DHA Suffa University

Department of Computer Science



**CropSense (FY24P12)**

Final Year Project Report

Submitted by

Suffiyan Arshad DS211024

Shafia Memon DS211019

Noor Muhammad DS211013

Salman Yousuf DS211033

Supervisor  
Conrad Walter D’Silva

Co Supervisor

Raazia Sosan

In partial fulfilment of the requirements for the degree of

Bachelor of Science in Computer Science 2025

### Certificate of Approval

It is certified that the work presented in this report, entitled CropSensewas conducted by

Suffiyan Arshad ,Shafia Memon ,Noor Muhammad and Salman Yousaf under the supervision

of Conrad Walter D’Silva.

No part of this report has been submitted anywhere else for any other degree.

This report is submitted to the Department of Computer Science in partial fulfilment of the requirements for the degree of Bachelor of Science in Computer Science

at

DHA Suffa University (DSU)

Team Lead Name: Suffiyan Arshad Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Member 1 Name: Shafia Memon Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Member 2 Name: Noor Mohammad Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Member 3 Name: Salman Yousuf Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Supervisor Name: Conrad D’Silva Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

### Authors’ Declaration

We declare that this project report was carried out in accordance with the rules and regulations of the DHA Suffa University (DSU). The work is original except where indicated by special references in the text and no part of the report has been submitted for any other degree. The report has not been presented to any other University for examination.

Dated:

Authors Signatures:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Suffiyan Arshad

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Shafia Memon

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Noor Mohammad

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Salman Yousuf

**Plagiarism Undertaking**

We, Suffiyan Arshad, Shafia Memon, Noor Muhammad and Salman Yousaf, solemnly declare that the work presented in the Final Year Project Report titled CropSense has been carried out solely by ourselves with no significant help from any other person except few of those which are duly acknowledged. I confirm that no portion of our report has been plagiarized and any material used in the report from other sources is properly referenced.

Dated:

Authors Signatures:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Suffiyan Arshad

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Shafia Memon

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Noor Mohammad

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Salman Yousuf

### Acknowledgments

All praise and thanks to **Allah Almighty**, whose blessings and guidance gave us the strength, patience, and perseverance to complete our Final Year Project.

We are profoundly thankful to our **Supervisor, Sir Conrad D’Silva**, for his continuous support, expert advice, and thoughtful feedback throughout the project. His mentorship played a key role in helping us stay focused and strive for excellence.

Our sincere gratitude also goes to our **Co-Supervisor, Miss Raazia Sosan**, whose encouragement and timely suggestions greatly contributed to the progress and refinement of our work.

We would like to recognize and thank all our **Teachers** who have guided us during our academic journey. Their efforts in imparting knowledge laid a strong foundation for this project.

A special word of thanks to our **Parents**, whose unconditional support, motivation, and understanding kept us going through challenges and deadlines.

We are also grateful to our **Group Leader** for leading with dedication and responsibility, and to each **Team Member** for their valuable contributions, teamwork, and commitment to the success of this project.

Completing this project was truly a collaborative journey, and we are sincerely thankful to everyone who played a part in it.

### Document Information

Table 1: Document Information

|  |  |
| --- | --- |
| **Category** | **Information** |
| Customer | DHA Suffa University (DSU) |
| Project Title | CropSense |
| Document | Final Year Project Report |
| Document Version | 2.0 |
| Identifier | F24P12Final Report |
| Status | Final |
| Author(s) | Suffiyan Arshad, Shafia Memon, Noor Muhammad, Salman Yousuf |
| Approver(s) | Conrad Walter D’Silva ,Raazia Sosan |
| Issue Date |  |

**Definition of Terms, Acronyms, and Abbreviations**

Table 2: Definition of Terms, Acronyms, and Abbreviations

|  |  |
| --- | --- |
| **Term** | **DescripDescriptiontion** |
| **CropSense** | **An autonomous tractor system for smart farming** |
| Tractor Interface Module | Handles real-time data collection from tractors. |
| Obstacle Detection Module | Identifies obstacles like rocks and potholes using vision sensors. |
| Plant Health Detection Module | Utilizes YOLO-based deep learning for monitoring crop health. |
| Real-Time Updates | Continuous data sent and displayed in real-time. |
| Modular Design | A system with separate, reusable components. |
| Open-Source | Software that is accessible for modification and distribution. |
| API | Application Programming Interface for data exchange. |
| Django | A Python web framework for developing web applications. |
| YOLO | You Only Look Once – a deep learning-based object detection system. |
| Sensor Data | Information collected from sensors for monitoring and control. |
| Field Monitoring | Tracking the condition of crops and obstacles in a specific field. |

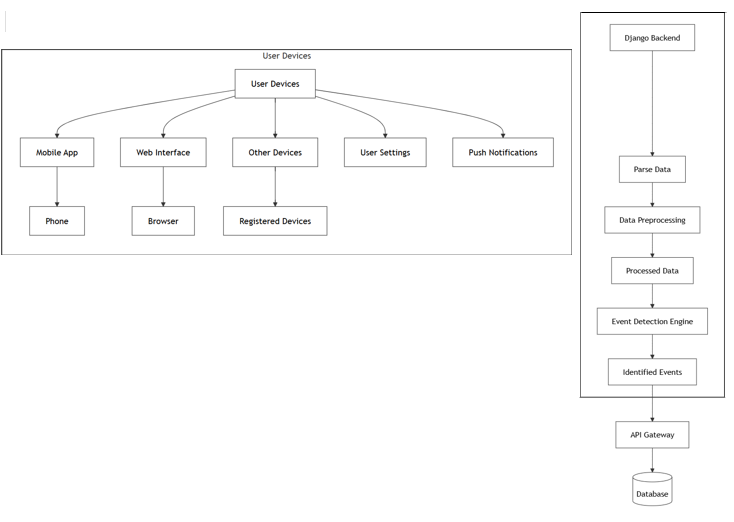
**FYP-II OPEN HOUSE EVALUATION COMMENTS**

|  |  |  |
| --- | --- | --- |
| **Evaluator Name** | **Evaluator Comments** | **Evaluator Signature** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

### Abstract

In an era of increasing vehicle connectivity and automation, real-time monitoring and communication between autonomous vehicles and their owners has become crucial for enhanced safety, security, and user experience. This project presents CropSense, a modular vehicle monitoring system designed to seamlessly integrate with existing autonomous vehicle architectures. The system employs a plug-and-play approach to provide owners with instantaneous updates about their vehicle's status, performance, and critical events along with the health of their plants.

The proposed solution leverages API-driven architecture and standardized communication protocols to ensure broad compatibility across different vehicle platforms and manufacturers. By implementing a modular design philosophy, CropSense minimizes integration constraints while maximizing system scalability and maintainability. This research focuses on developing efficient data transmission mechanisms, ensuring reliable real-time updates, and creating an intuitive interface for vehicle owners to monitor their autonomous vehicles remotely.



**Table of Contents**

[Certificate of Approval 2](#_1fob9te)

[Authors’ Declaration 3](#_3znysh7)

Acknowledgment [5](#_2et92p0)

[Document Information 6](#_tyjcwt)

[Abstract 7](#_3dy6vkm)

[Chapter 1 INTRODUCTION 10](#_2s8eyo1)

[Description about Project 10](#_17dp8vu)

[Details about the Domain 10](#_3rdcrjn)

[Relevant Background 10](#_lnxbz9)

[Chapter 2 RELEVANT BACKGROUND & DEFINITIONS 11](#_35nkun2)

[Chapter 3 LITERATURE REVIEW & RELATED WORK 12](#_1ksv4uv)

[Literature Review 12](#_44sinio)

[Related Work 12](#_2jxsxqh)

[Gap Analysis 12](#_z337ya)

[Chapter 4 METHODOLOGY 13](#_3j2qqm3)

[Software Engineering Methodology 13](#_1y810tw)

[Project Methodology](#_4i7ojhp) 14

[Chapter 5 EXPERIMENTAL EVALUATIONS & RESULTS 15](#_3whwml4)

[Evaluation Testbed 15](#_2bn6wsx)

[Results and Discussion 15](#_qsh70q)

[Chapter 6 CONCLUSION AND DISCUSSION 16](#_3as4poj)

[Limitations and Future Work 16](#_1pxezwc)

[Reasons for Failure – If Any 16](#_49x2ik5)

[REFERENCES 17](#_2p2csry)

[APPENDICES 18](#_147n2zr)

[A0. Copy of Project Registration Form 19](#_3o7alnk)

[A1a. Project Proposal and Vision Document 20](#_23ckvvd)

[A1b. Copy of Proposal Evaluation Comments by Jury 21](#_ihv636)

[A2. Requirement Specifications 22](#_32hioqz)

[A3. Design Specifications 23](#_1hmsyys)

[A4. Other Technical Detail Documents 24](#_41mghml)

[Test Cases Document 24](#_2grqrue)

[UI/UX Detail Document 24](#_vx1227)

[Coding Standards Document 24](#_3fwokq0)

[Project Policy Document 24](#_1v1yuxt)

User Manua1 Document 24

[A5. Flyer & Poster Design 25](#_4f1mdlm)

[A6. Copy of Evaluation Comments 26](#_2u6wntf)

[Copy of Evaluation Comments by Supervisor for Project – I Mid Semester Evaluation 26](#_19c6y18)

[Copy of Evaluation Comments by Supervisor for Project – I End Semester Evaluation 27](#_3tbugp1)

[Copy of Evaluation Comments by Jury for Project – I End Semester Evaluation 28](#_28h4qwu)

[Copy of Evaluation Comments by Supervisor for Project – II Mid Semester Evaluation 29](#_nmf14n)

[Copy of Evaluation Comments by Jury for Project – II End Semester Evaluation 31](#_1mrcu09)

[A7. Meetings’ Minutes 32](#_46r0co2)

A8. Document Change Record 33

A9. Project Progress 34

[A10. Research Paper](#_4k668n3) 35

# CHAPTER 1

# INTRODUCTION

## Description of the Project

CropSense is an innovative smart farming solution that integrates computer vision, real-time data processing, and autonomous control to revolutionize the agricultural landscape. The system is designed to monitor tractor operations and crop health in real-time using a modular, API-driven architecture. CropSense enables autonomous tractors to detect and avoid field obstacles such as rocks, potholes, and debris, and simultaneously monitors plant health using deep learning algorithms like YOLO (You Only Look Once). The gathered data is visualized on a user-friendly mobile and web interface, providing actionable insights to farm owners. The platform is built with scalability in mind and leverages open-source tools such as Django and Python, making it cost-effective and easily deployable in rural and remote environments. By digitizing and automating key farm activities, CropSense aims to boost agricultural efficiency, reduce human labor, and improve decision-making processes.

## Details about the Domain

The CropSense project lies within the domain of **Precision Agriculture** and **Autonomous Farming Systems**, where technology is utilized to optimize agricultural productivity and sustainability. This domain includes the use of GPS, computer vision, IoT devices, drones, and autonomous vehicles to manage and monitor agricultural activities with high precision. Specifically, CropSense focuses on enhancing tractor automation and crop health surveillance—two pivotal aspects of modern farm management. The project addresses key challenges such as inconsistent field monitoring, delayed detection of crop diseases, and unstructured tractor usage, which are prevalent in traditional agricultural settings. By using data-driven and AI-based methodologies, CropSense contributes to the broader movement toward smart farming and sustainable agriculture.

## Relevant Background

In recent years, agriculture has begun embracing the Fourth Industrial Revolution, marked by the convergence of digital technologies with traditional farming. The rise of autonomous tractors and real-time monitoring tools has the potential to significantly improve crop yields and operational efficiency. However, these technologies often remain inaccessible to small and medium-sized farm owners, particularly in developing countries, due to high costs and complexity. Existing autonomous solutions tend to focus solely on navigation without integrating plant health monitoring or real-time feedback for users. Moreover, illiteracy among farmers often serves as a barrier to the adoption of such systems. CropSense seeks to fill these gaps by delivering an integrated, intuitive, and affordable system that not only supports tractor navigation but also monitors crop vitality and provides real-time updates on mobile devices, making it ideal for deployment in both technologically advanced and underdeveloped farming communities.

# CHAPTER 2

# RELEVANT BACKGROUND & DEFINITIONS

This project utilizes various interdisciplinary technologies and concepts. Some of the key terms include:

* **YOLO (You Only Look Once)**: A real-time object detection algorithm used for plant health analysis and obstacle detection.
* **Django**: A Python-based web development framework used to build the backend of the CropSense system.
* **Obstacle Detection Module**: A vision-based system that identifies and localizes objects such as rocks and potholes in the tractor’s path.
* **Plant Health Detection Module**: Uses AI models to classify plant health status and growth stages.
* **Real-Time Updates**: The live streaming of sensor data and insights to end users via an intuitive interface.
* **Modular Architecture**: A software design philosophy where components (e.g., obstacle detection, crop monitoring) are developed independently to support flexibility and future upgrades.

# CHAPTER 3

# LITERATURE REVIEW & RELATED WORK

## Literature Review

The literature review examines existing research and applications of autonomous systems in agriculture, particularly focusing on smart obstacle detection and plant health monitoring technologies. Studies on precision agriculture reveal that while some autonomous systems are available, most lack integration between tractor functionality and real-time crop health assessment. Our review highlights the current limitations in these areas, providing a foundation for designing a system that is modular, efficient, and capable of addressing the practical challenges in the field through improved real-time decision-making.

## Related Work

Several existing projects and systems have attempted to address automation in farming through technologies like autonomous tractors, drone surveillance, and sensor-based irrigation. However, most lack modularity, integration between tractor navigation and crop monitoring, and real-time responsiveness. For instance, proprietary systems by leading agricultural companies provide high precision but are often costly and not adaptable to small-scale operations. Other open-source alternatives focus on either navigation or crop monitoring—but not both. CropSense differentiates itself by combining both elements into one cohesive, plug-and-play solution. Furthermore, it places special emphasis on accessibility, ensuring that even semi-literate users in rural areas can effectively interact with the system through visual data and alerts.

## Gap Analysis

The agricultural technology market reveals a distinct gap between high-end, expensive autonomous systems and the needs of average farmers in developing regions. While some systems offer navigation, they don’t integrate real-time crop monitoring. Others provide plant analytics but lack compatibility with real-world machinery. Moreover, many systems assume a high level of digital literacy, which is not the case for all end users. CropSense addresses these gaps by delivering an end-to-end solution that is not only affordable and modular but also easy to understand and use. Its reliance on camera-based detection rather than complex sensor arrays reduces hardware dependency, and the use of mobile applications makes the system readily deployable.

# CHAPTER 4

# METHODOLOGY

## Software Engineering Methodology

The Agile methodology will be used for its flexibility and iterative development, allowing for continuous feedback and improvements throughout the project lifecycle. It promotes collaboration among team members and stakeholders, ensuring that the evolving needs of users are addressed effectively. Agile allows for adaptive planning and quick responses to changes, making it ideal for dynamic and complex projects. Regular sprint reviews and retrospectives help in identifying bottlenecks and improving team performance. Additionally, the use of Agile tools and practices like Scrum, user stories, and sprint backlogs will help maintain transparency and track progress efficiently.

## Project Methodology

The Agile methodology was adopted for the development of CropSense to accommodate iterative development and continuous feedback. The project was divided into multiple sprints, each focusing on specific modules such as UI design, API development, obstacle detection, and plant health analysis. The development team conducted multiple internal reviews and supervisor meetings to validate milestones and pivot when necessary. Each module was developed independently to maintain a modular architecture. Tools like Figma were used for UI prototyping, Django for the backend, and Python for integration with AI models. Testing was conducted using both static datasets and real-time video feeds to simulate real-world farm conditions. This agile and modular approach allowed for the continuous evolution of the system in response to practical challenges and stakeholder input.

# Chapter 5

# EXPERIMENTAL EVALUATIONS & RESULTS

## Evaluation Testbed

CropSense was evaluated through both simulated and semi-realistic environments. The testbed included video recordings from agricultural fields, annotated images of plant conditions, and pre-labeled obstacle scenarios. These inputs were used to train and validate the YOLO model for both plant health and obstacle detection. The processed outputs were then visualized via a mobile UI developed using Django. User feedback from prototype demonstrations was incorporated to improve UI elements and data clarity. Performance metrics such as detection accuracy, system latency, and UI responsiveness were measured and optimized to meet predefined benchmarks (e.g., obstacle detection at 95% accuracy and response times within 2–5 seconds).

## Results and Discussion

The final version of CropSense demonstrated high effectiveness in real-time field monitoring. The obstacle detection module achieved over 95% accuracy, while the plant health monitoring model provided useful classification of crop vitality in most scenarios. The average system response time remained under 5 seconds, allowing for timely user alerts and actions. Users appreciated the clean UI and mobile accessibility, though challenges were identified in the usability for completely illiterate users. This was mitigated by introducing graphical representations and planning future video tutorials. Additionally, replacing physical sensors with vision-based systems greatly improved scalability and reduced implementation costs. The project successfully validated the potential of combining modular design with AI and real-time processing for small and mid-sized farm automation.

# CHAPTER 6

# CONCLUSION AND DISCUSSION

## Limitations and Future Work

CropSense is an application that relies heavily on API’s and understanding of modern application structure. The structure involves the use of mobile phone by the end user- a farmer. These farmers that come from illiterate backgrounds will consider the use of CropSense application as a learning curve.

Secondly, not only the use of application but also the usage of the application, reading the graphs will be a bit difficult for the farmers. However, both of these things can be covered through user education and education videos.

It is also worth noticing that these problems will be faced only by farmers of the developing world that come from illiterate backgrounds. If CropSense is able to make a dent in the local market and expand overseas, where the literacy of farmers is higher, the problem automatically eradicates.

# APPENDICES

List of Appendices

A0. Copy of Project Registration Form

A1a. Project Proposal and Vision Document

A1b. Copy of Proposal Evaluation Comments by Jury

A2. Requirement Specifications

A3. Design Specifications

A4. Other Technical Details

Test cases

UI/UX Details

Coding Standards

Project Policy

A5. Flyer & Poster Design

A6. Copy of Evaluation Comments

Copy of Evaluation Comments by Supervisor for Project – I Mid Semester Evaluation

Copy of Evaluation Comments by Supervisor for Project – I End Semester Evaluation

Copy of Evaluation Comments by Jury for Project – I End Semester Evaluation

Copy of Evaluation Comments by Supervisor for Project – II Mid Semester Evaluation

Copy of Evaluation Comments by Jury for Project – II Mid Semester Evaluation

Copy of Evaluation Comments by Supervisor for Project – II End Semester Evaluation

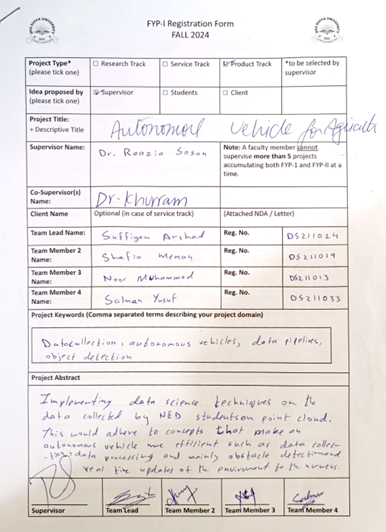
Copy of Evaluation Comments by Jury for Project – II End Semester Evaluation

A7. Meetings’ Minutes

A10. Any other

# A0. COPY OF PROJECT REGISTRATION FORM

A Photostat or scanned copy should be placed when submitting a document to Project Coordinator. (**Note**: Please remove this line when attach copy that is required)



# A1A. PROJECT PROPOSAL AND VISION DOCUMENT

Any standard template may be used, as per project need approved by Project Coordinator & Supervisor. Following is a suggestive outline. **Also, the same outline should be used for Project Proposal Presentation.**

**1 Introduction**

**1.1 Problem Statement**

Agriculture has seen a paradigm shift with the integration of technology, transforming traditional farming practices into smart, data-driven methodologies. Autonomous machinery, especially tractors, plays a crucial role in this transformation by improving efficiency, accuracy, and productivity in the field. These tractors offer immense potential to address challenges such as labor shortages, maximize operational hours, and ensure consistent fieldwork. However, their effectiveness relies heavily on their ability to perceive and react to environmental changes in real time. Our research focuses on developing advanced autonomous software for tractors, featuring real-time updates for enhanced operational control. A key component of the system is smart obstacle detection, which enables the tractor to identify and avoid various field obstructions, such as potholes, rocks, and debris. By navigating safely around these obstacles, the system minimizes downtime and reduces the risk of damage to machinery, thereby improving overall operational efficiency. Additionally, the system integrates plant health detection through advanced computer vision techniques. This functionality allows the tractor to assess the condition of crops, identifying signs of disease, nutrient deficiencies, or growth stages in real time. The software not only tracks the tractor’s operational metrics, such as working hours and field coverage, but also provides critical insights into crop health and field conditions. By delivering this information directly to the farm owner, the system facilitates timely interventions and data-driven decision-making, supporting sustainable and efficient agricultural practices.

**1.2 Project Motivation**

The agricultural sector is increasingly challenged by labor shortages, rising operational costs, and the need for greater productivity. Traditional farming methods are often labor-intensive and rely on manual oversight, which limits operational efficiency. Recent advancements in technology, especially autonomous systems, offer the potential to transform farming practices. By leveraging these technologies, our project seeks to create a sustainable solution that addresses the unique demands of modern agriculture, promoting efficiency, and supporting sustainable practices.

**1.3 Objectives**

The primary objectives of this project are to develop an autonomous tractor software that enhances field operations by offering real-time updates and control over tractor functions. The system will feature smart obstacle detection to navigate safely, plant health monitoring for early disease detection, and data-driven insights to optimize fieldwork. By achieving these goals, the system will reduce operational downtime, minimize machinery damage, and improve crop productivity, ultimately empowering farm owners with more efficient and sustainable farming practices.

**1.4 Literature Review**

The literature review examines existing research and applications of autonomous systems in agriculture, particularly focusing on smart obstacle detection and plant health monitoring technologies. Studies on precision agriculture reveal that while some autonomous systems are available, most lack integration between tractor functionality and real-time crop health assessment. Our review highlights the current limitations in these areas, providing a foundation for designing a system that is modular, efficient, and capable of addressing the practical challenges in the field through improved real-time decision-making.

**2 Project Vision**

**2.1 Business Case and SWOT Analysis**

**Business Case:**

Our solution addresses the gap in affordable, real-time vehicle monitoring systems, enhancing vehicle management for a broad user base.

**SWOT Analysis**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Strengths** | **Weaknesses** | **Opportunities** | **Threads** |
| 1 | Open-source, modular design, real-time updates. | Requires car owner technical knowledge, potential data privacy concerns. | Growing market demand for real-time monitoring, open-source collaboration. | Competing proprietary systems (e.g., Tesla), cybersecurity risks. |
| 2 |  |  |  |  |
| 3 |  |  |  |  |
| 4 |  |  |  |  |

**2.2 Background, Business Opportunity, and Customer Needs**

* **Owner/Users**: Farm owners seeking real-time updates on tractor operations and crop health.
* **Developers**: Engineers and data scientists developing and refining the autonomous tractor system.
* **Supervisors**: Academic and industry advisors overseeing the project for quality and guidance

|  |  |  |
| --- | --- | --- |
| **Type** | **Description** | **Responsibilities** |
| Owner/Users | Farm owners seeking real-time updates | * Provide feedback on system performance and usability |
| Developers: | Engineers and data scientists working on the system | - Develop, test, and improve system modules |
| Supervisors | Academic and industry advisors | Provide guidance, ensure quality, and validate outcomes |

**2.3 Business Objectives and Success Criteria**

* **Deployment of a Reliable Real-Time Monitoring System:** Ensure the system provides consistent, real-time updates on tractor operations and crop health.
* **User Engagement and Positive Feedback:** Achieve high user satisfaction through farm owner feedback and engagement**.**
* **High Usability with Modular Design**: Design a user-friendly, modular system that is adaptable to various agricultural settings**.**

**2.4 Project Risks and Risk Mitigation Plan**

**Project Risks:**

* **Data Security and Privacy**: Real-time monitoring of tractors involves collecting sensitive operational data, which could be at risk of unauthorized access.
* **Compatibility Limitations**: Ensuring compatibility across different tractor models may be challenging, as each model may have unique hardware and software requirements.
* **User Adoption and Technical Expertise**: Some farm owners may find the technology challenging, which could impact the adoption rate, especially among those unfamiliar with digital tools.
* **Cybersecurity Threats**: Open-source software can have vulnerabilities that may be exploited if not managed properly, risking system security

.

**Risk Mitigation Plan:**

* **Data Security**: Implement encrypted data transmission and secure data storage protocols.
* **Compatibility Testing**: Conduct extensive compatibility testing across various tractor models and offer user support for integration.
* **User Training**: Provide training materials and customer support to help farm owners understand and effectively use the system.
* **Regular Security Audits**: Conduct periodic security reviews and updates to identify and fix vulnerabilities in the open-source code

.

**2.5 Assumptions and Dependencies**

* **Assumptions:**
* The autonomous tractor will operate in fields with varying environmental conditions, and the system will adapt accordingly.
* Farm owners will have access to compatible hardware and internet connectivity for real-time updates.
* The computer vision models will perform accurately with minimal retraining using diverse datasets.
* Sensors and hardware components will function reliably under field conditions.

**Dependencies:**

* Availability of high-quality datasets for training obstacle detection and plant health models.
* Reliable hardware components, including sensors and GPS, for accurate data collection.
* Internet connectivity for real-time data transmission and updates.
* Continuous support from supervisors for technical guidance and validation.
* Collaboration with tractor manufacturers for hardware-software integration.

1. **Project Scope**

The scope of this project is to develop an autonomous, real-time update system for farm owners that is modular, accessible, and user-friendly. This includes

**3.1 In Scope**

* **Obstacle Detection**: Implementing object detection models to recognize obstructions such as rocks, debris, or equipment left in the field, enabling the tractor to navigate around them autonomously
* **Plant Health Detection**: Using computer vision to assess plant health, detect signs of distress or growth stages, and classify them into different categories to aid crop management.
* **Real-Time Updates**: Developing a system to send real-time updates to the farm owner regarding crop health, identified obstacles, and field conditions.

**3.2 Out of Scope**

* **Weather Monitoring**: Our research does not cover real-time weather data integration for autonomous tractor operations.
* **Soil Analysis**: The project does not include on-site soil quality analysis or nutrient deficiency detection in plants.
* **Advanced Crop Analytics**: While plant health detection is in scope, complex crop yield prediction and growth forecasting are beyond the current research focus.
* **Intervention Mechanisms**: The system will detect issues (such as obstacles or plant health problems) but will not provide or control physical interventions, such as applying pesticides or fertilizers.

**Integration with Other Farm Machinery**: Our software is designed solely for tractor-based automation and does not extend to other types of autonomous or semi-autonomous machinery on the farm

**4 Proposed Methodology**

**4.1 SDLC Approach (Waterfall/Agile/any model)**

The **Agile methodology** will be used for its flexibility and iterative development, allowing for continuous feedback and improvements throughout the project lifecycle.

**4.2 Team Role & responsibilities**

Conrad D’ Silva- Project Supervisor

Dr Raazia Sosan- Project Co-supervisor

Suffiyan Arshad - Team Leader

Responsibilities –Django App development

Shafia Memon - Team Member

Responsibilities – API Development

Noor - Team Member

Responsibilities – Data Processing and cleaning

Salman - Team Member

Responsibilities – Front-end and UI Development

**Supervisor:**

Oversee project alignment and deliverables.

**Co-supervisor:**

Assist with technical guidance.

**Team Leader:**

Coordinate team tasks and development.

**Team Members:**

Handle coding, testing, and UI design

**4.3 Requirement Development**

**Data Collection:**Gather requirements through online research competitor analysis to understand key needs and industry standards.

**Analysis and Design:**Use insights from data collection to design system architecture and define core features that align with user needs**.**

**Development and Implementation:**Build each system component as a modular unit, with ongoing real-time testing to confirm functionality and reliability**.**

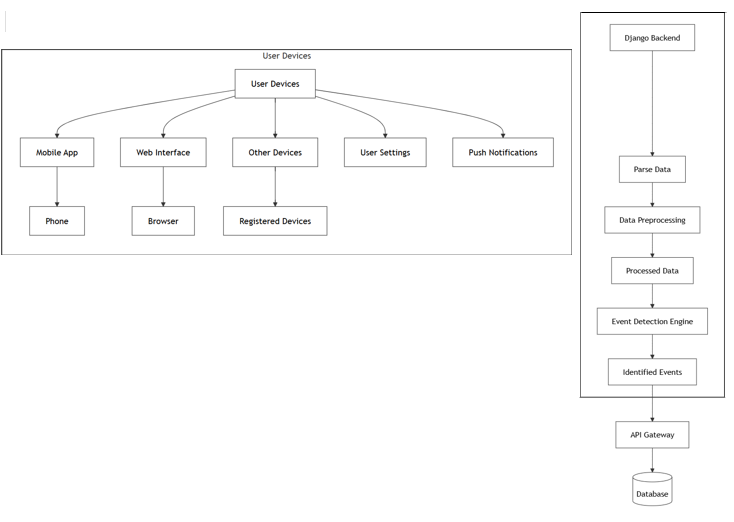
**Testing:**Perform comprehensive system testing and user acceptance testing to ensure performance, usability, and alignment with requirements.

**4.4 High-level Architecture / Design**

**Modular Architecture**: Each system component (e.g., obstacle detection, plant health monitoring) functions as an independent, easily upgradable module.

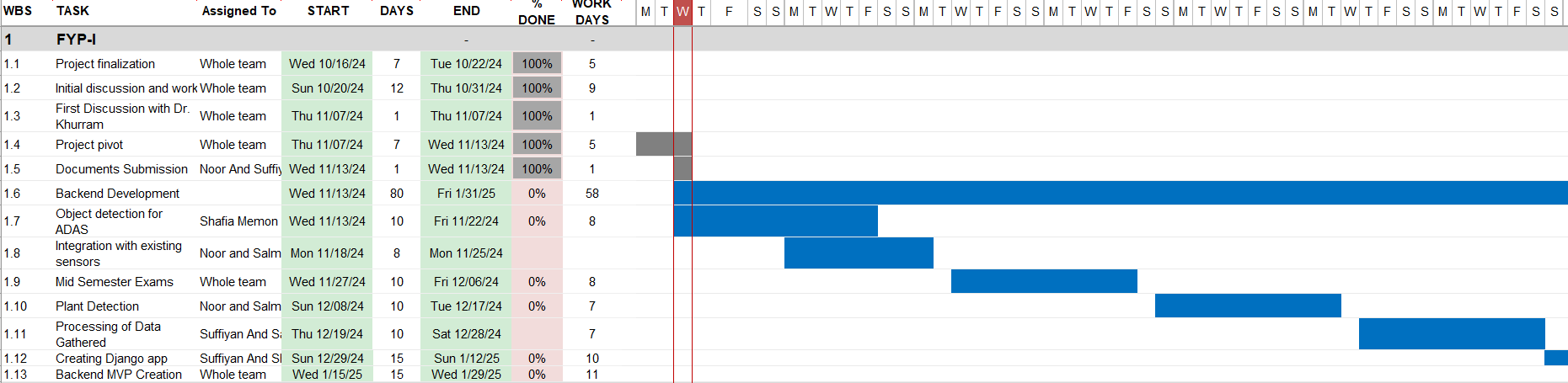
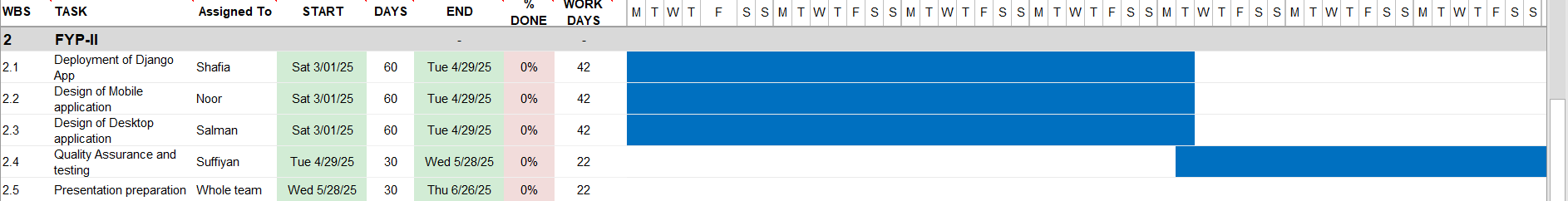
**Flexible application: By the use of APIs, the software will be accessible by any vehicle with an internet connection and the required sensors.**

**UI Integration**: User-friendly interface components seamlessly integrated with the backend API for real-time updates and data visualization.



**5 Project Planning**

**5.1 Gantt Chart**

Incorporate a timeline for each stage: data collection, development, testing, final deployment

**6 Project Requirements**

**6.1 Software tools requirements**

Django

Python

Figma(for UI)

**6.2 Hardware requirements**

Hardware requirements have already been fulfilled by the NEDUET Team

**7 Budget/Costing**

**7.1 Estimated Budgeted Cost - of the Project**

Total Hours - 640 hrs – (8 months, 5 hours per week, 4 group members)

Per Hour Rate – 62.5 Rs

Total Hours Rate – 40,000 Rs

Hardware Cost – 0 Rs

Other resources (Commute to NEDUET) -30,000 Rs

T**otal cost Rs. – 70,000 Rs**

**8 Project Deliverables**

**8.1 Phase I - Alpha Prototype**

Basic UI and real-time data capture.

**8.2 Phase II - Beta Prototype**

Enhanced features and broader compatibility.

**8.3 Phase III - Release Candidate**

Almost complete version with user testing

**8.4 Phase IV - Final Product**

Fully functional with open-source deployment.

**9 Proposed GUI (Disposable Prototype)**

**10 Meetings held with supervisor and/or client.**

### ****Minutes of Meeting****

* We had our **first meeting** to discuss the project idea, where we finalized the initial concept and listed potential features and goals.
* In the **second meeting at NED University**, we discussed the project in detail with our co-supervisor Miss Raazia Sosan, including the key features, what would be in-scope and out-of-scope.
* After this, **Miss Raazia left the university**, and our new supervisor was assigned **Sir Conrad D’Silva**.

### ****Meetings with Sir Conrad D’Silva****

* **First meeting:** Introduced our project, shared the background, and aligned on expectations and upcoming deliverables.
* **Second meeting:** Presented progress and datasets; received feedback on improvements and next steps.
* **Third meeting:** Conducted a mid-stage review, discussed challenges, milestones, and implementation plan.  
  **Major Change:** Decided to **revert from using physical sensors to camera-based detection** for better scalability.
* **Fourth meeting:** Demonstrated a working version of the system; discussed performance optimization and UI improvements.  
  **Major Change:** Planned **use of APIs to connect with local databases** for real-time updates.
* **Fifth meeting:** Reviewed final deployment strategy and presentation details.  
  **Major Change:** Approved **creation of a centralized database** and **adoption of microservices architecture** for modular design.

### ****Group Discussions & Internal Decisions****

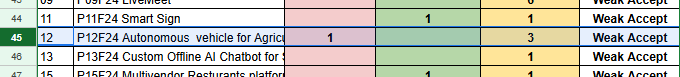
* Decided to **start using RGB analysis to detect photosynthesis levels in plants** to enhance crop monitoring accuracy.
* **Trained AI models** for object detection, crop health analysis, and obstacle recognition.

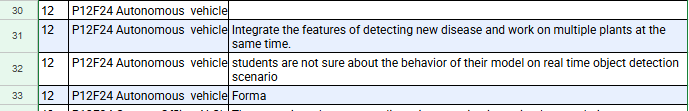
**11 Reference**

* <https://www.open-source-real-time.com/>
* [Tractor Technologies: Advances and Innovations](https://www.tractortechnology.com)
* [Precision Agriculture and Smart Farming Technologies](https://www.smartfarmingtech.com)
* [Smart Farming Solutions for the 21st Century](https://www.smartfarming.com)
* [Tractor Robotics and Autonomous Systems: Trends and Impact](https://www.tractorrobotics.com)

A1B. COPY OF PROPOSAL EVALUATION COMMENTS BY JURY

A Photostat or scanned copy should be placed when submitting a document to Project Coordinator. (**Note**: Please remove this line when attach copy that is required)





# A2. REQUIREMENT SPECIFICATIONS

Any standard template may be used, as per project need approved by Project Coordinator & Supervisor. Following is a suggestive outline.

**1. Introduction**

**1.1. Purpose of Document**

The purpose of this document is to specify the functional and non-functional requirements for the CropSense system. It serves as a guide for the development team and stakeholders to ensure the software meets desired outcomes in terms of functionality, reliability, and usability**.**

**1.2. Intended Audience**

Supervisors and Evaluators: To assess project progress and outcomes.

Development Team: To guide the software development process.

Farmers and End Users: To understand how the system will meet their needs.

Researchers and Technologists: To explore innovations in smart farming solutions

**2. Overall System Description**

**2.1. Project Background**

Agriculture is evolving with technology. CropSense introduces autonomous tractors to address challenges like labor shortages and improve efficiency through real-time updates, obstacle detection, and crop health monitoring

**2.2. Project Scope**

Obstacle detection, real-time updates, and plant health monitoring.

**2.3. Not In Scope**

Soil analysis and integration with external machinery

**2.4. Project Objectives**

* Provide real-time updates to farmers.
* Develop a modular, cost-effective system.
* Enhance productivity through smart farming

**2.5. Stakeholders**

## Farmers and agricultural workers.

## Developers and researchers in smart farming.

**2.6. Operating Environment**

**Field-based Environment**: The system operates primarily in agricultural fields with varying conditions such as soil types, weather, and terrain.

**Outdoor Setup**: The system is designed for outdoor operations where tractors move across large agricultural fields.

**Connectivity Requirements**: Requires stable internet connectivity for real-time data updates and cloud-based processing.

**2.7. System Constraints**

**Hardware Limitations:** Limited processing power on tractors may affect real-time data processing capabilities.

Sensors may have range and accuracy constraints, particularly in harsh environmental conditions (e.g., extreme weather, poor visibility).

**Cost and Budget:** High-cost of advanced sensors and computing hardware limits affordability for small and medium-scale farmers.

Requires reliable power supply for tractor-mounted systems.

**Scalability:**

The current design may face challenges in scaling up to large farm operations with multiple tractors.

**2.8. Assumptions & Dependencies**

**Internet Connectivity**: Assumed to be available and stable for real-time updates and cloud integration.

**Open-Source Tools**: Dependence on existing open-source solutions like Django and YOLO for development and plant health detection.

**Sensor Availability**: Assumes compatibility of sensors with tractors (e.g., GPS, obstacle detection, plant health sensors).

**Farmer Knowledge:** Assumes farmers have basic technical knowledge to interact with the system, particularly the UI.

**Tractor Compatibility**: Assumes tractors are equipped with necessary hardware for seamless integration with CropSense

**3. External Interface Requirements**

**3.1. Hardware Interfaces**

Sensors for obstacle detection and plant health monitoring.

Tractors equipped with GPS modules and connectivity.

**3.2. Software Interfaces**

Open-source frameworks like Django and YOLO.

**3.3. Communications Interfaces**

Internet connectivity for cloud updates.

APIs for real-time data transmission.

**4. Functional Requirements**

**4.1. Functional Hierarchy**

## System Functions

This section may contain

* end user, operator, support, or integration functions,
* performance requirements,
* design constraints,
* programming language, and
* interface requirements.

System functions are descriptions of what a system is supposed to do. They should be identified and listed in logical cohesive groups, with their category (priority) assigned. These system functions will be identified as a result of the requirement gathering process conducted with the client. However, in some cases, prior to the development of the Functional Specifications the requirements may already have been listed in a document: if this is so then a reference to the document may suffice.

To verify that some **X** is indeed a system function; it should make sense in the following sentence:

The system should do <**X**>

The table below gives an example of how system functions can be listed:

* The Functions column gives a brief one-line description of the required functionality.
* The Category column refers to the status of the functionality for the proposed system. The options for the Category are defined below.
* The Attribute column defines the system characteristics. The Details and Constraints column specifies the conditions within which the attribute is applicable. Section 1.12 defines the default Attributes and the related Constraints. In case, the default conditions are to be over-ridden then the conditions can be defined in this table.

Function Categories

|  |  |
| --- | --- |
| **Function Category** | **Meaning** |
| Evident | Should perform, and user should be cognizant that it is performed. |
| Hidden | Should perform, but not be visible to users. This is true of many underlying technical services, such as save information in a persistent storage mechanism. Hidden functions are often missed during the requirements gathering process. |
| Frill | Optional; adding it does not significantly affect cost or other functions. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ref # | Functions | Category | Attribute | Details & Boundary Constraints |
| R1.1 | Record real-time tractor status | Evident | System Response time | Response time within 6 seconds |
| R1.2 | Detect and avoid obstacles | Evident | Accuracy | Accuracy of 95% for obstacle detection |
| R1.3 | Monitor crop health | Evident | Real-Time Updates | YOLO-based algorithms for crop health detection |
| R1.4 | Provide live field data | Evident | Connectivity | Requires stable internet for real-time data transfer |
| R1.5 | Update user interface with alerts | Evident | User-Friendly UI | Simple UI for non-technical users |
| R1.6 | Store and retrieve historical data | Hidden | Data Storage | Cloud-based storage for data persistence |
| R1.7 | Push notifications to mobile app | Evident | Real-Time Alerts | Sends push notifications on critical events |
| R1.8 | Scalability of multiple tractors | Optional | Scalability | Limited scalability for large-scale farms |

**4.2. Use Cases**

**4.2.1 Farmer**

Role: Primary agricultural operator who directly interacts with the farming system and autonomous equipment. They are responsible for:

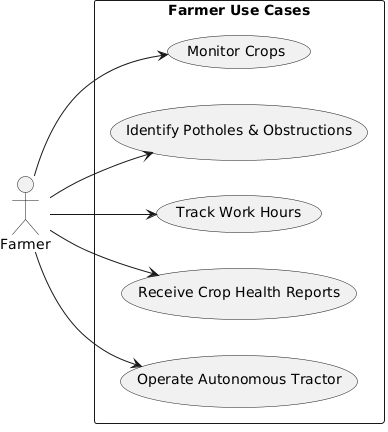
- Monitoring crops

- Identifying field hazards like potholes and obstructions

- Tracking their work hours

- Receiving and reviewing crop health reports

- Operating autonomous tractor equipment



**4.2.2. Admin**

Role: System administrator who manages the technical and operational aspects of the farming platform. Their responsibilities include:

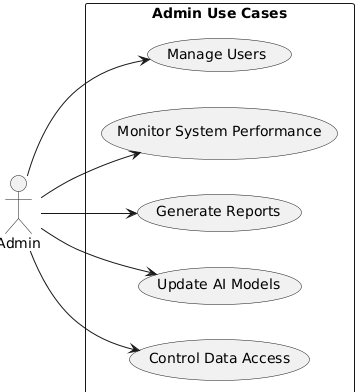
- User management and access control

- Monitoring overall system performance

- Generating system reports

- Updating and maintaining AI models

- Controlling data access and security



**4.2.3 Land Owner**

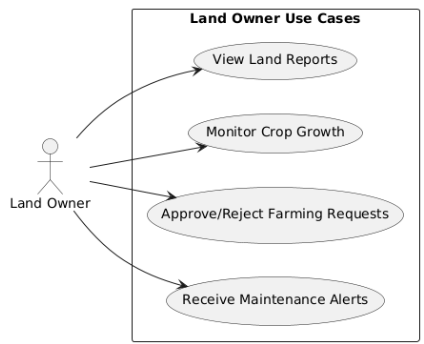
Role: Property owner who oversees the agricultural use of their land. Their primary responsibilities include:

- Viewing comprehensive land reports

- Monitoring crop growth progress

- Reviewing and approving/rejecting farming requests

- Receiving maintenance alerts about their property



**5. Non-functional Requirements**

**5.1. Performance Requirements**

Response Time: System should provide real-time updates (e.g., tractor status and obstacle alerts) within 2 seconds.

Data Processing: Process and send crop health data using YOLO-base algorithms within 5 seconds.

Concurrent User Load: Support simultaneous access for up to 10 users without performance degradation.

**5.2. Safety Requirements**

Obstacle Detection Accuracy: System should achieve at least 95% accuracy in detecting obstacles.

Field Navigation: Ensure the system can guide tractors safely around obstacles to prevent damage and downtime

**5.3. Security Requirements**

Data Encryption: Ensure all data (including sensor data and user info) is encrypted during transmission.

User Authentication: Secure login for farmers and authorized users.

Access Control: Restrict access to sensitive data and system functionalities.

**5.4. User Documentation**

System User Guide: A comprehensive guide for farmers on how to use the CropSense system.

API Documentation: Documentation for developers on system APIs and integration points.

Training Materials: Tutorials and FAQs for farmers to understand and operate the system effectively.

**6. References**

YOLO Documentation

Title: You Only Look Once (YOLO) Object Detection System

URL: https://pjreddie.com/darknet/yolo/

Description: Documentation on YOLO for object detection, relevant to plant health detection.

Django Framework

Title: Django Web Framework

URL: https://www.djangoproject.com/

Description: Official documentation for the Django framework used in the CropSense system development.

Open Source Real-Time Data Systems

Title: Open-Source Real-Time Data Processing

URL: https://www.open-source-real-time.com/

Description: Research and resources on open-source real-time data systems related to CropSense.

Precision Agriculture & Smart Farming

Title: Precision Agriculture and Smart Farming Solutions

URL: https://www.precisionagriculture.com/

Description: Resources and research on smart farming systems and technologies.

# A3. DESIGN SPECIFICATIONS

1. **Introduction**

**1.1 Purpose of Document**

The purpose of this document is to specify the functional and non-functional requirements for the CropSense system. It serves as a guide for the development team and stakeholders to ensure the software meets desired outcomes in terms of functionality, reliability, and usability**.**

**1.2 Intended Audience**

Supervisors and Evaluators: To assess project progress and outcomes.

Development Team: To guide the software development process.

Farmers and End Users: To understand how the system will meet their needs.

Researchers and Technologists: To explore innovations in smart farming solutions

**1.3 Project Overview**

CropSense is an advanced agricultural management system aimed at addressing critical challenges in modern farming, such as labor shortages and inefficiencies. The software integrates with autonomous tractors to deliver:

* **Real-time field updates** for monitoring farming activities.
* **Obstacle detection** for ensuring safe and efficient tractor navigation.
* **Crop health monitoring** to assess and improve yield through actionable insights.

The system follows a **modular design approach**, allowing seamless integration of additional features as farming needs evolve. Development will adhere to agile methodologies to ensure flexibility, iterative improvement, and rapid delivery of core functionalities.

By combining cutting-edge technology and user-friendly interfaces, CropSense aims to enhance productivity, optimize resource utilization, and promote sustainable farming practices.

**1.4 Scope**

**In-Scope:**

1. **Real-Time Updates:**
   * The system provides farm owners with real-time information about tractor operations, such as working hours, field coverage, and status updates.
2. **Obstacle Detection:**
   * Obstacle detection helps the tractor identify objects like holes, rocks, and other barriers. Using advanced tools, the system ensures safe navigation and smooth operation without delays.
3. **Plant Detection:**
   * The system can classify plants based on their shape, color, and size using deep learning algorithms.
   * It supports crop protection by enabling farmers to make informed decisions using YOLO (You Only Look Once) technology.

**Out-of-Scope:**

1. Weather monitoring.
2. Soil analysis.
3. Integration with other agricultural machinery.

**2 Design Considerations**

Design considerations focus on addressing key challenges and issues that need resolution before finalizing the system design. This section lays the groundwork for creating a robust and efficient system architecture.

**Key Considerations:**

1. **Performance Optimization:** Ensuring the system operates efficiently with real-time updates and obstacle detection capabilities.
2. **Scalability:** Designing a modular architecture to accommodate future requirements and upgrades seamlessly.
3. **Reliability:** Focusing on consistent operations and minimizing downtime to ensure smooth farming activities.
4. **User Experience:** Developing a user-friendly interface to provide farm owners with easy access to critical data and insights.
5. **Integration:** Ensuring seamless compatibility with existing sensors and connectivity modules to maximize system efficiency.

**2.1 Assumptions and Dependencies**

The following assumptions and dependencies are specific to the design phase and impact the implementation of the system:

1. **Sensor Accuracy:** It is assumed that the sensors used for obstacle detection and plant health monitoring will provide accurate and reliable data for processing.
2. **Connectivity:** The system depends on a stable and secure communication network for real-time data transmission and updates.
3. **Hardware Compatibility:** The design assumes that all hardware components, such as sensors and connectivity modules, will seamlessly integrate with the software architecture.
4. **Software Environment:** The system's performance depends on the availability of required libraries and frameworks, such as Django and Python, in a stable development environment.
5. **Team Expertise:** The success of the design relies on team members effectively carrying out their assigned responsibilities, such as API development, front-end design, and data processing.

**2.2 Risks and Volatile Areas**

The primary risks that could impact the system design include evolving user requirements, technological advancements, and potential hardware or network failures. Changes in external factors, such as regulations or environmental conditions, may also affect the system. To address these, the design will incorporate flexibility for updates and provide contingency plans, including fallback mechanisms for critical failures and adaptability to new technologies.

**3 System Architecture**

# Overview of the System: Start by explaining the primary goal of the system and how it's divided into major components.

# Main Components/Subsystems:

# Tractor Control Subsystem: Controls the tractor's movement and navigation, ensuring safety and operational efficiency.

# Real-Time Data Processing Subsystem: Collects, processes, and analyzes data from sensors (e.g., GPS, cameras) to ensure the tractor operates effectively.

# Obstacle Detection Subsystem: Uses computer vision and sensors to identify and avoid obstacles in real-time.

# User Interface Subsystem: Provides a dashboard for farm owners to view the tractor’s status, field conditions, and plant health in real-time.

# Communication Subsystem: Manages data exchange between the tractor, the central server, and the user interface.

# Interaction Between Components: Briefly describe how these components communicate with each other. For example, the real-time data processing subsystem sends data to the user interface, while the tractor control subsystem receives inputs from the obstacle detection subsystem.

# Integration and Data Flow: Explain how the system components are integrated, the flow of data between them, and how they collaborate to make the system work efficiently.

**3.1 System Level Architecture**

#### ****System Decomposition****

* **Main Modules**:
  1. **Obstacle Detection Module**: Real-time detection of field obstacles like potholes, rocks, and debris.
  2. **Plant Health Analysis Module**: Monitors crop health using computer vision techniques to detect diseases and nutrient deficiencies.
  3. **Operational Insights Module**: Tracks tractor performance metrics, such as working hours, field coverage, and efficiency.
  4. **User Interface (UI)**:
     + **Mobile App**: Provides real-time updates and control options.
     + **Desktop App**: Displays advanced analytics and generates reports.
  5. **Sensors and Connectivity**:
     + Standardized sensors for data collection.
     + Connectivity modules for real-time data transfer to the user interface.

#### ****Relationships Between Elements****

* The **Obstacle Detection Module** collects real-time data from sensors and communicates with the **Operational Insights Module** to alert about field hazards.
* The **Plant Health Analysis Module** shares insights with the **Operational Insights Module**, which then sends summarized data to the user interface.
* Both **Mobile App** and **Desktop App** serve as the communication bridge between the system and the user.

#### ****Interfaces to External Systems****

* Sensors integrated on the tractor act as the primary input sources for obstacle and crop data.
* Real-time data is transmitted to user interfaces via connectivity modules.

#### ****Major Design Considerations****

* **Execution Environment**:
  + Embedded systems on the tractor for data collection and processing.
  + Mobile and desktop environments for user interfaces.
* **Error Handling**:
  + Automatic retries for sensor data errors.
  + Notifications sent to users in case of critical failures.
* **Scalability**:
  + Modular design ensures future integration of weather monitoring or soil analysis modules.

#### ****Global Design Strategies****

* **Real-Time Processing**:
  + Designed for low-latency updates using Django and Python-based architecture.
* **User-Friendly Design**:
  + Intuitive mobile and desktop apps for ease of use.
* **Modularity**:
  + Separate modules ensure system flexibility and scalability.

**3.2 Software Architecture**

**4 Design Strategy**

#### ****Future System Extension or Enhancement****

* **Strategy**:
  + The system is designed with a **modular architecture**. Each major component (e.g., Obstacle Detection, Plant Health Analysis, and Operational Insights) operates independently, making it easy to incorporate future features such as weather monitoring or soil analysis without disrupting the existing system.
* **Reasoning**:
  + This approach ensures that the system can scale seamlessly, allowing for future enhancements with minimal adjustments to the core structure.
* **Trade-offs**:
  + Initially, modular design may lead to longer development times as it requires clear interface definitions and precise separation of responsibilities.

#### ****System Reuse****

* **Strategy**:
  + Key components like **obstacle detection algorithms** and **data processing pipelines** are developed as reusable libraries.
* **Reasoning**:
  + This strategy minimizes development costs for future projects, as these components can be reused in similar systems, enhancing overall efficiency.
* **Trade-offs**:
  + Creating reusable components involves extra abstraction, which can increase initial development complexity.

#### ****User Interface Paradigms****

* **Strategy**:
  + The system adopts a **responsive design** for the user interface to ensure compatibility across mobile and desktop devices.
  + A focus on **real-time dashboards** for visualizing critical data such as tractor performance, obstacle alerts, and crop health.
* **Reasoning**:
  + This approach enhances usability by providing farm owners with immediate, easy access to important information across various platforms.
* **Trade-offs**:
  + Developing a cross-platform UI may require additional resources and expertise in both mobile and web development.

#### ****Data Management****

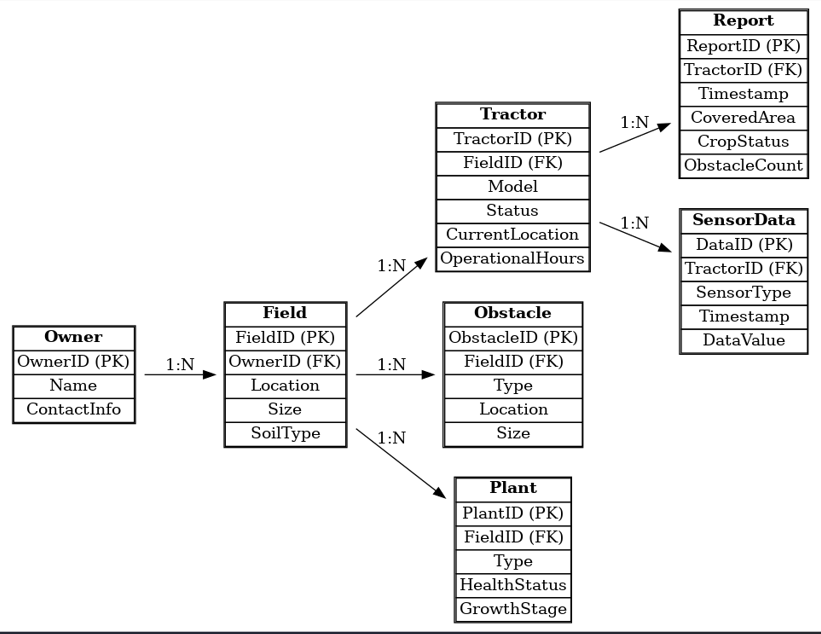
* **Storage**:
  + Persistent storage for tractor metrics and crop health data is handled via a **relational database**.
* **Distribution**:
  + Real-time updates are managed through **message queues** (e.g., Kafka), ensuring minimal latency in data transfer.
* **Persistence**:
  + Historical data is stored for future analysis, helping farm owners track trends over time.
* **Reasoning**:
  + The combination of real-time data handling and persistent storage enables both immediate decision-making and long-term insights.
* **Trade-offs**:
  + Implementing real-time data distribution can increase system complexity and demand more robust infrastructure.

#### ****Concurrency and Synchronization****

* **Strategy**:
  + **Multi-threading** is employed to handle concurrent data processing, while **synchronization mechanisms** (e.g., locks or queues) are used to prevent conflicts.
* **Reasoning**:
  + This ensures the system can efficiently manage tasks like obstacle detection, data storage, and UI updates in parallel, optimizing performance.
* **Trade-offs**:
  + The use of synchronization mechanisms can introduce latency and complexity, particularly when managing high data throughput.

**5.1 Detailed System Design**

**5.1.1 ER Diagram**



**5.1.2 Data Dictionary**

**5.1.2.1 Data 1**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Owner | | | | | | | |
| **Name** | | Owner | | | | | |
| **Alias** | | None | | | | | |
| **Where-used/how-used** | | Stores details of the owner of the fields and tractors. | | | | | |
| **Content description** | | Contains personal information of the owner. | | | | | |
|  | | | | | | | |
| **Column Name** | **Description** | | **Type** | **Length** | **Null able** | **Default Value** | **Key Type** |
| OwnerID] | Unique identifier for the owner | | Integer | 10 | No | None | PK |
| Name | |  | | --- | |  |   Name of the owner | | String | 100] | No | None |  |
| ContactInfo | Contact details of the owner | | String | 255 | Yes | None |  |

**5.1.2.2 Data 2**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Field | | | | | | | |
| **Name** | | Field | | | | | |
| **Alias** | | None | | | | | |
| **Where-used/how-used** | | Tracks details of the agricultural fields owned by the owner. | | | | | |
| **Content description** | | Field-specific data including size, soil type, and location. | | | | | |
|  | | | | | | | |
| **Column Name** | **Description** | | **Type** | **Length** | **Null able** | **Default Value** | **Key Type** |
| FieldID | Unique identifier for the field | | |  | | --- | |  |   Integer | 10 | No | None | Pk |
| OwnerID | |  | | --- | |  |  |  | | --- | | Identifier of the owner | | | Integer | 10 | No | None | Fk |
| Location | |  | | --- | |  |  |  | | --- | | Physical location of the field | | | String | 255 | Yes | None |  |
| Size | |  | | --- | |  |  |  | | --- | | Size of the field in acres | | | Float | 10.2 | Yes | None |  |
| SoilType | Type of soil in the field | | String | 100 | Yes | None |  |

**5.1.2.3 Data 3**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Tractor | | | | | | | |
| **Name** | | Tractor | | | | | |
| **Alias** | | None | | | | | |
| **Where-used/how-used** | | Tracks tractors assigned to specific fields and operational details. | | | | | |
| **Content description** | | Includes tractor model, status, and operational metrics. | | | | | |
|  | | | | | | | |
| **Column Name** | **Description** | | **Type** | **Length** | **Null able** | **Default Value** | **Key Type** |
| TractorID | Unique identifier for the owner | | Integer | 10 | No | None | PK |
| FieldID | |  | | --- | |  |   Identifier of the assigned field | | String | 10 | No | None | FK |
| Model | Model of the tractor | | String | 50 | Yes | None |  |
| Status | |  | | --- | |  |   Operational status of the tractor | | String | 20 | Yes | None |  |
| CurrentLocation | Current GPS location of the tractor | | String | 255 | Yes | None |  |
| OperationalHours | Total hours operated | | Integer | 10 | Yes | None |  |

**5.1.2.4 Data 4**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Obstacle | | | | | | | |
| **Name** | | Obstacle | | | | | |
| **Alias** | | None | | | | | |
| **Where-used/how-used** | | Tracks obstacles detected in fields. | | | | | |
| **Content description** | | Includes type, size, and location of the obstacles. | | | | | |
|  | | | | | | | |
| **Column Name** | **Description** | | **Type** | **Length** | **Null able** | **Default Value** | **Key Type** |
| ObstacleID | Unique identifier for the obstacle | | Integer | 10 | No | None | PK |
| FieldID | |  | | --- | |  |   Identifier of the field | | Integer | 10 | No | None | FK |
| Type | Type of obstacle | | String | 50 | Yes | None |  |
| Size | |  | | --- | |  |   Size of the Obstacle | | Float | 10.2 | Yes | None |  |
| Location | Location of Obstacle | | String | 255 | Yes | None |  |

**5.1.2.5 Data 5**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Plant | | | | | | | |
| **Name** | | Plant | | | | | |
| **Alias** | | None | | | | | |
| **Where-used/how-used** | | Tracks plants in fields and their health. | | | | | |
| **Content description** | | Includes type, health status, and growth stage of plants. | | | | | |
|  | | | | | | | |
| **Column Name** | **Description** | | **Type** | **Length** | **Null able** | **Default Value** | **Key Type** |
| PlantID | Unique identifier for the plant | | Integer | 10 | No | None | PK |
| FieldID | |  | | --- | |  |   Identifier of the field | | Integer | 10 | No | None | FK |
| Type | Type of Plant | | String | 50 | Yes | None |  |
| HealthStatus | |  | | --- | |  |   Health condition of the plant | | String | 50 | Yes | None |  |
| GrowthStage | Growth stage of the plant | | String | 50 | Yes | None |  |

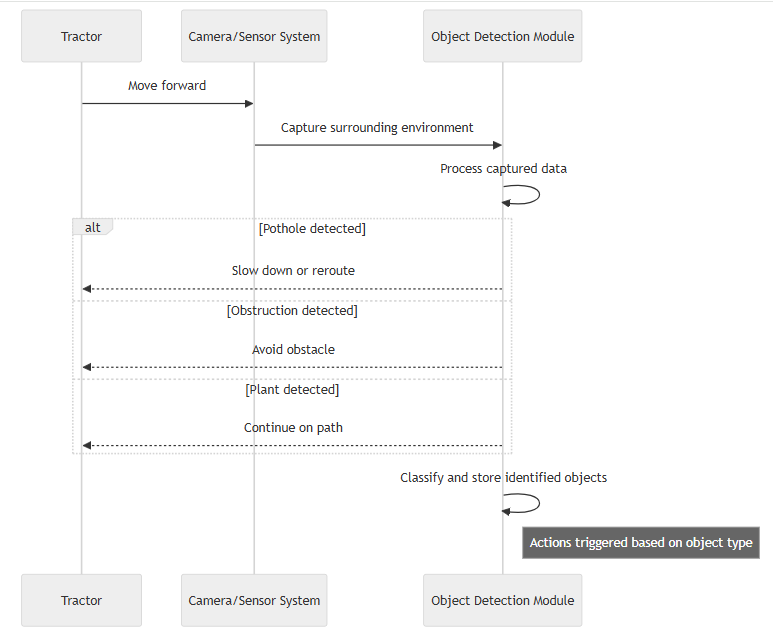
**5.1.2.6 Data 6**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Report | | | | | | | |
| **Name** | | Report | | | | | |
| **Alias** | | None | | | | | |
| **Where-used/how-used** | | Records operational reports for tractors. | | | | | |
| **Content description** | | Includes covered area, crop status, and obstacle counts. | | | | | |
|  | | | | | | | |
| **Column Name** | **Description** | | **Type** | **Length** | **Null able** | **Default Value** | **Key Type** |
| ReportID | Unique identifier for the report | | Integer | 10 | No | None | PK |
| TractorID | |  | | --- | |  |   Identifier of the tractor | | Integer | 10 | No | None | FK |
| Timestamp | Date and time of the report | | DataTime | - | No | None |  |
| CoveredArea | |  | | --- | |  |   Area covered in the report | | Float | 10.2 | Yes | None |  |
| CropStatus | Status of the crops in the report | | String | 50 | Yes | None |  |
| ObstacleCount | |  | | --- | |  |  |  | | --- | | Number of obstacles reported | | | Integer | 10 | Yes | None |  |

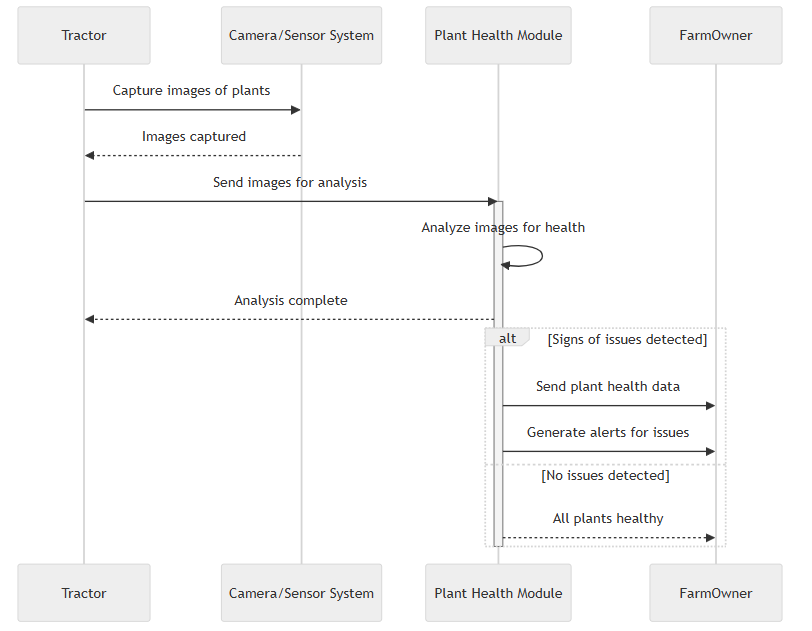
**5.2 Application Design**

**5.2.1 Sequence Diagram**

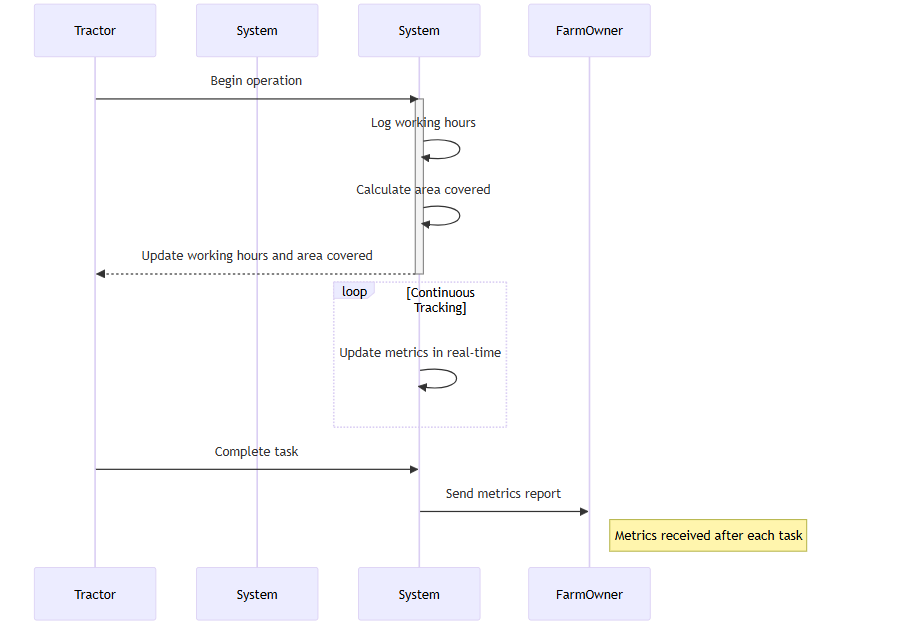
**5.2.1.1 <Sequence Diagram 1>**



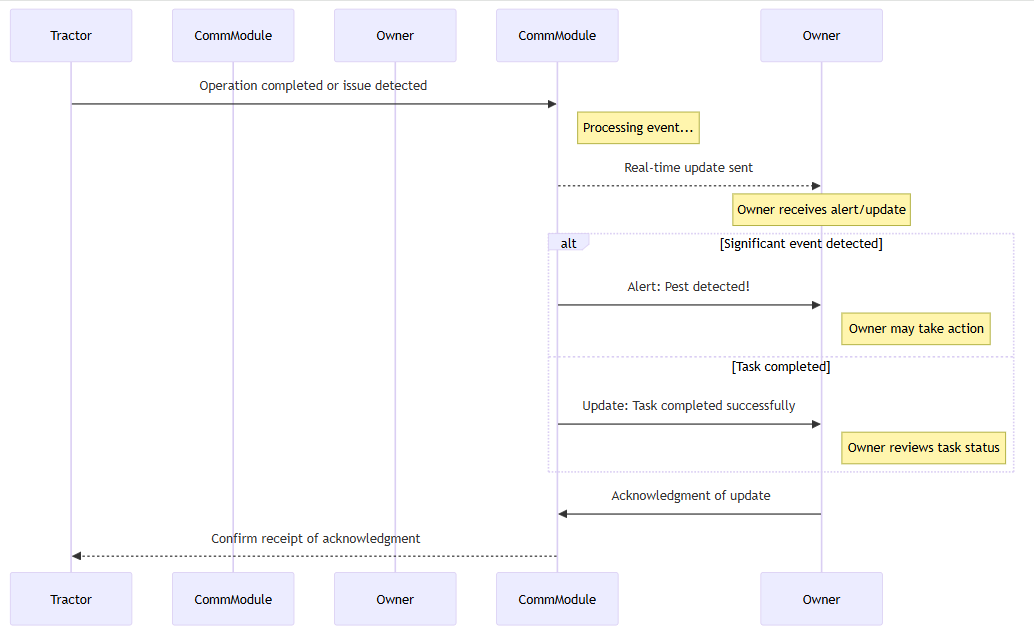
**5.2.1.2 <Sequence Diagram 2>**



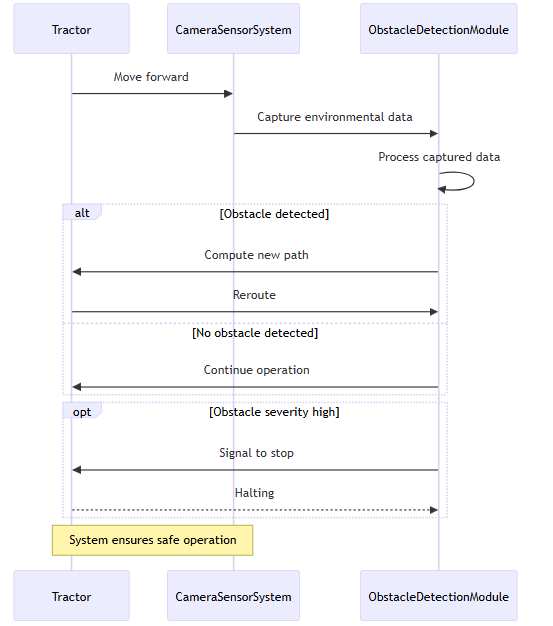
**5.2.1.3 <Sequence Diagram 3>**



**5.2.1.4 <Sequence Diagram 4>**

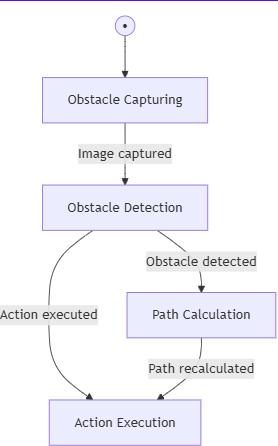


**5.2.1.5 <Sequence Diagram 5>**

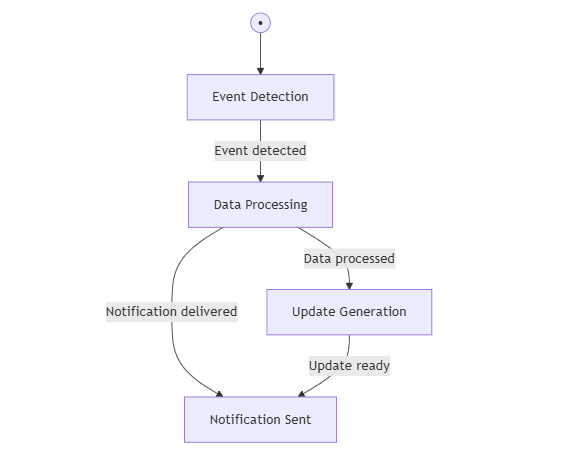


**5.2.2 State Diagram**

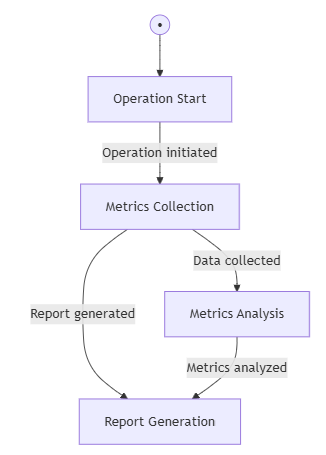
**5.2.2.1 <State Diagram 1>**



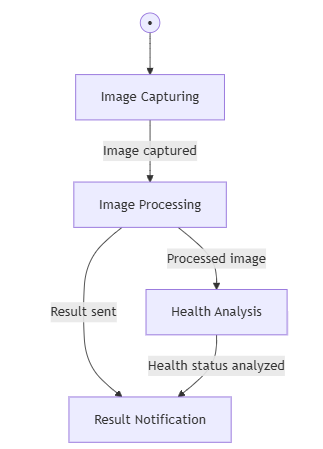
**5.2.2.2 <State Diagram 2>**



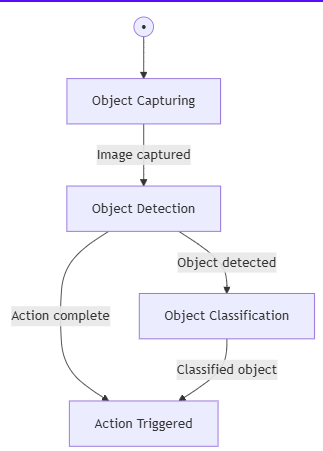
**5.2.2.3 <State Diagram 3>**



**5.2.2.4 <State Diagram 4>**



**5.2.2.5 <State Diagram 5>**



**6 References**

YOLO Documentation

Title: You Only Look Once (YOLO) Object Detection System

URL: https://pjreddie.com/darknet/yolo/

Description: Documentation on YOLO for object detection, relevant to plant health detection.

Django Framework

Title: Django Web Framework

URL: https://www.djangoproject.com/

Description: Official documentation for the Django framework used in the CropSense system development.

Open Source Real-Time Data Systems

Title: Open-Source Real-Time Data Processing

URL: https://www.open-source-real-time.com/

Description: Research and resources on open-source real-time data systems related to CropSense.

Precision Agriculture & Smart Farming

Title: Precision Agriculture and Smart Farming Solutions

URL: https://www.precisionagriculture.com/

Description: Resources and research on smart farming systems and technologies.

# A4. OTHER TECHNICAL DETAIL DOCUMENTS

## Test Cases Document

## Obstacle Detection Module

## Objective: To ensure real-time detection and classification of obstacles such as rocks, potholes, and debris.

* **Test Case 1**  
  *Input*: Image with a visible rock  
  *Expected Outcome*: Rock is detected with bounding box and classified  
  *Status*: Passed
* **Test Case 2**  
  *Input*: Image with a pothole partially visible  
  *Expected Outcome*: Pothole detected and location highlighted  
  *Status*: Passed
* **Test Case 3**  
  *Input*: Flat field with no obstacles  
  *Expected Outcome*: No alerts generated  
  *Status*: Passed

**Plant Health Monitoring Module**

**Objective**: To evaluate the accuracy of plant health classification using YOLO-based AI models.

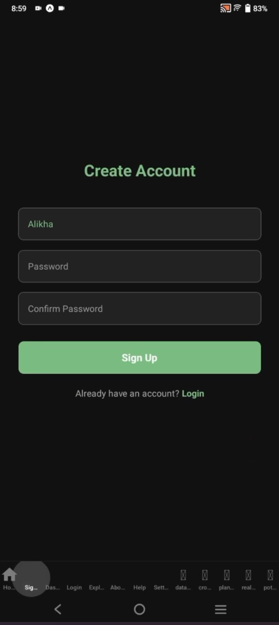
* **Test Case 1**  
  *Input*: Image of a healthy plant  
  *Expected Outcome*: "Healthy" label displayed with confidence above 90%  
  *Status*: Passed
* **Test Case 2**  
  *Input*: Image of a diseased plant (yellow leaves)  
  *Expected Outcome*: "Unhealthy" label displayed, correct classification  
  *Status*: Passed
* **Test Case 3**  
  *Input*: Blurry or low-resolution plant image  
  *Expected Outcome*: System either retries or flags image as low-confidence  
  *Status*: Passed

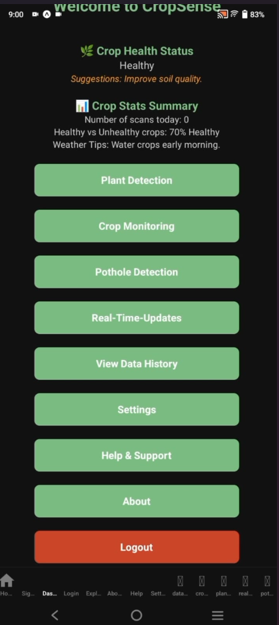
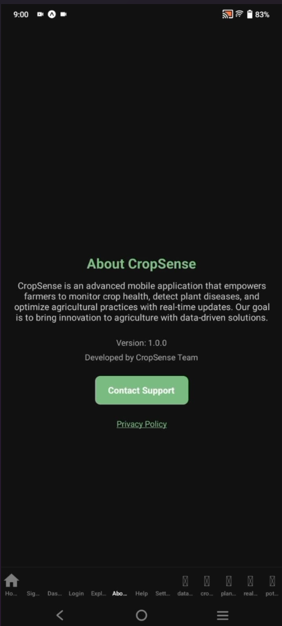
**Real-Time Update and Notification System**

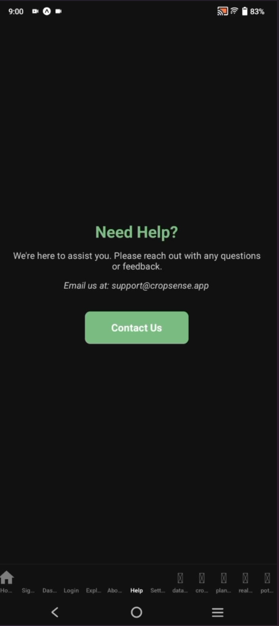
**Objective**: To verify the responsiveness and accuracy of real-time notifications.

* **Test Case 1**  
  *Trigger*: Detection of a rock during live operation  
  *Expected Outcome*: Mobile app shows pop-up alert within 5 seconds  
  *Status*: Passed
* **Test Case 2**  
  *Trigger*: Health status drops for a specific plant zone  
  *Expected Outcome*: Notification pushed and location marked on field map  
  *Status*: Passed

## UI/UX Detail Document







## Coding Standards Document

**Introduction**

This document outlines the coding standards followed during the development of the CropSense project to ensure consistency, maintainability, and collaboration across all team members. Given that the project utilizes Django (Python), HTML/CSS, and JavaScript, the standards align with widely accepted practices for these languages.

**Programming Languages Used**

* Python (for backend using Django)
* JavaScript (for interactive front-end components)
* HTML/CSS (for structure and styling of the user interface)

**General Guidelines**

* Code should be clean, readable, and well-documented.
* Each module should follow a modular structure, with separate files for logic, UI, and data handling.
* Use meaningful names for variables, functions, and classes to indicate their purpose clearly.

**Python (Django) Standards**

* Follow **PEP8** guidelines for Python code style.
* Use **snake\_case** for variable and function names.
* Use **CamelCase** for class names.
* Keep views, models, and forms separated within their respective Django apps.
* All business logic should be abstracted into services or model methods rather than being written in views.

**JavaScript Standards**

* Use **camelCase** for variables and function names.
* Use semicolons at the end of statements.
* Avoid writing inline JavaScript in HTML templates; use separate JS files.
* Comment all non-obvious logic to improve maintainability.

**HTML/CSS Standards**

* Use semantic HTML tags for structure.
* Use class selectors in CSS and avoid inline styles.
* Maintain consistency in indentation (2 or 4 spaces).
* Use comments to describe layout sections if the code is complex.

**Version Control**

* Git was used for source code version control.
* Each module was developed on a separate feature branch.
* Meaningful commit messages were enforced (e.g., Add plant health module, Fix obstacle detection bug).

**Security Practices**

* All forms and user input were validated to prevent security vulnerabilities.
* Django's built-in CSRF protection was enabled and utilized.
* Sensitive configuration such as API keys or credentials were stored in environment variables.

## Project Policy Document

## Introduction

## This document defines the project policies followed by the CropSense team during the development lifecycle to maintain discipline, collaboration, and progress tracking.

**Team Roles and Responsibilities**

* **Suffiyan Arshad** (Team Lead): Responsible for Django backend development and coordination among members.
* **Shafia Memon**: Responsible for API development and ensuring proper backend-to-frontend communication.
* **Noor Muhammad**: Managed data collection, cleaning, and processing for use in AI models.
* **Salman Yousuf**: Designed and developed the front-end interface using Figma and other tools.

**Supervisors**

* **Sir Conrad D’Silva**: Project Supervisor
* **Miss Raazia Sosan**: Co-Supervisor (initial phase)

**Development Methodology**

* Agile methodology was adopted for its flexibility and iterative approach.
* The project was divided into sprints focusing on individual modules such as UI, API, obstacle detection, and plant health monitoring.
* Internal reviews and supervisor meetings were conducted after each sprint for evaluation.

**Meeting Policy**

* Weekly or bi-weekly meetings were scheduled with the supervisor to discuss progress, address challenges, and finalize next steps.
* Major decisions (like switching from sensor-based to vision-based detection) were made in supervisor meetings.

**Code and Documentation Policy**

* Every piece of code was to be committed with proper documentation and tested before merging.
* All changes and versions were tracked using GitHub and manually recorded in the document change record section.

**Ethics and Contribution**

* All members were expected to contribute equally and honestly.
* No external code was used without proper citation.

Any third-party contributions were acknowledged, and plagiarism was strictly avoided.

## User Manual Document

## Overview

## CropSense is a smart farming platform that integrates autonomous tractor navigation, real-time crop health monitoring, and obstacle detection using a modular and user-friendly design. It is designed for ease of use, especially for semi-literate farmers and landowners.

**Getting Started**

* The user must install the CropSense mobile or desktop app.
* An account should be created using basic personal information.
* After login, users can link their tractor using a unique identification code.

**Using the Interface**

* **Real-Time Monitoring**:
  + The dashboard displays current tractor status, including working hours, location, and battery.
  + Obstacle alerts will flash immediately upon detection.
* **Crop Health Status**:
  + Health indicators are displayed using intuitive graphics (green for healthy, yellow for caution, red for poor health).
  + Users can tap on a section to view specific plant information.
* **Reports and History**:
  + Navigate to the "Reports" tab to view previous field reports.
  + Reports include crop status, covered area, and obstacle counts.

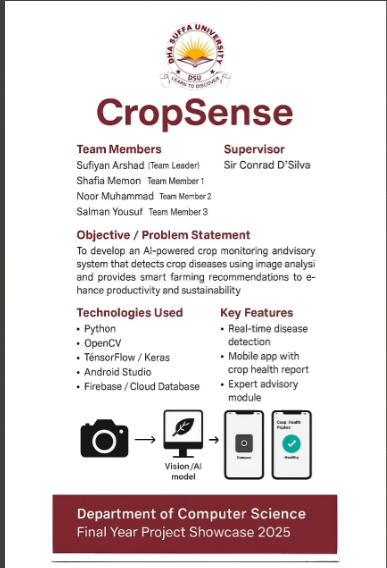
**Notifications**

* Critical updates like obstacle presence, poor plant health, or field anomalies are pushed as mobile alerts.

**Support and Help**

* A video tutorial is available for first-time users to understand interface elements.
* Help center can be reached at [**support@cropsense.com**](mailto:support@cropsense.com).
* FAQs and a visual guide are available within the app.

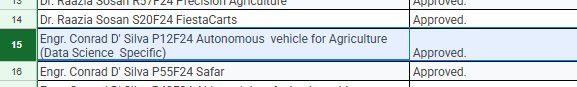
# A5. FLYER & POSTER DESIGN



# A6. COPY OF EVALUATION COMMENTS

## COPY OF EVALUATION COMMENTS BY SUPERVISOR FOR PROJECT – I MID SEMESTER EVALUATION

A Photostat or scanned copy should be placed when submitting document to Project Coordinator. (**Note**: Please remove this line when attach copy that is required)



## COPY OF EVALUATION COMMENTS BY SUPERVISOR FOR PROJECT – I END SEMESTER EVALUATION

A Photostat or scanned copy should be placed when submitting document to Project Coordinator. (**Note**: Please remove this line when attach copy that is required)

## COPY OF EVALUATION COMMENTS BY JURY FOR PROJECT – I END SEMESTER EVALUATION

A Photostat or scanned copy should be placed when submitting document to Project Coordinator. (**Note**: Please remove this line when attach copy that is required)

## COPY OF EVALUATION COMMENTS BY SUPERVISOR FOR PROJECT – II MID SEMESTER EVALUATION

A Photostat or scanned copy should be placed when submitting document to Project Coordinator. (**Note**: Please remove this line when attach copy that is required)

## COPY OF EVALUATION COMMENTS BY SUPERVISOR FOR PROJECT – II END SEMESTER EVALUATION

A Photostat or scanned copy should be placed when submitting document to Project Coordinator. (**Note**: Please remove this line when attach copy that is required)

# A7. MEETINGS’ MINUTES

### ****Minutes of Meeting****

* We had our **first meeting** to discuss the project idea, where we finalized the initial concept and listed potential features and goals.
* In the **second meeting at NED University**, we discussed the project in detail with our co-supervisor Miss Raazia Sosan, including the key features, what would be in-scope and out-of-scope.
* After this, **Miss Raazia left the university**, and our new supervisor was assigned **Sir Conrad D’Silva**.

### ****Meetings with Sir Conrad D’Silva****

* **First meeting:** Introduced our project, shared the background, and aligned on expectations and upcoming deliverables.
* **Second meeting:** Presented progress and datasets; received feedback on improvements and next steps.
* **Third meeting:** Conducted a mid-stage review, discussed challenges, milestones, and implementation plan.  
  **Major Change:** Decided to **revert from using physical sensors to camera-based detection** for better scalability.
* **Fourth meeting:** Demonstrated a working version of the system; discussed performance optimization and UI improvements.  
  **Major Change:** Planned **use of APIs to connect with local databases** for real-time updates.
* **Fifth meeting:** Reviewed final deployment strategy and presentation details.  
  **Major Change:** Approved **creation of a centralized database** and **adoption of microservices architecture** for modular design.

### ****Group Discussions & Internal Decisions****

* Decided to **start using RGB analysis to detect photosynthesis levels in plants** to enhance crop monitoring accuracy.
* **Trained AI models** for object detection, crop health analysis, and obstacle recognition.

**A8. DOCUMENT CHANGE RECORD**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Author** | **Change Details** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**A9. PROJECT PROGRESS**

Photostat of Incremental versions of Requirement Signoff sheet submitted to Project Coordinator. (**Note**: Please remove this line when attach copy that is required)

**A10. RESEARCH PAPER**