Software Engineering Department

Braude College

Capstone Project Phase B

An Internet of Things Sensor's System – Early Detection System for Avocado Tree Diseases

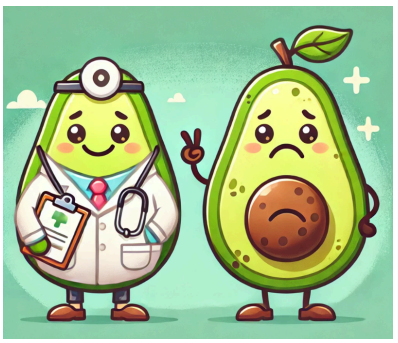
By:

Bahaldeen Swied

Rabea Lahham

Advisor:

Dr. Naomi Unkelos Shpigel



Project Code:

25-1-D-13

Phase A: https://github.com/oRABiiA/AvoTech

Phase B: https://github.com/oRABiiA/avocado-disease-detection-system

Table of Contents

Abstract 5
1. Introduction 5
2. Literature Review 6
2.1 Wireless Sensor Network (WSN) in Agriculture 6
2.2 IOT in Agriculture 6
2.3 Precision Agriculture Using IoT 8
2.3.1 Sensors 8
2.3.1.1 Air Humidity Sensor 8
2.3.1.2 Soil Moisture Sensor 8
2.3.1.3 Cameras for Early Disease Detection 8
2.4. Image Analysis in Agriculture 9
2.5 Avocado Diseases 9
2.5.1 Root rot 9
2.5.2 laurel wilt 10
3. Research 10
3.1 Scheduled Meetings 10
3.1.1 Meetings Summary 10
4. Engineering Process 11
4.1 Development Process 11
4.2 User-Centered Design Approach 11
4.3 Workflow 12
4.3.1 Phase A 12
4.3.2 Phase B 12
4.3.3 Pictures from farms and meetings 15
4.4 Solution 15
4.4.1 Algorithm 15
4.4.1.1 Data Collection 15
4.4.1.2 Image Analysis Using GPT-4o-mini API 15
4.4.1.3 Data Integration and Analysis 16
4.4.1.5 Generating Actionable Insights 16
4.4.1.6 Real-Time Updates 16
4.4.2 Engineering Design 18
4.5 Technologies Review 20
4.5.1 Hardware Side 20
4.5.1.1 Technical Details 20
4.5.2 Software Side 20
4.5.2.1 Client-Side Technologies 20
4.5.2.2 Server-Side Technologies 20
4.6 Architecture Diagram 21
4.6.1 Software Architecture 21
4.6.2 Hardware Architecture 21

5. Work Artifacts 22
5.1 Requirements 22
Table 2. Functional Requirements 22
Table 3. Non-Functional Requirements 23
5.2 Use Case Diagram 24
Table 4. Test Cases 27
6. User Guide 30
6.1 Login Page 30
6.2 Dashboard Screen: 31
6.3 Navigation Header: 33
6.4 Sidebar Menu: 34
6.5 Alert Popup: 35
6.6 About Page: 36
6.7 Contact Us: 37
6.8 FAQ Page: 38
6.9 Sensor Data Page: 39
6.10 Calendar Page: 40
6.11 Capture Live Photo Page: 41
6.12 Features Page: 42
7. Maintenance Guide 43
7.1 Installations 43
7.1.1 Prerequisites 43
7.1.2 Project Dependencies 43
7.1.2.1 Core Frameworks & Libraries: 43
7.1.2.2 State Management: 43
7.1.2.3 Firebase Integration: 43
7.1.2.4 Real-Time Communication: 43
7.1.2.5 Styling: 43
7.1.2.6 UI/UX & Animations: 44
7.1.2.7 Utilities: 44
7.1.2.8 Code Quality: 44
7.1.3 Developer Setup Instructions 44
7.1.3.1 Clone the repository 44
7.1.3.2 Install dependencies 44
7.1.3.3 Run development server 44
7.1.4 Firebase Databse 44
7.1.5 Running The Standalone Backend Server 46
7.1.5.1 AI_model.py 46
7.1.5.2 picture_request.py 47
7.1.5.3 Firebase AdminSDK 47
8. Evaluation 48
8.1 Reflection 1-Interview with Mr. Nir Yakoby -Farmer of Avocado field near Hamat Gader 48
8.2 Reflection 2-Interview with Mr. Foaad Hmeed and Mr. Saher Hmeed Veteran

farmers in Galil **48**
8.3 Areas for Enhancement **48**
8.3.1 Hamat Gader Farmer Feedback **48**
8.3.2 Galil Farmers Feedback **49**
8.4 SUS Survey **50**
9. Conclusion and Summary **53**
9.1 Key Findings and Achievements **53**
9.2 Challenges and Solutions **53**
9.3 Implications and Future Work **54**
9.4 Personal Learning Outcomes **54**
9.5 Final Thoughts **55**
10. Appendices **56**
10.1 Model Placement **56**
10.1.1 Soil and Air Humidity Sensors **56**
10.1.2 M5Stick C Plus 2 Controller **56**
10.1.3 Camera **58**
References **59**

Abstract

In recent years, the Internet of Things (IoT) has emerged as a transformative force in modern agriculture, revolutionizing data collection, analysis, and utilization across orchards and research environments. This project addresses a specific challenge avocado growers face: monitoring various environmental and plant health parameters to detect diseases early. The main obstacle lies in lacking a comprehensive system to effectively gather, analyze, and transmit live data from diverse orchard sensors.

To address this challenge, we developed an IoT-based sensor system aimed at detecting diseases in avocado trees. This solution establishes a cohesive platform for collecting, monitoring, and analyzing data, integrating a network of sensors that measure temperature, humidity, soil moisture, and leaf conditions within a cloud-connected environment. By centralizing and visualizing real-time sensor data, the system allows growers and researchers to monitor changes, swiftly recognize disease indicators, and obtain more profound insights into plant health and orchard conditions.

A key contribution of this project lies in its capacity for remote communication and data-driven decision-making, serving as a proof of concept for an integrated IoT framework in precision agriculture. Collaboration between local farmers and agricultural research institutions underpins this initiative, showcasing the potential for cooperative efforts in enhancing sustainable farming practices. Throughout the development process, we embraced Agile methodologies, ensuring flexibility and responsiveness to evolving requirements.

As one of the first targeted IoT applications for avocado disease detection within our institution, this project paves the way for further advancements in smart farming. Our work demonstrates the potential of inter-institutional partnerships in fostering innovation, setting a foundation for future developments in IoT-enabled agriculture, and reinforcing the critical role of data-driven strategies in modern crop management.

1. Introduction

Avocados have surged in global popularity due to their distinctive flavor and nutritional benefits, driving substantial growth in production across various regions. According to recent data [13], Mexico leads the world in avocado output with an impressive **2.53 million** tons, followed by Colombia and Peru. Notably, Israel also ranks among the top ten producers, contributing approximately **189.67 thousand** tons annually [13]. This significant share underscores the fruit's strategic importance in Israeli agriculture, where high-quality yields and efficient farming practices are paramount. However, as global demand for avocados intensifies, so do the challenges associated with diseases that can drastically impact crop health and productivity.

Avocado trees are highly susceptible to a variety of diseases, such as Armillaria Root Rot, Verticillium Wilt, and Phytophthora Canker, which can lead to significant yield losses and reduced fruit quality [14]. These diseases often exhibit overlapping symptoms or remain undetected until the tree's health deteriorates severely, complicating treatment efforts. Additionally, other factors such as nutritional deficiencies (e.g., boron and zinc), pest damage (mites, thrips, and the Melolonthidae complex), and abiotic stresses (sunburn, hailstone damage, and overripe fruits) further contribute to losses during preharvest and

harvest stages. also, there are some issues in packinghouses, These challenges collectively lead to an average economic loss of $80.29 USD per ton of produced fruit, with rejection rates of 5.78% on farms and 5.68% in packinghouses [1].
Traditional detection and management methods, which rely heavily on manual inspection, are time-consuming, error-prone, and often inadequate for early-stage identification of these issues.
To address these challenges, This project presents an IoT-based sensor system integrated with machine learning and image processing to enhance early disease detection in avocado farming. By collecting real-time environmental data and analyzing images of affected plant parts, the system provides accurate and timely insights. Machine learning models classify diseases based on patterns in the data, improving precision and reliability. This innovative approach promotes sustainable farming, reduces chemical use, increases productivity, and strengthens Israel's position in the global avocado market, showcasing the transformative potential of precision agriculture.

2. Literature Review

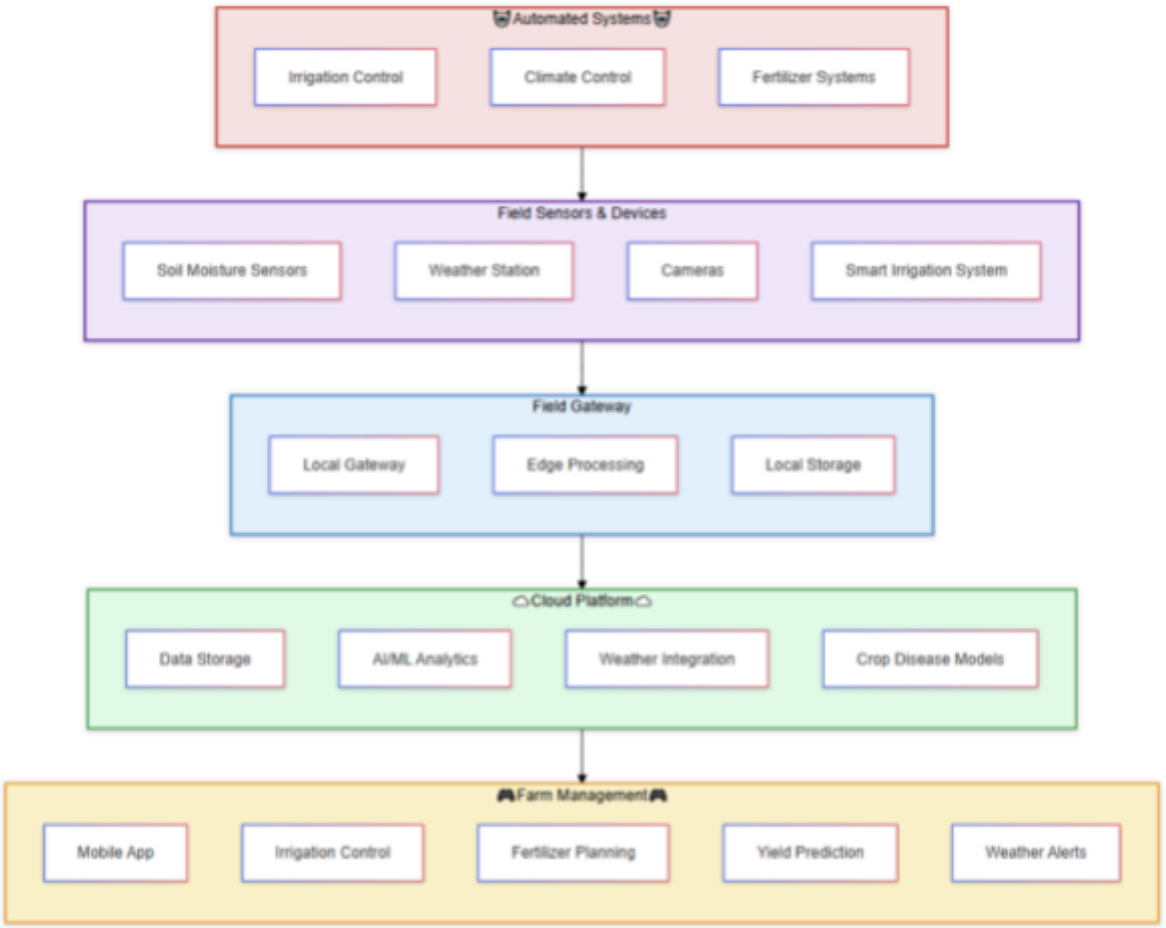
2.1 Wireless Sensor Network (WSN) in Agriculture

The integration of advanced technologies into agriculture has been a focus recently, aiming to enhance efficiency, sustainability, and productivity.
WSNs have emerged as a cornerstone for precision agriculture due to their adaptability, low cost, and energy efficiency. These networks facilitate real-time monitoring of environmental parameters like soil moisture, temperature, humidity, and pH, which are critical for optimizing agricultural practices. Studies have demonstrated the effectiveness of WSNs in monitoring agricultural fields through nodes powered by solar energy, which ensures uninterrupted communication and data transmission. For instance, the use of solar-powered routing protocols has proven to be energy-efficient and reliable, particularly in environments with limited infrastructure. By integrating advanced communication technologies, energy-efficient systems, and automation, smart agriculture can address pressing challenges such as resource scarcity and environmental sustainability [2].

2.2 IOT in Agriculture

In the field of agriculture, low-power wireless sensor networks (WSNs) and IoT platforms play a critical role in enabling real-time monitoring and management of environmental and crop-related factors. Programmable, open-source devices are often preferred for their flexibility in customizing sensor behavior, integrating new peripherals, and ensuring compatibility with the deployment environment. However, agricultural IoT deployments face unique challenges, such as interference caused by crop movement, environmental factors like humidity and temperature fluctuations, and power constraints in battery-operated nodes. Wireless communication protocols, such as Bluetooth, ZigBee, and LoRaWAN, are commonly used in agricultural IoT systems, each offering trade-offs in range, power consumption, and data transfer rates. Additionally, high humidity and extreme temperatures significantly impact communication reliability and sensor performance, highlighting the need for durable, low-power hardware and efficient communication protocols to ensure long-term

stability and accurate data collection in harsh agricultural environments. These insights guide the design of robust IoT systems tailored for precision agriculture, such as disease detection in avocado trees [3] Figure 1 explains the IoT Architecture.



(Figure 1. IoT Architecture)

2.3 Precision Agriculture Using IoT

2.3.1 Sensors

2.3.1.1 Air Humidity Sensor

The integration of intelligent humidity sensors in Wireless Sensor Networks (WSNs) offers a transformative approach to precision agriculture by optimizing data handling and energy efficiency. Traditional humidity sensors generate excessive data, increasing power consumption and processing demands. The intelligent sensor model, as proposed, integrates a conventional sensor with an embedded processor to process data using advanced algorithms. This approach reduces data collection by up to 50% and minimizes post-processing latency, thereby enhancing power efficiency and overall system performance [4]. Beyond improving operational efficiency, these sensors play a crucial role in disease detection by monitoring environmental conditions that influence pathogen growth. For example, they detect variations in soil and atmospheric moisture levels that favor fungal or bacterial diseases, enabling early warnings and localized control measures. By integrating sensor data with disease prediction models, farmers can proactively mitigate risks, optimize pesticide usage, and prevent outbreaks, making intelligent humidity sensors indispensable for sustainable farming and plant health management [4].

2.3.1.2 Soil Moisture Sensor

Soil moisture sensors play a vital role in modern agriculture by enabling efficient water management and supporting precision irrigation, ultimately conserving water and improving crop yields. These sensors provide real-time data about soil moisture levels, ensuring optimal irrigation scheduling and reducing overwatering, which can lead to plant diseases caused by excessive soil moisture. Various types of sensors are available, each with unique advantages. Besides improving irrigation practices, these sensors contribute to disease management by preventing conditions favorable for pathogen growth, such as excessive humidity in the root zone. Early detection of moisture imbalances helps in identifying potential disease outbreaks and maintaining soil health, demonstrating the sensors' dual role in enhancing productivity and mitigating risks [5].

2.3.1.3 Cameras for Early Disease Detection

There are several diseases the avocado plant can suffer from, such as Phytophthora root rot (Prr), and laurel wilt (Lw).
Utilizing cameras and classification methods, the system can do early-stage detection using a Machine Learning process.
Current detection methods, such as visual scouting, are labor-intensive, expensive, and prone to inaccuracies, especially during early, asymptomatic stages. Alternatives like microarray and DNA-based methods are accurate but costly and slow. Automated techniques using multispectral imaging and artificial intelligence offer a promising solution by leveraging spectral reflectance data to differentiate between healthy and stressed plants.

These methods have shown success in other crops, prompting their application in this study to develop a low-cost, efficient detection system for avocado diseases and nutrient deficiencies [6].

2.4. Image Analysis in Agriculture

We implemented a novel image analysis approach by integrating a regular camera with the ChatGPT API to detect early signs of diseases in avocado trees. Unlike previous plans involving local deep learning models, the current system relies on capturing images of avocado trees in the field and sending them to a backend server. From there, the images are forwarded to ChatGPT with the rest of the sensors data via a carefully designed prompt. ChatGPT analyzes the image and returns a response that classifies the tree as either healthy or infected, along explanation and tips that based on the trees image and sensors data with a natural-language.

This method offers several advantages. First, it eliminates the need for heavy computation on the edge device, which simplifies the hardware setup and reduces costs. Second, the textual feedback provided by ChatGPT is easy to understand, making it more accessible for farmers and agricultural staff who may not have technical expertise. The responses often include descriptions such as “the leaves appear discolored” or “branches show signs of wilting,” which helps build trust in the system’s diagnosis.

OpenAI reports that GPT-4o-mini performs well on visual understanding tasks, especially with clear and relevant prompts. Based on our development experience, the model provided consistent and reasonable answers when images were clear and well-lit.

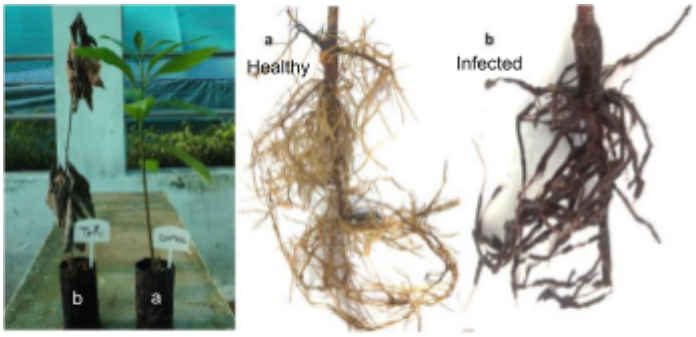
Reported performance: GPT-4o mini is better than other small models at reasoning tasks involving both text and vision, scoring 82.0% on MMLU, a textual intelligence and reasoning benchmark, as compared to 77.9% for Gemini Flash and 73.8% for Claude Haiku. [15]

By combining simple hardware with a powerful cloud-based AI model, our system presents a lightweight yet effective solution for disease detection in agriculture. It demonstrates how generative AI can be used not only for text but also for interpreting visual data in real-world field conditions.

2.5 Avocado Diseases

2.5.1 Root rot

Phytophthora root rot (PRR), is a critical issue in avocado cultivation, leading to root damage, reduced water and nutrient uptake, canopy decline, and tree mortality. The fungus thrives in poorly drained soils and has been reported globally, though its prevalence varies by region [8,9]. Traditional visual assessment of PRR severity is limited by assessor expertise and environmental factors, prompting the development of



(Figure 2. healthy root vs infected root [12])

digital methods. Image analysis using RGB photography and satellite imagery has shown strong correlations between canopy porosity and disease severity, offering more accurate and scalable assessment tools [8]. Figure 2 showcases PRR effects on avocado trees.

2.5.2 laurel wilt

Laurel Wilt (LW) is a lethal disease caused by the fungus Raffaelea lauricola, spread by the redbay ambrosia beetle. It disrupts tree vascular systems, causing rapid death. Since its U.S. discovery in 2002, LW has severely impacted avocado trees and other Lauraceae species, threatening Florida's $35 million avocado industry and altering ecosystems [10,11].

(Figure 3. laurel wilt infected stages [16])

Early detection through remote sensing techniques enables targeted control, while management includes tree removal, fungicides, and public awareness. However, challenges like beetle reproduction and the lack of biological controls persist. LW poses a global threat due to expanding beetle ranges and trade [10,11]. Figure 3 shows how Laurel wilt affects avocado trees.

3. Research

3.1 Scheduled Meetings

Throughout the project, we held several meetings with the students and Professor Uzzi Rozen from the Mechanical Engineering department. Regularly engaging with the mechanical engineering students was a key aspect of our work. During these meetings, we presented the progress made and the tasks completed within the given period. Additionally, these discussions helped clarify topics and address any uncertainties we had. After each meeting, we scheduled the next session and outlined the progress expected to be achieved by then.

3.1.1 Meetings Summary

Over the course of four meetings, we laid the foundation for our project and established a strong collaboration with the mechanical engineering team. The first meeting (20.11.2024) was held with Keren, the farm manager, where we discussed key challenges in agriculture and explored possible project directions. In the second meeting (27.11.2024), we evaluated proposed projects from mechanical engineering students via Zoom, assessing their compatibility with our goals. During the third meeting (18.12.2024) at the FAB Lab, we reviewed presentations and selected the project focused on early disease detection in avocado trees due to its preparedness and alignment with our vision. In the final meeting (8.1.2025), also held at the FAB Lab, we met with the selected team and their professor to discuss technical components, especially the sensors and camera, raising questions and

identifying potential integration issues to ensure smooth progress moving forward.

4. Engineering Process

4.1 Development Process

The development of our system followed an **Agile methodology**, allowing us to adapt to changes and incrementally build a scalable, user-friendly, and intelligent platform for early disease detection in avocado trees. The primary objective was to create a system that can collect real-time environmental data, analyze tree health through image interpretation, and provide actionable insights to farmers.

The **backend** was developed using **Python**, responsible for managing sensor data collection, image processing, API communication, and data integration. Our **frontend** was built with **Next.js**, ensuring a modern, fast, and responsive user experience. The system uses **Firebase** to store user data and analysis results, while **Mosquitto** (MQTT) facilitates real-time communication with field sensors.

The data collection process begins with a **regular camera** installed in the field, capturing two images daily — once in the morning and once in the afternoon (15:00). The server, located on-site, receives the images, resizes them to reduce latency and cost, and retrieves real-time data from the environmental sensors (e.g., temperature, humidity, soil moisture). This combined data package (image + sensor readings) is then sent to **GPT-4o-mini** through the ChatGPT API for analysis. The model returns a natural-language response indicating the health status of the tree along with personalized recommendations. This response is stored in Firebase and made available immediately on the system's web interface.

To enhance usability, the platform includes an **alert mechanism**: if the model detects an unhealthy tree, a notification is automatically triggered and sent to the farmer's dashboard. The recommendations vary depending on the tree’s condition. For healthy trees, the system provides general maintenance tips; for infected trees, the GPT model generates specific suggestions based on visible symptoms and sensor values.

While formal testing is still in progress, initial **manual testing** of AI responses and sensor readings confirmed the system's functionality. The web application currently supports **multiple farmers, multiple trees per user**, and real-time data visualization for the selected sensors. Work with the mechanical engineering team was limited to sensor hardware provision — all system design, software development, AI integration, and cloud infrastructure were implemented by our team.

This development process reflects a balance between field practicality and modern AI-driven agriculture, building a foundation for a scalable and intelligent crop management system.

4.2 User-Centered Design Approach

Our project adopts a **user-centered design (UCD)** approach, placing the needs and experiences of **avocado growers and agricultural managers** at the core of the development process. This methodology ensures that the system not only meets technical and functional specifications but also delivers practical, intuitive, and effective tools tailored to real-world agricultural workflows.

To ensure that the platform truly meets user needs, we conducted **direct meetings with farmers**, where we discussed their daily routines, challenges, and expectations. As part of our usability evaluation, we distributed the **System Usability Scale (SUS)** questionnaire to multiple users. Based on their responses, the system achieved an **average SUS score of 87.14**, which is considered **excellent usability**. This strong score confirms that users found the platform easy to use, well-integrated, and suitable for practical, day-to-day use in the field.

These meetings and feedback sessions are **still ongoing**, and user input continues to guide refinements to the interface, alert mechanisms, and recommendation formats. By involving real users throughout the development process, we have built a solution that is not only technologically advanced but also **genuinely usable and impactful in real-world agriculture**.

4.3 Workflow

4.3.1 Phase A

The workflow for this project began with an initial discussion with Dr. Noami, during which the potential focus areas were explored, ultimately leading to the decision to pursue a project in the field of smart agriculture. This was followed by a comprehensive literature review, where articles and research papers were analyzed to gain a deeper understanding of the technologies and challenges associated with smart agriculture. Several meetings were held with farm & mechanical engineering teams to refine the project direction. In the first meeting with Keren, challenges in agriculture were discussed, and various project ideas were explored. Subsequent meetings included reviewing project proposals from mechanical engineering students and evaluating their feasibility, as well as selecting the early disease detection system for avocado trees after presentations at the FAB Lab. Another meeting at the FAB Lab allowed for the clarification of sensor requirements and the addressing of technical questions collaboratively. These discussions and insights led to the refinement of the final project idea, defining the key technical and engineering requirements needed for implementation.

4.3.2 Phase B

began in **April 2025 (semester B)**, marking the transition from planning to full-scale system development. The first step involved setting up the **backend infrastructure in Python**, followed by the integration of **frontend components** using **Next.js**, and connecting the system to **Firebase** and **Mosquitto** for real-time data handling and storage.

One of the major turning points occurred early in development when we realized that the originally planned camera module (Unity V2 Camera) was not suitable for our needs. This challenge led us to consult with two engineering students, who introduced the idea of using a regular camera with **OpenAI's GPT API** as a new AI solution. After conducting internal research and tests, we adopted the **GPT-4o-mini** model, which became the core of our disease detection logic.

Over the course of Phase B, we implemented and manually tested every part of the system: from **photo capture and resolution checks** to optimizing the **image prompt structure** sent to the GPT API, and verifying **sensor data accuracy**. We iterated through multiple API calls until we consistently received useful, customized responses. These responses were validated by comparing GPT's output with the actual sensor data and visual inputs, confirming that the model was responding with **relevant, non-generic insights**. Local **farmers also validated** the usefulness of these responses, stating that they were **understandable and informative** for practical use.

Once a working prototype was available, we initiated **user testing** with real farmers. Based on their interactions and feedback, we made several improvements to the user interface and functionality. We also conducted a **System Usability Scale (SUS)** survey, resulting in a high average score of **87.14**, indicating excellent user satisfaction and usability.

As part of our outreach efforts, we also presented the project to engineers and students from other universities in the **Association of Engineers in Israel**. These presentations allowed us to showcase the system’s capabilities, receive valuable external feedback, and generate interest for potential collaborations beyond our institution.

Throughout this phase, we implemented solutions to practical issues—such as **resizing images** to reduce token usage in the GPT API and fixing bugs related to data formatting and transmission. In parallel, we expanded the system to support **multiple users, farms, and trees**, preparing it for real-world deployment in varied agricultural settings.

Currently, the system is **fully functional**, ready for use in farms that have access to electricity and Wi-Fi. To promote our solution, we **shared it on social media**, which led to **growing interest from the agricultural community**. As a result, we have begun receiving **invitations to present the system to farmers, including those from outside Israel**, highlighting its potential global relevance. Looking ahead, we aim to expand support for other types of **fruits and vegetables** and continue gathering farmer feedback to further improve accuracy, usability, and scalability.

Figure 4 shows visual diagram of our workflow on the project.

(Figure 4. Workflow)

4.3.3 Pictures from farms and meetings

4.4 Solution

4.4.1 Algorithm

4.4.1.1 Data Collection

The system begins with data collection, utilizing sensors distributed across the field. These sensors continuously monitor various environmental and tree-specific parameters, such as soil hydration levels, temperature, humidity, and tree images. The data is stored in the cloud for further processing.

4.4.1.2 Image Analysis Using GPT-4o-mini API

Images captured by a standard camera are sent to a local server twice a day. The server resizes the images to optimize performance and cost, then gathers real-time sensor data from the field. This combined information—image and sensor readings—is sent to the GPT-4o-mini API for analysis. The model processes the data and returns a detailed response indicating whether the tree is healthy or infected, along with reasoning and recommendations. The result is saved to Firebase and displayed on the user dashboard.

4.4.1.3 Data Integration and Analysis

Once the detection results are saved in the Mosquito cloud, the system retrieves them along with other environmental data from the field sensors. This includes soil hydration levels and atmospheric conditions. By integrating the detection results from the camera with environmental data, the system conducts a detailed analysis to provide a holistic understanding of the field's overall condition. This integrated approach ensures the detection of potential issues that may not be apparent from the images alone.

4.4.1.5 Generating Actionable Insights

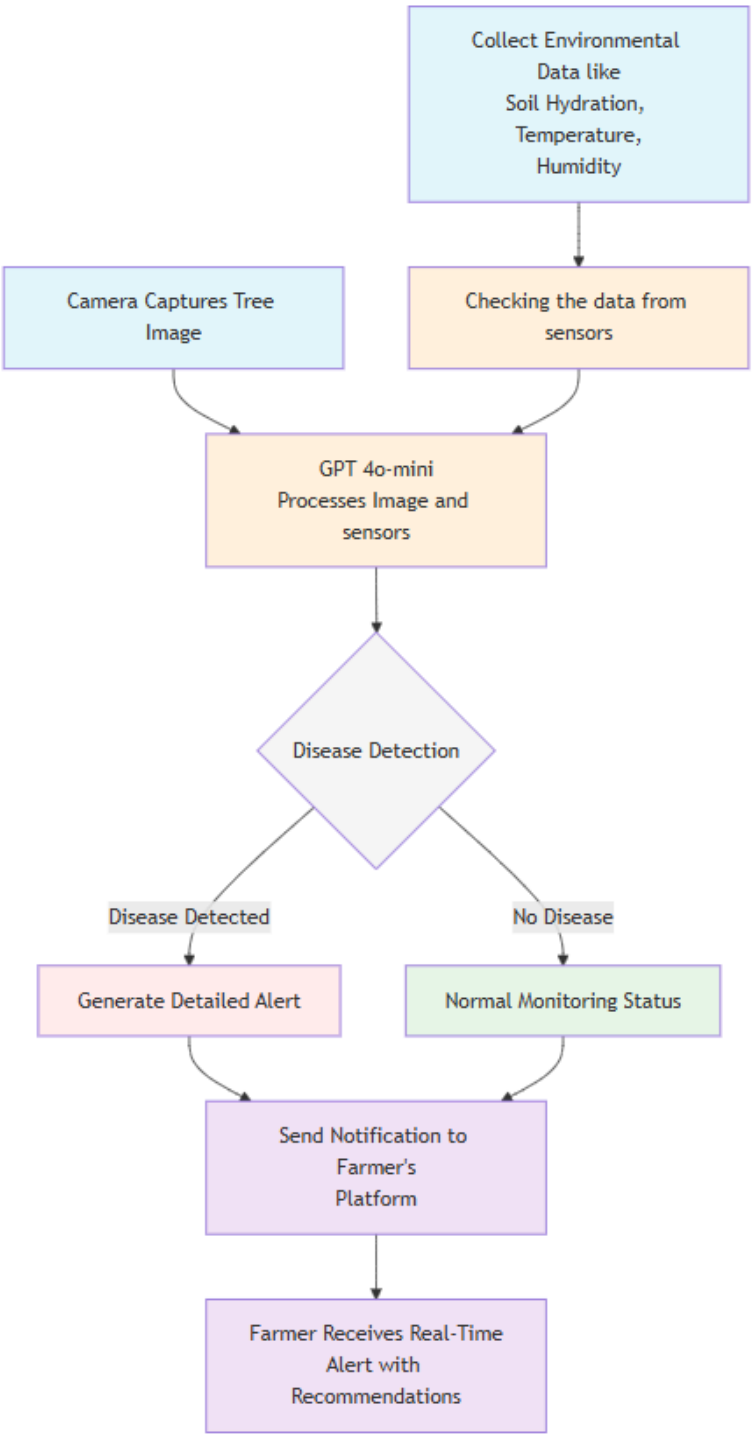
The results of the analysis are compiled into actionable insights. The system generates alerts and sends notifications directly to the farmer's profile through our platform. These alerts include a detailed report on the following:

- Whether trees require attention
- Potential issues identified
- Recommendations for addressing them

For example, if low soil hydration levels are detected, the system may recommend irrigation. If the detection results from the model indicate disease presence, specific treatment actions are suggested.

4.4.1.6 Real-Time Updates

Our algorithm ensures timely updates by conducting multiple checks during the day. This schedule allows the system to provide farmers with up-to-date information, empowering them to make informed decisions to maintain the health and productivity of their fields. By leveraging image analysis through the GPT-4o-mini API and integrating it with real-time environmental sensor data, the system provides a robust and intelligent solution for proactive field management.



(Figure 6. Diseases Detection Algorithm)

4.4.2 Engineering Design

Our proposed solution is an advanced software system connected to a cloud server, designed to retrieve and process information collected by sensors strategically placed in the field near avocado trees. The model is divided into two main parts (See Figure 6 and Figure 7). The first part, the bracket model, incorporates the M5StackStick C Plus2, the YL69 ground moisture unit, and the ENV IV sensor. This component is positioned directly beneath the tree to optimize data collection. The second part, the camera, is placed a few steps away from the tree to capture a wider view, ensuring a better angle for capturing the entire tree and monitoring its structure and surroundings. Both parts are connected to the cloud, with the M5StackStick C Plus2 transmitting collected data in real-time to the Mosquitto cloud using a JSON format and a communication protocol determined by mechanical engineering students, MQTT.

The system decodes the sensor data and presents it to users in an intuitive and organized format, ensuring ease of access and interpretation. Additionally, the camera captures images of the trees, which are then sent to the server and analyzed using the GPT-4o-mini API. This cloud-based image analysis approach eliminates the need for onboard processing or external deep-learning infrastructure. By identifying signs of disease through visual and environmental data, the system reduces the need for manual inspection, enhances early detection, and enables timely intervention—ultimately improving efficiency and supporting proactive plant care.

Beyond disease detection, the system functions as a comprehensive agricultural management tool. It includes features such as:

- Reminders for Key Agricultural Tasks: Schedule and track events like irrigation, fertilization, and pest control.
- Harvest Planning: Estimates optimal fruit-picking times based on visual and environmental data.

This integrated solution delivers a full suite of tools tailored to the farmer's needs, providing a seamless and efficient way to manage and optimize avocado cultivation. By minimizing manual intervention and offering actionable insights, the system empowers farmers to focus on growing healthy, productive crops with greater ease and confidence. Figures 7, 8 explain this process, and Table 1 presents the sensors used in the system.

(Figure 7. Camera)

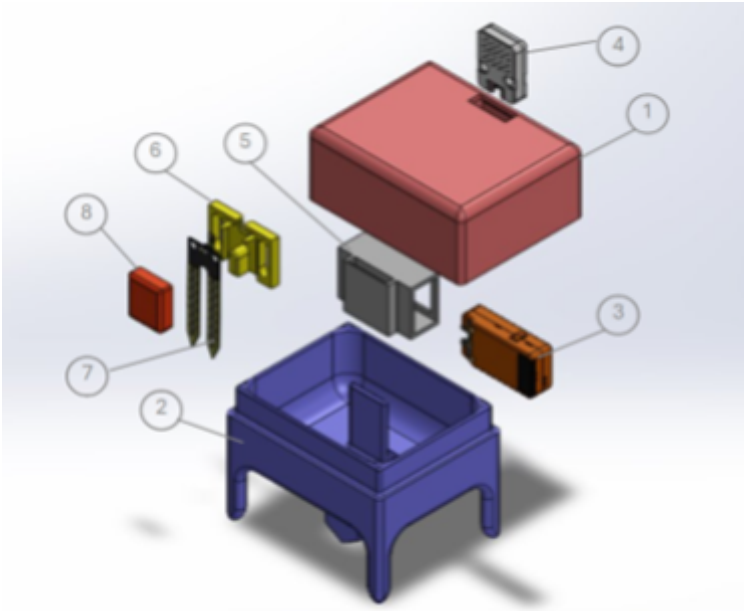


Figure 8. Sensor's Bracket Model

(Table 1 Figure 9. Sensor's Description)

Part number	Description	Quantity
1	Front cover	1
2	Bottom cover	1
3	M5StackStick C Plus2	1
4	ENV IV – Sensor unit	1
5	Controller chassis	1
6	Back cover chassis for ground monitor unit	1
7	YL69 ground moisture unit	1
8	Front cover for ground monitor unit chassis	1

4.5 Technologies Review

4.5.1 Hardware Side

4.5.1.1 Technical Details

The hardware setup of our system consists of a regular digital camera positioned to capture images of avocado trees twice daily, and a set of environmental sensors including soil moisture, temperature, and humidity sensors. These components are connected to an M5Stack microcontroller, which collects sensor data in real time. The camera sends images to a local server PC, where they are resized and packaged together with the latest sensor readings. The M5Stack communicates with the server via Wi-Fi, using the MQTT protocol through Mosquitto to ensure efficient, lightweight data transmission. This setup enables continuous monitoring of tree health and environmental conditions, forming the foundation for automated, AI-driven disease detection.

4.5.2 Software Side

4.5.2.1 Client-Side Technologies

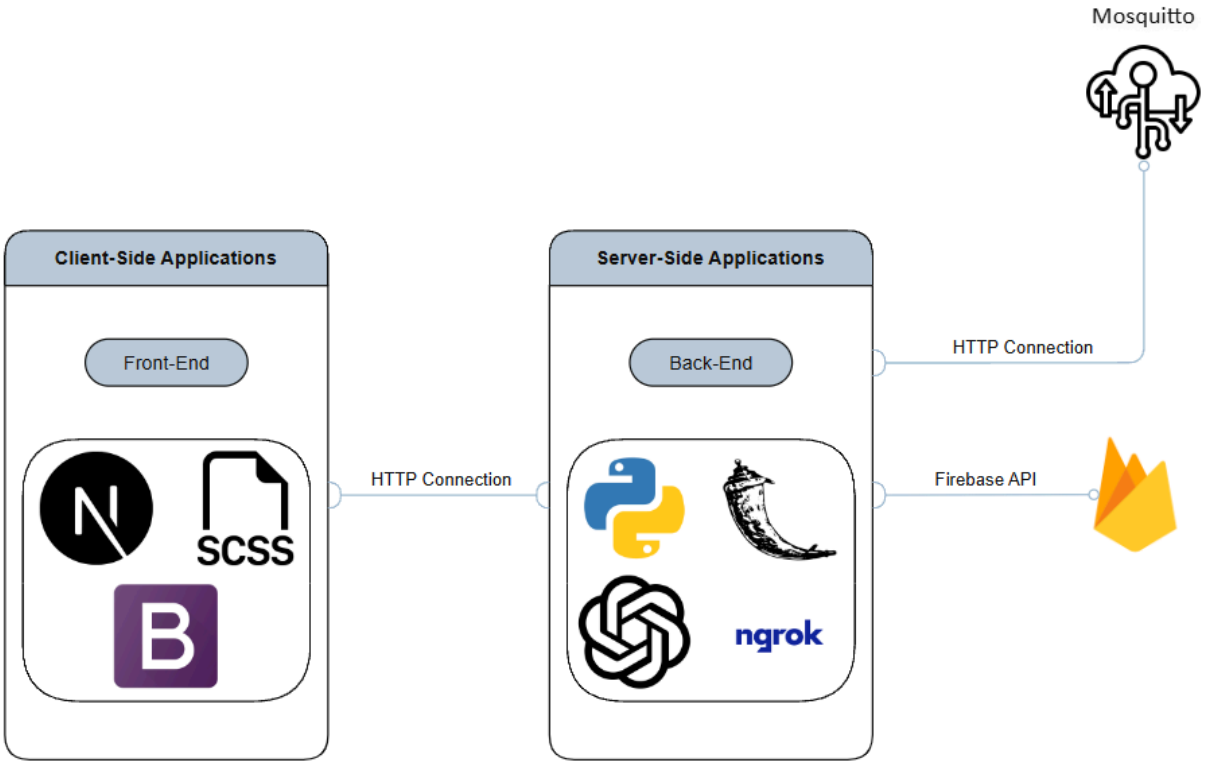
Our client-side development is built using **Next.js**, a powerful React-based framework that enables the creation of fast, scalable, and user-friendly web applications. Next.js allows for server-side rendering, optimized routing, and seamless integration of dynamic components, making it ideal for real-time agricultural monitoring systems. It is used in conjunction with **JavaScript**, a versatile and event-driven scripting language that enables dynamic interaction with webpage elements, seamless integration with HTML and CSS, and efficient handling of asynchronous tasks such as data fetching. To enhance data visualization, we integrated **Chart.js**, a lightweight and easy-to-use JavaScript library for creating responsive, HTML-based charts. These charts visually present real-time data collected from field sensors, providing farmers and agricultural managers with clear and actionable insights.

4.5.2.2 Server-Side Technologies

Our server-side development is based on **Python**, a powerful and widely used programming language known for its simplicity, readability, and extensive library support. We used **Flask**, a lightweight web framework in Python, to handle the backend logic and API communication, along with **ngrok** to expose our local server securely during development and testing. While Python is commonly used for machine learning and data manipulation tasks using libraries such as Pandas, NumPy, and TensorFlow, in our project it primarily manages backend operations and coordinates the flow between sensor data, image processing, and AI analysis via the GPT-4o-mini API. For real-time data storage and synchronization, we integrated the **Firebase Realtime Database**, a cloud-hosted NoSQL database that stores data as JSON and allows seamless syncing across multiple devices and users. This infrastructure enables real-time updates, user-specific data access, and a reliable backend foundation for the entire system.

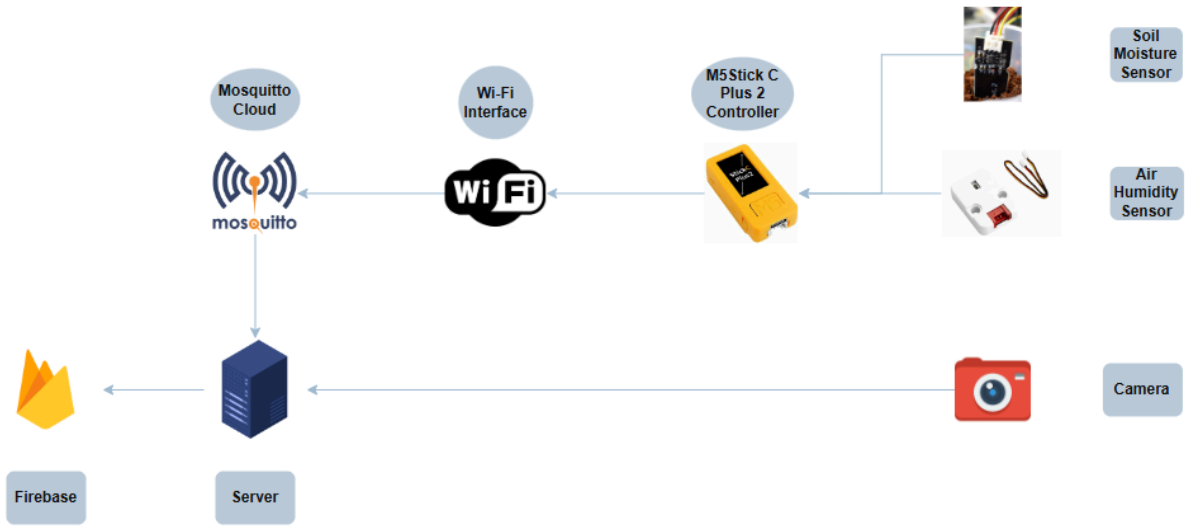
4.6 Architecture Diagram

4.6.1 Software Architecture



(Figure 10. Software Architecture)

4.6.2 Hardware Architecture



(Figure 11. Hardware Architecture)

5. Work Artifacts

5.1 Requirements

The system seeks to provide a scalable and efficient solution for farmers and agricultural managers to monitor and maintain avocado tree's health effectively. Tables 2 and 3 presents Functional & Non-Functional Requirements of our system.

Table 2. Functional Requirements

No.	Requirement
1	The system should support user login.
2	The system should allow users to access it via the WEB application.
3	The system shall allow multiple users to access data for the same farm.
4	The system should show notifications for users when new data has been fetched.
5	The system should notify users when a disease is detected, specifying the type of disease and severity.
6	The system should provide recommendations for intervention, such as treatment options or preventive measures.
7	The system should identify key visual features, such as spots, discoloration, or patterns indicative of disease.
8	The system should classify images into predefined disease categories or "healthy" using machine learning models.
9	The system should identify unknown patterns that could indicate emerging diseases.
10	The system should store historical data for trend analysis and predictive modeling.
11	The system should provide a user-friendly dashboard to visualize environmental data, images, and analysis results.
12	The system should allow chart generation to visualize the collected data.

Table 3. Non-Functional Requirements

No.	Requirement	Type
1	The image analysis for disease detection shall be completed within 30 seconds per image under normal conditions.	Performance
2	The system shall be able to handle data from more than 1 sensor simultaneously without performance degradation.	
3	The cloud infrastructure shall scale dynamically to handle increased workloads during peak data transmission periods.	Scalability
4	The system shall maintain an uptime of 99%, ensuring minimal downtime for data access and processing.	Availability
5	The system shall provide real-time data even in locations with limited network connectivity	
6	The system shall provide an intuitive and user-friendly interface, accessible via web browsers and mobile devices.	Usability
7	Users shall require no more than 20 minutes of training to navigate and operate the system efficiently.	
9	The system shall ensure accurate data transmission with no error rate	Reliability
10	The system shall automatically retry data transmission in case of communication failures of less than 20 sec.	
11	The system architecture shall be modular, enabling developers to update or replace individual components without affecting the entire system.	Maintainability
12	The system shall support custom configurations for sensor thresholds, task reminders, and notification preferences based on user requirements.	Flexibility
13	The system shall accommodate various types of sensors and communication protocols without major modifications.	
14	The interface shall be optimized for use on devices with varying screen sizes, including smartphones and tablets.	Accessibility

5.2 Use Case Diagram

The **Avocado Disease Detection System** allows farmers to monitor and manage the health of their avocado trees using real-time data, image analysis, and alert notifications. The system integrates multiple services (GPT-4o-mini for image analysis, Mosquitto Cloud for sensor data, and Firebase Cloud for data storage) See figure 12.

Actors:

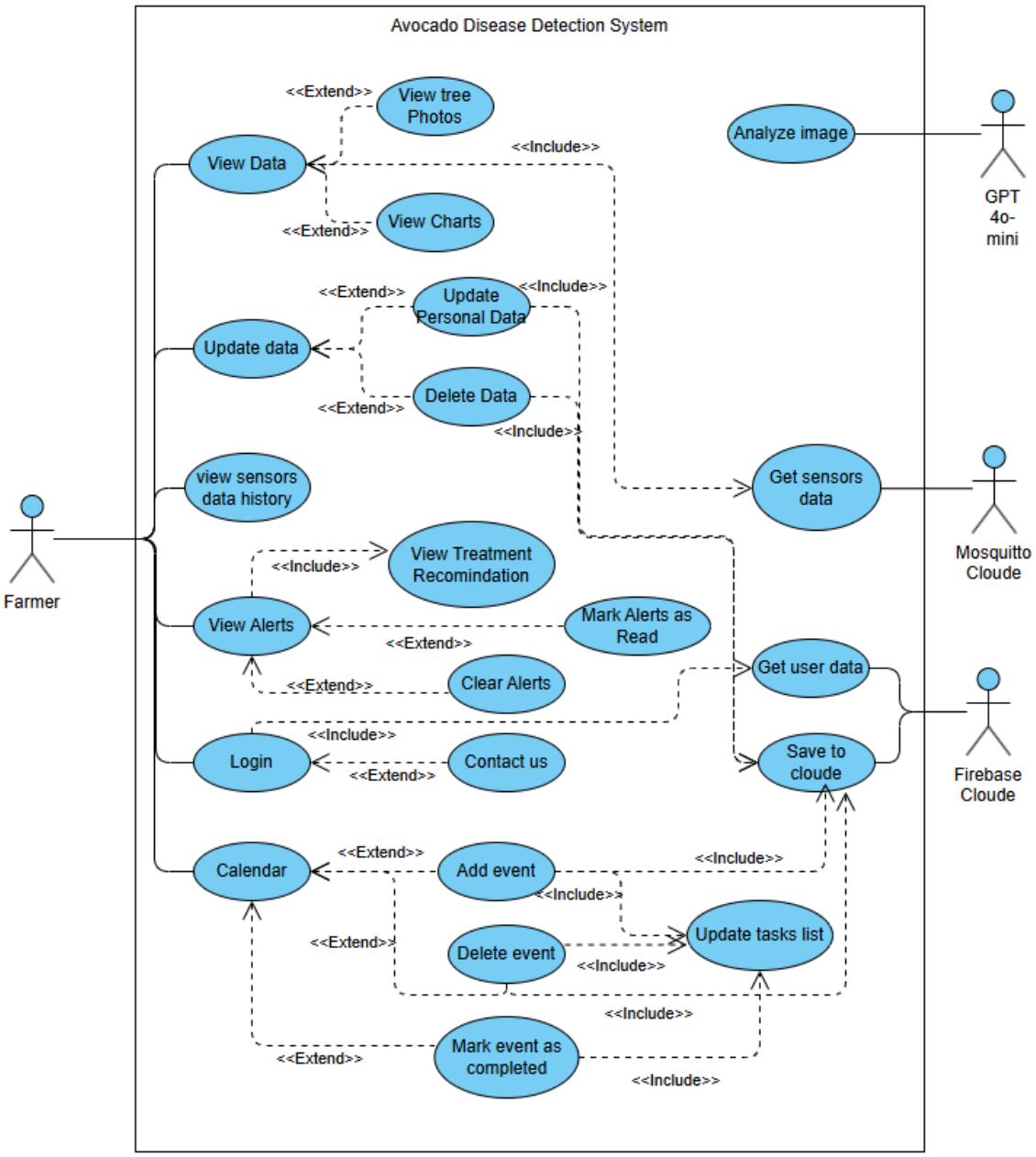
- **Farmer**: Main user interacting with the system.
- **GPT-4o-mini**: Processes tree photos to analyze disease symptoms.
- **Mosquitto Cloud**: Provides real-time sensor data (e.g., temperature, humidity, soil moisture).
- **Firebase Cloud**: Stores user data, alerts, and events in the cloud.

Key Functionalities:

1. **Data Management**
    - Farmers can view data in multiple formats, such as tree photos, charts, and historical sensor data.
    - They can update, delete, and personalize stored data.
    - All data updates are synchronized with Firebase Cloud.
2. **Disease Detection & Recommendations**
    - Farmers can upload tree images.
    - GPT-4o-mini analyzes the image to detect potential diseases.
    - Based on detection results, the system provides treatment recommendations.
3. **Sensor Data Monitoring**
    - The system retrieves live environmental data from Mosquitto Cloud.
    - Farmers can track sensor history for better decision-making.
4. **Alerts System**
    - The system sends alerts for potential diseases or abnormal sensor readings.

- Farmers can view, clear, or mark alerts as read.
- Alerts may include treatment suggestions.

5. **Calendar & Task Management**
    - Farmers can add, update, delete, and complete events.
    - Events and alerts are linked to a task list that keeps the farmer updated.
    - Calendar changes are stored in Firebase Cloud.
6. **User Account Management**
    - Farmers can log in to access their personal dashboard.
    - Contact options are available for support or inquiries.



(Figure 12. Use Case Diagram)

Table 4. Test Cases

No.	TestID	Precondition	Expected Result	Description	Test Result
1	SensorConfigured	IoT sensors are properly connected and configured	Sensor data is accurately collected and transmitted	Test the functionality of IoT sensors to ensure they capture and send environmental data correctly	Passed
2	SystemNetworkConnectivity	The system has access to stable network connectivity	Data from sensors updates on the dashboard	Test data flow from IoT sensors to the dashboard.	Passed
3	UserLogin	Test user has access to the system	User interface displays data in a clear, user-friendly way	Verify the usability and accessibility of the dashboard for end-users	Passed
4	EnviromentalFactors	Sensors are placed in varying environmental conditions	System collects and processes data accurately in all conditions	Evaluate the robustness of sensors under diverse environmental factors	Passed
5	InconvenienceData	Data irregularity is introduced in the sensor readings	System detects and flags the irregularity	Test the system's ability to flag irregularities	Passed

6	SystemComponents	All system components are operational	System performs end-to-end disease detection and reporting	Conduct full system testing from data collection to actionable insights	Passed
7	HistoricalData	Historical data is stored in the system	System generates trends and predictions based on data	Test the analytics and predictive capabilities of the system using historical data	Passed
8	UserInputError	User provides incorrect or incomplete input on the dashboard/Login	System provides appropriate error messages or guidance	Validate error handling and feedback mechanisms in the user interface	Passed
9	PlatformCompatibility	Dashboard accessed on multiple devices or browsers	Dashboard functions consistently across platforms	Test the cross-platform compatibility of the user interface	Passed
10	DataLossTransmission	Data packet loss during transmission	System retransmits or handles missing data gracefully	Evaluate the system's ability to handle data loss during transmission	Passed
11	DiseaseIdentification	Real-world disease symptoms on avocado trees	System correctly detects and classifies diseases	Conduct field testing to verify disease detection in real-world scenarios	Passed

12	ChartGeneration	Generating a Chart based on the provided data from sensors	System correctly generates and showcases the chart to the user on the dashboard	Evaluate the system's ability to generate the correct chart with readable resolution	Passed

6. User Guide

6.1 Login Page

- **Purpose:** The login page serves as the entry point to the system for registered users, primarily farmers and agricultural managers. Its purpose is to ensure that access to the system's functionalities and farm-specific data is secured and personalized using their credentials.

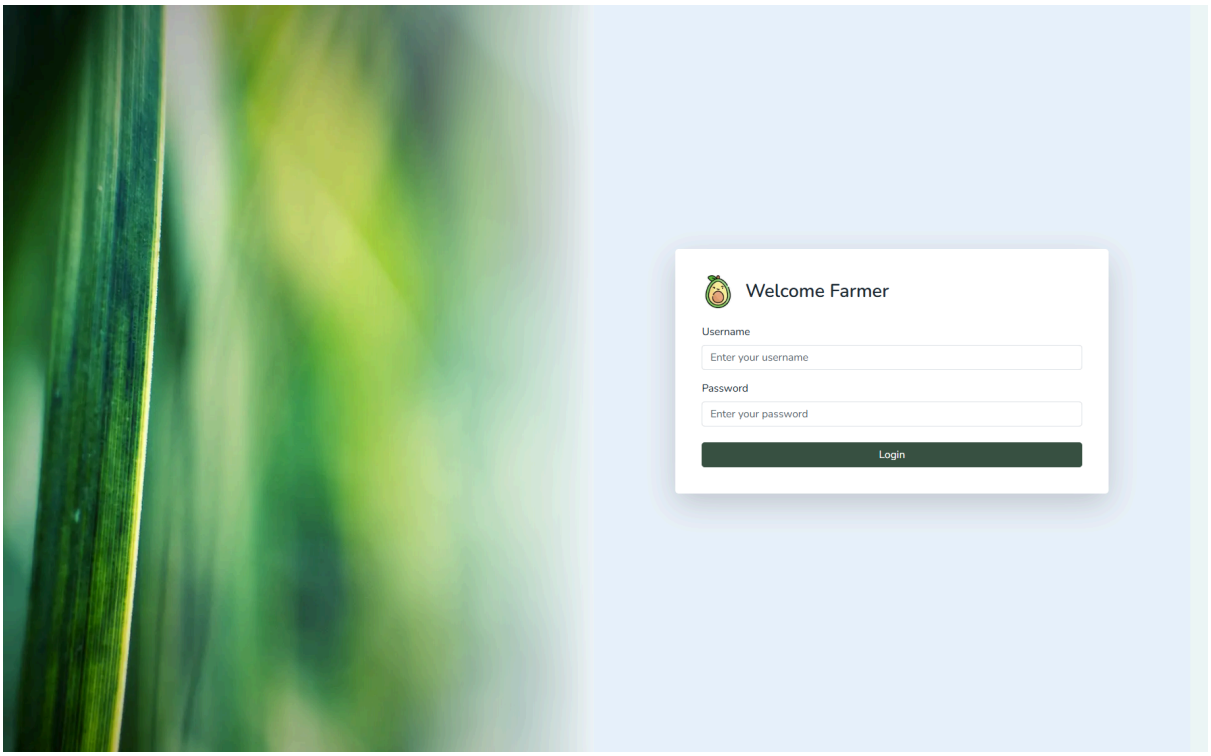
- **Key Elements:**

    - **Username Input Field:** A text input box labeled "Username" where users enter their unique login identifier.

    - **Password Input Field:** A secured password field labeled "Password" that masks input to protect user credentials and ensure secure login.

    - **Login Button:** A dark green rectangular button labeled "Login" that triggers the authentication process. Once clicked, it checks the entered credentials against the Firebase Realtime Database.

    - **Error Message:** Displays a red alert box with messages like "Incorrect password" or "User not found" if login fails.



(Figure 13. Landing Screen Page)

6.2 Dashboard Screen:

- **Purpose:** The dashboard is the central interface of the AvoTech system, providing farmers with a clear and actionable overview of their orchard's health, sensor data, and recent activity. It is designed to offer real-time insights, streamline decision-making, and serve as a daily operations hub.

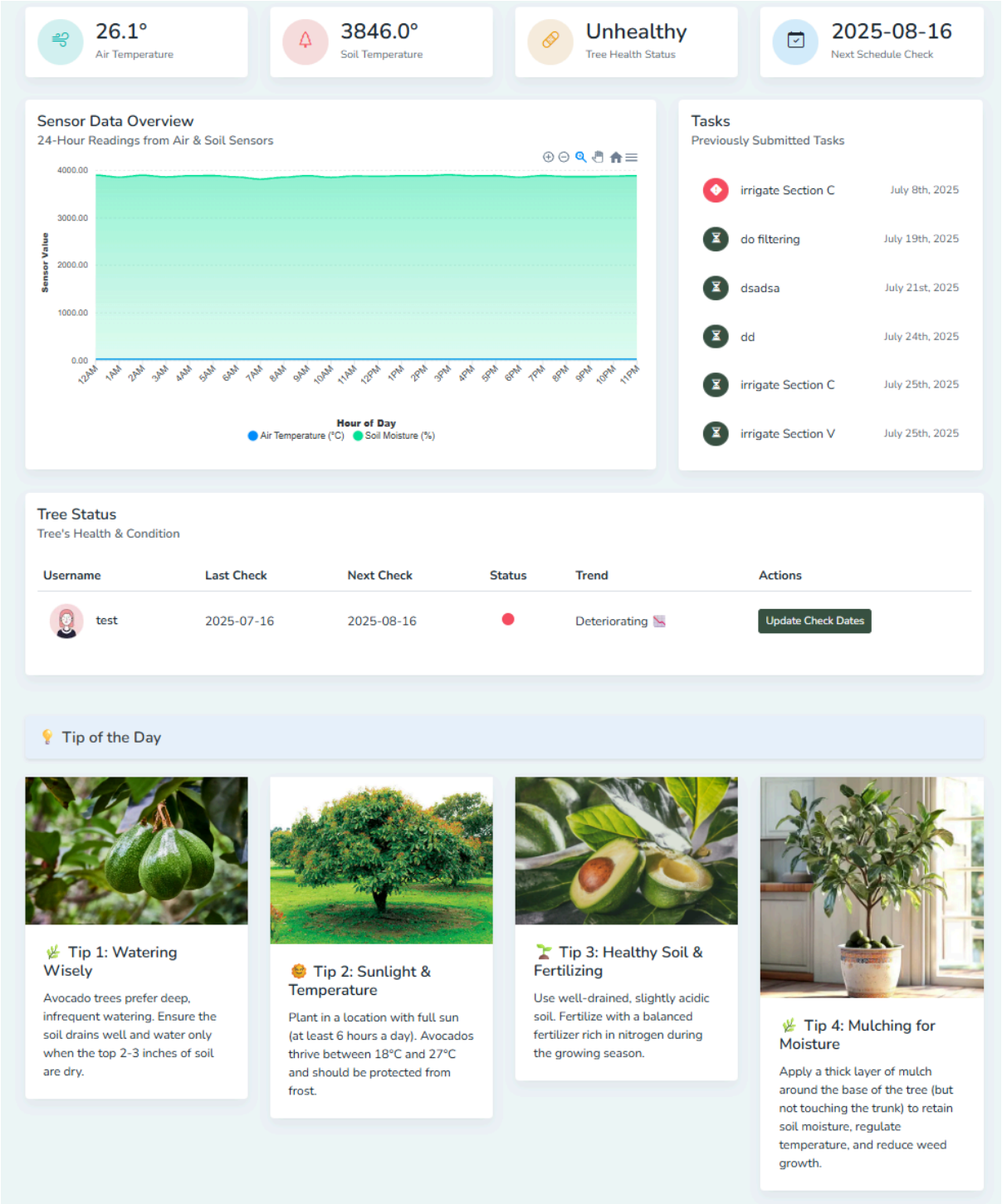
- **Key Elements:**

    - **Top Status Indicators (Colored Boxes):** these boxes display the current health status summary across the farm.
        - **wind icon:** real time air temperature indicator from the field.
        - **tree icon:** real time soil moisture indicator from the field.
        - **bandage icon:** indicates if all trees are healthy or not with explaining on which tree is not healthy.
        - **calendar icon:** date of the day.

    - **Sensor Data Overview Chart (Center Graph):** A visual chart representing real-time sensor readings (e.g., temperature and soil moisture) over time, Helps users quickly spot trends and anomalies in environmental conditions.

    - **Tasks Panel (Top Right Box):** Displays recent tasks and alerts such as Irrigation schedules,Fertilization reminders and Harvest readiness alerts,it uses icons and timestamps to help users stay organized and proactive.

    - **Tree Status Table (Bottom Section):** A detailed table listing individual tree data like, last checked date, next check date, current health status, submit tree got checked button and a trend which tells the farmer if the tree health is stable, improving or deteriorating. It offers a quick overview of all monitored trees, helping farmers identify which need attention.



(Figure 14. Dashboard Screen Page)

6.3 Navigation Header:

- **Purpose:** The Navigation Header provides users with a consistent and accessible way to navigate between different sections of the application, access account settings, and log out securely.

- **Key Elements:**

    - **Mobile Menu Toggle:** A hamburger icon that triggers the side navigation drawer on small screens.

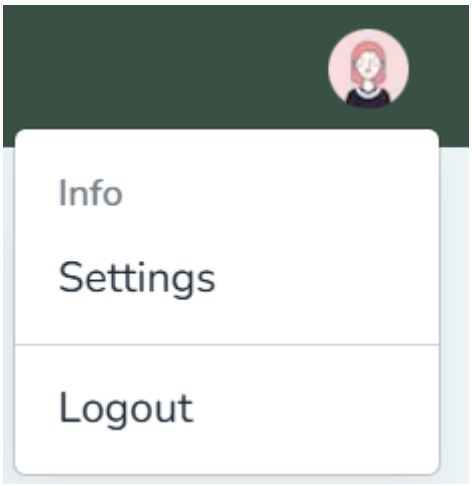
    - **Navigation Links:** A horizontal set of links for quick access to:

        - **About:** Learn more about the system.
        - **Contact Us: C**ontact information or support.
        - **FAQ:** Frequently asked questions.

        - **User Profile Dropdown:** Profile icon that opens a dropdown menu with:

            - **Settings:** (placeholder, no functionality yet).
            - **Logout:** Ends the session, clears storage, and redirects to the login page.

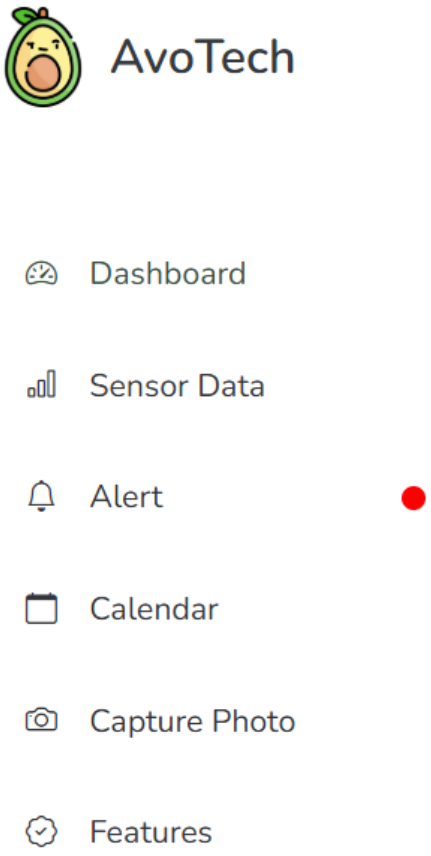
(Figure 20. Navigation Header)



(Figure 15. Dropdown User Profile Header)

6.4 Sidebar Menu:

- **Purpose:** The Sidebar Menu acts as the main navigation hub for the user. It allows seamless access to all core areas of the dashboard, including monitoring, tools, and system features. It also includes a live notification area (alerts).
- **Key Elements:**
    - **Dashboard:** Main overview page.
    - **Sensor Data:** View temperature and moisture data.
    - **Alerts:** Displays real-time alerts or system warnings. A red notification badge is shown if there are **unread alerts**.
    - **Calendar:** View and manage tasks and events.
    - **Capture Photo:** Access camera functionality for AR or image capture.
    - **Features:** Highlights available app features or tools.



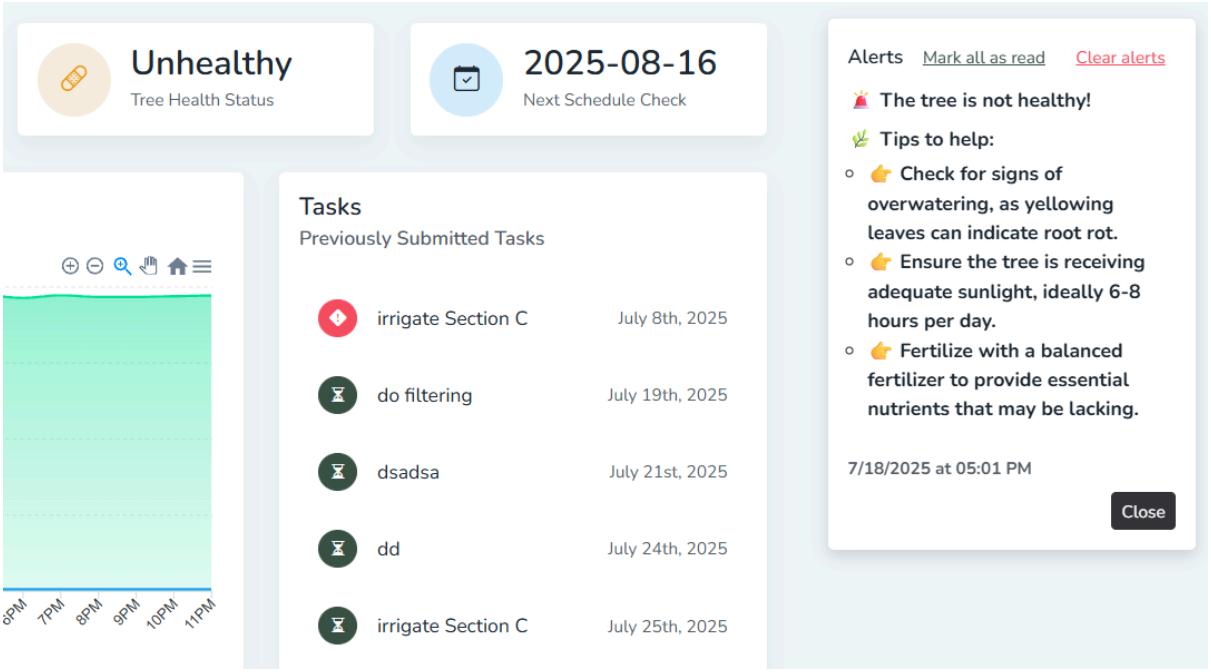
(Figure 16. Sidebar Menu)

6.5 Alert Popup:

- **Purpose:** The Alert Popup provides users with real-time system alerts or warnings. It's a compact notification center that allows users to review, mark, and clear alerts without navigating away from their current page.

- **Key Elements:**

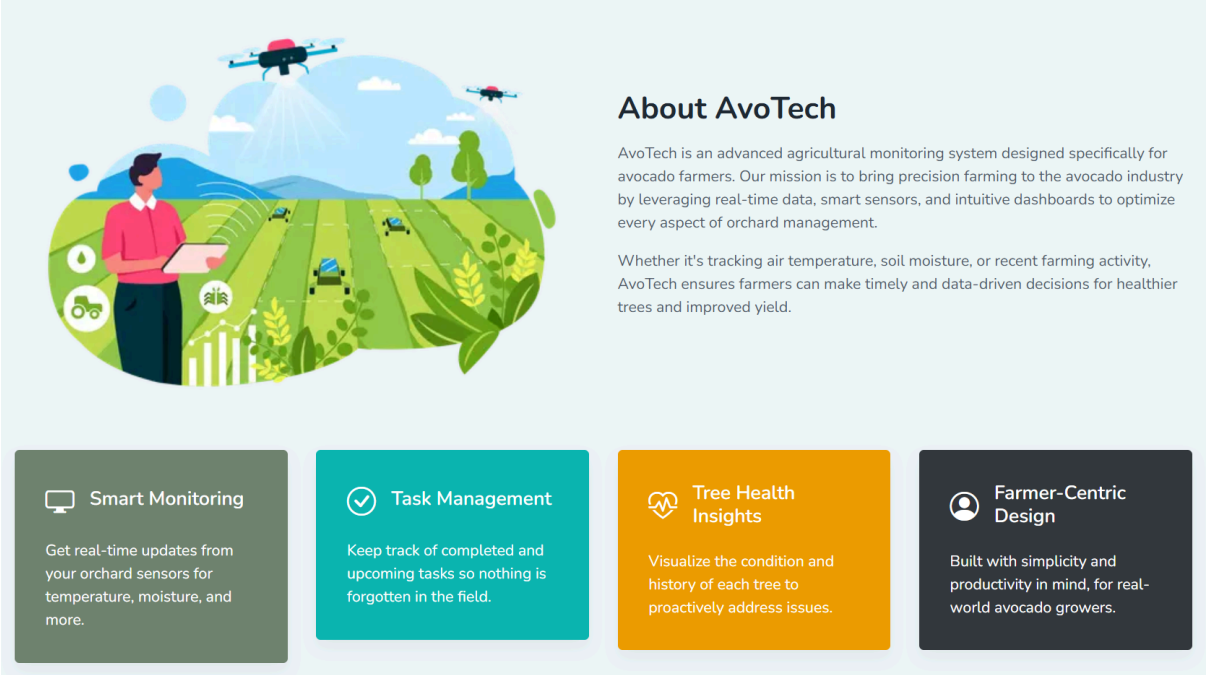
    - **Alerts List:** A list of all alerts with:
        - **Unread alerts shown in bold.**
        - **Read alerts shown in muted text.**
        - **Each alert includes the message and datetime**
        - **Mark all as read:** A button that marks all alerts as read using.
        - **Clear alerts:** Deletes all alerts from the list using.
        - **Close button:** Closes the popup without modifying the alert list by setting.



(Figure 17. Alert Popup Notification)

6.6 About Page:

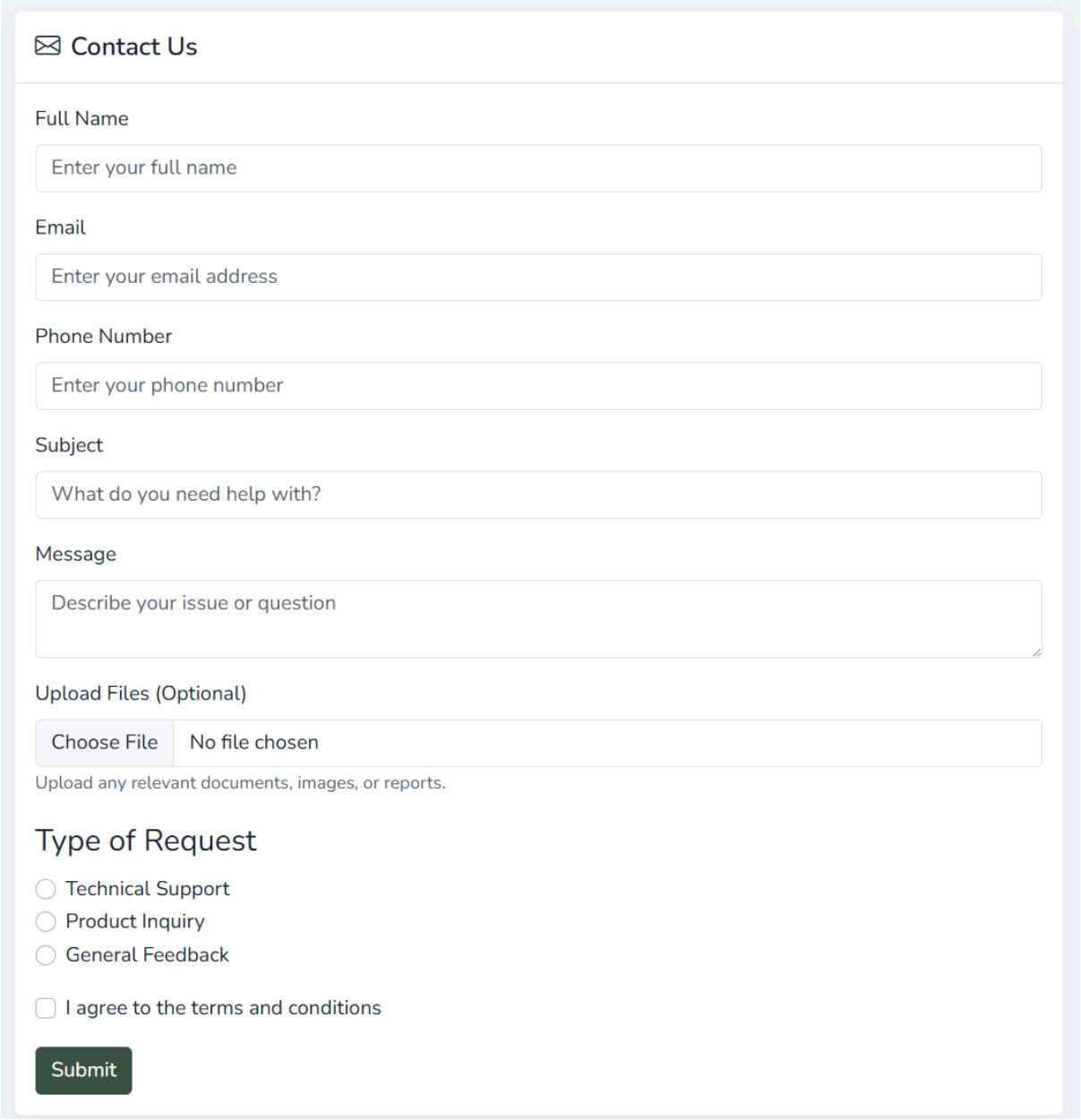
- **Purpose:** The About Page introduces users to the AvoTech platform—explaining its mission, features, and the problems it solves for avocado farmers. It sets the context for the application and builds user trust.



(Figure 18. About Page)

6.7 Contact Us:

- **Purpose:** The Contact Us page allows users to reach out for help, share feedback, or ask questions. It ensures users feel supported and connected to the AvoTech team for any kind of assistance.



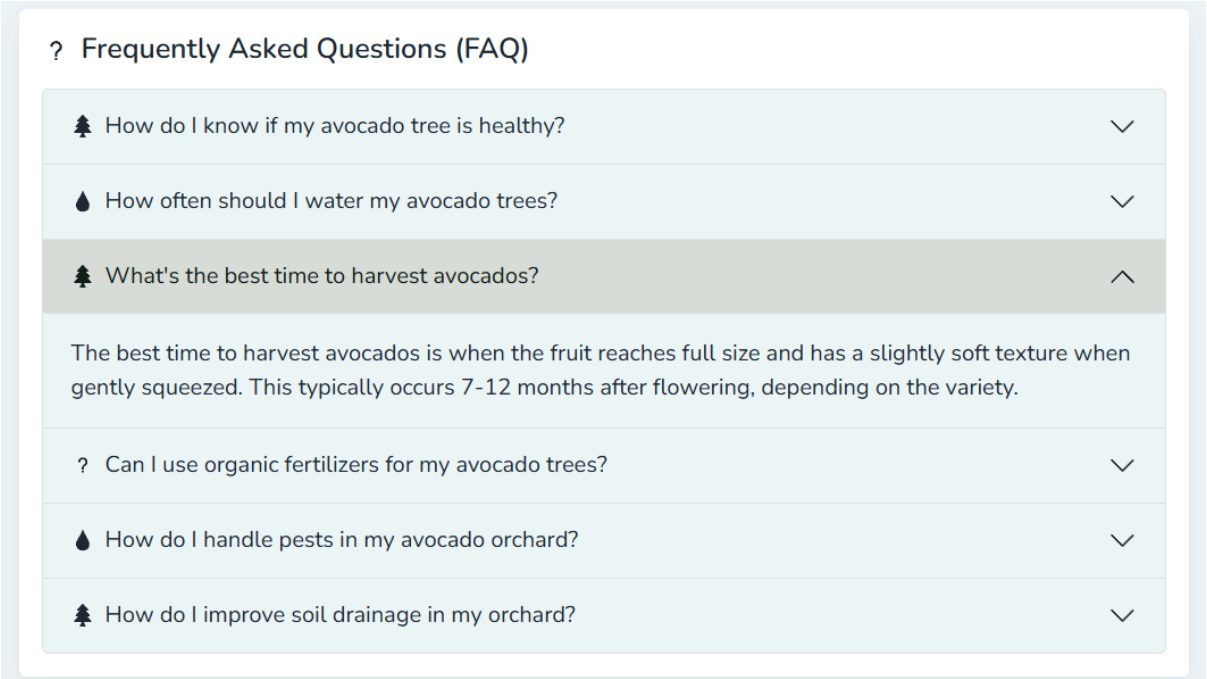
(Figure 19. Contact Us Page)

6.8 FAQ Page:

- **Purpose:** The FAQ page provides users with quick answers to common questions regarding avocado tree care, farming best practices, and how to make the most of AvoTech's smart monitoring tools. It helps users solve problems independently without needing direct support.

- **Key Elements:**

    - **Accordion Interface:** Questions and answers are displayed in collapsible accordion format for a clean, organized view.



(Figure 20. FAQ Page)

6.9 Sensor Data Page:

- **Purpose:** The Sensor Data page provides users with live monitoring of air temperature and soil moisture levels in their avocado orchard. It helps farmers make timely decisions for irrigation, disease prevention, and overall orchard management.

- **Key Elements:**

    - **Real-Time Charts:** Two interactive area charts show the temperature and soil moisture data. Data is continuously streamed and updated using MQTT protocol via a custom `Mqtt` hook.

    - **Air Temperature Chart:** Displays hourly temperature readings in °C.

    - **Soil Moisture Chart:** Shows moisture levels to help prevent overwatering or drought stress.

(Figure 21. Sensor Data Page)

6.10 Calendar Page:

- **Purpose:** The Calendar Page serves as a **task management and note-taking system** for daily farm activities. It allows users to plan, track, and mark farming tasks throughout the month. After making new tasks, they will also appear on the taks table on the dashboard.

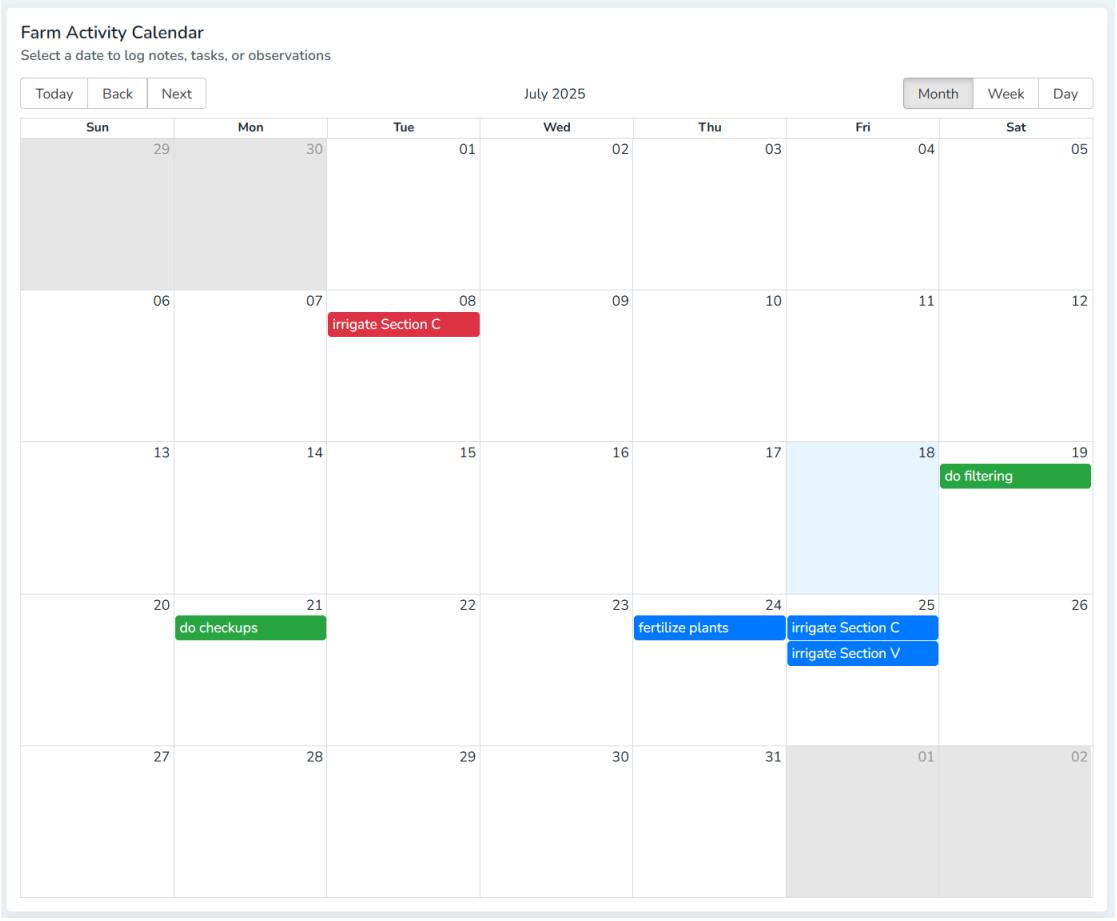
- **Key Elements:**

    - **Interactive Calendar:** Users can choose between view modes: Month/Week/Day. Users also can click any date/time slot to **add new notes or tasks**.

    - **Task Creation Modal:** Lets users define task title and automatically schedules it for the desired day.

    - **Event Interaction:** On clicking a task, a modal allows:
      Mark as Done/Delete/Cancel

    - **Task Status Colors:**
        - **Green** – Task marked as completed.
        - **Red** – Overdue task (not marked done and past date).
        - **Blue** – Upcoming or current tasks.



(Figure 22. Calendar Page)

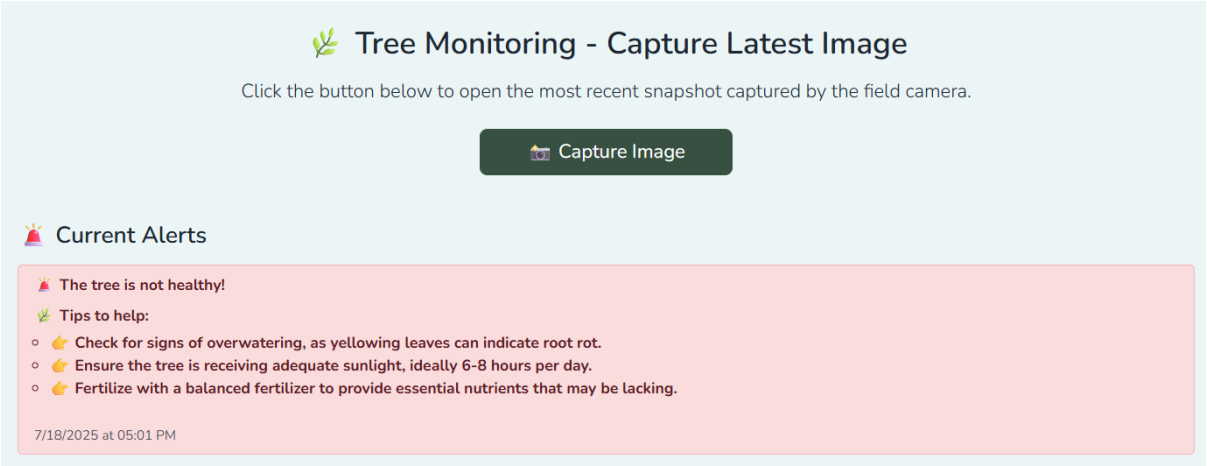
6.11 Capture Live Photo Page:

- **Purpose:** The Capture Photo Page enables users to remotely view the latest snapshot taken by the field camera. It's designed to support visual monitoring of tree health or farm conditions in near real-time.

- **Key Elements:**

    - **Capture Image Button:** Opens a **new browser tab** with the most recent image captured by the farm camera. Appends a **timestamp** to the image URL to ensure the latest version is loaded (avoids caching). If no image URL is found or an error occurs, users are alerted.

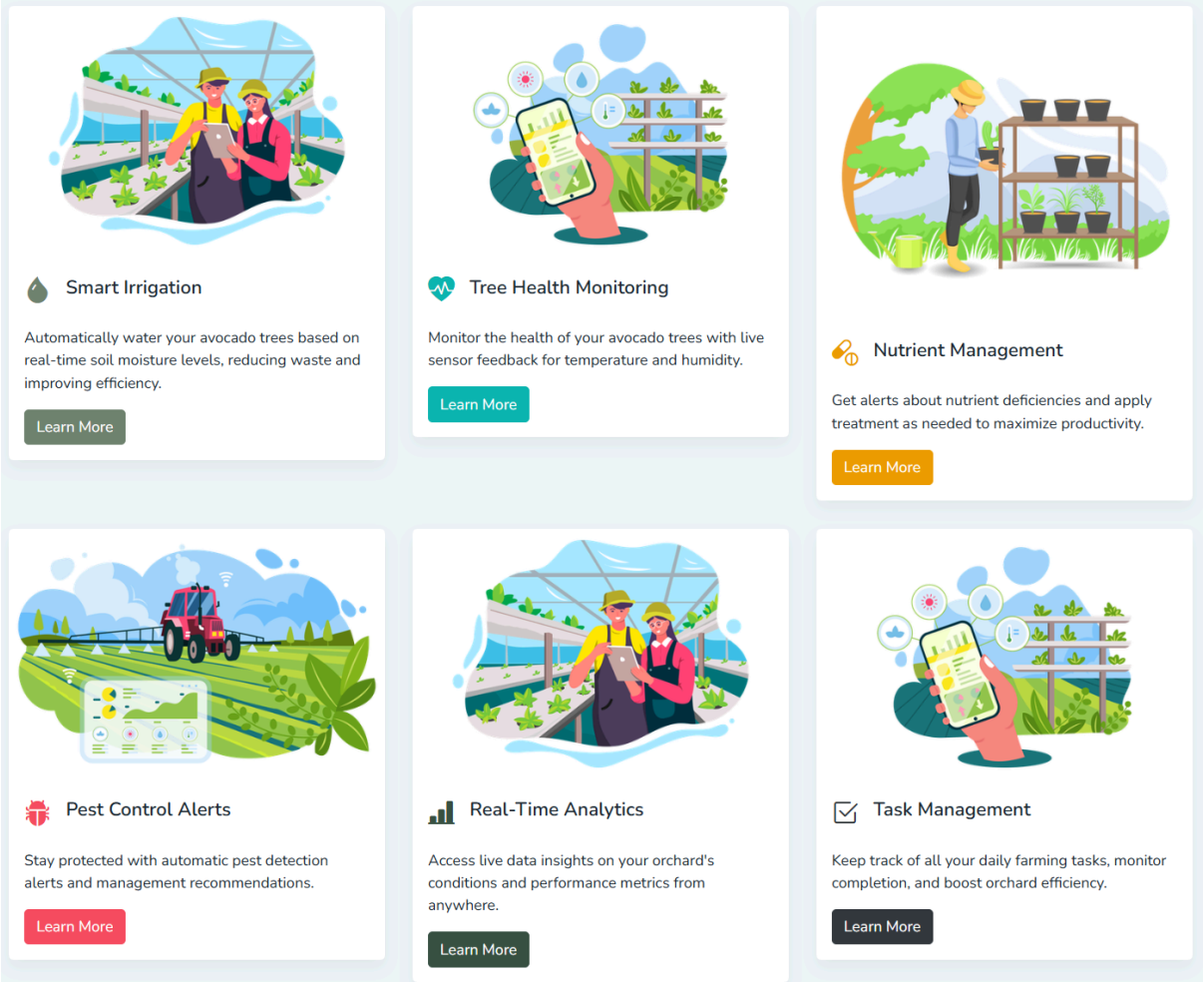
    - **Alert Display Section:** Displays the current active alerts from the shared Alert Context. Each alert shows a **message**, along with its **timestamp**. If no alerts exist, a helpful message ("No alerts at the moment") is shown.



(Figure 23. Capture Live Photo Page)

6.12 Features Page:

- **Purpose:** The Features Page highlights the core capabilities of the smart farming platform. It provides an overview of how the system helps monitor, automate, and manage different aspects of avocado farming using sensor data and intelligent alerts.



(Figure 24. Features Page)

7. Maintenance Guide

7.1 Installations

7.1.1 Prerequisites

Before setting up the project, ensure the following are installed:

- **Next.js**
- **Node.js**: Version **16.x or later** (to support Next.js and all packages properly)
- **npm**
- **Git** for version control
- **Firebase**
- **Python 3+**
- **ngrok**

7.1.2 Project Dependencies

The following libraries and frameworks will be automatically installed via npm install:

7.1.2.1 Core Frameworks & Libraries:

- next@^13.5.11 – React framework for SSR and routing.
- react@18.2.0 & react-dom@18.2.0 – React library.

7.1.2.2 State Management:

- @reduxjs/toolkit@1.9.5
- react-redux@8.1.2

7.1.2.3 Firebase Integration:

- firebase@^11.6.1 – For backend/auth/database integration.

7.1.2.4 Real-Time Communication:

- mqtt@^5.13.0 – For real-time sensor data using MQTT protocol.

7.1.2.5 Styling:

- bootstrap@5.3.1
- bootstrap-icons@^1.10.5

- sass@1.64.2
- reactstrap@9.2.0 – React-based Bootstrap components.

7.1.2.6 UI/UX & Animations:

- framer-motion@^12.9.1
- react-icons@^5.5.0
- react-feather@2.0.10
- simplebar-react@3.2.4 – For custom scrollbars.
- react-apexcharts@1.4.1 – For charts and graphs.
- react-big-calendar@^1.18.0 – For calendar views.

7.1.2.7 Utilities:

- date-fns@^4.1.0 – For date handling.
- prop-types@15.8.1 – Type checking for props.

7.1.2.8 Code Quality:

- eslint@8.46.0
- eslint-config-next@13.4.12

7.1.3 Developer Setup Instructions

7.1.3.1 Clone the repository

- **git clone https://github.com/oRABiiA/avocado-disease-detection-system.git**
- **cd avocado-disease-detection-system**

7.1.3.2 Install dependencies

- **npm install**

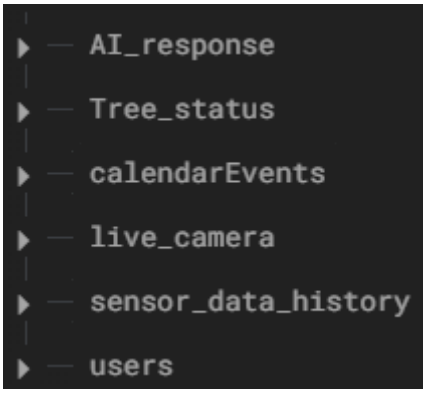
7.1.3.3 Run development server

- **npm run dev**

7.1.4 Firebase Databse

Our Database is based on Realtime Database in Firebase.

The Realtime Databse is structured as follows:



- **AI_response:** Provides the AI's latest analysis of the avocado tree's health, along with care suggestions.

  Fields:

  - Healthy (string): "YES" or "NO" – Indicates if the tree is considered healthy.
  - Tip1, Tip2, Tip3 (string): Practical care suggestions based on the analysis.
  - timestamp (string, ISO 8601): Time when the AI response was generated.

- **Tree_status:** Tracks the overall health trend and upcoming maintenance/checkup schedules.

  Fields:

  - last_check_date (string, YYYY-MM-DD): Last time the tree's condition was assessed.
  - next_check_date (string, YYYY-MM-DD): Scheduled date for the next check.
  - trend (string): "up" | "down" | "stable" – Direction of health change.

- **calendarEvents:** Stores task-based events (e.g. irrigation, fertilization) grouped by date. Each task is stored under a unique ID.

  Fields:

  - title (string): Task description.
  - start, end (string, ISO 8601): Task start/end datetime.
  - done (boolean): Task completion status.

- **live_camera:** Provides the real-time streaming URL for camera access.

Fields:

- url (string): A live camera feed endpoint (e.g., via Ngrok).

- **sensor_data_history:** Captures hourly environmental readings uploaded by sensors.

  Fields:

  - moisture (number): Moisture level from the soil sensor.
  - temperature (number): Air temperature in Celsius.
  - time_of_upload (string): The hour label for reference (e.g., "10AM").
  - Entries are labeled by hour from 12AM to 11PM.

- **users:** Basic user login credentials.

  Fields:

  - userId (string): Unique identifier.
  - password (string): Plain-text password

7.1.5 Running The Standalone Backend Server

Inside the repository, you'll find a folder named Standalone Server. The Standalone Server folder in the repository contains two key Python scripts that run independently from the main client-side app but are essential for AI-powered analysis and data ingestion into Firebase. Make sure the following files are **all located in the same directory**:

```
Standalone Server/
├── AI_model.py
├── capture_image.py
├── firebase_credentials.json
```

7.1.5.1 AI_model.py

This script runs as part of the **Standalone Server**. It performs **automated avocado tree health analysis** using:

- An image captured from a webcam
- Real-time sensor data (temperature & soil moisture) via MQTT
- OpenAI's vision model for image + text analysis

- Firebase to store health reports and suggestions

The script uses OpenAI's Vision model gpt-4o-mini to analyze avocado tree images and generate care recommendations. This requires authentication with a valid API key. To use the AI-powered tree health analysis, **each developer must register for their own OpenAI API key**.

How to set it up:

1. **Create an OpenAI account** (if you don't have one):
   https://platform.openai.com/signup

2. **Generate an API key**:
   Go to https://platform.openai.com/api-keys and click **"Create new secret key."**

3. **Insert the key into** AI_model.py:
   client = OpenAI(api_key="your-api-key-here")

7.1.5.2 picture_request.py

This Python script is a Flask-based lightweight HTTP server that captures and serves avocado tree photos from a webcam. It allows other services (like your AR app or AI model) to access the latest tree image via a simple URL endpoint.

7.1.5.3 Firebase AdminSDK

This file is a Firebase Admin SDK private key used to securely connect server-side applications (like Python scripts) to your Firebase project, bypassing client-side authentication.

- The project admin must download the Admin SDK file from the github repository.
- Place the file in the same directory as AI_model.py or wherever it's being used.
- Update the path in the code if necessary:
  credentials.Certificate("path/to/your-adminsdk-file.json")

8. Evaluation

Our evaluation process focused on assessing the system's **performance and usability** using two key methodologies: **expert interviews** and **user testing**. To evaluate the system's real-world effectiveness, we conducted **in-depth interviews with farmers**, gathering feedback on practicality, clarity, and usefulness. In addition, we administered a **System Usability Scale (SUS) survey** to **seven potential users** after demonstrating the system's core functionalities. These combined methods allowed us to measure both perceived usability and practical value in real agricultural contexts.

8.1 Reflection 1-Interview with Mr. Nir Yakoby -Farmer of Avocado field near Hamat Gader

A significant validation of the AvoTech system's relevance came during an on-site meeting and system demonstration with avocado farmers in an orchard near Hamat Gader. The visit provided valuable insights into the daily challenges farmers face and highlighted the need for digital tools to support efficient decision-making. One key takeaway was that farmers managing large numbers of trees struggle to track how much water and fertilizer to apply, often leading to wasted resources. They expressed strong interest in a solution like AvoTech, which could help them optimize resource usage, reduce costs, and save time by providing clear data and AI-driven recommendations. This feedback reinforced the system's potential to deliver real-world impact in commercial-scale farming.

8.2 Reflection 2-Interview with Mr. Foaad Hmeed and Mr. Saher Hmeed Veteran farmers in Galil

A meaningful validation of the AvoTech system's value came during an on-site meeting and demonstration with small-scale farmers in the Galil region, whose focus includes a variety of fruit trees, not just avocados. These farmers emphasized the importance of early disease detection, especially when managing a limited number of trees. Since they dedicate significant time and care to each tree, losing even one to disease results in a major financial and emotional loss. They described how undetected issues often lead to tree death after years of effort, costing them both time and money. The feedback highlighted a strong need for a system like AvoTech, which can alert them early, provide clear insights, and help them take timely action to protect their crops and avoid preventable losses.

8.3 Areas for Enhancement

8.3.1 Hamat Gader Farmer Feedback

During our on-site meeting and demonstration in the Hamat Gader orchard, the farmer provided valuable suggestions for improving the AvoTech system's capabilities. One key area for enhancement is the integration of a sensor that can measure the specific fertilizer needs of each tree. The farmer emphasized that such a feature would allow for precise, tree-level nutrient management, leading to better crop health and significant cost savings.

In addition, the farmer recommended upgrading the soil moisture sensor, noting that more accurate readings would improve irrigation decisions and enhance the system's overall reliability. These insights underscore the importance of continually refining our hardware components to better support the practical needs of large-scale farmers and strengthen AvoTech's role as a decision-support tool in the field.

8.3.2 Galil Farmers Feedback

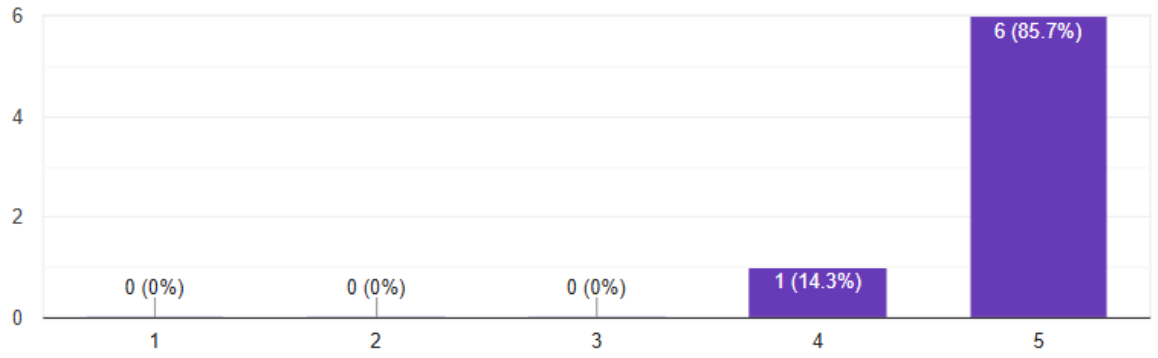
During our visit to small-scale farmers in the Galil region, we received constructive feedback highlighting opportunities to expand the system's usefulness. The farmers expressed a strong interest in seeing AvoTech adapted to support additional types of trees and crops, beyond just avocados. Since their orchards include a variety of fruit trees, they noted that expanding the system's capabilities would make it more relevant and beneficial across their entire farm.

Additionally, the farmers emphasized the importance of having a mobile application version of the system. They explained that accessing the platform directly from their phones while in the field would greatly improve convenience and usability, making it easier to monitor tree health and receive alerts in real time. These insights point to clear paths for future development, including multi-crop support and mobile accessibility, to better meet the needs of diverse and mobile farming environments.

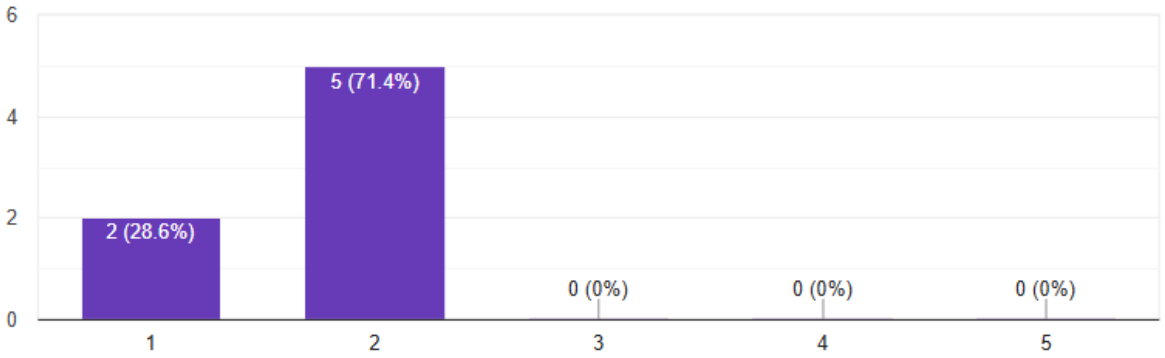
8.4 SUS Survey

From May 2025, we conducted a System Usability Scale (SUS) survey with 7 potential users, including farmers and individuals involved in agricultural work, to evaluate the usability of the AvoTech system. The SUS is a widely recognized and reliable tool for assessing system usability across various domains. Participants responded to 10 standardized questions using a 5-point Likert scale ranging from 1 (Strongly Disagree) to 5 (Strongly Agree). Below, we present the results for each question along with bar charts illustrating the distribution of responses.

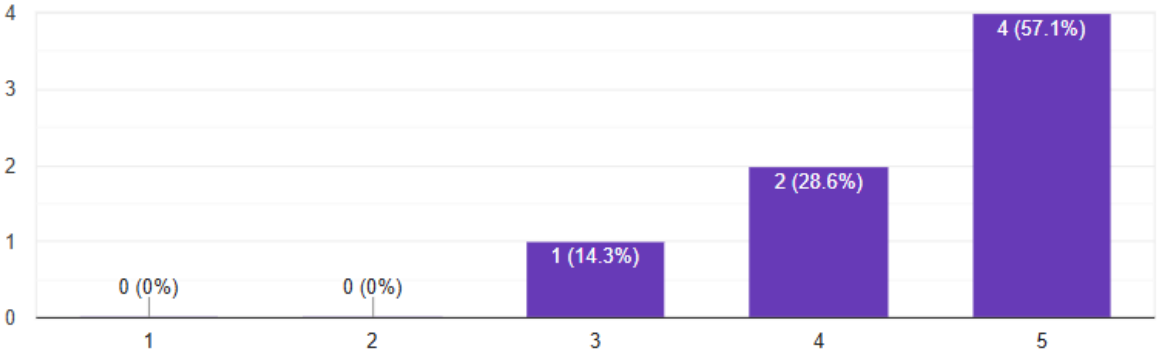
1. I think I would like to use this site often



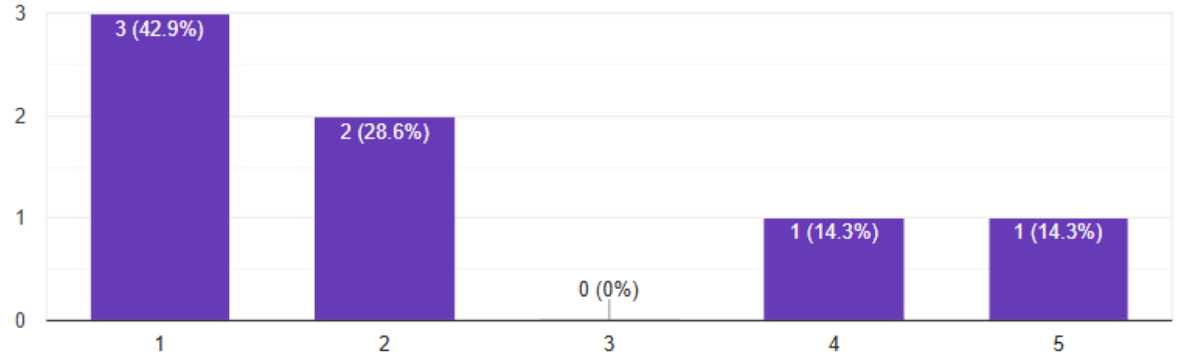
2. I found the site unnecessarily complex



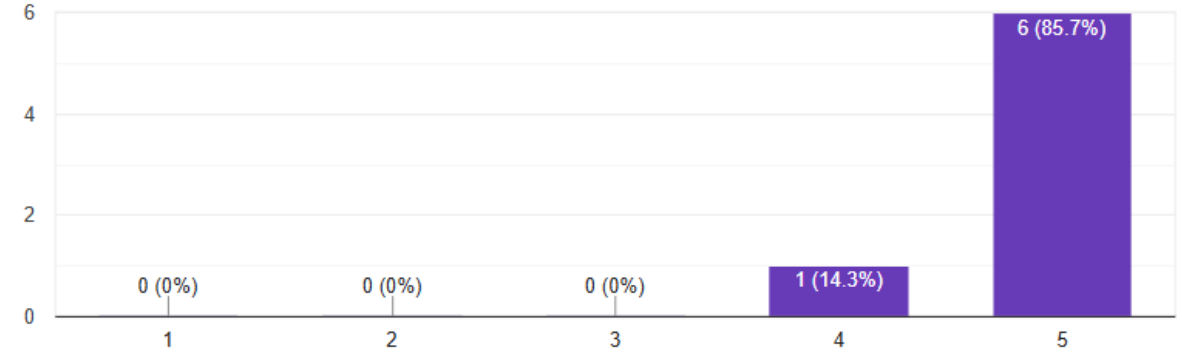
3. I thought the website was easy to use



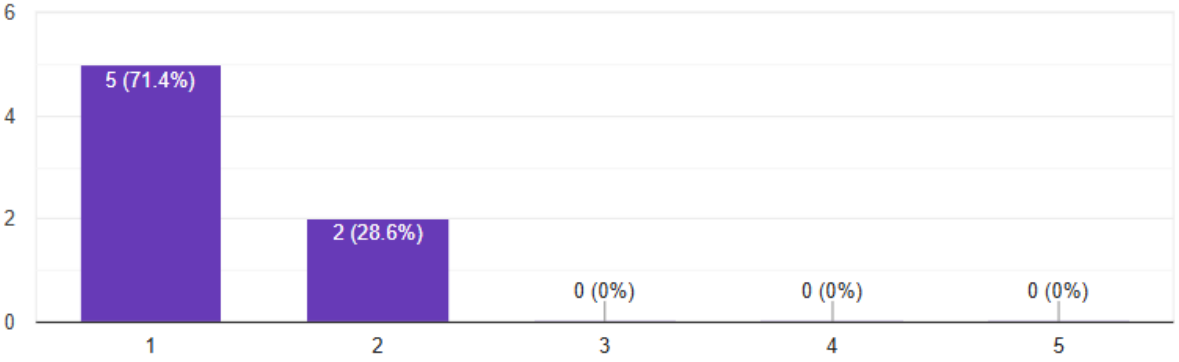
4. I think I will need technical support to use this site



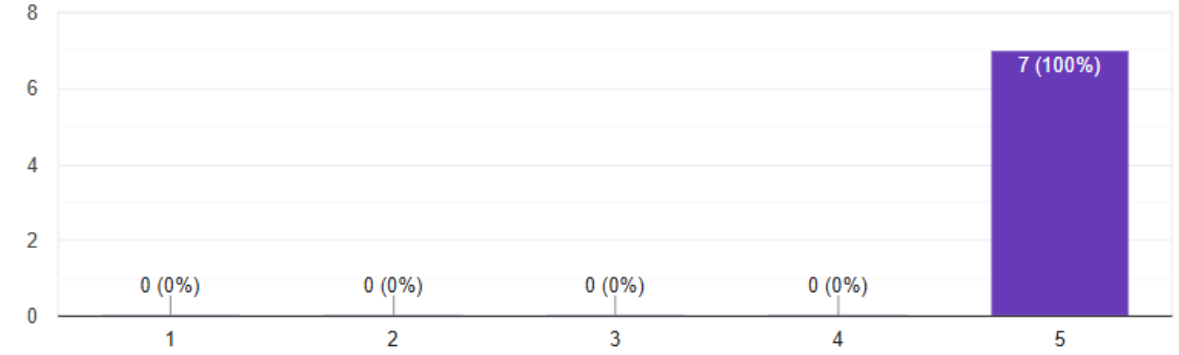
5. I found the various features on the site to be well integrated



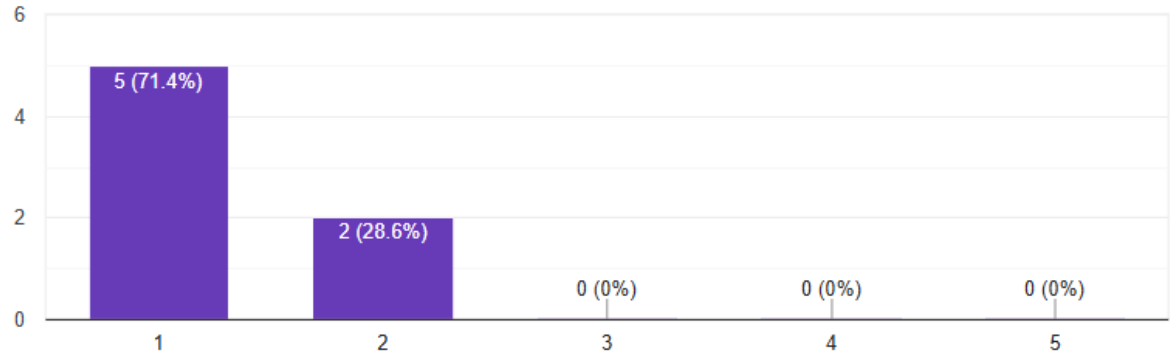
6. I thought there was too much inconsistency on this site



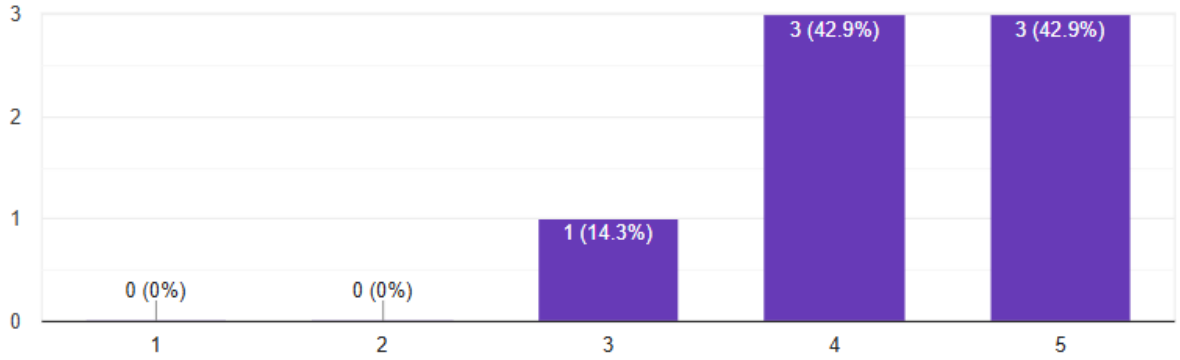
7. I assume most people will learn to use this site quickly



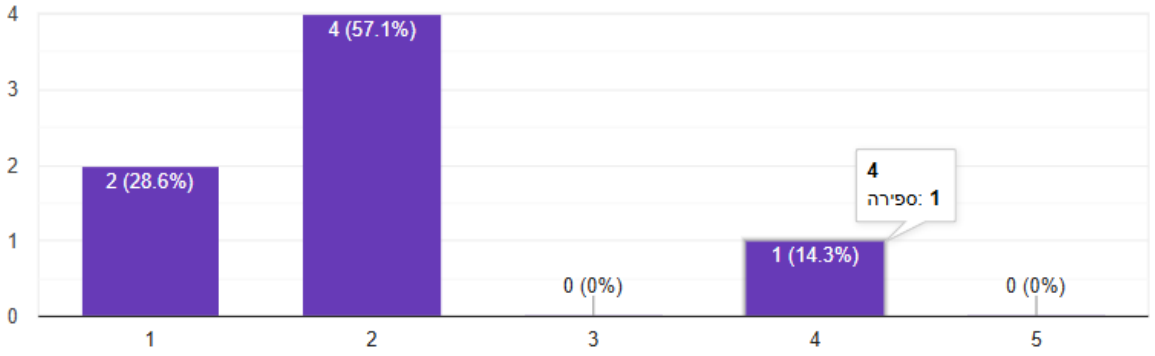
8. I found the site very cumbersome to use



9. I felt very safe using this site



10. I had to learn a lot of things before I was able to use this site



The results of the System Usability Scale survey indicate that users found the AvoTech system to be highly usable and intuitive. Across all 10 questions, responses showed consistently positive trends, with most participants agreeing that the system is easy to use, well-integrated, and suitable for their needs in the field.

The calculated average **SUS score was 87.14 out of 100**, which falls into the “Excellent” usability range according to industry benchmarks.

9. Conclusion and Summary

This report presented a comprehensive overview of AvoTech, an intelligent, IoT-based system designed to assist farmers in monitoring avocado tree health through real-time sensor data and AI-driven image analysis. The project evolved significantly from its initial concept in Phase A to a working, user-tested system in Phase B, demonstrating strong adaptability, technical depth, and a commitment to solving real agricultural problems.

9.1 Key Findings and Achievements

- User-Centered Development
  Through meetings with farmers in Hamat Gader and the Galil region and more, we applied a user-centered design process that shaped our features and interface to meet real needs—ranging from disease alerts for small growers to resource optimization for large-scale farmers.

- Functional System Implementation
  The system successfully integrates a wide range of capabilities: real-time sensor monitoring, image analysis using the GPT-4o-mini API, user management, and alert notifications—all within a responsive web application.

- Technological Integration
  Our stack includes Python (with Flask) for the backend, Next.js and Chart.js for the frontend, Firebase for real-time database syncing, and Mosquitto (MQTT) for lightweight sensor communication. We also used ngrok during development for secure remote testing.

- Adaptation and Innovation
  A key pivot occurred when we replaced the planned smart camera with a standard camera and shifted from on-device CNN models to cloud-based analysis using GPT-4o-mini. This change allowed for higher flexibility, lower hardware cost, and easier scaling.

- External Recognition
  We presented the system to engineers and students from other universities and shared it on social media, resulting in growing interest from agricultural communities in Israel and abroad.

9.2 Challenges and Solutions

- **Hardware Limitations**
  The original camera (Unity V2) failed to meet our requirements for disease detection.
  Solution: Replaced it with a basic camera, using cloud-based AI for image analysis.

- **Cost and API Efficiency**
  Sending high-resolution images to GPT-4o-mini was costly and inefficient.
  Solution: Added automatic image resizing to reduce token usage and improve response speed.

- **Sensor Accuracy**
  Farmers requested more accurate soil moisture readings and fertilizer recommendations.
  Solution: Documented this as an area for future enhancement involving advanced sensors.

- **User Interface Accessibility**
  Farmers wanted mobile access for easier use in the field.
  Solution: Planned a mobile version for future development.

- **Cross-Crop Support**
  Small growers working with various tree types requested support beyond avocados.
  Solution: Designed the architecture to support easy extension to other crops in the future.

- **External Circumstances – Regional Conflict**
  During the final stages of the project, the war situation in Israel imposed significant limitations on our work. Safety concerns, restricted movement, and limited access to resources and testing locations slowed development and team coordination.
  Solution: Despite these constraints, we maintained progress through remote collaboration and focused on completing a stable, functional version of the system. We believe that under normal circumstances, we could have reached an even higher level of refinement and system maturity.

9.3 Implications and Future Work

The successful implementation of AvoTech positions it as a valuable platform for smart agriculture. Potential directions for future development include:

- Expanding support for other tree types and crops
- Creating a dedicated mobile app for field use
- Adding advanced sensors for precise fertilizer and soil analysis
- Implementing multilingual support and offline capabilities
- Enhancing system personalization and analytics for large farms

9.4 Personal Learning Outcomes

This project offered us valuable experience beyond technical development, including:

- Applying Agile development in a real-world project
- Collaborating with stakeholders and gathering field-specific feedback

- Designing scalable, data-driven systems
- Gaining hands-on experience with cloud platforms, sensor integration, and AI APIs
- Understanding the importance of real-world usability through farmer testing

9.5 Final Thoughts

AvoTech is more than just a smart agriculture concept—it is a working prototype that addresses the urgent needs of modern farming. By combining real-time environmental data with AI-powered image analysis, it empowers farmers with insights that can save time, reduce costs, and prevent crop loss. The enthusiastic feedback from users and external interest confirm that the system has real potential for adoption and expansion.

As we move forward, we are committed to refining the platform, extending its capabilities, and helping shape the future of precision agriculture through innovation, accessibility, and sustainability.
While we are proud of what we achieved, we acknowledge that the recent war in Israel imposed limitations on our work. Restricted access to resources, field sites, and collaboration opportunities prevented us from fully realizing certain improvements. Nevertheless, we delivered a stable and impactful system, and we remain highly motivated to continue its development beyond this phase.

10. Appendices

10.1 Model Placement

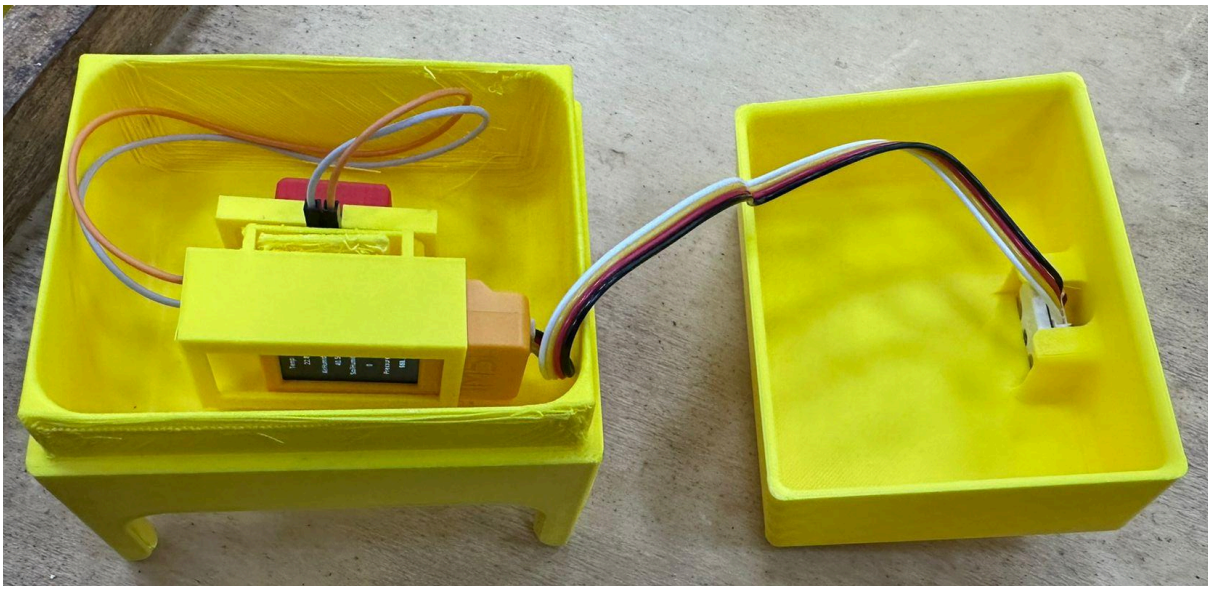
10.1.1 Soil and Air Humidity Sensors

Enable continuous monitoring of critical environmental conditions.

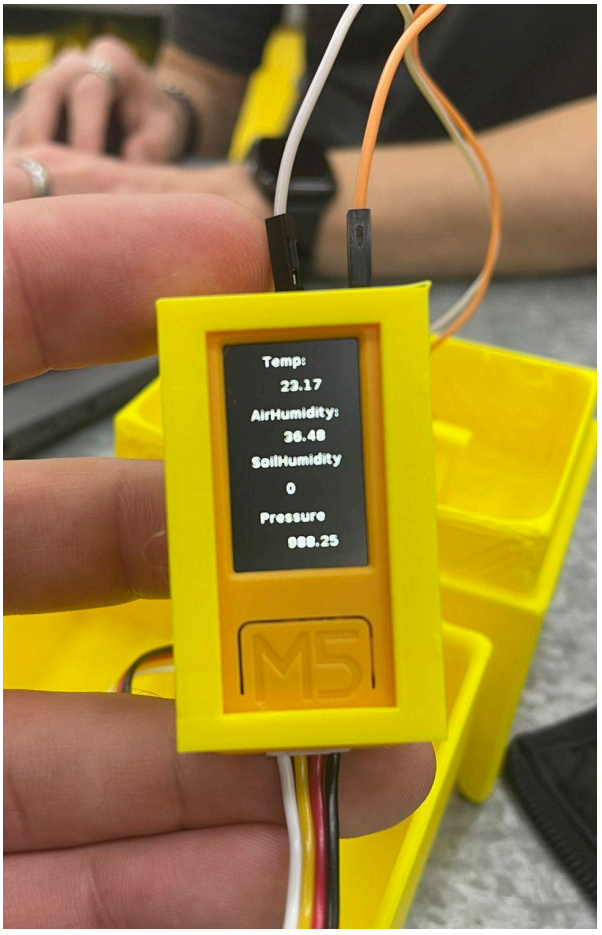
(Figure 25. Sensor Model Bracket)

10.1.2 M5Stick C Plus 2 Controller

The M5Stick C Plus 2 controller acts as the central hub of the system. It integrates data from the sensors and the camera, processes it in real time, and facilitates visualization and communication. Its compact design and built-in Wi-Fi capabilities make it ideal for IoT applications in agriculture. see Figure 22 and Figure 23.



(Figure 26. M5Stack Controller)



(Figure 27. M5Stack Controller)

10.1.3 Camera

Captures leaf images and analyzes them to detect disease symptoms.

(Figure 28. Camera)

References

[1] Ramírez-Gil, J. G., López, J. H., & Henao-Rojas, J. C. (2019). Causes of Hass avocado fruit rejection in preharvest, harvest, and packinghouse: economic losses and associated variables. Agronomy, 10(1), 8.

[2] Lakshmi, V. M., & Prabhu, T. (2024). In-depth Analysis and Investigation of Internet of Things and WSN for Precision Agriculture. COMPUTER, 24(11).

[3] Tzounis, A., Katsoulas, N., Bartzanas, T., & Kittas, C. (2017). Internet of Things in agriculture, recent advances and future challenges. Biosystems engineering, 164, 31-48.

[4] Shinghal, D. K., Noor, A., Srivastava, N., & Singh, R. (2011). Intelligent humidity sensor for-wireless sensor network agricultural application. International Journal of Wireless & Mobile Networks (IJWMN), 3(1), 118-128.

[5] Garg, A., Munoth, P., & Goyal, R. (2016, December). Application of soil moisture sensor in agriculture. In Proceedings of Internation Conference on Hydraulic (pp. 8-10).

[6] Abdulridha, J., Ehsani, R., Abd-Elrahman, A., & Ampatzidis, Y. (2019). A remote sensing technique for detecting laurel wilt disease in avocado in presence of other biotic and abiotic stresses. Computers and electronics in agriculture, 156, 549-557.

[8] Salgadoe, A. S. A., Robson, A. J., Lamb, D. W., Dann, E. K., & Searle, C. (2018). Quantifying the severity of phytophthora root rot disease in avocado trees using image analysis. Remote Sensing, 10(2), 226.

[9] Zentmyer, G. A. (1984). Avocado diseases. International Journal of Pest Management, 30(4), 388-400.

[10] Olatinwo, R. O., Fraedrich, S. W., & Mayfield III, A. E. (2021). Laurel wilt: current and potential impacts and possibilities for prevention and management. Forests, 12(2), 181.

[11] De Castro, A. I., Ehsani, R., Ploetz, R., Crane, J. H., & Abdulridha, J. (2015). Optimum spectral and geometric parameters for early detection of laurel wilt disease in avocado. Remote Sensing of Environment, 171, 33-44.

[12] Ayyandurai, M., Theradimani, M., Harish, S., Manonmani, K., Madhu, G. S., Raja, I. Y., ... & Pushpam, A. K. (2024). Bioprospecting of microbial agents and their metabolites as potential inhibitors of Phytophthora cinnamomi, the causal agent of avocado root rot. Physiological and Molecular Plant Pathology, 133, 102362.

[13]https://www.statista.com/statistics/593211/global-avocado-production-by-country/

[14] https://greenarborists.com/five-common-diseases-of-avocado-trees/

[15]https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/?utm_source=chatgpt.com

[16] https://ucanr.edu/blogs/blogcore/postdetail.cfm?postnum=24850