultimately leading to improved food security and environmental stewardship.

1.4 Scope

This website serves as a starting point for showcasing the potential of precision farming using machine learning (ML) and deep learning (DL) techniques. It focuses on educating users about the applications of these technologies in agriculture, rather than providing real-world decision-making capabilities due to data limitations.The website offers three main features:

- **Crop Recommendation System:** Using ML algorithms, the system recommends crops based on the user's location and soil data. By analyzing factors such as soil nutrient ratios, humidity, and temperature, the model suggests suitable crops for the given conditions. This feature aims to demonstrate how ML can assist farmers in making informed decisions about crop selection.
- **Fertilizer Suggestion System:** The fertilizer recommendation system utilizes ML to provide suggestions based on the user's soil data and the selected crop. By assessing the nutrient content of the soil and the specific requirements of the chosen crop, the model identifies nutrient deficiencies or excesses and recommends appropriate fertilizers. This feature showcases the potential of ML in optimizing fertilizer usage and improving soil health.
- **Disease Detection System:** The disease detection system employs simple deep learning techniques to predict crop diseases from uploaded leaf images. By training the model on a dataset of diseased and healthy plant leaves, the system can classify the uploaded image and provide information about the detected disease, its causes, and suggested prevention or treatment methods. This feature demonstrates the application of DL in early disease detection and prevention.

It is important to note that while these features showcase the potential of ML/DL in precision farming, the website's functionality is limited due to the use of custom-built datasets with potential limitations. The data may not be comprehensive or representative of real-world scenarios, and the models may not be suitable for making actual farming decisions. The primary focus of this website is on education and demonstrating the possibilities of ML/DL in agriculture. By showcasing these applications, the website aims to raise awareness about the potential of these technologies in improving crop yields, reducing resource waste, and enhancing overall farm productivity. However, for real-world decision-making, more robust and reliable models trained on large-scale, high-quality, and verified data would be necessary.

1.5 Infrastructure

This website will require a combination of front-end and back-end components:

Front-End: HTML, CSS, and JavaScript

User Interface (UI):

Back-End: Python and various machine learning libraries like scikit-learn, XGBoost, and TensorFlow

Server: Heroku

Database: Collected from kaggle and associated legal agricultural webpages.

Machine Learning (ML) Models: Decision Tree, Guassian Naive Bayes, Support Vector Machine (SVM), Logistic Regression, Random Forest, XGBoost.

Deep Learning (DL) Model: CNN ,Resnet 9, Torch,

API Integrations: Flask Api (Home, Crop recommendation, Fertilier recommendation , Disease Detection)

1.6 Literature Survey

1 **TITLE:** " PREDICTION OF CROP YIELD AND FERTILIZER RECOMMENDATION USING MACHINE LEARNING ALGORITHMS"

AUTHORS: Devdatta A. Bondre, Mr. Santosh Mahagaonkar

YEAR: 2019

DESCRIPTION:

The paper "PREDICTION OF CROP YIELD AND FERTILIZER RECOMMENDATION USING MACHINE LEARNING ALGORITHMS" presents a comprehensive system that combines IoT and machine learning to optimize crop production. The system uses sensors and IoT devices to collect data such as soil moisture, temperature, humidity, and nutrient levels. This data is then analyzed using machine learning algorithms to make informed decisions about which crops to plant, when to plant them, and how much fertilizer and water to apply.

KEY POINTS

Crop Yield Prediction: The system uses machine learning algorithms to predict crop yield based on soil parameters like nitrogen, phosphorus, pH of soil, and weather parameters like temperature and humidity.

Fertilizer Recommendation: The system recommends suitable fertilizers for the predicted crop based on fertilizer data, crop, and location.

IoT Integration: The system integrates IoT sensors and GPS modules to collect data and provide real-time information to farmers.

Machine Learning Algorithm: The system uses the KNN algorithm to classify data and recommend crops.

Accuracy: The system achieves an accuracy of 85% in crop recommendation.

Future Scope

Increasing Volume of Observation: The system can be further improved by increasing the volume of observation, i.e., soil test data.

AI Models: AI models can be used to get more accurate results.

CONCLUSION

The paper concludes that crop recommendation using IoT and machine learning has great potential to revolutionize the agricultural industry by optimizing resource utilization, reducing input costs, and increasing crop yields while maintaining sustainability.

2 **TITLE:** "Crop Recommendation System using Machine Learning Techniques for Sustainable Agriculture"

AUTHORS: PRABU AND A. RAJESH

YEAR:2024

DESCRIPTION: The research paper proposes a crop recommendation system that focuses on sustainable agriculture practices, utilizing machine learning algorithms to suggest crops that require fewer resources and have a lower environmental impact.

KEY HIGHLIGHTS:

Sustainable Crop Recommendation:

- The system aims to promote sustainable agriculture by recommending crops that are optimized for resource efficiency and environmental impact.
- This is in contrast to traditional crop recommendation systems that may focus solely on maximizing yield without considering sustainability factors.

Machine Learning Algorithms:

- The system employs neural networks and genetic algorithms to analyze various parameters and make crop recommendations.
- Neural networks are used to model the complex relationships between crop characteristics, environmental factors, and resource requirements.
- Genetic algorithms are utilized to optimize the crop selection process, exploring a wide range of possibilities to identify the most sustainable options.

Factors Considered:

- The system takes into account factors such as water usage, fertilizer requirements, pest resistance, and carbon footprint to determine the most sustainable crop choices.

- By considering these factors, the system can recommend crops that not only have high yields but also have a lower environmental impact and require fewer resources to cultivate.

Benefits of the System:

- **Improved Crop Yields:** The machine learning-based recommendations can help farmers achieve higher crop yields by selecting the most suitable and sustainable crops for their local conditions.
- **Reduced Costs:** By optimizing resource usage, the system can help farmers lower their input costs for water, fertilizers, and other agricultural inputs.
- **Sustainable Agriculture Practices:** The system promotes the adoption of sustainable agriculture practices, contributing to the long-term environmental and economic viability of farming operations.

Potential Impact:

- The research highlights the potential of machine learning in revolutionizing crop recommendation systems, moving beyond traditional approaches to embrace sustainable agriculture practices.
- By integrating neural networks and genetic algorithms, the system can provide tailored recommendations that balance productivity and environmental considerations, addressing the growing need for sustainable food production.
- The proposed crop recommendation system represents a significant step forward in leveraging machine learning to support sustainable agriculture practices. By considering a broader range of factors beyond just yield maximization, the system can help farmers make informed decisions that contribute to the long-term sustainability of their operations and the environment.

CHAPTER 2: REVIEW OF LITERATURE

2.1 Research Methodologies

The methodology for the Harvesting system project that includes crop recommendation, fertilizer recommendation, and plant disease classification using machine learning and a web application for front-end using Python Flask would involve several steps:

- **Data Collection and Preprocessing:** The first step would be to collect relevant data for each module such as soil type, weather conditions, historical crop yields, soil nutrient levels, crop types, and fertilization history. The collected data would then be preprocessed to remove any noise or inconsistencies.
- **Model Development and Training:** The next step would be to develop and train the machine learning models. For the crop recommendation and fertilizer recommendation models, we would use the random forest algorithm to predict suitable crops and fertilizers based on the collected data. For the plant disease classification model, we would use the Resnet 9 algorithm to classify plant diseases based on images of plant leaves and disease labels. The models would be developed using Python libraries such as scikit-learn and Kera's.
- **Integration and Deployment:** Once the models are trained and validated, we can integrate them into a single system using Python Flask. Flask is a lightweight web application framework that allows us to develop a web application with Python. The web application can provide a user-friendly interface for farmers to input their data and receive recommendations for crops, fertilizers, and disease management
- **System Maintenance and Updates:** The final step would involve maintaining and updating the system over time. This can include monitoring the performance of the machine learning models, updating the models with new data, and updating the web application with new features and improvements based on user feedback. Flask provides a flexible and extensible architecture that makes it easy to update and maintain the web application. Overall, the

methodology for the Harvesting system project would involve a combination of data collection, machine learning models, and a web application developed using Python Flask. The project would require expertise in Python programming, machine learning algorithms, and web application development.

2.2 Feasibility Studies

Analysis is the process of finding the best solution to the problem. System analysis is the process by which we learn about the existing problems, define objects and requirements and evaluates the solutions. It is the way of thinking about the organization and the problem it involves, a set of technologies that helps in solving these problems. Feasibility study plays an important role in system analysis which gives the target for design and development.

- **Economical Feasibility**

  This study is carried out to check the economic impact that the system will have on the organization. Since the project is Machine learning based, the cost spent in executing this project would not demand cost for software and related products, as most of the products are open source and free to use. Hence the project would consumed minimal cost and is economically feasible.

- **Technical Feasibility**

  This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Since machine learning algorithms is based on pure math there is very less requirement for any professional software. And also, most of the tools are open source. The best part is that we can run this software in any system without any software requirements which makes them highly portable. Also, most of the documentation and tutorials make easy to learn the technology.

- **Social Feasibility**

  The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The main purpose of this project which is based on crop prediction is to prevent the farmer from incurring losses and improve productivity. This also ensures that there is no scarcity of food as lack of production may lead to severe consequences. Thus, this is a noble cause for the sake of the society, a small step taken to achieve a secure future.

CHAPTER 3: PROPOSED SYSTEM

3.1 SDLC

AGILE MODEL

- The Agile model is an iterative approach to software development that emphasizes flexibility, collaboration, and customer feedback. Unlike the Waterfall model, which follows a linear sequence of phases, Agile breaks development into small, manageable increments called iterations, usually lasting one to four weeks.
- Each iteration involves planning, development, testing, and review, enabling teams to deliver functional components regularly and adapt quickly to changes. Collaboration is key, with cross-functional teams working closely together and holding daily stand-up meetings, sprint planning sessions, and retrospectives.
- Customer feedback is integral, with regular demonstrations at the end of each iteration allowing stakeholders to provide input that can be incorporated into the next cycle. This approach ensures the product continuously evolves to meet user needs, leading to higher-quality software.

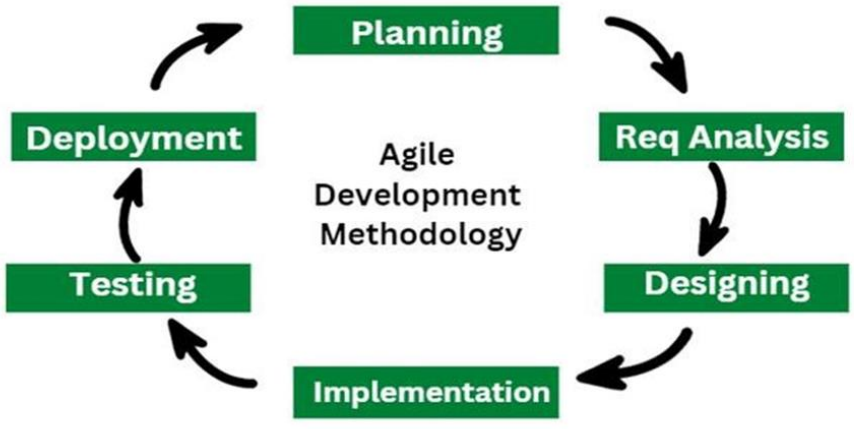

Figure 3.1.1: Agile Development Methodology

KEY PRINCIPLES OF THE AGILE MODEL:

- **Iterative Development:** Agile projects are divided into small iterations, known as sprints, typically lasting from one to four weeks. Each sprint delivers a potentially shippable product increment, allowing for continuous improvement and adaptation.
- **Customer Collaboration:** Agile teams prioritize customer collaboration and engagement throughout the development process. Customers or stakeholders are actively involved in sprint planning, reviews, and retrospectives, providing feedback and clarifying requirements as needed.
- **Adaptability:** The Agile model embraces change as a natural part of the development process. Requirements are not fixed upfront but evolve over time based on feedback and changing market conditions. Agile teams are flexible and responsive, adjusting their plans and priorities as needed to deliver the highest possible value.
- **Continuous Delivery:** The goal of Agile development is to deliver working software early and often. Each sprint produces a potentially shippable product increment, allowing stakeholders to see tangible progress and provide feedback that can be incorporated into future iterations.
- **Cross-functional Teams:** Agile teams are self-organizing and crossfunctional, comprising members with diverse skills and expertise. By working collaboratively and sharing knowledge, Agile teams are better equipped to solve complex problems and deliver high-quality solutions.

PHASES OF THE AGILE MODEL:

- **Backlog Refinement:** The Agile process begins with backlog refinement, where the product backlog is populated with user stories or requirements. The backlog is continuously refined and prioritized based on customer feedback and business value.
- **Sprint Planning:** At the start of each sprint, the Agile team selects a set of user stories from the product backlog to be completed during the sprint. Sprint

goals and tasks are defined, and the team commits to delivering the selected work within the sprint timeframe.

- **Sprint Execution:** During the sprint, the development team works collaboratively to implement the selected user stories. Daily stand-up meetings are held to track progress, discuss any obstacles, and ensure alignment with the sprint goals.
- **Sprint Review:** At the end of the sprint, the team presents the completed user stories to stakeholders during the sprint review. Stakeholders provide feedback on the delivered functionality, validate the work completed, and identify any changes or refinements needed.
- **Sprint Retrospective:** Following the sprint review, the Agile team conducts a retrospective meeting to reflect on the sprint process. The team identifies what went well, what could be improved, and any action items for future sprints, fostering a culture of continuous improvement.

AGILE MODEL APPLICATION:

The Agile model is well-suited for projects where:

- **Requirements are expected to evolve or change over time:** Agile development allows for flexibility in managing changing requirements. It emphasizes continuous improvement and adaptation to new requirements, ensuring that the product meets evolving customer needs.
- **Customer collaboration and feedback are essential for project success:** Agile emphasizes customer collaboration and feedback throughout the development process. This ensures that the product meets customer expectations and that any changes can be incorporated quickly.
- **Rapid delivery of value is required to meet changing business needs or market demands:** Agile's iterative approach enables rapid delivery of value through short sprints and incremental development. This allows for quick adaptation to changing business needs and market demands.

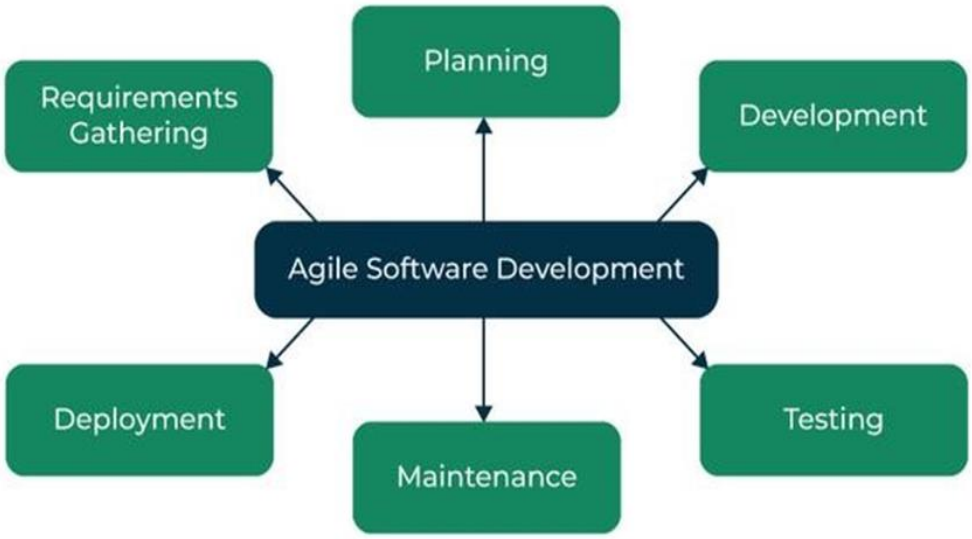

Figure 3.1.2: Agile Software Development

AGILE MODEL-ADVANTAGES:

- **Flexibility:** Easily adapt to changing requirements and priorities, keeping the project aligned with current needs.
- **Continuous Value Delivery:** Frequent releases deliver functional parts of the product regularly, enhancing customer satisfaction.
- **Improved Collaboration:** Emphasis on teamwork and communication ensures better alignment and understanding among team members and stakeholders.
- **Quick Feedback Response:** Rapidly incorporate user feedback and adapt to market changes, reducing time-to-market and increasing competitiveness.

AGILE MODEL-DISADVANTAGES:

- **Stakeholder Involvement:** Requires continuous engagement and commitment from stakeholders, which can be challenging to maintain.
- **Implementation Challenges:** Difficult to adopt in organizations with rigid processes or hierarchical structures, necessitating a cultural shift.
- **Effort in Collaboration:** High levels of collaboration and communication require additional effort and resources.

3.2 Existing System

The research works you mentioned aim to address the challenges faced by Indian farmers in selecting the most suitable crops for their land and optimizing agricultural practices. These efforts demonstrate the potential of machine learning and data-driven approaches in precision farming.

- **Regularized Greedy Forest for Crop Sequencing:** One of the approaches utilizes the Regularized Greedy Forest (RGF) algorithm to determine the optimal crop sequence at a given time stamp. By analyzing factors such as soil characteristics, weather patterns, and market demands, the RGF model suggests a sequence of crops that maximizes yield and profitability while considering the constraints and preferences of the farmer. This technique helps in optimizing crop rotation and diversification, leading to more sustainable and efficient farming practices.
- **Yield Prediction using Historical Weather Data:** Another research work proposes a model that leverages historical meteorological data as the training set. The model is trained to identify weather conditions that are detrimental to apple production. By analyzing monthly weather patterns, the model can efficiently predict apple yields, enabling farmers to make informed decisions regarding crop management and resource allocation. This approach demonstrates the potential of machine learning in mitigating the risks associated with unpredictable weather conditions and optimizing crop yields.
- **Crop Selection based on Predicted Yield Rate:** The research work also showcases the use of various algorithms, including Artificial Neural Network (ANN), K Nearest Neighbors (KNN), and Regularized Greedy Forest (RGF), in selecting crops based on predicted yield rates. These algorithms consider multiple parameters, such as soil fertility, water availability, and market demand, to predict the yield rate of different crops. By selecting crops with higher predicted yield rates, farmers can optimize their resources and maximize their returns. Additionally, the system incorporates features like pesticide prediction and online trading based on agricultural commodities, further enhancing the decision-making capabilities of farmers.

These research works highlight the growing interest among researchers in addressing the challenges faced by Indian farmers through the application of machine learning and data-driven approaches. By leveraging historical data, weather patterns, and market trends, these models aim to provide farmers with accurate predictions, recommendations, and decision support tools to improve crop selection, yield optimization, and overall farm productivity. As more researchers dedicate their efforts to this field, the potential for precision farming in India continues to grow, promising a future where data-driven insights empower farmers to make informed decisions and ensure sustainable agricultural practices.

3.2.1 Disadvantages of Existing System

The identified shortcoming in the notable published works is the focus on a single parameter (either weather or soil) for predicting the suitability of crop growth, rather than considering both factors simultaneously for the most accurate prediction. Integrating both weather and soil parameters is crucial because the interaction between these factors significantly impacts crop growth and yield outcomes.

- **Weather and Soil Interaction:**
  - **Soil Suitability:** Different soil types have varying nutrient levels, drainage capacities, and pH levels, which directly influence crop growth. For instance, certain crops thrive in well-drained soils with specific nutrient compositions, while others require different soil conditions for optimal growth.
  - **Weather Conditions:** Weather factors like temperature, rainfall, humidity, and sunlight play a vital role in determining crop success. Some crops are sensitive to temperature fluctuations, while others require specific humidity levels or sunlight exposure for healthy growth.

- **Comprehensive Prediction Approach:**
  - **Combined Impact:** By considering both weather and soil parameters together, researchers can provide a more holistic view of crop

suitability. For example, a soil type may be ideal for a particular crop, but if the region experiences extreme weather conditions like drought or excessive rainfall, the crop yield may be compromised.

- **Optimized Crop Selection:** Integrating weather and soil data allows for a more nuanced understanding of how these factors interact and influence crop growth. This comprehensive approach enables farmers to make informed decisions about crop selection, planting schedules, and resource allocation based on the combined impact of weather and soil conditions.

- **Enhanced Prediction Accuracy:**
  - **Improved Decision-Making:** By incorporating both weather and soil parameters into predictive models, researchers can enhance the accuracy of crop yield predictions. This integrated approach provides a more realistic representation of the challenges and opportunities associated with crop cultivation in specific regions.
  - **Risk Mitigation:** Considering both factors concurrently helps in identifying potential risks and vulnerabilities in crop production. Farmers can proactively adjust their farming practices, irrigation schedules, and crop selection based on a more comprehensive understanding of the environmental conditions affecting their fields.

In conclusion, while the existing research has made significant strides in predicting crop yields using either weather or soil data, there is a clear opportunity to enhance prediction accuracy by integrating both factors. By adopting a comprehensive approach that considers the synergistic effects of weather and soil conditions on crop growth, researchers can provide more robust insights and recommendations to support sustainable and productive agricultural practices.

3.3 Proposed Methods

To address the limitations of existing crop recommendation systems, we propose an Intelligent Crop Recommendation System that considers all the appropriate parameters, including temperature, rainfall, location, and soil condition, to predict crop suitability accurately. This system aims to serve as an Agro Consultant, providing reliable crop recommendations to farmers based on advanced machine learning algorithms.

Key features of the proposed system:

- **Comprehensive Data Integration:** The system integrates data from various sources, including weather stations, soil testing laboratories, and agricultural databases, to create a comprehensive dataset encompassing temperature, rainfall, soil characteristics (such as pH, nutrient levels, and texture), and location-specific information. This data is preprocessed, cleaned, and standardized to ensure high quality and consistency.
- **Machine Learning-based Crop Recommendation:** The system employs advanced machine learning algorithms to analyze the integrated dataset and provide crop recommendations tailored to the user's specific location and soil conditions. These algorithms, such as Random Forest, Support Vector Machines (SVM), and Artificial Neural Networks (ANN), are trained on historical crop performance data to learn the complex relationships between environmental factors and crop suitability.
- **Profit Analysis and Visualization:** To aid farmers in making informed decisions, the system provides profit analysis for various crops grown in different states. This feature allows users to compare the profitability of crops based on factors such as market prices, production costs, and yield potential. The results are presented through intuitive visualizations, such as charts and graphs, to help users easily interpret the data and plan their crop selection accordingly.
- **User-friendly Interface:** The system features a user-friendly interface that guides farmers through the process of providing their location and soil data. Based on this input, the system generates crop recommendations, along with

detailed information about each recommended crop, including sowing time, water requirements, and potential yield. The interface also provides access to the profit analysis feature, enabling farmers to make well-informed decisions.

- **Continuous Learning and Improvement:** The system is designed to continuously learn from user feedback and real-world outcomes. As farmers implement the recommended crops and provide feedback on their performance, the machine learning models are updated to refine the recommendations and improve the system's accuracy over time. This adaptive approach ensures that the system remains relevant and effective in the face of changing environmental conditions and market dynamics.

By integrating comprehensive data, advanced machine learning algorithms, profit analysis, and user-friendly features, the proposed Intelligent Crop Recommendation System aims to empower farmers with reliable and actionable insights to optimize their crop selection and improve overall farm productivity and profitability.

3.3.1 Module Description

The implementation steps of the Intelligent Crop Recommendation System involve the following key processes:

- **Acquisition of Training Dataset:**
    - The first step involves acquiring a comprehensive training dataset that forms the foundation for the machine learning models used in the system.
    - Various datasets are downloaded from government websites and platforms like Kaggle to gather essential agricultural data for analysis and prediction.
    - The datasets include information such as the cost of cultivation per hectare for major crops in each state, yield data, modal prices of crops, standard prices of crops, soil nutrient content, rainfall data, and temperature data.

- **Data Preprocessing:**
    - Data preprocessing is a crucial step to ensure the quality and accuracy of the dataset before feeding it into the machine learning models.
    - Null and zero values in the dataset, especially for yield data, are replaced with -1 to prevent any adverse effects on the overall prediction process.
    - Encoding of the dataset is performed to transform the data into a format that can be effectively utilized by the neural network and other machine learning algorithms.
- **Training Model and Crop Recommendation:**
    - Following data preprocessing, the next step involves training different machine learning models using the prepared dataset to generate accurate crop recommendations.
    - Machine learning models such as neural networks and linear regression are utilized to analyze the dataset and predict crop suitability based on factors like soil conditions, weather patterns, and historical agricultural data.
    - The models are trained to achieve the highest possible accuracy in recommending the best crops for specific locations, considering parameters like soil nutrient content, rainfall, temperature, and economic factors.
    - The trained models are then used to provide crop recommendations to farmers, considering all the relevant parameters and offering insights to optimize crop selection, maximize yield, and enhance farm profitability.

By following these systematic steps of dataset acquisition, data preprocessing, and model training, the Intelligent Crop Recommendation System can effectively leverage machine learning algorithms to provide farmers with valuable recommendations for crop selection, ultimately improving agricultural productivity and profitability.

3.4 System Architecture

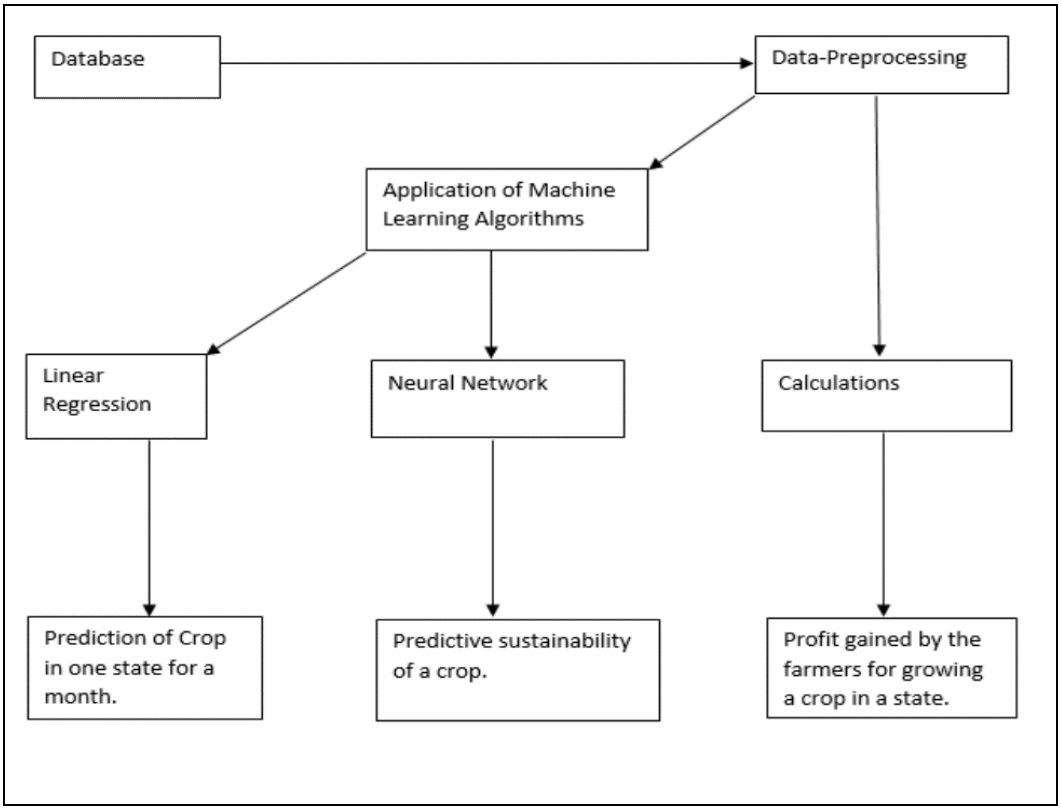

Figure 3.4: System Architecture

A system architecture is a conceptual model using which we can define the structure and behavior of that system. It is a formal representation of a system. Depending on the context, system architecture can be used to refer to either a model to describe the system or a method used to build the system. Building a proper system architecture helps in analysis of the project, especially in the early stages. Figure depicts the system architecture and is explained in the following section.

3.4.1 Proposed System Advantages

It offers several advantages over traditional methods of crop selection and management:

- **Increased Productivity:** AGRONOMISAGE can recommend the best crop to grow based on your specific location, soil conditions, and climate. This can lead to higher yields and increased profits for farmers.
- **Improved Crop Health:** By recommending the right fertilizers and identifying potential diseases early, AGRONOMISAGE can help farmers keep their crops healthy and reduce losses.
- **Reduced Costs:** AGRONOMISAGE can help farmers save money by optimizing their resource use. The system can recommend the right amount of fertilizer to apply, preventing waste. Early disease detection can also prevent costly interventions later.
- **Accessibility of Information:** AGRONOMISAGE can provide farmers with easy access to the latest information on crop science and best practices. This can be especially helpful for smallholder farmers who may not have access to extension services.
- **Data-Driven Decisions:** AGRONOMISAGE can help farmers make data-driven decisions about their crops. The system can track crop performance over time and identify trends that can be used to improve future yields.

Overall, AGRONOMISAGE has the potential to revolutionize agriculture by making it more productive, profitable, and sustainable.

3.5 Project Description

AGRONOMISAGE is your one-stop shop for smart farming! This user-friendly website leverages the power of machine learning (ML) and deep learning (DL) to empower you, the farmer, to get the most out of your land.

3.5.1 General

Here's how it works:

- **Planting Power:** Provide AGRONOMISAGE with some basic details about your field, like soil composition and climate. Using this information, AGRONOMISAGE recommends the ideal crop to cultivate for optimal yield.
- **Fertilizer Focus:** No more guessing about fertilizers! AGRONOMISAGE analyzes your chosen crop and soil conditions to suggest the most effective fertilizer for a healthy harvest.
- **Disease Diagnosis:** Ever worried about crop illness? AGRONOMISAGE has you covered! The system can identify potential plant diseases early on, allowing you to take swift action and minimize damage.

AGRONOMISAGE puts the power of data and technology directly in your hands, helping you make informed decisions to maximize your yield and become a data-driven farming pro!

3.5.2 Modules Description

For the purposes of this project we have used three popular algorithms: Linear regression, Logistic regression and Neural network. All the algorithms are based on supervised learning. Our overall system is divided into three modules:

- **Crop Recommender:** This is like your personal farming consultant! Tell AGRONOMISAGE about your land - details like sunlight hours, average rainfall, and soil type. AGRONOMISAGE uses this information, along with a massive dataset of crop performance factors, to recommend the most suitable crop for your specific conditions. This can help you maximize yield and profits by selecting a crop that thrives in your environment.
- **Fertilizer Recommender:** Stop wasting money on unnecessary fertilizers! AGRONOMISAGE analyzes your chosen crop and your

soil's composition. Based on this data, it recommends the optimal fertilizer type and amount to ensure your plants receive the exact nutrients they need for healthy growth. This can improve your crop's quality and reduce costs.

- **Crop Disease Predictor:** Plant illness can be a farmer's worst nightmare. AGRONOMISAGE acts as your early warning system! The system can analyze data and images (potentially uploaded by you) to identify signs of potential diseases in your crops. Early detection allows you to take preventive measures and minimize damage to your harvest.

3.5.3 Given Input Expected Output

The Agronomisage Platform project expects different types of inputs depending on the application being used, and it provides relevant outputs based on those inputs.

- **Crop recommendation:**
  - **Input:** Soil data such as pH levels, moisture content, and nutrient composition (nitrogen, phosphorus, potassium, etc.)
  - **Output:** Prediction of the most suitable crops to grow based on the soil conditions
- **Fertilizer recommendation:**
  - **Input:** Soil data (pH, moisture, nutrients) and the crop being grown
  - **Output:** Recommendation of appropriate fertilizers based on nutrient deficiencies or excesses in the soil
- **Plant disease prediction:**
  - **Input:** Image of a diseased plant leaf
  - **Output:** Prediction of the specific disease affecting the plant and suggestions for prevention and treatment

For example, if a user inputs the following soil data for the crop recommendation application:

- PH: 6.5
- Moisture: 25%

- Nitrogen: 20 ppm
- Phosphorus: 10 ppm
- Potassium: 15 ppm

The expected output would be a list of crops that are most suitable for growing in those soil conditions, such as:

- Tomatoes
- Peppers
- Eggplants

Similarly, if a user inputs soil data and specifies that they are growing tomatoes, the fertilizer recommendation application might suggest:

- Nitrogen-rich fertilizer to address phosphorus deficiency
- Calcium supplement to prevent blossom end rot

And if a user uploads an image of a tomato leaf with early blight disease, the plant disease prediction application should output:

- **Disease:** Early blight
- **Description:** Early blight is a fungal disease that causes brown or black spots on tomato leaves.
- **Prevention:** Use disease-resistant varieties, practice crop rotation, and apply fungicides as needed.

By providing relevant inputs, users can obtain valuable insights and recommendations from the Agronomisage Platform project to optimize their agricultural practices and improve crop yields.

3.6 System Requirement Specification

3.6.1 Hardware Requirements

- Processor: 2 gigahertz (GHz) or faster processor or SoC.
- RAM: 6 gigabyte (GB) for 32-bit or 8 GB for 64-bit.
- Hard disk space: =16GB.

3.6.2 Software Requirements

- Operating System: Windows XP/7/8/8.1/10, Linux and Mac

- Coding Language: Python
- Tools:
    - Pandas
    - Numpy
    - Tensorflow
    - Keras
    - Sickitlearn
    - Matplotlib
    - Flask
    - HTML
    - CSS
    - JavaScript
    - Anaconda

Machine Learning Tools

There are many different software tools available to build machine learning models and to apply these models to new, unseen data. There are also a large number of well defined machine learning algorithms available. These tools typically contain libraries implementing some of the most popular machine learning algorithms. They can be categorized as follows :

- Pre-built application-based solutions.
- Programming languages which have specialized libraries for machine learning

Using programming languages to develop and implement models is more flexible and gave us better control of the parameters to the algorithms. It also allows us to have a better understanding of the output models produced. Some of the popular programming languages used in the field of machine learning are:

- **Python:** Python is an extremely popular choice in the field of machine learning and AI development. Its short and simple syntax make it extremely easy to learn
- **R:** R is one of the most effective and efficient languages for analyzing and manipulating data in statistics. Using R, we can easily produce well-designed publication-quality plot, including mathematical symbols and formulae where needed. Apart from being a general-purpose language, R has numerous of packages like RODBC, Gmodels, Class and Tm which are used in the field of machine learning. These packages make the implementation of machine learning algorithms easy, for cracking the business associated problems
- **Tensorflow:** TensorFlow is an end-to-end open source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries, and community resources that lets researchers push the state-of-the-art in ML and developers easily build and deploy ML-powered applications. TensorFlow was originally developed by researchers and engineers working on the Google Brain team within Google's Machine Intelligence Research organization to conduct machine learning and deep neural networks research. The system is general enough to be applicable in a wide variety of other domains, as well. TensorFlow provides stable Python and C++ APIs, as well as non-guaranteed backward compatible API for other languages.

1. <u>SciKit-learn</u>

SciKit learn is an open source machine learning library built for python. Since its release in 2007, Scikit-learn has become one of the most popular open source machine learning libraries. Scikit-learn (also called sklearn) provides algorithms for many ma- chine learning tasks including classification, regression, dimensionality reduction and clustering. The documentation for scikit-learn is comprehensive, popular and well maintained. Sklearn is built on mature Python

Libraries such as NumPy, SciPy, and matplotlib. While languages such as R and MATLAB are extremely popular and useful for machine learning, we decided to choose Python along with its SciKit-learn libraries as our programming language of choice. The reasons for this are:

- We already have some familiarity and exposure to Python, and thus have a smaller learning curve.
- Both Python and Scikit-learn have excellent documentation and tutorials available online
- The number of classic machine learning algorithms that come with Scikitlearn, and the consistent patterns for using the different models i.e., each model can be used with the same basic commands for setting up the data, training the model and using the model for prediction. This makes it easier to try a range of machine learning algorithms on the same data.
- The machine learning algorithms included with sklearn have modifiable parameters known as hyperparameters that effect the performance of the model. These usually have sensible default values, so that we can run them without needing a detailed knowledge or understanding of their semantics.
- The IPython notebook, which is an interactive computational environment for Python, in which a user can combine code execution, rich text, mathematics and plots in a web page. This functionality allows us to provide the notebooks we used to run our experiments almost as an audit and in a presentable.

2. Dataset

For the system, we are using various datasets all downloaded for government website and kaggle.

Datasets include:-

- Cost of cultivation per ha dataset for major crops in each state Yield dataset
- Modal price of crops Standard price of crops
- Soil nutrient content dataset Rainfall Temperature dataset

A brief description of the datasets:

- **Yield Dataset:** This dataset contains yield for 16 major crops grown across all the states in kg per hectare.Yield of 0 indicates that the crop is not cultivated in the respective state.
- **Cost of Cultivation dataset:** This dataset provides the cost of cultivation for each crop in Rs. per hectare.
- **Modal price of crops:** This dataset gives the average market prices for those crops over a period of two months
- **Standard price of crops:** This dataset gives the current market price of the crops in Rs per hectare.
- **Rainfall Temperature dataset:** This dataset contains crops ,max and min rainfall, max and min temperature, max and min rainfall and ph values.
- **Soil nutrient content dataset:** This dataset has five columns with the attributes in the order-State, Nitrogen content, Phosphorous content, Potassium content and average ph. The nutrient content is represented with encoded alphabets-VL, L, M, H, VH with the meaning: VL -Very Low, L-Low, M-Medium, H-High, VH-Very high.

3. <u>Data Preprocessing:</u>

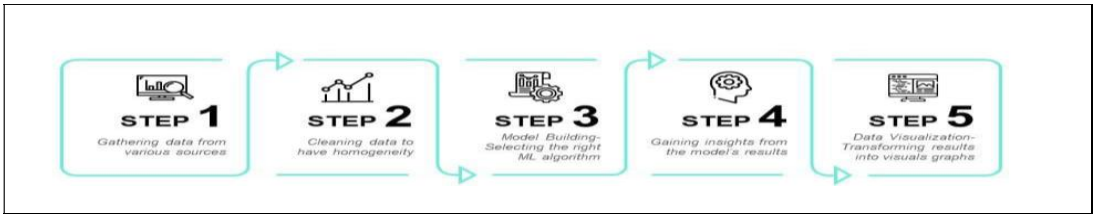

Figure 3.6.2: Data Preprocessing

This step includes replacing the null and 0 values for yield by -1 so that it does not effect the overall prediction. Further we had to encode the data-set so that it could be fed into the neural network.

Machine Learning Algorithms

Machine Learning algorithms used in the recommendation system are:

- **Linear Regression:** Linear regression is a linear approach to modeling the relationship between a scalar response (or dependent variable) and one or more explanatory variables (or independent variables). Linear regression is used for finding linear relationship between target and one or more predictors. It fits a linear model with coefficients w = (w1, . . . , wp) to minimize the residual sum of squares between the observed targets in the dataset, and the targets predicted by the linear approximation. Linear regression is used for finding linear relationship between target and one or more predictors.
- **Logistic Regression:** Logistic Regression is the appropriate regression analysis to conduct when the dependent variable is dichotomous (binary). The logis- tic model (or logit model) is used to model the probability of a certain class or event existing such as pass/fail, win/lose, alive/dead or healthy/sick. This can be extended to model several classes of events such as determining whether an image contains a cat, dog, lion, etc. Each object being detected in the image would be assigned a probability between 0 and 1 and the sum adding to one.
- **Neural Network:** Neural networks are a set of algorithms, modeled loosely after the human brain, that are designed to recognize patterns. They interpret sensory data through a kind of machine perception, labeling or clustering raw input. The patterns they recognize are numerical, contained in vectors, into which all real-world data, be it images, sound, text or time series, must be translated. Neural networks help us cluster and classify. Neural Networks

are themselves general function approximations, which is why they can be applied to almost any machine learning problem about learning a complex mapping from the input to the output space.

3.6.3 Deliverables

- Application
- Documentation
- Manuals
- System Specification
- Runnable CD for Direct Installation of Setup etc.,

CHAPTER 4: TEST STRATEGIES AND METHODOLOGIES

4.1 Test Strategies And Methods

Software testing is performed to verify that the completed software package functions according to the expectations defined by the requirements/specifications. The overall objective to not to find every software bug that exists, but to uncover situations that could negatively impact the customer, usability and/or maintainability.

Testing is an important aspect of any software development project. It ensures that the software is functioning as expected and meets the requirements of the users. There are different testing strategies and methodologies that can be used to test software.

TYPES OF TESTING

- WHITE- BOX TESTING

  White-box testing (also known as clear box testing, glass box testing, transparent box testing and structural testing, by seeing the source code) tests internal structures or workings of a program, as opposed to the functionality exposed to the end-user. In this testing approach, the tester has knowledge of the internal workings of the software. This approach is used to test the code quality, performance, and security of the software. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs. This is analogous to testing nodes in a circuit, e.g. in-circuit testing (ICT).While white-box testing can be applied at the unit, integration and system levels of the software testing process, it is usually done at the unit level. It can test paths within a unit, paths between units during integration, and between subsystems during a system–level test. Techniques used in white-box testing include:

  - API testing – testing of the application using public and private APIs.
  - Code coverage – creating tests to satisfy some criteria of code coverage.
  - Fault injection methods – intentionally introducing faults to gauge the efficacy of testing strategies.

- Mutation testing methods
- Static testing methods

- BLACK-BOX TESTING

  Black-box testing treats the software as a "black box", examining functionality without any knowledge of internal implementation, without seeing the source code. The testers are only aware of what the software is supposed to do, not how it does it. Black-box testing methods include: equivalence partitioning, boundary value analysis, all-pairs testing, state transition tables, decision table testing, fuzz testing, model-based testing, use case testing, exploratory testing and specification-based testing. One advantage of the black box technique is that no programming knowledge is required. Whatever biases the programmers may have had, the tester likely has a different set and may emphasize different areas of functionality. This method of test can be applied to all levels of software testing: unit, integration, system and acceptance.

- GREY BOX TESTING

  Grey box testing is a software testing technique that combines aspects of both black box and white box testing. In grey box testing, the tester has some knowledge of the internal workings of the software, but not a complete understanding. This approach is used to test the functionality, performance, and security of the software.

  Grey box testing involves testing the software at different levels of abstraction. For example, the tester might have access to the source code, but not the specific implementation details of certain functions. Alternatively, the tester might be able to see the data flowing through the system, but not the specific algorithms used to process it.

  Grey box testing can be particularly useful in testing complex systems, where it can be difficult to understand the internal workings of the software. It can also be used to test software that is undergoing changes or modifications, where there may be a need to test new features or functionality.

However, grey box testing can also be more time-consuming and complex than black box testing. It requires testers to have a certain level of technical expertise and knowledge of the software being tested.

TESTING METHODOLOGIES

- **AGILE TESTING:** Agile testing is a software testing approach that is part of the Agile software development methodology. Agile testing focuses on continuous testing throughout the software development process, rather than testing at the end of the development cycle. The goal of Agile testing is to ensure that the software is of high quality and meets the needs of the users.
  Agile testing involves working closely with the development team to ensure that testing is integrated into the development process. Testers and developers work together to identify and resolve defects as they arise, rather than waiting until the end of the development cycle to test the software. Agile testing involves testing small units of code and integrating them into the software as they are developed.
- **WATERFALL TESTING:** Waterfall testing is a software testing approach that is part of the Waterfall software development methodology. In the Waterfall methodology, the software development process is divided into sequential phases, with each phase completed before moving on to the next one. The goal of Waterfall testing is to ensure that the software meets the requirements and specifications of the project before it is released to the end-users.Waterfall testing is often sequential, with each phase completed before moving on to the next one. This approach can lead to a more thorough testing process, as each phase builds upon the previous one. However, it can also be time-consuming and inflexible, as changes to the software can be difficult to implement once a phase is completed.
- **EXPLORATORY TESTING:** Exploratory testing is a software testing approach that emphasizes the freedom and creativity of the tester in designing and executing tests. In exploratory testing, the tester relies on their experience, knowledge, and intuition to identify defects and other issues in the

software.Unlike traditional testing approaches, exploratory testing does not rely on predefined test cases or scripts. Instead, the tester actively explores the software, attempting to identify issues as they arise. The tester may use various techniques such as boundary value analysis, equivalence class partitioning, decision tables to guide their exploration.

4.2 Testing Strategy

For our AI content generation project, the testing strategy will encompass a comprehensive approach to ensure the functionality, performance, and security of the software. The testing phases will include:

- **Unit Testing:** This phase will focus on testing the smallest units of the system. We will utilize testing frameworks like Jest for unit tests on individual components of the AI algorithms.
- **Integration Testing:** Integration testing will examine the interactions between different modules of the system. In our project, integration tests will assess the seamless integration between AI models, databases, and external APIs.
- **System Testing:** System testing will evaluate the overall functionality of the system, including user interfaces and backend processes. Manual testing will validate the user experience, while automated testing tools like Cypress will ensure the robustness of the system.
- **Acceptance Testing:** This phase will verify that the system meets stakeholder requirements. Acceptance tests will be designed to cover various scenarios, including positive and negative test cases, ensuring alignment with stakeholder expectations. Additionally, user feedback will play a crucial role in refining the system during acceptance testing.

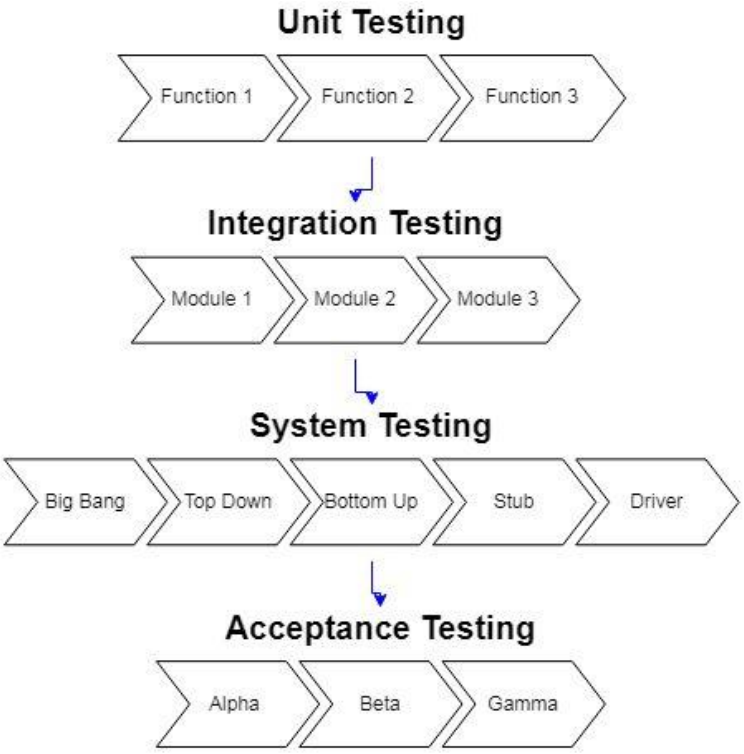

Figure 4.2 Test Procedure

4.3 Test Cases

VALIDATION TEST CASES

In software testing, validation testing plays a crucial role in ensuring that a system or component functions correctly and meets the specified requirements. This type of testing focuses on evaluating the software to determine whether it satisfies the needs and expectations of the client. Validation testing is performed after the completion of the testing phase, encompassing various testing types such as functional, nonfunctional, and user acceptance testing. The primary goal of validation testing is to confirm that the software product is ready for release, ensuring it performs as intended in a real-world environment. This process involves verifying that all features and functionalities work correctly, that the software is reliable, and that it meets the defined business requirements. By conducting validation testing, organizations can identify and rectify any issues or defects that

may impact the user experience, thereby delivering a high-quality product that aligns with the client's specifications and needs.

Test Case Description	Expected Output	Actual Output	Tester Name	Date
Unable to enter the crop details	To be entered desired soil input values	Unable to enter soil PH value	Md Mehboob	28-03-2024
Unable to enter the crop details	To be entered desired soil input values	Able to enter soil PH value	Md Mehboob	29-03-2024

Table 4.3.1: Test case 1

Test Case Description	Expected Output	Actual Output	Tester Name	Date
Unable To Predict The Recommended Crop	Predict The Recommended Crop	Showing “Internal Server Error”	Syed Ansar Ahmed Yaqoob	01-04-2024
Unable To Predict The Recommended Crop	Predict The Recommended Crop	Recommended Crop Details Are Shown	Syed Ansar Ahmed Yaqoob	01-04-2024

Table 4.3.2: Test Case 2

Test Case Description	Expected Output	Actual Output	Tester Name	Date
To Upload disease aided plant image	Image to be uploaded	Unable to upload image	Md Mehboob	03-04-2024
To Upload disease aided plant image	Image to be uploaded	Image uploaded successfully	Md Mehboob	03-04-2024

Table 4.3.3: Test case 3

Test Case Description	Expected Output	Actual Output	Tester Name	Date
To Check the interface link between home page and fertilizer recommendation page	To be directed to fertilizer recommendation page	Remain unchanged	Mohammed Daniyaalullah Khan	05-04-2024
To Check the interface link between home page and fertilizer recommendation page	To be directed to fertilizer recommendation page	Successfully directed to fertilizer recommendation page	Mohammed Daniyaalullah Khan	05-04-2024

Table 4.3.4: Test case 4

CHAPTER 5: DESIGN AND IMPLEMENTATION

5.1 Unified Modelling Language

5.1.1 Model

- A model is a simplification of reality.
- A model provides the blueprints of a system.
- A model may be structural, emphasizing the organization of the system, or it may be behavioral, emphasizing the dynamics of the system.
- We build models so that we can better understand the system we are developing.
- We build models of complex systems because we cannot comprehend such a system in its entirety.

Through modeling, we achieve four aims -

- Models help us to visualize a system as it is or as we want it to be.
- Models permit us to specify the structure or behavior of a system.
- Models give us a template that guides us in constructing a system.
- Models document the decisions we have made

5.1.2 Principles of Modelling

- The choice of what models to create has a profound influence on how a problem is attacked and how a solution is shaped
- Every model may be expressed at different levels of precision
- The best models are connected to reality
- No single model is sufficient. Every nontrivial system is best approached through a small set of nearly independent models

UML is a graphical notation used to visualize, specify, construct and document the artifact of software-intensive. UML is appropriate for modeling systems ranging from Enterprise Information Systems to Distributed Web-based

Applications and even to Hard Real-time Embedded systems. UML effectively starts with forming conceptual modeling of the language.

5.1.3 Applications of UML

UML is intended primarily for software-intensive systems. It has been use effectively for such domains as

1. Enterprise Information Systems
2. Banking and Financial Services
3. Telecommunications
4. Transportation
5. Defense and Aerospace
6. Retail
7. Medical Electronics
8. Scientific
9. Distributed Web-based Service.

5.1.4 Use Case Diagram

In the UML, a use case diagram will summarize the main points of your`s system's users (also referred to as actors) and their interactions with the system. A use case diagram is used to represent the dynamic behavior of a system. It encapsulates the system's functionality by incorporating use cases, actors, and their relationships.

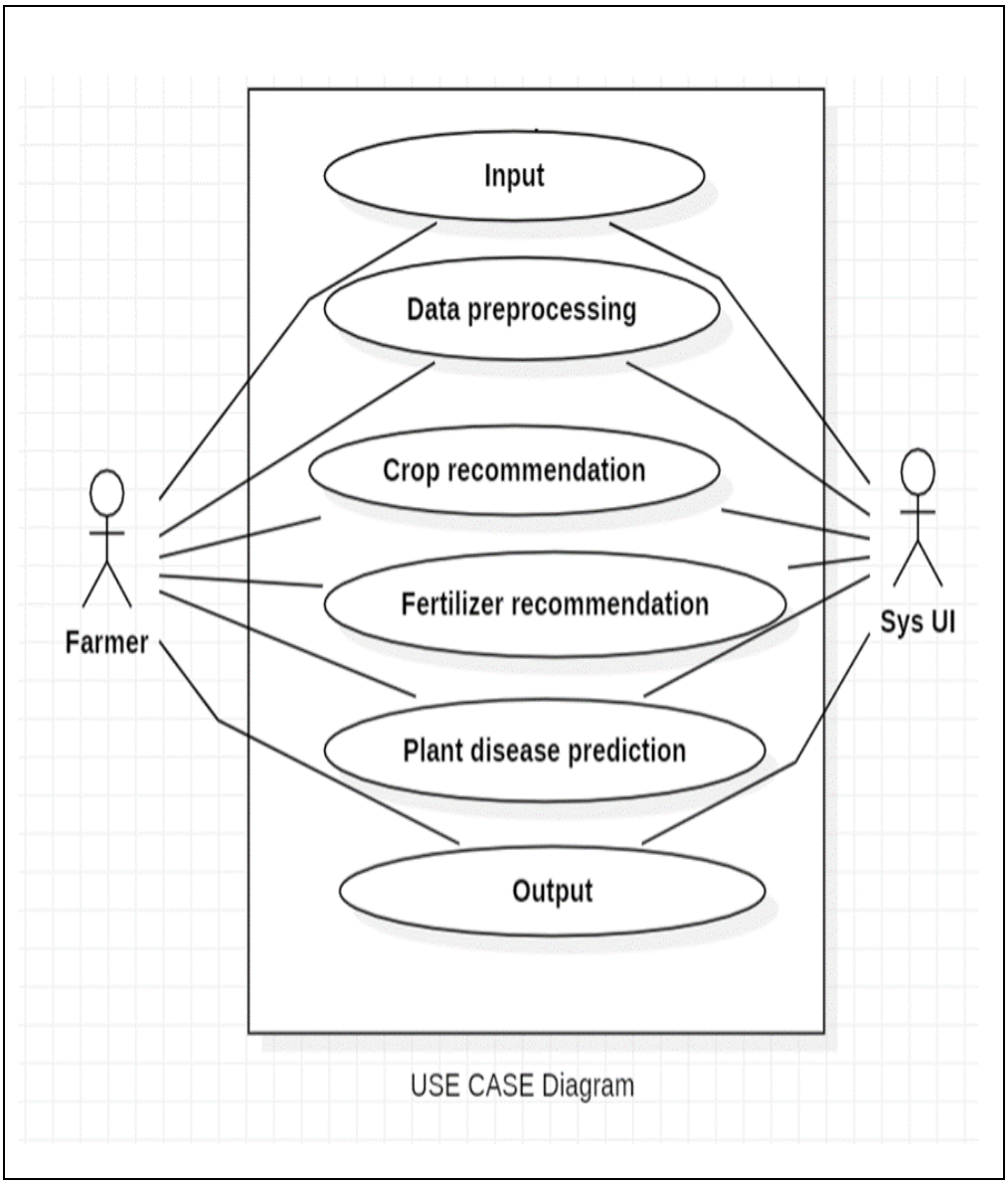

Figure 5.1.4 Use Case Diagram

5.1.5 Class Diagram

In software package engineering, a class diagram at intervals of the Unified Modelling Language (UML) can be the fashion of a static structure diagram that describes the structure of a system by showing the system`s categories, attributes, operations, and additionally the relationships among objections.

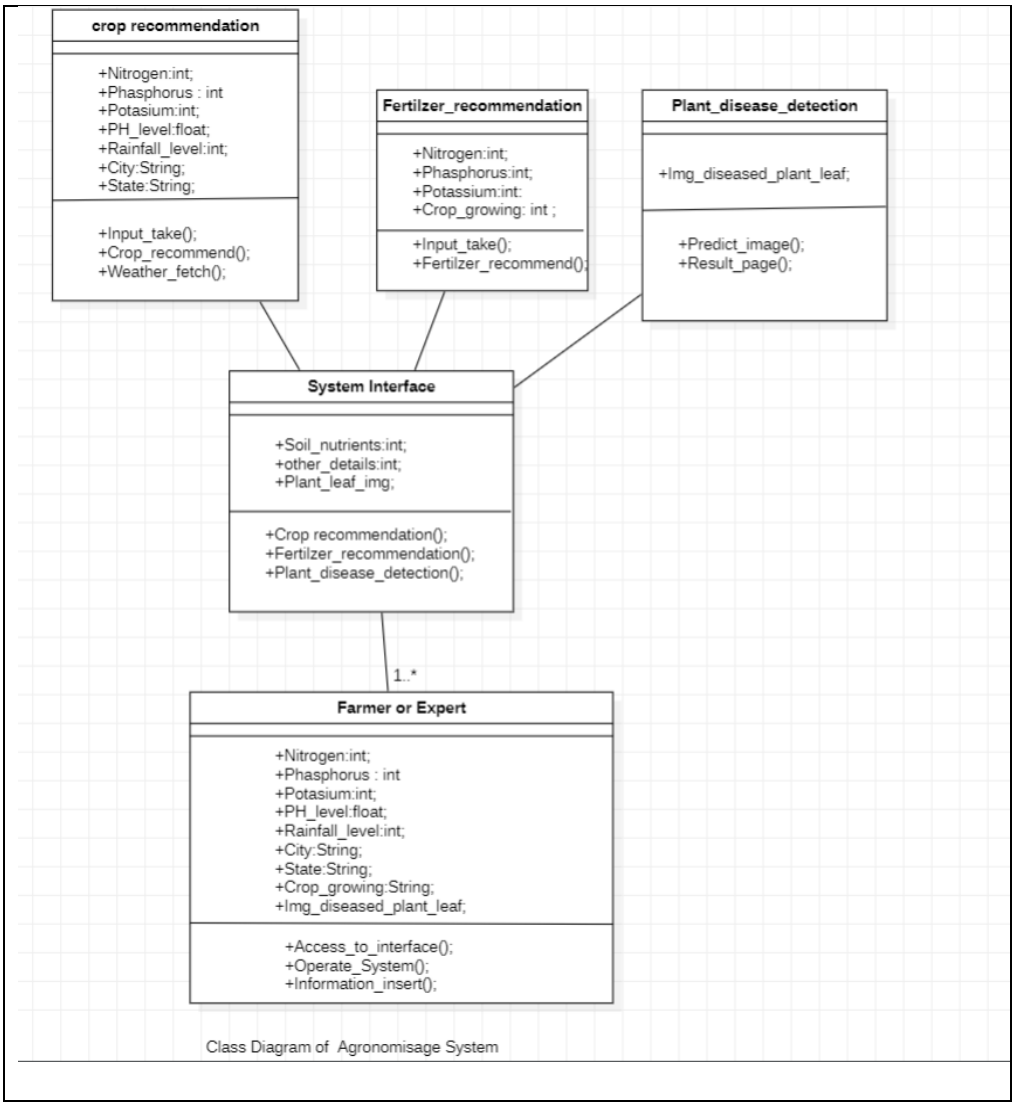

Figure 5.1.5 Class Diagram

5.1.6 Sequence Diagram

A sequence diagram or system sequence diagram (SSD) shows object interactions arranged in time sequence within the field of software package engineering. It depicts the objects involved within the state of affairs and also the sequence of messages changed between the objects required to hold out the practicality of the state of affairs.

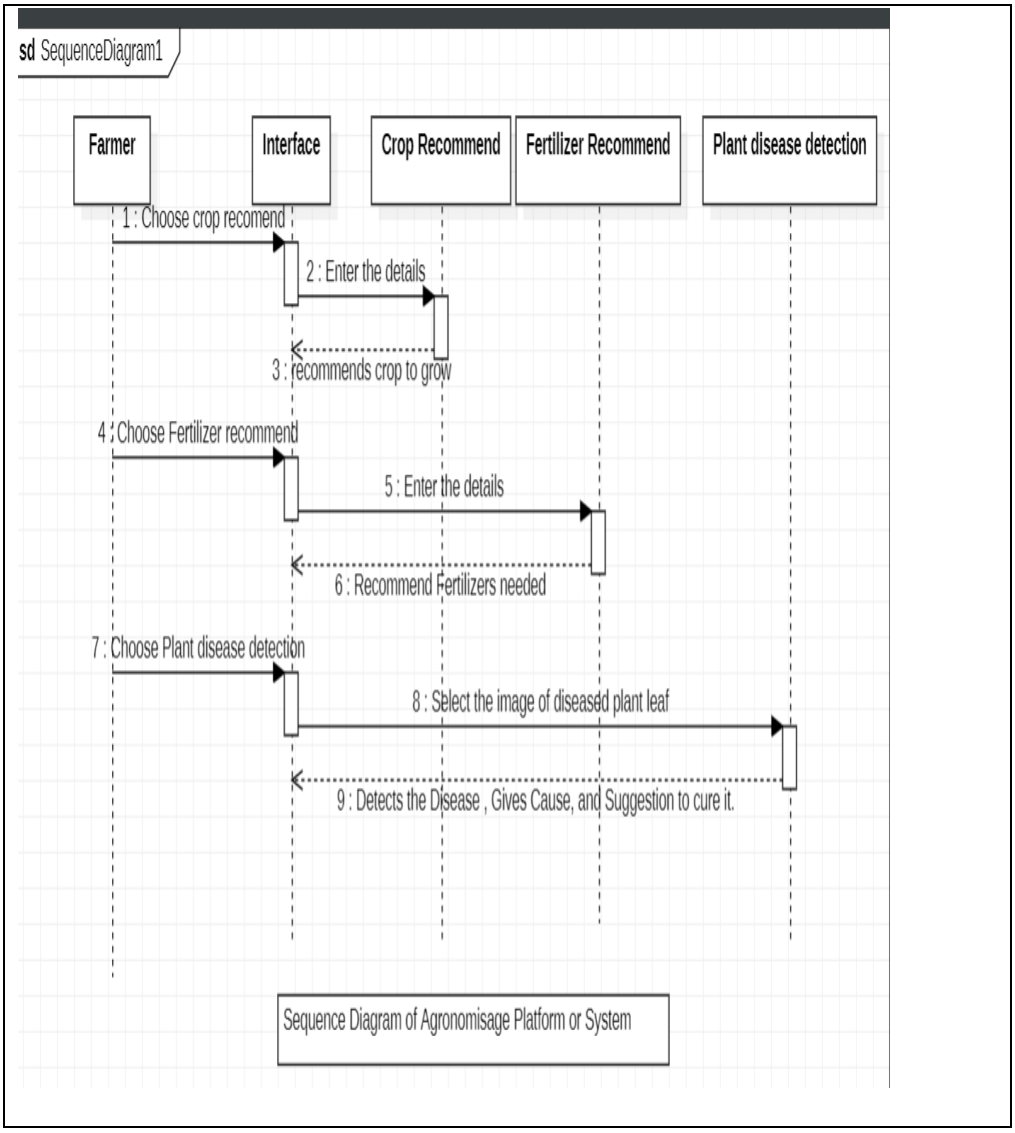

Figure 5.1.6 Sequence Diagram

5.1.7 Activity Diagram

The activity diagram could also be a graphical illustration for representing the flow of interactions at intervals and specific eventualities. It's sort of a flowchart at intervals that varied activities can be performed at intervals the system area unit portrayed.

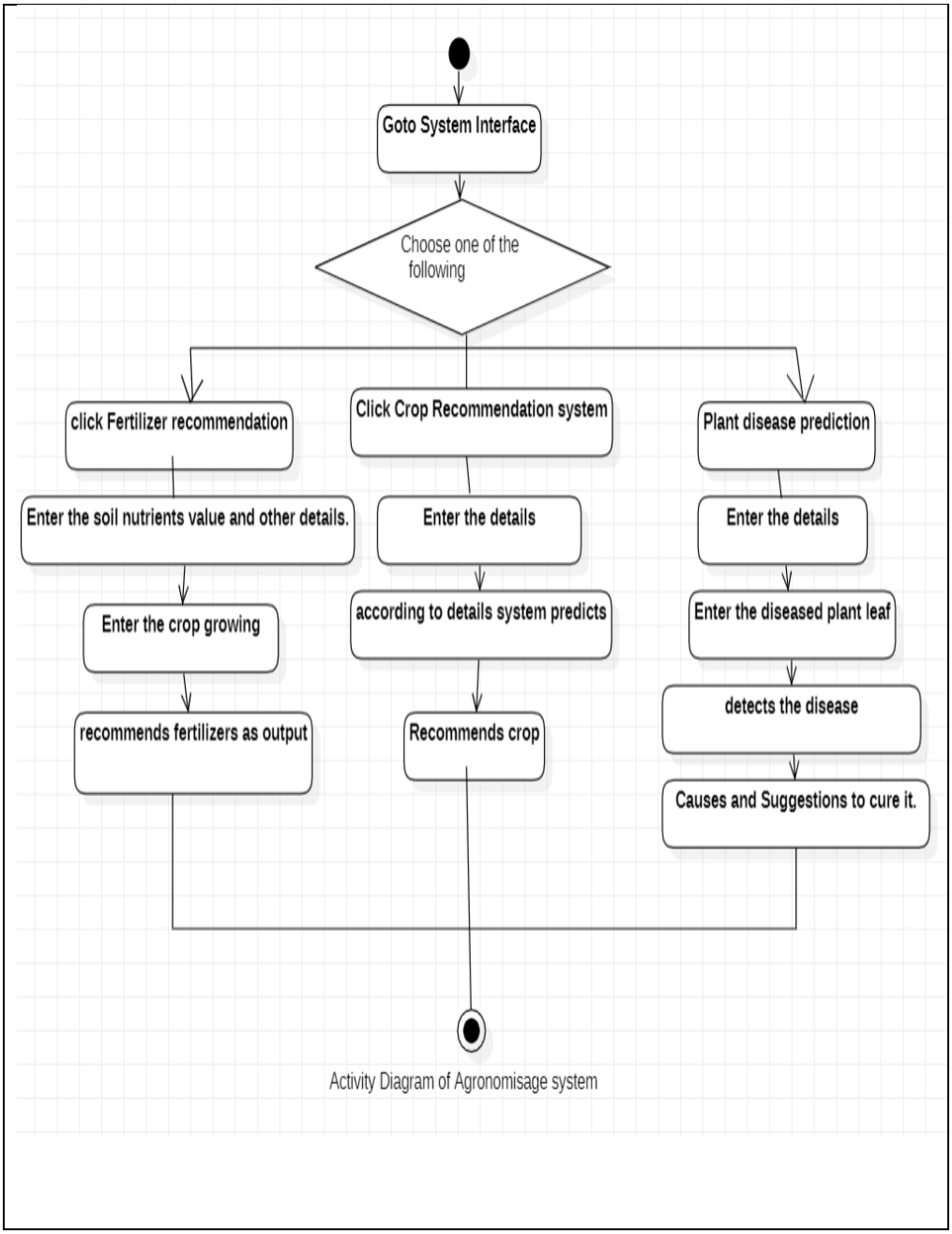

Figure 5.1.7 Activity Diagram

5.2 Sample Code

```
# Importing essential libraries and modules
from flask import Flask, render_template, request, Markup
import numpy as np
import pandas as pd
import requests
import pickle
import io
import config
import torch
from torchvision import transforms
from PIL import Image
from utils.model import ResNet9
#==========================================================
# Custom functions for calculations
def weather_fetch(city_name):
    """
    Fetch and returns the temperature and humidity of a city
    :params: city_name
    :return: temperature, humidity
    """
    api_key = config.weather_api_key
    base_url = "http://api.openweathermap.org/data/2.5/weather?"
    complete_url = base_url + "appid=" + api_key + "&q=" + city_name
```

```
    response = requests.get(complete_url)
    x = response.json()
    print("Response from OpenWeatherMap API:", x)  # Debugging print
    if 'main' in x:
        y = x["main"]
        temperature = round((y["temp"] - 273.15), 2)
        humidity = y["humidity"]
        return temperature, humidity
    else:
        return None
def predict_image(img, model):
    """
    Transforms image to tensor and predicts disease label
    :params: image
    :return: prediction (string)
    """
    transform = transforms.Compose([
        transforms.Resize(256),
        transforms.ToTensor(),
    ])
    image = Image.open(io.BytesIO(img))
    img_t = transform(image)
    img_u = torch.unsqueeze(img_t, 0)
    # Get predictions from model
```

```
    yb = model(img_u)
    # Pick index with highest probability
    _, preds = torch.max(yb, dim=1)
    prediction = disease_classes[preds[0].item()]
    # Retrieve the class label
    return prediction
#============================================================
=#FLASK APP
app = Flask(_name_)
# render home page
@ app.route('/')
def home():
    title = 'Harvestify - Home'
    return render_template('index.html', title=title)
# render crop recommendation form page
@ app.route('/crop-recommend')
def crop_recommend():
    title = 'Harvestify - Crop Recommendation'
    return render_template('crop.html', title=title)
# render fertilizer recommendation form page
@ app.route('/fertilizer')
def fertilizer_recommendation():
    title = 'Harvestify - Fertilizer Suggestion'
    return render_template('fertilizer.html', title=title)
# render disease prediction input page
```

```
#=============================================================
# RENDER PREDICTION PAGES

# render crop recommendation result page

@ app.route('/crop-predict', methods=['POST'])

def crop_prediction():

    title = 'Harvestify - Crop Recommendation'

    if request.method == 'POST':

        # Obtain form data

        N = int(request.form['nitrogen'])

        P = int(request.form['phosphorous'])

        K = int(request.form['pottasium'])

        ph = float(request.form['ph'])

        rainfall = float(request.form['rainfall'])

        city = request.form.get("city")

        # Fetch weather data

        if weather_fetch(city) is not None:

            temperature, humidity = weather_fetch(city)

            data = np.array([[N, P, K, temperature, humidity, ph, rainfall]])

            my_prediction = crop_recommendation_model.predict(data)

            final_prediction = my_prediction[0]

            return render_template('crop-result.html', prediction=final_prediction,
title=title)

        else:

            return render_template('try_again.html', title=title)

# render fertilizer recommendation result page
```

```
@ app.route('/fertilizer-predict', methods=['POST'])

# utils/fertilizer.py

fertilizer_dic = {

    "NHigh": "Your soil has a high nitrogen content. You should use fertilizers with lower nitrogen content.",

    "Nlow": "Your soil has a low nitrogen content. You should use fertilizers with higher nitrogen content.",

    "PHigh": "Your soil has a high phosphorus content. You should use fertilizers with lower phosphorus content.",

    "Plow": "Your soil has a low phosphorus content. You should use fertilizers with higher phosphorus content.",

    "KHigh": "Your soil has a high potassium content. You should use fertilizers with lower potassium content.",

    "Klow": "Your soil has a low potassium content. You should use fertilizers with higher potassium content."

}

def fert_recommend():

    title = 'Harvestify - Fertilizer Suggestion'

    crop_name = str(request.form['cropname'])

    N = int(request.form['nitrogen'])

    P = int(request.form['phosphorous'])

    K = int(request.form['pottasium'])

    # Process fertilizer recommendation

    df = pd.read_csv('Data/fertilizer.csv')

    nr = df[df['Crop'] == crop_name]['N'].iloc[0]

    pr = df[df['Crop'] == crop_name]['P'].iloc[0]

    kr = df[df['Crop'] == crop_name]['K'].iloc[0]
```

```
    n = nr - N
    p = pr - P
    k = kr - K
    temp = {abs(n): "N", abs(p): "P", abs(k): "K"}
    max_value = temp[max(temp.keys())]
    if max_value == "N":
        key = 'NHigh' if n < 0 else "Nlow"
    elif max_value == "P":
        key = 'PHigh' if p < 0 else "Plow"
    else:
        key = 'KHigh' if k < 0 else "Klow"
    response = Markup(str(fertilizer_dic[key]))
    return render_template('fertilizer-result.html', recommendation=response,
title=title)

# render disease prediction result page

@ app.route('/disease-predict', methods=['GET', 'POST'])

def disease_prediction():
    title = 'Harvestify - Disease Detection'
    if request.method == 'POST':
        if 'file' not in request.files:
            return redirect(request.url)
        file = request.files.get('file')
        if not file:
            return render_template('disease.html', title=title)
        try:
```

```
            img = file.read()
            prediction = predict_image(img, disease_model)
            prediction = Markup(str(disease_dic[prediction]))
            return render_template('disease-result.html', prediction=prediction,
title=title)
        except Exception as e:
            print(e)
            pass
    return render_template('disease.html', title=title)
#==============================================================
if _name_ == '_main_':
    app.debug = True
    app.run()
```

5.3 Screenshots

5.3.1 Home page

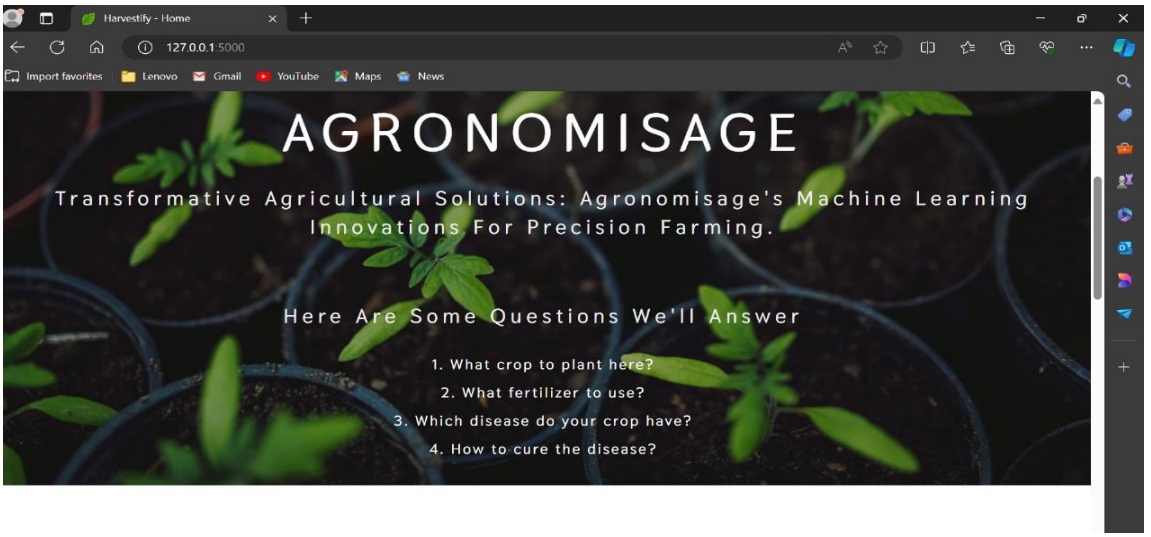

Figure 5.3.1 Home page

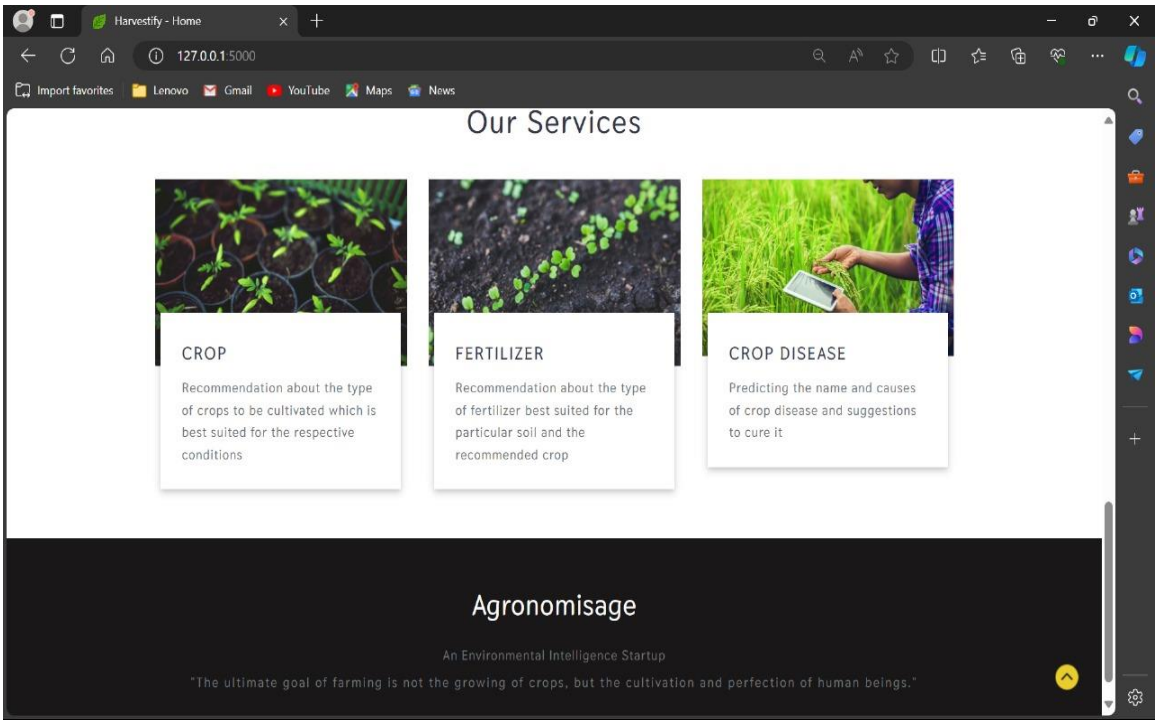

Figure 5.3.1.1 Home page

5.3.2 About Us

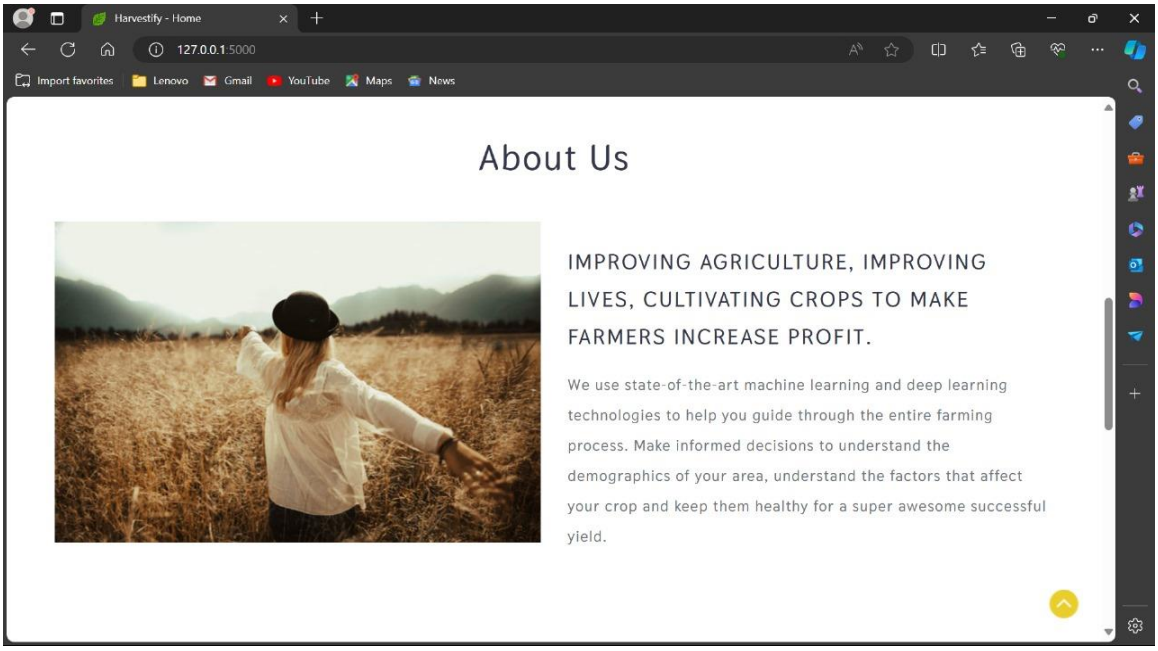

Figure 5.3.2 About Us

5.3.3 Fertilizer Recommendation

Input

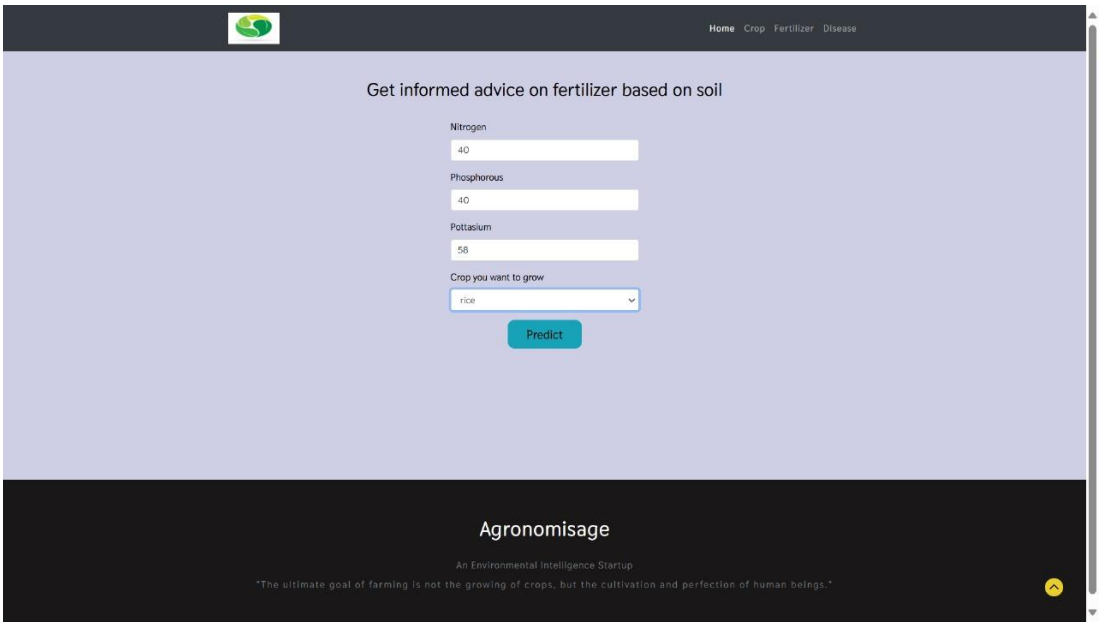

Figure 5.3.3 Fertilizer Recommendation Input

Output

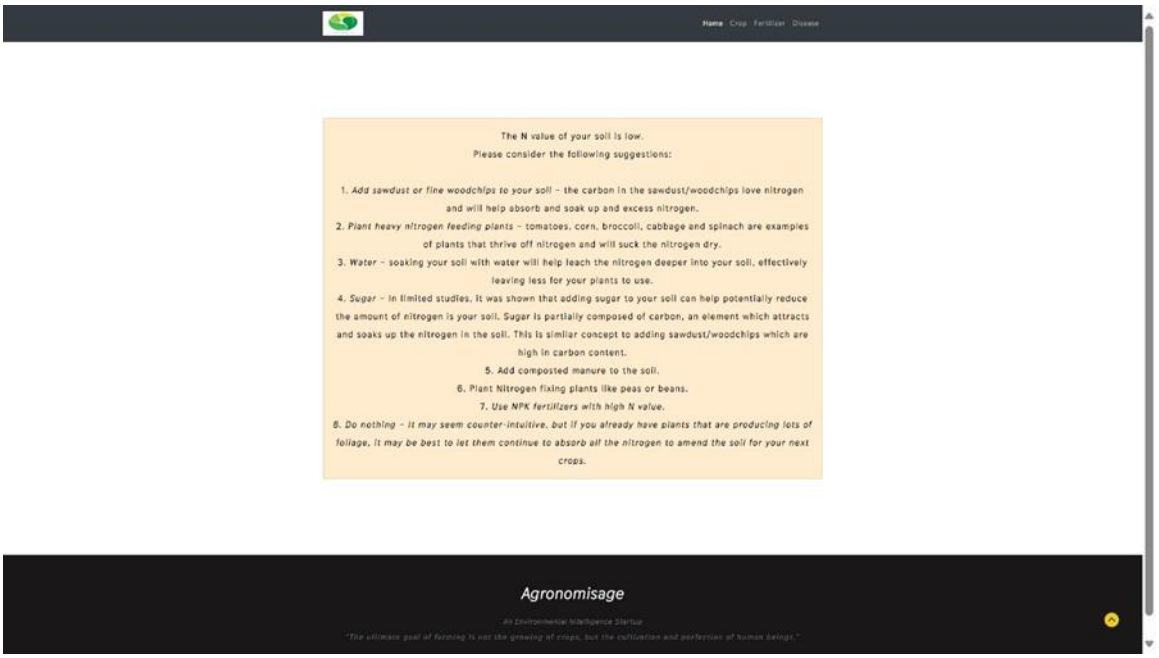

Figure 5.3.3.1 Fertilizer Recommendation output

5.3.4 Plant Disease Prediction

Input

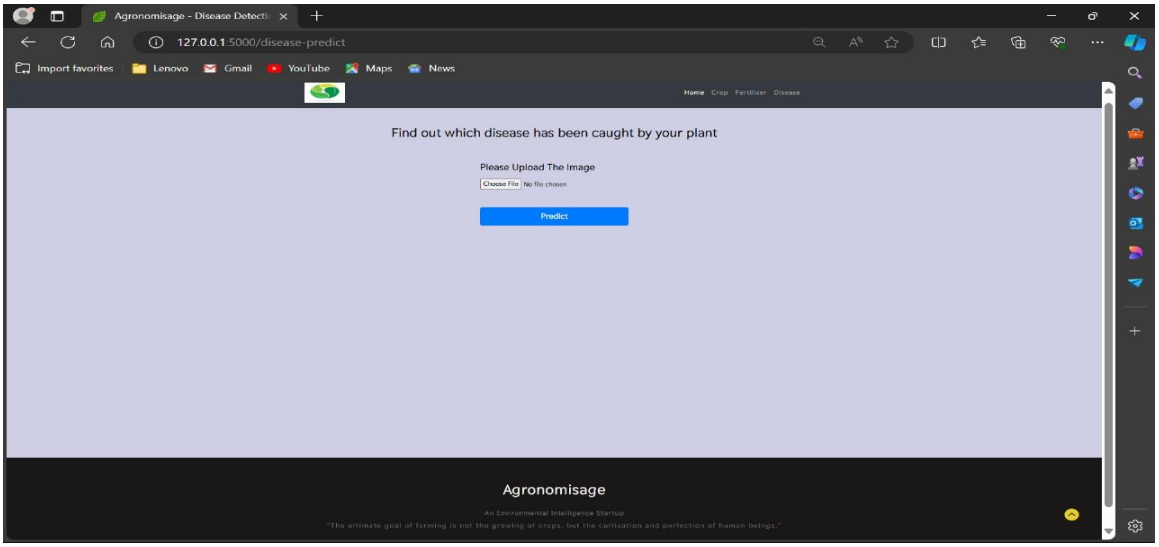

Figure 5.3.4 Plant Disease Prediction Input

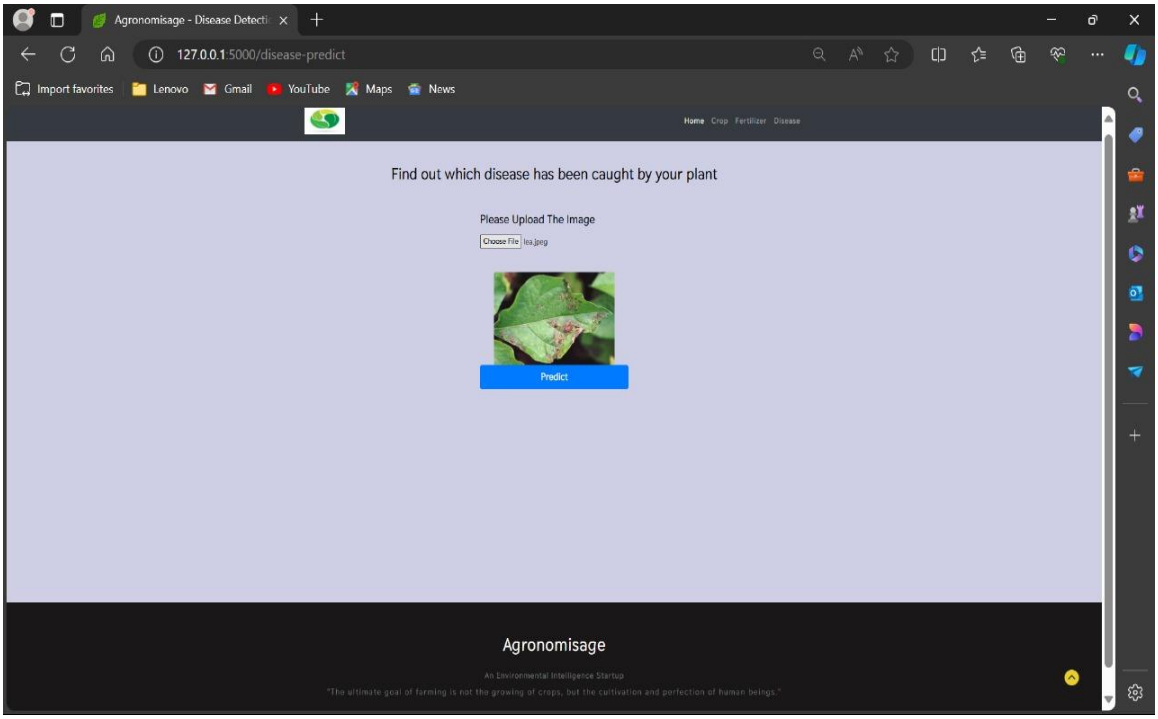

Figure 5.3.4 Plant Disease Prediction input

Output

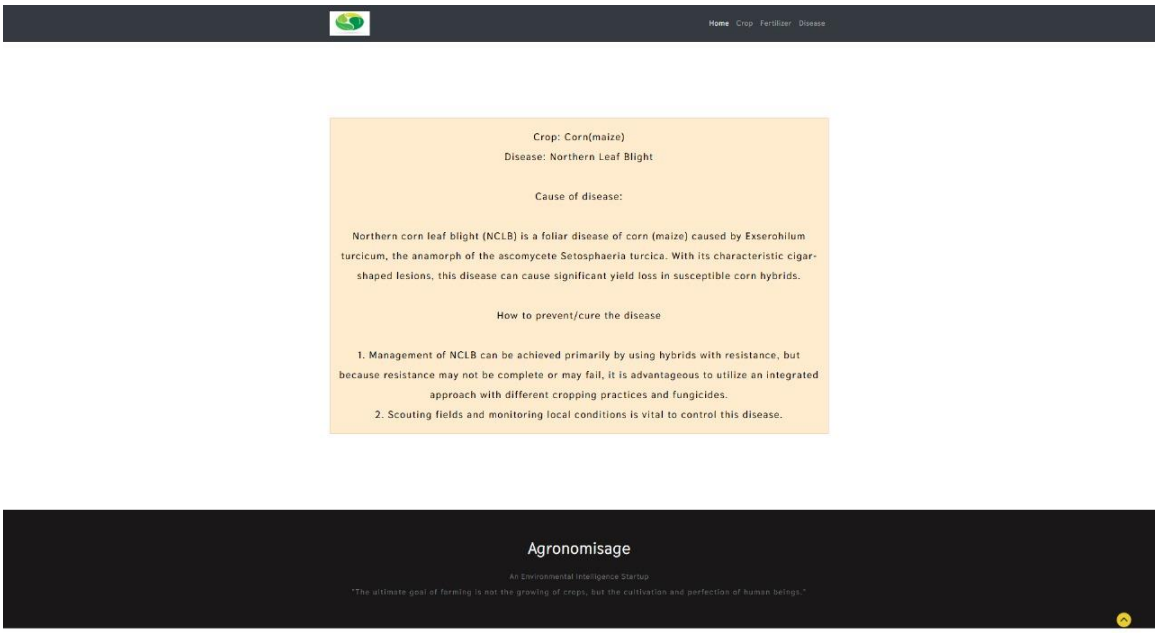

Figure 5.3.4.1 Plant Disease Prediction output

5.3.5 Crop Recommendation

Input

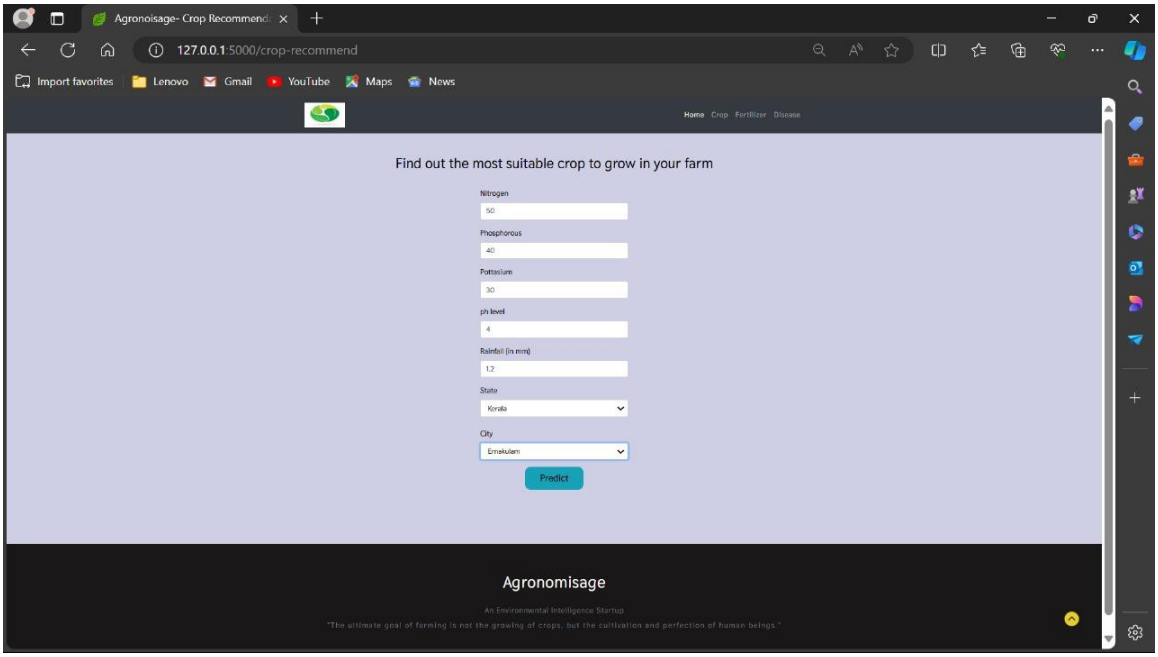

Figure 5.3.5 Crop Recommendation Input

Output

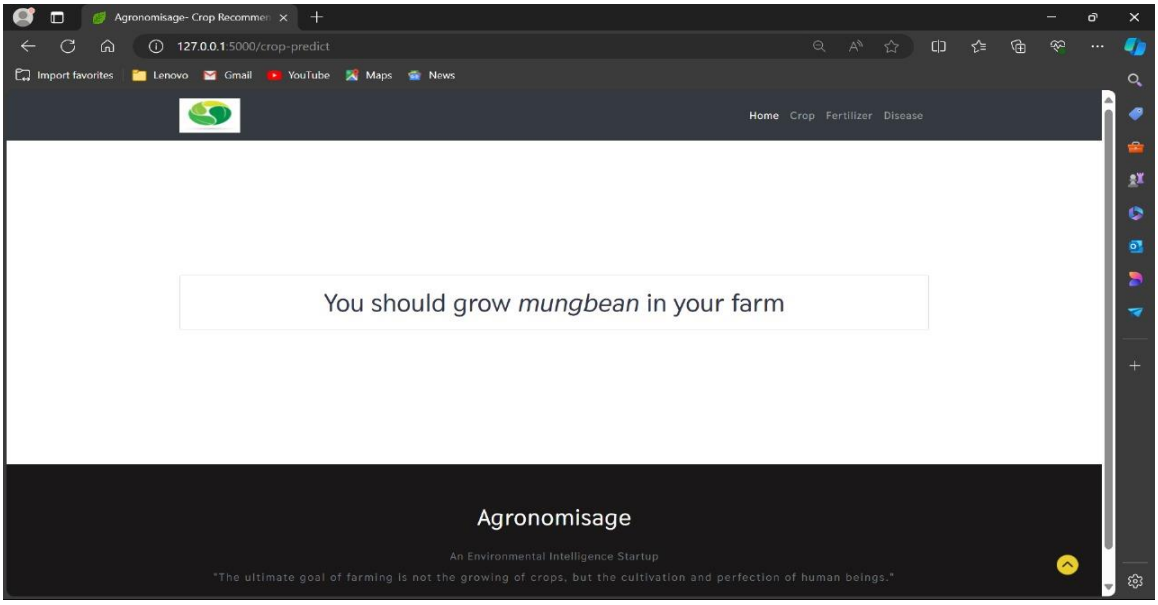

Figure 5.3.5.1 Crop Recommendation output

CHAPTER 6: JUSTIFICATION AND DISCUSSION OF THE RESULTS

6.1 Theoretical Justification

Overview on Machine Learning:

Machine learning is an application of artificial intelligence (AI) that gives systems the ability to automatically learn and evolve from experience without being specially programmed by the programmer. The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. The main aim of machine learning is to allow computers to learn automatically and adjust their actions to improve the accuracy and usefulness of the program, without any human intervention or assistance.

Traditional writing of programs for a computer can be defined as automating the procedures to be performed on input data in order to create output artifacts. Almost always, they are linear, procedural and logical. A traditional program is written in a programming language to some specification, and it has properties like:

- We know or can control the inputs to the program.
- We can specify how the program will achieve its goal.
- We can map out what decisions the program will make and under what conditions it makes them.
- Since we know the inputs as well as the expected outputs, we can be confident that the program will achieve its goal

Traditional programming works on the premise that, as long as we can define what a program needs to do, we are confident we can define how a program can achieve that goal. This is not always the case as sometimes, however, there are problems that you can represent in a computer that you cannot write a traditional program to solve. Such problems resist a procedural and logical solution. They have properties such as:

- The scope of all possible inputs is not known beforehand.

- You cannot specify how to achieve the goal of the program, only what that goal is.
- You cannot map out all the decisions the program will need to make to achieve its goal.
- You can collect only sample input data but not all possible input data for the program.

Machine learning techniques can be broadly categorized into the following types:

- **Supervised Learning:-** Supervised learning takes a set of feature/label pairs, called the training set. From this training set the system creates a generalised model of the relationship between the set of descriptive features and the target features in the form of a program that contains a set of rules. The objective is to use the output program produced to predict the label for a previously unseen, unlabeled input set of features, i.e. to predict the outcome for some new data. Data with known labels, which have not been included in the training set, are classified by the generated model and the results are compared to the known labels. This dataset is called the test set. The accuracy of the predictive model can then be calculated as the proportion of the correct predictions the model labeled out of the total number of instances in th test set.
- **Unsupervised Learning:-** Unsupervised learning takes a dataset of descriptive features without labels as a training set. In unsupervised learning, the algorithms are left to themselves to discover interesting structures in the data. The goal now is to create a model that finds some hidden structure in the dataset, such as natural clusters or associations. Unsupervised learning studies how systems can infer a function to describe a hidden structure from unlabeled data. The system does not figure out the right output, but it explores the data and can draw inferences from datasets to describe hidden structures from unlabeled data. Unsupervised learning can be used for clustering, which is used to discover any inherent grouping that are already present in the data. It can also be used for association problems, by creating rules based on the data and finding relationships or associations between them.

- **Semi supervised Learning: -** Semi-supervised machine learning falls somewhere in between supervised and unsupervised learning, since they use both labeled and unlabeled data for training typically a small amount of labeled data and a large amount of unlabeled data. The systems that use this method are able to considerably improve learning accuracy. Usually, semi- supervised learning is chosen when the acquired labeled data requires skilled and relevant resources in order to train it / learn from it. Otherwise, acquiring labeled data generally does not require additional resources.
- **Reinforcement Learning: -** Reinforcement machine learning algorithms is a learning method that interacts with its environment by producing actions and discovers errors or rewards. Machine learning algorithms are tools to automatically make decisions from data in order to achieve some over-arching goal or requirement. The promise of machine learning is that it can solve complex problems automatically, faster and more accurately than a manually specified solution, and at a larger scale. Over the past few decades, many machine learning algorithms have been developed by researchers.

6.2 Benefit Discussion of The Scheme

The Agronomisage Platform project offers a range of potential benefits for farmers, aiming to improve crop yields, farm efficiency, and potentially profits. Here's a detailed breakdown of its advantages:

Improved Decision-Making:

- **Data-Driven Recommendations**: Agronomisage Platform analyzes soil data, weather patterns, and even user-uploaded pictures. This data-driven approach helps farmers make informed decisions about crop selection, fertilizer use, and potential disease threats. Traditionally, these decisions might rely on experience or intuition, which can be prone to errors.
- **Reduced Risk**: By analyzing soil conditions and weather patterns, Agronomisage Platform can predict potential issues like crop

failure or disease outbreaks. This allows farmers to take preventive measures, reducing the risk of losing crops and income.

Increased Efficiency and Productivity:

- **Optimized Fertilizer Use:** Agronomisage Platform recommends the specific type and amount of fertilizer needed based on soil analysis. This prevents over-fertilization, which is wasteful and can harm the environment, while ensuring crops get the nutrients they need to thrive.
- **Targeted Crop Selection:** Agronomisage Platform suggests crops that are most likely to succeed based on specific conditions. This reduces the risk of planting crops that won't grow well in the local environment, leading to wasted resources and effort.

Potential for Increased Profits:

- **Improved Yields:** By recommending the right crops and optimizing fertilizer use, Agronomisage Platform can help farmers achieve higher crop yields.
- **Reduced Costs:** Optimized fertilizer use and reduced risk of crop failure can potentially lead to lower farm operating costs.
- **Higher Quality Crops:** Early detection of plant diseases through image analysis can help take timely action and potentially maintain higher crop quality.

Additional Benefits:

- **Accessibility of Knowledge:** Agronomisage Platform can be a valuable tool for new or less experienced farmers by providing them with data-driven insights that might otherwise require extensive knowledge or experience.
- **Sustainability:** Through optimized fertilizer use, Agronomisage Platform can potentially contribute to more sustainable agricultural practices.

It's important to consider these points as well:

- **Project Stage:** Agronomisage Platform is still under development, so its accuracy and reliability might improve over time.
- **Data Quality:** The quality of recommendations heavily relies on the accuracy of the data entered by the user and the comprehensiveness of Agronomisage Platform's internal data sets.
- **Human Expertise:** Machine learning is a powerful tool, but it shouldn't replace the knowledge and experience of farmers. Agronomisage Platform should be seen as a valuable aid, not a complete substitute for human judgment.

Overall, the Agronomisage Platform project holds immense potential to benefit farmers by improving decision-making, increasing efficiency and productivity, and potentially leading to higher profits. As the project matures and data quality improves, it could become a critical tool for modern agriculture.

CHAPTER 7: CONCLUSION AND FUTURE ENHANCEMENT

7.1 Conclusion

This system helps the farmer to choose the right crop by providing insights that ordinary farmers don't keep track of thereby decreasing the chances of crop failure and increasing productivity. It also prevents them from incurring losses. The system can be extended to the web and can be accessed by millions of farmers across the country. We could achieve an accuracy of 98.88 percent from the neural network and an accuracy of 99.93 percent from the linear regression model

Crop Disease Prediction Module is worth noting that the approach presented here is not intended to replace existing solutions for disease diagnosis, but rather to supplement them. Laboratory tests are ultimately always more reliable than diagnoses based on visual symptoms alone, and oftentimes early-stage diagnosis via visual inspection alone is challenging. Nevertheless, given the expectation of more than 5 Billion smartphones in the world by 2020 of which almost a Billion in Africa (GSMA Intelligence, 2016) we do believe that the approach represents a viable additional method to help prevent yield loss.

What's more, in the future, image data from a smartphone may be supplemented with location and time information for additional improvements in accuracy. Last but not least, it would be prudent to keep in mind the stunning pace at which mobile technology has developed in the past few years, and will continue to do so. With ever improving number and quality of sensors on mobiles devices, we consider it likely that highly accurate diagnoses via the smartphone are only a question of time.

Further development is to integrate the crop recommendation system, Fertilizer

Recommendation, Crop Disease Predictor with another subsystem, yield predictor that would also provide the farmer an estimate of production if he plants the recommended crop.

7.2 Future Enhancement

- **Yield Forecasting:**
  - Implement a feature to forecast crop yields based on historical data and current conditions.
  - Provide farmers with insights into expected harvest quantities for better planning.
- **Weather Integration:**
  - Enhance the system by integrating real-time weather data.
  - Adjust recommendations based on current weather conditions to optimize farming strategies.
- **Mobile Application:**
  - Develop a mobile application for convenient access to Harvestify.
  - Allow farmers to input data, receive recommendations, and monitor their crops on the go.
- **Community Collaboration Platform:**
  - Create a platform for farmers to share experiences, challenges, and success stories.
  - Foster a collaborative environment for knowledge exchange within the agricultural community.
- **Continuous Model Training:***
  - Implement a mechanism for continuous model training with updated datasets.
  - Improve model accuracy over time by incorporating the latest agricultural data.
- Expand Crop Support:*
  - Include a broader range of crops to support diverse farming practices.
  - Ensure the system accommodates a variety of regional and seasonal crops.
- User Education Resources:*
  - Provide educational resources within the platform to help users understand the science behind recommendations.

- Offer tutorials and guides on interpreting soil data, understanding diseases, and implementing suggested practices.

- **Multi-language Support:**
  - Introduce support for multiple languages to cater to farmers worldwide.
  - Facilitate global adoption by overcoming language barriers.
- **Blockchain Integration for Data Security:***
  - Explore the integration of blockchain technology for enhanced security of user data.
  - Ensure transparency and integrity in data handling processes.

These future enhancements aim to further empower farmers, promote sustainable practices, and strengthen Harvestify's role in modernizing agriculture.

7.3 References

1 S. D. Khirade and A. B. Patil, "Plant Disease Detection Using Image Processing," 2015 International Conference on Computing Community Control and Automation, pages 768-771, doi: IC-CUBEA.2015.153, 10.1109

2 "Image-based Plant Diseases Detection Using Deep Learning," by A. V. Panchal,S. C. Patel, K. Bagya lakshmi, P. Kumar, I. Raza Khan, and M. Soni, Materials Today: Proceedings, 2021, https://doi.org/10.1016/j.matpr.2021.07.281, ISSN 2214-7853.

3 "ImageNet:" by J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, and Li Fei-Fei A huge scope various leveled picture information base", 2009 IEEE Meeting on PC Vision and Example Acknowledgment, 2009, pp. 248-255, doi: 10.1109/CVPR.2009.5206848.

4 Russakovskyetal., " ImageNet Large Scale Visual Recognition Challenge," International Journal of Computer Vision, 115(3), pp. 211–252, 2015. https://doi.org/10.1007/s11263-015-0816-y.

5 George A. Mill operator, "WordNet: A Lexical Database for English," ACM Communications, Vol. 38, No. 11: 39-41.

6 "ImageNet Classification with Deep Convolutional Neural Networks," in Advances in Neural Information Processing Systems 25 (NIPS 2012), by A. Krizhevsky, I. Sutskever, and G. E. Hinton.

7 “Very Deep Convolutional Networks for Large-Scale Image Recognition,” K. Simonyan and A. Zisserman, 2014, arXiv 1409.1556.

8 "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770-778, doi: 10.1109/CVPR.2016.90.

9 Howard and others, MobileNets: Productive Convolutional Brain Organizations for Versatile Vision Applications", 2017, arXiv 1512.03385

10 M. Tan, Q. Le,(2019), "EfficientNet: Reexamining Model Scaling for Convolutional Brain Organizations", Procedures of the 36th Global Meeting on AI, in Procedures of AI Exploration, 2019, 97:6105-6114.

11 "An open access repository of images on plant health to enable the development of mobile disease diagnostics," by D. P. Hughes and M. Salathe, 2015, arXiv 1511.08060.

12 SP Mohanty, DP Hughes DP, M. Salath e', "Involving Profound Learning for Picture Based Plant Infection Recognition", Front. Plant Phys. 2016, 7:1419. doi: 10.3389/fpls.2016.01419.

13 "Plant Disease Detection and Classification by Deep Learning," in Plants, 2019, 8, no. 11: 468. https://doi.org/10.3390/plants8110468.

14 "Plant diseases and pests detection based on deep learning:" by J. Liu and X. Wang a review,” Plant Methods 17, 22 https://doi.org/10.1186/s13007-021-00722-9.

15 "Image-Based Detection of Plant Diseases:" by R. U. Khan, K. Khan, W. Albattah, and A. M. Qamar. From Classical Machine Learning to the Journey of Deep Learning,” in Wireless Communications and Mobile Computing, vol. 2021, Article ID 5541859, 13 pages, 2021. https://doi.org/10.1155/2021/5541859.

16 Computers and Electronics in Agriculture, Volume 161, 2019, Pages 272-279, ISSN 0168-1699, https://doi.org/10.1016/j.compag.2018.03.032, “A comparative study of fine-tuning deep learning models for plant disease identification,” by E. C. Too, L. Yujian, S. Njuki, and L. Yingchun.

17 S. M. Pande, P. K. Ramesh, A. Anmol, B. R. Aishwarya, K. Hilla, and K. Shaurya, "Crop Recommender System Using Machine Learning Approach," 2021,5th

International Conference on Computing Methodologies and Communication (ICCMC), pages 1066-1071, doi: 10.1109/ICCMC51019.2021.9418351.

18 "Crop recommendation system for precision agriculture," in S. Pudumalar, E. Ramanujam, R. H. Rajashree, C. Kavya, T. Kiruthika, and J. Nisha's 2016 Eighth International Conference on Advanced Computing (ICoAC), 2017, pp. 32-36, doi: 10.1109/ICoAC.2017.7951740.

19 "Design and Implementation of Fertilizer Recommendation System for Farmers," by K.S. Subramanian in 2020.

20 A. Pratap, R. Sebastian, N. Joseph, R.K. Eapen, S. Thomas, "Soil Fertil-ity Examination and Manure Suggestion Framework", Procedures of In-ternational Gathering on Headways in Processing and The board (ICACM) 2019, Accessible at SSRN: http://dx.doi.org/10.2139/ssrn.3446609 or https://ssrn.com/abstract=3446609

21 A. Palaniraj, A. S. Balamurugan, R. D. Prasad, and P. Pradeep, "Crop and Fertilizer Recommendation System using Machine Learning," International Research Journal of Engineering and Technology (IRJET) 2021, Volume 08, Issue 04

22 "Agro based crop and fertilizer recommendation system using machine learning," in European Journal of Molecular & Clinical Medicine, 7, 4, 2020, 2043-2051, by G. Preethi, P. Rathi, S. M. Sanjula, S. D. Lalitha, and B. V. Bindhu.

23 "Why Should I Trust You?" by C. Guestrin, S. Singh, and M. T. Ribeiro: In the proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16), the article titled "Explaining the Predictions of Any Classifier" appears. New York, NY, USA: Association for Computing Machinery, 2016, pp. 1135–1144. DOI: https://doi.org/10.1145/2939672.2939778.

24 "Grad-CAM::" by R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Visual Explanations from the Deep Net Using Gradient-Based Localization," in Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), 618-626, doi: 10.1109/ICCV.2017.74.